Assignment 4

Solution (a)

C $\approx \gamma 2^{A/2}$

=> $ln(C) \approx ln(\gamma) + A/2 * ln(2)$

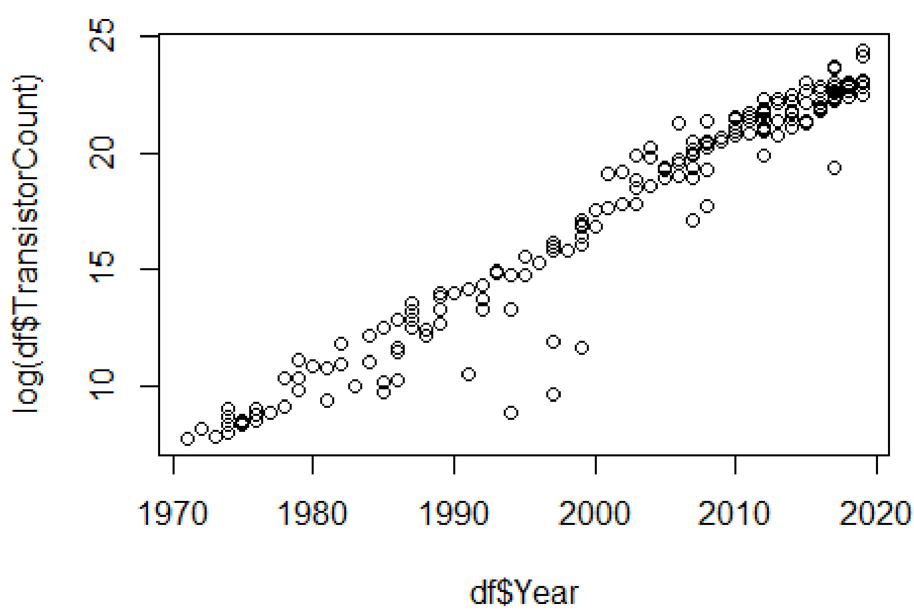
This equation is of the form $y \approx \beta_1 + \beta_2 x$

where $y = ln(C), x = A, \beta_1 = 1, \beta_2 = ln(2)/2$

The coefficent of A - β_2 is ln(2)/2 = 0.3465736

Solution (b)

```
df = read.csv("C:\\temp\\mooreslawdata.csv",header = TRUE)
plot(df$Year, log(df$TransistorCount))
```




```
#add new columns for year and log of transitor count to be used in JAGS model
df$A = df$Year
df$C_log = log(df$TransistorCount)
```

Solution (c)(i)

model {

for (j in 1:length(C_log)) {

```
  C_log[j] ~ dnorm(beta1 + beta2*(A[j] - mean_A), sigmasqinv)

  }
```

beta1 ~ dnorm(0,1/1000^2)

beta2 ~ dnorm(0,1/1000^2)

sigmasqinv ~ dgamma(0.001, 0.001)

