Problem Set 3

Adam Ten Hoeve

Introduction

Please complete the following tasks regarding the data in R. Please generate a solution document in R markdown and upload the .Rmd document and a rendered .doc, .docx, or .pdf document. Your work should be based on the data's being in the same folder as the .Rmd file. Please turn in your work on Canvas. Your solution document should have your answers to the questions and should display the requested plots.

These questions were rendered in R markdown through RStudio (https://www.rstudio.com/wp-content/uploads/2015/02/rmarkdown-cheatsheet.pdf, http://rmarkdown.rstudio.com).

Question 1

(10 points)

Create 10,000 samples of size 10 from the standard Normal distribution and calculate the maximum likelihood values of μ and σ^2 for each. Compute the mean of the μs and the mean of the σ^2s. Does the maximum likelihood estimate of μ seem have produce values with the mean equal the true value in the long run? (You may repeat the experiment to help answer this question.) Does the maximum likelihood estimate of σ^2 seem have produce values with the mean equal the true value in the long run? (Try comparing with the adjusted estimates produced by dividing the sum of the squared differences by 9 instead of 10.)

```
set.seed(45678)
mat<-matrix(rnorm(10000*10),ncol=10)

k <- 10000
n <- 10

means <- numeric(k)
vars <- numeric(k)
vars2 <- numeric(k)

# Simulate k times, where we pull n times from a std normal
for (i in 1:k){
    samp <- rnorm(n)
    # Add the sample to the matrix
    mat[i, ] <- samp

    # Calculate the mus and sigma^2 from the theoretical values
    # mu_hat = x_bar
    means[i] <- mean(samp)
    # \hat{sigma^2} <- 1/n * sum(x_i - x_bar)^2
    vars[i] <- 1/n * sum((samp - mean(samp))^2)
    # \hat{sigma_2^2} <- 1/(n-1) * sum(x_i - x_bar)^2
    vars2[i] <- 1/(n-1) * sum((samp - mean(samp))^2)
```

```
}

# Compute the mean of the mu estimates
cat("The mean of the estimated mu's is ", mean(means), ".\n", sep="")
```

```
## The mean of the estimated mu's is 0.001471441.
```

```
# Compute the means of the sigma^2 estimates
cat("The mean of the estimated sigma^2s is ", mean(vars), ".\n", sep="")
```

```
## The mean of the estimated sigma^2s is 0.9027457.
```

```
# Compute the means of the alternative sigma^2 estimates.
cat("The mean of the adjusted estimed sigma^2s is ", mean(vars2), ".\n", sep="")
```

```
## The mean of the adjusted estimed sigma^2s is 1.003051.
```

We can see that the maximum likelihood estimate μ value does approach the true mean of 0 as the number of simulations increases.

We can also see that the maximum likelihood estimate of σ^2 is approaching 0.9, which is not the true variance. However, the adjusted estimate, which was when we calculated each $\hat{\sigma}_i^2$ by dividing by $(n-1)$ instead of n, does approach 1.0 as the number of simulations increases.

Question 2

(10 points)

The exponential distributions are a one parameter family of continuous distributions, $Exp(\lambda)$. Given λ, the sample space is $[0, \infty)$ and the probability density function is $f(x) = \lambda \exp(-\lambda x)$. Thus if $x_1 ... x_n$ are n independent draws from an exponential distribution with parameter λ, the likelihood function of this sample is $\prod_{i=1}^{n} \lambda \exp(-\lambda x_i)$. Please derive the maximum likelihood value of λ as a function of $x_1 ... x_n$. That is, given $x_1 ... x_n$, what value of λ maximizes $\prod_{i=1}^{n} \lambda \exp(-\lambda x_i)$?

We are given the likelihood $\mathcal{L} = \prod_{i=1}^{n} \lambda e^{(-\lambda x_i)}$. Let's convert this to the log-likelihood so the eventual derivative will be a lot easier. We can do this because the logarithm function is monotonicly increasing.

$$\mathcal{L} = \prod_{i=1}^{n} \lambda e^{(-\lambda x_i)}$$

$$\ell = \log(\mathcal{L}) = \log\left[\prod_{i=1}^{n} \lambda e^{(-\lambda x_i)}\right]$$

$$\ell = \sum_{i=1}^{n} \log\left(\lambda e^{-\lambda x_i}\right)$$

$$\ell = \sum_{i=1}^{n} \log(\lambda) - \lambda x_i$$

$$\ell = n \log(\lambda) - \sum_{i=1}^{n} \lambda x_i$$

Now we want to find the critical points. We can do this by taking the derivative of our log-likelihood ℓ and setting it equal to 0.

$$\begin{aligned}
\frac{\partial}{\partial\lambda}\ell &= \frac{\partial}{\partial\lambda}\Big[n\log(\lambda) - \sum_{i=1}^{n}\lambda x_i\Big] = 0 \\
0 &= \frac{\partial}{\partial\lambda}\Big[n\log(\lambda) - \lambda x_1 - \lambda x_2 - \cdots - \lambda x_n\Big] \\
0 &= \frac{n}{\lambda} - x_1 - x_2 - \cdots - x_n \\
0 &= \frac{n}{\lambda} - \sum_{i=1}^{n} x_i \\
\lambda &= n\Big(\sum_{i=1}^{n} x_i\Big)^{-1}
\end{aligned}$$

Recall that $\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i \to \frac{1}{\bar{x}} = n\Big(\sum_{i=1}^{n} x_i\Big)^{-1}$, so we can plug that into our equation to get our final answer:

$$\boxed{\hat{\lambda} = \frac{1}{\bar{x}}}$$

Question 3

3.a.

The value of the R function "mean" applied to a vector $\mathbf{v} = (v_1, v_2, ...v_n)$ is the arithmentic mean of the vector: $\bar{v} = \frac{1}{n}\sum_{i=1}^{n} v_i$. The value of the R function "var" applied to the vector $\mathbf{v}$ equals $\frac{1}{n-1}\sum_{i=1}^{n}(v_i - \bar{v})^2$, a measure of how much the values differ from the mean. For $\lambda \in \{4, 25, 100\}$, create samples of size 100,000 from the Poisson distribution with parameter λ and the Normal distribution with mean equal to λ and sd equal to $\sqrt{\lambda}$. Please compare the values of "mean", "var", "max", and "min" for the Poisson and the Normal samples corresponding to each λ.(5 points)

```
k <- 100000
lambdas <- c(4, 25, 100)

# Create a dataframe to store the resulting statistics
df <- data.frame(lambda=rep(lambdas, each=2),
                 dist=rep(c("Poisson", "Normal"), times=3))
df$mean <- numeric(nrow(df))
df$var <- numeric(nrow(df))
df$max <- numeric(nrow(df))
df$min <- numeric(nrow(df))

for (i in seq(1, nrow(df), 2)){
    # Get the lambda value
    l = df$lambda[i]

    # Generate k samples from both a Poisson and Normal distribution
    samp.Pois <- rpois(n=k, lambda=l)
```

```
    samp.Normal <- rnorm(n=k, mean=l, sd=sqrt(l))

    # Calculate and record the mean, var, max and min of each sample.
    df$mean[i] <- mean(samp.Pois)
    df$var[i] <- var(samp.Pois)
    df$max[i] <- max(samp.Pois)
    df$min[i] <- min(samp.Pois)

    df$mean[i+1] <- mean(samp.Normal)
    df$var[i+1] <- var(samp.Normal)
    df$max[i+1] <- max(samp.Normal)
    df$min[i+1] <- min(samp.Normal)
}

df
```

```
##   lambda    dist      mean       var       max       min
## 1      4 Poisson   3.99722  4.011112  15.00000  0.000000
## 2      4  Normal   3.99312  3.987294  12.87166 -4.098430
## 3     25 Poisson  24.99989 24.912659  47.00000  6.000000
## 4     25  Normal  24.98625 24.956128  46.28450  4.149583
## 5    100 Poisson 100.07006 99.643708 144.00000 58.000000
## 6    100  Normal 100.02125 99.583464 140.66983 57.944840
```

