BIM 105
Fall 2024

Lab 4: The Poisson Distribution and RNA-Seq

Instructions: Read the prompts and use MATLAB to answer the questions. Submit your solution as a single script. A template script is available on Canvas.

Part 1. Poisson Distribution

The Poisson distribution tells us the probability of a certain number of events occurring in a fixed interval of time and/or space. It's an approximation of the Binomial distribution where n is large and p is small. When these conditions are met, the Poisson distribution provides a robust approximation to the Binomial, and it's defined as

$$\text{Prob}(k \text{ events}) = e^{-\lambda}\frac{\lambda^k}{k!} \qquad \text{where } k \in \mathbb{N} \cup \{0\} \text{ and } \lambda > 0.$$

Deriving the Poisson from the Binomial distribution starts with defining a parameter $\lambda = np$. This number corresponds to the "rate" of events occurring in your system. For example, if a typist types 10,000 characters per page and each character has a 0.001 probability of being a mistake, λ would be equal to 10 mistakes per page. The Poisson distribution provides the probability of a page having k mistakes, given λ.

Recall that the Binomial distribution gives the probability of a number of "successes" in a certain number of trials, n, given that a success occurs with probability p and trials are independent.

The expected outcome of X is $\mathrm{E}[X] = \lambda$, but perhaps more interestingly the variance of a Poisson random variable is also λ (i.e., $\mathrm{Var}(X) = \lambda$).

Question 1: Use MATLAB to draw 1000 samples of a Poisson random variable with the function **`poissrnd`**. Assume that $\lambda = 10$. What are the mean and variance of your 1000 samples?

Solution:

```
r = poissrnd(10, 1000, 1);
mr = mean(r); = 9.8650
vr = var(r); = 10.2530
```

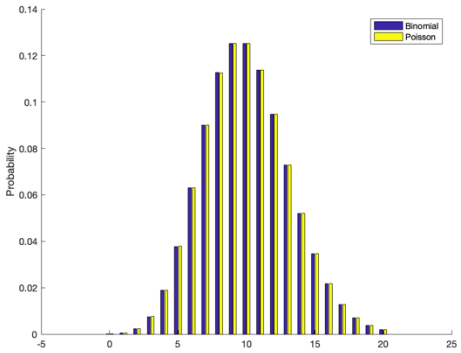
Simulation result can vary each time

Question 2: Use $n = 10{,}000$ and $p = 0.001$ to plot a Binomial distribution over $k \in [0, 1, \ldots, 20]$. Then, plot a Poisson distribution with $\lambda = np = 10$ over the same interval. Do the two distributions agree? (*Hint: Use* `binopdf(k,n,p)` *and* `poisspdf(k,lambda)` *to generate the points of the distributions*)

Solution:

```
n = 10000;
p = 0.001;
points = 0:1:20;
points_bino = binopdf(points, n, p);
points_poiss = poisspdf(points, n*p);
figure;
hold on
bar(points, [points_bino' points_poiss']);
ylabel('Probability')
legend('Binomial', 'Poisson')
```

The Poisson and binomial distribution are very similar and agree with one another.

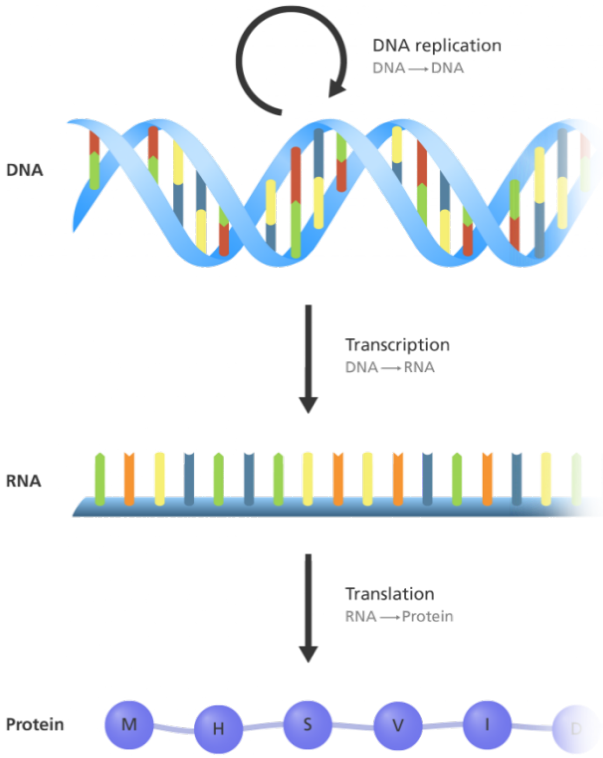


The distributions are nearly identical.

