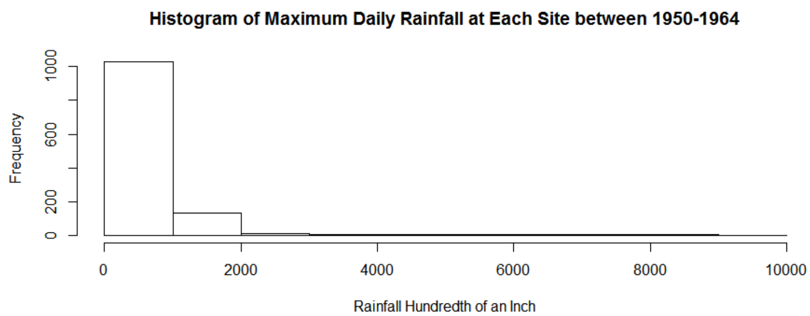
Daniel Briggs
ST 495

Introduction

Extreme rainfall events and the resulting floods often cause a great deal of damage to life and property. Determination of the magnitude of these events have implications in the management of watersheds and design of hydraulic structures. Due to climate change, these extreme precipitation events may be changing. Fortunately, the US Historical Climate Network has recorded daily precipitation measured in hundredths of an inch for 1,218 weather stations across the continental



United States from 1900-2014 allowing for easy study. Figure 1 shows the distribution of the most extreme daily precipitation events between 1950 and 2014.

Figure 1. The maximal 24-hour rainfall at each site between 1950 and 1964.

Data and Methods

To study the change in extreme precipitation events, yearly maximums were considered for the years 1950 to 2014. Y_{it} be the maximum of the daily precipitation values for monitor *i* and year *t*. Data for n = 1, 218 weather stations were considered initially. The maximum rainfall event for each year *t* was recorded for each station *i* and saved. If the number of days in year *t* at station *i* had more than 50 missing observations, then Y_{ij} was set to missing. After discarding all the stations *i* with more than 10 years of missing observations, only 921 stations were considered. A major assumption was that no station had no rainfall for year *t* so any values of 0 were changed to be missing. A total (n = 2) values of 0 were cast to be missing. These new missing values can represent a failure in the system that was not checked or poorly input data. But, due to the sensitivity of rainfall collection systems and the tendency of the systems to fail, it is unreasonable to assume that no rainfall was observed. $X_t = (t - 1950)/10$ was created such that each β_{i2} represents the increase in log mean per decade.

Four models were fit. Each model specified different priors for each beta coefficient. Extreme precipitation events were known to follow the likelihood distribution $Y_{it} \sim \text{Gamma}(a_{it}, b_{it})$ where a_{it} and b_{it} are set so that $\log[E(Y_{it})] = \beta_{i1} + X_t\beta_{i2}$ and $\log[Var(Y_{it})] = \beta_{i3}$. Table 1 shows the distributions for each β_{ij}

and any other additional requirements. An initial 2,000 samples were taken as burn in, and then 500 CODA samples were taken. 1,000 DIC samples were then taken.

	Model 1	Model 2	Model 3	Model 4
Prior Description	Gaussian	Gaussian with Priors on variance	Cauchy Priors	Bayesian LASSO
β_1	Normal(0,10)	Normal($0,\sigma^2_{b1}$)	$t_1(0,\sigma^2_{b1})$	DoubleExpo($0,\sigma^2_{b1}$)
β_2	Normal(0,10)	Normal($0,\sigma^2_{b2}$)	$t_1(0,\sigma^2_{b2})$	DoubleExpo($0,\sigma^2_{b2}$)
β_3	Gamma(0.1,0.1)	Gamma(0.1,0.1)	Gamma(0.1,0.1)	Gamma(0.1,0.1)
σ^2_{b1}		Gamma(0.1,0.1)	Gamma(0.1,0.1)	Gamma(0.1,0.1)
σ^2_{b2}		Gamma(0.1,0.1)	Gamma(0.1,0.1)	Gamma(0.1,0.1)

Table 1. Chart illustrating the distribution of each β_{ij} and hyper parameters σ^2_{bi}.

