Bayesian Modelling for Biostatistics

Alexandra Posekany

https://github.com/alexposekany/workshopbayesbiomed/

Thank you for having me here!

NOVAMATH
CENTER FOR MATHEMATICS
+ APPLICATIONS

REPÚBLICA
PORTUGUESA

Probabilistic Learning

Machine Learning is based on data with one or more of the following properties:

- **Stochastic** and/or generated by a complex non-deterministic or not fully understood process
- Noisily observed
- Partially observed

Probability theory is a wide field of research focussed on expressing, modelling such uncertainties and finding appropriate data generating processes. Among those are quite prominently Probabilistic models. **Probabilistic** is directly connected with the term "probability".

Some fields of unsupervised learning where no Outcome data is availble for backpropagation and Training of an algorithm, introduced probabilities rather than deterministic decisions. Among those is naive Bayesian filtering which is applied e.g. for filtering spam emails.

Probabilistic Modelling

Probabilistic models do not simply transform information in the data into single number outputs, but allow us to include previous knowledge and provide an information about how probable any one possible value is based on the information contained in the data.

Therefore instead of fitting a deterministic model to the data, as is done by regression where the only stochastic part is left in the residuals, we will fit models where each part, the data, the parameters/model coefficients are inherently random. As random variables, we describe them through the means probabilities and their distributions. Based on these, algorithmic "learning" happens through "updating" information based on our observed data.

Introduction to Bayesian Inference

Bayes' Theorem

$$\pi(\theta|x) = \frac{\pi(\theta)f(x|\theta)}{\int_\Theta \pi(\theta)f(x|\theta)d\theta}.$$

The denominator contains the **marginal distribution**, $m(x) = \int_\Theta \pi(\theta)f(x|\theta)d\theta$ which has the normalisation purpose of assuring that $\pi(\theta|x)$ is a probability distribution.
The quintessential parts of the Bayesian model are:

- the **prior distribution** $\pi(\theta)$ which expresses the uncertainty about a model parameter θ from parameter space Θ;
- the **Likelihoodfunction** $f(x|\theta)$ which transforms the information contained in the data to the model structure and evaluates its 'fittingness',
- and the resulting **posterior distribution** $\pi(\theta|x)$.

Why bother working with distributions?

Prior distributions introduce information available before the statistical analysis from any external sources into the model. This distribution basically assigns a probability to every possible value of the parameter for being a "proper" choice for the current model

The **Likelihood function** as in all statistical models weighs how well the observed data fit into the chosen model based on for a certain choice of parameter. The dependence on the choice of parameter is relevant, as it is exactly this parameter we wish to learn about in our Bayesian model.

In all our previous methods for regression with and without regularisation, cross validation or Monte Carlo simulation where we aimed for estimating a single specific value of parameters/a set of model coefficients/an area under a curve etc. Contrary to all of them, probabilistic modelling aims for obtaining a **probability** of being "appropriate" for **every single value** considered. The probability distribution of every single parameter value after combining previous information (prior) and data information (likelihood) is the **posterior distribution**.

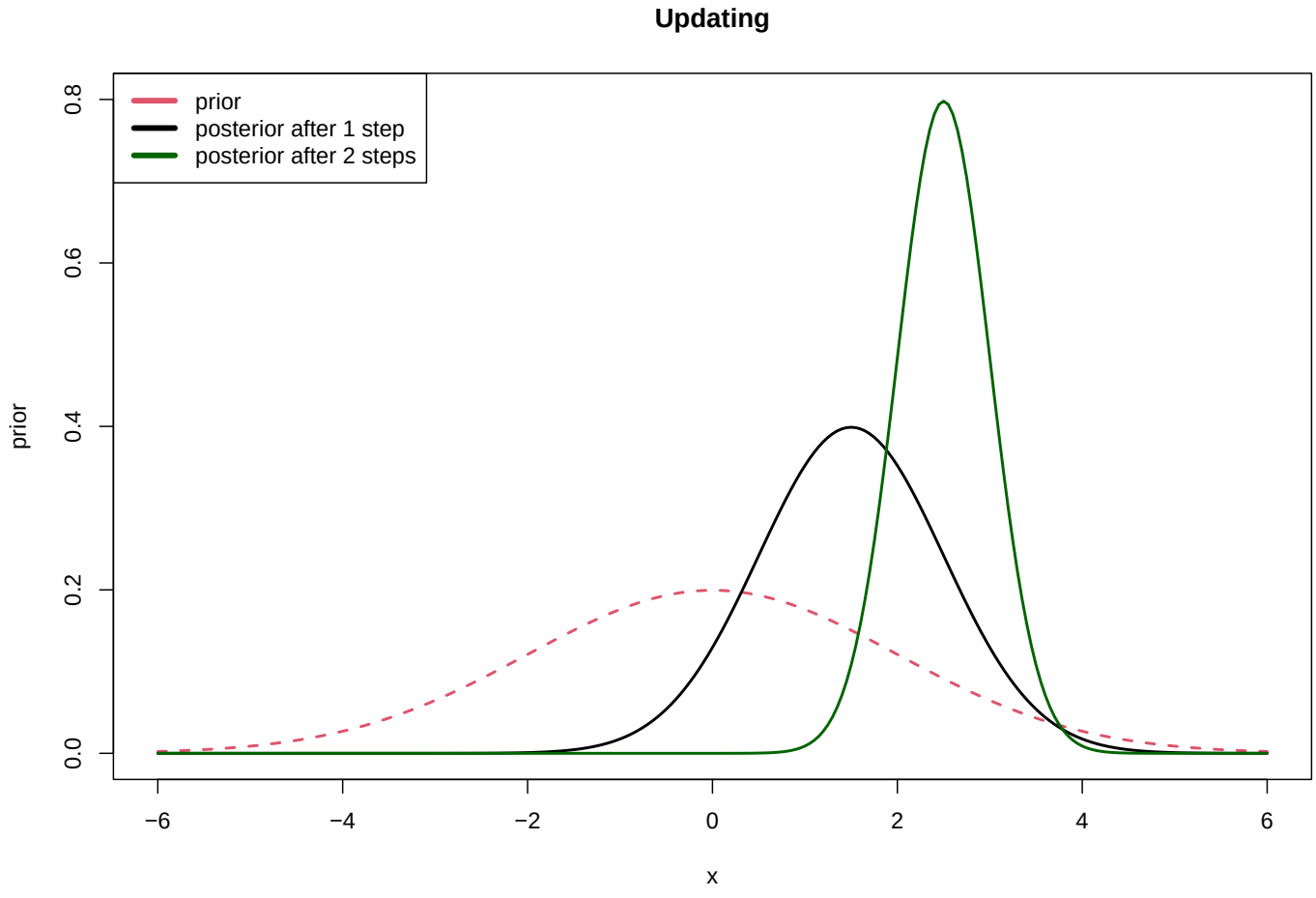
Updating Information

Maximum Likelihood estimation and Frequentist estimation can only use the whole sample at once and not include additional information.

Bayesian estimation allows for updating: taking the posterior after including the first observation as prior for including the second information results in the same posterior as including both observations at once. "**Updating**''

$$\begin{aligned} p(\theta|x_1, x_2) &\propto f(x_1, x_2|\theta)p(\theta) \\ &\propto f(x_2|\theta)f(x_1|\theta)p(\theta) \\ &\propto f(x_2|\theta)p(\theta|x_1) \end{aligned}$$

Updating


Updating
prior
posterior after 1 step
posterior after 2 steps
prior
0.0
0.2
0.4
0.6
0.8
−6
−4
−2
0
2
4
6
x

Why Bayesian methods? - Advantages

- All uncertainty is modelled as probability, probability laws obeyed.
  Complex hierarchical models are possible.

- Allows the incorporation of (prior) scientific information.

- Appropriateness of methods does not depend on having large sample sizes (asymptotics). Relevant for medical research!

- Can easily obtain inferences for any quantity of interest in an intuitively interpretable manner.
  - Direct probability interpretations.

  - Prediction is straightforward.

Why Bayesian methods? - Potential disadvantages

- Inferences depend on choice of prior and likelihood function, which may be incorrectly specified.
- Real prior distributions can be difficult to obtain in complicated models.
- In fact, bad prior information may be worse than no prior information.
- Sensitivity analysis is necessary.

The battle between Frequentist and Bayesian statistics

- **Frequentist**
  - procedure that quantifies uncertainty (e.g., p-value, confidence interval, etc.) in terms of repeating the process that generated the data many times
  - parameters are a single fixed value and unknown
  - unbiased estimation
  - makes probability statements only about the data
  - unconditional probabilities
- **Bayesian**
  - procedure that represents the uncertainty about parameters with probability distributions
  - parameters are random variables and unknown
  - biased estimation
  - makes probability statements about model parameters and the data
  - conditional probabilities

Probability

- The main difference between classical and Bayesian statistics is the concept of probability
  - from the classical point of view, probability is an "objective" concept
  - from the Bayesian point of view, probability is an "subjective" concept, as probabilities are conditional
- Both approaches have pros and cons. However, when they are both applicable, they are unlikely to give different results.

Revision of Distributions

Distributions will by the building stones of Bayesian inference. We will therefore revise them - we already used them for random number generation. Now we also add the consideration for which type of scenarios which distributions are reasonable as likelihood functions and priors.

Discrete distributions will be in use for selected scenarios:

- Actual categories: univariately we start with a uniform distribution and will end up with different weights for categories a-posteriori.
- Indicators of groups: These are drawn from a Bernoulli/Binomial distribution in the dichotomous case or a multinomial distribution for more than 2 groups.
- Counts: typically we model counts without a known maximum number of counts and apply Poisson or negative binomial distributions.

Applying continuous distributions

For continuous distributions again several scenarios exist:

- **Modelling continuous observations**: The most frequently used distribution for modelling anything is the normal distribution due to the mean and standard deviation having a direct interpretation and simple to calculate estimators.

  Depending on the domain of the data and the typical shape of their values other distributions become appropriate:

  - For skewed data with positive values log-normal distributions or Gamma distributions are applied.
  - For bounded data Beta distributions may be applied.
  - For overdispersed data with heavy tails student's t distributions can be utilized.

- **Modelling residuals**: Assuming a normal distribution of residuals is common for estimating confidence bounds of linear, generalised linear and regularised regression settings, as well as other algorithms based on least squares estimation such as LDA. This distribution assumption directly transfers the models into the Bayesian setting with the option of choosing different residual distributions.

Types of prior Distributions - Informative priors

- **Natural conjugate prior distribution**
  The prior has the same "structure" as the likelihood.
  Therefore we can obtain a solution in closed form.
  Because prior, likelihood and posterior have a common "structure" we are also able to interpret the prior, data and resulting posterior information in terms of this structure and the model's parameters or distributional shape and properties.

Theorem (Pitman-Koopman Lemma)

If for large enough sample size there exists a sufficient statistic of constant dimension for a family of distributions $f(.|\theta)$, then this family is exponential, if the support of $f(.|\theta)$ is independent of θ.