Part 2. RNA-Seq

Background

DNA contains the genetic information that cells need to grow, develop, function, and reproduce. In order to use the information encoded in their DNA, cells have enzymes that copy DNA sequences into another molecule called RNA. This process of reading a DNA sequence into RNA is called transcription. After transcription, the RNA molecule is processed with other enzymes to produce the protein encoded by the gene in the original DNA in a step called translation.



Source: YourGenome.com

(https://www.yourgenome.org/facts/what-is-the-central-dogma)

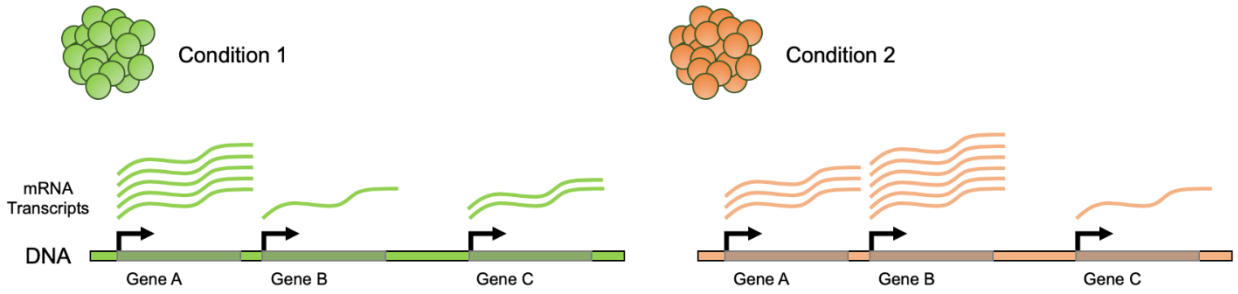
A cell does not equally utilize all of the genes coded within their DNA. Depending on a cell's state, different proteins are needed to maintain cellular function. As such, certain genes will be transcribed into RNA more frequently if the proteins associated with those genes are in high demand. How much a gene is transcribed into RNA and subsequently translated into a protein is referred to the gene's "expression."

The regulation of gene expression in cells is a complex process. The subject has been the center of research studies for over half a century. In short, cells have multiple mechanisms by which genes can be controlled, and all of them are not fully understood. Proteins in the nucleus can directly bind DNA to promote or inhibit a gene; DNA itself can be chemically altered in the nucleus to effect expression; large-scale expression changes can be induced by certain hormones; and many other mechanisms affect gene expression.

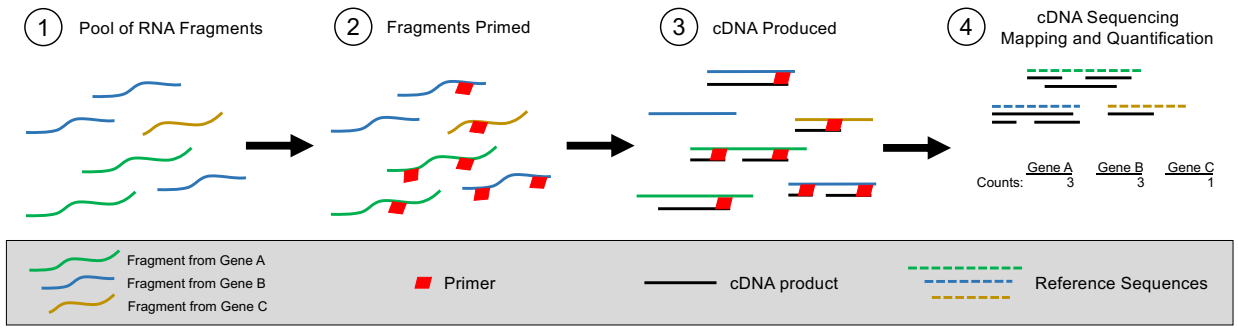
To learn about the role a particular protein, gene, or stimuli plays in the cell, researchers often perform what is called a "differential analysis" study of gene expression. Generally speaking, a cell will be placed in two different conditions, and the relative amount of gene expression for cells within each condition is determined. The changes in gene expression can then be used by researchers to disentangle how the feature of interest affects cells at the gene level.

An experiment that measures the amount of expression for each gene is RNA-Seq. The experiment works by sequencing RNA fragments in a cell, similarly to how we sequence DNA to determine its sequence. In this case, however, sequencing the entire pool of RNA in cells allows us to infer the relative abundance of genes. For example, consider the illustration below, which shows cells under two conditions with different gene expression profiles. In Condition 1, Gene A is producing many RNA fragments. This gene is highly expressed relative to Gene B and Gene C. In Condition 2, however, the genes are expressed at different rates. Gene B seems to be highly

expressed, while Gene A and Gene C are expressed at a lower rate. The subsequent changes in abundance of RNA fragments is what can be measured during an RNA-Seq experiment.



