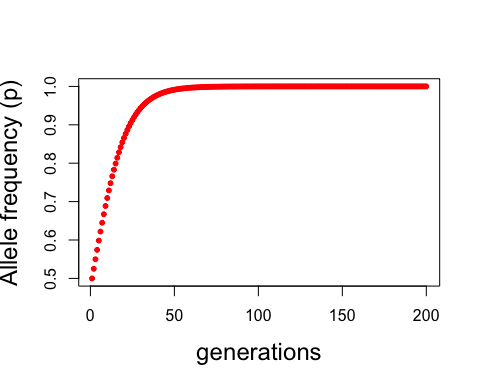
Assignment4Nov29

Zaid Sameer

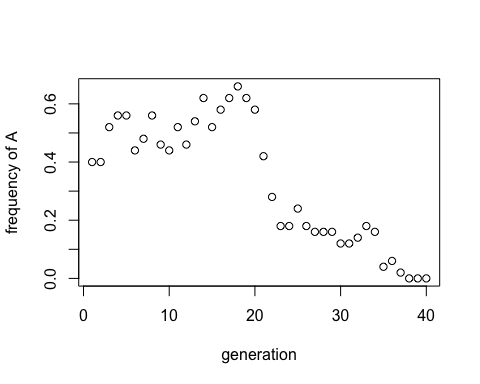
2018-11-29

#Question 2  
  
diploid\_selection <- function(p0 = 0.5, wAA = 1.1, wAa = 1.0, waa = 0.9, n = 200) {  
   
 # Initialize vectors to store allele frequencies and mean pop fitness  
 p <- rep(NA,n) # a vector to store allele frequencies  
 delta\_p <- rep(NA, n)   
 w\_bar <- rep(NA, n)  
   
 # starting conditions  
 p[1] <- p0 # starting allele frequencies  
 delta\_p[1] <- 0 #change in allele frequency  
 w\_bar[1] <- (p[1])^2 \* (wAA) + 2\*p[1]\*(1- p[1] )\*wAa + ((1- p[1])^2) \* waa  
   
 # now we need to loop from generation to generation  
 for ( i in 2:n) {  
 w\_bar[i - 1] <- (p[i - 1])^2 \* (wAA) + 2\*p[i - 1]\*(1- p[i - 1] )\*wAa + ((1- p[i - 1])^2) \* waa   
 p[i] <- (p[i - 1])^2 \* (wAA/w\_bar[i-1]) + (p[i - 1])\*(1- p[i - 1])\*wAa/w\_bar[i-1]  
 delta\_p[i] <- p[i] - p[i-1]   
 }  
 if (any(p > 0.9999)) {  
 fixation <- min(which.max(p > 0.9999))  
 cat("fixation for A1 occurs approximately at generation:", fixation )   
 } else {  
 maxAlleleFreq <- max(p)  
 cat("fixation of A1 does not occur, max. allele frequency is:", print(maxAlleleFreq, digits = 2) )  
 }  
 plot(x = 1:n, y = p,   
 xlab="generations",   
 ylab="Allele frequency (p)",   
 pch = 20, col = "red", cex.lab = 1.5)  
}  
  
diploid\_selection(p0 = 0.5, wAA = 1.1, wAa = 1.0, waa = 0.9, n = 200)

## fixation for A1 occurs approximately at generation: 97



# Question 3   
  
#fixed if pA gets to zero or one  
#trying to add diploid or haploid  
  
genetic\_drift\_function <- function(numofgen=20, pA0=0.5, sst0=20, diploid=FALSE){  
   
 pA <- rep(NA,numofgen) # a vector to store allele frequencies  
  
 pA[1] <- pA0 # starting allele frequencies  
   
 if (diploid == TRUE) {sst0 <- 2\*sst0}  
 # now we need to loop from generation to generation  
 for ( i in 2:numofgen) {  
allele\_counts <- sample(c("a", "A"),  
 size = sst0,  
 replace = TRUE,  
 prob = c((1-pA[i-1]), pA[i-1]))  
  
   
pA[i] <-sum(allele\_counts=="A")/length(allele\_counts)   
 }  
plot(pA,   
 ylab = "frequency of A",   
 xlab = "generation")  
}  
  
genetic\_drift\_function(numofgen=40, pA0=0.4, sst0=50, diploid=FALSE)



