MCB137L/237L: Physical Biology of the Cell
Spring 2025
Homework 6
(Due 3/4/25 at 2:00pm)

Hernan G. Garcia

"Mathematics, rightly viewed, possesses not only truth, but supreme beauty cold and austere, like that of sculpture, without appeal to any part of our weaker nature, without the gorgeous trappings of painting or music, yet sublimely pure, and capable of a stern perfection such as only the greatest art can show. The true spirit of delight, the exaltation, the sense of being more than Man, which is the touchstone of the highest excellence, is to be found in mathematics as surely as in poetry." - Bertrand Russel in *Study of Mathematics*

1 Synthesizing a Transcriptome: Big Data in Transcription

In class, we briefly discussed the myriad of different ways to measure gene expression. Writ large, we can either find ways to count the mRNA transcripts or the protein products that result from these transcripts. For example, when properly calibrated, the green fluorescent protein (GFP) in conjunction with fluorescence microscopy is a favorite approach for measuring protein copy numbers. Recently, a different way to engage in the dialogue between theory and experiment has been afforded by the advent of technologies that make it possible to take a census of the full complement of transcripts inside individual cells.

One of the key applications of single-cell mRNA sequencing has been its use to identify "transcriptional fingerprints" that define discrete cell types within a population containing cells that have committed to multiple possible fates. One of the best examples of this application of single-cell transcriptome-wide sequencing comes from projects such as the *Tabula muris*. This project measured RNA counts for tens of thousands of genes within tens of thousands of individual cells in the mouse, derived from tens of distinct organs and tissues. Each single cell transcriptome is a giant $\approx$ 10,000 dimensional vector with the i^{th} entry corresponding to the mRNA count of the i^{th} gene.

One widespread approach to visualizing the results from these types of experiments is shown in Figure 1. In the figure, each point corresponds to an individual cell whose transcriptome

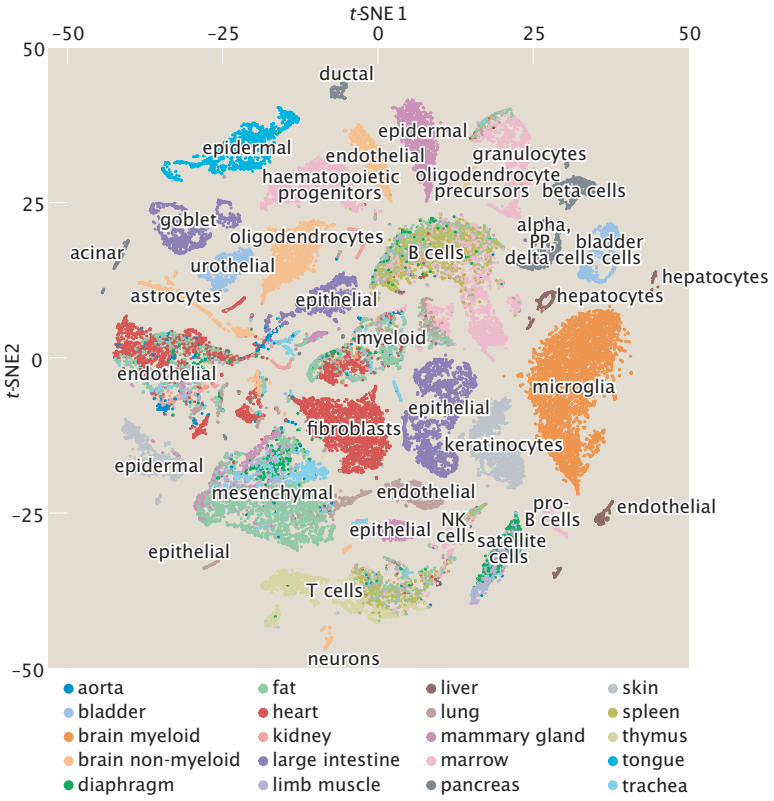

Figure 1: Graphical representation of the *Tabula muris* single-cell sequencing data. Individual cells of different organs in the mouse were subjected to single-cell transcriptome sequencing. Each dot represents a single cell, with its high-dimensional gene expression vector reduced to a two t-SNE lower dimensional representation. Clustering and manual annotation reveal different tissues and cell types. Adapted from The Tabula Muris Consortium et al., *Nature* 562:367-372, 2018.

was sequenced. Here, the extremely high dimensional data resulting from single-cell RNA sequencing (i.e., the number of mRNA molecules corresponding to each of $\approx$ 10,000 genes in each cell) was projected onto two dimensions using methods we will later explore. Further, once this projection is performed, cells are grouped in clusters. The idea is that cells within a cluster share much of their gene expression profile and are therefore identified as unique cell types corresponding to different tissues within the mouse. In this problem, we will attempt to build some intuition for how this identification of unique cell types is achieved by working with a synthetic transcriptome that we build ourselves using our understanding of the constitutive promoter. Obviously this is a caricature of the real situation where most genes are *not* constitutively expressed.

