TDDE07: Lab 1 report

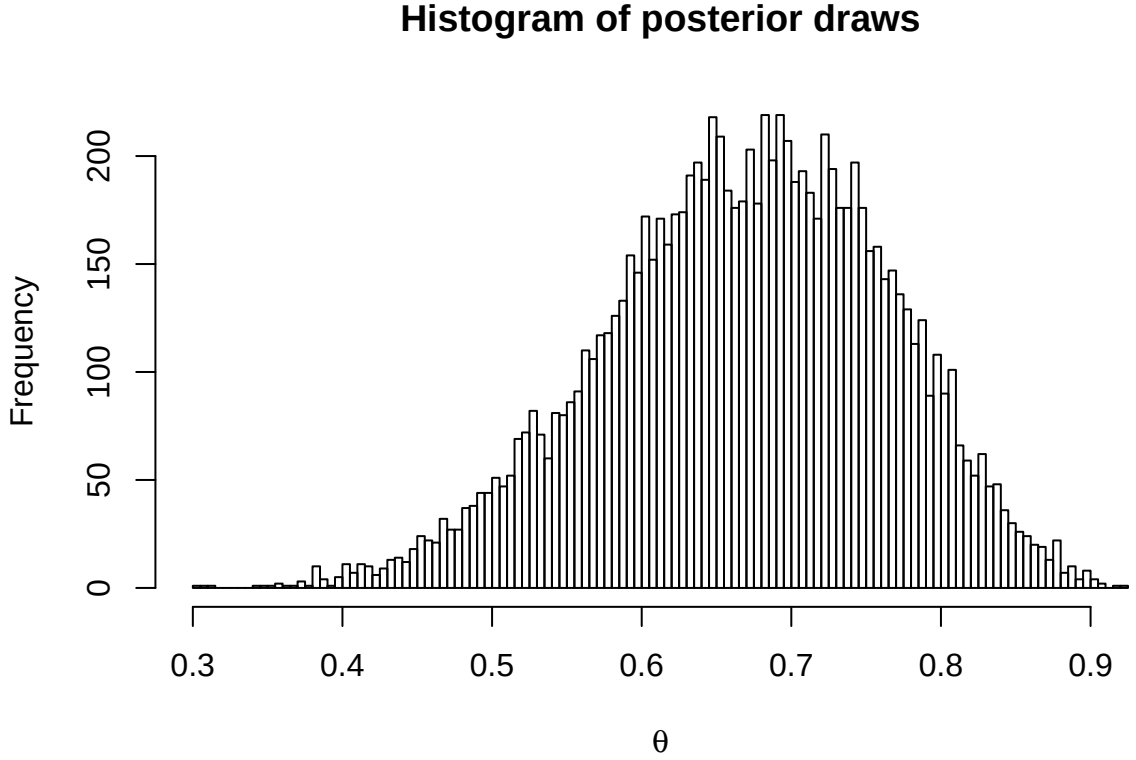
Sebastian Callh (sebca553), Jacob Lundberg (jaclu010)

10 april 2018

1. Bernoulli ... again

a)

The mean of a Bernoulli distribution is given by $p = \frac{s}{n} = \frac{14}{20} \approx 0.7$ which is very close to the mean of the simulated posterior distribution plotted below.



The posterior mean and the true mean are close to each other and increasing the number of draws will let them come arbitrarily close.

```
## [1] "Mean of posterior draws 0.668710"
```

b)

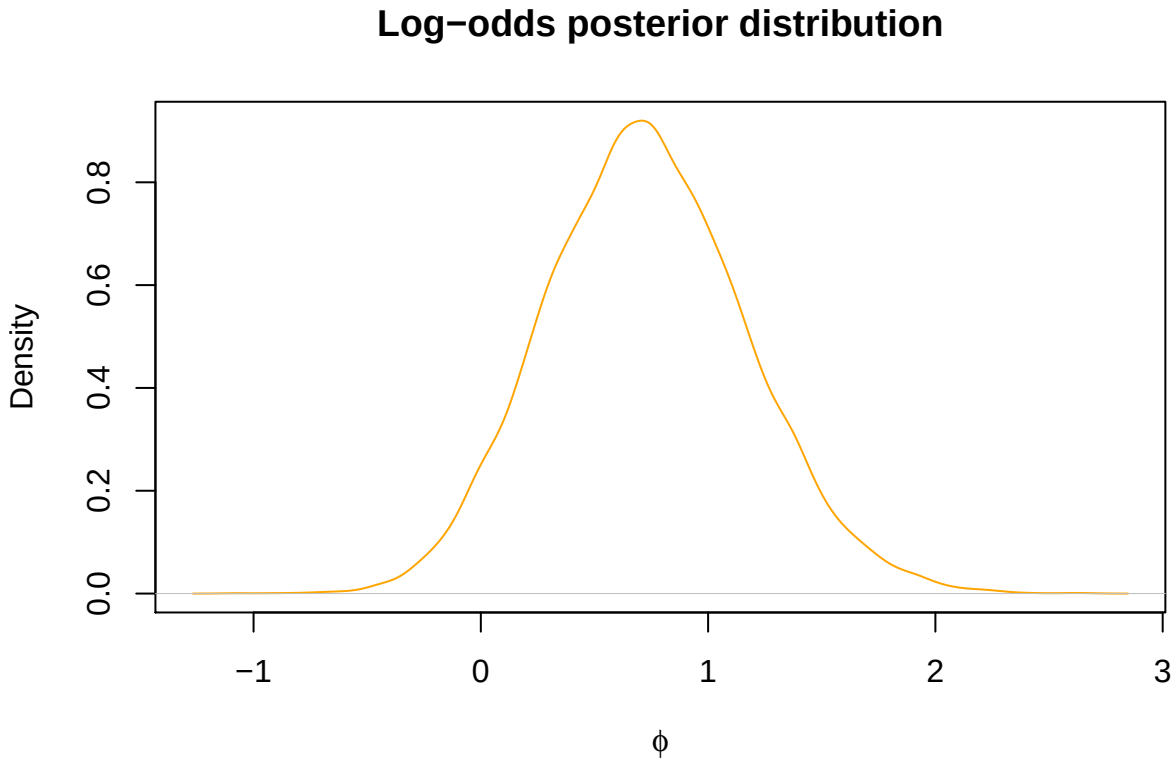
The posterior probability and the true probability for $P(\theta < 0.4)$ lie very close to each other. As with the mean value, increasing the number of draws will let the posterior probability approach the true probability.

```
## [1] "Simulated probability of theta < 0.4: 0.003200"
```

```
## [1] "Exact probability of theta < 0.4: 0.003973"
```

c)

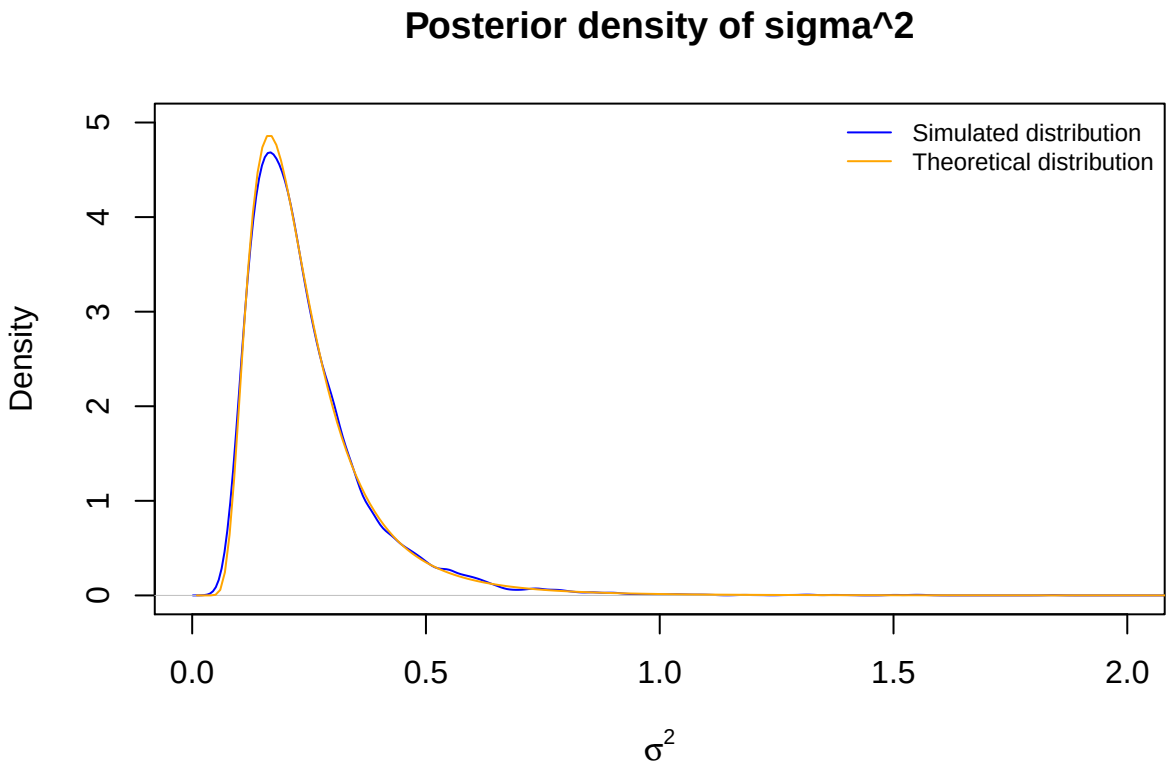
The log-odds posterior distribution can be seen in the plot below.



2. Log-normal distribution and the Gini coefficient.

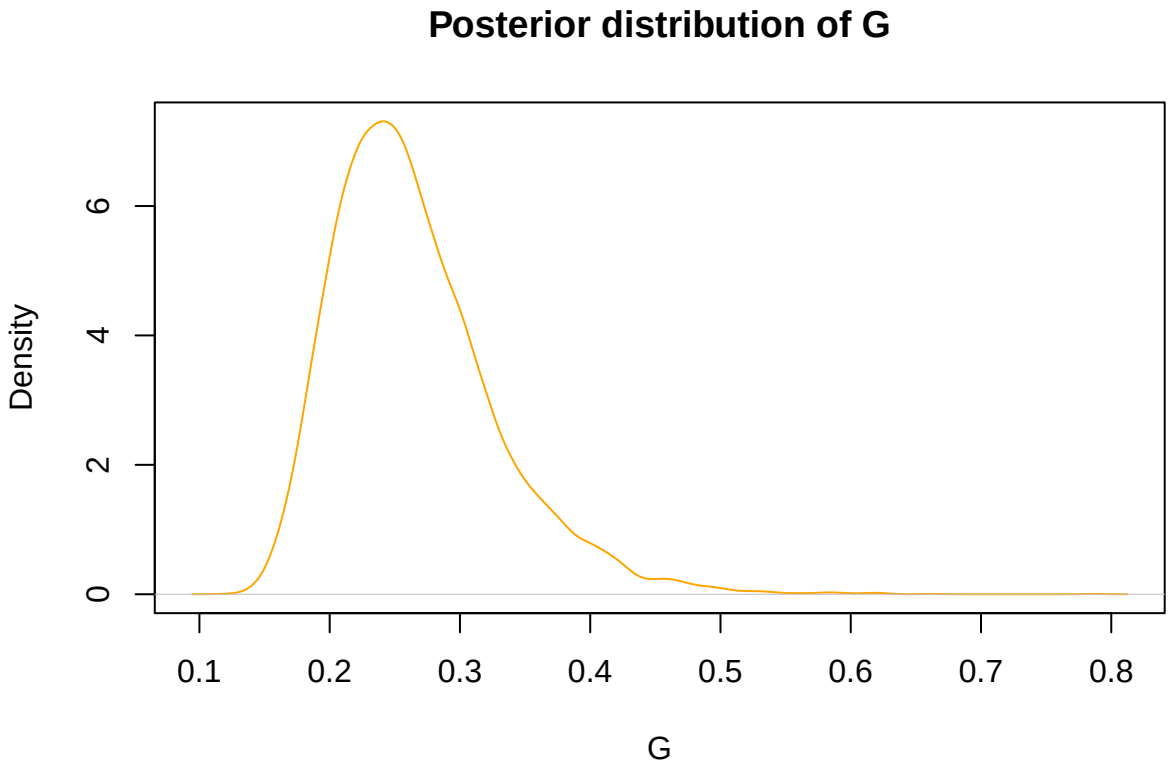
a)

Simulating from the posterior and plotting the approximated density together with the theoretical density reveals that the approximation is quite good, which can be seen in the plot below.



b)

Using the samples from a) and computing the posterior distribution of the Gini coefficient G results in the plot below.



c)

Computing the 95% credible interval results in

```
## [1] "Credible interval lower bound: 0.172934"
```

```
## [1] "Credible interval upper bound: 0.416341"
```

