Computational Biology I: Quantitative Data Analysis
Frühjahrsemester 2019

Exercise 8

Athos Fiori and Pascal Grobecker

Analysing gene expression by principal component analysis

One central question in the study of gene regulation is how the same genetic information can give rise to different cellular phenotypes. Gene expression studies across different human tissues have revealed that cell types can be distiguished by their gene expression profile. For example, hepatocytes, which are the primary cells present in liver tissue, have a distinct expression of genes related to liver-specific protein synthesis and metabolism.
Generally, tissue are highly-organised stuctures formed by different cell types. For example, the liver also contains sinusoidal endothelial cells, Kupffer cells and hepatic stellate cells whose specialised functions constrain them to an expression pattern which partly differs from that of hepatocytes. Therefore, when measuring gene expression from a complex tissue like liver, the resulting profile will be a mixture of the gene expression profiles of the different cell types forming the tissue. A natural question is therefore to ask: given the gene expression profile of an uncharacterised tissue sample, can we distinguish different cell types within the sample? If so, how many are there, and what is their relative fraction within the tissue?
As a step towards these questions, you will be applying *principal component analysis* (PCA) to a gene expression data set containing the expression of several thousand genes (25550) across 7 samples. All samples come from the same tissue but differ in their cell type composition. PCA allows you to quantify the sample-to-sample as well as gene-to-gene variation in terms of its (linearly) independent components. This might give you a hint to the number of (indepenent) cell types that generated the observed variation in gene expression.

PCA

Let's first recap what's PCA and where it comes from. Assume that in 2D you are in a reference frame (u, w) where your data-points $\{(u_1, w_1), (u_2, w_2), ..., (u_n, w_n)\}$ follow some linear relation. The goal is to find a new reference frame (x, y) in which the data points $\{(x_1, y_1), (x_2, y_2), ..., (x_n, y_n)\}$ do not correlate anymore i.e. $P(x, y) = P(x)P(y)$. Let's assume for simplicity that in this new coordinate system (called natural coordinate system) $P(x)$ and $P(y)$ are Gaussian distributed with mean zero and variance σ_x^2 and σ_y^2 respectively so the likelihood in this system reads

$$P(D|\sigma_x, \sigma_y) = \frac{1}{(2\pi\sigma_x^2\sigma_y^2)^{\frac{n}{2}}} \exp\left[-\sum_i \frac{x_i^2}{2\sigma_x^2}\right] \exp\left[-\sum_i \frac{y_i^2}{2\sigma_y^2}\right]$$

since the natural coordinate system and the original one are related by the relation

$$x_i = u_i \cos\theta + w_i \sin\theta - x_0 \tag{1}$$
$$y_i = w_i \cos\theta - u_i \sin\theta - y_0 \tag{2}$$

the likelihood in the original reference system reads

$$P(D|\sigma_x, \sigma_y, \theta, x_0, y_0) = \frac{1}{(2\pi\sigma_x^2\sigma_y^2)^{\frac{n}{2}}} \exp\left[-\sum_i \frac{(u_i \cos\theta + w_i \sin\theta - x_0)^2}{2\sigma_x^2}\right] \exp\left[-\sum_i \frac{(w_i \cos\theta - u_i \sin\theta - y_0)^2}{2\sigma_y^2}\right] \tag{3}$$

As said, our main goal is to find how the natural frame and the original frames are related i.e. we want to find θ, x_0, y_0. To find the most likely x_0 we can either marginalise over $\sigma_x, \sigma_y, \theta, y_0$ and look to the x_0 which maximise the posterior or, since we are using uniform prior, the x_0 which maximise the likelihood is the same

as the one which maximise the posterior[1] thus

$$\frac{\partial P(D|\sigma_x,\sigma_y,\theta,x_0,y_0)}{\partial x_0} = 0 \Rightarrow x_0 = \bar{u}\cos\theta + \bar{w}\sin\theta \tag{4}$$

and similar

$$y_0 = \bar{w}\cos\theta - \bar{u}\sin\theta \tag{5}$$

Equation (3) becomes, using (4), (5)

$$P(D|\sigma_x,\sigma_y,\theta) = \frac{1}{(2\pi\sigma_x\sigma_y)^{n-1}}\exp\left[-n\frac{X^2}{2\sigma_x^2} - n\frac{Y^2}{2\sigma_y^2}\right]$$

where X and Y are defined in the lecture. In order to find the best θ we first have to marginalise over the other parameters i.e.