- **Elicitated prior**
  A prior is built 'manually' in such a way that specific 'weights' are put on specific values of the parameter based on concrete information.

Important Conjugate prior combinations for discrete likelihoods

- **Binomial distribution** $y \sim Bin(n, p)$
  conjugate prior for proportion p: $p \sim Beta(a, b)$
- **Negative Binomial distribution** $y \sim NBin(n, p)$
  conjugate prior for proportion p: $p \sim Beta(a, b)$
- **Multinomial distribution**
  $y \sim Multi((n_1, \ldots, n_m), (p_1, \ldots, p_m))$
  conjugate prior for proportions $p_1, \ldots, p_m$:
  $p \sim Dir(\alpha, (\theta_1, \ldots, \theta_m))$
- **Poisson distribution** $y \sim Poi(\lambda)$
  conjugate prior for rate λ: $\lambda \sim Ga(a, b)$

Important Conjugate prior combinations for continuous likelihoods

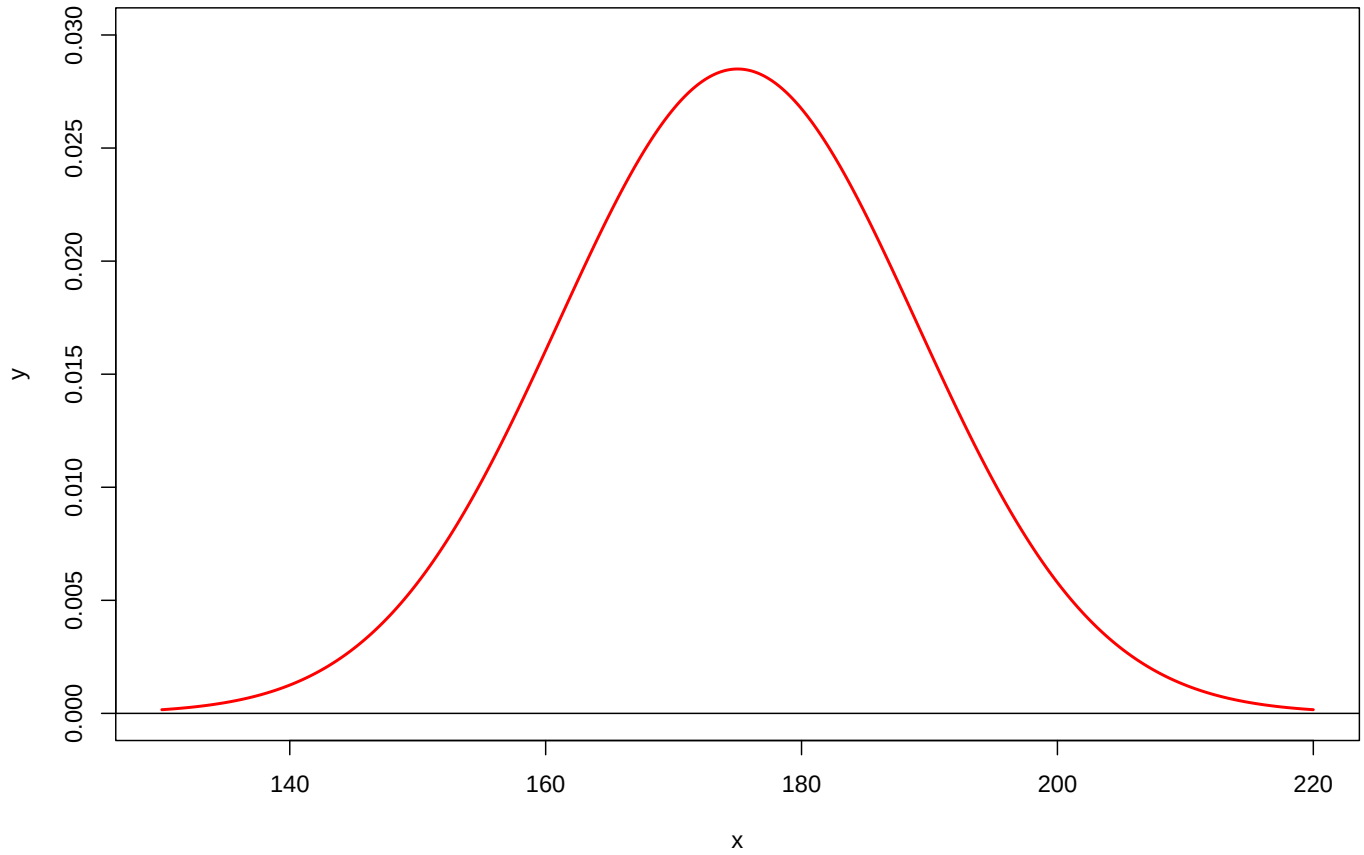
- **Normal distribution** $y \sim N(\mu, \sigma^2)$
  - conjugate prior for mean μ: $\mu \sim N(m, s^2)$
  - conjugate prior for variance σ^2: $\sigma^2 \sim IG(a, b)$ which is equivalent to
  - conjugate prior for precision $\lambda = \frac{1}{\sigma^2}$: $\lambda \sim G(a, b)$
- **Exponential distribution** $y \sim Ex(\lambda)$
  conjugate prior for rate λ: $\lambda \sim Ga(a, b)$

Non-informative Priors

- **'noninformative prior'**
  - **Jeffrey's prior**
    Idea: invariant under diffeomorph parameter transformations
    It is determined based on the Fisher information of the likelihood function.
    $\pi_J = [det(\mathcal{I})]^{-\frac{1}{2}}$
  - **reference priors**
    These are specifically chosen priors which are linked to asymptotic properties of the posterior.
  - **Maximum Entropy prior**
    maximizes the entropy, i. e. prior uncertainty about the parameter with side conditions

Simple Example - body sizes - prior

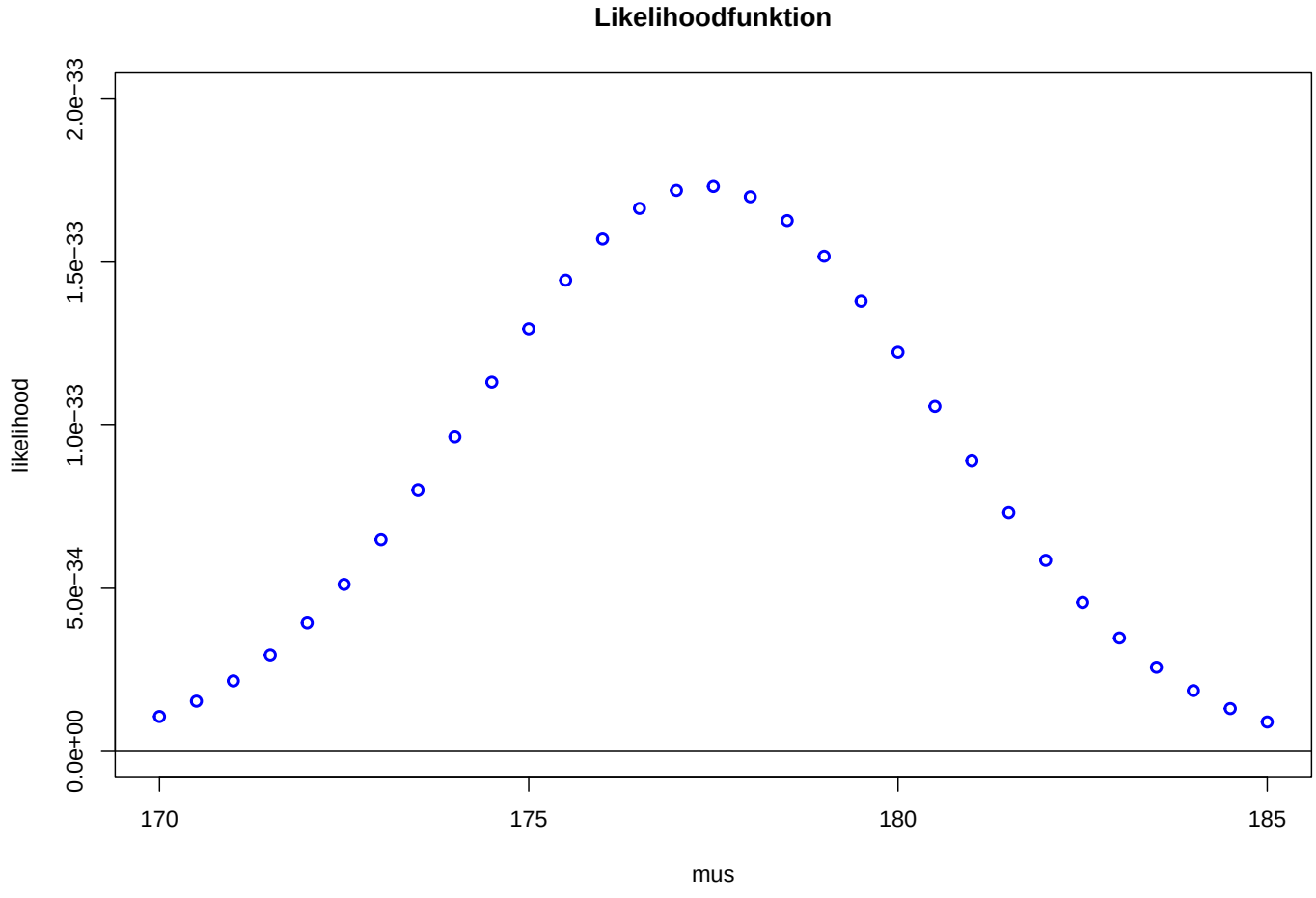
The prior information is that body sizes are normally distributed with a mean of 175 cm and a standard deviation of 14 cm.



Simple Example - body sizes - Likelihood

We start out with body sizes of persons measured in cm: 167, 169, 189, 182, 187, 173, 184, 181, 178, 182, 187, 184, 187, 187, 162, 179, 163, 159, 169, 179

This results in the following Likelihood function $\prod_{i=1}^{n} f(x_i|\mu)$ for some selected values of μ



Simple Example - body sizes - posterior

We obtain the posterior distribution by combining the **prior distribution**

$$\mu \sim \mathcal{N}(175, 14)$$

$$\pi(\mu) = \frac{1}{\sqrt{2\pi}14} e^{-\frac{1}{2}\left(\frac{\mu-175}{14}\right)^2}$$

and the **Likelihood function**

$$L(x|\mu) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}15} e^{-\frac{1}{2}\left(\frac{x_i-\mu}{15}\right)^2}$$

calculating

$$\pi(\mu|x) \propto L(x|\mu) \cdot \pi(\mu)$$

As the marginal distribution is simply a constant for specific data we leave it out of the calculations like all other constants which will have to be adapted in such a way that the posterior distribution is actually a probability distribution with area under the curve $= 1$.