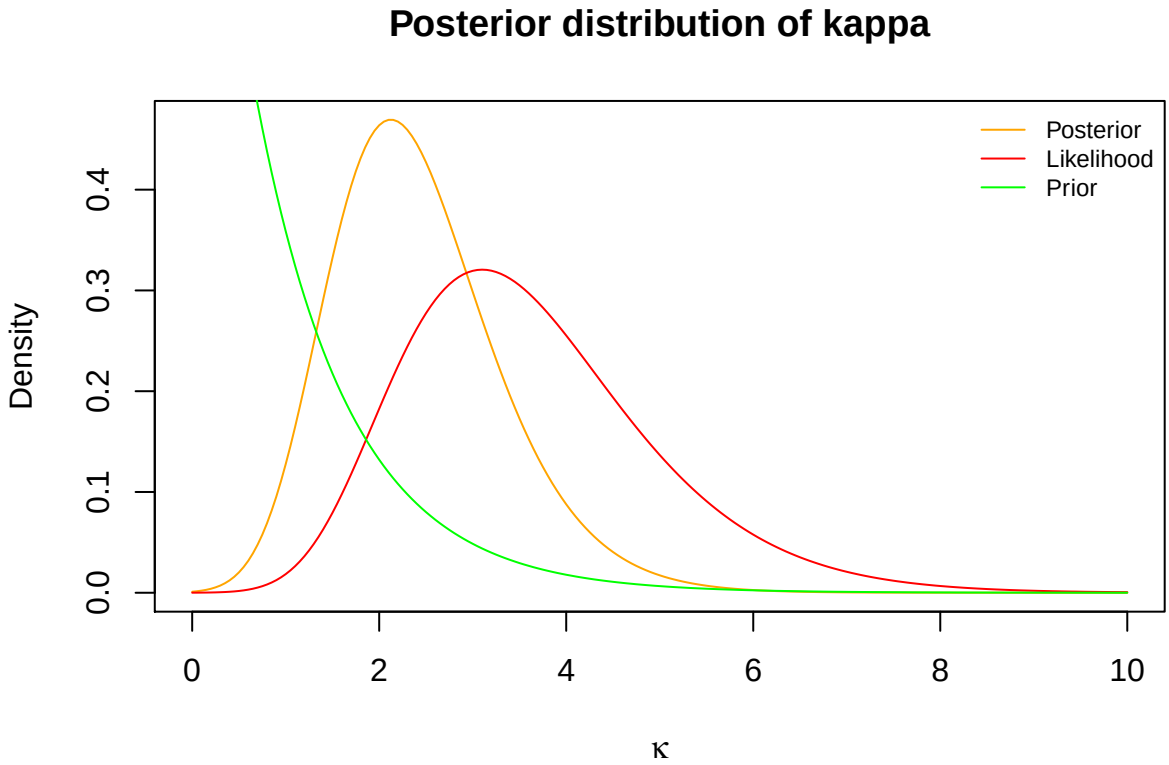
and computing the Highest Posterior Density interval results in

```
## [1] "HPD lower bound: 0.159075"
```

```
## [1] "HPD upper bound: 0.392269"
```

3. Bayesian inference for the concentration parameter in the Von Mises distribution

Plotting the posterior distribution of κ together with the prior and likelihood results in the plot below.



Which has it's mode at

```
## [1] "Posterior mode: 2.10"
```

```r
# 1
# a) Show that posterior converges to 14/20 approx. 0.7
alpha0 <- 2
beta0  <- 2
nDraws <- 10000
n      <- 20
s      <- 14
f      <- n - s
post   <- rbeta(nDraws, alpha0 + s, beta0 + f)
hist(post,
     breaks = 100,
     main   = "Histogram of posterior draws",
     xlab   = expression(theta))

sprintf("Mean of posterior draws %f", mean(post))

# b) Simulation to compute posterior probability P(theta < 0.4 | y)
thetas <- rbeta(nDraws, alpha0 +s , beta0 + f)
sprintf("Simulated probability of theta < 0.4: %f", length(thetas[thetas<0.4]) / length(thetas))  # Sim
sprintf("Exact probability of theta < 0.4: %f", pbeta(0.4, alpha0 + s, beta0 + f))                # Exa

# c) Log odds posterior distribution
phi <- sapply(thetas, function(theta) { log(theta / (1 - theta)) })
phi.dens <- density(phi)
plot(phi.dens,
     main = "Log-odds posterior distribution",
     ylab = "Density",
     xlab = expression(phi),
     col  = posterior.col)

# 2
incomes <- c(14, 25, 45, 25, 30, 33, 19, 50, 34, 67)
mu      <- 3.5
n       <- length(incomes)
tau2    <- sum((log(incomes) - mu)^2) / n
nDraws  <- 10000
dinvgamma <- function(x, a, b) {
  (b^a)/gamma(a) * x^(-a-1) * exp(-b/x)
}

dinvchisq2 <- function(x, n, tau2) {
  a <- n / 2
  b <- n*tau2 / 2
  dinvgamma(x, a, b)
}

# a) Simulate posterior draws
X           <- rchisq(nDraws, n)
sigma2      <- n*tau2/X
sigma2.dens <- density(sigma2)

plot(sigma2.dens, col = "blue",
     main = "Posterior density of sigma^2",
```

```
     xlab = expression(sigma^2),
     ylim = c(0,5),
     xlim = c(0,2))

# Theoretical sigmas
delta  <- 0.01
grid   <- seq(0.01, 2.5, delta)
sigmas <- dinvchisq2(grid, n, tau2)
lines(grid, sigmas/(sum(sigmas)*delta), type = 'l', col = "orange")
legend("topright",
       c("Simulated distribution", "Theoretical distribution"),
       col = c("blue", "orange"),
       lty = 1,
       bty='n',
       cex=.75)

# b) Gini coefficient posterior
G      <- 2*pnorm(sqrt(sigma2/2)) - 1
#G.hist <- hist(G, breaks = 100, freq = FALSE, xlim = 0:1)

# c) G Credible interval
G.slice <- sort(G)[(0.025*nDraws):(0.975*nDraws)]
cred.lower.bound <- min(G.slice)
cred.upper.bound <- max(G.slice)
G.dens <- density(G)
plot(G.dens,
     main = "Posterior distribution of G",
     xlab = "G",
     col  = posterior.col)

sprintf("Credible interval lower bound: %f", cred.lower.bound)
sprintf("Credible interval upper bound: %f", cred.upper.bound)

# c) G Highest Posterior Density (HPD)
i      <- order(G.dens$y, decreasing = TRUE)
csum   <- cumsum(G.dens$y[i])
m      <- length(csum[csum < sum(G.dens$y)*0.95])
H      <- G.dens$x[i][1:m]
hpd.lower.bound <- min(H)
hpd.upper.bound <- max(H)
sprintf("HPD lower bound: %f", hpd.lower.bound)
sprintf("HPD upper bound: %f", hpd.upper.bound)


# 3

Von.Mises <- function (y, mu, k){
  exp(k*cos(y - mu)) / (2*pi*besselI(k, 0))
}

Y          <- c(-2.44, 2.14, 2.54, 1.83, 2.02, 2.33, -2.79, 2.23, 2.07, 2.02)
mu         <- 2.39
lambda     <- 1
```

```r
delta      <- 0.05
kappas     <- seq(0, 10, delta)
likelihood <- sapply(kappas, function (k) { prod(Von.Mises(Y, mu, k)) })
prior      <- dexp(kappas, lambda)
posterior  <- likelihood * prior

plot(kappas, posterior/(sum(posterior)*delta),
     main = "Posterior distribution of kappa",
     xlab = expression(kappa),
     ylab = "Density",
     col  = posterior.col,
     type = 'l')
lines(kappas, likelihood/(sum(likelihood)*delta), col ="red")
lines(kappas, prior/(sum(prior)*delta), col ="green")
legend("topright",
       c("Posterior", "Likelihood", "Prior"),
       lty=1,
       col=c(posterior.col, likelihood.col, prior.col),
       bty='n',
       cex=.75)

sprintf("Posterior mode: %.2f", kappas[which.max(posterior)])
```

TDDE07 - Lab 2

Sebastian Callh, Jacob Lundberg

18 april 2018

Assignment 1 - Linear and polynomial regression

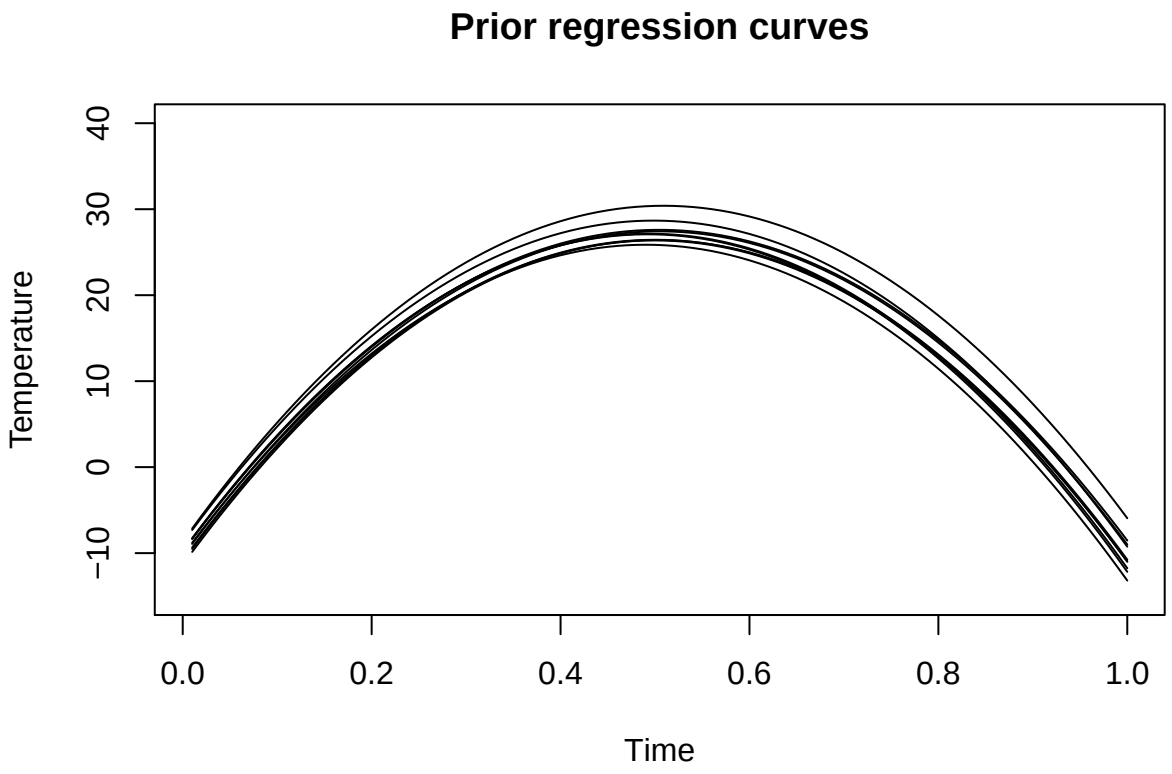
a)

When selecting the prior hyperparameters we had some domain knowledge from the course TDDE01, where we had processed weather data before. From that we knew that the mean temperature in Sweden was about 8 degrees. Of course, we also knew that it is warmer during the middle half of the year than the earlier/later. Apart from that we knew very little, so in the light of that knowledge we assigned our priors by iteratively plotting a collection of prior regression curves and tweaking our values. The final paremetrisation was

$$\mu_0 = (-10, 150, -150), \nu_o = 20, \Omega_0 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \sigma^2 = 2.$$

A collection of prior prediction curves from the final parametrisation can be seen below.

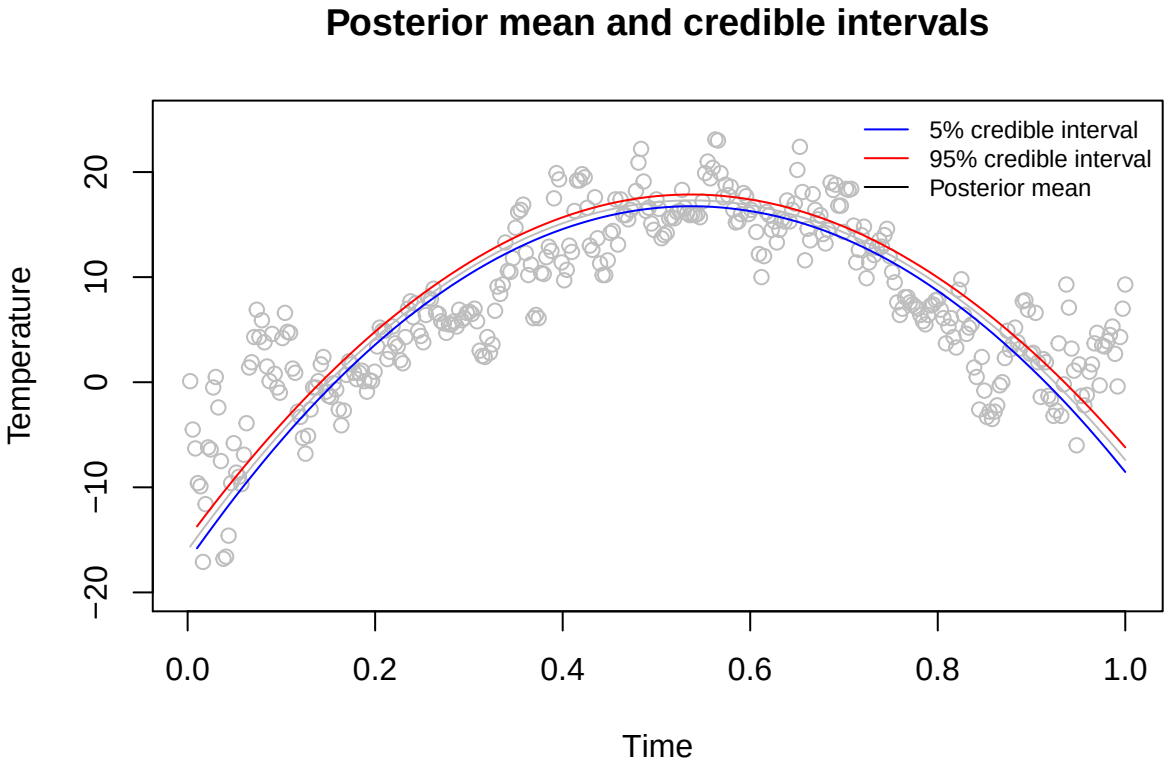
b)



c)

Since a conjugate prior was used the joint posterior distribution of β and σ^2 was computed by the posterior mapping. A scatter plot with the temperature data and the regression function mean with a 90% credible