$$P(D|\theta) = \int d\sigma_x d\sigma_y P(D|\sigma_x,\sigma_y,\theta)P(\sigma_x|I)P(\sigma_y|I) \propto \left[X^2Y^2\right]^{-\frac{n-1}{2}}$$

where we use uniform priors. If we define $\alpha = \tan\theta$, the slope which maximise (α_1^*) and minimise (α_2^*) the posterior $P(\theta|D)$ read

$$\alpha_{1,2}^* = \frac{V_{uu}-V_{ww}}{2V_{uw}} \pm \sqrt{1+\left(\frac{V_{uu}-V_{ww}}{2V_{uw}}\right)^2}$$

therefore the line $\begin{pmatrix} 1 \\ \alpha_1^* \end{pmatrix}$ is the "best" and $\begin{pmatrix} 1 \\ \alpha_2^* \end{pmatrix}$ the "worst" line.

Matrix notation Let's now go trough the same steps but using the matrix notation. If we combine (1), (2), (4) and (5) we find

$$x_i = (u_i - \bar{u})\cos\theta + (w_i - \bar{w})\sin\theta \tag{6}$$
$$y_i = (w_i - \bar{w})\cos\theta - (u_i - \bar{u})\sin\theta \tag{7}$$

By defining, $\vec{x} = \begin{pmatrix} x \\ y \end{pmatrix}$, $\vec{v} = \begin{pmatrix} u-\bar{u} \\ w-\bar{w} \end{pmatrix}$ and $R = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$ equation (6),(7) read

$$\vec{x} = R\cdot\vec{v}$$

In matrix notation, equation (3) reads

$$P(\vec{v}|\vec{\sigma}) = \frac{1}{2\pi\sigma_x^2\sigma_y^2}\exp\left[-\frac{1}{2}\vec{v}^T\cdot M\cdot\vec{v}\right] \quad where \quad M = R^{-1}\underbrace{\begin{pmatrix} \frac{1}{\sigma_x^2} & 0 \\ 0 & \frac{1}{\sigma_y^2} \end{pmatrix}}_{:=D} R$$

The natural reference frame is the one that diagonalise M, in fact only if M is diagonal we can write the total likelihood as a product of independent likelihood. From linear algebra it's well known that the eigenvectors basis diagonalise M and therefore it provides the natural reference frame. Let's thus solve the eigenvector equation $M\vec{x} = \lambda_x\vec{x}$. Before computing the eigenvector of M let's first remark that the general multivariate gaussian distribution reads

$$\mathcal{N}(\vec{\mu},C) = \frac{\exp\left[-\frac{1}{2}(\vec{x}-\vec{\mu})^T\cdot C^{-1}\cdot(\vec{x}-\vec{\mu})\right]}{\sqrt{\det(2\pi C)}}$$

where C is the covariance matrix. It's easy to show that **the inverse of the covariance matrix C gives the precision matrix M so eigenvectors of C are also eigenvectors of M**. The covariance matrix which maximise the likelihood reads

$$C = \begin{pmatrix} V_{uu} & V_{uw} \\ V_{wu} & V_{ww} \end{pmatrix} \quad where \quad V_{rt} = \frac{1}{n}\sum_{i=1}^{n}(r_i-\bar{r})(t_i-\bar{t})$$

[1] This is not true in general.

We can now solve the eigenvector equation for C and we find $\begin{pmatrix} 1 \\ \alpha_1^* \end{pmatrix}$ and $\begin{pmatrix} 1 \\ \alpha_2^* \end{pmatrix}$ to be the two eigenvectors. The optimal line is the one with highest eigenvalue in perfect agreement with previous procedure. This procedure (PCA) can be generalised in higher dimension and can be summarised as follow

- Consider a scatter of points deriving from a multi-variate Gaussian distribution
- We want to find a coordinate system (natural coordinate system) where the data are independent Gaussian distributed
- The precision matrix M represent the rotation we have to apply to our coordinate system in order to be the natural one
- The inverse of M is the covariance matrix C thus **eigenvecors of C gives the natural coordinate system**. This procedure is called PCA.
- The component are sorted from the highest to the lowest variance captured in the axis i.e. the first component is the one with largest eigenvalue, second component with second largest eigenvalue, and so on..
- By projecting the data on these components we can represent high dimensional data in lower dimension

