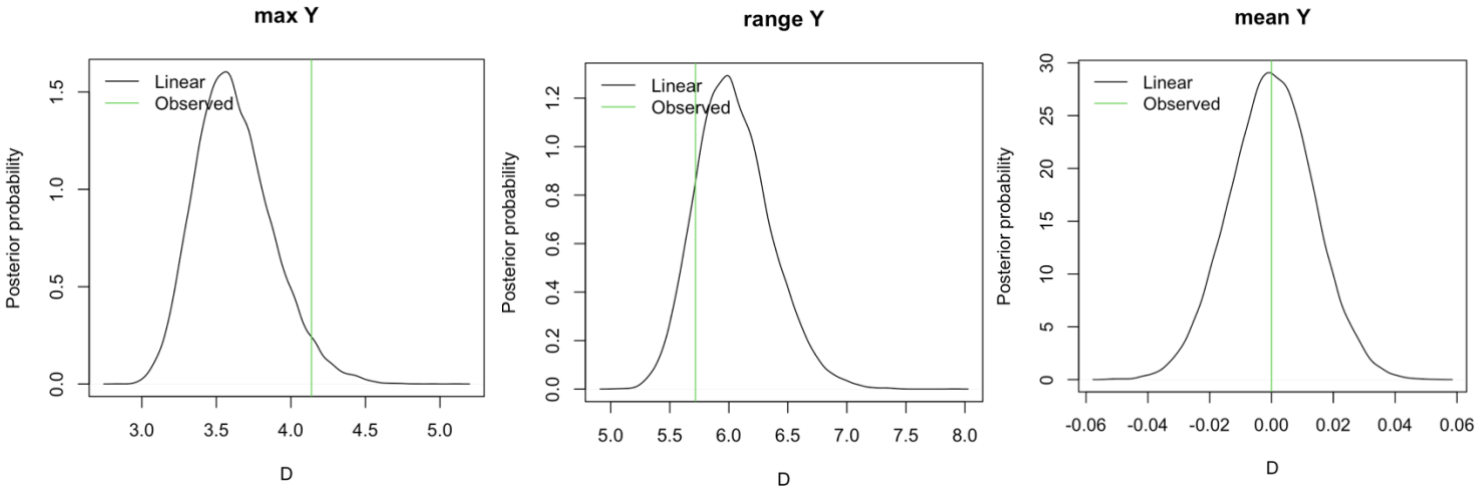
1. **Model Description:** The model I chose to fit is a Bayesian multiple linear regression model. The likelihood of this model is $Y_i \sim \text{Normal}(\beta_0 + \sum_{k=1}^{p} X_{ik}\,\beta_k\,, \tau)$ and the uninformed priors are β_k, $\beta_0 \sim \text{Normal}(0, 100)$ and $\tau \sim \text{Gamma}(0.1, 0.1)$. The predictor variables in this model are every variable in the original dataset (except for basin and VMAX) and "atlantic", a predictor variable I created based on the value of the basin variable. If basin is equal to "atlantic", the atlantic predictor variable is set to 1 and 0 otherwise. The response variable is VMAX. I used 5000 burn-in samples and 2 MCMC chains with 10000 iterations each. This is a reasonable approach because I noticed that many of the predictor variables have a linear relationship with VMAX. Additionally, VMAX roughly follows a normal distribution which is what a response variable follows in multiple linear regression.

2. **Model Comparisons:** I fit two Bayesian multiple linear regression models. They had the same likelihood and priors as described in the "Model Description" section, but the predictor variables that were included in the model fitting varied. The predictors in model 1 were the variables that seemed to have strong linear relationships with VMAX based on scatter plot visualizations. The predictors in model 2 included all variables in the original dataset (except for basin and VMAX) and atlantic. Models 1 and 2 converged according to the trace plots and Geweke diagnostics. The table below summarizes the included predictors and DIC values for models 1 and 2.