From the table, we can compare the Poisson and Normal samples. We can see that the means and variances were very similar between the two. The maximums and minimums are also similar, but with slightly larger differences between the values, and have the posibility of being negative, which is impossible for a Poisson random variable. It appears that as λ gets larger, the Poisson and Normal samples get more and more similar.

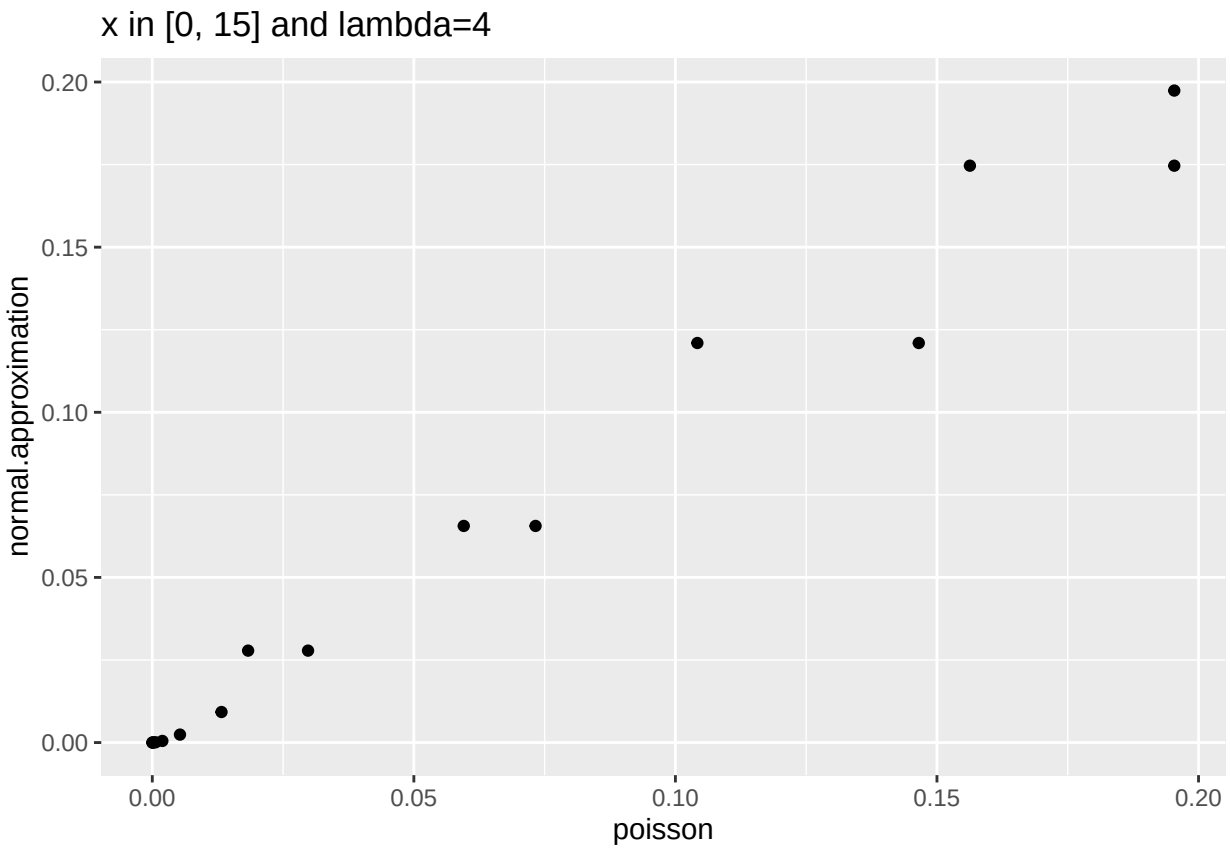
3.b.

For the values of λ and for integers x with the lower bound of 0 and the upper bounds of 15, 60, and 150, respectively, provide visualizations comparing the probability of an outcome equal to x under $Poisson(\lambda)$ and a value in $(x - .5, x + .5)$ under the Normal distribution with $\mu = \lambda$ and $\sigma^2 = \lambda$. Also, for each λ, give the sum of the absolute differences of the two probabilities for all the values of x assessed. (5 points)

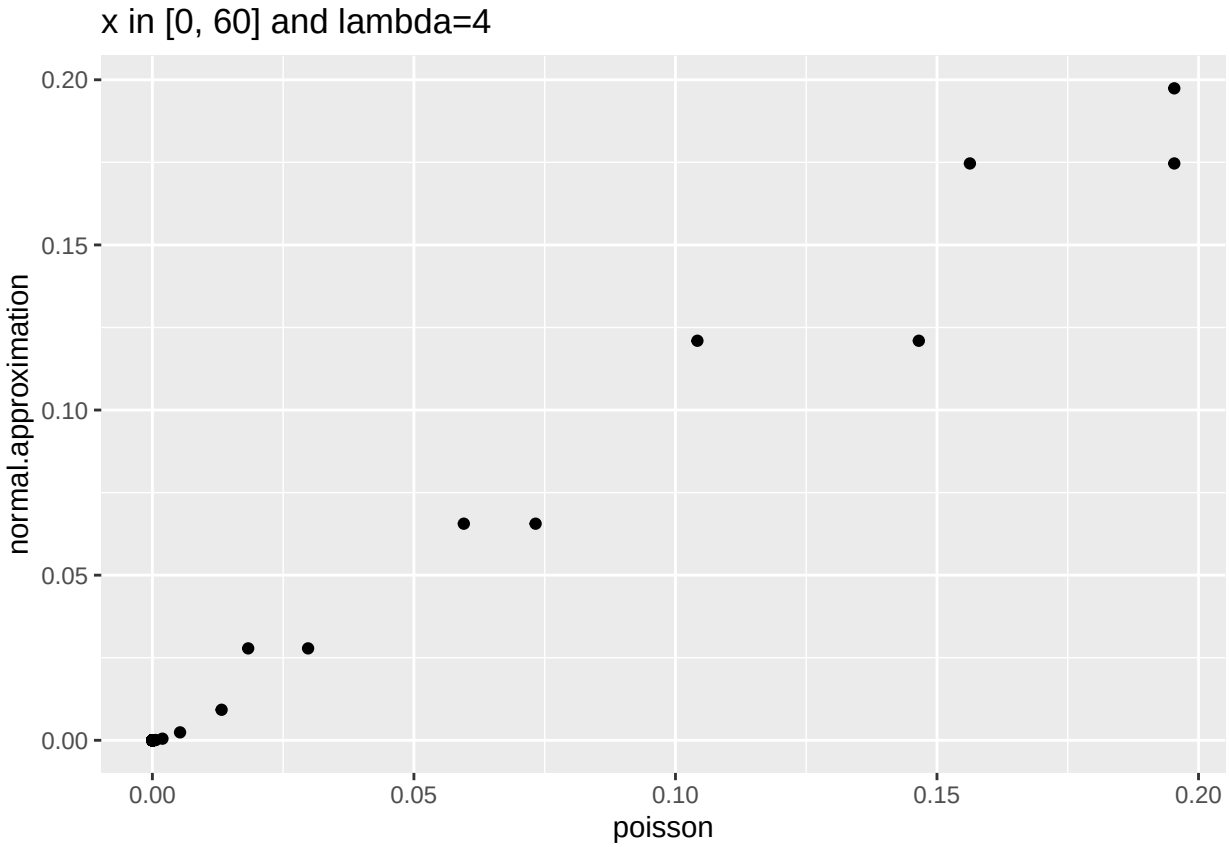
```
upper.bound = c(15, 60, 150)
errors = numeric(9)

for (lambda in lambdas){
  for (u in upper.bound){
    x.range = 0:u
    # Calculate the actual poisson probabilities
    probs.pois = dpois(x.range, lambda=lambda)
    # Calculate the approximate probabilities from the normal.
    probs.normal = pnorm(x.range+0.5, mean=lambda, sd=sqrt(lambda)) - pnorm(x.range-0.5, mean=lambda, s
    # Plot the relation between the two
    df.pois = data.frame(poisson=probs.pois, normal.approximation=probs.normal)
    title = sprintf("x in [0, %s] and lambda=%s", u, lambda)
    g = ggplot(df.pois, aes(x=poisson, y=normal.approximation)) +
          geom_point() +
```

