Introduction to Computational Biology
Lecture # 10: The use of gene expression data

Efrat Mashiach
5.1.2006

1 Introduction
On the previous lesson we talked about gene expression analysis by comparing the expression of the genes in different conditions. Suppose we discover groups of genes with a similar expression pattern, what can we do next?

• **Annotations:** Find the group’s characteristics. For example, if most of the genes are related to the mitochondria, and they express differently in a sick person and in a healthy person, can we conclude that the cell is in a different metabolic state during the sickness? Or did we get many mitochondrial genes in the group by chance...

• **Guilt by association:** Learn about an unknown gene from the common characteristics of the rest of the genes in it’s group.

• **Predictions:** Predict the state of a patient and his response to drugs according to the expression of his genes.

• **Regulation:** Learn about the regulation factors that affect the expression of the genes in the group.

2 Annotations

**Definition 2.1** The group of genes that we will use are:

- \( G \) - a group of the studied genes (the whole genome for example).
- \( G_d \) - a subgroup of differentially expressed genes from \( G \) (\( G_d \subseteq G \)).
- \( A \) - a subgroup of genes from \( G \) with a particular annotation (\( A \subseteq G \)).

We would like to know if the annotation of group \( A \) characterizes the genes in \( G_d \). There are two possibilities in which we’ll say that \( G_d \) is characterized by the annotation:

1. A significantly large part of \( G_d \) is in \( A \): \( \left| \frac{A \cap G_d}{|A|} \right| > \left| \frac{A \cap G_d}{|G|} \right| \)

2. A significantly large part of \( A \) is in \( G_d \): \( \left| \frac{A \cap G_d}{|A|} \right| > \left| \frac{G_d}{|G|} \right| \)

Our \( H_0 \) assumption is that the overlapping between \( A \) and \( G_d \) is random, and there is no real connection between the genes in \( G_d \) and the annotation. We can choose \( A' \) (or \( G_d' \)) randomly and evaluate the probability:

\[ P(|A' \cap G_d| \geq |A \cap G_d|) \]
2.1 The Hyper-Geometric distribution

The story: There is a bag with \(N\) balls - \(n\) red balls, and \((N - n)\) black balls. We take out \(K\) balls, what is the probability that \(m\) out of the \(K\) balls would be red?

\[
P_{HG}(m \mid N, n, K) = \binom{n}{m} \binom{N-n}{K-m} \binom{N}{K}
\]

The Hyper-Geometric p-value is:

\[
p-value = \min(n, K) \sum_{m'=m}^{\min(n, K)} P_{HG}(m' \mid N, n, K)
\]

In our case:

\(|G| = N, \quad |G_d| = n, \quad |A| = K, \quad |A \cap G_d| = m\)

2.2 What are the problems we face, while looking for annotations?

1. There are many different annotations.
2. The annotations of the genes in the DB are partial. There are genes that were never tested for certain annotations.
3. There are no 'negative annotations', so the reference group isn’t clear.
4. Many annotations have a hierarchical structure. For Example, the annotation 'Enzyme' includes the annotation 'Oxidizing Enzymes'. For that reason, the tests we do for the annotations are dependant.

3 Predictions

In the medicine field for example, we would like to find out why some patients response to a certain drug and the others don’t.

The naive method:

1. Get a matrix of the gene expression of the group of patients that respond to the drug, and a matrix of the gene expression of the group that doesn’t respond.
2. Run a learning algorithm on the data.
3. Use the result algorithm to predict the group of a new patient, by his gene expression.

The problems:

1. There is a small number of patients compared to the number of genes.
2. The learning algorithm may find an over-fitting solution, that will work for the data but won’t work in general cases, i.e. the solution describes the noise.
3. If we’ll have \(k\) patients, the mistakes percentage will be in jumps of 100/k, and won’t be accurate.

Solutions:
1. **Cross Validation (CV)**
   Let the algorithm learn from a subgroup of the data, and test the results on the rest of the data.
   In the *K-fold CV* method, we divide the group of samples (patients) into K subgroups, and in the \(i^{th}\) iteration \((1 \leq i \leq K)\), remove the \(i^{th}\) subgroup, learn from the rest of the group, and then test the results on the removed subgroup. Usually we use \(5 \geq K \geq 10\).
   When \(K\) is equal to the number of samples, in each iteration only one sample is removed. In this case, the method is called *Leave-one-out CV*.
   The CV is used to evaluate the quality of the learned algorithm. It enables us to calculate the mistake percentage in an unbiased way.

2. **Feature Selection**
   In this method, we identify the meaningful genes, and we choose a representative gene from each group of genes that has a similar expression. In this way we reduce the size of the genes group while preserving as much information as we can.
   What is the difference between finding differentially expressed genes and feature selection? While finding differentially expressed genes helps us describe the differences between the expression patterns in different groups of genes, the feature selection method isolates the information about the differences.

4. **Clustering**
   Finding a partition of the genes into groups, according to a common expression pattern.
   If we measure the expression of the genes in \(m\) different conditions, than we can represent the expression of each gene as a vector in \(\mathbb{R}^m\).

