OCN 750: Homework 2

Maia Kapur

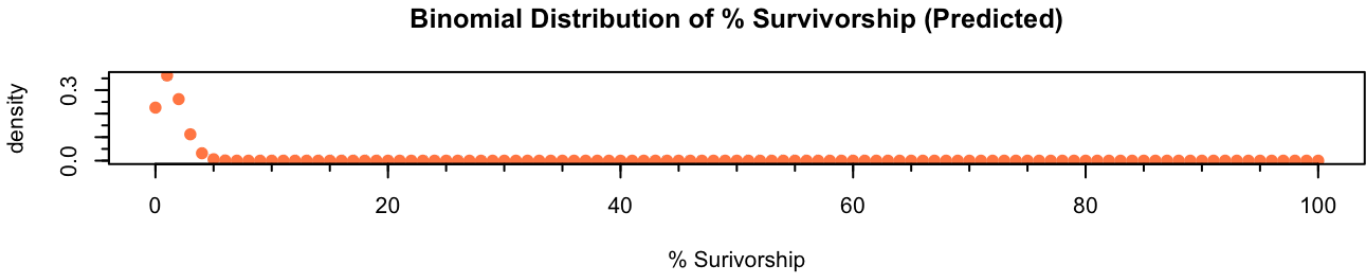
September 2, 2015

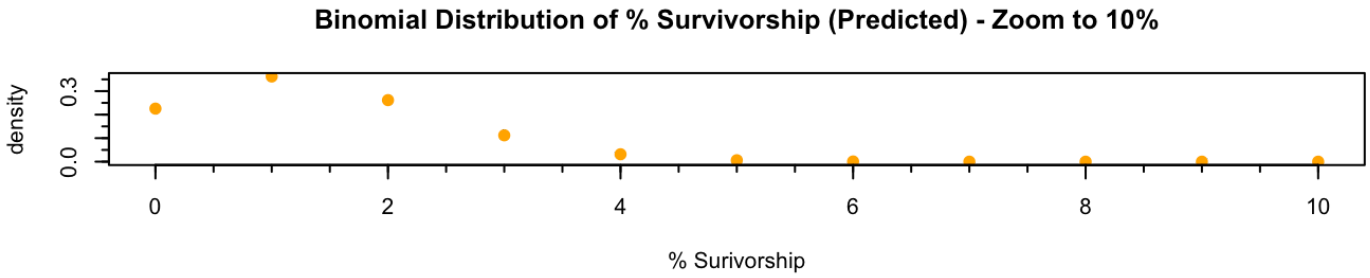
This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com (http://rmarkdown.rstudio.com).

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

Question 1: The column “n” lists the number of flies in each experiment (it’s always 10). The column “Survival” lists the number of survivors for each experiment. Make a plot with one column and two rows. On the top row, make a histogram of survivors across all experiments.On the bottom row, make a plot of the expected probabilities for a binomial distribution with n = 10 and with p equal to the mean proportion of survivors across all experiments (you will need to calculate this mean). You can calculate the expected binomial probabilities using dbinom(). The argument “size” is how you specify the number of trials (n), and the argument “prob” is how you specify the probability of survival (p). The argument “x” is the number for which you want to find out the corresponding probability, i.e. if you set x = 1, you are asking what is the probability that there will be 1 survivor.

```
#load packages
#install.packages("Hmisc")
library(Hmisc)


#Set working directory
#setwd("~/Users/maiakapur/Dropbox/2015 Fall/OCN 750")
#Import CSV
flies = read.csv("heat_shock_survival.csv")
#rename a column
colnames(flies)[8] <- "surv"
colnames(flies)[9] <- "perc_surv"

#create plotting device with one column and three rows
par(mfrow = c(3,1))

#create histogram - this plots the frequency of absolute survivors
hist(flies$surv, xlab = "Survivors", main = "Histogram of Observed Survivorship, a
ll Experiments", col = "coral")

#Plot Expected probabilities with binomial distribution, n = 10, p = mean survivor
s

#calculate mean % survivorship across all experiments
p = mean(flies$perc_surv)/100

#dbinom function: (vector, no trials, prob vector, log t/f)
#create vector of x values (these are your "test" cases -- must be integers)

#In this case I did from 0 to 100, with 100 being 100% survivorship
x = seq(0,100, by = 1)

#write equation for binomial distribution and plot
y = dbinom(x, size = 10, prob = p)

plot(x,y, ylab = "density", col = "coral", main = "Binomial Distribution of % Surv
ivorship (Predicted)", bg = "coral", pch = 21, type = "p", xlab = "% Surivorship")
minor.tick(nx = 4)

#it looks pretty flat after about 10 percent. Let's make another one zoomed in and
place it on top of the original plot.
k = seq(0,10, by = 1)
#write equation for binomial distribution and plot
y = dbinom(k, size = 10, prob = p)
plot(k,y, ylab = "density", col = "orange", bg = "orange", pch = 21, main = "Binom
ial Distribution of % Survivorship (Predicted) - Zoom to 10%", xlab = "% Surivorsh
ip")
minor.tick(nx = 4)
```





How would you characterize the similarities and differences between the observed distribution and the predicted distribution from the binomial function?

