

KERNEL MATRIX COMPLETION

Rachelle Alvarez Rivero



*A dissertation submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy*

Kato Laboratory
Domain of Electronics and Informatics, Mathematics and Physics
Graduate School of Science and Technology
Gunma University

August 2018

© 2018
Rachelle Alvarez Rivero
ALL RIGHTS RESERVED

“It always seems impossible until it’s done.”

Nelson Mandela

GUNMA UNIVERSITY

Abstract

KERNEL MATRIX COMPLETION

Rachelle Alvarez Rivero

In this study, in order to improve the accuracy of machine learning, methodologies for completing multiple incomplete data representations have been developed. In data analysis including pattern recognition and clustering, each data object is not limited to a single representation. In many applications, several measurement methods are available to objects to be analyzed, yielding multiple data representations. For example, in the task of function prediction of proteins in cells, each protein can be represented with its amino acid sequence, cubic structure, interactions with other proteins, and expression data. Each data representation provides useful information for function prediction. Proteins that have homology in their amino acid sequences are likely to have same functions in cells. The cubic structures of proteins determine functions. Many function mechanisms in cells depend on multiple interacted proteins. Proteins with same functions express the same conditions in cells. Thus, each of these representations is informative for function prediction. Research reports have shown that analysis accuracy is improved by combining multiple data representations. However, an issue in data analysis based on multiple data representations is that data examples lacking any representation cannot be included in machine learning. In this thesis, several new methods for completing incomplete data are presented. To assess the effectiveness of the new data completion methods, experiments on real-world data are carried out. The results are then reported.

Acknowledgments

First and foremost, I would like to give my sincerest appreciation and gratitude to my adviser, Professor Tsuyoshi Kato, for all his patience and hard work in guiding me throughout my PhD journey, and for training me to become a better researcher. This doctoral dissertation would not have been possible without his supervision.

I would also like to thank my thesis panel: Professor Naoya Ohta, Professor Yoichi Seki, Professor Hidetoshi Yokoo, and Professor Hirofumi Yokouchi, for their time and effort in reviewing this thesis, and for their invaluable comments that led to its improvement.

I am grateful too to my professors in graduate courses and in Japanese Language classes, for all the patience and knowledge they imparted. I also thank all the staff of the Student Support section and of this department, who assisted me in all my university needs.

I thank my past and present labmates for all the smiles and kindness, especially to my tutors: Raissa Relator, Eisuke Ito, and Ayumi Shimoyama, for their research inputs and for helping me in processing many documents written in Japanese.

Thank you also to all my friends that made my stay here in Japan more meaningful: to the members of the English Lunch Group; to Dari, Ratana, Ji-Shiang Zhou, Thip, and Pikky; to Mr. and Mrs. Aoki; to ate Mitch, kuya, Antoniette and Danielle; and most especially to Fumiaki Higuchi who is always willing to extend his generosity and imparts his knowledge to me about many things. I also thank my friends and past students in the Philippines who never forget to say 'hi' to me from time to time.

I am also indebted to Gunma University and the University of the Philippines for the education and scholarships that I have received, especially to the Japanese Government (Monbukagakusho/MEXT) Scholarship for funding my PhD studies, and for the University of the Philippines for my study leave. I also give my appreciation to my professors in UP who believed in me and had been my source of encouragement.

All thanks to my family, especially to Leo, for the encouragement, love, and support. I love you all.

Finally, I thank You, Lord, for all the grace and blessings. All for Your greater glory.

Contents

Abstract	v
Acknowledgments	vii
List of Figures	xiii
List of Tables	xv
List of Abbreviations	xvii
Nomenclature	xix
1 Introduction	1
1.1 Thesis Outline and Contributions	2
2 Preliminaries	5
2.1 Learning from Features	5
2.1.1 Support Vector Machine	6
2.2 Kernel Methods	9
2.2.1 The Kernel Trick	10
2.2.2 Constructing Kernels	12
2.3 Summary	13
3 Mutual Kernel Matrix Completion	15
3.1 Related Works	18
3.2 Problem Setting	18
3.3 MKMC Method	19
3.3.1 The Objective Function	19
3.3.2 MKMC Algorithm	20
3.4 Statistical Framework of MKMC	22
3.4.1 An Overview of the EM Algorithm	22
3.4.2 MKMC Algorithm is an EM Algorithm	24
3.5 Experiments and Results	27
3.5.1 Experimental Setting	27
Data Set	28
Completion Task	29
Classification Task	29

3.5.2	Experimental Results	29
	Classification Performance	29
	Completion Accuracy	32
3.6	Summary	33
3.7	Derivations and Some Discussions	33
3.7.1	Derivation of (3.28) and (3.29)	33
3.7.2	Derivation of (3.32) and (3.33)	34
3.7.3	Derivation of the Model Parameter Update	35
4	Parametric Models for MKMC	37
4.1	Overview of Probabilistic PCA and Factor Analysis	40
4.1.1	Principal Component Analysis	40
	PCA as Maximum Variance	40
	PCA as Minimum Mean of Projection Errors	41
4.1.2	Probabilistic PCA	41
	Closed-Form Solutions	42
	EM algorithm for PCA	44
4.1.3	Factor Analysis	45
4.2	Parametric Models for MKMC	47
4.2.1	PCA-MKMC	47
4.2.2	FA-MKMC	48
4.3	Statistical Framework of the Parametric Models	51
4.3.1	EM Algorithm for PCA-MKMC	51
4.3.2	EM Algorithm for FA-MKMC	51
4.4	Experiments and Results	53
4.4.1	Experimental Setting	53
	Data Set	54
	Completion Methods	54
4.4.2	Classification Performance Result	55
4.5	Summary	55
4.6	Proofs and Derivations	56
4.6.1	Derivation of the second moments for FA.	56
4.6.2	Derivation of the model parameter updates for FA-MKMC.	57
4.6.3	Proof of Proposition 4.2.1	57
5	Conclusion	59
A	Some Formulas and Identities	61
A.1	Probability Densities	61
A.2	Multivariate Gaussian Distribution	61
A.2.1	Partitioned Gaussians	63
	Marginal Distribution	63

Conditional Distribution	63
A.3 Matrix Properties	64
A.4 Matrix Derivatives	66
Bibliography	67

List of Figures

1.1	Thesis contributions.	2
2.1	The optimal separating hyperplane.	6
2.2	Embedding of data points into a feature space.	9
3.1	Different representations of a protein.	16
3.2	Problem settings.	17
3.3	Experimental set-up.	27
3.4	Classification performance for varying number of missing entries.	30
3.5	Classification performance for each kernel matrix.	31
3.6	Prediction accuracy of the completion methods.	32
4.1	Experimental set-up.	53

List of Tables

4.1	Classification performance for 20% missed entries.	55
-----	--	----

List of Abbreviations

EM Algorithm	E xpectation- M aximization Algorithm
FA	F actor A nalysis
KL Divergence	K ullback- L eibler Divergence
MKMC	M utual K ernel M atrix C ompletion
PCA	P rincipal C omponent A nalysis
ROC	R eceiver O perating C haracteristic
SVM	S upport V ector M achines

Nomenclature

$ \mathbf{A} $	the determinant of matrix \mathbf{A}
\mathbf{A}^\top	the transpose of matrix \mathbf{A}
\mathbf{A}^{-1}	the inverse of matrix \mathbf{A}
$\text{diag}(\mathbf{x})$	an $n \times n$ diagonal matrix whose diagonal entries are the elements of vector $\mathbf{x} \in \mathbb{R}^n$
$\text{diag}(\mathbf{X})$	an n -dimensional vector containing the diagonal elements of matrix $\mathbf{X} \in \mathbb{R}^{n \times n}$
$\mathbb{E}[\cdot]$	expectation of a random variable
\mathcal{F}	feature space
\mathbf{G}	Gram matrix
\mathbf{I}_n	the $n \times n$ identity matrix; the subscript is often omitted
\mathbf{K}	kernel matrix
$\text{KL}(\cdot, \cdot)$	KL divergence
$\log(\cdot)$	natural logarithm
$\text{logdet}(\mathbf{A})$	the natural logarithm of the determinant of matrix \mathbf{A}
\mathbb{N}	the set of natural numbers
\mathbb{N}^n	set of n -dimensional natural vectors
\mathbb{N}_n	set of natural numbers less than or equal to n , for any $n \in \mathbb{N}$
$\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution of \mathbf{x} with mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\Sigma}$
$p(\cdot)$	probability
\mathbb{R}	the set of real numbers
\mathbb{R}^n	set of n -dimensional real vectors
$\mathbb{R}^{m \times n}$	set of $m \times n$ real matrices
\mathbb{R}^+	the set of positive real numbers
\mathbb{S}_+^n	set of $n \times n$ positive semidefinite symmetric matrices
\mathbb{S}_{++}^n	set of $n \times n$ positive definite symmetric matrices
$\text{Tr}(\mathbf{A})$	trace of (square) matrix \mathbf{A}
\mathbf{u}_i	i -th eigenvector
\mathcal{X}	input space
$\langle \cdot, \cdot \rangle$	inner product
$\ \cdot\ $	Euclidean norm
$\ \cdot\ _F$	Frobenius norm
$\mathbf{0}_{m \times n}$	the $m \times n$ zero matrix; the subscript is often omitted
$\kappa(\cdot, \cdot)$	kernel function

$\mathbf{\Lambda}$	diagonal matrix of eigenvalues
λ_i	i -th eigenvalue
ϕ	mapping from \mathcal{X} to \mathcal{F}
ψ	diagonal matrix of noise variances
σ^2	variance
Θ	set of model parameters

This thesis is dedicated to my family.

Chapter 1

Introduction

With the constant development of new technology, many new types of data in the field of biotechnology have become available. However, many of these biological data are expensive and are still difficult to obtain. Hence, optimal utilization of such data is of utmost importance to researchers in this field. Since biological data poses several challenges to computational biology such as high dimensionality, heterogeneous representation, and the need to combine or integrate such heterogeneous representations, active development of learning algorithms that can deal with these difficult aspects of the data have become imperative.

Kernel methods, especially support vector machines (SVMs), provided a major leap in the advancement of computational biology as powerful mechanisms in the analysis, processing, and handling of many data types. Kernel methods rely on the use of positive semidefinite functions called *kernels*, and, together with the feature space view by Boser et al. [9], led to new algorithm designs. Kernels, being identical to a dot product in a (usually) higher dimensional space (or feature space), addressed the problem of high dimensionality in the data; and through this implicit embedding of the data in a feature space, linear methods can be used for learning tasks such as classification and regression. Kernels also work with non-vectorial data, providing standardized solutions in handling structured and complex data types, common in biological data. Finally, the success of constructing kernels that are specifically designed to various biological data, and the ability to combine different kernels made it possible to learn from integrated heterogeneous data.

A collection of recent trends in kernel methods has been compiled in [44]. From kernel design, to learning from heterogeneous data and advanced application of kernels and SVMs, these researches have proven that advancement in computational biology had really come a long way, and will continue to be as the need to extract knowledge from data becomes increasingly valuable.

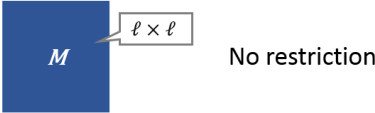
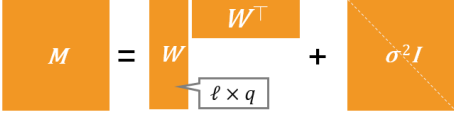

	Model Definition	Degrees of Freedom
MKMC		$\ell(\ell + 1)/2$
PCA-MKMC		$\ell q + 1 - q(q - 1)/2$
FA-MKMC		$\ell q + \ell - q(q - 1)/2$

FIGURE 1.1: Thesis contributions.

The MKMC algorithm presented in this thesis introduces a model matrix that serves as a single representation of the empirical data. Here, the model matrix has no restriction, thereby requiring large degrees of freedom in parameter estimation. In order to have control of model flexibility, variants of the MKMC model are introduced: the PCA-MKMC and FA-MKMC. Here, the model matrices are restricted through the use of an $\ell \times q$ matrix and a diagonal matrix; and by adjusting the value of q , the number of degrees of freedom is controlled.

1.1 Thesis Outline and Contributions

The studies in this thesis rely on the use of kernel methods; the main contribution is to provide solutions to the incomplete-data problem common to biological researches. As obtaining high quality biological data is expensive and time-consuming, many researchers lean towards the use of machine learning methods, such as the ones presented in this thesis, due to their convenience and practicality.

After providing basic familiarity with binary classification, SVM, and kernel methods in Chapter 2, the primary contributions of this thesis are listed as follows:

- Chapter 3 presents a novel kernel-based method of mutually inferring the missing entries in kernel matrices corresponding to a set of different but relevant data sources. The method employs Kullback-Leibler divergences between the kernel matrices through a model kernel matrix, and aims at minimizing these divergences when inferring the missing entries. Minimization is done via an iterative EM algorithm, which results to completion of the kernel matrices and a means of combining them for further application. Such application is the task of classifying yeast proteins as either membrane or non-membrane, through a learned SVM classifier.

- Chapter 4 extends the model presented in Chapter 3 by allowing control of the flexibility of the model kernel matrix. As the model kernel matrix in the previous chapter is full covariance, the risk of overfitting is present, that is, a classifier trained on this matrix can classify a training set very well, but cannot generalize a new data properly. Through suitable adjustments of model flexibility, the risk of overfitting (and underfitting) can be avoided. The parametric models introduced in this chapter are based on probabilistic latent variable models, such as the probabilistic PCA and factor analysis models. These methods are applied to the task of protein functional classification in yeast to demonstrate effectiveness with regards to classification performance.

An overview of these contributions is illustrated in Fig. 1.1. Here, the differences in the structure of the model matrices of the introduced algorithms are shown. Finally, Chapter 5 concludes and summarizes the overall contributions of this thesis.

The works in this thesis have contributed to a journal publication and a research paper under review in the Institute of Electronics, Information and Communication Engineers (IEICE) Transactions on Information & Systems [41, 40].

Chapter 2

Preliminaries

This chapter covers some concepts that are essential in the development of this thesis. Readers may refer to [47, 8, 10, 15, 22, 37, 45] for in-depth discussions and as supplementary materials about machine learning, kernel methods, and other relevant topics. Also for quick reference, an appendix of formulas and identities and a page for nomenclature have been provided.

2.1 Learning from Features

Sometimes synonymously referred to as machine learning, *pattern recognition* or *pattern analysis* is a field of study that aims at detecting patterns and regularities in a given set of data, the ultimate goal of which is to be able to make correct predictions about a new data point coming from the same source. To discover the underlying patterns in the data, learning methodologies are used. *Supervised learning* is a type of learning methodology wherein a given set of data with their corresponding labels, the underlying relationship between the input (data) and the output (labels) is learned, so that when a new data point arrives, it will be correctly assigned a label. These labels can be nominal, in which case the learning task is called *multi-class classification*. In the case where the label or output is real, the learning task is called *regression*. On the other hand, in an *unsupervised learning*, there are no associated labels or outputs to the data, and the learning task is to identify patterns that are innate in the data set, and group the data accordingly from the identified patterns. In this way, the data points are clustered, and the learning task is called *clustering*.

In order for machine learning algorithms to operate and statistical analysis to be applied, numerical representation of objects is required. These measurable properties of the data are called *features*, and are conveniently expressed as vectors. For instance, a blue pixel can be represented by its RGB (0,114,178), or by its CMYK (100,50,0,0)¹.

¹RGB (red, green, blue) and CMYK (cyan, magenta, yellow, black or “key”) are the standard colors used by designers; the former is typically used for web pages while the latter is for printed materials.

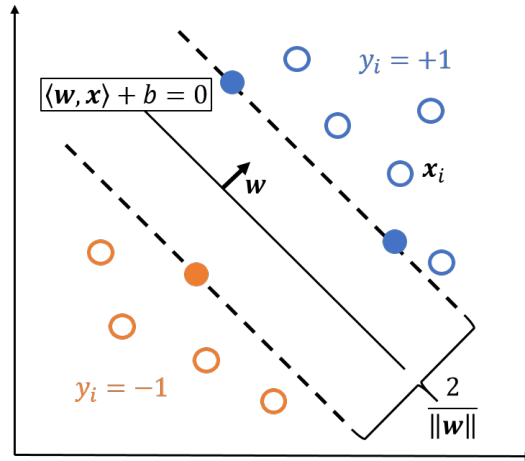


FIGURE 2.1: The optimal separating hyperplane.

SVM finds the hyperplane (the line with equation $\langle \mathbf{w}, \mathbf{x} \rangle + b = 0$ and normal vector \mathbf{w}) that optimally separates the two classes. Here, the data points on the left side of the optimal hyperplane belong to the negative class $y = -1$, while those on the right side belong to the positive class $y = +1$. Each margin has width $1/\|\mathbf{w}\|$, and the shaded points on the dashed lines are the support vectors.

A protein, on the other hand, may have a feature vector $(1,1,0,1,0,0)$, where 1 implies interaction with another protein in the data set, while 0 implies no interaction. Such feature vectors are employed by machine learning algorithms to exploit knowledge from a sample data in order to infer solutions to certain tasks.

The focus of the succeeding discussions will be on the learning methodology employed in this thesis—the two-class classification or simply, *binary classification* in supervised learning. This classification problem can be formally described as follows: consider the training data \mathcal{D} consisting of ℓ input-output pairs

$$\mathcal{D} = \{(\mathbf{x}_i, y_i) \mid \mathbf{x}_i \in \mathbb{R}^n, y_i \in \{1, -1\}\}_{i=1}^{\ell}.$$

The input vectors \mathbf{x}_i are usually called *features*, while the output y_i is sometimes called the *target*. From the training set \mathcal{D} , a learning algorithm tries to learn a *decision function* $f: \mathbf{x}_i \mapsto y_i$ so that when given a new data point \mathbf{x} , f can predict an output y for that data.

2.1.1 Support Vector Machine

A learning algorithm that solves the classification task described above is support vector machine (SVM) [9, 14, 52], the intuition of which is shown in Fig. 2.1. In this case, a hyperplane

$$\langle \mathbf{w}, \mathbf{x} \rangle + b = 0, \tag{2.1}$$

where $\mathbf{w} \in \mathbb{R}^n$ is a vector normal to the plane and $b \in \mathbb{R}$, separates the positive training data from the negative ones through the decision function

$$f(\mathbf{x}) = \text{sign}(\langle \mathbf{w}, \mathbf{x} \rangle + b). \quad (2.2)$$

Here, f assigns \mathbf{x} to the positive group ($y = 1$) if $z = \langle \mathbf{w}, \mathbf{x} \rangle + b > 0$, and to the negative group ($y = -1$) if $z < 0$. In this setting, the data points are assumed to be correctly classified by the linear classifier, i. e., $y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) > 0$ for all $i \in \mathbb{N}_\ell$. Among the many hyperplanes possible, SVM finds the optimal hyperplane

$$\langle \mathbf{w}_0, \mathbf{x} \rangle + b_0 = 0, \quad (2.3)$$

in which the distance of the closest data point to the decision boundary (called *margin*) is maximal. In this way, the risk of overfitting is minimized and better generalization can be obtained. To proceed with the optimization process, consider the perpendicular distance of any point \mathbf{x}_i from the hyperplane:

$$\frac{y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b)}{\|\mathbf{w}\|}. \quad (2.4)$$

Then the maximum margin is the solution of the maximization problem

$$\operatorname{argmax}_{\mathbf{w}, b} \left\{ \frac{1}{\|\mathbf{w}\|} \min_i [y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b)] \right\}. \quad (2.5)$$

By setting

$$y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) = 1 \quad (2.6)$$

for the points closest to the hyperplane, then all the data points satisfy

$$y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) \geq 1, \quad \text{for all } i = 1, \dots, \ell. \quad (2.7)$$

The complex problem (2.5) then takes the equivalent form

$$\operatorname{argmin}_{\mathbf{w}, b} \frac{1}{2} \|\mathbf{w}\|^2, \quad (2.8)$$

subject to the constraint (2.7). This optimization problem can now be easily solved by constructing the Lagrangian

$$L(\mathbf{w}, b, \boldsymbol{\alpha}) = \frac{1}{2} \|\mathbf{w}\|^2 - \sum_{i=1}^{\ell} \alpha_i [y_i (\langle \mathbf{w}, \mathbf{x}_i \rangle + b) - 1], \quad (2.9)$$

and finding the saddle point, where (2.9) is minimized with respect to \mathbf{w} and b , and maximized with respect to

$$\alpha_i \geq 0, \quad \text{for all } i = 1, \dots, \ell. \quad (2.10)$$

Minimization of (2.9) with respect to \mathbf{w} and b yields

$$\mathbf{w} = \sum_{i=1}^{\ell} \alpha_i y_i \mathbf{x}_i, \quad \text{and} \quad (2.11)$$

$$\sum_{i=1}^{\ell} \alpha_i y_i = 0, \quad (2.12)$$

respectively. Using (2.11) and (2.12), the *dual* form of (2.9) is obtained:

$$W(\boldsymbol{\alpha}) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j \langle \mathbf{x}_i, \mathbf{x}_j \rangle, \quad (2.13)$$

and is then maximized with respect to $\boldsymbol{\alpha}$ subject to the constraints (2.10) and (2.12). The vector \mathbf{w} in (2.11) is then optimal when specified by the optimal solution $\boldsymbol{\alpha}^0 = (\alpha_1^0, \dots, \alpha_{\ell}^0)$:

$$\mathbf{w}_0 = \sum_{i=1}^{\ell} \alpha_i^0 y_i \mathbf{x}_i, \quad (2.14)$$

and for chosen b_0 that maximizes the margin, the optimal hyperplane in (2.3) can then be written as

$$\sum_{i=1}^{\ell} \alpha_i^0 y_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b_0 = 0. \quad (2.15)$$

In constrained optimization problems, the optimal solution is required to satisfy the Karush-Kuhn-Tucker (KKT) conditions (2.10), (2.7), and

$$\alpha_i^0 [y_i (\langle \mathbf{w}_0, \mathbf{x}_i \rangle + b_0) - 1] = 0. \quad (2.16)$$

It follows that for nonzero α_i^0 ,

$$y_i (\langle \mathbf{w}_0, \mathbf{x}_i \rangle + b_0) = 1, \quad (2.17)$$

which are precisely the vectors \mathbf{x}_i on the margin. Since these are the only vectors that matter for nonzero α_i^0 in the summation for \mathbf{w}_0 in (2.14), these vectors are appropriately called the *support vectors*. The decision function (2.2) can now be rewritten in terms of the support vectors

$$f(\mathbf{x}) = \text{sign} \left(\sum_{i \in \mathcal{S}} \alpha_i^0 y_i \langle \mathbf{x}_i, \mathbf{x} \rangle + b_0 \right), \quad (2.18)$$

where the indices of the support vectors are denoted by \mathcal{S} .² It is noteworthy to mention that in the dual form (2.13) of the Lagrangian that had to be maximized, the information from the training data enters the equation only through the inner

²The parameter b_0 can also be obtained in terms of the support vectors; the solution can be found in [8].

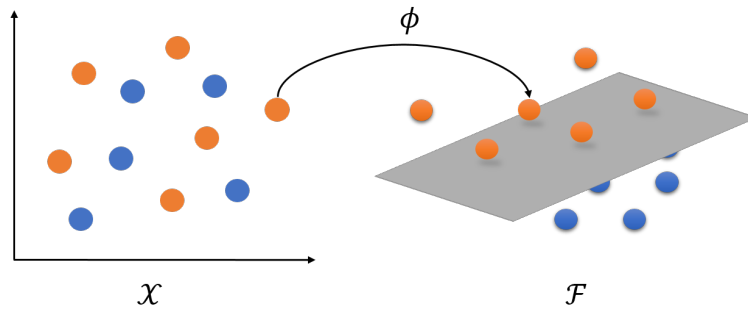


FIGURE 2.2: Embedding of data points into a feature space.

In the input space \mathcal{X} , no linear classifier could be found to separate the two classes (blue and orange). However, when the data points were mapped into a higher dimensional space \mathcal{F} , a linear classifier was found.

product. Likewise, once the classifier has been trained, a new point \mathbf{x} can be classified by (2.18) only through the inner product of \mathbf{x} with the support vectors, which makes SVM very efficient to use. The SVM algorithm was originally developed by Vapnik and Chervonenkis in 1963, and gained popularity when Boser, Guyon, and Vapnik extended SVM to nonlinear classifiers in 1992, where kernel trick was applied.