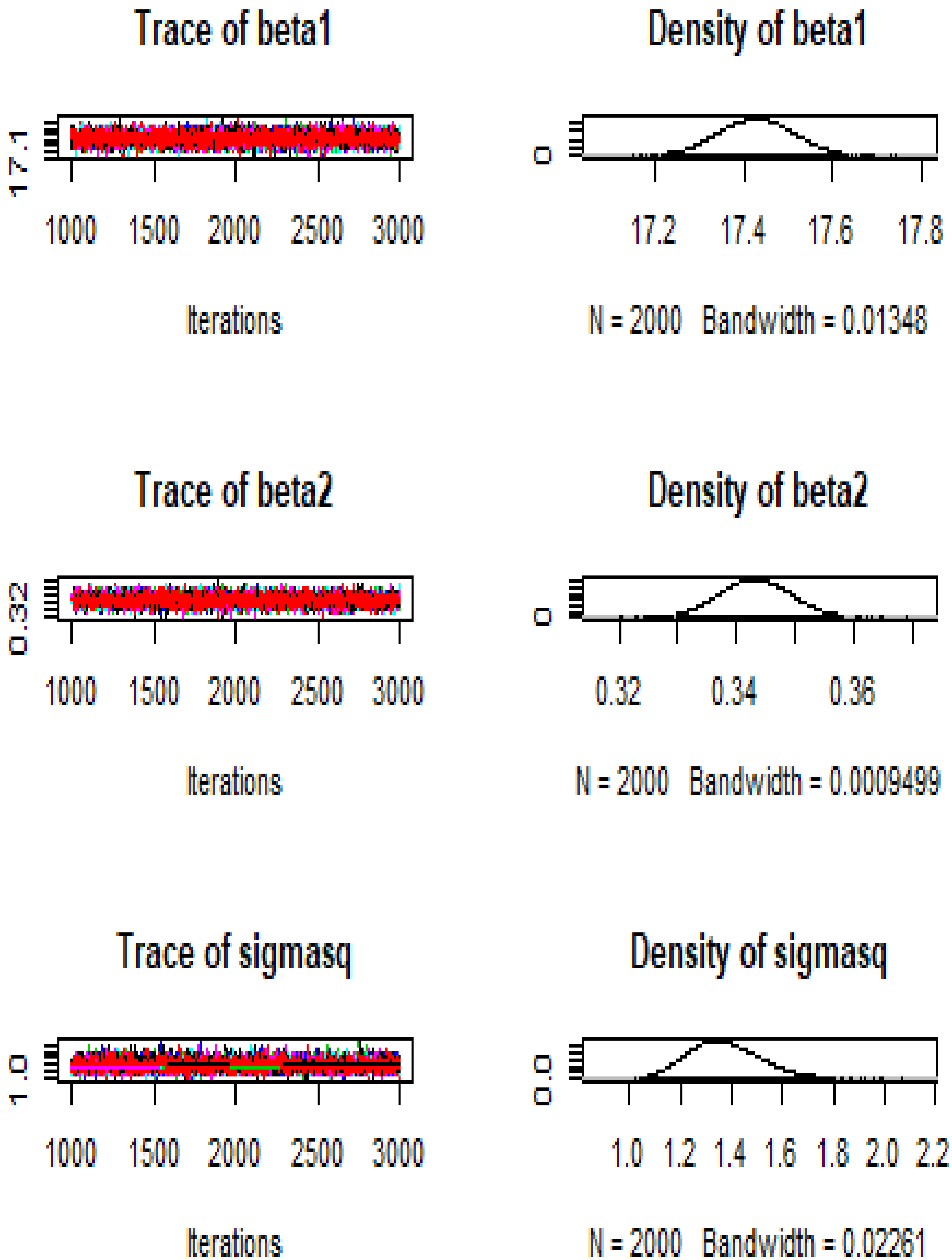
sigmasq <- 1/sigmasqinv

mean_A <- mean(A)