Convergence and Model Selection

All models were fit in R using JAGS and MCMC sampling with three Markov chains. After discarding 2,000 iterations as burn in samples, and using the next 500 samples for estimating the posterior of the model parameters, each model was checked for convergence. The trace plots showing the generation of the posterior distribution resemble the white noise process illustrated by Figure 2 showing that the all the models converge.

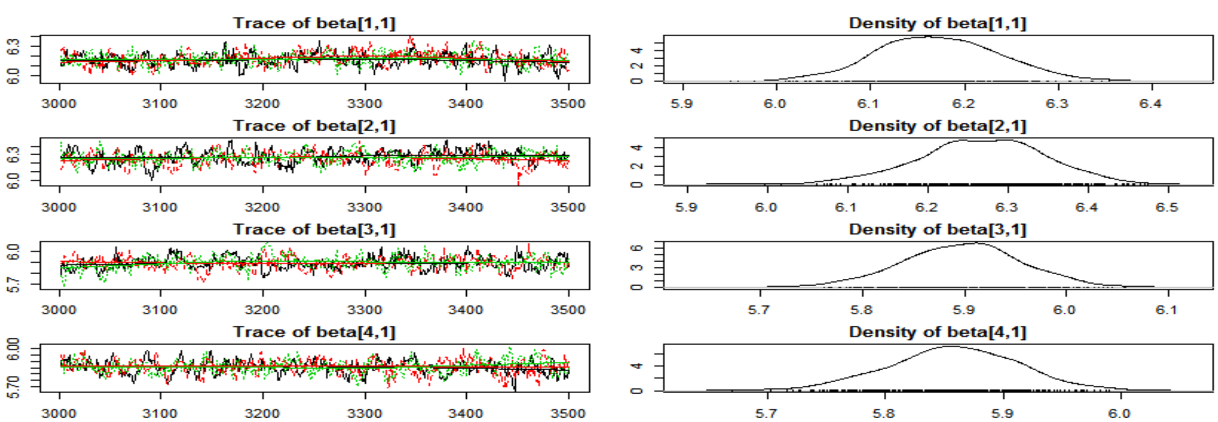


Figure 2. Illustration of the convergence of the Cauchy model with 500 samples.

Because all the models converged, all models were considered. Our model was selected using the deviance information criterion (DIC). Table 2 shows the DIC values for each model. The Bayesian Lasso approach was chosen because the penalized deviance was the smallest.

	Model 1	Model 2	Model 3	Model 4
Model Description	Gaussian	Gaussian with Priors on variance	Cauchy Priors	Bayesian LASSO
Mean Deviance	660207	660015	660024	660175
Penalty	2871	2481	2534	2142
Penalized Deviance	663078	662496	662558	662317

Table 2. DIC for the Four Models

Diagnostics were performed to check if the MCMC algorithm was producing a reliable output for the final model. The Gelman and Rubins shrink factor plots tend to a value of 1 as the number of iterations reaches 3,500 which is additional evidence for convergence of the model. Given that only 500 samples were taken, the effective sample size should not be greater than 500, but the effective sample size obtained a minimum value for a random sample of β_{ij} at 169.1910. This suggests that the samples are not highly correlated. The auto correlation plots were generated to confirm this. The ACF decays after a lag of 5, and is insignificant further supporting the idea that the samples are independent. Thus, it can be assumed that MCMC algorithm produced reliable output for the model parameters.

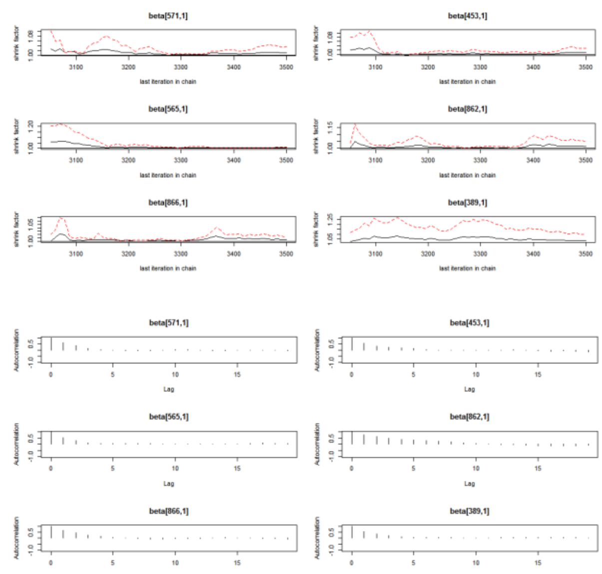


Figure 3. Autocorrelation plots and Gelman and Rubins shrink factor plots for a random sample of β_{ij}.

