Math 612: Single Cell Analysis **2019W Term 1**

Lecture 6: September 26

Lecturer: Geoffrey Schiebinger *Scribe: Min Jun Jo*

6.1 Discovering heterogeneity from a sample

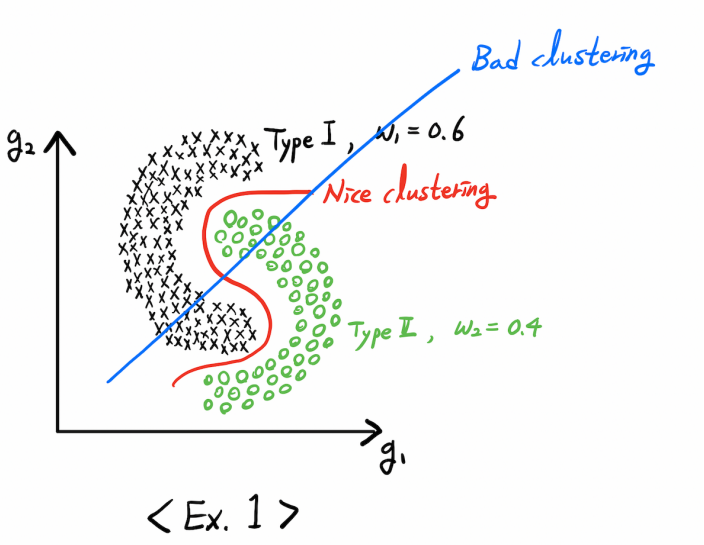
When we take samples from a given space of cells, we often encounter a mixture of different types of cells. In those cases, appropriate **clustering** would be a crucial starting point. The question is, how can we do the appropriate clustering for given data?

6.1.1 Introduction

Let $X_1, X_2, \cdots, X_n$ be n i.i.d. samples. We will talk about modeling this as a mixture of different cell types; specifically, we will deal with mixture modeling, clustering, and differential expression.

A mixture model (or distribution) is a special kind of distribution that we will use to think about different cell types. Mathematically, a mixture distribution $\overline{P}$ can be defined by $\overline{P} = w_1 P_1 + \cdots + w_k P_k$ where w_i's stand for the corresponding "mixture weights" satisfying $w_1 + \cdots + w_k = 1$ and P_i's are distributions called "mixture components."

Suppose that we have 2 types of cells. If we label the cells with respect to two genes, say g_1 and g_2, then we obtain, for examples, the following graphs: the first one <Ex.1> describes a case that we need a more sophisticated method of clustering, and the second one <Ex.2> is the simplest case on which a crude clustering still works.



< Ex. 1 >

To sample from a mixture model,

1) Pick cell type T as the following:

$$T = \begin{cases} 1 \text{ with probability } w_1 \\ 2 \text{ with probability } w_2 \\ \vdots \\ k \text{ with probability } w_k \end{cases}$$

2) Sample expression profile $X \sim P_T$. Then we can indeed check that $X \sim \overline{P}$.

6.1.2 Clustering - unsupervised learning

Goal of clustering: Given $X_1, \cdots, X_n \sim \overline{P}$, group $X_1, \cdots, X_n$ into groups by cell types.

Given groupings, we can try to learn what makes cell types different.

6.1.3 Differential expression - supervised learning

The simplest approach can be described as the following four steps:

i. Find the mean of whole population $\overline{m}$

ii. Find the means of X_i's in each group, say (m_i, T_i)'s

iii. Fold change $\frac{m_i}{\overline{m}}$

iv. Sort to find top genes over-expressed in each group

2) Train classifier

Given n samples $X_1, X_2, \cdots, X_n$, the most popular simple approach is called k-means that consists of the four steps below.

i. Initialize mean vectors $m_1, m_2, \cdots, m_k \in \mathsf{R}^{20,000}$

ii. Assign each point X_i to the closest mean vector

iii. Update mean vector to be the mean of X_i assigned to it

iv. Repeat ii. and iii. until convergence

This train classifier is applicable to <Ex. 2>. But it's flawed in being applied to <Ex. 1>.

