TÉCNICO LISBOA

Latent Factors of Multi-Omics Data and Clustering

Xia Anbang

Thesis to obtain the Master of Science Degree in

Electrical and Computer Engineering

Supervisor(s): Prof. Alexandra Sofia Martins de Carvalho
Prof. Susana de Almeida Mendes Vinga Martins

Examination Committee

Chairperson: Prof. Teresa Maria Sá Ferreira Vazão Vasques
Advisor: Prof. Susana de Almeida Mendes Vinga Martins
Members of the Committee: Prof. Alexandra Sofia Martins de Carvalho

December 2019

Declaration

I declare that this document is an original work of my own authorship and that it fulfills all the requirements of the Code of Conduct and Good Practices of the Universidade de Lisboa.

Acknowledgments

First of all, I would like to thank my supervisors, Prof. Alexandra Carvalho and Prof. Susana Vinga for all the support they have given me and guidance for development of this master thesis. I would like to thank André Veríssimo for helping in acquiring data set from TCGA, server issues and other helps. I also would like to thank all my friends, my family and my girl friend Jun Chen for all the emotional supports.

This work was partially supported by national funds from Fundação para a Ciência e a Tecnologia (FCT), through projects PTDC/EMS-SIS/0642/2014, PTDC/CCI-CIF/29877/2017, PTDC/EEI-SII/1937/2014, UID/EEA/50008/2013, UID/CEC/50021/2019, and UID/EMS/50022/2019.

Abstract

Cancer is a very complex disease; often, its types cannot easily be classified simply by its location or manifested characteristics. In these circumstances, it is critical to find the group of patients who have similar underlying biological information and apply similar treatment for the person that belongs to the same group.

In the area of medicine, with the development of new technologies, nowadays we can collect enormous amounts of data and build databases of genes, for example, The Cancer Genome Atlas (TCGA) database which focuses on cancer diseases. TCGA database contains various types of omic data, such as mutations, RNA expressions and DNA methylation. In the genome-related database it is common to have more variables than samples and facing the curse of dimensionality problem.

With this multi-omic data, this work aims to discover unknown factors common to the three data types using factor analysis tools such as iCluster and MOFA; this dimension reduction process can select more relevant information for further analysis.

This thesis proposes a methodology that compares MOFA and iCluster by finding the underlying latent factors and perform a clustering of patients who share biological similarities in the same group according to the obtained latent factors.

After testing both methods first on the synthetic data and comparing their abilities to recover the underlying factors and clusters, we decide to apply MOFA method for the Ovarian Carcinoma(OV) data extracted from TCGA, to find latent factors and the relevant clustering results.

From synthetic data analysis, we conclude that the MOFA has better performance. For real data, the genes find are cancer related but the cluster results are insignificant.

Keywords: TCGA, FA, PCA, MOFA, Clustering, iCluster, K-means, Curse of dimensionlaity

Resumo

Com o desenvolvimento tecnologia experimental celular e molecular, tem-se assisitdo a um aumento significativo da informação gerada e disponível para análise. Na área da medicina foi possível construir base de dados acerca de genes, nos quais se destaca The Cancer Genome Atlas (TCGA) em que se foca nas doenças de cancro. Na base de dados de TCGA contém vários tipos de dados omicos, tais como mutações, expressões de RNA, e metilação de DNA.Com estes dados heterogéneos, este trabalho pretende descobrir factores escondidos comuns aos três tipos de dados através do uso de técnicas de análise fatorial, através de ferramentas existentes, nomeadamente iCluster e MOFA. Este processo de redução de dimensionalidade consegue selecionar informações mais relevantes para posterior análise.

O cancro é considerada uma doença complexa, muitas vezes não se consegue classificá-los facilmente o seu tipo apenas através do local da doença ou por fen]otipos. Nestas circunstâncias, procurar pacientes que tenham informações biológicas semelhantes para formar um grupo com o mesmo tratamento pode ser uma tarefa importante.

Esta tese propõe a metodologia de comparar duas técnicas de reduzir a dimensão dos dados e procurar informação escondidas e usar esta informação para posterior análise.

Primeiro vai ser gerado um conjunto de dados sintéticos para testar a capacidade de encontrar informação escondida e recuperar os grupos, depois vai ser utilizado dados de cancro ovário de TCGA para encontrar grupos de pacientes similiares e com que se baseia esta classificação.

Através do análise dos dados sintéticos, concluiu-se que MOFA apresenta desempenho melhor. Com os dados de cancro ovário, conseguiu-se obter um conjunto de genes relacionados com cancro.

Keywords: TCGA, FA, PCA, MOFA, Clustering, iCluster

Contents

List of Tables

List of Figures

Acronyms

ARD Automatic Relevance Determination. 16

BIC Bayesian Information Criteria. 25

CHAT The Cancer Hallmarks Analytics Tool. 32

ELBO Evidence Lower Bound. 17

FA Factor Analysis. 2, 8

FMI Fowlkes and Mallows Index. 12

FPKM Fragments Per Kilobase per Million. 21

HGP Human Genome Project. 1

LF Latent Factor. xiii, 27, 28, 32–36

MOFA Multi-Omic Factor Analysis. 2, 15–17, 30, 41

OV Ovarian Carcinoma. 21, 22, 30, 41

PCA Principal Component Analysis. 2–4, 6, 7, 13

PPCA Probabilistic Component Analysis. 4, 7

STD Standard Deviation. xiii, 26, 27

TCGA The Cancer Genome Atlas. 1, 2, 21, 30, 41

Chapter 1

Introduction

1.1 Motivation

The data explosion in recent years is bringing enormous opportunities to machine learning, allowing the development of new algorithms that can achieve surprising results in various areas.

In the medical area, some gene related project has produced a huge amount of high dimensional data. Human Genome Project (HGP) with objective of determining sequences of nucleotide base pairs that make up human DNA. The Cancer Genome Atlas (TCGA), a project that wants to catalogue genetic mutations responsible for mutations using genome sequence and bioinformatics. Other technological advances increasingly enable multiple data types across different biological layers ranging from genome, epigenome, trancriptome, proteome and metabolome to phenome profiling [3].

When apply machine learning in medicine, instead like doctor approaches problems and finding solutions through constant learning and progressing during the career, it tries learning rules from data. Starting with patient-level observations, it tries to find algorithm to deal with vast number of variables, looking for combination that can predict the outcomes. Where machine learning shines is in handle with enormous number of features, remarkably in cases where predictor is more than observations which is called ”curse of dimensionality” [4], and needs to be dealt by combining in non-linear and highly iterative ways [5].

Machine learning has demonstrated potentials in analyzing large, complex biological data [6], through this capacity allows us to use in new appeared kind of data, whose sheer volume or complexity would previously have made analyzing them unimaginable [7].

An important technique to make machine learning algorithms to having feasible results is reducing the number of features. In general there exists two types of dimension reduction technique. First type, which reduce the feature numbers by selecting the most relevant features that have influence in the result. The representative of this type of technique is when use, for example optimization functions as LASSO [8] and Elastic Net[9] penalties in regression models. Another type consist using mathematical techniques that transform the original features space into new subspace. In this subspace the data points are more closer and the density of points is higher, the distance calculation is easier too. Principal Component Analysis and Factor Analysis are most used technique in this family. And this work are focused on methods that are based on the last type.

When applying these methods to analyze cancer, we are assuming that the development of cancer on the human body is influenced by some genes. And when reducing the dimension of features we need to take into account the interpretability of results.

Finally, with reduced data of gene sequences, we can cluster the patients into different clusters in way

that the patients come from same clustering are similar and different from inter-clusters. The grouping of patients is useful for medical treatment, because the similar patients may be treated efficiently by same treatment, and it is possible to separate the cancers by their intrinsic characteristics in various ways.

1.2 Objectives

As previously mentioned, the high dimensional data-set faces the problem ”curse of dimentionality”, there exists several machine learning techniques to deal with this problem by using dimension reduction techniques. This paper focuses on the techniques that are Principal Component Analysis (PCA) and Factor Analysis (FA) based.

This work considers using the biological heterogeneous data to find a latent space with latent factors that are common on all types of data. From the latent factors discovered, the data have much fewer dimensions, so we can apply other analysis algorithms to them and the result is more interpret-able.

The specific purpose is to use two types of framework that already exist in the literature, Multi-Omic Factor Analysis (MOFA) [10] and iCluster[11]. First of all, we apply them to the generated synthetic data that has characterization of multi-omic biological data, analyze their results and interpretability for testing the viability of framework. Then we apply the framework to the real data from TCGA for getting the latent factors. After obtain the latent factors, we use the clustering algorithms to grouping the patients.

The clustering on latent factors is based on the assumption that, the patients who suffer the same type of cancer might have similarities among each other, and by this similarity we can divide them by sub-types. Different from the traditional pathogenic pathway to identifying the sub-types, machine learning algorithms want to group them by genomic profiling. Using unsupervised dimension reduction and clustering techniques, we can deal with the huge amount of information which cannot be interpreted efficiently by Human to produce clusters of patients that share genomic level similarities. At the same time, we can be aware of the importance of each gene feature in the clustering criteria. After that, the specialists in cancer diseases can study the significance of this grouping, the influence of important features, the possibility for using the same treatment to the patients in the same group, etc.

1.3 Document outline

The remaining parts of this document are organized as follows. Chapter 2 introduces the theoretical basis of dimension reduction techniques. Chapter 3 introduces the cluster evaluation methods and the K-means algorithm. Chapter 4 introduces the methodology used, which explains the integrated frameworks, data selection, processing and codification, the way to use frameworks and how to compare both integrated frameworks. Chapter 5 shows the results obtained by using synthetic data, the analysis of the results for evaluating the frameworks and finally the application of the methodology on real data. Chapter 6 describes the achievements obtained by this work and possible future works.

Chapter 2

Dimensionality reduction

When we have a set of data with large number of dimensions, it is often desired to reduce the dimension, simplifying the data-set as much as possible while keeping the original information. For the biomedical data analysis, we often have high dimensional data-sets, then applying dimension reduction techniques are useful.

This thesis is focused in the factor analysis related methods. Like PCA, probabilistic PCA, factor analysis and integrated frameworks that can infer hidden factors.

2.1 Principal Component Analysis

The principal component analysis is a well-established statistical procedure for resolving dimension reduction problem through transforming a set of observations that maybe are correlated into a uncorrelated lower dimension set.

Considering a set of observed data matrix $Y \in \mathbb{R}^{n \times d}$, with n samples and d features. The objective is reducing the original data dimension d into lower dimension k, such that $k < d$. PCA approximates the original matrix Y by component weights W and principal components Z,

$$Y \approx ZW^T, \tag{2.1}$$

where $Z \in \mathbb{R}^{n \times k}$ and $W \in \mathbb{R}^{d \times k}$.

In the case of PCA, we can have two different interpretations:

1. Minimize the error, or in other words, minimizes the average of projection cost, defined as the mean squared distance between the data points and their projections(Pearson, 1901):

$$\underset{Z \subset \mathbb{R}^{n \times k}, W \subset \mathbb{R}^{d \times k}}{\arg\min} ||Y - ZW^T||_F = \sum_{i=1}^{n} \sum_{j=1}^{d} (y_j^i - w_j^T z^i)^2. \tag{2.2}$$

2. Maximize the variance of projected data(Hotelling, 1933). Assuming that the data have zero mean,

$\mu_z = 0$ and $z^i = Wy^i$. Then,

$$\begin{aligned}
&\arg\max_{W \subset \mathbb{R}^{d \times k}} \sum_{i=1}^{n} ||z^i - u_z|^2 = ||Wy^i||^2 = \\
&= \sum_{i=1}^{n} Tr((x^i)^T W^T W x^i) = Tr(W^T W \sum_{i=1}^{n} x^i (x^i)^T) \\
&= Tr(W^T W X^T X).
\end{aligned} \tag{2.3}$$

In fact, both interpretations lead to the same result as shows by following proof[12]:
Starting by distance to the hyper-plane(minimize the distance),

$$\begin{aligned}
||ZW - Y||^2 = ||YW^T W - Y||^2 = Tr((YW^T W - Y)(YW^T W - Y)) = \\
= Tr(W^T W X^T X W^T W) - 2Tr(W^T W X^T X) + Tr(X^T X) = \\
= Tr(W^T W W^T W X^T X) - 2Tr(W^T W X^T X) + Tr(X^T X) = \\
= -Tr(W^T W X^T X) + \text{constant}.
\end{aligned} \tag{2.4}$$

2.2 Factor analysis

Factor analysis is a model that seeks to relate a d-dimensional observation vector y to corresponding k-dimensional vector of latent(unobserved) variables z, represented by following linear relationship:

$$y = Wz + \mu + \epsilon, \tag{2.5}$$

where W is the weight matrix that relates the observed set with hidden set. The parameter μ allows the model to have noon-zero mean, and ϵ represents the residual error. Conventionally the latent variables, $z \approx$ N(0,I) is defined to be independent and Gaussian with unit variance. By defining the error or noise term ϵ to be likewise Gaussian $\epsilon \approx$ N(0,Ψ), induces corresponding observations y to be Gaussian $y \approx$ N(μ, $WW^T + \Psi$). The parameters are determined by maximum-likelihood algorithm, because there is no closed-form analytic solution for finding W and Ψ.

2.3 Probabilistic principal component analysis

The normal PCA formulation discussed previously is based on the linear projection of original data into lower dimensional subspace data. PCA also can be expressed in the maximum likelihood solution of a probabilistic latent variable model named as probabilistic PCA(Probabilistic Component Analysis (PPCA)) [13] proposed by M. Tipping and C. Bishop at 1999, which is a special case of latent variable model resulting when using the isotropic Gaussian noise model in the ϵ term present in the equation 2.5 $\epsilon \sim$ N(0, $\sigma^2 I$).

Consider a prior distribution over latent variable z given by zero-means with unit variance,

$$p(z) = N(z|0, I). \tag{2.6}$$

And the conditional distribution for the observed variable y conditioned on the value of the latent

variable as:

$$p(y|z) = N(y|Wz + \mu, \sigma^2 I), \tag{2.7}$$

where the mean of y is a linear combination of matrix W with matrix of samples in latent space added with vector μ, and the variance is giving by $\sigma^2 I$. With this assumption, the observed values y are mapped by 2.5 with $\epsilon = \sigma^2 I$. Note that, in this framework the variables are mapped from the latent space to the observed space, the reverse mapping is obtained by using Bayes's Theorem for use it in the practical situations to find the latent variables z.

We need to determine the values of the parameters W, μ and σ^2, first by using maximum likelihood. To write down the likelihood function, we need the marginal distribution of the observed data y which is obtained by integrating out the latent variables:

$$p(y) = \int_z p(y|z)p(z)dz, \tag{2.8}$$

which is also Gaussian like, and it's given by

$$p(y) = N(\mu, C), \tag{2.9}$$

where the observation co-variance model is specified by $C = WW^T + \sigma^2 I$.

From the equation 2.5 and considering to be Gaussian, it's mean and co-variance will be

$$E[y] = E[Wz + \mu + \epsilon] = \mu \tag{2.10}$$

$$\begin{aligned} cov[y] &= E[(Wz + \epsilon)(Wz + \epsilon)^T] \\ &= E[Wzz^T W^T] + E[\epsilon\epsilon^T] = WW^T + \sigma^2 I \end{aligned} \tag{2.11}$$

We first introduce here C^{-1} which involves the inversion of $D \times D$ size matrix, and it will be used in later calculations. The computation to invert C can be reduced by formulate as [14],

$$C^{-1} = \sigma^{-1} I - \sigma^{-2} W M^{-1} W^T \tag{2.12}$$

where M is a $M \times M$ size matrix defined by

$$M = W^T W + \sigma^2 I. \tag{2.13}$$

By inverting M in spite of invert C directly, the computational cost is reduced from $\mathcal{O}(D^3)$ to $\mathcal{O}(M^3)$.