Simple Example - body sizes - posterior

$$\pi(\mu|x) \propto \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}15} e^{-\frac{1}{2}\left(\frac{x_i-\mu}{15}\right)^2} \cdot \frac{1}{\sqrt{2\pi}14} e^{-\frac{1}{2}\left(\frac{\mu-175}{14}\right)^2}$$

is expanded to $\pi(\mu|x) \propto e^{-\frac{1}{2}\sum_{i=1}^{n}\left(\frac{x_i-\mu}{15}\right)^2 - \frac{1}{2}\left(\frac{\mu-175}{14}\right)^2}$ and reordering the elements and dropping constants leads us to

$$\pi(\mu|x) \propto e^{-\frac{1}{2}\left(\frac{1}{15^2}\sum_{i=1}^{n}\left(x_i^2-2\mu x_i+\mu^2\right)-\frac{1}{14^2}\left(\mu^2-2\mu 175+175^2\right)\right)}$$

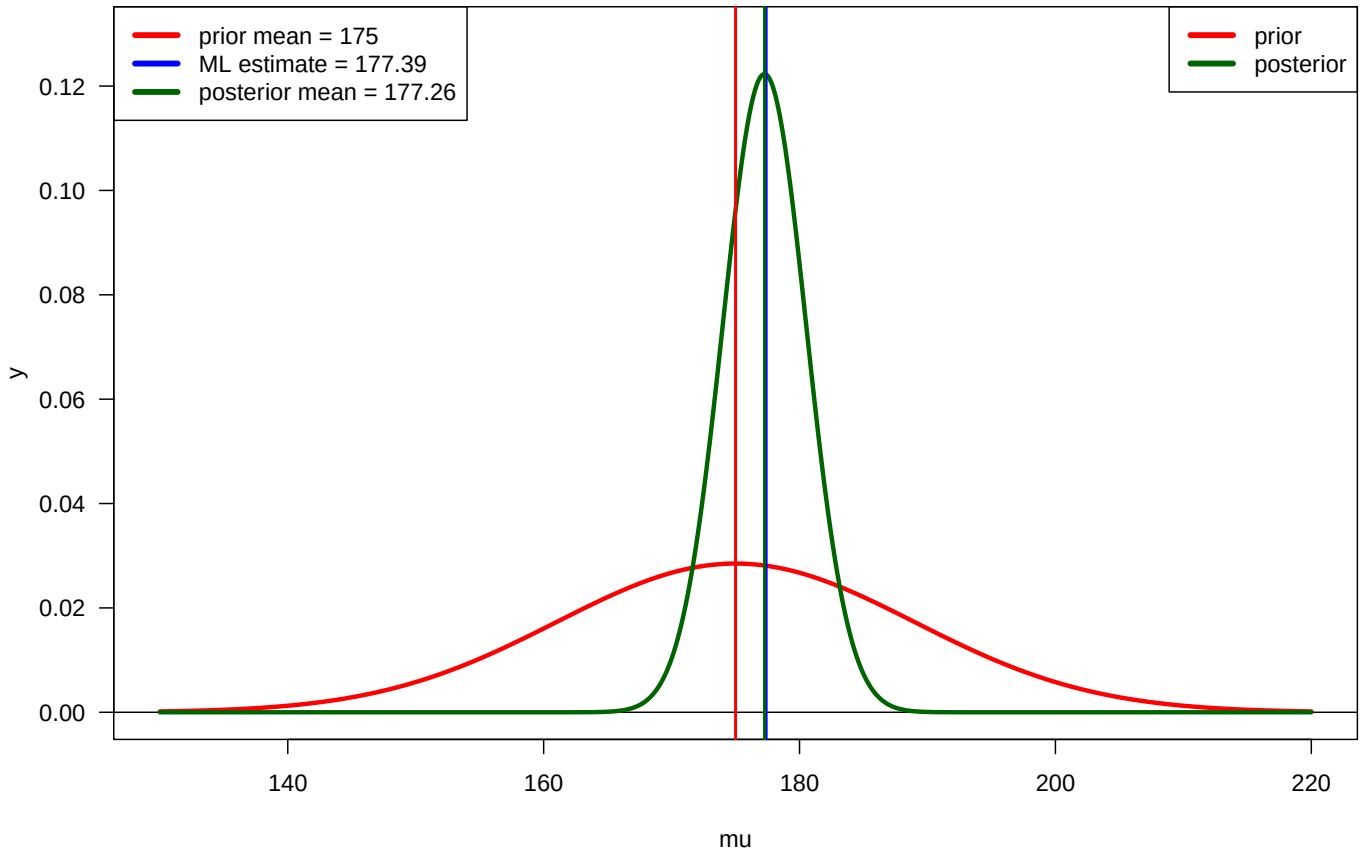
$$\pi(\mu|x) \propto e^{-\frac{1}{2}\left(\frac{1}{15^2}\mu^2 n-2\mu\frac{1}{15^2}\sum_{i=1}^{n} x_i+\frac{1}{14^2}\mu^2-2\mu\frac{1}{14^2}175\right)}$$

$$\pi(\mu|x) \propto e^{-\frac{1}{2}\left(\mu^2\left(\frac{1}{15^2}n+\frac{1}{14^2}\right)-2\mu\left(\frac{1}{15^2}\sum_{i=1}^{n} x_i-\frac{1}{14^2}175\right)\right)}$$

The trained eye can identify the structure of a normal distribution again

$$\boldsymbol{\pi(\mu|x)} \sim \mathcal{N}\left(\frac{\frac{1}{15^2}\sum_{i=1}^{n} x_i - \frac{1}{14^2}175}{\frac{1}{15^2}n+\frac{1}{14^2}}, \left(\frac{1}{15^2}n+\frac{1}{14^2}\right)^{-1}\right)$$

Updating


prior mean = 175
ML estimate = 177.39
posterior mean = 177.26
prior
posterior
0.12
0.10
0.08
0.06
0.04
0.02
0.00
y
140
160
180
200
220
mu

How to get information out of this posterior?

- All Inference about the parameter of interest θ is based on the posterior distribution (and therefore also on the prior).
- The information contained in the posterior distribution can be summarised in different ways as appropriate to the inference goal, e.g.
  - Means, standard deviations, medians. (point estimation)
  - Probability of exceeding a certain threshold, say θ_0, $\Pr(\theta > \theta_0 \mid \mathbf{y})$. (hypothesis tests)
  - Credibility intervals. (interval estimation)

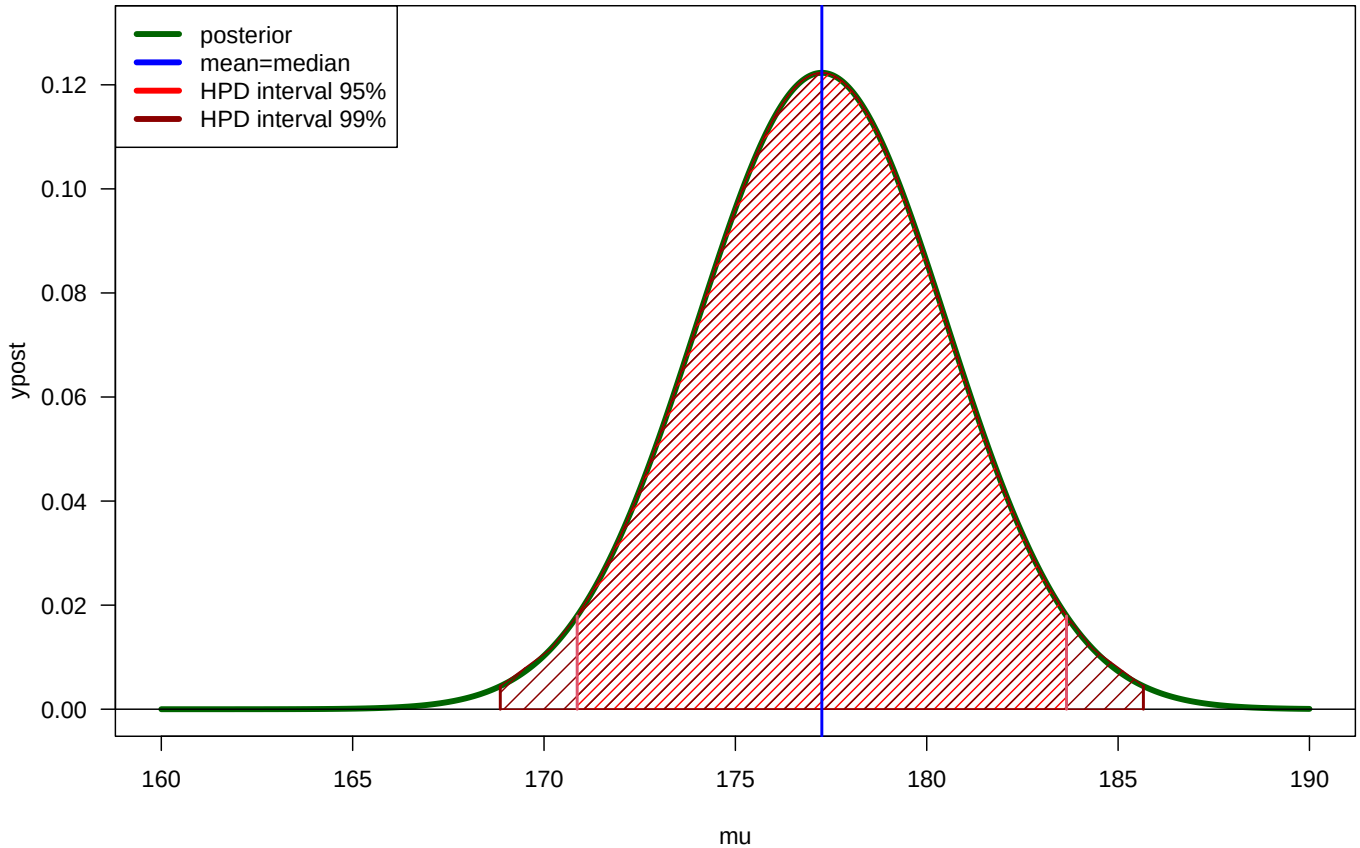
Bayesian estimators and HPD-Intervals

posterior distribution $\rightarrow$ many ways of defining point and interval estimators

The basis for this are "**Loss functions**""

- **posterior mean** $\leftrightarrow$ quadratic L_2 loss
- **posterior median** $\leftrightarrow$ absolute L_1 loss
- **posterior mode** $\leftrightarrow$ 0-1 loss
- **Highest Posterior Density Interval**
  shortest possible interval for a given coverage probability

