Course summary: Introduction to Bayesian Statistics

Jasper Ginn

March 2019

1 Introduction (week 1)

A Bayesian statistical analysis consists of several steps.

1. **Research question & data preview**. This is the most important step. We think about *what* it is we want to know and *how* we could represent this using data.
2. **Density of the data**. Here, we construct the likelihood of the data given our problem statement.
3. **Prior distribution for the parameters of the density of the data**. Any likelihood function has parameters. The prior distributions allow us to represent our knowledge of these parameters.
4. **Data collection**.
5. **Posterior distribution**. We construct the posterior distribution using Bayes' rule and sample it repeatedly.
6. **Inference for transformation of the parameters**. Using the samples we drew from the posterior, we can transform the parameters in a lot of ways to obtain new and interesting statistics.
7. **The posterior predictive distribution of the data**. We can use the posterior distribution to repeatedly create predictions, which gives us detailed information about our problem.

1.1 Bayes' rule

Bayesian statistics heavily rely on **Bayes' rule**, which is given by:

$$f(\theta|x) = \frac{f(x|\theta)f(\theta)}{f(x)} \tag{1.1}$$

Where:

- $f(\theta|x)$ is the posterior density.
- $f(x|\theta)$ is the likelihood of the data.

- $f(\theta)$ is the prior distribution of the parameter θ.
- $f(x)$ is a normalizing constant to ensure that the posterior distribution is a 'proper' probability distribution.

1.2 Constructing the density of the data (likelihood)

The density of the data, or *likelihood* of the data, is the **probability/likelihood** of the data *given* the parameter θ. The likelihood of the data depends on the problem under consideration. The likelihood of the data is defined for *every data point*. That is, for one data point, we have

$$f(x_i|\theta)$$

By virtue of independence, we can take the product of the individual likelihoods. So, for n data points, we have

$$f(\vec{x}|\theta) = \prod_{i=1}^{n} f(x_i|\theta)$$

1.3 Prior distributions

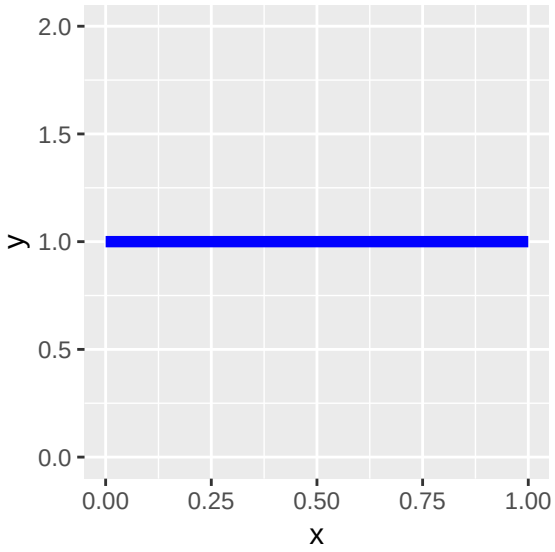
The parameter θ is not treated as a fixed but unknown parameter as we would do in frequentist statistics. Rather, we approach it as a random variable. The 'knowledge' that we have about our parameter is summarized in the *prior* $f(\theta)$. In equation 1.1, we observe that the prior is 'combined' with the likelihood of the data to form the posterior. Hence, we can see that the posterior is a mix between the evidence from observed data and the knowledge we add to the model by means of a prior.

Because of this 'mixing', it would be reasonable to expect that, if we knew a lot about the problem at hand and we could specify this in the prior, the posterior would be heavily affected by this knowledge. Conversely, if we knew nothing about the distribution of our parameters, the posterior would be affected primarily by the evidence we collected in the form of the data.

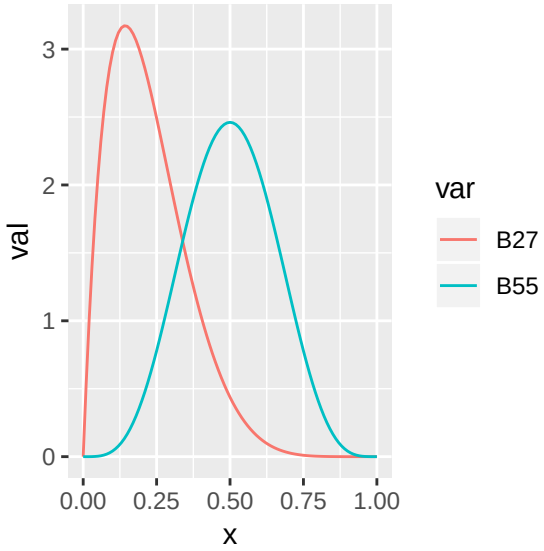
A prior can be:

1. **Uninformative**. This means that we do not know much about the form of θ, which is reflected in the prior distribution. Suppose that the prior distribution for θ is a beta distribution $\theta \sim B(\alpha, \beta)$.

Then, we could represent our lack of knowledge by using $B(1,1)$, which evaluates to 1 for every value of $x \in (0,1)$



2. **Informative**. This means that we have some idea about the distribution of θ prior to the analysis. We can reflect the strength of our *belief* in our knowledge in the way we construct the prior distribution. That is, in our beta distribution above, we can choose different parameters for α and β that will change the way the beta distribution looks.



In general, high dispersion (variance) indicates more uncertainty about the prior distribution whereas low dispersion indicates more certainty about our prior distribution.

A prior distribution is called *conjugate* if it has the same mathematical form as the density of the data. For example, if we have a binomial likelihood and a beta prior, then we get:

$$\begin{aligned} f(\theta|x) &\propto f(x|\theta)f(\theta) \\ &\propto \left[\prod_{i=1}^{n} \binom{n}{x} \theta^x (1-\theta)^{n-x}\right] \theta^{\alpha-1}(1-\theta)^{\beta-1} \\ &\propto \theta^{\sum x}(1-\theta)^{\sum n-x}\theta^{\alpha-1}(1-\theta)^{\beta-1} \\ &\propto \theta^{(\sum x_i)+\alpha-1}(1-\theta)^{(\sum n-x_i)+\beta-1} \end{aligned} \tag{1.3.1}$$

The resulting posterior is mathematically similar to the prior.

1.4 Constructing the posterior distribution

After we define the likleihood of the data and the priors, and we have collected our data for analysis, we can construct the posterior probability density $f(\theta|x)$, which is given by

$$f(\theta|x) = f(x|\theta)f(\theta) \tag{1.4.1}$$

Using a sampling procedure, we can then repeatedly sample the posterior distribution. This (hopefully) gives us a good approximation of the posterior.

1.5 Evaluation of the results

To evaluate the results, we compute the *Maximum a Posteriori* (MAP), which is usually the mean of the posterior distribution, but can also be the median or the mode. Furthermore, we calculate the posterior standard deviation of the sampled values and the 95% credible interval. This is analogous to the confidence interval in classical statistics with a much simpler interpretation: the 95% credible interval (CI) has a 95% probability of containing the true value of interest.

1.6 Why Bayesian statistics is called 'subjective'

The use of a prior in Bayesian statistics raises the question *which* prior should be considered best. Ultimately, this is a subjective choice, and a function of

1. (Subjective) prior knowledge.
2. Attitude with respect to the use of prior knowledge on historical data.
3. Evaluation of the 'fit' between study subjects and application in a previous study and current study.
4. Attitude with respect to the use of prior distributions to represent this prior knowledge.

