A joint finite mixture model for clustering genes from beta, Gaussian and Bernoulli distributed data

Master's Thesis
Bioinformatics Masters
Degree Programme,
Institute of Medical Technology,
University of Tampere, Finland
Xiaofeng Dai
October, 2009

MASTER'S THESIS

Place:	University of Tampere, Bioinformatics Masters Degree Programme, Faculty of Medicine, Institute of Medical Technology, Tampere, Finland
Author:	Xiaofeng Dai
Title:	A joint finite mixture model for clustering genes from beta, Gaussian and Bernoulli distributed data
Pages:	55 pp + appendices 12 pp
Supervisor:	Prof. Harri Lähdesmäki
Reviewers:	Prof. Mauno Vihinen, Prof. Harri Lähdesmäki
Time:	October, 2009

Abstract

Background: Expression and protein-protein interaction data are often coupled in gene clustering, which has succeeded in many applications such as pathway discovery and function inference. However, asynchronous relations, which can be measured by protein-DNA binding data, are also crucial in regulatory network and should be taken into account. Thus, how to make efficient use of gene expression, protein-protein interaction and protein-DNA binding data has posed us a huge challenge.

Method: A beta-Gaussian-Bernoulli mixture model (BGBMM) is proposed to solve the aforementioned problem, assuming that protein-DNA binding probabilities, gene expression data and protein protein interactions can be modeled as beta, Gaussian and Bernoulli distributions, respectively. BGBMM is a natural extension of the beta mixture model, the Gaussian mixture model and the Bernoulli mixture model, which differs from other mixture model based methods by fusing three heterogeneous data types into a unified probabilistic modeling framework with each data type modeled as one component. BGBMM is demonstrated to be an efficient model for data fusion, and is applicable to any data sources that can be modeled as beta, Gaussian or Bernoulli distributed random variables. Further, it is easily extendable to data of any other parametric distributions in principle. A joint standard expectation maximization algorithm is developed to estimate parameters involved in BGBMM. Four approximation-based model selection methods, i.e., the Akaike information criterion, a modified AIC, the Bayesian information criterion, and the integrated classification likelihood-Bayesian information criterion, are compared in BGBMM, based on which the best performing ones are suggested for future use.

Results: The simulation tests and real case application show that combining three data sources into a single joint mixture model with each data type modeled as one component can highly improve the clustering accuracy. Also, applying BGBMM in mouse data reveals genes involved in the process of Toll-like receptor stimulated macrophage activation.

Preface

This work was completed under the supervision of Prof. Harri Lähdesmäki, who has offered me tremendous suggestions during this project. I, thereby, pay my greatest gratitude to him for his superb guidance and enormous help. Also, my deepest acknowledgement goes to Prof. Mauno Vihinen for his effort on carefully reviewing my thesis, offering constructive comments and also for his special care during my whole study process.

I will never forget the teachers who introduced me to the field of bioinformatics. Special thanks go to MSc. Martti Tolvanen, MSc. Filip Ginter, MSc. Pentti Riikonen and Dr. Bairong Shen, for the life beneficial knowledge they have taught me.

I would also like to thank the Tampere Graduate School in Information Science and Engineering (TISE) for financially supporting this project.

Last but not least is my special gratitude towards my family and friends, who have always been encouraging and supporting me. Without their love and understanding, this thesis could not be completed smoothly.

November, 2009

Xiaofeng Dai
Bioinformatics Programme,
Faculty of Medicine,
Institute of Medical Technology,
University of Tampere, Finland

Abbreviations

AP-MS	Affinity Purification followed by Mass Spectrometry
AIC	Akaike Information Criterion
AIC3	modified Akaike Information Criterion
BerMM	Bernoulli finite Mixture Model
BGMM	Beta-Gaussian joint finite Mixture Model
BGBMM	Beta-Gaussian-Bernoulli joint finite Mixture Model
BMM	Beta finite Mixture Model
BIC	Bayesian Information Criterion
ChIP	Chromatin Immunoprecipitation
cDNA	complementary DNA
EM	Expectation Maximization algorithm
GMM	Gaussian finite Mixture Model
GO	Gene Ontology
GRN	Gene Regulatory Network
ICL-BIC	Integrated Classification Likelihood - Bayesian Information Criterion
MCMC	Markov Chain Monte Carlo
mRNA	messenger RNA
PCR	Polymerase Chain Reaction
PPI	Protein-Protein Interaction
sBGMM	stratified Beta-Gaussian joint finite Mixture Model
TF	Transcription Factor
TLR	Toll-Like Receptor
Y2H	Yeast two-Hybrid

Contents

Chapter 1

Introduction

1.1 Background and motivation

Cluster analysis is a standard computational method for gene function discovery and many other explanatory data analysis. It is commonly applied in gene expression data assuming that genes share similar expression profiles have similar cellular functions and are likely to be involved in the same processes [1]. Although the feasibility of using expression data for gene clustering has been validated by many applications [2–5], this assumption is often violated by the transcriptional coherence of uncorrelated genes in response to, e.g., environmental stresses and challenged by the undetectable correlations of co-regulated genes due to their involvement in multiple pathways and high genomic noises [6].

A natural way of solving this problem is through data fusion. Information from multiple data sources reinforce each other to reduce genome-level noise within each data source and provide us a holistic view of the system from different perspectives. With the emergence of new techniques, many biological data sources other than gene expression data have become available, such as protein-protein interactions (PPIs), protein-DNA binding probabilities and DNA sequences. Among others, PPI data is often coupled with expression data in clustering, with the assumption that genes in the same pathway exhibit similar expression profiles due to synchronous activation and their protein products coordinate to achieve a particular task [7]. Also, 70% to 80% interacting protein pairs are shown to share at least one function [8]. Besides, successful applications on coupling expression and PPI data are frequently reported, such as pathway discovery [7] and gene function inference [6]. However, genes involved in the same

pathway or network also include asynchronous relations [9], which is not revealable by observing gene expression profiles and PPIs alone. Protein-DNA binding data, which mainly refers to transcription factor and promoter binding information, is essential in understanding a transcriptional regulatory network. Thus, how to make efficient use of protein-DNA binding, gene expression and PPI data is critical in understanding the mechanism behind a gene regulatory network.

1.2 Objectives

The objective of this thesis is to cluster genes by efficiently utilizing information at the gene, protein and gene products' interaction level, so that the genes clustered together are more likely to be involved in the same gene regulatory network (GRN), facilitating further data analysis. Specifically, the main objectives are listed below.

1. Developing an efficient model to cluster genes from protein-DNA binding probabilities, gene expression and protein protein interaction data, assuming that they are of beta, Gaussian and Bernoulli distributions, respectively.

2. Exploring the rules for efficiently utilizing heterogeneous data sources under the current clustering framework by comparing the current model with other models.

3. Applying the developed model to mouse data, aiming at performance test and novel information discovery.

1.3 Significance

BGBMM can be used to find genes that are involved in the same GRN. Thus, novel genes and/or their new functions can be found and/or predicted by referencing to the known genes that are clustered together with them. Also, the clustering results can be used as the prior for network construction.

BGBMM is not restricted to analyze protein-DNA binding probabilities, gene expression and PPI data, and can be applied to any data sources that are of beta, Gaussian and Bernoulli distributions. Further, the current framework is not limited to the specific type of problems

analyzed here, which is extendable to data of any parametric distributions in principle and applicable to problems in other research domain as well.

Chapter 2

Literature Review

2.1 Algorithm

2.1.1 Clustering algorithms

Clustering, defined as grouping objects into subsets such that objects within each group share more common features than those do not [10; 11], is a traditional unsupervised technique widely applied in many fields. There are many clustering algorithms [12–19], among which the typical approaches can be classified into three categories, i.e., the hierarchical methods, the partitioning methods, and the model-based methods [20].

Hierarchical methods

There are two types of hierarchical clustering algorithms, namely the agglomerative method and the divisive method, which recursively combines or splits a set of objects into bigger or smaller groups based on a certain criterion [21; 22]. Commonly applied criteria include single linkage [22], complete linkage [22], average linkage [22], group average linkage [22] and Ward's linkage [12], whose formulations are shown, respectively, in Equations 2.1 to 2.5 [12; 22]. Notice in these equations that $D(X, Y)$ and $d(\mathbf{x}, \mathbf{y})$ each represents the distance between two groups (X and Y) and two objects ($\mathbf{x}$ and $\mathbf{y}$, $\mathbf{x} \in X, \mathbf{y} \in Y$), respectively, the number of objects within groups X or Y is shown as n_X or n_Y, and ESS is the abbreviation of 'error sum of squares'.

$$D(X,Y) = \min_{\mathbf{x}\in X,\mathbf{y}\in Y} d(\mathbf{x},\mathbf{y}) \tag{2.1}$$

$$D(X,Y) = \max_{\mathbf{x}\in X,\mathbf{y}\in Y} d(\mathbf{x},\mathbf{y}) \tag{2.2}$$

$$D(X,Y) = \frac{\sum_{i=1}^{n_X}\sum_{j=1}^{n_Y} d(x_i,y_j)}{n_X \times n_Y} \tag{2.3}$$

$$D(X,Y) = d(\frac{\sum_{i=1}^{n_X} x_i}{n_X}, \frac{\sum_{j=1}^{n_Y} y_j}{n_Y}) \tag{2.4}$$

$$D(X,Y) = ESS(XY) - ESS(X) - ESS(Y), \text{ where} \tag{2.5}$$
$$ESS(X) = \sum_{i=1}^{n_X} |x_i - \frac{1}{n_X}\sum_{j=1}^{n_X} x_j|^2$$

As shown in these criteria, the group similarity is often scaled by distance, for which different measurements can be employed depending on the purpose and the objects' characteristics. Among others, Euclidean distance [23], Mahalanobis distance [24], Manhattan distance [25], and hamming distance [26] are most commonly seen. These distances can be computed from Equations 2.6 to 2.9, respectively, where p ($p \in \{1, \infty\}$) is the dimension of each observation and 'Cov' represents the covariance matrix of two objects.

$$d(\mathbf{x},\mathbf{y}) = \sqrt{\sum_{i=1}^{p}(x_i - y_i)^2} \tag{2.6}$$

$$d(\mathbf{x},\mathbf{y}) = \sqrt{(\mathbf{x}-\mathbf{y})^T \mathrm{Cov}^{-1}(\mathbf{x}-\mathbf{y})} \tag{2.7}$$

$$d(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{p} |x_i - y_i| \tag{2.8}$$

$$d(\mathbf{x},\mathbf{y}) = \sum_{i=1}^{p} h_i, \quad h_i = \begin{cases} 1 & \text{if } x_i \neq y_i \\ 0 & \text{if } x_i = y_i \end{cases} \tag{2.9}$$

Hierarchical clustering is favored due to its simple yet intuitively reasonable principle which, however, requires expert domain knowledge to define the distance measurement for a particular problem. For example, Euclidean distance is suitable when the data is representable in vector space but should be avoided in high-dimensional text clustering [27]. Moreover, the number of clusters depends highly on the granularity chosen by the user, rendering the results subjective to the pre-assumptions [20]. Also, outliers, if exist, may distort the clustering results.

Partitioning methods

Partitioning methods is another class of heuristic methods besides hierarchical clustering. The principle is to iteratively reallocate data points across groups until no further improvement is obtainable [13; 20]. K-means [13] is a typical and the most representative partitioning algorithm. It is based on the criterion that each object belongs to its closest group, where the group is represented by the mean of its objects. In particular, with a given k, the algorithm partitions n observations, $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, into k groups ($\mathbf{G} = \{G_1, G_2, \ldots G_k\}$) by minimizing the total intra-cluster variance, i.e., $\underset{\mathbf{G}}{\operatorname{argmin}} \sum_{i=1}^{k} \sum_{\mathbf{x}_j \in \mathbf{G}_i} (\mathbf{x}_j - \mu_i)^2$, where μ_i is the mean of $\mathbf{G}_i$.

It is seen from K-means that the number of clusters has to be pre-assumed or known. Also, the clustering results may be contaminated by outliers [20]. Successive efforts have been devoted to search their remedies which, however, mostly involve techniques out of the domain of partitioning methods. For example, X-means (extended from K-means) solves the problem of selecting the number of clusters via using model selection criteria [28].

Despite those disadvantages, partitioning methods is widely applied due to their simplicities. Many algorithms, such as fuzzy C-means [29], quality threshold clustering [15] and partitioning around medoids [30], also belong to this category. Specifically, 'fuzzy C-means' assigns each data point to each cluster with a certain probability [29], 'quality threshold' only groups data points whose similarities are high enough [15], and 'partitioning around medoids' minimizes a sum of dissimilarities and allows the user to choose the number of clusters through graphical display [30].

Model based methods

Model based methods attempt to optimize the fitness between the data and the model where the data is assumed to be generated [16; 17; 31; 32]. Model based methods can be further

classified into finer groups, such as finite mixture models [16], infinite mixture models [17], model based hierarchical clustering [31], and specialized model-based partitioning clustering algorithm [32], among which finite model based methods are most commonly seen.

In finite model-based clustering, each observation $\mathbf{o}_j$, where $j = 1, \ldots, n$ and n is the number of genes, is drawn from a finite mixture distribution with the prior probability π_i, component-specific distribution $f_i^{(g)}$ and its parameters θ_i. The formula is given in Equation 2.10 [16], where $\theta = \{(\pi_i, \theta_i) : i = 1, \ldots, g\}$ is used to denote all the unknown parameters, with the restriction that $0 < \pi_i \leq 1$ for any i and $\sum_{i=1}^{g} \pi_i = 1$. Note that g is the number of components in this model, and the superscript (g) is ignored from $f_i^{(g)}$ for simplicity in Chapter 3.

$$f(\mathbf{o}_j|\theta) = \sum_{i=1}^{g} \pi_i f_i^{(g)}(\mathbf{o}_j|\theta_i) \tag{2.10}$$

The parameters of model based methods are generally estimated by expectation maximization (EM) algorithm, which is commonly used to obtain maximum likelihood estimation from incomplete data [16; 20; 31–33] as discussed in subsection 2.1.2. The choice of the number of clusters is often casted as model selection problems, i.e., based on certain model selection criteria [16; 20; 31–33] as reviewed in subsection 2.1.3.

As aforementioned, the problem of choosing the number of clusters which is generically inherited by heuristic methods can be naturally solved by model based methods [20]. Also, outliers can be treated as distinct mixture components [16; 20; 33]. Further, the statistical formulation endows model based methods with more superiority over their heuristic alternatives [20].

2.1.2 Expectation maximization algorithm

Expectation maximization algorithm, abbreviated as EM algorithm, is an iterative method that estimates the parameters in a probabilistic model by maximizing its likelihood, where the model is assumed to depend on unobserved latent variables [34]. In particular, the maximum likelihood estimation is found by iterating over an expectation (E) step and a maximization (M) step [34], i.e.,

- E step: at the m^{th} iteration, compute the expected value of the log-likelihood function, $Q(\theta|\theta^{(m)})$, given the observations X under the parameter estimation at that iteration, i.e.,

$\theta^{(m)}$.

$$Q(\theta|\theta^{(m)}) = E_{\mathbf{c}|X,\theta^{(m)}}\left[\log(L|X,\theta)\right].$$

- M step: update the parameters at the $(m+1)^{\text{th}}$ iteration by maximizing $Q(\theta|\theta^{(m)})$, i.e.,

$$\theta^{(m+1)} = \underset{\theta}{\text{argmax}} Q(\theta|\theta^{(m)}).$$

2.1.3 Model selection criteria

Generally applied model selection criteria can be roughly classified as likelihood-based methods [35] and approximation-based methods [33; 36–40], which are introduced, separately, below.

Likelihood based model selection criteria

Likelihood-based methods can be further divided into the bootstrap method and the cross-validation method [35].