2.2 Kernel Methods

For data sets with more complex pattern, a linear classifier may not be found in the original input space \mathcal{X} . In such cases, the data is said to be non-linearly separable in \mathcal{X} . However, when these data points are mapped to a higher-dimensional space \mathcal{F} , referred to as *feature space*, a linear classifier can be found, as in the case in Fig. 2.2. Situations like these require learning algorithms to change the representation of the data through some user-specified nonlinear mapping function $\phi : \mathcal{X} \rightarrow \mathcal{F}$, and is permissible to do so since changing the data's representation or coordinates does not change the underlying patterns or regularities in the data [47].

A class of pattern recognition algorithms, the *kernel methods*, maps the data into a higher-dimensional space where the patterns can be easily detected. This efficient mapping of the data in the feature space is done via a kernel function, formally defined as follows:

Definition 2.2.1 (Kernel). *A kernel is a function κ such that for all $\mathbf{x}, \mathbf{z} \in \mathcal{X}$ satisfies*

$$\kappa(\mathbf{x}, \mathbf{z}) = \langle \phi(\mathbf{x}), \phi(\mathbf{z}) \rangle,$$

where ϕ is a mapping from an input space \mathcal{X} to an (inner product) feature space \mathcal{F}

$$\phi : \mathbf{x} \mapsto \phi(\mathbf{x}) \in \mathcal{F}.$$

Simply put, a kernel function takes the inner product of two data images under an embedding ϕ .

2.2.1 The Kernel Trick

Popularly termed as *kernel trick*, one can conveniently replace any inner product of two data images appearing in a learning algorithm, by a valid kernel function. To see this, suppose a linear classifier is being constructed in a feature space \mathcal{F} for the data images under a mapping $\phi : \mathbf{x} \mapsto \phi(\mathbf{x}) \in \mathcal{F}$. By kernel trick, the dual form in (2.13) can be rewritten as

$$W(\boldsymbol{\alpha}) = \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \quad (2.19)$$

$$= \sum_{i=1}^{\ell} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{\ell} \alpha_i \alpha_j y_i y_j \kappa(\mathbf{x}_i, \mathbf{x}_j), \quad (2.20)$$

while the decision function in (2.18) can be expressed as

$$\begin{aligned} f(\mathbf{x}) &= \text{sign} \left(\sum_{i \in \mathcal{S}} \alpha_i^0 y_i \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}) \rangle + b_0 \right) \\ &= \text{sign} \left(\sum_{i \in \mathcal{S}} \alpha_i^0 y_i \kappa(\mathbf{x}_i, \mathbf{x}) + b_0 \right), \end{aligned} \quad (2.21)$$

which corresponds to the optimal hyperplane separating the data images in the feature space.

Observe that through kernels, one does not have to explicitly compute the coordinates of the data in its new representation; rather, one has just be able to evaluate the corresponding kernel.

As an illustration, given the input data $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_\ell)^\top \in \mathbb{R}^{\ell \times n}$, the SVM algorithm considered in the previous section operates only through the inner product of the input:

$$\mathbf{G} = \begin{pmatrix} \langle \mathbf{x}_1, \mathbf{x}_1 \rangle & \langle \mathbf{x}_1, \mathbf{x}_2 \rangle & \cdots & \langle \mathbf{x}_1, \mathbf{x}_\ell \rangle \\ \langle \mathbf{x}_2, \mathbf{x}_1 \rangle & \langle \mathbf{x}_2, \mathbf{x}_2 \rangle & \cdots & \langle \mathbf{x}_2, \mathbf{x}_\ell \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \mathbf{x}_\ell, \mathbf{x}_1 \rangle & \langle \mathbf{x}_\ell, \mathbf{x}_2 \rangle & \cdots & \langle \mathbf{x}_\ell, \mathbf{x}_\ell \rangle \end{pmatrix} = \mathbf{X} \mathbf{X}^\top, \quad (2.22)$$

which is called the *Gram matrix*³ of \mathbf{X} . Note that since all the data information needed in the algorithm is contained in this matrix, the original data can be discarded.

³The matrix is named after the Danish actuary and mathematician Jorgen P. Gram (1850–1916).

This is highly desirable especially when the number of objects ℓ is much smaller than the dimension n , which is usually the case.

Now, when the data points are mapped under ϕ , the information needed in (2.19) is

$$\begin{aligned} \mathbf{K} &= \begin{pmatrix} \langle \phi(\mathbf{x}_1), \phi(\mathbf{x}_1) \rangle & \langle \phi(\mathbf{x}_1), \phi(\mathbf{x}_2) \rangle & \cdots & \langle \phi(\mathbf{x}_1), \phi(\mathbf{x}_\ell) \rangle \\ \langle \phi(\mathbf{x}_2), \phi(\mathbf{x}_1) \rangle & \langle \phi(\mathbf{x}_2), \phi(\mathbf{x}_2) \rangle & \cdots & \langle \phi(\mathbf{x}_2), \phi(\mathbf{x}_\ell) \rangle \\ \vdots & \vdots & \ddots & \vdots \\ \langle \phi(\mathbf{x}_\ell), \phi(\mathbf{x}_1) \rangle & \langle \phi(\mathbf{x}_\ell), \phi(\mathbf{x}_2) \rangle & \cdots & \langle \phi(\mathbf{x}_\ell), \phi(\mathbf{x}_\ell) \rangle \end{pmatrix} \\ &= \begin{pmatrix} \kappa(\mathbf{x}_1, \mathbf{x}_1) & \kappa(\mathbf{x}_1, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_1, \mathbf{x}_\ell) \\ \kappa(\mathbf{x}_2, \mathbf{x}_1) & \kappa(\mathbf{x}_2, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_2, \mathbf{x}_\ell) \\ \vdots & \vdots & \ddots & \vdots \\ \kappa(\mathbf{x}_\ell, \mathbf{x}_1) & \kappa(\mathbf{x}_\ell, \mathbf{x}_2) & \cdots & \kappa(\mathbf{x}_\ell, \mathbf{x}_\ell) \end{pmatrix}, \end{aligned} \quad (2.23)$$

which is called the *kernel matrix*. It can be observed that the entries of the kernel matrix can be obtained in only one operation, i. e., through a kernel function, instead of mapping the data and then computing the inner product. As a result, the mapping ϕ can be left implicit. Moreover, since the kernel is operating on pairs of data points, it can be viewed as a similarity measure between those two points, and hence the kernel matrix represents some generalized similarity measure between input vectors.

It was mentioned earlier that a valid kernel is needed in order to exploit the kernel trick. Fortunately, there is no need to construct the mapping explicitly to test whether or not a function constitutes a valid kernel—a necessary and sufficient condition is that for all the possible choices of the (training) data, the kernel matrix must be positive semidefinite⁴, as stated in *Mercer's Theorem*:

Theorem 2.2.1 (Mercer's Theorem). *A symmetric function $\kappa(\mathbf{x}_i, \mathbf{x}_j)$ can be expressed as an inner product*

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

for some ϕ if and only if the kernel matrix (2.23) is positive semidefinite for any $\mathbf{x}_i, \mathbf{x}_j \in \mathbb{R}^n$, for $i, j = 1, \dots, \ell$.

This theorem serves as a powerful tool that allows us to handle kernels without caring how the mapping ϕ nor the corresponding feature space look like. As long as the necessary condition for kernel matrices—the positive semidefiniteness—is maintained, a valid kernel is guaranteed, as well as the existence of its corresponding feature space.

As a matrix containing the kernel values for every pair of data points, the kernel matrix (as well as the Gram matrix) is positive semidefinite, and the proof is rather

⁴A matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ is positive semidefinite if for all $\boldsymbol{\alpha} \in \mathbb{R}^n$, $\boldsymbol{\alpha}^\top \mathbf{A} \boldsymbol{\alpha} \geq 0$.

straightforward. Indeed, letting $\mathbf{K}_{i,j} = \kappa(\mathbf{x}_i, \mathbf{x}_j) = \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle$ be the i, j -th entry of kernel matrix \mathbf{K} , and for any $\boldsymbol{\alpha} \in \mathbb{R}^n$,

$$\begin{aligned} \boldsymbol{\alpha}^\top \mathbf{K} \boldsymbol{\alpha} &= \sum_{i,j=1}^{\ell} \alpha_i \alpha_j \mathbf{K}_{ij} = \sum_{i,j=1}^{\ell} \alpha_i \alpha_j \langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle \\ &= \left\langle \sum_{i=1}^{\ell} \alpha_i \phi(\mathbf{x}_i), \sum_{j=1}^{\ell} \alpha_j \phi(\mathbf{x}_j) \right\rangle \\ &= \left\| \sum_{i=1}^{\ell} \alpha_i \phi(\mathbf{x}_i) \right\|^2 \geq 0. \end{aligned}$$

2.2.2 Constructing Kernels

Given the characterizations for valid kernels, one can now create functions derived from simpler kernels without explicitly constructing the feature space. Provided that the new function created is positive semidefinite, the function is a valid kernel, and the feature space where the inner product is computed by this new function exists.

The following proposition shows the closure property of kernels under some operations, i. e., when one or more kernels undergo with such operations, the positive semidefinite property of kernels is preserved [47]:

Proposition 2.2.1 (Closure properties). *Let κ_1 and κ_2 be kernels over $\mathcal{X} \times \mathcal{X}$, $\mathcal{X} \subseteq \mathbb{R}^n$, $a \in \mathbb{R}^+$, $f(\cdot)$ a real-valued function on \mathcal{X} , $\phi: \mathcal{X} \mapsto \mathcal{F}$ with κ_3 a kernel over $\mathcal{F} \times \mathcal{F}$, and \mathbf{B} a symmetric positive semidefinite $n \times n$ matrix. Then the following functions are kernels:*

- (i) $\kappa(\mathbf{x}, \mathbf{z}) = \kappa_1(\mathbf{x}, \mathbf{z}) + \kappa_2(\mathbf{x}, \mathbf{z});$
- (ii) $\kappa(\mathbf{x}, \mathbf{z}) = a\kappa_1(\mathbf{x}, \mathbf{z});$
- (iii) $\kappa(\mathbf{x}, \mathbf{z}) = \kappa_1(\mathbf{x}, \mathbf{z})\kappa_2(\mathbf{x}, \mathbf{z});$
- (iv) $\kappa(\mathbf{x}, \mathbf{z}) = f(\mathbf{x})f(\mathbf{z});$
- (v) $\kappa(\mathbf{x}, \mathbf{z}) = \kappa_3(\phi(\mathbf{x}), \phi(\mathbf{z}));$
- (vi) $\kappa(\mathbf{x}, \mathbf{z}) = \mathbf{x}^\top \mathbf{B} \mathbf{z}.$

The proofs for (i) and (ii) can be shown by utilizing Mercer's theorem, where it is only necessary to show that the induced kernel matrix is positive semidefinite. Indeed, given kernel matrices \mathbf{K}_1 and \mathbf{K}_2 obtained by restricting κ_1 and κ_2 to $\{\mathbf{x}_1, \dots, \mathbf{x}_\ell | \mathbf{x}_i \in \mathbb{R}^n\}$, and given any $\boldsymbol{\alpha} \in \mathbb{R}^\ell$,

- (i) $0 \leq \boldsymbol{\alpha}^\top \mathbf{K}_1 \boldsymbol{\alpha} + \boldsymbol{\alpha}^\top \mathbf{K}_2 \boldsymbol{\alpha} = \boldsymbol{\alpha}^\top (\mathbf{K}_1 + \mathbf{K}_2) \boldsymbol{\alpha}$; and
- (ii) $0 \leq a \boldsymbol{\alpha}^\top \mathbf{K}_1 \boldsymbol{\alpha} = \boldsymbol{\alpha}^\top a \mathbf{K}_1 \boldsymbol{\alpha}$.

Since $\mathbf{K}_1 + \mathbf{K}_2$ and $a\mathbf{K}_1$ are positive semidefinite, $\kappa_1 + \kappa_2$ and $a\kappa_1$ are kernels.

Given the above closure property for kernels, more complex kernels derived from simple kernels can also be constructed, as follows:

- (i) $\kappa(\mathbf{x}, \mathbf{z}) = p(\kappa_1(\mathbf{x}, \mathbf{z}))$, for a polynomial function p with positive coefficients;
- (ii) $\kappa(\mathbf{x}, \mathbf{z}) = \exp(\kappa_1(\mathbf{x}, \mathbf{z}))$;
- (iii) $\kappa(\mathbf{x}, \mathbf{z}) = \exp\left(-\|\mathbf{x} - \mathbf{z}\|^2 / (2\sigma^2)\right)$, for $\sigma \in \mathbb{R}^+$.

The kernel function in (iii) is also known as the Gaussian radial basis kernel, one of the most widely-used type of kernel function. Other most recognized kernels in practice are the linear kernel $\kappa(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle$, and the polynomial kernel $\kappa(\mathbf{x}, \mathbf{z}) = \langle \mathbf{x}, \mathbf{z} \rangle^d + c$, $c \in \mathbb{R}$, $d \in \mathbb{N}$.

2.3 Summary

Kernel methods make it possible to detect nonlinear patterns in the data by constructing a feature space through a nonlinear map, and then searching for linear patterns in that space. Through the use of valid kernel functions, very efficient computations can be made since kernels are operated using the input data and hence, explicit computation of the mapping is no more needed.

Characterization of valid kernels allows one to use the kernel trick on any learning algorithms that employs the inner product of input data, either in the original input space or in the higher-dimensional feature space. Moreover, a valid kernel assures the existence of a feature space where its corresponding inner product is defined. Since different embeddings of the data correspond to different kernels, a kernel can be constructed in a way that it captures the similarity between data points in the intended application. By evaluating the kernel on all pairs of data points, a symmetric positive semidefinite matrix containing some notion of similarity between the data points is produced. A kernel matrix, thus, can be viewed as a representation of some generalized similarity measure of the data. Given these kernels corresponding to different embeddings of the data, and given that simple algebraic combinations of valid kernels induce another valid kernel, different representations of the data can then be integrated, paving the way for kernel approach to data fusion.

Chapter 3

Mutual Kernel Matrix Completion

From the discussions about kernel methods in the previous chapter, it can now be demonstrated how kernel methods can be utilized in the analysis of data, especially in the field of biology, where an object can be described in more than one way. For example, a bacterium can be represented by several marker sequences [51], while a protein can be described through its 3-D structure or its gene expression and amino acid sequences [30]. Learning from different descriptions or partial *views* of an object is commonly referred to as *multiview learning* [50], and several studies [7, 13] employed multiview learning to solve some clustering tasks. The availability of such information provided us with different views of the data, and the question of how to unify these different views to provide a complete description of the data arose. An impediment, however, in unifying these complementary information about a certain biological data, is that the descriptions come in a diverse format, as illustrated in Fig. 3.1. For instance, 3-dimensional structures of proteins are shown in diagrams, while protein-protein interactions are best represented as graphs, where nodes represent proteins and adjacent nodes means there is an interaction between the two proteins; gene expressions are represented as heat maps, and protein sequences as a 20-symbol alphabet string, to name a few. Lanckriet et al. [30] demonstrated how these partial information in diverse formats can be unified through the use of kernel methods. In their work, each partial information of the data are transformed into a common format—as kernel matrices—whose entries are generalized similarity relationships among data points, defined by kernel functions. Through kernel methods, different notions of similarity in the data are captured. For example, if two genes are close to one another or are very similar, then the amino acid sequences of their protein products would be very similar. Likewise, their gene expression measurements would be similar. The similarity measurements in the space derived from amino acid sequences is quite different from those in the space derived from gene expression measurements. Thus, through different embeddings of the data (corresponding to different kernels), the different notions of similarity are captured. Now that the

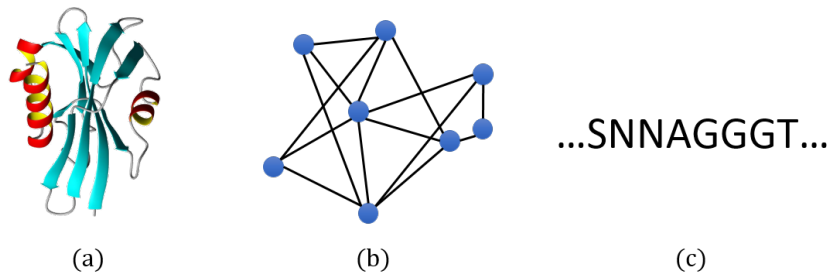


FIGURE 3.1: Different representations of a protein.

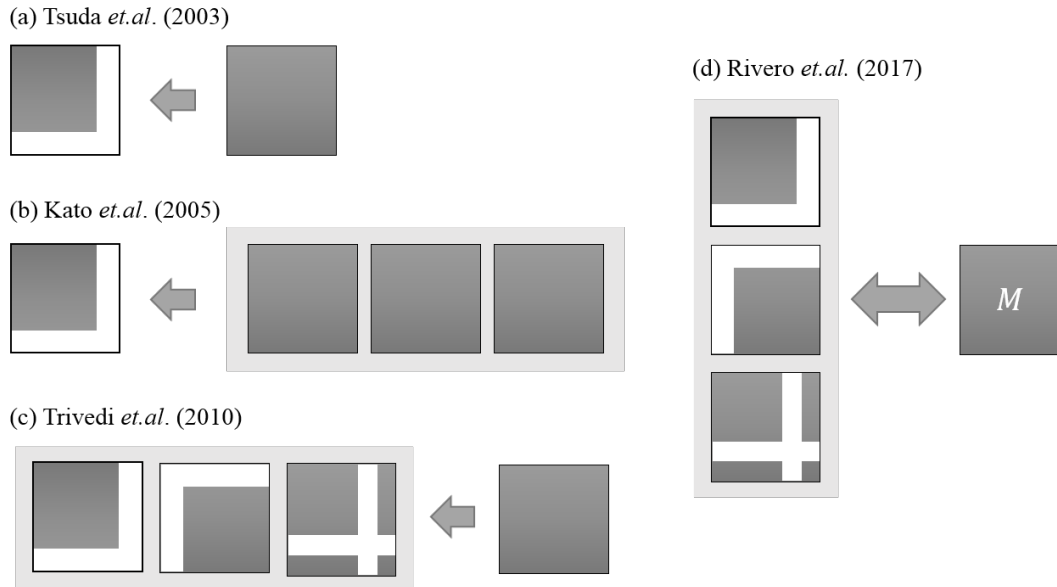
(a) 3-D structure of a protein;¹(b) Protein-protein interaction network, best represented as a graph; (c) A protein sequence as a 20-symbol alphabet string.

heterogeneity of the data representations is addressed through kernel methods, the different views of the data can now be integrated or combined.

Simultaneous use of multiple kernel matrices for learning, also referred to as *multiple kernel learning*, has been proven to be beneficial in the analysis of data. Some real-world applications of multiple kernel learning include: classifying mines and clutter using 4-view land mine detection data [56], and web page clustering task [50]; but multiple kernel learning is mostly applied to biology-related tasks such as prediction of psychopathological disorders and of drug sensitivity in breast cancer cell lines [6]; protein classification tasks [27, 31, 30, 32]; and inference of protein networks [26], among others. Lanckriet et al. [30] have shown in their work that integrating different representations of a genome leads to better protein classification, as opposed to using a single genome representation alone. While [30] used semidefinite programming (SDP) techniques to optimally combine the multiple kernel representations, a more recent approach by Gönen [21] uses a fully conjugate Bayesian formulation that allows a large number of kernels to be combined efficiently.

Since high quality data are expensive and acquiring them requires extensive work, it is usually the case that only a handful of good information can be obtained. In some situations, obtaining noisy or erroneous data is inevitable. Hence, the kernel matrices derived from such incomplete data will have missing entries corresponding to the unavailable samples. Such kernel matrices are called *incomplete*. Incomplete kernel matrices cannot be combined directly, and cannot be given to machine learning. Since there are far fewer missing entries in a kernel matrix than the data matrix where it is derived from, it is therefore more advantageous (in addition to the fact that machine learning methods like support vector machines (SVMs) work with kernel matrices) to infer the missing entries of the kernel matrix, than those of the data matrix [29]. To address this incomplete kernel matrix problem, many completion techniques have been introduced. An early work by Tsuda et al. [51] utilized a complete auxiliary

¹David E. Volk, *Ribbonr Diagram of a Protein* (USA in citizendum, 2007)

FIGURE 3.2: Problem settings.²

(a) A single auxiliary complete matrix is used to complete a single incomplete matrix; (b) Multiple auxiliary complete matrices are used to complete a single incomplete matrix; (c) A single auxiliary complete matrix is used to complete multiple incomplete matrices; (d) This study introduces a complete model matrix M and involves repetition of two steps: first, M is fitted to the current empirical matrices; then, the missing entries are inferred from the fitted model matrix M .

kernel matrix to infer the missing entries of a single incomplete kernel matrix, as shown in Fig. 3.2 (a). Subsequent completion techniques take advantage of multiview learning, where multiple complete auxiliary kernel matrices are used to complete an incomplete kernel matrix, such as in the work of Kato *et al.* [26], Fig. 3.2 (b).

However, the aforementioned kernel matrix completion techniques assume an ideal setting in which many data sources are completely available, while in real-world setting this is not always the case—there will be instances when all that can be acquired are data representations with some lacking information. This situation gives rise to multiple kernel matrix completion problem. Recently, Trivedi *et al.* proposed a multiple kernel matrix completion technique that uses kernel canonical correlation analysis, but with the assumption that at least one kernel matrix must be complete [50], as shown in Fig. 3.2 (c).

In this chapter, a novel algorithm that solves the multiple kernel matrix completion problem is introduced, in a setting where it is possible that all of the empirical kernel matrices have missing entries, as shown in Fig. 3.2 (d). We call our proposed model the *Mutual Kernel Matrix Completion* (MKMC) model. As the name implies, the method completes all the incomplete kernel matrices mutually or simultaneously. Here, a model kernel matrix M fitted to the incomplete kernel matrices is introduced.

²The figures are adapted from [41, 40].

Then, the missing entries of the kernel matrices are inferred from the fitted model matrix \mathbf{M} . The model matrix is in turn updated by fitting it to the completed kernel matrices. The updated model matrix is then used to further improve the estimates for the kernel matrix completion, and the cycle continues. In theory, we introduce an objective function that takes the sum of the Kullback-Leibler (KL) divergences from the empirical kernel matrices to the model matrix. This objective function is then minimized via Expectation-Maximization (EM) algorithm. Interestingly enough, the optimal solutions given by MKMC for the E-step and M-step are in closed forms and hence, the kernel completion problem can be solved efficiently.

3.1 Related Works

A recent work that is similar to our problem setting is [6], where the kernel matrices need not be complete a priori. The completion techniques, however, are different. In [6], an incomplete kernel matrix is expressed as a convex combination of the other kernel matrices through the use of learned reconstruction weights. On the other hand, following [55, 26, 51], our work relates the kernel matrices to the covariance of zero-mean Gaussians, allowing us to use the EM algorithm to minimize the objective function [1]. Moreover, to assess the distances between the kernel matrices to fit a parametric model, [6] employs the Euclidean metric and in doing so, the positive-definiteness of the resulting kernel matrices is not assured. Our work follows that of [51], where KL-divergence and in turn, LogDet divergence [33, 16] is employed, keeping the positive-definiteness of the kernel matrices [40].

In this chapter, the problem setting for this study is first described. Then, the proposed algorithm, the MKMC method, will be presented, followed by a discussion of MKMC algorithm in a statistical framework. The efficacy of the method is then demonstrated through experiments, and the chapter is capped off with some derivations not shown in the main discussion.

