PHP 2530: Bayesian Statistical Methods HW III

Nick Lewis

04/26/2022

```
library(ggplot2) #this makes better looking plots in R
theme_set(theme_minimal())
library(patchwork) #for plot manipulatios (i.e. subplots)
library(Cairo) #Windows is bad at makinf good ggplots so this helps with resolution
library(tidyr)
library(gridExtra)
library(latex2exp)
```

```
#This function is used to make computing our statistics easier
quantities <- function(x,p,params,sigfig){
  '
    PARAMETERS:
        X - data matrix
        p - the quantiles you wish to calculate
        names - names you want to give the parameters
        sigfig - number of significant digits you want
    Return:
        stats - a datframe detailing statistics about the parameters
    '
  stats <- data.frame(mean = apply(x,2,mean),
                      std = apply(x,2,sd) )

  #add quantile columns
  for (y in p){stats[paste0(100*y,"%")] <- apply(x,2,quantile,probs=y)}
  rownames(stats) <- params
  return( round(stats,sigfig) )
}
```

Problem 1: (BDA 3rd Ed., Exercise 5.2)

a) Reproduce the computations in Section 5.5 for the educational testing example. Use the posterior simulations to estimate (i) for each school j, the probability that its coaching program is the best of the eight; and (ii) for each pair of schools, j and k, the probability that the coaching program in school j is better than that in school k.

```
### Problem 1 (BDA 3rd Ed. Exercise 5.3)

#Part a
#data for school
```

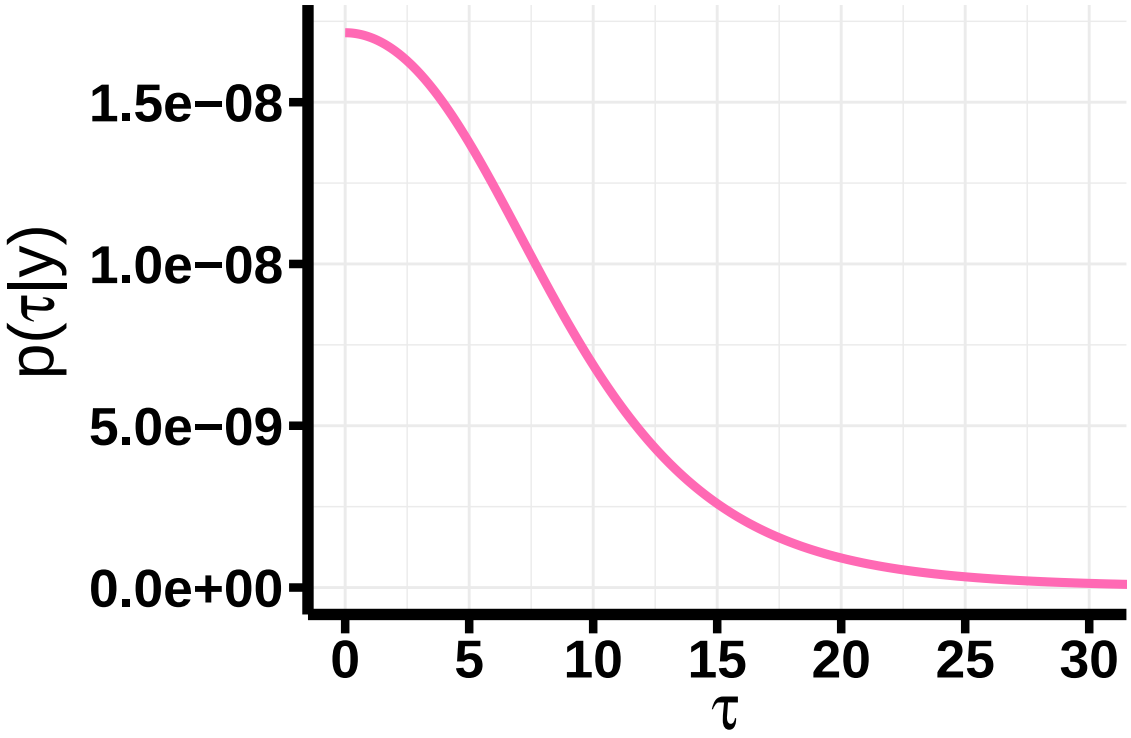
```r
school.data <- data.frame(y = c(28,8,-3,7,-1,1,18,12),
                          sd = c(15,10,16,11,9,11,10,18) )

#Number of iterations
N <- 500
#range of tau as described on page 121
tau <- seq(from = 0.0001, to = 40, length.out = N)
#best guess of mu
mu <- seq(from = -30, to = 30, length.out = N)


#posterior density for (mu,tau)|y (bottom of page 116)
school_prior <- function(x,y) 1
school <- function(a,b,data,sd,prior){
  '
    PARAMETERS:
    ----------
        data - estimated means
        sd - accompanying standard deviations
        a - gridspace for mu
        b - grid space for tau

    Returns:
    -------
    logpost:natural logarithm of unnormalized posterior density
    '
  loglik  <- function(x,y,m,s){
   -((m - x)^2 / (2*(y^2 + s^2))) - 0.5*log(y^2 + s^2)
  }
  logpost <- log(prior(a,b)) #initialize logposterior with prior

  for(j in 1:length(data)) {logpost <- logpost + loglik(a,b,data[j],sd[j])}

return( logpost )
}


#calculates the logposterior
school.post <- outer(mu,tau,school,
                     data=school.data[,"y"],
                     sd=school.data[,"sd"],prior = school_prior)
#calculates the posterior
school.post <- exp(school.post)

#posterior used to select random numbers that correspond to indices of our grid
samples <- sample(length(school.post), size = 2000,replace = T,
                  prob = c(school.post) )

#Random jitter
#plug sample values in to give us result + random jitter
d.mu <- diff(mu)[1]/2
d.tau <- diff(tau)[1]/2
```

```
#I don't declare these anymore to save on memory
mu.post <- rep(mu, times = length(tau))[samples]+runif(2000,-d.mu,d.mu)
tau.post <- rep(tau, each = length(mu))[samples]+runif(2000,0,d.tau)
```

#Reproducing Figures 5.5-7 from Section 5.5

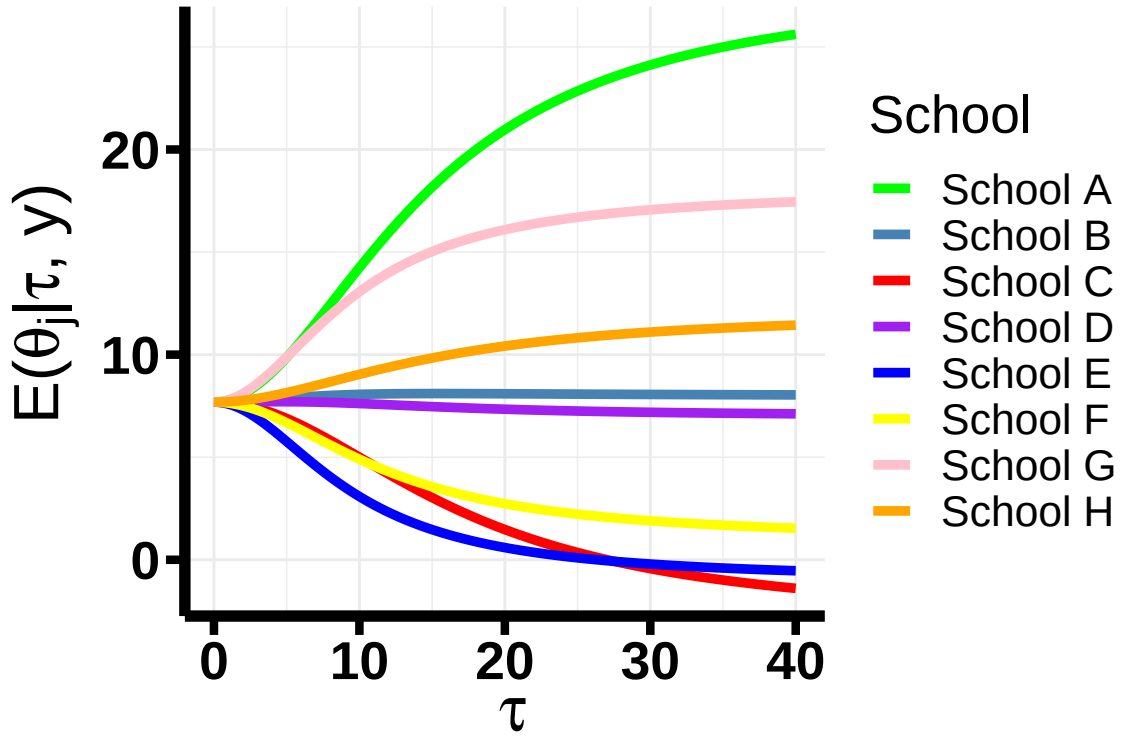
$p(\tau \mid y)$

```
#Plots p(tau |y)
tau.plot <- ggplot()+
  geom_line(aes(x=tau,y  = colSums(school.post)),size=1.60,col="hotpink") +
  coord_cartesian(xlim = c(0,30)) + #chooses limits for x,y axis
  scale_x_continuous(breaks=seq(0,30,by=5))+ #breaks the x-axis into pieces
  theme(axis.line = element_line(colour = "black",size=2),
        text = element_text(size=20),
        axis.text = element_text(colour = "black",size = 20,face="bold"),
        axis.title = element_text(size = 24,face="bold"),
        axis.ticks.length=unit(.25, "cm"),
        axis.ticks = element_line(colour = "black", size = 1.5),
        legend.background = element_blank())+
  ylab(~ paste("p(",tau,"|y)"))+
  xlab(~paste(tau))

tau.plot
```