}

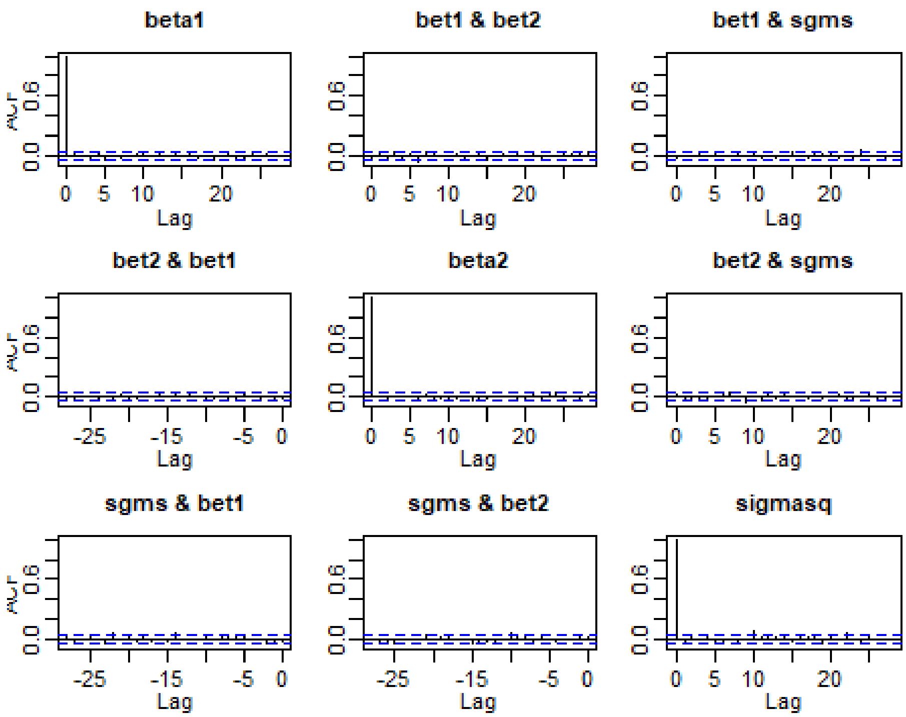
```
inits <- list(list(beta1=10000, beta2=10000,sigmasqinv=0.01,.RNG.name="base::
Wichmann-Hill",.RNG.seed=2),
               list(beta1=10000, beta2=10000,sigmasqinv=0.00001,.RNG.name="ba
se::Wichmann-Hill",.RNG.seed=3),
               list(beta1=10000, beta2=-10000,sigmasqinv=0.01,.RNG.name="base
::Wichmann-Hill",.RNG.seed=4),
               list(beta1=10000, beta2=-10000,sigmasqinv=0.00001,.RNG.name="b
ase::Wichmann-Hill",.RNG.seed=5),
               list(beta1=-10000, beta2=10000,sigmasqinv=0.01,.RNG.name="base
::Wichmann-Hill",.RNG.seed=6),
               list(beta1=-10000, beta2=10000,sigmasqinv=0.00001,.RNG.name="b
ase::Wichmann-Hill",.RNG.seed=7),
               list(beta1=-10000, beta2=-10000,sigmasqinv=0.01,.RNG.name="bas
e::Wichmann-Hill",.RNG.seed=8),
               list(beta1=-10000, beta2=-10000,sigmasqinv=0.00001,.RNG.name="
base::Wichmann-Hill",.RNG.seed=9)
               )
library(rjags)

## Warning: package 'rjags' was built under R version 3.6.3

## Loading required package: coda

## Warning: package 'coda' was built under R version 3.6.3

## Linked to JAGS 4.3.0

## Loaded modules: basemod,bugs

m1 <- jags.model("C:\\temp\\mooreslaw.bug", df, inits, n.chains=8)
```

```
## Warning in jags.model("C:\\temp\\mooreslaw.bug", df, inits, n.chains = 8):
## Unused variable "Processor" in data

## Warning in jags.model("C:\\temp\\mooreslaw.bug", df, inits, n.chains = 8):
## Unused variable "Year" in data

## Warning in jags.model("C:\\temp\\mooreslaw.bug", df, inits, n.chains = 8):
## Unused variable "TransistorCount" in data

## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 178
##    Unobserved stochastic nodes: 3
##    Total graph size: 516
##
## Initializing model

update(m1, 1000) # burn-in
x1 <- coda.samples(m1, c("beta1","beta2","sigmasq"),n.iter=2000)

plot(x1)
```



Trace of beta1
17.1
1000 1500 2000 2500 3000
Iterations
Density of beta1
0
17.2 17.4 17.6 17.8
N = 2000 Bandwidth = 0.01348
Trace of beta2
0.32
1000 1500 2000 2500 3000
Iterations
Density of beta2
0
0.32 0.34 0.36
N = 2000 Bandwidth = 0.0009499
Trace of sigmasq
1.0
1000 1500 2000 2500 3000
Iterations
Density of sigmasq
0.0
1.0 1.2 1.4 1.6 1.8 2.0 2.2
N = 2000 Bandwidth = 0.02261

```r
acf(x1[[1]])
```




```r
gelman.diag(x1, autoburnin = FALSE)

## Potential scale reduction factors:
##
##          Point est. Upper C.I.
## beta1             1          1
## beta2             1          1
## sigmasq           1          1
##
## Multivariate psrf
##
## 1
```