Model	Included Predictors	DIC
1	MINSLP, RHLO, CAPE1, CAPE3, TCOND7002, INST2, TCONDSYM2, COUPLSYM3, HWFI, HWRF	1803
2	StormID, Date, LAT, LON, MINSLP, SHR_MAG, STM_SPD, SST, RHLO, CAPE1, CAPE3, SHTFL2, TCOND7002, INST2, CP1, TCONDSYM2, COUPLSYM3, HWFI, VMAX_OP_T0, HWRF, atlantic	1752

I chose DIC as the metric to compare these two models since both models have the same likelihood. The DIC values have a difference greater than 10 which indicates that there is a substantial difference between the two values. I selected the model with the smallest DIC value which was model 2.

3. **Goodness of Fit:** I used posterior predictive checks to verify that model 2 fits the data well. The posterior probability distributions for maximum, range, and mean of VMAX are shown below. The p-values for these statistics are 0.0374, 0.855, and 0.503 for maximum, range, and mean, respectively.



The distributions for range and mean include the observed range and mean values from the data. Also, the p-values for the range and mean are not very close to 0 or 1. Although the p-value for the maximum is close to 0, the distribution of the maximum includes the observed maximum value from the data. Thus, the model mostly fits the data.

4. **Variable Importance:** All predictors in model 2 have an inclusion posterior probability of 1, so they are all important. The table below shows a summary of the predictors' effects. The β_k Mean column is the mean of the β_k MCMC samples.

Predictor	**β_k Mean**	**Predictor**	**β_k Mean**
StormID	0.018	SHTFL2	0.013
Date	-0.022	TCOND7002	-0.056
LAT	-0.067	INST2	0.071
LON	0.012	CP1	-0.0063
MINSLP	-0.10	TCONDSYM2	0.044
SHR_MAG	-0.057	COUPLSYM3	0.10
STM_SPD	-0.049	HWFI	0.50
SST	-0.027	VMAX_OP_T0	-0.039
RHLO	0.048	HWRF	0.35
CAPE1	-0.015	atlantic	-0.012
CAPE3	0.095		

Overall, the effects of the predictors are relatively small. The predictor with the largest absolute magnitude effect size is HWFI and the predictor with the smallest absolute magnitude effect size is CP1.

5. **Prediction:** Using 5-fold cross-validation, the mean absolute error for model 2 is 8.863. The mean absolute error for the HWRF model is 10.006 which means that model 2 performs slightly better than the HWRF model. The coverage of 95% intervals for model 2 is 0.937.

Code

```
#Midterm 2 Code

#libraries
library(tidyverse)
library(rjags)
library(dplyr)
library(splines)
library(knitr)

#read in dataset
data <- read_delim('/Users/adarekar/Documents/College/Senior/ST
440/Midterm 2/E2_data.csv', delim = ',')

#getting rid of lead_time since it is always equal to 24
data <- select(data, -c("lead_time"))

#splitting dataset based on if vmax is missing or not
vmax_miss <- filter(data, is.na(data$VMAX))
vmax_no_miss <- filter(data, !is.na(data$VMAX))

#converting basin to numeric
atlantic <- c()
for (i in 1:nrow(vmax_no_miss)) {

  if (vmax_no_miss$basin[i] == 'atlantic') {
    atlantic <- append(atlantic, 1)
  }
  else {
    atlantic <- append(atlantic, 0)
