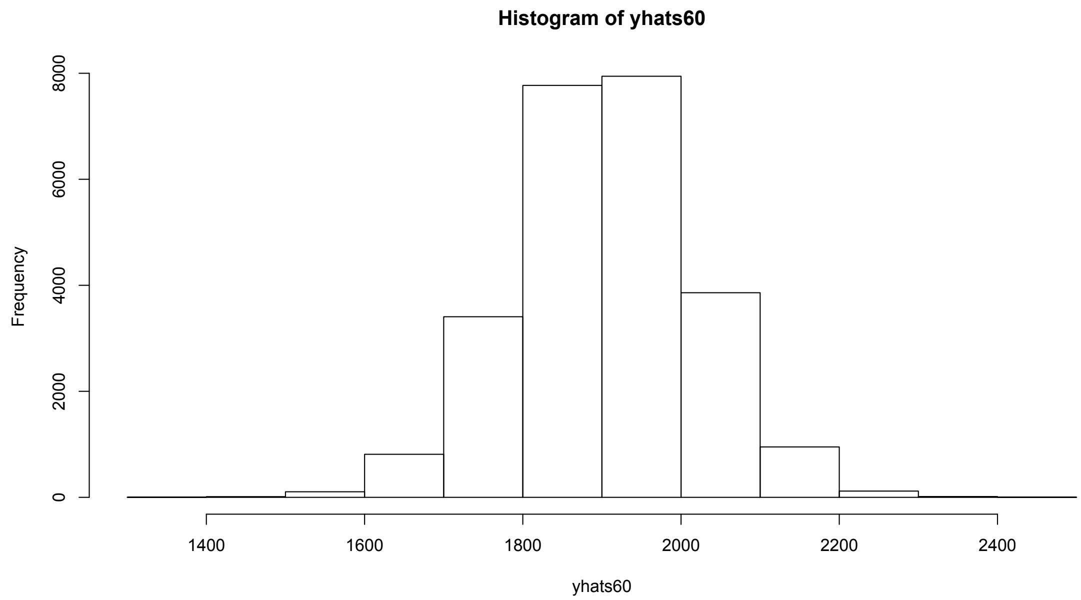
COMMENTS FROM GRADED EXAM: The 95% credible interval for the posterior predictive distribution of IgG measurements for the next person who enters with uptake x=60 was too narrow. The error was that I took the credible interval of the N regression means instead drawing N random normal values with each mean as a center and each corresponding sigsqs value as the variance.



Mean: 1904.294
95% Credible Interval: (1678.518, 2128.151)

```
#Read file and attach
data = read.csv(file.choose(), header=TRUE)
attach(data)

#Set gibbs parameters

y = data$IgG
x = data$Max.O2.Uptake

output = summary(glm(y ~ x + I(x^2)))

N = 25000
lag = 500
burnin = 0
b0 = output$coef[1,1]
b1 = output$coef[2,1]
b2 = output$coef[3,1]
```

```
#Gibbs function receives eight parameters: "x" and "y" data values; "b0", "b1", and
"b2" prior regression coefficients, "N" for number of realizations, "lag" for
determining how many realizations to skip between saves, and "burnin" for
determining how many realizations to skip before starting to save. Gibbs generates
N independent random normal values for each regreesion coefficient and N
independent inverse gamma values for the regression variance.
gibbs <- function(x,y,b0,b1,b2,N,lag,burnin) {

        #obtain length of data
        n = length(x)
        #Set N to be N*lag+burnin
        N <- N*lag + burnin

        #Initialize vectors to hold the beta coefficients and sigsq realizations
        b0s = NULL
        b1s = NULL
        b2s = NULL
        s2s = NULL

        for(i in 1:N) {

                #Generate a sigsq, s2, based on current regression coefficients b0, b1,
b2
                s2 = 1/rgamma(1, n/2, sum((y-b0-b1*x-b2*x^2)^2)/2)
                b0 = rnorm(1, sum(y-b1*x-b2*x^2)/n, sqrt(s2/n))
                b1 = rnorm(1, sum(x*(y-b0-b2*x^2))/sum(x^2), sqrt(s2/sum(x^2)))
                b2 = rnorm(1, sum(x^2*(y-b0-b1*x))/sum(x^4), sqrt(s2/sum(x^4)))


                #if i is greater than burnin and if i is a multiple of the lag, store alpha,
beta, and siqsq
  if(i > burnin) {
         if(i %% lag == 0) {
                b0s <- c(b0s,b0)
                b1s <- c(b1s,b1)
                b2s <- c(b2s,b2)
                s2s <- c(s2s,s2)
   }
  }
        }

        vectors <- list("beta0s" = b0s, "beta1s" = b1s, "beta2s" = b2s, "sigsqs" = s2s)
        return(vectors)

}
```

```
#ii USE GIBBS TO SAMPLE FROM THE MARGINAL POSTERIOR OF EACH OF THE
FOUR PARAMETERS
v = gibbs(x,y,b0,b1,b2,N,lag,burnin)

beta0s = v$beta0s
beta1s = v$beta1s
beta2s = v$beta2s
sigsqs = v$sigsqs

mean(beta0s)
#[1] -1452.651
mean(beta1s)
#[1] 87.85156
mean(beta2s)
#[1] -0.5320706
mean(sigsqs)
#[1] 12219.16

#Plot each parameter with ts.plot, hist, and acf

par(mfrow=c(3,1)) #split plotting window into 3 rows and 1 column
ts.plot(beta0s,xlab="Iterations")
hist(beta0s,probability=T, cex.lab=1.5, cex.axis=1.5)
acf(beta0s,lag.max=500)

par(mfrow=c(3,1)) #split plotting window into 3 rows and 1 column
ts.plot(beta1s,xlab="Iterations")
hist(beta1s,probability=T, cex.lab=1.5, cex.axis=1.5)
acf(beta1s,lag.max=500)

par(mfrow=c(3,1)) #split plotting window into 3 rows and 1 column
ts.plot(beta2s,xlab="Iterations")
hist(beta2s,probability=T, cex.lab=1.5, cex.axis=1.5)
acf(beta2s,lag.max=500)

par(mfrow=c(3,1)) #split plotting window into 3 rows and 1 column
ts.plot(sigsqs,xlab="Iterations")
hist(sigsqs,probability=T, cex.lab=1.5, cex.axis=1.5)
acf(sigsqs,lag.max=500)

#Create a matrix of regression coefficients - N rows and 3 columns - using the
posterior distribution of each beta (USE THE POSTERIOR TO SET THE CURVES)
k=1
mcoef <- matrix(, nrow = N, ncol = 3)
for(i in 1:N) {
```

```
        coord = c(beta0s[i], beta1s[i], beta2s[i])
        mcoef[k ,] = coord
        k = k + 1
}
```