As mentioned previously more important than the predictive distribution $p(y)$ is posterior distribution $p(z|y)$ of the latent variables giving the observation y. Which can be calculated using the Bayes's rule, and it's also Gaussian given by:

$$p(z|y) = \mathcal{N}(z|M^{-1}W^T(y - \mu), \sigma^{-2}M^{-1}). \tag{2.14}$$

Nothing that the posterior μ depends on y, but the co-variance it's independent of y.

2.3.1 Parameter estimation by ML

Firstly we consider determining parameters W and σ^2 of the model by using maximum likelihood. Giving a data set $Y = y_n$ with relation expressed by equation 2.9, the corresponding log likelihood function is

giving by

$$\begin{aligned}\log p(Y|\mu, W, \sigma^2) &= \sum_{n=1}^{N} \log p(y_n|\mu, W, \sigma^2) \\ &= -\frac{ND}{2}\log(2\pi) - \frac{N}{2}\log|C| - \frac{1}{2}\sum_{n=1}^{N}(y_n - \mu)^T C^{-1}(y_n - \mu)\end{aligned} \tag{2.15}$$

Derivative of log likelihood with respect to μ gives

$$\frac{\partial \mathcal{L}}{\partial \mu} = 0 \implies \mu = \bar{y} \tag{2.16}$$

where $\bar{y}$ is the data mean. The log likelihood is a quadratic function of μ, this is unique maximum of function. Going back to 2.15 and substituting the μ we can simplifying the log likelihood term to

$$\mathcal{L} = -\frac{N}{2}D\log(2\pi) + \log|C| + Tr(C^{-1}S), \tag{2.17}$$

where S is the data co-variance matrix defined by

$$S = \frac{1}{N}\sum_{n=1}^{N}(y_n - \mu)(y_n - \mu)^T. \tag{2.18}$$

Maximization with respect to W and σ^2 is more complex, but it was already proved in [13] that it has exact closed form solution and all of the stationary points can be written in the form

$$W_{ML} = U_M(L_M - \sigma^2 I)^{1/2}R, \tag{2.19}$$

where U_M is a $D \times M$ matrix whose columns are the principal eigenvectors of S, the L_M is a $M \times M$ diagonal matrix and R is an arbitrary $M \times M$ orthogonal matrix. Other combinations of eigenvectors(i.e. non-principal ones) correspond to saddle-pints of the likelihood function.

Assuming the matrix U have the eigenvectors arranged in order of decreasing values of corresponding eigenvalues, in this case the columns of $W = W_{ML}$ defines the subspace of standard PCA, and the corresponding maximum likelihood solution for σ^2 is given by

$$\sigma^2_{ML} = \frac{1}{D-M}\sum_{i=M+1}^{D}\lambda_i, \tag{2.20}$$

that corresponds the average variance associated with the dimensions discarded.

To obtain latent values from observed data y, we can reverse this mapping using Bayes's rule. From equation 2.14, the expected value is given by

$$E[z|x] = M^{-1}W_{ML}^T(y - \bar{y}) \tag{2.21}$$

where M is given by 2.13.

2.3.2 Parameter estimation by EM

The probabilistic PCA model can be expressed in terms of a marginalization over a continuous latent space z for each data point y_n, we can make use the EM algorithm to find the maximum likelihood estimates of the model parameters. This may seem useless when we have a closed form solution

of ML, but in problems of high dimensional, the EM algorithm gives computational advantages when working with iterative procedures than sample co-variance matrix and ability to handle missing data.

We can use general framework for EM to find the parameters for probabilistic PCA. The data points are assumed to be independent, then the log-likelihood functions is given by

$$\log p(Y, Z|\mu, W, \sigma^2) = \sum_{i=1}^{n} \log p(y_n|z_n) + \log p(z_n). \tag{2.22}$$

In the E-step, the expectation with respect to the posterior distribution over the latent variables takes account with 2.6 and conditional expression 2.7 , is showed in [14] the expectation is given by,

$$\begin{aligned} E[\log p(Y, Z|\mu, W, \sigma^2)] = -\sum_{n=1}^{N} \Big\{ \frac{D}{2} \log(2\pi\sigma^2) + \frac{1}{2} Tr(E[z_n z_n^T]) + \frac{1}{2\sigma^2} ||x_n - \mu||^2 - \\ \frac{1}{\sigma^2} E[z_n]^T W^T (x_n - \mu) + \frac{1}{2\sigma^2} Tr(E[z_n z_n^T] W^T W) \Big\}. \end{aligned} \tag{2.23}$$

Is used old parameter values to evaluate:

$$E[z_n] = M^{-1} W^T (\bar{y} - y) \tag{2.24}$$

$$E[z_n z_n^T] = \sigma^2 M^{-1} + E[z_n] E[z_n]^T \tag{2.25}$$

which follows directly from posterior distribution 2.14 and the standard result $E[z_n z_n^T] = cov[z_n] + E[z_n]E[z_n]^T$.

In the M step, we maximize with respect to W and σ^2, keeping the posterior statistics fixed. Maximization with respect to σ^2 is straightforward. For the maximization with respect to W is given by [13],

$$W_{new} = SW(\sigma^2 I + M^{-1} W^T S W^{-1}), \tag{2.26}$$

$$\sigma^2_{new} = \frac{1}{d} Tr(S - S W M^{-1} W^T_{new}). \tag{2.27}$$

Where, S is given by 2.18 and M by 2.13.

With EM algorithm for solving the PPCA problem, it proceeds by initializing the parameters values then alternately computing the vales of 2.23 using 2.24 and 2.25 and updating the parameters values in M-step using 2.26 and 2.27.

Comparison of dimension reduction approaches The formulation as PPCA compared with a conventional PCA brings several advantages:

- We can use EM algorithms to find the solution, which is more computational efficiently in some situations by avoiding to calculate the covariance matrix when only a few leading eigenvectors are required.
- The combination of probabilistic model with EM allows the algorithm works when have missing values data.
- The existence of likelihood function allows comparison of model's efficiency with other probabilistic density models.

Factor analysis is a linear-Gaussian latent variable model that is closely related to probabilistic PCA, differs only in that FA have a diagonal covariance Ψ in spite of isotropic covariance.

Chapter 3

Clustering

In the unsupervised learning, the objects are no assigned to any label, the objective is to find the underlaying pattern and intrinsic information of data-set for posterior analysis. Is used clustering techniques for this types of task.

In the biomedical data analysis, the clustering methods has been proved to have great efficient to discover the underlaying patterns[15]. Because in the medical treatments, most of times there do not exist the clear grouping definition for each group. And grouping patients with similarities together allow specialist in the area to treat them more efficiently, after identifying the group can use most efficient treatment directly avoiding experimental treatments.

Clustering tries to separates the objects from data-set to various non overlapping subsets, each subset is denominated by cluster. From this separation, we can find patterns, information, rules of each subset and using them.

Assuming we have a data matrix D = $\{x_1, x_2,...,x_m\}$, which contains m unlabeled objects, for each object x_i = (x_{i1}, ..., x_{ip}) is a p dimensional vector. the objective of clustering are separates the data-set D into k clusters, $\{C_l \mid l$ = 1,2,...,k$\}$ such that the intersection of each cluster is null . All the objects are attributed with one and only one label. The algorithm returns a p dimensional vector with respective object label.

The separation on clusters are through similarity(or dissimilarity), the most popular dissimilarity measure for metric representations is the distance, for instance the Euclidean.

Clustering techniques can be roughly divided into two categories:

- Hierarchical.
- Partitioning.

Hierarchical clustering techniques are able to find structures which can be further divided in substructures and so on recursively. The result is a hierarchical structure of groups known as dendrogram. And also the hierarchical clustering algorithms can be divided into two categories. Agglomerative, a bottom-up approach which each single object are considered to be a cluster, then go up through merging the two closest clusters until forming one single group that contains all the objects. It is divisive, which is opposite of agglomerative, by starting from one single group then recursively separates into two clusters.

In partitioning clustering, the objects are separated into disjoint sets by some criterion, distant based is the most used one. And the closest objects are grouped to form a cluster.

3.1 Clustering algorithms - K-means

In this section is introduced the most used clustering algorithm. There exist much more clustering algorithms, but in this work was only used a K-means.

Giving a data-set D = $\{x_1, x_2,...,x_m\}$, the K-means algorithm will cluster the samples into k clusters C = $\{C_1, C_2,...,C_k\}$, the objective is minimizing the within-cluster sum of squares. Formally is to find:

$$\arg\min_{c} \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||_2^2, \tag{3.1}$$

which μ_i = $\frac{1}{|C_i|} \sum_{x \in C_i} x$ is mean of points in C_i. The formula 3.1 minimizes the sum square distance of all points to each mean point of each cluster, the lowest the sum the more similar are samples in cluster C. To find the global minimum in this problem is difficult, it need take consideration to all samples in the cluster C, this is a NP hard problem [16]. However there are efficient heuristic, greedy, dynamic programming algorithms to find the local minimum quickly. Follow is presented the general K-means algorithm.

Algorithm 1 K-Means

1: *Input:* D = $\{x_1, x_2,...,x_m\}$
2: **procedure** K-MEANS
3: *Random select k samples from D to initialize the vector* $\{\mu_1, \mu_2,...,\mu_k\}$
4: *repeat*:
5: **for** `i=1,2...,k` **do**
6: C_i = Ø
7: **for** `j=1,2...,m` **do**
8: **for** `i=1,2...,k` **do**
9: $d_{ji} = ||x_j - \mu_i||_2$
10: $\lambda_j = \arg\min_{i \in \{1,2,...,k\}} d_{ji}$
11: $C_{\lambda_j} = C_{\lambda_j} \cup \{x_j\}$
12: **for** `i=1,2...,k` **do**
13: $\mu_i^{'} = \frac{1}{|C_i|} \sum_{x \in C_i} x_i$
14: **if** $\mu_i^{'} \neq \mu_i$ **then**
15: $\mu_i^{'} = \mu_i$
16: *until no more upgrade of* μ *values.*

Lines 5 and 6 are to initialize the Clusters with empty set. Line 9 is for calculating the distances between samples and cluster's centroids. Then in line 9 are calculated which cluster are closest, after the sample are classified to the closest cluster. Line 13 are recalculated the cluster's centroids, if cluster have modification then the mean are updated. If all cluster's centroids are not modified anymore, then the algorithm had achieved solution.

3.2 Clustering evaluation

The fact that the objects have no labels, the evaluation metrics for classifying the models performance are not so obvious. There exist 2 fundamental characteristics should need pay attention on clustering problems, validity index and distance measurement.

Considering the objective of clustering are grouping objects of data-set D in a way that the groups do not have overlapping samples. From the logical reasoning, we want similar objects clustering together and the difference of clusters as big as possible, i.e. maximize intra-cluster similarity and minimize inter-cluster similarity. Mainly there exist two types of validity index, external index which exists an external

reference model can be used to compare the prediction result. And internal index, evaluation metric that doesn't take account of any external reference. In this job is created a synthetic data with real labels, so will use external references.

Considering D = $\{x_1, x_2,...,x_m\}$, suppose from clustering we got C = $\{C_1, C_2,...,C_k\}$, and the comparison model with C^* = $\{C_1^*, C_2^*,...,C_k^*\}$, and corresponding obtained label λ and λ^*. We can pair-wise all the configuration, and defining

$$a = |SS|, SS = \{(x_i, x_j)|\lambda_i = \lambda_j, \lambda_i^* = \lambda_j*, i < j\}, \tag{3.2}$$

$$b = |SD|, SD = \{(x_i, x_j)|\lambda_i = \lambda_j, \lambda_i^* \neq \lambda_j*, i < j\}, \tag{3.3}$$

$$c = |DS|, DS = \{(x_i, x_j)|\lambda_i \neq \lambda_j, \lambda_i^* = \lambda_j*, i < j\}, \tag{3.4}$$

$$d = |DD|, DD = \{(x_i, x_j)|\lambda_i \neq \lambda_j, \lambda_i^* \neq \lambda_j*, i < j\}. \tag{3.5}$$

In the set SS containing the samples that are classified to same cluster in C and same also belongs to cluster in C^*, in the set SD containing the samples that are classified to same cluster in C and different cluster in C^*, DS containing the samples that are classified to different cluster in C and same cluster in C^* and DD containing the samples that are not classified to same cluster in C neither in C^*. The constrain $i < j$ makes the combination of pairs and total number of pairs are $a+b+c+d = m(m-1)/2$.

From equations 3.2, 3.3, 3.4 and 3.5 can infer the following index.

3.2.1 Precision and recall

The precision coefficient is defined as the proportion of points that are rightly grouped together in the C, that mean that it is also grouped together according to the reference C^*. It is defined by:

$$P = \frac{a}{a+c} \tag{3.6}$$

Similar with the recall coefficient, is defined as the proportion of points that are grouped together in the reference cluster C^*, which is also grouped together in C. And is defined by:

$$R = \frac{a}{a+b} \tag{3.7}$$

Once we have precision and recall, we can infer F-measure coefficient, which is the harmonic mean of the precision and recall, defined as:

$$F = \frac{2}{\frac{1}{P} + \frac{1}{r}} = \frac{2PR}{P+R} = \frac{2a}{2a+b+c} \tag{3.8}$$

3.2.2 Jaccard coefficient(JC)

The Jaccard Coefficient is defined as:

$$JC = \frac{a}{a+b+c}, \tag{3.9}$$

is a statistic used for comparing the similarity and diversity of samples. And are defined as the size of intersection over size of union.

Class	B_1	B_2	...	B_j	Sums
A_1	n_{11}	n_{12}	...	n_{1j}	a_1
A_2	n_{21}	n_{22}	...	n_{2j}	a_2
...	...	...	...	...	...
A_i	n_{i1}	n_{i2}	...	n_{ij}	a_i
Sums	b_1	b_2	...	b_j	N

Table 3.1: Contigency table.

3.2.3 Rand index

The Rand index is defined by:

$$RI = \frac{2(a+d)}{m(m-1)} = \frac{a+d}{a+b+c+d}. \tag{3.10}$$

This metric measures the similarity between two clusters C and C^* by measuring the pairwise agreements amount clusters C_i and C_j^*. It is ranged between 0 to 1, with 0 meaning that the results have 0 agreed pair and with 1 meaning that the clusters are exactly the same. But this measurement index have a problem when uses two random clusters, the result index value is bigger than zero and it is variable.

3.2.4 Adjusted rand index

To solve the problem previous mentioned for Rand index, there is proposed a corrected-for-chance version.

Giving a set D of m elements, and two groups of clusters C and C^*, with C = $\{C_1, C_2,...,C_k\}$ and C^* = $\{C_1^*, C_2^*,...,C_k^*\}$, the overlap between C and C^* can summarized in a contingency table $[n_{ij}]$ where each n_{ij} denotes the number of objects that are common in C and C^*: n_{ij} = $\mid C_i \cap C_j^* \mid$

The Adjusted Rand Index is defined as:

$$ARI = \frac{\text{Index} - \text{Expected Index}}{\text{Maximum Index} - \text{Expected Index}} \tag{3.11}$$

$$ARI = \frac{\sum_{ij}\binom{n_{ij}}{2} - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{N}{2}}{\frac{1}{2}[\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}] - [\sum_i \binom{a_i}{2} \sum_j \binom{b_j}{2}]/\binom{n}{2}} \tag{3.12}$$

where n_{ij}, a_i and b_j are values from contingency table 3.1.

3.2.5 Fowlkes and Mallows index

Fowlkes and Mallows Index (FMI) is an external evaluation method that is used to determine the similarity between two clusters. This method was used to measure similarity between two hierarchical clusters or clustering and benchmark classification. Defined by:

$$FMI = \sqrt{\frac{a}{a+b} \cdot \frac{a}{a+c}}. \tag{3.13}$$

All above index are valued between 0 to 1. The greater the better are the result.

