PHP 2530: BAYESIAN STATISTICAL METHODS
HOMEWORK II R APPENDIX

NICK LEWIS

Code Appendix

```
library(DirichletReg) # allows us to sample from a dirichlet distribution
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


'
Use this pieces of code to get smooth ggplots for windows. I dont like this
method since it uses too much memory in declaring, but do this to save your plots
'

#ggsave(bike.hist,path="~/Bayesian Statistical Methods/HW2/Figures",
#       filename = 'Problem 4 Histograms.png', dpi = 300, type = 'cairo',
#       width = 11.28, height = 7.38, units = 'in',bg="white")
```

Problem 1

```
### PROBLEM 1 (BDA 3rd Ed. Exercise 3.2)

### METHOD 1: SAMPLE FROM THE DIRICHLET DISTRIBUTIONS DIRCECTLY

##pre-debate proportions
pre.theta <- rdirichlet(10000, c(295,308,39))
##post-debate proportions
post.theta <- rdirichlet(10000, c(289,333,20))

#Distribution of those who preferred bush to Dukakis before the debate
pre.alpha <- pre.theta[,1]/(pre.theta[,1] + pre.theta[,2])
#Distribution of those who preferred bush to Dukakis after the debate
post.alpha <- post.theta[,1]/(post.theta[,1] + post.theta[,2])

diff <- post.alpha-pre.alpha

ggplot() +aes(x=diff) +
  geom_histogram(color="black", fill="blue")+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        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")+
```

```r
  xlab(~ paste(alpha[post]," - ", alpha[pre]))

sprintf("The posterior probability of a shift towards Bush is %s ",mean(diff > 0))

### METHOD 2: SAMPLING DIRECTLY FROM THE DISTRIBUTION OF ALPHA

#Distribution of those who preferred bush to Dukakis before the debate
pre.alpha <- rbeta(n=10000,shape1=295, shape2=308)
#Distribution of those who preferred bush to Dukakis after the debate
post.alpha <-  rbeta(n=10000,shape1=289, shape2=333)

diff <- post.alpha-pre.alpha

ggplot() +aes(x=diff) +
  geom_histogram(color="black", fill="blue")+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        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(alpha[post]," - ", alpha[pre]))

sprintf("The posterior probability of a shift towards Bush is %s ",mean(diff > 0))
```

Problem 2

```r
### PROBLEM 2 (BDA 3rd Ed. Exercise 3.3)

#treatment group
#sample size
n.t <- 36; mean.t <- 1.173; sd.t <- 0.20 / sqrt(n.t)

#distribution of treatment group mean (t-distribution is scale-loc family)
mu.t <- sd.t*rt(10000,n.t-1) + mean.t

#control group
#sample size
n.c <- 32; mean.c <- 1.013; sd.c <- 0.24 / sqrt(n.c)
#distribution of control group mean
mu.c <- sd.c*rt(10000,n.c-1) + mean.c

# Our difference in means
mu <- mu.t - mu.c

#mean, standard deviation and 95% credible interval
mu.int <- round(quantile(mu,probs = c(0.025,0.975)),3)


ggplot() +aes(x=mu) +
  geom_histogram(color="black", fill="yellow")+
```

```
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        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("Frequency")+
  xlab(~ paste(mu[treated]," - ", mu[control]))

sprintf("Our mean for the difference is %.5s",mean(mu))
sprintf("Our standard deviation for the difference is %.5s",sd(mu))
sprintf("The Credible Interval for the Difference is [%s]",
        paste0(mu.int, collapse = ', '))
```

Problem 3

```
### PROBLEM 3 (BDA 3rd Ed. Exercise 3.5)

#unrounded/rounded values
w <- c(10,10,12,11,9)

#number of grid points
A <- 200

#picks arbitrary values for mu, log(sigma)
moo <- seq(from = 1, to = 20, length.out = A)
lsig <- seq(from = -3, to = 3, length.out = A)

#PART A : Assume unrounded measurements
unrounded <- function(a, b, x){
     '
    Parameters:
        a - grid space for mean parameter
        b - grid space for standard deviation parameter
        x - data vector
    Returns:
        natural log of unnormalized posterior
    '
  prior <- 1 #prior
  #sample size, mean and variance
  n <- length(x); v <- mean(x); s <- var(x)
  # translate log(sigma) back to sigma;
  b <- exp(b)
  loglik <- function(a,b) -n*log(b) - ( ((n-1)*s + n*(v-a)^2) / (2*b^2) )
  #using p(mu,log(sigma)|y), the prior on p(log(sigma)) propto 1
  logpost <- loglik(a,b) + log(prior)
  return(logpost )
}

#fast way to calculate posterior distribution
system.time(unrounded.post <- outer(moo, lsig,unrounded,w))
#calculates the posterior
```

```r
unrounded.post <- exp(unrounded.post)