Results and Predictions

The output from the MCMC sampling algorithm was analyzed to determine the values and uncertainty in the coefficients β_{ij}. The The 2.5% and 97.5% quantile values for the posterior distribution of all the parameters were examined to determine 95% credible sets. The posterior mean was evaluated to determine a point estimate of β_{ij}.

	Mean	2.5% Quant.	97.5% Quant.	95% Credible Set	SE	Time-series SE
$\beta_{571,2}$	-1.049e-03	-0.021799	0.019894	(-0.021799, 0.019894)	0.010108	0.0005599
$\beta_{453,2}$	4.460e-03	-0.014472	0.026328	(-0.014472, 0.026328)	0.009751	0.0004630
$\beta_{565,2}$	-3.417e-03	-0.025796	0.013741	(-0.025796, 0.013741)	0.009501	0.0005071
$\beta_{862,2}$	1.536e-02	-0.003432	0.042354	(-0.003432, 0.042354)	0.011984	0.0009413
$\beta_{389,2}$	1.243e-02	-0.007944	0.040078	(-0.007944, 0.040078)	0.012835	0.0007262

Table 3. Posterior distribution Quantiles and Credible Sets and Standard Errors

All of the credible sets contain zero, suggesting that the slopes are not unique. To further illustrate the validity of the model, the models were plotted for the random sample.

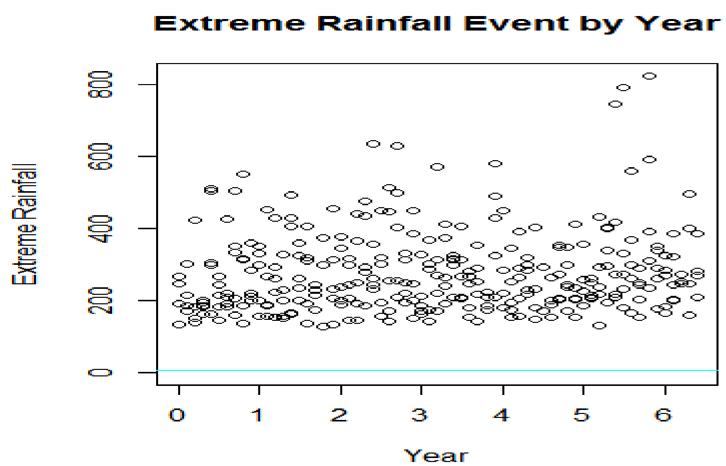


Figure 4. The model does not appear to model the data well.

The model does not seem to model the data well. Some potential solutions to this include increasing the variance on the model, or including another model that will predict the data better. Another solution to modeling extreme rainfall events would be to fit a different model. Fitting a single model with a generalized extreme value distribution, with a large variance would create a single generalizable model that would be able to predict the rainfall everywhere. Another alternative model would be to fit a multiple slopes model between years.

Summary of Results

The change in extreme rainfall events in the continental United States was modeled using a Bayesian approach and an MCMC algorithm. Year was used to model the maximum rainfall event. The final model used a Bayesian LASSO approach to fit the data. The model converges, and converges veryefficiently and neatly, but does not generate accurate results. A new model needs to be developed.

Daniel Briggs
ST 495

Appendix:

```
#refresh
rm(list = ls())

#get data
wd = 'C:/Users/debri/Documents/ST 495 Applied Bayesian Statistics/Final'
setwd(wd)
data = load('USHCNprcpSetup.RData')

#create informative names
siteNames = c()
for(i in 1:1218){
  siteName = paste("site", i, sep = '_')
  siteNames = c(siteNames, siteName)
}

#add years and days to make more informative
year = sort(rep(1900:2014, 365))
PRCP = cbind(year, PRCP)
day = rep(1:365,(dim(PRCP)[1]/365))
PRCP = cbind(day, PRCP)

#implement informative names and subset to years >=1950
colnames(PRCP)[3:dim(PRCP)[2]] = siteNames
PRCP.redox = subset(PRCP, year > 1949)
#additional removal of all NA columns
PRCP.redox = PRCP.redox[, colSums(is.na(PRCP.redox)) != nrow(PRCP.redox)]

#data frame that will be used to create new data frame to solve
PRCP.redox = as.data.frame(PRCP.redox)
print(dim(PRCP.redox)[2] == (length(siteNames) + 2))

#new data frame
cols = length(siteNames)
rows = 2014 - 1950 + 1
Y = as.data.frame(matrix(nrow = cols, ncol = rows))
#Y column names
Ycol = c()
for(i in 1950:2014){
  y = paste('Year', i, sep = '_')
  Ycol = c(Ycol, y)
}
colnames(Y) = Ycol

#populate data frame
for(i in 1950:2014){
```