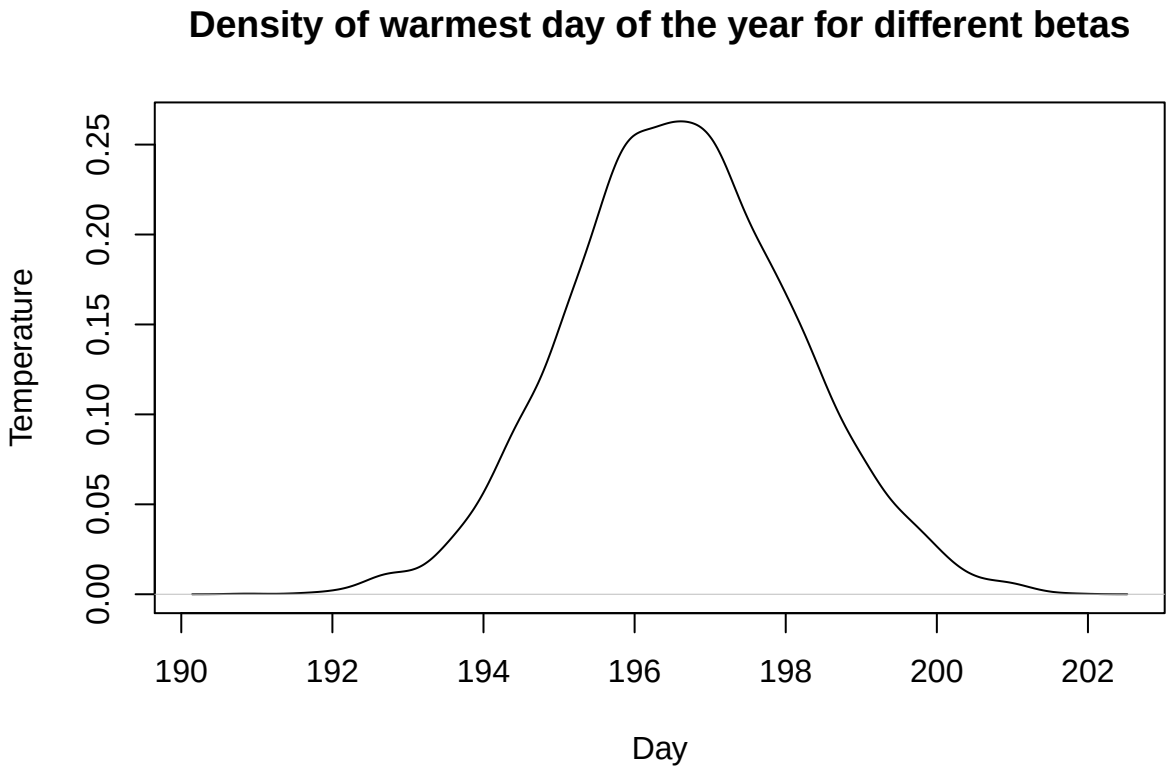
interval can be seen below.



The interval barely contains any data. It should not, since it only quantifies the uncertainty about the data mean and not the actual data.

d)

Finding the warmest day is done by finding the maximum of a second degree polynomial. This is done by differentiating the function, setting it to 0 and solving for t which gives $t = -\frac{\beta_1}{2\beta_2}$. Computing the maximum for several posterior draws of β yields the distribution plotted below which has the mean value $196.6476 \approx 197$.



e)

Since the higher order terms are not believed to be needed they can be assigned $\mu_i = 0$ and $\Omega_{0,ii}$ very large. That is, we assign the prior formalizing our beliefe that they should be very small. That will reduce the models flexibility and combat overfitting.

2. Posterior approximation for classification with logistic regression

a)

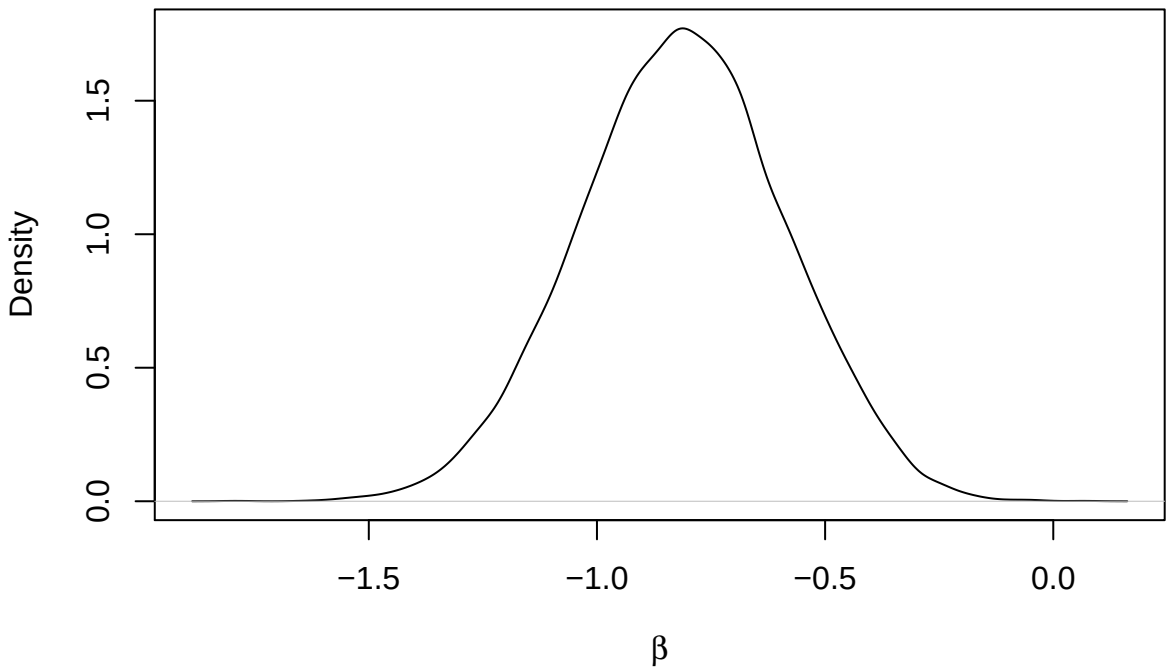
No questions asked on this part.

b)

Using `optim.R` to find $\hat{\beta} = (0.38, -0.01, 0.11, 0.10, -0.09, -0.05, -0.82, -0.01)$ and

$$J_y^{-1}(\hat{\beta}) = \begin{bmatrix} -105.78 & -2216.83 & -1311.95 & -1047.17 & -156.42 & -4451.79 & -27.35 & -137.88 \\ -2216.83 & -60854.93 & -28817.20 & -21918.24 & -3196.04 & -94428.20 & -542.68 & -2776.39 \\ -1311.91 & -28817.20 & -16881.39 & -12957.13 & -1907.13 & -55135.90 & -361.45 & -1657.08 \\ -1047.17 & -21918.24 & -12957.13 & -15641.80 & -2930.44 & -46403.13 & -208.47 & -1061.18 \\ -156.42 & -3196.04 & -1907.13 & -2930.43 & -645.15 & -7243.13 & -21.19 & -124.80 \\ -4451.79 & -94428.20 & -55135.90 & -46403.13 & -7243.13 & -194210.76 & -930.93 & -5397.63 \\ -27.35 & -542.68 & -361.45 & -208.48 & -21.20 & -930.93 & -34.64 & -40.13 \\ -137.88 & -2776.32 & -1657.09 & -1061.18 & -124.80 & -5397.64 & -40.13 & -368.02 \end{bmatrix}$$

The plot below shows the distribution of the `NSmallChild` variable. 95% credible interval limits are included.




```
## [1] "Credible interval 2.5 % limit: -1.259324"
```

```
## [1] "Credible interval 97.5% limit: -0.381006"
```

Comparing the posterior mean of the other β show that the β corresponding to `NSmallChild` is the biggest, and because of that `NSmallChild`is an important factor. The actuall numbers can be seen below.

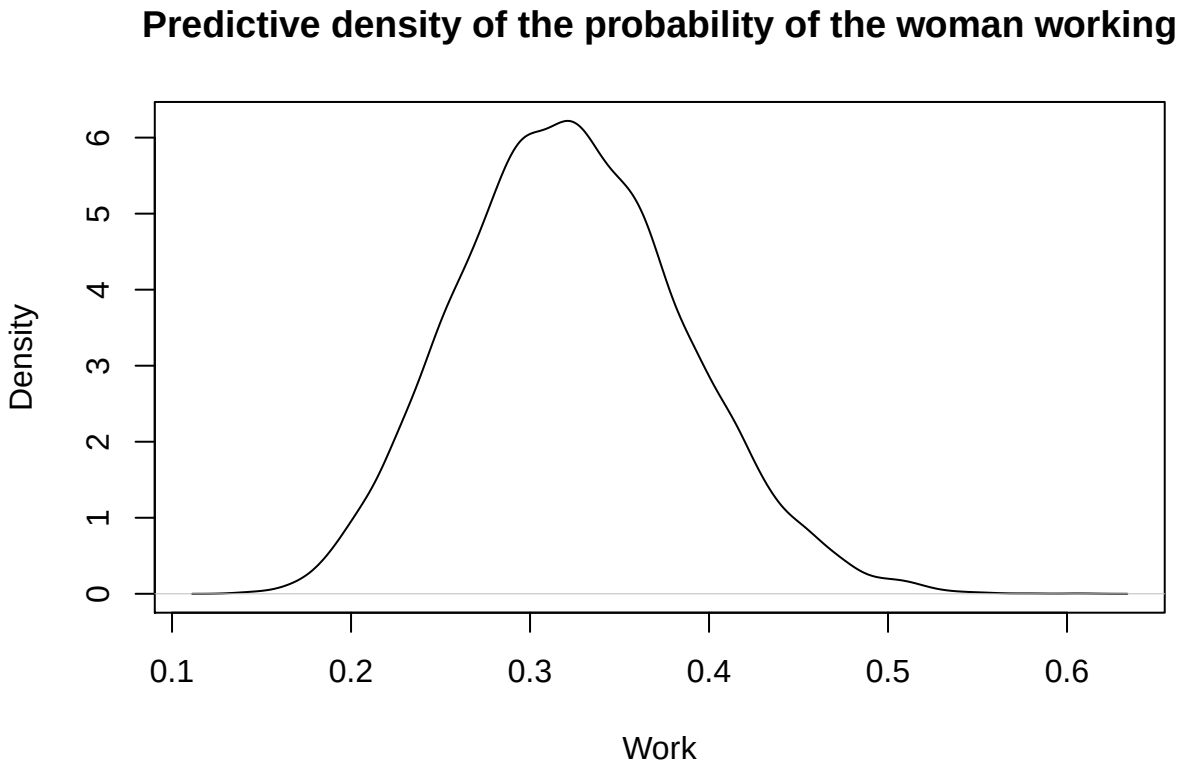
```
##
## Call:
## glm(formula = Work ~ 0 + ., family = binomial, data = data2)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1662  -0.9299   0.4391   0.9494   2.0582
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## Constant     0.64430    1.52307   0.423 0.672274
## HusbandInc  -0.01977    0.01590  -1.243 0.213752
## EducYears    0.17988    0.07914   2.273 0.023024 *
## ExpYears     0.16751    0.06600   2.538 0.011144 *
## ExpYears2   -0.14436    0.23585  -0.612 0.540489
## Age         -0.08234    0.02699  -3.050 0.002285 **
## NSmallChild -1.36250    0.38996  -3.494 0.000476 ***
## NBigChild   -0.02543    0.14172  -0.179 0.857592
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 277.26  on 200  degrees of freedom
```

```
## Residual deviance: 222.73  on 192  degrees of freedom
## AIC: 238.73
##
## Number of Fisher Scoring iterations: 4
```

The `Estimate` column shows a significantly larger value for `NSmallChild` compared to the others. Thus it is important.

c)

