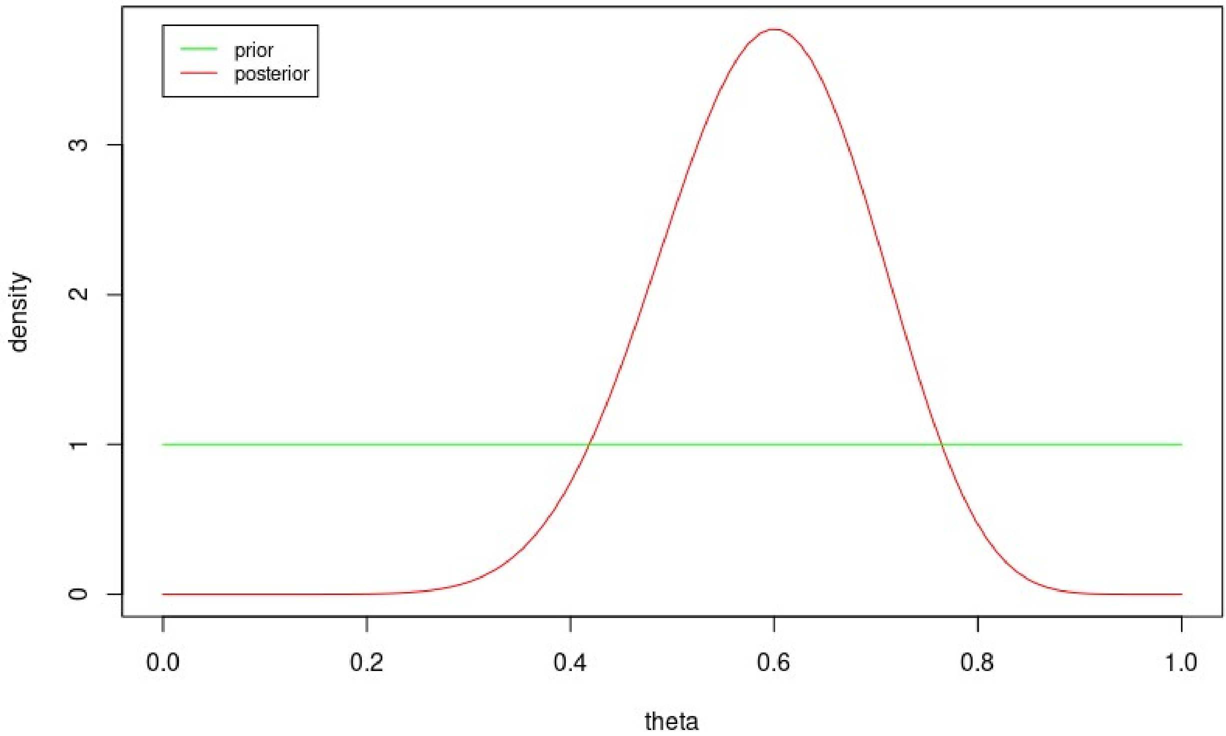
Introduction to Bayesian analysis 1, Take-home task
Tuomas Porkamaa

1.
Likelihood:
$Y = 12|\ \theta \sim Binom(n = 20, \theta), p(Y = 12|\ \theta) = \binom{20}{12}\theta^{12}(1-\theta)^{8}$

a.)
Prior:
$\theta \sim Unif(0, 1)$
Posterior:
p$(\theta|\ Y = 12) \propto p(\theta)p(Y = 12|\ \theta) \propto \theta^{12}(1-\theta)^{8}$
$\therefore \theta|\ Y = 12 \sim Beta(13, 9)$
$\therefore E(\theta|\ Y = 12) = 0.59, Var(\theta|\ Y = 12) = 0.0105$

R-Codes and plots:

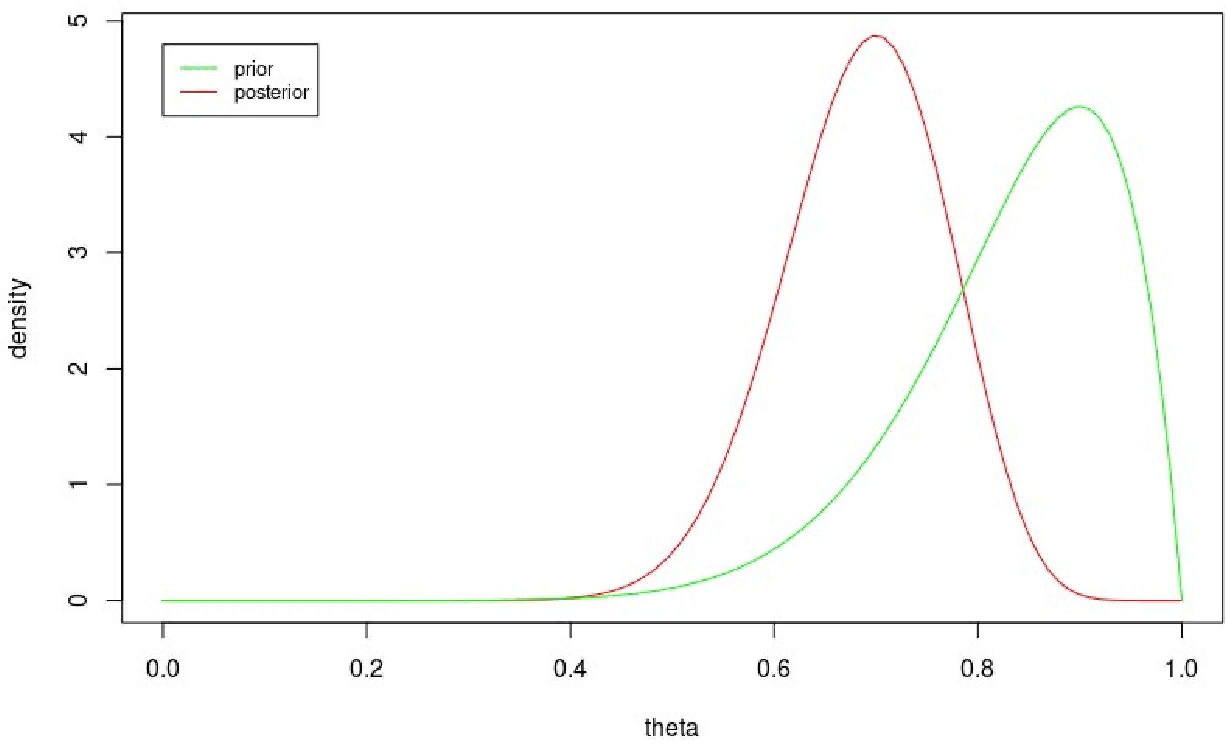
```
theta = seq(0,1,length.out = 100)
thetaPostDist <- dbeta(theta, 13, 9)
thetaPriorDist <- dunif(theta, 0, 1)

plot(theta, thetaPostDist, type = "l", col="red", ylab="density")
lines(theta, thetaPriorDist, type = "l", col="green")
legend(0, 3.8, legend = c("prior", "posterior"), col = c("green", "red"), lty =
1:1, cex = 0.8)
```