The bootstrap method proceeds with the null hypothesis H_0 that the model has g_0 components, then it evaluates the likelihood ratio, λ, between the model with g_0 mixture components and that with $g_0 + 1$ components for many (denote it as K) repetitions to approximate the null distribution of $-2\log\lambda$, and finally the p-value is assessed by reference with respect to the ordered bootstrap replications of $-2\log\lambda$, where the j^{th} order statistic of the K replicates is taken as an estimate of the quantile of order $j/(K+1)$ [41]. This method is so far only reported to be efficient in solving small problems, i.e., problems with small data size and a few number of clusters [41; 42]. Also, the results are shown to be slightly biased towards the null hypothesis [41].

The cross-validation method includes many alternatives depending on how the partitions are chosen. Typically, it partitions the data into complementary subsets, computes the 'test log-likelihood' of the fitted model by fitting the model with data from one subset (training data) and evaluating the log-likelihood of the model with data from another (testing data), divides the 'test log-likelihood' by the number of data points in the testing data, and evaluates the average of this expectation over several repetitions. In principle, besides the 'test log-likelihood', any function that scores the fitness between the model and the data is feasible to be implemented under this framework [35]. However, as seen from its procedure, cross-validation method is inefficient in data usage since only partial data is involved in model training [35].

Likelihood based methods have some successful applications in solving model selection problems [35; 41; 42] which, however, are still limited in use due to their heavy computational costs [35].

Approximation based model selection criteria

Approximation-based methods are a class of model selection criteria most widely applied and favored due to their computational efficiency and simplicity. These methods include 'closed-form approximations to the Bayesian solution', 'Monte Carlo sampling of the Bayesian solution', and 'penalized likelihood methods' [35].

The 'closed-form approximations to the Bayesian solution' and the 'Monte Carlo sampling of the Bayesian solution' treat the number of components as a parameter and estimate its posterior distribution from the data and the model by the Bayesian approach [35]. These two approaches differ in that the 'closed-form approximations to the Bayesian solution' approximates this posterior analytically and the other one estimates it via Markov chain Monte Carlo (MCMC) sampling [35].

Penalized likelihood methods select the model based on its data log-likelihood and a penalty factor that increases with the model complexity [35; 43]. Various penalization methods exist for model selection, such as Bayesian information criterion (BIC) [37; 40], integrated classification likelihood-BIC (ICL-BIC, simplified as ICL in this thesis) [33], Akaike information criterion (AIC) [36; 39], and modified AIC (such as AIC3 [38; 39]).

Approximation-based methods suffer from the theoretical limitation that the results may be subjective to the underlying approximations, since the penalty term is often approximated as the data size approaches infinity [35]. However, they are still the most popular methods for model selection due to their simple yet powerful implementation compared with other methods [35].

2.2 Data

Gene expression data, protein-DNA binding probabilities, and protein-protein interactions (PPI) are jointly utilized in gene clustering in this thesis, with the assumption that these data are of Gaussian, beta and Bernoulli distributions, respectively.

2.2.1 Gene expression data

Gene expression data is perhaps the most widely applied information source in gene clustering. Many techniques, including hierarchical methods [21; 22], partitioning methods [13; 22], and model based methods [16; 17; 22; 31; 32], are applied to this type of data [40; 44–50], with the aim of, e.g., finding functional related genes.

Gene expression data is typically obtained from DNA microarrays, which is a high-throughput technique that measures the cellular abundance of messenger RNA (mRNAs) [25]. DNA microarray has two commonly used platforms, which are the spotted microarrays and oligonucleotide microarrays [25; 51]. The two types of microarrays differ mainly in whether the probes of the interested genes are built on the chips or not. Specifically, in spotted DNA microarrays, the probes, i.e., complementary DNAs (cDNA), short polymerase chain reaction (PCR) products, or oligonucleotide chains of the genes of interest are spotted on the chip; and in oligonucleotide chips, oligonucleotides are built or synthesized *in situ* on the microarrays [51]. There are many different techniques of building probes *in situ*, among which Affymetrix chips and Agilent chips are widely used which build probes on microarray chips through a photolithographic process and an ink-jet synthesizer, respectively [51].

DNA microarray experiments can be divided into two types, i.e., double-channel experiment and single-channel experiment [25; 51], whose choice largely depends on the experimental purpose and the chip type. The main difference between these two kinds of experiments are that the outputs of a double-channel experiment are the intensity ratios and those obtained from a single-channel experiment are the raw intensities. Which type of experiment could be carried out is sometimes linked with the chip types. For example, cDNA chips are typically double-channel microarrays, and Affymetrix could only be carried out in a single channel manner [51].

There are also, among others, two kinds of experimental designs commonly carried out by microarrays, which are 'time series', a series of samples following each other in time, and 'conditions' which are discrete time points or states [52]. Generally, time series is used to study a development in time, and conditions are typically employed, e.g., to infer gene functions or genes' relationships [52]. It is worth knowing that several replicates of each condition are often carried out, where the number of replicates is chosen such that it can provide the necessary information to judge the significance of the conclusions [52].

Errors may be introduced to the microarray results at each step due to systematic variations caused by, e.g., hybridization of asynchronous cells and cross-hybridization, and random

mistakes because of, e.g., human operations [51]. Thus, the raw data needs to be preprocessed and normalized before being analyzed [51]. Typically, filtered log-transformed DNA microarray data is often assumed to be of Gaussian distribution [53] and widely used for many applications, such as gene clustering [44; 45] which assumes that genes share similar expression profiles are functionally related or synchronously expressed [45].

2.2.2 Protein-DNA binding data

Protein-DNA interactions are an essential regulator in gene expression [9; 54], because of which enormous experimental [55–59] and computational [60–71] efforts are devoted to measure or estimate these physical binding mechanisms and affinities.

ChIP (chromatin immunoprecipitation) related techniques are typically adopted experimentally to measure the protein-DNA interactions. It uses formaldehyde to crosslink proteins to DNA in the living cells, shears the chromatin into random-sized pieces by sonication, enriches the protein-bound DNA fragments of interest by immunoprecipitation using the corresponding protein specific antibody, reverses the protein-DNA cross-links, purifies the interested fragments via PCR, and uses some downstream techniques to determine the DNA sequence [72]. Based on the different downstream methods, currently widely acknowledged techniques include ChIP-chip [55; 56] and ChIP-seq [57–59; 73]. ChIP-chip is a technique that uses DNA microarrays to detect the bounded DNA sequences [55; 56], and ChIP-seq employs the next generation massively parallel sequencing for this task [57–59; 73]. Although the application of ChIP-seq is currently limited by its high cost, it is now on the trend of taking the dominance by its high resolution and low requirement of the amount of ChIP DNAs [59].

Besides the emergence of new experimental techniques, many computational methods are also developed aiming at protein-DNA binding sites discovery and/or their binding affinities prediction [60–70; 74]. Efforts on this field can be viewed as from two perspectives, i.e., by analyzing protein-DNA binding complex or studying DNA sequences alone. From the first perspective, protein-DNA binding sites are initially identified by the amino acids located at the protein-DNA binding interface alone [63; 74], which are characterized later by also integrating protein structural information [64; 65]. An alternative approach from this perspective is to predict the binding affinities through all-atom modeling, i.e., modeling all the atoms in a protein-DNA complex where the binding energies at all the locations are evaluated [66–69]. From the second aspect, the motif discovery algorithms, which focus on searching for novel

binding motifs from a collection of short sequences that are assumed to contain a common regulatory motif, is typically used to discover protein-DNA binding sites [70], and the protein-DNA binding site prediction algorithms, which predict the putative binding sites based on the given binding specificities (being represented, e.g., as position specific weight matrix), is generally used to do further predictions based on the information observed from motif discoveries or experimental evidences [60–62]. Recently, an emerging trend is to predict functional binding sites [71], providing the possibility of obtaining data of functional protein-DNA interactions.

The protein-DNA binding data employed in this thesis is the computational predictions from ProbTF [62], a probabilistic TFBS prediction algorithm, assuming that the obtained probabilities are beta distributed.

2.2.3 Protein-protein interaction data

Besides protein-DNA interactions, PPIs also play a crucial regulatory role in many cellular process, since the formation of polymers is required for most proteins to exert their functions [9; 54].

The yeast two-hybrid (Y2H) system [75; 76] is a commonly used technique for PPI detection. The Y2H system uses a reporter gene to signal whether two proteins (called the 'prey' and the 'bait') interact [75; 76]. Specifically, the transcription factor (TF) that controls the expression of the reporter gene has two function domains, i.e., the binding domain and the activation domain, which are split and fused with the 'bait' and the 'prey', respectively; when the two proteins interact, the two domains combine and activate the expression of the reporter gene [75; 76]. Large-scale Y2H could now be carried out by using, e.g., a colony-array format [77–79],

Another widely applied technique to find PPIs is the affinity purification followed by mass spectrometry (AP-MS) [76; 80]. In such an experiment, the two potential interactive proteins are also named 'bait' and 'prey'. In the AP step, the 'bait' is tagged and extracted together with 'prey' through co-immunoprecipitation or tandem affinity purification [76; 80]. Then, the extracted proteins are identified by MS [76; 80], which determines the identity of a protein by ionizing the molecule and measuring its mass-to-charge ratio [81]. Applying AP-MS in a high throughput fashion has now become available, e.g., in yeast [82–84].

Y2H and AP-MS, although both used for PPI detection, each has its own strength and is suitable for different tasks. For example, Y2H is able to find transient interactions which

is beyond AP-MS's reach; AP-MS is biased by the amount of proteins whereas Y2H is not; AP-MS can be used to find one-to-many relationships whereas Y2H is only limited to detecting binary interactions; the interactions found by AP-MS exist *in vivo*, where those discovered by Y2H may not exist under physiological conditions; AP-MS can find weak interactions which are not detectable by Y2H [76].

PPIs are often stored in databases such as DIP [85; 86], MINT [87], MIPS/MPact [88], IntAct [89], BioGRID [90], and HPRD [91]. In practice, it is often to integrate and use information from multiple databases [76] since large discrepancies and contradictions exist among different sources [92–97].

PPIs used in this thesis is queried from six databases, i.e., DIP [85; 86], MINT [87], MIPS/MPact [88], IntAct [89], BioGRID [90], and HPRD [91] via a data integration platform, PINA [98].

Chapter 3

Method

This chapter presents the framework of BGBMM, its EM algorithm and the tested approximation-based model selection criteria.

3.1 Clustering framework

In BGBMM, define $\theta = [\theta_1, \theta_2, \theta_3, \pi]^T$, $\pi = [\pi_1, \ldots, \pi_g]^T$, $\theta_1 = [\alpha_{11}, \ldots, \alpha_{gp_1}, \beta_{11}, \ldots, \beta_{gp_1}]^T$, $\theta_2 = \left[\mu_{11}, \ldots, \mu_{gp_2}, \sigma_1^2, \ldots, \sigma_{p_2}^2\right]^T$, and $\theta_3 = [q_{11}, \ldots, q_{gp_3}]^T$, where p_1, p_2 and p_3 each represents the dimension of the observations in beta, Gaussian and Bernoulli mixture model, respectively. Also denote X, Y and Z as the observations of beta, Gaussian and Bernoulli distributed data, respectively, function f of $\mathbf{x}$, $\mathbf{y}$, $\mathbf{z}$ as the density function of beta, Gaussian and Bernoulli distribution, respectively, and $\mathbf{o} = [\mathbf{x}^T, \mathbf{y}^T, \mathbf{z}^T]^T$.

BGBMM is a joint mixture model of beta, Gaussian and Bernoulli distributions, with the assumption that, for each component i, data of the three distributions are independent.

In the part of beta mixture model (BMM), each component is assumed to be the product of p_1 independent beta distributions, whose probability density function is defined in Equation 3.1, where $\theta_{1i} = [\alpha_{i1}, \ldots, \alpha_{ip_1}, \beta_{i1}, \ldots, \beta_{ip_1}]$ and $\mathbf{x} = [x_1, \ldots, x_{p_1}]^T$.

$$f_i(\mathbf{x}|\theta_{1i}) = \prod_{u=1}^{p_1} \frac{x_u^{\alpha_{iu}-1}(1-x_u)^{\beta_{iu}-1}}{B(\alpha_{iu}, \beta_{iu})} \tag{3.1}$$

Likewise, each component is assumed to follow a Gaussian distribution in the Gaussian

mixture model (GMM), whose probability density function of each component for each gene is defined in Equation 3.2, where $\theta_{2i} = [\mu_{i1}, \ldots, \mu_{ip_2}, \sigma^2_{i1}, \ldots, \sigma^2_{ip_2}]$, $\mu_i = [\mu_{i1}, \ldots, \mu_{ip_2}]$. Note that a diagonal covariance matrix, $V = \text{diag}(\sigma^2_1, \sigma^2_2, \ldots, \sigma^2_{p_2})$ ($|V| = \prod_{v=1}^{p_2} \sigma^2_v$), is used in the GMM part to reduce the number of parameters need to be estimated, which is especially useful when dealing with high-dimensional data.

$$f_i(\mathbf{y}|\theta_{2i}) = \frac{1}{(2\pi)^{\frac{p_2}{2}}|V|^{\frac{1}{2}}} \exp\big(-\frac{1}{2}(\mathbf{y}-\mu_i)^T V^{-1}(\mathbf{y}-\mu_i)\big), \tag{3.2}$$

In the part of Bernoulli mixture model (BerMM), each component is modeled as Bernoulli distribution, with the probability density function for each gene defined in Equation 3.3, where $\theta_{3i} = [q_{i1}, \ldots, q_{ip_3}]$.

$$f_i(\mathbf{z}_w|\theta_{3i}) = \prod_{w=1}^{p_3} q_{iw}^{\mathbf{z}_w}(1-q_{iw})^{(1-\mathbf{z}_w)}, \tag{3.3}$$

3.2 Expectation maximization algorithm

A standard EM algorithm is applied to jointly estimate the parameters, θ, iteratively, where the data log-likelihood (natural logarithm is referred to throughout this thesis) is written as Equation 3.4. Recall that $\mathbf{o}_j$ represents the observation j ($j = 1, \ldots, n$), n is the number of genes, g is the number of components in this model, π_i is the prior probability of drawing an observation from the component i ($0 < \pi_i \leq 1$, $\sum_{i=1}^{g} \pi_i = 1$), $f_i^{(g)}$ is the component-specific distribution, and θ_i represents the parameters of component i ($\theta = \{(\pi_i, \theta_i) : i = 1, \ldots, g\}$).

$$\log L(\theta) = \sum_{j=1}^{n} \log(\left[\sum_{i=1}^{g} \pi_i f_i(\mathbf{o}_j|\theta_i)\right]) \tag{3.4}$$

The direct maximization of Equation 3.4 is difficult, which can be casted in the framework of incomplete data. Since it is assumed that data of different distributions are independent, L_c can be factored as Equation 3.5,

$$L_c(\theta) = f(X|\mathbf{c}, \theta) f(Y|\mathbf{c}, \theta) f(Z|\mathbf{c}, \theta) f(\mathbf{c}|\theta). \tag{3.5}$$

If define $c_j \in \{1, \ldots, g\}$ as the clustering membership of $\mathbf{o}_j$, then the complete data log-

likelihood can be written as Equation 3.6, where $\chi(c_j = i)$ is the indicator function of whether $\mathbf{o}_j$ is from the i^{th} component or not.

$$\log L_c(\theta) = \sum_{j=1}^{n} \sum_{i=1}^{g} \chi(c_j = i) \log \left(\pi_i f_i(\mathbf{o}_j|\theta_i)\right), \tag{3.6}$$

In the EM algorithm, E step computes the expectation of the complete data log-likelihood as shown in Equation 3.7 [99].