Chapter 4

Proposed methodology

The methodology used for this project is present in this chapter. First we will introduce the integrated frameworks used. Then the data used in this work, namely synthetic and real.

4.1 Integrated frameworks

The integrated frameworks used was factor analysis based. Able to deal with multiple heterogeneous data, inferring a set of hidden factors that capture source of variation across multiple data-types. Additionally, the integrated frameworks are so called because they can do other tasks, for example clustering in iCluster and factor analysis in MOFA.

4.1.1 iCluster

Ronglai Sheng [11] has proposed in his original paper in 2009 a framework that includes latent variable model with clustering, the resulting methodology is called iCluster. iCluster incorporates flexible modeling of the associations between different data types and the variance-covariance with data types, while simultaneously reducing the dimension of data-sets in a single model. For the dimension reduction part, it is based on the probabilistic PCA model and extended to multiple data-types. For the cluster, the framework is K-means based as explained below.

Clustering method - eigengene K-means algorithm

The standard K-means algorithm was introduced in chapter 3 with expression described in 4.1, which is sensitive to the choice of starting point and might iterate to the local minima rather than to the global minimum. A better optimization scheme for K-means arises from PCA, proposed by Zha et al [17]. Let $Z = (z_1, ..., z_k)^T$ with k-th row being indicator vector of cluster k and normalized to have unit length:

$$\arg\min_{c} \sum_{i=1}^{k} \sum_{x \in C_i} ||x - \mu_i||_2^2, \tag{4.1}$$

$$z_k^T = (0, ...0, \frac{1}{\sqrt{n_k}}, ..., \frac{1}{\sqrt{n_k}}, 0, ..., 0), \tag{4.2}$$

where n_k is the number of samples in cluster k and $\sum_{k=1}^{K} n_k = n$. With cluster assign matrix Z, the optimal of clustering solution can achieved by minimizing the within-cluster variance. Let XX^T be the

Gram matrix of the sample, then the K-means loss function in 4.1 can be expressed as:

$$\text{trace}(XX^T) - \text{trace}(ZX^TXZ^T), \tag{4.3}$$

which is the total variance minus the between-cluster variance. The total variance is constant given the data, the problem in 4.1 is equivalent to maximizing the between-cluster variance:

$$\max_{ZZ^T=I_k} \text{trace}(ZX^TXZ^T). \tag{4.4}$$

Considering a continuous $Z^\star$ that satisfies all the conditions of Z except for the discrete structure i.e. $z_k^\star$ can take values different of zero or one(achieved by square-root scale in 4.2). Then the problem is equivalent to the eigenvalue decomposition of S. Then the closed-form solution for 4.4 is $\widehat{Z}^\star = E$, where $E = (e_1, ..., e_k)^T$ are the eigenvectors corresponding to the K largest eigenvalues from the eigen-decomposition of S. Ding and He also published in [18] that the redundancy in Z such that the K-means solution can be defined by the first $K-1$ eigenvectors.

Although defining the Z as continuous makes some problem in interpretability of the cluster indicator matrix, but it is necessary to achieve for find closed-form solution for K-means. The interpretability can restored easily by standard K-means algorithm invoked on $Z^\star$. Since we are in genomic data context, the algorithm described is named as eigengene K-means[11].

Gaussian latent variable model

Consider again a Gaussian latent variable model representation of the eigenvalue K-means clustering:

$$Y = WZ + \epsilon. \tag{4.5}$$

The formulation is similar than 2.5, where Y is mean-centered in this case.Different from normal formulation, $Z = (z_1, ..., z_k)^T$ is the cluster indicator matrix of dimension $(K-1) \times n$ as previous defined. W is the coefficient matrix of dimension $p \times (K-1)$, and ϵ is a error term with zero mean and a diagonal covariance matrix Ψ = diag(Ψ_1,...,Ψ_p). Considering a continuous parameterization $Z^\star$ of Z and additional assumption that $Z^\star \sim$ N(0,I) and $\epsilon \sim$ N(0,Ψ). Then the K-means problem with likelihood-based solution is available through model (4.5). The parameters inference will be based on the posterior mean of $Z^\star$ given the data and the inference method are already presented in the section Probabilistic PCA.

Joint latent variable model

Since the objective is studying biological genomic data, there exist more than one type of data to explain the under-layer disease and sub-types of patient. So we need generalize the formulation present in 4.5 to multi-omic formulation. This means estimating matrix of indicator $Z = (z_1, ..., z_{K-1})$ by,

$$\begin{aligned} Y_1 &= W_1Z + \epsilon_1 \\ Y_2 &= W_2Z + \epsilon_2 \\ &\vdots \\ Y_m &= W_mZ + \epsilon_m, \end{aligned}$$

where m is the number of genomic data type available from the same set of samples. $Z \sim$ N(0,I) is common for all data types, $\epsilon_i \sim$ N(0,Ψ) is error term that is independent from each other and W_i is a coefficient matrix. Then the marginal distribution data matrix $Y_1, ..., Y_m$ are multivariate normal with mean zero and covariance matrix $C = WW^T + \Psi$, the corresponding log-likelihood function of the data is(equal as definied in equation 2.17),

$$l(W, C) = -\frac{1}{2}(\sum_{i=1}^{m} p_i \ln(2\pi) + \ln|C| + tr(C^{-1}S)). \tag{4.6}$$

The parameter inference is obtained by EM algorithm to obtain the maximum likelihood estimates of W and Ψ, dealing with complete-data log-likelihood,

$$l_c(W, \Psi) = -\frac{n}{2}\Big\{ \sum_{i=1^m} p_i \ln(2\pi) + \ln|\Psi| \Big\} - \frac{1}{2}\Big\{ tr((X - WZ^\star)^T \Psi^{-1}(X - WZ^\star)) + tr(Z^{\star T} Z^\star) \Big\}, \tag{4.7}$$

which is more efficient than maximizing directly the marginal data likelihood.

A sparse solution

The framework is applied to the biological data sets, which normally the number of features p are much bigger than number of samples n. In this cases, the regularization method is important. The sparse solution is applied to W, and the complete data-likelihood with sparse solution is,

$$l_{c,p}(W, \Psi) = l_c(W, \Psi) - J_\lambda(W), \tag{4.8}$$

where J_λ is a penalty term on W with non negative regularization parameter λ. Authors of iCluster adopted a lasso penalty, which takes the form,

$$J_\lambda(W) = \lambda \sum_{i=1}^{m} \sum_{k=1}^{K-1} \sum_{j=1}^{p_i} |w_{ikj}|. \tag{4.9}$$

4.1.2 MOFA

Multi-Omics Factor Analysis(MOFA) [10] was proposed by R. Argelaguet et al. at 2018, a computational method for discovering the principal sources of variation in multi-omics data-set. This method can infer a set of hidden factors across various types of measures. The result learnt factors can be used for downstream analysis.

Considering we have M data matrices Y^1,..,Y^M of dimensions $N \times D_m$, where N is the number of samples and D_m the number of features in data matrix m. The objective of MOFA is try find hidden factor matrix Z(common for all data matrices) such that,

$$Y^m = ZW^{mT} + \epsilon^m \tag{4.10}$$

where m = 1,..,M, W^m denotes weight matrix for each data matrix m and ϵ^m denotes error term for each data matrix, it's depend on data type. MOFA is formulated in the probabilistic Bayesian framework, proposing prior distributions on all unobserved variables, i.e. for Z, W and ϵ.

Starting by assuming ϵ^m to be Gaussian like (similar in standard factor analysis model), while allowing

heteroscedasticity across features, we get:

$$p(\epsilon_d^m) = N(\epsilon_d^m|0, 1/\tau_d^m). \tag{4.11}$$

With previous assumptions, we get:

$$p(y_{nd}^m) = N(y_{nd}^m|z_{n,:}w_{d,:}^{mT}, 1/r_d^m) \tag{4.12}$$

where $w_{d,:}^m$ denotes the d-th row of the loading matrix W^m and $z_{n,:}$ the n-th row of the latent factor matrix Z. For probabilistic treatments, MOFA assumes prior distributions for the weights W^m, the latent factors Z and error term τ^m. It assumes standard Gaussian prior for latent factors and conjugate Gamma for error:

$$p(z_{n,k}) = N(z_{n,k}|0, 1) \tag{4.13}$$

$$p(\tau_d^m) = \Gamma(\tau_d^m|a_0^\tau, b_0^\tau) \tag{4.14}$$

Model regularization

While working with biological high dimensional data sets, it's essential to have sparse and interpret able results for posterior analysis. And MOFA was created under this acknowledge, having two types of regularization on weights W^m: a view- and factor-wise sparsity and a feature-wise sparsity. The view- and factor-wise allows to directly identify which factor is active in which view, and feature-wise sparsity puts zero weights on individual features from active factors.

This regularization is achieved by putting appropriate priors on the weight matrices. MOFA uses Automatic Relevance Determination (ARD) prior for view- and factor-wise sparsity and spike-and-slab prior for feature-wise sparsity.

The spike-and-slab prior contains Dirac delta functions,

$$p(\omega) = (1 - \theta)\delta(\omega) + \theta N(\omega|0, 1/\alpha) \tag{4.15}$$

which is problematic for inference. For solving this problem is required to re-parameterize weights w as a product of a Gaussian random variable $\hat{w}$ with Bernoulli random variable s, resulting following prior:

$$p(\hat{w}_{d,k}^m, s_{d,k}^m) = N(\hat{w}_{d,k}^m|0, 1/\alpha_k^m)Ber(s_{d,k}^m|\theta_k^m) \tag{4.16}$$

which α_k^m controls strength of factor k in view m and θ_k^m controls contribution of spike term affecting feature-wise sparsity of factor k in view m. For automatically learning these parameters, it assumes following priors:

$$p(\theta_k^m) = Beta(\theta_k^m|a_0^\theta, b_o^\theta) \tag{4.17}$$

$$p(\alpha_k^m) = \Gamma(\alpha_k^m|a_0^\alpha, b_0^\alpha) \tag{4.18}$$

with hyper-parameters a_0^θ and b_0^θ = 1 and a_0^α, b_0^α = $1e^{-14}$ to get uninformative priors. A value of θ_k^m close to 0 implies that most of the weights of factor k in view m is shrunk to 0.

The ARD prior yields a matrix α with dimensions $M \times K$ that defines four different types of factors [10]:

- Factors that do not explain variation in any data set (inactive factors): all values in the corresponding columns of α are large. These factors are actively removed from the model during training.
- Factors that explain variation in all data sets (fully shared factors): all M values in the corresponding columns of α are small.
- Factors that explain variation in a single data set (unique factors): all values in the corresponding columns of α are very large, except one.
- Factors that explain variation in a subset of data sets (partially shared factors): some values in the corresponding columns of α are very large whereas others are small.

the previous prior distributions, the joint probability density function is given by [10]:

$$\begin{aligned} p(Y, \hat{W}, S, Z, \Theta, \alpha, \tau) = & \prod_{m=1}^{M} \prod_{n=1}^{N} \prod_{d=1}^{D_m} N\Big(y_{nd}^m | \sum_{1}^{K} s_{dk}^m \hat{w}_{dk}^m z_{nk}, 1/\tau_d\Big) \\ & \prod_{m=1}^{M} \prod_{d=1}^{D_m} \prod_{k=1}^{K} N\Big(\hat{w}_{dk}^m | 0, 1/\alpha_k^m\Big) Ber\Big(s_{d,k}^m | \theta_k^m\Big) \\ & \prod_{n=1}^{N} \prod_{k=1}^{K} N\Big(z_{nk} | 0, 1\Big) \\ & \prod_{m=1}^{M} \prod_{k=1}^{K} Beta\Big(\theta_k^m | a_0^\theta, b_0^\theta\Big) \\ & \prod_{m=1}^{M} \prod_{k=1}^{K} G\Big(a_k^m | a_0^\alpha, b_0^\alpha\Big) \\ & \prod_{n=1}^{N} \prod_{k=1}^{K} G\Big(\tau_d^m | a_0^\tau, b_0^\tau\Big) \end{aligned} \tag{4.19}$$

And the graphic representation of model is illustrated in 4.1.

Parameter inference

For inference of parameters, MOFA use a variational Bayesian framework, which is essentially a mean field approximation for approximate inference[19]. The key idea is to approximate the intractable posterior distribution using a simpler class of distributions by minimizing the Kullback–Leibler divergence to the exact posterior, or equivalently maximizing the Evidence Lower Bound (ELBO). Convergence of the algorithm can be monitored based on the ELBO. More details can be found on [2].

Method

MOFA framework was available in [20]. For using it is need to have data sets in the required format and set the training options for run it. The framework starts with a established number of latent factors and can drop out according to a threshold the factor that do not have significant explained variance of original data.

MOFA uses a iterative approximations to infer the parameters and do not have a global best solution, so is necessary to run the model multiple times and select the best trained model. The model selection is based on the ELBO value, do not take account on the number of latent factors obtained. After having best model, it is possible to analyze directly the latent factors. Then was used K-means clustering algorithm to the obtained results for finding groups of samples.

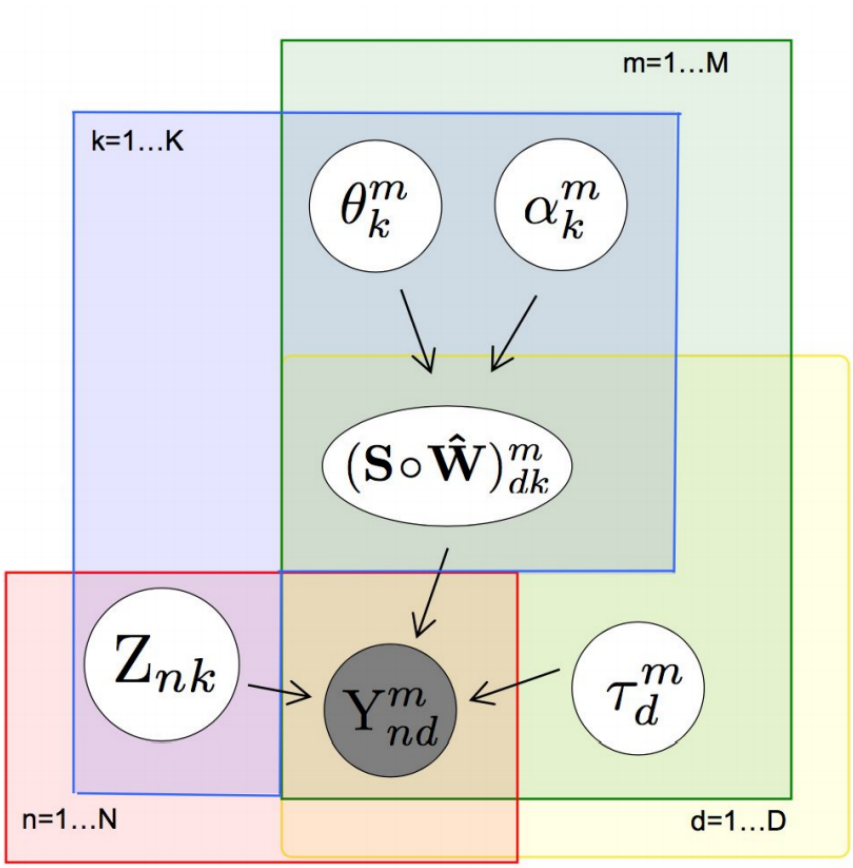


Figure 4.1: Graphical model representation of MOFA(taken from[2]).

4.2 Data

Before using the previous introduced frameworks on real data, was generated synthetic data for validating and testing the viability of methods.