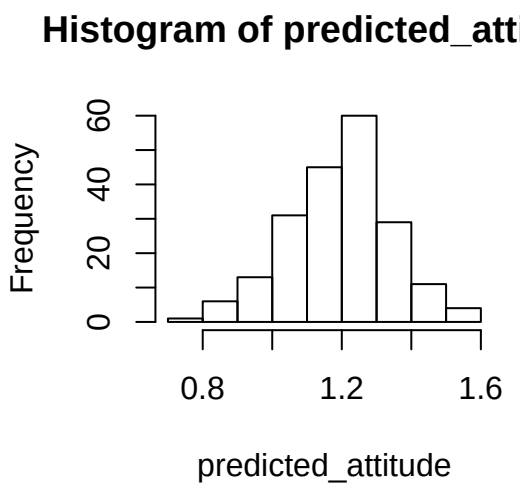
Ultimately, we should reasoned and transparent thinking to select priors of interest.

If we care more about historical results, we could choose to weight historical results more heavily, which results in *power priors*. We weight the likelihood of the historical data by some value α_0 to indicate that it should be more or less important than our believe of the distribution of the parameters. **This practice requires domain expertise**.

1.7 Transformation of posterior densities & posterior predictive distributions

Besides retrieving point estimates, we can also transform posterior distribution as we see fit after we have repeatedly sampled it to compute ratios, differences and other statistics. Such variables are treated as any other, meaning that we similarly calculate metrics like the EAP etc.

Unlike classical statistics, which focuses mainly on the computation of point estimates, bayesian statistics allows us more interesting analysis of posterior predictive results. For example, suppose that we run a linear regression on a dataset which predicts an attitude score from agreeableness and extraversion. Then a posterior predictive distribution of a person with an agreaableness score of 0.5 and an extraversion score of 2.5 would be predicted to have an attitude score that lies between 0.8 and 1.6 with the mean at an attitude score of 1.16.



Hence, posterior predictive distributions help us understand future predictions.

1.8 Practical using R and JAGS

We first need to specify the data in R. This we can do by copying it from the table

```
dat <- list(y.PE=58, n.PE=141, y.PC=40, n.PC=143)
```

Next, we specify a JAGS model. Given that we want to test the effectiveness of the treatment PE against PC, we will model the success probability θ, which indicates the probability of being cured.

Notice that JAGS doesn't require a specific order of the variables in the model file. This means that you can reference items before assignment.

We also specify an additional variable that we want to collect at each iteration called RR or the relative risk of remaining sick in the PC condition versus the PE condition:

$$RR = \frac{\theta_{PC}}{\theta_{PE}}$$

```
# Specify model
mod1 <- 'model{

  # Priors (uninformative)
```

```
  theta.PE ~ dbeta(1,1)
  theta.PC ~ dbeta(1,1)

  # Likelihood of the data
  y.PE ~ dbin(theta.PE, n.PE)
  y.PC ~ dbin(theta.PC, n.PC)

  # Contrast (RR)
  RR <- theta.PC / theta.PE

}'

# Load rjags
library(rjags)
con <- textConnection(mod1)
# Create a jags model
jmod <- jags.model(file=con, data=dat, n.chains=2)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 2
##    Unobserved stochastic nodes: 2
##    Total graph size: 8
##
## Initializing model
```

```
close(con)
```

At this point, we have initialized the model with uninformative beta priors. Next, we specify the burn-in period and burn 1.000 samples.

```
update(jmod, n.iter = 1000)
```

Finally, we specify the parameters of interest and run the sampler to obtain a sample of the posterior distribution.

```
params <- c("theta.PE", "theta.PC", "RR")
res <- coda.samples(jmod, variable.names = params, n.iter=10000)
```

Normally, we would first inspect convergence before viewing the results, but since we have not yet done this we will skip this step. We use the `summary()` function to inspect the results.

```
summary(res)
```

```
##
## Iterations = 2001:12000
## Thinning interval = 1
## Number of chains = 2
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##             Mean      SD  Naive SE Time-series SE
## RR       0.6920 0.11467 0.0008109      0.0010385
## theta.PC 0.2829 0.03728 0.0002636      0.0003359
## theta.PE 0.4128 0.04040 0.0002857      0.0003598
##
## 2. Quantiles for each variable:
##
##             2.5%    25%    50%    75%  97.5%
## RR       0.4918 0.6112 0.6844 0.7631 0.9381
## theta.PC 0.2124 0.2571 0.2817 0.3077 0.3592
## theta.PE 0.3348 0.3852 0.4122 0.4399 0.4925
```

We observe the following:

- The probability of recovering under treatment PC is roughly 28%, whereas the probability of recovering under treatment PE is roughly 41%.
- The relative risk of θ_{PC}/θ_{PE} is 0.68, 95% CCI $= [0.49, 0.94]$%, meaning that the risk of remaining ill under the treatment PC is $1/0.68 \approx 1.47$ higher than the treatment PE. Since the ratio is below 1, we can say that we are 95% certain that PE treatment gives a higher chance of recovery than PC treatment.

Notice that, for this model, it is also possible to derive the solution analytically. The mean for the posterior distribution is given by

$$\hat{\theta}_i = \frac{(\alpha_i + y_i)}{(\alpha_i + y_i) + (\beta_i + n_i - y_i)}$$

Which gives us

$$\theta_{PC} = \frac{1 + 40}{1 + 40 + 1 + 143 - 40} \approx 0.2828$$
$$\theta_{PE} = \frac{1 + 58}{1 + 58 + 1 + 141 - 58} \approx 0.4126$$

The RR then becomes $0.2828/0.4126 \approx 0.6854$. These results are the same as the results we obtained using sampling.

With respect to using informative hypotheses using historical data, I would argue that both datasets fit reasonably well with our current research goal. Under the assumption that the treatments PC and PE are relatively static (don't change much), and that PTSD is a stable condition (which is likely given that the impact of war is horrible in almost any situation), we could use the results from the previous studies by pooling the data together and using these to construct the prior distributions.

In the model file below, we specify the informative distributions.

```
mod2 <- 'model{

  # Priors (informative)
  theta.PE ~ dbeta(161,191)
  theta.PC ~ dbeta(126,281)
```

```
  # Likelihood of the data
  y.PE ~ dbin(theta.PE, n.PE)
  y.PC ~ dbin(theta.PC, n.PC)

  # Contrast (RR)
  RR <- theta.PC / theta.PE

}'

# Load rjags
con <- textConnection(mod2)
# Create a jags model
jmod <- jags.model(file=con, data=dat, n.chains=2)
close(con)
# Run the model
update(jmod, n.iter = 1000)
params <- c("theta.PE", "theta.PC", "RR")
res <- coda.samples(jmod, variable.names = params, n.iter=10000)
```

```
summary(res)
```

```
##
## Iterations = 2001:12000
## Thinning interval = 1
## Number of chains = 2
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##            Mean      SD  Naive SE Time-series SE
## RR       0.6803 0.05643 0.0003990      0.0005147
```

```
## theta.PC 0.3014 0.01962 0.0001388      0.0001824
## theta.PE 0.4442 0.02257 0.0001596      0.0002003
##
## 2. Quantiles for each variable:
##
##             2.5%    25%    50%    75%  97.5%
## RR       0.5755 0.6412 0.6782 0.7171 0.7958
## theta.PC 0.2638 0.2882 0.3012 0.3145 0.3403
## theta.PE 0.4004 0.4288 0.4442 0.4595 0.4881
```

The posterior mean for RR is 0.68 with 95% CCI = $[0.58, 0.8]$. The CCIs are smaller and still do not include 1. Hence, the result of including prior knowledge has quite a big effect.