(a) Let's start by creating a mental picture of the high dimensionality of single-cell sequencing data by picturing how this data is stored. Specifically, think of a matrix $\mathbf{G}$ where you store the RNA counts for 10,000 genes measured in 1,000 cells where each row of the matrix corresponds to a given cell. How many rows and columns would this matrix have? Draw this matrix schematically, clearly indicating what each dimension of the matrix represents. Further, identify the gene expression vector that corresponds to the number of mRNA molecules detected for all species in cell number 1.

(b) To begin to get a feeling for this kind of data, we imagine an experiment on cells containing only two genes. These cells can adopt three different fates based on the expression state of these genes (i.e., low/low, low/high and high/high). Further, let's assume that these two genes are constitutively expressed, and that low and high gene expression levels correspond to an average of 10 and 35 mRNA molecules per cell, respectively. To remind ourselves of what the null hypothesis for constitutive promoters looks like, write the chemical master equation for a constitutive promoter and show that solving this equation in steady state results in a Poisson distribution. In the case of the low and high expression levels, give the formula for the specific Poisson distribution for those two cases.

(c) Plot histograms of the number of mRNA molecules of gene 1 and gene 2 for each cell type, assuming 1,000 cells of each type. This means that you will invoke the Poisson distribution you derived in the previous part of the problem and use it to describe the distribution of mRNA counts for the different cell types.

(d) Generate a synthetic transcriptome matrix $\mathbf{G}$ with 1,000 cells of each type (for a total of 3,000 cells in your dataset) by sampling from the Poisson distributions that you derived above. Make a plot of this low-dimensional synthetic transcriptome data set consisting of number of mRNA molecules of gene 2 vs. number of mRNA molecules of gene 1, where each dot within the plot corresponds to an individual cell.

Now, we will imagine that we are given this transcriptome data without any more information than the fact that there should be three cell types within it. Note that in reality we will rarely have information about number of cell types within a sample a priori. However, this is a good first step toward building intuition about the challenges of analyzing single-cell sequencing data.

In order to find cell types in our synthetic transcriptome, we will resort to so-called k-means clustering. The steps of this algorithm are illustrated in figure 2 and can be enumerated as follows:

1. The transcriptome data is plotted. In this case, because we only have two genes, this corresponds to a two dimensional plot of the number of mRNA molecules of gene 2 as a function of the number of mRNA molecules of gene 1 for each single cell. A set of N random points within this data set are then selected, with N being the number of clusters we are trying to identify. These N points will be called the centroids.

2. The distance of every data point to the centroids is calculated. Each data point is assigned to its closest centroid. This is our first approximation to the assignment of cells to our three clusters.

3. Based on the categorization of data points, new centroids are calculated. For each cluster, calculate their corresponding centroids by taking the average values of expression for the two genes.

4. Data points are reassigned to their closest centroid. This means that we now need to take every data point and compute the distance to all three updated centroids and then to assign them to the centroid they are closest to.

5. Steps (3) and (4) are repeated until convergence is achieved.

(e) Write a k-means algorithm to find 3 clusters in your synthetic transcriptome data set. In doing so, generate intermediate plots for the iterations of the algorithm such as those shown in Figure 2.

(f) One of the biggest drawbacks of k-means clustering is that we need to commit to a given number of clusters in advance. Explore what happens if you tell your algorithm to look for two and four clusters instead of three. Document some of the final answers from the algorithm and comment on why it converged to that answer. Comment on how all of these answers correspond to what you actually know about the system given that you generated the transcriptomes!

Finally, it is important to note that all algorithms are limited in the sense that they require commitments by specifying parameters. In k-means, we had to commit to a number of clusters. However, there are other approaches to finding clusters that do not require specifying cluster number a priori such as DBSCAN.

(g) Read about DBSCAN and explain how it works by drawing a graphical example (this can be in cartoon form). For this algorithm, what are the parameters we need to commit to?