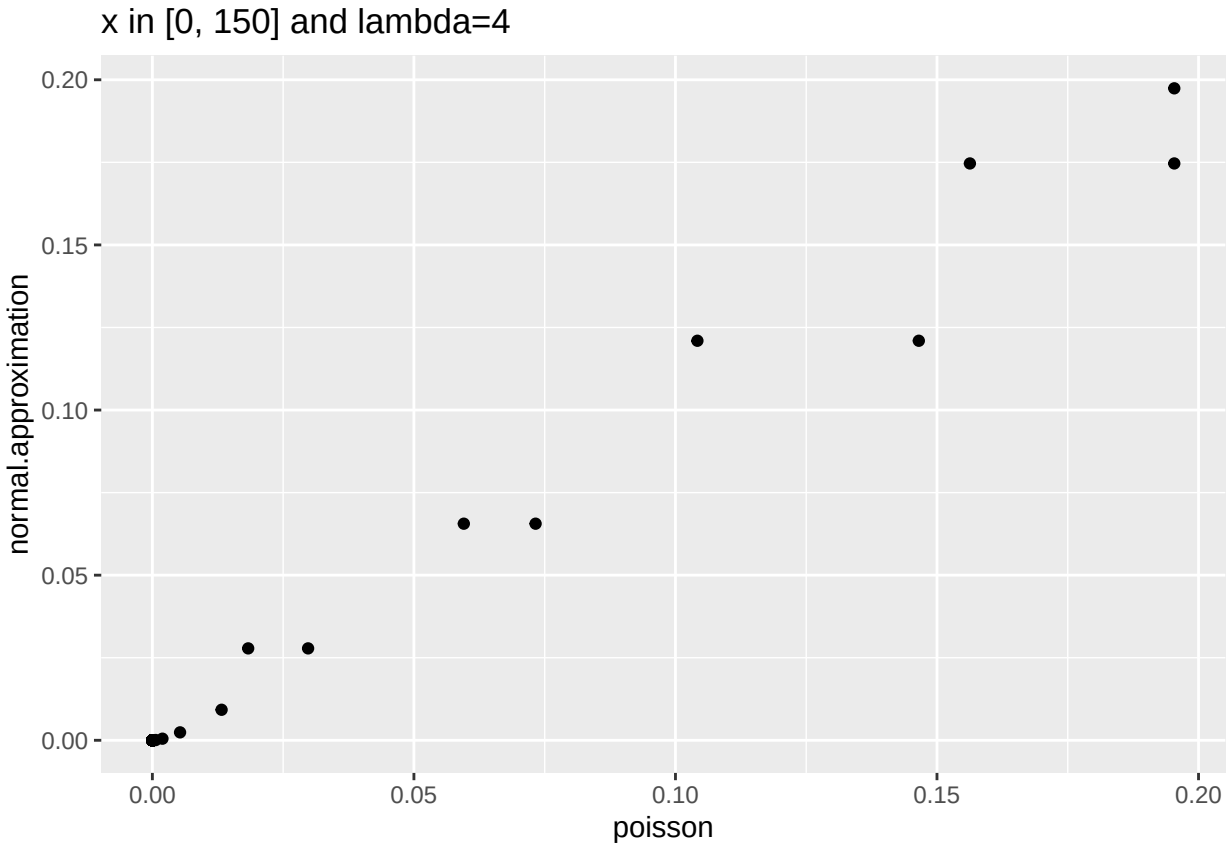
```
          labs(title=title)
    print(g)
    # Calculate the sum of the errors.
    diff = sum(abs(probs.pois - probs.normal))
    out = sprintf("The sum of the absolute differences when lambda=%s and x in [0, %s] is %s", lambda, u
    print(out)
  }
}
```