b.)
Prior:
$\theta \sim Beta(10, 2)$
Posterior:
p$(\theta|\ Y = 12) \propto p(\theta)p(Y = 12|\ \theta) \propto \theta^9(1-\theta)\theta^{12}(1-\theta)^8 = \theta^{21}(1-\theta)^9$
$\therefore \theta|\ Y = 12 \sim Beta(22, 10)$
$\therefore E(\theta|\ Y = 12) = 0.6875, Var(\theta|\ Y = 12) = 0.0065$

R-Codes and plots:

```
data <- c(rep(1,12), rep(0,8))
thetaPostInfo <- bb.sum(data, 1, 10, 2)
thetaPostDist <- dbeta(theta, thetaPostInfo$alfa1, thetaPostInfo$beta1)
thetaPriorDist <- dbeta(theta, 10, 2)

plot(theta, thetaPostDist, type = "l", col="red", ylab="density")
lines(theta, thetaPriorDist, type = "l", col="green")
legend(0, 4.8, legend = c("prior", "posterior"), col = c("green", "red"), lty =
1:1, cex = 0.8)
```



c.)
In case a.) the non-informative prior distribution is used, so our posterior model is completely data driven. Our question in hand was formed as; Is an annual rate of degradation of gravestones more than 0.12mm, where θ is the probability of this phenomenon. Based on the first model we could estimate θ to fall in range [0.42, 0.76] with 90% probability. Intuitively this range seems a kind of “neutral” and so, in this specific graveyard, the gravestones doesn’t degrade significantly fast.

In case b.) the experimenters had some prior information about θ, which can be modelled as Beta(10, 2) distribution. Based on the second plot, it can be seen that the prior distribution is strongly skewed to larger values of θ, so compared to case a.) we might expect a bit more “extreme” posterior inference about θ, especially because the sample size is small. Based on the posterior model we could estimate θ to fall in range [0.55, 0.81] with 90% probability, so we get a bit more significant results compared to case a.). It might be a reasonable to say that in this particular cemetery, it is more likely that gravestones degrades more than 12mm per year in most cases. If we want to made more detailed conclusions, we might want to take a look to methods how experimenters constructed their prior believes in a form of Beta(10, 2), because in this case that distribution plays significant role also in posterior inference.

2.

a.)

Likelihood:

$$p(\mathbf{Y}_A|\ \lambda_A) = \prod_{i=1}^{10} p(y_{Ai}|\ \lambda_A) = \prod_{i=1}^{10} \lambda_A^{y_{Ai}} e^{-\lambda_A} \frac{1}{y_{Ai}!} = \lambda_A^{\sum y_{Ai}} e^{-10\lambda_A} \prod_{i=1}^{10} \frac{1}{y_{Ai}!}$$

Prior:

$$\lambda_A \sim Gamma(120, \frac{1}{10})$$

Posterior:

$$p(\lambda_A|\mathbf{Y}_A) \propto p(\lambda_A)p(\mathbf{Y}_A|\ \lambda_A) \propto \lambda_A^{120-1} e^{-10\lambda_A} \lambda_A^{\sum y_i} e^{-10\lambda_A} = \lambda_A^{120+\sum y_i -1} e^{-20\lambda_A}$$

$$\therefore \lambda_A|\ \mathbf{Y}_A \sim Gamma(120 + \sum_{i=1}^{10} y_{Ai}, \frac{1}{20}) \equiv Gamma(237, \frac{1}{20})$$

$$\therefore E(\lambda_A|\ \mathbf{Y}_A) = 11.85, Var(\lambda_A|\ \mathbf{Y}_A) = 0.5925$$

R-codes:

```
dataA <- c(12,9,12,14,13,13,15,8,15,6)
theta <- seq(7,17,length.out = 200)
thetaAPriorAlfa <- 120
thetaAPriorBeta <- 1/10 # 1/10
thetaAPriorDist <- dgamma(theta, thetaAPriorAlfa, scale=thetaAPriorBeta)
thetaAPostInfo <- pg.sum(dataA, thetaAPriorAlfa, 1/thetaAPriorBeta)
thetaAPostDist <- dgamma(theta, thetaAPostInfo$alfa1, rate=thetaAPostInfo$beta1)
thetaAPostMean <- thetaAPostInfo$Mean
thetaAPostVar <- thetaAPostInfo$Var
thetaAPostHDI <- pg.hdi(dataA, thetaAPriorAlfa, thetaAPriorBeta, 0.9)

cat("Theta A:",
    "\nPosterior mean:", thetaAPostMean,
    "\nPosterior variance:", thetaAPostVar,
    "\n90% HDI: [", thetaAPostHDI$Ala,", ", thetaAPostHDI$Yla,"]\n")

Code output:
Theta A:
Posterior mean: 11.85
Posterior variance: 0.5925
90% HDI: [ 10.58073 ,  13.10933 ]
```

b.)
Likelihood:

$$p(\mathbf{Y}_B|\ \lambda_B) = \lambda_B^{\sum y_{Bi}} e^{-13\lambda_A} \prod_{i=1}^{13} \frac{1}{y_{Bi}!}$$

Prior:

$\lambda_B \sim Gamma(12, 1)$

Posterior:

$$p(\lambda_B|\mathbf{Y}_B) \propto p(\lambda_B)p(\mathbf{Y}_B|\ \lambda_B) \propto \lambda_B^{12-1} e^{-\lambda_B} \lambda_B^{\sum y_{Bi}} e^{-13\lambda_B} = \lambda_B^{12+\sum y_{Bi}-1} e^{-14\lambda_B}$$

$$\therefore \lambda_B|\ \mathbf{Y}_B \sim Gamma(12 + \sum_{i=1}^{13} y_{Bi}, \frac{1}{14}) \equiv Gamma(125, \frac{1}{14})$$

$$\therefore E(\lambda_B|\ \mathbf{Y}_B) = 8.93, Var(\lambda_B|\ \mathbf{Y}_B) = 0.6378$$