$$\begin{aligned}
& Q(\theta|\theta^{(m)}) \\
= \; & E_{\mathbf{c}|\mathbf{x}_j,\mathbf{y}_j,\mathbf{z}_j,\theta^{(m)}}(\log L_c) \\
= \; & E_{\mathbf{c}|X,Y,Z,\theta^{(m)}}\left[\log\left(f(X|\mathbf{c},\theta)f(Y|\mathbf{c},\theta)f(Z|\mathbf{c},\theta)f(\mathbf{c}|\theta)\right)\right] \\
= \; & E_{\mathbf{c}|X,Y,Z,\theta^{(m)}}\left[\log\left(\prod_{j=1}^{n} f(\mathbf{x}_j|c_j,\theta)f(\mathbf{y}_j|c_j,\theta)f(\mathbf{z}_j|c_j,\theta)f(c_j|\theta)\right)\right] \\
= \; & E_{\mathbf{c}|X,Y,Z,\theta^{(m)}} \\
& \left[\log\left(\prod_{j=1}^{n}\left(\prod_{u=1}^{p_1} f(x_{ju}|c_j,\theta)\right)\left(\prod_{v=1}^{p_2} f(y_{jv}|c_j,\theta)\right)\left(\prod_{w=1}^{p_3} f(z_{jw}|c_j,\theta)\right)\left(f(c_j|\theta)\right)\right)\right] \\
= \; & \sum_{j=1}^{n} E_{c_j|\mathbf{x}_j,\mathbf{y}_j,\mathbf{z}_j,\theta^{(m)}} \\
& \left[\log\left(\prod_{u=1}^{p_1} f(x_{ju}|c_j,\theta)\right) + \log\left(\prod_{v=1}^{p_2} f(y_{jv}|c_j,\theta)\right) + \log\left(\prod_{w=1}^{p_3} f(z_{jw}|c_j,\theta)\right) + \log\left(f(c_j|\theta)\right)\right] \\
= \; & \sum_{j=1}^{n} E_{c_j|\mathbf{x}_j,\mathbf{y}_j,\mathbf{z}_j,\theta^{(m)}} \\
& \left[\sum_{u=1}^{p_1} \log\left(f(x_{ju}|c_j,\theta)\right) + \sum_{v=1}^{p_2} \log\left(f(y_{jv}|c_j,\theta)\right) + \sum_{w=1}^{p_3} \log\left(f(z_{jw}|c_j,\theta)\right) + \log\left(f(c_j|\theta)\right)\right] \\
= \; & \sum_{j=1}^{n} E_{c_j|\mathbf{x}_j,\mathbf{y}_j,\mathbf{z}_j,\theta^{(m)}}\left[\log\left(f(\mathbf{x}_j|c_j,\theta_1)\right)\right] + \sum_{j=1}^{n} E_{c_j|\mathbf{x}_j,\mathbf{y}_j,\mathbf{z}_j,\theta^{(m)}}\left[\log\left(f(\mathbf{y}_j|c_j,\theta_2)\right)\right] \\
& + \sum_{j=1}^{n} E_{c_j|\mathbf{x}_j,\mathbf{y}_j,\mathbf{z}_j,\theta^{(m)}}\left[\log\left(f(\mathbf{z}_j|c_j,\theta_3)\right)\right] + \sum_{j=1}^{n} E_{c_j|\mathbf{x}_j,\mathbf{y}_j,\mathbf{z}_j,\theta^{(m)}}\left[\log\left(f(c_j|\pi)\right)\right],
\end{aligned} \tag{3.7}$$

By computing the expectation, Equation 3.7 becomes Equation 3.8, where τ is calculated by Equation 3.8 according to Bayes' rule [99].

$$\begin{aligned} Q(\theta|\theta^{(\mathbf{m})}) &= \sum_{j=1}^{n}\sum_{i=1}^{g}\tau_{ji}^{(m)}\log\left(f_i(\mathbf{x}_j|\theta_{1i})\right) + \sum_{j=1}^{n}\sum_{i=1}^{g}\tau_{ji}^{(m)}\log\left(f_i(\mathbf{y}_j|\theta_{2i})\right) \\ &\quad + \sum_{j=1}^{n}\sum_{i=1}^{g}\tau_{ji}^{(m)}\log\left(f_i(\mathbf{z}_j|\theta_{3i})\right) + \sum_{j=1}^{n}\sum_{i=1}^{g}\tau_{ji}^{(m)}\log(\pi_i) \\ &= \sum_{j=1}^{n}\sum_{i=1}^{g}\tau_{ji}^{(m)}\log(\pi_i f_i(\mathbf{x}_j|\theta_{1i})f_i(\mathbf{y}_j|\theta_{2i})f_i(\mathbf{z}_j|\theta_{3i})) \end{aligned}$$

$$\begin{aligned} \tau_{ji} &= f(c_j = i|\mathbf{x}_j, \mathbf{y}_j, \mathbf{z}_j, \theta_i^{(m)}) \\ &= \frac{f(\mathbf{x}_j|c_j = i, \theta_{1i}^{(m)})f(\mathbf{y}_j|c_j = i, \theta_{2i}^{(m)})f(\mathbf{z}_j|c_j = i, \theta_{3i}^{(m)})f(c_j = i|\pi_i^{(m)})}{\sum_{i'=1}^{g} f(\mathbf{x}_j|c_j = i', \theta_{1i'}^{(m)})f(\mathbf{y}_j|c_j = i', \theta_{2i'}^{(m)})f(\mathbf{z}_j|c_j = i', \theta_{3i'}^{(m)})f(c_j = i'|\pi_{i'}^{(m)})} \\ &= \frac{\pi_i^{(m)} f_i(\mathbf{x}_j|\theta_{1i}^{(m)})f_i(\mathbf{y}_j|\theta_{2i}^{(m)})f_i(\mathbf{z}_j|\theta_{3i}^{(m)})}{\sum_{i'=1}^{g}\pi_{i'}^{(m)} f_{i'}(\mathbf{x}_j|\theta_{1i'}^{(m)})f_{i'}(\mathbf{y}_j|\theta_{2i'}^{(m)})(\mathbf{z}_j|\theta_{3i'}^{(m)})} \end{aligned}$$

Note that $\tau_{ji}^{(m)}$ is the estimated posterior probability of $\mathbf{o}_j$ coming from component i at iteration m, and each $\mathbf{o}_j$ can be assigned to its component based on $\left\{i_0|\tau_{ji_0} = \max_i(\tau_{ji})\right\}$. Equations 3.7 and 3.8 show that the assumption that the beta, Gaussian and Bernoulli distributed data are independent carries over to the expected log-likelihood as well.

Define Equations 3.8 to 3.11, then Equation 3.8 becomes Equation 3.12 [99].

$$
\begin{aligned}
Q_1(\theta_1) &= \sum_{j=1}^{n}\sum_{i=1}^{g} \tau_{ji}^{(m)} \log\left(f_i(\mathbf{x}_j|\theta_{1i})\right) && (3.8)\\
Q_2(\theta_2) &= \sum_{j=1}^{n}\sum_{i=1}^{g} \tau_{ji}^{(m)} \log\left(f_i(\mathbf{y}_j|\theta_{2i})\right) && (3.9)\\
Q_3(\theta_3) &= \sum_{j=1}^{n}\sum_{i=1}^{g} \tau_{ji}^{(m)} \log\left(f_i(\mathbf{z}_j|\theta_{3i})\right) && (3.10)\\
Q_4(\pi) &= \sum_{j=1}^{n}\sum_{i=1}^{g} \tau_{ji}^{(m)} \log(\pi_i) && (3.11)\\
Q(\theta) &= Q_1(\theta_1) + Q_2(\theta_2) + Q_3(\theta_3) + Q_4(\pi) && (3.12)
\end{aligned}
$$

Now the problem is converted to the convex optimization problem, with the Lagrangian function shown as Equation 3.13 [99].

$$
\begin{aligned}
\mathcal{L}(\theta) &= \mathcal{L}(\theta_1, \theta_2, \theta_3, \pi)\\
&= Q_1(\theta_1) + Q_2(\theta_2) + Q_3(\theta_3) + Q_4(\pi) + \lambda\left(1 - \sum_{i'=1}^{g}\pi_{i'}\right) && (3.13)
\end{aligned}
$$

It is seen from Equation 3.13 that the standard EM for BGBMM will reduce to the standard EM for beta, Gaussian, and Bernoulli distribution, respectively, when the dimensions of the data of the other two distributions go to zero. In the EM algorithm of BGBMM, the parameters of BMM is estimated with the Newton-Raphson method, and those of the GMM and BerMM are updated with closed formulas.

Parameter estimation in BMM

Specifically, let $\theta_{1i} = (\alpha_i, \beta_i)$, then the new estimate $\theta_{1i}^{(m+1)}$ is obtained following Equation 3.14 [99], where $H^{-1}(\theta_{1i}^{(m)})$ is the Hessian matrix evaluated at $\theta_{1i}^{(m)}$.

$$
\theta_{1i}^{(m+1)} = \theta_{1i}^{(m)} - H^{-1}(\theta_{1i}^{(m)})\nabla_{\theta_{1i}}\mathcal{L}(\theta^{(m)}) \qquad \theta_{1i} \geq \mathbf{1} \tag{3.14}
$$

The derivation and formula [99] of $\theta_{1i}^{(m)}$ are given below, where Ψ and Ψ' represent the digamma (the first logarithmic derivative of the gamma function) and trigamma (the second logarithmic derivative of the gamma function) functions in Matlab, respectively, and $\{u = 1, \ldots, p_1\}$.

$$\begin{aligned} \because Q_1(\theta_1) &= \sum_{j=1}^{n}\sum_{u=1}^{p_1}\sum_{i=1}^{g}\tau_{ji}\left((\alpha_{iu}-1)\log(x_{ju}) + (\beta_{iu}-1)\log(1-x_{ju}) - \log(\frac{\Gamma(\alpha_{iu})\Gamma(\beta_{iu})}{\Gamma(\alpha_{iu}+\beta_{iu})})\right) \\ \nabla_{\theta_1}\mathcal{L}(\theta) &= \nabla_{\theta_1}Q_1(\theta_1) \end{aligned} \tag{3.15}$$

$$\begin{aligned} \therefore \frac{\partial}{\partial\alpha_{iu}}\mathcal{L}(\theta^{(m)}) &= \sum_{j=1}^{n}\tau_{ji}^{(m)}\left(\log(x_{ju}) - \Psi(\alpha_{iu}^{(m)}) + \Psi(\alpha_{iu}^{(m)}+\beta_{iu}^{(m)})\right) \\ \frac{\partial}{\partial\beta_{iu}}\mathcal{L}(\theta^{(m)}) &= \sum_{j=1}^{n}\tau_{ji}^{(m)}\left(\log(1-x_{ju}) - \Psi(\beta_{iu}^{(m)}) + \Psi(\alpha_{iu}^{(m)}+\beta_{iu}^{(m)})\right) \\ \frac{\partial^2}{\partial\alpha_{iu}^2}\mathcal{L}(\theta^{(m)}) &= \sum_{j=1}^{n}\tau_{ji}^{(m)}\left(\Psi'(\alpha_{iu}^{(m)}+\beta_{iu}^{(m)}) - \Psi'(\alpha_{iu}^{(m)})\right) \\ \frac{\partial^2}{\partial\beta_{iu}^2}\mathcal{L}(\theta^{(m)}) &= \sum_{j=1}^{n}\tau_{ji}^{(m)}\left(\Psi'(\alpha_{iu}^{(m)}+\beta_{iu}^{(m)}) - \Psi'(\beta_{iu}^{(m)})\right) \\ \frac{\partial^2}{\partial\alpha_{iu}\partial\beta_{iu}}\mathcal{L}(\theta^{(m)}) &= \sum_{j=1}^{n}\tau_{ji}^{(m)}\left(\Psi'(\alpha_{iu}^{(m)}+\beta_{iu}^{(m)})\right) \end{aligned}$$

$$\begin{aligned} H^{-1}(\theta_{1i}^{(m)}) &= \begin{bmatrix} \frac{\partial^2}{\partial\alpha_{iu}^2}\mathcal{L}(\theta^{(m)}) & \frac{\partial^2}{\partial\alpha_{iu}\partial\beta_{iu}}\mathcal{L}(\theta^{(m)}) \\ \frac{\partial^2}{\partial\alpha_{iu}\partial\beta_{iu}}\mathcal{L}(\theta^{(m)}) & \frac{\partial^2}{\partial\beta_{iu}^2}\mathcal{L}(\theta^{(m)}) \end{bmatrix}^{-1} \\ \nabla_{\theta_{1i}}\mathcal{L}(\theta^{(m)}) &= \begin{bmatrix} \frac{\partial}{\partial\alpha_{iu}}\mathcal{L}(\theta^{(m)}) \\ \frac{\partial}{\partial\beta_{iu}}\mathcal{L}(\theta^{(m)}) \end{bmatrix} \end{aligned}$$

Parameter estimation in GMM

The parameters of the GMM part, μ_{iv}'s and σ_v^2's ($\{v = 1, \ldots, p_2\}$), in BGBMM can be estimated by the standard EM algorithm of GMM with diagonal covariance matrix. The derivations are shown below [16], which result in Equations 3.16 and 3.17 for parameter updates.

$$
\begin{aligned}
\because Q_2(\theta_2) &= \sum_{j=1}^{n}\sum_{v=1}^{p_2}\sum_{i=1}^{g}\tau_{ji}^{(m)}\left(-\frac{1}{2}\log\left(2\pi\sigma_v^2\right)-\frac{1}{2\sigma_v^2}(y_{jv}-\mu_{iv})^2\right)\\
\nabla_{\theta_2}\mathcal{L}(\theta) &= \nabla_{\theta_2}Q_2(\theta_2)
\end{aligned}
$$

$$
\begin{aligned}
\therefore \frac{\partial}{\partial\mu_{iv}}\mathcal{L}(\theta) &= \sum_{j=1}^{n}\tau_{ji}^{(m)}\left(\frac{1}{\sigma_v^2}(y_{jv}-\mu_{iv})\right)\\
&= 0\\
\frac{\partial}{\partial\sigma_v^2}\mathcal{L}(\theta) &= \sum_{j=1}^{n}\sum_{i=1}^{g}\tau_{ji}^{(m)}\left(-\frac{1}{2\sigma_v^2}+\frac{1}{2(\sigma_v^2)^2}(y_{jv}-\mu_{iv})^2\right)\\
&= 0\\
\hat{\mu}_{iv}^{(m+1)} &= \sum_{j=1}^{n}\tau_{ji}^{(m)}y_{jv}/\sum_{j=1}^{n}\tau_{ji}^{(m)} \qquad (3.16)\\
\hat{\sigma}_v^{2,(m+1)} &= \sum_{j=1}^{n}\sum_{i=1}^{g}\tau_{ji}^{(m)}(y_{jv}-\mu_{iv}^{(m)})^2/n \qquad (3.17)
\end{aligned}
$$

Parameter estimation in BerMM

The parameters of BerMM, q_{iw}'s ($\{w = 1, \ldots, p_3\}$), are derived below, whose update formula is given by Equation 3.18.

$$
\begin{aligned}
\because Q_3(\theta_3) &= \sum_{j=1}^{n}\sum_{w=1}^{p_3}\sum_{i=1}^{g} \tau_{ji}^{(m)} \left(log(q_{iw}^{z_{jw}}) + log((1-q_{iw})^{(1-z_{jw})}) \right) \\
\nabla_{\theta_3}\mathcal{L}(\theta) &= \nabla_{\theta_3} Q_3(\theta_3) \\
\therefore \frac{\partial}{\partial q_{iw}}\mathcal{L}(\theta) &= \sum_{j=1}^{n} \tau_{ji}^{(m)} \left(\frac{z_{jw} - q_{iw}}{q_{iw}(1-q_{iw})} \right) \\
&= 0 \\
\sum_{j=1}^{n} \tau_{ji}^{(m)} \hat{\mathbf{q}}_i^{(m+1)} &= \sum_{j=1}^{n} \tau_{ji}^{(m)} \mathbf{z}_j \\
\hat{\mathbf{q}}_i^{(m+1)} &= \frac{\sum_{j=1}^{n} \tau_{ji}^{(m)} \mathbf{z}_j}{\sum_{j=1}^{n} \tau_{ji}^{(m)}}
\end{aligned} \tag{3.18}
$$