2 Sampling the posterior: Gibbs method (week 2)

Until now, we have seen that:

$$f(\theta|y) = \frac{f(y|\theta)f(\theta)}{f_\theta(y)}$$

1. $f(y|\theta)$ is the density of the data (likelihood)
2. $f(\theta)$ is the prior distribution of the parameter θ
3. $f_\theta(y) = \int f(y|\theta)f(\theta)d\theta$ is the marginal distribution
4. $f(\theta|y)$ is the posterior distribution

We also know that the marginal distribution serves the role as a normalizing constant. That is, its role is to ensure that $f(\theta|y)$ is a proper probability distribution. Accordingly, we can state that

$$f(\theta|y) \propto f(y|\theta)f(\theta)$$

In practice, we have three options for sampling from the posterior distribution.

1. If θ is a single parameter, we can derive the posterior distribution analytically.

2. If θ is a vector containing multiple parameters, then we could derive conditional posterior distributions and sample the posterior using a Gibbs sampler.
3. If the conditional posteriors derived under 2 are of an unknown form, we cannot sample directly from them and so we use the Metropolis-Hastings algorithm.

2.1 Two parameter case

Suppose that we have $f(y|\theta_1, \theta_2)$. Then the joint posterior distribution will be:

$$p(\theta_1, \theta_2|y) \propto f(y|\theta_1, \theta_2)f(\theta_1, \theta_2)$$

However, it is more convenient to derive the conditional posterior distributions

$$f(\theta_1|\theta_2, y) \propto f(y|\theta_1, \theta_2)f(\theta_1|\theta_2) \tag{1}$$

$$f(\theta_2|\theta_1, y) \propto f(y|\theta_1, \theta_2)f(\theta_2|\theta_1) \tag{2}$$

Note that we often assume that the priors are independent. By definition, if we have independent events $f(\theta_1|\theta_2)$ and $f(\theta_2|\theta_1)$ then reduce to $f(\theta_1)$ and $f(\theta_2)$.[1]

To sample from the posterior distribution, we repeatedly and iteratively sample the conditional posteriors by using the following procedure:

1. Set $h = 0$ and H to a large number.
2. Choose the initial values for $\theta_1, \theta_2 \longrightarrow \theta_1^{(0)}, \theta_2^{(0)}$.
3. Set $h = h + 1$.
4. Sample $\theta_1^{(h)}$ from the conditional posterior $f(\theta_1|\theta_2^{(h-1)}, y)$.
5. Sample $\theta_2^{(h)}$ from the conditional posterior $f(\theta_2|\theta_1^{(h-1)}, y)$.
6. Repeats steps $3 - 5$ many times until $h = H$.

This procedure is called the *Gibbs sampler*.

[1] See e.g. this page

2.2 Evaluating convergence

There are several ways to check convergence of the model. Keep in mind that:

- **We should always use a combination of these methods**.
- **We can never prove that a chain has converged**.

1. **Autocorrelation plot**: Because the Gibbs sampler is an MC method, we by definition compute the current value on the basis of the previous one. High autocorrelation means that we are 'stepping through' the posterior space very slowly. This is a problem because we may not observe the entire posterior space. Autocorrelation further occurs because of **(1) mixed distributions** (e.g. a mixed (multimodal) normal distribution) and **(2) high autocorrelation between variables**.

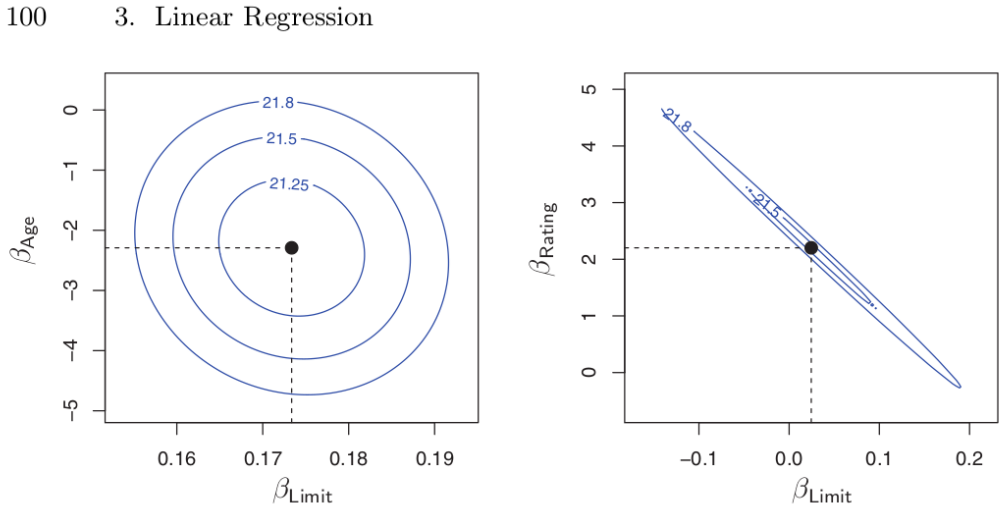
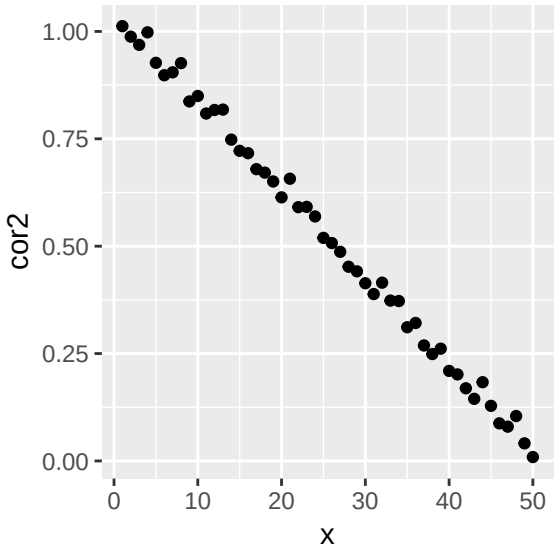


Figure 1: Uncorrelated versus correlated variables. Taken from: James, Gareth et al. Introduction to statistical learning. p.100

Autocorrelation disturbs the estimate of the posterior SD and reduces the *effective* sample size (and therefore the information) that we have collected from the posterior. The effective sample size would be the point where the autocorrelation is close to 0; around about every 40^{th} iteration in the plot below.



In general, the lower the autocorrelation, the more information we have about the posterior. This is referred to as *mixing*.

2. **Trace/history plots**: We check the values of the parameters against the index over multiple chains to see whether the estimate of the parameter stabilizes over time and to check whether the chains overlap. The value to which the chains coverge is called the **stationary distribution**. It is important that we 'burn' (remove) a set of n (usually close to 1.000) values from the start of the sequence to give the algorithm time to converge to a stable state.