$E(\theta_j \mid \tau, y)$ Plots

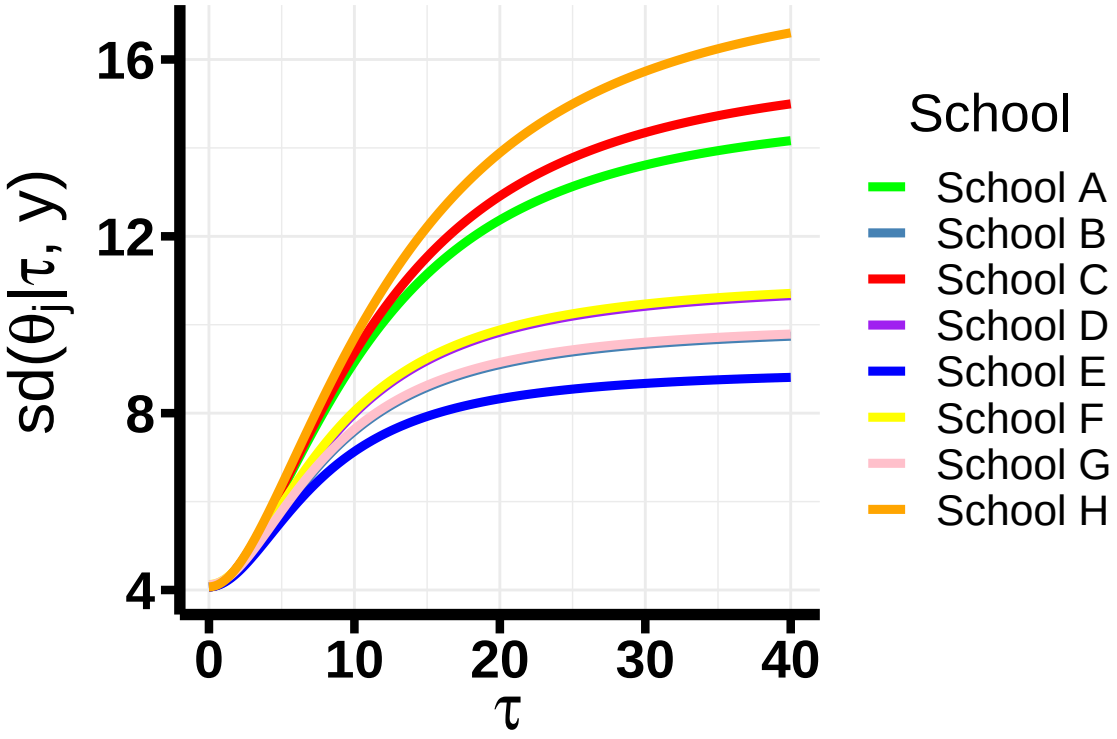
```r
#Graphs of E(theta_j | tau,y)
expected.tau <- function(x,data,sd){
  '
    PARAMETERS:
    ----------
        x - grid for tau
        data - estimated means
        sd - accompanying standard deviations

    Returns:
    -------
    E(theta_j | tau,y) for every school
  '

  mu.hat <- sapply(x, function(t) sum(data/(sd^2 + t^2)) / sum(1/(sd^2+t^2)) )
  V.tau <- outer(x,sd, function(x,y) 1 / ((1/x)^2 + (1/y)^2 ) )
  theta.taus <- outer(mu.hat/x^2,data/sd^2,function(x,y) x+y)*V.tau
  return(theta.taus)
}

school.mean <- expected.tau(x=tau,
                            data=school.data[,"y"],
                            sd=school.data[,"sd"])

letters <- sapply(LETTERS[1:8],function(x) paste0("School ", x))
colnames(school.mean) <- letters
school.mean <- as.data.frame(cbind(tau,school.mean)) %>% gather(school,p,-tau)

ggplot(data=school.mean)+
  geom_line(aes(x=tau,y = p,color=school,group=school),size=1.7) +
  theme(axis.line = element_line(colour = "black",size=2),
        text = element_text(size=20),
        axis.text = element_text(colour = "black",size = 20,face="bold"),
        axis.title = element_text(size = 24,face="bold"),
        axis.ticks.length=unit(.25, "cm"),
        axis.ticks = element_line(colour = "black", size = 1.5),
        legend.background = element_blank())+
  scale_color_manual(name="School", values=c("green",
                                             "steelblue",
                                             "red","purple","blue","yellow",
                                             "pink","orange"),
                     labels=unique(school.mean[,"school"]) )+
  ylab(~ paste("E(",theta[j],"|",tau,", y)"))+
  xlab(~paste(tau))
```



$sd(\theta_j \mid \tau, y)$ Plots

```
#Graphs of sd(theta_j | tau,y)

sd.tau <- function(x,data,sd){
  '
    PARAMETERS:
    ----------
        x - grid for tau
        data - estimated means
        sd - accompanying standard deviations

    Returns:
    -------
    sd(theta_j | tau,y) for every school
  '

  mu.hat.V <- sapply(x, function(t) 1 / sum(1/(sd^2+t^2)) )
  V <- outer(x,sd, function(x,y) 1 / ((1/x)^2 + (1/y)^2 ) )
  V.taus <- outer(x,sd, function(x,y) (1/x)^2 / ((1/x)^2 + (1/y)^2 ) )
  return(sqrt( V+mu.hat.V*V.taus^2 ))
}

school.sd <- sd.tau(x=tau,
                    data=school.data[,"y"],
                    sd=school.data[,"sd"])

#add little jitter so these are visible
school.sd[,7] <- school.sd[,7]+75*diff(school.sd[,7])[1]
```

```r
school.sd[,6] <- school.sd[,6]+75*diff(school.sd[,6])[1]


colnames(school.sd) <- letters
school.sd <- as.data.frame(cbind(tau,school.sd)) %>% gather(school,p,-tau)

ggplot(data=school.sd)+
  geom_line(aes(x=tau,y = p,color=school,group=school),size=1.60) +
  theme(axis.line = element_line(colour = "black",size=2),
        text = element_text(size=20),
        axis.text = element_text(colour = "black",size = 20,face="bold"),
        axis.title = element_text(size = 24,face="bold"),
        axis.ticks.length=unit(.25, "cm"),
        axis.ticks = element_line(colour = "black", size = 1.5),
        legend.background = element_blank())+
  scale_color_manual(name="   School", values=c("green",
                                                 "steelblue",
                                                 "red","purple","blue","yellow",
                                                 "pink","orange"),
                     labels=unique(school.sd[,"school"]) )+
  ylab(~ paste("sd(",theta[j],"|",tau,", y)"))+
  xlab(~paste(tau))
```




```r
#Probability Calculations
theta.post <- function(x,y,data,sd){
  '
    PARAMETERS:
    ----------
```

```
        x - posterior draws for mu
        y - posterior draws for tau
        data - estimated means
        sd - accompanying standard deviations

    Returns:
    -------
    Posterior draws theta_j | tau, mu, y for each school
  '

  #lengths of posterior draws and data
  N <- length(x); n <- length(data)
  #variance
  V <- outer(y,sd, function(x,y) 1 / sqrt((1/x)^2 + (1/y)^2) )
  #mean
  mu <- outer(x/y^2,data/sd^2, function(x,y) x+y)*V^2

  #use updates to now make informed draws of theta_j
  thetas <- sapply(1:n,function(j) rnorm(n = N,mean = mu[,j], sd = V[,j]))
  return(thetas)
}

thetas <- theta.post(mu.post,tau.post,
                     data=school.data[,"y"],
                     sd=school.data[,"sd"])
```

Probability of School θ_j being better than school θ_k

```
#nicer way to write our probability matrix

#I don't like the way it looks so I transpose it
Probabilities <- apply(thetas, 2, function(x) 100*round(colMeans(x > thetas),3))

#turns array into matrix and turns it into dataframe
Probabilities <- as.data.frame(t(Probabilities))
rownames(Probabilities) <- colnames(Probabilities) <- letters

grid.table(Probabilities)
```

School A	School B	School C	School D	School E	School F	School G
0	64.2	69.4	65.1	75	69	52.8
35.8	0	57.7	51.5	61.9	56	38
30.6	42.3	0	43.2	54.8	49.6	32.7
34.8	48.5	56.8	0	62.3	55.2	36.4
25	38.2	45.2	37.8	0	45.4	26.7
31	44	50.4	44.8	54.5	0	32.2
47.2	62.1	67.3	63.6	73.3	67.8	0
37.2	51.1	57.6	53.7	63	58.6	40.5

Probability of School θ_k being the best.

```
'smoother way to calculate best probability. This gives us the col number of the
best, then organizes them (table function), and then we average over our draws.'
```

```
## [1] "smoother way to calculate best probability. This gives us the col number of the \nbest, then or
```

```
Best <- table( apply(thetas,1, which.max) ) / nrow(thetas)

#Just some organizational work
Best <- 100*round(matrix(Best,ncol=nrow(school.data)),3)
colnames(Best) <- letters; rownames(Best) <-"Pr"

grid.table(Best)
```

	School A	School B	School C	School D	School E	School F	School G	School H
Pr	24.9	11	8.8	8.9	4.7	8	20.3	13.5

(b) Repeat (a), but for the simpler model with τ set to ∞ (that is, separate estimation for the eight schools). In this case, the probabilities (ii) can be computed analytically.

```
### PART B
theta.inf <- sapply(1:nrow(school.data), function(x) rnorm(n=2000,
                                mean=school.data[x,"y"],sd=school.data[x,"sd"]))
```

```
#nicer way to write our probability matrix

#I don't like the way it looks so I transpose it
Prob.inf <- apply(theta.inf, 2, function(x) 100*round(colMeans(x > theta.inf),3))

#turns array into matrix and turns it into dataframe
Prob.inf <- as.data.frame(t(Prob.inf))
rownames(Prob.inf) <- colnames(Prob.inf) <- letters

grid.table(Prob.inf)
```

	School A	School B	School C	School D	School E	School F	School G	School H
School A	0	85.5	92.4	87	95	92.4	69.9	73.4
School B	14.5	0	72.1	52.3	74.1	69	23	41.1
School C	7.6	28	0	29.6	43.7	42	12.2	24
School D	13	47.7	70.5	0	70.1	66.4	23.8	39.4
School E	5.1	25.9	56.3	30	0	44.8	7.6	24.8
School F	7.6	31	58	33.6	55.2	0	12.2	29.3
School G	30.1	76.9	87.8	76.2	92.4	87.7	0	60.2
School H	26.6	58.9	76	60.7	75.2	70.7	39.8	0

Probability of School θ_k being the best.

```
Best.inf <- table( apply(theta.inf,1, which.max) ) / nrow(theta.inf)

#Just some organizational work
Best.inf <- 100*round(matrix(Best.inf,ncol=nrow(school.data)),3)
colnames(Best.inf) <-letters; rownames(Best) <-"Pr"

grid.table(Best.inf)
```

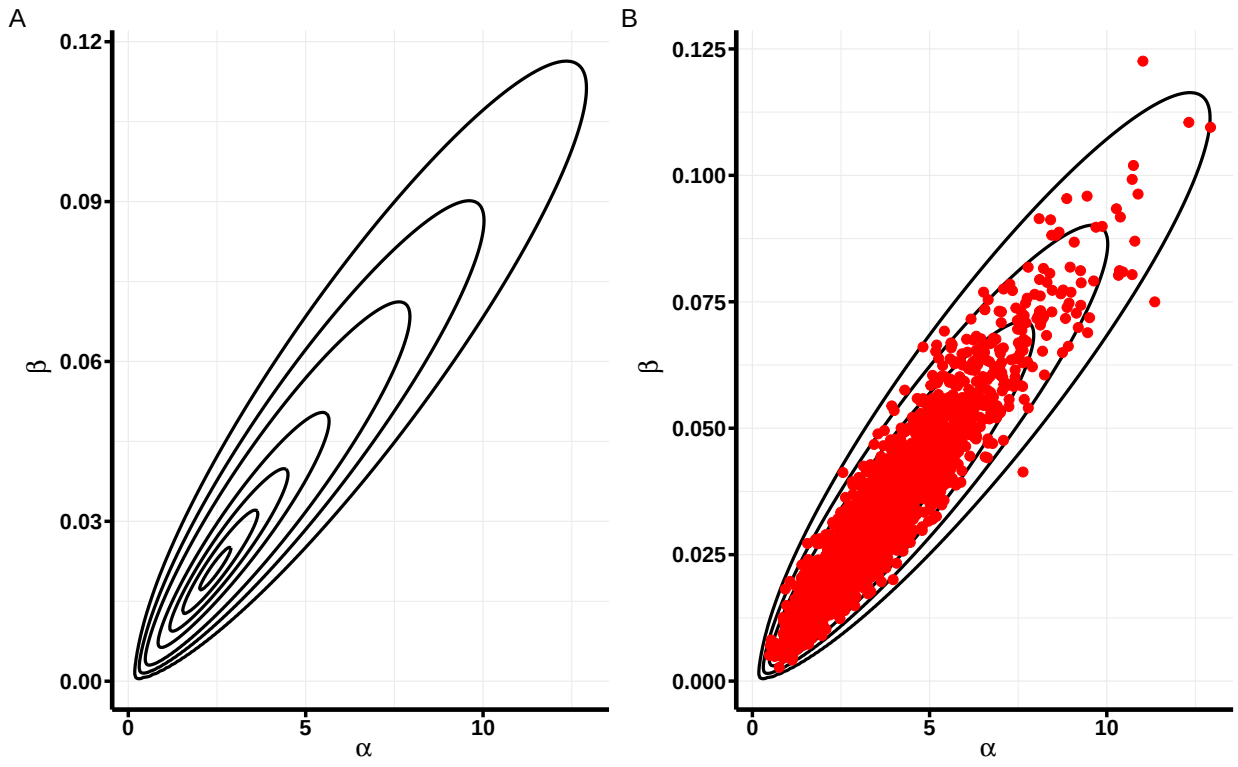
School A	School B	School C	School D	School E	School F	School G	School H
52.4	4	2.4	3.8	0.3	1.1	17.5	18.6

Problem 4 (BDA 3rd Ed. Exercise 5.14) Hierarchical Poisson model: consider the dataset in the previous problem, but suppose only the total amount of traffic at each location is observed.