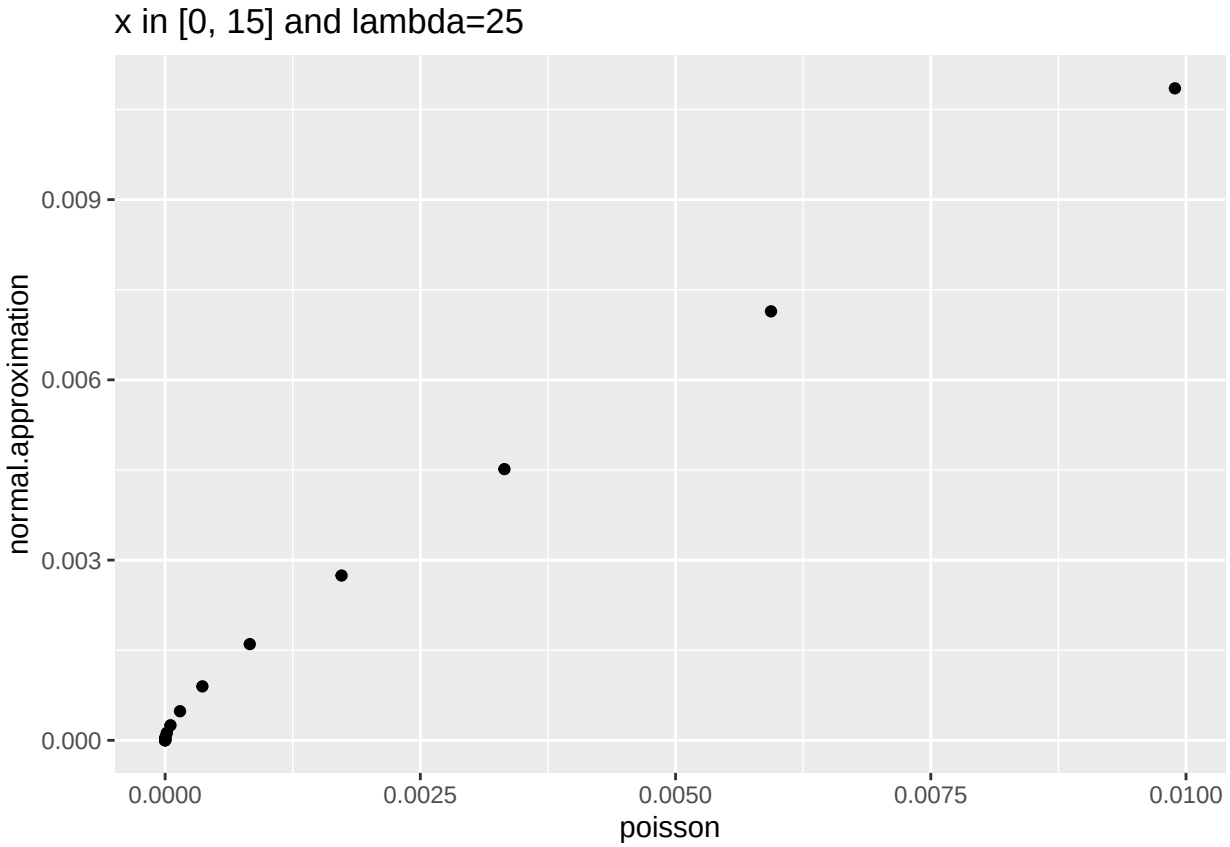

```
## [1] "The sum of the absolute differences when lambda=4 and x in [0, 15] is 0.117759485103899"
```



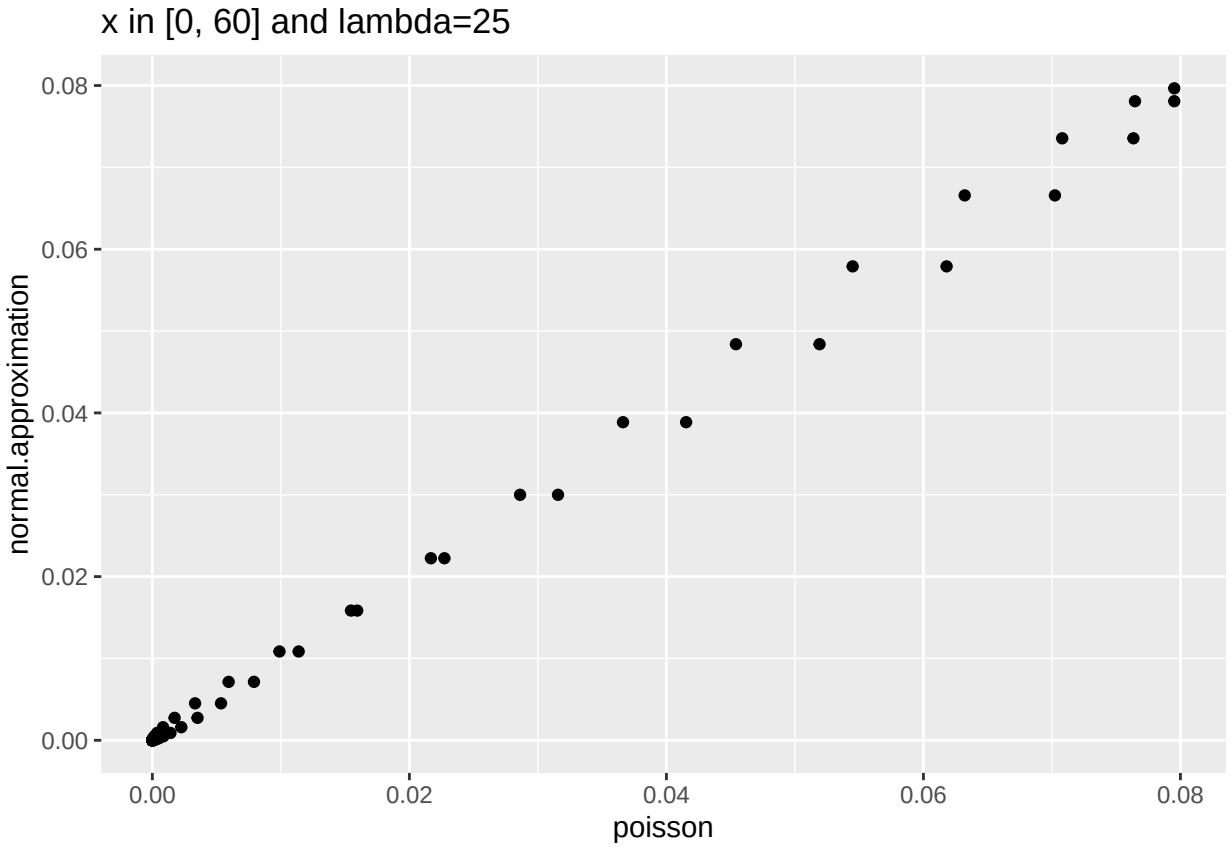

```
## [1] "The sum of the absolute differences when lambda=4 and x in [0, 60] is 0.117764373252447"
```



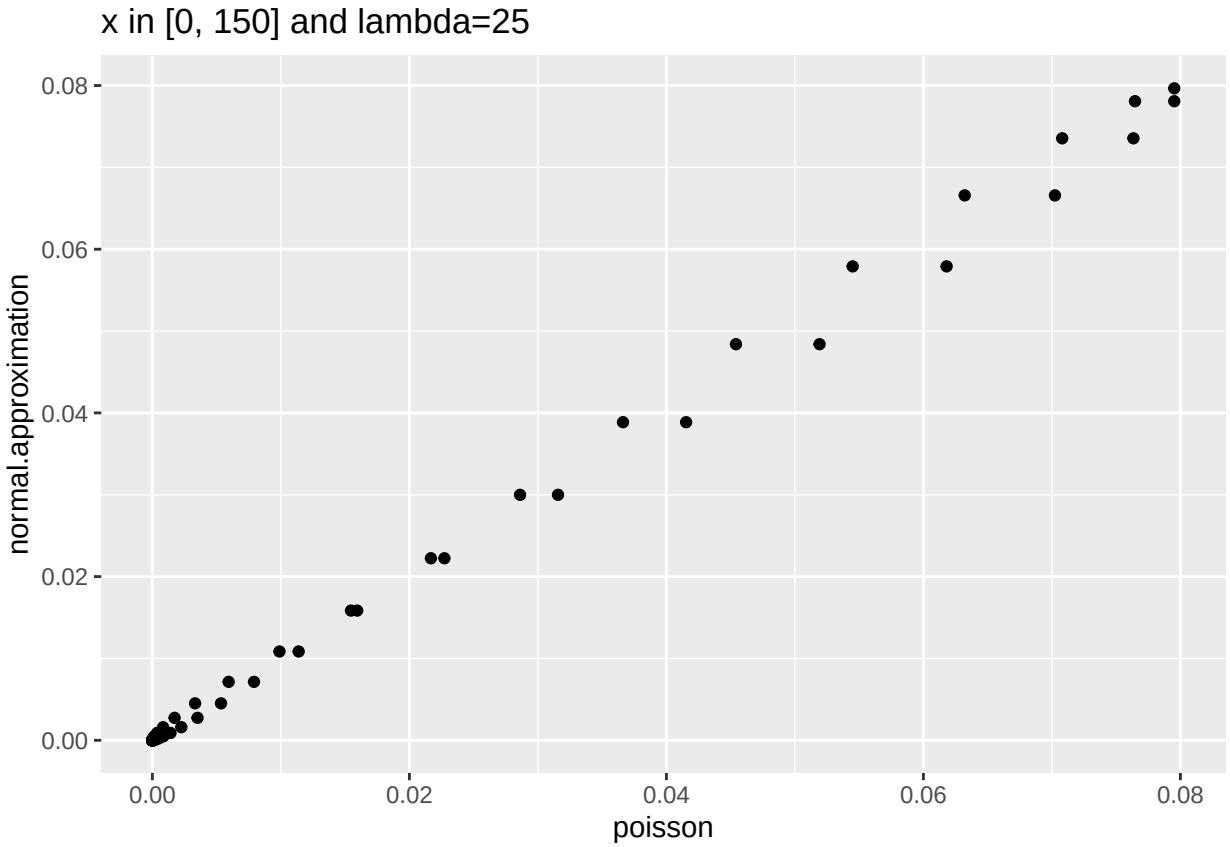

```
## [1] "The sum of the absolute differences when lambda=4 and x in [0, 150] is 0.117764373252447"
```