3.2 Problem Setting

Suppose that in the analysis of ℓ objects, K relevant data sources are available. Let the corresponding $\ell \times \ell$ symmetric kernel matrices derived from these data sources be denoted as $\mathbf{Q}^{(1)}, \dots, \mathbf{Q}^{(K)}$, whose entries are the relationships among the ℓ objects in the corresponding data sources. Suppose also that some information about the objects at hand are not available, leading to kernel matrices with missing entries in rows and columns, as depicted in Fig. 3.2 (d). When the rows and columns of $\mathbf{Q}^{(k)}$ are rearranged such that the information is available for the first $n_k < \ell$ objects and

unavailable for the remaining $m_k := \ell - n_k$ objects, the result is a block matrix

$$\mathbf{Q}_{vh,vh}^{(k)} = \begin{pmatrix} \mathbf{Q}_{v,v}^{(k)} & \mathbf{Q}_{v,h}^{(k)} \\ \mathbf{Q}_{h,v}^{(k)} & \mathbf{Q}_{h,h}^{(k)} \end{pmatrix}, \quad (3.1)$$

where the submatrix $\mathbf{Q}_{v,v}^{(k)} \in \mathbb{S}_{++}^{n_k}$ have visible entries; while the $n_k \times m_k$ submatrix $\mathbf{Q}_{v,h}^{(k)}$, $m_k \times n_k$ submatrix $\mathbf{Q}_{h,v}^{(k)} = \left(\mathbf{Q}_{v,h}^{(k)}\right)^\top$, and $m_k \times m_k$ submatrix $\mathbf{Q}_{h,h}^{(k)}$ have entries that are yet to be inferred.

The proposed MKMC algorithm mutually completes the kernel matrices by inferring the missing entries in $\mathbf{Q}_{v,h}^{(1)}, \dots, \mathbf{Q}_{v,h}^{(K)}$ and $\mathbf{Q}_{h,h}^{(1)}, \dots, \mathbf{Q}_{h,h}^{(K)}$, through the available information from $\mathbf{Q}_{v,v}^{(1)}, \dots, \mathbf{Q}_{v,v}^{(K)}$. After inference of the missing entries, the rows and columns of $\mathbf{Q}_{vh,vh}^{(k)}$ are reordered back to $\mathbf{Q}^{(k)}$.

3.3 Mutual Kernel Matrix Completion Method

3.3.1 The Objective Function

To formulate the objective function, the distance between two matrices must first be specified so as to fit a parametric model. In this study, the Kullback-Leibler (KL) divergence, defined as the distance between two probability distributions [2], is used:

$$\text{KL}(q, p) = \int q(\mathbf{x}) \log \frac{q(\mathbf{x})}{p(\mathbf{x})} d\mathbf{x}, \quad (3.2)$$

where q is called the empirical distribution, p is the model distribution, and \mathbf{x} is an ℓ -dimensional random variate.

Now, consider a positive definite matrix \mathbf{M} and associate it to the model distribution

$$p := \mathcal{N}(\mathbf{0}, \mathbf{M}) \quad (3.3)$$

as the covariance of a zero-mean Gaussian. Similarly, let the kernel matrices $\mathbf{Q}^{(1)}, \dots, \mathbf{Q}^{(K)}$ be associated to the covariance of zero-mean Gaussian distributions as follows:

$$q_1 := \mathcal{N}(\mathbf{0}, \mathbf{Q}^{(1)}), \dots, q_K := \mathcal{N}(\mathbf{0}, \mathbf{Q}^{(K)}). \quad (3.4)$$

Doing so, the KL divergence of the model matrix \mathbf{M} from $\mathbf{Q}^{(k)}$ can be obtained as

$$\text{KL}(\mathbf{Q}^{(k)}, \mathbf{M}) = \frac{1}{2} \left[\log \det \mathbf{M} - \log \det \mathbf{Q}^{(k)} + \text{Tr}(\mathbf{M}^{-1} \mathbf{Q}^{(k)}) - \ell \right], \quad (3.5)$$

where the size of matrices \mathbf{M} and $\mathbf{Q}^{(k)}$ is $\ell \times \ell$.

Now that the distance between two matrices has been determined, the objective function can now be formulated. The objective function is taken as the sum of the KL divergences:

$$J(\mathcal{H}, \mathbf{M}) := \sum_{k=1}^K \text{KL}(\mathbf{Q}^{(k)}, \mathbf{M}), \quad (3.6)$$

where

$$\mathcal{H} := \left\{ \mathbf{Q}_{v,h}^{(k)}, \mathbf{Q}_{h,h}^{(k)} \right\}_{k=1}^K \quad (3.7)$$

denotes the set of submatrices with missing entries, and \mathbf{M} is the model matrix.

The objective function can also be taken as a sum of LogDet divergences, where the LogDet divergence is defined as

$$\text{LogDet}(\mathbf{Q}^{(k)}, \mathbf{M}) := \frac{1}{2} \left[\log \det \mathbf{M} - \log \det \mathbf{Q}^{(k)} + \langle \mathbf{M}^{-1}, \mathbf{Q}^{(k)} - \mathbf{M} \rangle \right].$$

Indeed, from (3.6) and (3.5):

$$\begin{aligned} J(\mathcal{H}, \mathbf{M}) &= \frac{K}{2} \log \det \mathbf{M} - \frac{1}{2} \sum_{k=1}^K \log \det \mathbf{Q}^{(k)} + \frac{1}{2} \sum_{k=1}^K \text{Tr}(\mathbf{M}^{-1} \mathbf{Q}^{(k)}) - \frac{K\ell}{2} \\ &= \frac{K}{2} \log \det \mathbf{M} - \frac{1}{2} \sum_{k=1}^K \log \det \mathbf{Q}^{(k)} + \frac{1}{2} \sum_{k=1}^K \text{Tr}(\mathbf{M}^{-1} \mathbf{Q}^{(k)} - \mathbf{I}_\ell) \\ &= \frac{K}{2} \log \det \mathbf{M} - \frac{1}{2} \sum_{k=1}^K \log \det \mathbf{Q}^{(k)} + \frac{1}{2} \sum_{k=1}^K \langle \mathbf{M}^{-1}, \mathbf{Q}^{(k)} - \mathbf{M} \rangle \\ &= \sum_{k=1}^K \text{LogDet}(\mathbf{Q}^{(k)}, \mathbf{M}). \end{aligned} \quad (3.8)$$

An advantage of using LogDet divergence is its automatic enforcement of positive-definiteness to the resulting kernel matrices [33, 16]. Having introduced the model matrix \mathbf{M} and the objective function J , MKMC algorithm infers the missing entries by minimizing J with respect to \mathcal{H} and \mathbf{M} , so that the kernel matrices $\{\mathbf{Q}^{(k)}\}_{k=1}^K$ are as close to each other as possible through \mathbf{M} .

3.3.2 MKMC Algorithm

The MKMC algorithm minimizes the objective function (3.6) by iteratively doing the following steps:

1. *Imputation step.* Fix the model matrix \mathbf{M} and minimize J with respect to \mathcal{H} :

$$\mathcal{H} := \underset{\mathcal{H}}{\text{argmin}} J(\mathcal{H}, \mathbf{M}); \quad \text{and} \quad (3.9)$$

Algorithm 1 MKMC Algorithm.

Input: Incomplete kernel matrices $\mathbf{Q}^{(1)}, \dots, \mathbf{Q}^{(K)}$.
Output: Completed kernel matrices $\mathbf{Q}^{(1)}, \dots, \mathbf{Q}^{(K)}$.

```

1: begin
2: Initialize  $\{\mathbf{Q}^{(k)}\}_{k=1}^K$  by imputing zeros in the missing entries;
3: Initialize the model matrix as  $\mathbf{M} := \frac{1}{K} \sum_{k=1}^K \mathbf{Q}^{(k)}$ ;
4: repeat
5:   for all  $k \in \{1, \dots, K\}$  do
6:     Reorder and partition  $\mathbf{Q}^{(k)}$  as  $\mathbf{Q}_{vh,vh}^{(k)}$ ;
7:     Reorder and partition  $\mathbf{M}$  as  $\mathbf{M}_{vh,vh}^{(k)}$ ;
8:      $\mathbf{Q}_{v,h}^{(k)} := \mathbf{Q}_{v,v}^{(k)} \left( \mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{M}_{v,h}^{(k)}$ ;
9:      $\mathbf{Q}_{h,h}^{(k)} := \mathbf{M}_{h,h}^{(k)} - \mathbf{M}_{h,v}^{(k)} \left( \mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{M}_{v,h}^{(k)}$ 
        $+ \mathbf{M}_{h,v}^{(k)} \left( \mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{Q}_{v,v}^{(k)} \left( \mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{M}_{v,h}^{(k)}$ ;
10:   end for
11:   Update  $\mathbf{M}$  as  $\mathbf{M} := \frac{1}{K} \sum_{k=1}^K \mathbf{Q}^{(k)}$ .
12: until convergence
13: end.

```

2. *Model update step.* Minimize J with respect \mathbf{M} while \mathcal{H} is held fixed:

$$\mathbf{M} := \underset{\mathbf{M}}{\operatorname{argmin}} J(\mathcal{H}, \mathbf{M}). \quad (3.10)$$

These steps correspond to the E-step and M-step of the EM algorithm, the details of which will be discussed in the next section. Iteratively doing these steps decreases the objective function monotonically.

As mentioned earlier, the optimal solutions for the E- and M-steps of MKMC algorithm are given in closed forms. The closed form solutions are:

$$\mathbf{Q}_{v,h}^{(k)} := \mathbf{Q}_{v,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{M}_{v,h}^{(k)}; \quad (3.11)$$

$$\begin{aligned} \mathbf{Q}_{h,h}^{(k)} := & \mathbf{M}_{h,h}^{(k)} - \mathbf{M}_{h,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{M}_{v,h}^{(k)} \\ & + \mathbf{M}_{h,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{Q}_{v,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{M}_{v,h}^{(k)} \end{aligned} \quad (3.12)$$

for the imputation step, and

$$\mathbf{M} := \frac{1}{K} \sum_{k=1}^K \mathbf{Q}^{(k)} \quad (3.13)$$

for the model update step. Observe that the model update, which resulted from minimizing J with respect to \mathbf{M} , is simply the average of the updated kernel matrices (proof in Sect. 3.7.3). Meanwhile, the values inferred for the submatrices $\mathbf{Q}_{v,h}^{(k)}$ and $\mathbf{Q}_{h,h}^{(k)}$ are in terms of the available values from the submatrix $\mathbf{Q}_{v,v}^{(k)}$, and of those in the

submatrices of the reordered and partitioned \mathbf{M} :

$$\mathbf{M}_{vh,vh}^{(k)} = \begin{pmatrix} \mathbf{M}_{v,v}^{(k)} & \mathbf{M}_{v,h}^{(k)} \\ \mathbf{M}_{h,v}^{(k)} & \mathbf{M}_{h,h}^{(k)} \end{pmatrix}, \quad (3.14)$$

where the sizes of the submatrices of $\mathbf{M}_{vh,vh}^{(k)}$ are same as those of the k -th reordered kernel matrix $\mathbf{Q}_{vh,vh}^{(k)}$. The MKMC algorithm can now be outlined as in Alg. 1.

3.4 MKMC Algorithm in a Statistical Framework

This section shows how the MKMC algorithm infers the missing entries in a vectorial data through a probabilistic approach. As mentioned in the previous section, MKMC algorithm minimizes the objective function by repeating two steps: one that minimizes the objective function with respect to the unobserved values in the vectorial data; and one that minimizes the objective function with respect to the model parameter. In theory, MKMC algorithm introduces a model parameter and finds its best fit to a probabilistic model by maximizing the model parameter's likelihood. It then shows that the steps involved in the MKMC algorithm comprise the E-step and M-step in the EM algorithm. In this section, the connection between the MKMC and EM algorithms will be established after revisiting some details about the EM algorithm.

3.4.1 An Overview of the EM Algorithm

The EM algorithm finds the maximum likelihood solutions for probabilistic models having latent (or unobserved) variables [8, 34, 17]. The algorithm works by first computing the expectation of the log-likelihood of the current parameter estimates (the Expectation or E-step), then finding the parameter values that maximize the expectation found in the E-step (the Maximization or M-step). These steps are repeated alternately to find the best fit for the parameters.

To begin with, consider a model parameter Θ of a probabilistic model $p(\mathbf{V} | \Theta)$ from observed data \mathbf{V} . The log-likelihood of the model parameter Θ given the observed data \mathbf{V} is

$$L(\Theta; \mathbf{V}) := \log p(\mathbf{V} | \Theta). \quad (3.15)$$

Taking the expectation of (3.15) under an empirical distribution $q(\mathbf{V})$, the following generalized form can now be considered [1]:

$$L(\Theta; q) := \mathbb{E}_{q(\mathbf{V})} [\log p(\mathbf{V} | \Theta)]. \quad (3.16)$$

This function can now be maximized with respect to Θ by alternately repeating the E- and M-steps.

However, the unobserved data \mathbf{H} must also be taken into account, comprising the *complete* data (\mathbf{V}, \mathbf{H}) . The joint distribution of the complete data is in the exponential family [53, 19], and its log-likelihood function takes the form

$$\log p(\mathbf{V}, \mathbf{H} | \Theta) = \langle \mathbf{S}(\mathbf{V}, \mathbf{H}), \mathbf{G}(\Theta) \rangle - \mathbf{A}(\Theta), \quad (3.17)$$

for some vector-valued or matrix-valued functions $\mathbf{A}(\cdot)$, $\mathbf{G}(\cdot)$, and $\mathbf{S}(\cdot)$.

Since our knowledge about the unobserved data depends only on the posterior distribution of \mathbf{H} , the complete-data log-likelihood cannot be used. The posterior distribution of \mathbf{H} , denoted by $p(\mathbf{H} | \mathbf{V}, \Theta^{(t-1)})$, by the way, can simply be interpreted as the probability obtained after data \mathbf{V} has been observed. Hence, the expected value of the complete-data log-likelihood under the posterior distribution of \mathbf{H} is used instead:

$$\mathbb{E}_{p(\mathbf{H} | \mathbf{V}, \Theta^{(t-1)})} \mathbb{E}_{q(\mathbf{V})} \left[\log p(\mathbf{V}, \mathbf{H} | \Theta^{(t-1)}) \right], \quad (3.18)$$

which corresponds to the E-step of EM algorithm. Maximizing (3.18) gives rise to a new parameter estimate $\Theta^{(t)}$, which on the other hand corresponds to the M-step.

Going back to the joint distribution of \mathbf{V} and \mathbf{H} , an important property of exponential family distribution is that the maximum likelihood estimator depends on the data only through $\mathbf{S}(\mathbf{V}, \mathbf{H})$, hence termed as the *sufficient statistics* of the distribution [8]. Thus, in the E-step of the t -th iteration, the expectation in (3.18) is computed as

$$\mathbf{S}^{(t)} := \mathbb{E}_{t-1} [\mathbf{S}(\mathbf{V}, \mathbf{H})], \quad (3.19)$$

where $\mathbb{E}_{t-1} := \mathbb{E}_{p(\mathbf{H} | \mathbf{V}, \Theta^{(t-1)})} \mathbb{E}_{q(\mathbf{V})}$, for simplicity. Moving on to the M-step, the computed expected value, denoted by $Q(\Theta)$, is maximized to obtain the model parameter update $\Theta^{(t)}$:

$$\Theta^{(t)} = \underset{\Theta}{\operatorname{argmax}} Q(\Theta), \quad (3.20)$$

where the Q -function is defined as

$$\begin{aligned} Q(\Theta) &:= \mathbb{E}_{t-1} [\log p(\mathbf{V}, \mathbf{H} | \Theta)] \\ &= \langle \mathbf{S}^{(t)}, \mathbf{G}(\Theta) \rangle - \mathbf{A}(\Theta). \end{aligned} \quad (3.21)$$

Meanwhile, the convergence of the EM algorithm is ensured from the following property:

Proposition 3.4.1. *The EM algorithm produces a nondecreasing series of the log-likelihood:*

$$L(\Theta^{(t-1)}; q) \leq L(\Theta^{(t)}; q), \quad \text{for } t \in \mathbb{N}.$$

Proof. For any unobserved data \mathbf{H} and $p(\mathbf{H}|\mathbf{V}, \Theta) > 0$, the following holds:

$$\log p(\mathbf{V}|\Theta) = \log p(\mathbf{V}, \mathbf{H}|\Theta) - \log p(\mathbf{H}|\mathbf{V}, \Theta). \quad (3.22)$$

Then, taking the expectation of both sides under $q(\mathbf{V})$ and $p(\mathbf{H}|\mathbf{V}, \Theta^{(t-1)})$, Eq. (3.22) becomes

$$L(\Theta; q) = \mathcal{Q}(\Theta) - \mathbb{E}_{t-1}[\log p(\mathbf{H}|\mathbf{V}, \Theta)]. \quad (3.23)$$

Since $\mathcal{Q}(\Theta^{(t-1)}) \leq \mathcal{Q}(\Theta^{(t)})$ from the update rule in M-step, and it follows directly from the nonnegativity of the KL divergence that $\mathbb{E}_{t-1}[\log p(\mathbf{H}|\mathbf{V}, \Theta^{(t-1)})] \geq \mathbb{E}_{t-1}[\log p(\mathbf{H}|\mathbf{V}, \Theta^{(t)})]$ [8], the following inequality is obtained:

$$\begin{aligned} L(\Theta^{(t-1)}; q) &= \mathcal{Q}(\Theta^{(t-1)}) - \mathbb{E}_{t-1}[\log p(\mathbf{H}|\mathbf{V}, \Theta^{(t-1)})] \\ &\leq \mathcal{Q}(\Theta^{(t)}) - \mathbb{E}_{t-1}[\log p(\mathbf{H}|\mathbf{V}, \Theta^{(t)})] \\ &= L(\Theta^{(t)}; q). \end{aligned} \quad (3.24)$$

□

It has been shown that for every complete iteration of the EM algorithm, the log-likelihood of the model parameter increases monotonically.

3.4.2 MKMC Algorithm is an EM Algorithm

The previous discussion has shown how EM algorithm fits a model parameter to a model distribution. The expectation of the log-likelihood function under a posterior distribution of the unobserved data is computed in the E-step, then the obtained expectation function is maximized in the M-step. In this section, the connection between MKMC algorithm and EM algorithm will be established.

First off, let the vectors $\mathbf{x}_1, \dots, \mathbf{x}_K \in \mathbb{R}^\ell$ represent the complete data, where each entry of \mathbf{x}_k is a row or column in $\mathbf{Q}^{(k)}$. Also, let the subvectors $\mathbf{v}_k \in \mathbb{R}^{n_k}$ and $\mathbf{h}_k \in \mathbb{R}^{m_k}$ of \mathbf{x}_k correspond to the observed and unobserved data in the k -th kernel matrix, respectively. Finally, let $\mathbf{V} := (\mathbf{v}_1, \dots, \mathbf{v}_K)$ and $\mathbf{H} := (\mathbf{h}_1, \dots, \mathbf{h}_K)$. From here, the empirical and model distributions can now be discussed.

From (3.4), the empirical distribution of the observed data \mathbf{v}_k can be expressed as

$$q_k(\mathbf{v}_k) = \mathcal{N}(\mathbf{v}_k | \mathbf{0}, \mathbf{Q}_{\mathbf{v}, \mathbf{v}}^{(k)}), \quad (3.25)$$

so accordingly, $q(\mathbf{V})$ can be written as a product of Gaussians:

$$q(\mathbf{V}) = \prod_{k=1}^K \mathcal{N}(\mathbf{v}_k | \mathbf{0}, \mathbf{Q}_{\mathbf{v}, \mathbf{v}}^{(k)}), \quad (3.26)$$

and the second moment of $q(\mathbf{V})$ is thus given by

$$\mathbb{E}_{q(\mathbf{V})} [\mathbf{v}_k \mathbf{v}_k^\top] = \mathbf{Q}_{v,v}^{(k)}. \quad (3.27)$$

Following the description from (3.3), the model distribution of the complete data is given by

$$p(\mathbf{V}, \mathbf{H} | \mathbf{M}) = \prod_{k=1}^K \mathcal{N}(\mathbf{x}_k | \mathbf{0}, \mathbf{M}), \quad (3.28)$$

and its logarithm takes the form

$$\log p(\mathbf{V}, \mathbf{H} | \mathbf{M}) = \langle \mathbf{S}(\mathbf{V}, \mathbf{H}), \mathbf{G}(\mathbf{M}) \rangle - \mathbf{A}(\mathbf{M}), \quad (3.29)$$

the derivations of which are given in Sect. 3.7. Here, the functions $\mathbf{A}(\cdot)$, $\mathbf{G}(\cdot)$, and $\mathbf{S}(\cdot)$ are defined as

$$\begin{aligned} \mathbf{S}(\mathbf{V}, \mathbf{H}) &:= \sum_{k=1}^K \mathbf{x}_k \mathbf{x}_k^\top, & \mathbf{G}(\mathbf{M}) &:= -\frac{1}{2} \mathbf{M}^{-1}, \\ \text{and } \mathbf{A}(\mathbf{M}) &:= \frac{K}{2} \log \det \mathbf{M} + \text{const.} \end{aligned} \quad (3.30)$$

Observe that (3.29) is now similar in form as (3.17). Having established the distributions, the E-step and M-step can now be tackled.

Since the joint distribution in (3.28) is in the exponential family, it suffices to consider only the expectation of the sufficient statistics:

$$\mathbb{E}_{t-1} [\mathbf{S}(\mathbf{V}, \mathbf{H})] = \sum_{k=1}^K \mathbb{E}_{t-1} [\mathbf{x}_k \mathbf{x}_k^\top]. \quad (3.31)$$

However, since vector \mathbf{x}_k consists of a subvector \mathbf{v}_k for the observed data and a subvector \mathbf{h}_k for the unobserved data, the following submatrices of $\mathbb{E}_{t-1} [\mathbf{x}_k \mathbf{x}_k^\top]$ need to be considered:

$$\mathbb{E}_{t-1} [\mathbf{v}_k \mathbf{v}_k^\top], \quad \mathbb{E}_{t-1} [\mathbf{v}_k \mathbf{h}_k^\top], \quad \text{and} \quad \mathbb{E}_{t-1} [\mathbf{h}_k \mathbf{h}_k^\top].$$

Note that from (3.27), $\mathbb{E}_{t-1} [\mathbf{v}_k \mathbf{v}_k^\top] = \mathbb{E}_{p(\mathbf{H} | \mathbf{V}, \mathbf{M}^{(t-1)})} [\mathbf{Q}_{v,v}^{(k)}] = \mathbf{Q}_{v,v}^{(k)}$, where here, $\mathbf{M}^{(t-1)}$ denotes the current value of the model parameter \mathbf{M} . Meanwhile, it can be

shown that the expectations of the remaining submatrices are

$$\begin{aligned}\mathbb{E}_{t-1} \left[\mathbf{v}_k \mathbf{h}_k^\top \right] &= \mathbf{Q}_{v,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{M}_{v,h}^{(k)} \\ &= \mathbf{Q}_{v,h}^{(k)},\end{aligned}\tag{3.32}$$

$$\begin{aligned}\mathbb{E}_{t-1} \left[\mathbf{h}_k \mathbf{h}_k^\top \right] &= \mathbf{M}_{h,h}^{(k)} - \mathbf{M}_{h,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{M}_{v,h}^{(k)} \\ &\quad + \mathbf{M}_{h,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{Q}_{v,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)} \right)^{-1} \mathbf{M}_{v,h}^{(k)} \\ &= \mathbf{Q}_{h,h}^{(k)},\end{aligned}\tag{3.33}$$

which coincide with the closed-form solutions (3.11) and (3.12) of MKMC algorithm in the imputation step. The expectation of the sufficient statistics is therefore

$$\mathbf{S}^{(t)} = \mathbb{E}_{t-1} [\mathbf{S}(\mathbf{V}, \mathbf{H})] = \sum_{k=1}^K \mathbb{E}_{t-1} [\mathbf{x}_k \mathbf{x}_k^\top] = \sum_{k=1}^K \mathbf{Q}^{(k)},\tag{3.34}$$

and the corresponding \mathcal{Q} -function is given by

$$\begin{aligned}\mathcal{Q}(\mathbf{M}) &= \left\langle \mathbf{S}^{(t)}, \mathbf{G}(\mathbf{M}) \right\rangle - \mathbf{A}(\mathbf{M}) \\ &= \left\langle \sum_{k=1}^K \mathbf{Q}^{(k)}, -\frac{1}{2} \mathbf{M}^{-1} \right\rangle - \frac{K}{2} \log \det \mathbf{M} + \text{const.}\end{aligned}\tag{3.35}$$

Optimizing this function with respect to \mathbf{M} gives the model parameter update (3.13). The derivation for the model parameter update and for (3.32) and (3.33) can be found in Sect. 3.7.

