MATH 420 Homework #3 SOLUTIONS

Instructions: A printed copy of your homework should be handed in at **the start of class** the day it is due. If you have any supplementary electronic files you wish to turn in (e.g. R scripts or wxMaxima files) email them to the instructor prior to class with file name format: `Lastname-hwX.ext`. Each part of each exercise is worth 10 points unless stated otherwise.

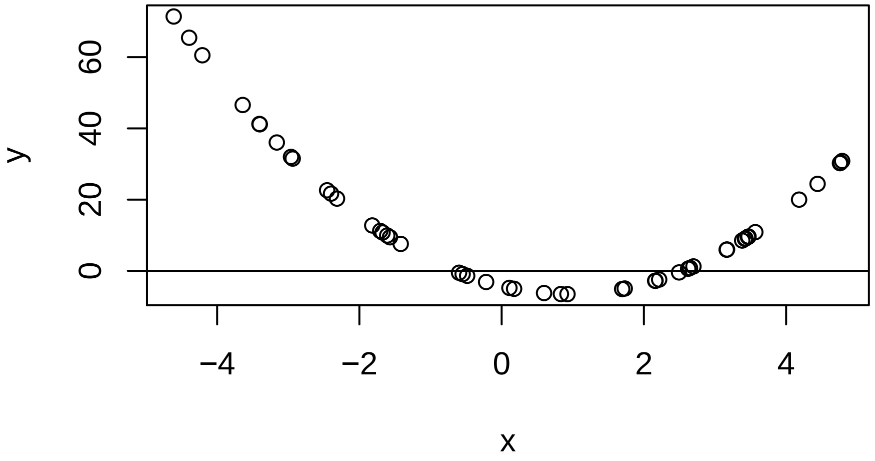
Exercise 1: Suppose the results of a Poisson Regression analysis done on transect count data (similar to the one done in class) reveals a best fit model of $y_i \sim \text{Poisson}(\lambda = 0.75\,\text{Length}_i)$, where the data do not appear to be overdispersed. Based on the underlying model, what would you estimate to be the average distance between organisms along the transects? Justify your answer.

Ans: Recall that for a homogeneous Poisson Process with rate r the intervals between events have lengths that are Exponentially distributed with rate r and the number of events in an interval of length L is Poisson with mean $\lambda = r\,L$. Therefore, if our best fit model is Poisson with mean $\frac{3}{4}\,L$, the distances between organisms along the transects should be Exponentially distributed with rate $r = \frac{3}{4}$, and should have an average distance of $\frac{1}{r} = \frac{4}{3}$.

Exercise 2: See the file rssmle.R on the course website.

(a) First, generate a fake data set of 50 points along a parabola (i.e. a quadratic polynomial) with two real roots. Whether or not you add noise is up to you.

```r
set.seed(8675309)
x=runif(50, -5,5)
pars <- runif(3, -5,5) # mystery parameters!
y=pars[1]*(x-pars[2])*(x-pars[3])
plot(x,y); abline(h=0)
```



(b) Estimate the parameters r_i in $y = (x - r_1)(x - r_2)$ or $-(x - r_1)(x - r_2)$ based on graphically inspecting the data (e.g. plot it and carefully read off where the roots occur).

Ans: In the figure above, around -0.5 and 2.5.

(c) Fit the function $y = a + b\,x + c\,x^2$ to your simulated data using two different methods in `optimx()` (e.g., use `method=c("Nelder-Mead","BFGS")`). Use the quadratic formula to calculate the roots r_i, and compare to part (b).
Ans: Here we'll use minimization of the RSS$= \sum_i \left(y_i - (a + b\,x_i + c\,x^2)\right)^2$.

```
library(optimx)
RSS <- function(ps) {
  a=ps[1]; b=ps[2]; c=ps[3];
  return(sum((y-(a+b*x+c*x^2))^2))
}
fit=optimx(c(a=1,b=0,c=1),RSS,method=c("Nelder-Mead","BFGS","nlm"))
fit

##                     a        b       c        value fevals gevals niter
## Nelder-Mead -4.28141 -4.76560 2.52773 3.11290e-06    228     NA    NA
## BFGS        -4.28134 -4.76562 2.52771 8.41885e-19     27      8    NA
## nlm         -4.28132 -4.76562 2.52770 1.83487e-08     NA     NA    19
##             convcode  kkt1 kkt2 xtimes
## Nelder-Mead        0 FALSE TRUE      0
## BFGS               0  TRUE TRUE      0
## nlm                0 FALSE TRUE      0

## BFGS gave the best fit.
c(a=fit$a[2], b=fit$b[2], c=fit$c[2])

##        a        b        c
## -4.28134 -4.76562  2.52771

## TRUE vs ESTIMATED ROOTS

r1 = (-fit$b[2]-sqrt(fit$b[2]^2-4*fit$a[2]*fit$c[2]))/(2*fit$c[2])
r2 = (-fit$b[2]+sqrt(fit$b[2]^2-4*fit$a[2]*fit$c[2]))/(2*fit$c[2])

c(pars[2], r1) # TRUE vs ESTIMATED root 1

## [1] -0.664309 -0.664309

c(pars[3], r2) # TRUE vs ESTIMATED root 2

## [1] 2.54966 2.54966
```

