**Project II**

**Deterministic and Stochastic Simulations in R**

For this project, you are going to generate some simulation tools in R to understand the dynamics of selection and drift. Results will be compared against theory to see if the simulations are behaving as predicted.

**R pointers:**

**Assign a variable**

x <- 5

x

[1] 5

**Adding a value to a variable**, example:

x <-x + 1

x

[1] 6

**Making a list(vector)**

x <- c(1,2,3,4,5)

x

1,2,3,4,5

**Calculations can be performed on a vector.**

y <- x/20

y

[1] 0.45 0.60 0.70 0.65 0.45 0.60 0.50 0.30 0.50 0.55

**Adding something to a vector**

x <- c() establishing a null vector

x <- c(x,10)

x

[1] 10

x <- c(x,10,20,30)

x

[1] 10 10 20 30

**If statements and logical operations**

if (test\_expression) {

statement

}

x <- 0

y <- 0

z <- 0

if(x==y) {z<-z+1} #**NOTE the double equals!**

z

[1] 1

if(x==z) {z<-z+1} #**NOTE the double equals!**

z

[1] 1

if(x==y) {z<-z+1}

z

[1] 2

Also, other logical operators:

x < y

[1] FALSE

z >x

[1] TRUE

not equals:

x != y

[1] FALSE

**For Loops**

for (i in 1:10) {...

}

Example:

x<-c()

for (i in 1:10) {

x <- c(x,2\*i)

print(x)

}

x<-c()

for (i in 1:100) {

z<-sample(1:100,1) #samples a random number from 1:100

x <- c(x,z)

}

x

hist(x)

**While loops**

**The logic of a while loop is this - you want something to happen until something is achieved.** Suppose you want to simulate making new mutants until you obtain one mutant. Then, you want to see how long it took until you obtained the mutation. Finally, you want to record how long it took to obtain a mutation for one experiment. By simulating many experiments, you want to get a sense as to how long it will typically take to obtain your first mutant.

while (test\_expression) {

statement

}

index <-0

num\_mutant\_found <-0

while (num\_mutant\_found == 0){

 x <- sample(1:10000, 1)

 if (x < 101) {num\_mutant\_found <- num\_mutant\_found +1}

 index <- index + 1

}

**Simulating Random Processes.**

**sample(a:b,x)**

Generates **x** random number between **a** and **b**

z<-sample(1:100,1) #samples 1 random number from 1:100

The default is to sample WITHOUT replacement.

>sample(1:10,10)

[1] 10 7 3 4 8 1 6 2 5 9

>sample(1:10,11)

Error.

> sample(1:10,11,replace=TRUE)

 [1] 9 3 1 4 4 5 5 9 5 3 3

**rbinom(x,y,z)**

Generate **x** number of random binomial numbers, with sample size **y** and probability **z** (**z** is between 0 and 1)

x<-rbinom(10,20,0.5)

x

[1] 9 12 14 13 9 12 10 6 10 11

**rmultinom(x,y,c(probabilities))**

Generates **x** samples of **y** counts in proportion to values in probability vector. Probabilities do not need to add to 1, they are normalized.

> rmultinom(1,100,c(0.25,0.25,0.25,1))

 [,1]

[1,] 15

[2,] 18

[3,] 19

[4,] 48

> rmultinom(2,100,c(0.25,0.25,0.25,10))

 [,1] [,2]

[1,] 2 3

[2,] 3 3

[3,] 1 0

[4,] 94 94

**Summarizing data**

hist(x): histogram

note: limits of X on histogram can be set with xlim. For example, hist(x,xlim = c(0,200)) sets the x values between 0 and 200

plot(x): scatter plot, where x is plotted as Y axis and x-axis is an index

mean(x): mean

var(x):variance

**Assignment**

**1. Deterministic simulations of selection**.

Build a deterministic diploid simulator that starts with a given frequency of an allele and plots the trajectory for an arbitrary number of generations. Relative fitness of the three genotypes (AA, Aa and aa) will be provided in the code. Also allow for mutation (u). To do this, use the relevantfunctions from class.

**To hand in:**

R program

Test and discuss (with plots) the following.

Does the program:

1. Show the predicted dynamics of increase for dominant (Plot 1) vs. recessive (Plot 2) beneficial mutations.

2. Show the predicted dynamics of overdominance (Plot 3) and underdominance (Plot 4).

3. Show the predicted dynamics of mutation-selection balance for a dominant deleterious allele (Plot 5) recessive deleterious allele (Plot6)

4. Is there such a thing as fixation in a deterministic model of selection?

**2. Stochastic simulation of drift and selection.**

Build a stochastic simulator using the framework above. For simplicity, you may assume haploidy.

**To hand in:**

R program

Test and discuss the following.

Does the program:

1. Show that the expected fixation probability for a neutral mutation is equal to the starting frequency? (For this, you will have to run many simulations and automate the process for counting probability of fixation or loss. I recommend you do this with modest population sizes so the runs don't take too long)

2. Show that the expected fixation probability for a beneficial mutation can be approximated as 2s ?

3. Is the Kimura probability of fixation for a beneficial allele better?