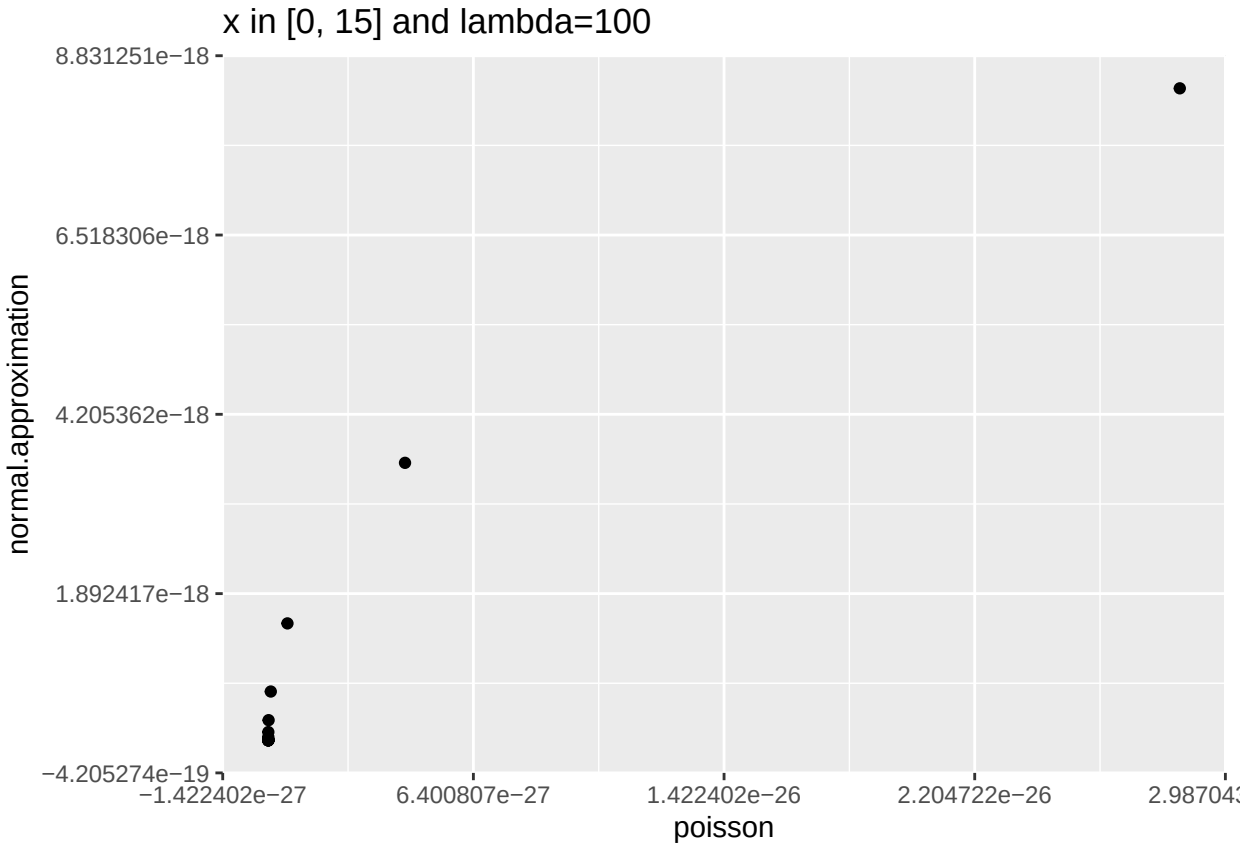

```
## [1] "The sum of the absolute differences when lambda=25 and x in [0, 15] is 0.00642336868189 58"
```



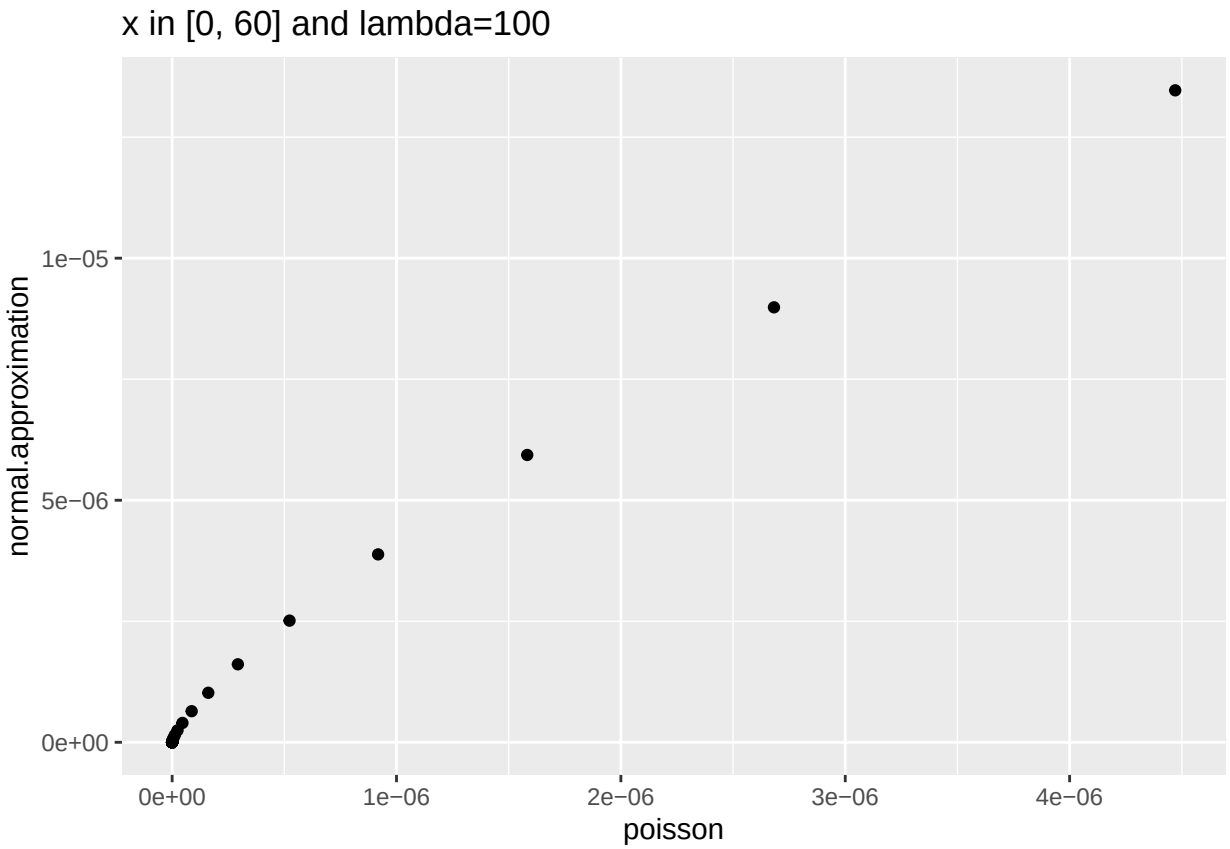

```
## [1] "The sum of the absolute differences when lambda=25 and x in [0, 60] is 0.0505178669005517"
```