(b) Compute the marginal posterior density of the hyperparameters and plot its contours. Simulate random draws from the posterior distribution of the hyperparameters and make a scatterplot of the simulation draws.

```r
### PROBLEM 4 (BDA 3rd Ed, Exercise 5.14)

#Data for this problem
#y-bikes for streets w/ bike lanes;v- vehicles for streets w/ bike lanes
res.data <- data.frame(bikes = c(16, 9, 10, 13,19, 20, 18, 17,35, 55) ,
                       vehicles = c(58,90, 48, 57, 103, 57, 86,112, 273, 64))

res.data["total"] <- res.data["bikes"]+res.data["vehicles"]

vehicle_prior <- function(x,y) (x+y)^(-5/2)
vehicle.post <- function(a,b,v,prior){
  '
  PARAMETERS:
  ----------
    v - vehicle data
    a - alpha
    b - beta
    prior - prior distribution for alpha and beta
  Returns:
  -------
    logpost: natural logarithm of the unnormalized posterior density
  '
  N <- length(v) #length of the data

  loglik <- function(a,b,v){
    lgamma(a+v) + a*log(b/(b+1)) - (v)*log(b+1) - lgamma(a) - lgamma(v+1)
  }

  logpost <- log(prior(a,b))
  for (j in 1:N){logpost <- logpost + loglik(a,b,v = v[j])}

  #subract maximum to reduce overflow
  return( logpost )
}

alpha <- seq(from=0.0001, to = 15, length.out=500)
beta <- seq(from=0.0001, to = 0.3, length.out=500)

#Calculation of the log Posterior Distributions
vehicles <- outer(alpha,beta,vehicle.post,v=res.data[,"total"],prior = vehicle_prior)

#calculates posterior
vehicles <- exp(vehicles - max(vehicles))

#Make vectors that contain all pairwise combinations of A and B
alpha.grid <- rep(alpha, times = length(beta))
beta.grid <- rep(beta, each = length(alpha))

v.samps <- 2000
samples <- sample(length(vehicles), size = v.samps,replace = T,
                  prob = c(vehicles) )

#plug sample values in to give us result + random jitter
```

```r
d.alpha <- diff(alpha)[1]/2
d.beta <- diff(beta)[2]/2

alphas.post <-  alpha.grid[samples]+runif(v.samps,min = -d.alpha, max = d.alpha)
beta.post <- beta.grid[samples]+runif(v.samps,min = -d.beta, max = d.beta)

# This is another Posterior plot, but this has the simulated points built on top of it
vehicle.data <- data.frame(alpha = alpha.grid,
                           beta = beta.grid,
                           post = c(vehicles))
point.data <- data.frame(x=alphas.post,y=beta.post,z=c(vehicles)[samples])
#contour levels for flight posterior
levels <- c(0.001,0.01,0.05,0.25,0.50,0.75,0.95)
vehicle.cont <- quantile(seq(min(vehicles),max(vehicles),length.out=1e5),levels)

# Contour Plot of Observed Vehicles Posterior
vehicle.plot <- ggplot(vehicle.data, aes(x=alpha, y= beta, z=post))+
  stat_contour(breaks= vehicle.cont,color="black",size = 1.4)+ #contour levels
  scale_fill_gradient(low = 'yellow', high = 'red', guide = "none") +
  scale_alpha(range = c(0, 1), guide = "none")+
  theme(axis.line = element_line(colour = "black",size=2),
        text = element_text(size=20),
        axis.text = element_text(colour = "black",size = 20,face="bold"),
        axis.title = element_text(size = 24,face="bold"),
        axis.ticks.length=unit(.25, "cm"),
        axis.ticks = element_line(colour = "black", size = 1.5))+
  ylab(~ paste(beta))+
  xlab(~ paste(alpha))

(vehicle.plot | vehicle.plot+geom_point(aes(x=x,y=y,z=z),point.data,colour="red",size=4)) +
  plot_annotation(tag_levels = 'A')
```



(e) Draw samples from the joint posterior distribution of the parameters and hyperparameters, by analogy to the method used in the hierarchical binomial model

```
#Here we get our posterior draws for the parameters and hyperparameters
theta.params <- sapply(1:nrow(res.data), function(x)
                  rgamma(n=v.samps,
                  shape = alphas.post+res.data[x,"total"],
                    rate = beta.post+1))

vehicle.params <- cbind(alphas.post,beta.post,theta.params)
colnames(vehicle.params) <- c("alpha","beta",
                              sapply(1:nrow(res.data),function(j) paste0("theta ",j)) )
#Summary Statistics
grid.table( quantities(x = vehicle.params,
                       p = c(0.025,0.25,0.50,0.75,0.975),
                       params = colnames(vehicle.params), sigfig=2) )
```

	mean	std	2.5%	25%	50%	75%	97.5%
alpha	3.78	1.83	1.12	2.43	3.47	4.85	8.18
beta	0.03	0.02	0.01	0.02	0.03	0.04	0.07
theta 1	75.63	8.5	59.48	70.03	75.27	81.08	93.13
theta 2	99.51	10.03	81.08	92.69	99.22	105.72	120.94
theta 3	59.81	7.83	45.73	54.42	59.39	64.77	75.71
theta 4	71.51	8.46	55.84	65.6	71.02	76.95	88.88
theta 5	121.63	11.06	101.34	114.08	121.35	129.06	143.58
theta 6	77.94	8.64	62.24	71.78	77.59	83.91	95.07
theta 7	103.61	10.28	84.27	96.56	103.2	110.21	125.16
theta 8	128.67	11.14	107.45	121.24	128.43	135.88	152.24
theta 9	302.23	17.28	269.37	291.02	302.15	313.81	336.74
theta 10	118.62	10.57	98.85	111.25	118.32	125.86	139.39

Problem 5: (BDA 3rd Ed., Exercise 6.2) Model checking: in Exercise 2.13, the counts of airline fatalities in 1976–1985 were fitted to four different Poisson models.

```
### PROBLEM 5 (BDA 3rd Ed., Exercise 6.2)

#data for the problem
df <- data.frame(year =1:10 ,
                 accidents=c(24, 25, 31, 31, 22, 21, 26, 20, 16, 22),
                 deaths=c(734, 516, 754, 877, 814, 362, 764, 809, 223, 1066),
                 rate =c(0.19, 0.12, 0.15, 0.16, 0.14, 0.06, 0.13, 0.13, 0.03, 0.15)
)
df["miles"] <- df["deaths"]*(1e8)/df["rate"]

#Prior distribution parameters (here so you can adjust for different priors)
prior.shape <- 0; prior.rate <- 0

R <- 1000
## APPROACH: FIND POSTERIOR PREDICTIVE DISTRIBUTION

sizes <- c(sum(df["accidents"]),sum(df["deaths"])) + prior.shape

#corresponding probability parameters
probs1 <- rep(nrow(df)/(nrow(df)+1+prior.rate),10)
probs2 <-sum(df["miles"])/(sum(df["miles"])+df[,"miles"]+prior.rate)
```

```
# Replication Draws for each model
M1.draws <- replicate(R, rnbinom(n=10,size = sizes[1],prob= probs1) )
M2.draws <- replicate(R, rnbinom(n=10,size = sizes[1],prob= probs2) )
M3.draws <- replicate(R, rnbinom(n=10,size = sizes[2],prob= probs1) )
M4.draws <- replicate(R, rnbinom(n=10,size = sizes[2],prob= probs2) )
```

(a) For each of the models, set up posterior predictive test quantities to check the following assumptions: (1) independent Poisson distributions, (2) no trend over time.

To test (1) Independent Poisson Distributions, we use the 1st lag of the autocorrelations.

$$acf(x)_k = \frac{\sum_{t=1}^{n-k} y_t y_{t+k}}{\sum_{t=1}^{n} y_t}$$

To test (2) No Trend over time we use spearman correlations

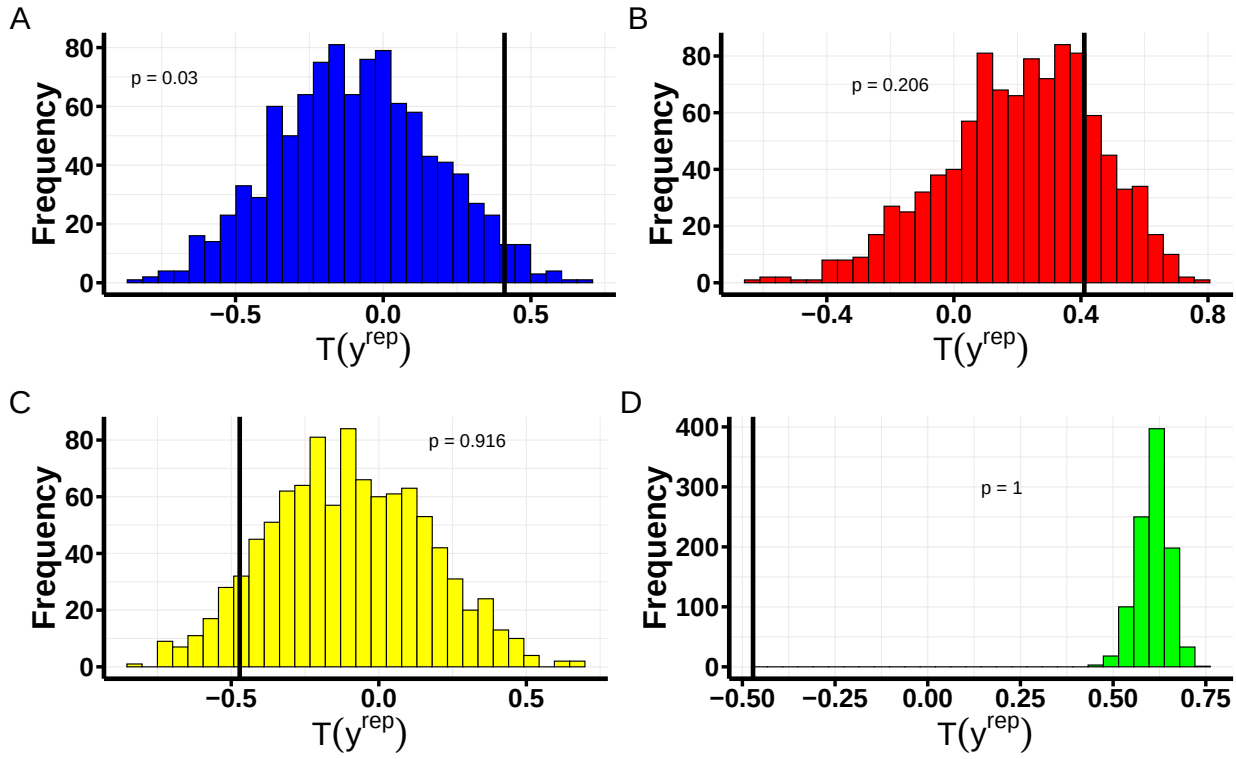
$$r_s = corr(r_X, r_Y) = \frac{Cov(r_X, r_Y)}{sd(r_X)sd(r_Y)}$$
$$= \frac{\sum_{t=1}^{n-k}(r_{t,X} - \bar{r}_X)(r_{t,Y} - \bar{r}_Y)}{\sqrt{\sum_{t=1}^{n}(r_{t,X} - \bar{r}_X)^2}\sqrt{\sum_{t=1}^{n}(r_{t,Y} - \bar{r}_Y)^2}}$$