Daniel Briggs
ST 495

```
  for(j in 3:dim(PRCP.redox)[2]){
    bin = which(PRCP.redox$year == i)
    colindex = j - 2
    rowindex = i - 1949
    if(sum(is.na(PRCP.redox[bin,j])) > 50){
      Y[colindex, rowindex] = NA
    } else {
      Y[colindex, rowindex] = max(PRCP.redox[bin,j], na.rm = T)
    }
  }
}

#check Y
max(Y, na.rm=T)
Y = Y[rowSums(is.na(Y)) <= 10, ]
##any zero value absolutely makes zero sense anywhere in the country
Y[Y == 0] = NA
#only two zero values anyhow
dim(Y)
sum(is.na(Y))

#wrapper function for X
year.scale = function(t = t){
  X = (t - 1950)/10
}
#so we make X
X = year.scale(1950:2014)


#time for some Bayes
nt = length(X)
ns = dim(Y)[1]
library(rjags)
model.1 = "model{

  # Likelihood

  for(i in 1:ns){for(t in 1:nt){
     Y[i,t] ~ dgamma(a[i,t],b[i,t])
     a[i,t] <- pow(mu[i,t],2)/v[i,t]
     b[i,t] <- mu[i,t]/v[i,t]
     log(v[i,t]) <- beta[i,3]
     log(mu[i,t]) <- beta[i,1] + X[t] * beta[i,2]
  }}


  #Priors for betas
  for(i in 1:ns){
```

Daniel Briggs
ST 495

```
    beta[i,1] ~ dnorm(0, 0.1)
    beta[i,2] ~ dnorm(0, 0.1)
    beta[i,3] ~ dgamma(0.1, 0.1)
  }

}"
#data
dat = list(Y = Y, X = X, ns = ns, nt = nt)

#model 1 Super simple
model1 = jags.model(textConnection(model.1), data = dat, n.chains = 3)
update(model1, 2000)
codasamp1 = coda.samples(model1, 500, variable.names = c('beta'))
dicsamp1 = dic.samples(model1, 1000)

#plot traces
par(mar = rep(2, 4))
#these traces look good

#just check it
gelman.plot(codasamp1)
autocorr.plot(codasamp1)
effectiveSize(codasamp1)
plot(codasamp1)
summary(codasamp1)
dicsamp1



#fit another model Hyper Parameters
model.2 = "model{

# Likelihood

for(i in 1:ns){for(t in 1:nt){
Y[i,t] ~ dgamma(a[i,t],b[i,t])
a[i,t] <- pow(mu[i,t],2)/v[i,t]
b[i,t] <- mu[i,t]/v[i,t]
log(v[i,t]) <- beta[i,3]
log(mu[i,t]) <- beta[i,1] + X[t] * beta[i,2]
}}


#Priors for betas
for(i in 1:ns){
beta[i,1] ~ dnorm(0, invar.b1)
beta[i,2] ~ dnorm(0, invar.b2)
beta[i,3] ~ dgamma(0.1, 0.1)
```

```
}

#priors for inverse variance of Beta[,1] and Beta[,2]
invar.b1 ~ dgamma(0.1, 0.1)
invar.b2 ~ dgamma(0.1, 0.1)

}"

model2 = jags.model(textConnection(model.2), data = dat, n.chains = 3)
update(model2, 2000)
codasamp2 = coda.samples(model2, 500, variable.names = c('beta'))
dicsamp2 = dic.samples(model2, 1000)

#plot traces
par(mar = rep(2, 4))
#these traces look good

#just check it
gelman.plot(codasamp2)[,4]
autocorr.plot(codasamp2)
effectiveSize(codasamp2)
plot(codasamp2)[,4]
summary(codasamp2)
dicsamp2

#fit another model Bayesian LASSO
model.3 = "model{

# Likelihood

for(i in 1:ns){for(t in 1:nt){
Y[i,t] ~ dgamma(a[i,t],b[i,t])
a[i,t] <- pow(mu[i,t],2)/v[i,t]
b[i,t] <- mu[i,t]/v[i,t]
log(v[i,t]) <- beta[i,3]
log(mu[i,t]) <- beta[i,1] + X[t] * beta[i,2]
}}


#Priors for betas
for(i in 1:ns){
beta[i,1] ~ ddexp(0, invar.b1)
beta[i,2] ~ ddexp(0, invar.b2)
beta[i,3] ~ dgamma(0.1, 0.1)
}

#priors for inverse variance of Beta[,1] and Beta[,2]
invar.b1 ~ dgamma(0.1, 0.1)
```