#PART B: posterior without rounding

rounded <- function(a,b,x){
  '
    Parameters:
        a - grid space for mean parameter
        b - grid space for standard deviation parameter
        x - data vector
    Returns:
        natural log of unnormalized posterior
    '
  prior <- 1 #prior
  b <- exp(b)   #translate log(sigma) to sigma
  logpost <- log(prior)
  loglik <- function(mu,sig,y) log1p(pnorm(y+0.5,mu,sig) - pnorm(y-0.5,mu,sig)-1)
  #log posterior calculation
  for (j in 1:length(x)){ logpost <-  logpost + loglik(a,b,x[j]) }
  return( logpost )
}


#fast way to calculate posterior distribution
system.time(rounded.post <- outer(moo, lsig,rounded,w))
#calculates the posterior
rounded.post <- exp(rounded.post)

## PART C: COMPARING THE POSTERIOR DISTRIBUTIONS AND CONTOUR PLOTS

#FIRST, WE SAMPLE FROM THE DISTRIBUTIONS

#simulated points from marginal posteriors (unrounded)
B <- 10000
#sample size, sample mean, sample variance
r <- length(w); mu.w <- mean(w); var.w = var(w)

#marginal posterior pdf's for mu and sigma.
sigma.unrounded <- sqrt(( (r-1)*var.w) / (rchisq(B, r-1)) )
mu.unrounded  <- rnorm(B, mu.w, sigma.unrounded/sqrt(r))

#simulated points from marginal posteriors (rounded)

#there is no nice posterior distribution for these so we have to try something else

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


'
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.
'
```

```r
samples <- sample(length(rounded.post), size = B,replace = T,
                  prob = c(rounded.post) )


#sampling this way basically gives us the same pairs + random jitter (see BDA 3rd Ed. pg. 76

#step size for mu, and log sigma grid
d.moo <- diff(moo)[1]/2
d.lsig <- diff(lsig)[1]/2
#add random jitter to make samples continuous
mu.rounded <- moo.grid[samples]+runif(B,min = -d.moo, max = d.moo)
sigma.rounded <- exp(sigma.grid[samples]+runif(B,min = -d.lsig, max = d.lsig))


#create all of the combinations of mu and log(sigma) for contour plot
unrounded.data <- data.frame(mu = moo.grid,
                             logsig = sigma.grid,
                             prob = c((unrounded.post)))
rounded.data <- data.frame(mu = moo.grid,
                             logsig = sigma.grid,
                             prob = c((rounded.post)))
#CONTOUR PLOT OF UNROUNDED

#contour levels
'
ggplot works in a similar method to python. It post the function height,
not the quantile
'
levels <- c(0.0001, 0.001, 0.01,0.05,0.25,0.50,0.75,0.95)
cont1 <- quantile(seq(min(unrounded.post),max(unrounded.post),length.out=1e5),levels)
cont2 <- quantile(seq(min(rounded.post),max(rounded.post),length.out=1e5),levels)

#Unrounded Posterior
unrounded.plot <- ggplot(unrounded.data, aes(x=mu, y= logsig,z=prob))+
  stat_contour(breaks= cont1,color="black",size = 1.4)+ #contour levels
  coord_cartesian(xlim = c(4,18), ylim = c(-3, 3)) + #chooses limits for x,y axis
  scale_x_continuous(breaks=seq(4,18,by=2))+ #breaks the x-axis into pieces
  scale_y_continuous(breaks=seq(-3,3,by=1))+ #breaks y - axis into pieces
  scale_fill_gradient(low = 'yellow', high = 'red', guide = "none") +
  scale_alpha(range = c(0, 1), guide = "none")+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        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("log(",sigma,")"))+
  xlab(~ paste(mu))

unrounded.plot
```

```r
#Rounded Posterior
rounded.plot <- ggplot(rounded.data, aes(x=mu, y= logsig,z=prob))+
  stat_contour(breaks= cont2,color="black",size = 1.4)+ #contour levels
  coord_cartesian(xlim = c(4,18), ylim = c(-3, 3)) + #chooses limits for x,y axis
  scale_x_continuous(breaks=seq(4,18,by=2))+ #breaks the x-axis into pieces
  scale_y_continuous(breaks=seq(-3,3,by=1))+ #breaks y - axis into pieces
  scale_fill_gradient(low = 'yellow', high = 'red', guide = "none") +
  scale_alpha(range = c(0, 1), guide = "none")+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),  #all three make grid disappear
        axis.line = element_line(colour = "black",size=2),
        text = element_text(size=20), #increases text size
        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("log(",sigma,")"))+
  xlab(~ paste(mu))

rounded.plot

#merges plots together
(unrounded.plot | rounded.plot) +
  plot_annotation(tag_levels = 'A')