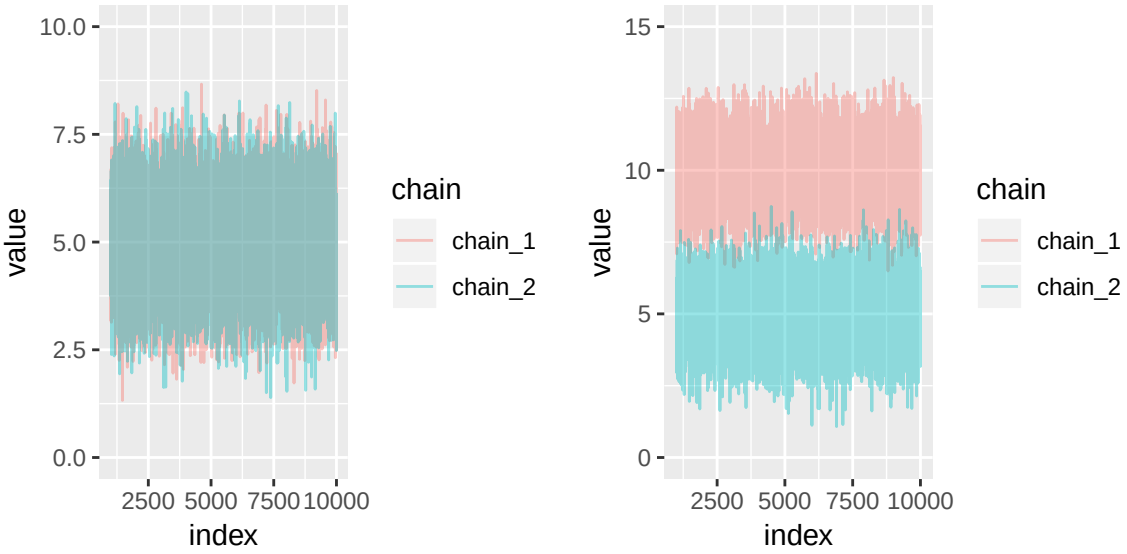


Figure 2: A converged model (left) and a model that has not converged (right) because the two chains do not overlap.

3. **Gelman-Rubin Statistic**: The GR statistic assesses convergence of multiple chains. The statistic compares the variance *within* chains to variance *between* chains. Its value should be close to 1.

4. **MCError**: The MC error is computed as

$$\text{MCERR} = \frac{\text{SD}}{\sqrt{\text{iterations}}}$$

Where the SD value is the standard deviation of the MAP estimates of a parameter. The MC error decreases as the number of iterations $\to \infty$, and should not be larger than 5% of the sample standard deviation.

2.3 What to do if convergence failed?

There are several things we can do if the model fails to converge.

1. Use more iterations
2. Re-parametrize the model, for example by centering the independent variables.
3. Use different priors for correlated variables.
4. Use different initial values.
5. Increase the thinning parameter.

2.4 Practical II using R and JAGS

The goal is to build a regression model that models people's attitude as a function of their agreeableness and extraversion:

$$\text{attitude}_i = \beta_0 + \beta_1 \text{agreeableness}_i + \beta_2 \text{extraversion}_i + e_i$$

Where $e_i \sim N(0, \sigma^2)$.

We assume the following priors:

- For $\beta_0, \beta_1, \beta_2$ we assume a normal prior. This yields six hyperparameters; a prior mean and variance for each coefficient.
- For σ^2 we assume an inverse-gamma prior. This yields two hyperparameters; a shape parameter and a scale parameter.

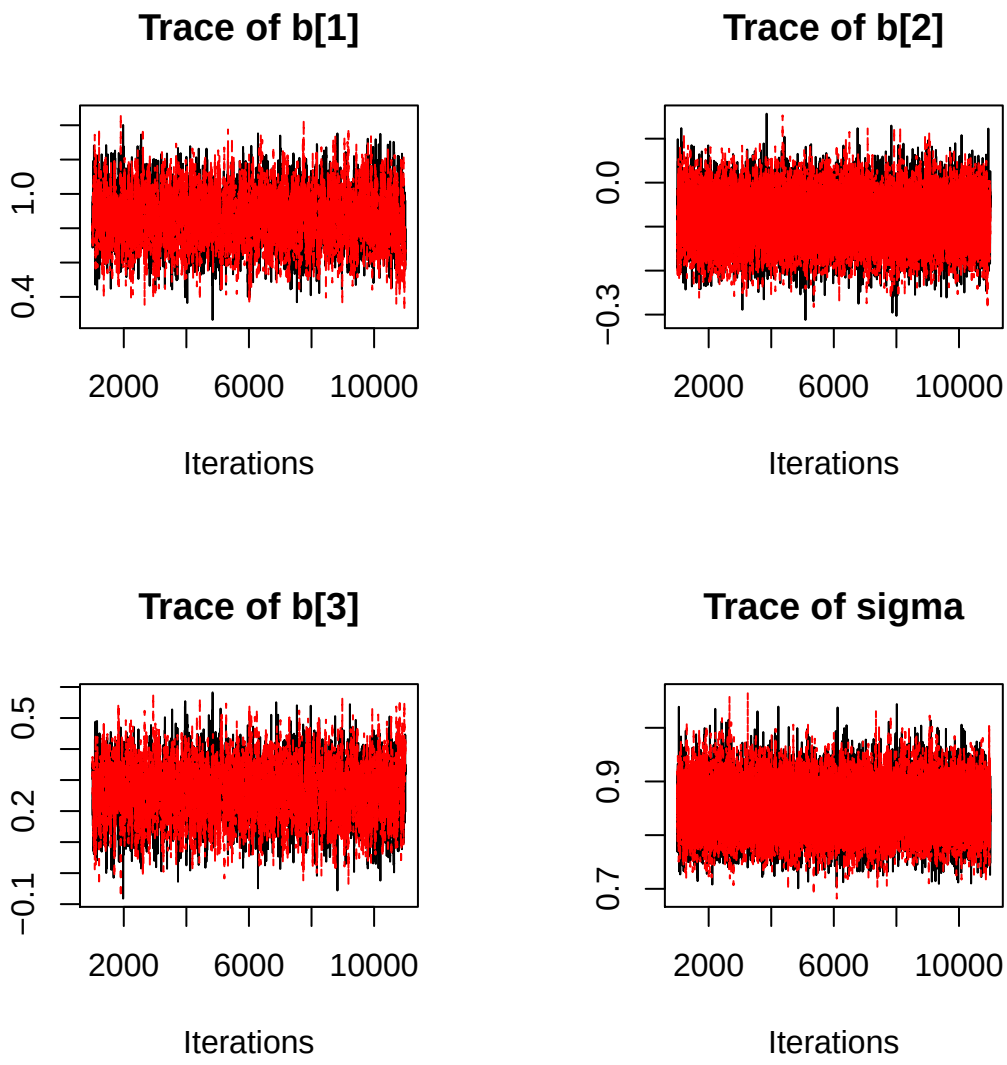
We specify the model as follows. Note the following two things:

1. We are using **uninformative priors**.
2. Since we are using uninformative priors, we draw the coefficients from a normal with mean $\mu_{0,0}$ and variance $\tau^2_{0,0}$. This happens in the line `b[coef] ~ dnorm(0, 1/10000)` below. We **don't see** the part where the likelihood of the data and the prior are merged together into the posterior.
3. In JAGS, we use *precision* instead of *variance*, hence the 1/10000.