4.2.1 Synthetic data

For validating the reliability of iCluster and MOFA, is essential to have some simpler and easier interpretable results. For this propose are used a generated data-set for testing and interpret them. Once we want grouping patients into subcategories based on the reduced features, the validation data-set should also have this characteristic. The generation of synthetic data are proceeded by the following steps. First of all, is generated a latent variable matrix Z using python sklearn package[21], with predefined number of clusters, samples and features. Then is transformed into multi-omic and higher dimensional space according to the number of views and features selected. Then is added the error term and transformed to the corresponding distribution(Gaussian, Poisson and Bernoulli).

Toy example

We use the following toy example to explain in detail the synthetic data generation. Using sklearn package to generate 50 samples with 2 latent factors distributed along 4 centers with standard deviation equals to 1 as illustrated in Figure 5.1a.

Then is applied a transformation to Z such that the original data is transformed into a Gaussian 3-view data set, each view with 20 features as illustrated in Figure 4.2.

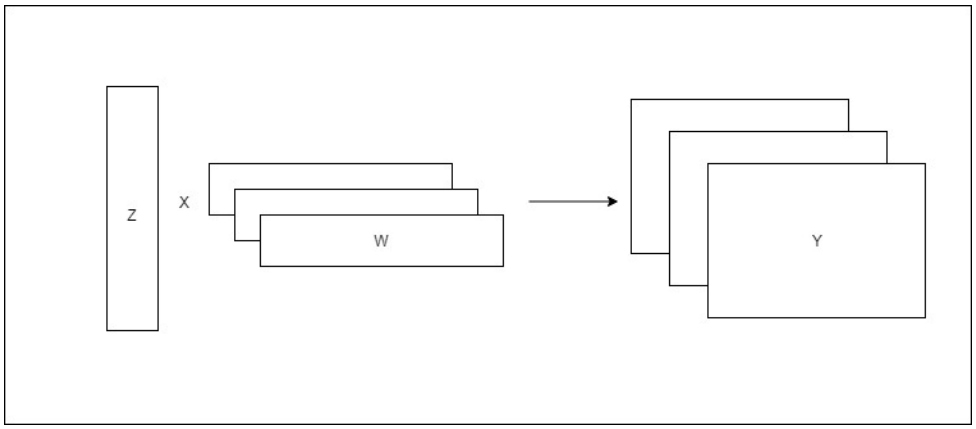


Figure 4.2: Transformation of synthetic data in latent space to multi-omics space.

The transformation is done by generating 3 matrix of W with random normal distribution $N \sim$ N(0, $\frac{1}{\sqrt{\alpha}}$), with α as a random $k \times n$ matrix with value 1 or 1000. For simulating a sparse solution the weight matrix W is multiplied by a $S \sim B(p = 0,5)$ matrix to simulates that a latent factor are affect only by some features. Depending which type of distribution is needed, is applying a transformation to ZW.

$$\tilde{Y} = log(1 + e^{ZW}), \tag{4.20}$$

for Poisson distribution data and,

$$\tilde{Y} = \frac{1}{1 + e^{ZW}}, \tag{4.21}$$

for the binomial distribution.

Then the result is obtained by

$$Y = \tilde{Y} + \epsilon. \tag{4.22}$$

which the ϵ corresponds the noise term, the distribution is different depending in the predefined data distribution.

The resulted Y is object for methodology validation.

Set of synthetic data

For more general view, will create a big set of synthetic data with different values of parameters for testing the performance of frameworks according to the dimension size of parameters. The synthetic data for this purpose will have following parameters depending on Z or Y. For the Z, we need to consider the number of test samples, number of latent factors, number of cluster and number of cluster's standard derivation values. And for simulate the visible variables Y, it need to take account of number of samples, number of visible features, number of latent variables, number of clusters and number of types of data. The works turns too complex if we test the performance changing all the mentioned parameters, so for simplicity we only choose some of them that was more important for testing, and let the others to be constant.

We choose the samples size as 100 patients, with number of latent factors lf = [3, 5, 10, 15], the number of cluster k=[2, 3, 4, 5, 6, 7, 8, 9], the standard derivation of each cluster as STD = [0.25, 0.5, 0.75, 1.0, 1.25, 1.5, 1.75, 2.0] for generating testing data Z. And for Y, the number of samples are same, from each number of latent factors in Z generates features with dimensions of F = [20, 50, 100, 500, 1000] with 3 types data with Gaussian distribution. The data generation of Z is same way as explained in the toy example.

In total there was generated 1280 samples, resulted from combination of 4 sizes of latent factors, 8 sizes of cluster numbers, 8 sizes of standard variation and 4 types of features size as illustrated in Figure 4.3. All samples was run in the MOFA framework, and only some of then was run with iCluster due to running time problem.

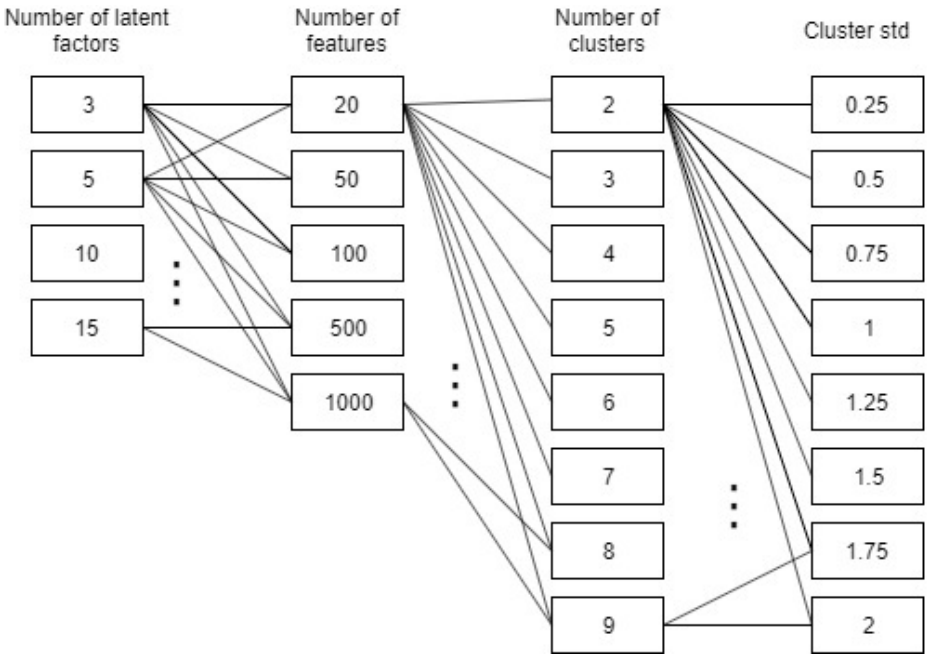


Figure 4.3: Values and 1280 combinations of testing sample parameters.

4.2.2 Real biological data

The biological data selected by this project are related with cancer disease. There exist more than a hundred types of cancer, most of them are caused by mutations on the DNA, bad regularization system, inability to restore the anomalies of DNA resulting uncontrollable multiplication of cells in the body. The Cancer Genome Atlas (TCGA) [22] database is tested in this project, a landmark genomics program involved with over 20000 primary cancer samples across 33 types of cancer, generated over 2.5 petabytes of genomic, epigenomic, transcriptomic, and proteomic data.

OV data-set For this study, are only selected DNA methylation, RNA sequence and mutation types of Ovarian Carcinoma (OV) cancer data from TCGA. The DNA methylation data was collect by Illumina Human Methylation 450k. The RNA-seq have 56830 features and 379 patients, and data are represented with counts and Fragments Per Kilobase per Million (FPKM), for statistical reason we select the FPKM representation. The DNA methylation have 25978 features and 602 patients. And for mutation data we have single cell experiment with 13536 features with 432 patients.

Gene codification When dealing with bioinformatics data, such as RNA-seq data, chip data, or data downloaded from the TCGA database. This data features are encrypted with variety of IDs, these IDs are like the ID cards of each of us, they are used to identify genes. Since IDs come from different databases, or the intent of naming is different, there are always multiple different IDs for the same gene, such as Entrze ID, Ensembl ID, HGNC ID, refseq ID, etc.

The Entrze ID is a gene identifier used in the NCBI database, usually represented by a pure number. For example, the Entrze ID of the human TP53 gene is 7157 (note that the gene IDs of different species are different). The Ensembl ID is codification used in the genes of EMBL(European Bioinformatics Institute) database. The identifiers are all started with four uppercase letters of ENSG (ensembl gene), followed by 11 digits, so the length of the Ensembl ID is usually 15 digits. For example, the Ensembl ID of the human TP53 gene is ENSG00000141510. Noting that the Ensembl ID It contains not only more than 20,000 protein-coding genes, but also many pseudo-genes, miRNAs, etc., so it has more than 60,000 IDs, which is much more than the number of genes known to humans. HGNC ID refers to The gene identifier specified by the HUGO Gene Nomenclature Committee. The committee usually assigns a name to the gene and an ID, such as the human TP53 gene. The standard symbol is TP53, equivalent to the abbreviation of the standard name Tumor Protein p53 and HGNC ID is 11998. Refseq is a gene standard sequence (reference sequence) database provided by NCBI in the United States, in which the ID of the human TP53 gene is NG_017013.

Data pre-processing In the RNA-seq data-set, there exist two types of cancer, primary with 374 patients and recurrent with 5 patients, for simplicity this 5 patients samples are ignored. In the set of 56830 genes, only are selected subset of 19941 variables corresponding to the protein-coding genes reported from Ensemble genome browser [23] and the Consensus CDS Project [24]. This pre-selection is based that protein-code genes are more important in the study of cancer.

This data-set is not perfect, the patients in 3 types of data are different, for simplicity is only select the intersection information across all data, resulting a set with 269 patients.

Then is used a variance filter to select the most variate features. Resulting 1500 features for RNA-seq, 1000 for DNA methylation and 500 for mutations.

The initial mutation matrix are composed by counts in each gene, but for this work, it was changed to binary matrix.

4.3 Proposed pipeline

After introducing the frameworks and data used in this job, this section will introduce the full pipeline.

First was used the toy example for explain in detail the way was generated the synthetic data. Choosing the best method for determining the number of cluster, which is Silhouette method. This method will be used for automatically choosing the best number of cluster in next simulations.

Then the set of 1280 test samples will be used for analysing the MOFA and 768 test samples for iCluster, the samples with feature dimension 500 and 1000 wasn't tested in the iCluster due to training time problem.

After analysing the results of with synthetic data to get performance of frameworks, the best one will be used for OV data analysis. iCluster was actually used too, but due to high dimensional and software problem, the tests was failed.

In the real data, the variables are genes, after obtain the results, is possible to analyse the gene functionalities from other studies.

In the general way, the result pipeline is illustrated in Figure 4.4.

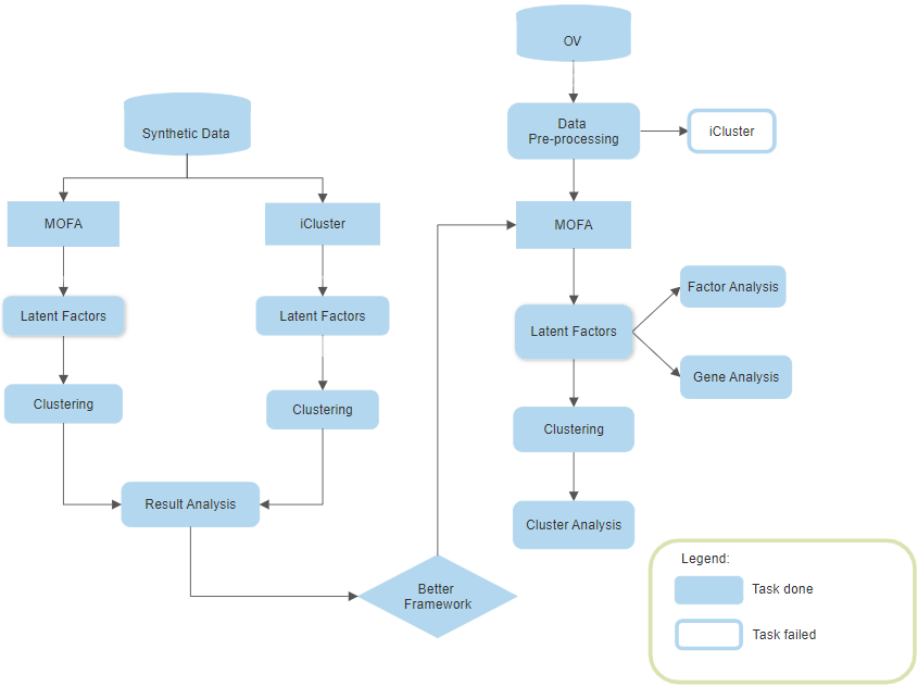


Figure 4.4: Flowchart of methodology.

Chapter 5

Results

This part will present the results that was obtained from using the method described in the previous chapter. This chapter contains two sections, the first one is result from applying methodology to the synthetic data and the second one was used real data-set.

5.1 Synthetic data results

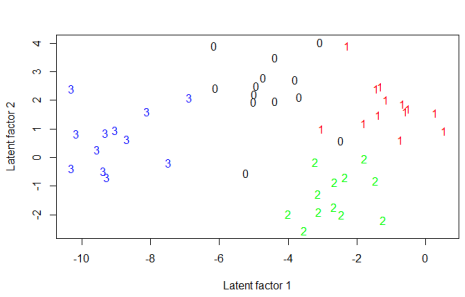
As mentioned before, the synthetic data was used for validating the availability of frameworks, so is essential to have some simple and easy interpret-able results. This section presents the result obtained from Synthetic data.

Toy example results

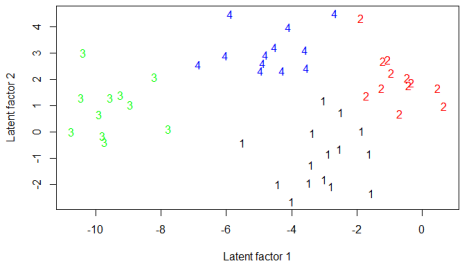
The original cluster of toy example are present in Figure 5.1a, in Figure 5.1b shows the clusters resulted using K-means on original data, is possible to visualize that without any transformation the clusters result have wrong classification on some points. This was created on purpose for having average value standard deviation of points to induce misclassification, because in real data the dispersion of points is large. In Figure 5.1 shows the cluster points plotted in the latent space found by MOFA, the way for achieving this result was explained later. It is possible to visualize that the recovered space is very similar that then original one, the cluster's position are also similar, only the points have some deviation.

Because of simplicity of data, there is no need for pre-processing toy example before using it in the frameworks.

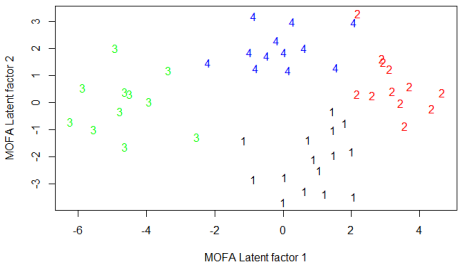
Using MOFA For MOFA, was chosen default parameters to train the data set except the DropFactorThreshold. This was used to drop out the factors that have low variance explained for automatically left only the most explainable factors. Then run the framework 25 times for choosing the best model based on the ELBO value, the models can have different number of factors. After obtained the best model, we got results for posterior analysis and clustering. For this example, was obtained 2 latent factors as illustrated in Figure 5.2, which $R2$ represents the variance captured by each factor and view represents the data types. From analysing the variance explained per factor graphic, we can observe that majority of information present across 3 data types was captured only by 2 factors. From graphic below, the view_3, which represents the data type 3 is almost explained only by the latent factor number 2(LF2), and view_1 and view_3 was captured by LF1. And from total variance explained graphic shows



(a) Original labels of data set.



(b) Labels obtained using K-means on original latent space.



(c) Labels obtained using K-means on MOFA's latent space.