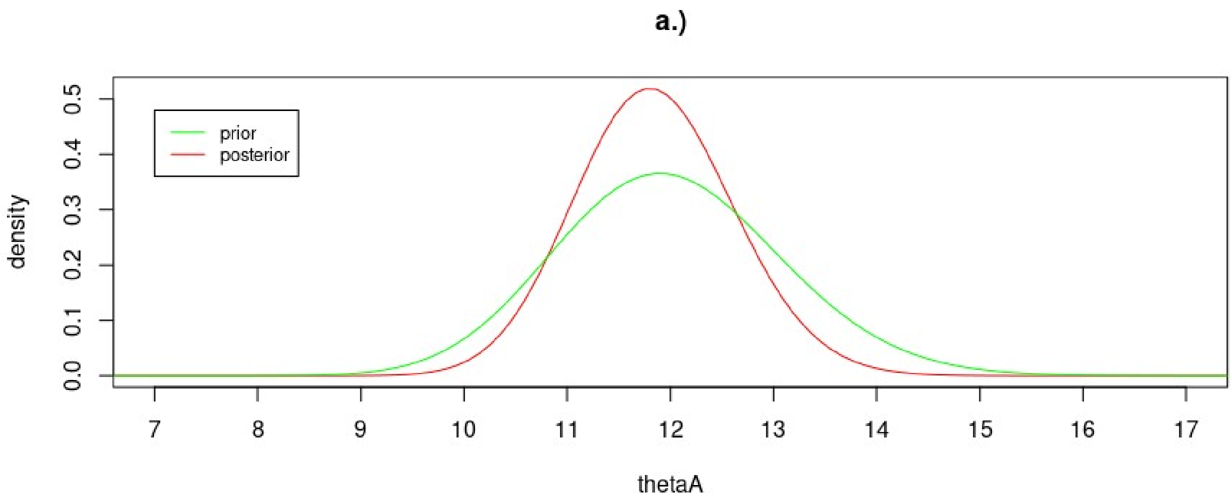
R-codes:

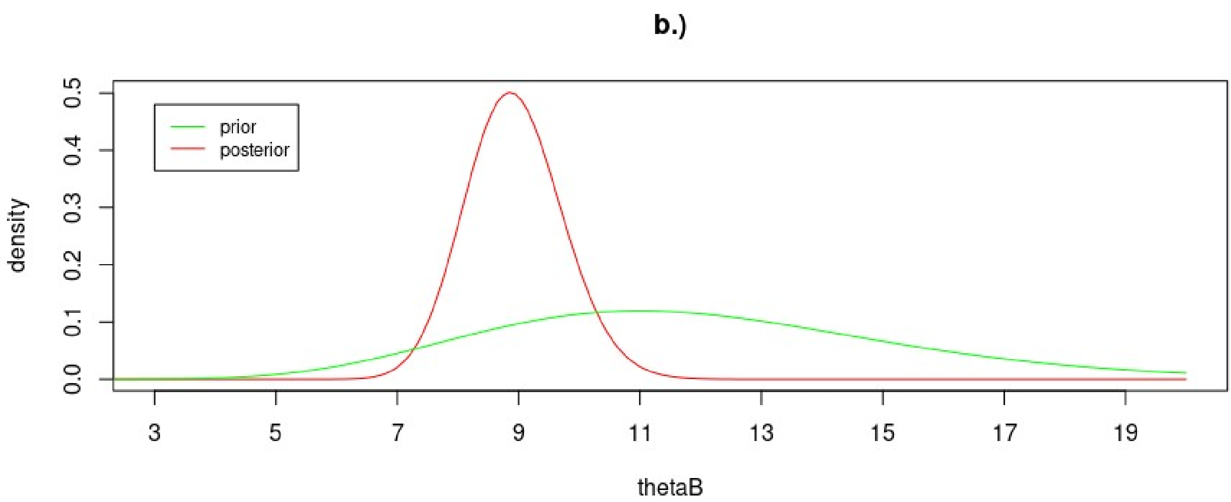
```
dataB <- c(11,11,10,9,9,8,7,10,6,8,8,9,7)
theta <- seq(0,20,length.out = 200)
thetaBPriorAlfa <- 12
thetaBPriorBeta <- 1
thetaBPriorDist <- dgamma(theta, thetaBPriorAlfa, thetaBPriorBeta)
thetaBPostInfo <-pg.sum(dataB, thetaBPriorAlfa, thetaBPriorBeta)
thetaBPostDist <- dgamma(theta, thetaBPostInfo$alfa1, thetaBPostInfo$beta1)
#1/thetaBPostInfo$beta1
thetaBPostMean <- thetaBPostInfo$Mean
thetaBPostVar <- thetaBPostInfo$Var
thetaBPostHDI <- pg.hdi(dataA, thetaBPriorAlfa, thetaBPriorBeta, 0.9)

cat("Theta B:",
    "\nPosterior mean:", thetaBPostMean,
    "\nPosterior variance:", thetaBPostVar,
    "\n90% HDI: [", thetaBPostHDI$Ala,", ", thetaBPostHDI$Yla,"]\n")

Code output:
Theta B:
Posterior mean: 8.928571
Posterior variance: 0.6377551
90% HDI: [ 10.02459 ,  13.41231 ]
```

c.)

R-codes and plots:

```
par(mfrow=c(2,1))
# Plotting thetaA
xaxis <- seq(7, 17, by = 1)
plot(theta, thetaAPostDist, type = "l", col = "red", ylab = "density", xlab =
"thetaA", xlim = c(7, 17), xaxt = 'n', main = "a.)")
axis(1, at = xaxis, las = 1)
lines(theta, thetaAPriorDist, type = "l", col = "green")
legend(7, 0.48, legend = c("prior", "posterior"), col = c("green", "red"), lty =
1:1, cex = 0.8)

# Plotting thetaB
xaxis <- seq(3, 20, by = 2)
plot(theta, thetaBPostDist, type = "l", col = "red", ylab = "density", xlab =
"thetaB", xlim = c(3, 20), xaxt = 'n', main = "b.)")
axis(1, at = xaxis, las=1)
lines(theta, thetaBPriorDist, type = "l", col = "green")
legend(3, 0.48, legend = c("prior", "posterior"), col = c("green", "red"), lty =
1:1, cex = 0.8)
```





d.)