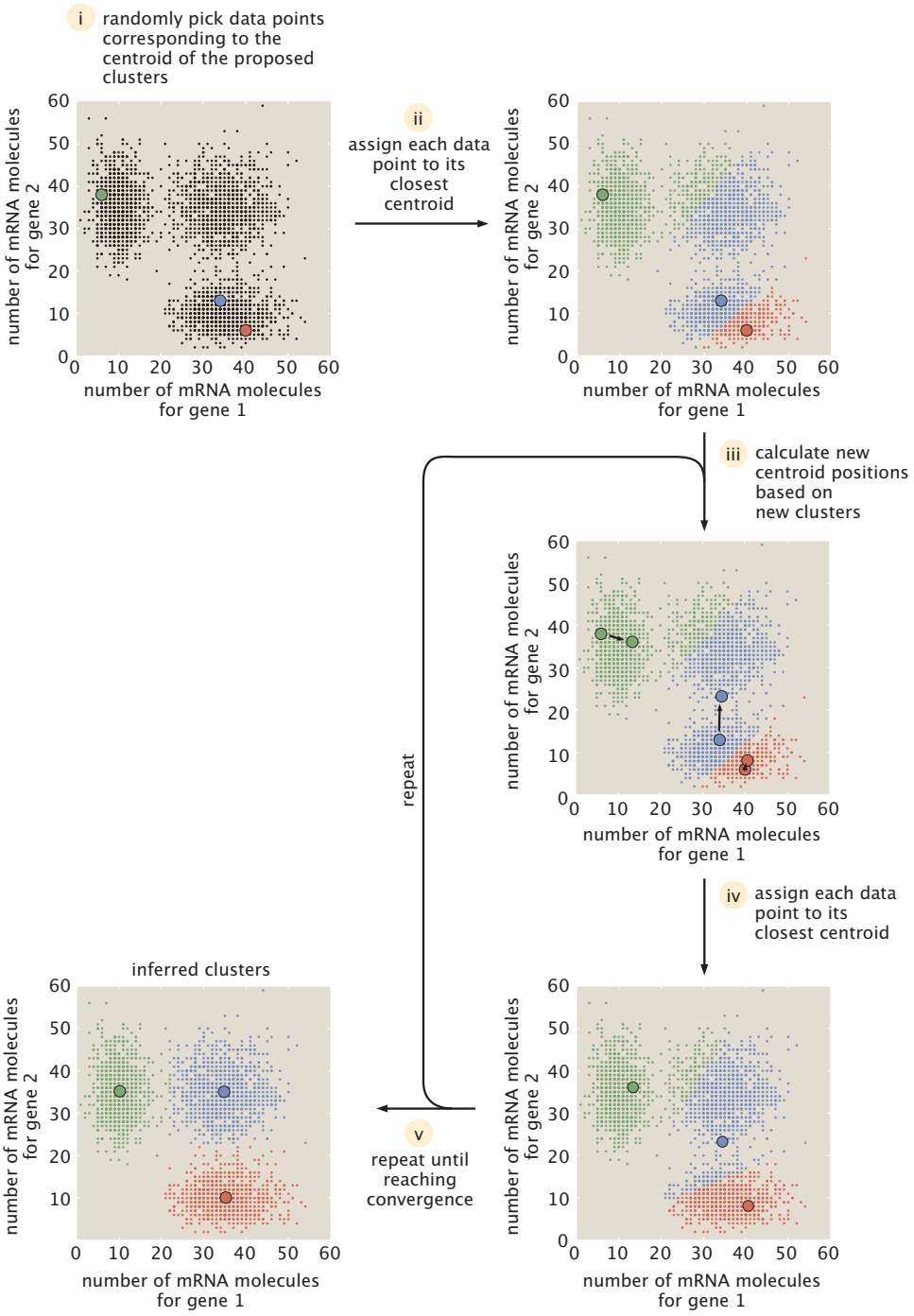

Figure 2: The k-means clustering algorithm. (i) A set of N points are chosen randomly from the dataset to become the centroids of the N clusters to identify. (ii) Each data point is assigned to its closest centroid. (iii) New centroids are calculated for each new cluster. (iv) Data points are reassigned to their new centroids. By iteratively repeating steps (iii) and (iv) convergence can be ultimately reached.

2 Mutation correlation and physical proximity on the gene

Do problem 4.4 from PBoC2 shown in Figure 3. You might find it useful to read section "Flies and the Rise of Modern Genetics" starting on page 170 of PBoC2.