  }

}

vmax_no_miss$atlantic = atlantic

#visualizations
plot(vmax_no_miss$Date, vmax_no_miss$VMAX)
plot(vmax_no_miss$LAT, vmax_no_miss$VMAX)
plot(vmax_no_miss$LON, vmax_no_miss$VMAX)
plot(vmax_no_miss$MINSLP,vmax_no_miss$VMAX, xlab = 'MINSLP', ylab =
"VMAX")
plot(vmax_no_miss$SHR_MAG,vmax_no_miss$VMAX)
plot(vmax_no_miss$STM_SPD,vmax_no_miss$VMAX)
plot(vmax_no_miss$SST,vmax_no_miss$VMAX)
```

```
plot(vmax_no_miss$RHLO,vmax_no_miss$VMAX)
plot(vmax_no_miss$CAPE1,vmax_no_miss$VMAX)
plot(vmax_no_miss$CAPE3,vmax_no_miss$VMAX)
plot(vmax_no_miss$SHTFL2,vmax_no_miss$VMAX)
plot(vmax_no_miss$TCOND7002,vmax_no_miss$VMAX, xlab = "TCOND7002",
ylab = "VMAX")
plot(vmax_no_miss$INST2,vmax_no_miss$VMAX)
plot(vmax_no_miss$CP1,vmax_no_miss$VMAX)
plot(vmax_no_miss$TCONDSYM2,vmax_no_miss$VMAX)
plot(vmax_no_miss$COUPLSYM3,vmax_no_miss$VMAX)
plot(vmax_no_miss$HWFI,vmax_no_miss$VMAX)
plot(vmax_no_miss$VMAX_OP_T0,vmax_no_miss$VMAX)
plot(vmax_no_miss$HWRF,vmax_no_miss$VMAX)
plot(vmax_no_miss$atlantic,vmax_no_miss$VMAX)

plot(density(vmax_no_miss$VMAX))


#trying multiple linear regression since a lot of the variables seem
to have a linear relationship with VMAX
X1 <- cbind(vmax_no_miss$MINSLP, vmax_no_miss$RHLO,
vmax_no_miss$CAPE1, vmax_no_miss$CAPE3,
           vmax_no_miss$TCOND7002,
           vmax_no_miss$INST2, vmax_no_miss$TCONDSYM2,
vmax_no_miss$COUPLSYM3, vmax_no_miss$HWFI,
           vmax_no_miss$VMAX_OP_T0, vmax_no_miss$HWRF)

Y1 <- (vmax_no_miss$VMAX -
mean(vmax_no_miss$VMAX))/sd(vmax_no_miss$VMAX) #scaling
X1 <- as.matrix(scale(X1))
n1 <- length(Y1)
p1 <- ncol(X1)

data1 <- list(Y = Y1, X = X1, n = n1, p = p1)

#defining model
model_string1 <- textConnection("model{
   # Likelihood
    for(i in 1:n){
      Y[i] ~ dnorm(beta0+inprod(X[i,],beta[]),taue)
    }
   # Priors
    for(j in 1:p){
      beta[j] ~ dnorm(0,0.01)
    }
    beta0 ~ dnorm(0,0.01)
```

```
    taue  ~ dgamma(0.1, 0.1)

    # Predictive checks

    for(i in 1:n){
    Y2[i] ~ dnorm(beta0+inprod(X[i,],beta[]),taue)

  }


    D[1] <- max(Y2[])
    D[2] <- max(Y2[])-min(Y2[])
    D[3] <- mean(Y2[])



 }")

model1 <- jags.model(model_string1,data = data1, n.chains = 2,
quiet=TRUE)
update(model1, 10000, progress.bar="none")
samples1 <- coda.samples(model1, variable.names=c("beta0", "beta"),
n.iter=20000, progress.bar="none")

par(mar=c(5, 5, 2, 2))
plot(samples1) #trace plots indicate convergence

#checking for convergence
range(autocorr(samples1[[1]],lag=1)) #autocorrelation exists
range(effectiveSize(samples1)) #effective sample sizes are at around
1000 or above so indicates convergence
geweke.diag(samples1[[1]]) #all are less than absolute value of 2 so
this suggests convergence


#compute DIC
DIC1 <- dic.samples(model1,n.iter=10000,progress.bar="none")
print(DIC1)
#Mean deviance:  1790
#penalty 13.12
#Penalized deviance: 1803

#posterior predictive checks
#defining model
model_string1 <- textConnection("model{
   # Likelihood
```

```
    for(i in 1:n){