#Question 4 (100 generations, running model a 1000 times, 3 different pA0 (0.5, 0.25 and 0.1) 200 diploid individuals)  
  
genetic\_drift\_function\_TH <- function(numofgen=1000, pA0=0.5, sst0=200, diploid=FALSE){  
   
 pA <- rep(NA,numofgen) # a vector to store allele frequencies  
  
 pA[1] <- pA0 # starting allele frequencies  
   
 if (diploid == TRUE) {sst0 <- 2\*sst0}  
 # now we need to loop from generation to generation  
 for ( i in 2:numofgen) {  
allele\_counts <- sample(c("a", "A"),  
 size = sst0,  
 replace = TRUE,  
 prob = c((1-pA[i-1]), pA[i-1]))  
  
   
pA[i] <-sum(allele\_counts=="A")/length(allele\_counts)   
 }  
   
if (pA[numofgen] == 0) {print("Y")}  
 else {print("N")} #function prints Y if the allele is ever lost (otherwise it prints N)  
}  
  
  
athousand\_0.5 <- replicate(1000, genetic\_drift\_function\_TH(numofgen=100, pA0=0.5, sst0=200, diploid=TRUE))

## [1] "N"  
## [1] "N"  
## [1] "N"  
…

sum(athousand\_0.5=="Y")/length(athousand\_0.5)

## [1] 0.004

athousand\_0.25 <- replicate(1000, genetic\_drift\_function\_TH(numofgen=100, pA0=0.25, sst0=200, diploid=TRUE))

## [1] "N"  
## [1] "N"  
## [1] "N"  
## [1] "N"  
…

sum(athousand\_0.25=="Y")/length(athousand\_0.25)

## [1] 0.109

athousand\_0.1 <- replicate(1000, genetic\_drift\_function\_TH(numofgen=100, pA0=0.1, sst0=200, diploid=TRUE))

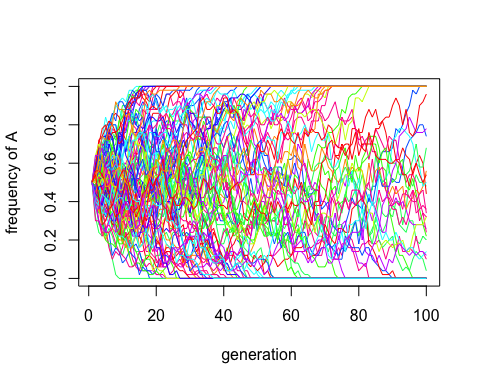
## [1] "Y"  
## [1] "N"  
## [1] "Y"  
## [1] "Y"  
…

sum(athousand\_0.1=="Y")/length(athousand\_0.1)

## [1] 0.436

#Question 5  
genetic\_drift\_function\_100\_plot <- function(numofgen=20, pA0=0.5, sst0=20, diploid=FALSE){  
   
 pA <- rep(NA,numofgen) # a vector to store allele frequencies  
  
 pA[1] <- pA0 # starting allele frequencies  
   
 if (diploid == TRUE) {sst0 <- 2\*sst0}  
 # now we need to loop from generation to generation  
 for ( i in 2:numofgen) {  
allele\_counts <- sample(c("a", "A"),  
 size = sst0,  
 replace = TRUE,  
 prob = c((1-pA[i-1]), pA[i-1]))  
  
   
pA[i] <-sum(allele\_counts=="A")/length(allele\_counts)   
 }  
 print(pA)  
}  
  
for (i in 1:100){  
 pA\_lines <- genetic\_drift\_function\_100\_plot(numofgen=100, pA0=0.5, sst0=50, diploid=FALSE)  
  
 if (i == 1) {plot(pA\_lines, type="n", ylim = c(0,1), ylab = "frequency of A",   
 xlab = "generation" )  
 lines(x=(1:100),y=pA\_lines,col=sample(rainbow(10))) }  
 else  
 {lines(x=(1:100),y=pA\_lines,col=sample(rainbow(10)))}  
  