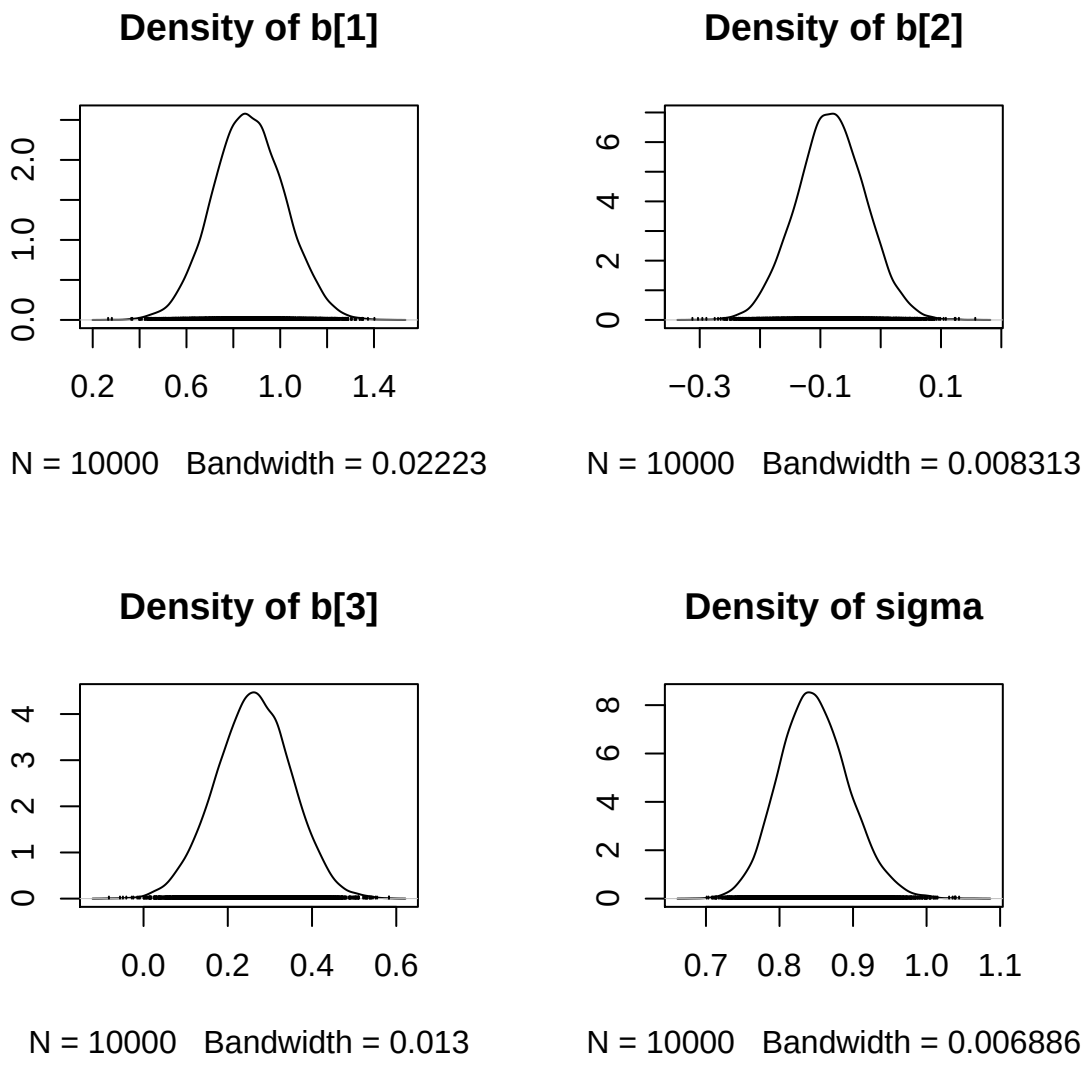
```
# Load data
d <- haven::read_spss("data/Exercise 2 - Data.sav")


# Make model
linm <- "model{


  #### Priors


  # Sigma^2
  tau ~ dgamma(0.001, 0.001)


  # Coefficients
  for(coef in 1:3) {
    b[coef] ~ dnorm(0, 1/10000)
  }


  #### Likelihood ==> for each example in the data SEPARATELY


  for(i in 1:length(attitude)) {
    # Calculate the mean value
    mu[i] <- b[1] + b[2] * extraversion[i] + b[3] * agreeableness[i]


    # Use dnorm to calculate the density at the mean value
    attitude[i] ~ dnorm(mu[i], tau)
  }
```

```
  #### Variables of interest
  sigma2 <- 1/tau
  sigma <- sqrt(sigma2)

}"

# Specify initial values for the chains
# (this is optional)
init <- list(
  init1 <- list(b = c(7, 0, -5), tau=1),
  init2 <- list(b = c(-5, 0, 7), tau=2)
)

# Text connection
con <- textConnection(linm)

# Jags model
library(rjags)
jmod <- jags.model(file=con, data=d, n.chains=2, inits = init)
close(con)
# Run the model
update(jmod, n.iter = 1000)

# Specify parameter names
params <- c("b", "sigma")
# Run the chain
res <- coda.samples(jmod, variable.names = params, n.iter=10000)
```

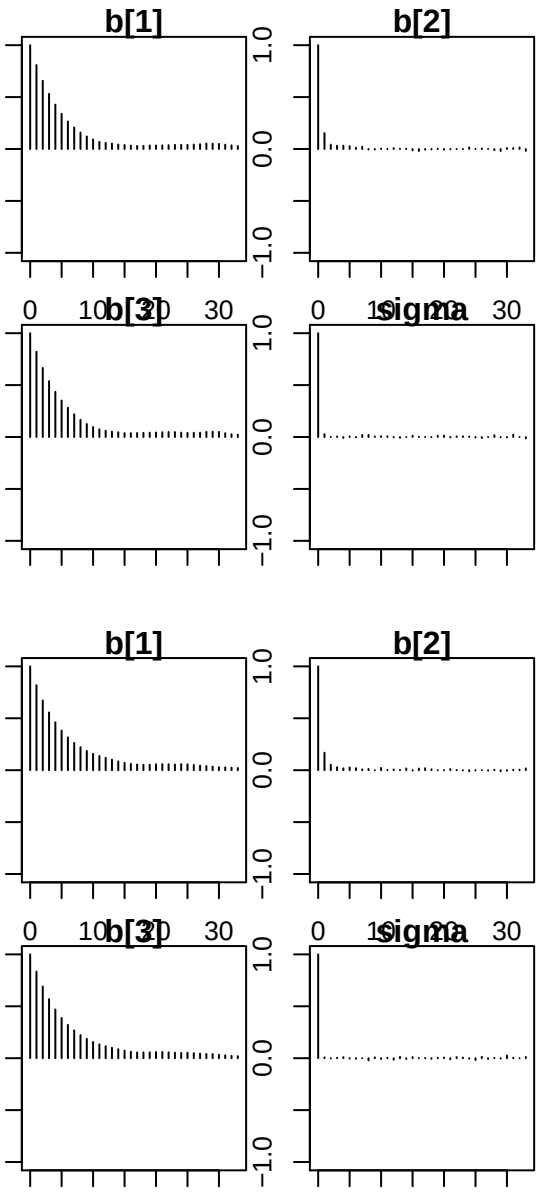
We first inspect convergence before we continue on and look at the results of the model.