Example body sizes



Code for graphics, posterior and HPD

```
x<-seq(160,190,by=0.05)
y<-dnorm(x,mean = 175,sd=14)  # prior
sdpost=1/sqrt(1/15^2*length(groessen)+1/14^2)
meanpost=(1/15^2*sum(groessen)+175/14^2)/(1/15^2*length(groessen)+1/14^2)
ypost<-dnorm(x,mean = meanpost,sd=sdpost) # posterior
plot(x,ypost,lwd=4,col="darkgreen",type="l",ylim=c(0,0.13),xlab="mu",las=1)
abline(h=0)
tabpost<-cbind(qnorm(seq(0.025,0.975,by=0.01),mean=meanpost,sd=sdpost),
               dnorm(qnorm(seq(0.025,0.975,by=0.01),mean=meanpost,sd=sdpost),mean=meanpost,sd=sdpost))
tabpost2<-cbind(qnorm(seq(0.005,0.995,by=0.01),mean=meanpost,sd=sdpost),
                dnorm(qnorm(seq(0.005,0.995,by=0.01),mean=meanpost,sd=sdpost),mean=meanpost,sd=sdpost))
polygon(rbind(c(qnorm(0.025,mean=meanpost,sd=sdpost),0),tabpost,c(qnorm(0.975,mean=meanpost,sd=sdpost),0)
polygon(rbind(c(qnorm(0.005,mean=meanpost,sd=sdpost),0),tabpost2,
              c(qnorm(0.995,mean=meanpost,sd=sdpost),0),c(qnorm(0.005,mean=meanpost,sd=sdpost),0)),
        col="darkred",density = 10)
abline(v=meanpost,col="blue",lwd=2)
lines(x=c(qnorm(0.025,mean=meanpost,sd=sdpost),qnorm(0.025,mean=meanpost,sd=sdpost)),
      y=c(0,dnorm(qnorm(0.025,mean=meanpost,sd=sdpost),mean=meanpost,sd=sdpost)),col=2,lwd=2)
lines(x=c(qnorm(0.975,mean=meanpost,sd=sdpost),qnorm(0.975,mean=meanpost,sd=sdpost)),
      y=c(0,dnorm(qnorm(0.975,mean=meanpost,sd=sdpost),mean=meanpost,sd=sdpost)),col=2,lwd=2)
lines(x=c(qnorm(0.005,mean=meanpost,sd=sdpost),qnorm(0.005,mean=meanpost,sd=sdpost)),
      y=c(0,dnorm(qnorm(0.005,mean=meanpost,sd=sdpost),mean=meanpost,sd=sdpost)),col="darkred",lwd=2)
lines(x=c(qnorm(0.995,mean=meanpost,sd=sdpost),qnorm(0.995,mean=meanpost,sd=sdpost)),
      y=c(0,dnorm(qnorm(0.995,mean=meanpost,sd=sdpost),mean=meanpost,sd=sdpost)),col="darkred",lwd=2)
legend("topleft",legend=c("posterior","mean=median","HPD interval 95%","HPD interval 99%"),lwd=4,
       col=c("darkgreen","blue","red","darkred"))
```

Bayesian medical testing

We consider the example of the clinical trial of a simple drug which is meant to find the effectiveness of a treatment for a certain disease. In phase 1 patients are treated based previous findings in vitro and in vivo studies in animals and/or humans. In phase 2 a large number of patients receives treatment based on outcomes of the phase 1 study.

In this sense this is the classical way of Bayesian learning through "updating" which is the reason why in the U.S. the FDA wants all trials to be evaluated according to this Bayesian scheme, as it saves patient numbers due to retaining information. In the end, our goal is to learn about the probability of the drug showing a positive effect for a patient p which is basically the proportion of an underlying Binomial process.

How to do Bayesian Inference for Clinical Trials?

Let us consider $Y \sim Bin(n_1, \theta)$ with n_1 being the total number of patients treated in the Phase 1 trial and θ the unknown proportion of positive treatment outcomes. Here Y counts the number of patients with positive outcomes.

We note that the likelihood

$$L(\theta|y_1) = \binom{n_1}{y_1}\theta^{y_1}(1-\theta)^{n_1-y_1} \propto \theta^{y_1}(1-\theta)^{n_1-y_1} = \theta^{\alpha-1}(1-\theta)^{\beta-1}$$

which is the kernel of a $Beta(\alpha, \beta)$ distribution with $\alpha = y_1 + 1$ and $\beta = n_1 - y_1 + 1$.

Therefore, the parameters a and b refer to the number of paitents with positive outcomes and non-desired outcomes respectively regularised by adding 1.

Assume that 50 patients were treated and out of those, 30 had a desired outcome. Then, y=30 and $n_1 - y_1$=20.

How to intrepret conjugate priors' hyperparameters?

We will show that

$$\theta \sim Beta(a, b) \quad prior \quad ----->$$
$$\theta|y \sim Beta(a + y_1, b + n_1 - y_1) \quad posterior$$

The trick here is that the prior distribution and the likelihood distribution have the same basic structure, as we have already seen in the normal distribution example for human sizes. Further assume that we start with **prior** information on the drug efficacy rate in a "naive" way and assume a 50% probability, then we can encode this for example as $Beta(5, 5)$ **prior** for θ. The parameters a and b of the Beta distribution mean that out of 10 "previous" patients a=5 had a positive effect and b=5 had no desired effect.

Under these circumstances, given the observed sample, one could learn about the proportion of patients with positive effects as an estimate of the drug efficacy rate that it follows a Beta distribution $\theta|y_1 \sim Beta(35, 25)$.

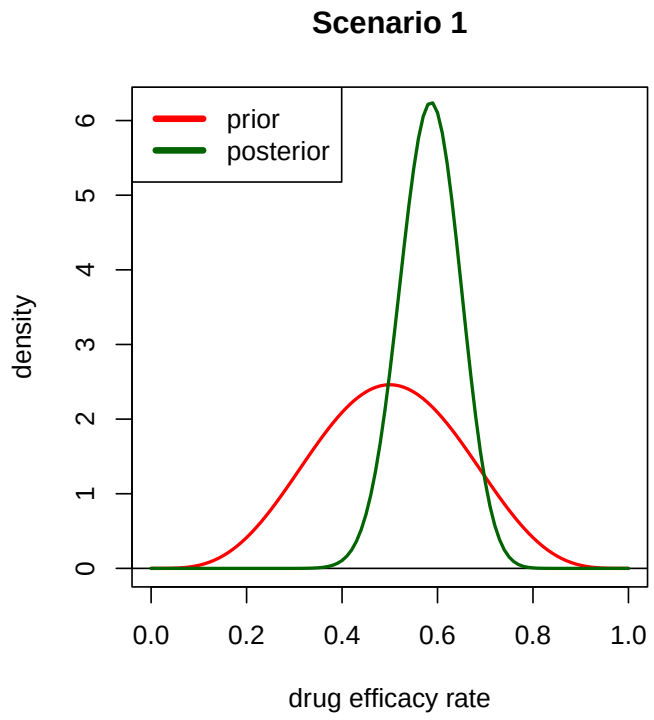
Changing the prior - Getting information into the model

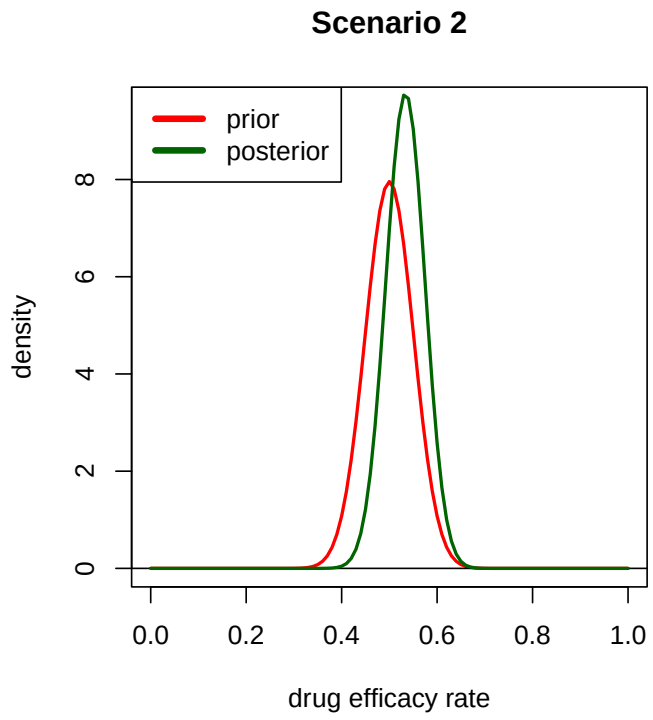
If we had used a different prior distribution which carried the same information such as a $Beta(50, 50)$ **prior** for θ. Then parameters a and b of the Beta distribution mean that out of 100 "previous" patients a=50 had a positive effect and b=50 had no desired effect.

Combining this prior information with the observed sample, one could learn about the proportion of spam emails that it follows a Beta distribution $\theta|y_1 \sim Beta(80, 70)$.

Obviously, we have much more prior evidence and therefore the prior observations have more influence compared to the data. This effect is called **informative** and such a prior is an **informative prior**.

Clinical Trial example - Updating the drug efficacy rate





Phase 2 Trial - Informativeness as a Feature

Similar to before $Y \sim Bin(n_2, \theta)$ with n_2 being the total number of patients treated in the Phase 2 trial and θ the unknown proportion of positive treatment outcomes, again Y counts the number of patients with positive outcomes.

We already saw that with a prior of $\theta \sim Beta(a, b)$ we receive a posterior $\theta|y_1 \sim Beta(a + y_1, b + n_1 - y_1)$ after the Phase 1 of clinical trials which provides the most reasonable prior $\theta \sim Beta(a + y_1, b + n_1 - y_1)$ for the Phase 2 clinical trials which will produce $\theta \sim Beta(a + y_1 + y_2, b + (n_1 - y_1) + (n_2 - y_2))$ as the appropriate posterior.

Clinical Relevance

Again based on the absolute numbers of patients in the Phase 2 trial (n_2, y_2) relative to the absolute numbers of patients in the Phase 1 trial (n_1, y_1) the prior will be more or less informative. In any case the Bayesian Inference allows to naturally blend Phase 1 and 2 trials (and possible lab trials) as if they had been a single larger trial, whereas this is impossible for classical inference resulting in smaller required patient number for trials to obtain similar precision of results (HPDI).

Because of this natural feature of Bayesian updating, the FDA has decided that for ethical reasons all drug tests are to be planned and evaluated based on this Bayesian Inference Procedure.

