Lab 07 Simulating Sequences

The goal of Lab 07 | Simulating Sequences is to use a Markov model to simulate realistic DNA sequences. In the first part of the lab you will quantify nucleotide and dinucleotide frequencies to determine whether a given sequence has dinucleotide biases. Then you will use a Markov chain to simulate random sequences with the same empirical distribution of dinucleotide frequencies as found in the observed sequence.

The lab is split into the following sections:

- Simulation
- Methylation

Assignment

Follow the instructions in this document and answer the questions in the cell below each question. Submit your answers by uploading a PDF file to gradescope. To generate the pdf, first export the notebook as HTML: >File, >Export to ..., >HTML. Then, open the HTML in a browser and use your browser to print to PDF.

Check to make sure all your cells have been run and the **results** displayed in the PDF file.

Reminder, provide comments for any code you write to ensure partial credit.

Simulation

Simulation is the imitation of a real-world process or system. The act of simulating something first requires that a model be developed; the model represents the key characteristics, behaviors and functions of the system or process. The model represents the system itself, whereas the simulation generates possible outcomes of the system.

Some simulations are **deterministic**, where the outcome is always the same from a given starting condition. Deterministic simulations can be used to explore how initial conditions affect the outcome of a complex process.

Some simulations are **stochastic**, where the outcome depends on a stochastic aspect of the model, typically encoded by random number generators. Stochastic simulations provide a way to assess the many potential outcomes of a system.

Traditionally, the formal modeling of systems has been via a mathematical model, which attempts to find analytical solutions enabling the prediction of the behaviour of the system from a set of parameters and initial conditions. Computer simulation is often used as an adjunct to, or substitution for, modeling systems for which closed form analytic solutions are not possible. There are many different types of computer simulation, the common feature they all share is the attempt to generate

a sample of representative scenarios for a model in which a complete enumeration of all possible states would be prohibitive or impossible.

Simulations in biology

Computer simulations provide a diverse resource for understanding biological systems. For example, they can be used to:

1. test or benchmark algorithms or computational methods,
2. test hypotheses that can't otherwise be tested,
3. determine whether a model can explain or predict empirical data.

Take influenza epidemiology as an example. Given a simple infectious disease model with parameters for the number of susceptible and resistant individuals, rate of transmission, etc, does this model predict the number flu cases in a given season (#3)? Or, given the number of flu shots and rate of resistance they provide how many additional flu cases would we expect to observed if no one received a flu shot (#2)? And finally, if you devise an algorithm that predicts the number of flu cases each year given previous year data, simulations can be used to assess performance of the model with limited empirical data (#1).

Methylation

DNA methylation is a process by which methyl groups are added to the DNA molecule. Methylation can change the activity of a DNA segment without changing the sequence. When located in a gene promoter, DNA methylation typically acts to repress gene transcription. DNA methylation is essential for normal development and is associated with a number of key processes including genomic imprinting, X-chromosome inactivation, repression of transposable elements, aging and carcinogenesis.

Two of DNA's four bases, cytosine and adenine, can be methylated. Cytosine methylation is widespread in both eukaryotes and prokaryotes, but the rate of cytosine DNA methylation can differ greatly between species.

Methylation of cytosine to form 5-methylcytosine occurs at the same 5 position on the pyrimidine ring where the DNA base thymine's methyl group is located. Spontaneous deamination of 5-methylcytosine converts it to thymine. This results in a T:G mismatch. Repair mechanisms then correct it back to the original C:G pair; alternatively, they may substitute G for A, turning the original C:G pair into an A:T pair, effectively changing a base and introducing a mutation.

CpG Sites

The CpG sites are regions of DNA where a cytosine nucleotide is followed by a guanine nucleotide in the linear sequence of bases along its 5' → 3' direction. CpG is shorthand for 5'—C—phosphate—G—3' , that is, cytosine and guanine separated by only one phosphate group; phosphate links any two nucleosides together in DNA. The CpG notation is used to distinguish this single-stranded

linear sequence from the CG base-pairing of cytosine and guanine for double-stranded sequences. The CpG notation is therefore to be interpreted as the cytosine being 5 prime to the guanine base. CpG should not be confused with GpC, the latter meaning that a guanine is followed by a cytosine in the 5' → 3' direction of a single-stranded sequence.

Cytosines in CpG dinucleotides can be methylated to form 5-methylcytosine. In mammals, methylating the cytosine within a gene can change its expression, a mechanism that is part of a larger field of epigenetics. Enzymes that add a methyl group are called DNA methyltransferases. In mammals, 70% to 80% of CpG cytosines are methylated.