(b) For each of the models, use simulations from the posterior predictive distributions to measure the discrepancies. Display the discrepancies graphically and give p-values.

Independent Poisson Distributions

```
#First Test Statistic

acf_reps <- cbind(apply(M1.draws,2,function(x) acf(x,plot=F,lag.max=1)$acf[2]),
                          apply(M2.draws,2,function(x) acf(x,plot=F,lag.max=1)$acf[2]),
                          apply(M3.draws,2,function(x) acf(x,plot=F,lag.max=1)$acf[2]),
                          apply(M4.draws,2,function(x) acf(x,plot=F,lag.max=1)$acf[2]) )

acf_obs <- c(acf(df["accidents"],plot=F,lag.max=1)$acf[2],
                 acf(df["accidents"],plot=F,lag.max=1)$acf[2],
                 acf(df["deaths"],plot=F,lag.max=1)$acf[2],
                 acf(df["deaths"],plot=F,lag.max=1)$acf[2])

hist_col <- c("blue","red","yellow","green")

acf_x <- c(-0.74,-0.20,0.3,0.2)
acf_y <- c(70,70,80,300)
acf_hist.plots <- lapply(1:4, function(j){
  ggplot() +aes(x=acf_reps[,j]) +
    geom_histogram(color="black", fill=hist_col[j])+
    geom_vline(xintercept = acf_obs[j],size=2)+
    annotate("text",x = acf_x[j], y= acf_y[j], size = 6,label =
```

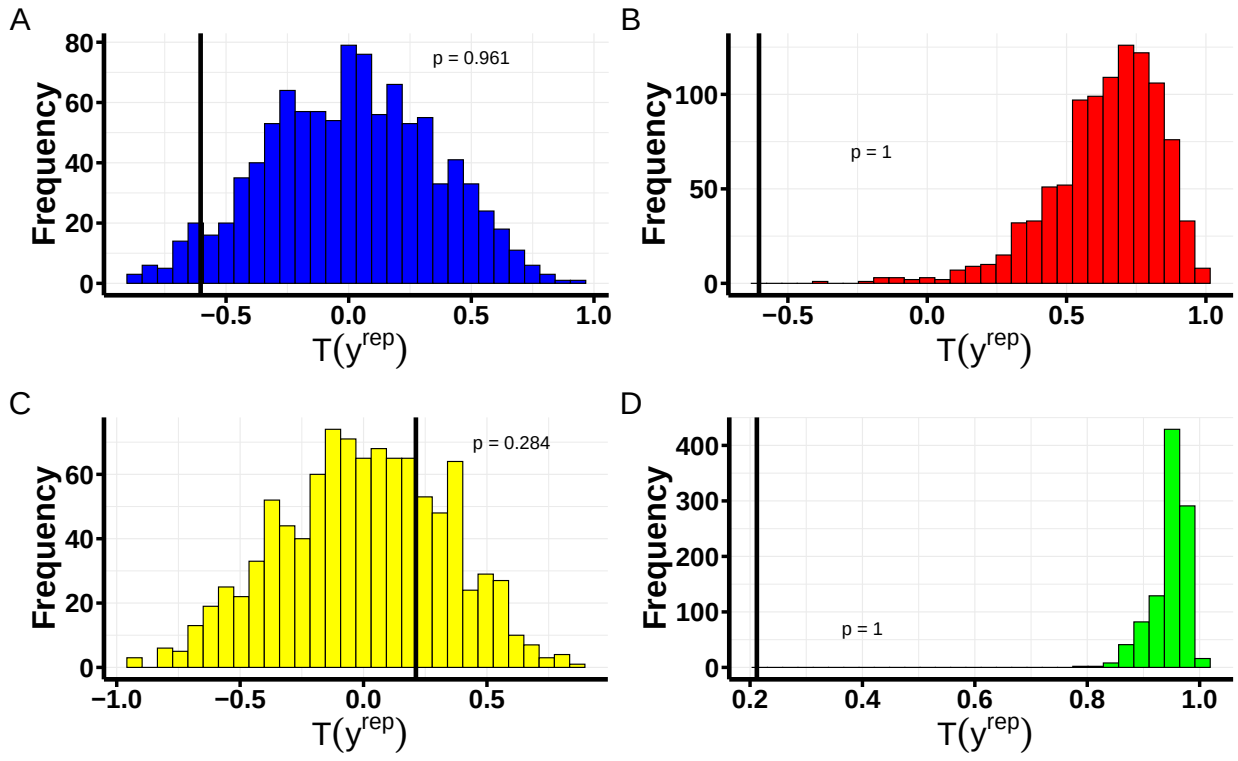
```
                paste0("p = ",mean(acf_reps[,j] >= acf_obs[j])))+
    theme(axis.line = element_line(colour = "black",size=2),
          text = element_text(size=24),
          axis.text = element_text(colour = "black",size = 24,face="bold"),
          axis.title = element_text(size = 30,face="bold"),
          axis.ticks.length=unit(.25, "cm"),
          axis.ticks = element_line(colour = "black", size = 1.5))+
    ylab("Frequency")+
    xlab(~ paste(T(y^{rep})))
}
)

(acf_hist.plots[[1]] | acf_hist.plots[[2]]) /
  (acf_hist.plots[[3]] |  acf_hist.plots[[4]] ) +
  plot_annotation(tag_levels = 'A')
```



No Trend Over Time

```
#Second Test Statistic

#Replications

sp_reps <- cbind(apply(M1.draws,2,function(x) cor(x,df[,"year"],method = "spearman")),
                 apply(M2.draws,2,function(x) cor(x,df[,"year"],method = "spearman")),
                 apply(M3.draws,2,function(x) cor(x,df[,"year"],method = "spearman")),
                 apply(M4.draws,2,function(x) cor(x,df[,"year"],method = "spearman")) )
```

```r
sp_obs <- c(cor(df[,"accidents"],df[,"year"],method = "spearman"),
            cor(df[,"accidents"],df[,"year"],method = "spearman"),
            cor(df[,"deaths"],df[,"year"],method = "spearman"),
            cor(df[,"deaths"],df[,"year"],method = "spearman"))

hist_col <- c("blue","red","yellow","green")

sp_x <- c(0.50,-0.20,0.6,0.4)
sp_y <- c(75,70,70,70)

spearman_hist.plots <- lapply(1:4, function(j){
  ggplot() +aes(x=sp_reps[,j]) +
    geom_histogram(color="black", fill=hist_col[j])+
    geom_vline(xintercept = sp_obs[j],size=2)+
    annotate("text",x = sp_x[j], y= sp_y[j], size = 6,label =
               paste0("p = ",mean(sp_reps[,j] >= sp_obs[j])))+
    theme(axis.line = element_line(colour = "black",size=2),
          text = element_text(size=24),
          axis.text = element_text(colour = "black",size = 24,face="bold"),
          axis.title = element_text(size = 30,face="bold"),
          axis.ticks.length=unit(.25, "cm"),
          axis.ticks = element_line(colour = "black", size = 1.5))+
    ylab("Frequency")+
    xlab(~ paste(T(y^{rep})))
}
)

(spearman_hist.plots[[1]] | spearman_hist.plots[[2]]) /
  (spearman_hist.plots[[3]] |  spearman_hist.plots[[4]] ) +
  plot_annotation(tag_levels = 'A')
```



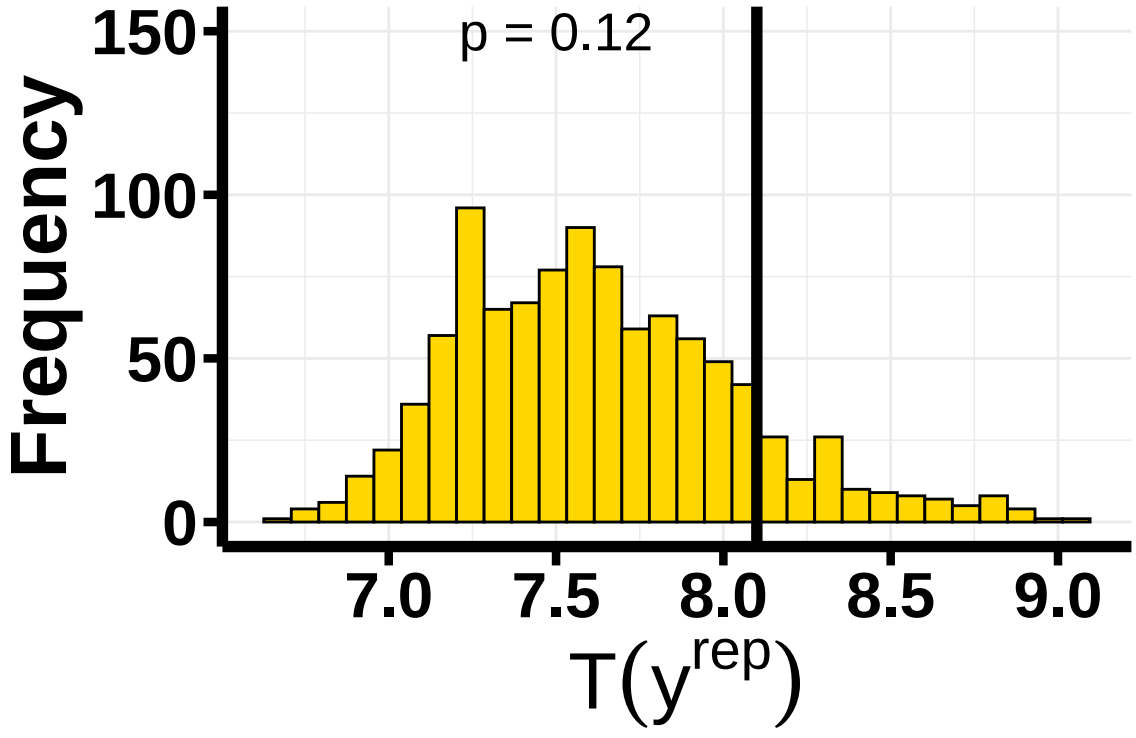
Problem 6: (BDA 3rd Ed. Exercise 6.5)

Prior vs. posterior predictive checks (from Gelman, Meng, and Stern, 1996): consider 100 observations, $y_1, \ldots, y_n$, modeled as independent samples from a $N(\theta, 1)$ distribution with a diffuse prior distribution, say, $p(\theta) = \frac{1}{2A}$ for $\theta \in [-A, A]$ with some extremely large value of A, such as \$10^5 \$. We wish to check the model using, as a test statistic, $T(y) = \max_j |y_j|$: is the maximum absolute observed value consistent with the normal model? Consider a dataset in which $\bar{y} = 5.1$ and $T(y) = 8.1$.