In order to consider the statistical properties of an RNA-Seq experiment, we first need to describe more of the details of how the experimental procedure works. A simplified schematic that highlights the features relevant to this lab is included below. First, we start with a pool of RNA fragments that we wish to quantify. Secondly, the RNA fragments are mixed with primers which are required to initiate the sequencing reaction. The primers bind the RNA fragments in a random fashion. Third, reverse transcriptase (RT) is used to reverse transcribe the primed RNA fragments into their complementary DNA (cDNA) products. The cDNA products are then sequenced and mapped to a reference genome to determine which gene each fragment is associated with. The number of fragments mapped to a gene is tabulated and is referred to as "counts" for that gene.



Assumption

A Simplified Simulation

Let's consider each step in more detail, starting with what comprises the input RNA. Typically, it's a diverse pool of RNA fragments extracted from a group of living cells. This pool of RNA would contain millions of RNA fragments originating from possibly thousands of different genes. For our purposes here, though, we will simplify some aspects of the input RNA pool. Let's assume that the input is a collection of N RNA fragments from G different genes. Suppose that a gene i is responsible for n_i of the total fragments, meaning that $N = \sum_i^G n_i$. Further assume that all fragments produced by a gene i are of identical length t_i. The total number of RNA nucleotides in the pool of fragments is thus $\sum_i^G n_i t_i$.

Under these assumptions, we can encode the pool of RNA with two vectors, one to indicate the length of each gene and another to indicate the number of fragments from each gene. Let's use a simple example in which a pool of RNA from five genes is used. Let the gene lengths and number of fragments be the following

```
% 5 transcripts in simulation
ti = [200 300 1000 500 450]; % 5 genes, each with a particular length
ni = [5 3 5 7 10]; % Copy numbers of each transcript
```

In order to detect a fragment, we first need to add primers to the RNA. Primers are molecules which bind to the RNA to initiate transcription of the complementary sequence. In our simulations, we will assume that primers have a length of 6 nucleotides and bind all possible sites with equal probability. The length of the primer is important because it constrains where along the RNA a primer may bind. It can only bind where all six nucleotides are available (i.e., where the primer doesn't hang off the edge of a fragment). This means if a fragment has length t, there are $(t+1)-6=t-5$ sites where a primer can bind. We assume the primers may bind any viable site with equal probability and thus the occurrences of binding can be thought of as a uniform sampling of available sites. For instance, if we wanted to simulate the binding of 200 primers, we could say:

```
% Primers
plen = 6; % Length

% Enumerate possible binding sites
binding_sites = (ti+1-plen).*ni; % Number of binding sites in pool, per gene

% Probability that a primer binds to each gene
probs = binding_sites./sum(binding_sites);

np = 200; % How many primers we are going to simulate

% Sample transcripts that produce fragments
binds = 1+sum(repmat(rand(1, np), length(ti), 1)>repmat(cumsum(probs'), 1, np), 1);
```

binding_sites = [975 885 4975 3465 4450]

probs = [0.0661 0.0600 0.3373 0.2349 0.3017]

generate a 5x200 matrix the cumulative probs of each gene, rows are identical

generate a 5x200 matrix that consist of 0-1 random number, rows are identical
each rpw correspond a gene

sum(…,1): sum over the rows

The result is a vector **`binds`** which contains a list of genes corresponding to the fragments bound by our 200 primers.

binds: an array, each element corresponds to one of the 200 primers, indicating which gene that primer has bound to

Genes with more binding sites (either due to longer transcripts or higher copy numbers) have a higher chance of attracting a primer.

The next step is reverse-transcription of the primed fragments into cDNA. The enzyme of interest here is reverse transcriptase (RT), which starts from a primed site to produce a sequence complementary to the RNA fragment. This process is an imperfect procedure as the enzyme has a tendency to fall-off the strand as it progresses. This means the length of the reverse-transcribed fragment is itself a random variable. We will assume that RT has probability $p=0.01$ of dropping off at each nucleotide, and subsequently that the length of a cDNA fragment is random variable $L\sim\text{Geom}(p)$. We will also need to constrain this length so that if the RT reaches the end of the original strand, elongation stops. In MATLAB we can perform this procedure with:

```
% Initialize fragments stemming from each binding site
frags = zeros(np, 3);                    see it as a dictionary, each column store info

p = 0.01; % Probability of RT falling off

for i = 1:length(binds) % For each primer...

    frags(i, 1) = binds(i); % Save transcript   % 1st col: save which transcript (gene) the primer binds to
    t = ti(binds(i)); % Transcript length      % Length of the transcript to which the primer binds

    nuc = randi(t); % Binding position      % Random starting position for RT on the transcript
    frags(i, 2) = nuc; % Save position      % 2nd col: save binding position
    len = 6 + geornd(p); % Length of reverse transcribe (including primer)

    % Handle if fragment length goes beyond end of transcript
    if nuc + len > t
        len = t-nuc;      % if it exceeds the end of the transcript: overall length of transcript - before binding position length
    end

    frags(i, 3) = len; % Length of fragment      % 3rd col: length of fragment of cDNA

end
```