CpG dinucleotides have long been observed to occur with a much lower frequency in the sequence of vertebrate genomes than would be expected due to random chance. For example, in the human genome, which has a 42% GC content, a pair of nucleotides consisting of cytosine followed by guanine would be expected to occur 0.21 * 0.21 = 4.41% of the time. The frequency of CpG dinucleotides in human genomes is 1%—less than one-quarter of the expected frequency. It was proposed that the CpG deficiency is due to an increased vulnerability of methylcytosines to spontaneously deaminate to thymine in genomes with CpG cytosine methylation. In mammalian genomes, CpG islands are typically 300-3,000 base pairs in length, and have been found in or near approximately 40% of promoters of mammalian genes. About 70% of human promoters have a high CpG content.

CpG Expectation

The expected number of CpG sites can be calculated by:
(number of C *number of G) / length of the sequence*
or
(frequency of C) (frequency of G) * length of the sequence

Python functions

Built in string functions: `str.lower()` `str.upper()`
Return a copy of the string with all the cased characters converted to lowercase.

In [1]:
```
s = 'AGCcct'
print( str.lower(s) )
print( str.upper(s) )
```

```
agccct
AGCCCT
```

Numpy function: `np.mean()`
Compute the arithmetic mean along the specified axis.

Numpy function: `np.std()`
Compute the standard deviation along the specified axis.

In [2]:
```
import numpy as np
```

```
a = np.array([[1, 2], [3, 4]])
np.mean(a)
```

Out[2]:
```
2.5
```

In [3]:
```
np.mean(a, axis=0)
```

Out[3]:
```
array([2., 3.])
```

In [4]:
```
np.mean(a, axis=1)
```

Out[4]:
```
array([1.5, 3.5])
```

In [5]:
```
np.std(a)
```

Out[5]:
```
1.118033988749895
```

Random module

This module implements pseudo-random number generators and associated functions for various distributions.
https://docs.python.org/3.7/library/random.html
A similar set of functions are available through the Numpy package.
https://docs.scipy.org/doc/numpy-1.15.1/reference/routines.random.html

Almost all module functions depend on the basic function random(), which generates a random float uniformly in the semi-open range `[0.0, 1.0)` .

`random.choices(population, weights=None, cum_weights=None, k=1)`
Return a k sized list of elements chosen from the population with replacement. If a weights sequence is specified, selections are made according to the relative weights. Alternatively, if a cum_weights sequence is given, the selections are made according to the cumulative weights. For example, the relative weights `[10, 5, 30, 5]` are equivalent to the cumulative weights `[10, 15, 45, 50]` . If neither weights nor cum_weights are specified, selections are made with equal probability.

`random.random()`
Return the next random floating point number in the range `[0.0, 1.0)` .

`random.choice(seq)`
Return a k sized list of elements chosen from the population with replacement.

In [6]:
```
import random as r
print( r.random() )
print( r.choice('AGCT') )
a = r.choices(['A', 'G', 'C','T'], [0.1, 0.4, 0.4, 0.1], k=50)
print( ''.join(a) )
```

```
0.5026772334808797
T
CGCCTGAGCGGCGGCACACGCCGCCCTGCCCCGGCTCCCGAGGCCGCGCG
```

Comparing observed and simulated numbers

The Central limit theorem:
Let $X_1, X_2, \ldots X_n$ be a random sample of size n from a distribution of X with mean μ and variance σ^2.
Then, for large n,

$$\frac{\bar{X} - \mu}{\sigma/sqrt(N)} \sim N(0,1) \qquad (1)$$

We can get the Normal function from `scipy.stats.norm` :
https://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.stats.norm.html#scipy.stats.norm

And, we can use the interval function to get a confidence interval given a mean and standard deviation:

```
stats.norm.interval(alpha, loc=0, scale=1)
```

Lets simulate (generate) a bunch of exponentially distributed random variables with $\lambda = 0.10$ and then see whether the true mean $1/\lambda$ is within the 95% confidence interval of estimated mean from the simulation.