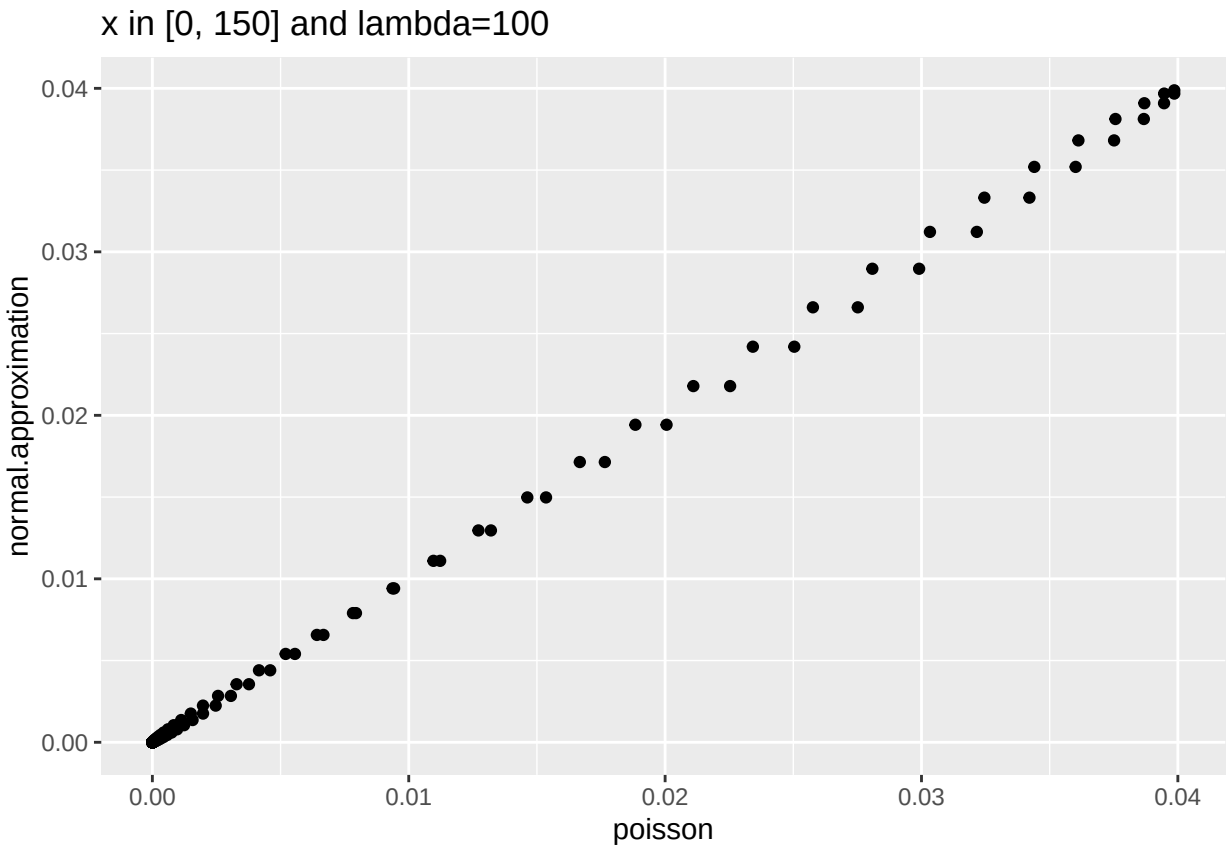

```
## [1] "The sum of the absolute differences when lambda=25 and x in [0, 150] is 0.0505178677563507"
```




```
## [1] "The sum of the absolute differences when lambda=100 and x in [0, 15] is 1.45651364991376e-17"
```




```
## [1] "The sum of the absolute differences when lambda=100 and x in [0, 60] is 2.82633784275434e-05"
```




```
## [1] "The sum of the absolute differences when lambda=100 and x in [0, 150] is 0.0251847140719218"
```

Question 4

Please supply the missing code where indicated.

The data sets in these questions were downloaded 1/17/2020 from https://ourworldindata.org/

The code chunks below read in a data frame of world populations and a data frame of world population densities.

```
dat.pop<-read.csv("population.csv",stringsAsFactors = FALSE)
dat.den<-
  read.csv("population-density.csv",stringsAsFactors = FALSE)
names(dat.den)[4]<-"density"
```

Write code to restrict both data frames to cases in which the value of "Year" is 2000 and the value of "Code" is not the empty string, "", or the value for the whole world,"OWID_WRL". (2 points)

```
dat.pop <- dat.pop[(dat.pop$Year==2000 & dat.pop$Code!="" & dat.pop$Code!="OWID_WRL"), ]
dat.den <- dat.den[(dat.den$Year==2000 & dat.den$Code!="" & dat.den$Code!="OWID_WRL"), ]
```

Merge the data sets.

```
dat.both<-inner_join(dat.den,dat.pop,by="Code")
# check: This should equal 1 if the restriction above is correct.
mean(dat.both$Entity.x==dat.both$Entity.y)
```

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

Write code to find the indices in "dat.both" at which the population takes on its minimum or maximum value and at which the density takes on its minimum or maximum value. Store the resulting indices in a vector named "inds". Create a data frame "dat.text" from dat.both that includes only the rows containing these extremes. (3 points)

```
# Find the indexes of minimum and maximum of both pop and density
idx.min.pop <- which.min(dat.both$Population)
idx.max.pop <- which.max(dat.both$Population)
idx.min.den <- which.min(dat.both$density)
idx.max.den <- which.max(dat.both$density)

# Create a new dataframe of just the rows with the min and max values
inds <- c(idx.min.pop, idx.max.pop, idx.min.den, idx.max.den)
dat.text <- dat.both[inds, ]
dat.text
```

```
##      Entity.x Code Year.x      density  Entity.y Year.y Population
## 200    Tuvalu  TUV   2000 3.140000e+02    Tuvalu   2000       9000
## 42      China  CHN   2000 1.344925e+02     China   2000 1290551040
## 77  Greenland  GRL   2000 1.369229e-01 Greenland   2000      56000
## 114     Macao  MAC   2000 2.139895e+04     Macao   2000     428000
```