The predictive distribution that a woman works that is 40 years old, with two children (aged 3 and 9), 8 years of education, 10 years of experience and a husband with income 10 is described by the following plot.




```
## [1] "The mean probability that the woman works is 0.32"
```

```r
library(mvtnorm)
data <- read.csv("TempLinkoping.txt", sep = "\t")
X <- cbind(rep(1, length(data$time)), data$time, data$time^2)
Y <- data$temp

# 1. Linear and polynomial regression

rinvchisq <- function(vx, sigmax, draws) {
  vx*sigmax/rchisq(draws, vx)
}

#a) Determining the prior distribution
delta <- 0.01
times <- seq(delta, 1, delta)
nDraws  <- 5000
posteriorDraws <- 5000
mu0    <- as.matrix(c(-10, 150, -150))   # From tuning the plotted polynomial
omega0 <- 1*diag(3)           # same variance for betas, we dont know much
v0      <- 20
sigma0  <- 2
sigma2s <- rinvchisq(v0, sigma0, nDraws)#v0*sigma0/rchisq(nDraws, v0)
n       <- dim(X)[1]

glmthing <- lm(temp ~ time + I(time^2), data = data)
summary(glmthing)

# Plot of prior variance
plot.prior.varaince <- function() {
  plot(density(sigma2s),
       main = "Sigma^2 prior",
       xlab = expression(sigma^2),
       ylab = "Density",
       type = 'l')
}

# Thetas are matrix of theta - all parameters with priors
thetas <- cbind(t(sapply(sigma2s, function(sigma2) {
  rmvnorm(1, mu0, sigma2*solve(omega0))
})), sigma2s)

# y is the prior distribution of the model
prior.temp <- apply(thetas[seq(1, 10),], 1, function(theta) {
  sapply(times, function(time) {
    theta[1] + theta[2]*time + theta[3]*time^2 #+ rnorm(1, 0, theta[4])
  })
})

#b) Check if prior distribution is sensible
m <- dim(prior.temp)[1]
x.axis <- (1:m) / m

plot.prior.regression.curves <- function () {
  plot(x.axis, prior.temp[,1],
```

```
      type = 'l',
      ylim = c(-15, 40),
      xlab = "Time",
      ylab = "Temperature",
      main = "Prior regression curves")
  for (i in 2:10) {
    lines(x.axis, prior.temp[,i])
  }
}

# c) Simulating from posterior distribution

# Posterior mapping
beta.hat <- solve(t(X)%*%X)%*%t(X)%*%Y # OLS
A   <- t(X) %*% X
B   <- as.numeric(t(Y) %*% Y)
mun      <- solve(A + omega0) %*% (A%*%beta.hat + omega0%*%mu0)
omegan   <- A + omega0
vn       <- v0 + n
sigma2n  <- as.numeric((v0%*%sigma0 + (B + t(mu0)%*%omega0%*%mu0 - t(mun)%*%omegan%*%mun))/vn)

post.sigma2s <- rinvchisq(vn, sigma2n, posteriorDraws)
post.thetas  <- cbind(t(sapply(post.sigma2s, function(post.sigma2) {
  rmvnorm(1, mun, post.sigma2 * solve(omegan))
})), post.sigma2s)

post.betas <- post.thetas[,1:3]
theta.mean <- apply(post.thetas, 2, mean)
post.pred.mean <- X  %*% as.matrix(theta.mean[1:3])

nSamples <- 5000
bounds <- as.matrix(sapply(times, function(t) {
  ys <- as.matrix(apply(post.betas[1:nSamples,], 1, function(beta) {
    c(1, t, t^2) %*% as.matrix(beta)
  }))

  lower.bound <- nSamples * 0.05
  upper.bound <- nSamples * 0.95
  bounded.y   <- sort(ys)[lower.bound:upper.bound]
  list(upper = head(bounded.y, 1), lower = tail(bounded.y, 1))
}))

plot.posterior.betas <- function() {
  plot(data$time,
       data$temp,
       ylim = c(-20,25),
       col  = "Gray",
       xlab = "Time",
       ylab = "Temperature",
       main = "Posterior mean and credible intervals")
  lines(data$time, post.pred.mean, col = "Gray")
  lines(x=times,y=bounds[1,], col = 'Blue')
  lines(x=times,y=bounds[2,], col = 'Red')
```

```r
  legend("topright",
         c("5% credible interval", "95% credible interval", "Posterior mean"),
         lty=1,
         col=c("Blue", "Red", "Black"),
         bty='n',
         cex=.75)
  }

# d)
x_tilde <- which.max(post.pred.mean)
# Hottest day of the year (# 197)


# Distribution over the hottest day
day <- as.matrix(apply(post.betas, 1, function(beta) {
  (-beta[2] / (2*beta[3]))*n
}))
mean(day)
plot.day.density <- function() {
  plot(density(day),
       main = "Density of warmest day of the year for different betas",
       xlab = "Day",
       ylab = "Temperature")
}

# e)
# mu0: set the last 4 values to 0 to delete the effect of the higher order terms
# omega: set the 4 last values to very large to prevent high variance for the higher order terms


# 2. Posterior approximation for classification with logistic regression

# a)
data2 <- read.table("WomenWork.dat", header=TRUE)
y    <- as.vector(data2[,1])
X    <- as.matrix(data2[,-1])

glmModel <- glm(Work ~0 + ., data=data2, family = binomial)
#summary(glmModel)

# b)
# want posterior
# to get that first we need prior and likelihood

beta.log.posterior <- function(betas, X, y, mu, sigma) {
  d    <- length(betas)
  pred <- X %*% betas

  log.likelihood <- sum(y*pnorm(pred, log.p = TRUE) + (1-y)*pnorm(pred, log.p = TRUE, lower.tail = FALSE
  log.prior      <- dmvnorm(betas, matrix(mu, d, 1), sigma*diag(d), log = TRUE)

  log.likelihood + log.prior
}
```

```r
tau2 <- 10^2
mu   <- 0
d <- dim(X)[2]
beta.initial <- rep(0, d)
result       <- optim(beta.initial, beta.log.posterior, gr = NULL,
                      X, y, mu, tau2,
                      hessian = TRUE,
                      method  = c("BFGS"),
                      control = list(fnscale=-1))

beta_hat <- result$par
J        <- result$hessian

# numerical values should be in the report
beta.post <- rmvnorm(10000, beta_hat, -solve(J))

#NSmallChild
nsc <- beta.post[,7]
q1 <- quantile(nsc, c(0.025,0.975))
plot.density.nsc <- function() {
  plot(density(nsc),
       main = "Posterior density of beta corresponding to NSmallChild",
       xlab = expression(beta))
}

# compute 95% credible interval for NSmallChild?
x <- c(1, 10, 8, 10, 1, 40, 1, 1)
y_hat <- x %*% t(beta.post)
p <- 1 / (1 + exp(-y_hat))
plot.does.work <- function() {
  plot(density(p),
       xlab = "Work",
       main = "Predictive density of the probability of the woman working")
  sprintf("The mean probability that the woman works is %.2f", mean(p))
}
```

TDDE07 - Lab 3

Sebastian Callh, Jacob Lundberg

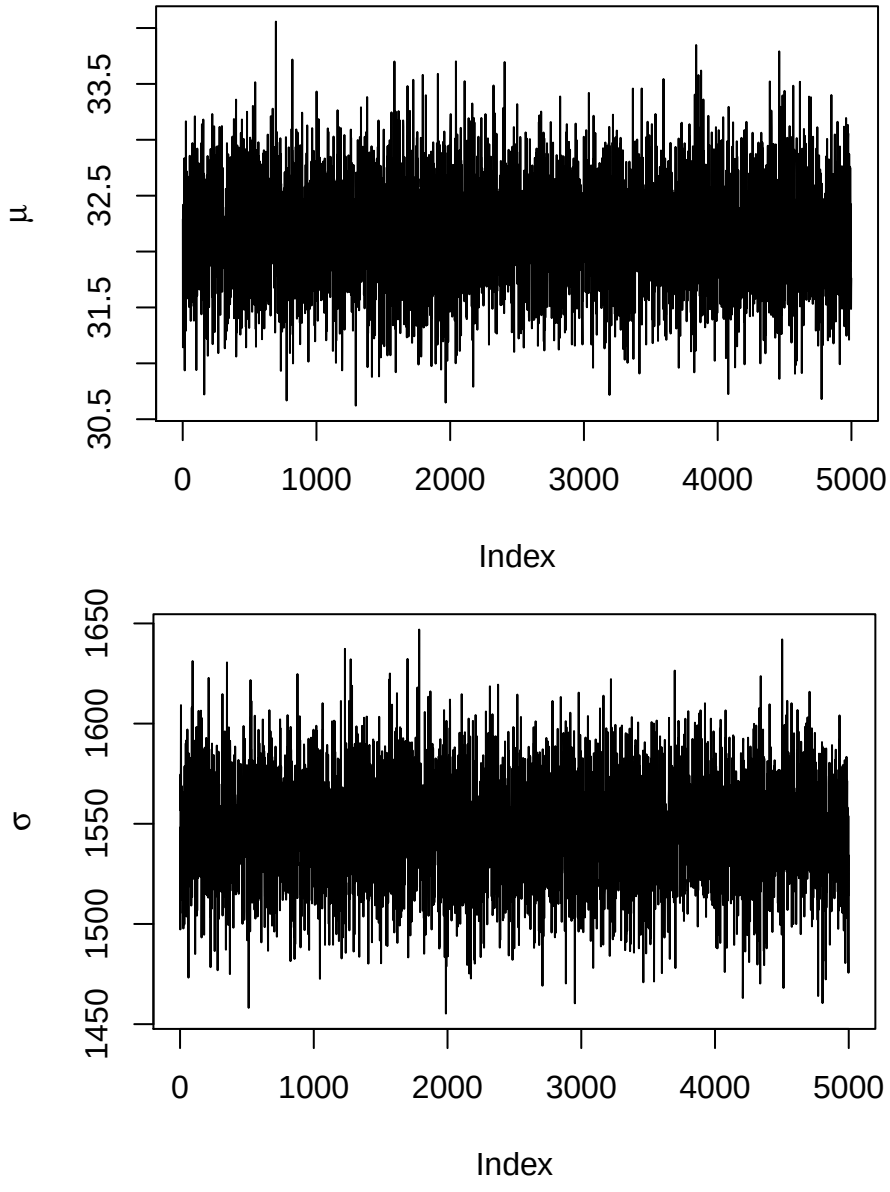
8 maj 2018

1. Normal model, mixture of normal model with semi-conjugate prior

a) - Normal model

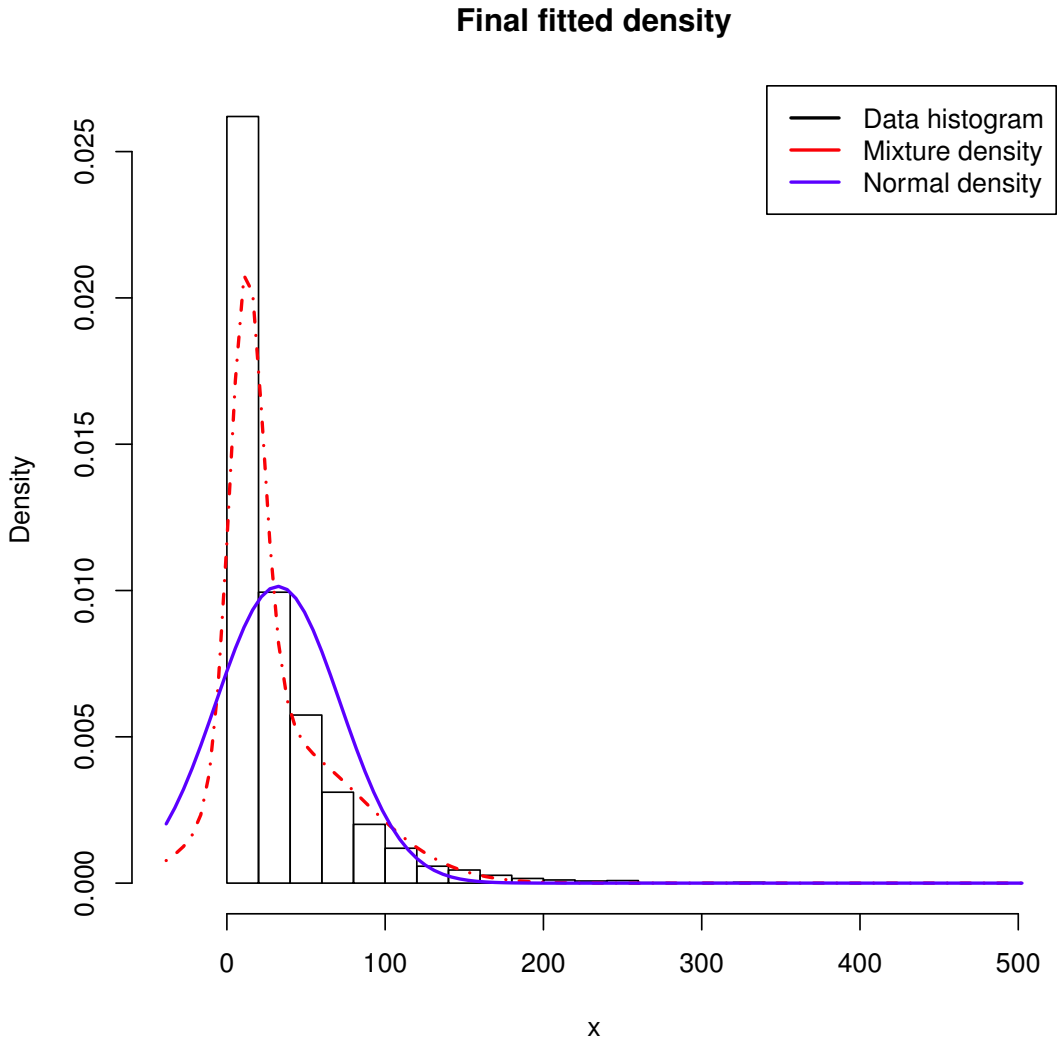
(ii) - Analysis

Looking at the trajectories for μ and σ it can be seen that the values are not very auto correlated, which implies good draws. They also oscillate around a stable value.

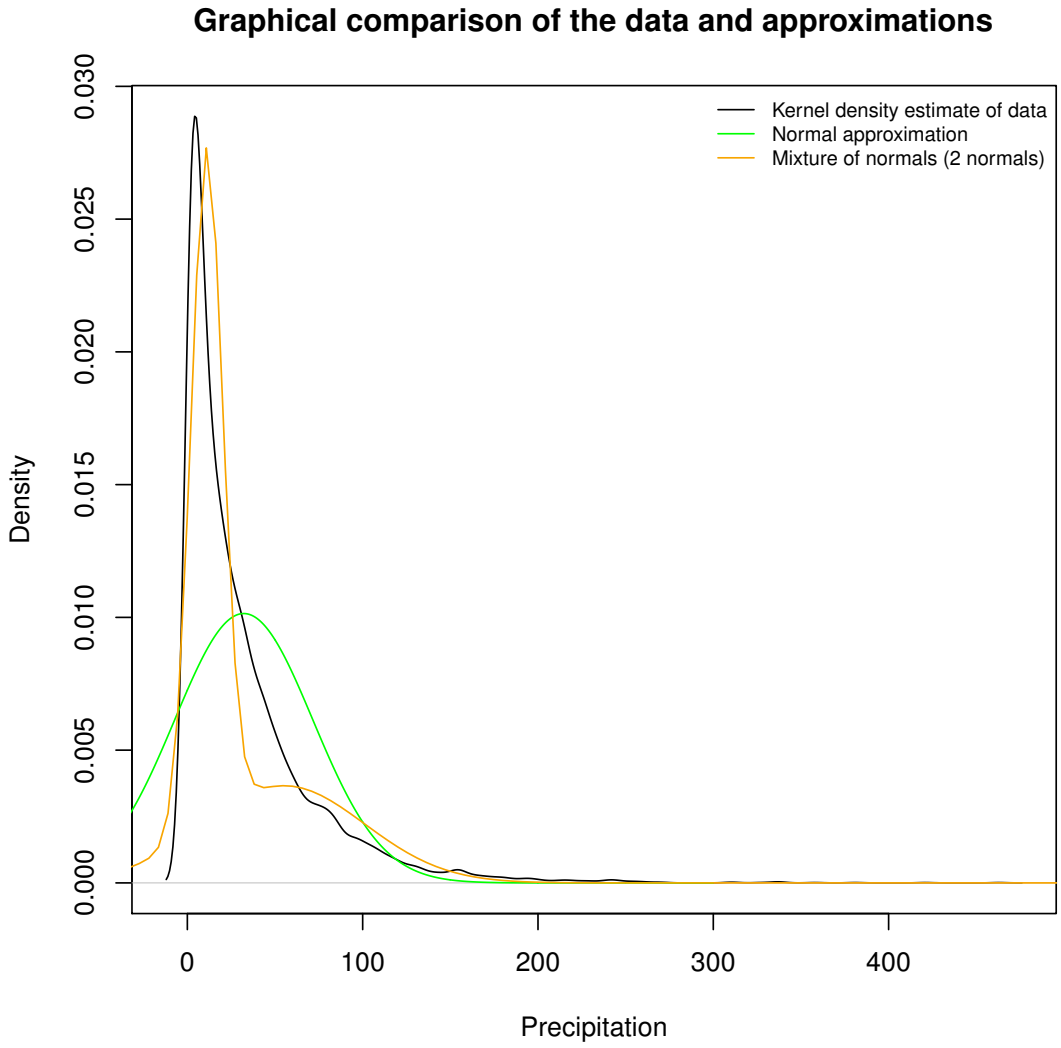


b) Mixture normal model

After running the `NormalMixtureGibbs.R` for 30 iterations the mixture model had converged to the distribution seen in the plot below.