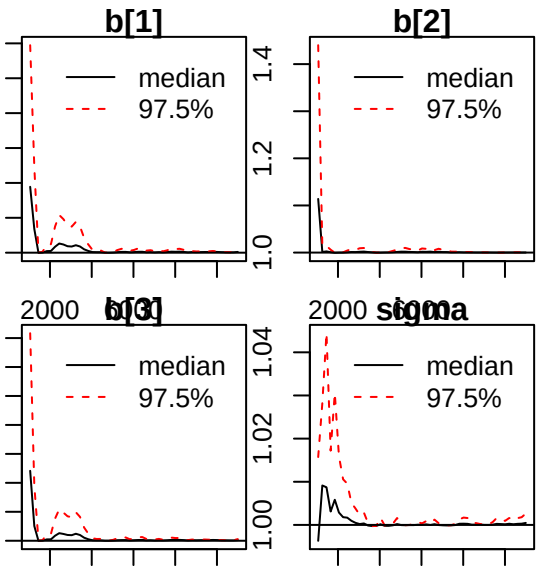
The traceplot looks converged for each parameter. The values of the different chains largely overlap and are within reasonable bounds (e.g. the variance is small). The parameters seem to have converged to a stationary distribution.



Similarly, the density plots show a nice, unimodal normal-like distribution.



There appears to be some evidence of autocorrelation, especially for the residual error and for the coefficient of agreeableness. The autocorrelation dissappears around the 15^{th} lag. To remedy this, we perform grand-mean centering on the variables.



The gelman-rubin plot shows that the between and within variance of the chains are close to one, which is desired. This means that the chains have 'forgotten' their initial states and have converged to the same values.

```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 2
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##           Mean      SD  Naive SE Time-series SE
## b[1]   0.86625 0.15198 0.0010746      0.0033570
## b[2]  -0.08233 0.05685 0.0004020      0.0005109
## b[3]   0.26138 0.08889 0.0006286      0.0020081
## sigma  0.84763 0.04708 0.0003329      0.0003376
##
## 2. Quantiles for each variable:
##
```

```
##           2.5%     25%      50%      75%   97.5%
## b[1]   0.57379  0.7624  0.86348  0.96926 1.16713
## b[2]  -0.19337 -0.1202 -0.08232 -0.04408 0.03003
## b[3]   0.08481  0.2017  0.26215  0.32139 0.43293
## sigma  0.76024  0.8150  0.84555  0.87813 0.94576
```

From the results, we observe the following:

1. The coefficient for extraversion has a negative sign (and therefore impact) on the outcome attitude with $\beta_2 = -0.082,\ 95\%\ \text{CI} = [-.19, .03]$. Because the credible interval contains 0, we conclude that there is no evidence to support the claim that a person's extraversion score impacts their attitudes towards pets.
2. The coefficient for agreeableness has a positive sign with mean $\beta_3 = 0.26,\ 95\%\ \text{CI} = [.082, .444]$. Therefore, for each one-point increase in the agreeableness score, the attitude towards pets increases by a value of 0.26. Given that the credible interval does not contain 0, we may conclude that agreeableness positively impacts a person's attitude towards pets.

Next, we run the model with an interaction effect.

```
# Load data
d <- haven::read_spss("data/Exercise 2 - Data.sav")


# Make model
linm <- "model{

  #### Priors

  # Sigma^2
  tau ~ dgamma(0.001, 0.001)

  # Coefficients
  for(coef in 1:4) {
    b[coef] ~ dnorm(0, 1/10000)
  }
```

```
  #### Likelihood ==> for each example in the data SEPARATELY
  for(i in 1:length(attitude)) {
    # Calculate the mean value
    mu[i] <- b[1] + b[2] * (extraversion[i] - mean(extraversion)) + b[3] * (agreeableness[i] - mean(agr
    # Use dnorm to calculate the density at the mean value
    attitude[i] ~ dnorm(mu[i], tau)
  }

  #### Variables of interest
  sigma2 <- 1/tau
  sigma <- sqrt(sigma2)

}"

# Specify initial values for the chains
# (this is optional)
init <- list(
  init1 <- list(b = c(7, 0, -5, 12), tau=1),
  init2 <- list(b = c(12, -5, 0, 7), tau=2)
)

# Text connection
con <- textConnection(linm)

# Jags model
library(rjags)
jmod <- jags.model(file=con, data=d, n.chains=2, inits = init)
close(con)
# Run the model
update(jmod, n.iter = 1000)
```

```r
# Specify parameter names
params <- c("b", "sigma")
# Run the chain
res <- coda.samples(jmod, variable.names = params, n.iter=10000)
```

We skip checking convergence for now and go straight to the results.

```
##
## Iterations = 1001:11000
## Thinning interval = 1
## Number of chains = 2
## Sample size per chain = 10000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##           Mean      SD  Naive SE Time-series SE
## b[1]   1.22545 0.06716 0.0004749      0.0004963
## b[2]  -0.08625 0.05665 0.0004006      0.0004491
## b[3]   0.27867 0.09077 0.0006418      0.0007254
## b[4]   0.11109 0.06708 0.0004743      0.0005193
## sigma  0.84352 0.04663 0.0003297      0.0003413
##
## 2. Quantiles for each variable:
##
##           2.5%      25%      50%      75%   97.5%
## b[1]   1.09347  1.18054  1.22503  1.27076 1.35852
## b[2]  -0.19787 -0.12456 -0.08618 -0.04815 0.02476
## b[3]   0.10203  0.21811  0.27830  0.33870 0.45928
## b[4]  -0.02009  0.06589  0.11106  0.15619 0.24380
## sigma  0.75816  0.81105  0.84123  0.87378 0.94000
```

From the results, we observe that the 95% CI of the interaction effect contains 0, and as such we do not have evidence that the effect of agreeableness on liking pets is moderated by extraversion.

3 Sampling the posterior: Metropolis-Hastings method

The Metropolis-Hastings algorithm is an MC method for sampling the posterior that allows us to sample from a generic probability distribution that we call the 'target distribution', even if we do not know the normalizing constant of the distribution we are looking for. This may occur because e.g. finding the marginal distribution is difficult or impossible because we cannot integrate the function.

