On Matters Of Nature and Science

L A Liggett

2019-04-05

Contents

Chapter 1

Prerequisites

This is a *sample* book written in **Markdown**. You can use anything that Pandoc's Markdown supports, e.g., a math equation $a^2 + b^2 = c^2$.

The **bookdown** package can be installed from CRAN or Github:

```r
install.packages("bookdown")
# or the development version
# devtools::install_github("rstudio/bookdown")
```

Remember each Rmd file contains one and only one chapter, and a chapter is defined by the first-level heading `#`.

To compile this example to PDF, you need XeLaTeX. You are recommended to install TinyTeX (which includes XeLaTeX): https://yihui.name/tinytex/.

Chapter 2

Genetics and Genomics

2.1 Introduction

A haplotype block is a set of closely linked alleles or markers on a chromosome that tend to be inherited together over evolutionary time.

Across Eukaryotes, the frequency of recombination is inversely proportional to overall genome size. The result is that yeast have a recombination rate a few orders of magnitude higher than that of humans (She and Jarosz, 2018).

2.1.1 Subpoint

This is some sub info

2.1.2 Second subpoint

A haplotype block is a set of closely linked alleles or markers on a chromosome that tend to be inherited together over evolutionary time.

Across Eukaryotes, the frequency of recombination is inversely proportional to overall genome size. The result is that yeast have a recombination rate a few orders of magnitude higher than that of humans (She and Jarosz, 2018).

2.2 DNA Replication

When the origin of replication(s) is removed from bacteria or eukaryotes, growth and division is restricted or entirely eliminated, but in some strains of archaea like H volcanii, deletion of the origin of replication accelerates cell growth rates. It turns out that this archaea can use a process that is similar to homologous recombination to create a replication fork and replicate its chromosome (Hawkins et al., 2013).

2.3 Cloning

A macaque was the first primate to be cloned by SCNT (Liu et al., 2018).

2.4 Mutation Rate

Using whole-genome sequencing or next-gen sequencing to determine mutation rates by base, it appears that C>T mutations at CpG sites mutate at a frequency of 10-7 changes/cell division, and all other sites are within the range of 10-8-10-9 base changes per division (Arnheim and Calabrese, 2009, 2016; Campbell and Eichler, 2013; Ségurel et al., 2014).

Providing an example of how human mutation rates can differ by geographic origination, europeans compared to african/asian populations have a 1.6 increased mutation rate of a particular mutation (Harris, 2015).

It appears that humans have the highest germline mutation rate of all analyzed species (Lynch, 2016)

2.5 Mutation Hotspots

There are a number of sporadic mutation hotspots associated with disease incidence, like achondroplasia which has a sporadic incidence rate of 4.5 x 10-5 per generation (Arnheim and Calabrese, 2016; Waller et al., 2008). This disease originates from a single mutation in the FGFR3 gene at a mutation rate ~450 times higher than what would ordinarily be expected at a CpG site (Bellus et al., 1995; Rousseau et al., 1994; Shiang et al., 1994).

2.6 Mutation Detection

Pyrophosphorolysis-activated polymerization is a mutation detection method that can detect a single mutant molecule of DNA within 25,000 genomes (Liu and Sommer, 2004; Qin et al., 2007).

2.7 Genetic Modifications

Caffeine was cloned to allow for caffeine-deficient coffee and teas without the decaffeination process (Kato et al., 2000).

When a DNA-associating protein from tardigrades was cloned into mammalian cells, they became about 40% more tolerant to radiation (Hashimoto et al., 2016).

2.8 Sequencing Methods

Using a new sequencing method called sci-RNA-seq, the transcriptome of every cell of 762 cells in C Elegans was sequenced to yield single-cell sequencing results and transcriptome profiling of every cell in the body. The way this is done is by methanol fixing nuclei and then incorporating a UMI when converting to cDNA, then mixing cells again and incorporating another UMI when synthesizing the other strand (Cao et al., 2017).

It appears that DAPI does not increase sequencing error rates by Illumina sequencing (Leung et al., 2016).

One group came up with a method that is essentially identical to mine in which they use barcoded probes to detect leukemia but they tracked the mutation manually and ignored background (Wong et al. 2015)

2.9 Diagnostics

In Li and Snyder Cell 2018, the EHR from hospitals is used to integrate with a machine learning algorithm trained on aneurysm detection. Patients are then whole genome sequenced, and the genome sequencing plus the lifestyle of the individual on EHR is then used to predict if the person has an aneurysm. They were able to achieve pretty robust detection results that could then be used in a prediction setting in the clinic.

Chapter 3

JupyterLab

Here is a simple template that I use that controls a couple useful things when starting a new notebook.

```
import sys
sys.path.append('../util')

%reload_ext autoreload
%autoreload 2

from util import *
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns

sns.set_palette('pastel')
sns.set_style('ticks')
sns.set_context('paper', font_scale=1)
```

It is often convenient to have a notebook automatically refresh the imported libraries so that they can be modified while working on a JupyterLab notebook.

```
%reload_ext autoreload
%autoreload 2
```

To allow directory organization, dependcies can be separated into different directories and imported into a jupyter notebook using the following import statement.

```
import sys
sys.path.append('../util')
```

Chapter 4

Visualization

4.1 Matplotlib

Plotting a heatmap.

```
import matplotlib.pyplot as plt
import numpy as np
a = np.random.random((16, 16))
plt.imshow(a, cmap='RdBu'', interpolation='nearest')
plt.show()
```