Figure 5.1: Cluster's label comparison with K-means algorithm in original data points and in latent space found by MOFA.

that the information of view_3 and view_1 was mostly recovered and the view_2 only retains around 25% of variance.

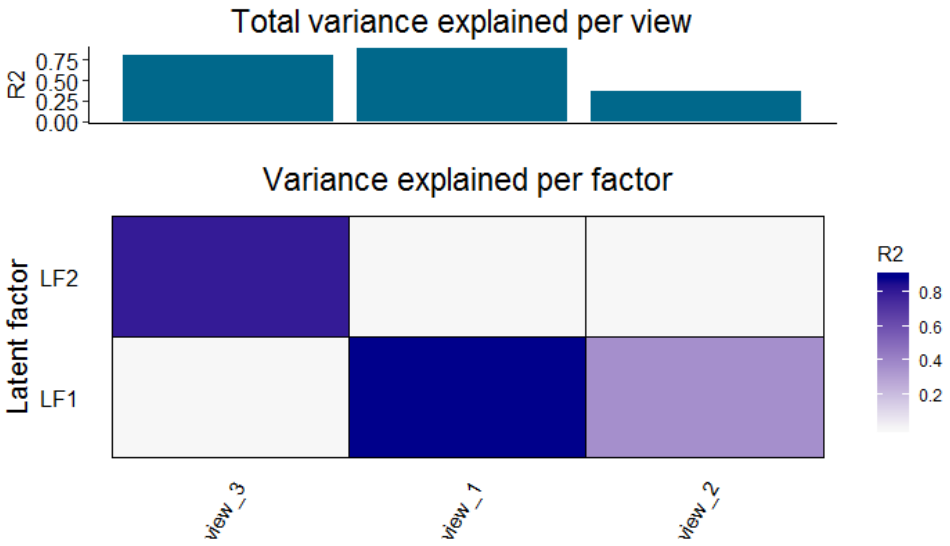
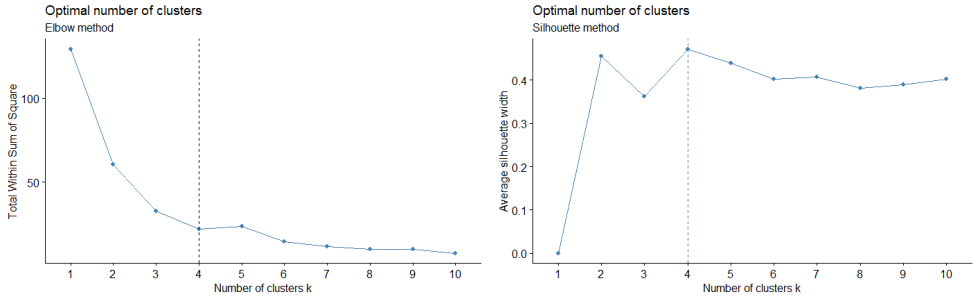


Figure 5.2: Latent factors obtained from MOFA.

Then is applied K-means clustering to the obtained latent factors. For determining the best k for this data set, was used 3 different way as showed in Figure 5.3. In Figure 5.3a is illustrated the k selection using elbow method, that suggests 4 clusters. In Figure 5.3b is though Silhouette, that suggests 4 clusters and in Figure 5.3c is using Gap statistic that suggest 3 clusters.

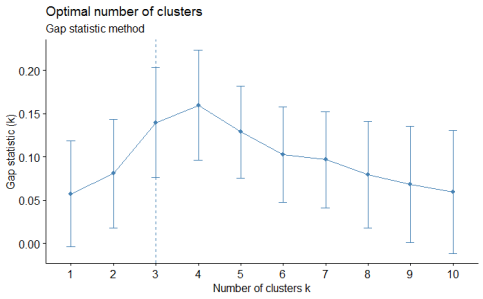
In this case, 2 of 3 methodologies suggests $k = 4$, then for the cluster analysis was assumed to have

4 clusters. The plot of cluster samples are present in Figure 5.1c.



(a) k determination by Elbow method.

(b) k determination by Silhouette method.



(c) k determination by Gap statistic method.

Figure 5.3: Different methods for determining number of cluster k on latent factors obtained by using MOFA.

Using iCluster iCluster framework fits a regularized latent factor variable model based clustering that generates an integrated cluster assignment based on joint inference across data types. The method was available in bioconductor with package name "iClusterPLus"[25]. The framework allows to set number of latent factors and number of clusters k, because the framework is subject to have one less dimension in latent space (latent space with dimension k-1). And value of parameter set lambda(a parameter set to control how many genomics features have non-zero weights on the latent factor). We have to optimize these two parameters. The number of lambda points to sample depends on the number of data types and can take the values present in Table 5.1. Toy example have data types, was chose 185 lambda values for training the data.

Number of data types	*Number of lambda*
1	any
2	8, 13, 21, 34, 55, 89, 144, 233, 377, 610
3	35, 101, 135, 185, 266, 418, 597, 828, 1010
4	307, 562, 701, 1019, 2129, 3001, 4001, 5003, 6007

Table 5.1: Number of lambda values to fit iCluster, table from iCluster manual[1].

For each k, the lambda selection criteria is based on Bayesian Information Criteria (BIC). To choose the best k, iCluster uses $\%$ explained variation. The result is illustrated in Figure 5.4, then is chosen $k = 3$ as the best value for the posterior analysis. By selecting 3 clusters, the number of latent factors are automatically fixed as 2.

Cluster result evaluation After getting cluster results of samples in MOFA's latent space and iCluster's latent space, it allows to evaluate the performance of both frameworks by comparing the real label and

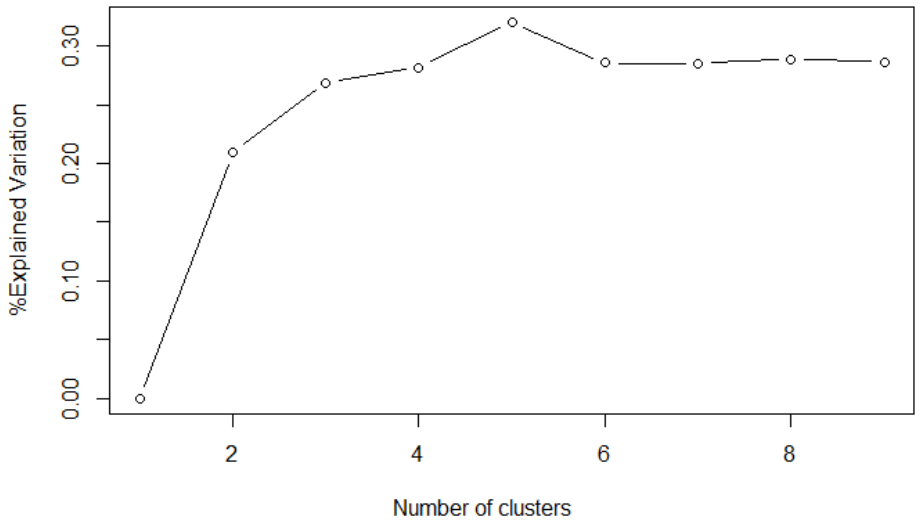


Figure 5.4: Number of cluster vs percent of variance explained.

labels obtained. In Table 5.2 shows the values of index that was introduced in Chapter 3.

Source of LF	*Precision*	*Recall*	*Jaccard*	*Rand*	*Folkes Mallows*
Original	0.84375	0.8321918	0.7210683	0.9232653	0.8379509
MOFA	0.8402778	0.8373702	0.7223881	0.9240816	0.8388228
iCluster	0.9583333	0.6216216	0.6052632	0.8530612	0.7718295

Table 5.2: Cluster index of toy example.

The first row of table presents the index values obtained by comparing the real label with labels obtained by applying K-means in the original latent space. The values are not perfect, because the sample in example are not perfect cluster-able by K-means.

The second entry refers index values obtained by comparing real labels with labels provided by K-means in MOFA's latent space.

And the last entry refers index values come from iCluster's latent space.

From visualizing the results on Figure 5.1 and Table 5.2, MOFA present better results in toy example. Is able to recover majority of cluster labels, the samples localization on latent space and the precision, recall, Jaccard, Rand and Folkes Mallows index values are very close to index values obtained by applying K-means directly to the original space. For the iCluster, the restriction of relation between number of cluster and latent space dimension makes the framework have worse performance almost in all indexes except in precision, but this maybe dues to have less cluster and this gain is paid by low recall value.

Set of synthetic data results

Next are present results obtained from testing the 1280 synthetic samples created as described in chapter 4.

Cluster's STD effect First of all, we want to study how the cluster's STD affects the model performance. And the result are showed in Figure 5.5. The figures shows how the 5 clustering index vary

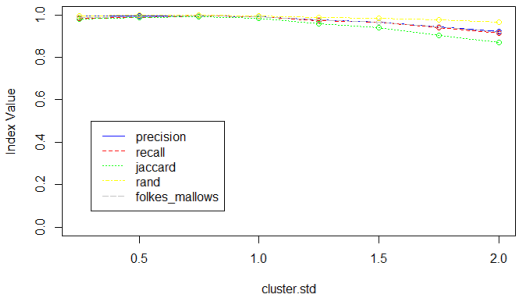
when the clustering standard deviation changes. Was tested with 160 data-sets with different parameters for each STD value to produce mean for analysis.

In Figure 5.5a shows the index values resulted by comparing the real labels with label resulted by applying K-means on original latent factors, this figure can show how the data points are dispersed. For standard derivation less than 1.0, the problem is trivial on original space. Almost all index have high values, K-means can group the correct clusters.When STD increases, the misclassification error also increases due to the points of different groups are intercepting.

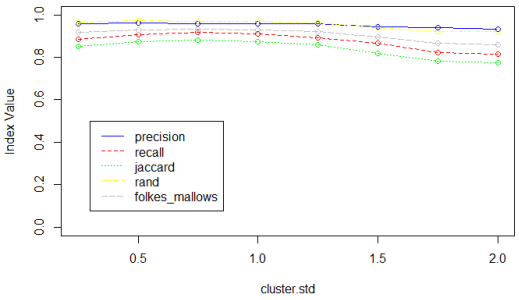
In second Figure 5.5b shows the index values come from the comparison between real labels of samples and labels resulted by K-means applied to the latent factors resulted using MOFA with Silhouette as selection criteria for find k. The index values have best performance on STD values between 0.5-1.0. Theoretically, small STD values of clusters will get better index values, and this didn't happen may due to space transformations on data-set.

In figure 5.5c is similar than previous one, but the k selection criteria is elbow method. From comparing this graphic with Figure 5.5b, the results are worse in almost all index except in precision. The index values doesn't change too much along different STD values. Beyond analysing the STD effect, We also can conclude that, for this type of data, the number of cluster obtained by Silhouette method is better than elbow method.

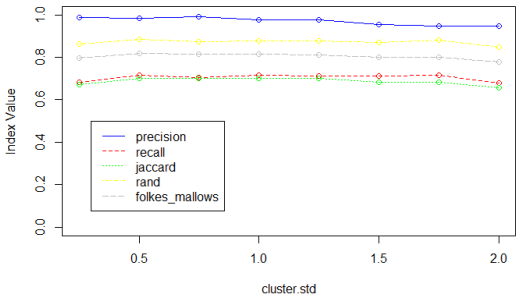
In Figure 5.5d are presented clustering index obtained by applying the iCluster framework. For this case, the samples used to obtain the result are less than others cases due to time cost problem for training medium and big size data. The index values decreases when the original data points dispersion increases.



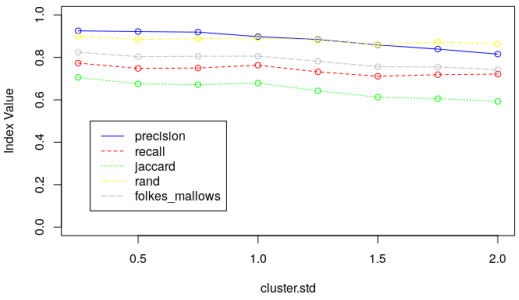
(a) Index values using original factors.



(b) Index values using MOFA's factors by Silhouette.



(c) Index values using MOFA's factors by elbow.



(d) Index values using iCluster's factors by elbow.

Figure 5.5: The cluster's STD effect on index values.

Original LF recovering It was also tested the relation between the number of original latent factors with the number of features.

1. MOFA

   The result are illustrated in Figure 5.6 for MOFA and more detail can be found in Table 5.3. The table and graphic represent the same result. In the table, the row entries represents the original latent space dimension of samples, and column entries refers the dimension of result 3 data type dimension, for simplicity all the data types have same dimension. Are testing 64 samples for each entry in table, in entry of tables the #LF represent the dimension of latent space recovered and #Obs represents the how many tests achieved this dimension. From analysing the table, we can see that the framework can recover the original dimension of test samples when the original dimension are 3 and 5. When the original dimension increases, the result turns bad, the possible number of dimension recovered turns dispersed as shows the column when original #LF are 15, the test have dimension from 1 to 11 or more.

   The cause possibly comes from that the increasing of complexity of data and the original representation can also be simplified and the dimension can be reduced.

   In Figure 5.6, the x axis represents each table entry in Table 5.3, composed by 64 tests. And the graphic represents the distribution along recovered space of LF. Which can be visualized that the tests with original space 3 and 5 have high density concentrated in 3 and 5 respectively.

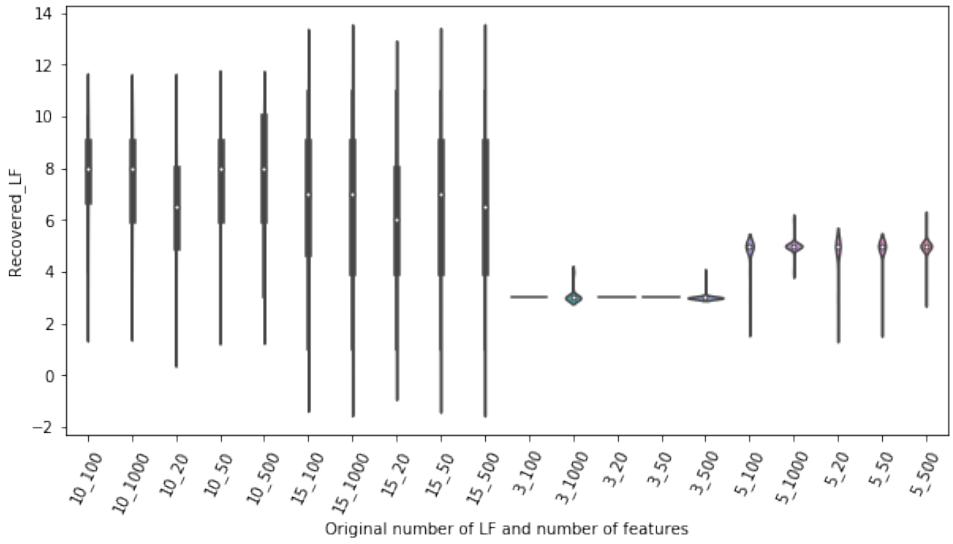


Figure 5.6: Effects of feature number on latent numbers with MOFA in graphic representation. In the x axis, the first element represents original #LF, and second element is dimension of samples in feature space.

2. iCluster

   The same analysis is applied to iCluster, the results are present in Figure 5.7 and in Table 5.4. For the iCluster case, the sample's feature dimension only used are 20, 50 and 100. As previously mentioned, the iCluster takes too much time for running the samples with high dimension.

   Differently than the MOFA's results, iCluster also have worse performance when the original latent space is reduced. For number of original LF equals to 3, the results are distributed along various values, more than half of cases don't hit the correct solution, but general view, the solutions have precision. For other cases, the solution is very dispersed, the same conclusion can be visualized on the Figure 5.7, the density was concentrated on the values between 2-4.