We can see from plots, chains are sampling from same regions, have indepdnent samples and gelman factors are 1 hence indicating convergence

Solution (c)(ii)

```r
summary(x1)

##
## Iterations = 1001:3000
## Thinning interval = 1
## Number of chains = 8
## Sample size per chain = 2000
##
```

```
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##             Mean       SD  Naive SE Time-series SE
## beta1   17.4258 0.088159 6.970e-04      6.910e-04
## beta2    0.3432 0.006212 4.911e-05      4.922e-05
## sigmasq  1.3774 0.148321 1.173e-03      1.185e-03
##
## 2. Quantiles for each variable:
##
##             2.5%    25%     50%     75%   97.5%
## beta1   17.2521 17.366 17.4259 17.4847 17.5982
## beta2    0.3311  0.339  0.3432  0.3474  0.3555
## sigmasq  1.1175  1.273  1.3666  1.4710  1.7020
```

Solution (c)(iii)

The slope is β_2, as we can see from (c)(ii) posterior mean is 0.3432 The 95% posterior interval is (`0.3311,0.3555`) and contains the value 0.3465736 from part a)

Solution (c)(iv)

The intercept is β_1, and its posterior mean is 17.4258 The 95% psterior interval is (17.2521, 17.5982)

Solution (d)(i)

$JAGS - MODEL$

model {

for (j in 1:length(C_log)) {

```
  C_log[j] ~ dnorm(beta1 + beta2*(A[j] - mean_A), sigmasqinv)

  }
```

beta1 ~ dnorm(0,1/1000^2)

beta2 ~ dnorm(0,1/1000^2)

sigmasqinv ~ dgamma(0.001, 0.001)

sigmasq <- 1/sigmasqinv

mean_A <- mean(A)

A_pred <- 2021

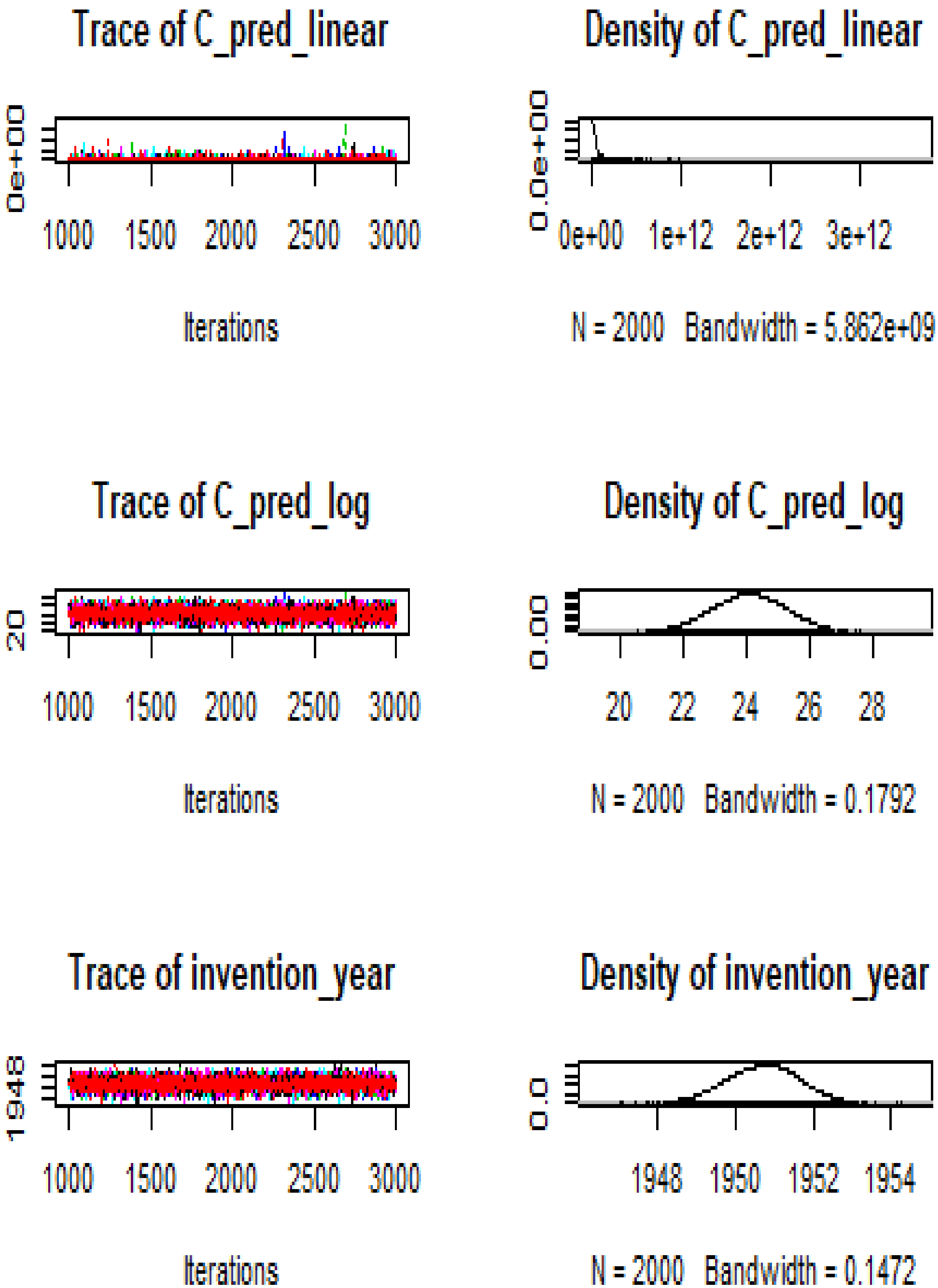
C_pred_log ~ dnorm(beta1 + beta2*(A_pred - mean_A), sigmasqinv)