Parameter estimation of the prior probability

The prior probability, π, can be computed as Equation 3.19 [99],

$$
\begin{aligned}
\because Q_4(\pi) &= \sum_{j=1}^{n}\sum_{i=1}^{g} \tau_{ji} \log(\pi_i) \\
\nabla_{\pi}\mathcal{L}(\theta) &= \nabla_{\pi} Q_4(\pi) - \lambda \mathbf{1} \\
\therefore \frac{\partial}{\partial \pi_i}\mathcal{L}(\theta) &= \sum_{j=1}^{n} \tau_{ji}^{(m)} \frac{1}{\pi_i} - \lambda \\
\hat{\pi}_i^{(m+1)} &= \sum_{j=1}^{n} \tau_{ji}^{(m)} / \lambda.
\end{aligned} \tag{3.19}
$$

Since

$$\begin{aligned}\sum_{i=1}^{g}\pi_i &= \sum_{i=1}^{g}\frac{1}{\lambda}\sum_{j=1}^{n}\tau_{ji}\\ &= \frac{1}{\lambda}\sum_{j=1}^{n}\sum_{i=1}^{g}\tau_{ji}\\ &= \frac{1}{\lambda}\sum_{j=1}^{n}1\\ &= \frac{n}{\lambda},\end{aligned}$$

and $\sum_{i=1}^{g}\pi_i = 1$, thus $\lambda = n$. By plugging it into Equation 3.19, Equation 3.20 [99] is obtained to estimate the prior probabilities in the joint mixture model.

$$\hat{\pi}_i^{(m+1)} = \sum_{j=1}^{n}\tau_{ji}^{(m)}/n \tag{3.20}$$

3.3 Model selection

Four well-known approximation-based model selection criteria, i.e., BIC [37; 40], ICL [33], AIC [36; 39], and AIC3 [38; 39] are compared in BGBMM, according to which the best-performing one is chosen. Calculations for the above criteria are defined in Equations 3.21 to 3.24, respectively, where d is the number of free parameters, M is the total amount of the data ($M = \sum_{w=1}^{W} M_w$, M_w is the size of data set w and W is the number of input data sets), and $-2\sum_{j=1}^{n}\sum_{i=1}^{g}\tau_{ji}\log(\tau_{ji})$ is the estimated entropy of the fuzzy classification matrix $C_{ji} = ((\tau_{ji}))$ [33].

$$BIC = -2\log L(\hat{\theta}) + d\log(nM), \tag{3.21}$$

$$\begin{aligned}ICL &= -2\log L(\hat{\theta}) + d\log(nM)\\ &\quad -2\sum_{j=1}^{n}\sum_{i=1}^{g}\tau_{ji}\log(\tau_{ji}),\end{aligned} \tag{3.22}$$

$$AIC = -2\log L(\hat{\theta}) + 2d, \tag{3.23}$$

$$AIC3 = -2\log L(\hat{\theta}) + 3d, \tag{3.24}$$

The number of free parameters d in BGBMM is $d_{BGB} = 2gp_1 + p_2 + p_2g + p_3g + g - 1$ since there are p_1g free α_{iu}'s, p_1g free β_{iu}'s, p_2 free σ_v's, p_2g free μ_{iv}'s, p_3g free q_{iw}'s, and $g - 1$ free π_i's.

3.4 Algorithm implementation

BGBMM is implemented with a fourth generation programming language, i.e., Matlab [100]. In order to avoid the possible local maxima, the algorithm is run multiple times with different initial values for each clustering event. The parameters α_{iu}'s and β_{iu}'s ($u \in \{1, \ldots, p_1\}$) for each dimension of the beta distribution are initialized by method-of-moments so that their means are randomly distributed within the range of $x_{1u}, \ldots, x_{nu}$ and the variances are equal for all clusters (g). The parameters μ_{iv}'s and σ_v^2's in the GMM part are obtained from the randomly initialized fuzzy C-means clustering results. The parameters of the BerMM part, q_{iw}'s, are initialized with random probabilistic values whose summation over i is 0.5. Finally, π_i's are initialized with the uniform probability $1/g$.

Since the Newton method used for parameter estimation in the BMM part may stuck into local minimum, thus, for each clustering event, the EM algorithm runs several times until convergence. The convergence threshold (where the absolute difference of Q is used to monitor the convergence) and the maximum number of iterations are combined by an 'and' operator to surveil the convergence, which are set to 0.0001 and 100, respectively. Further, for each simulation and the real case study, 100 rounds of clustering are run, from which the one with the maximum Q is used for model selection.

The Matlab codes can be found in the appendix.

Chapter 4

Results

This chapter presents the performance test results of BGBMM by comparing it with two other joint mixture models, BGMM [99] and sBGMM [101], using simulated and real data.

BGBMM differs from BGMM in employing one more data source, i.e., Bernoulli distributed data, and distinguishes from sBGMM in utilizing the third information source as one component of the joint mixture model instead of converting it into the model prior.

All the clustering events have reached their convergence according to the statistics stored during each run.

4.1 Performance test with artificial data

Two questions are answered from the simulation tests of BGBMM, i.e., how does adding one component of Bernoulli distribution to the joint clustering framework improve the clustering ('Performance test 1'), and how does the quality of Bernoulli distributed data affect the prediction accuracy ('Performance test 2').

4.1.1 Performance evaluation system

The scoring system, namely the 'E score' and presented in [99], is used to evaluate the simulation accuracy. As described in Equations 4.1 to 4.2, T_j denotes the ground truth clustering membership of data j. R stands for all the possible associating ways between the estimated and the true clusters, where r_i is the label of data belonging to component i predicted by the clus-

tering algorithm and r is chosen from labels $1, 2, \ldots, \max\{\hat{g}, g\}$ ($\hat{g}$ and g are the largest labels in the estimated and ground truth clustering, respectively). Also, e represents the individual score of each gene, E is the average score of all the genes for each repetition, 'E score' of each repetition is the one corresponding to the optimal Q, and the final 'E score' of each scenario is the median of the 10 'E score's (10 times clusterings are done for each data scenario with different seeds using Matlab). This scoring system not only records the clustering accuracy but also reflects the influence of the model selection criterion.

$$
\begin{aligned}
e_j(r) &= \begin{cases} 1 & if \;\; \hat{c}_j = i \;\; and \;\; r_i = T_j \\ 0 & \text{otherwise} \end{cases} \\
E &= \max_{r \in R} \sum_{j=1}^{n} e_j(r)/n \qquad (4.1) \\
R &= \big\{ r = (r_1, \ldots, r_{\hat{g}}) : \forall i \neq j \;\; r_i \neq r_j; \\
&\qquad r_i \in \{1, \ldots, \max\{\hat{g}, g\}\} \big\}. \qquad (4.2)
\end{aligned}
$$

4.1.2 Data

Information from different sources may differ in the underlying data structure since they cover different perspectives of a system [102]. As illustrated in Fig. 4.1, data can be divided into five regions. Particularly, all the data share the same structure in 'Region 1', two out of the three data types agree on the underlying structure in 'Region 2' (excludes Gaussian), 'Region 3' (no beta) and 'Region 4' (except for Bernoulli), and none of the data sources has the same underlying structure in 'Region 5'. In this study, only the data from the first four regions are chosen for the performance test since the ground truth of the fifth scenario is undecidable.

The simulated Gaussian and beta distributed data, which can be in the form of gene expression and protein-DNA binding data in practice, are listed in Table 4.1. The sparsity patterns of the generated Bernoulli distributed data or, e.g., PPIs in reality is shown in Fig. 4.2. For each data source, both data with low and high degrees of noise are considered, and the noise source is also taken into account for Gaussian distributed data. In particular, the small letters 'g' and 'b' at the beginning of the name of each data type stand for the quality of the data source, i.e., 'good' (less noise) and 'bad' (more noise), respectively. The capital letter 'B', 'G' and 'P' represent the data type, i.e., the beta, Gaussian and Bernoulli distributed data. The subfixes 'm' and 'v' of Gaussian distributed data show the noise source, i.e., caused by close means

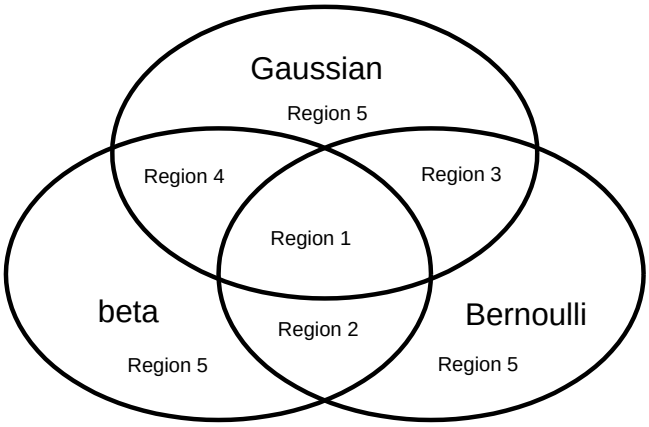

Figure 4.1: Region divisions of input data. In Region 1, the data structure is the same for all data sources. In Region 2, beta and Bernoulli distributed data have the same structure. In Region 3, Gaussian and Bernoulli distributed data share the same structure. In Region 4, beta and Gaussian distributed data have the same structure. In Region 5, no data share the same structure.

('m') and large variances ('v') of the components. For example, 'bG$_m$' means bad Gaussian distributed data whose noise is originated from close means of its components. Further, define the number of genes to be 100 ($n = 100$), and the second dimension of the beta and Gaussian distributed data to be four ($p_1 = p_2 = 4$). All the simulations are repeated 10 times with randomly generated data sets.

4.1.3 Model selection criteria selection

The four model selection criteria, BIC [37; 40], ICL [33], AIC [36; 39] and AIC3 [38; 39] are compared for data from 'Region1' to 'Region4' with both possible underlying ground-truths, i.e., two or three clusters in this study, if the data structure differs among data sources. The corresponding average E scores (i.e., the average of all the possible data combinations under each scenario with each ground-truth assumption) are shown in Table 4.2.

It is seen from Table 4.2 that, AIC and AIC3 perform similarly and outweigh the other crite-

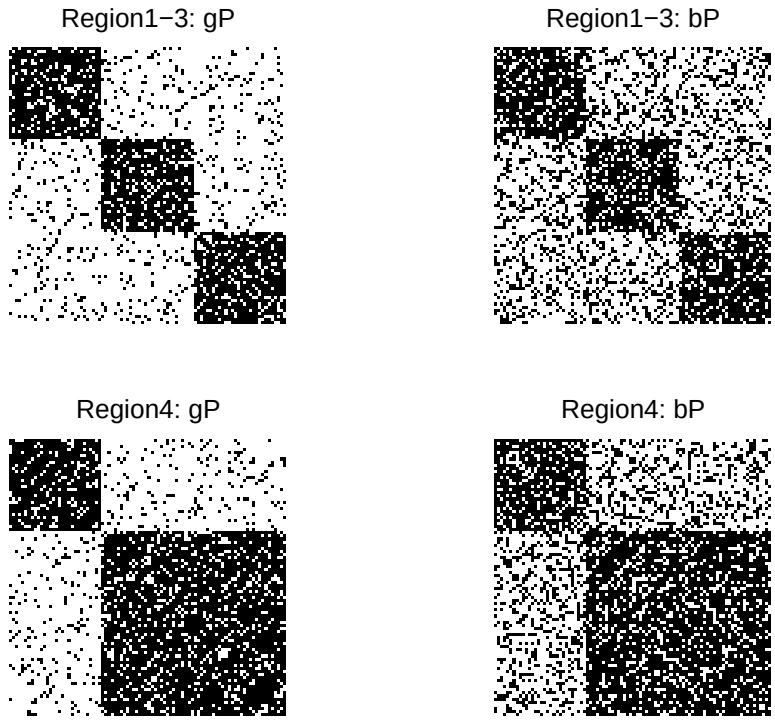

Figure 4.2: Sparsity patterns of the contact matrixes of the artificial PPI data sets. 'gP' and 'bP' stand for PPI data with less and more noises, respectively.

ria in generating higher E scores, especially when the finer grouping (more number of clusters) is assumed to be the ground truth. Thus, AIC and AIC3 are selected as the model selection criteria in BGBMM due to their superior (five out of seven tested scenarios) performance compared with the others.

4.1.4 Performance test

Performance test 1

The performance of BGBMM is compared with BGMM and sBGMM for scenarios corresponding to 'Region 1' to 'Region 4'. The comparison results of BGBMM with BGMM and sBGMM are shown in Fig. 4.3. In cases where the data structure differ, both possible ground-truths are assumed.

The results show that there is no ambiguity in the first scenario, where no discrepancy

exists among different data sources. Further, BGBMM is much more robust than BGMM and sBGMM regarding its higher accuracy and more stable performance.

For 'Region 2' data, where the number of clusters for beta and Bernoulli distributed data are both three, BGBMM clearly outweighs the other models in most cases (see Fig. 4.3 (b)), demonstrating its efficiency in data fusion. However, this superiority does not show for two cases, i.e., whenever the beta distributed data is combined with 'gG+bP', where the number of clusters favors two instead of three (see the extruding bars of BGBMM in Fig. 4.3 (c)). This error is caused by using less noisy but erroneous ('erroneous' refers to not consistent with the dominant data structure or data structure agreed by most data sources) Gaussian distributed data and accurate but highly noisy Bernoulli distributed data. Also, notice that errors caused by the above reason can be reversed in sBGMM when 'gB' is jointly used with 'gG+bP', which, however, is not possible in BGBMM, indicating a stronger role of Bernoulli distributed data in BGBMM compared with that in sBGMM. These results together suggest that, BGBMM can make a more efficient use of information but requires higher data quality.

For 'Region 3' data, where Gaussian and Bernoulli distributed data both have three underlying clusters, the superiority of BGBMM over the other models is obvious as seen in Fig. 4.3 (d).

In 'Region 4', only the Bernoulli distributed data indicates two clusters, because of which the results obtained from BGBMM favor two clusters (see the extruding bars of BGBMM in Fig. 4.3 (g)) when the Gaussian distributed data is noisy. This error, originated from using erroneous Bernoulli distributed data and noisy Gaussian distributed data, is, however, recoverable by accurate and less noisy beta distributed data in sBGMM. Nevertheless, except for the few cases where the error is recovered in sBGMM, BGBMM still clearly shows its superiority (see the extruding bars of BGBMM in Fig. 4.3 (f)). These results again illustrate BGBMM's efficiency in data usage, but on the other hand indicate its high requirement on data quality.

Taken together, BGBMM is highly efficient regarding data usage, which also depends on the data quality. In other words, data of higher quality or importance is better combined as one component of the joint mixture model; otherwise, it is safer to convert the data into the model prior to facilitate clustering.

Performance test 2

Although the superiority of combining Bernoulli distributed data as one component of the joint mixture model is well demonstrated in Fig. 4.3, it is still not known yet how much the joint

performance is affected by the quality of this third data source.

In this simulation, the noisy beta and Gaussian distributed data, i.e., 'bG$_m$+bB' (the worst case according to Fig. 4.3 (a)), are used. The noise level of the Bernoulli distributed data (abbreviated as 'PPI' below) is controlled by $\frac{0.6-\alpha}{0.4+\alpha}$, where α varies from 0.1 to 0.5 with step size 0.1. The simulations are done for 'Region 1' data, and the results are shown in Fig. 4.4.