Possible heatmap colors are:

```
Accent, Accent_r, Blues, Blues_r, BrBG, BrBG_r, BuGn, BuGn_r, BuPu, BuPu_r, CMRmap, CMRmap_r, Dark2, Da
Set1_r, Set2, Set2_r, Set3, Set3_r, Spectral, Spectral_r, Wistia, Wistia_r, YlGn, YlGnBu, YlGnBu_r, YlG
gist_stern, gist_stern_r, gist_yarg, gist_yarg_r, gnuplot, gnuplot2, gnuplot2_r, gnuplot_r, gray, gray_
twilight, twilight_r, twilight_shifted, twilight_shifted_r, viridis, viridis_r, vlag, vlag_r, winter, w
```

A simple venn diagram.

```
from matplotlib_venn import venn2
venn2(subsets = (3, 2, 1))
```

A more complicated venn diagram.

```
from matplotlib import pyplot as plt
import numpy as np
from matplotlib_venn import venn3, venn3_circles
plt.figure(figsize=(4,4))
v = venn3(subsets=(1, 1, 1, 1, 1, 1, 1), set_labels = ('A', 'B', 'C'))
v.get_patch_by_id('100').set_alpha(1.0)
v.get_patch_by_id('100').set_color('white')
v.get_label_by_id('100').set_text('Unknown')
v.get_label_by_id('A').set_text('Set "A"')
c = venn3_circles(subsets=(1, 1, 1, 1, 1, 1, 1), linestyle='dotted')
c[0].set_lw(1.0)
c[0].set_ls('dotted')
```

```
plt.title("Sample Venn diagram")
plt.annotate('Unknown set', xy=v.get_label_by_id('100').get_position() - np.array([0, 0.05]), xytext=(-
             ha='center', textcoords='offset points', bbox=dict(boxstyle='round,pad=0.5', fc='gray', al
                          arrowprops=dict(arrowstyle='->', connectionstyle='arc3,rad=0.5',color='gray')
                          plt.show()
```

4.2 Seaborn

Here is a general bar plot that includes some commonly used parameters.

```
# fits my 22 inch monitor
plt.figure(figsize=(19.17,11.98))
# order controls the display order of the samples
sns.catplot(x="Sample", y="Somatic", kind="bar", data=var_counts, order=labels);
# keeps x-axis labels, but eliminates the tick mark
plt.tick_params(labelbottom=True, bottom=False)
# trim off the x-axis
sns.despine(offset=10, trim=True, bottom=True)
# labels
plt.title('')
plt.ylabel('')
plt.xlabel('')
# manual control of xlabels
labels = ['Indiv_1-a','Indiv_2','Indiv_3','Indiv_1-b']
# control xtick order
plt.xticks(range(len(labels)), labels, rotation=45)
# control the number of x-ticks
plt.locator_params(axis='x', nbins=10)
# legend positioning
plt.legend(loc='upper right')
# log scale
plt.gca().set_yscale('log')
# this is better if neg values are needed
plt.gca().set_yscale('symlog')
# fit plot to display
plt.tight_layout()
plt.show()
# save figure with tight_layout
plt.savefig("test.svg", format="svg", bbox_inches="tight")
```

Signifance information can be added by including p-values and label bars using the following code.

```
x1, x2 = 0, 1 # columns to annotate on the plot
y2, y1 = 20, 15 # placement of the line and how for down the vertical legs go
plt.plot([x1,x1, x2, x2], [y1, y2, y2, y1], linewidth=1, color='k') # stats line
plt.text((x1+x2)*.5, y2+2, "p=0.09", ha='center', va='bottom') # p-value or sig
```

Chapter 5

Biology

5.1 General

Some helpful commands for genetic sequence.

```
from string import ascii_uppercase # python 3
from string import upper, lower # python 2
upper('tcga')
lower('TCGA')
title('tcga') # capitalize the first letter
```

5.2 Biopython

Reverse complement of sequence

```
from Bio.Seq import Seq
str(Seq(i).reverse_complement())
```

5.3 UCSC Genome Browser

Get sequence from UCSC genome browser

```
from subprocess import check_output, STDOUT
temp = check_output('wget -qO- http://genome.ucsc.edu/cgi-bin/das/hg19/dna?segment=%s:%s,%s' % (vcfObj.
```

5.4 Ref Genome

Get sequence from reference genome

```
from subprocess import check_output, STDOUT
temp = check_output('samtools faidx %s %s:%s-%s' % (ref, vcfObj.chrom, low, high), stderr=STDOUT, shell
finalSeq = ''
for line in temp.decode('UTF-8').split('\n'):
for line in temp.decode('UTF-8').split('\n'): # this is only necessary in python 3 to convert binary to
    if '>' not in line:
        finalSeq += line

finalSeq = finalSeq.upper()
```

5.5 Personal Information

```
# parse vcf file with parseline
if '#' not in line and 'chr' in line: # skip the info
# vcf handling
from parseline import VCFObj
# or
from util import VCFObj
vcfObj = VCFObj(vcfLine)
# available attributes: ao, dp, af, wt, var, chrom, location
```

Chapter 6

Data I/O

6.1 Reading Data Files

Opening .gz files

```
import gzip
for line in gzip.open('myFile.gz'):
    print line
```

6.2 Pickles

Writing data in pickle format

```
import pickle
p = open('principle.pkl', 'wb')
pickle.dump(principleData, p)
p.close()
```

Reading data in pickle format

```
import pickle
p = open('principle.pkl', 'rb')
principleData = pickle.load(p)
p.close()
```