Thus, our procedure for estimating the two roots was a success.

Exercise 3: Fit the rate parameter of an exponential distribution from a simulated data set (use `set.seed(123); x=rexp(200,rate=pi);`) using an MLE based optimization using `optimize()` and the `dexp()` function in R (i.e., don't use the analytical formula for the MLE; see Exercise 4).

Ans:

```
set.seed(123); x=rexp(200,rate=pi);
nLL = function(r) { -sum(dexp(x,rate=r,log=TRUE)) }
fit=optimize(nLL,c(1e-15, 10)); fit


## $minimum
## [1] 3.11903
##
## $objective
## [1] -27.5045


# An acceptable alternative with some helpful extras...
fit=optimx(1, nLL, method="L-BFGS-B", lower=1e-15,upper=10); fit


##               p1    value fevals gevals niter convcode kkt1 kkt2 xtimes
## L-BFGS-B 3.11903 -27.5045      9      9    NA        0 TRUE TRUE      0
```

Exercise 4: Next, repeat the above parameter estimation but here find an analytical expression for the MLE $\widehat{r}$ by differentiating the negative log likelihood function for the scenario above (i.e., for an exponential distribution with rate r) for a given data set $(x_1, \ldots, x_N)$. Recall

$$f(x) = r\ \exp(-r\,x), \quad \text{for} \quad x \geq 0.$$

Ans: Differentiating (with respect to r) the negative log likelihood function

$$\begin{aligned} nLL(r|\mathbf{x}) &= -\ln\left(\prod_{i=1}^{N} r\ \exp(-r\,x_i)\right) \\ &= -\ln\left(r^N \exp\left(-r\sum_{i=1}^{N} x_i\right)\right) = r\,N\,\bar{x} - N\ln(r) \end{aligned}$$

yields

$$\frac{\partial}{\partial r} nLL(r|x) = \left(\bar{x} - \frac{1}{r}\right) N$$

which, when set to zero to find the value of r that minimizes nLL, gives the MLE

$$\widehat{r} = \frac{1}{\bar{x}}.$$

Exercise 5:
(420 Students Only) Use the `boot` package in R to calculate 95% confidence intervals for the estimate of r used in Exercise 4 above (for example, see `https://www.statmethods.net/advstats/bootstrapping.html`).

Ans: Building on the code above...

```
library(boot)
r.boot <- function(xs,indx) { 1/mean(xs[indx]) }
bs <- boot(data=x, statistic=r.boot, R=50000)
## Which type? See http://www.tau.ac.il/~saharon/Boot/10.1.1.133.8405.pdf
boot.ci(bs,conf=0.95, type=c("basic","bca","perc"))


## BOOTSTRAP CONFIDENCE INTERVAL CALCULATIONS
## Based on 50000 bootstrap replicates
##
## CALL :
## boot.ci(boot.out = bs, conf = 0.95, type = c("basic", "bca",
##     "perc"))
##
## Intervals :
## Level      Basic              Percentile            BCa
## 95%   ( 2.661,  3.498 )   ( 2.740,  3.577 )   ( 2.712,  3.537 )
## Calculations and Intervals on Original Scale


## basic is less reliable than bca and percentile methods
```