Fig. 4.4 shows a logistic regression curve between PPI's noise level and BGBMM's performance. It is worth noticing the outstanding E score, i.e., beyond 0.9, when PPI's noise level is below $\frac{0.3}{0.7}$ ($\alpha > 0.3$), and the slight performance improvement at $\alpha = 0.1$, where PPI data is extremely noisy. Recall that this simulation is done with highly noisy beta and Gaussian distributed data.

4.2 Performance test with real data

4.2.1 Gene ontology validation

Gene ontology (GO), which covers annotations on four aspects of genes and their products' properties [103], is used to validate the clustering results of real data. Among the four domains covered by GO, three are shared by all organisms, which are 'molecular function', 'cellular component' and 'biological process' [103]. Specifically, 'molecular function' refers to the biochemical activity of a gene product, 'cellular component' shows the cellular location where a gene product is active, and 'biological process' means the biological objective of a gene or gene product or, in other words, the biological process it is involved in [104–107].

To find the significantly annotated terms by looking at the probabilities that the terms are counted by chance, the hypergeometric probability distribution is used to compute the p-values of the gene enrichment score (called 'p-value' in this thesis for simplicity) for each clustering event (reference Bioinformatics Toolbox 3.1 in Matlab). GO is composed of ontology itself, where a vocabulary of terms and their relations are stored, and annotations, which describe the associations between the terms in the ontology [103–107]. The statistics is computed among the terms appeared in the ontology based on the information stored in the annotations. In particular, Equation 4.3 is used to calculate the p-value of the hypothesis that the ontology term t is associated with genes in cluster i, where the formula of hypergeometric distribution is shown in Equation 4.4. Note in these equations that N is the total number of annotations for

the genes in the data set, M_i is the number of annotations associated with genes in cluster i, n_t is the total number of annotations of term t in ontology, and $k_{t,i}$ is the number of annotations associated with the term t of genes in cluster i. The p-value of genes that form the cluster i is thus the minimum of all the associated annotated terms, i.e., $p_i = \min_{\mathbf{t}}(p_{t,i})$, and the smaller the p-value is, the smaller chance that genes grouped in cluster i is obtained by chance.

$$\begin{aligned} p_{t,i} &= 1 - \sum_{j=0}^{k_{t,i}-1} f(j, N, M_i, n_t) \\ &= 1 - F(k_{t,i} - 1, N, M_i, n_t) \qquad (4.3) \\ f(k_{t,i}, N, M_i, n_t) &= \frac{\binom{M_i}{k_{t,i}}\binom{N-M_i}{n_t-k_{t,i}}}{\binom{N}{n_t}} \qquad (4.4) \end{aligned}$$

The ontology is not static but updated frequently as old terms becoming obsolete with the keep emerging new ones [103–107]. The ontology used for all the clustering events in this study is retrieved at the time the work was conducted.

4.2.2 Data

BGBMM is applied to mouse protein-DNA binding probabilities (predicted from a transcription factor binding site prediction algorithm, ProbTF [62]), gene expression data and PPI data, which are naturally modeled as beta, Gaussian and Bernoulli distributions, respectively.

The protein-DNA binding data contains the probabilities of 266 transcription factors (TF) binding to 20397 genes, computed with mouse-specific position weight matrices from the TRANSFAC database [62]. The gene expression data is composed of 1960 genes measured from 95 conditions [108]. PPI data are taken from six public curated PPI databases, i.e., MINT [87], IntAct [89], DIP [85; 86], BioGRID [90], HPRD [91], and MIPS/MPact [88], and queried through an online integration platform PINA [98].

There are 1502 genes measured in all three data sets, which are used in this study. In particular, the binding probabilities of seven TF genes to the focused genes are used as beta distributed data. The seven TFs are 'E2F6', 'E2f7', 'Foxm1', 'Nfatc1', 'Rest', 'Stat1' and 'Mxd1', which are always grouped together when TF genes are clustered by BGMM. To avoid the possible violence of the independence assumption in the GMM part, gene expression data of

23 conditions, i.e., the midpoints of the time series (as shown in Table 4.3), is used for further analysis. In the PPI data, among 1586 interactions where at least one interactor is within the data set, 221 interactions have both interactors fall in the test set. These interactions are thereby used as the Bernoulli distributed data.

4.2.3 Performance test

To see how stable BGBMM works with the tested real data and whether 100 iterations for each clustering events are enough for convergence, five repetitions are done for each model, i.e., BGBMM, BGMM and sBGMM. The clustering results of the 1502 genes are comparatively stable according to the test, where all five iterations of BGBMM and BGMM generate the same results, respectively, and three tests of sBGMM result in the same clustering. The clustering results (the most frequent result is shown if the results differ among different runs) are compared and evaluated by GO, which are shown in Table 4.4.

It is clear from Table 4.4 that, BGBMM significantly outperforms BGMM and sBGMM from almost all aspects, and sBGMM is better than BGMM except when all GO aspects are simultaneously considered. Moveover, although maybe data dependent, BGBMM tends to be more stable than sBGMM according to the repetitive test (five same results in BGBMM vs. three in sBGMM). These results highlight again the advantage of data fusion given the high-level genomic noise, and also strengthen the efficiency of combining additional data sources into the joint mixture model as model components if the data is of high quality. Notice that PPIs used in this study are confirmed from six PPI databases.

4.2.4 Exploration of the clustering results

The best clustering results predicted from BGBMM, i.e., the one chosen by AIC and AIC3, are further explored. As shown in Table 4.5, the genes are clustered into two groups, and the p-values of group 2 is much lower than that of group 1 regarding all the considered aspects. It is found that all the 32 TF-genes of the test data set are within group 2. Also recall that the gene expression data is obtained with Toll-like receptor (TLR)-stimulated macrophage activation (see Table 4.3). Thus, it is plausible to deduce that genes within group 2 are involved in the TLR-stimulated macrophage program given that BGBMM considers information from all three levels, i.e., gene, protein, and gene products' physical binding levels.

Data set 1		cluster 1				cluster 2				cluster 3			
gB	alpha	20	5	3	30	20	25	30	35	2	15	33	4
	beta	2	15	33	4	20	25	30	35	20	5	3	30
bB	alpha	33	30	22	20	30	27	20	18	27	24	18	16
	beta	30	33	20	22	27	30	18	20	24	27	16	18
gG	mean	5	-8	20	15	10	1	-20	0	-10	8	5	15
	variance	1	2	3	2.5	1	2	3	2.5	1	2	3	2.5
bG_m	mean	3	15	5	11	2	13	6	9	1	14	7	10
	variance	1	2	3	2.5	1	2	3	2.5	1	2	3	2.5
bG_v	mean	5	-8	20	15	10	1	-20	0	-10	8	5	15
	variance	10	20	30	25	10	20	30	25	10	20	30	25
Data set 2		cluster 1				cluster 2				cluster 3			
gB	alpha	20	5	3	30	20	25	30	35	2	15	33	4
	beta	2	15	33	4	20	25	30	35	20	5	3	30
bB	alpha	33	30	22	20	30	27	20	18	27	24	18	16
	beta	30	33	20	22	27	30	18	20	24	27	16	18
gG	mean	10		1		-20		0		-10	8	5	15
	variance	1		2		3		2.5		1	2	3	2.5
bG_m	mean	2		13		6		9		1	14	7	10
	variance	1		2		3		2.5		1	2	3	2.5
bG_v	mean	10		1		-20		0		-10	8	5	15
	variance	10		20		30		25		10	20	30	25
Data set 3		cluster 1				cluster 2				cluster 3			
gB	alpha	20		5		3		30		2	15	33	4
	beta	2		15		33		4		20	5	3	30
bB	alpha	30		27		20		18		27	24	18	16
	beta	27		30		18		20		24	27	16	18
gG	mean	5	-8	20	15	10	1	-20	0	-10	8	5	15
	variance	1	2	3	2.5	1	2	3	2.5	1	2	3	2.5
bG_m	mean	3	15	5	11	2	13	6	9	1	14	7	10
	variance	1	2	3	2.5	1	2	3	2.5	1	2	3	2.5
bG_v	mean	5	-8	20	15	10	1	-20	0	-10	8	5	15
	variance	10	20	30	25	10	20	30	25	10	20	30	25

Note: 'gB' and 'bB' each stands for 'beta' distributed data that are of 'good' and 'bad' quality respectively; 'gG', 'bG_m' and 'bG_v' each represents 'Gaussian' distributed data that are of 'good' quality and 'bad' quality with respect to close means and large variances respectively; '||' separate the parameters of different clusters, and '|' separate the parameters of different dimensions (2^{nd} dimension) within the same cluster. Parameters of beta and Gaussian distributions are the same for data from 'Region 1' and 'Region 4'.

Table 4.1: Parameters of beta and Gaussian distributed data.

Model	Region	N	AIC	AIC3	BIC	ICL	Best
BGBMM	R1	3	0.9893	0.9896	0.9258	0.9258	AIC3
	R2	3	0.9183	0.9184	0.8228	0.8228	AIC3
	R3	3	0.9750	0.9755	0.8981	0.8981	AIC3
	R4	3	0.8274	0.8274	0.7272	0.7272	AIC|AIC3
	R2	2	0.7341	0.7343	0.7629	0.7629	BIC|ICL
	R3	2	0.6836	0.6836	0.6804	0.6804	AIC|AIC3
	R4	2	0.8253	0.8251	0.8916	0.8916	BIC|ICL
sBGMM	R1	3	0.8534	0.8646	0.8694	0.8707	ICL
	R2	3	0.7522	0.7633	0.7836	0.7821	BIC
	R3	3	0.7517	0.7660	0.7816	0.7812	BIC
	R4	3	0.8500	0.8604	0.8666	0.8653	BIC
	R2	2	0.6299	0.6609	0.7348	0.7443	ICL
	R3	2	0.6398	0.6613	0.7175	0.7227	ICL
	R4	2	0.6121	0.6231	0.6495	0.6505	ICL
BGMM	R1	3	0.8171	0.7989	0.8007	0.8038	AIC
	R2	3	0.7540	0.7235	0.7460	0.7460	AIC
	R3	3	0.7289	0.6810	0.7450	0.7333	BIC
	R4	3	0.8080	0.7978	0.8136	0.8069	BIC
	R2	2	0.7201	0.7005	0.7342	0.7342	BIC|ICL
	R3	2	0.7185	0.7146	0.7229	0.7181	BIC
	R4	2	0.6519	0.6372	0.6450	0.6480	AIC

Note: Values shown here are the averages of E scores over all the possible data combinations in each scenario ('gB+gG+gP', 'gB+gG+bP', bB+gG+gP', 'bB+gG+bP', 'gB+bG$_m$+gP', 'gB+bG$_m$+bP', 'bB+bG$_m$+gP', 'bB+bG$_m$+bP', 'gB+bG$_v$+gP', 'gB+bG$_v$+bP', 'bB+bG$_v$+gP', 'bB+bG$_v$+bP') selected by each criterion in each model. 'ICL' is short for 'ICL-BIC'. 'N' stands for ground-truth of the number of underlying clusters. 'Best' is the best criterion with respect to highest average E scores and used in drawing Fig. 4.3. All values are rounded to four decimal points.

Table 4.2: Comparison of different model selection criteria in BGBMM, sBGMM and BGMM.

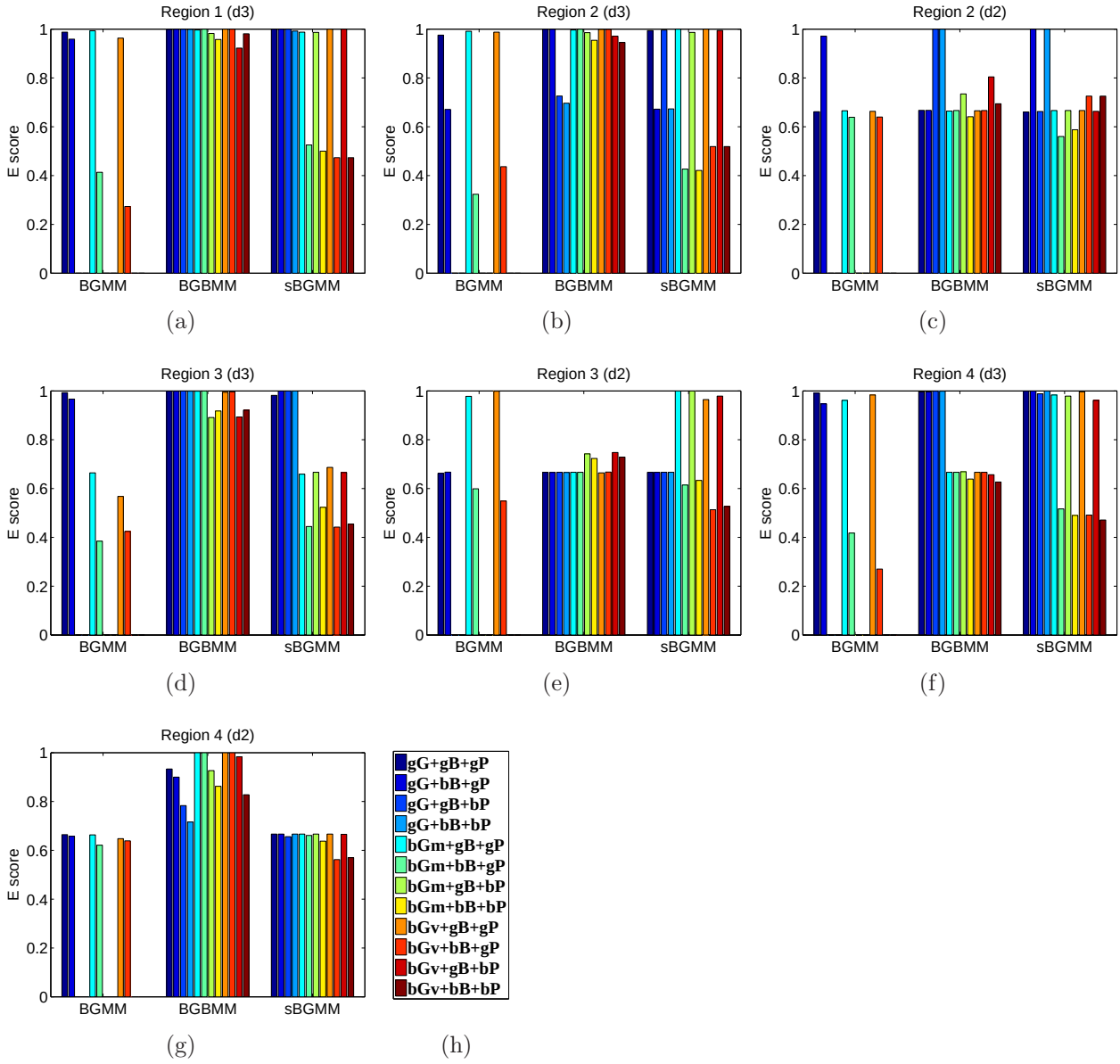

Figure 4.3: Performance comparison results of BGBMM with BGMM and sBGMM for (a) 'Region 1' data, (b) 'Region 2' data with three clusters as the ground-truth, (c) 'Region 2' data with two clusters as the ground-truth, (d) 'Region 3' data with three clusters as the ground-truth, (e) 'Region 3' data with two clusters as the ground-truth, (f) 'Region 4' data with three clusters as the ground-truth, and (g) 'Region 4' data with two clusters as the ground-truth. The legend is shown in sub-figure (h). All the criteria are chosen based on average accuracy. '1', '2', '3' in the x-axis represent 'BGMM', 'BGBMM' and 'sBGMM', respectively. The y-axis shows the E scores. Missing bars in BGMM are the combinations that do not exist in this model.