R Code for scenario

```
xrate<-seq(0,1,by=0.01)
prior1<-dbeta(xrate,shape1 = 5,shape2 = 5)
posterior1<-dbeta(xrate,shape1 = 35,shape2 = 25)
prior2<-dbeta(xrate,shape1 = 50,shape2 = 50)
posterior2<-dbeta(xrate,shape1 = 80,shape2 = 70)
par(mfrow=c(1,2))
plot(xrate,prior1,xlab="spam rate",ylab="density",main="Scenario 1",
     type="l",col="red",ylim=c(0,6.2),lwd=2)
abline(h=0)
lines(xrate,posterior1,col="darkgreen",lwd=2)
legend("topleft",legend=c("prior","posterior"),col=c("red","darkgreen"),lwd=4)
plot(xrate,prior2,xlab="spam rate",ylab="density",main="Scenario 2",
     type="l",col="red",ylim=c(0,9.52),lwd=2)
abline(h=0)
lines(xrate,posterior2,col="darkgreen",lwd=2)
legend("topleft",legend=c("prior","posterior"),col=c("red","darkgreen"),lwd=4)
```

Exercises - Task 1

We recalculate the estimation of the prevalence of Covid19 in spring 2020. Samples from 1279 persons were analysed with PCR testing procedures. Out of all those not a single randomly selected person was tested positively. This obviously breaks standard testing mechanisms for extimating the proportion of infected person in Austria.

However, additional information is available from similar tests in Germany which had a comparable behaviour of the spread of the disease at that time. In the same time span 4 positive cases out of 4068 had been found.

1. Build a Beta prior distribution for this Binomial scenario, which encodes the information of the German study.

   Reweight both parameters compared to the original counts with a factor of $\frac{1}{10}$

   .
2. Build the corresponding Binomial model for the number of people suffering from the disease based on the 1279 test. Obtain the theoretical posterior distribution for this scenario.
3. Plot the posterior density and obtain the point estimators and 95% Highest posterior density interval of the prevalence of Covid19 (=proportion of inhabitants suffering from the disease).
4. Explain why Statistik Austria chose this method instead of simulation-based or frequentist inference for obtaining intervals of the prevalence.

Exercises - Task 2

We revisit linear models and their residual distributions. We have already learned that the distribution of residuals is assumed to be normal. Therefore, the Bayesian linear modelling will assume a normal distribution for the data

$$y \sim N(x^T \beta, \sigma^2)$$

for a single explanatory variable scenario, we will therefore consider the inference of the linear model's coefficient β and the residual variance σ^2.

1. Define conjugate priors for the coefficient parameter and the residual means independently. Explain how the parameters can be set to be uninformative. Compare different choice of prior parameters.
2. Build the corresponding normal model the regression inference. Obtain the theoretical posterior distribution for both parameters separately assuming the other one to be "known".
3. Provide the formulae for point estimators and 95% Highest posterior density interval of the regression parameters separately assuming the other one to be "known".
4. Test this with the data from R: dataset DNase and model

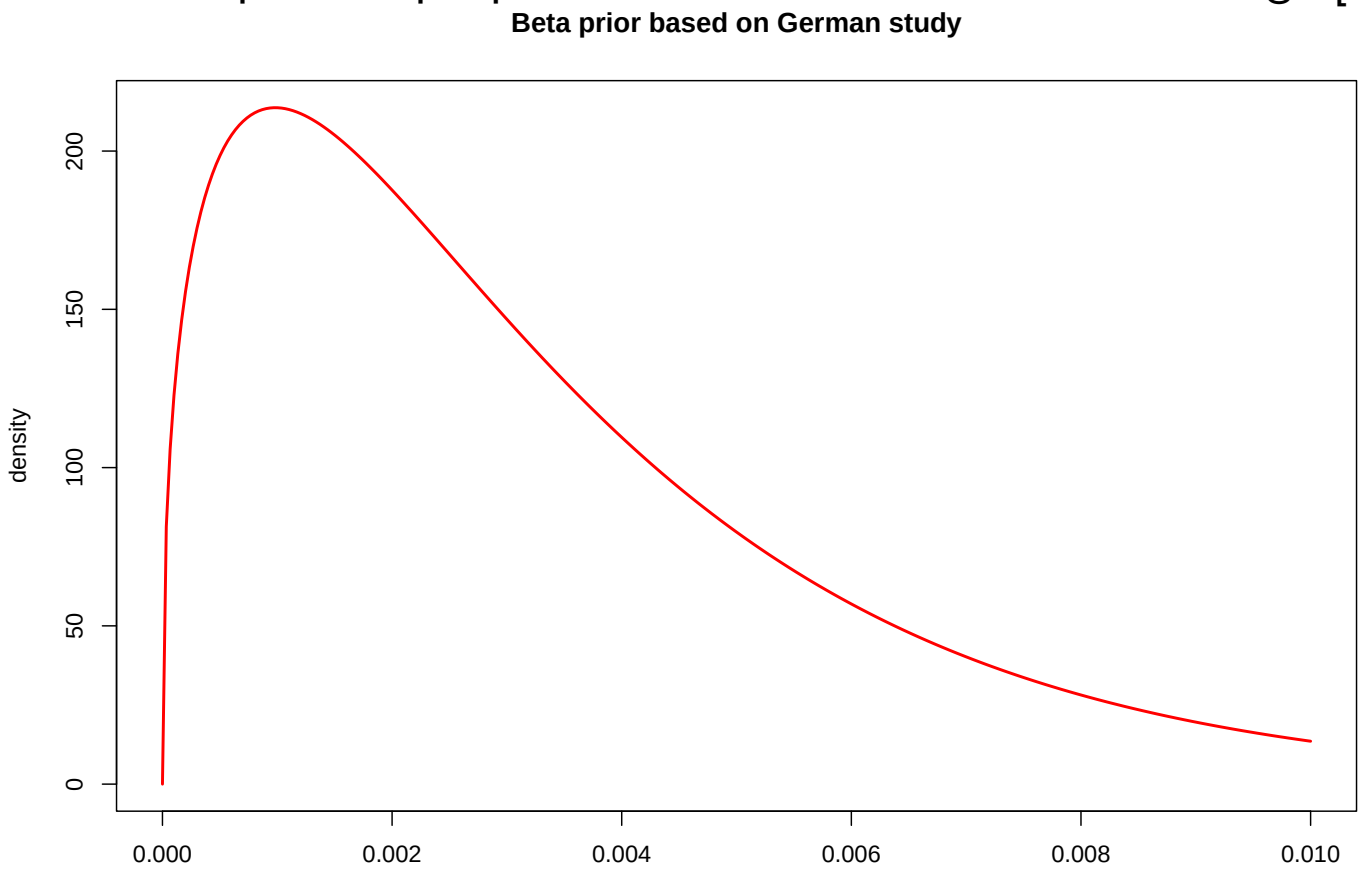
```
lm(density~I(conc)^(1/2),data=DNase)
```

Compare the Bayesian against the frequentist results.

Stepwise Solution - Exercise 1

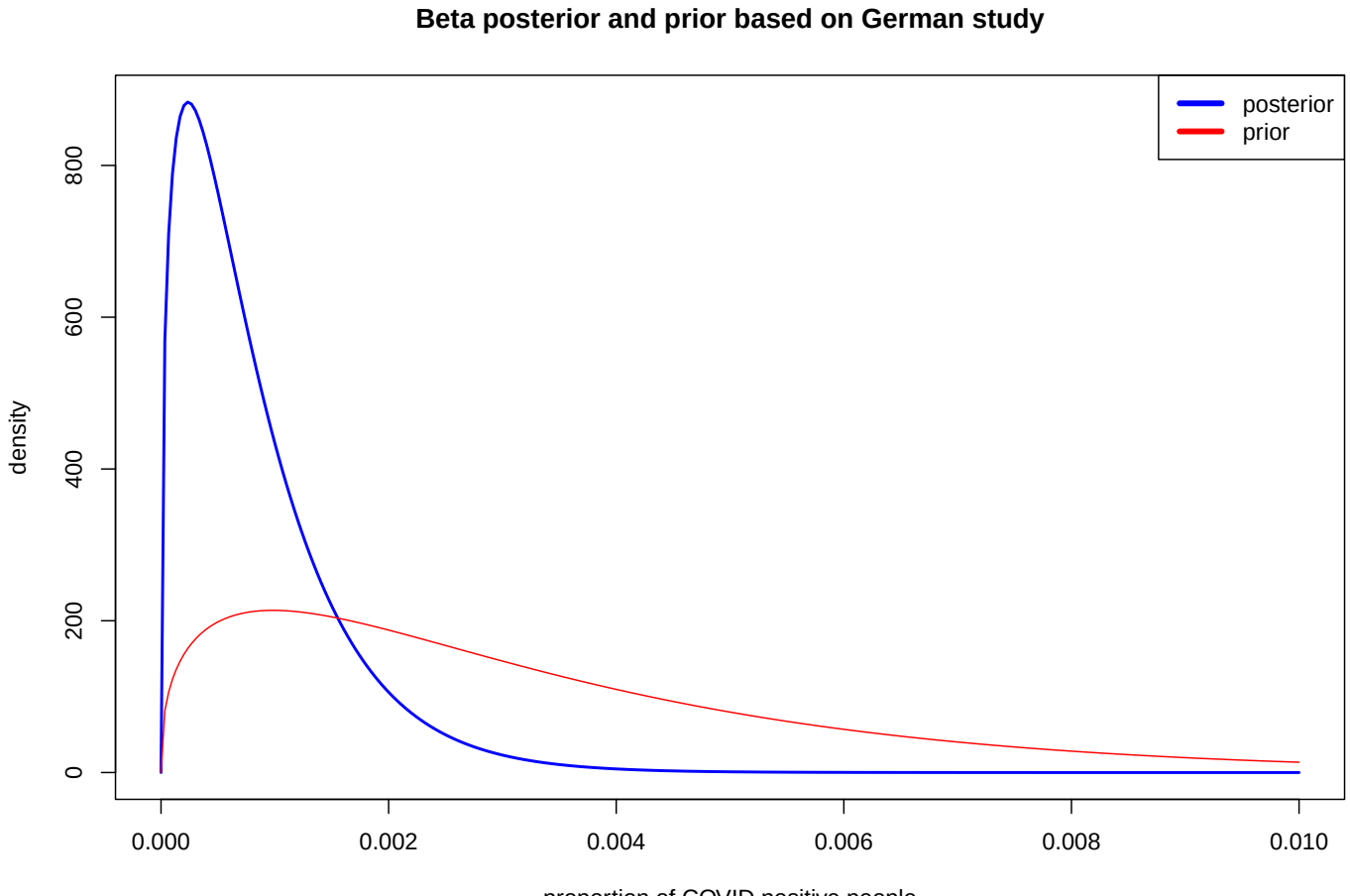
Beta Prior Distribution

We first build a beta prior distribution for the given binomial scenario which encodes the information of the German study. To that end, we choose a beta prior distribution with parameters $\alpha = y + 1 = 4/10 + 1 = 1.4$ and $\beta = (n - y)/10 + 1 = (4,068 - 4)/10 + 1 = 407.4$ (using the same notation as given in the lecture slides). Here, we have reweighted the German observations with a factor of $\frac{1}{10}$. We now plot the resulting prior that models the proportion of COVID positive people. We restrict the x-axis to the range $[0\%, 1\%]$.