From the discussions above, it can be seen that the steps involved in MKMC algorithm correspond to the steps in the EM algorithm. To see why the optimal solution from the EM algorithm is also the optimal solution for MKMC algorithm, it should be noted that when the expressions for the log-likelihood function $L(\mathbf{M})$ and the objective function $J(\mathcal{H}, \mathbf{M})$ are examined,

$$\begin{aligned}L(\mathbf{M}) &:= \sum_{k=1}^K \log p(\mathbf{V}, \mathbf{H} | \mathbf{M}) \\ &= -\frac{K}{2} \log \det \mathbf{M} - \frac{1}{2} \left\langle \mathbf{S}(\mathbf{V}, \mathbf{H}), \mathbf{M}^{-1} \right\rangle + \text{const.}\end{aligned}\tag{3.36}$$

$$\begin{aligned}J(\mathcal{H}, \mathbf{M}) &= \sum_{k=1}^K \text{KL}(\mathbf{Q}^{(k)}, \mathbf{M}) \\ &= \frac{K}{2} \log \det \mathbf{M} + \frac{1}{2} \left\langle \mathbf{S}(\mathbf{V}, \mathbf{H}), \mathbf{M}^{-1} \right\rangle \\ &\quad - \frac{1}{2} \sum_{k=1}^K \log \det \mathbf{Q}^{(k)} + \text{const.}\end{aligned}\tag{3.37}$$

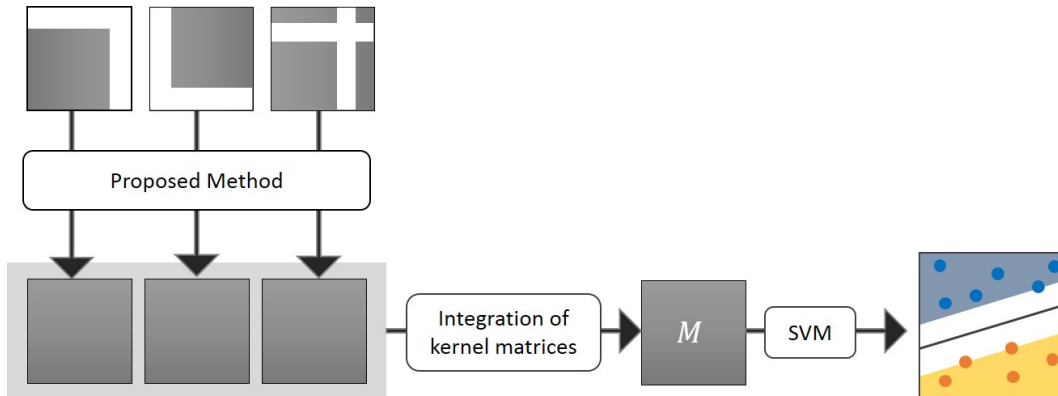


FIGURE 3.3: Experimental set-up.

The proposed method completes the incomplete kernel matrices, then integrates them for an SVM classifier to train on a subset of data in \mathbf{M} (training data). The trained classifier then predicts which of the remaining data points (testing data) are membrane and which are non-membrane proteins.

the terms involving their sufficient statistic are inversely proportional. Hence, the log-likelihood solution for the maximum of $L(\mathbf{M})$ is also the solution for the minimum of $J(\mathcal{H}, \mathbf{M})$.

3.5 Experiments and Results

As representations of generalized similarities among data points, the completed kernel matrices must be able to characterize data well so that underlying patterns can be easily detected. Hence, when inferring missing kernel matrix values, the relationships among the data points must also be considered. In this section, the efficacy of MKMC model is tested not only by the model's prediction accuracy, but by its ability to retain the interrelationships among the data points as well.

In this experiment, the proposed method is applied to the problem of protein membrane classification, following the work of [30].

3.5.1 Experimental Setting

The experiment proceeds as follows: the incomplete kernel matrices will be completed by MKMC algorithm by inferring the missing entries, then integrates the completed kernel matrices to be used for SVM training. The MKMC algorithm's model update step provides a way for the completed kernel matrices to integrate, that is, via their average. From a subset of these integrated matrices, called *training data*, an SVM classifier will be trained. Classification performance is then evaluated on the remaining data points that were not included in the training phase (*testing data*),

through the number of times the classifier has correctly categorized the testing data. Figure 3.3 illustrates the setting for the experiment.

Data Set

The data set³ consists of seven kernel matrices derived from various yeast protein data types, such as protein-protein interaction data, gene expression data, and amino-acid sequence data. These information are important in membrane classification task, as they provide the characteristics in common with membrane proteins and non-membrane proteins. For instance, membrane proteins must have more similarities in their amino-acid sequences, or in their gene expressions, compared to proteins that are non-membrane. Moreover, membrane proteins must have interacted more often than the non-membrane ones. Due to the valuable information that these data types provide, they can be utilized as data sources for membrane classification task.

Membrane proteins are proteins that are anchored in biological membranes (selective penetrable barriers within living things), either permanently (integral membrane proteins) or temporarily (peripheral membrane proteins). Much research has been dedicated to membrane protein classification, as most modern medicinal drugs are targeted towards membrane proteins.

Lanckriet et al. [30] provided descriptions on how the kernel matrices were derived from real data. For instance, pairwise comparison of amino-acid sequences of proteins via BLAST⁴ algorithm yields a matrix of score vectors. Since this matrix is nonpositive semidefinite, a linear kernel is evaluated to each matrix entries, resulting to a valid kernel matrix (i. e., a symmetric, positive semidefinite matrix), containing all the information acquired from amino-acid sequences of a set of proteins.

Another kernel matrix formulation from real-world data is through an interaction matrix, where the rows and columns correspond to proteins, and the binary entries correspond to whether or not two proteins interact. The kernel matrix corresponding to this protein-protein interaction is then formed from the inner products of rows and columns of the said binary matrix, where larger inner products indicate more similar interaction pattern.

Of the seven kernel matrices used for this experiment, four are from primary sequence data, two are from protein-protein interaction data, and one is from gene expression data. The annotations (or labels) for the proteins are provided by CYGD [35]. Membrane proteins are labeled as +1 while non-membrane proteins are labeled as -1. Of the 2,318 proteins considered, 497 belong to the positive class.

³The data set is obtained from <https://noble.gs.washington.edu/proj/sdp-svm/>.

⁴Basic Local Alignment Search Tool is one of the algorithms available for comparing primary biological sequences.

Completion Task

To generate incomplete kernel matrices, randomly-picked rows and columns are “removed” from each data source, i. e., the selected rows and columns are replaced by undetermined values (such as zero) so as not to change the size of the matrix. Various percentage of missed data is considered, from 0% (as baseline) to 90%, where selection of rows and columns to be missed is done by increments of 10%.

Traditional completion methods that fit the problem setting are considered for comparison. Such methods are the zero-imputation method and mean-imputation method. From the names themselves, zero-imputation method imputes zeros on the “missing” entries (which is also used to initialize the kernel matrices in MKMC algorithm), while mean-imputation method⁵ imputes the mean of the remaining entries in the corresponding rows and columns.

In this task, each of the completion methods (MKMC algorithm, zero-imputation, and mean-imputation) is performed on each percentage of missing entries (10% to 90%) in each of the kernel matrices ($\mathbf{Q}^{(1)}, \dots, \mathbf{Q}^{(7)}$), repeated five times.

Classification Task

For this task, after completion of the kernel matrices, an SVM classifier [12] is trained on a subset of the combined kernel matrices. Several number of training points are considered, including 200 and 1,000 training data points. Similar to the completion task, the selection of training data is done in random. The remaining data points served as the testing data, where classification performance is evaluated.

3.5.2 Experimental Results

Classification Performance

The classification performance of each completion method is evaluated using receiver operating characteristic (ROC) score, or area under the ROC curve (AUC). This curve captures the categorization ability of a binary classifier as the discrimination threshold is varied. To compare the generalization performance of certain models, the ROC scores (or AUCs) of each model are being compared. Models with higher ROC scores have better classification performance than the other models. ROC scores may range from 0.5 (random prediction) to 1.0 (perfect prediction). The classification performance of the three completion methods (when the ratio of missed entries had

⁵A description on how the kernel matrices are completed by this method can be found in [41].

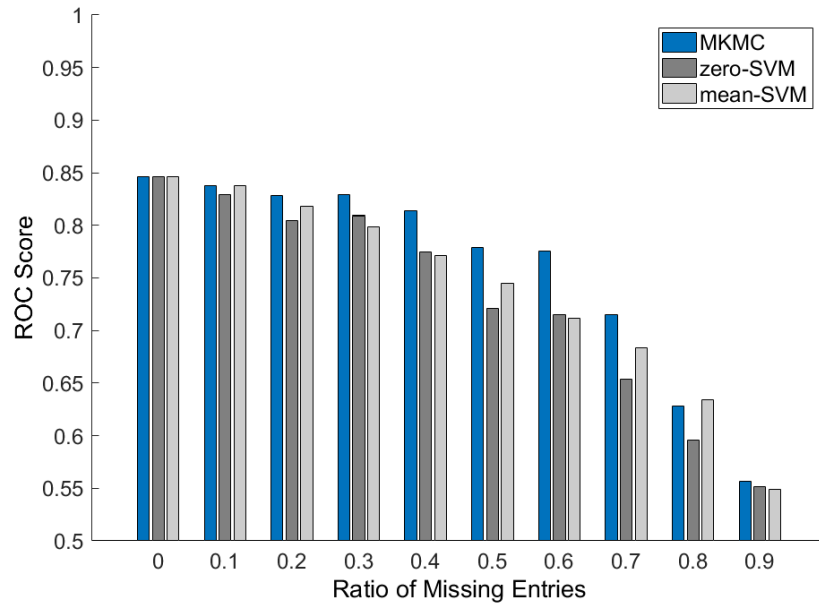
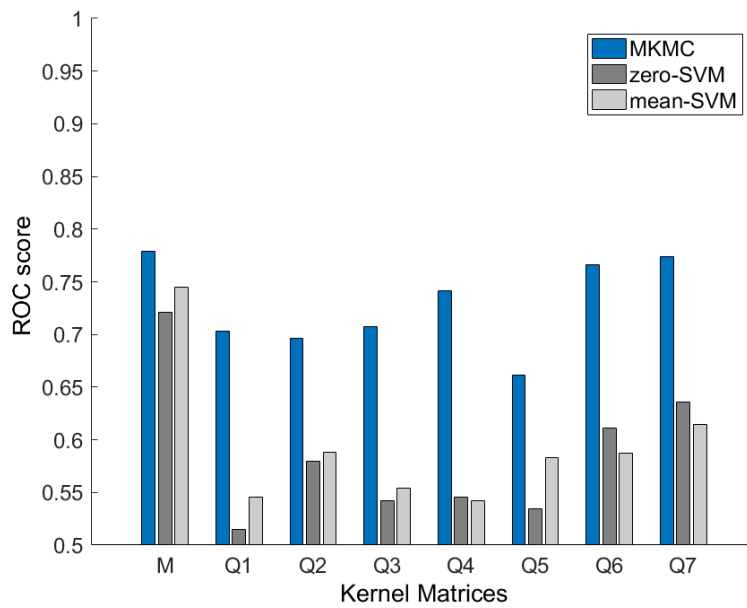


FIGURE 3.4: Classification performance for varying number of missing entries.

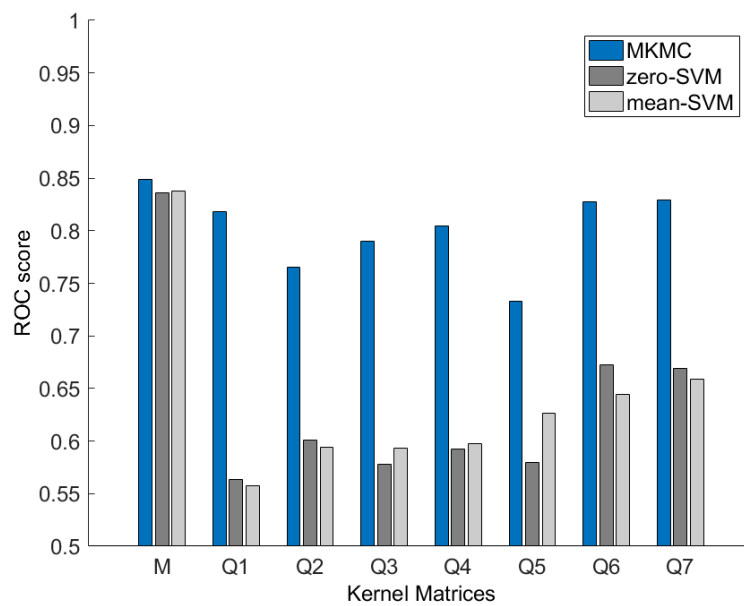
The height of the bars represent the ROC scores of the completion methods when the ratio of missing entries is varied. Here, the SVM classifier is trained on 200 data points of the model matrix.

been varied) is illustrated in Fig. 3.4. Here, the methods zero-imputation and mean-imputation are renamed as zero-SVM and mean-SVM, respectively, after an SVM classifier has been trained on their respective integrated kernel matrices (average of the kernel matrices). Although the classification performance of MKMC model (as well as those of zero-SVM and mean-SVM) declines as the number of missed entries increases, it has remained the victor over the other two completion methods in most of the cases, especially at 40% to 70% missed entries.

The classification performance was also tested on the individual completed kernel matrices to see whether or not the completion methods have preserved the relationships among the data. As shown in Fig. 3.5, the proposed model MKMC bested the other two completion methods on the individual classification performance. From a one-sample t -test, the mean ROC score of MKMC model (0.779) is statistically significant from those of zero-SVM (0.721) and mean-SVM (0.745), with P-values $8.32 \cdot 10^{-5}$ and $1.40 \cdot 10^{-3}$, respectively, for 200 training data points. In the case of 1,000 training data points, the mean ROC score of MKMC model (0.848) is also statistically significant from zero-SVM (0.836, P-value = $3.45 \cdot 10^{-2}$) and mean-SVM (0.832, P-value = $4.60 \cdot 10^{-3}$).



(a) 200 training data



(b) 1,000 training data

FIGURE 3.5: Classification performance for each kernel matrix.

The height of each bar shows the ROC scores of the completion methods for 50% missed data. The SVM classifier was trained on (a) 200 and (b) 1,000 data points.

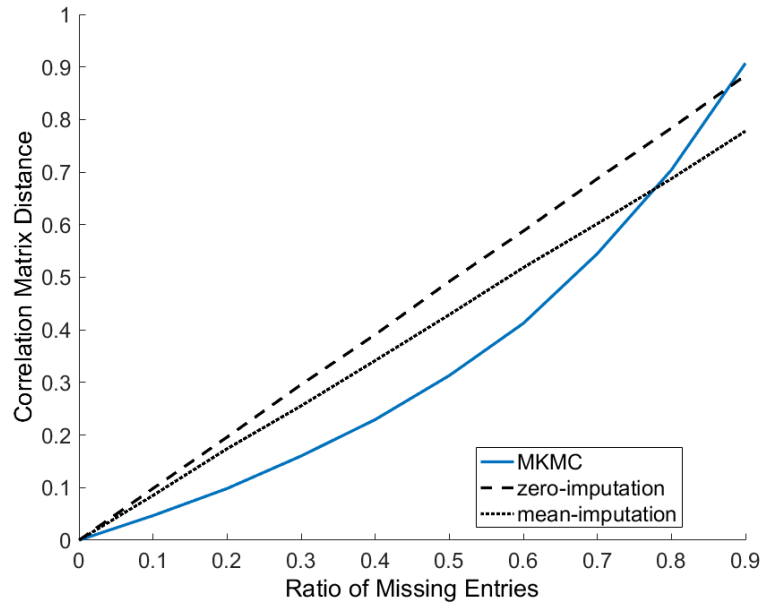


FIGURE 3.6: Prediction accuracy of the completion methods.

The lines show the prediction accuracy of the completion methods as the number of missing entries increases. Better prediction accuracy is implied by lower correlation matrix distance value.

Completion Accuracy

The criterion used in evaluating how accurate the missing entries were recovered is the mean correlation matrix distance between the original and completed kernel matrices:

$$\frac{1}{K} \sum_{k=1}^K \left(1 - \frac{\text{Tr}(\mathbf{Q}^{(k)} \hat{\mathbf{Q}}^{(k)})}{\|\mathbf{Q}^{(k)}\|_F \|\hat{\mathbf{Q}}^{(k)}\|_F} \right), \quad (3.38)$$

where $\|\cdot\|_F$ is the Frobenius norm, $\mathbf{Q}^{(k)}$ denotes the “true” kernel matrix while $\hat{\mathbf{Q}}^{(k)}$ denotes the corresponding estimated or completed kernel matrix. In this criterion, lower correlation matrix distance value implies better prediction accuracy. As shown in Fig. 3.6, with the exception of 80% to 90% missed entries, the MKMC model obtains the best prediction accuracy than the other two completion methods in each ratio of missing entries.

3.6 Summary

In this chapter, a novel kernel-based method for mutual completion of kernel matrices is introduced. The proposed algorithm, the MKMC model, relies upon the use of kernel matrices in order to fuse partial descriptions of the data into a single, big picture of the data set at hand. This, in turn, was used to infer the missing entries in each of the kernel matrices representing the relevant data sources for prediction. Aside from the promising capabilities of the MKMC model as shown in the experimental results on real-world data set, a statistical description of the model was also presented, giving the proposed algorithm a principled way to handle and integrate kernel matrices with missing data.

3.7 Derivations and Some Discussions

This section presents the derivations of some expressions and of the closed-form solutions for the MKMC algorithm.

3.7.1 Derivation of (3.28) and (3.29)

From (3.28),

$$\begin{aligned}
 p(\mathbf{V}, \mathbf{H} | \mathbf{M}) &= \prod_{k=1}^K \mathcal{N}(\mathbf{x}_k | \mathbf{0}, \mathbf{M}) \\
 &= \prod_{k=1}^K \frac{1}{(2\pi)^{\frac{\ell}{2}} |\mathbf{M}|^{\frac{1}{2}}} \exp \left\{ -\frac{1}{2} \mathbf{x}_k^\top \mathbf{M}^{-1} \mathbf{x}_k \right\} \\
 &= \frac{1}{(2\pi)^{\frac{K\ell}{2}} |\mathbf{M}|^{\frac{K}{2}}} \exp \left\{ -\frac{1}{2} \sum_{k=1}^K \langle \mathbf{x}_k \mathbf{x}_k^\top, \mathbf{M}^{-1} \rangle \right\} \\
 &= \frac{1}{(2\pi)^{\frac{K\ell}{2}} |\mathbf{M}|^{\frac{K}{2}}} \exp \left\{ \left\langle \sum_{k=1}^K \mathbf{x}_k \mathbf{x}_k^\top, -\frac{1}{2} \mathbf{M}^{-1} \right\rangle \right\},
 \end{aligned} \tag{3.39}$$

and observe that $p(\mathbf{V}, \mathbf{H} | \mathbf{M})$ is in the exponential family. Now, taking the logarithm of both sides, the same expression as (3.29) is obtained:

$$\begin{aligned}
 \log p(\mathbf{V}, \mathbf{H} | \mathbf{M}) &= \left\langle \sum_{k=1}^K \mathbf{x}_k \mathbf{x}_k^\top, -\frac{1}{2} \mathbf{M}^{-1} \right\rangle - \frac{K}{2} \log \det \mathbf{M} + \text{const.} \\
 &= \langle \mathbf{S}(\mathbf{V}, \mathbf{H}), \mathbf{G}(\mathbf{M}) \rangle - \mathbf{A}(\mathbf{M}),
 \end{aligned} \tag{3.40}$$

where the functions $\mathbf{A}(\cdot)$, $\mathbf{G}(\cdot)$, and $\mathbf{S}(\cdot)$ are defined as

$$\begin{aligned} \mathbf{S}(\mathbf{V}, \mathbf{H}) &:= \sum_{k=1}^K \mathbf{x}_k \mathbf{x}_k^\top, & \mathbf{G}(\mathbf{M}) &:= -\frac{1}{2} \mathbf{M}^{-1}, \\ \text{and } \mathbf{A}(\mathbf{M}) &:= \frac{K}{2} \log \det \mathbf{M} + \text{const.} \end{aligned} \quad (3.41)$$

3.7.2 Derivation of (3.32) and (3.33)

Let the parameter model \mathbf{M} be also denoted by Θ . Using the posterior distribution of the unobserved data \mathbf{h}_k

$$p(\mathbf{h}_k | \mathbf{v}_k, \mathbf{M}) = \mathcal{N}\left(\mathbf{h}_k | \mathbf{M}_{h,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)}\right)^{-1} \mathbf{v}_k, \mathbf{M}_{h|v}^{(k)}\right),$$

with

$$\mathbf{M}_{h|v}^{(k)} := \mathbf{M}_{h,h}^{(k)} - \mathbf{M}_{h,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)}\right)^{-1} \mathbf{M}_{v,h}^{(k)},$$

the expectation in (3.32) is derived as follows:

$$\begin{aligned} \mathbb{E}_{t-1} \left[\mathbf{v}_k \mathbf{h}_k^\top \right] &= \mathbb{E}_{q(\mathbf{V})} \left[\mathbf{v}_k \mathbb{E}_{p(\mathbf{H} | \mathbf{V}, \Theta^{(t-1)})} \left[\mathbf{h}_k^\top \right] \right] \\ &= \mathbb{E}_{q(\mathbf{V})} \left[\mathbf{v}_k \left(\mathbf{M}_{h,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)}\right)^{-1} \mathbf{v}_k \right)^\top \right] \\ &= \mathbb{E}_{q(\mathbf{V})} \left[\mathbf{v}_k \mathbf{v}_k^\top \right] \left(\mathbf{M}_{v,v}^{(k)}\right)^{-1} \mathbf{M}_{v,h}^{(k)} \\ &= \mathbf{Q}_{v,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)}\right)^{-1} \mathbf{M}_{v,h}^{(k)} \\ &= \mathbf{Q}_{v,h}^{(k)}. \end{aligned}$$

On the other hand, the expectation in (3.33) is computed as

$$\begin{aligned} \mathbb{E}_{t-1} \left[\mathbf{h}_k \mathbf{h}_k^\top \right] &= \mathbb{E}_{q(\mathbf{V})} \left[\mathbf{M}_{h|v}^{(k)} + \left(\mathbf{M}_{h,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)}\right)^{-1} \mathbf{v}_k \right) \left(\mathbf{M}_{h,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)}\right)^{-1} \mathbf{v}_k \right)^\top \right] \\ &= \mathbf{M}_{h|v}^{(k)} + \mathbf{M}_{h,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)}\right)^{-1} \mathbf{Q}_{v,v}^{(k)} \left(\mathbf{M}_{v,v}^{(k)}\right)^{-1} \mathbf{M}_{v,h}^{(k)} \\ &= \mathbf{Q}_{h,h}^{(k)}. \end{aligned}$$

3.7.3 Derivation of the Model Parameter Update

The model parameter update is the optimal solution of the \mathcal{Q} -function. To obtain the optimal solution, first let the \mathcal{Q} -function be rewritten as

$$\begin{aligned}\mathcal{Q}(\mathbf{M}) &= \left\langle \sum_{k=1}^K \mathbf{Q}^{(k)}, -\frac{1}{2}\mathbf{M}^{-1} \right\rangle - \frac{K}{2} \log \det \mathbf{M} + \text{const.} \\ &= -\frac{1}{2} \sum_{k=1}^K \text{Tr} \left(\mathbf{M}^{-1} \mathbf{Q}^{(k)} \right) + \frac{K}{2} \log \det \mathbf{M}^{-1} + \text{const.}\end{aligned}$$

Taking the derivative of the \mathcal{Q} -function with respect to \mathbf{M}^{-1} , then setting to zero gives

$$-\frac{1}{2} \sum_{k=1}^K \mathbf{Q}^{(k)} + \frac{K}{2} \mathbf{M} = \mathbf{0}.$$

Finally, solving for \mathbf{M} yields $\mathbf{M} = \frac{1}{K} \sum_{k=1}^K \mathbf{Q}^{(k)}$.