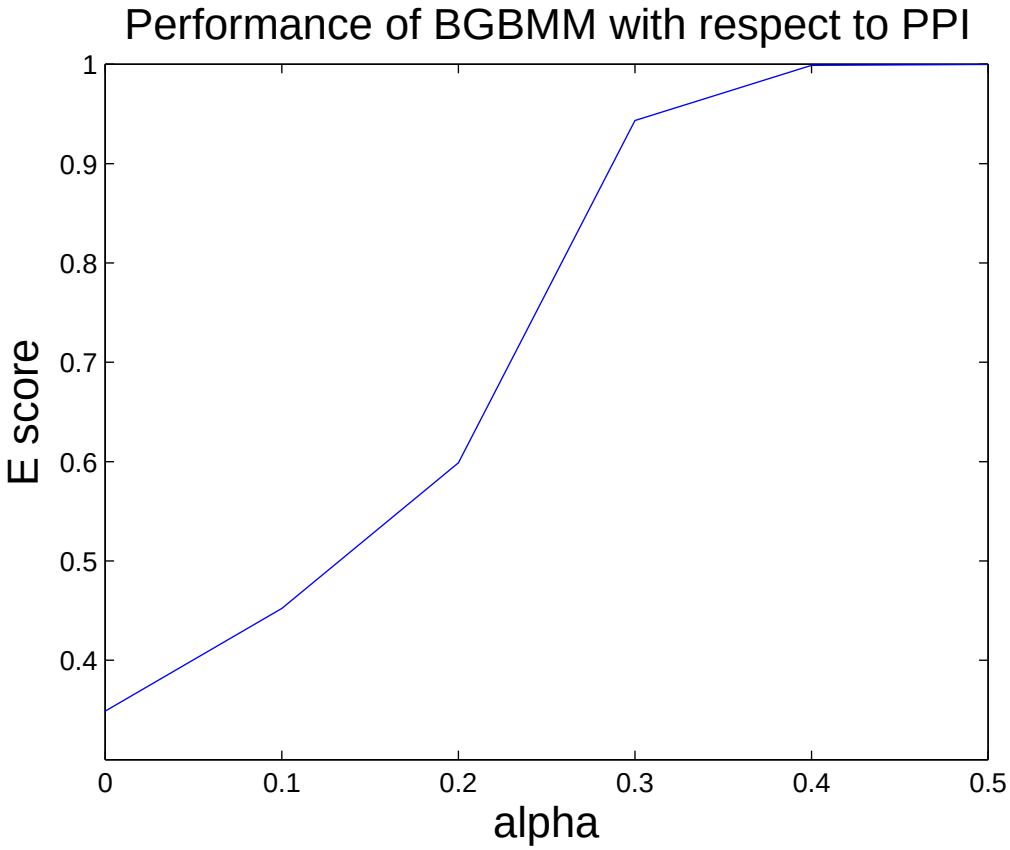

Figure 4.4: The performance of BGBMM with respect to the quality of the Bernoulli distributed data. The x-axis represents the noise level of Bernoulli distributed data. The y-axis shows the E score. E score at point '0' is the score for BGMM when Bernoulli distributed data is not taken into account.

Point	Treatment	Time (min)
1	Atf_3^-	0
2	C_pG+Atf_3^-	120
3	LPS+Atf_3^-	240
4	Pam_2CSK_4+Atf_3^-	120
5	poly I:C+Atf_3^-	120
6	$Crem^-$	0
7	LPS+$Crem^-$	240
8	poly I:C+$Crem^-$	360
9	$Myd88^-$	0
10	LPS+$Myd88^-$	60
11	Pam_3CSK_4+$Myd88^-$	60
12	poly I:C+$Myd88^-$	60
13	$TicamI^-$	0
14	LPS+$TicamI^-$	120
15	LPS+Pam_2CSK_4+$TicamI^-$	120
16	no	0
17	C_pG	60
18	LPS	360
19	Pam_2CSK_4	80
20	Pam_3CSK_4	240
21	Pam_3CSK_4+poly I:C	60
22	poly I:C	120
23	R848	120

Note: 'Point' refers to the labels of the x axis; '-' means the mutant strain that does not have the particular gene; time point chosen for the treatment is shown in the column 'Time'. Treatments after point 16 were all applied to the wild type.

Table 4.3: Treatments of the gene expression data

		All		F		C		P		
Moldel	Crit	M1	M2	M1	M2	M1	M2	M1	M2	N
BG	1∼4	0.1077	0.0255	0.2191	0.2192	0.2632	0.3067	0.2368	0.2603	4
sBG	1∼2	0.1278	0.1238	0.2019	0.2051	0.2292	0.2135	0.2126	0.1914	14
	3∼4	0.1436	0.1473	0.2142	0.2180	0.2372	0.2285	0.2241	0.2028	13
BGB	1∼2	0.0898	0.0388	0.1849	0.0727	0.2257	0.2086	0.2085	0.1670	5
	3∼4	0.0484	0.0484	0.1128	0.1128	0.1700	0.1700	0.1499	0.1499	2

Note: 'F', 'C', 'P' each represents one aspect of Gene Ontology, i.e., 'molecular function', 'cellular component' and 'biological process', respectively. 'All' means all three aspects are included. 'M1' and 'M2' stand for the mean and median of the p-values across all the clusters, respectively. 'Model' and 'Crit' represent the model and model selection criteria, respectively. 'BG', 'sBG', 'BGB' stand for BGMM, sBGMM and BGBMM, respectively. '1' to '4' each represents model selection criterion BIC, ICL, AIC, AIC3 respectively. 'N' means the number of clusters generated by each model. All fractions are rounded to four decimal points.

Table 4.4: Performance test results of BGBMM with real data.

Model	Group	All	F	C	P	N
BGB	1	0.0957	0.2148	0.3130	0.2910	633
	2	0.0011	0.0108	0.0270	0.0087	869

Note: 'F', 'C', 'P' represent the three aspects of gene ontology, and 'All' means all three aspects are included. '1' and '2' represent the group number of the clustering result. 'N' means the number of genes involved in each cluster. All fractions are rounded to four decimal points.

Table 4.5: Clustering results obtained by BGBMM and selected with BIC and ICL.

Chapter 5

Discussion

5.1 Summary and conclusion

This thesis proposes a joint mixture model, namely BGBMM, to cluster genes from protein-DNA binding probabilities, gene expression data and PPIs, assuming that these information sources are of beta, Gaussian and Bernoulli distribution, respectively.

BGBMM distinguishes itself from BGMM [99] in its simultaneous utilization of information from three heterogeneous data sources, leading to the feasibility of clustering genes from gene, protein, and gene products' physical binding levels. Thus, genes clustered together by BGBMM are more likely to be involved in the same regulatory network, assuming that one gene corresponds to one protein, and genes within the same regulatory module are more likely to interact with each other and share more similar expression profiles than those from different clusters.

Although both employing three information sources, BGBMM differs from sBGMM [101] in combining the third information source as a model component instead of converting it into the prior (in sBGMM the third data source is used to build the prior). The results show that it is more efficient to integrate data as a model component which, however, renders the results more sensitive to data quality than converting the information into the model prior. Thus, depending on the problem, BGBMM is more efficient if the Bernoulli distributed data is complete, less noisy or of more importance; otherwise, sBGMM is a safer choice.

In this thesis, beta, Gaussian and Bernoulli distributions are used to model protein-DNA binding probabilities, gene expression and PPIs. However, BGBMM is not limited to the specific problems studied here. Many other data sources can be modeled as these distributions. For

example, any data that is of probabilistic form such as correlations [33] can be modeled as beta distribution, and PPI data is not limited to be symmetric but can be of any binary form such as literature-derived PPI information or promoter states. Also, BGBMM can be further extended to incorporate data of any other parametric distributions, in principle, to solve problems even beyond this field.

5.2 Limitation and future direction

The diagonal covariance matrix is used in the GMM part of BGBMM to reduce the number of parameters to be estimated from $(p_2^2 + p_2)/2$ to p_2, which is a significant reduction when p_2 is huge. However, this approximation assumes no correlations among Gaussian distributed data, which may be violated if time series is used and when the data correlations need to be preserved. To solve this problem, other modeling strategies could be adopted. For example, one can develop similar estimation algorithms for a covariance model where off-diagonal elements are assumed to be constant.

Also, besides the current modeling strategy, other approaches such as hierarchical Bayes model [109] may also be used to cluster genes from multiple data sources, which may even work better when different data sources disagree on the underlying ground-truth, since it models the ensemble data structure while allowing each data source having its individual underlying structure.

Bibliography

[1] D X Jiang, C Tang, and A D Zhang. Cluster analysis for gene expression data: a survey. *IEEE Transactions on knowledge and data engineering*, 16(11):1370–1386, 2004.

[2] T R Hvidsten, J komorowski, A K Sandvik, and A Laegreid. Predicting gene function from gene expressions and ontologies. In *Pacific Symposium on Biocomputing*, pages 299–310. World Scientific, 2001.

[3] M Kuramochi and O karypis. Gene classification using expression profiles: a feasibility study. In *Proceedings of the 2nd IEEE International Symposium on Bioinformatics and Bioengineering. Bethesda, Maryland, USA*, 2001.

[4] X Zhou, M C Kao, and W H Wong. Transitive functional annotation by shortest-path analysis of gene expression data. In *Proceedings of the National Academy of Sciences of the United States of America*, pages 12783–12788. National Academy of Sciences, 2002.

[5] A Lagreid, T R Hvidsten, H Midelfart, J Komorowski, and A K Sandvik. Predicting gene ontology biological process from temporal gene expression patterns. *Genome Research*, 13(5):965–979, 2003.

[6] K Tu, H Yu, and Y X Li. Combing gene expression profiles and protein-protein interaction data to infer gene functions. *Journal of Biotechnology*, 124:475–485, 2006.

[7] E Segal, H Wang, and D Koller. Discovering molecular pathways from protein interaction and gene expression data. *Bioinformatics*, 19:i264–i272, 2003.

[8] A Vazquez, A Flammini, A Maritan, and A Vespignani. Global protein function prediction from protein-protein interaction networks. *Nature Biotechnology*, 21:697–700, 2003.

[9] B Lewin. *Genes VIII*. Pearson Education (US), Upper Saddle River, 2004.

[10] X Xiao, E R Dow, R Eberhart, Z B Miled, and R J Oppelt. Gene clustering using self-organizing maps and particle swarm optimization. In *Proceedings of the 17th International Parallel and Distributed Processing Symposium (IPDPS'03)*, page 154.2. IEEE Computer Society, 2003.

[11] G M Artmann and S Chien. *Bioengineering in Cell and Tissue Research*. Springer-Verlag Berlin Heidelberg, Germany, 2008.

[12] J H Ward. Hierarchical grouping to optimize an objective function. *Journal of the American Statistical Association*, 58(301):234–244, 1963.

[13] J MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297. University of California Press, 1967.

[14] X Xu, M Ester, H P Kriegel, and J Sander. A distribution-based clustering algorithm for mining in large spatial databases. In *Proceedings of the 14th International Conference on Data Engineering (ICDE'98)*, pages 324–331. IEEE Computer Society, 1998.

[15] L J Heyer, S Kruglyak, and S Yooseph. Exploring expression data: Identification and analysis of coexpressed genes. *Genome Research*, 9(11):1106–1115, 1999.

[16] G J McLachlan and D Peel. *Finite mixture models*. John Wiley & Sons, New York, USA, 2000.

[17] M Medvedovic and S Sivaganesan. Bayesian infinite mixture model based clustering of gene expression profiles. *Bioinformatics*, 18(9):1194–1206, 2002.

[18] B Z Qiu, X Z Zhang, and J Y Shen. Grid-based clustering algorithm for multi-density. In *Proceedings of the Fourth International Conference on Machine Learning and Cybernetics*, pages 18–21. Springer, 2005.

[19] K Ovaska, M Laakso, and S Hautaniemi. Fast gene ontology based clustering for microarray experiments. *BioData Mining*, 1(1):11, 2008.

[20] C Fraley and A E Raftery. Model-based clustering, discriminant analysis, and density estimation. *Journal of the American Statistical Association*, 97:611–631, 2002.

[21] S C Johnson. Hierarchical clustering schemes. *Psychometrika*, 32(3):241–254, 1967.

[22] T Hastie, R Tibshirani, and J Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer Science + Business Media, New York, USA, 2009.

[23] D W Mount. *Bioinformatics: sequence and genome analysis (2nd edition)*. John Inglis, New York, 2004.

[24] K McGarigal, S Cushman, and S G Stafford. *Multivariate statistics for wildlife and ecology research*. Springer-Verlag, New York, 2000.

[25] H C Causton, J Quackenbush, and A Brazma. *Microarray/gene expressions data analysis: a beginner's guide*. Blackwell Science, UK, 2003.

[26] D Russell. *The principles of computer networking*. Cambridge University Press, Cambridge, UK, 1989.

[27] M Steinbach, G Karypis, and V Kumar. A comparison of document clustering techniques. In *KDD Workshop on Text Mining*, 2000.

[28] D Pelleg and A Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 727–734. Morgan Kaufmann, 2000.

[29] P Melin and O Castillo. *Hybrid intelligent systems for pattern recognition using soft computing: an evolutionary approach for neural networks and fuzzy systems*. Springer-Verlag, Berlin, Germany, 2005.

[30] L Kaufman and P J Rousseeuw. *Finding Groups in Data: An introduction to Cluster Analysis*. Wiley, New York, 1990.

[31] S Vaithyanathan and B Dom. Model-based hierarchical clustering.

[32] S Zhong and J Ghosh. A unified framework for model-based clustering. *Journal of Machine Learning Research*, 4:1001–1037, 2003.

[33] Y Ji, C Wu, P Liu, J Wang, and R K Coombes. Applications of beta-mixture models in bioinformatics. *Bioinformatics*, 21(9):2118–2122, 2005.

[34] A P Dempster, N M Laird, and D B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39(1):1–38, 1977.

[35] P Smyth. Model selection for probabilistic clustering using cross-validated likelihood. *Statistics and Computing*, 9(1):63–72, 2000.

[36] H Akaike. A new look at the statistical identification model. *IEEE Transactions on Automatic Control*, 19:716–723, 1974.

[37] G Schwarz. Estimating the dimension of a model. *The Annals of Statistics*, 6(2):461–464, 1978.

[38] H Bozdogan. Model selection and akaike information criterion (AIC): The general theory and its analytic extensions. *Psychometrika*, 52(3):345–370, 1987.

[39] C Biernacki and G Govaert. Choosing models in model-based clustering and discriminant analysis. *Journal of Statistical Computarion and Simularion*, 64(1):49–71, 1999.

[40] W Pan. Incorporating gene functions as priors in model-based clustering of microarray gene expression data. *Bioinformatics*, 22(7):795–801, 2006.

[41] G J McLachlan and D Peel. *Computing Science and Statistics*. Interface Foundation of North America, Fairfax Station, Virginia, 1997.

[42] G J McLachlan. On bootstrapping the likelihood ratio test statistic for the number of components in a normal mixture. *Applied Statistics*, 36(3):318–324, 1987.

[43] D M Chickering and D Heckerman. Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, 29(2/3):181–244, 1997.

[44] K Kim, S Zhang, K Jiang, L Cai, I B Lee, L J Feldman, and et al. Measuring similarities between gene expression profiles through new data transformations. *BMC Bioinformatics*, 8(29):1–14, 2007.

[45] Hestilow T J and Huang Y. Clustering of gene expression data based on shape similarity. *EURASIP Journal on Bioinformatics and Systems Biology*, 2009:195712, 2009.

[46] Z Bar-Joseph, G K Gerber, D K Gifford, T S Jaakkola, and I Simon. Continuous representations of time-series gene expression data. *Journal of Computational Biology*, 10(3-4):341–356, 2003.

[47] J Monod and F Jacob. Clustering of unevenly sampled gene expression time-series data. *Fuzzy Sets and Systems*, 152(1):49–66, 2005.