Daniel Briggs
ST 495

```
invar.b2 ~ dgamma(0.1, 0.1)

}"

model3 = jags.model(textConnection(model.3), data = dat, n.chains = 3)
update(model3, 2000)
codasamp3 = coda.samples(model3, 500, variable.names = c('beta'))
dicsamp3 = dic.samples(model3, 1000)

#plot traces
par(mar = rep(2, 4))
#these traces look good

#just check it
gelman.plot(codasamp3[,c(randomBetas)])
autocorr.plot(codasamp3[,c(randomBetas)])
effectiveSize(codasamp3[,c(randomBetas)])
plot(codasamp3)
summary(codasamp3[,c(randomBetas)])
dicsamp3


#fit another model Cauchy
model.4 = "model{

# Likelihood

for(i in 1:ns){for(t in 1:nt){
Y[i,t] ~ dgamma(a[i,t],b[i,t])
a[i,t] <- pow(mu[i,t],2)/v[i,t]
b[i,t] <- mu[i,t]/v[i,t]
log(v[i,t]) <- beta[i,3]
log(mu[i,t]) <- beta[i,1] + X[t] * beta[i,2]
}}


#Priors for betas
for(i in 1:ns){
beta[i,1] ~ dt(0, invar.b1,1)
beta[i,2] ~ dt(0, invar.b2,1)
beta[i,3] ~ dgamma(0.1, 0.1)
}

#priors for inverse variance of Beta[,1] and Beta[,2]
invar.b1 ~ dgamma(0.1, 0.1)
invar.b2 ~ dgamma(0.1, 0.1)

}"
```

Daniel Briggs
ST 495

```
model4 = jags.model(textConnection(model.4), data = dat, n.chains = 3)
update(model4, 2000)
codasamp4 = coda.samples(model4, 500, variable.names = c('beta'))
dicsamp4 = dic.samples(model4, 1000)

#plot traces
par(mar = rep(2, 4))
#these traces look good

#just check it
gelman.plot(codasamp4[,1])
autocorr.plot(codasamp4[,1])
effectiveSize(codasamp4)
plot(codasamp4)
summary(codasamp4)
dicsamp4


randomBetas = sample(1:ns, 6)
beta = c(571,453,565,862,389)
B1 = as.vector(summary(codasamp4)[[1]][beta,1])
B2 = c(-1.049e-03, 4.460e-03, -3.417e-03, 1.536e-02, 1.243e-02)
plot(x = NULL, y = NULL, ylim =c(0,max(Y[beta,],na.rm=T)), xlim = c(0,max(X)), xlab = 'Year', ylab =
'Extreme Rainfall', main = 'Extreme Rainfall Event by Year')
for(i in 1:5){
  abline(B1[i], B2[i], col = i)
}
for(j in 1:5){
  for(i in 1:length(X)){
    points(X[i], Y[beta[j],i])
  }
}
maxprcp = c()
for(i in 3:dim(PRCP)[2]){
  max = max(PRCP[,i], na.rm = T)
  maxprcp = c(maxprcp, max)
}

hist(maxprcp, main = "Histogram of Maximum Daily Rainfall at Each Site between 1950-1964", xlab =
"Rainfall Hundredth of an Inch")
```