Chapter 4

Parametric Models for Mutual Kernel Matrix Completion

This chapter presents alternative methods in addressing the incomplete multiview learning problem, in which the complexity of the model can be controlled. These methods, as variants of the MKMC model introduced in the previous chapter, are appropriately named PCA-MKMC and FA-MKMC. PCA-MKMC employs the probabilistic Principal Component Analysis (PCA) approach in restricting the covariance model, whereas FA-MKMC follows that of the Factor Analysis (FA) model. Probabilistic PCA¹ and FA are linear Gaussian models that model the covariance structure of the data via a small number of latent variables. In particular, many data sets have similar structures since they are all associated with a latent. For instance, in a particular survey question, many respondents may have similar answers or opinions based on their religion. The similarity in responses are then generated by the latent religion. As illustrated in the example, the data points are generated by latent variables, and data structures may lie in a (much) lower dimensional space than the original input space. In such situations, fitting a full covariance model in the input space may seem inappropriate as it considers each and every correlation in the data, and a large number of free parameters needs to be considered.² In essence, dimensionality reduction methods such as (probabilistic) PCA and FA “compress” highly correlated features. When a learning classification algorithm such as SVM is fitted to a data set with highly correlated features, the learned classifier might become prone to overfitting, i. e., the classifier is trained so well in the training set that it cannot correctly classify a new, unseen data very well. Thus, even when there is some information loss in dimensionality reduction, this loss in information might become useful for the learned model to generalize well.

¹The notion of PCA is based on a linear projection of the data onto a lower dimensional subspace to reduce data dimensionality. Tipping and Bishop [49] and Roweis [42] reformulated PCA as the maximum likelihood solution of a probabilistic latent variable model, and called it *probabilistic PCA*.

²Fitting a full covariance matrix $\mathbf{A} \in \mathbb{R}^{\ell \times \ell}$ uses $\ell(\ell + 1)/2$ free parameters.

From the above description, a data point $\mathbf{x}_k \in \mathbb{R}^\ell$ can be thought of as being generated by the underlying data structure $\mathbf{z}_k \in \mathbb{R}^q$ of much lower dimensionality ($q \ll \ell$), that is,

$$\mathbf{x}_k = \mathbf{W} \mathbf{z}_k + \boldsymbol{\epsilon}_k, \quad (4.1)$$

where \mathbf{x}_k is expressed as a linear combination of the basis vectors $[\mathbf{w}_1, \dots, \mathbf{w}_q] = \mathbf{W} \in \mathbb{R}^{\ell \times q}$, plus noise $\boldsymbol{\epsilon}_k \in \mathbb{R}^\ell$.

The density of the latent variable \mathbf{z}_k is spherical Gaussian $\mathcal{N}(\mathbf{z}_k; \mathbf{0}, \mathbf{I}_q)$, and $\boldsymbol{\epsilon}_k$ is a normally distributed random variable with zero mean and covariance $\boldsymbol{\psi} \in \mathbb{R}^{\ell \times \ell}$, where $\boldsymbol{\psi}$ is diagonal. Conditioned on the latent variable, the data point \mathbf{x}_k then has conditional density $\mathcal{N}(\mathbf{x}_k; \mathbf{W} \mathbf{z}_k, \boldsymbol{\psi})$. From these assumptions, the probability of observing \mathbf{x}_k can be obtained:

$$\mathcal{N}(\mathbf{x}_k; \mathbf{0}, \mathbf{W} \mathbf{W}^\top + \boldsymbol{\psi}). \quad (4.2)$$

The MKMC model variants PCA-MKMC and FA-MKMC learn the model matrix \mathbf{M} described in the previous chapter in a constrained setting. FA-MKMC adopts the covariance matrix in (4.2), expressed as an outer product of two skinny matrices plus a diagonal matrix

$$\mathbf{M} = \mathbf{W} \mathbf{W}^\top + \boldsymbol{\psi}, \quad (4.3)$$

where control of the model complexity enters through the choice of q , i. e., the number of columns of \mathbf{W} . Meanwhile, $\boldsymbol{\psi}$ is the covariance matrix of the observed noises, and its diagonal entries are called *uniquenesses*. The goal of factor analysis is to learn the parameters \mathbf{W} and $\boldsymbol{\psi}$ such that (4.3) is as close to the sample data covariance as possible.

When employing the expression (4.3) for the model parameter, restrictions must be made to $\boldsymbol{\psi}$ for the model to gain information from \mathbf{x}_k . Given the goal of FA mentioned earlier, observe that when $\boldsymbol{\psi}$ is full covariance, maximum likelihood estimation of \mathbf{M} leads to simply choosing $\mathbf{W} = \mathbf{0}$ and setting $\boldsymbol{\psi}$ to be the sample data covariance, and all the data structures are then explained by the noise, as can be observed from (4.1). The maximum likelihood solution obtained in this kind of setting corresponds to an *unconstrained* Gaussian distribution [8, 43], which is the case of the (full covariance) MKMC model introduced in the previous chapter.

In factor analysis, the matrix $\boldsymbol{\psi}$ is restricted to be diagonal, i. e., all off-diagonal entries are zero. In this setting, fitting the model matrix \mathbf{M} requires only estimation of $\ell q + \ell - q(q - 1) / 2$ free parameters³, where the first term corresponds to estimating the $\ell \times q$ matrix \mathbf{W} ; the second term is for estimating the diagonal matrix $\boldsymbol{\psi}$; and the third term corresponds to redundancy to axis rotations and flips in the latent space.

³The number of free parameters are also alternatively called *degrees of freedom*.

A similar model to FA, but not identical to, is the probabilistic PCA model. Here, the noise covariance matrix is further restricted to $\sigma^2 \mathbf{I}$, making the expression for the model matrix be given as

$$\mathbf{M} = \mathbf{W}\mathbf{W}^\top + \sigma^2 \mathbf{I}. \quad (4.4)$$

In contrast with FA where the noise variance is independent for each dimension, probabilistic PCA assumes the same noise value for all dimension. Here, the number of degrees of freedom is $\ell q + 1 - q(q-1)/2$. PCA-MKMC adopts the expression in (4.4) for its model matrix, and learns the parameters \mathbf{W} and σ^2 through probabilistic PCA.

As mentioned earlier, (probabilistic) PCA and FA models are for reducing the dimension of the data and, consequently, leads to some information loss due to discarded correlations in the input data. The amount of information to retain (and to lose) therefore lies on the choice of q , as each factor (or column in \mathbf{W}) contains a certain amount of the overall variance, measured by its corresponding eigenvalue. For instance, a factor that explains more variance than a single observed variable has an eigenvalue greater than one. Intuitively, those factors that least contribute to the explanation of the variance are generally discarded, as shown by Tipping and Bishop [49]. There still has no standard approach in choosing the number of factors to retain, but a widely-used criterion is the Kaiser criterion, where all factors with eigenvalues greater than one are retained [36], and the Guttman-Kaiser criterion, where all factors with eigenvalues greater than the mean of the eigenvalues are retained [57]. Other factor retention methods have been proposed, such as via the scree-plot, total variance explained, and most recently, the empirical Kaiser criterion [11].

As well-established dimensionality reduction techniques, PCA and FA have been well-researched and applied. Two definitions of PCA are attributed to Hotelling and Pearson: Hotelling defined PCA as a maximum variance formulation that maximizes the variance of the projected data onto the principal subspace [24]; while Pearson defined it as a linear projection that minimizes the mean projection error [39]. Jolliffe, on the other hand, demonstrated how PCA can be used for applications such as data compression and data visualization, among others [25]. Variants of PCA such as PPCA and mixtures of PPCA were introduced by Tipping and Bishop [48, 49]. Links between FA [5, 4, 20] and PCA were also investigated [23, 54, 3].

The rest of this chapter introduces the proposed variants of MKMC model: the PCA-MKMC and FA-MKMC, and demonstrates how learning in these models are unified by the EM algorithm.

4.1 Overview of Probabilistic PCA and Factor Analysis

The following section gives an overview of the fundamentals of PCA, probabilistic PCA, and FA for a smooth understanding on how the proposed parametric models PCA-MKMC and FA-MKMC are formulated.

4.1.1 Principal Component Analysis

Principal component analysis finds a subspace onto which the projected data points have (1) maximum variance [24]; and (2) minimum mean projection error [39].

PCA as Maximum Variance

Consider K data points $\{\mathbf{x}_1, \dots, \mathbf{x}_K\}$, $\mathbf{x}_k \in \mathbb{R}^\ell$ and for simplicity, suppose they are centered in advance. Also, consider orthonormal vectors⁴

$$[\mathbf{u}_1, \dots, \mathbf{u}_q] =: \mathbf{U}_q \in \mathbb{R}^{\ell \times q}. \quad (4.5)$$

Then the q -dimensional subspace where the projection points have maximum variance is determined by \mathbf{U}_q , where $\mathbf{u}_1, \dots, \mathbf{u}_q$ are the eigenvectors corresponding to the dominant eigenvalues $\lambda_1 \geq \dots \geq \lambda_q$ ($\geq \lambda_{q+1} \geq \dots \geq \lambda_K$) in the eigendecomposition of \mathbf{S} :

$$\mathbf{S}\mathbf{U} = \mathbf{\Lambda}\mathbf{U}, \quad (4.6)$$

where

$$\mathbf{S} = \frac{1}{K} \sum_{k=1}^K \mathbf{x}_k \mathbf{x}_k^\top \quad (4.7)$$

is the data covariance matrix, $\mathbf{U} = [\mathbf{u}_1, \dots, \mathbf{u}_K]$, and $\mathbf{\Lambda} = \text{diag}(\lambda_1, \dots, \lambda_K)$. The subspace obtained from \mathbf{U}_q is therefore called the *principal subspace*, with *principal components* $\mathbf{u}_1, \dots, \mathbf{u}_q$.

To see why this is the case, consider projection onto \mathbb{R} in the direction of say, \mathbf{u}_1 . Then the projection variance is given by

$$\frac{1}{K} \sum_{k=1}^K \left(\mathbf{u}_1^\top \mathbf{x}_k \right)^2 = \mathbf{u}_1^\top \mathbf{S} \mathbf{u}_1, \quad (4.8)$$

and is maximized subject to the constraint $\mathbf{u}_1^\top \mathbf{u}_1 = 1$ (from the orthonormal assumption earlier). The equivalent Lagrange formulation is

$$\max_{\mathbf{u}_1} \left\{ \mathbf{u}_1^\top \mathbf{S} \mathbf{u}_1 + \lambda \left(1 - \mathbf{u}_1^\top \mathbf{u}_1 \right) \right\}, \quad (4.9)$$

⁴Orthonormal vectors are unit vectors that are orthogonal to each other.

where λ is the introduced Lagrange multiplier. The solution for this maximization problem is

$$\mathbf{S}\mathbf{u}_1 = \lambda\mathbf{u}_1, \quad (4.10)$$

implying that \mathbf{u}_1 is an eigenvector of \mathbf{S} with eigenvalue λ . Rewriting the above equation as

$$\mathbf{u}_1^\top \mathbf{S}\mathbf{u}_1 = \lambda, \quad (4.11)$$

it can be observed that the LHS of this equation is maximum when \mathbf{u}_1 is chosen such that it corresponds to the largest eigenvalue.

For the general case of q -dimensional subspace, the principal components are obtained by choosing the q eigenvectors of \mathbf{S} corresponding to the q largest eigenvalues.

PCA as Minimum Mean of Projection Errors

Another way to look at the PCA formulation is as follows: given the q -dimensional principal subspace determined by q eigenvectors, the remaining $\ell - q$ eigenvectors are therefore orthogonal to the said subspace. The lengths of these remaining eigenvectors, as scaled by their corresponding eigenvalues, represent the projection errors; and the mean of the projection errors

$$\frac{1}{\ell - q} \sum_{i=q+1}^{\ell} \lambda_i \quad (4.12)$$

is minimum when the eigenvectors corresponding to $\ell - q$ smallest eigenvalues are chosen, which is equivalent to choosing the eigenvectors with largest eigenvalues as principal components.

4.1.2 Probabilistic PCA

As a dimensionality reduction tool, PCA can be given a probabilistic reformulation so that it can be used as a constrained density model, whilst capturing dominant data correlations.

In probabilistic PCA, the generative model $\mathbf{x}_k = \mathbf{W}\mathbf{z}_k + \boldsymbol{\epsilon}_k$ in (4.1) is assumed to have the following Gaussian densities:

$$p(\mathbf{z}_k) = \mathcal{N}(\mathbf{z}_k; \mathbf{0}, \mathbf{I}_q); \quad (4.13)$$

$$p(\boldsymbol{\epsilon}_k) = \mathcal{N}(\boldsymbol{\epsilon}_k; \mathbf{0}, \sigma^2 \mathbf{I}_\ell); \quad \text{and} \quad (4.14)$$

$$p(\mathbf{x}_k | \mathbf{z}_k, \mathbf{W}, \sigma^2) = \mathcal{N}(\mathbf{x}_k; \mathbf{W}\mathbf{z}_k, \sigma^2 \mathbf{I}_\ell). \quad (4.15)$$

From here, the Gaussian density of the complete data can be computed:

$$\begin{aligned} p((\mathbf{x}_k, \mathbf{z}_k) | \mathbf{W}, \sigma^2) &= p(\mathbf{x}_k | \mathbf{z}_k, \mathbf{W}, \sigma^2) p(\mathbf{z}_k) \\ &= \frac{1}{(2\pi\sigma^2)^{\ell/2}} \exp\left(-\frac{\|\mathbf{x}_k - \mathbf{W}\mathbf{z}_k\|^2}{2\sigma^2}\right) \\ &\quad \cdot \frac{1}{(2\pi)^{q/2}} \exp\left(-\frac{\|\mathbf{z}_k\|^2}{2}\right) \end{aligned} \quad (4.16)$$

$$= \mathcal{N}\left((\mathbf{x}_k, \mathbf{z}_k); \mathbf{0}, \begin{bmatrix} \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I} & \mathbf{W} \\ \mathbf{W}^\top & \mathbf{I} \end{bmatrix}\right). \quad (4.17)$$

From this joint density, the marginal density (or the probability of observing \mathbf{x}_k) can be obtained:

$$p(\mathbf{x}_k | \mathbf{W}, \sigma^2) = \mathcal{N}\left(\mathbf{x}_k; \mathbf{0}, \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}\right), \quad (4.18)$$

as well as the posterior distribution

$$\begin{aligned} p(\mathbf{z}_k | \mathbf{x}_k, \mathbf{W}, \sigma^2) &= \mathcal{N}\left(\mathbf{z}_k; \mathbf{W}^\top (\mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I})^{-1} \mathbf{x}_k, \right. \\ &\quad \left. \mathbf{I} - \sigma^2 \mathbf{W}^\top (\mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I})^{-1} \mathbf{W}\right) \\ &= \mathcal{N}\left(\mathbf{z}_k; \mathbf{C}^{-1} \mathbf{W}^\top \mathbf{x}_k, \sigma^2 \mathbf{C}^{-1}\right). \end{aligned} \quad (4.19)$$

Here, matrix inversion formula is applied so that the $\ell \times \ell$ matrix $(\mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I})^{-1}$ can be efficiently evaluated by computing instead the inverse of a smaller $q \times q$ matrix \mathbf{C} , as follows:

$$\left(\mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}\right)^{-1} = \frac{1}{\sigma^2}\mathbf{I} - \frac{1}{\sigma^2}\mathbf{W}\mathbf{C}^{-1}\mathbf{W}^\top, \quad (4.20)$$

where

$$\mathbf{C} := \mathbf{W}^\top \mathbf{W} + \sigma^2 \mathbf{I}_q. \quad (4.21)$$

It can be observed that the obtained conditional density in (4.19) has mean that is a linear function of the data variable \mathbf{x}_k , and covariance that is independent of \mathbf{x}_k . Probabilistic PCA is hence an example of a linear-Gaussian model [43, 37], which usually involves maximization of expressions such as the complete-data log-likelihood function.

Closed-Form Solutions

In probabilistic PCA, the goal is to learn the parameters \mathbf{W} and σ^2 so as to make the covariance $\mathbf{M} = \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}$ of the predictive distribution (4.18) be as close as possible to the data covariance \mathbf{S} in (4.7). From the marginal density $p(\mathbf{x}_k | \Theta)$, the

corresponding log-likelihood function is then

$$\begin{aligned}
\log p(\mathbf{X}|\Theta) &= \sum_{k=1}^K \log p(\mathbf{x}_k|\Theta) \\
&= -\frac{K\ell}{2} \log(2\pi) - \frac{K}{2} \log \det \mathbf{M} - \frac{K}{2} \langle \mathbf{M}^{-1}, \mathbf{S} \rangle \\
&=: \mathcal{L}.
\end{aligned} \tag{4.22}$$

Fortunately, from standard matrix differentiation results of [28], the following partial derivative can be obtained:

$$\frac{\partial \mathcal{L}}{\partial \mathbf{W}} = K (\mathbf{M}^{-1} \mathbf{S} \mathbf{M}^{-1} \mathbf{W} - \mathbf{M}^{-1} \mathbf{W}) = \mathbf{0} \tag{4.23}$$

yielding the stationary points

$$\mathbf{S} \mathbf{M}^{-1} \mathbf{W} = \mathbf{W}, \tag{4.24}$$

whose trivial solutions are $\mathbf{W} = \mathbf{0}$ and $\mathbf{M} = \mathbf{S}$. Of course, the desired case is when $\mathbf{W} \neq \mathbf{0}$ and $\mathbf{M} \neq \mathbf{S}$, the solution of which was shown by Tipping and Bishop [49]. Making use of the singular value decomposition of \mathbf{W} leads to the potential solutions

$$\mathbf{W} = \mathbf{U}_q (\mathbf{L}_q - \sigma^2 \mathbf{I})^{1/2} \mathbf{R}, \tag{4.25}$$

where the $\ell \times q$ matrix \mathbf{U}_q consists of concatenation of any eigenvectors of the data covariance \mathbf{S} . The $q \times q$ diagonal matrix \mathbf{L}_q have diagonal entries λ_i , the eigenvalues corresponding to the eigenvectors in \mathbf{U}_q . Meanwhile, the matrix \mathbf{R} is an arbitrary orthogonal matrix, and hence can be treated as the identity matrix \mathbf{I}_q .

However, the maximum likelihood solution is obtained when the columns of \mathbf{U}_q are the eigenvectors of \mathbf{S} corresponding to the dominant eigenvalues $\lambda_1 \geq \dots \geq \lambda_q$ ($\geq \lambda_{q+1} \geq \dots \geq \lambda_\ell$), and σ^2 is the average of the $\ell - q$ smallest eigenvalues. Thus, for eigenvectors $\mathbf{u}_1, \dots, \mathbf{u}_\ell$ corresponding to the eigenvalues $\lambda_1 \geq \dots \geq \lambda_\ell$, the maximum likelihood solutions \mathbf{W}_{ML} and σ_{ML}^2 are therefore

$$\mathbf{W}_{\text{ML}} = \mathbf{U}_q (\mathbf{\Lambda}_q - \sigma^2 \mathbf{I})^{1/2}, \text{ for } \mathbf{\Lambda}_q = [\mathbf{u}_1, \dots, \mathbf{u}_q] \tag{4.26}$$

$$\sigma_{\text{ML}}^2 = \frac{1}{\ell - q} \sum_{i=q+1}^{\ell} \lambda_i, \tag{4.27}$$

which are analogous to obtaining the principal components in Sect. 4.1.1. As can be observed, the model covariance

$$\mathbf{M} = \mathbf{W}_{\text{ML}} \mathbf{W}_{\text{ML}}^\top + \sigma_{\text{ML}}^2 \mathbf{I} \tag{4.28}$$

captures the dominant correlations in the data through \mathbf{W}_{ML} , and treats the correlations lost from dimensionality reduction as noise through $\sigma_{\text{ML}}^2 \mathbf{I}$.

EM algorithm for PCA

As the maximum likelihood solutions have already been found, pushing through with the EM algorithm at this point may already seem worthless. However, using an iterative procedure such as the EM algorithm is deemed computationally advantageous over eigendecomposition of the sample covariance matrix in high-dimensional spaces. Also, the discussion on EM algorithm for probabilistic PCA can be extended to factor analysis, as can be seen in the next section. The EM algorithm for probabilistic PCA then proceeds as follows:

E-step. In this step, the sufficient statistics of the (latent space) posterior distribution in (4.19) is computed, using the current parameter values $\Theta^{(t-1)} = (\mathbf{W}^{(t-1)}, \sigma_{t-1}^2)$:

$$\mathbb{E}[\mathbf{z}_k] = \left(\mathbf{C}^{(t)}\right)^{-1} \left(\mathbf{W}^{(t-1)}\right)^\top \mathbf{x}_k \quad (4.29)$$

$$\mathbb{E}[\mathbf{z}_k \mathbf{z}_k^\top] = \sigma_{t-1}^2 \left(\mathbf{C}^{(t)}\right)^{-1} + \mathbb{E}[\mathbf{z}_k] \mathbb{E}[\mathbf{z}_k]^\top, \quad (4.30)$$

where $\mathbf{C}^{(t)} = (\mathbf{W}^{(t-1)})^\top \mathbf{W}^{(t-1)} + \sigma_{t-1}^2 \mathbf{I}$.

M-step. For this step, the model parameter updates $\Theta^{(t)} = (\mathbf{W}^{(t)}, \sigma_t^2)$ are obtained by maximizing the expected log-likelihood of the complete data, as follows:

$$\mathbf{W}^{(t)} = \operatorname{argmax}_{\mathbf{W}} \sum_{k=1}^K \mathbb{E}[\log p(\mathbf{x}_k, \mathbf{z}_k)]; \quad (4.31)$$

$$\sigma_t^2 = \operatorname{argmax}_{\sigma^2} \sum_{k=1}^K \mathbb{E}[\log p(\mathbf{x}_k, \mathbf{z}_k)], \quad (4.32)$$

under the sufficient statistics in the E-step. Thus, for the M-step, consider the complete-data log-likelihood

$$\begin{aligned} \log p(\mathbf{X}, \mathbf{Z} | \Theta) &= \prod_{k=1}^K \mathcal{N}(\mathbf{x}_k; \mathbf{W} \mathbf{z}_k, \sigma^2 \mathbf{I}) \mathcal{N}(\mathbf{z}_k; \mathbf{0}, \mathbf{I}_q) \\ &= \sum_{k=1}^K (\log p(\mathbf{x}_k | \mathbf{z}_k, \Theta) + \log p(\mathbf{z}_k)), \end{aligned} \quad (4.33)$$

and take the expectation under the posterior distribution in (4.19) to obtain

$$\begin{aligned}
\mathbb{E}[\log p(\mathbf{X}, \mathbf{Z} | \Theta)] &= -\frac{K\ell}{2} \log(2\pi\sigma^2) - \frac{1}{2\sigma^2} \sum_{k=1}^K \|\mathbf{x}_k\|^2 \\
&\quad - \frac{1}{2} \text{Tr} \left(\sum_{k=1}^K \mathbb{E}[\mathbf{z}_k \mathbf{z}_k^\top] \right) + \frac{1}{\sigma^2} \sum_{k=1}^K \mathbb{E}[\mathbf{z}_k]^\top \mathbf{W}^\top \mathbf{x}_k \\
&\quad - \frac{1}{2\sigma^2} \left\langle \mathbf{W}, \mathbf{W} \sum_{k=1}^K \mathbb{E}[\mathbf{z}_k \mathbf{z}_k^\top] \right\rangle \\
&\quad - \frac{q}{2} \log(2\pi). \tag{4.34}
\end{aligned}$$

Keeping the posterior statistics (4.29) and (4.30) fixed, (4.34) is maximized with respect to \mathbf{W} and σ^2 . The model parameter updates are then obtained as follows:

$$\mathbf{W}^{(t)} = \left(\sum_{k=1}^K \mathbf{x}_k \mathbb{E}[\mathbf{z}_k]^\top \right) \left(\sum_{k=1}^K \mathbb{E}[\mathbf{z}_k \mathbf{z}_k^\top] \right)^{-1} \tag{4.35}$$

$$\begin{aligned}
\sigma_t^2 &= \frac{1}{K\ell} \sum_{k=1}^K \left[\|\mathbf{x}_k\|^2 - 2\mathbb{E}[\mathbf{z}_k]^\top \left(\mathbf{W}^{(t)} \right)^\top \mathbf{x}_k \right. \\
&\quad \left. + \text{Tr} \left(\mathbb{E}[\mathbf{z}_k \mathbf{z}_k^\top] \left(\mathbf{W}^{(t)} \right)^\top \mathbf{W}^{(t)} \right) \right]. \tag{4.36}
\end{aligned}$$

The EM algorithm is then carried out by first initializing the parameters, and then repeatedly doing (4.29), (4.30), (4.35), and (4.36) in order until convergence.