C_pred_linear <-exp(C_pred_log)

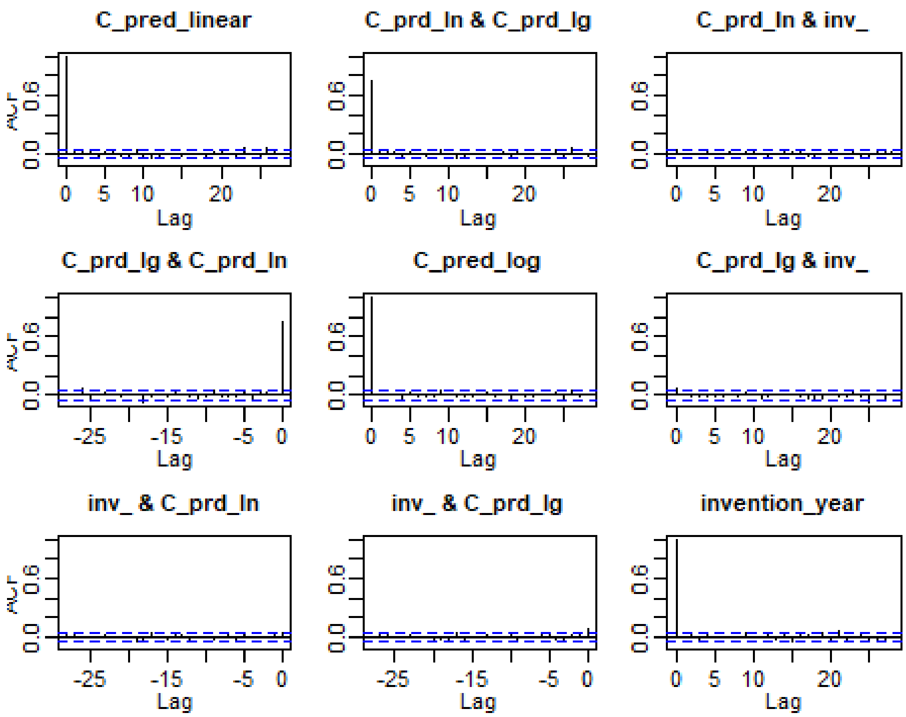
invention_year <- mean_A - beta1/beta2

}

```
m1 <- jags.model("C:\\temp\\mooreslawPred.bug", df, inits, n.chains=8, n.adap
t=1000)

## Warning in jags.model("C:\\temp\\mooreslawPred.bug", df, inits, n.chains =
8, :
## Unused variable "Processor" in data

## Warning in jags.model("C:\\temp\\mooreslawPred.bug", df, inits, n.chains =
8, :
## Unused variable "Year" in data

## Warning in jags.model("C:\\temp\\mooreslawPred.bug", df, inits, n.chains =
8, :
## Unused variable "TransistorCount" in data

## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 178
##    Unobserved stochastic nodes: 4
##    Total graph size: 524
##
## Initializing model

update(m1, 1000) # burn-in
x2 <- coda.samples(m1, c("C_pred_log","C_pred_linear","invention_year"),n.ite
r=2000)

plot(x2)
```


Trace of C_pred_linear
0e+00
1000 1500 2000 2500 3000
Iterations
Density of C_pred_linear
0.0e+00
0e+00 1e+12 2e+12 3e+12
N = 2000 Bandwidth = 5.862e+09
Trace of C_pred_log
20
1000 1500 2000 2500 3000
Iterations
Density of C_pred_log
0.00
20 22 24 26 28
N = 2000 Bandwidth = 0.1792
Trace of invention_year
1948
1000 1500 2000 2500 3000
Iterations
Density of invention_year
0.0
1948 1950 1952 1954
N = 2000 Bandwidth = 0.1472

```r
acf(x2[[1]])
```




```r
gelman.diag(x2, autoburnin = FALSE)

## Potential scale reduction factors:
##
##                  Point est. Upper C.I.
## C_pred_linear          1.01       1.01
## C_pred_log             1.00       1.00
## invention_year         1.00       1.00
##
## Multivariate psrf
##
## 1
```

We can see from plots, chains are sampling from same regions, have indepdnent samples and gelman factors are close to 1 hence indicating convergence

Solution (d)(ii)

```r
summary(x2)

##
## Iterations = 1001:3000
## Thinning interval = 1
## Number of chains = 8
## Sample size per chain = 2000
##
```

```
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                     Mean        SD  Naive SE Time-series SE
## C_pred_linear  5.802e+10 9.874e+10 7.806e+08      7.721e+08
## C_pred_log     2.410e+01 1.172e+00 9.263e-03      9.213e-03
## invention_year 1.951e+03 9.628e-01 7.612e-03      7.612e-03
##
## 2. Quantiles for each variable:
##
##                     2.5%       25%       50%       75%     97.5%
## C_pred_linear  2.899e+09 1.345e+10 2.947e+10 6.482e+10 2.821e+11
## C_pred_log     2.179e+01 2.332e+01 2.411e+01 2.489e+01 2.637e+01
## invention_year 1.949e+03 1.950e+03 1.951e+03 1.951e+03 1.953e+03
```

Solution (d)(iii)

95% posterior predictive confidence interval for year 2021 is (2.899e+09,2.821e+11)

Solution (d)(iv)

The year indicated by $\overline{A} - \beta 1/\beta 2$ is the solution to the regression equation

$log(C) \approx \beta_1 + \beta_2(A_i - \overline{A})$ when when log(C) = 0 i.e. C = 1 (transitor count of 1)

The 95% posterior interval for invention year is (1949,1953)