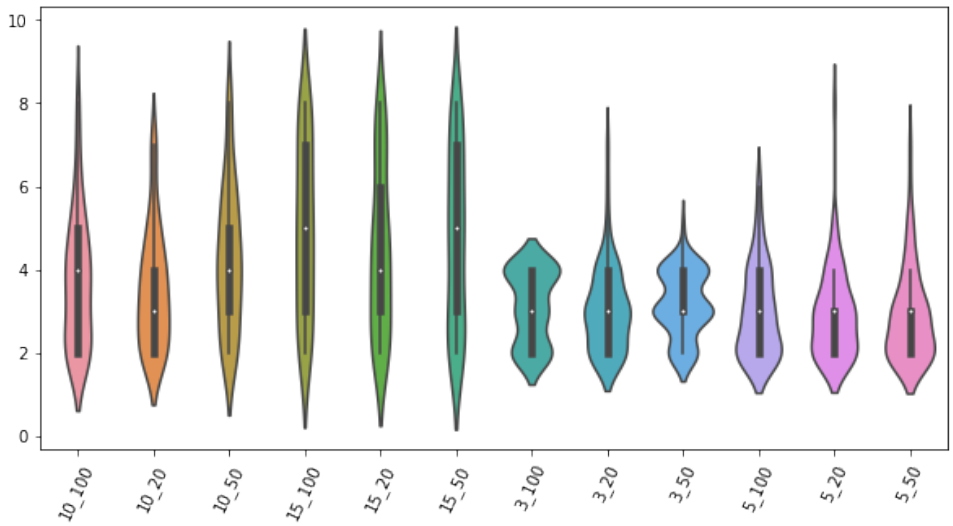


Figure 5.7: Effects of feature number for finding latent space dimension by iCluster in graphic representation.

Recovering number of cluster

Figure 5.8 and Table 5.5 presents the influence of number of factors in terms of recovering the number of cluster in the data samples. In table, the columns represents the number of dimension in original latent space, rows refers the real number of clusters in test sample and the total column represents the total observations in each k found in total 160 test across all $\#$LF. For this case, each entry is composed by 40 tests. The test samples contains samples with 2-9 clusters, and the objective is find small number of clusters, for this reason, the search space for number of cluster is limited to 10. For samples with original k equals 2, 3 and 4, MOFA can find exact number of cluster more than half of cases. From Figure 5.8 also can be visualized that the first elements have density concentrated in the correct number of k, after original number of k are greater or equal to 5, starts the dispersion of k found across various values.

The results point that the framework have high accuracy and average precision for finding the exact number of clusters. The results also shows that the number of latent factors doesn't affects too much in the task for finding a correct number of clusters k. In the each entry was incorporated tests with different number of features, this means that this result ignores the influence of feature dimension in this task.

For the iCluster's case, the framework have the constrain in the problem formulation, that imposes the relation number of cluster k equal number of latent variable plus one.

5.1.1 Time consumption

Table 5.6 presents approximate time interval used for training each dimension of data. For MOFA, it runs 10 random initialization and iCluster uses 135 lambda values with 10 initialization. For iCluster, the time consumption is large because the framework trains one model per each value of lambda and for each value of k, number of clusters.

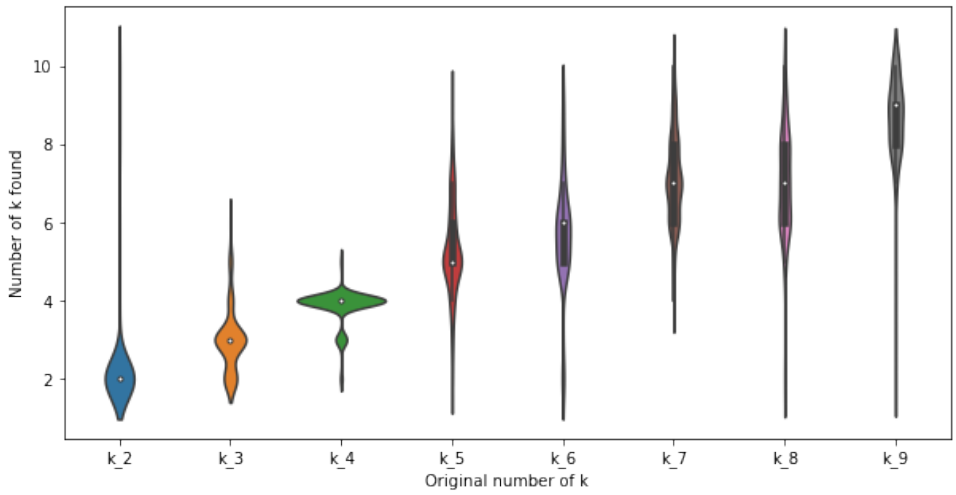


Figure 5.8: Graphic of experiment for testing the effect of number of latent factors in number of clusters using MOFA.

Feature dimension	*iCluster*	*MOFA*
20	5-7min	$<$ 1min
50	15-45min	1-2min
100	1-3h	1-3min
500	5-20h	2-5min
1000	22-40h	15-25min

Table 5.6: Training time consumption.

Result analysis From the previous results, it shows that generally MOFA presents better results than iCluster. MOFA is able to apply higher dimensional data within acceptable time thanks to fewer parameters to test , to choose and better optimization algorithm, can get the latent factors and clusters separately, ability to deal with missing value samples.

5.2 Real biological data results

After testing these two frameworks with synthetic data for studying the availability of finding the latent factors and clusters, this part will use the framework for real data-set analysis.

As described in chapter 4 , was used OV data from TCGA database for analysis. The data processing was already done, now we run the data with MOFA framework. For train the model, most of parameters are the default ones. There was used 25 latent factors to start, DropFactorThreshold as 0.02 for drop out the factors that explain less 0.02 variance in all data types. The maximum iterations for train the model are 4000 and the stop parameter ELBO difference value was set to 0.2.

The problem is complex, the framework can not find the global optimal solution for this problem. So the best model of 25 random initialization was used for later analysis based on the ELBO value.

Latent factor analysis

With this data-set the best model have only 3 latent variables, the variance captured and explained by each factor are present in Figure 5.9.

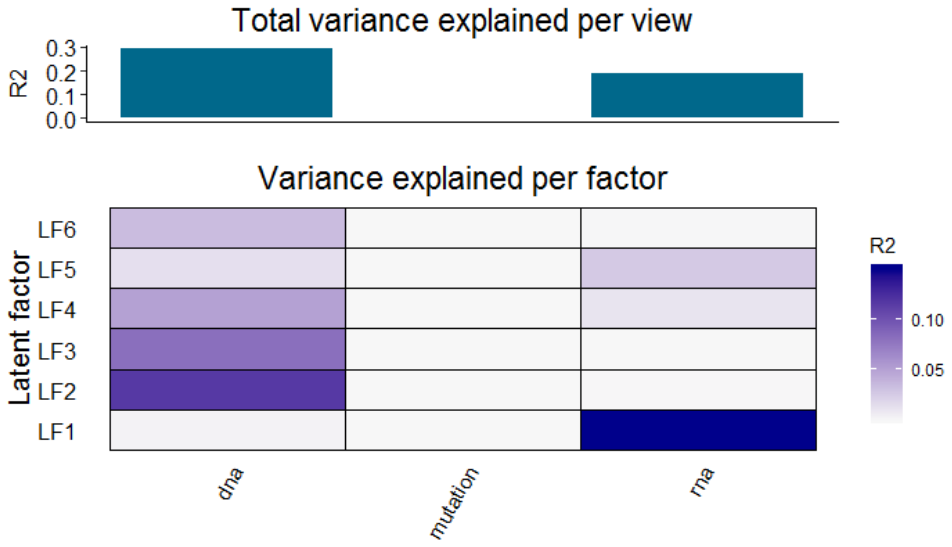
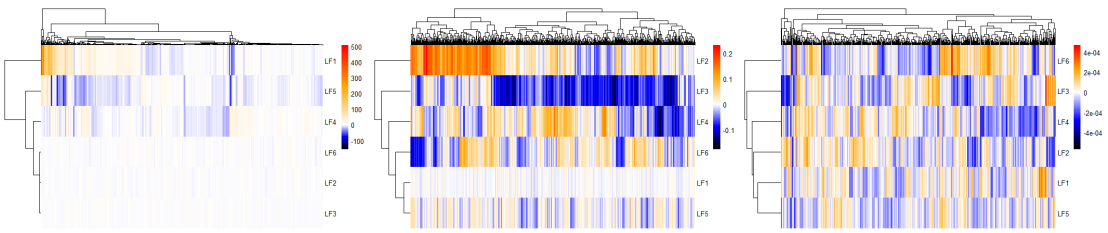


Figure 5.9: Variance explained by each factor.

From analysing the figure, we can see that the DNA methylation information is the most captured one, was retained about 0.30 variance of total variance, then the RNA-seq with approximately 0.20 and for mutation data is show that the variance captured is almost null. This is happens because the mutation data is a very sparse binomial matrix, in this case the way to calculate the variance captured is not very effective.

In detail, we can see that the DNA methylation information is mostly present in LF2, LF3, LF4 and LF5. RNA-seq information in LF1 and LF5.

After obtain the factors, we can do further analysis with them. Each factor are composed by linear combination of feature weights in sparse space which are plotted in Figure 5.10a.



(a) Heat-map plot of weights in mRNA.

(b) Heat-map plot of weights in DNA.

(c) Heat-map plot of weights in mutation.

Figure 5.10: Heat-map plot of weights in each type of data.

We can see from figure, that RNA-seq information is highly concentrated in few features, that makes the weight of each feature having big values comparing with other data. In DNA methylation, LF2 is positive correlated with a group of features and LF3 is composed by features with negative correlation. For mutation data, the weight matrix have very small weights and the features don't form a clear group.

Gene analysis

In order to understand the functionalities of each factor, there was selected top 15 features that are most important in each type of data in Table 5.7. The RNA-seq data and mutation was converted from Ensemble ID to standard symbol for better interpretation.

LF1			LF2			LF3		
mRNA	DNA m.	Mutation	mRNA	DNA m.	Mutation	mRNA	DNA m.	Mutation
TMSB10	cg03668539	TENM1	TIMP2	cg21312148	VCAN	WNT11	cg13603171	LRRC7
RPL18A	cg05406101	LAMA3	NDUFB4	cg20676475	CDH10	B4GALT1	cg25509184	TRIO
RPS12	cg00448720	WNK1	GNAS	cg12348970	KMT2C	KHSRP	cg22396755	DNAH5
RPL35	cg27462398	PTPRH	CCNE1	cg25856811	FAT1	PTN	cg18952647	SYNE2
RPLP1	cg15821095	MYH7	URI1	cg00152644	MYH1	MRPL37	cg07027513	KMT2C
RPS11	cg21591452	AHNAK	ATP6V0E1	cg05440289	TSHZ3	DPEP3	cg22881914	MDN1
RPL13A	cg15679651	HMCN1	CANX	cg21754343	FLG	MXRA8	cg09107315	FMN2
RPS8	cg16979445	OBSCN	MRPS12	cg07014174	ADAMTS19	HNRNPK	cg08575537	CSMD3
RPL8	cg01281904	TLN2	POMP	cg11750883	FAM135B	STOML2	cg09492887	PKHD1
RPL29	cg16773028	DNAH2	NHP2	cg24423088	FMN2	ZNF503	cg19308222	TLN2
RPS21	cg01541443	AMER1	ANXA5	cg17298704	CSMD3	TRIM8	cg01033938	CNTNAP2
RPLP2	cg01495509	F5	MYADM	cg08763351	LRP1B	GNB2	cg06958829	SAMD9L
RPLS27	cg13688966	DMD	NPM1	cg14258236	FAM47C	TUFM	cg02930996	CSMD1
RPL10A	cg12967560	NOTCH4	TCEAL9	cg25119415	RYR2	SELENOM	cg07773116	RYR1
RPS18	cg22289837	PRRC2A	HNRNPAB	cg25259754	MYH4	OSTC	cg11004890	FAM47C
LF4			LF5			LF6		
mRNA	DNA m.	Mutation	mRNA	DNA m.	Mutation	mRNA	DNA m.	Mutation
IGFBP2	cg27285720	HECW1	RPLP0	cg01497527	CFH	APLP2	cg26164184	CDH10
C3	cg23563234	BRCA1	PFN1	cg19205041	BRCA1	MLF2	cg12864235	CDH9
ITM2C	cg14011639	CDH10	RPL5	cg15914863	SPEN	LTBP4	cg02473123	PDE11A
RPL10	cg17612991	MAP2	RPS2	cg10861599	APOB	TSPO	cg11884243	PCDH15
SLC34A2	cg18493147	PZP	IFI27	cg21238818	MDN1	CUTA	cg14173969	ASTN1
PTMS	cg21614638	MACF1	B2M	cg16517394	FLG	ALKBH7	cg19475870	OBSCN
B2M	cg02335804	PDE11A	EEF2	cg08507270	PPP1R3A	FGFRL1	cg17677397	SPTA1
MARCKSL1	cg06550629	MYH8	RPL4	cg15746187	TTN	EIF2S3	cg22463915	LRP1B
TACSTD2	cg25151806	FLG	ISG15	cg26620356	SORCS3	ARF3	cg25336198	HAS2
CLDN6	cg11809091	CNTNAP2	RPL10A	cg19537511	CYP11B1	YIPF3	cg13745346	FGA
MUC1	cg26282384	MUC16	HLA-C	cg15149645	COL22A1	TMBIM6	cg05331214	TLN2
HLA-DRB1	cg07080946	NEB	TMSB4X	cg04806409	PKHD1	PPP1R14B	cg21006686	CSMD1
TUBB	cg17714030	CSMD1	HLA-A	cg20022122	MUC16	AKIRIN1	cg08573687	AHNAK2
OST4	cg08108641	DMD	GPX1	cg06417962	CSMD1	RNF5	cg12150401	ZNF441
HLA-B	cg12788467	MYH4	HLA-B	cg14832904	IQGAP3	ANKRD65	cg18344063	IGHG1

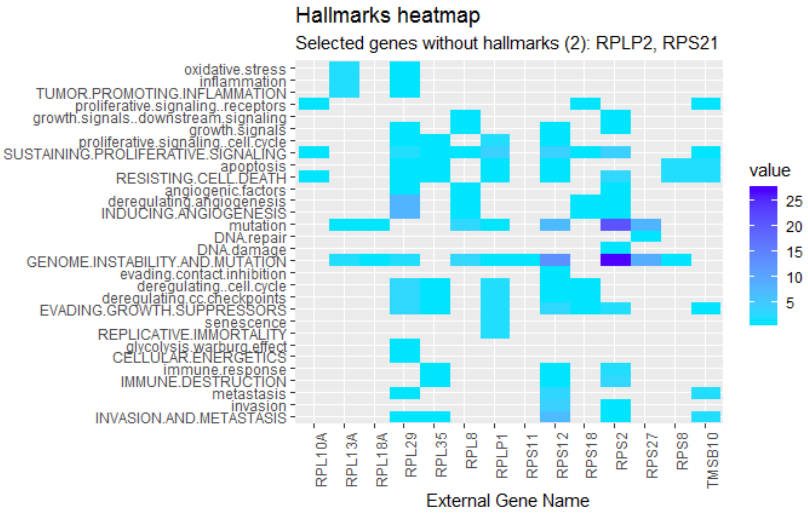
Table 5.7: Top-15 features in each latent factor in each type of data.

Was selected gene obtained from RNA-seq and mutation for cancer hallmark analysis based on the information made available by The Cancer Hallmarks Analytics Tool (CHAT) [26], the the results are present in Figure 5.11 and 5.12 respectively.