      Y[i] ~ dnorm(beta0+inprod(X[i,],beta[]),taue)
    }
   # Priors
    for(j in 1:p){
      beta[j] ~ dnorm(0,0.01)
    }
    beta0 ~ dnorm(0,0.01)
    taue  ~ dgamma(0.1, 0.1)

    # Predictive checks

    for(i in 1:n){
    Y2[i] ~ dnorm(beta0+inprod(X[i,],beta[]),taue)

  }


    D[1] <- max(Y2[])
    D[2] <- max(Y2[])-min(Y2[])
    D[3] <- mean(Y2[])



 }")

model1 <- jags.model(model_string1,data = data1, n.chains = 2,
quiet=TRUE)
update(model1, 10000, progress.bar="none")
samples1 <- coda.samples(model1, variable.names=c("D", "Y2", "beta0",
"beta"), n.iter=20000, progress.bar="none")

ds1 <- rbind(samples1[[1]], samples1[[2]])
d01 <- c(max(Y1), max(Y1)-min(Y1), mean(Y1))

dnames <- c("max Y", "range Y", "mean Y")

pval1 <- rep(0,3)
names(pval1)<-dnames


for(j in 1:3){
  plot(density(ds1[,j]),
       xlab="D",ylab="Posterior probability",
       main=dnames[j])
  abline(v=d01[j],col=3)
```

```
  legend("topleft",c("Linear", "Observed"),lty=1,col=c(1, 3),bty="n")

  pval1[j] <- mean(ds1[,j]>d01[j])

}

print(pval1)
#max Y  range Y   mean Y
#0.047425 0.878425 0.501775


#second model with more covariates
X2 <- select(vmax_no_miss, -c("basin", "VMAX"))
X2$Date <- as.numeric(X2$Date)
Y2 <- (vmax_no_miss$VMAX -
mean(vmax_no_miss$VMAX))/sd(vmax_no_miss$VMAX) #scaling
X2 <- as.matrix(scale(X2))
n2 <- length(Y2)
p2 <- ncol(X2)

data2 <- list(Y = Y2, X = X2, n = n2, p = p2)

#defining model
model_string2 <- textConnection("model{
   # Likelihood
    for(i in 1:n){
      Y[i] ~ dnorm(beta0+inprod(X[i,],beta[]),taue)
    }
   # Priors
    for(j in 1:p){
      beta[j] ~ dnorm(0,0.01)
    }
    beta0 ~ dnorm(0,0.01)
    taue  ~ dgamma(0.1, 0.1)

    # Predictive checks

    for(i in 1:n){
    Y2[i] ~ dnorm(beta0+inprod(X[i,],beta[]),taue)

  }


    D[1] <- max(Y2[])
    D[2] <- max(Y2[])-min(Y2[])
    D[3] <- mean(Y2[])
```

```
 }")

model2 <- jags.model(model_string2,data = data2, n.chains = 2,
quiet=TRUE)
update(model2, 5000, progress.bar="none")
samples2 <- coda.samples(model2, variable.names=c("beta0", "beta"),
n.iter=10000, progress.bar="none")

#effects of predictors
beta_mean <- colMeans(rbind(samples2[[1]], samples2[[2]]))
print(format(beta_mean, scientific = FALSE))


par(mar=c(5, 5, 2, 2))
plot(samples2) #trace plots indicate convergence

#checking for convergence
range(autocorr(samples2[[1]],lag=1)) #autocorrelation exists
range(effectiveSize(samples2)) #some parameters have effective sample
size less than 1000
geweke.diag(samples2[[1]]) #all less than absolute value of 2,
convergence indicated


#compute DIC
DIC2 <- dic.samples(model2,n.iter=10000,progress.bar="none")
print(DIC2)
#Mean deviance:  1729
#penalty 23.08
#Penalized deviance: 1752

#posterior predictive checks
#defining model
model_string2 <- textConnection("model{
   # Likelihood
    for(i in 1:n){
      Y[i] ~ dnorm(beta0+inprod(X[i,],beta[]),taue)
    }
   # Priors
    for(j in 1:p){
      beta[j] ~ dnorm(0,0.01)
    }
    beta0 ~ dnorm(0,0.01)
```

```