Theoretical Posterior Distribution

We now build the corresponding binomial model for the number of people suffereing from the disease based on the 1,279 people with negative tests to obtain the theoretical posterior distribution. To that end, we know from the lecture that if the prior is given by $Beta(\alpha, \beta)$, the posterior is given for a binomial likelihood function by $Beta(\alpha + y, \beta + n - y)$, where n is in our case 1,279 and y corresponds to the number of positive tests 0. Hence, the posterior distribution is given by $Beta(\alpha, \beta + 1,279)$, where α and β are defined above.



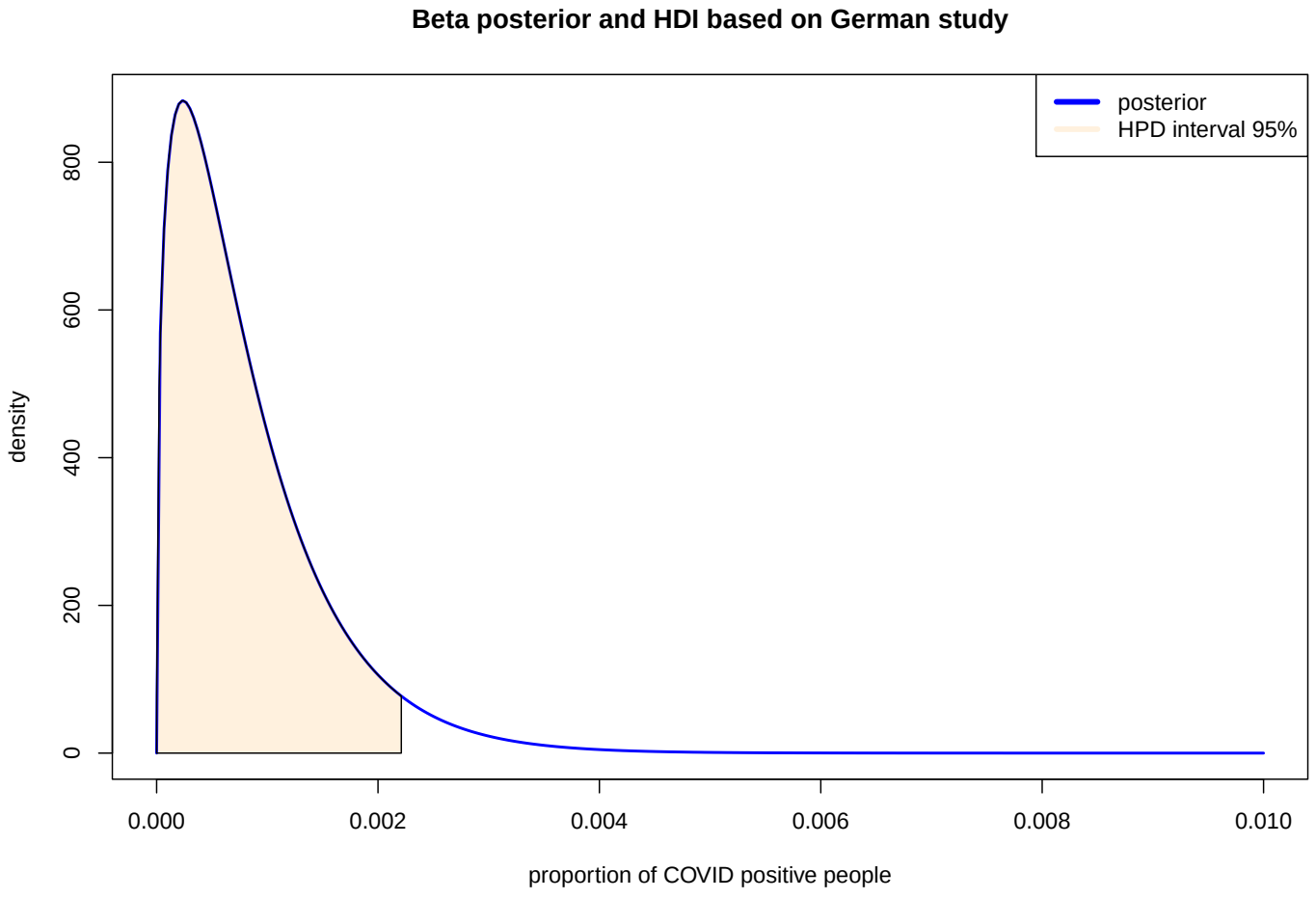
Plotting Posterior, Point Estimators and HPD

We now calculate the mean, mode and median of the posterior distribution. As we already know that the posterior follows a beta distribution, we can directly calculate those parameters by using the explicit formulas for the beta distribution. Note that for the median we take a common approximation $\textit{median} = \frac{\alpha - \frac{1}{3}}{\alpha + \beta - \frac{2}{3}}$, since there is no closed-form formula of the median. Furthermore, we determine the highest posterior density (HPD) interval by using the package `HDInterval`.

Tabelle 1: Point esimators for the posterior distribution based on the prior that uses German COVID-19 cases

Estimator	Posterior Point Estimate
Mean	0.0008294330232833 7
Mode	0.0002372619965597 01
Median	0.0006321986684315 55
lower 95%-HPD boundary	1.972466469196 19e-07
upper 95%-HPD boundary	0.0022106040984371 6

Plot of posterior density with the 95% HPD interval



Explanation why Statistik Austria chose this method instead of simulation based or frequentist inference for obtaining intervals of the prevalence

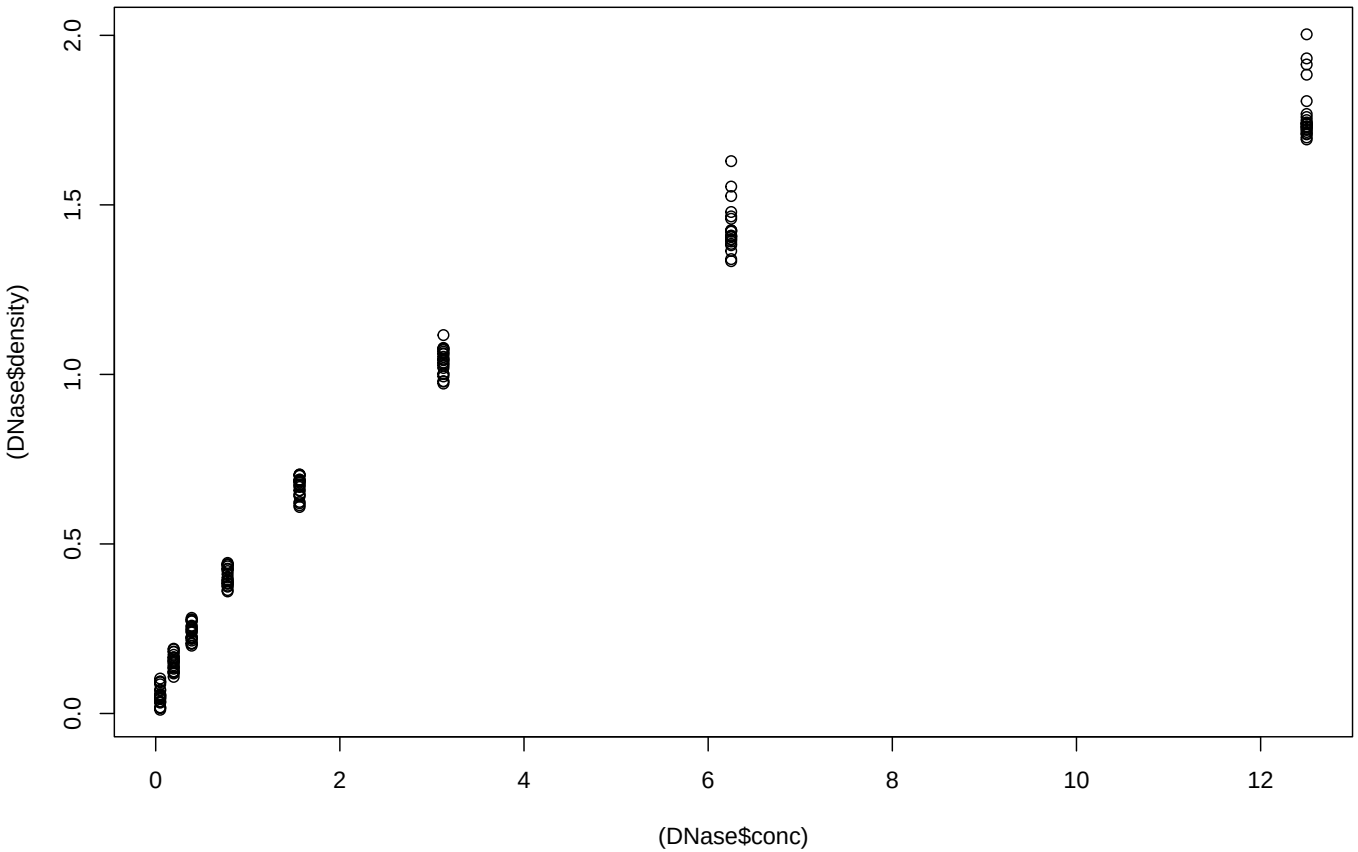
Statistik Austria had to choose a Bayesian approach, because in the Austrian data, there were no positive tests. Therefore, any frequentist inference would have resulted in point estimates of 0%. Still, we know that the true estimate of COVID-19 prevalence is positive (and not 0).

On the contrary, a Bayesian approach allows us to consider all parameters and model coefficients to be random. We can include prior information on the COVID-19 prevalance which we can update based on the available data. In our case, the German data seems to be relatively representative of Austria and can therefore be used to construct a good prior. Hence, the Bayesian approach estimates the uncertainty of the COVID-19 prevalance more sound than a simulation based or frequentist inference approach on the Austrian data.

Stepwise Solution - Exercise 2

For Bayesian linear modelling, we assume in the following a normal distribution for the data

$$y \sim N(x\beta, \sigma^2).$$



Bayesian Regession Model

We now define conjugate priors for the coefficient parameter β and the residual variance σ^2 independently. We start by defining the prior of β to be normally distributed with mean m and variance s^2, i.e.

$$\beta \sim N(m, s^2).$$

We then define the prior of σ to be inverse-Gamma distributed with shape parameters a, b, i.e.

$$\sigma^2 \sim IG(a, b).$$

Equivalently, we can also consider $\lambda = \frac{1}{\sigma^2} = G(a, b)$. These priors are the conjugate priors of the likelihood $y \sim N(x\beta, \sigma^2)$.

To make the priors informative, we would have to choose a small variance parameter s^2 of the normal distribution for the prior of β (e.g. to $s^2 = 0.1$). We can take $m = 0$ as a mean of the normal distribution (which corresponds to the case when the independent variable does not explain the dependent variable). For λ, we would have to choose $a = b$ both large. This corresponds to a prior mean of σ being 1, while the prior has small variance (e.g. $a = b = 100$, which corresponds to a variance of $1/100$).