What is the variance in #survivors for the observed data?

```
#A function that gives variance of binomial distribution when given n and p
vdbinom = function(n,p){n * p *(1-p)}
vdbinom(10,p)
```

```
## [1] 1.191897
```

Give a reason why the variance in the observed data might be larger than the predicted variance from the binomial distribution.

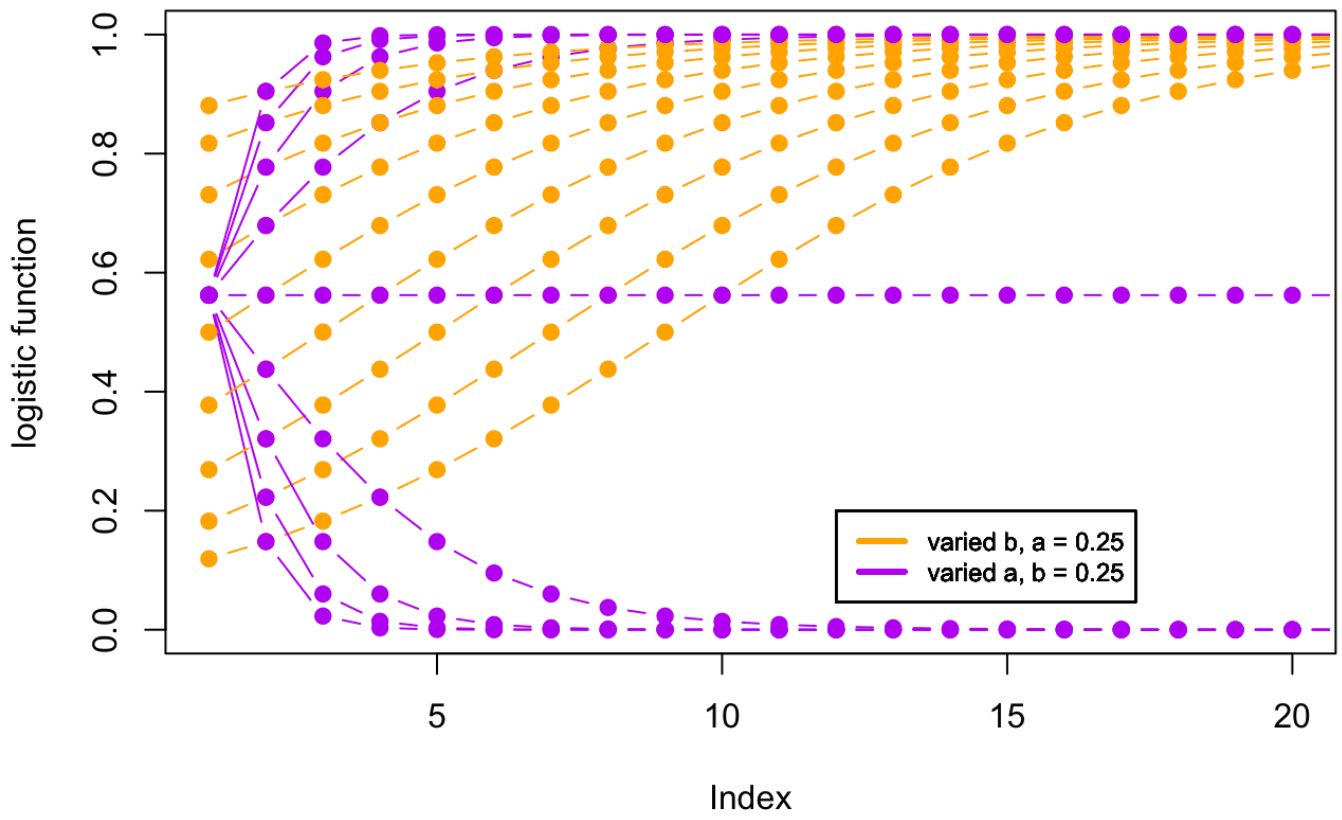
```
#Comparison of variances
vars = c(var(flies$surv), vdbinom(10,p))
names(vars) = c("observed", "predicted")
vars
```

```
##  observed predicted
##  3.065598  1.191897
```

```
#The variance is smaller in the predicted model because of differences in the way
variance is calculated between the two distributions; while the binomial distribut
ion is a function of the probabilities and no. of trials, the observed variance is
scaled by the "noisy" differences from the mean.
```