6.1.4 Spectral clustering

6.1.4.1 Introduction

Spectral clustering is a nonlinear dimensionality reduction that would transform data into k-means through which we can see the equivalent but well-clustered data. This method is twofold:

1) Nonlinear embedding into lower dimensional space R^k where k is the number of cell types we are looking for. This nonlinear embedding is called a kind of "diffusion components."

2) k-means in R^k

6.1.4.2 Settings

We fix an arbitrary *kernel function* $\kappa : \chi \times \chi \to \mathsf{R}^+$. An example of our target space χ is $\mathsf{R}^{20,000}$, a space of genes. Here $\kappa(x, y)$ stands for the "similarity" between two points x and y: roughly speaking, if $\kappa(x, y) \sim 0$, we can say that x and y are different, and if $\kappa(x, y) \sim 1$, then we say that x and y are similar.

Example 1. Linear kernel $\kappa(x, y) = x^T y$

Example 2. Polynomial kernel $\kappa(x, y) = (1 + x^T y)^2$

Example 3. Gaussian kernel $\kappa(x, y) = e^{\frac{||x-y||^2}{2\ 2}}$ with *bandwidth* σ

Using the kernel function κ we define the *kernel matrix* K by $(K)_{ij} = \kappa(x_i, x_j)$ for each $(i, j) \in [n] \times [n]$. By the definition of K, K is a symmetric matrix which is poisitive semi-definite, i.e., $K \geq 0$.

6.1.4.3 Diffusion component embedding (Laplacian embedding)

This embedding consists these five steps:

I. Compute the kernel matrix

Given a kernel K, we do kernel PCA: compute the eigenvectors of K.

II. "Normalize" the kernel matrix

Why do we normalize it? PCA is going to find the subspace that maximizes the variance of the data. Therefore, if 90% of the data (just a rough percentage signaling its dominance) is in one cluster, then our top k eigenvectors would all be related to the dominant cluster. Thus we need to find a different matrix whose top eigenvectors would correspond to different clusters, i.e., we should leverage a more appropriate matrix for clustering.

The normalized Laplacian L is defined by

$$L = D^{-1/2} K D^{-1/2},$$

where the diagonal matrix D can be expressed component-wisely by $d_{ii} = \sum_{j=1}^{n} \kappa(x_i, x_j)$ for each i.

III. Compute eigenvectors of L: consider $Lv = \lambda v$.

Take the top k eigenvectors, say $v_1, v_2, \cdots, v_k$.

IV. Embed the data

Note that the top eigenvectors were derived from the evaluation of the kernel function κ over the samples $X_1, X_2, \cdots, X_n$. So, for each $j \in [n]$ (L is a $n \times n$ matrix), indeed we can see each component of the eigenvector v_j as a function:

$$v_j = \begin{pmatrix} v_j(X_1) \\ \vdots \\ v_j(X_n) \end{pmatrix}$$

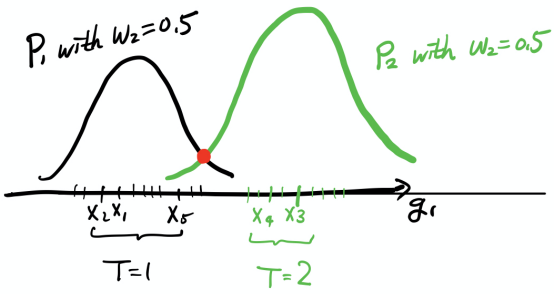
Here is the k-dimensional embedding: $X_i \longmapsto \big(v_1(X_i), v_2(X_i), , v_k(X_i)\big)$.

V. k-means in $\mathbb{R}^k$

For brevity, consider the case $k = 2$ with one dimensional space $\chi = \mathbb{R}$ for g_1.

Consider the mixture model $\overline{P} = w_1 P_1 + w_2 P_2$. Suppose that we have the samples described in <Ex.3>. Then we can project the data onto two-dimensional space spanned by P_1 and P_2, as presented in <Ex.4>.

The final result of the k-dimensional embedding would be seen in <Ex.5>.



<Ex. 3>