The result of this is an array **`frags`** where each row corresponds to a cDNA fragment. The first column holds which gene the fragment corresponds to, the second column holds the initial binding position, and the third column holds the final length of the reverse-transcribed fragment.

From here, the experiment is almost complete. What remains is the sequencing of the cDNA fragments, the mapping of their sequences to a reference genome, and couting the number of fragments per gene. For the purposes of this lab, we will assume that the cDNA sequencing is performed perfectly (in reality, we know that this is not always the case). In regard to mapping, only sequences which are length 30 or longer can be reliably mapped to unique sites. Therefore, we will only detect fragments in **`frags`** that are 30 or greater in length. We can obtain these fragments with

```
detected_frags = frags(frags(:, 3) >= 30);
```

identify frag arrays that 3rd col > 30

Finally, we need to enumerate how many fragments ("counts") came from each of the five genes. To do this, we can call the built-in function **`hist`** to tabulate how many times each gene appears in our detected fragments.

```
counts = hist(detected_frags, 1:1:length(ti));
```

This simulation process has been pre-coded into a function **`rna_seq_sim(ti, ni, np)`**. Download the function from Canvas and use it to explore RNA-Seq with the following questions.

Question 3: Simulate an RNA-Seq experiment with the parameters from the prompt above:

```
ti = [200 300 1000 500 450];
ni = [5 3 5 7 10];
np = 200;
```

Solution:

```
ti = [200 300 1000 500 450];
ni = [5 3 5 7 10];
np = 200;
counts = rna_seq_sim(ti, ni, np);
```

Relative abundance

Suppose that we wanted to compute the relative abundance of each gene. Because each gene has a different RNA fragment length, we need to account for the fact that longer RNA fragments will be primed more often than smaller fragments. Also recall that the entire length of an RNA fragment is not available for primer binding – the "effective length" of a gene i is $t_i - 5$. Thus, the relative proportion of fragments from a particular gene i is computed as:

$$p_i = \frac{c_i/(t_i - 5)}{\sum_j^G c_j/(t_j - 5)}$$

normalized observed gene counts by the its effective length of the gene. This normalization is crucial because it allows for a fair comparison between genes of different lengths. Without this adjustment, longer fragments would inherently have a higher count simply due to having more potential binding sites, not necessarily because they are expressed more abundantly.

where c_i is the observed counts for gene i.

Question 4: Use the equation above to compute the relative abundance of fragments for each of the five genes from your results of Question 3.

Solution:

```
pg = (counts./(ti-5))/(sum(counts./(ti-5)));
= (answers will vary) [0.12, 0.10, 0.15, 0.30, 0.32]
```

Question 5: We know the underlying abundance of fragments in the simulation is **`ni = [5, 3, 5, 7, 10]`**, meaning that the underlying relative abundance (**`ni/sum(ni)`**) is **`[0.167, 0.100, 0.167, 0.233, 0.333]`**. How do these values compare to your solution from Question 4?

Solution:

The results of the experiment seem to agree with the underlying abundances.

The normalized abundance of the fragment counts is very similar to the actual proportions.

Up until now we've only been concerned with results from a single experiment. However, especially in biological systems, it's often necessary to perform several experiments to more strongly assess differences across conditions. For this reason, researchers are often interested in modelling the *distribution* of counts for a gene or genes across multiple experiments.

Question 6: Perform 1000 experiments using the parameters from above. Save the counts for the first gene (the gene with length 200) from each experiment and plot a histogram of the 1000 observations.

Question 7: Assume that the observed distribution is Poisson. The best fit of the data is a Poisson distribution where λ is the sample average. Compute the mean of the counts and use this parameter to generate a Poisson distribution to overlay onto your plot from Question 6. How well do the data agree with a Poisson distribution?

