Damon Frezza  
3/15  
ISyE 6420  
Midterm Report

**Problem 1.**

1. Using BUGS to investigate, we find that the precision measurements for the different mirrors are statistically different. Here we can see that the effects of both mirror 1 and mirror 4 are significantly different from zero.

**mean sd MC\_error 2.5% med 97.5% start sample**

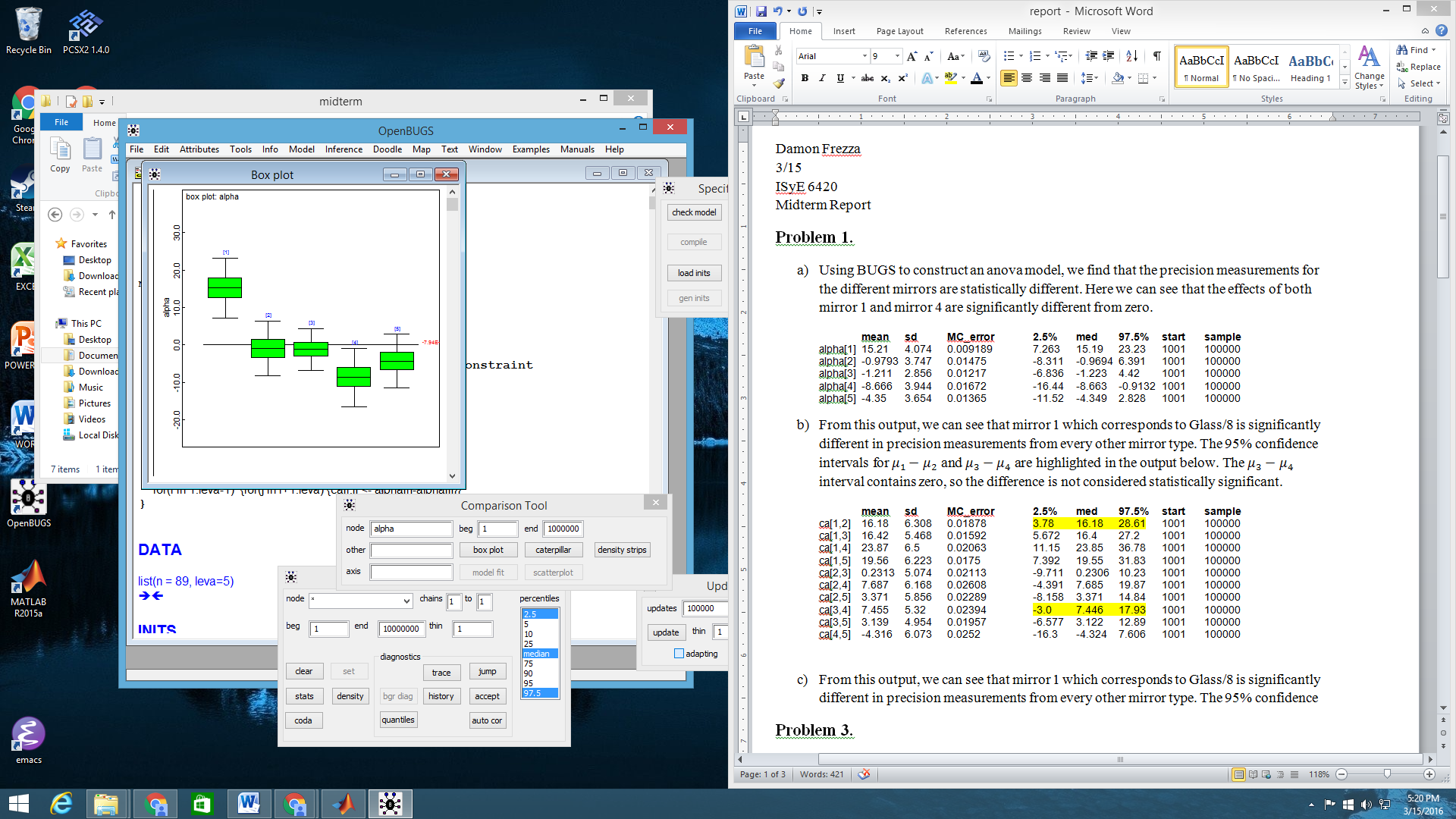
alpha[1] 15.21 4.074 0.009189 7.263 15.19 23.23 1001 100000