[48] P Ma, C I Castillo-Davis, W Zhong, and J S Liu. A data-driven clustering method for time course gene expression data. *Nucleic Acids Research*, 34(4):1261–1269, 2006.

[49] Y Yuan and C-T Li. Unsupervised clustering of gene expression time series with conditional random fields. In *Proceedings of the Inaugural IEEE International Conference on Digital EcoSystems and Technologies (DEST'07)*, pages 571–576. IEEE Computer Society, 2007.

[50] L Rueda, A Bari, and A Ngom. *Transactions on Computational System Biology X*. Springer-Verlag, Berlin, Germany, 2008.

[51] R Autio. *Computational Methods for High-Throughput Data Analysis in Cancer Research*. Tampere University of Technology, Doctoral Thesis, 2008.

[52] H C Causton, J Quackenbush, and A Brazma. *Handbook of Statistical Genetics (3rd edition)*. John Wiley & Sons, England, 2007.

[53] G K Smyth, Y H Yang, and T Speed. *Functional Genomics: Methods and Protocols*. Humana Press, Totowa, NJ, 2002.

[54] B Alberts, A Johnson, J Lewis, M Raff, K Roberts, and P Walter. *Molecular Biology of the Cell (Fifth edition)*. Garland Science, New York, 2008.

[55] B Ren, F Robert, J J Wyrick, O Aparicio, E G Jennings, I Simon, and et al. Genome-wide locatin and function of DNA binding proteins. *Science*, 290(5500):2306–2309, 2000.

[56] V R Iyer, C E Horak, C S Scafe, D Botstein, M Snyder, and P O Brown. Genomic binding sites of the yeast cell-cycle transcription factors SBF and MBF. *Nature*, 409(6819):533–538, 2001.

[57] D S Johnson, A Mortazavi, R M Myers, and B Wold. Genome-wide mapping of *in vivo* protein-DNA interactions. *Science*, 316(5830):1497–1502, 2007.

[58] G Robertson, M Hirst, M Bainbridge, M Bilenky, Y Zhao, T Zeng, and et al. Genome-wide profiles of STAT1 DNA association using chromatin immunoprecipitation and massively parallel sequencing. *Nature Methods*, 4(8):651–657, 2007.

[59] J P Peter. ChIP-seq: advantages and challenges of a maturing technology. *Nature Reviews Genetics*, 10(10):669–680, 2009.

[60] J M Claverie and S Audic. The statistical significance of nucleotide position-weight matrix matches. *Bioinformatics*, 12(5):431–439, 1996.

[61] Y Barash, G Elidan, T Kaplan, and N Friedman. CIS: compound importance sampling method for protein-DNA binding site p-value estimation. *Bioinformatics*, 21(5):596–600, 2005.

[62] H Lähdesmäki, A G Rust, and I Shmulevich. Probabilistic inference of transcription factor binding from multiple data sources. *PLoS ONE*, 3(3):e1820, 2008.

[63] M Suzuki and N Yagi. DNA recognition code of transcription factors in the helix-turn-helix, probe helix, hormone receptor, and zinc finger families. *Proceedings of the National Academy of Sciences*, 91(26):12357–12361, 1994.

[64] C O Pabo and L Nekludova. Geometric analysis and comparison of protein-DNA interfaces: Why is there no simple code for recognition? *Journal of Molecular Biology*, 301(3):597–624, 2000.

[65] N M Luscombe, R A Laskowski, and J M Thornton. Amino acid-base interactions: a three-dimensional analysis of protein-DNA interactions at an atomic level. *Nucleic Acids Research*, 29(13):2860–2874, 2001.

[66] R G Endres, T C Schulthess, and N S Wingreen. Toward an atomistic model for predicting transcription factor binding sites. *Proteins*, 57(2):262–268, 2004.

[67] G Paillard and R Lavery. Analyzing protein-DNA recognition mechanism. *Structure*, 12(1):113–122, 2004.

[68] G Paillard, G Deremble, and R Lavery. Looking into DNA recognition: zinc finger binding specificity. *Nucleic Acids Research*, 32(22):6673–6682, 2004.

[69] A V Morozov, J J Havranek, D Baker, and E D Siggia. Protein-DNA binding specificity predictions with structural models. *Nucleic Acids Research*, 33(18):5781–5798, 2005.

[70] K D MacIsaac and E Fraenkel. Practical strategies for discovering regulatory DNA sequence motifs. *PLoS Computational Biology*, 2(4):e36, 2006.

[71] A Beyer, C Workman, J Hollunder, D Radke, U Möller, T Wilhelm, and et al. Integrated assessment and prediction of transcription factor binding. *PLoS Computational Biology*, 2(6):e70, 2006.

[72] O Aparicio, J V Geisberg, and S Kevin. *Chromatin immunoprecipitation for determining the association of proteins with specific genomic sequences in vivo. In* Current Protocols in Cell Biology. California, USA, 2004.

[73] R Jothi, S Cuddapah, A Barski, K Cui, and K Zhao. Genome-wide identification of *in vivo* protein-DNA binding sites from ChIP-Seq data. *Nucleic Acids Research*, 36(16):5221–5231, 2008.

[74] N C Seeman, J M Rosenberg, and A Rich. Sequence-specific recognition of double helical nucleic acids by proteins. *Proceedings of the National Academy of Sciences*, 73(3):804–808, 1976.

[75] S Fields and O Song. A novel genetic system to detect protein-protein interactions. *Nature*, 340(6230):245–246, 1989.

[76] A Panchenko and T Przytycka. *Protein-protein interactions and networks: identification, computer, analysis, and prediction.* Springer-Verlag, London, 2008.

[77] P Uetz, L Giot, G Cagney, T A Mansfield, R S Judson, J R Knight, and et al. A comprehensive analysis of protein-protein interactions in *Saccharomyces cerevisiae*. *Nature*, 403(6770):623–627, 2000.

[78] T Ito, T Chiba, R Ozawa, M Yoshida, M Hattori, and Y Sakaki. A comprehensive two-hybrid analysis to explore the yeast protein interactome. *Proceedings of the National Academy of Sciences*, 98(8):4569–4574, Apr. 2001.

[79] P Uetz. Two-hybrid arrays. *Current Opinion in Chemical Biology*, 6(1):57–62, 2002.

[80] A Kumar and M Snyder. Protein complexes take the bait. *Nature*, 415(6868):123–124, 2002.

[81] O D Sparkman. *Mass spectrometry desk reference (2nd edition)*. Global View Publishing, 1, 2006.

[82] Y Ho, A Gruhler, A Heilbut, G D Bader, L Moore, S L Adams, and et al. Systematic identification of protein complexes in *Saccharomyces cerevisiae* by mass spectrometry. *Nature*, 415(6868):123–124, 2002.

[83] N J Krogan, W T Peng, G Cagney, M D Robinson, R Haw, G Zhong, and et al. High-definition macromolecular composition of yeast RNA-processing complexes. *Molecular Cell*, 13(2):225–239, 2004.

[84] A C Gavin, P Aloy, P Grandi, R Krause, M Boesche, M Marzioch, and et al. Proteome survey reveals modularity of the yeast cell machinery. *Nature*, 440(7084):637–643, 2006.

[85] I Xenarios, D W Rice, L Salwinski, M K Baron, E M Marcotte, and D Eisenberg. Dip: the database of interacting proteins. *Nucleic Acids Research*, 28(1):289–291, 2000.

[86] L Salwinski, C S Miller, A J Smith, F K Pettit, J U Bowie, and D Eisenberg. The database of interacting proteins: 2004 update. *Nucleic Acids Research*, 32(Database issue), 2004.

[87] A Chatraryamontri, A Ceol, L M Palazzi, G Nardelli, M V Schneider, L Castagnoli, and et al. MINT: the Molecular INTeraction database. *Nucleic Acids Research*, 35(Database issue):D572–D574, 2006.

[88] U Güldener, M Münsterkötter, M Oesterheld, P Pagel, A Ruepp, H W Mewes, and et al. MPact: the MIPS protein interaction resource on yeast. *Nucleic Acids Research*, 34(Database issue):D436–D441, 2006.

[89] S Kerrien, Y Alam-Faruque, B Aranda, I Bancarz, A Bridge, C Derow, and et al. IntAct-open source resource for molecular interaction data. *Nuleic Acids Research*, 35(Database issue):D561–D565, 2006.

[90] D Fiedler, H Braberg, M Mehta, G Chechik, G Cagney, P Mukherjee, and et al. Functional organization of the *S. cerevisiae* phosphorylation network. *Cell*, 136(5):952–963, 2009.

[91] T S K Prasad, R Goel, K Kandasamy, S Keerthikumar, S Kumar, S Mathivanan, and et al. Human protein reference database-2009 update. *Nuleic Acids Research*, 37(Database issue):D767–D772, 2009.

[92] R Mrowka, A Patzak, and H Herzel. Is there a bias in proteome research? *Genome Research*, 11(12):1971–1973, 2001.

[93] A M Edwards, B Kus, R Jansen, D Greenbaum, J Greenblatt, and M Gerstein. Bridging structural biology and genomics: assessing protein interaction data with known complexes. *Trends in Genetics*, 18(10):529–536, 2002.

[94] C von Mering, R Krause, B Snel, M Cornell, S G Oliver, S Fields, and et al. Comparative assessment of large-scale data sets of protein-protein interactions. *Nature*, 417(6887):399–403, 2002.

[95] J Gagneur, L David, and L M Steinmetz. Capturing cellular machines by systematic screens of protein complexes. *Trends in Microbiology*, 14(8):336–339, 2006.

[96] J Goll and P Uetz. The elusive yeast interactome. *Genome Biology*, 7(6):223, 2006.

[97] G T Hart, A K Ramani, and E M Marcotte. How complete are current yeast and human protein-interaction networks? *Genome Biology*, 7(11):120, 2006.

[98] J Wu, T Vallenius, K Ovaska, J Westermarck, T P Makela, and S Hautaniemi. Integrated network analysis platform for protein-protein interactions. *Nature Methods*, 6(1):75–77, 2009.

[99] X F Dai, T Erkkilä, O Yli-Harja, and H Lähdesmäki. A joint finite mixture model for clustering genes from independent Gaussian and beta distributed data. *BMC Bioinformatics*, 10:165, 2009.

[100] *MATLAB technical documentation*. http://www.mathworks.com/access/helpdesk/help/techdoc/index.html?/access/helpdesk/help/techdoc/matlab.html&http://en.wikipedia.org/wiki/MATLAB.

[101] X F Dai, H Lähdesmäki, and O Yli-Harja. A stratified beta-Gaussian finite mixture model for clustering genes with multiple data sources. *International Journal On Advances in Life Sciences*, 1(1):14–25, 2009.

[102] M J Herrgård, B-S Lee, V Portnoy, and B ø Palsson. Integrated analysis of regulatory and metabolic networks reveals novel regulatory mechanisms in *Saccharomyces*. *Genome Research*, 16(5):627–635, 2006.

[103] The Gene Ontology Consortium. The gene ontology project in 2008. *Nucleic Acids Research*, 36(Database issue):D440–D444, 2008.

[104] The Gene Ontology Consortium. Gene Ontology: tool for the unification of biology. *Nature Genetics*, 25:25–29, 2000.

[105] The Gene Ontology Consortium. Creating the gene ontology resource: design and implementation. *Genome Research*, 11:1425–1433, 2001.

[106] The Gene Ontology Consortium. The gene ontology (go) database and informatics resource. *Nucleic Acids Research*, 32:D258–D261, 2004.

[107] The Gene Ontology Consortium. The gene ontology project in 2006. *Nucleic Acids Research*, 34:D322–D326, 2006.

[108] S A Ramsey, S L Klemm, D E Zak, K A Kennedy, V Thorsson, B Li, and et al. Uncovering a macrophage transcriptional program by integrating evidence from motif scanning and expression dynamics. *PLoS Computational Biology*, 4(2):e1000021, 2008.

[109] A Gelman, J B Carlin, H S Stern, and D B Rubin. *Bayesian Data Analysis (2nd edition)*. Chapman & Hall/CRC, Boca Raton, F.L., USA, 2004.