Solution:

```
Ns = 1000;
counts_A = NaN(Ns, 1);
for i = 1:Ns
    counts = rna_seq_sim(ti, ni, np);
    counts_A(i) = counts(1);
end
figure;
hold on
h = histogram(counts_A, 'Normalization', 'pdf');
```

The data fits well with a Poisson distribution based on the graph

In reality, there is a significant amount of *biological variation* across experiments. This means that over successive experiments, the underlying abundance of each gene is not held constant. Rather, the underlying abundances of RNA fragments differ due to uncontrollable circumstances surrounding the cell.

We can incorporate these differences in our simulations to investigate what effect, if any, they have on the experimental results. Instead of using `ni = [5 3 5 7 10]` for every simulation, we can let the number of fragments from each gene be Poisson random variables with averages corresponding to the values in `ni`. In other words, we will call a simulation with: `counts = rna_seq_sim(ti, poissrnd(ni), np)`. This means the underlying fragment abundances will vary, but the average abundance over many simulations will remain at the same level as the original example.

Not fixed abundances for each fragment, but wriggle around that number

Question 8: Perform 1000 simulations of the RNA-Seq experiment with the same parameters as above, but replace `ni` with a random Poisson sample for each simulation as described. Plot a histogram of the counts over the 1000 simulations for the first gene.

Solution:

```
Ns = 1000;
counts_A = NaN(Ns, 1);
for i = 1:Ns
    counts = rna_seq_sim(ti, poissrnd(ni), np);
    counts_A(i) = counts(1);
end
figure;
hold on
h = histogram(counts_A, 'Normalization', 'pdf');
```

Question 9: Assume that the observed distribution is Poisson. Compute the mean of the counts and use this parameter to generate a Poisson distribution to overlay onto your histogram. How well do the new data agree with a Poisson distribution?

Solution:

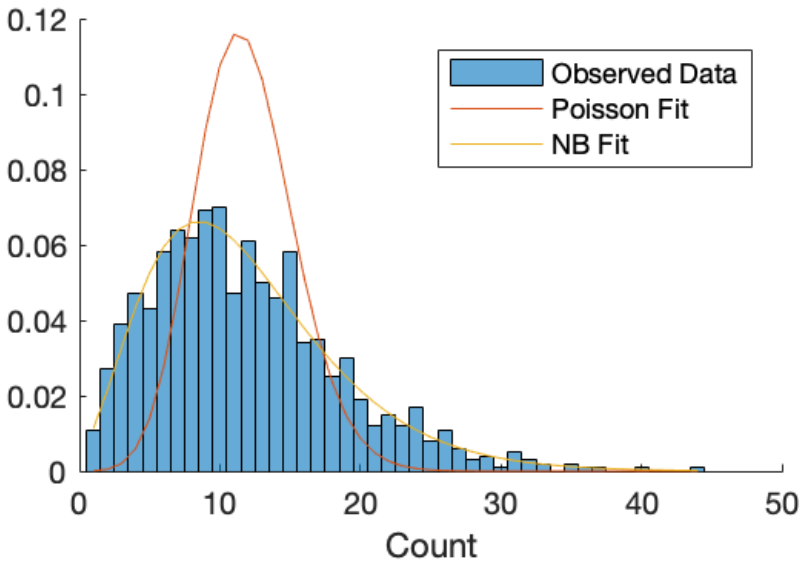
```
mc = mean(counts_A);
points = min(counts_A):1:max(counts_A);
plot(points, poisspdf(points, mc));
```

The data do not agree with a Poisson distribution.

Question 10: Researchers often model real RNA-Seq data as a Negative Binomial Distribution. In short, the Negative Binomial is a generalization of the Poisson that allows some flexibility for the variance to grow from the mean. Use **`nbinfit`** to fit a Negative Binomial to your counts from Question 8, and plot this fitted distribution onto your histogram of counts. Does this model capture the data better than the Poisson?

Solution:

```
params = nbinfit(counts_A);
plot(points, nbinpdf(points, params(1), params(2)));
```



The data agree much better with a NB distribution the negative binomial fits better

Question 11: The Negative Binomial can be derived mathematically by considering a random variable $X \sim \text{Poisson}(Y)$, where Y itself is a random variable (specifically, a "Gamma" random variable such that $Y \sim \text{Gamma}(\alpha, \beta)$, but the details of the Gamma distribution are beyond the scope of this lab). Given this, do you think a Negative Binomial is a suitable choice to model RNA-Seq counts? How might this relate to capturing biological variation?

Solution:

The Negative Binomial can be interpreted as a mixture of Poisson distributions where the parameter lambda is itself a random variable. This interpretation indicates the suitability of the distribution to capture the RNA-Seq process with biological variation.