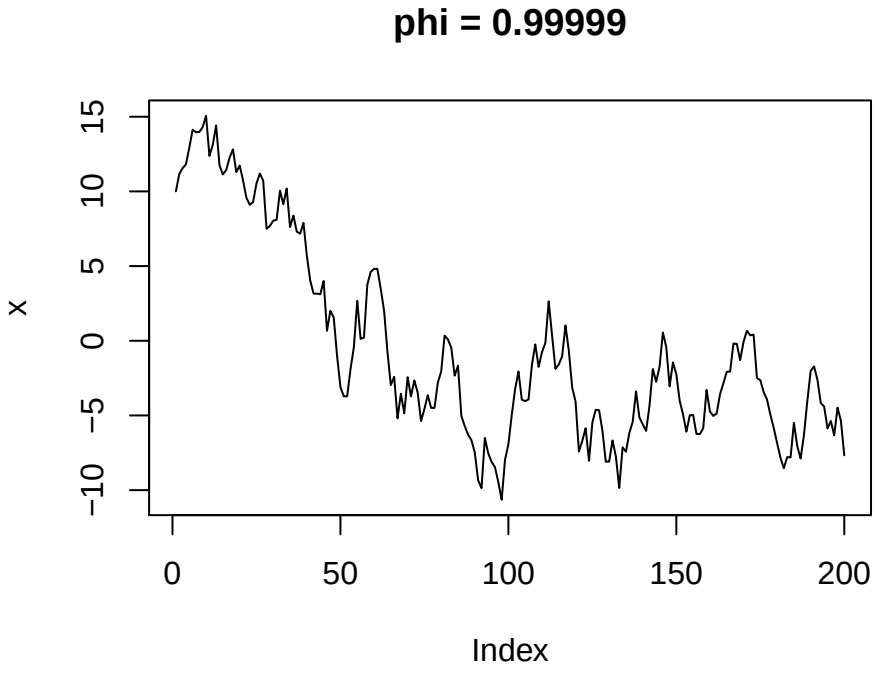
c) Graphical comparison

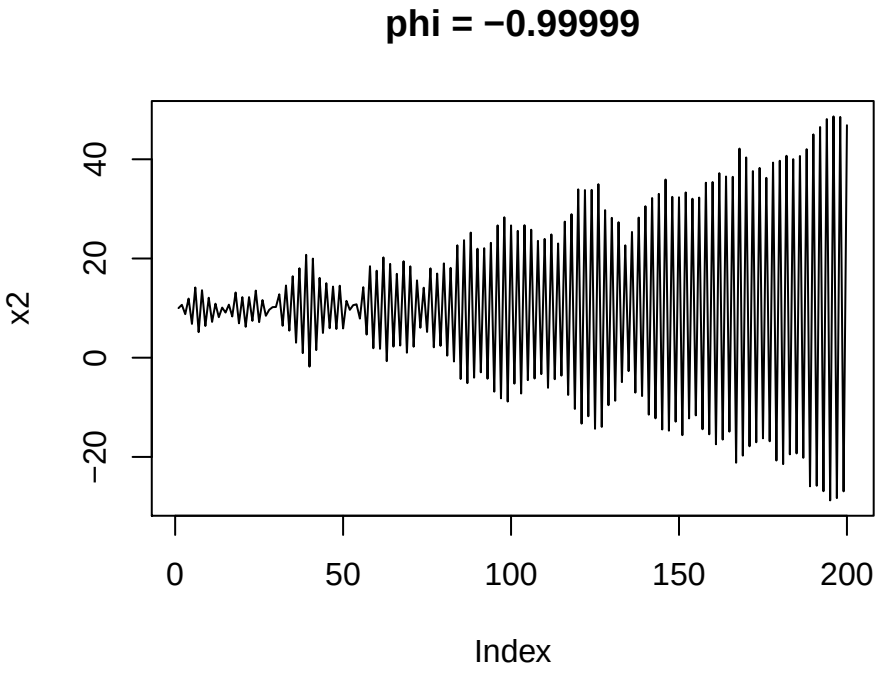


2. Time-series models in Stan

a)

Plotting realizations of the AR(1)-process shows that a high ϕ gives a smoothing effect and a small (into the negatives) gives an oscillating effect. When ϕ is close to zero all contributions of x_{t-1} is supressed and only white noise remains.





b)

i)

Report of the values of the three estimated parameters μ, σ and ϕ can be seen below. For process x the parameters were fairly well approximated (the μ was still quite bad), but for process y the approximated μ was terrible. We were not able to estimate the true value of μ. The estimation was done with 4000 samples.

Table 1: Estimated parameters of AR(1) process X ($\phi = 0.3$)

	μ	σ	ϕ
Mean	7.25	1.38	0.28
2.5%	5.83	1.24	0.14
97.5%	8.62	1.53	0.42
Eff. samples	1108	1450	1098
True value	10	1.41	0.3

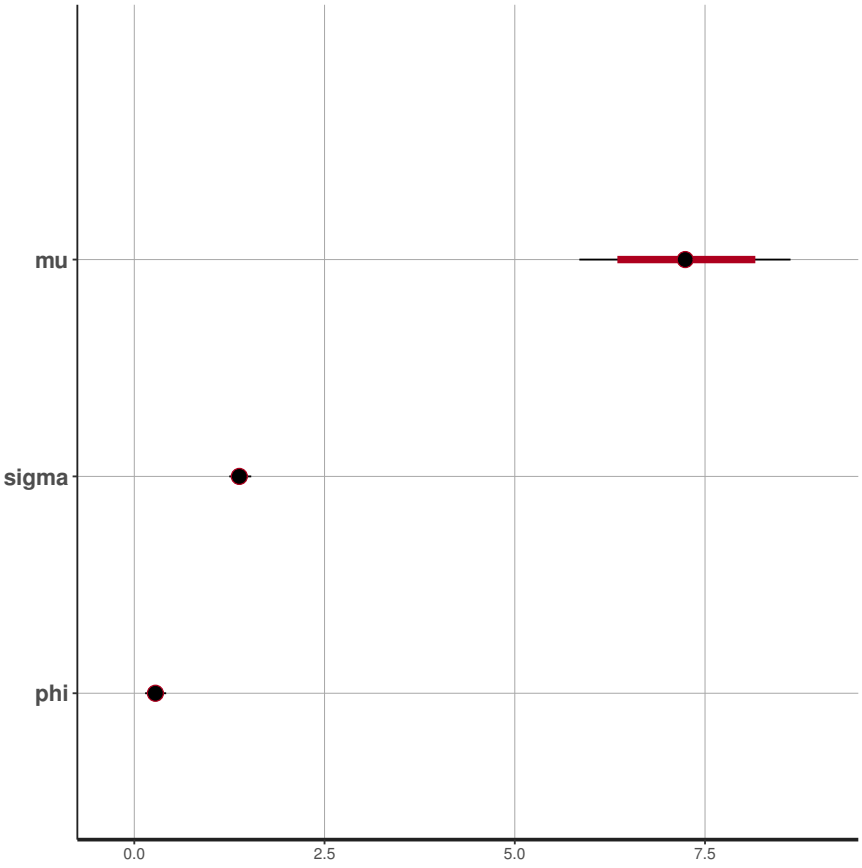
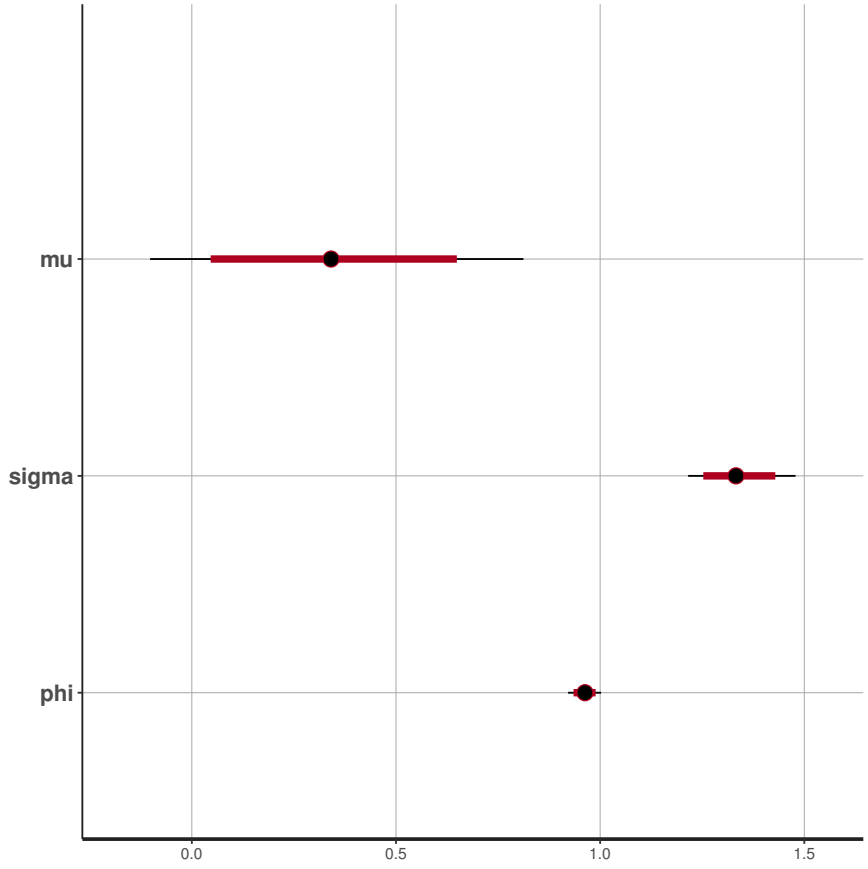


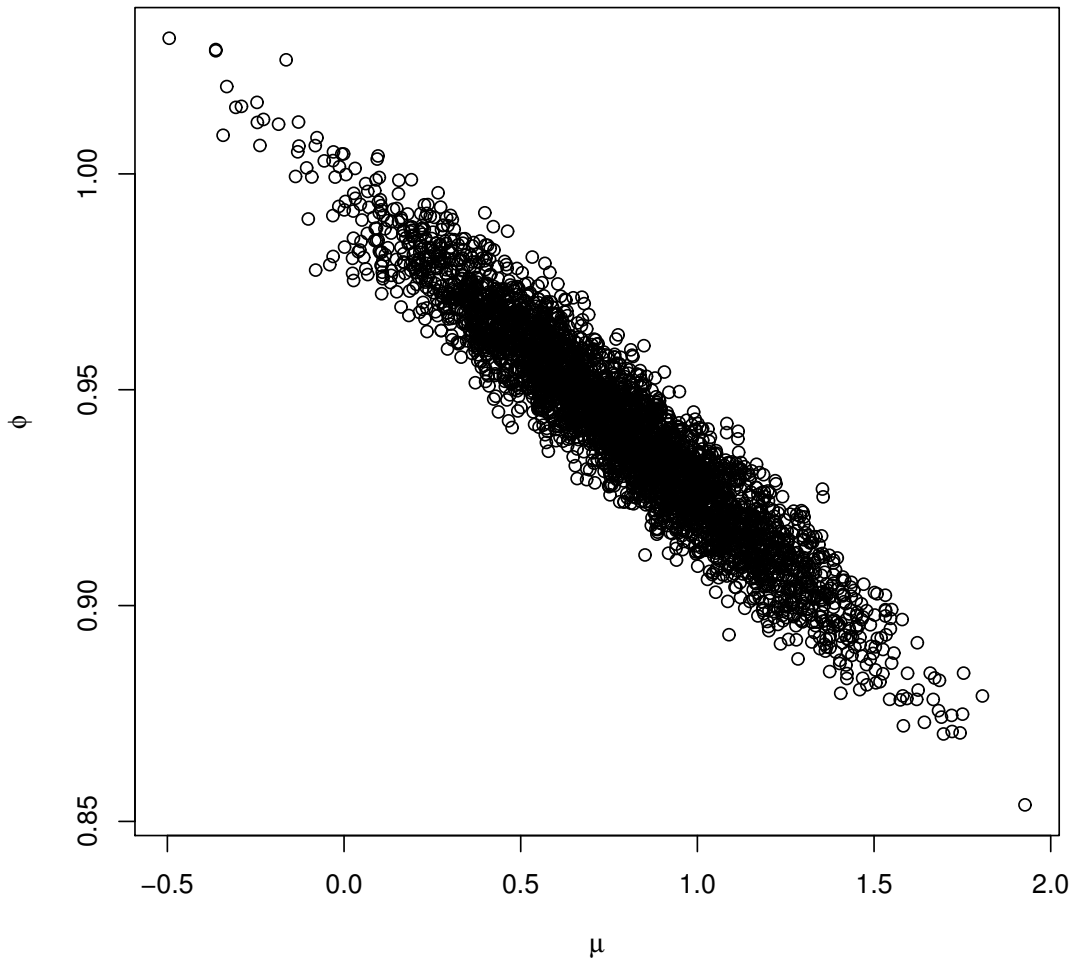
Table 2: Estimated parameters of AR(1) process Y ($\phi = 0.95$)