To get uniformative priors, we can increase the variance s^2 of the prior for β (e.g. $s^2 = 1,000$). We can again take $m = 0$ as a mean of the normal distribution. For λ, we can set $a = b = 0.5$, which would correspond to a prior mean of σ being 1, while the prior has comparably large variance 2 (and is in this sense uniformative).

Posterior distribution of Regression Coefficients

We now build the corresponding normal model for the regression inference. We obtain the theoretical posterior distribution for both parameters β, λ separately assuming the other one to be "known". We start with β and assume λ (or equivalently $1/\sigma^2$) to be known:

$$\begin{aligned}\pi(\beta|x,y) &\propto \mathcal{L}(x,y|\beta)\cdot\pi(\beta)\\ &= \prod_i \frac{1}{\sqrt{2\pi}\sigma}\exp\left(-\frac{1}{2\sigma^2}(x_i\beta-y_i)^2\right)\cdot\frac{1}{\sqrt{2\pi}s}\exp\left(-\frac{1}{2s^2}(m-\beta)^2\right)\\ &\propto \exp\left(-\frac{1}{2\sigma^2}\sum_i(x_i\beta-y_i)^2-\frac{1}{2s^2}(m-\beta)^2\right)\end{aligned}$$

By properly scaling, we therefore find

$$\pi(\beta|x,y)\sim N\left(\frac{\left(\frac{\sum_i x_i y_i}{\sigma^2}+\frac{m}{s^2}\right)}{\left(\frac{1}{s^2}+\sum_i\frac{x_i^2}{\sigma^2}\right)},\left(\frac{1}{s^2}+\sum_i\frac{x_i^2}{\sigma^2}\right)^{-1}\right).$$

Posterior Distribution of the Model Precision

We now proceed by determining the posterior distribution of λ (and assume that β is known):

$$\begin{aligned}
\pi(\lambda|x,y) &\propto \mathcal{L}(x,y|\lambda)\cdot\pi(\lambda) \\
&= \prod_{i=1}^{n}\frac{\sqrt{\lambda}}{\sqrt{2\pi}}\exp\left(-\frac{\lambda}{2}(x_i\beta-y_i)^2\right)\cdot\frac{\lambda^{a-1}\exp(-b\lambda)b^a}{\Gamma(a)} \\
&\propto \lambda^{n/2+a-1}\exp\left(-\frac{\lambda}{2}\sum_i(x_i\beta-y_i)^2-b\lambda\right) \\
&= \lambda^{n/2+a-1}\exp\left(-\lambda\left(\frac{\sum_i(x_i\beta-y_i)^2}{2}+b\right)\right).
\end{aligned}$$

By properly scaling, we therefore find

$$\pi(\lambda|x,y)\sim G\left(\frac{n}{2}+a,\frac{\sum_i(x_i\beta-y_i)^2}{2}+b\right).$$

Deriving HPDIs

Based on the posterior distributions that we found in the previous task, we can now determine the mean, mode and median of the posterior distributions for β and σ. We use the well-known property of normal distributions that their mean, median and mode coincide. As there is no closed-form solution for the median of the Gamma distribution, we can only provide the mode and mean as explicit formulae:

$$\mathbb{E}(\pi(\beta|x,y)) = \mathit{median}(\pi(\beta|x,y)) = \mathit{mode}(\pi(\beta|x,y)) = \frac{\left(\frac{\sum_i x_i y_i}{\sigma^2} + \frac{m}{s^2}\right)}{\left(\frac{1}{s^2} + \sum_i \frac{x_i^2}{\sigma^2}\right)}$$

$$\mathbb{E}(\pi(\lambda|x,y)) = \frac{\frac{n}{2} + a}{\frac{\sum_i (x_i\beta - y_i)^2}{2} + b}$$

For the 95%-HPD interval of the posterior of β, we can use the inverse distribution function of $N\left(\frac{\left(\frac{\sum_i x_i y_i}{\sigma^2} + \frac{m}{s^2}\right)}{\left(\frac{1}{s^2} + \sum_i \frac{x_i^2}{\sigma^2}\right)}, \left(\frac{1}{s^2} + \sum_i \frac{x_i^2}{\sigma^2}\right)^{-1}\right)$, i.e. the quantile function (since the normal distribution is symmetric).

HPD Interval for regression

$$HDP_{lower} = N^{-1}\left(0.025, mean = \frac{\left(\frac{\sum_i x_i y_i}{\sigma^2} + \frac{m}{s^2}\right)}{\left(\frac{1}{s^2} + \sum_i \frac{x_i^2}{\sigma^2}\right)}, variance = \left(\frac{1}{s^2} + \sum_i \frac{x_i^2}{\sigma^2}\right)^{-1}\right),$$

$$HDP_{upper} = N^{-1}\left(0.975, mean = \frac{\left(\frac{\sum_i x_i y_i}{\sigma^2} + \frac{m}{s^2}\right)}{\left(\frac{1}{s^2} + \sum_i \frac{x_i^2}{\sigma^2}\right)}, variance = \left(\frac{1}{s^2} + \sum_i \frac{x_i^2}{\sigma^2}\right)^{-1}\right).$$

As the Gamma distribution is not symmetric, we cannot find the HDP of the posterior of σ in a simple manner. Instead, we have to determine it numerically (e.g. via `hdi` of `HDInterval`).

Bayesian Regression - practical howto

```
library(bayesreg)
bayesregDNase<-bayesreg(density~I(conc^(1/2)),data=DNase)
summary(bayesregDNase)
```

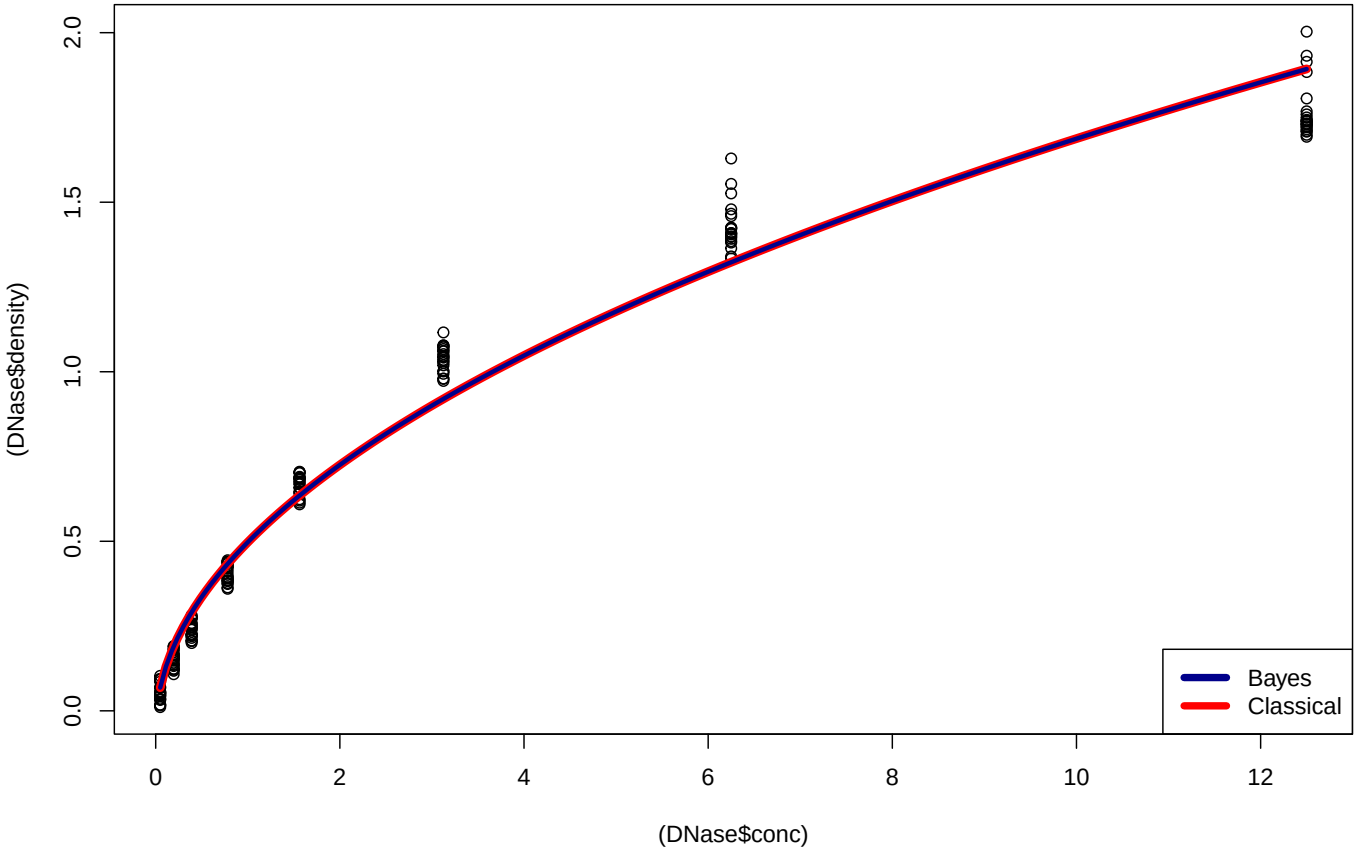
```
## ================================================================================
## |                   Bayesian Penalised Regression Estimation ver. 1.2          |
## |                     (c) Enes Makalic, Daniel F Schmidt. 2016-2021            |
## ================================================================================
## Bayesian Gaussian ridge regression                          Number of obs    =      176
##                                                             Number of vars   =        1
## MCMC Samples    =   1000                                    std(Error)       = 0.089716
## MCMC Burnin     =   1000                                    R-squared        =   0.9778
## MCMC Thinning   =      5                                    WAIC             =   -173.5
##
## --------------+-----------------------------------------------------------------
##     Parameter |  mean(Coef)  std(Coef)    [95% Cred. Interval]      tStat    Rank        ESS
## --------------+-----------------------------------------------------------------
## I(conc^(1/2)) |     0.55038    0.00651      0.53864    0.56372     84.568       1 **    1000
##         _cons |    -0.05329    0.01114     -0.07474   -0.03241          .       .          .
## --------------+-----------------------------------------------------------------
```

Classical Regression - a comparison

```
lmDNase<-lm(density~I(conc^(1/2)),data=DNase)
summary(lmDNase)
```

```
##
## Call:
## lm(formula = density ~ I(conc^(1/2)), data = DNase)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.20009 -0.05060 -0.01110  0.05827  0.30600
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)   -0.053297   0.011081   -4.81 3.26e-06 ***
## I(conc^(1/2))  0.550521   0.006287   87.57  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08897 on 174 degrees of freedom
## Multiple R-squared:  0.9778, Adjusted R-squared:  0.9777
## F-statistic:  7668 on 1 and 174 DF,  p-value: < 2.2e-16
```