In [7]:

```
from scipy import stats
from math import sqrt

N = 100
rexp = np.random.exponential(scale=10, size=N) # Scale = 1/rate or 1/lambda
mu = np.mean(rexp)
sigma = np.std(rexp)
print("Observed mean: ", mu)
ci = stats.norm.interval(0.95, loc=mu, scale=sigma/sqrt(N))
print("95% confidence interval: ", ci)

outside = 0
for i in range(1000):
    rexp = np.random.exponential(scale=10, size=N) # Scale = 1/rate or 1/lambda
    mu = np.mean(rexp)
    sigma = np.std(rexp)
    ci = stats.norm.interval(0.95, loc=mu, scale=sigma/sqrt(N))
    if (ci[0] > 10 or ci[1] < 10):
        outside += 1
print("Number of times the true value (10) is outside our confidence interval ba
print("We expect it to be outside 5% or 50/1000 times")
```

```
Observed mean:  8.648052349051115
95% confidence interval:  (7.110097653068429, 10.1860070450338)
Number of times the true value (10) is outside our confidence interval based on
1000 simulations:  60
We expect it to be outside 5% or 50/1000 times
```

Question 1

Write a function that takes a nucleotide sequence as input and outputs a vector (list) of counts for the number of occurrences of `['A','G','C','T','CG']`. Specificially, 'CG' is the number of occurrences of the dinucleotide 5'-CpG-3'. The function should work using sequences with either an uppercase or lowercase input sequence.

Use the function from Question 1 to calculate the observed number of 'CpG' dinucleotides present in the fasta file, lab07.fasta.

(2 points)

In [8]:
```
#
```

Question 2

What is the observed and expected number of 'CpG' dinucleotides in the sequence? What is the GC content, i.e. the frequency of G or C in the sequence?

(2 points)

In [9]:
```
#
```

Question 3

Simulate (generate) 100 sequences with the same length as lab07.fasta. Use the observed nucleotide frequencies in lab07.fasta for the simulation. For example if the frequency of 'G' is 25%, the probability of generating a 'G' is 25%.

Output the mean frequency of `[A, G, C, T]` across the simulated sequences and the mean frequency of 'CpG' dinucleotides in the simulated sequences.

Make sure the frequencies of A + G + C + T = 1

(2 points)

In [10]:
```
#
```

The simulated frequency of CpG sites should be much higher than what is observed in `lab07.fasta`. To improve our simulation we can use a first order Markov model.

Question 4

Write a function that takes a sequence as input and returns a first order Markov model of base composition (1st order refers to the probability of a transition depends only on the 1st state preceeding it, as opposed to the last two or three states). The returned Markov model should be a transition probability matrix `P[ i , j ]`, a 4x4 matrix of probabilities of moving to state j given a current state i, where states are `[A,G,C,T]`.

The transition matrix can be obtained by counting all of the observed transitions in the sequence and then dividing each row by its sum. For example, the sequence: ACGTACAA has two P(A,C) transitions and one P(A,A) transition and so the first row of the transition matrix would be [1/3,0,2/3,0] in the order A,G,C,T.

Use the function to calculate a transition matrix for `lab07.fasta` and print to the screen.

(2 points)

In [11]:
```
#
```

Question 5

What is the least frequent transition probability and what is its transition probability in the matrix from Question 5?

Given that lab07.fasta is from the human genome, explain why C to G transitions in the Markov model are rare?

Calculate the equilibrium frequency of `[A, G, C, T]` for the Markov model using the transition matrix. (Hint: review Lab05 and remember that you can get very accurate equilibrium frequencies without 'going to infinity'. How do you know your numbers are accurate? Calculate one more step and see if they change.)

Calculate the expected frequency of CpG sites for the Markov model using the transition matrix. (They should be very close to the observed frequencies).

(2 points)

In [12]:
```
#
```

Question 6

Simulate 100 sequences with the same length as lab07.fasta using your Markov model generated from lab07.fasta.

What is the mean and standard deviation of the number of occurrences of 'CpG' in the simulated sequences and are the observed values within one standard deviation of the mean simulated values. (Which would indicate the model adequately simulates the observed CpG frequency)

(4 points)

In [13]:
```
#
```

Question 7

Using your first order Markov chain as an expectation, what 3 bp sequence is most under-represented in lab07.fasta, and which is the most over-represented?

For example, the expected frequency of AAA in the first order Markov chain is 0.04357 and the observed frequency is 0.05091, so AAA is over-represented by 0.05091/0.04357 = 1.168 fold.

(4 points)

In [14]:
```
#
```

Question 8

If CpG sites are being eliminated by hypermutation, we would expect them to increase in frequency in a sequence that evolved without hypermutation of CpG sites.

Simulate evolution of the `lab07.fasta` sequence without high CpG mutation rates. Start with the observed sequence and generate 30,000 substitutions. The positions of the substitutions should be uniform across the sequence and substitutions at previously substituted sites should be allowed. Use the observed nucleotide frequencies to choose what substitution is made (but don't substitute 'A' for 'A', etc).

What is the frequency of CpG sites in the simulated sequence?

(2 points)

In [15]:
```
#
```

In [ ]:
```

```