To do this, we construct - and sample from - a Markov Chain whose stationary distribution is the distribution that we are looking for. That is:

$$f(\theta) \propto g(\theta)$$

The MH algorithm then proceeds as follows:

1. Select an initial value $\theta^{(0)}$.
2. For $i = 1, \ldots, m$, repeat:
   a. draw a candidate value from a proposal distribution $q(\theta)$ s.t. $\theta^* \sim q(\theta^*|\theta^{(i-1)})$.
   b. Set $\alpha = \frac{g(\theta^*)/q(\theta^*|\theta^{(i-1)})}{g(\theta^{(i-1)})/q(\theta^*|\theta^{(i-1)})} = \frac{g(\theta^*)q(\theta^{(i-1)}|\theta^*)}{g(\theta^{(i-1)})q(\theta^*|\theta^{(i-1)})}$
   c. If $\alpha \geq 1$: Accept θ^* and set $\theta^{(i)} \leftarrow \theta^*$
   d. If $\alpha < 1$: Accept θ^* and set $\theta^{(i)} \leftarrow \theta^*$ with probability α

The basic idea of the algorithm is this: if moving is **adventageous** ($\alpha \geq 1$), which you can think of as e.g. being in the tails of the distribution and having a candidate at the peak of the distribution, then *we will always move*. If moving is **not adventageous** ($\alpha < 1$), then we may still move *but only with probability* α.

3.1 Selecting a proposal density

The proposal density may or may not depend on the previous value of theta. If it does, then the MH algorithm is called a **random-walk MH algorithm**. If it does not, then we must make sure that $q(\theta) \sim f(\theta)$.

In a random walk MH, the proposal distribution is centered on the value of theta of the previous iteration ($\theta^{(i-1)}$). If we choose $q(\theta)$ to be a symmetric distribution (e.g. a normal), we have a neat additional feature; it means that in step 2*b*, we only have to calculate the ratio $\alpha = \frac{g(\theta^*)}{g(\theta^{(i-1)})}$. This works because, for symmetric distributions, $q(\theta^{(i-1)}|\theta^*) = q(\theta^*|\theta^{(i-1)})$.

The MH algorithm has an **acceptance rate** which is basically the proportion of accepted candidates. High acceptance rates means that we are stepping through the posterior space too slow (slow mixing). Typically, we want an acceptance rate between $20\% \sim 50\%$. Similarly, we want low autocorrelation, but in practice the autocorrelation will be quite high because samples generated from an MH are not i.i.d. (in this case we can always use longer chains).

The Gibbs sampler can be viewed as an MH sampler with a perfect proposal density. Hence, the acceptance rate is 1 and $q(\theta) = f(\theta)$ exactly.

4 Model selection using DIC (week 7)

With any model, we like to be able to quantify the 'quality' of that model given the problem we are working on. There are many forms of 'information criteria' (IC) such as the AIC, BIC etc. that do this. All IC methods are based on the same principles:

1. We like well-fitting models.
2. We like parsimonious (specific) models that are not complex.

Given that 'well fitting' refers to the fit of the model on the data, it would be reasonable to evaluate the *likelihood* of the data given the parameters, or $f(data|parameters)$. Parsimony is defined by using its antonym 'complexity'. Hence, we have two parts to any IC method:

1. A measure of **fit**. Usually the log-likelihood of the data given the parameters.
2. A measure of **complexity**. Usually a penalty measure involving the number of parameters.

Which sum together as follows:

$$\text{IC} = \text{model misfit} + \text{model complexity}$$

We prefer the model with the smallest IC.

Hence, IC methods give us the ability to compare models that are not nested[2] and to compare more than two models at the same time. It is a standardized metric, and therefore comparable across models and methods.

[2] meaning that the models are not of a form s.t. model A fully encompasses model B

4.1 Kullback-Leibler distance

The K-L distance minimizes the distance between the **truth** $p(.)$ and the **model under consideration** $f(.|\theta)$:

$$\text{K-L} = E_{p(y)}\left[\log p(y) - \log f(y|\theta)\right] \tag{KL}$$

The 'truth' $p(y)$ is a constant and can be ignored when comparing models. Furthermore, since $\log f(y|\theta) \geq \log p(y)$ we maximize $E\left[\log f(y|\theta)\right]$[3].

To compute $f(y|\theta)$ for a measure like the AIC, we use $\hat{\theta}_y$. This gives us a biased estimate because we are using y both to estimate the model **and** to evaluate the fit. Hence, we need a penalty term to correct for overfitting.

4.2 Deviance information criterion (DIC)

The DIC minizes the loss function / deviance of a model using the posterior mean:

$$\text{DIC} = -2\log f(y|\bar{\theta_y}) + 2_{PD}$$

Like all IC measures, the DIC consists of two parts:

1. The **likelihood** of the data, computed as the log of the density of the data using the posterior mean.
2. A measure of **model complexity**, which will be estimated as the effective number of parameters used in the model.

In a complex model or when using informative priors, the number of parameters (dimensionality) is difficult to define. The DIC estimates this number as:

$$\begin{aligned}\text{PD} &= -2\left[E_{p(\theta|y)}\left\{\log f(y|\theta)\right\} - \log f(y|\bar{\theta}_y)\right] \\ &\approx -2\left[\left\{\frac{1}{Q}\sum_{i=1}^{Q}\log f(y|\theta^q)\right\} - \log f(y|\frac{1}{Q}\sum_{i=1}^{Q}\theta^q)\right]\end{aligned}$$

When computing the DIC, we prefer the model that has the lower value. Given two models, we then say that model A (lower DIC) provides a better description of the data than model B (higher DIC).

[3]In equation KL we are taking a difference. If we drop $\log p(y)$ then we are left with $-\log f(y|\theta)$. In this case we would *minimize* this value, but since we like to maximize things instead we multiply by -1.

4.3 Practical: model DIC using R and JAGS

We will use JAGS to obtain the posterior for a logistic regression model. Further, we will make use of the DIC to select a best model.

Our goal is to make a clinical prediction rule to detect clinical depression in primary care. We use low informative priors.

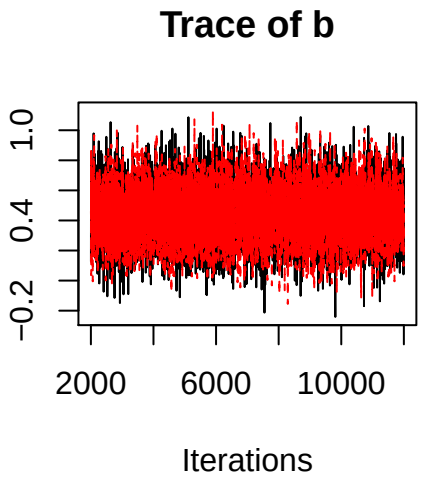
Next, we specify the logistic regression model with only `gender` as a predictor

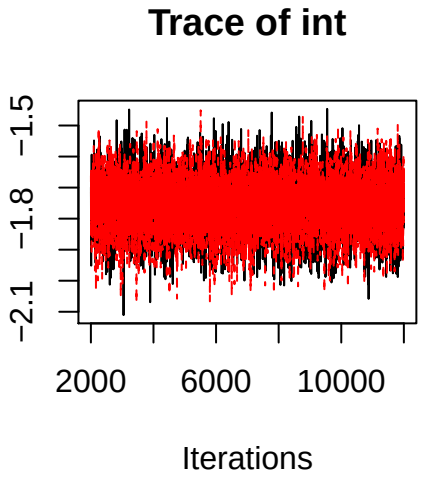
```
# depGP is outcome variable
jmodl <- "model {

  ### Likelihood
  for(i in 1:length(depr)) {

    # Outcome variable distributed as bernoulli with theta == p
    depr[i] ~ dbern(p[i])

    # Calculate theta
    logit(p[i]) <- int + b[1]*(gender[i] - mean(gender))

  }

  ### Priors

  # Intercept (normal)
  int ~ dnorm(0, 1/1000)
  # Coefficients
  for(j in 1:1) {
    b[j] ~ ddexp(0, sqrt(2)) # OR: b[j] ~ dnorm(0, 1/10000)
  }

  # Variables of interest
  odds_gender <- exp(b[1])
```