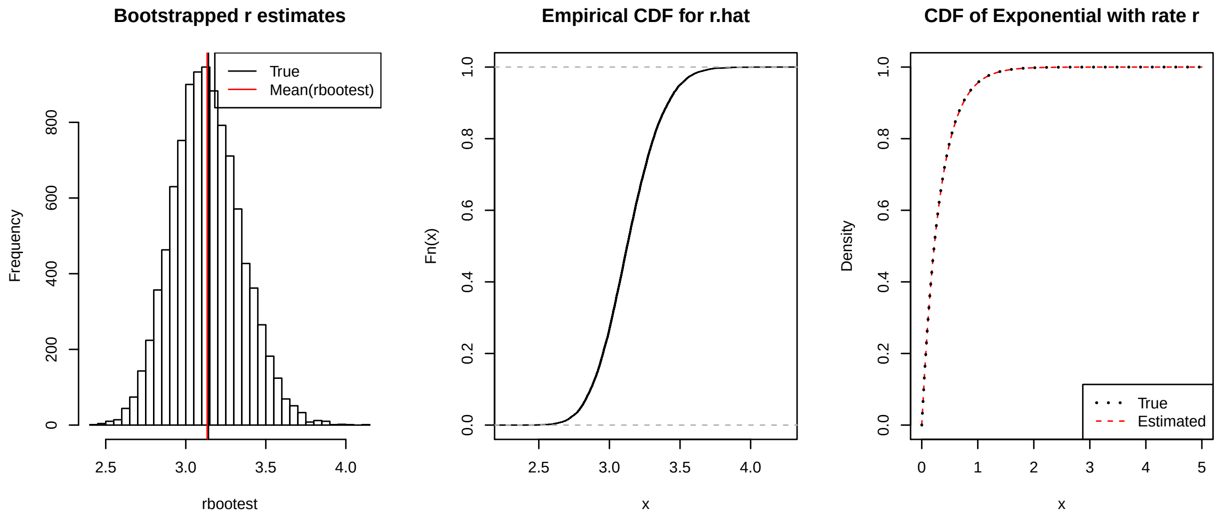
(620 Students Only) The **non-parametric bootstrap** is a resampling-based method of constructing a distribution for an estimator that can then be used to create confidence intervals. In short, iteratively sample from your data with replacement a new data set with the same sample size and use it to recalculate your estimate. Repeat this 10,000 or more times and each time store the estimate, then use that set of 10,000+ estimates as a sample from the estimator distribution to construct the confidence interval. Implement this for the estimation procedure in Exercise 3 above, plot a histogram of the bootstrapped estimates, an empirical CDF (using `plot(ecdf(rbootest))` if the vector of bootstrap estimates is stored in `rbootest`), a CDF plot using the estimate $\widehat{r}$ (use `curve(pexp(x,r.est),0,2.8,col="red",type="l",lty=2` or overlay it on the empirical CDF by adding the argument `add=TRUE`).

a. Does this estimate seemed biased?

b. What is the 95% Confidence Interval on r? (Hint: if the bootstrap estimates are in a vector *rbootest*, the 2.5% and 97.5% percentile of r can be obtained using the R code `quantile(rbootest, c(0.025, 0.975))`).

Ans:

```
r.hat <- function(xs) { 1/mean(xs) }
r.est <- r.hat(x) # our actual r estimate for sample x
rbootest = c() # we'll fill this up with bootstrapped estimates
for(i in 1:10000) {
    rbootest[i] = r.hat(sample(x,length(x),replace=TRUE))
}
par(mfrow=c(1,3))
hist(rbootest,main="Bootstrapped r estimates",60)
abline(v=pi,col="black")
abline(v=mean(rbootest), col="red")
legend("topright",c("True","Mean(rbootest)"),lty=c(1,1),col=c("black","red"))
plot(ecdf(rbootest),main="Empirical CDF for r.hat")
curve(pexp(x,rate=r.est),0,5,col="red",type="l",lty=2,
    main="CDF of Exponential with rate r", ylab="Density")
curve(pexp(x,rate=pi),0,5,col="black",type="l",lty=3, lwd=2, add=TRUE)
legend("bottomright",c("True","Estimated"),lty=c(3,2),col=c("black","red"), lwd=c(2,1))
```



From these results it seems that (a) the estimates aren't significantly biased, and (b) using a crude approach to finding the confidence interval, we have

```
quantile(rbootest, c(0.025, 0.975))


##    2.5%   97.5%
## 2.73714 3.57140
```