}

## [1] 0.50 0.50 0.42 0.42 0.42 0.36 0.34 0.28 0.26 0.22 0.24 0.16 0.16 0….



## [1] 0.50 0.52 0.58 0.62 0.62 0.74 0.72 0.74 0.76 0.76 0.86 0.86 0.84 0.8…

#question 6  
question\_6\_function<- function(intercept=0.5, slope=0.1, numofobs=20, standdev=2){  
 x <- seq(from=1, to=10, length.out = numofobs)  
  
 a <- intercept  
 b <- slope  
  
y\_deterministic <- b\*x + a  
y\_simulated <- rnorm(length(x), mean=y\_deterministic, sd=standdev)  
mod\_sim <- lm(y\_simulated ~ x)  
p\_val\_slope <- summary(mod\_sim)$coeff[2,4]  
print(p\_val\_slope)  
  
}  
  
#to check  
  
set.seed(720)  
question\_6\_function()

## [1] 0.3620625

set.seed(720)

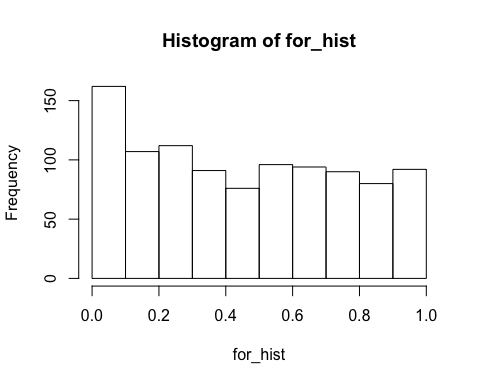
x <- seq(from=1, to=10, length.out = 20)  
  
a <- 0.5  
 b <- 0.1  
  
y\_deterministic <- b\*x + a  
y\_simulated <- rnorm(length(x), mean=y\_deterministic, sd=2)  
mod\_sim <- lm(y\_simulated ~ x)  
p\_val\_slope <- summary(mod\_sim)$coeff[2,4]  
print(p\_val\_slope)

## [1] 0.3620625

#now to generate histograms  
  
for\_hist <- replicate(1000, question\_6\_function())

## [1] 0.3250347  
## [1] 0.6773251  
## [1] 0.1576037  
## [1] 0.5441501  
…

hist(for\_hist)



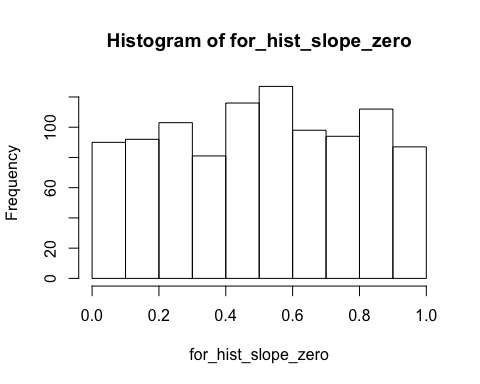
sum(for\_hist < 0.05)/length(for\_hist)

## [1] 0.107

for\_hist\_slope\_zero <- replicate(1000, question\_6\_function(slope=0))

## [1] 0.06871395  
## [1] 0.7673239  
## [1] 0.832518

…..  
hist(for\_hist\_slope\_zero)



sum(for\_hist\_slope\_zero < 0.05)/length(for\_hist\_slope\_zero)

## [1] 0.048

#less proportion is has a p < 0.05 when slope is zero