# WE CALCULATE THE MEAN AND VARIANCE FOR mu and sigma.
df.stats  <- cbind(mu.unrounded, sigma.unrounded,mu.rounded, sigma.rounded)
df.stats  <- data.frame(v = apply(df.stats,2,mean),
                        w = apply(df.stats,2,var),
                        x = apply(df.stats,2,quantile,probs=0.025),
                        y = apply(df.stats,2,quantile,probs=0.50),
                        z = apply(df.stats,2,quantile,probs=0.975))
df.stats <- round(df.stats,4)
rownames(df.stats) <- c("Unrounded mu","Unrounded sigma","Rounded mu",
                        "Rounded sigma")
colnames(df.stats) <- c("Mean","Variance","2.5%","50%","97.5%")

print(df.stats)

#PART D: Calculate mean of (z1-z2)^2

'
NOTE: The Inverse cdf method for the normal distribution works as follows:
1). Let F be the normal cdf. F:[a,b] -> [F(a),F(b)], so F^-1:[F(a),F(b)] -> [a,b].
2). Note [F(a),F(b)]= F(a) + (F(b)-F(a))*[0,1] so F^-1(F(a) + (F(b)-F(a))*[0,1])
    maps those values to [a,b].
'

#repeat B length vector r times; repeat r length vector B times
#I dont like loops so I just made this one big vector
mu.reps <- rep(mu.rounded,times=r); sig.reps <- rep(sigma.rounded,times=r)

#This calculates the cdfs.
upper <- pnorm(rep(w,each=B) + 0.5,mean = mu.reps, sd = sig.reps)
```

```r
lower <- pnorm(rep(w,each=B) - 0.5,mean = mu.reps, sd = sig.reps)

#inverse cdf samples
inv.cdf.samps <- lower + runif(r*B)*(upper-lower)
val <- qnorm(inv.cdf.samps,mean = mu.reps,sd = sig.reps)

#take the r*B length vector and turn into matrix.
#Each column corresponds to samples of unrounded measurements

z <- matrix(val, ncol = r,byrow=F)
mean ((z[,1]-z[,2])^2)
```

Problem 4

```r
### PROBLEM 4 (BDA 3rd Ed. Exercise 3.8)

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

#z-bikes for streets w/o bike lanes;v- vehicles for streets w/o bike lanes
z <- c(12, 1, 2, 4, 9, 7, 9, 8)
w <- c(113, 18, 14, 44,208, 67, 29, 154)
n.z <- w + z

### APPROACH 1: BETA DISTRIBUTION
bike.post <- function(a,b,p){
  '
  PARAMETERS:
    a - alpha values
    b - beta values
    p - proportions
  '
  post <- function(a,b,p){ log(dbeta(p,a,b))}
  q <- 0
  for (j in 1:length(p)){
    q <- q + outer(a,b,post,p[j])
  }
  return( exp(q) )
}

#since our priors are uniform, all we have to do is restrict the grid

alpha <- seq(from=0.001, to = 100, length.out=500)
beta <- seq(from=0.001, to = 100, length.out=500)

#Calculation of the Posterior Distributions
bike.prop.y <- bike.post(alpha,beta,y/n.y)
bike.prop.z <- bike.post(alpha,beta,z/n.z)

#get samples for posterior draws
samples.y <- sample(length(bike.prop.y), size = 1000,replace = T,
```

```
                  prob = c(bike.prop.y) )
samples.z <- sample(length(bike.prop.z), size = 1000,replace = T,
                    prob = c(bike.prop.z) )

#Posterior Draws: I should add a random jitter here, but I don't feel like it

alpha.y.post <-  rep(alpha, times = length(beta))[samples.y]
beta.y.post <- rep(beta, each = length(alpha))[samples.y]

alpha.z.post <-  rep(alpha, times = length(beta))[samples.z]
beta.z.post <- rep(beta, each = length(alpha))[samples.z]

# Difference in Proportions
diff1 <- rbeta(1000,alpha.y.post,beta.y.post)-rbeta(1000,alpha.z.post,beta.z.post)
sprintf("The Difference in proportions for Method 1 is %.5s",mean(diff1))

#METHOD 2: BINOMIAL LIKELIHOOD
#Prior parameters in case you wish to change it around
a2 <- 5; b2 <- 5

#Posterior Distribution for Residential Streets with Bike Lanes
theta.y <- rbeta(n=1000,shape1=a2+sum(y), shape2=b2+sum(n.y)-sum(y))
#Posterior Distribution for Residential Streets without Bike Lanes
theta.z <- rbeta(n=1000,shape1=a2+sum(z), shape2=b2+sum(n.z)-sum(z))

#takes average proportion for each draw
y.samples <- apply(outer(n.y,theta.y, rbinom,n=10000), 2, function(x) x/n.y)
z.samples <- apply(outer(n.z,theta.z, rbinom,n=8000), 2, function(x) x/n.z)

#Difference in Proportions for Method 2