4.4 Mutation correlation and physical proximity on the gene

In Section 4.6.1, we briefly described Sturtevant's analysis of mutant flies that culminated in the generation of the first chromosome map. In Table 4.2, we show the crossover data associated with the different mutations that he used to draw the map. A crossover refers to a chromosomal rearrangement in which parts of two chromosomes exchange DNA. An illustration of the process is shown in Figure 4.26. The six factors looked at by Sturtevant are B, C, O, P, R, and M. Flies recessive in B, the black factor, have a yellow body color. Factors C and O are completely linked, they always go together and flies recessive in both of these factors have white eyes. A fly recessive in factor P has vermilion eyes instead of the ordinary red eyes. Finally, flies recessive in R have rudimentary wings and those recessive in M have miniature wings. For example, the fraction of flies that presented a crossover of the B and P factors is denoted as BP. Assume that the frequency of recombination is proportional to the distance between loci on the chromosome.

Reproduce Sturtevant's conclusions by drawing your own map using the first seven data points from Table 4.2.

Keep in mind that shorter "distances" are more reliable than longer ones because the latter are more prone to double crossings. Are distances additive? For example, can you predict the distance between B and P from looking at the distances B(C,O) and (C,O)P? What is the interpretation of the two last data points from Table 4.2?

Table 4.2: Fraction of crossovers of six sex-linked factors in *Drosophila*. (Adapted from A. H. Sturtevant, *J. Exp. Zool.* 14:43, 1913.)

Factors	Fraction of crossovers
BR	115/324
B(C,O)	214/21736
(C,O)P	471/1584
(C,O)R	2062/6116
(C,O)M	406/898
PR	17/573
PM	109/458
BP	1464/4551
BM	260/693

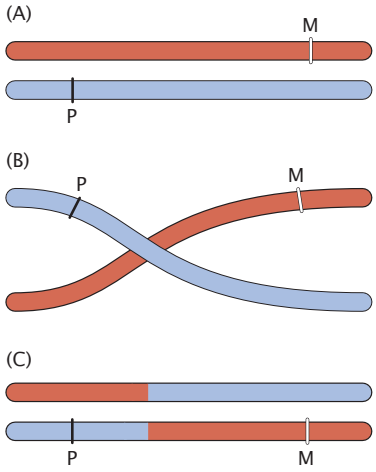

Figure 4.26: Crossing over of chromosomes. (A) Chromosomes before crossing over showing two loci labeled P and M. (B) Illustration of the crossing-over event. (C) Chromosomes after crossover.

Figure 3: Problem 4.4 from PBoC.

3 Counting Proteins with Partitioning Statistics

One of the great challenges in quantitative cell biology is to be able to turn the fluorescence values obtained from fusions to proteins to an actual absolute number of proteins. While there are many ways to "calibrate" such measurements using standards of a known concentration, in this problem, we will explore how we can use bacterial cell division, pure thought and the binomial distribution in order to calibrate a fluorescent protein.

(a) Begin by reading the paper by Rosenfeld *et al.* entitled "Gene Regulation at the Single-Cell Level" (posted on the website with the homework) and write a one paragraph commentary on the paper with special reference to how they used the binomial partitioning as a way to count repressor proteins. What is the experiment they did and what were they trying to learn?

In the rest of the problem we work out for ourselves the ideas about binomial partitioning introduced in the Rosenfeld *et al.* paper in order to consider the concentration of proteins as a function of time in dividing cells. In particular, the point of this problem is to work out the concentration of protein given that we start with a single parental cell that has N copies of this protein. In the Rosenfeld experiment, at some point while the culture is growing, the production of the protein is stopped by providing a chemical in the medium and then the number of copies per cell is reduced as a result of dilution as the cells divide.

Interestingly, this problem opens the door to one of the most important themes in physics, namely, that of fluctuations. In particular, as the cells divide from one generation to the next, each daughter does not really get $N/2$ copies of the protein since the dilution effect is a stochastic process. Rather the partitioning of the N proteins into daughter cells during division follows the binomial distribution. Analyzing these fluctuations can actually lead to a quantification of the number of copies of a protein in a cell.

(b) We think of the N copies of the protein as being divided between the two daughters with N_1 going to daughter 1 and $N - N_1$ going to daughter 2. Explain how the probability of N_1 proteins going to daughter cell one is given by the binomial distribution

$$P(N_1, N) = \binom{N}{N_1} p^{N_1} q^{N-N_1}, \tag{1}$$

where the probability of a protein going to daughter cell 1 is p, and the probability of a one protein going to daughter 2 is $q = 1 - p$. For your explanation you can choose to show a formal mathematical derivation, or qualitatively walk us through the meaning of each term in the equation. Remember that, while for most of the course we could use the "stadium seating" approximation to think about how to place N_1 spectators in N seats, here N and N_1 are of comparable magnitude. This situation, which already encountered in the context of the DNA entropic spring, calls for the binomial coefficient $\binom{N}{N_1}$.