alpha[2] -0.9793 3.747 0.01475 -8.311 -0.9694 6.391 1001 100000

alpha[3] -1.211 2.856 0.01217 -6.836 -1.223 4.42 1001 100000

alpha[4] -8.666 3.944 0.01672 -16.44 -8.663 -0.9132 1001 100000

alpha[5] -4.35 3.654 0.01365 -11.52 -4.349 2.828 1001 100000



1. From this output, we can see that mirror 1 which corresponds to Glass/8 is significantly different in precision measurements from every other mirror type. Only difference including mirror 1 are statistically significant. The 95% confidence intervals for and are highlighted in the output below. The interval contains zero, so the difference is not considered statistically significant.

**mean sd MC\_error 2.5% med 97.5% start sample**  ca[1,2] 16.18 6.308 0.01878 3.78 16.18 28.61 1001 100000

ca[1,3] 16.42 5.468 0.01592 5.672 16.4 27.2 1001 100000

ca[1,4] 23.87 6.5 0.02063 11.15 23.85 36.78 1001 100000

ca[1,5] 19.56 6.223 0.0175 7.392 19.55 31.83 1001 100000

ca[2,3] 0.2313 5.074 0.02113 -9.711 0.2306 10.23 1001 100000

ca[2,4] 7.687 6.168 0.02608 -4.391 7.685 19.87 1001 100000

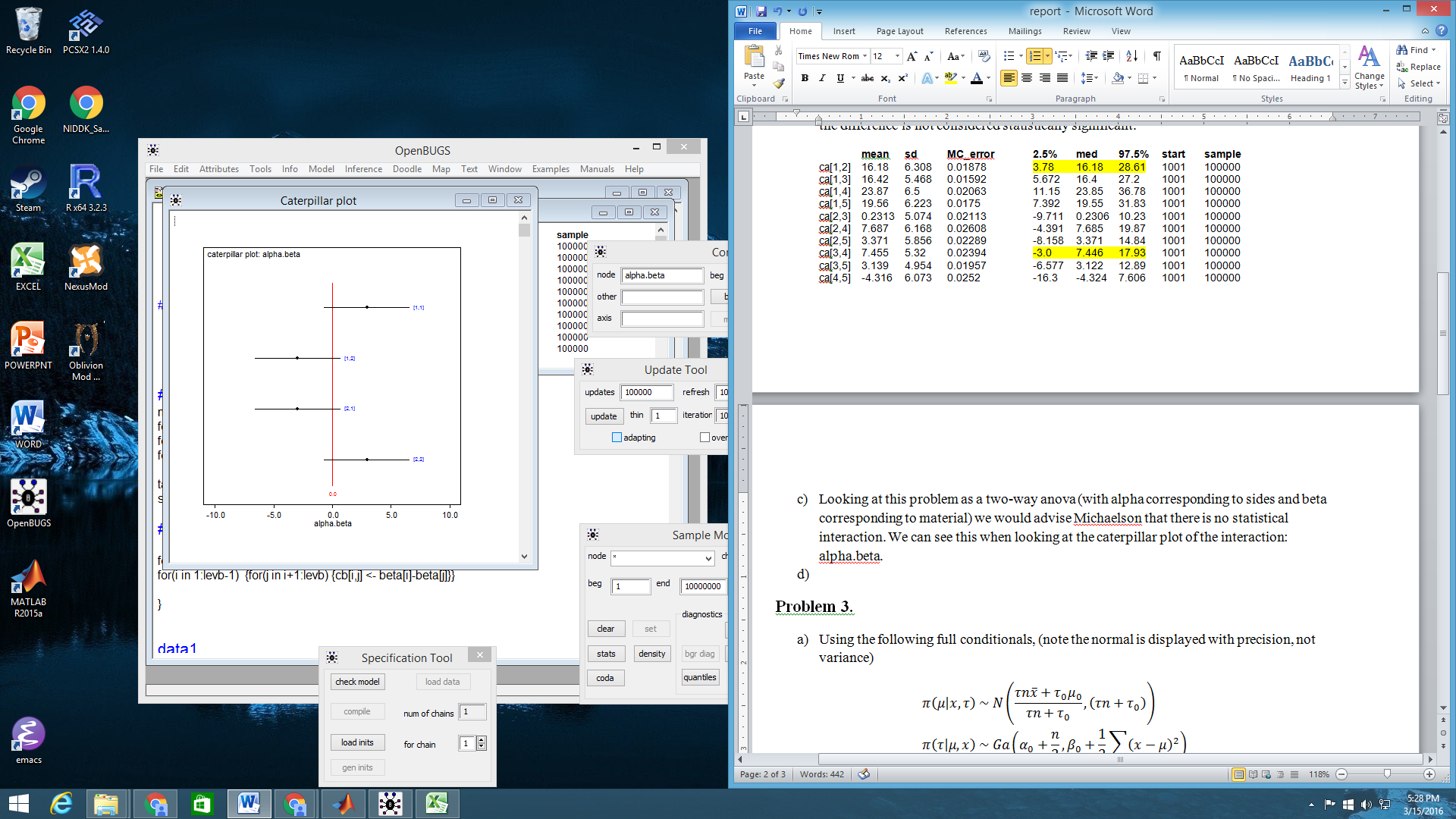
ca[2,5] 3.371 5.856 0.02289 -8.158 3.371 14.84 1001 100000