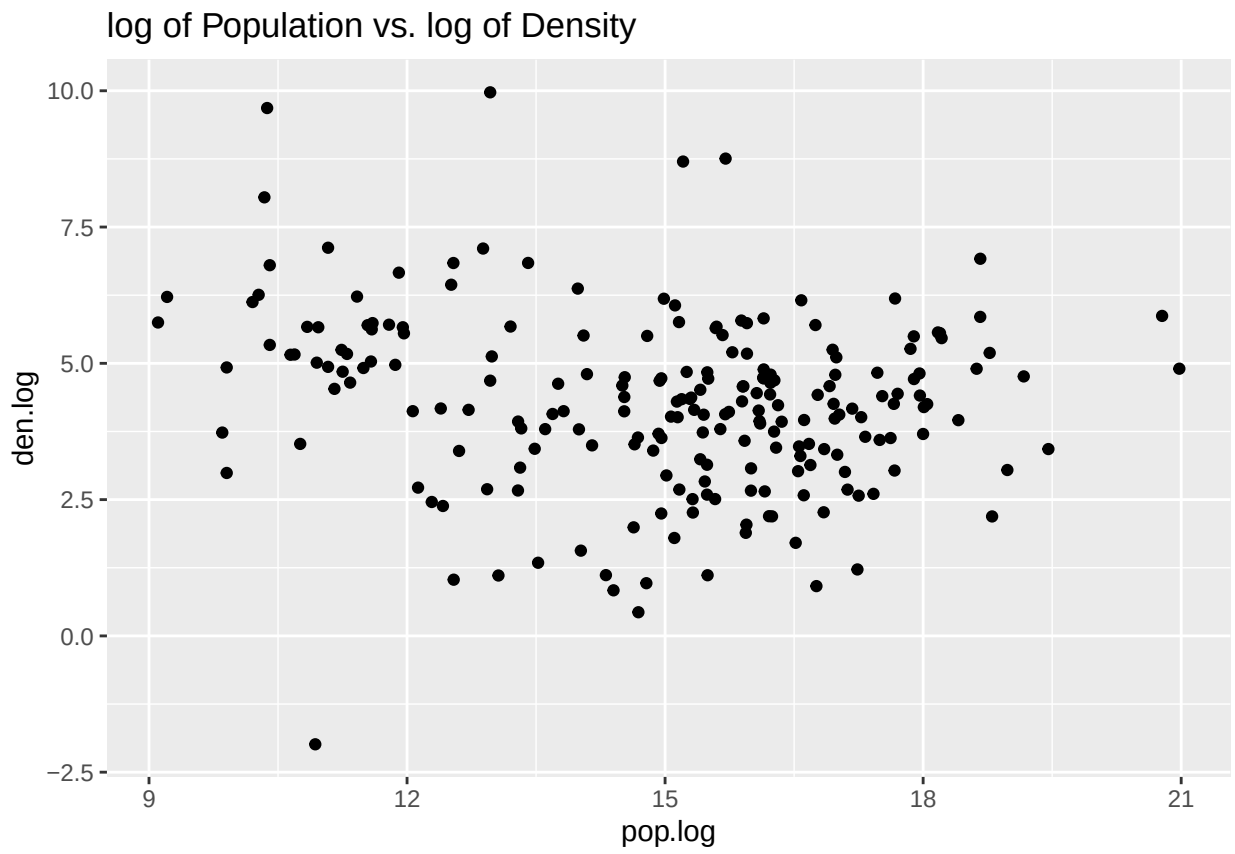
Use "transmute" from dplyr to create a data frame from dat.both with the value for "entity",the log of "density" in "den.log", and the log of "Population" in "pop.log".(3 points)

```
dat.tran <- transmute(dat.both, entity=Entity.x, den.log=log(density), pop.log=log(Population))
head(dat.tran)
```

```
##           entity  den.log  pop.log
## 1    Afghanistan 3.426802 16.84950
## 2        Albania 4.725068 14.95622
## 3        Algeria 2.572063 17.25085
## 4 American Samoa 5.661588 10.96820
## 5        Andorra 4.935392 11.08214
## 6         Angola 2.579274 16.61249
```

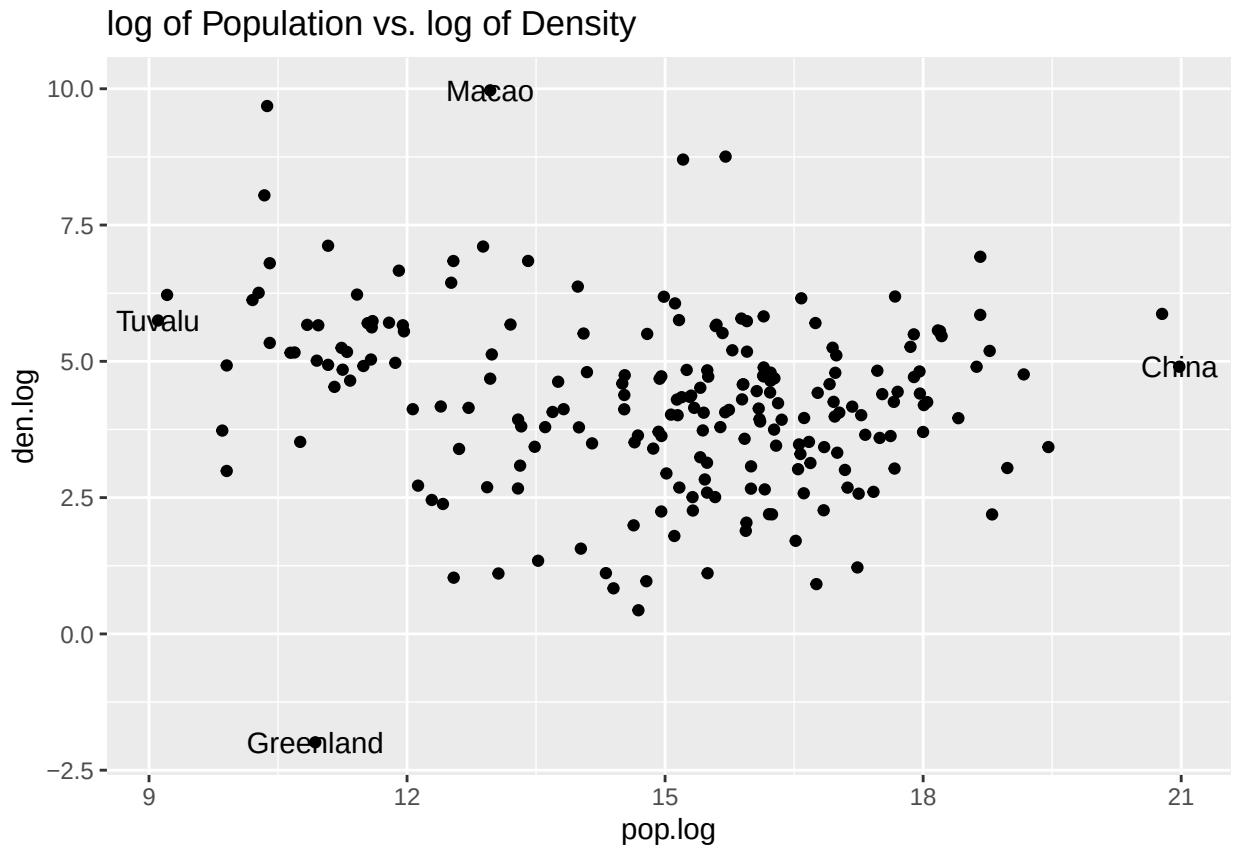
Use "ggplot" to create a point plot of the log of population (on the x-axis) versus the log of density. Store the plot in the variable g. Display the plot. (2 points)

```
g <- ggplot(dat.tran, aes(x=pop.log)) + geom_point(aes(y=den.log)) + labs(title="log of Population vs. l
g
```