We can also calculate the mean of the probability distribution (also called the first moment

of the distribution) by invoking a cool trick using the derivative with respect to p

$$\langle N_1 \rangle = \sum_{N_1=0}^{N} N_1 \binom{N}{N_1} p^{N_1} q^{N-N_1} = p \frac{\partial}{\partial p} \sum_{N_1=0}^{N} \binom{N}{N_1} p^{N_1} q^{N-N_1}. \tag{2}$$

This equation can be rewritten as

$$\langle N_1 \rangle = p \frac{\partial}{\partial p} \left((p+q)^N \right) = p N_{mother} (p+q)^{N-1}, \tag{3}$$

where we made use of the fact that

$$\sum_{N_1=0}^{N} P(N_1, N) = (p+q)^N. \tag{4}$$

Using $p + q = 1$, Equation 3 leads to

$$\langle N_1 \rangle = pN. \tag{5}$$

(c) Work out the expected averaged fluctuations squared in the partitioning process after each division by noting that the averaged fluctuations can be written as $\langle (N_1 - N_2)^2 \rangle$, where N_1 and N_2 are the number of proteins that end up in daughter cells 1 and 2, respectively. Show that, if $p = q = 0.5$, the partitioning error is given by $\langle (N_1 - N_2)^2 \rangle = N$. To make this possible, use the derivative trick twice such that

$$\langle N_1^2 \rangle = \sum_{N_1=0}^{N} N_1^2 \binom{N}{N_1} p^{N_1} q^{N-N_1} = p \frac{\partial}{\partial p} \left[p \frac{\partial}{\partial p} \left(\sum_{N_1=0}^{N} \binom{N}{N_1} p^{N_1} q^{N-N_1} \right) \right] \tag{6}$$

as well as the result $\langle N_1 \rangle = pN$ described above. In addition, use the fact that $N = N_1 + N_2$, in order to calculate the average partioning error as

$$\langle (N_1 - N_2)^2 \rangle = \langle [N_1 - (N - N_1)]^2 \rangle = \langle (2N_1 - N)^2 \rangle. \tag{7}$$

Remember that $\langle N \rangle = N$, as N is a constant in our problem.

(d) Next, look at the Rosenfeld paper and explain how measuring fluorescence variations can be used to calibrate the exact number of copies of the fluorescent protein in a cell. Specifically, assume that the fluorescence intensity in each cell can be written as $I = \alpha N$, where α is an as-yet unknown calibration factor and N the number of proteins in the cell. Explain what this equation means and why you think it is justified. Derive an expression relating I_1, I_2 and I_{tot} using the result of part (c). Make a qualitative schematic showing a plot of $\langle (I_1 - I_2)^2 \rangle$ versus I_{tot} and explain how to get the calibration factor α from this plot. Note that we're asking to draw up an explanation, not to actually make a plot with Python..

(e) Now we are going to repeat the Rosenfeld experiment numerically in order to *fit* the calibration factor. Consider a fluorescent protein such that the calibration factor between

the intensity and the number of fluorophores is 50, that is $I = 50N$. Generate intensity data by choosing $N_1 + N_2 = 10, 50, 100, 1000$ and 5000 and for each case, "partition" the proteins from the mother cell to the two daughters 100 times (i.e. as if you are looking at 100 mother cells divide for each choice of the protein copy number).

To make this possible, flip a coin for each molecule in order to decide whether the molecule is going to daughter cell 1 or 2. Specifically, for every molecule to be partitioned, draw a random number between 0 and 1 using the Python ramdom.randint function (remember to import the `random` package). Then, use the `if` function to decide whether a molecule will be partitioned to daugther cell 1 or 2.

Finally, make a plot of the resulting $\langle (I_1 - I_2)^2 \rangle$ vs I_{tot} just as we did analytically in the previous problem. What I mean is that you need to make a plot of all of your simulation results. Then, do a fit to your "data" using a numpy function (see the note below) and see how well you recover the calibration factor that you actually put in by hand. Plot the fit on the same graph as all of the "data".

Note: You can use `numpy.polyfit` to perform a linear fit to your "data" using the syntax `numpy.polyfit(x, y, deg)` where `x` is the data x-coordinate, `y` is the data y-coordinate, and `deg` is the degree of the polynomial you'd like to fit to your data (for instance, you would use `deg = 1` for a linear fit). You can also use `numpy.linalg.lstsq` if you'd rather phrase the problem as a matrix equation (this is reasonably simple to do as well, and an example of a linear fit performed using this function is provided in the Numpy documentation linked to above).