diff2 <- colMeans(y.samples) - colMeans(z.samples)
sprintf("The Difference in proportions for Method 2 is %.5s",mean(diff2))

#APPROACH 3:

#METHOD 3: Separate Bikes and Vehicles

#Parameters for our gamma priors (y- bicycles, vehichles. z - bicyles, vehicles)
a.b <- 15; a.v <- 85; b <- 1

#Posterior distributions for y, v based off choice of prior
theta.by <- rgamma(n=10000,shape = a.b+sum(y), rate=(b+length(y)))
theta.vy <-rgamma(n=10000,shape = a.v+sum(v), rate=(b+length(v)))

#Posterior distributions for z,w based off choice of prior
theta.bz <- rgamma(n=10000,shape = a.b+sum(z), rate=(b+length(z)))
theta.vz <-rgamma(n=10000,shape = a.v+sum(w), rate=(b+length(w)))


#sum of the rates
yrate <- rpois(n=10000,lambda = theta.by)+rpois(n=10000,lambda = theta.vy)
zrate <- rpois(n=10000,lambda = theta.bz)+rpois(n=10000,lambda = theta.vz)

#this is the proportion of bikes that we see
```

```r
y.bikes <- rpois(n=10000,lambda = theta.by) / yrate
z.bikes <- rpois(n=10000,lambda = theta.bz) / zrate

#Taking difference of proportions between street w/ bike lane vs without
diff3 <- y.bikes - z.bikes
sprintf("The Difference in proportions for Method 3 is %.5s",mean(diff3))

#PART D: Histograms

method.1 <- ggplot() +aes(x=diff1) +
  geom_histogram(color="black", fill="blue")+
  geom_vline(xintercept = mean(diff1),size=2)+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        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(mu[y]," - ", mu[z]))

method.2 <- ggplot() +aes(x=diff2) +
  geom_histogram(color="black", fill="red")+
  geom_vline(xintercept = mean(diff2),size=2)+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        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(mu[y]," - ", mu[z]))

method.3 <- ggplot() +aes(x=diff3) +
  geom_histogram(color="black", fill="green")+
  geom_vline(xintercept = mean(diff3),size=2)+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        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(mu[y]," - ", mu[z]))

(method.1 | method.2) /
  (plot_spacer() + method.3 + plot_spacer() + plot_layout(widths = c(1,2,1))) +
  plot_annotation(tag_levels = 'A')
```

Problem 5

```r
### PROBLEM 5 (BDA 3rd. Ed. Exercise 3.12)

#Data for the problem at hand
#years correspond to 1976,1977,1978,1979,1980,1981,1982,1983,1984,1985
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)
                   )

#Part b: informative Prior

#number of draws from prior
N <- 1000
#grid for alpha and beta
alpha <- seq(from = 10, to = 70, length.out = N)
betas <-  seq(from = -5, to = 5, length.out = N)

priors <- function(a,b){
  # Calculate density on grid
  prior <- dgamma(a,shape=50,rate=1)*dnorm(b,mean=0,sd = sqrt(0.5))
  return(prior)
}

informative.prior <- outer(alpha,betas,priors)

#part e
'
the estimates from this regression give you the mode of the posterior under
uniform prior
'
fit <- glm(accidents ~ year, data = df,family=poisson(link="identity"))
fit <- lm(accidents ~ year, data = df)

sprintf("\U03B1 = %s, \U03B2 = %s", round(coef(fit)[1],3), round(coef(fit)[2],3))
print("The covariance matrix is: ")
round(vcov(fit),3)

flight.post <- function(a,b,t,y){
    '
    Parameters:
        a - grid space for alpha
        b - grid space for beta
        t - time data
        y - number of fatal accidents
    '
  logl <- function(M,y){
    rate <- ifelse(M> 0, M,0)      #M represents the kernel of the prior
    y*log(rate) - (rate) - lfactorial(y)
  }
  z <- 0 #initialize value for the for loop
  for (j in 1:length(t)) {
    #sums the log likelihoods
```

```
    z <- z + logl(M = outer(a,b,function(x,y,s) x+y*s,s = t[j]) , y = y[j])
  }
return(exp(z) / sum(exp(z)) )  }


system.time(flights <- flight.post(a=alpha,b=betas, t=df[,'year'], y=df[,'accidents']))

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

'
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.