Exercise 6 (Parametric Bootstrap): Consider this small study of human height versus shoe size at (https://www.statcrunch.com/5.0/viewreport.php?reportid=35115), summarized in Figure 1 below, which found the following best fit SLR regression model correlating U.S. shoe size and height in inches: $Height = 50.87 + 1.657\,ShoeSize$ with $\widehat{\sigma} = 3.78$ which yielded an R^2 value of 0.387. This analysis is repeated in the R script **hw3-shoesize-height.R** using the data in **hw3-sullivan-statistical-survey-data.xlsx**.

```
## Shoe size data from https://www.statcrunch.com/5.0/viewreport.php?reportid=35115
## Data from https://www.statcrunch.com/app/index.php?dataid=450205

library(readxl)
ssdat <- read_excel("hw3-sullivan-statistical-survey-data.xlsx",1)
# Redo their analysis:
fit0 <- lm(Height~Foot, data=ssdat)
summary(fit0)
```

```
##
## Call:
## lm(formula = Height ~ Foot, data = ssdat)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -18.75  -2.20   0.56   2.53  12.56
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   50.875      1.531    33.2   <2e-16 ***
## Foot           1.657      0.148    11.2   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.78 on 197 degrees of freedom
## Multiple R-squared:  0.387,Adjusted R-squared:  0.384
## F-statistic:  124 on 1 and 197 DF,  p-value: <2e-16
```

```
# We can access elements of the object returned
# by a function by treating the function call
# as if it were an object, e.g. a data frame:
Rsq0 <- summary(fit0)$adj.r.squared
Rsq0
```

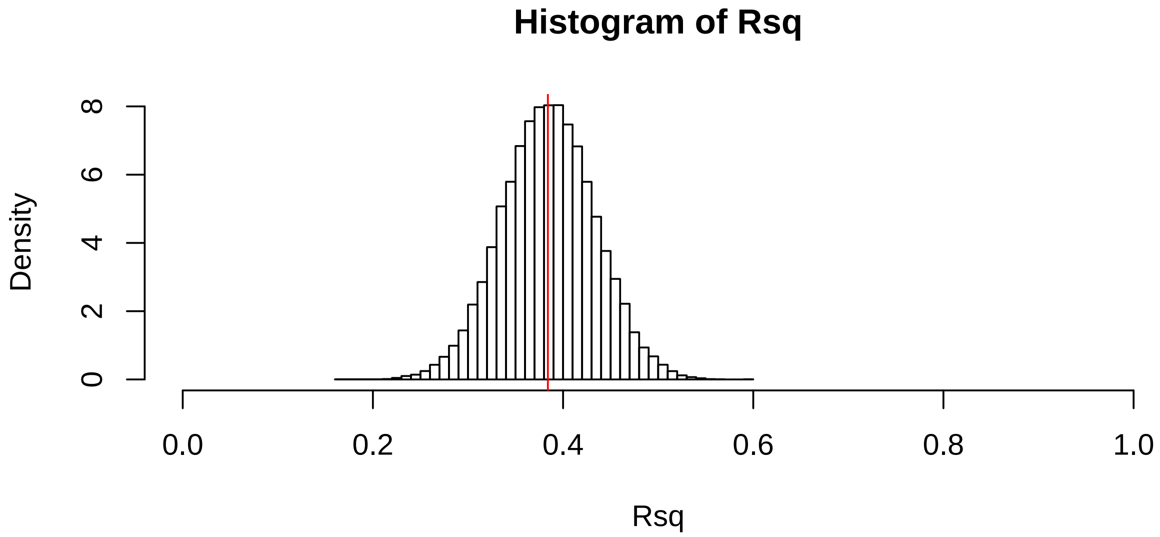
```
## [1] 0.38402
```

```
# From the regression on the website listed above (compare to above)
B0 = 50.874798;
B1 = 1.6565183;
sd = 3.7840705;
N = nrow(ssdat); N # sample size
```

```
## [1] 199
## Create vectors ShoeSize and ExpectedHeight.
ShoeSize = ssdat$Foot
ExpectedHeight = B0 + B1 * ShoeSize

## Parametric Bootstrap...
K=50000
Rsq = rep(NA,K) # place holder for Rsq values
for(i in 1:K) {
  # Here I'm packing the process of sampling from the best fit model
  # and re-running the regressio to get Rsq all into a single line of code ...
  Rsq[i] =  summary(lm(Height~ShoeSize, data=data.frame(ShoeSize = ssdat$Foot,
                Height = rnorm(N,mean=ExpectedHeight,sd=sd))))$adj.r.squared
}

hist(Rsq,60,xlim=c(0,1),freq=FALSE)
abline(v=Rsq0,col="red")
```



a. What is the basic definition of R^2 in this context, and how should we interpret R^2 values?

Ans: Apart from the technical definition, R^2 is a measure of how much of the variance is explained by our model. Thus large values are interpreted as indicating that the model does a good job ‘‘explaining the data”.

