~~~
---
title: "STAT 261 Lab 2"
author: "Fill in your name"
date: "Fill in the date"
output:
  pdf_document: default
  html_document:
    df_print: paged
---

# General Instructions

* Execute each chunk of code to ensure that your code works properly.
* Save this .Rmd file and then knit the entire document to pdf.

# Learning outcomes

* use of `hist()` to plot histograms of generated/observed data (and appearance
customization!)
* Adding density lines to histograms to model the relationship between the Chi-square and
standard normal distributions
* use of geometric distribution functions, `rgeom()` and `dgeom()`
* understanding how to code (log-)likelihood and (log-)relative likelihood functions in R
* use of `optimize()` to find the MLE of your (log-)likelihood function
* use of `uniroot()` to find 100p% likelihood intervals
* use of `round()` to present rounded numerical values

## 1.  The Normal(0,1) and Chi-squared distribution.

In this section, we see the relationship between Normal(0,1) random variables
and the Chi-squared(1) distribution.  Normal(0,1)^2 ~ Chi-squared(1)

```{r chi}
# First generate 1000 random observations from N(0,1)
set.seed(12345)
myz <- rnorm(1000)

# Plot a scaled histogram and overlay normal pdf
h <- hist(myz, breaks=10, col="red",
        main="Histogram with Normal Curve", prob=TRUE)
xfit<-seq(min(myz), max(myz), length=40) #sequence of length 40
yfit<-dnorm(xfit, mean=mean(myz), sd=sd(myz)) #normal density
lines(xfit, yfit, col="forestgreen", lwd=50) #overlay fitted normal density

# as a little exercise, change the color in line 39 above to "forestgreen" and
# the linewidth to 50. Congratulations, you have just made your
# first piece of Data Art!! A very hungry caterpillar :^)

# Square these observations, plot a histogram and overlay the chi-square(1) pdf
myz2 <- myz^2
h <- hist(myz2, breaks=10, col="red",
        main="Histogram with Chi-squared(1) Curve", prob=TRUE)
xfit <- seq(min(myz2), max(myz2), length=40)
yfit <- dchisq(xfit, 1)    #chi-squared(1) density
lines(xfit, yfit, col="blue", lwd=2) #overlay Chi-squared(1) curve


```



## 2.  Generate n=10 observations from the Geometric distribution.

_type ?rgeom() to read which form of the PMF R computes (i.e. does it include or exclude
the final successful trial?)_
~~~

~~~
```{r Sample}
set.seed(54321)
n <- 10
geo.dat <- rgeom(n, prob=.07)   # this generates n geometric observations

geo.mle <- 1/(1+mean(geo.dat)) #We derived this MLE during the basketball tryouts example
geo.mle
```

## 3. Write a function to compute the log-likelihood given the n observations and plot the
log-likelihood.

In this section, we write a function which computes the Geometric log-likelihood given
arguments:
* theta, a scalar or vector of probabilities and
* x, a vector of observations.


```{r log-like}
# Write a function to compute geometric log-likelihood with arguments
#  theta = scalar or vector of geometric probabilities
#  x = vector of observations

ell <-  function(theta, x){
  loglike <- 0
  for (i in 1:length(x)){
    loglike <- loglike + dgeom(x[i], prob=theta, log=TRUE)
      #computes the sum of the log of geometric probabilities over each of the
observations
  }
  return(loglike)
}

theta <- seq(0.01, 0.15, by=.005)  #a vector sequence of theta values
gloglike <- ell(theta, geo.dat)    #compute the log-likelihood for each value in theta,
given data geo.dat
head(cbind(theta, gloglike))    #head prints the top values
plot(gloglike ~ theta, ylab='Log-Likelihood', xlab='theta', type='l') #plot(y ~ x) version
title(paste('Geometric Log-likelihood for Lab 2, n=', n, sep=''))
```

## 4. Compute the MLE using the function we defined.

```{r MLE}
#?optimize   #see the arguments and outputs for optimize; see optional arguments ...
# we pass a function to optimize, starting interval, and any other arguments required by
ell

thetahat <- optimize(ell, c(.02, .10), maximum=TRUE, x=geo.dat)
thetahat   #how does the maximum compare with 1/(1+mean(geo.dat)) computed above?
thetahat$maximum  #extract the maximum
thetahat$objective  #extract value of function ell at maximum
```


## 5.  Write a function to compute the log relative likelihood, r(theta) and graph.

```{r log_relative}
# Function to compute the log relative likelihood, r(theta)
#  theta = scalar or vector of Binomial probabilities
#  thetahat = the MLE of theta
#  x = vector of observations

logR <- function(theta, thetahat, x){
  ell(theta, x) - ell(thetahat, x)
~~~

~~~
}

logR(theta, thetahat$maximum, geo.dat)

# Function to compute the log relative likelihood - ln(p) for
#      100p% Likelihood interval computations
#  theta = scalar or vector of Binomial probabilities
#  thetahat = the MLE of theta
#  x = vector of observations
logR.m.lnp <- function(theta, thetahat, x, p) {logR(theta, thetahat, x) - log(p)}

p <- .1  #10% likelihood interval
plot(logR.m.lnp(theta, thetahat$maximum, geo.dat, p) ~ theta, ylab='r(theta)-ln(p)',
     xlab='theta', type='b')
abline(h=0)  #add horizontal line at zero
title('Lab 2, Log Relative Likelihood - ln(p)')
```

## 6. Compute the 10% Likelihood Interval as the roots of r(theta) - ln(.10) = 0

```{r roots}
#?uniroot  #see the arguments for uniroot
# use the graph to obtain starting interval for root finding search

#Likelihood intervals, supply the function, starting interval and arguments
lower <- uniroot(logR.m.lnp, c(.01, .04), thetahat$maximum, geo.dat, p)
lower

upper <- uniroot(logR.m.lnp, c(.06, .10), thetahat$maximum, geo.dat, p)
upper

```

# Summary:

The maximum likelihood estimate of theta is, `r thetahat$maximum` and its 10% likelihood
interval is (`r lower$root`, `r upper$root`).

(Rounded version)
The maximum likelihood estimate of theta is, `r round(thetahat$maximum, 3)` and its 10%
likelihood
interval is (`r round(lower$root, 3)`, `r round(upper$root, 3)`).
~~~