(a) What is the posterior predictive distribution for y^{rep}? Make a histogram for the posterior predictive distribution of $T(y^{rep})$ and give the posterior predictive p-value for the observation $T(y) = 8.1$.

```
### PROBLEM 6 (BDA 3rd Ed. Exercise 6.7)

#PART A
post.pred <- function(y,N){
  '
  PARAMETERS:
  ----------
    y - mean of the data
    N - number of samples in the data

  Returns:
  -------
      test statistic : the absolute value of the maximum from 100 samples
  '
  p <- rnorm(n=N,mean=y,sd=sqrt(1+1/N))
```

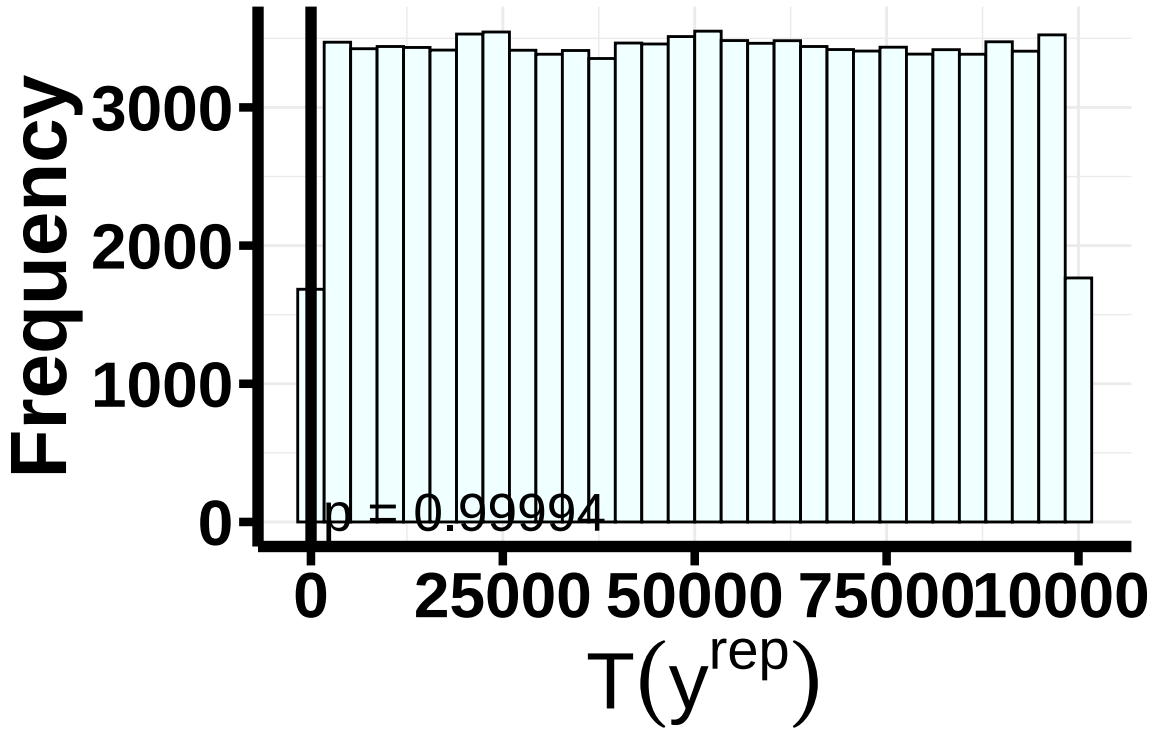
```
  return( abs(max(p)) )
}

sample.max <- replicate(1000,post.pred(y=5.1,N=100))

ggplot() +aes(x=sample.max) +
  geom_histogram(color="black", fill="gold")+
  geom_vline(xintercept = 8.1,size=2)+
  annotate("text",x = 7.5, y=150, size = 7,label =
             paste0("p = ",mean(sample.max > 8.1)))+
  theme(axis.line = element_line(colour = "black",size=2),
        text = element_text(size=24),
        axis.text = element_text(colour = "black",size = 24,face="bold"),
        axis.title = element_text(size = 30,face="bold"),
        axis.ticks.length=unit(.25, "cm"),
        axis.ticks = element_line(colour = "black", size = 1.5))+
  ylab("Frequency")+
  xlab(~ paste(T(y^{rep})))
```



(b) The prior predictive distribution is $p(y^{rep}) = \int p(y^{rep}|\theta)p(\theta)d\theta$. (Compare to equation (6.1).) What is the prior predictive distribution for y^{rep} in this example? Roughly sketch the prior predictive distribution of $T(y^{rep})$ and give the approximate prior predictive p-value for the observation $T(y) = 8.1$.

```
#PART B
prior.pred <- function(n,A){
  '
  PARAMETERS:
```

```
  ----------
    n - number of draws
    A - bounds of the uniform prior on theta (i.e. theta ~Unif(-A,A))

  Returns:
  -------
    test statistic : the absolute value of the maximum from 100 samples
  '
  theta <- runif(n,-A,A)
  draws <- sapply(1:n, function(x)  rnorm(100,theta[x],1))
  reps <- apply(draws, 2, function(x) abs(max(x)) )
  return( reps )
}

#y.prior <- seq(from=-1e5,to=1e5,length.out=2e5)
prior.max <- prior.pred(n=100000,A=1e5)

ggplot() +aes(x=prior.max) +
  geom_histogram(color="black", fill="azure")+
  geom_vline(xintercept = 8.1,size=2)+
  annotate("text",x = 20000, y=75, size = 7,label =
             paste0("p = ",mean(prior.max> 8.1)))+
  theme(axis.line = element_line(colour = "black",size=2),
        text = element_text(size=24),
        axis.text = element_text(colour = "black",size = 24,face="bold"),
        axis.title = element_text(size = 30,face="bold"),
        axis.ticks.length=unit(.25, "cm"),
        axis.ticks = element_line(colour = "black", size = 1.5))+
  ylab("Frequency")+
  xlab(~ paste(T(y^{rep})))
```



Problem 7: (BDA 3rd Ed. Exercise 6.9) Model checking: check the assumed model fitted to the rat tumor data in Section 5.3. Define some test quantities that might be of scientific interest, and compare them to their posterior predictive distributions.

Now we define our test quantities of interest. More or less there are a couple pertinent features that we would like to be seen in the model. Firstly, we want to observe a similar proportion of tumors in our replications as was seen in the the observed data. This would tell us that our model is capable of producing experiments with reasonable tumor frequencies. Secondly, we wouldn't like for our model to over- or under-predict the number of tumors present in the rats. We therefore propose the maximum as test statistic. The maximum will tell us whether or not our model is overpredicting or underpredicting the extremes. We would like to do the same thing with the minimum but the data is extremely sparse; there are many 0's. One option would be to define the number of 0's present as a test statistic to get around this conundrum. Our three statistics will then be:

$$T_1(y) = \max_j \{y_1, y_2, \ldots, y_J\}$$

$$T_2(y) = \frac{1}{J} \sum_{j=1}^{J} \frac{y_j}{n_j}$$

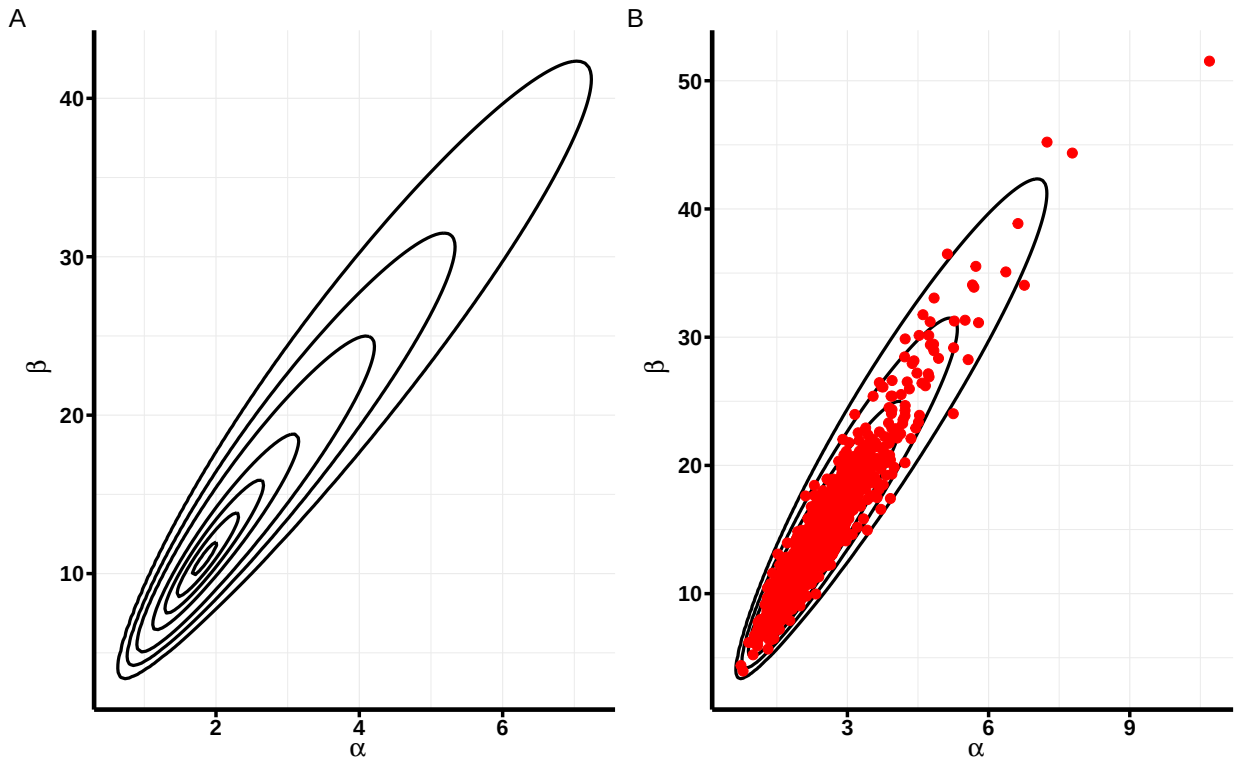
$$T_3(y) = \sum_{j=1}^{J} I_{\{y_j = 0\}}$$

```r
### PROBLEM 7 (BDA 3rd Ed. Exercise 6.9)

#Rat Tumor Data for all 71 Experiments

rat.data <- data.frame(
  tumors = c( 0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  0,  1,  1,  1,
  1,  1,  1,  1,  1,  2,  2,  2,  2,  2,  2,  2,  2,  2,  1,  5,  2,
  5,  3,  2,  7,  7,  3,  3,  2,  9, 10,  4,  4,  4,  4,  4,  4,  4,
  10,  4,  4,  4,  5, 11, 12,  5,  5,  6,  5,  6,  6,  6,  6, 16, 15,
  15,  9,  4),

  rats = c(20, 20, 20, 20, 20, 20, 20, 19, 19, 19, 19, 18, 18, 17, 20, 20, 20,
  20, 19, 19, 18, 18, 25, 24, 23, 20, 20, 20, 20, 20, 20, 10, 49, 19,
  46, 27, 17, 49, 47, 20, 20, 13, 48, 50, 20, 20, 20, 20, 20, 20, 20,
  48, 19, 19, 19, 22, 46, 49, 20, 20, 23, 19, 22, 20, 20, 20, 52, 46,
  47, 24, 14)
  )

R2 <- 200

ralpha <- seq(from = 0.5, to = 12, length = R2)
rbetas <- seq(from = 3, to = 53, length = R2)

#of form seen in derivation
rat.post <- function(y,n,a,b,prior){
  '
  PARAMETERS:
  ----------
    y - data on bicyle proportions on residential street
    n - number of vehicles seen in total
    a - alpha parameter
    b - beta parameter
    prior - prior distribution for alpha, beta
  Returns:
  -------
    logpost: natural logarithm of the unnormalized posterior density
  '

#length of the data
N <- length(y)

# for brevity, split the likelihood into a numerator term and denominator
loglik <- function(a,b,y,n){
  upper <- lgamma(a+b)+lgamma(a + y)+lgamma(b + (n - y))
  lower <- lgamma(a)+lgamma(b)+lgamma(a + b + n)
  return(upper - lower)
    }
logpost <- log(prior(a,b)) #initialize value for the for loop with log prior

for (j in 1:N) { logpost <- logpost + loglik(a,b,y=y[j],n=n[j]) }

return(logpost)
}
```

```
#calculates log posterior
rats <- outer(ralpha,rbetas,rat.post,y=rat.data[,"tumors"],
              n=rat.data[,"rats"], prior = vehicle_prior)

rats <- exp(rats-max(rats)) #calculates the posterior

#Make vectors that contain all pairwise combinations of A and B
ralpha.grid <- rep(ralpha, times = length(rbetas))
rbeta.grid <- rep(rbetas, each = length(ralpha))

'
unravel matrix going row to row instead of column to column. This way we sample
(mu, sigma) instead of having to sample mu, then sigma.
'
```