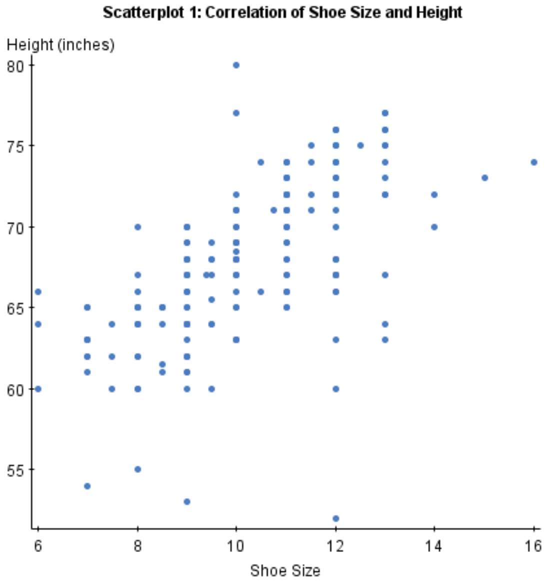
b. How expected or unexpected is this level of explanatory power (i.e., the $R^2 = 0.387$ value) under the assumption that this best fit model is true, and given these sample sizes? To answer that question, use the following number of individuals with each shoe size in the data set to simulate synthetic data under their best fit linear model given above. Repeat this data simulation 50,000 times using a for loop (*for(i in 1:50000) { ...)* and each time store the R^2 value using something like `fit=lm(...); fitsum <- summary(fit); Rsq[i]<-fitsum$adj.r.squared;`. Plot a histogram of these R^2 values, use *abline()* to draw a vertical line indicating the

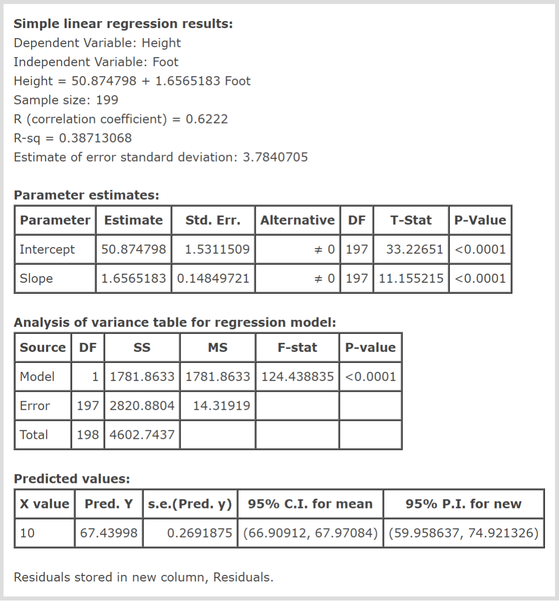
$R^2 = 0.387$, and see Exercise 5 above, and calculate a corresponding p-value.

Ans: Under the (strong!) assumption that the best-fit model is the “true” model, we see a range of Rsq values from around 0.2 to 0.6 for similarly distributed data sets. Thus, we would not expect to see very high R^2 values for such data given this amount of “noise” and this sample size.

c. Discuss what your results mean in terms of how well this particular data set meets the assumptions of the SLR model.

Ans: It does not seem to be obviously inconsistent with the assumed modeling framework, however additional checking (e.g., the usual statistical model diagnostic checks) is needed to fully answer this question.





Simple linear regression results:
Dependent Variable: Height
Independent Variable: Foot
Height = 50.874798 + 1.6565183 Foot
Sample size: 199
R (correlation coefficient) = 0.6222
R-sq = 0.38713068
Estimate of error standard deviation: 3.7840705

Parameter estimates:

Parameter	Estimate	Std. Err.	Alternative	DF	T-Stat	P-Value
Intercept	50.874798	1.5311509	≠ 0	197	33.22651	<0.0001
Slope	1.6565183	0.14849721	≠ 0	197	11.155215	<0.0001

Analysis of variance table for regression model:

Source	DF	SS	MS	F-stat	P-value
Model	1	1781.8633	1781.8633	124.438835	<0.0001
Error	197	2820.8804	14.31919		
Total	198	4602.7437			

Predicted values:

X value	Pred. Y	s.e.(Pred. y)	95% C.I. for mean	95% P.I. for new
10	67.43998	0.2691875	(66.90912, 67.97084)	(59.958637, 74.921326)

Residuals stored in new column, Residuals.

Figure 1: Source: https://www.statcrunch.com/5.0/viewreport.php?reportid=35115 based on data from the Sullivan Statistics Survey at https://www.statcrunch.com/app/index.php?dataid=450205.