4.1.3 Factor Analysis

Closely related to probabilistic PCA, factor analysis also models data points as a linear combination of the factors, with specified error terms. The only difference of FA from probabilistic PCA is that the noise covariance, which is also the covariance of the conditional distribution of \mathbf{x}_k given the latent \mathbf{z}_k , is diagonal:

$$p(\boldsymbol{\epsilon}_k) = \mathcal{N}(\boldsymbol{\epsilon}_k; \mathbf{0}, \boldsymbol{\psi}); \quad \text{and} \tag{4.37}$$

$$p(\mathbf{x}_k | \mathbf{z}_k, \mathbf{W}, \boldsymbol{\psi}) = \mathcal{N}(\mathbf{x}_k; \mathbf{W} \mathbf{z}_k, \boldsymbol{\psi}), \tag{4.38}$$

instead of an isotropic $\sigma^2 \mathbf{I}$. The probability of observing \mathbf{x}_k is therefore given by the marginal distribution

$$p(\mathbf{x}_k | \mathbf{W}, \boldsymbol{\psi}) = \mathcal{N}(\mathbf{x}_k; \mathbf{0}, \mathbf{W} \mathbf{W}^\top + \boldsymbol{\psi}), \tag{4.39}$$

and the posterior distribution is then

$$\begin{aligned}
p(\mathbf{z}_k | \mathbf{x}_k, \mathbf{W}, \boldsymbol{\psi}) &= \mathcal{N}\left(\mathbf{z}_k; \mathbf{W}^\top \left(\mathbf{W}\mathbf{W}^\top + \boldsymbol{\psi}\right)^{-1} \mathbf{x}_k, \right. \\
&\quad \left. \mathbf{I}_q - \mathbf{W}^\top \left(\mathbf{W}\mathbf{W}^\top + \boldsymbol{\psi}\right)^{-1} \mathbf{W}\right) \\
&= \mathcal{N}\left(\mathbf{z}_k; \mathbf{G}\mathbf{W}^\top \boldsymbol{\psi}^{-1} \mathbf{x}_k, \mathbf{G}\right), \tag{4.40}
\end{aligned}$$

where

$$\begin{aligned}
\mathbf{G} &:= \mathbf{I}_q - \mathbf{W}^\top \left(\mathbf{W}\mathbf{W}^\top + \boldsymbol{\psi}\right)^{-1} \mathbf{W} \\
&= \left(\mathbf{I}_q + \mathbf{W}^\top \boldsymbol{\psi}^{-1} \mathbf{W}\right)^{-1}, \tag{4.41}
\end{aligned}$$

by utilizing again the matrix inversion formula to efficiently evaluate the inverse.

Unlike in the probabilistic PCA model, the maximum likelihood solution for factor analysis has no closed form, that is, the model parameters are coupled in a way such that the maximum likelihood estimates cannot be solved directly. The optimal solution must therefore be found iteratively, which can be done via EM algorithm.

The EM algorithm for FA is analogous to that of probabilistic PCA. The sufficient statistics of the (latent space) posterior distribution are given by

$$\begin{aligned}
\mathbb{E}[\mathbf{z}_k] &= \mathbf{W}^\top \left(\mathbf{W}\mathbf{W}^\top + \boldsymbol{\psi}\right)^{-1} \mathbf{x}_k \\
&= \mathbf{G}\mathbf{W}^\top \boldsymbol{\psi}^{-1} \mathbf{x}_k \tag{4.42}
\end{aligned}$$

$$\begin{aligned}
\mathbb{E}[\mathbf{z}_k \mathbf{z}_k^\top] &= \mathbf{I}_q - \mathbf{W}^\top \left(\mathbf{W}\mathbf{W}^\top + \boldsymbol{\psi}\right)^{-1} \mathbf{W} + \mathbb{E}[\mathbf{z}_k] \mathbb{E}[\mathbf{z}_k]^\top \\
&= \mathbf{G} + \mathbb{E}[\mathbf{z}_k] \mathbb{E}[\mathbf{z}_k]^\top. \tag{4.43}
\end{aligned}$$

For the M-step, the complete-data log-likelihood for FA is as follows:

$$\begin{aligned}
\log p(\mathbf{X}, \mathbf{Z} | \boldsymbol{\Theta}) &= \prod_{k=1}^K \mathcal{N}(\mathbf{x}_k; \mathbf{W} \mathbf{z}_k, \boldsymbol{\psi}) \mathcal{N}(\mathbf{z}_k; \mathbf{0}, \mathbf{I}_q) \\
&= \sum_{k=1}^K (\log p(\mathbf{x}_k | \mathbf{z}_k, \boldsymbol{\Theta}) + \log p(\mathbf{z}_k)). \tag{4.44}
\end{aligned}$$

Holding the sufficient statistics fixed, the model parameter updates are obtained by setting the gradient of the above complete-data log-likelihood function to zero. The model parameter updates are then

$$\mathbf{W}^{(t)} = \left(\sum_{k=1}^K \mathbf{x}_k \mathbb{E}[\mathbf{z}_k]^\top \right) \left(\sum_{k=1}^K \mathbb{E}[\mathbf{z}_k \mathbf{z}_k^\top] \right)^{-1} \tag{4.45}$$

$$\boldsymbol{\psi}^{(t)} = \text{diag} \left(\mathbf{S} - \mathbf{W}^{(t)} \frac{1}{K} \sum_{k=1}^K \mathbb{E}[\mathbf{z}_k] \mathbf{x}_k^\top \right), \tag{4.46}$$

where the expectations are evaluated using the current parameter values $\Theta^{(t-1)} = (\mathbf{W}^{(t-1)}, \boldsymbol{\psi}^{(t-1)})$, and *diag* sets the off-diagonal entries of $\boldsymbol{\psi}^{(t)}$ to zero.

The EM algorithm goes forth by first initializing⁵ the parameters \mathbf{W} and $\boldsymbol{\psi}$, then performing the following repeatedly in order: solving for the $q \times q$ matrix \mathbf{G} ; evaluating the expectations (4.42) and (4.43) for the E-step; and revising the parameters in (4.45) and (4.46) for the M-step.

4.2 Parametric Models for MKMC

Given how the maximum likelihood solutions for probabilistic PCA and FA are obtained by EM algorithm, the parametric models PCA-MKMC and FA-MKMC can now be introduced. These parametric models are the constrained counterparts of the (full covariance) MKMC model introduced in the previous chapter. The aim of PCA-MKMC and FA-MKMC is to provide control of model complexity, in order to improve the generalization performance of the proposed models.

4.2.1 PCA-MKMC

For its model parameter matrix, PCA-MKMC adopts the covariance matrix of the predictive probability described in probabilistic PCA model. The model matrix for PCA-MKMC is hence expressed as

$$\mathbf{M} = \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}, \quad (4.47)$$

where $\mathbf{W} \in \mathbb{R}^{\ell \times q}$ and $\sigma^2 \in \mathbb{R}$ are the model parameters. The objective function for this model is given by

$$J_{\text{PCA}}(\mathcal{H}, \mathbf{W}, \sigma^2) := \sum_{k=1}^K \text{LogDet}(\mathbf{Q}^{(k)}, \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}), \quad (4.48)$$

where \mathcal{H} is the set of submatrices of missing entries, as described in (3.7). The optimal model parameter estimates are obtained by repeating the following steps:

1. *Imputation step.* Using the current parameter estimates $\mathbf{W}^{(t-1)}$ and σ_{t-1}^2 , minimize J_{PCA} with respect to \mathcal{H} :

$$\mathcal{H}^{(t)} := \underset{\mathcal{H}}{\text{argmin}} J_{\text{PCA}}(\mathcal{H}, \mathbf{W}^{(t-1)}, \sigma_{t-1}^2); \quad (4.49)$$

⁵Since the maximum likelihood (ML) solutions in probabilistic PCA can be obtained directly from the eigendecomposition of the data covariance matrix, these ML solutions can be used to initialize the parameters in the EM algorithm for FA.

2. *Model update step.* Fixing the update $\mathcal{H}^{(t)}$, minimize J_{PCA} with respect to \mathbf{W} and σ^2 :

$$\left(\mathbf{W}^{(t)}, \sigma_t^2\right) := \underset{(\mathbf{W}, \sigma^2)}{\operatorname{argmin}} J_{\text{PCA}} \left(\mathcal{H}^{(t)}, \mathbf{W}, \sigma^2\right). \quad (4.50)$$

Here, the imputation step is analogous to the imputation step in the (full covariance) MKMC in Chap. 3, but with the model matrix defined as (4.47). The kernel matrix $\mathbf{Q}^{(k)}$ is hence updated using (3.11) and (3.12), and let the updated $\mathbf{Q}^{(k)}$ be denoted as $\mathbf{Q}^{(t,k)}$.

Fixing the updated empirical kernel matrices, the objective function in (4.48) is now expressed as

$$\begin{aligned} J_{\text{PCA}} \left(\mathcal{H}^{(t)}, \mathbf{W}, \sigma^2\right) &:= \frac{K}{2} \log \det \left(\mathbf{W}\mathbf{W}^\top + \sigma^2 \mathbf{I}\right) \\ &+ \frac{K}{2} \left\langle \left(\mathbf{W}\mathbf{W}^\top + \sigma^2 \mathbf{I}\right)^{-1}, \mathbf{S}^{(t)} \right\rangle \\ &+ \text{const.}, \end{aligned} \quad (4.51)$$

where 'const.' denotes the terms independent of the model parameters, and

$$\mathbf{S}^{(t)} := \frac{1}{K} \sum_{k=1}^K \mathbf{Q}^{(t,k)}, \quad (4.52)$$

where $\mathbf{Q}^{(t,k)}$ is the value of $\mathbf{Q}^{(k)}$ in the t -th iteration.

The closed-form solutions in the model update step are obtained in the following manner. Letting $\mathbf{u}_1, \dots, \mathbf{u}_\ell$ be the eigenvectors corresponding to the eigenvalues $\lambda_1 \geq \dots \geq \lambda_\ell$ of $\mathbf{S}^{(t)}$, the model parameter updates $\mathbf{W}^{(t)}$ and σ_t^2 are given as

$$\sigma_t^2 = \frac{1}{\ell - q} \sum_{i=q+1}^{\ell} \lambda_i \quad (4.53)$$

$$\mathbf{W}^{(t)} = \mathbf{U}_q \left(\mathbf{\Lambda}_q - \sigma_t^2 \mathbf{I}\right)^{1/2}, \quad (4.54)$$

where $\mathbf{U}_q := [\mathbf{u}_1, \dots, \mathbf{u}_q]$, and $\mathbf{\Lambda}_q := \operatorname{diag}(\lambda_1, \dots, \lambda_q)$. The outline of PCA-MKMC algorithm is given in Alg. 2.

4.2.2 FA-MKMC

In FA-MKMC, the model matrix takes the form

$$\mathbf{M} = \mathbf{W}\mathbf{W}^\top + \boldsymbol{\psi}, \quad (4.55)$$

where the model parameters are the diagonal matrix $\boldsymbol{\psi}$ and $\mathbf{W} \in \mathbb{R}^{\ell \times q}$. As this model has more freedom in the variance of each variable, FA-MKMC is a bit more

Algorithm 2 PCA-MKMC Algorithm.

Input: Incomplete kernel matrices $\mathbf{Q}^{(1)}, \dots, \mathbf{Q}^{(K)}$.
Output: Completed kernel matrices $\mathbf{Q}^{(1)}, \dots, \mathbf{Q}^{(K)}$.

- 1: **begin**
- 2: Initialize $\{\mathbf{Q}^{(k)}\}_{k=1}^K$ by imputing zeros in the missing entries;
- 3: Initialize the model matrix as $\mathbf{M} := \frac{1}{K} \sum_{k=1}^K \mathbf{Q}^{(k)}$;
- 4: **repeat**
- 5: **for all** $k \in \{1, \dots, K\}$ **do**
- 6: Update $\mathbf{Q}_{v,h}^{(k)}$ and $\mathbf{Q}_{h,h}^{(k)}$ using (3.11) and (3.12);
- 7: **end for**
- 8: Set \mathbf{S} as (4.52);
- 9: Update \mathbf{W} and σ^2 using (4.53) and (4.54);
- 10: Set \mathbf{M} as (4.47);
- 11: **until** convergence
- 12: **end.**

flexible than PCA-MKMC, but is more rigid than the (full covariance) MKMC. The objective function for FA-MKMC is given by

$$J_{\text{FA}}(\mathcal{H}, \mathbf{W}, \boldsymbol{\psi}) := \sum_{k=1}^K \text{LogDet} \left(\mathbf{Q}^{(k)}, \mathbf{W}\mathbf{W}^\top + \boldsymbol{\psi} \right). \quad (4.56)$$

Fitting the FA parameters to the empirical kernel matrices is also analogous to those of (full covariance) MKMC and PCA-MKMC, where two steps (imputation and model update steps) are repeated. However, as previously mentioned, the FA parameters have no closed-form solutions since its maximum likelihood estimates cannot be solved directly. Hence, inference in FA-MKMC is done by repeatedly performing the following steps:

1. *Imputation step.* Using the current parameter estimates $\mathbf{W}^{(t-1)}$ and $\boldsymbol{\psi}^{(t-1)}$, minimize J_{FA} with respect to \mathcal{H} :

$$\mathcal{H}^{(t)} := \underset{\mathcal{H}}{\text{argmin}} J_{\text{FA}} \left(\mathcal{H}, \mathbf{W}^{(t-1)}, \boldsymbol{\psi}^{(t-1)} \right); \quad (4.57)$$

2. *Model update step.* Fixing the update $\mathcal{H}^{(t)}$, find the model parameter updates $\mathbf{W}^{(t)}$ and $\boldsymbol{\psi}^{(t)}$ such that

$$J_{\text{FA}} \left(\mathcal{H}^{(t)}, \mathbf{W}^{(t-1)}, \boldsymbol{\psi}^{(t-1)} \right) \geq J_{\text{FA}} \left(\mathcal{H}^{(t)}, \mathbf{W}^{(t)}, \boldsymbol{\psi}^{(t)} \right). \quad (4.58)$$

Similar to (full covariance) MKMC and PCA-MKMC, inference of the missing entries in the imputation step can be done using (3.11) and (3.12).

Algorithm 3 FA-MKMC Algorithm.**Input:** Incomplete kernel matrices $\mathbf{Q}^{(1)}, \dots, \mathbf{Q}^{(K)}$.**Output:** Completed kernel matrices $\mathbf{Q}^{(1)}, \dots, \mathbf{Q}^{(K)}$.

```

1: begin
2: Initialize  $\{\mathbf{Q}^{(k)}\}_{k=1}^K$  by imputing zeros in the missing entries;
3: Initialize the model matrix as  $\mathbf{M} := \frac{1}{K} \sum_{k=1}^K \mathbf{Q}^{(k)}$ ;
4: Initialize  $\mathbf{W}$  and  $\boldsymbol{\psi}$  using (4.53) and (4.54);
5: repeat
6:   for all  $k \in \{1, \dots, K\}$  do
7:     Update  $\mathbf{Q}_{v,h}^{(k)}$  and  $\mathbf{Q}_{h,h}^{(k)}$  using (3.11) and (3.12);
8:   end for
9:   Set  $\mathbf{S}$  as (4.52);
10:  Update  $\mathbf{W}$  and  $\boldsymbol{\psi}$  using (4.59) and (4.60);
11:  Set  $\mathbf{M}$  as (4.55);
12: until convergence
13: end.

```

On the other hand, the model parameter updates that satisfy (4.58) are computed as

$$\mathbf{W}^{(t)} := \mathbf{S}_{xz}^{(t)} \left(\mathbf{S}_{zz}^{(t)} \right)^{-1}; \quad (4.59)$$

$$\boldsymbol{\psi}^{(t)} := \text{diag} \left(\mathbf{S}^{(t)} - \mathbf{W}^{(t)} \left(\mathbf{S}_{xz}^{(t)} \right)^\top \right), \quad (4.60)$$

where

$$\begin{aligned}
\mathbf{S}_{zz}^{(t)} &:= \mathbf{G}^{(t)} + \mathbf{B}^{(t)} \mathbf{S}_{xz}^{(t)} \\
\mathbf{S}_{xz}^{(t)} &:= \mathbf{S}^{(t)} \left(\mathbf{B}^{(t)} \right)^\top \\
\mathbf{B}^{(t)} &:= \mathbf{G}^{(t)} \mathbf{F}^{(t)} \\
\mathbf{G}^{(t)} &:= \left(\mathbf{I}_q + \mathbf{F}^{(t)} \mathbf{W}^{(t-1)} \right)^{-1} \\
\mathbf{F}^{(t)} &:= \left(\mathbf{W}^{(t-1)} \right)^\top \left(\boldsymbol{\psi}^{(t-1)} \right)^{-1}.
\end{aligned} \quad (4.61)$$

The monotonic decrease in (4.58) during optimization is guaranteed by the following proposition⁶:

Proposition 4.2.1. *The inequality (4.58) always holds in every iteration in Alg. 3.*

⁶Proof is shown in Sect. 4.6.3.

4.3 PCA-MKMC and FA-MKMC in a Statistical Framework

Following the same course of action as in the previous chapter, the following section shows how EM algorithm is derived for the probabilistic models PCA-MKMC and FA-MKMC, and then proceeds to showing how to arrive at the optimal solutions.

4.3.1 EM Algorithm for PCA-MKMC

Recall from Sect. 3.4.2 that each vector $\mathbf{x}_k \in \mathbb{R}^\ell$ consists of the subvectors $\mathbf{v}_k \in \mathbb{R}^{n_k}$ and $\mathbf{h}_k \in \mathbb{R}^{m_k}$, corresponding to the observed and unobserved data in $\mathbf{Q}^{(k)}$, respectively. Thus, analogous to (4.22), the expected log-likelihood of the joint density

$$\begin{aligned} p_{\text{PCA}}(\mathbf{v}_k, \mathbf{h}_k | \mathbf{W}, \sigma^2) &= p_{\text{PCA}}(\mathbf{x}_k | \mathbf{W}, \sigma^2) \\ &= \mathcal{N}(\mathbf{x}_k; \mathbf{0}, \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}), \end{aligned} \quad (4.62)$$

is the \mathcal{Q} -function

$$\begin{aligned} \mathcal{Q}_t^{\text{PCA}}(\mathbf{W}, \sigma^2) &:= \sum_{k=1}^K \mathbb{E} [\log p_{\text{PCA}}(\mathbf{x}_k | \mathbf{W}, \sigma^2)] \\ &= -\frac{K\ell}{2} \log(2\pi) - \frac{K}{2} \log \det(\mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}) \\ &\quad - \frac{1}{2} \left\langle (\mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I})^{-1}, \sum_{k=1}^K \mathbb{E} [\mathbf{x}_k \mathbf{x}_k^\top] \right\rangle. \end{aligned} \quad (4.63)$$

Here, the expectation is evaluated under the posterior $q(\mathbf{h}_k | \mathbf{v}_k)q(\mathbf{v}_k)$ using current parameter values $\mathbf{W}^{(t-1)}$ and σ_{t-1}^2 , yielding $\mathbb{E}[\mathbf{x}_k \mathbf{x}_k^\top] = \mathbf{Q}^{(k)}$. The above \mathcal{Q} -function results to an equivalent expression as (4.22), with $\mathbf{M} = \mathbf{W}\mathbf{W}^\top + \sigma^2\mathbf{I}$ and $\mathbf{S} = \frac{1}{K} \sum_{k=1}^K \mathbf{Q}^{(k)}$, of which, the maximum likelihood solutions are (4.26) and (4.27). Moreover, as can be observed, the resulted \mathcal{Q} -function is the negative of the objective function J_{PCA} in (4.51), up to some constant. Hence, the maximum likelihood solutions of PCA-MKMC are indeed those given in (4.53) and (4.54).

4.3.2 EM Algorithm for FA-MKMC

In FA-MKMC, a latent variable $\mathbf{z}_k \in \mathbb{R}^q$, drawn from a standard Gaussian $\mathcal{N}(\mathbf{z}_k; \mathbf{0}, \mathbf{I}_q)$, is introduced for each data point \mathbf{x}_k . The complete data for the k -th data source is

$(\mathbf{x}_k, \mathbf{z}_k)$; and the joint density takes the form

$$\begin{aligned} p_{\text{FA}}(\mathbf{x}_k, \mathbf{z}_k | \mathbf{W}, \boldsymbol{\psi}) &= \mathcal{N}(\mathbf{x}_k; \mathbf{W} \mathbf{z}_k, \boldsymbol{\psi}) \mathcal{N}(\mathbf{z}_k; \mathbf{0}, \mathbf{I}_q) \\ &= \mathcal{N}\left((\mathbf{x}_k, \mathbf{z}_k); \mathbf{0}, \begin{bmatrix} \mathbf{W} \mathbf{W}^\top + \boldsymbol{\psi} & \mathbf{W} \\ \mathbf{W}^\top & \mathbf{I} \end{bmatrix}\right). \end{aligned} \quad (4.64)$$

Also from here, the marginal density of \mathbf{x}_k can be obtained:

$$p_{\text{FA}}(\mathbf{x}_k | \mathbf{W}, \boldsymbol{\psi}) = \mathcal{N}(\mathbf{x}_k; \mathbf{0}, \mathbf{W} \mathbf{W}^\top + \boldsymbol{\psi}). \quad (4.65)$$

E-step. From the above result, the posterior distribution of the latent variable is obtained (4.40), as well as the sufficient statistics (4.42) and (4.43).

Meanwhile, the expected complete-data log-likelihood is given by the following \mathcal{Q} -function:

$$\begin{aligned} \mathcal{Q}_t^{\text{FA}}(\mathbf{W}, \boldsymbol{\psi}) &:= \sum_{k=1}^K \mathbb{E}[\log p_{\text{FA}}(\mathbf{x}_k, \mathbf{z}_k | \mathbf{W}, \boldsymbol{\psi})] \\ &= \sum_{k=1}^K \mathbb{E}\left[-\frac{1}{2} \langle \mathbf{x}_k - \mathbf{W} \mathbf{z}_k, \boldsymbol{\psi}^{-1} (\mathbf{x}_k - \mathbf{W} \mathbf{z}_k) \rangle - \frac{1}{2} \log \det \boldsymbol{\psi} + \text{const.}\right] \\ &= \left\langle \mathbf{W}, \boldsymbol{\psi}^{-1} \sum_{k=1}^K \mathbb{E}[\mathbf{x}_k \mathbf{z}_k^\top] \right\rangle - \frac{1}{2} \left\langle \mathbf{W}, \boldsymbol{\psi}^{-1} \mathbf{W} \sum_{k=1}^K \mathbb{E}[\mathbf{z}_k \mathbf{z}_k^\top] \right\rangle \\ &\quad - \frac{1}{2} \left\langle \boldsymbol{\psi}^{-1}, \sum_{k=1}^K \mathbb{E}[\mathbf{x}_k \mathbf{x}_k^\top] \right\rangle - \frac{K}{2} \log \det \boldsymbol{\psi} + \text{const.}, \end{aligned} \quad (4.66)$$

where the expectation is taken under the joint posterior density

$$\begin{aligned} q_t(\mathbf{x}_k, \mathbf{z}_k) &= q_t(\mathbf{z}_k, \mathbf{h}_k | \mathbf{v}_k) q(\mathbf{v}_k) \\ &= p_{\text{FA}}(\mathbf{z}_k | \mathbf{h}_k, \mathbf{v}_k, \boldsymbol{\Theta}^{(t-1)}) p_{\text{FA}}(\mathbf{h}_k | \mathbf{v}_k, \boldsymbol{\Theta}^{(t-1)}) q(\mathbf{v}_k) \\ &= p_{\text{FA}}(\mathbf{z}_k | \mathbf{x}_k, \boldsymbol{\Theta}^{(t-1)}) p_{\text{FA}}(\mathbf{h}_k | \mathbf{v}_k, \boldsymbol{\Theta}^{(t-1)}) q(\mathbf{v}_k) \end{aligned} \quad (4.67)$$

from the current model parameter values $\boldsymbol{\Theta}^{(t-1)} = (\mathbf{W}^{(t-1)}, \boldsymbol{\psi}^{(t-1)})$. The second moments appearing in (4.66) are obtained⁷ as

$$\begin{aligned} \sum_{k=1}^K \mathbb{E}[\mathbf{x}_k \mathbf{x}_k^\top] &= K \mathbf{S}, & \sum_{k=1}^K \mathbb{E}[\mathbf{x}_k \mathbf{z}_k^\top] &= K \mathbf{S}_{xz}, \\ \text{and } \sum_{k=1}^K \mathbb{E}[\mathbf{z}_k \mathbf{z}_k^\top] &= K \mathbf{S}_{zz}. \end{aligned} \quad (4.68)$$

⁷The derivations are given in Set. 4.6.1.