'

samples <- sample(length(flights), size = 1000,replace = T,
                  prob = c(flights) )

#plug sample values in to give us result + random jitter
d.alpha = (alpha[2]-alpha[1])
d.beta = (betas[2]-betas[1])

alphas.post <-  alpha.grid[samples]+runif(1000,min = -d.alpha/2, max = d.alpha/2)
beta.post <- beta.grid[samples]+runif(1000,min = -d.beta/2, max = d.beta/2)

# This is another Posterior plot, but this has the simulated points built on top of it
flight.data <- data.frame(alpha = alpha.grid,
                           beta = beta.grid,
                          prior = c(informative.prior),
                           post = c(flights))

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

# Contour Plot of Flight Accidents Posterior
flight.prior.plot <- ggplot(flight.data, aes(x=alpha, y= beta, z=prior))+
  geom_contour(color="black",size = 1.4)+ #contour levels
  coord_cartesian(xlim = c(30,70), ylim = c(-2,2)) +
  scale_fill_gradient(low = 'yellow', high = 'red', guide = "none") +
  scale_alpha(range = c(0, 1), guide = "none")+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        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))

flight.prior.plot
```

```r
# Contour Plot of Flight Accidents Posterior
flight.post.plot <- ggplot(flight.data, aes(x=alpha, y= beta, z=post))+
  stat_contour(breaks= flight.cont,color="black",size = 1.4)+ #contour levels
  coord_cartesian(xlim = c(10,50), ylim = c(-5, 5)) +
  scale_fill_gradient(low = 'yellow', high = 'red', guide = "none") +
  scale_alpha(range = c(0, 1), guide = "none")+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        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))
flight.post.plot

(flight.prior.plot | flight.post.plot)  + plot_annotation(tag_levels = 'A')
#Now we plot our histogram
ggplot() +aes(x=alphas.post+beta.post*11) +
  geom_histogram(color="black", fill="lightgreen")+
  theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(),
        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(alpha," + ", "1986",beta))

#95%,- confidence interval
pos <- quantile(rpois(10000,alphas.post + beta.post*11), c(0.025,0.975))
sprintf("The Credible Interval for Fatal Accidents in 1986 is [%s]",
        paste0(pos, collapse = ', '))

### LINE CODE AS SEEN IN SOLN MANUAL
x <- -1000:1000

#defines piecewise function for shading
fx <- (x > 0) * (-x/10) + (x <= 0) * (-x)
df <- data.frame(x = x,
                 y1 = -x,
                 y2= -x/2,
                 y3=-x/3,
                 y10= -x/10,
                 piece = fx)

ggplot(df, aes(x=x)) +
  geom_line(aes(y = y1,color="green"),size=1.7) +
  geom_line(aes(y = y2,color="steelblue"), size=1.7) +
```

```r
geom_line(aes(y = y3,color="red"), size=1.7) +
geom_line(aes(y = y10,color="purple"), size=1.7) +
theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
      panel.background = element_blank(),
      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.spacing.x = unit(2.0, 'cm'))+
geom_vline(xintercept = 0,size=2)+
geom_hline(yintercept=0,size=2)+
geom_text(x=500,y=500,label="\u03A9",size=40,color="black")+
geom_ribbon(aes(ymin = piece,ymax=1000), fill = "lightgrey", alpha = .5)+
scale_x_continuous(limits = c(-1000, 1000))+
scale_y_continuous(limits = c(-1000, 1000))+
scale_color_manual(name="Equation", values=c("green"="green",
                                             "steelblue"="steelblue",
                                             "red"="red","purple"="purple"),
                   labels=expression(alpha + beta*'t'[1],
                                     alpha + beta*'t'[2],
                                     alpha + beta*'t'[3],
                                     alpha + beta*'t'[N]),
                   guide = guide_legend(override.aes = list(
                     shape = rep(16,4) ) ) )+
ylab(~ paste(beta))+
xlab(~paste(alpha))
```