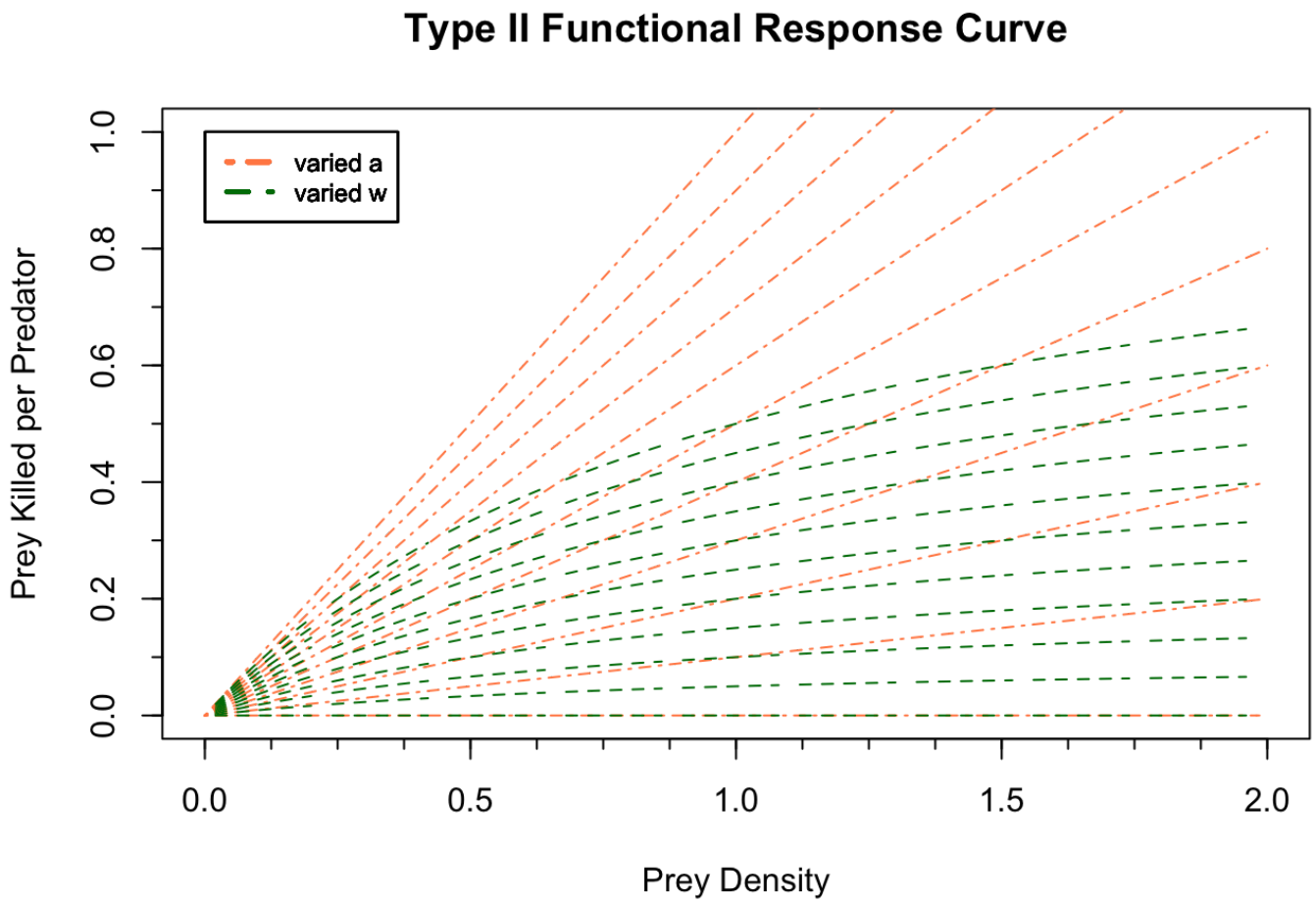
2 Use curve() to show how the logistic curve changes as each parameter (a and b) varies from positive to negative. Make two plots, one where you vary a over a number of values, and one where you vary b over a number of values. Use the option add=TRUE to plot the different curves on the same plot, with different colors. Use legend() to add a legend for the different colors.

```
#Overwrite the graphic device from Q1
par(mfrow = c(1,1))

#Write the logistic function (there may already be a package)
logistic = function(a,b,x){
  exp(a + b*x)/(1 + exp(a + b*x))}

#Set up a vector of values to plot along the curve
#tries = seq(0,10,1)
a = seq(-2,2,0.5)
b = seq(-2,2,0.5)
#x = seq(0,10,0.5)

#a vector of colors (for plotting purposes)
linecols = c("orange", "purple")

#logistic(1,0.1,x) # a from 0 to 1 by 100

#plot(logistic(a[1],5,w), col = "red")
#points(logistic(a[2],5,w), col = "green")
#points(logistic(a[3],1,w), col = "black")
#points(logistic(1,b[1],w), col = "black")

#a for-loop that plots various values of A against b = 0.25
for (i in 1:length(a)) {
  if (i == 1){
    #this plots the first and second values of a with b = 0.25
    plot(logistic(a[i],0.25,x), xlim = c(1,20), ylim = c(0, 1.0), type = "b", ylab
= "logistic function", main = "Logistic Curves for Various Values of a and b", col
= linecols[1], pch = 21, bg = linecols[1], cex = 1)
    points(logistic(a[i+1],0.25,x), col = linecols[1], type = "b", pch = 21, bg =
linecols[1])
    #same thing for various values of b, with a fixed at 0.25
    points(logistic(0.25,b[i],x), type = "b", pch = 21, bg = linecols[2], col = li
necols[2])
    points(logistic(0.25,b[i+1],x), col = linecols[2], type = "b", pch = 21, bg =
linecols[2])
    }
  else {
    #plots points for indices 2 and beyond (e.g. a[3])
    points(logistic(a[i+1],0.25,x), col = linecols[1], pch = 21, bg = linecols[1],
type = "b")
     points(logistic(0.25,b[i+1],x), col = linecols[2], pch = 21, bg = linecol
s[2], type = "b")
    }
legend(12,0.2, c("varied b, a = 0.25", "varied a, b = 0.25"), cex = 0.75, lt
y=c(1,1), lwd=c(2.5,2.5),col=c(linecols[1],linecols[2]))
}
```