M-step. When the above results for the second moments are substituted directly to (4.66), the following expression for Q_t^{FA} is obtained:

$$\begin{aligned} Q_t^{\text{FA}}(\mathbf{W}, \boldsymbol{\psi}) = & \langle \mathbf{W}, K\boldsymbol{\psi}^{-1}\mathbf{S}_{xz} \rangle - \frac{K}{2} \langle \mathbf{W}, \boldsymbol{\psi}^{-1}\mathbf{W}\mathbf{S}_{zz} \rangle \\ & - \frac{K}{2} \langle \boldsymbol{\psi}^{-1}, \mathbf{S} \rangle - \frac{K}{2} \log\det \boldsymbol{\psi} + \text{const.} \end{aligned} \quad (4.69)$$

Setting its gradient to zero yields the parameter updates (4.59) and (4.60).⁸

4.4 Experiments and Results

To test how changing the flexibility of a kernel matrix affects a learning classifier's generalization capability, experiments on real-world data are conducted. The proposed parametric models are applied to functional classification prediction of yeast proteins, following the works of [18, 31, 38].

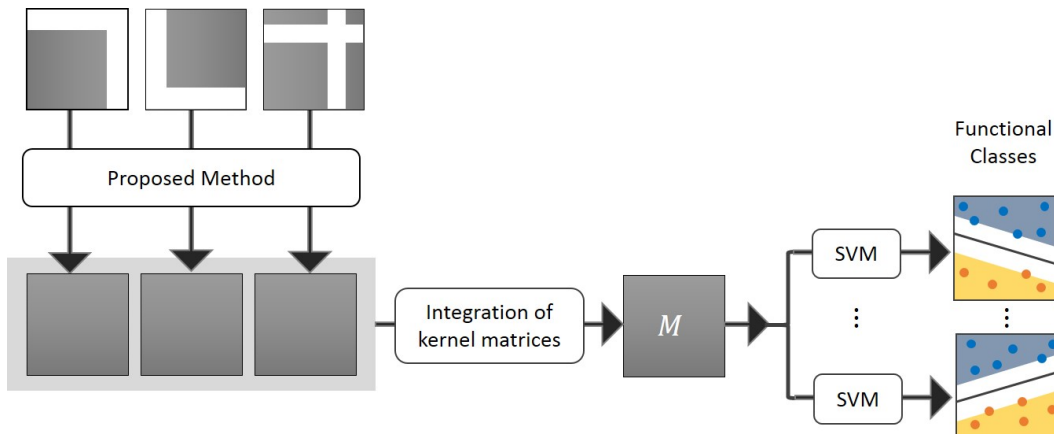


FIGURE 4.1: Experimental set-up.

The proposed method completes the incomplete kernel matrices, then integrates them for an SVM classifier to train on a subset of data in M (training data). The trained classifier then predicts which of the remaining data points (testing data) belong to a certain functional class or not.

4.4.1 Experimental Setting

The setting for this experiment (Fig. 4.1) is similar to that in the previous chapter, where the incomplete kernel matrices are first completed by the proposed methods, then integrated to form a single kernel matrix consisting of all the information derived from different but relevant data sources. An SVM classifier is then trained on a subset of this integrated matrix, and the learned classifier then categorizes the test data. The

⁸The derivation for the model parameter update appears in Sect. 4.6.2.

only difference in the setting is that instead of classifying the proteins as membranes or non-membranes, the proteins in this experiment are classified as to whether they perform a certain function or not. For instance, if a protein is involved in DNA processing and in transcription, but not in metabolism, then this protein is labeled +1 in the former classes (DNA processing and transcription), and -1 in the latter (metabolism). In this experiment, 13 functional classes⁹ are considered, leading to 13 binary classification problems.

Data Set

The data set consists of 3,588 yeast genes with known functions (from the 13 functional classifications). The corresponding kernel matrices are hence of size $3,588 \times 3,588$, and the kernel matrices used in this study are derived from the following different data sources: gene expression data, genetic interaction data, protein-protein interaction data, Pfam domain structure data, Smith-Waterman¹⁰ pairwise sequence comparison data, and protein interaction via TAP data. A detailed description of these six kernel matrices can be found in [31].

Completion Methods

The completion methods involved in this experiment are the traditional methods (zero-imputation and mean-imputation, as described in the previous chapter), the (full covariance) MKMC method, and the parametric methods PCA-MKMC and FA-MKMC. In choosing the flexibility of the parametric methods, two criteria are considered: the Kaiser criterion (greater-than-one) and Guttman-Kaiser criterion (greater-than-mean). Hence, in this experiment, PCA-GK and PCA-K correspond to PCA-MKMC using Guttman-Kaiser and Kaiser criterion, respectively. Similarly for FA-MKMC, FA-GK and FA-K are considered. In total, seven completion methods are tested for classification performance.

In this experiment, incomplete kernel matrices are generated in the same fashion as in the previous experiment, where rows and columns to be missed are chosen at random. The completion methods then infer the missing entries to be able to complete an incomplete kernel matrix. The fused completed kernel matrices (the model kernel matrix) for each completion method is then used to train an SVM classifier for function prediction task.

⁹The complete list of the functional classes is given in [31].

¹⁰Smith-Waterman algorithm determines similar regions between two protein sequence strings through sequence alignment.

TABLE 4.1: Classification performance for 20% missed entries.

The SVM classifier is trained on the integrated completed kernel matrices of each method, for 20% training data. The mean ROC scores are then tabulated, where each row corresponds to the ROC score of the completion methods on a given functional class. The largest ROC score per class is boldfaced, while those with no significant difference from the highest ones are underlined.

Class	zero-SVM	mean-SVM	(FC) MKMC	PCA-GK	PCA-K	FA-GK	FA-K
1	0.7914	0.7915	0.7995	0.8015	0.8022	0.8010	0.8006
2	0.7918	0.7925	0.7975	<u>0.8025</u>	0.8032	0.8014	<u>0.8021</u>
3	0.7941	0.7933	0.8000	<u>0.8045</u>	0.8052	0.8029	0.8032
4	0.8418	0.8431	0.8497	0.8529	0.8534	0.8516	0.8519
5	0.8839	0.8844	0.8956	0.8972	0.8979	0.8961	0.8967
6	0.7665	0.7669	0.7745	<u>0.7780</u>	0.7783	<u>0.7770</u>	0.7770
7	0.8321	0.8328	0.8414	0.8437	0.8444	0.8429	<u>0.8440</u>
8	0.7336	0.7336	0.7354	<u>0.7407</u>	0.7418	0.7391	0.7386
9	0.7621	0.7630	0.7651	<u>0.7706</u>	0.7714	<u>0.7694</u>	<u>0.7695</u>
10	0.7441	0.7445	0.7485	0.7551	0.7570	0.7525	<u>0.7556</u>
11	0.5766	0.5757	0.5825	<u>0.5791</u>	<u>0.5807</u>	0.5793	0.5772
12	0.9357	0.9347	0.9435	<u>0.9448</u>	0.9453	0.9443	0.9444
13	<u>0.6818</u>	<u>0.6845</u>	0.6794	0.6913	<u>0.6911</u>	0.6840	0.6838

4.4.2 Classification Performance Result

The experiment is performed ten times for each completion method, and the mean ROC scores are recorded in Tab. 4.1.

Indeed, classification performance is improved when the model kernel matrix is not too flexible. The parametric models PCA-MKMC and FA-MKMC introduced in this chapter have performed better than the (full covariance) MKMC model, in particular, the highest ROC scores are obtained by PCA-MKMC using Kaiser criterion, with the exception from the two functional classes where ROC scores are the least (Classes 11 and 13). To determine the statistical difference of PCA-K among the other completion methods, a one-sample t -test is again employed. Indeed, the parametric models have no significant difference in general, but are statistically significant over the other completion methods.

4.5 Summary

In this chapter, two variants of the (full covariance) MKMC model are developed in order for the user to gain control of the model flexibility. These parametric models, the PCA-MKMC and FA-MKMC, justified the claim that by reducing a model's flexibility, overfitting is avoided, suggesting better generalization of the learned classifier. Indeed, the parametric models performed the best among the other completion

methods in the task of functional classification prediction in yeast proteins. Similar to (full covariance) MKMC model, theoretical backgrounds are also provided for PCA-MKMC and FA-MKMC models.

4.6 Proofs and Derivations

4.6.1 Derivation of the second moments for FA.

From the product rule of probabilities, the joint distribution $q_t(\mathbf{x}_k, \mathbf{z}_k)$ can be expressed as a product of a marginal and a conditional distribution:

$$q_t(\mathbf{x}_k, \mathbf{z}_k) = q_t(\mathbf{z}_k | \mathbf{x}_k) q_t(\mathbf{x}_k) \quad (4.70)$$

$$= q_t(\mathbf{x}_k | \mathbf{z}_k) q_t(\mathbf{z}_k). \quad (4.71)$$

On the other hand, the marginal $q_t(\mathbf{x}_k)$ can be obtained by marginalizing out \mathbf{z}_k from the joint distribution:

$$\begin{aligned} q_t(\mathbf{x}_k) &= \int q_t(\mathbf{x}_k, \mathbf{z}_k) d\mathbf{z}_k \\ &= \int q_t(\mathbf{x}_k | \mathbf{z}_k) q_t(\mathbf{z}_k) d\mathbf{z}_k. \end{aligned} \quad (4.72)$$

From (4.42), (4.43), (4.61), (4.70), and (4.72), the following second moments are obtained:

$$\begin{aligned} \mathbb{E}_{q_t(\mathbf{x}_k, \mathbf{z}_k)} [\mathbf{x}_k \mathbf{x}_k^\top] &= \mathbb{E}_{q_t(\mathbf{z}_k | \mathbf{x}_k)} [\mathbb{E}_{q_t(\mathbf{x}_k)} [\mathbf{x}_k \mathbf{x}_k^\top]] \\ &= \mathbf{Q}^{(k)}; \end{aligned} \quad (4.73)$$

$$\begin{aligned} \mathbb{E}_{q_t(\mathbf{x}_k, \mathbf{z}_k)} [\mathbf{x}_k \mathbf{z}_k^\top] &= \mathbb{E}_{q_t(\mathbf{x}_k)} [\mathbf{x}_k \mathbb{E}_{q_t(\mathbf{z}_k | \mathbf{x}_k)} [\mathbf{z}_k^\top]] \\ &= \mathbb{E}_{q_t(\mathbf{x}_k)} [\mathbf{x}_k \mathbf{x}_k^\top] (\mathbf{B}^{(t)})^\top \\ &= \mathbf{Q}^{(k)} (\mathbf{B}^{(t)})^\top; \end{aligned} \quad (4.74)$$

and

$$\begin{aligned} \mathbb{E}_{q_t(\mathbf{x}_k, \mathbf{z}_k)} [\mathbf{z}_k \mathbf{z}_k^\top] &= \mathbb{E}_{q_t(\mathbf{x}_k)} [\mathbb{E}_{q_t(\mathbf{z}_k | \mathbf{x}_k)} [\mathbf{z}_k \mathbf{z}_k^\top]] \\ &= \mathbf{G}^{(t)} + \mathbb{E}_{q_t(\mathbf{x}_k)} \left[\mathbf{B}^{(t)} \mathbf{x}_k \mathbf{x}_k^\top (\mathbf{B}^{(t)})^\top \right] \\ &= \mathbf{G}^{(t)} + \mathbf{B}^{(t)} \mathbf{Q}^{(k)} (\mathbf{B}^{(t)})^\top. \end{aligned} \quad (4.75)$$

It then follows that

$$\begin{aligned} \sum_{k=1}^K \mathbb{E} [\mathbf{x}_k \mathbf{x}_k^\top] &= K \mathbf{S}, & \sum_{k=1}^K \mathbb{E} [\mathbf{x}_k \mathbf{z}_k^\top] &= K \mathbf{S}_{xz}, \\ \text{and } \sum_{k=1}^K \mathbb{E} [\mathbf{z}_k \mathbf{z}_k^\top] &= K \mathbf{S}_{zz}. \end{aligned} \quad (4.76)$$

4.6.2 Derivation of the model parameter updates for FA-MKMC.

The model parameter update $\mathbf{W}^{(t)}$ is obtained as follows:

$$\begin{aligned} \frac{\partial \mathcal{Q}_t^{\text{FA}}(\mathbf{W}, \boldsymbol{\psi})}{\partial \mathbf{W}} &= \boldsymbol{\psi}^{-1} \left(K \mathbf{S}_{xz}^{(t)} - K \mathbf{W} \mathbf{S}_{zz}^{(t)} \right) = 0. \\ \implies \mathbf{W}^{(t)} &= \mathbf{S}_{xz}^{(t)} \left(\mathbf{S}_{zz}^{(t)} \right)^{-1}. \end{aligned}$$

For the model update $\boldsymbol{\psi}^{(t)}$,

$$\begin{aligned} \frac{\partial \mathcal{Q}_t^{\text{FA}}(\mathbf{W}, \boldsymbol{\psi})}{\partial \boldsymbol{\psi}^{-1}} &= \frac{K}{2} \boldsymbol{\psi} - \frac{K}{2} \mathbf{S}^{(t)} + \frac{K}{2} \mathbf{W}^{(t)} \mathbf{B}^{(t)} \mathbf{S}^{(t)} \\ \implies \boldsymbol{\psi} &= \mathbf{S}^{(t)} - \mathbf{W}^{(t)} \mathbf{B}^{(t)} \mathbf{S}^{(t)} \\ &= \mathbf{S}^{(t)} - \mathbf{W}^{(t)} \left(\mathbf{S}_{xz}^{(t)} \right)^\top \\ \implies \boldsymbol{\psi}^{(t)} &= \text{diag} \left(\mathbf{S}^{(t)} - \mathbf{W}^{(t)} \left(\mathbf{S}_{xz}^{(t)} \right)^\top \right), \end{aligned}$$

where *diag* sets the off-diagonal entries of $\boldsymbol{\psi}^{(t)}$ to zero.

4.6.3 Proof of Proposition 4.2.1

Let $\boldsymbol{\Theta} = (\mathbf{W}, \boldsymbol{\psi})$. With the use of empirical distribution $q_t(\mathbf{x}_k)$ and marginal density (4.65), the objective function (4.56) in the model update step can be expressed as follows:

$$\begin{aligned} J_{\text{FA}}(\mathcal{H}^{(t)}, \boldsymbol{\Theta}) &= \sum_{k=1}^K \mathbb{E}_{q_t(\mathbf{x}_k)} [\log q_t(\mathbf{x}_k)] - \sum_{k=1}^K \mathbb{E}_{q_t(\mathbf{x}_k)} [\log p_{\text{FA}}(\mathbf{x}_k | \boldsymbol{\Theta})] \\ &= \sum_{k=1}^K \mathbb{E}_{q_t(\mathbf{x}_k)} [\log q_t(\mathbf{x}_k)] - \sum_{k=1}^K \mathbb{E}_{q_t(\mathbf{x}_k, \mathbf{z}_k)} \left[\log \frac{p_{\text{FA}}(\mathbf{x}_k, \mathbf{z}_k | \boldsymbol{\Theta})}{p_{\text{FA}}(\mathbf{z}_k | \mathbf{x}_k, \boldsymbol{\Theta})} \right], \end{aligned}$$

where the second equality follows from

$$p_{\text{FA}}(\mathbf{x}_k, \mathbf{z}_k | \Theta) = p_{\text{FA}}(\mathbf{z}_k | \mathbf{x}_k, \Theta) p_{\text{FA}}(\mathbf{x}_k | \Theta), \quad (4.77)$$

for any $\mathbf{z}_k \in \mathbb{R}^q$, for $k = 1, \dots, K$.

Likewise, the Q -function for FA-MKMC can be rewritten as follows:

$$\begin{aligned} \mathcal{Q}_t^{\text{FA}}(\Theta) &= \sum_{k=1}^K \mathbb{E}_{q_t(\mathbf{x}_k, \mathbf{z}_k)} [\log p_{\text{FA}}(\mathbf{x}_k, \mathbf{z}_k | \Theta)] \\ &= \sum_{k=1}^K \mathbb{E}_{q_t(\mathbf{x}_k, \mathbf{z}_k)} \left[\log \frac{p_{\text{FA}}(\mathbf{x}_k, \mathbf{z}_k | \Theta)}{p_{\text{FA}}(\mathbf{z}_k | \mathbf{x}_k, \Theta)} \right] \\ &\quad + \sum_{k=1}^K \mathbb{E}_{q_t(\mathbf{x}_k)} [\mathbb{E}_{q_t(\mathbf{z}_k | \mathbf{x}_k)} [\log p_{\text{FA}}(\mathbf{z}_k | \mathbf{x}_k, \Theta)]] \\ &= -J_{\text{FA}}(\mathcal{H}^{(t)}, \Theta) + \sum_{k=1}^K \mathbb{E}_{q_t(\mathbf{x}_k)} [\log q_t(\mathbf{x}_k)] \\ &\quad + \sum_{k=1}^K \mathbb{E}_{q_t(\mathbf{x}_k)} [\mathbb{E}_{q_t(\mathbf{z}_k | \mathbf{x}_k)} [\log p_{\text{FA}}(\mathbf{z}_k | \mathbf{x}_k, \Theta)]] . \end{aligned}$$

From the monotonic increase of the Q -function for each model parameter update $\Theta^{(t)} = (\mathbf{W}^{(t)}, \boldsymbol{\psi}^{(t)})$, and from the non-negativity of the KL-divergence between two probabilistic density functions, the following inequality holds:

$$\begin{aligned} 0 &\leq \mathcal{Q}_t^{\text{FA}}(\Theta^{(t)}) - \mathcal{Q}_t^{\text{FA}}(\Theta^{(t-1)}) \\ &= -J_{\text{FA}}(\mathcal{H}^{(t)}, \Theta^{(t)}) + J_{\text{FA}}(\mathcal{H}^{(t)}, \Theta^{(t-1)}) \\ &\quad - \sum_{k=1}^K \text{KL} \left(p_{\text{FA}}(\mathbf{z}_k | \mathbf{x}_k, \Theta^{(t-1)}), p_{\text{FA}}(\mathbf{z}_k | \mathbf{x}_k, \Theta^{(t)}) \right) \\ &\leq -J_{\text{FA}}(\mathcal{H}^{(t)}, \Theta^{(t)}) + J_{\text{FA}}(\mathcal{H}^{(t)}, \Theta^{(t-1)}) . \end{aligned}$$

It goes to show that at each update of the model parameters, the objective function decreases monotonically. \square

Chapter 5

Conclusion

This thesis has successfully addressed the problem of learning from incomplete data in a kernel-based setting, by introducing a novel method that mutually infers the missing entries of kernel matrices representing different data types. This method, the (full covariance) MKMC model, not only infers the missing entries whilst preserving the relationships in the data, but also provides a way of unifying the data sources, backed by a statistical analysis of the algorithm. As this model has full covariance, two variants of this model are also introduced: the PCA-MKMC and FA-MKMC, derived from the notions of probabilistic PCA and factor analysis. These parametric models have further improved detection of underlying structures in the data by providing a means of adjusting the number of free parameters or degrees of freedom. By adjusting the number of degrees of freedom, model flexibility is controlled, providing a real data structure for a learning classifier to exploit. As the demand for finding patterns and hidden structures in the data increases, with the availability of multiple information sources, statistical learning algorithms such as the ones presented in this thesis are highly sought-after.

Succinctly, a mutual kernel matrix completion method, called MKMC model, is introduced in Chapter 3 to solve the multiview learning problem with missing information. The proposed model, applied to the membrane protein classification problem, has demonstrated its efficacy over the conventional methods through classification performance and completion accuracy.

In Chapter 4, parametric models are introduced, combining the notions of probabilistic PCA and factor analysis to the (full covariance) MKMC model in Chapter 3. These parametric models allow control of model flexibility, and have shown their positive influence in the generalization capability of a learned classifier through experiments on protein function classification on yeast proteins.

Appendix A

Some Formulas and Identities

This appendix is added as a quick reference for the formulas and identities that were extensively used in this thesis.

A.1 Probability Densities

Let $\mathbf{x} = (x_1, \dots, x_\ell)$ consist of continuous variables x_1, \dots, x_ℓ . The multivariate probability density then satisfies

$$p(\mathbf{x}) \geq 0; \tag{A.1}$$

$$\int p(\mathbf{x}) d\mathbf{x} = 1. \tag{A.2}$$

From sum and product rules of probability, the following hold for multivariate probability densities:

$$p(\mathbf{x}) = \int p(\mathbf{x}, \mathbf{y}) d\mathbf{y}; \tag{A.3}$$

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{y}|\mathbf{x})p(\mathbf{x}) \tag{A.4}$$

$$= p(\mathbf{x}|\mathbf{y})p(\mathbf{y}), \tag{A.5}$$

for some continuous vector \mathbf{y} .

A.2 Multivariate Gaussian Distribution

The Gaussian (or normal) distribution is a continuous probability distribution for real-valued random variables.

For a single variable x with a Gaussian distribution, its probability density function takes the form

$$p(x|\mu, \sigma^2) = \mathcal{N}(x; \mu, \sigma^2) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{-\frac{1}{2\sigma^2}(x - \mu)^2\right\}, \quad (\text{A.6})$$

where μ and σ^2 are the mean and variance, respectively.

In the case of an ℓ -dimensional vector \mathbf{x} , the multivariate Gaussian distribution is given by

$$\begin{aligned} p(\mathbf{x}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) &= \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma}) \\ &= \frac{1}{(2\pi)^{\ell/2}} \frac{1}{|\boldsymbol{\Sigma}|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^\top \boldsymbol{\Sigma}^{-1}(\mathbf{x} - \boldsymbol{\mu})\right\}, \end{aligned} \quad (\text{A.7})$$

where $\boldsymbol{\mu} \in \mathbb{R}^\ell$ is the mean vector, and $\boldsymbol{\Sigma} \in \mathbb{R}^{\ell \times \ell}$ is the covariance matrix, with ℓ and $\ell(\ell + 1)/2$ independent parameters, respectively.

Some remarks about multivariate Gaussian:

1. The argument of the exponential in (A.7) is in *quadratic form*, i. e., of the form $\mathbf{a}^\top \mathbf{A} \mathbf{a}$, for some vector $\mathbf{a} \in \mathbb{R}^n$ and symmetric matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$;
2. The mean vector is the expectation of \mathbf{x} :

$$\boldsymbol{\mu} = \mathbb{E}[\mathbf{x}], \quad (\text{A.8})$$

also called the *first moment* of \mathbf{x} ;

3. The *covariance* of \mathbf{x} is given by

$$\text{cov}[\mathbf{x}] = \mathbb{E}\left[(\mathbf{x} - \mathbb{E}[\mathbf{x}])(\mathbf{x} - \mathbb{E}[\mathbf{x}])^\top\right] = \boldsymbol{\Sigma}; \quad (\text{A.9})$$

4. The *second moment* of \mathbf{x} is

$$\begin{aligned} \mathbb{E}[\mathbf{x}\mathbf{x}^\top] &= \boldsymbol{\mu}\boldsymbol{\mu}^\top + \text{cov}[\mathbf{x}] \\ &= \boldsymbol{\mu}\boldsymbol{\mu}^\top + \boldsymbol{\Sigma}; \end{aligned} \quad (\text{A.10})$$

5. The first and second moments of \mathbf{x} entirely characterizes multivariate Gaussian distributions.
6. For $\mathbf{X} = (\mathbf{x}_1, \dots, \mathbf{x}_K)$ consisting of independent variables $\mathbf{x}_k \in \mathbb{R}^\ell$ drawn from a Gaussian distribution (in this case, the variables are called *independent and*

identically distributed, or i.i.d.), the (marginal) probability of \mathbf{X} is given by

$$p(\mathbf{X} | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \prod_{k=1}^K \mathcal{N}(\mathbf{x}_k | \boldsymbol{\mu}, \boldsymbol{\Sigma}). \quad (\text{A.11})$$

7. Given a joint Gaussian density $p(\mathbf{x}_a, \mathbf{x}_b | \boldsymbol{\mu}, \boldsymbol{\Sigma})$, then the conditional densities $p(\mathbf{x}_a | \mathbf{x}_b, \boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $p(\mathbf{x}_b | \mathbf{x}_a, \boldsymbol{\mu}, \boldsymbol{\Sigma})$, and the marginal densities $p(\mathbf{x}_a | \boldsymbol{\mu}, \boldsymbol{\Sigma})$ and $p(\mathbf{x}_b | \boldsymbol{\mu}, \boldsymbol{\Sigma})$, are also Gaussians.