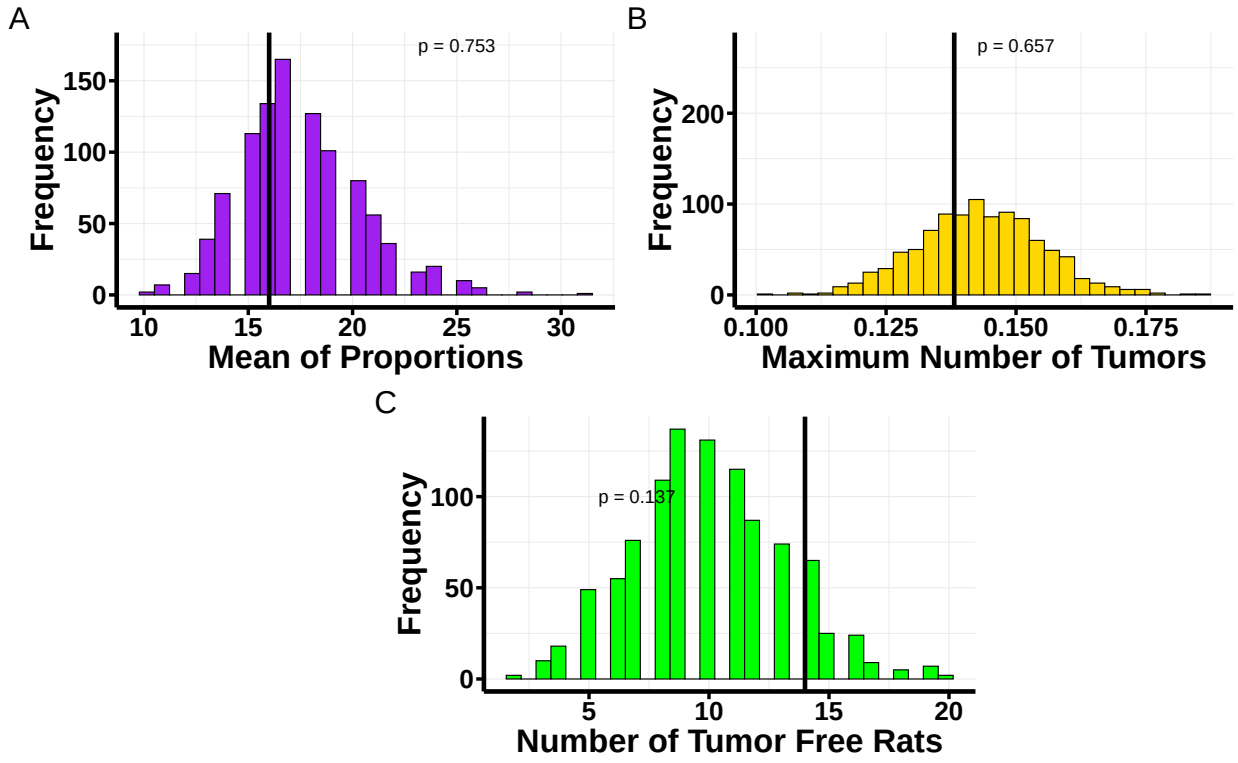
```
## [1] "\nunravel matrix going row to row instead of column to column. This way we sample \n(mu, sigma)
```

```
ratsamps <- 1000
samples <- sample(length(rats), size = ratsamps,replace = T,
                  prob = c(rats) )

#plug sample values in to give us result + random jitter
d.alpha <- diff(ralpha)[1]/2
d.beta <- diff(rbetas)[1]/2

ralpha.post <-  ralpha.grid[samples]+runif(ratsamps,-d.alpha, d.alpha)
rbeta.post <- rbeta.grid[samples]+runif(ratsamps,-d.beta,d.beta)

# This is another Posterior plot, but this has the simulated points built on top of it
rat.stuff <- data.frame(alpha = ralpha.grid,
                        beta = rbeta.grid,
                        post = c(rats))
point.data <- data.frame(x=ralpha.post,y=rbeta.post,z=c(rats)[samples])

#contour levels for flight posterior
levels <- c(0.001,0.01,0.05,0.25,0.50,0.75,0.95)
rat.cont <- quantile(seq(min(rats),max(rats),length.out=1e5),levels)

rat.plot <- ggplot(rat.stuff, aes(x=alpha, y= beta, z=post))+
  stat_contour(breaks= rat.cont,color="black",size = 1.4)+ #contour levels
  scale_fill_gradient(low = 'yellow', high = 'red', guide = "none") +
  scale_alpha(range = c(0, 1), guide = "none")+
  theme(axis.line = element_line(colour = "black",size=2),
        text = element_text(size=20),
        axis.text = element_text(colour = "black",size = 20,face="bold"),
        axis.title = element_text(size = 24,face="bold"),
        axis.ticks.length=unit(.25, "cm"),
        axis.ticks = element_line(colour = "black", size = 1.5))+
  ylab(~ paste(beta))+
  xlab(~ paste(alpha))

 (rat.plot | rat.plot+geom_point(aes(x=x,y=y,z=z),point.data,colour="red",size=4)) +
  plot_annotation(tag_levels = 'A')
```




```
#theta parameters
rat.params <- sapply(1:nrow(rat.data), function(x)
        rbeta(n = ratsamps,
             shape1 = ralpha.post+rat.data[x,"tumors"],
             shape2 = rbeta.post+(rat.data[x,"rats"]-rat.data[x,"tumors"]) ))

rat.draws <- sapply(1:nrow(rat.data), function(x)
                    rbinom(n=ratsamps,size = rat.data[x,"rats"], prob = rat.params[,x]))

# test statistics for Rat Experiment

# (i) Maximum Number of Tumors
# (ii) Proportion of Tumor affected rats
# (iii) Number of Tumorless rats

rat_reps <- cbind(apply(rat.draws,1,max),
                  apply(rat.draws,1,function(x) mean(x/rat.data[,"rats"])),
                       apply(rat.draws,1,function(x) sum(x==0) ))

rat_obs <- c(max(rat.data[,"tumors"]),mean(rat.data[,"tumors"]/rat.data[,"rats"]),
             sum(rat.data[,"tumors"]==0))

rat_labels <- c('Mean of Proportions','Maximum Number of Tumors','Number of Tumor Free Rats')

rat_x <- c(25,0.15,7)
rat_y <- c(175,275,100)

rat_col <- c("purple","gold","green")
```

```
rat_hist <- lapply(1:3, function(j){
  ggplot() +aes(x=rat_reps[,j]) +
    geom_histogram(color="black", fill=rat_col[j])+
    geom_vline(xintercept = rat_obs[j],size=2)+
    annotate("text",x = rat_x[j], y= rat_y[j], size = 6,label =
               paste0("p = ",mean(rat_reps[,j] >= rat_obs[j])))+
    theme(axis.line = element_line(colour = "black",size=2),
          text = element_text(size=24),
          axis.text = element_text(colour = "black",size = 24,face="bold"),
          axis.title = element_text(size = 30,face="bold"),
          axis.ticks.length=unit(.25, "cm"),
          axis.ticks = element_line(colour = "black", size = 1.5))+
    ylab("Frequency")+
    xlab(rat_labels[j])
}
)

( rat_hist[[1]] | rat_hist[[2]] ) /
  (plot_spacer() + rat_hist[[3]] + plot_spacer() + plot_layout(widths = c(1,2,1))) +
  plot_annotation(tag_levels = 'A')
```



Problem 9: BDA 3rd Ed. Exercise 7.6

Fitting a power-transformed normal model: Table 7.3 gives short-term radon measurements for a sample of houses in three counties in Minnesota (see Section 9.4 for more on this example). For this problem, ignore