Solution (e)(i)

```
m1 <- jags.model("C:\\temp\\mooreslaw.bug", df, inits, n.chains=8, n.adapt=10
00)

## Warning in jags.model("C:\\temp\\mooreslaw.bug", df, inits, n.chains = 8,
:
## Unused variable "Processor" in data

## Warning in jags.model("C:\\temp\\mooreslaw.bug", df, inits, n.chains = 8,
:
## Unused variable "Year" in data

## Warning in jags.model("C:\\temp\\mooreslaw.bug", df, inits, n.chains = 8,
:
## Unused variable "TransistorCount" in data

## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 178
##    Unobserved stochastic nodes: 3
##    Total graph size: 516
```

```
##
## Initializing model

df$adjusted_year = df$Year - mean(df$Year)
update(m1, 1000) # burn-in
x3 <- coda.samples(m1, c("beta1","beta2","sigmasq"),n.iter=2000)

beta1_sim <- as.matrix(x3)[, "beta1"]
beta2_sim <- as.matrix(x3)[, "beta2"]

#combine beta1 and beta2 into a matrix
beta.sim <- cbind(beta1_sim,beta2_sim)

sigma2_sim <- as.matrix(x3)[, "sigmasq"]
Nsim <- nrow(beta.sim)

mod <- lm(C_log ~adjusted_year, data=df)
X <- model.matrix(mod)

error.std.sim <- matrix(NA,Nsim,nrow(df))

#Simulated standard Error
for (s in 1:Nsim)
  error.std.sim[s,] <- (df$C_log - X %*% cbind(beta.sim[s,])) /sqrt(sigma2_si
m[s])
```

Solution (e)(ii)

Replicate standard error is expected to have a standard normal distribution. Hence rnorm can be used to generate it.

```
#Simulate replicate Error from rnrom
error.rep.std.normal <-matrix(rnorm(Nsim*nrow(df)),Nsim,nrow(df))
```

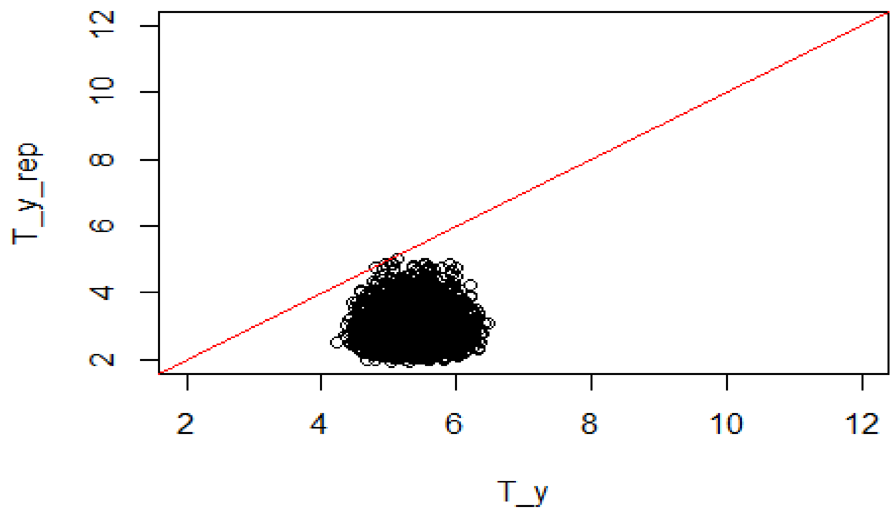
Solution (e)(iii)

Compute T(y,X,θ) and T(y^{rep},X,θ)

```
T_y <- apply(abs(error.std.sim), 1, max)
T_y_rep<- apply(abs(error.rep.std.normal), 1, max)
```

Solution (e)(iv)

```
plot(T_y,T_y_rep,xlim = c(2,12), ylim = c(2,12) )
abline(0,1,col ="red")
```



Solution (e)(v)

Compute p value for the outlier test statistic

```
mean(T_y_rep >= T_y)

## [1] 0.000125
```

Since p value is very low it indicates presence of outliers

Solution (e)(vi)

The exterme outlier can be defined as microprocessor that achieves the maximum difference for each simulation from standard normal error with highest probablity

```
N_y <- apply(abs(abs(error.std.sim) - abs(error.rep.std.normal)), 1, which.ma
x)

#Sort by highest incidence of a uP being maximum outlier wrt to the metric an
d print top 3
sort(table(N_y),decreasing=TRUE)[1:3]

## N_y
##   63   55   66
## 9730 5900  347
```

Index no 63 (F21 processor) has the highest occurrence (9730) out of 16000 simulations as extreme i.e. probability .608 for it bring the extreme outlier