R-codes:

```
dataB <- c(11,11,10,9,9,8,7,10,6,8,8,9,7)
#theta <- seq(0,20,length.out = 200)
postMeans <- data.frame(Prior_alpha=integer(0), Prior_beta=integer(0),
Post_alpha=integer(0), Post_beta=double(0), Post_mean=double(0))
for(n in 1:50){
    thetaBPriorAlfa <- 12*n
    thetaBPriorBeta <- n
    thetaBPostInfo <- pg.sum(dataB, thetaBPriorAlfa, 1/thetaBPriorBeta)
    postMeans[nrow(postMeans)+1,] <- c(thetaBPriorAlfa, thetaBPriorBeta,
                                       thetaBPostInfo$alfa1,
                                       thetaBPostInfo$beta1,
                                       thetaBPostInfo$Mean)
}
postMeans
```

```
Code output:
   Prior_alpha Prior_beta Post_alpha Post_beta Post_mean
1           12          1        125  14.00000  8.928571
2           24          2        137  13.50000 10.148148
3           36          3        149  13.33333 11.175000
4           48          4        161  13.25000 12.150943
5           60          5        173  13.20000 13.106061
6           72          6        185  13.16667 14.050633
7           84          7        197  13.14286 14.989130
8           96          8        209  13.12500 15.923810
9          108          9        221  13.11111 16.855932
10         120         10        233  13.10000 17.786260
11         132         11        245  13.09091 18.715278
12         144         12        257  13.08333 19.643312
13         156         13        269  13.07692 20.570588
14         168         14        281  13.07143 21.497268
15         180         15        293  13.06667 22.423469
16         192         16        305  13.06250 23.349282
17         204         17        317  13.05882 24.274775
18         216         18        329  13.05556 25.200000
19         228         19        341  13.05263 26.125000
20         240         20        353  13.05000 27.049808
21         252         21        365  13.04762 27.974453
22         264         22        377  13.04545 28.898955
23         276         23        389  13.04348 29.823333
24         288         24        401  13.04167 30.747604
25         300         25        413  13.04000 31.671779
26         312         26        425  13.03846 32.595870
27         324         27        437  13.03704 33.519886
28         336         28        449  13.03571 34.443836
29         348         29        461  13.03448 35.367725
30         360         30        473  13.03333 36.291560
31         372         31        485  13.03226 37.215347
32         384         32        497  13.03125 38.139089
33         396         33        509  13.03030 39.062791
34         408         34        521  13.02941 39.986456
35         420         35        533  13.02857 40.910088
36         432         36        545  13.02778 41.833689
37         444         37        557  13.02703 42.757261
```

38	456	38	569	13.02632	43.680808
39	468	39	581	13.02564	44.604331
40	480	40	593	13.02500	45.527831
41	492	41	605	13.02439	46.451311
42	504	42	617	13.02381	47.374771
43	516	43	629	13.02326	48.298214
44	528	44	641	13.02273	49.221640
45	540	45	653	13.02222	50.145051
46	552	46	665	13.02174	51.068447
47	564	47	677	13.02128	51.991830
48	576	48	689	13.02083	52.915200
49	588	49	701	13.02041	53.838558
50	600	50	713	13.02000	54.761905

It can be seen that the posterior expectation of θ_B is closest to that of to be θ_A, when the posterior distribution is formed as (output line 4)

$\theta_B|\ \mathbf{Y}_B \sim Gamma(161, \dfrac{1}{13.25})$

and prior distribution for θ_B is formed as

$\theta_B \sim Gamma(48, 4)$

To get the prior distribution for θ_B, we have to examine and reorganize the new posterior pdf:

$$p(\theta_B|\ \mathbf{Y}_B) \propto \theta_B^{48+\sum y_{Bi}-1} e^{-\theta_B(\frac{1}{4}+13} \propto \theta_B^{48-1} e^{-\frac{1}{4}\theta_B} \theta_B^{\sum y_{Bi}} e^{-13\theta_B}$$

Thus our updated prior believes about θ_B can be modelled by distribution:

$\theta_B \sim Gamma(48, 4)$

3.)
a.)
R-codes:

```
fn <- "/home/tuomas/R/Projects/Bayesian analysis/Take_home_task/task3.jag"
cat(
  "model
  {
    # Likelihood:
    for(i in 1:N)
    {
      x[i] ~ dnegbin(p[i],1)
      # Prior for p's
      p[i] ~ dbeta(a, b)
    }
    # Hyperpriors for a and b
    a ~ dgamma(1, 1)
    b ~ dgamma(1, 1)
    #b ~ dgamma(1, 0.5)

    theta <- a/(a+b)
  }
  ",
  file = fn )

task3.data <- list(x=c(4,0,1,7,3,2,8,0), N=8)
task3.init <- list(p=c(0.5,0.5,0.5,0.5,0.5,0.5,0.5,0.5), a=11, b=1)
task3.model <- jags.model(file = fn, data = task3.data, inits = task3.init,
n.chains = 3, n.adapt = 2000)

task3.par <- c("p[1]","p[2]","p[3]","p[4]","p[5]","p[6]","p[7]","p[8]", "theta")
task3.result <- coda.samples(model = task3.model, variable.names = task3.par,
n.iter = 10000, thin = 10)
summary(task3.result)

Output:
1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

        Mean     SD Naive SE Time-series SE
p[1]  0.2661 0.1547 0.002824       0.002825
p[2]  0.5494 0.2440 0.004455       0.004362
p[3]  0.4230 0.2143 0.003912       0.003836
p[4]  0.1928 0.1182 0.002157       0.002105
p[5]  0.3032 0.1703 0.003109       0.003049
p[6]  0.3581 0.1903 0.003475       0.003490
p[7]  0.1762 0.1105 0.002018       0.002018
p[8]  0.5431 0.2475 0.004519       0.004518
theta 0.3824 0.1087 0.001985       0.001985

2. Quantiles for each variable:

         2.5%     25%    50%    75%  97.5%
p[1]  0.03300 0.14723 0.2450 0.3633 0.6256
p[2]  0.10663 0.36323 0.5497 0.7388 0.9774
p[3]  0.06973 0.25587 0.4030 0.5709 0.8590
p[4]  0.02422 0.10579 0.1730 0.2620 0.4709
```

```
p[5]  0.04192 0.17055 0.2817 0.4185 0.6797
p[6]  0.04959 0.20897 0.3376 0.4905 0.7604
p[7]  0.02223 0.09242 0.1559 0.2407 0.4362
p[8]  0.10373 0.34988 0.5397 0.7438 0.9762
theta 0.19396 0.30421 0.3742 0.4562 0.6130
```