ca[3,4] 7.455 5.32 0.02394 -3.0 7.446 17.93 1001 100000

ca[3,5] 3.139 4.954 0.01957 -6.577 3.122 12.89 1001 100000

ca[4,5] -4.316 6.073 0.0252 -16.3 -4.324 7.606 1001 100000

1. Looking at this problem as a two-way anova (with alpha corresponding to sides and beta corresponding to material) we would advise Michelson that there is no statistical interaction. We can see this when looking at the caterpillar plot of the interaction, alpha.beta, and noting that each interval line crosses the 0 line.



Additionally, looking at the following output, we can say that both the material of the mirror and number of sides that the mirror has cause significant differences in the precision of the measurements. Glass mirrors and 8 sided mirrors tended to cause higher numbers than steel mirrors and 12 sided mirrors.

**mean sd 2.5% med 97.5% start sample**  alpha[1] 5.122 1.862 1.457 5.12 8.773 1001 100000

alpha[2] -5.122 1.862 -8.773 -5.12 -1.456 1001 100000

alpha.beta[1,1] 2.965 0.005953 -0.6932 2.967 6.626 1001 100000

alpha.beta[1,2] -2.965 0.005953 -6.626 -2.967 0.6935 1001 100000

alpha.beta[2,1] -2.965 0.005953 -6.626 -2.967 0.6935 1001 100000

alpha.beta[2,2] 2.965 0.005953 -0.6932 2.967 6.626 1001 100000

beta[1] 6.818 1.863 3.153 6.822 10.47 1001 100000

beta[2] -6.818 1.863 -10.47 -6.822 -3.153 1001 100000

ca[1,2] 10.24 3.725 2.913 10.24 17.55 1001 100000

cb[1,2] 13.64 3.726 6.306 13.64 20.95 1001 100000

**Problem 3.**

1. Using the following full conditionals, (note the normal is displayed with precision, not variance)

we developed a Gibbs sampling algorithm and found the following estimates for and:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Mean | Standard Dev | 2.5% | Median | 97.5% |
|  | 10.119 | 0.208 | 9.710 | 10.120 | 10.530 |
|  | 0.463 | 0.0893 | 0.3048 | 0.4578 | 0.6541 |

1. Putting an inverse gamma prior on changes the full conditional to be:

Where and as seen in the sigma prior.

Clearly, a difference in the full conditionals will cause a change in the posterior but this is a small change so the output is similar.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Mean | Standard Dev | 2.5% | Median | 97.5% |
|  | 10.115 | 0.1914 | 9.739 | 10.116 | 10.489 |
|  | 0.5507 | 0.099 | 0.3744 | 0.5446 | 0.7631 |

1. Next we develop a Metropolis algorithm for the priors as in part(a) and simulate the same posterior. Using scaled uniform(-1,1) distributions for proposal distributions, we get an acceptance rate of 90% which is efficient. The outcomes are close to those we obtained by Gibbs sampling in part(a).