Appendix

Clustering framework of BGBMM

```
function [StructG,cluster,conver,time] = cluster_betanormbernoulli(Data_bind,
                                    Data_expr,Data_ppi,minDistribution,
                                    maxDistribution,gov,genes,Ann,GO,aspect)
% INPUT:
% Data_bind         - A m-by-y matrix containing beta distributed data.
% Data_expr         - A m-by-x matrix containing Gaussian distributed data.
% Data_ppi          - A m-by-m binary contact matrix.
% maxDistribution   - The maximum number of possible underlying distributions.
% minDistribution   - The minimum number of possible underlying distributions.
% gov               - 0: no GO validation included.
%                     1: GO validation included.
% genes             - Gene names for GO validation.
% Ann               - Gene annotation for GO validation.
% GO                - Gene ontology.
% aspect            - F: molecular function.
%                     C: cellular component.
%                     P: biological process.
%                     All: all three aspects are included.
% OUTPUT:
% StructG           - Clustering results
%                     Field 'GeneCluster': the clustering results.
%                     Field 'Likelihood': the likelihood of the
                                        clustering results.
%                     Field 'NumberOfDistributions': the estimated
```

```
                                        number of clusters.
%                     Field 'AssumedNumberOfDistributions': the
                                        true number of clusters.
%                     'StructG1' to 'StructG4' correspond to BIC,
                                    ICL, AIC, AIC3, respectively.
% conver            - Convergence score.
% time              - Time needed for a clustering event.

tic;
D1 = size(Data_bind,1);
D2_beta = size(Data_bind,2);
D2_norm = size(Data_expr,2);
D2_bernoulli = size(Data_ppi,2);
Data_bind(Data_bind < 0.0001) = 0.0001;
Data_bind(Data_bind > 0.9999) = 0.9999;
iteration = 100;
l = cell(iteration,1);
Z = cell(iteration,1);
C = cell(iteration,1);
cluster = cell(maxDistribution-minDistribution+1,1);
ll = cell(maxDistribution-minDistribution+1,1);
num = 1;
z = cell(maxDistribution-minDistribution+1,1);
for mm = minDistribution:maxDistribution
    for iter = 1:iteration
        [C{iter},q(iter),l{iter},Z{iter},con(iter)] =
        em_betanormbernoulli_standard(Data_bind,Data_expr,Data_ppi,mm);
    end
    conver(num) = sum(con);
    [Q(num),ind] = max(q);
    c{num} = C{ind};
    ll{num} = l{ind};
    z{num} = Z{ind};
    for j = 1:num
        for i = 1:length(c{j})
```

```
            if cellfun('isempty',c{j}(i))
                continue;
            else
                cluster{j}(cell2num(c{j}(i))) = i;
            end
        end
    end
if gov == 1
        GOV(num) = 0;
        for j = 1:length(aspect)
            temp = calculateES(cluster{num},genes,Ann,GO,aspect{j},0);
            GOV(num) = GOV(num) + mean(temp) + median(temp);
        end
    elseif gov == 0
        BIC(num) = -2*sum(log(sum(ll{num},2)))
                   +(D2_norm+mm*D2_norm+2*mm*D2_beta+mm*D2_bernoulli+mm-1)
                   *log(D1*(D2_norm+D2_beta+D2_bernoulli));
        ICL(num) = -2*sum(log(sum(ll{num},2)))
                   +(D2_norm+mm*D2_norm+2*mm*D2_beta+mm*D2_bernoulli+mm-1)
                   *log(D1*(D2_norm+D2_beta+D2_bernoulli))
                   -2*sum(log(prod(z{num}.^z{num},2)));
        AIC2(num) = -2*sum(log(sum(ll{num},2)))
                    +(D2_norm+mm*D2_norm+2*mm*D2_beta+mm*D2_bernoulli+mm-1)*2;
        AIC3(num) = -2*sum(log(sum(ll{num},2)))
                    +(D2_norm+mm*D2_norm+2*mm*D2_beta+mm*D2_bernoulli+mm-1)*3;
    else
        GOV(num) = 0;
        for j = 1:length(aspect)
            temp = calculateES(cluster{num},genes,Ann,GO,aspect{j},0);
            GOV(num) = GOV(num) + mean(temp) + median(temp);
        end
        BIC(num) = -2*sum(log(sum(ll{num},2)))
                   +(D2_norm+mm*D2_norm+2*mm*D2_beta+mm*D2_bernoulli+mm-1)
                   *log(D1*(D2_norm+D2_beta+D2_bernoulli));
        ICL(num) = -2*sum(log(sum(ll{num},2)))
```

```
                    +(D2_norm+mm*D2_norm+2*mm*D2_beta+mm*D2_bernoulli+mm-1)
                    *log(D1*(D2_norm+D2_beta+D2_bernoulli))
                    -2*sum(log(prod(z{num}.^z{num},2)));
        AIC2(num) = -2*sum(log(sum(ll{num},2)))
                     +(D2_norm+mm*D2_norm+2*mm*D2_beta+mm*D2_bernoulli+mm-1)*2;
        AIC3(num) = -2*sum(log(sum(ll{num},2)))
                     +(D2_norm+mm*D2_norm+2*mm*D2_beta+mm*D2_bernoulli+mm-1)*3;
    end
    num = num + 1;
end
if gov == 1
    [B,M] = min(GOV);
    ClusterG_mult = cluster{M};
    BpG_mult = ll{M};
    DisG_mult = numel(intersect([1:(minDistribution+M-1)],ClusterG_mult));
    StructG = struct('GeneCluster',ClusterG_mult,'Likelihood',BpG_mult,
              'NumberOfDistributions', DisG_mult,'AssumedNumberOfDistributions',
              minDistribution+M-1);
elseif gov == 0
    [B,M1] = min(BIC);
    ClusterG_mult1 = cluster{M1};
    BpG_mult1 = ll{M1};
    DisG_mult1 = numel(intersect([1:(minDistribution+M1-1)],ClusterG_mult1));
    StructG1 = struct('GeneCluster',ClusterG_mult1,'Likelihood',BpG_mult1,
               'NumberOfDistributions',DisG_mult1,'AssumedNumberOfDistributions',
               minDistribution+M1-1);
    [B,M2] = min(ICL);
    ClusterG_mult2 = cluster{M2};
    BpG_mult2 = ll{M2};
    DisG_mult2 = numel(intersect([1:(minDistribution+M2-1)],ClusterG_mult2));
    StructG2 = struct('GeneCluster',ClusterG_mult2,'Likelihood',BpG_mult2,
              'NumberOfDistributions',DisG_mult2,'AssumedNumberOfDistributions',
              minDistribution+M2-1);
    [B,M3] = min(AIC2);
    ClusterG_mult3 = cluster{M3};
```

```
    BpG_mult3 = ll{M3};
    DisG_mult3 = numel(intersect([1:(minDistribution+M3-1)],ClusterG_mult3));
    StructG3 = struct('GeneCluster',ClusterG_mult3,'Likelihood',BpG_mult3,
               'NumberOfDistributions',DisG_mult3,'AssumedNumberOfDistributions',
               minDistribution+M3-1);
    [B,M4] = min(AIC3);
    ClusterG_mult4 = cluster{M4};
    BpG_mult4 = ll{M4};
    DisG_mult4 = numel(intersect([1:(minDistribution+M4-1)],ClusterG_mult4));
    StructG4 = struct('GeneCluster',ClusterG_mult4,'Likelihood',BpG_mult4,
               'NumberOfDistributions',DisG_mult4,'AssumedNumberOfDistributions',
               minDistribution+M4-1);
    StructG = {StructG1,StructG2,StructG3,StructG4};
else
    [B,M1] = min(BIC);
    ClusterG_mult1 = cluster{M1};
    BpG_mult1 = ll{M1};
    DisG_mult1 = numel(intersect([1:(minDistribution+M1-1)],ClusterG_mult1));
    StructG1 = struct('GeneCluster',ClusterG_mult1,'Likelihood',BpG_mult1,
               'NumberOfDistributions',DisG_mult1,'AssumedNumberOfDistributions',
               minDistribution+M1-1);
    [B,M2] = min(ICL);
    ClusterG_mult2 = cluster{M2};
    BpG_mult2 = ll{M2};
    DisG_mult2 = numel(intersect([1:(minDistribution+M2-1)],ClusterG_mult2));
    StructG2 = struct('GeneCluster',ClusterG_mult2,'Likelihood',BpG_mult2,
               'NumberOfDistributions',DisG_mult2,'AssumedNumberOfDistributions',
               minDistribution+M2-1);
    [B,M3] = min(AIC2);
    ClusterG_mult3 = cluster{M3};
    BpG_mult3 = ll{M3};
    DisG_mult3 = numel(intersect([1:(minDistribution+M3-1)],ClusterG_mult3));
    StructG3 = struct('GeneCluster',ClusterG_mult3,'Likelihood',BpG_mult3,
               'NumberOfDistributions',DisG_mult3,'AssumedNumberOfDistributions',
               minDistribution+M3-1);
```

```
    [B,M4] = min(AIC3);
    ClusterG_mult4 = cluster{M4};
    BpG_mult4 = ll{M4};
    DisG_mult4 = numel(intersect([1:(minDistribution+M4-1)],ClusterG_mult4));
    StructG4 = struct('GeneCluster',ClusterG_mult4,'Likelihood',BpG_mult4,
               'NumberOfDistributions',DisG_mult4,'AssumedNumberOfDistributions',
               minDistribution+M4-1);
    [B,M5] = min(GOV);
    ClusterG_mult5 = cluster{M5};
    BpG_mult5 = ll{M5};
    DisG_mult5 = numel(intersect([1:(minDistribution+M5-1)],ClusterG_mult5));
    StructG5 = struct('GeneCluster',ClusterG_mult5,'Likelihood',BpG_mult5,
              'NumberOfDistributions',DisG_mult5,'AssumedNumberOfDistributions',
              minDistribution+M5-1);
    StructG = {StructG1,StructG2,StructG3,StructG4,StructG5};
end
time = toc;
```

EM algorithm of BGBMM

```
function [cluster,Q,ll,z,convergence] = em_betanormbernoulli_standard
         (data_bind,data_expr,data_ppi,k)
% INPUT:
% data_bind         - A m-by-y matrix containing beta distributed data.
% data_expr         - A m-by-x matrix containing Gaussian distributed data.
% data_ppi          - A m-by-m binary contact matrix.
% k                 - The number of clusters.
% OUTPUT:
% cluster           - Clustering results.
% Q                 - The expectation of the log likelihood.
% ll                - Likelihood.
% z                 - The estimated prior probabilities.
% convergence       - Convergence score: the number of iterations that have
%                     reached convergence.
```

```
max_iter = 100;
convergence = 1;
verbose = 0;
n = size(data_expr,1);
m_beta = size(data_bind,2);
m_norm = size(data_expr,2);
m_bernoulli = size(data_ppi,2);
if k>1
    mu_norm = fcm(data_expr, k, [2.0 100 1e-3 verbose]).';
else
    mu_norm = mean(data_expr, 1).';
end
sigma_norm = covfixer2(diag(diag(cov(data_expr))));
for i = 1:m_beta
    alpha(i,:) = randint(1,k,[i*10+1 (i+1)*10]);
    beta(i,:) = randint(1,k,[(i-1)*10+1 i*10]);
end
for i = 1:m_bernoulli
    tmp(:,i) = 10.*i.*rand(k,1);
end
proto = tmp./repmat((sum(tmp,1)*2),k,1);
cluster = cell(k,1);
prob = ones(k,1) * (1/k);
Q = 0;
iter = 0;
changed = 1;
p = zeros(n,k);
p_beta = zeros(n,k);
p_norm = zeros(n,k);
p_bernoulli = zeros(n,k);
very_small_norm = mean(std(data_expr))*(1/k)*0.0001;
very_small_H = mean(std(data_bind))*(1/k)*0.0001;
Qs = [];
while (iter < max_iter) & changed
```

```
iter = iter + 1;
old_Q = Q;
Qs = [Qs,old_Q];
if or(sum(sum(isnan(sigma_norm))),sum(sum(isinf(sigma_norm))))
    break;
end
tol = m_norm*norm(sigma_norm)*eps*4;
corank = rank(sigma_norm,tol);
d = prod(diag(sigma_norm));
if corank == m_norm & d > very_small_norm
    invsigma_norm = inv(sigma_norm);
else
    sigma_norm = sigma_norm + eye(m_norm)*3*max(tol,very_small_norm);
    invsigma_norm = inv(sigma_norm);
    d = prod(diag(sigma_norm));
end
for i = 1:k
    dist_norm = data_expr - repmat(mu_norm(:,i)',n,1);
    p_norm(:,i) = (1/sqrt(d))*exp(-0.5*(sum((dist_norm*invsigma_norm)
                  .*dist_norm,2)));
    for j = 1:m_beta
        tmppdf = betapdf(data_bind(:,j),alpha(j,i),beta(j,i));
        tmppdf(tmppdf==0) = eps;
        pdf_beta(:,j) = tmppdf;
    end
    for j = 1:m_bernoulli
        pdf_bernoulli(:,j) = proto(i,j).^data_ppi(:,j).*((1-proto(i,j))
                             .^(1-data_ppi(:,j)));
    end
    p_beta(:,i) = prod(pdf_beta,2);
    p_bernoulli(:,i) = prod(pdf_bernoulli,2);
    p(:,i) = p_beta(:,i).*p_norm(:,i).*p_bernoulli(:,i);
end
ll = [];
for w = 1:k
```

```
    ll = [ll, prob(w)*p(:,w)];
end
nf = sum(ll,2);
nf2 = nf.*(nf > 0) + (nf == 0);
z = ll./repmat(nf2,1,k);
z(find(nf == 0),:) = 1/k;
likelihood = ll.^z;
Q = sum(log(sum(likelihood,2)));
for ind = 1:n
    [v(ind),posterior(ind)] = max(ll(ind,:));
end
[posteriorV,posteriorI] = sort(posterior);
for i = min(posterior):max(posterior)
    cluster{i} = posteriorI(posteriorV == i);
end
for i = 1:k
    prob(i) = mean(z(:,i));
    proto(i,:) = sum(repmat(z(:,i),1,m_bernoulli).*data_ppi,1)./sum(z(:,i));
    if prob(i) <= 0
        prob(i) = 0.001;
        fprintf(1,'betanormbernoulli_standard: prob smaller than 0\n');
    end
    for j = 1:m_beta
        if or(alpha(j,i)<1,beta(j,i)<1)
            break;
        end
        tmpa = z(:,i).*(psi(0,alpha(j,i))-psi(0,alpha(j,i)+beta(j,i))
               -log(data_bind(:,j)));
        funa = sum(tmpa(~isnan(tmpa)));
        tmpb = z(:,i).*(psi(0,beta(j,i))-psi(0,beta(j,i)+alpha(j,i))
               -log(1-data_bind(:,j)));
        funb = sum(tmpb(~isnan(tmpb)));
        tmpa2 = z(:,i).*(psi(1,alpha(j,i))-psi(1,alpha(j,i)+beta(j,i)));
        funa2 = sum(tmpa(~isnan(tmpa2)));
        tmpb2 = z(:,i).*(psi(1,beta(j,i))-psi(1,beta(j,i)+alpha(j,i)));
```

```
funb2 = sum(tmpa(~isnan(tmpb2)));
tmpab = z(:,i).*(-psi(1,alpha(j,i)+beta(j,i)));
funab = sum(tmpa(~isnan(tmpab)));
H = [funa2,funab;funab,funb2];
if isnan(sum(sum(H)))
    fprintf(1,'betanormbernoulli: NaN H\n');
elseif isinf(sum(sum(H)))
    fprintf(1,'betanormbernoulli: Inf H\n');
else
    tolH =2*norm(H)*eps*4;
    corankH = rank(H,tolH);
    dH = prod(diag(H));
    if ~(corankH == 2 & dH > very_small_H)
        H = H + eye(2)*3*tolH/very_small_H;
    end
end
temp = ([alpha(j,i);beta(j,i)] - inv(H)*[funa;funb])';
alpha_new(j,i) = max(1,temp(1));
beta_new(j,i) = max(1,temp(2));
newS1 = sign(alpha(j,i)-alpha_new(j,i));
newS2 = sign(beta(j,i)-beta_new(j,i));
flag = 1;
iteration = 1;
while and(iteration < max_iter,flag)
    oldS1 = newS1;
    oldS2 = newS2;
    tmpa = z(:,i).*(psi(0,alpha_new(j,i))-psi(0,alpha_new(j,i)
           +beta_new(j,i))-log(data_bind(:,j)));
    funa = sum(tmpa(~isnan(tmpa)));
    tmpb = z(:,i).*(psi(0,beta_new(j,i))-psi(0,beta_new(j,i)
           +alpha_new(j,i))-log(1-data_bind(:,j)));
    funb = sum(tmpb(~isnan(tmpb)));
    tmpa2 = z(:,i).*(psi(1,alpha_new(j,i))-psi(1,alpha_new(j,i)
            +beta_new(j,i)));
    funa2 = sum(tmpa2(~isnan(tmpa2)));
```

```
tmpb2 = z(:,i).*(psi(1,beta_new(j,i))-psi(1,beta_new(j,i)
        +alpha_new(j,i)));
funb2 = sum(tmpb2(~isnan(tmpb2)));
tmpab = z(:,i).*(-psi(1,alpha_new(j,i)+beta_new(j,i)));
funab = sum(tmpab(~isnan(tmpab)));
clear tmpa tmpb tmpa2 tmpb2 tmpab
alpha(j,i) = alpha_new(j,i);
beta(j,i) = beta_new(j,i);
H = [funa2,funab;funab,funb2];
if isnan(sum(sum(H)))
    fprintf(1,'betanormbernoulli_standard: NaN H\n');
elseif isinf(sum(sum(H)))
    fprintf(1,'betanormbernoulli_standard: Inf H\n');
else
    tolH =2*norm(H)*eps*4;
    corankH = rank(H,tolH);
    dH = prod(diag(H));
    if ~(corankH == 2 & dH > very_small_H)
        H = H + eye(2)*3*tolH/very_small_H;
    end
end
temp = ([alpha(j,i);beta(j,i)] - inv(H)*[funa;funb])';
alpha_new(j,i) = temp(1);
beta_new(j,i) = temp(2);
if or(or(alpha_new(j,i)<1,beta_new(j,i)<1),
   or(isnan(alpha_new(j,i)),isnan(beta_new(j,i))))
    alpha_new(j,i) = alpha(j,i);
    beta_new(j,i) = beta(j,i);
    break
end
difference1 = alpha(j,i)-alpha_new(j,i);
difference2 = beta(j,i)-beta_new(j,i);
newS1 = sign(difference1);
newS2 = sign(difference2);
if ~or(and(oldS1 == newS1,abs(difference1) > 0.01),
```

```
                    and(oldS2 == newS2,abs(difference2) > 0.01))
                     flag = 0;
                end
                iteration = iteration + 1;
            end
            alpha(j,i) = alpha_new(j,i);
            beta(j,i) = beta_new(j,i);
        end
        mu_norm(:,i) = (sum(repmat(z(:,i),1,m_norm).*data_expr)/sum(z(:,i)))';
    end
    for j = 1:m_norm
        for u = 1:k
            dist_norm = data_expr(:,j) - mu_norm(j,u);
            S(j,u) = sum(z(:,u).*dist_norm.*dist_norm);
        end
        sig(j) = sum(S(j,:))/n;
    end
    sigma_norm = diag(sig);
    changes = sum(sum(abs(Q - old_Q)))/(n*k);
    changed = changes > 0.0001;
end
if iter == max_iter
    convergence = 0;
end
```