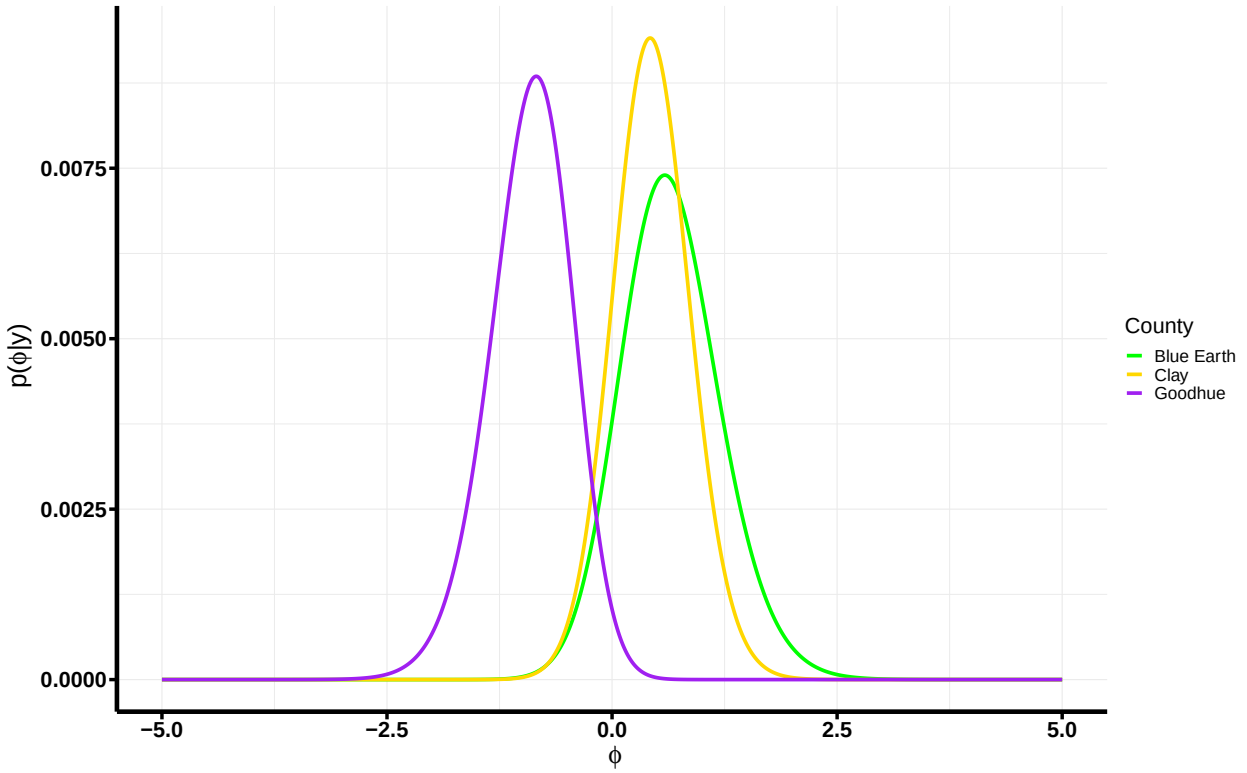
the first-floor measurements (those indicated with asterisks in the table).

(a) Fit the power-transformed normal model from Exercise 7.5(b) to the basement measurements in Blue Earth County.

```
### Problem 9 (BDA 3rd Ed, Exercise 7.6)

blue.earth <- c(5,13,7.2,6.8,12.8,9.5,6,3.8,1.8,6.9,4.7,9.5)
clay <- c(12.9,2.6,26.6,1.5,13,8.8,19.5,9.0,13.1,3.6)
goodhue <- c(14.3,7.6,2.6,43.5,4.9,3.5,4.8,5.6,3.5,3.9,6.7)
#data put into this form since I have to use pooled model
radon <- list(
  "Blue Earth" =  c(5,13,7.2,6.8,12.8,9.5,6,3.8,1.8,6.9,4.7,9.5),
  "Clay" = c(12.9,2.6,26.6,1.5,13,8.8,19.5,9.0,13.1,3.6),
  "Goodhue" =  c(14.3,7.6,2.6,43.5,4.9,3.5,4.8,5.6,3.5,3.9,6.7)
)

boxcox <- function(data,phi) {
  data <- unlist(data) #just in case a list is inputted
  if (min(data) <= 0){
    print("All data must be positive")
  }
  else {
    sapply(data, function(x) ifelse(phi !=0 , (x^(phi) - 1) / (phi), log(x) ) )
  }
}
```

```
#PART A

N9 <- 1000
phi.grid <- seq(from = -5, to = 5 , length.out = N9)

#prior distribution for phi
phi.prior <- function(x) 1

#with our draw of blue earth, we will update our values of phi
phi.post <- function(y,phi,prior){
  '
    PARAMETERS:
    ----------
        y-data
        phi - phi parameter
        prior - prior distribution for phi
    Returns:
    -------
      post: the unnormalized posterior density
    '
n <- length(y) #number of observations

boxcox.var <- var(boxcox(data=y,phi=phi)) #sample variance of boxcox

log_phi.prior <- log(prior(phi)) + (phi-1)*(1-(1/n))*sum(log(y))  #p(phi)
```

```
logpost <- log_phi.prior - ((n-1)/2)*log(boxcox.var) #log posterior

return(logpost)
}

#posteriors for blue earth, clay and goodhue + pooled data
counties.df <- data.frame(x = phi.grid,
                          blueearth = sapply(phi.grid, function(x) phi.post(y = radon[["Blue Earth"]], pl
                           clay = sapply(phi.grid, function(x) phi.post(y = radon[["Clay"]], phi = x,prio
                           goodhue = sapply(phi.grid, function(x) phi.post(y = radon[["Goodhue"]], phi =

#normalize densities
counties.df[,2:4] <- apply(counties.df[,2:4],2,function(x) exp(x)/ sum( exp(x) ))

county.df <- counties.df %>% gather(counties,p,-x)

ggplot(data=county.df) + aes(x=phi.grid)+
  geom_line(aes(x=x,y = p,color=counties,group=counties),size=1.60) +
  theme(axis.line = element_line(colour = "black",size=2),
        text = element_text(size=20),
        axis.text = element_text(colour = "black",size = 20,face="bold"),
        axis.title = element_text(size = 24,face="bold"),
        axis.ticks.length=unit(.25, "cm"),
        axis.ticks = element_line(colour = "black", size = 1.5),
        legend.background = element_blank())+
  scale_color_manual(name="County", values=c("green",
                                             "gold","purple"),
                     labels=c("Blue Earth","Clay","Goodhue"))+
  ylab(~ paste("p(",paste(phi),"|y)"))+
  xlab(~paste(phi))
```




```
#Statistics and values to use for parts a and b

#jitter
d.phi <- diff(phi.grid)[1]/2

#Draws
BE.draws <- ( sample(phi.grid,1000,replace=T,prob=counties.df[,"blueearth"])+
                runif(1000,-d.phi,d.phi) )

#PART A: Draws for mu, sigma for Blue Earth County

norm.param.draws <- function(y,phi){
  '
  PARAMETERS:
  ----------
    y - data
    phi - posterior draws of phi

  Returns:
  -------
    tuple of (mu,sigma) draws
  '

  #First, create your boxcox data
  #NOTE: this uses our external phi function so be wary
  boxcox.data <- boxcox(y,phi)

  #Second,now get sufficient statistics for the data
```

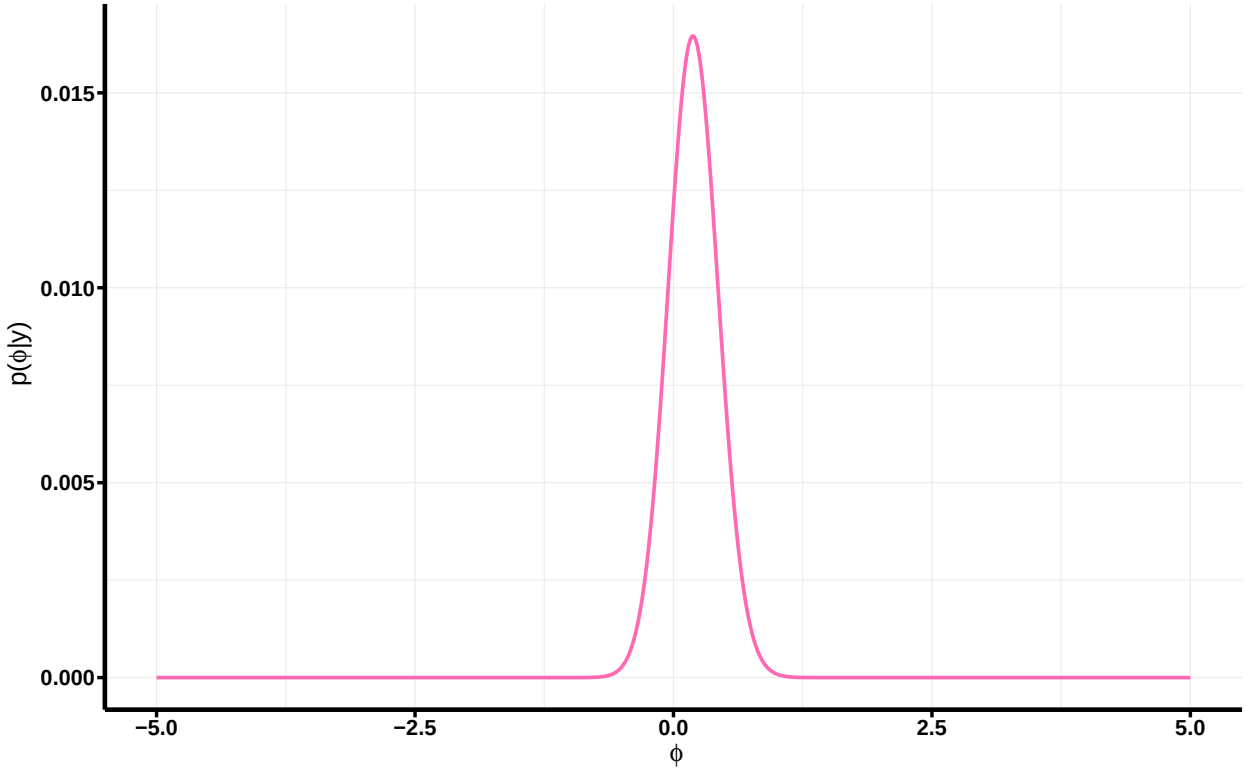
```
  y.mean <- apply(boxcox.data,1,mean)
  y.var <- apply(boxcox.data,1,var)
  n <- length(y)
  #Finally, get our samples of mu, sigma
  sigma <- sqrt( (n-1)*y.var/(rchisq(length(phi),n-1)) )
  mu <- rnorm(length(phi),y.mean, sigma/sqrt(n))

  return(cbind(mu,sigma))
}

# WE CALCULATE THE MEAN AND VARIANCE FOR mu and sigma.

#Summary Statistics
grid.table( quantities(x = cbind(norm.param.draws(radon[["Blue Earth"]],BE.draws),BE.draws),
           p = c(0.025,0.25,0.50,0.75,0.975), params = c("Mu","Sigma","Phi"),
           sigfig=3) )
```

	mean	std	2.5%	25%	50%	75%	97.5%
Mu	5.806	6.866	1.325	2.505	3.929	6.456	21.809
Sigma	3.661	6.107	0.317	0.95	1.886	3.75	17.326
Phi	0.673	0.543	−0.343	0.284	0.666	1.032	1.787