Mean and standard deviation for p_i's are listed in output table 1
95% credible interval for p_i's can be found in output table 2 so that
CI = [2.5% quantile, 97.5% quantile]

b.)

```
Code output when b~Gamma(1,2):
1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

        Mean     SD Naive SE Time-series SE
p[1]  0.2524 0.1430 0.002610       0.002597
p[2]  0.4707 0.2346 0.004283       0.004372
p[3]  0.3925 0.1968 0.003593       0.003591
p[4]  0.1917 0.1126 0.002055       0.001967
p[5]  0.2910 0.1604 0.002929       0.003109
p[6]  0.3294 0.1772 0.003234       0.003215
p[7]  0.1771 0.1088 0.001987       0.001928
p[8]  0.4752 0.2355 0.004300       0.004138

2. Quantiles for each variable:

         2.5%     25%    50%    75%  97.5%
p[1]  0.03867 0.14244 0.2322 0.3433 0.5793
p[2]  0.08765 0.28451 0.4492 0.6400 0.9487
p[3]  0.07190 0.24255 0.3724 0.5267 0.8186
p[4]  0.02832 0.10608 0.1729 0.2575 0.4539
p[5]  0.04844 0.16584 0.2696 0.3964 0.6437
p[6]  0.05440 0.19485 0.3055 0.4536 0.7109
p[7]  0.02465 0.09431 0.1566 0.2425 0.4357
p[8]  0.08782 0.29320 0.4516 0.6496 0.9421
```

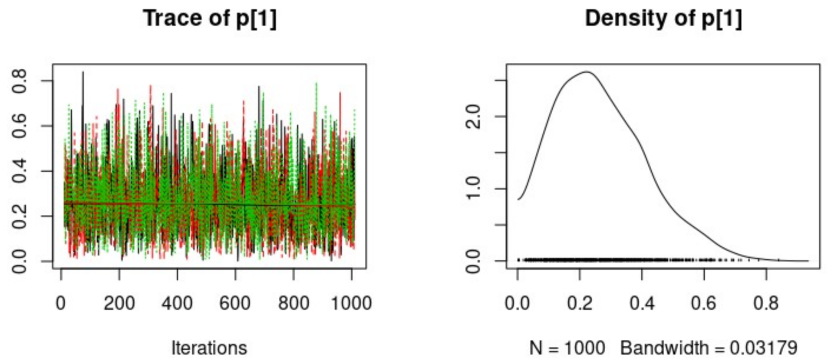
c.)
Posterior inference for $\theta = \frac{\alpha}{\alpha+\beta}$:

```
Empirical mean and standard deviation:
        Mean     SD Naive SE Time-series SE
theta 0.3819 0.1086 0.001984       0.001997
Quantiles:
         2.5%     25%    50%    75%  97.5%
theta 0.18885 0.30576 0.3740 0.4525 0.6144
```

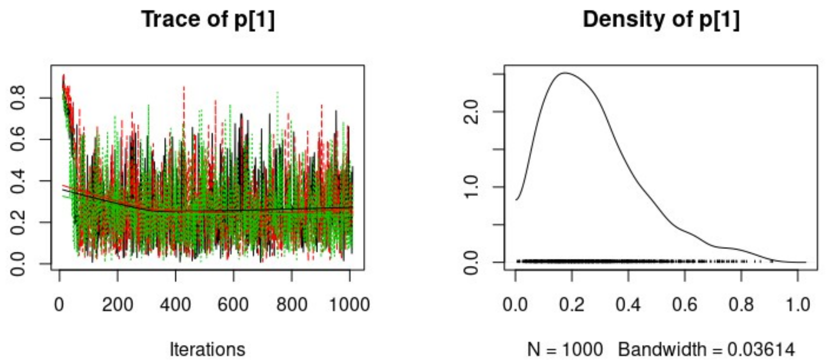
d.)
The choose of initial values for markov chain has effect only on so called burn-in-phase which is an early stage of simulated mcmc process. If the initial values are poorly chosen, it will take more time to markov chain to converge towards posterior distribution, and thus decreases the performance, which for example, can be seen from the trace-plot of mcmc process. There are quite a few of ways to select initial values for parameters, but some preferred methods are to use MLE-estimate of a parameter, or alternatively, when using multiple chains it is often preferred to start from dispersed points in parameter space. Below are listed traceplots using various initalization values. Number of

iterations were decreased to 1000, so that the burn-in phase could be seen more easily and n.adapt is set to 10 so we can inspect the early stage of markov chain more clearly.

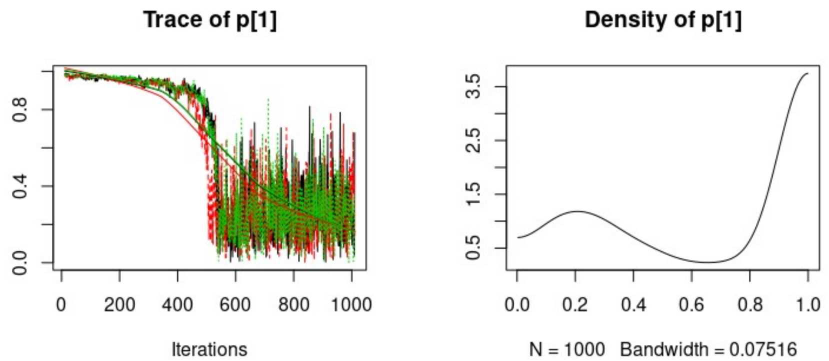
Case 1: a=11, b=1, n.adapt=10, n.iter=1000:



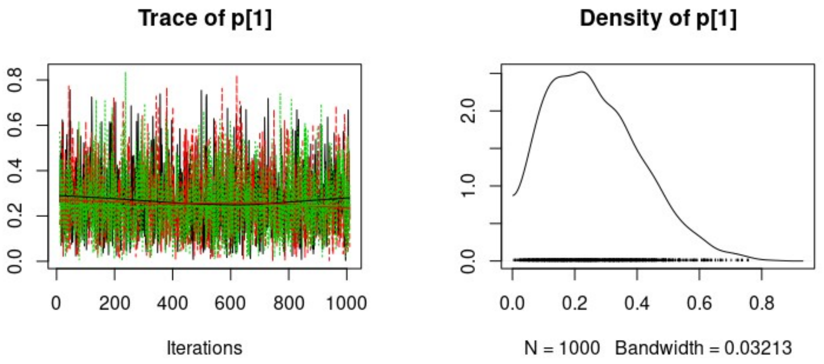
Case 2: a=111, b=1, n.adapt=10, n.iter=1000:



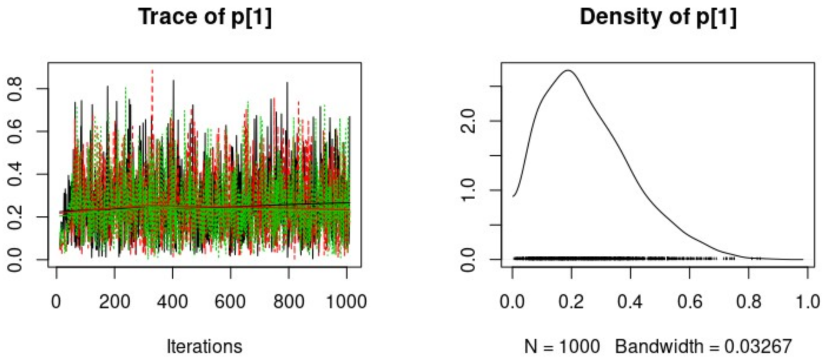
Case 3: a=1111, b=1, n.adapt=10, n.iter=1000:



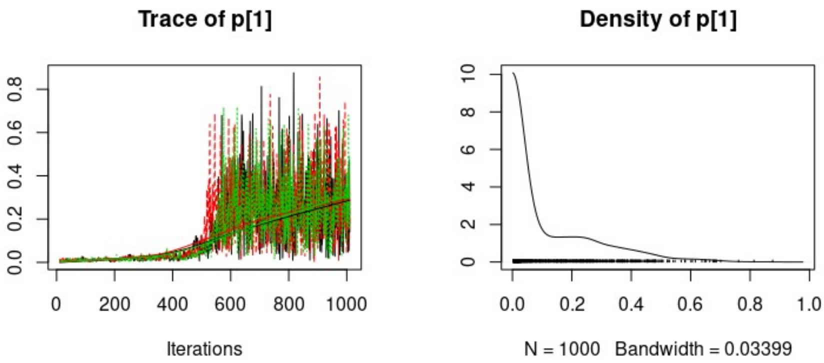
Case 4: a=1, b=11, n.adapt=10, n.iter=1000:



Case 5: a=1, b=111, n.adapt=10, n.iter=1000:



Case 6: a=1, b=1111, n.adapt=10, n.iter=1000:

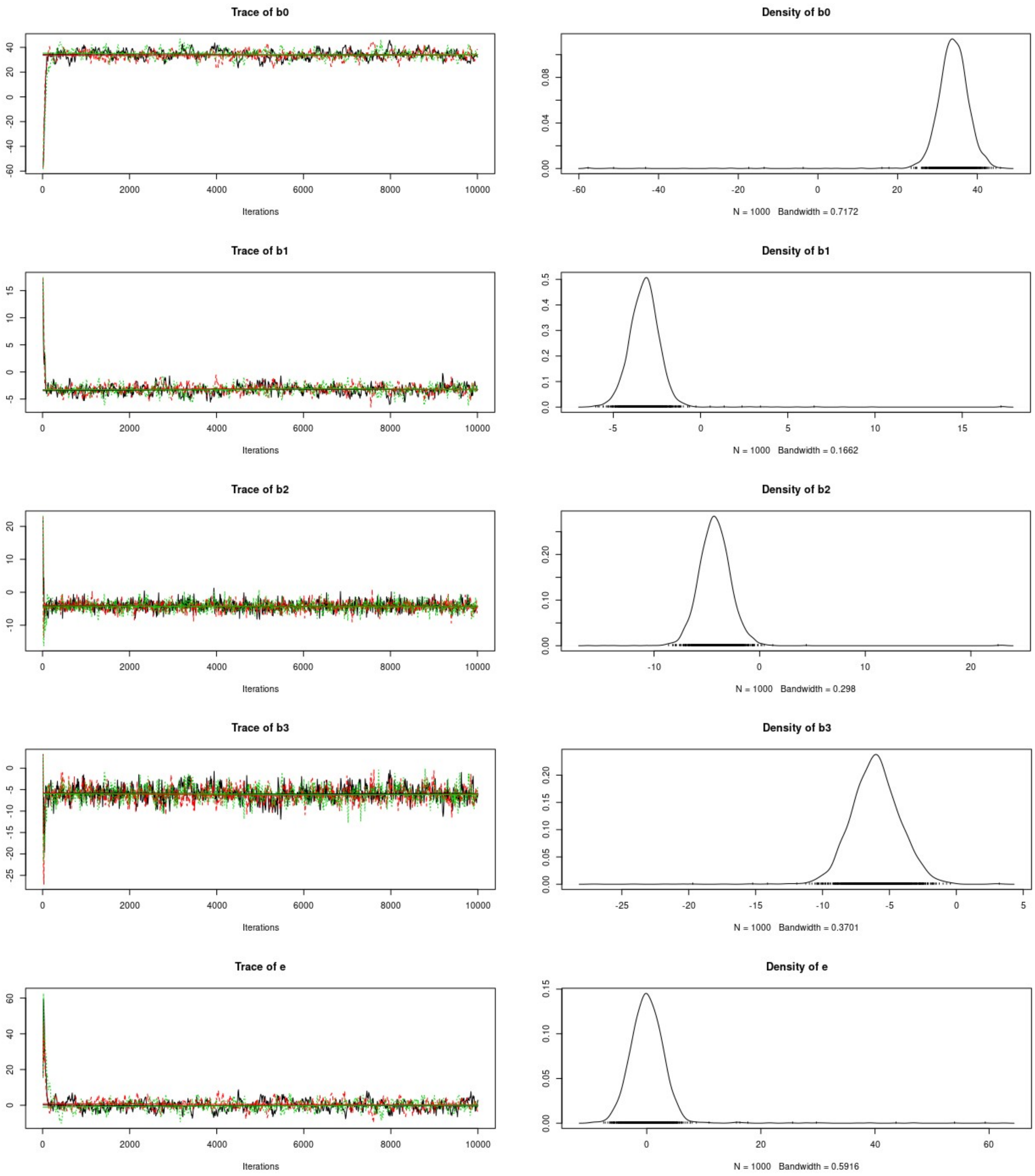


As can be seen, the effect of extreme parameter values dominances the early stage of mcmc process. However, in these examples only the first 10 samples are discarded from markov chain and a result from this will produce highly biased posterior information. If n.adapt is set as a larger value, the poorly selected parameter values doesn't have that much effect to the posterior inference.

4.