4.1 **Clustering methods:**
1. **k-means**: Iteratively find centroids (means), and then divide the genes into groups according to the closest centroid.
2. **Hierarchical clustering**: See Figure 1. The main advantage in this method is that we can divide the genes to groups in different resolutions, but it is not always clear which resolution to choose.

There are two ways of implementing the hierarchical clustering:

(a) **Top-down**: In each level in the recursion there is $2^n$ partition possibilities.

(b) **Bottom-up (Agglomerative Clustering)**: In each level in the recursion, there are $O(n^2)$ partition possibilities, so it is easier to compute, but here we focus on the details and we might miss the global phenomena.

The algorithm of this agglomerative clustering is similar to Huffman coding algorithm.

Both of the methods are problematic in that during the recursion, we can’t regret the decisions that were made in previous levels.

**Definition 4.1** We define distance functions as follow:

$d(a, b) = $ the distance between the vectors $a$ and $b$.

$d(A, B) = $ the distance between two groups of vectors: $A$ and $B$.

In these methods we need to calculate the distance between groups of vectors. This can be done in a few ways:

1. **Mean distance**:

   \[ d(A, B) = \frac{1}{|A| \cdot |B|} \sum_{a \in A, b \in B} d(a, b) \]

   i.e. calculate the distance between the groups by the distance between the average vectors of each group.

2. **Single link**:

   \[ d(A, B) = \min_{a, b} d(a, b) \]

   i.e. calculate the distance between the groups by the distance between the closest pair of vectors in the two groups. In the agglomerative clustering, this can be calculated using the ‘minimum spanning tree’ algorithm.

3. **Complete link**:

   \[ d(A, B) = \max_{a, b} d(a, b) \]

   This is harder to compute.

4.2 **A probability model of clustering**:

A gene expression is represented by \((x_1, x_2, \ldots, x_m)\) and we would like to calculate it’s probability. There is an additional hidden random variable $c$ (the cluster), and we assume that $P(c)$ is known.

\[
P(x_1, x_2, \ldots, x_m) = \sum_c P(c)P(x_1, x_2, \ldots, x_m \mid c) = \sum_c P(c) \prod_i P(x_i \mid c)
\]

Assuming that the $x$’s are i.i.d in dependence of $c$, i.e. $P(x_i \mid c) \sim N(\mu_{ci}, \sigma_{ci}^2)$.

We would like to find the parameters $\mu_{ci}, \sigma_{ci}^2$ that will give the maximum likelihood.

The posterior is calculated using Bayes’ theorem:

\[
P(c \mid x_1, x_2, \ldots, x_m) \propto \prod_i P(x_i \mid c)P(c) = P(c) \prod_i \frac{1}{\sqrt{2\pi \sigma_{ci}}} e^{-\frac{(x_i - \mu_{ci})^2}{2\sigma_{ci}^2}}
\]

If all the variances are constant (for example: $\forall i \sigma_{ci} = 1$) the Euclidean distance will determine the clusters (similar to k-means).

Different variances mean that for some clusters it is more important to be close to the expectancy, and in others it is less important, i.e. each cluster emphasizes different experiments.
4.3 EM Clustering Algorithm

This is an algorithm that is similar to the k-means algorithm. While in the k-means algorithm the expectancies and the variances of all the clusters are the same, in this EM algorithm these parameters can be different in different clusters. If the clusters of each gene are known, than we can find the expectatancies and the variances, using MLE method.

The sufficient statistics are:

\[ S^1 = \sum_i 1 \]
\[ S^2 = \sum_i x_i \]
\[ S^3 = \sum_i x_i^2 \]

**M-Step** - By using the MLE we get:

\[ \hat{\mu} = \frac{S^2}{S^1} \]
\[ \hat{\sigma}^2 = \frac{S^3}{S^1} - \hat{\mu}^2 \]

In our case:

\[ S^1_{c_i} = \sum_{n=1}^{N} 1\{C[n] = c\} \]
\[ S^2_{c_i} = \sum_{n=1}^{N} x_i[n] \cdot 1\{C[n] = c\} \]
\[ S^3_{c_i} = \sum_{n=1}^{N} x_i^2[n] \cdot 1\{C[n] = c\} \]

**E-Step** - The expected sufficient statistics are:

\[ \mathbb{E}(S^1_{c_i} \mid D) = \sum_{n=1}^{N} P(C[n] = c \mid x_1[n] \ldots x_m[n]) \]
\[ \mathbb{E}(S^2_{c_i} \mid D) = \sum_{n=1}^{N} x_i[n] \cdot P(C[n] = c \mid x_1[n] \ldots x_m[n]) \]
\[ \mathbb{E}(S^3_{c_i} \mid D) = \sum_{n=1}^{N} x_i^2[n] \cdot P(C[n] = c \mid x_1[n] \ldots x_m[n]) \]

Now, we can calculate each iteration in the EM.