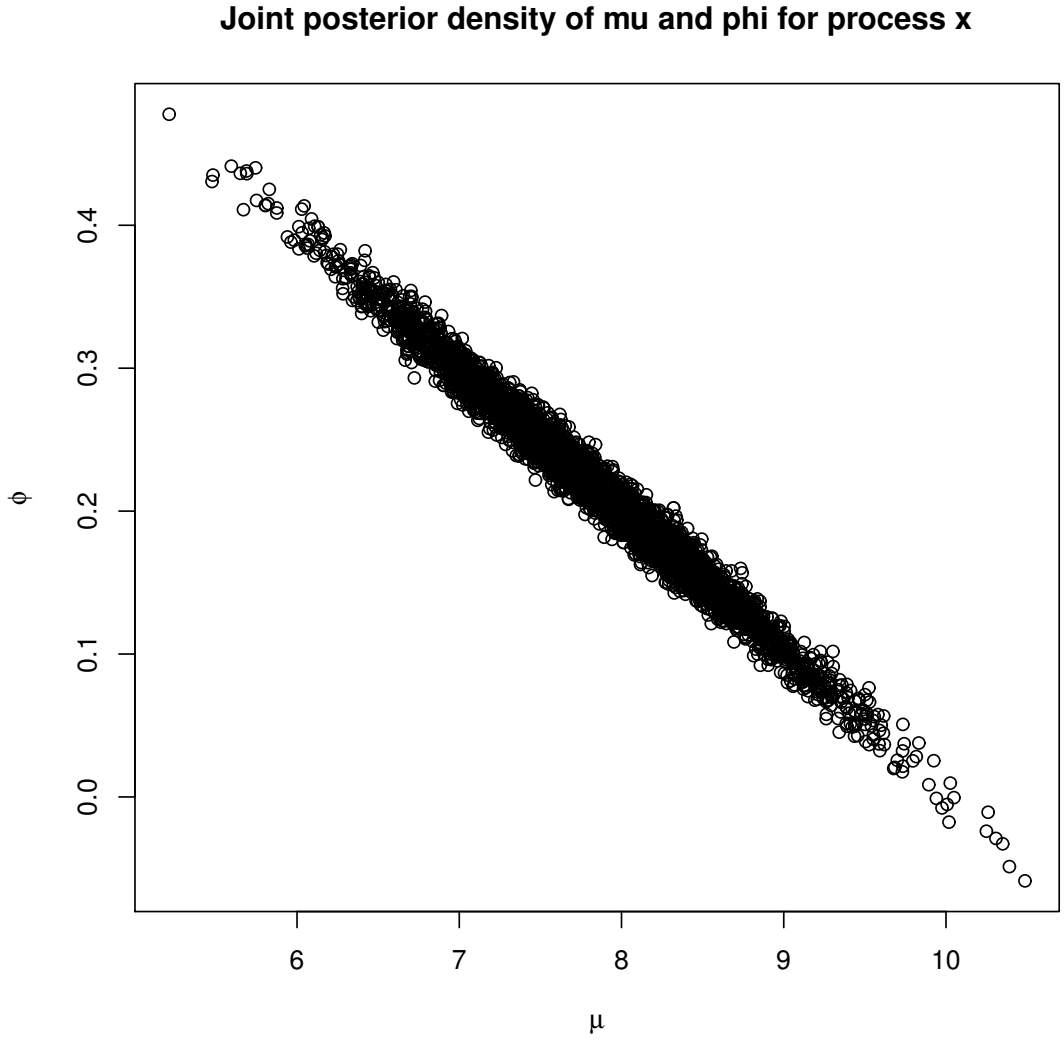
	μ	σ	ϕ
Mean	0.35	1.34	0.96
2.5%	-0.10	1.22	0.92
97.5%	0.81	1.48	1.00
Eff. samples	1726	2166	1729
True value	10	1.41	0.95



ii)

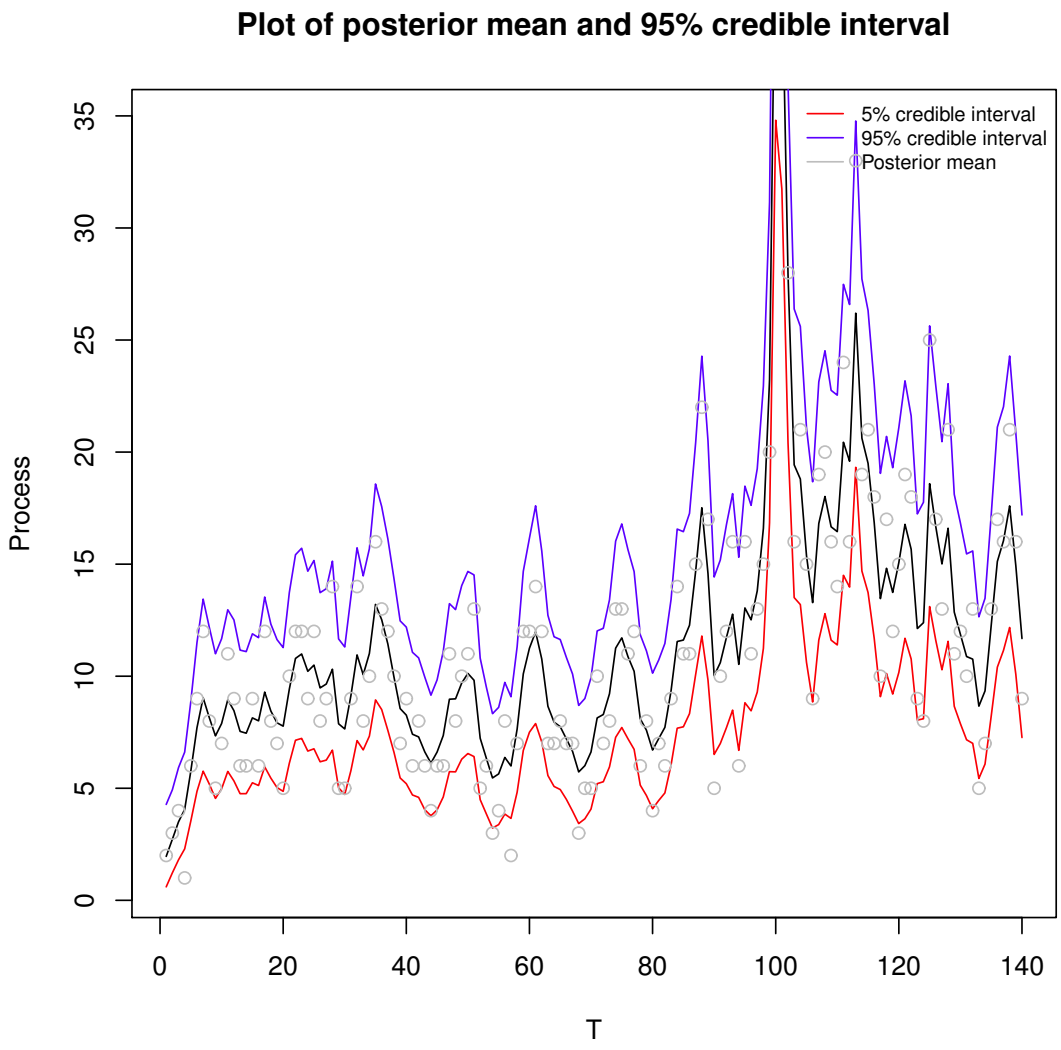
Plotting the joint posterior distributon of μ and ϕ shows that they are highly inversely correlated.


Joint posterior density of mu and phi for process y
ϕ
1.00
0.95
0.90
0.85
−0.5
0.0
0.5
1.0
1.5
2.0
μ



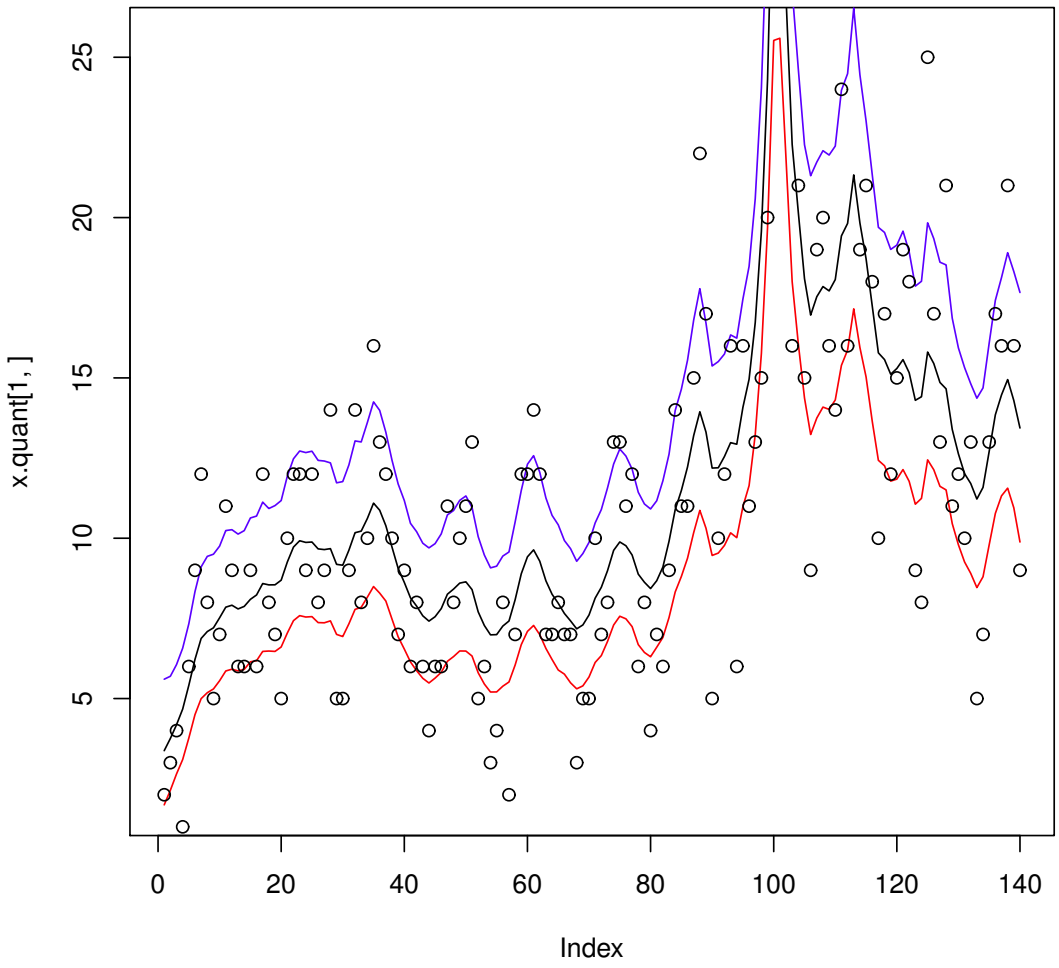
c)

Estimating the model with Stan and plotting the posterior mean with the data resulted in the following plot.



d)

Assuming a normal prior on σ^2 with $\mu = 0$, $\tau^2 = 0.02$ forces the process to vary more slowly. This is reflected in the posterior, which shows that the process is too slow to capture the data.