3. In lecture we talked about the Type II functional response and gave an example for wolves feeding on caribou. Use curve() to make some plots showing how the shape of this curve varies as you vary the handling time (h) and the attack rate (a). How do these parameters differ in how they control the shape of the curve, particularly (1) when prey density (R) is very low and (2) when prey density is very high? If you were a hungry wolf who was hunting for low-density caribou, would you rather increase your attack rate or decrease your handling time? Make sure you plot this with a large enough range on the x-axis to see how the curve saturates.

```
## The FRII equation can be rearranged and written in terms of an asymptote & hal
f-saturation constant. The fact that the Type II functional response and the Micha
elis-Menten equation have the same mathematical form is actually pretty intuitive.

#Set up a sequence of "test" values your parameters
#Modified from:  https://github.com/mattbarbour34/comprehensive_exams/blob/maste
r/functional_response_code.Rmd

a = seq(0,1,0.1) #attack rate of predator
w = seq(0,1,0.1) # maximum attack rate of predator in Type 2 or 3 functional respo
nse
D <- w/a # half saturation constant.

linecols = c("coral", "darkgreen")
for (i in 1:length(a)) {
  if (i == 1){
curve(a[i] * x, 0, 2, xlab = "Prey Density", ylab = "Prey Killed per Predator", yl
im = c(0,1), lty = 4, col = linecols[1], main = "Type II Functional Response Curv
e") # Type 2
curve(a[i+1] * x, 0, 2, add = TRUE, lty = 4, col = linecols[1])
curve(w[i] * x / (D + x), 0, 2, add = TRUE, lty = 2, col = linecols[2])
curve(w[i+1] * x / (D + x), 0, 2, add = TRUE, lty = 2, col = linecols[2])
}
  else {
  curve(a[i+1] * x, 0, 2, add = TRUE, lty = 4, col = linecols[1])
  curve(w[i+1] * x / (D + x), 0, 2, add = TRUE, lty = 2, col = linecols[2])
}
legend(0,1, c("varied a", "varied w"), cex = 0.75, lty=c(4,2), lwd=c(2.5,2.5),co
l=c(linecols[1],linecols[2]))
}
minor.tick(nx = 4)
```




```
#It appears that at all except very small prey densities, an increased attack rate
(a) would have a much greater benefit than decreasing your handling time. At low p
rey densities, e.g. <0.25, they seem comprable.
```

Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.