    taue  ~ dgamma(0.1, 0.1)

    # Predictive checks

    for(i in 1:n){
    Y2[i] ~ dnorm(beta0+inprod(X[i,],beta[]),taue)

  }


    D[1] <- max(Y2[])
    D[2] <- max(Y2[])-min(Y2[])
    D[3] <- mean(Y2[])



 }")

model2 <- jags.model(model_string2,data = data2, n.chains = 2,
quiet=TRUE)
update(model2, 5000, progress.bar="none")
samples2 <- coda.samples(model2, variable.names=c("D", "Y2", "beta0",
"beta"), n.iter=10000, progress.bar="none")


ds2 <- rbind(samples2[[1]], samples2[[2]])
d02 <- c(max(Y2), max(Y2)-min(Y2), mean(Y2))

dnames <- c("max Y", "range Y", "mean Y")

pval2 <- rep(0,3)
names(pval2)<-dnames


for(j in 1:3){
  plot(density(ds2[,j]),
       xlab="D",ylab="Posterior probability",
       main=dnames[j])
  abline(v=d02[j],col=3)
  legend("topleft",c("Linear", "Observed"),lty=1,col=c(1, 3),bty="n")

  pval2[j] <- mean(ds2[,j]>d02[j])

}

print(pval2)
```

```
#  max Y range Y  mean Y
#0.03715 0.85110 0.49995

#going with model 2 based on DIC value

#variable importance
beta <- ds2[, 1709:1729]
colnames(beta) <- colnames(X2)
Inc_Prob <- apply(beta!=0,2,mean)
Q <- t(apply(beta,2,quantile,c(0.5,0.05,0.95)))
out <- cbind(Inc_Prob,Q)

kable(round(out,2))
#all are important with inclusion probability of 1

#CV for selected model
set.seed(123)
fold <- rep(1:5,length(Y2)/5)
fold <- sample(fold)

Y2_mean <- matrix(NA, length(Y2), 1)
Y2_low <- matrix(NA, length(Y2), 1)
Y2_high <- matrix(NA, length(Y2), 1)

for (i in 1:5) {
  train_linear <- list(Y = Y2[fold!=i], X = X2[fold!=i,], n =
sum(fold!=i), p = p2)

  #fit linear regression model
  model_string2 <- textConnection("model{
   # Likelihood
    for(i in 1:n){
      Y[i] ~ dnorm(beta0+inprod(X[i,],beta[]),taue)
    }
   # Priors
    for(j in 1:p){
      beta[j] ~ dnorm(0,0.01)
    }
    beta0 ~ dnorm(0,0.01)
    taue  ~ dgamma(0.1, 0.1)
    sigma <- 1/sqrt(taue)



 }")
```

```
  model2 <- jags.model(model_string2,data = train_linear, n.chains =
2, quiet=TRUE)
  update(model2, 5000, progress.bar="none")
  samples2 <- coda.samples(model2, variable.names=c("beta0", "beta",
"sigma"),n.iter=10000, progress.bar="none")

  print('done with fitting linear model')


  #make predictions
  samples <- rbind(samples2[[1]], samples2[[2]])
  beta <- colMeans(samples[,1:21])
  beta0 <- mean(samples[,22])
  sigma <- mean(samples[,23])

  for(j in 1:length(Y2)) {
    if (fold[j] == i) {
      posterior <- rnorm(nrow(samples), beta0 + sum(X2[j, ]*beta),
sigma)
      Y2_mean[j, 1] <- mean(posterior)
      Y2_low[j,1] <- quantile(posterior,0.025)
      Y2_high[j, 1] <- quantile(posterior, 0.975)
    }

  }

  print(i)
  rm(model2)




}

#calculating metrics
#scaling back
Y2_mean_convert <- Y2_mean*sd(vmax_no_miss$VMAX) +
mean(vmax_no_miss$VMAX)
Y2_low_convert <- Y2_low*sd(vmax_no_miss$VMAX) +
mean(vmax_no_miss$VMAX)
Y2_high_convert <- Y2_high*sd(vmax_no_miss$VMAX) +
mean(vmax_no_miss$VMAX)

MAD <- colMeans(abs(Y2_mean_convert-vmax_no_miss$VMAX))
print(MAD)
```

```
#8.863433