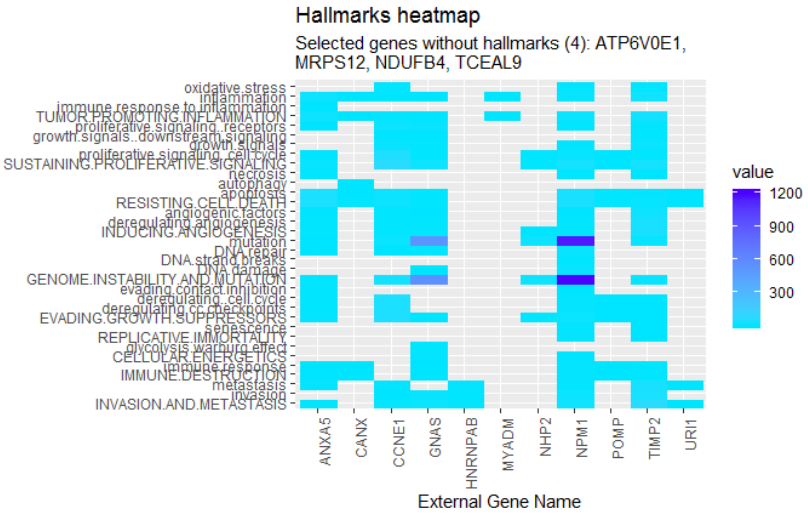
From observation on Figure 5.12 is visible that most of genes in all LF have influence in functions of ”genome instability and mutation” and ”mutation”. This result it was a bit of a surprise, because the variance capture on latent factor analysis is very low small. Even so, the most important genes selected by all latent factor are strong related with genome instability and mutation.

From Figure 5.11, is observable that the selected genes also have influence on ”genome instability and mutation”, ”mutation”, ”resistance on cell death”, ”sustaining proliferation signaling” and other cancer related causes.

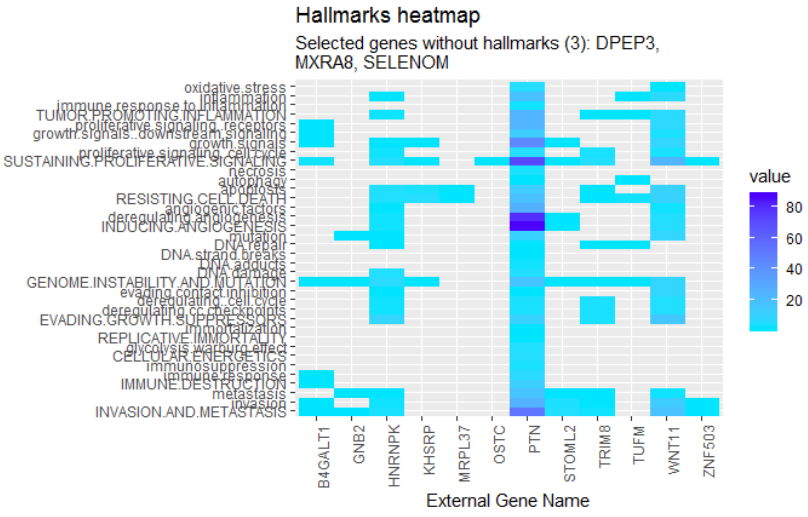
In general view, seems that MOFA have ability to select the features that are related with cancer disease, at same time for mutation data, the selected genes have few importance on other functionalities, which in my owner opinion is an advantage.



(a) Hallmark of LF1.

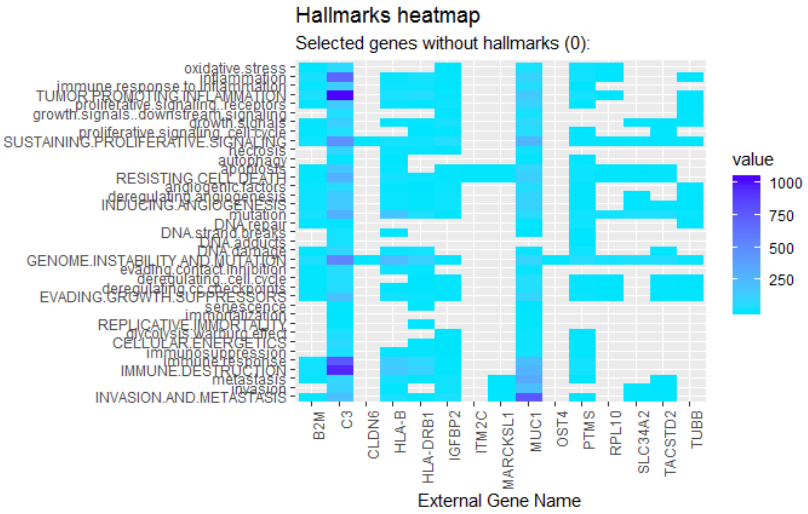


(b) Hallmark of LF2.

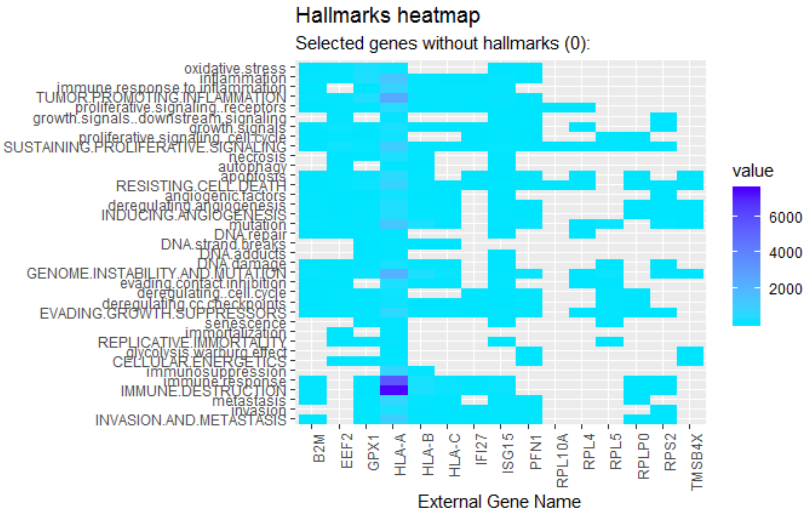


(c) Hallmark of LF3.

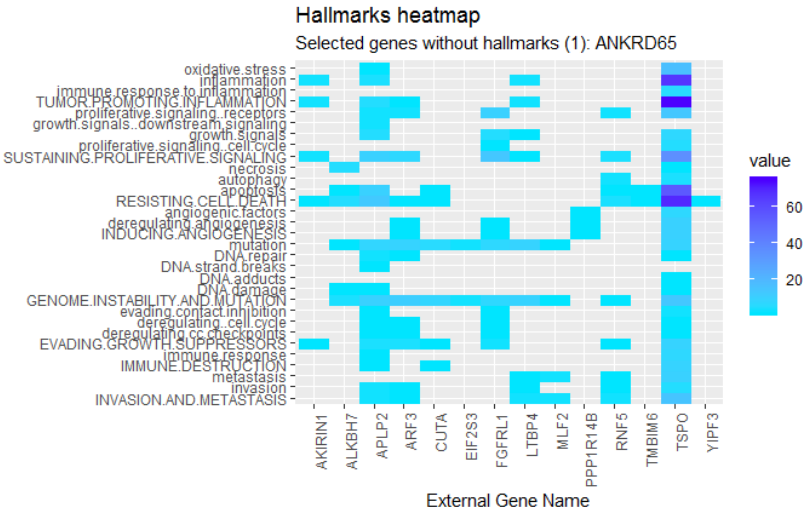
Figure 5.11: Hallmarks heat-map of top variate genes from LF in mRNA-sequence data.



(d) Hallmark of LF4.

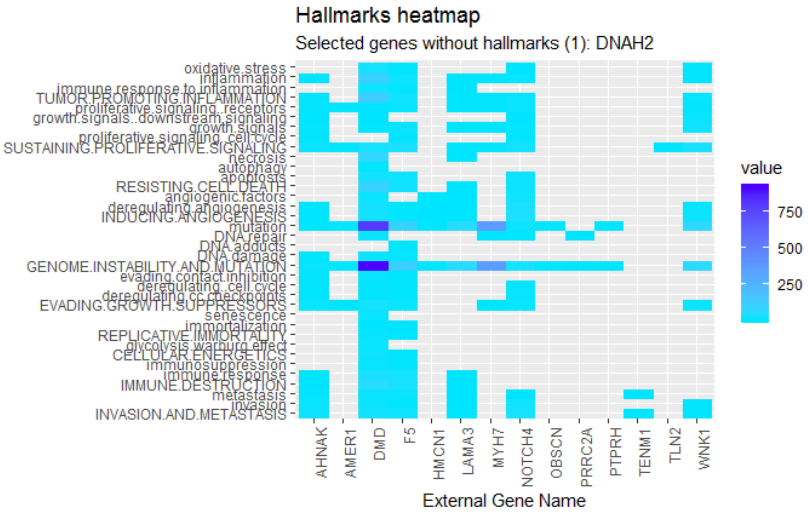


(e) Hallmark of LF5.

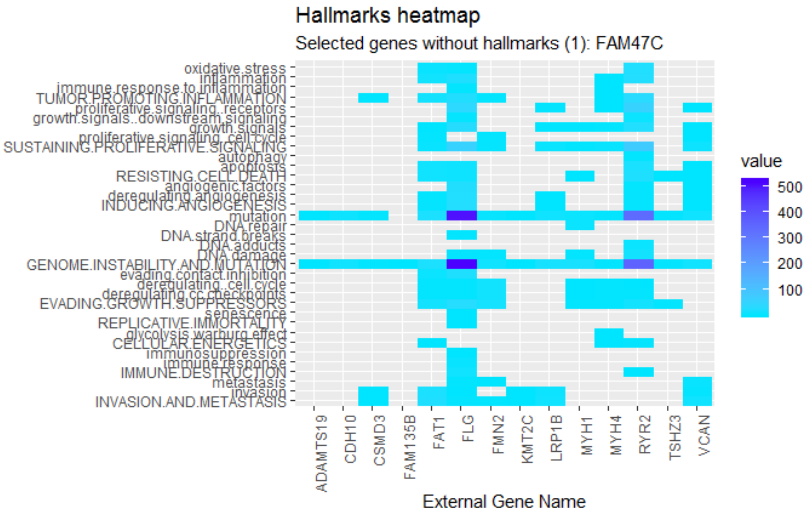


(f) Hallmark of LF6.

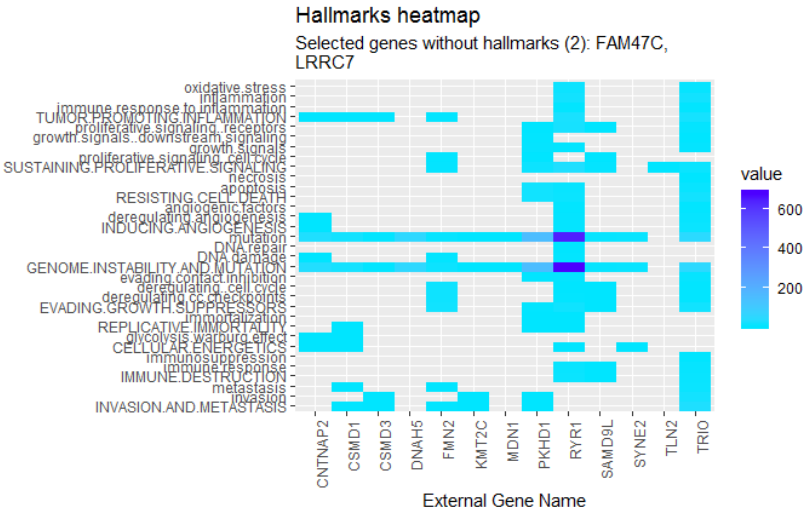
Figure 5.11: Hallmarks heat-map of top variate genes from LF in mRNA-sequence data(continuation).



(a) Hallmark of LF1.

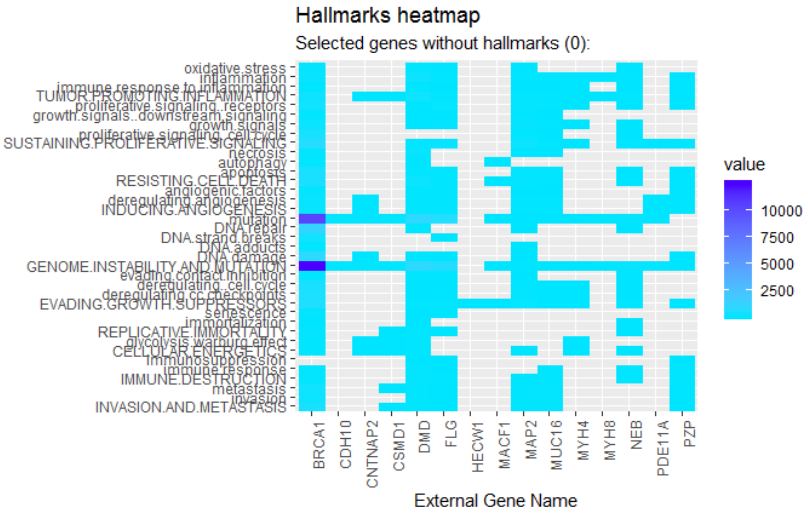


(b) Hallmark of LF2.

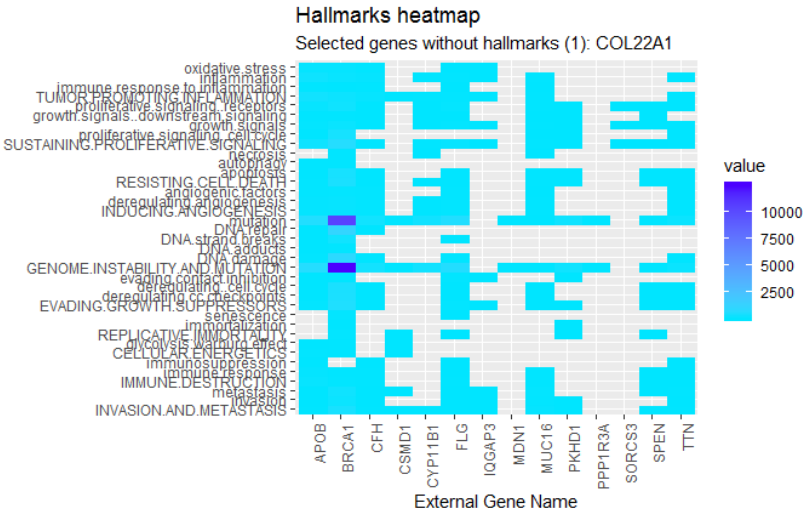


(c) Hallmark of LF3.

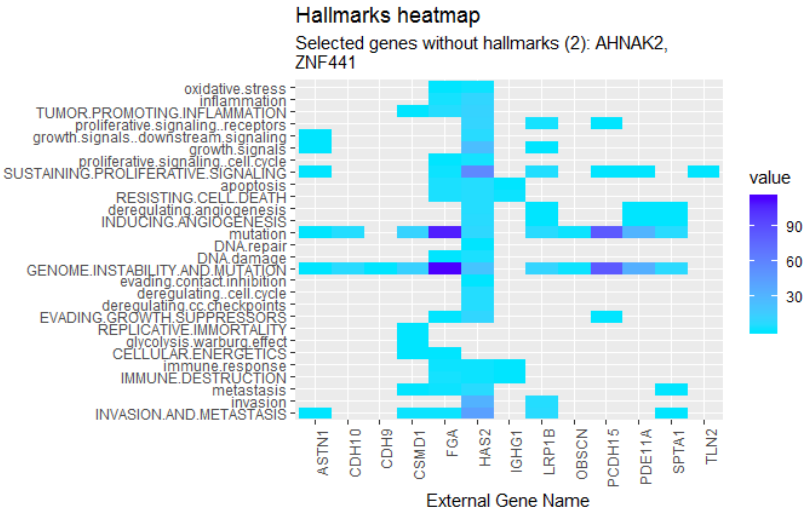
Figure 5.12: Hallmarks heat-map of top variate genes from LF in mutation data.