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Mean | Standard Dev | 2.5% | Median | 97.5% |
|  | 10.112 | 0.2049 | 9.711 | 10.114 | 10.504 |
|  | 0.4687 | 0.0777 | 0.3319 | 0.4659 | 0.6282 |

**Problem 4.**

1. Using multiple linear regression in BUGS, we estimate the missing values of Rostral length and Weight as well as the coefficients of the linear model, and the R-squared value for the model.

**mean sd 2.5% med 97.5% start sample**  Rlength[11] 1.364 0.5281 0.3541 1.324 2.532 1001 100000

Rsquared 0.9455 0.01337 0.9108 0.9488 0.961 1001 100000

Weight[21] 9.319 0.9167 7.492 9.32 11.13 1001 100000

beta[1] -5.975 1.038 -7.979 -5.997 -3.807 1001 100000

beta[2] 0.3585 2.455 -4.745 0.2773 5.461 1001 100000

beta[3] -4.164 2.902 -9.819 -4.206 1.763 1001 100000

beta[4] 3.697 6.475 -8.969 3.683 16.36 1001 100000

beta[5] 7.004 3.72 -0.3467 6.932 14.57 1001 100000

beta[6] 14.99 4.825 5.728 14.91 25.1 1001 100000 weightPred 3.486 1.42 0.6366 3.503 6.28 1001 100000

So the R-squared value is 94% which indicates that the model explains most of the variation in *Weight*, which is nice. The estimated *Rostral length* for the 11th observation is 1.364 inches. The estimated *Weight* for the 21st observation is 9.13 pounds. (It should be noted that 9.13 is the largest *Weight* value and should be used with the understanding that the relationship between *Weight* and the 5 predictors could be less linear towards the extremes.)

1. The weightPred variable in the model is constructed from the linear combination of the *betas* and the prediction vector given in the question. Hence, the prediction is 3.486 pounds and the 95% confidence interval is [0.6366, 6.28].
2. Supposing that one variable needs to be eliminated, we test out the 5 models which contain every combination of the 4 predictors and compare their Ibrahim-Laud criteria. For this problem, we use the estimates from part(a) for the missing data. The model we choose is model 1, which excludes the *Rostral length* variable. We choose this model because both the mean and median values of LI are the smallest among the 5 models.

**mean sd MC\_error 2.5% median 97.5% start sample**

LI[1] 5.525 0.6878 0.01096 4.431 5.438 7.112 1001 100000

LI[2] 5.83 0.7891 0.01602 4.596 5.724 7.684 1001 100000

LI[3] 5.548 0.692 0.009897 4.452 5.461 7.151 1001 100000

LI[4] 6.747 0.9882 0.01942 5.17 6.624 9.012 1001 100000

LI[5] 6.002 0.8246 0.01553 4.711 5.893 7.909 1001 100000

**Code:**

**Problem 1: a and b**

model{

for (i in 1:n){

measure[i] ~ dnorm( mu[i], tau )

mu[i] <- mu0 + alpha[type[i]]

}

#alpha[1] <- 0.0; #CR constraints

alpha[1] <- -sum( alpha[2:leva] ); #STZ Constraint

mu0 ~ dnorm(0, 0.0001)

for (j in 2:leva){

alpha[j] ~ dnorm(0, 0.0001)

}

tau ~ dgamma(0.001, 0.001)

sigma <- sqrt(1/tau)

for(i in 1:leva-1) {for(j in i+1:leva) {ca[i,j] <- alpha[i]-alpha[j]}}

}

DATA

list(n = 89, leva=5)

< 2 columns of data called measures[] and types[] >

INITS

list(mu0=0, alpha = c(NA,0,0,0,0), tau=1)

**Problem 1: c**model{

for(i in 1:n){

measure[i] ~ dnorm( mu[i], tau )

mu[i] <- mu0 + alpha[ sides[i] ] + beta[ material[i] ] + alpha.beta[ sides[i], material[i] ]