(b) Fit the power-transformed normal model to the basement measurements in all three counties, holding the parameter ϕ equal for all three counties but allowing the mean and variance of the normal distribution to vary

```
#PART B

phi.post.pooled <- function(data,phi,prior){
  '
    PARAMETERS:
    ----------
```

```
        data - list of county data (NOTE: in list form)
        phi - phi parameter
        prior - prior distribution for phi
    Returns:
    -------
      post: the unnormalized posterior density
    '
  y <- unlist(data); n <- sapply(data,length) #pooled data, number of observations
  N <- sum(n) #length of pooled data

  geo.mean <- (phi-1)*(1-(1/N))*sum(log(y)) #geometric mean in log form

  boxcox.vars <- sapply(1:length(n),
                function(x) ((n[x]-1)/2)*log(var(boxcox(data=data[x],phi=phi))))
  logpost <- log(prior(phi)) + geo.mean - sum(boxcox.vars) #log posterior
  return( exp(logpost))
}

counties.df[["pooled"]] <- sapply(phi.grid, function(x) phi.post.pooled(data = radon,phi = x,prior = phi

ggplot() + aes(x=phi.grid,y = counties.df[["pooled"]]/sum(counties.df[["pooled"]]) )+
  geom_line(size=1.60, color = "hotpink") +
  theme(axis.line = element_line(colour = "black",size=2),
        text = element_text(size=20),
        axis.text = element_text(colour = "black",size = 20,face="bold"),
        axis.title = element_text(size = 24,face="bold"),
        axis.ticks.length=unit(.25, "cm"),
        axis.ticks = element_line(colour = "black", size = 1.5),
        legend.background = element_blank())+
  ylab(~ paste("p(",paste(phi),"|y)"))+
  xlab(~paste(phi))
```




```
pooled.draws <- ( sample(phi.grid,1000,replace=T,prob=counties.df[["pooled"]])+
                  runif(1000,-d.phi,d.phi) )

df.countydata <- cbind(pooled.draws,norm.param.draws(radon[["Blue Earth"]],pooled.draws),
                       norm.param.draws(radon[["Clay"]],pooled.draws),
                       norm.param.draws(radon[["Goodhue"]],pooled.draws))

#Summary Statistics
grid.table( quantities(x = df.countydata,
                       p = c(0.025,0.25,0.50,0.75,0.975),
                       params = c("Phi","Blue Earth Mu","Blue Earth Sigma",
                                  "Clay Mu", "Clay Sigma","Goodhue Mu","Goodhue Sigma"),
                       sigfig=3) )
```

	mean	std	2.5%	25%	50%	75%	97.5%
Phi	0.199	0.249	−0.297	0.027	0.203	0.359	0.671
Blue Earth Mu	2.41	0.769	1.348	1.862	2.315	2.771	4.183
Blue Earth Sigma	0.935	0.499	0.324	0.593	0.821	1.142	2.089
Clay Mu	2.919	1.248	1.354	2.087	2.655	3.478	5.979
Clay Sigma	1.683	1.123	0.539	0.979	1.41	2.047	4.55
Goodhue Mu	2.473	1.229	1.191	1.746	2.23	2.828	5.124
Goodhue Sigma	1.798	1.783	0.409	0.86	1.336	2.133	5.914

(c) Check the fit of the model using posterior predictive simulations.

```
#PART C. Getting Samples

#phi to use

phi.mode <- phi.grid[which.max(counties.df[,"pooled"])]
#Inverts the boxcox transform
invboxcox <- function(data,phi) {
    data <- unlist(data)
    sapply(data, function(x) ifelse(phi !=0 , (phi*x+1)^(1/phi),  exp(x) ) )
}


boxcox.samples <- function(n,y,phi){
  '
  PARAMETERS:
  ----------
    n - number of samples
    y - data
    phi - posterior draw of phi

  Returns:
  -------
    data replications for a county
  '

  #draw samples
```

```
  #First, create your boxcox data, and unlist your data

  y <- unlist(y)
  #NOTE: this uses our external phi function so be wary
  boxcox.data <- boxcox(y,phi)

  #Second,now get sufficient statistics for the data
  y.mean <- mean(boxcox.data)
  y.var <- var(boxcox.data)
  N <- length(y)
  #get our samples of mu, sigma
  sigma <- sqrt( (N-1)*y.var/(rchisq(n,N-1)) )
  mu <- rnorm(n,y.mean, sigma/sqrt(N))

  #use samples of mu, sigma to get boxcox draws

  data <- sapply(1:n, function(x) rnorm(n=N,mean=mu[x],sd=sigma[x]) )
  #reverse samples
  invdata <- sapply(1:n, function(x) invboxcox(data[,x],phi))

  return(invdata)
}

BE.samples <- boxcox.samples(n = 1000,
                             y = radon["Blue Earth"],
                             phi = phi.mode)

#use this to make sure no missingness
sum(is.na(c(BE.samples))   )
```

```
## [1] 0
```

```
clay.samples <- boxcox.samples(n=1000,
                               y=radon["Clay"],
                               phi = phi.mode)

sum(is.na(c(clay.samples))   )
```

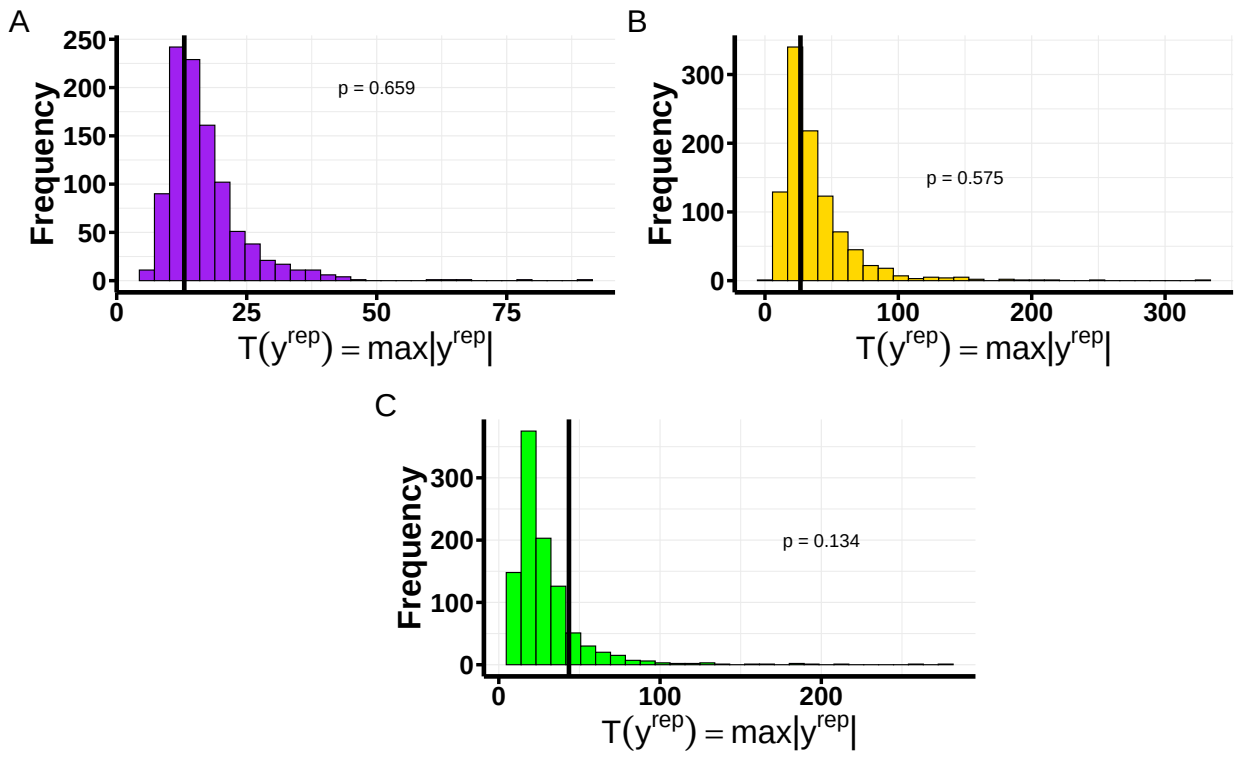
```
## [1] 0
```

```
goodhue.samples <- boxcox.samples(n=1000,
                                  y = radon["Goodhue"],
                                  phi = phi.mode)

sum(is.na(c(goodhue.samples))   )
```

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

Maximum Test Statistic

```r
#Maximum Test Statistic
BlueEarth.max <- apply(BE.samples,2,max,na.rm =TRUE)
Clay.max <- apply(clay.samples,2,max, na.rm = TRUE)
Goodhue.max <- apply(goodhue.samples,2,max, na.rm = TRUE)

# Histograms for Max test statistic

max_reps <- cbind(BlueEarth.max,Clay.max,Goodhue.max)

max_obs <- c(max(radon[["Blue Earth"]]),max(radon[["Clay"]]),max(radon[["Goodhue"]]))

max_x <- c(50,150,200)
max_y <- c(200,150,200)

county_col <- c("purple","gold","green")

max_hist <- lapply(1:3, function(j){
  ggplot() +aes(x=max_reps[,j]) +
    geom_histogram(color="black", fill=county_col[j])+
    geom_vline(xintercept = max_obs[j],size=2)+
    annotate("text",x = max_x[j], y= max_y[j], size = 6,label =
               paste0("p = ",mean(max_reps[,j] >= max_obs[j])))+
    theme(axis.line = element_line(colour = "black",size=2),
          text = element_text(size=24),
          axis.text = element_text(colour = "black",size = 24,face="bold"),
          axis.title = element_text(size = 30,face="bold"),
          axis.ticks.length=unit(.25, "cm"),
          axis.ticks = element_line(colour = "black", size = 1.5))+
    ylab("Frequency")+
    xlab(TeX('$\\T(y^{rep}) = max |y^{rep}|$'))
}
)


( max_hist[[1]] | max_hist[[2]] ) /
  (plot_spacer() + max_hist[[3]] + plot_spacer() + plot_layout(widths = c(1,2,1))) +
  plot_annotation(tag_levels = 'A')
```



Standard Deviation Test Statistic

```
#Standard Deviation Test Statistic

BlueEarth.sd <- apply(BE.samples,2,sd,na.rm=TRUE)
Clay.sd <- apply(clay.samples,2,sd,na.rm = TRUE)
Goodhue.sd <- apply(goodhue.samples,2,sd, na.rm = TRUE)

# Histograms for Standard Deviation test statistic

sd_reps <- cbind(BlueEarth.sd,Clay.sd,Goodhue.sd)

sd_obs <- c(sd(radon[["Blue Earth"]]),sd(radon[["Clay"]]),sd(radon[["Goodhue"]]))

sd_x <- c(15,40,30)
sd_y <- c(200,100,200)

county_col <- c("purple","gold","green")

sd_hist <- lapply(1:3, function(j){
  ggplot() +aes(x=sd_reps[,j]) +
    geom_histogram(color="black", fill=county_col[j])+
    geom_vline(xintercept = sd_obs[j],size=2)+
    annotate("text",x = sd_x[j], y= sd_y[j], size = 6,label =
               paste0("p = ",mean(sd_reps[,j] >= sd_obs[j])))+
    theme(axis.line = element_line(colour = "black",size=2),
          text = element_text(size=24),
          axis.text = element_text(colour = "black",size = 24,face="bold"),
```

```
          axis.title = element_text(size = 30,face="bold"),
          axis.ticks.length=unit(.25, "cm"),
          axis.ticks = element_line(colour = "black", size = 1.5))+
    ylab("Frequency")+
    xlab(TeX('$\\T(y^{rep})$ = $\\sd(y^{rep})$'))
}
)

( sd_hist[[1]] | sd_hist[[2]] ) /
  (plot_spacer() + sd_hist[[3]] + plot_spacer() + plot_layout(widths = c(1,2,1))) +
  plot_annotation(tag_levels = 'A')
```