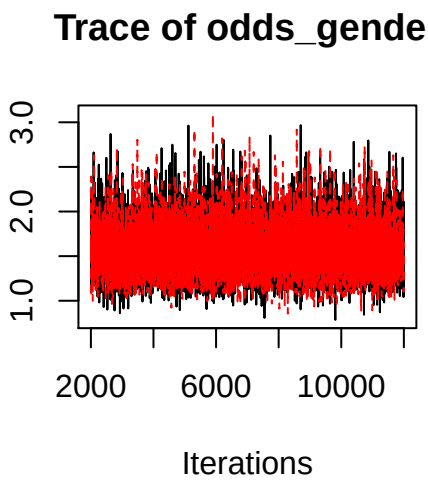
```
}"


# Parameters
params <- c("int", "b", "odds_gender", "prob_gender")
# Initial values
init <- list(list(int = 0, b=2),
             list(int = 2, b=4))


# Initiate JAGS model
library(rjags)
con <- textConnection(jmodl)
mod_gender <- jags.model(con, data=d, n.chains=2, inits=init)
close(con)


# Burn-in
update(mod_gender, n.iter = 1000)


# Sample posterior
samples <- coda.samples(mod_gender, variable.names=params, n.iter = 10000, thin=2)
```

In the model above, we also calculate the odds ratio for `gender` in the model. The advantage of this is that we do not have to do it outside of the model.

First, we assess convergence


Trace of b
1.0
0.4
−0.2
2000
6000
10000
Iterations

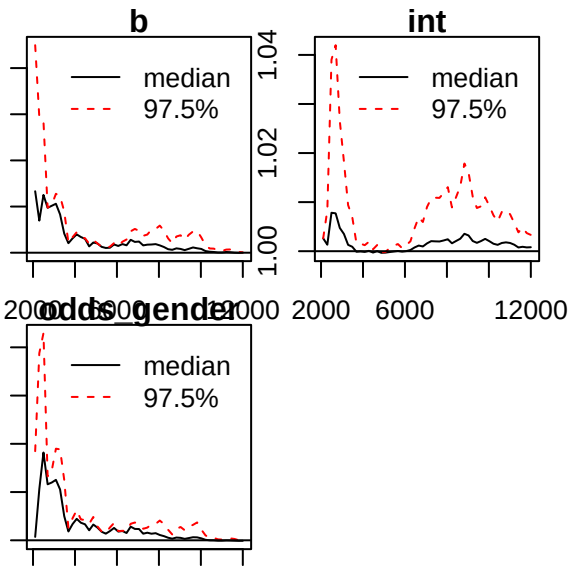

Trace of int
−1.5
−1.8
−2.1
2000
6000
10000
Iterations



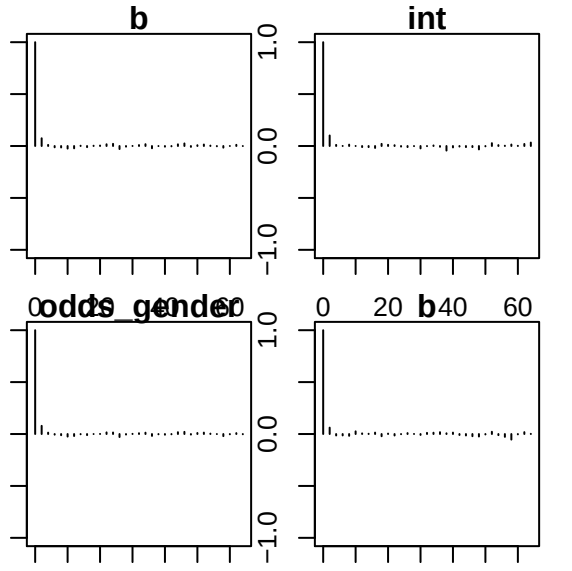
The trace plot indicates that the coefficients have converged to their stable distribution.

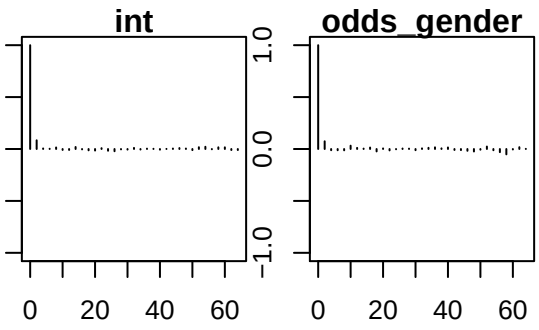
```
## Potential scale reduction factors:
##
##             Point est. Upper C.I.
## b                    1          1
## int                  1          1
## odds_gender          1          1
##
## Multivariate psrf
##
## 1
```

Similarly, we observe that from the gelman-rubin statistic that the two chains have converged as the GR statistic is close to 1.



Finally, we check the autocorrelation





Autocorrelation is quite an issue in this model. We can reduce it by increasing the thinning parameter and by centering the predictor (even though centering a binary variable is meaningless)

```
##
## Iterations = 2002:12000
## Thinning interval = 2
## Number of chains = 2
## Sample size per chain = 5000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##                Mean      SD  Naive SE Time-series SE
## b             0.4417 0.19049 0.0019049      0.0020405
## int          -1.7539 0.08912 0.0008912      0.0009784
## odds_gender   1.5840 0.30693 0.0030693      0.0033079
##
## 2. Quantiles for each variable:
##
##                 2.5%     25%     50%     75%   97.5%
## b             0.08179  0.3096  0.4385  0.5713  0.8214
```

```
## int         -1.93182 -1.8134 -1.7529 -1.6936 -1.5825
## odds_gender  1.08523  1.3629  1.5504  1.7706  2.2737
```

The coefficient for gender is positive, which indicates that females have a higher probability of depression than males $\beta_{gender} = 0.432,\ 95\%$ CCI $= [0.068, 0.822]$. The odds ratio does not include 1, and so we may conclude that gender is a predictor of depression.

Next, we define two other models using different predictors and compare the models using the DIC.

Next, we compare the models

```
dic_gender <- dic.samples(mod_gender, 1000, "pD")
dic_gender
```

```
## Mean deviance:  880.4
## penalty 1.274
## Penalized deviance: 881.6
```

```
dic_age <- dic.samples(mod_age, 1000, "pD")
dic_age
```

```
## Mean deviance:  882.7
## penalty 0.6103
## Penalized deviance: 883.3
```

```
dic_educ <- dic.samples(mod_educ, 1000, "pD")
dic_educ
```

```
## Mean deviance:  878.5
## penalty 0.3759
## Penalized deviance: 878.9
```

Whether or not we use the parameter estimates and DIC depends heavily on our application area. If we were just looking for the best fitting model, we wouldn't necessarily care about insignificant predictors all that much. However, if our goal is inference, we would defnitely want to make more parsimonious models and leave out predictors that don't add much to the model.