}

##STZ (sum-to-zero) constraints

alpha[1] <- - sum(alpha[2:leva])

beta[1] <- - sum(beta[2:levb])

for(a in 1:leva) {alpha.beta[a,1] <- - sum(alpha.beta[a, 2:levb])}

for(b in 2:levb) {alpha.beta[1,b] <- - sum(alpha.beta[2:leva, b])}

#PRIORS

mu0 ~ dnorm(0, 0.0001)

for(a in 2:leva) {alpha[a] ~ dnorm(0, 0.0001)}

for(b in 2:levb) {beta[b] ~ dnorm(0, 0.0001)}

for(a in 2:leva) {for(b in 2:levb){

alpha.beta[a,b] ~ dnorm(0, 0.0001) }}

tau ~ dgamma(0.001, 0.001)

s <- 1/sqrt(tau)

#PAIRWISE COMPARISONS

for(i in 1:leva-1) {for(j in i+1:leva) {ca[i,j] <- alpha[i]-alpha[j]}}

for(i in 1:levb-1) {for(j in i+1:levb) {cb[i,j] <- beta[i]-beta[j]}}

}

data1

list(n = 56, leva= 2, levb= 2)

data2 mirrors: measure sides material

inits

list(mu0=0, alpha=c(NA, 0), beta=c(NA,0),

alpha.beta = structure(.Data=c(NA, NA, NA,0),

.Dim=c(2,2)), tau = 1)

**Problem 3: a and b**

nn = 100000;

mus = [];

taus = [];

x = importdata('mushrooms.dat');

xbar = mean(x);

n = length(x);

mu = 12; % set the parameters as prior means

tau = 0.5;

tau0 = 0.25;

mu0 = 12;

alpha0 = 2; **#switch alpha0 and beta0 for the inverse gamma in part b**

beta0 = 4;

for i = 1 : nn

new\_mu = normrnd((tau\*n\*xbar + tau0\*mu0)/(tau\*n+tau0), sqrt(1/(tau\*n+tau0)) );

par = beta0+ sum((x-mu).^2)/2;

new\_tau =gamrnd(alpha0 + n/2, 1/par); **#augment this first argument by +2 for the inverse gamma in part b**

mus = [mus new\_mu];

taus = [taus new\_tau];

tau=new\_tau;

mu=new\_mu;

end

disp([mean(mus(burnin:nn)),sqrt(var(mus(burnin:nn))),median(mus(burnin:nn))])

disp([mean(taus(burnin:nn)),sqrt(var(taus(burnin:nn))),median(taus(burnin:nn))])

prctile(mus,[2.5,97.5])

prctile(taus,[2.5,97.5])

**Problem 3: c**

%--------------------------------

nn = 50500; % nn=number of metropolis iterations

burn=500; % burn = burnin amount

%---------------------------------------------

mus=[]; %

taus=[];

x = importdata('mushrooms.dat');

xbar = mean(x);

n = length(x);

muOld = 10; % start, theta\_0

tauOld = .5;

r = 0;

for i = 1:nn

muProp = muOld + (rand(1)-0.5)/5; %proposal from scaled uni(-1,1)

tauProp = tauOld + (rand(1)-0.5)/100;

u = rand(1,1);

tmp1 = 1;

tmp2 = 1;

for i2 = 1:n

tmp1 = tmp1 \* normpdf(x(i2), muOld, sqrt(1/(tauOld)));

tmp2 = tmp2 \* normpdf(x(i2), muProp, sqrt(1/(tauProp)));

end

post\_o = tmp1\*normpdf(muOld, 12, 2)\*gampdf(tauOld,2,0.25); %f(X|old theta)\*pi(old theta)

post\_p = tmp2\*normpdf(muProp, 12, 2)\*gampdf(tauProp,2,0.25); %f(X|proposal)\*pi(proposal)

newmu = muOld;

newtau = tauOld;

if u <= min(post\_p/post\_o, 1,'includenan')

r = r+1;

newmu = muProp; %accept proposal as 'new'

newtau = tauProp;

muOld = newmu; % and set 'old' to be the 'new'

tauOld = newtau;

end

taus = [taus, newtau]; %collect all theta's

mus = [mus, newmu];

end

disp(r/nn)

disp([mean(mus(burn:end)),sqrt(var(mus(burn:end))),median(mus(burn:end))])

disp([mean(taus(burn:end)),sqrt(var(taus(burn:end))),median(taus(burn:end))])

disp([prctile(mus(burn:end),2.5),prctile(mus(burn:end),97.5)])

disp([prctile(taus(burn:end),2.5),prctile(taus(burn:end),97.5)])