A.2.1 Partitioned Gaussians

Let $\mathbf{x} \in \mathbb{R}^\ell$ consist of two disjoint subsets $\mathbf{x}_a \in \mathbb{R}^n$ and $\mathbf{x}_b \in \mathbb{R}^{\ell-n}$, for $n < \ell$, and suppose that $\mathbf{x} \sim \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \boldsymbol{\Sigma})$, where

$$\boldsymbol{\mu} = \begin{pmatrix} \boldsymbol{\mu}_a \\ \boldsymbol{\mu}_b \end{pmatrix}, \quad \boldsymbol{\Sigma} = \begin{pmatrix} \boldsymbol{\Sigma}_{aa} & \boldsymbol{\Sigma}_{ab} \\ \boldsymbol{\Sigma}_{ba} & \boldsymbol{\Sigma}_{bb} \end{pmatrix}, \quad (\text{A.12})$$

are the corresponding mean vector and covariance matrix, respectively. Here, since $\boldsymbol{\Sigma}$ is a symmetric matrix, it follows that the submatrices $\boldsymbol{\Sigma}_{aa}$ and $\boldsymbol{\Sigma}_{bb}$ are symmetric, and $\boldsymbol{\Sigma}_{ba}^\top = \boldsymbol{\Sigma}_{ab}$. The following Gaussian distributions are then obtained:

Marginal Distribution

$$p(\mathbf{x}_a | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}_a; \boldsymbol{\mu}_a, \boldsymbol{\Sigma}_{aa}); \quad (\text{A.13})$$

$$p(\mathbf{x}_b | \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}_b; \boldsymbol{\mu}_b, \boldsymbol{\Sigma}_{bb}). \quad (\text{A.14})$$

Conditional Distribution

$$p(\mathbf{x}_a | \mathbf{x}_b, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}_a; \boldsymbol{\mu}_{a|b}, \boldsymbol{\Sigma}_{a|b}) \quad (\text{A.15})$$

$$p(\mathbf{x}_b | \mathbf{x}_a, \boldsymbol{\mu}, \boldsymbol{\Sigma}) = \mathcal{N}(\mathbf{x}_b; \boldsymbol{\mu}_{b|a}, \boldsymbol{\Sigma}_{b|a}), \quad (\text{A.16})$$

where

$$\boldsymbol{\mu}_{a|b} = \boldsymbol{\mu}_a + \boldsymbol{\Sigma}_{ab} \boldsymbol{\Sigma}_{bb}^{-1} (\mathbf{x}_b - \boldsymbol{\mu}_b); \quad (\text{A.17})$$

$$\boldsymbol{\mu}_{b|a} = \boldsymbol{\mu}_b + \boldsymbol{\Sigma}_{ba} \boldsymbol{\Sigma}_{aa}^{-1} (\mathbf{x}_a - \boldsymbol{\mu}_a); \quad (\text{A.18})$$

$$\boldsymbol{\Sigma}_{a|b} = \boldsymbol{\Sigma}_{aa} - \boldsymbol{\Sigma}_{ab} \boldsymbol{\Sigma}_{bb}^{-1} \boldsymbol{\Sigma}_{ba}; \quad (\text{A.19})$$

$$\boldsymbol{\Sigma}_{b|a} = \boldsymbol{\Sigma}_{bb} - \boldsymbol{\Sigma}_{ba} \boldsymbol{\Sigma}_{aa}^{-1} \boldsymbol{\Sigma}_{ab}. \quad (\text{A.20})$$

The covariances $\Sigma_{a|b}$ and $\Sigma_{b|a}$ are called the *Schur complement* of the submatrices Σ_{bb} and Σ_{aa} , respectively [46].

A.3 Matrix Properties

Let \mathbf{A} denote a matrix, whose element in the i th row and j th column is denoted by $A_{i,j}$. The following are some definitions and identities involving matrices:

1. The *transpose* of \mathbf{A} , denoted by \mathbf{A}^\top , has elements of the form $(\mathbf{A}^\top)_{i,j} = A_{j,i}$.
2. A square matrix \mathbf{A} is *symmetric* if

$$\mathbf{A} = \mathbf{A}^\top, \quad (\text{A.21})$$

i. e., the elements of \mathbf{A} are of the form $A_{i,j} = A_{j,i}$.

3. For two matrices $\mathbf{A} \in \mathbb{R}^{m \times p}$ and $\mathbf{B} \in \mathbb{R}^{p \times n}$,

$$(\mathbf{AB})^\top = \mathbf{B}^\top \mathbf{A}^\top. \quad (\text{A.22})$$

4. A matrix \mathbf{A} is said to be *invertible* if its inverse \mathbf{A}^{-1} exists such that

$$\mathbf{AA}^{-1} = \mathbf{A}^{-1}\mathbf{A} = \mathbf{I}. \quad (\text{A.23})$$

5. For invertible matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{n \times n}$,

$$(\mathbf{AB})^{-1} = \mathbf{B}^{-1}\mathbf{A}^{-1}; \quad (\text{A.24})$$

$$\left(\mathbf{A}^\top\right)^{-1} = \left(\mathbf{A}^{-1}\right)^\top. \quad (\text{A.25})$$

6. If $\mathbf{A} = \text{diag}(A_1, \dots, A_n)$ is a diagonal matrix, then its inverse is given by

$$\mathbf{A}^{-1} = \text{diag}(1/A_1, \dots, 1/A_n). \quad (\text{A.26})$$

7. For matrices $\mathbf{A}, \mathbf{B}, \mathbf{C}$, and \mathbf{D} of correct sizes, the *Woodbury formula* (or matrix inversion formula) is given by

$$\left(\mathbf{A} + \mathbf{BD}^{-1}\mathbf{C}\right)^{-1} = \mathbf{A}^{-1} - \mathbf{A}^{-1}\mathbf{B}\left(\mathbf{D} + \mathbf{CA}^{-1}\mathbf{B}\right)^{-1}\mathbf{CA}^{-1}. \quad (\text{A.27})$$

8. For any $a \in \mathbb{R}$ and square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, the trace of \mathbf{A} is given by

$$\text{Tr}(a\mathbf{A}) = a \text{Tr}(\mathbf{A}) = a \sum_{i=1}^n \mathbf{A}_{i,i}. \quad (\text{A.28})$$

9. For matrices \mathbf{A} , \mathbf{B} , and \mathbf{C} of corresponding sizes, the following hold:

$$\text{Tr}(\mathbf{AB}) = \text{Tr}(\mathbf{BA}); \quad (\text{A.29})$$

$$\text{Tr}(\mathbf{ABC}) = \text{Tr}(\mathbf{CAB}) = \text{Tr}(\mathbf{BCA}). \quad (\text{A.30})$$

10. If a matrix $\mathbf{A} \in \mathbb{R}^{m \times n}$ satisfies $\mathbf{A}^\top \mathbf{A} = \mathbf{I}_n$, then \mathbf{A} is said to be an *orthonormal matrix*.

11. For a square matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$, the *determinant* of \mathbf{A} is defined as

$$|\mathbf{A}| = \sum (\pm 1) A_{1,i_1} A_{2,i_2} \cdots A_{n,i_n}, \quad (\text{A.31})$$

where the coefficient is +1 if the permutation $i_1 i_2 \cdots i_n$ is even, and -1 if the permutation is odd.

12. For two square matrices \mathbf{A} and \mathbf{B} ,

$$|\mathbf{AB}| = |\mathbf{A}| |\mathbf{B}|. \quad (\text{A.32})$$

13. The determinant of an invertible matrix is given by

$$|\mathbf{A}^{-1}| = \frac{1}{|\mathbf{A}|}. \quad (\text{A.33})$$

14. For matrices $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n}$, their *inner product* is defined as

$$\langle \mathbf{A}, \mathbf{B} \rangle := \sum_{i=1}^m \sum_{j=1}^n A_{i,j} B_{i,j}. \quad (\text{A.34})$$

15. For matrices $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathbb{R}^{m \times n}$ and scalars $a, b \in \mathbb{R}$, the following hold:

$$\langle \mathbf{A}, \mathbf{B} \rangle = \langle \mathbf{B}, \mathbf{A} \rangle = \text{Tr}(\mathbf{A}^\top \mathbf{B}) = \text{Tr}(\mathbf{B}^\top \mathbf{A}); \quad (\text{A.35})$$

$$\langle a\mathbf{A}, \mathbf{B} \rangle = a \langle \mathbf{A}, \mathbf{B} \rangle = \langle \mathbf{A}, a\mathbf{B} \rangle; \quad (\text{A.36})$$

$$\langle a\mathbf{A}, b\mathbf{B} \rangle = ab \langle \mathbf{A}, \mathbf{B} \rangle; \quad (\text{A.37})$$

$$\langle \mathbf{A}, \mathbf{B} + \mathbf{C} \rangle = \langle \mathbf{A}, \mathbf{B} \rangle + \langle \mathbf{A}, \mathbf{C} \rangle. \quad (\text{A.38})$$

16. The *Frobenius norm* of a matrix \mathbf{A} is defined as

$$\|\mathbf{A}\|_F := \sqrt{\langle \mathbf{A}, \mathbf{A} \rangle} \geq 0. \quad (\text{A.39})$$

17. The *eigendecomposition* (or spectral decomposition) of an $n \times n$ symmetric matrix \mathbf{A} is given by

$$\mathbf{A} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top, \quad (\text{A.40})$$

where \mathbf{U} is an $n \times n$ orthonormal matrix whose columns are called *eigenvectors*, and $\mathbf{\Lambda} = \text{diag}(\lambda_1, \dots, \lambda_n)$ is a diagonal matrix whose diagonal entries are called *eigenvalues*.

18. An $n \times n$ symmetric matrix \mathbf{A} is said to be *positive semidefinite* if for any $\boldsymbol{\alpha} \in \mathbb{R}^n$,

$$\boldsymbol{\alpha}^\top \mathbf{A} \boldsymbol{\alpha} \geq 0. \quad (\text{A.41})$$

19. If \mathbf{A} is positive semidefinite then its eigenvalues are non-negative, and its determinant is also non-negative.

20. An $n \times n$ symmetric matrix \mathbf{A} is said to be (strictly) *positive definite* if for any $\boldsymbol{\alpha} \in \mathbb{R}^n$,

$$\boldsymbol{\alpha}^\top \mathbf{A} \boldsymbol{\alpha} > 0, \quad (\text{A.42})$$

and has positive determinant and eigenvalues.

A.4 Matrix Derivatives

The following are some properties involving derivatives of matrices, say matrices \mathbf{A} and \mathbf{B} :

$$\frac{\partial}{\partial \mathbf{A}} \text{Tr}(\mathbf{A}) = \mathbf{I} \quad (\text{A.43})$$

$$\frac{\partial}{\partial \mathbf{A}} \text{Tr}(\mathbf{A}\mathbf{B}) = \mathbf{B}^\top \quad (\text{A.44})$$

$$\frac{\partial}{\partial \mathbf{A}} \text{Tr}(\mathbf{A}^\top \mathbf{B}) = \mathbf{B} \quad (\text{A.45})$$

$$\frac{\partial}{\partial \mathbf{A}} \text{Tr}(\mathbf{A}\mathbf{B}\mathbf{A}^\top) = \mathbf{A}(\mathbf{B} + \mathbf{B}^\top) \quad (\text{A.46})$$

$$\frac{\partial}{\partial \mathbf{A}} \log \det(\mathbf{A}) = (\mathbf{A}^{-1})^\top. \quad (\text{A.47})$$

Bibliography

- [1] Shun-Ichi Amari. “Information Geometry of the EM and em Algorithms for Neural Networks”. In: *Neural Networks* 8.9 (Dec. 1995), pp. 1379–1408.
- [2] Shun-Ichi Amari and Hiroshi Nagaoka. *Methods of Information Geometry*. Vol. 191. Translations of Mathematical Monographs. American Mathematical Society, 2001.
- [3] Theodore Wilbur Anderson. “Asymptotic Theory for Principal Component Analysis”. In: *Ann. Math. Statist.* 34.1 (Mar. 1963), pp. 122–148. DOI: 10.1214/aoms/1177704248. URL: <https://doi.org/10.1214/aoms/1177704248>.
- [4] David J. Bartholomew et al. *Analysis of Multivariate Social Science Data, 2nd Ed.* Chapman & Hall/CRC Statistics in the Social and Behavioral Sciences. Taylor & Francis, 2008.
- [5] Alexander Basilevsky. *Statistical Factor Analysis and Related Methods: Theory and Applications*. Wiley Series in Probability and Statistics. John Wiley & Sons, Inc., 2008.
- [6] Sahely Bhadra, Samuel Kaski, and Juho Rousu. “Multi-view Kernel Completion”. In: *Machine Learning* 106.5 (May 2017), pp. 713–739. ISSN: 1573-0565.
- [7] Steffen Bickel and Tobias Scheffer. “Multi-View Clustering”. In: *ICDM*. 2004.
- [8] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Ed. by Michael Jordan, Jon Kleinberg, and Bernhard Schölkopf. 233 Spring Street, New York, NY 10013, USA: Springer Science+Business Media, LLC, 2006.
- [9] Bernhard E. Boser, Isabelle M. Guyon, and Vladimir N. Vapnik. “A Training Algorithm for Optimal Margin Classifiers”. In: *Proceedings of the Fifth Annual Workshop on Computational Learning Theory*. COLT '92. New York, NY, USA: ACM, 1992, pp. 144–152. ISBN: 0-89791-497-X. DOI: 10.1145/130385.130401. URL: <http://doi.acm.org/10.1145/130385.130401>.
- [10] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*. New York: Cambridge University Press, 2004. ISBN: 0521833787 9780521833783.
- [11] Johan Braeken and Marcel A. L. M. van Assen. “An Empirical Kaiser Criterion”. In: *Psychological Methods* 22 3 (2017), pp. 450–466.
- [12] Chih-Chung Chang and Chih-Jen Lin. “LIBSVM: A Library for Support Vector Machines”. In: *ACM Transactions on Intelligent Systems and Technology* 2 (2011). Software available at <http://www.csie.ntu.edu.tw/~cjlin/libsvm>, 27:1–27:27.

-
- [13] Kamalika Chaudhuri et al. “Multi-view Clustering via Canonical Correlation Analysis”. In: *ICML '09*. 2009, pp. 129–136.
- [14] Corinna Cortes and Vladimir Vapnik. “Support-Vector Networks”. In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. ISSN: 0885-6125.
- [15] Nello Cristianini and John Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*. 1st ed. Cambridge University Press, 2000. ISBN: 0521780195.
- [16] Jason V. Davis et al. “Information-Theoretic Metric Learning”. In: *Proceedings on International Conference on Machine Learning*. ACM, 2007, pp. 209–216.
- [17] Arthur P. Dempster, Nan M. Laird, and Donald B. Rubin. “Maximum Likelihood from Incomplete Data via the EM Algorithm”. In: *Journal of the Royal Statistical Society. Series B (Methodological)* 39.1 (1977), pp. 1–38.
- [18] Minghua Deng, Ting Chen, and Fengzhu Sun. “An Integrated Probabilistic Model for Functional Prediction of Proteins”. In: *Journal of Computational Biology* 11.2–3 (2004), pp. 463–475.
- [19] Richard O. Duda and Peter E. Hart. *Pattern Classification and Scene Analysis*. A Wiley Interscience Publication. Wiley, 1973.
- [20] Brian S. Everitt. *An Introduction to Latent Variable Models*. London: Chapman and Hall London Ltd, 1984. ISBN: 0412253100.
- [21] Mehmet Gönen. “Bayesian Efficient Multiple Kernel Learning”. In: *29th International Conference on Machine Learning*. 2012.
- [22] Trevor Hastie, Robert Tibshirani, and Jerome Harold Friedman. *The Elements of Statistical Learning*. Springer Verlag, Aug. 2001.
- [23] Geoffrey E. Hinton, Michael Revow, and Peter Dayan. “Recognizing Handwritten Digits Using Mixtures of Linear Models”. In: *Advances in Neural Information Processing Systems 7, [NIPS Conference, Denver, Colorado, USA, 1994]*. 1994, pp. 1015–1022. URL: <http://papers.nips.cc/paper/962-recognizing-handwritten-digits-using-mixtures-of-linear-models>.
- [24] Harold Hotelling. *Analysis of a Complex of Statistical Variables Into Principal Components*. Warwick & York, 1933.
- [25] Ian T. Jolliffe. *Principal Component Analysis*. New York: Springer Verlag, 2002.
- [26] Tsuyoshi Kato, Koji Tsuda, and Kiyoshi Asai. “Selective Integration of Multiple Biological Data for Supervised Network Inference”. In: *Bioinformatics* 21.10 (Feb. 2005), pp. 2488–2495.
- [27] Taishin Kin, Tsuyoshi Kato, and Koji Tsuda. “Protein Classification via Kernel Matrix Completion”. In: *Kernel Methods in Computational Biology*. Ed. by K. Tsuda In B. Schölkopf and J.P. Vert. The MIT Press, 2004. Chap. 3, pp. 261–274.
- [28] W. J. Krzanowski and F. H. C. Marriott. *Multivariate Analysis. Part II: Classification, Covariance Structures and Repeated Measurements*. Edward Arnold, 1994.

- [29] Ritwik Kumar et al. “Multiple Kernel Completion and Its Application to Cardiac Disease Discrimination”. In: *IEEE 10th International Symposium on Biomedical Imaging*. IBM Research. San Francisco, CA, USA, Apr. 2013, pp. 760–763.
- [30] Gert R. G. Lanckriet et al. “A Statistical Framework for Genomic Data Fusion”. In: *Bioinformatics* 20.16 (2004), pp. 2626–2635.
- [31] Gert R. G. Lanckriet et al. “Kernel-Based Data Fusion and Its Application to Protein Function Prediction in Yeast”. In: *Proceedings of the Pacific Symposium on Biocomputing*. 2004.
- [32] Gert R. G. Lanckriet et al. “Kernel-Based Integration of Genomic Data Using Semidefinite Programming”. In: *Kernel Methods in Computational Biology*. Ed. by K. Tsuda In B. Schölkopf and J.P. Vert. The MIT Press, 2004, pp. 231–259.
- [33] Tomoki Matsuzawa et al. “Stochastic Dykstra Algorithms for Metric Learning with Positive Definite Covariance Descriptors”. In: *The 14th European Conference on Computer Vision (ECCV2016)*. 2016, pp. 786–799.
- [34] Geoffrey J. McLachlan and Thriyambakam Krishnan. *The EM Algorithm and Extensions, 2nd ed.* Wiley Series in Probability and Statistics. Hoboken, NJ: Wiley, 2008.
- [35] Hans-Werner Mewes et al. *MIPS: A Database for Genomes and Protein Sequences*. *Nucleic Acids Res.*, 28. 2000.
- [36] Erik Mooi and Marko Sarstedt. “Factor Analysis”. In: *A Concise Guide to Market Research: The Process, Data, and Methods Using IBM SPSS Statistics*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 201–236. ISBN: 978-3-642-12541-6. DOI: 10.1007/978-3-642-12541-6_8. URL: https://doi.org/10.1007/978-3-642-12541-6_8.
- [37] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.
- [38] William Stafford Noble and Asa Ben-Hur. “Integrating Information for Protein Function Prediction”. In: *Bioinformatics—From Genomes to Therapies*. Weinheim, Germany: Wiley-VCH Verlag GmbH, 2008. Chap. 35, pp. 1297–1314.
- [39] Karl Pearson. “On Lines and Planes of Closest Fit to Systems of Points in Space”. In: *Philosophical Magazine* 2 (6 1901), pp. 559–572.
- [40] Rachele Rivero and Tsuyoshi Kato. *Parametric Models for Mutual Kernel Matrix Completion*. arXiv:1804.06095v1. Apr. 2018.
- [41] Rachele Rivero, Richard Lemence, and Tsuyoshi Kato. “Mutual Kernel Matrix Completion”. In: *IEICE Transactions on Information & Systems* E100-D.8 (Aug. 2017), pp. 1844–1851.
- [42] Sam Roweis. “EM Algorithms for PCA and SPCA”. In: *Proceedings of the 1997 Conference on Advances in Neural Information Processing Systems 10*. NIPS ’97. Cambridge, MA, USA: MIT Press, 1998, pp. 626–632. ISBN: 0-262-10076-2. URL: <http://dl.acm.org/citation.cfm?id=302528.302762>.

- [43] Sam Roweis and Zoubin Ghahramani. “A Unifying Review of Linear Gaussian Models”. In: 11.2 (Feb. 1999), pp. 305–345. ISSN: 0899-7667. DOI: 10.1162/089976699300016674. URL: <http://dx.doi.org/10.1162/089976699300016674>.
- [44] Bernhard Schölkopf, Koji Tsuda, and Jean-Philippe Vert. *Kernel Methods in Computational Biology*. Cambridge, Massachusetts: MIT Press, 2004.
- [45] Bernhard Schölkopf and Alexander J. Smola. *Learning with Kernels*. MIT Press, 2002.
- [46] J. Schur. “Über Potenzreihen, die im Innern des Einheitskreises beschränkt sind.” In: *Journal für die reine und angewandte Mathematik* 147 (1917). ISSN: 0075-4102; 1435-5345/e, pp. 205–232.
- [47] John Shawe-Taylor and Nello Cristianini. *Kernel Methods for Pattern Analysis*. Cambridge, UK: Cambridge University Press, 2004.
- [48] Michael E. Tipping and Christopher M. Bishop. “Mixtures of Probabilistic Principal Component Analyzers”. In: *Neural Computation* 11 (Feb. 1999), pp. 443–482.
- [49] Michael E. Tipping and Christopher M. Bishop. “Probabilistic Principal Component Analysis”. In: *Journal of the Royal Statistical Society, Series B* 21/3 (Jan. 1999), pp. 611–622.
- [50] Anusua Trivedi et al. “Multiview Clustering with Incomplete Views”. In: *NIPS*. 2010.
- [51] Koji Tsuda, Shotaro Akaho, and Kiyoshi Asai. “The em Algorithm for Kernel Matrix Completion with Auxiliary Data”. In: *Journal of Machine Learning Research* 4 (2003), pp. 67–81.
- [52] Vladimir N. Vapnik. *The Nature of Statistical Learning Theory*. New York, NY, USA: Springer-Verlag New York, Inc., 1995. ISBN: 0-387-94559-8.
- [53] Martin J. Wainwright and Michael I. Jordan. “Graphical Models, Exponential Families, and Variational Inference”. In: *Found. Trends Mach. Learn.* 1.1-2 (Jan. 2008), pp. 1–305.
- [54] Peter Whittle. “On Principal Components and Least Square Methods of Factor Analysis”. In: *Scandinavian Actuarial Journal* 1952.3-4 (1952), pp. 223–239. DOI: 10.1080/03461238.1955.10430696. eprint: <https://www.tandfonline.com/doi/pdf/10.1080/03461238.1955.10430696>. URL: <https://www.tandfonline.com/doi/abs/10.1080/03461238.1955.10430696>.
- [55] Christopher K. I. Williams and David Barber. “Bayesian Classification with Gaussian Processes”. In: *IEEE Trans. Pattern Anal. Mach. Intell.* 20.12 (Dec. 1998), pp. 1342–1351. ISSN: 0162-8828. DOI: 10.1109/34.735807. URL: <https://doi.org/10.1109/34.735807>.
- [56] David Williams and Lawrence Carin. “Analytical Kernel Matrix Completion with Incomplete Multi-View Data”. In: *Proceedings of the Workshop on Learning with Multiple Views*. 22nd ICML. Bonn, Germany, 2005.

-
- [57] Keith A. Yeomans and Paul A. Golder. “The Guttman-Kaiser Criterion as a Predictor of the Number of Common Factors”. In: *Journal of the Royal Statistical Society. Series D (The Statistician)* 31.3 (1982), pp. 221–229. ISSN: 00390526, 14679884. URL: <http://www.jstor.org/stable/2987988>.