COV <- colMeans((Y2_low_convert <= vmax_no_miss$VMAX) &
(vmax_no_miss$VMAX <= Y2_high_convert))
print(COV)
#0.9366569

MAD_model <- mean(abs(vmax_no_miss$HWRF-vmax_no_miss$VMAX))
print(MAD_model)
#10.00559
#my model has slightly lower MAD which means it is performing better

#predictions for missing VMAX
#converting basin to numeric
atlantic <- c()
for (i in 1:nrow(vmax_miss)) {

  if (vmax_miss$basin[i] == 'atlantic') {
    atlantic <- append(atlantic, 1)
  }
  else {
    atlantic <- append(atlantic, 0)
  }

}

vmax_miss$atlantic = atlantic

X2 <- select(vmax_no_miss, -c("basin", "VMAX"))
X2$Date <- as.numeric(X2$Date)
Y2 <- (vmax_no_miss$VMAX -
mean(vmax_no_miss$VMAX))/sd(vmax_no_miss$VMAX) #scaling
X2 <- as.matrix(scale(X2))
n2 <- length(Y2)
p2 <- ncol(X2)

data2 <- list(Y = Y2, X = X2, n = n2, p = p2)

model_string2 <- textConnection("model{
   # Likelihood
    for(i in 1:n){
      Y[i] ~ dnorm(beta0+inprod(X[i,],beta[]),taue)
    }
   # Priors
    for(j in 1:p){
      beta[j] ~ dnorm(0,0.01)
```

```
    }
    beta0 ~ dnorm(0,0.01)
    taue  ~ dgamma(0.1, 0.1)
    sigma <- 1/sqrt(taue)



 }")

model2 <- jags.model(model_string2,data = data2, n.chains = 2,
quiet=TRUE)
update(model2, 5000, progress.bar="none")
samples2 <- coda.samples(model2, variable.names=c("beta0", "beta",
"sigma"), n.iter=10000, progress.bar="none")

samples <- rbind(samples2[[1]], samples2[[2]])
beta <- colMeans(samples[,1:21])
beta0 <- mean(samples[,22])
sigma <- mean(samples[,23])

X2_miss <- select(vmax_miss, -c("basin", "VMAX"))
X2_miss$Date <- as.numeric(X2_miss$Date)
X2_miss <- as.matrix(scale(X2_miss))

Y_mean <- matrix(NA, nrow(vmax_miss), 1)
Y_low <- matrix(NA, nrow(vmax_miss), 1)
Y_high <- matrix(NA, nrow(vmax_miss), 1)

for(j in 1:nrow(vmax_miss)) {

  posterior <- rnorm(nrow(samples), beta0 + sum(X2_miss[j, ]*beta),
sigma)
  Y_mean[j, 1] <- mean(posterior)
  Y_low[j,1] <- quantile(posterior,0.025)
  Y_high[j, 1] <- quantile(posterior, 0.975)


}

Y_mean_convert <- Y_mean*sd(vmax_no_miss$VMAX) +
mean(vmax_no_miss$VMAX)
Y_high_convert <- Y_high*sd(vmax_no_miss$VMAX) +
mean(vmax_no_miss$VMAX)
Y_low_convert <- Y_low*sd(vmax_no_miss$VMAX) + mean(vmax_no_miss$VMAX)
```

```
#save to dataframe and csv file
predict.data <- data.frame(HWRF = vmax_miss$HWRF,
                           VMAX = Y_mean_convert,
                           L = Y_low_convert,
                           U = Y_high_convert)
predict.final <- data
predict.final$L <- rep(NA, nrow(predict.final))
predict.final$U <- rep(NA, nrow(predict.final))

j <- 1
for (i in 1:nrow(predict.final)) {
  if (is.na(predict.final$VMAX[i]) == TRUE) {
    predict.final$VMAX[i] <- predict.data$VMAX[j]
    predict.final$L[i] <- predict.data$L[j]
    predict.final$U[i] <- predict.data$U[j]
    j <- j +1


  }
}

predict.final <- select(predict.final, c("HWRF", "VMAX", "L", "U"))
write.csv(predict.final, '/Users/adarekar/Documents/College/Senior/ST
440/Midterm 2/DarekarAyesha.csv', row.names = FALSE)
```