**Problem 4: a and b**model{

for(i in 1:n){

Weight[i] ~ dnorm(mu[i],tau)

mu[i] <- beta[1] + beta[2]\*Rlength[i] +beta[3]\*Wlength[i] + beta[4]\*Rnotch[i] + beta[5]\*Wnotch[i] + beta[6]\*Width[i]

}

for(j in 1:6){

beta[j] ~ dnorm(0.0, 0.001)

}

for(k in 1:n){

residualvec[k] <- pow(mu[k]-Weight[k],2)

totalsvec[k] <- pow(Weight[k] - ybar,2)

}

ybar <- mean(Weight[])

Rsquared <- 1 - (sum(residualvec[])/sum(totalsvec[]))

tau ~ dgamma(0.001,0.001)

Rlength[11] ~ dnorm(1.5, 2)

weightPred <- beta[1] + beta[2]\*predStar[1] + beta[3]\*predStar[2] + beta[4]\*predStar[3] + beta[5]\*predStar[4] + beta[6]\*predStar[5]

}

Data

list(n=22,predStar=c(1.52, 1.12, 0.622, 0.917, 0.324))

Inits

list(beta=c(0,0,0,0,0,0),tau=1)

**Problem 4: c**model{

for(i in 1:n){

# 5 Competing Models

mu[1,i] <- beta1[1] +beta1[2]\*Wlength[i] + beta1[3]\*Rnotch[i] + beta1[4]\*Wnotch[i] + beta1[5]\*Width[i]

mu[2,i] <- beta2[1] + beta2[2]\*Rlength[i] + beta2[3]\*Rnotch[i] + beta2[4]\*Wnotch[i] + beta2[5]\*Width[i]

mu[3,i] <- beta3[1] + beta3[2]\*Rlength[i] +beta3[3]\*Wlength[i] + beta3[4]\*Wnotch[i] + beta3[5]\*Width[i]

mu[4,i] <- beta4[1] + beta4[2]\*Rlength[i] +beta4[3]\*Wlength[i] + beta4[4]\*Rnotch[i] + beta4[5]\*Wnotch[i]

mu[5,i] <- beta5[1] + beta5[2]\*Rlength[i] +beta5[3]\*Wlength[i] + beta5[4]\*Rnotch[i] + beta5[5]\*Width[i]

}

# LI predictive Criterion. Smaller is better

for(i2 in 1:5){

log(sigma[i2]) <- log.sigma[i2]

log.sigma[i2] ~ dunif(-10,10)

tau[i2] <- 1/pow(sigma[i2], 2)

LI[i2] <- sqrt( sum(D2[i2,]) + pow(sd( y.new[i2,]),2))

# for each different model

for(i3 in 1:n){

y[i2,i3] <- Weight[i3]

y[i2,i3] ~ dnorm(mu[i2,i3],tau[i2])

y.new[i2,i3] ~ dnorm(mu[i2,i3],tau[i2])

D2[i2,i3] <- pow(Weight[i3]-y.new[i2,i3],2)

}

}

for(j in 1:6){

beta[j] ~ dnorm(0.0, 0.001)

beta1[j] ~ dnorm(0.0, 0.001)

beta2[j] ~ dnorm(0.0, 0.001)

beta3[j] ~ dnorm(0.0, 0.001)

beta4[j] ~ dnorm(0.0, 0.001)

beta5[j] ~ dnorm(0.0, 0.001)

}

}

Data

list(n=22)

Inits

list(

beta1=c(0,0,0,0,0,0),

beta2=c(0,0,0,0,0,0),

beta3=c(0,0,0,0,0,0),

beta4=c(0,0,0,0,0,0),

beta5=c(0,0,0,0,0,0),

log.sigma=c(0,0,0,0,0))