Home work

---

1. We provide you the function `loadData` which loads the gene expression matrix from the data file `dataSet8.dat`. The structure is the following
   `rowNames ,colNames , E = loadData('dataSet8.dat')`
   where `colNames` contains all the 7 samples name, `rowNames` contains 25550 different genes and `E[i][j]` contains the gene expression of gene j in sample i.
2. We can see the problem in 2 different ways. Either we consider the 25550 dimensional gene space an represent every sample as a point in this space, or we consider the 7 dimensional sample space and consider every gene as a point in this space Fig 1. Clearly the second choice is much better since the space is lower dimensional and we have more data points in it.
   We have seen that to perform PCA we need to find the covariance matrix. First remark that the matrix with zero mean column $\langle E \rangle$ is simply computed with

   $$\langle E \rangle_{gs} = E_{gs} - \frac{1}{G} \sum_{g=1}^{G} E_{gs}$$

   Since we are in the sample space the covariance matrix is defined as

   $$\mathrm{Cov}(E) = \frac{1}{G} \sum_{g} \langle E \rangle_{gs} \langle E \rangle_{gs'} = \frac{1}{G} \langle E^T \rangle \langle E \rangle$$

   Complete the function `Covariance` which returns the covariance matrix in sample space. Note that you can use the numpy function `dot` to compute the matrix product.
3. Complete the function `PCA` , which returns the eigenvalues and eigenvectors of C, you can use the function `linalg.eig` of numpy.
4. The PCA gives you a base V, which characterize the type of variation in the data set mentioned above, i.e.,

   $$\mathrm{Cov}_S(E) \sim E^T E = V D V^T.$$

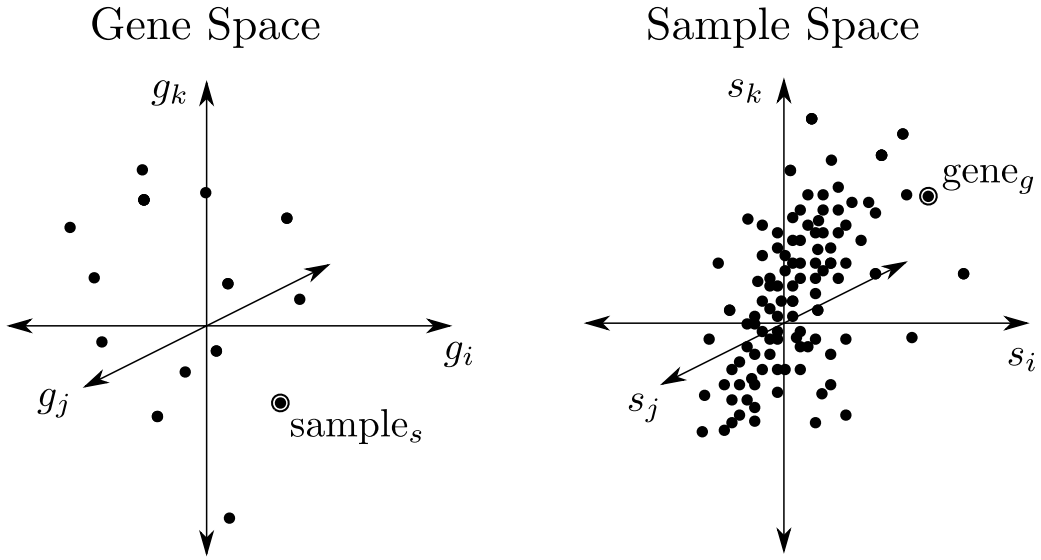

Figure 1: Centered gene expression data vizualized as a cloud of points in a 3-dimensional sub-space. Left: gene space; very dot represents the expression levels of a sample across all genes. Right: sample space; every dot is the expression of a particular gene across all samples. Note that in the example, we observe many more genes than samples, i.e., the gene expression matrix has many more rows than columns.

The columns (principal components) of V form a basis for the sample space. The diagonal elements of D quantify the amount of variation in the directions of the principal components. The fraction of explained variance (FOV) per principal component i can be quantified as

$$\text{FOV}_i = \frac{D_i}{\sum_j D_j}.$$

Complete the function `FOV` which returns the first 2 principal components and their fraction of explained variance. You can use the numpy function `argsort` to find how the array arguments are sorted.

5. Keeping only the first two principal components we can find the two-dimensional plane through the high-dimensional dataset in which the data is most spread out, so if the data contains clusters these too may be most spread out, and therefore most visible to be plotted out in a two-dimensional diagram. Therefore implement the function `Project2` which returns the projection of E in the 2 first principal component. (Additional) What do we observe by plotting them?