(d) Hallmark of LF4.



(e) Hallmark of LF5.



(f) Hallmark of LF6.

Figure 5.12: Hallmarks heat-map of top variate genes from LF in mutation data(continuation).

Clustering

In addition to the previous analysis, we also want to group the patient into clusters based on K-means clustering in the latent space. The choice of number of cluster is based on the Silhouette method, because this method shows better results in the synthetic data-set. The Silhouette method suggest 5 clusters for the OV data samples in the latent space. This data don't have well defined labels for comparison, the survival analysis on the samples of each cluster was made for analysis in Figure 5.13. From observation, the figure shows that the cluster are not well separate between clusters.

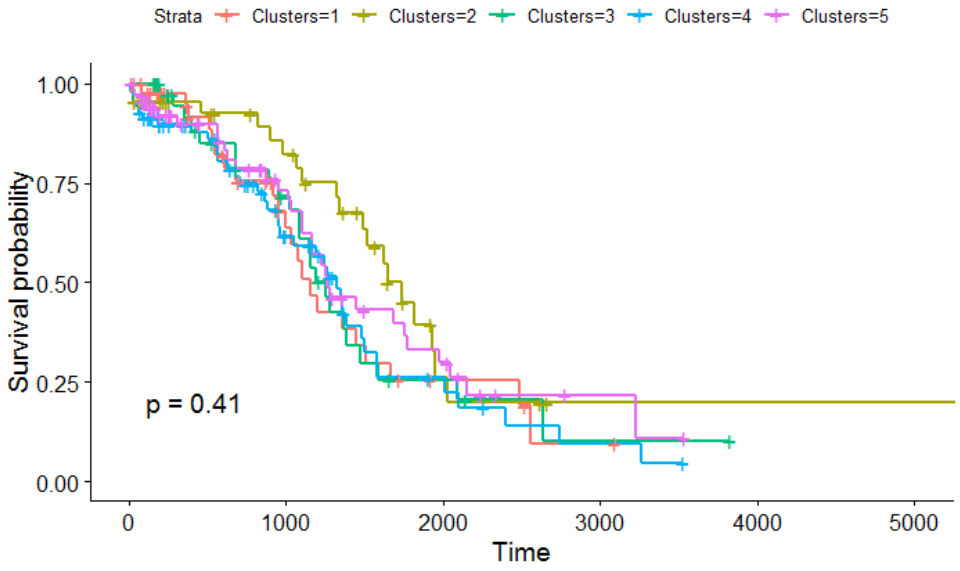


Figure 5.13: Survival plot with different clusters.

Original #LF	3	5	10	15
F = 20	#LF | #Obs 3 | 64	#LF | #Obs 2 | 2 3 | 7 4 | 4 5 | 51	#LF | #Obs 2 | 1 3 | 4 4 | 7 5 | 8 6 | 12 7 | 15 8 | 8 9 | 7 10 | 2	#LF | #Obs 1 | 1 2 | 3 3 | 6 4 | 10 5 | 9 6 | 6 7 | 12 8 | 10 9 | 5 10 | 1 11+ | 1
F = 50	#LF | #Obs 3 | 64	#LF | #Obs 2 | 1 3 | 2 4 | 5 5 | 56	#LF | #Obs 3 | 3 4 | 5 5 | 7 6 | 7 7 | 8 8 | 17 9 | 6 10 | 11	#LF | #Obs 1 | 1 2 | 2 3 | 8 4 | 6 5 | 8 6 | 5 7 | 8 8 | 8 9 | 4 10 | 8 11+ | 6
F = 100	#LF | #Obs 3 | 64	#LF | #Obs 2 | 1 3 | 2 4 | 3 5 | 58	#LF | #Obs 3 | 3 4 | 3 5 | 3 6 | 7 7 | 10 8 | 14 9 | 14 10 | 10	#LF | #Obs 1 | 1 2 | 5 3 | 5 4 | 5 5 | 7 6 | 7 7 | 10 8 | 7 9 | 7 10 | 4 11+ | 6
F = 500	#LF | #Obs 3 | 63 4 | 1	#LF | #Obs 3 | 1 4 | 2 5 | 57 6 | 3	#LF | #Obs 3 | 1 4 | 5 5 | 4 6 | 7 7 | 7 8 | 12 9 | 9 10 | 19	#LF | #Obs 1 | 1 2 | 4 3 | 7 4 | 8 5 | 6 6 | 6 7 | 8 8 | 6 9 | 5 10 | 3 11+ | 10
F = 1000	#LF | #Obs 3 | 59 4 | 5	#LF | #Obs 4 | 2 5 | 60 6 | 2	#LF | #Obs 3 | 1 4 | 4 5 | 3 6 | 9 7 | 9 8 | 11 9 | 14 10 | 13	#LF | #Obs 1 | 3 2 | 3 3 | 6 4 | 9 5 | 4 6 | 4 7 | 8 8 | 10 9 | 7 10 | 2 11+ | 8

Table 5.3: Effects of feature number on latent numbers with MOFA in table representation.

#LF	3	5	10	15
F = 20	#LF | #Obs 2 | 21 3 | 25 4 | 14 5 | 2 6 | 1 7 | 1	#LF | #Obs 2 | 26 3 | 23 4 | 11 5 | 3 8 | 1	#LF | #Obs 2 | 18 3 | 17 4 | 15 5 | 8 6 | 2 7 | 4	#LF | #Obs 2 | 14 3 | 10 4 | 11 5 | 11 6 | 3 7 | 9 8 | 6
F = 50	#LF | #Obs 2 | 13 3 | 27 4 | 23 5 | 1	#LF | #Obs 2 | 29 3 | 21 4 | 8 5 | 4 6 | 1 7 | 1	#LF | #Obs 2 | 12 3 | 11 4 | 15 5 | 11 6 | 9 7 | 2 8 | 4	#LF | #Obs 2 | 14 3 | 9 4 | 6 5 | 11 6 | 7 7 | 9 8 | 8
F = 100	#LF | #Obs 2 | 20 3 | 16 4 | 28	#LF | #Obs 2 | 27 3 | 17 4 | 14 5 | 4 6 | 2	#LF | #Obs 2 | 19 3 | 11 4 | 16 5 | 10 6 | 4 7 | 2 8 | 2	#LF | #Obs 2 | 10 3 | 9 4 | 9 5 | 10 6 | 8 7 | 8 8 | 10

Table 5.4: Effects of feature number for finding latent space dimension by iCluster in table representation.

#LF	3	5	10	15	Total
k = 2	#k | #Obs 2 | 29 3 | 1 4 | 1 5 | 1 6 | 1 8 | 2 9 | 2 10 | 3	#k | #Obs 2 | 39 4 | 1	#k | #Obs 2 | 39 10 | 1	#k | #Obs 2 | 40	#k | #Obs 2 | 147 3 | 1 4 | 2 5 | 1 6 | 1 8 | 2 9 | 2 10 | 4
k = 3	#k | #Obs 2 | 25 3 | 13 4 | 1 5 | 1	#k | #Obs 3 | 34 5 | 3 6 | 3	#k | #Obs 3 | 40	#k | #Obs 2 | 13 3 | 6 4 | 14 5 | 7	#k | #Obs 2 | 38 3 | 93 4 | 15 5 | 11 6 | 3
k = 4	#k | #Obs 2 | 2 3 | 12 4 | 26 5 | 1	#k | #Obs 2 | 1 4 | 39	#k | #Obs 3 | 2 4 | 35 5 | 3	#k | #Obs 3 | 3 4 | 36 5 | 1	#k | #Obs 2 | 3 3 | 17 4 | 136 5 | 5
k = 5	#k | #Obs 3 | 6 4 | 1 5 | 26 7 | 5 8 | 2	#k | #Obs 2 | 2 4 | 17 5 | 16 6 | 2 7 | 2 8 | 1	#k | #Obs 2 | 1 5 | 9 6 | 11 7 | 14 8 | 5 9 | 1	#k | #Obs 5 | 27 6 | 12 9 | 2	#k | #Obs 2 | 3 3 | 6 4 | 18 5 | 78 6 | 25 7 | 21 8 | 8 9 | 3
k = 6	#k | #Obs 2 | 10 3 | 1 5 | 12 6 | 17	#k | #Obs 5 | 2 6 | 38	#k | #Obs 5 | 34 6 | 3 7 | 3	#k | #Obs 5 | 6 6 | 4 7 | 15 8 | 8 9 | 6 10 | 1	#k | #Obs 2 | 10 3 | 1 5 | 54 6 | 62 7 | 18 8 | 8 9 | 6 10 | 1
k = 7	#k | #Obs 4 | 2 6 | 21 7 | 17	#k | #Obs 5 | 3 6 | 4 7 | 18 8 | 10 9 | 5	#k | #Obs 6 | 10 7 | 14 8 | 5 9 | 11	#k | #Obs 5 | 1 6 | 5 7 | 11 8 | 18 9 | 1 10 | 4	#k | #Obs 4 | 2 5 | 4 6 | 40 7 | 60 8 | 33 9 | 17 10 | 4
k = 8	#k | #Obs 2 | 2 5 | 6 6 | 32	#k | #Obs 5 | 3 6 | 14 7 | 13 8 | 3 9 | 3 10 | 4	#k | #Obs 6 | 2 7 | 16 8 | 19 10 | 3	#k | #Obs 6 | 1 7 | 8 8 | 18 9 | 12 10 | 1	#k | #Obs 2 | 2 5 | 9 6 | 49 7 | 46 8 | 40 9 | 15 10 | 8
k = 9	#k | #Obs 2 | 1 3 | 3 5 | 2 7 | 2 8 | 6 9 | 21 10 | 5	#k | #Obs 7 | 1 8 | 15 9 | 21 10 | 3	#k | #Obs 7 | 3 8 | 23 9 | 13 10 | 1	#k | #Obs 7 | 3 8 | 8 9 | 5 10 | 24	#k | #Obs 2 | 1 3 | 3 5 | 2 7 | 9 8 | 52 9 | 60 10 | 33

Table 5.5: Table of experiment for testing the effect of number of latent factors in number of clusters using MOFA.

Chapter 6

Conclusion

In this chapter, the conclusions about objectives of this work are presented. In the first section are discussed the main achievement followed by the possible future work.

6.1 Achievements

The goal of this work is to analyze the high dimensional bio-metrics data and to group the patients in various clusters that maybe share similar underlying biological information for later clinical treatments.

The proposed methodology uses iCluster and MOFA for finding the latent factors. The results of synthetic data demonstrate that the MOFA is more advantageous than iCluster in many ways. Also, MOFA has the ability for recovering the correct number of latent factors for small cases and it can recover approximately number of original clusters. And iCluster shows bad results in the parameters recovering in high dimensional data. From the time cost point of view, iCluster requires much more time for training the samples due to parameters numbers.

With OV data-set of TCGA, MOFA can infer few number of hidden factors, but due to high dimensional and complexity of data, the factors cannot capture too much variance from original data. Nevertheless the obtained factors can provide the most important genes that are composed by them. Moreover this work has selected some genes that possibly are important in the cancer disease, able to provide useful information for the specialists to analyze.

6.2 Future Work

Normally, the areas of latent factors and clusters don't have well defined labels for comparing the results. It is interesting to apply this methodology to other types of data which is easier to verify the performance. In this work, the genes are filtered by variance filter, while more complex and variate types of filter can be applied to test the results. For example, classic feature selection techniques as LASSO or Elastic-net can be applied further.

In this work, the synthetic data test and real data analysis only use K-means clustering algorithm, so another point to work on is to test more types of algorithms and other distance metrics for clustering.

Bibliography

[1] R. Shen, "Package ' iCluster '," 2015.

[2] R. Argelaguet, B. Velten, D. Arnol, S. Dietrich, T. Zenz, J. C. Marioni, F. Buettner, W. Huber, and O. Stegle, "Multi-Omics Factor Analysis—a framework for unsupervised integration of multi-omics data sets," *Molecular Systems Biology*, vol. 14, no. 6, 2018.

[3] Y. Hasin, M. Seldin, and A. Lusis, "Multi-omics approaches to disease," *Genome Biology*, vol. 18, no. 1, pp. 1–15, 2017.

[4] R. Bellman, *Dynamic Programming*. Dover Publications, 1957.

[5] S. Mullainathan and J. Spiess, "Machine learning: An applied econometric approach," *Journal of Economic Perspectives*, vol. 31, no. 2, pp. 87–106, 2017.

[6] C. Xu and S. A. Jackson, "Machine learning and complex biological data," *Genome Biology*, vol. 20, no. 1, pp. 1–4, 2019.

[7] Woalder, " HHS Public Access," *Physiology & behavior*, vol. 176, no. 1, pp. 139–148, 2017.

[8] R. Tibshirani, "Regression Shrinkage and Selection Via the Lasso," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[9] H. Zou and T. Hastie, "Regularization and variable selection via the elastic net," *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, vol. 67, no. 2, pp. 301–320, 2005.

[10] R. Argelaguet, B. Velten, D. Arnol, S. Dietrich, T. Zenz, J. C. Marioni, F. Buettner, W. Huber, and O. Stegle, "Multi-Omics Factor Analysis—a framework for unsupervised integration of multi-omics data sets," *Molecular Systems Biology*, vol. 14, no. 6, p. e8124, 2018.

[11] R. Shen, A. B. Olshen, and M. Ladanyi, "Integrative clustering of multiple genomic data types using a joint latent variable model with application to breast and lung cancer subtype analysis," vol. 25, no. 22, pp. 2906–2912, 2009.

[12] "Online source about pca." `https://www.cs.ubc.ca/~schmidtm/Courses/540-W18/L18.5.pdf`. Accessed: 2019-10-23.

[13] M. E. Tipping and C. M. Bishop, "Probabilistic principal component analysis," *Journal of the Royal Statistical Society. Series B: Statistical Methodology*, vol. 61, no. 3, pp. 611–622, 1999.

[14] C. M. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[15] R. Xu and D. C. Wunsch, "Clustering algorithms in biomedical research: A review," *IEEE Reviews in Biomedical Engineering*, vol. 3, pp. 120–154, 2010.

[16] D. Aloise, A. Deshpande, P. Hansen, and P. Popat, "NP-hardness of Euclidean sum-of-squares clustering," *Machine Learning*, vol. 75, no. 2, pp. 245–248, 2009.

[17] H. Zha, X. He, C. Ding, H. Simon, and M. Gu, "Spectral relaxation for k-rneans clustering," *Advances in Neural Information Processing Systems*, 2002.

[18] C. Ding and X. He, "Cluster structure of K-means clustering via principal component analysis," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 3056, pp. 414–418, 2004.

[19] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe, "Variational Inference: A Review for Statisticians," *Journal of the American Statistical Association*, vol. 112, no. 518, pp. 859–877, 2017.

[20] "Mofa source in github." `https://github.com/bioFAM/MOFA`. Accessed: 2019-10-23.

[21] "Python scikit package." `https://scikit-learn.org/stable/`. Accessed: 2019-10-23.

[22] "The cancer genome atlas." `https://cancergenome.nih.gov/`. Accessed: 2019-10-23.

[23] "Ensemble genome browser." `https://www.ensembl.org/info/about/species.html`. Accessed: 2019-10-23.

[24] "Consensus cds project." `https://www.ncbi.nlm.nih.gov/projects/CCDS/CcdsBrowse.cgi`. Accessed: 2019-10-23.

[25] "icluster package." `https://bioconductor.org/packages/release/bioc/html/iClusterPlus.html`. Accessed: 2019-11-10.

[26] "The cancer hallmarks analytics tool." `http://chat.lionproject.net/`. Accessed: 2019-11-10.