x.quant[1,]
5
10
15
20
25
0
20
40
60
80
100
120
140
Index

Appendix

```r
# 1 Normal model, mixture of normal model with semi-conjugate prior

# a) Normal model
data <- read.table("rainfall.dat", header=FALSE)
mu0    <- 20 # 0.5 cm
tau0   <- 5  # Variance of expected rainfall
nu0    <- 3  # Variance of expected value of variance
sigma0 <- 5  # Expected value of variance

nIter  <- 5000
theta0 <- c(20, 2)
x      <- data[,1]
n      <- length(x)

rinvchisq <- function(vx, sigmax, draws) {
  vx*sigmax/rchisq(draws, vx)
}

Mu.Conditional.Posterior.Draw <- function(sigma2) {
  taun <- sqrt(1/(n/sigma2 + 1/tau0^2))
  w    <- (n/sigma2) / (n/sigma2 + 1/tau0^2)
  mun  <- w*mean(x) + (1 - w) * mu0
  rnorm(1, mun, taun)
}

Sigma.Conditional.Posterior.Draw <- function(mu) {
  nun    <- n + nu0
  sigman <- (nu0*sigma0^2 + sum((x-mu)^2)) / nun
  rinvchisq(nun, sigman, 1)
}

Gibbs <- function(theta_t) {
  mu     <- Mu.Conditional.Posterior.Draw(theta_t[2])
  sigma2 <- Sigma.Conditional.Posterior.Draw(mu)
  c(mu, sigma2)
}

thetas     <- matrix(rep(0, nIter*2), nrow = nIter)
thetas[1,] <- theta0

for(i in 2:nIter) {
  thetas[i,] <- Gibbs(thetas[i-1,])
}

plot.normal.approximation <- function () {
  plot(thetas[-1,2],
       type = 'l',
       ylab = 'Value')
}

# b) Mixture normal model
```

```r
# Model options
nComp <- 2                  # Number of mixture components

# MCMC options
nIter <- 30 # Number of Gibbs sampling draws

# Prior options
alpha     <- 10*rep(1,nComp)   # Dirichlet(alpha)
muPrior   <- rep(mu0,nComp)    # Prior mean of mu
tau2Prior <- rep(tau0,nComp)   # Prior std of mu
sigma2_0  <- rep(sigma0,nComp) # s20 (best guess of sigma2)
nu0       <- rep(nu0,nComp)    # degrees of freedom for prior on sigma2

#setEPS()
#postscript("mixture-of-normals.eps")
#source("NormalMixtureGibbs.R")
#points(campy.data)
#dev.off()


# c) Graphical comparison

gibbs.mu    <- mean(thetas[,1])
gibbs.sigma <- sqrt(mean(thetas[,2]))
delta <- 0.05
grid  <- seq(-100, 300, delta)

plot.graphical.comparison <- function () {
    plot(density(x), col = 'black',
         main ="Graphical comparison of the data and approximations",
         ylab = "Density",
         xlab  = "Precipitation")
  lines(grid, dnorm(grid, gibbs.mu, gibbs.sigma), type = 'l', col = 'green')
  lines(xGrid, mixDens, type = 'l', col = "orange")
  legend("topright",
       c("Kernel density estimate of data", "Normal approximation", "Mixture of normals (2 normals)"),
       lty=1,
       col=c("black", "green", "orange"),
       bty='n',
       cex=.75)
}

# 2 Time series models in Stan
#library(rstan)
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)

# a)

ar.process <- function (phi, init, sigma, T) {
    y     <- rep (0, T)
    y[1]  <- init
    for (t in 2:T) {
```

```
        y[t] <- mu + phi*(y[t-1] - mu) + rnorm(1, 0, sigma)
    }
    y
}

T     <- 200
mu    <- 10
phi1  <- 0.3
phi2  <- 0.95
s <- sqrt(2)
x <- ar.process(phi1, mu, s, T)
y <- ar.process(phi2, mu, s, T)

plot.ar.process <- function () {
    plot(x, type ='l')
}

# b)

x.fit <- stan(file = "time-series.stan",
            data = list(
                x = x,
                N = T
            ))
y.fit <- stan(file = "time-series.stan",
            data = list(
                x = y,
                N = T
            ))

posterior.mean.x <- get_posterior_mean(x.fit)
posterior.mean.y <- get_posterior_mean(y.fit)

head(posterior.mean.y)
head(posterior.mean.x)
mu.x.post    <- posterior.mean.x[1, 5]
sigma.x.post <- posterior.mean.x[2, 5]
phi.x.post   <- posterior.mean.x[3, 5]
x.params <- extract(x.fit, pars = c("mu", "phi"))
y.params <- extract(y.fit, pars = c("mu", "phi"))
plot(x    = x.params$mu,
     y    = x.params$phi,
     main = 'Joint posterior density of mu and phi for process x',
     xlab = expression(mu),
     ylab = expression(phi))

plot(x    = y.params$mu,
     y    = y.params$phi,
     main = 'Joint posterior density of mu and phi for process y',
     xlab = expression(mu),
     ylab = expression(phi))

mu.y.post    <- posterior.mean.y[1, 5]
```

```
sigma.y.post <- posterior.mean.y[2, 5]
phi.y.post   <- posterior.mean.y[3, 5]

z.x <- ar.process(phi.x.post, mu.x.post, sigma.x.post, T)
z.y <- ar.process(phi.y.post, mu.y.post, sigma.y.post, T)

setEPS()
postscript("estimated-parameters-of-from-y.eps")
plot(y.fit)
dev.off()

setEPS()
postscript("estimated-parameters-of-from-x.eps")
plot(x.fit)
dev.off()

plot.sigma.prior <- function () {
    d <- 0.01
    grid <- seq(0, 2, d)
    plot(grid, dnorm(grid, 0, 0.02),
         type = 'l',
         main = '(Unnormalized) Prior density for sigma',
         xlab = expression(sigma),
         ylab = 'Density'
         )
}

# c)

campy.data <- as.vector(read.table("campy.dat", header = TRUE)[,1])
c.fit <- stan(file ="campy.stan",
              data = list (
                  c = campy.data,
                  N = length(campy.data)
              ))

params <- extract(c.fit, pars = c("mu", "sigma"))
x <- extract(c.fit, pars = "x")

mean(params$sigma)
theta.t <- exp(x$x)
x.mean <- apply(theta.t, 2, mean)
x.quant <- apply(theta.t, 2, quantile, probs=c(0.025,0.975))

#setEPS()
#postscript("posterior-with-sigma-0.02-prior.eps")
plot.ar.posterior.mean <- function () {
plot(x.quant[1,], type = 'l', col='red',
     main = "Plot of posterior mean and 95% credible interval",
     xlab = "T",
     ylab = "Process")
lines(x.quant[2,], type = 'l', col='blue')
lines(x.mean, type = 'l')
```

```
points(campy.data, col = 'grey')
legend("topright",
       c("5% credible interval", "95% credible interval", "Posterior mean"),
       lty=1,
       col=c("red", "blue", "grey"),
       bty='n',
       cex=.75)
}

c.df <- as.data.frame(fit)
head(as.matrix(c.fit)[,1])

# d)

# posterior sigma = 0.2603307 with uniform
# posterior sigma = 0.2481805 with 0.15
# posterior sigma = 0.1074629 with 0.02
```

Stan file

```
data {
  int<lower=0> N;
  vector[N] x;
}

parameters {
  real mu;y <- ar.process(phi2, mu, s, T)
  real<lower=0> sigma;
  real phi;
}

model {
  x[2:N] ~ normal(mu + phi * x[1:(N - 1)], sigma);
}
```

Stan file for `campy.dat`

```
data {
  int<lower=0> N;
  int c[N];
}

parameters {
  real mu;
  real<lower=0> sigma;
  real phi;
  vector[N] x;
}

model {
  // sigma  ~ normal(0, 0.02);
```

```
  phi    ~ normal(0, 0.6);
  x[2:N] ~ normal(mu + phi * x[1:(N - 1)], sigma);
  for (n in 1:N)
    c[n] ~ poisson(exp(x[n]));
}
```

TDDE07 - Lab 4

Sebastian Callh, Jacob Lundberg

16 may 2018

Poisson regression - the MCMC way

a)

Using `glm` to estimate β resulted in the coefficients in the table below.

Table 1: Point estimate of β.

(Intercept)	PowerSeller	VerifyID	Sealed	Minblem	MajBlem	LargNeg	LogBook	MinBidShare
1.072	-0.021	-0.395	0.444	-0.052	-0.221	0.071	-0.121	-1.894

It can be seen that the coefficients `VerifyID`, `Sealed`, `MajBlem`, `MinBidShare` are important, with `MinBidShare` being most important.

b)

Using `optim` to compute the posterior mode (and mean) of the coefficients for the normal approximation resulted in the coefficients in the table below.

Table 2: Point estimate of β.

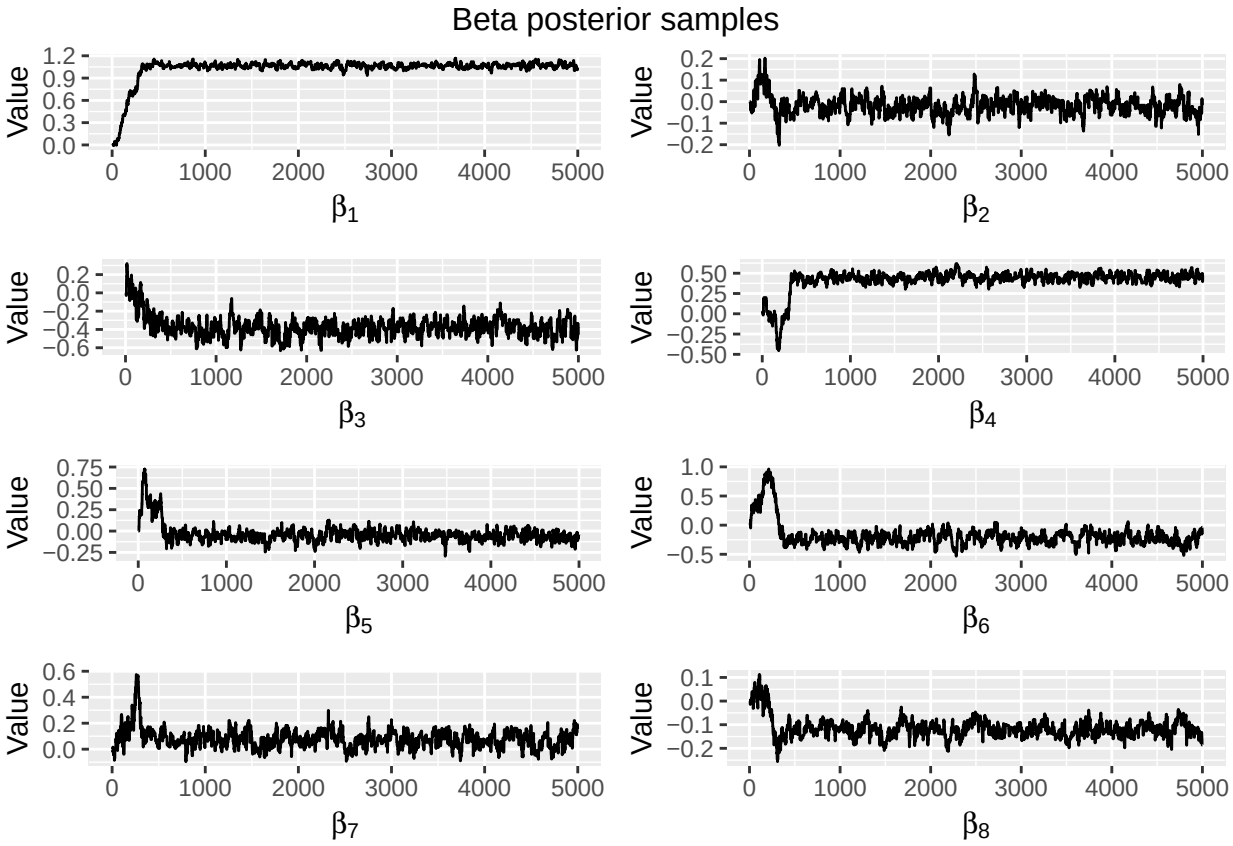
(Intercept)	PowerSeller	VerifyID	Sealed	Minblem	MajBlem	LargNeg	LogBook	MinBidShare
1.070	-0.021	-0.393	0.444	-0.052	-0.221	0.071	-0.120	-1.892

c)

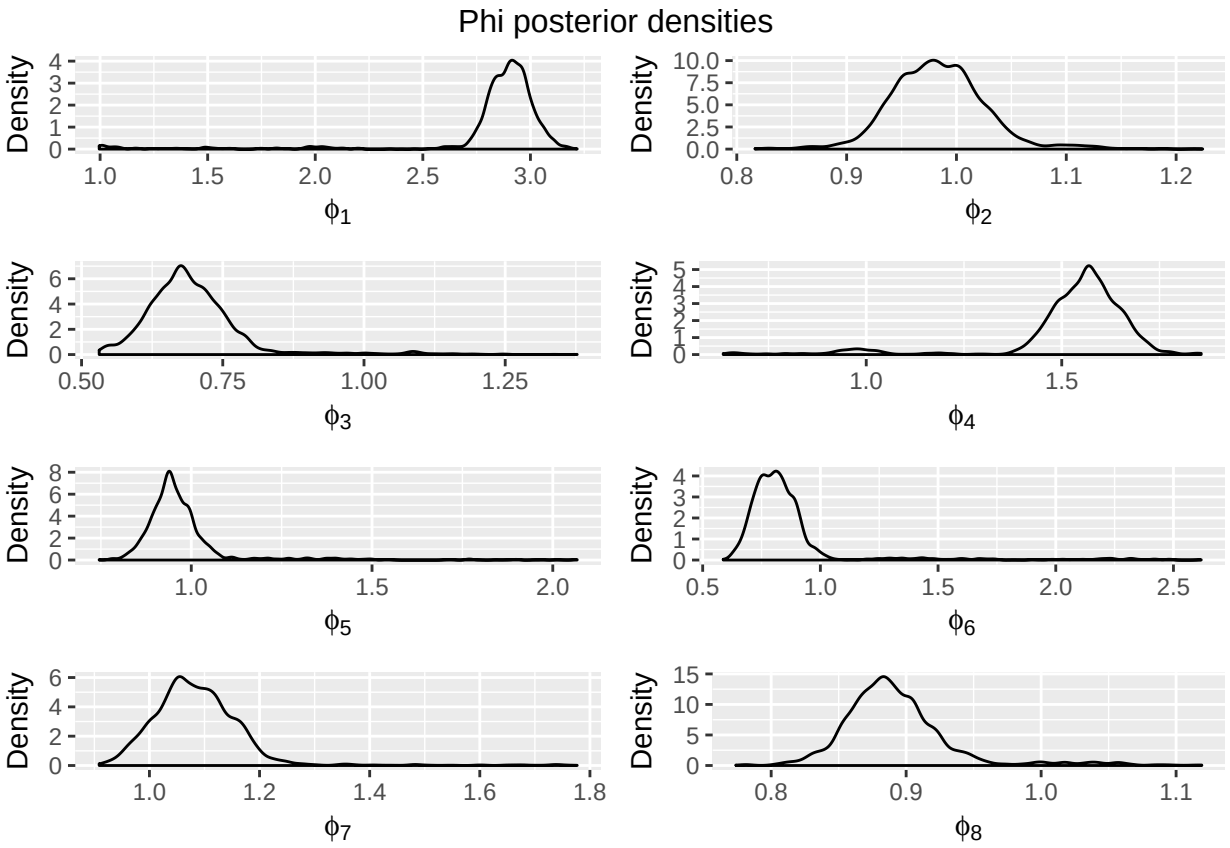
Using Metropolis random walk to simulate from the posterior β distribution resulted in the posterior mean coefficients which can be seen below.

Table 3: Point estimate of β.