#iii CREATE N CURVES AND SELECT THREE AT RANDOM TO SUPERIMPOSE THREE QUADRATIC CURVES

```
#Create a matrix of yhat values - N rows and 30 columns (30 x values) using the matrix of regression coefficients
yhats <- matrix(, nrow = N, ncol = length(x))
for(i in 1:N) {
    yhat <- mcoef[i,1] + mcoef[i,2]*x + mcoef[i,3]*x^2
    yhats[i ,] = yhat
}

#Select 3 random numbers between 0 and 25000 to represent the indices of the 3 randomly selected coefficient coordinates from the mcoef matrix
rLines = sample(1:25000, 3, replace=F)

#Create a 3-row matrix of yhat values for each x using the randomly selected coordinates
yhats3 <- matrix(, nrow = 3, ncol = length(x))
for(i in 1:3) {
    yhat <- mcoef[rLines[i],1] + mcoef[rLines[i],2]*x + mcoef[rLines[i],3]*x^2
    yhats3[i ,] = yhat
}
```

#iv SUPERIMPOSE POINTWISE MEAN REGRESSION CURVE

```
#Calculate the means of the columns (2) of the yhats matrix
means = apply(yhats, 2, mean)

#Plot x and y with possible fitted curves after spline smoothing
smooth1 = smooth.spline(x,yhats3[1,], spar=0.35)
smooth2 = smooth.spline(x,yhats3[2,], spar=0.35)
smooth3 = smooth.spline(x,yhats3[3,], spar=0.35)
smooth4 = smooth.spline(x, means, spar=0.35)

plot(x,y)
lines(smooth1, lty=2)
lines(smooth2, lty=2)
lines(smooth3, lty=2)
lines(smooth4, lwd=2.5)
```

#DRAW N REALIZATIONS FROM THE POSTERIOR PREDICTIVE DISTRIBUTION OF IgG MEASUREMENTS FOR THE NEXT PERSON WHO ENTERS WITH UPTAKE x=60

```
#Using the matrix of regression coefficients - mcoef - create a vector of mean yhat values with O2 uptake = 60 - each yhat value is the mean of a normal distribution (USE THE CURVE TO SET THE MEANS FOR EACH CURVE)
ymeans60 = NULL
for(i in 1:N) {
	yhat60 <- mcoef[i,1] + mcoef[i,2]*60 + mcoef[i,3]*60^2
	ymeans60 = c(ymeans60, yhat60)
}

#Using each mean yhat value as the center of a normal distribution, draw draw from each one random normal value N realizations - one for each curve (USE EACH MEAN TO GENERATE A SINGLE REALIZATION YIELDING N TOTAL REALIZATIONS)
yhats60 = NULL
for(i in 1:N) {
	yhats60 <- c(yhats60, rnorm(1, ymeans60[i], sqrt(sigsqs[i])))
}

hist(yhats60)

mean(yhats60)
#[1] 1904.294
quantile(yhats60, 0.025)
#[1] 1678.518
quantile(yhats60, 0.975)
#[1] 2128.151



detach(data)
```