R-codes:

```
fn <- "/home/tuomas/R/Projects/Bayesian analysis/Take_home_task/task4.jag"
cat("
  model{
  # Likelihood:
  for(i in 1:N){
    mu[i] <- b0 + b1*weight[i] + b2*sixcyl[i] + b3*eightcyl[i] + e
    mpg[i] ~ dnorm(mu[i], prec)
  }
  # Priors:
  b0 ~ dnorm(0, 0.0001) # Variance = 1000
  b1 ~ dnorm(0, 0.0001)
  b2 ~ dnorm(0, 0.0001)
  b3 ~ dnorm(0, 0.0001)
  e ~ dnorm(0, prec)
  prec ~ dgamma(0.01, 0.01) #?
  sigma2 <- 1/sqrt(prec)

  # Predictive distribution:
  mu.new <- b0 + b1*weight.new + e
  mpg.new ~ dnorm(mu.new, prec)

  }",
  file = fn )

task4.data <- list(mpg=c(21.0,21.0,22.8,21.4,18.7,18.1,14.3,24.4,22.8,19.2,17.8,
                         16.4,17.3,15.2,10.4,10.4,14.7,32.4,30.4,33.9,21.5,15.5,
                         15.2,13.3,19.2,27.3,26.0,30.4,15.8,19.7,15.0, 21.4),
                   weight=c(2.620, 2.875, 2.320, 3.215, 3.440, 3.460, 3.570,
                            3.190, 3.150, 3.440, 3.440, 4.070, 3.730, 3.780,
                            5.250, 5.424, 5.345, 2.200, 1.615, 1.835, 2.465,
                            3.520, 3.435, 3.840, 3.845, 1.935, 2.140, 1.513,
                            3.170, 2.770, 3.570, 2.780),
                   sixcyl=c(1, 1, 0, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,
                            0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0),
                   eightcyl=c(0, 0, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
                              0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0, 1, 0, 1, 0),
                   weight.new = 3.5,
                   mpg.new = NA,
                   N=32
                   )
task4.init <- list(b0=20,b1=20,b2=20,b3=20,e=0.5,prec=1)
task4.model <- jags.model(fn, data = task4.data, inits = task4.init,
                          n.chains = 3, n.adapt = 2000)

task4.pars <- c("b0", "b1", "b2", "b3", "e")
task4.postInfo <- coda.samples(model = task4.model, variable.names = task4.pars,
                               n.iter = 10000, thin = 10)
summary(task4.postInfo)
plot(task4.postInfo)
```


Trace of b0
Iterations
Density of b0
N = 1000 Bandwidth = 0.7172
Trace of b1
Iterations
Density of b1
N = 1000 Bandwidth = 0.1662
Trace of b2
Iterations
Density of b2
N = 1000 Bandwidth = 0.298
Trace of b3
Iterations
Density of b3
N = 1000 Bandwidth = 0.3701
Trace of e
Iterations
Density of e
N = 1000 Bandwidth = 0.5916

```
Console output:

Iterations = 10:10000
Thinning interval = 10
Number of chains = 3
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

      Mean    SD Naive SE Time-series SE
b0 33.6046 6.332  0.11560        0.33247
b1 -3.1780 1.168  0.02133        0.04888
b2 -4.2375 1.723  0.03145        0.03366
b3 -6.0599 1.939  0.03540        0.06144
e   0.2721 4.143  0.07563        0.22163

2. Quantiles for each variable:

     2.5%    25%      50%    75%  97.5%
b0 26.418 31.766 33.96030 36.262 41.099
b1 -4.902 -3.749 -3.20044 -2.707 -1.583
b2 -7.093 -5.194 -4.28532 -3.326 -1.235
b3 -9.527 -7.208 -6.05986 -4.887 -2.471
e  -5.305 -1.806  0.01488  1.903  5.709
```

b.)
95% equal tail probability intervals:

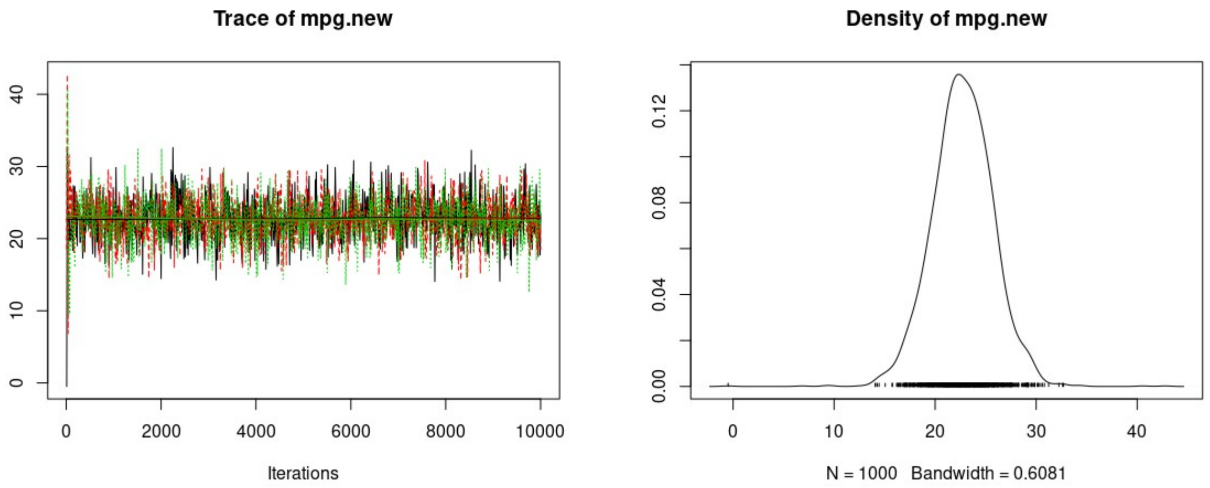
$\beta_0 : [27.004, 40.890]$
$\beta_1 : [-4.858, -1.584]$
$\beta_2 : [-7.170, -1.315]$
$\beta_3 : [-9.515, -2.502]$
$error : [-5.464, 5.925]$

c.)
There is not significant difference in mpg when comparing six and eight cylinder cars. When viewing the probability intervals in task b.) it can be seen that the intervals for β_2 and β_3 extensively overlaps each other. Taking into account also the dummy binary variables, the overall difference is not notable.

d.)