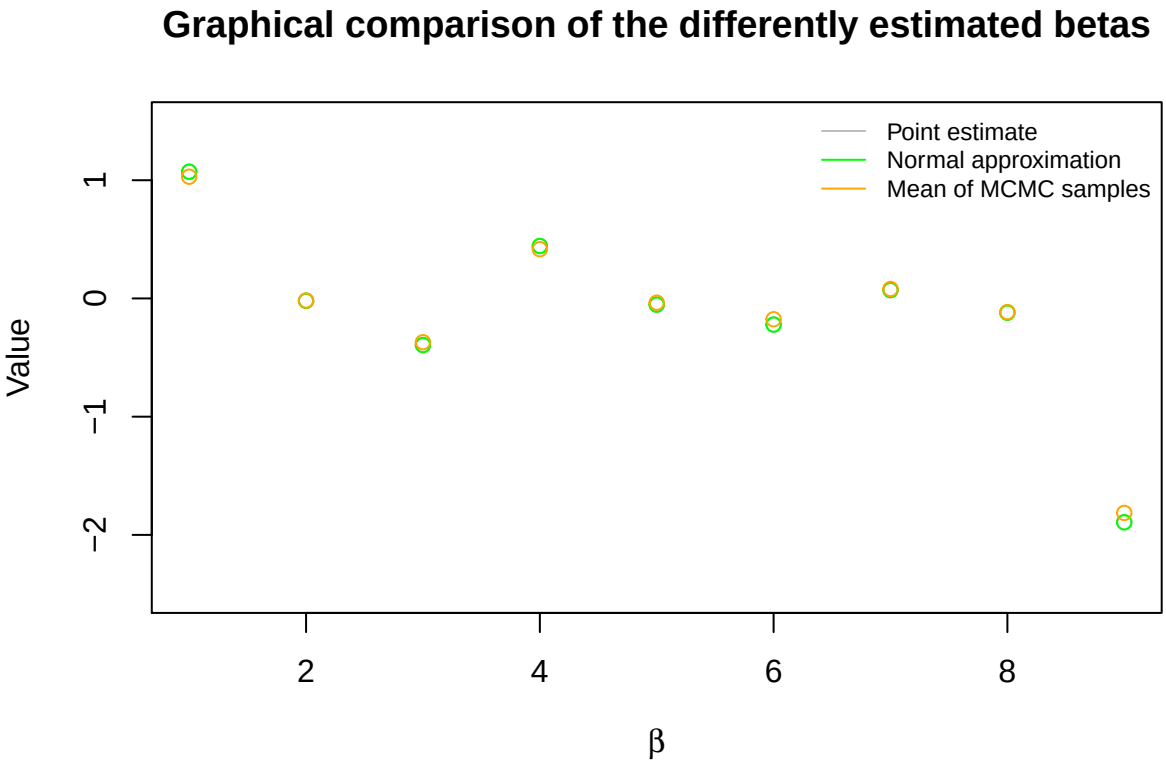
(Intercept)	PowerSeller	VerifyID	Sealed	Minblem	MajBlem	LargNeg	LogBook	MinBidShare
1.068	-0.017	-0.394	0.445	-0.051	-0.224	0.076	-0.121	-1.890



Observing the plots above it can be seen that all chains have converged to the static distribution.



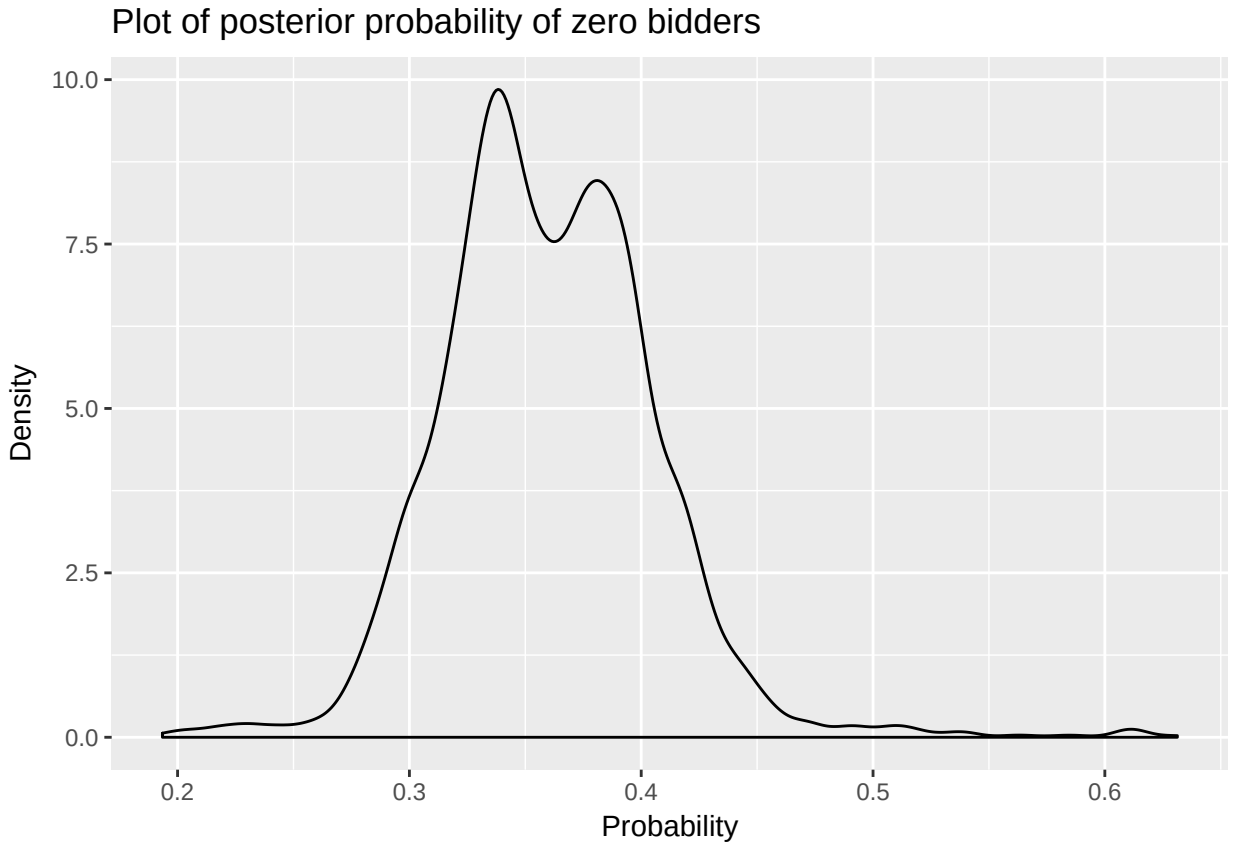
Mapping onto $\phi = \exp(\beta)$ and plotting the densities gives the plots that can be seen above.



Finally comparing the differenet estimated β shows that they all did a good job.

d)

Using the MCMC draws from c) to simulate from the predictive distribution of the provided data point resulted in the distribution that can be seen below. The expected value of no bidders was 0.36.



```r
# 1 - Poisson regression - the MCMC way

# a)
ebay.data <- read.table("eBayNumberOfBidderData.dat", header = TRUE)
ebay.glm <- glm(nBids ~ .-Const, data = ebay.data, family = poisson(link=log))
summary(ebay.glm)

# Estimate tells us: MinBidShare most important -1.89, sealed 0.44, veriID 0.39, majBlem -0.22 ,logBook

# b)
library(optimr)
library(mvtnorm)
library(ggplot2)
library(gridExtra)

log.posterior.of.beta <- function(beta, mu0, sigma0, X, Y) {
  beta.log.prior         <- dmvnorm(beta, mu0, sigma0, log = TRUE)
  poisson.log.likelihood <- sum(dpois(Y, exp(X %*% beta), log = TRUE))
  poisson.log.likelihood + beta.log.prior
}

Y      <- ebay.data[,1]
X      <- as.matrix(ebay.data[,2:10])
n      <- dim(X)[1]
p      <- dim(X)[2]
mu0       <- rep(0, p)
sigma0    <- 100*solve(t(X)%*%X)
beta.init <- rep(0, p)

optim.beta <- optim(beta.init,
                    log.posterior.of.beta,
                    gr      = NULL,
                    mu      = mu0,
                    sigma   = sigma0,
                    X       = X,
                    Y       = Y,
                    hessian = TRUE,
                    method  = c("BFGS"),
                    control = list(fnscale=-1))

log.approx.normal.posterior <- function (x) {
  dmvnorm(x, optim.beta$par, -solve(optim.beta$hessian), log = TRUE)
}

# c)
metropolis.sample <- function(c, sigma, logPostFunc, theta_c, ...) {
  theta_p <- as.vector(rmvnorm(1, theta_c, c*sigma))
  log.prob.theta_c <- logPostFunc(theta_c, ...)
  log.prob.theta_p <- logPostFunc(theta_p, ...)
  alpha  <- min(1, exp(log.prob.theta_p - log.prob.theta_c))
  accept <- runif(1, 0, 1) < alpha

  if (accept) {
```

```
    list(theta = theta_p, accept = accept)
  } else {
    list(theta = theta_c, accept = accept)
  }
}

# Initial beta is zero vector
metropolis <- function (nSamples, burnin) {
  c <- 0.5
  sigma.proposal <- -solve(optim.beta$hessian)
  betas   <- matrix(rep(rep(0, p), nSamples), ncol = p)
  accepts <- rep(0, nSamples-1)
  for (i in 2:nSamples) {
    result <- metropolis.sample(c, sigma.proposal,
                          log.posterior.of.beta,
                          betas[i-1,],
                          mu0,
                          sigma0,
                          X,
                          Y)
    betas[i,]    <- result$theta
    accepts[i-1] <- result$accept
  }

  valid.accepts <- accepts[burnin:nSamples]
  accept.ratio <- sum(valid.accepts) / length(valid.accepts)
  list(betas = betas, accept.ratio = accept.ratio)
}

nSamples <- 5000
burnin   <- 1000
res <- metropolis(nSamples, burnin)
betas <- res$betas
beta.post.mean <- apply(betas, 2, mean)

plot.graphical.comparison <- function () {
  plot(optim.beta$par, col ="gray",
       ylim = c(-2.5,1.5),
       xlab = expression(beta),
       ylab = "Value",
       main = "Graphical comparison of the differently estimated betas")
  points(ebay.glm$coefficients, col ="green")
  points(beta.post.mean, col="orange")
  legend("topright",
         c("Point estimate", "Normal approximation", "Mean of MCMC samples"),
                   lty=1,
                   col=c("gray", "green", "orange"),
                   bty='n',
                   cex=.75)
}

plot.posterior.beta.samples <- function () {
  x.axis <- seq(1, dim(betas)[1], 1)
```

```r
  beta.df <- data.frame(betas, x.axis)
  pb1 <- ggplot(beta.df, mapping = aes(x.axis, X1)) + geom_line() + labs(x = expression(beta[1]), y = "
  pb2 <- ggplot(beta.df, mapping = aes(x.axis, X2)) + geom_line() + labs(x = expression(beta[2]), y = "
  pb3 <- ggplot(beta.df, mapping = aes(x.axis, X3)) + geom_line() + labs(x = expression(beta[3]), y = "
  pb4 <- ggplot(beta.df, mapping = aes(x.axis, X4)) + geom_line() + labs(x = expression(beta[4]), y = "
  pb5 <- ggplot(beta.df, mapping = aes(x.axis, X5)) + geom_line() + labs(x = expression(beta[5]), y = "
  pb6 <- ggplot(beta.df, mapping = aes(x.axis, X6)) + geom_line() + labs(x = expression(beta[6]), y = "
  pb7 <- ggplot(beta.df, mapping = aes(x.axis, X7)) + geom_line() + labs(x = expression(beta[7]), y = "
  pb8 <- ggplot(beta.df, mapping = aes(x.axis, X8)) + geom_line() + labs(x = expression(beta[8]), y = "
  grid.arrange(pb1, pb2, pb3, pb4, pb5, pb6, pb7, pb8, ncol = 2, top ="Beta posterior samples")
}

plot.posterior.phis <- function () {
  phis <- exp(betas)
  phi.df <- data.frame(phis)
  pp1 <- ggplot(phi.df, aes(x = X1)) + geom_density() + labs(x = expression(phi[1]), y = "Density")
  pp2 <- ggplot(phi.df, aes(x = X2)) + geom_density() + labs(x = expression(phi[2]), y = "Density")
  pp3 <- ggplot(phi.df, aes(x = X3)) + geom_density() + labs(x = expression(phi[3]), y = "Density")
  pp4 <- ggplot(phi.df, aes(x = X4)) + geom_density() + labs(x = expression(phi[4]), y = "Density")
  pp5 <- ggplot(phi.df, aes(x = X5)) + geom_density() + labs(x = expression(phi[5]), y = "Density")
  pp6 <- ggplot(phi.df, aes(x = X6)) + geom_density() + labs(x = expression(phi[6]), y = "Density")
  pp7 <- ggplot(phi.df, aes(x = X7)) + geom_density() + labs(x = expression(phi[7]), y = "Density")
  pp8 <- ggplot(phi.df, aes(x = X8)) + geom_density() + labs(x = expression(phi[8]), y = "Density")
  grid.arrange(pp1, pp2, pp3, pp4, pp5, pp6, pp7, pp8, ncol = 2, top = "Phi posterior densities")
}

# d)

# New data point with const
x <- c(1, 1, 1, 1, 0, 0, 0, 1, 0.5)
beta <- metropolis(5000, 1000)$betas
posterior.prob <- dpois(0, exp(x %*% t(beta)))
plot.posterior.predictive.density <- function () {
  df <- data.frame(t(posterior.prob))
  ggplot(df, aes(x = t.posterior.prob.)) +
    geom_density() +
    xlab("Probability") +
    ylab("Density") +
    ggtitle("Plot of posterior probability of zero bidders")
}
mean(posterior.prob)
```