The following, when uncommented, should give the previous plot with the names of the entities having extreme population or extreme density, assuming that the result of the "transmute" call was stored back in "dat.both".

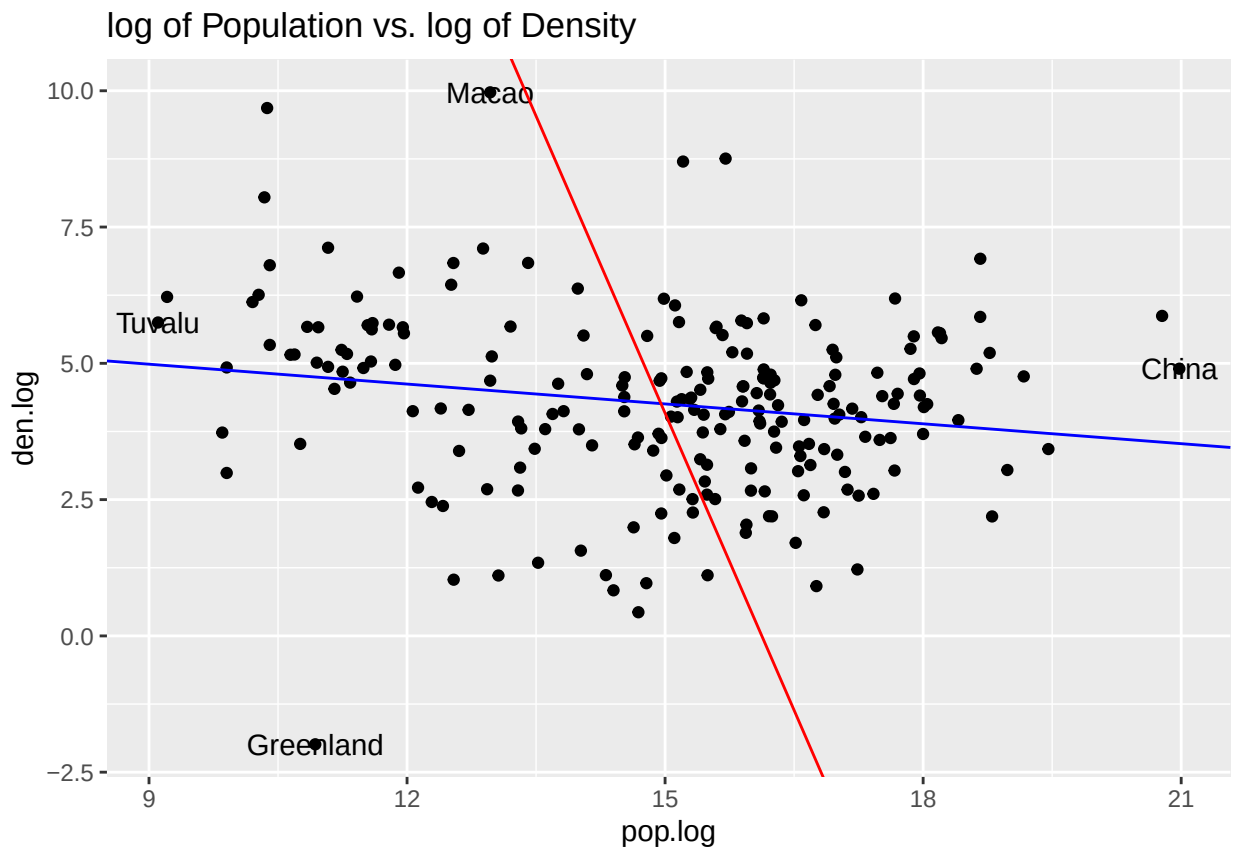
```
dat.text<-dat.tran[inds,]
g<-g+
    geom_text(data=dat.text,aes(x=pop.log,y=den.log,label=entity))
g
```



Please add the least squares best fit line with "pop.log" as the x-value and "den.log" as the y-value in $\mathbf{y} = m\mathbf{x} + b$. Also plot the line minimizing the squared error $\sum (x_i - (ly_i + c))^2)$ again with "pop.log" as the x-value and "den.log" as the y-value in such a way that the points (x, y) on the line are related by $x = ly + c$. That is, if f is the function giving "pop.log" as an affine function of "den.log", minimizing the square error $\sum (x_i - (ly_i + c))^2)$, plot the inverse function f^{-1}. (5 points)

```
# Linear model. population vs density
model = lm(den.log~pop.log, data=dat.tran)
# Inverse model. density vs. population
model.inv = lm(pop.log~den.log, data=dat.tran)

g2 <- g +
      # Linear regression
      geom_abline(slope=model$coef[2], intercept=model$coef[1], color="blue") +
      # Inverse: x = y/m - b/m
      geom_abline(slope=(1/model.inv$coef[2]), intercept=(-model.inv$coef[1]/model.inv$coef[2]), color=
g2
```



Question 5

In some cases, any theoretically useful line through paired data points $\{(x_1, y_1), (x_2, y_2), ...(x_n, y_n)\}$ can be assumed to be of the form $y = mx$ with the y-intercept equal to 0. For example, let x be the "percent remaining" reading on a battery and let y be the time to reach "percent remaining=0" under some regimen for discharging the battery. Derive the formula for m for the least squares best fit line for this model. (5 points)

To find our estimation for m, we are going to minimize the least squares $e = \sum_{i=1}^{n}(y_i - \hat{y})^2 = \sum_{i=1}^{n}(y_i - (mx_i))^2$ in terms of m.

$$
\begin{aligned}
0 &= \frac{\partial}{\partial m}\Big[\sum_{i=1}^{n}(y_i - (mx_i))^2\Big] \\
0 &= \frac{\partial}{\partial m}\big[(y_1 - mx_1)^2 + \cdots + (y_n - mx_n)^2\big] \\
0 &= -2x_1(y_1 - mx_1) + \cdots - 2x_n(y_n - mx_n) \\
0 &= -2\sum_{i=1}^{n} x_i(y_i - mx_i) \\
0 &= \sum_{i=1}^{n} x_i y_i - m\sum_{i=1}^{n} x_i^2 \\
\hat{m} &= \frac{\sum_{i=1}^{n} x_i y_i}{\sum_{i=1}^{n} x_i^2}
\end{aligned}
$$