R-codes and plots:

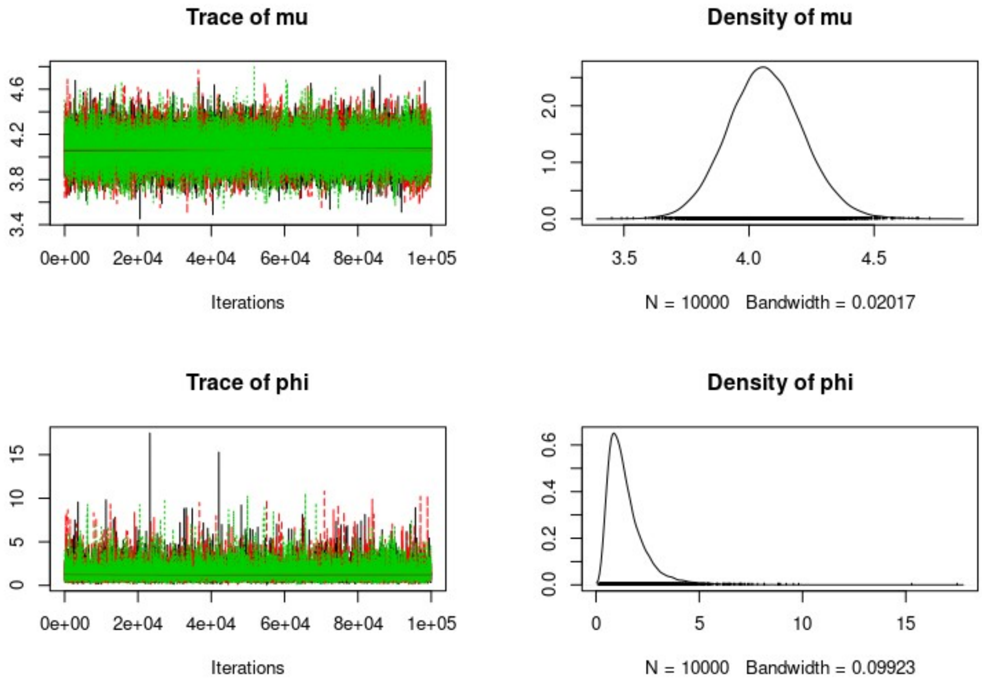
```
task4d.pars <- c("mpg.new")
task4d.postInfo <- coda.samples(model = task4d.model,
                                variable.names = task4d.pars,
                                n.iter = 10000, thin = 10)
summary(task4d.postInfo)
plot(task4d.postInfo)
```




```
Console output:
1. Empirical mean and standard deviation for each variable (mpg.new),
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
      22.89788        3.09286        0.05647        0.06324

2. Quantiles for each variable (mpg.new):

 2.5%   25%   50%   75% 97.5%
17.00 20.91 22.85 24.83 28.93
```

5.
a.)
R-codes and plots:

```
cat("
  model{
  # Likelihood:
  for(i in 1:N){
    y1[i] ~ dnorm(mu, sigma1)
    y2[i] ~ dnorm(mu, sigma2)

  }
  # Priors:
  mu ~ dnorm(4.7, 1/v)
  sigma1 ~ dgamma(4, 1/5)
  sigma2 ~ dgamma(4, 1/5)
  tau1 <- 1/sigma1
  tau2 <- 1/sigma2
  phi <- tau2/tau1
  }",
  file = fn)

# a)
task5.data <- list(y1 = c(4.3, 4.3, 2.7, 3.6, 3.5, 4.5),
                   y2 = c(3.9, 4.0, 4.5, 2.9, 5.2, 4.8),
                   v = 0.2,
                   N = 6)
task5.init <- list(mu=4.7, sigma1=1, sigma2=1)
task5.model <- jags.model(file = fn, data = task5.data, inits = task5.init,
n.chains = 3, n.adapt = 2000)

task5.pars <- c("mu", "phi", “tau1”, “tau2”)
task5.postInfo <- coda.samples(model = task5.model, variable.names = task5.pars,
                                 n.iter = 100000, thin = 10)

summary(task5.postInfo)
plot(task5.postInfo)

Code output:

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

       Mean     SD  Naive SE Time-series SE
mu   4.0682 0.1520 0.0008777      0.0008777
phi  1.4554 0.9384 0.0054178      0.0054354
tau1 0.2716 0.1312 0.0007572      0.0007676
tau2 0.3294 0.1493 0.0008619      0.0008619

2. Quantiles for each variable:

       2.5%    25%    50%    75%  97.5%
mu   3.7774 3.9651 4.0661 4.1685 4.3731
phi  0.3838 0.8260 1.2237 1.8127 3.8848
tau1 0.1170 0.1850 0.2414 0.3224 0.6083
tau2 0.1491 0.2305 0.2951 0.3896 0.7139
```



Inference about $\mu < 4.7$:
Based on 95% CI [3.7785, 4.377] of μ, it can be inferred that it is quite likely that $\mu < 4.7$

Inference about $\phi = \frac{\tau_2}{\tau_1}$:
Based on 95% credible interval and posterior mean of ϕ it may be estimated that τ_2 is roughly equal with τ_1. However, it must also be noted that the sample size was rather small and due to this, the prior distributions of $\tau' s$ will be in dominant role which in this case, are equal. In this sense it may be reasonable to say, that $\tau_2 > \tau_1$ because, although the sample is small, there is still slight difference between $\tau' s$ posterior information, even tought their prior distributions are equal.

b.)
When v grows indefinitely, the prior distribution of μ becomes more and more noninformative and thus the posterior distribution of μ becomes more and more data driven, a.k.a. the posterior mean of μ converges towards sample mean. This can be noted when sampling from posterior distribution at the same time when increasing values for v. Below are some numbers regarding to this method:

```
              v  mu_mean
1  1.500000e-01 4.092256
2  5.000000e-01 4.023562
3  1.000000e+00 4.006566
4  5.000000e+00 3.991787
5  1.000000e+01 3.991407
6  5.994843e+01 3.988174
7  3.593814e+02 3.988363
8  2.154435e+03 3.988377
9  1.291550e+04 3.988715
10 7.742637e+04 3.987182
11 4.641589e+05 3.987386
12 2.782559e+06 3.988744
13 1.668101e+07 3.988994
14 1.000000e+08 3.989758
```

c.)
Based on examination, it can be noted that the new process produces less wastage, compared to the standard process. Average wastage from standard process was measured to be 4.7% but by using updated process, the average wastage percentage was measured to be 4.1%. To made a more specific summary about updated process, it was measured that by 95% probability the wastage percentage falls between 3.78% and 4.38%.
When comparing the variances, it may be reasonable to assume that the variance regarding to operator 2 might be slightly larger than variance regarded to operator 1. However, the sample of given measurement data was rather small. To obtain more secure inferences about variances, more measurement data would be needed.