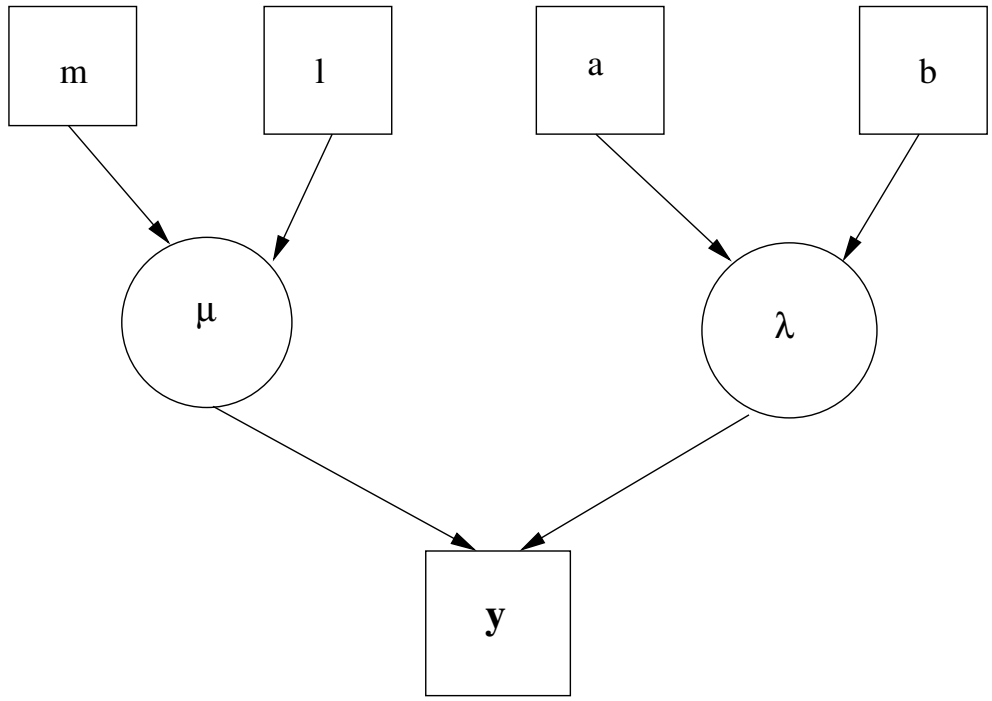
Plotting the results



Directed Acyclic Graphs

Bayesian paradigm:
consider parameters as random variables
$\rightarrow$ add prior on parameter, additional latent parameters

Directed acyclic graph (DAG): visualisation of hierarchical model



Mathematical Model

The DAG illustrates the following distributions and their parameters:

$$\begin{aligned} y|\mu,\lambda &\sim N(\mu,\lambda^{-1}) \\ \mu|m,l &\sim N(m,l^{-1}) \\ \lambda|a,b &\sim Gamma(a,b) \end{aligned}$$

based on a Likelihood function

$$f(y|\mu,\lambda) \propto \lambda^{n/2}\exp\left\{-\frac{\lambda}{2}\sum_{i=1}^{n}(y_i-\mu)^2\right\}.$$

The directions visualise the hierarchical model structure with the data and its likelihood at the centre and priors for these parameters on the next hierarchical level.

Mathematical Model

The Normal- and Gamma- distribution are natural conjugate priors for any models based on normally distributed likelihood functions, such as the equivalent of t-test for Bayesian settings based on normal distributions priors and posteriors of the mean, the linear regression model and probit regression model. All have posteriors and "full conditional" distributions of the following forms.

$$\begin{aligned} \mu|\lambda, y &\sim N(m^*, l^*) \\ m^* &= l^* \cdot (l \cdot m + \lambda \cdot n \cdot \bar{y}) \\ l^* &= l + n \cdot \lambda \end{aligned}$$

are the parameters of the normal posterior of the mean and

$$\begin{aligned} \lambda|\mu, y &\sim Gamma(a^*, b^*) \\ a^* &= a + \frac{n}{2} \\ b^* &= b + \frac{1}{2} \sum_{i=1}^{n} (y_i - \mu)^2 \end{aligned}$$

of the Gamma-posterior of the precision (inverse variance) where typical for Bayesian inference the Likelihood-based pivot is **biased** by the prior information.

Posterior and Full conditionals

In the above example we used the term posterior, although the normal posterior for the mean only applies, if the precision where known, and the Gamma posterior for the precision only applies, if the mean where known. You have dealt with these **1-dimensional posterior** scenarios in your last exercises.

If both parameters where to be determined simultaneously, a **two-dimensional posterior** would have to be constructed. Its **marginal distribution** for every value of λ would be the normal posterior of the mean μ, while its **marginal distribution** for every value of μ would be the Gamma posterior of the precision λ. Thus, conditional on the value of λ the 1-dimensional normal posterior for the mean is the **marginal distribution** of this 2-dimensional posterior and therefore referred to as **full conditional posterior**. These **full conditionals** wil be the basic building stones of Gibbs samplers for MCMC simulation.

Revision of Monte Carlo sampling

- Monte Carlo sampling is the predominant method of Bayesian inference because it avoids asymptotic approximations and can be used in high-dimensions.
- The main idea is to approximate posterior summaries by drawing samples from the posterior distribution, and then using these samples to approximate posterior summaries of interest.
- For example, if $\boldsymbol{\theta}^{(1)}, \ldots, \boldsymbol{\theta}^{(S)}$ are samples from $p(\boldsymbol{\theta} \mid \mathbf{y})$, then the mean of S samples can be used to approximate the posterior mean.
- This only provides approximations of the posterior summaries of interest.
- Many argue that this form of approximation is superior to asymptotic approximations because the Bayes CLT requires the sample size of the dataset to go to infinity and the Monte Carlo approximation requires the number of simulated values to go to infinity.
- In most cases, $S \to \infty$ is cheaper and more realistic than $n \to \infty$.
- But how to draw samples from some arbitrary distribution $p(\boldsymbol{\theta} \mid \mathbf{y})$?

Markov Chain Monte Carlo

Basic MCMC samplers:

- The **Metropolis-Hastings** sampler: most universal sampling scheme
- The **Gibbs** sampler: most commonly used, simple to understand and straightforward to calculate and implement
- The **Reversible Jump** sampler and the **Birth and Death** sampler: deal with varying parameter sizes
- **Hybrid** sampler: combines at least 2 of the sampling approaches

Markov Chain Monte Carlo

Idea: to obtain samples from a distribution without this distribution being explicitly available

Aim: constructing an ergodic Markov chain with stationary distribution ξ in order to acquire samples from that distribution.

plug sampled values into the Monte Carlo integration

in a Bayesian frame work: posterior distributions often analytically intractible for complex, e.g. hierarchical Bayesian models

Metropolis-Hastings sampler

Aim: drawing $(x^{(t)})$ such that $(x^{(t)})$, $t = 0, 1, \ldots$ are a Markov chain with stationary distribution being the objective **target density** ξ

Approach: auxiliary conditional distribution, **proposal density** $q(.|.)$ of a proposed value given the 'old' value.

good proposal

- easy to simulate from or
- symmetric (i.e.$q(x|y) = q(y|x)$) so that it cancels out in the acceptance probability

Metropolis-Hastings sampler

- For $t = 0$: take starting value x_0
- $t > 0$:
  1. generate proposal $Y_t \sim q(y|x^{(t-1)})$
  2. Either

     move to proposed value Y_t with $\alpha(x^{(t-1)}, Y_t)$ or

     stay at old value $x^{(t-1)}$ with $1 - \alpha(x^{(t-1)}, Y_t)$

     where $\alpha(x, y) = \min\left\{\frac{\xi(y)}{\xi(x)}\frac{q(x|y)}{q(y|x)}, 1\right\}$ is the acceptance probability.

     The transition kernel of the Metropolis-Hastings sampler is

$$\mathcal{K}(x, y) = \alpha(x, y)q(y|x) + (1 - \int \alpha(x, y)q(y|x)dy)\delta_x(y)$$

Gibbs sampler

Idea: use **full conditional distributions**

$$\xi_i(x_i|x_1, x_2, \ldots, x_{i-1}, x_{i+1}, \ldots, x_p) \quad i = 1, 2, \ldots, p$$

associated with the target distribution to generate samples from target distribution, if we can sample from these distributions

Thus, unlike the MH sampler the Gibbs sampler is by definition **multidimensional**! (at least two variables are required for the conditional distributions)

Gibbs sampler

- **Gibbs sampling** was proposed in the early 1990s (Geman and Geman, 1984; Gelfand and Smith, 1990) and fundamentally changed Bayesian computing.
  - It is attractive because it can sample from high-dimensional posteriors
  - The main idea is to break the problem of sampling from the high-dimensional joint distribution into a series of samples from low-dimensional conditional distributions. (the full conditional *posteriors*)
- The algorithm is straightforward:
  - One begins by setting initial values for all parameters, $\boldsymbol{\theta}^{(0)} = (\theta_1^{(0)}, \ldots, \theta_d^{(0)})$.
  - Variables are then sampled one at a time from their **full conditional distributions**

$$p(\theta_j \mid \theta_1, \ldots, \theta_{j-1}, \theta_{j+1}, \ldots, \theta_p, \mathbf{y}), \qquad j = 1, \ldots, d.$$

  - Rather than 1 sample from p-dimensional joint, we make p-dimensional samples.

Gibbs sampler

- Generally, given a parameter vector $\theta = (\theta_1, \ldots, \theta_d)$, the Gibbs sampler works as follows.

Step 1 Specify initial values $(\theta_1^{(0)}, \ldots, \theta_d^{(0)})$.

Step 2 For $t = 1, \ldots, T$

2.1 Simulate $\theta_1^{(t)} \sim p(\theta_1 \mid \mathbf{y}, \theta_2^{(t-1)}, \ldots, \theta_d^{(t-1)})$

2.2 Simulate $\theta_2^{(t)} \sim p(\theta_2 \mid \mathbf{y}, \theta_1^{(t)}, \theta_3^{(t-1)}, \ldots, \theta_d^{(t-1)})$

...

2.d Simulate $\theta_d^{(t)} \sim p(\theta_d \mid \mathbf{y}, \theta_1^{(t)}, \theta_2^{(t)}, \ldots, \theta_{d-1}^{(t)})$

Step 3 Discard the first k observations of the chain and compute the summary statistics from the *posterior* distribution based on $(\theta_1^{(k+1)}, \ldots, \theta_d^{(k+1)}), \ldots, (\theta_1^{(T)}, \ldots, \theta_d^{(T)})$

Example: Gibbs sampler

Gibbs Sampler for normal distribution with Normal- and Gamma- conjugate priors. All have posteriors and "full conditional" distributions of the following forms.

The full conditional distributions of

$$\begin{array}{rcc} \mu|\lambda, y & \sim & N(m^*, l^*) \\ & & m^* = l^* \cdot (l \cdot m + \lambda \cdot n \cdot \bar{y}) \\ & & l^* = l + n \cdot \lambda \\ \lambda|\mu, y & \sim & Gamma(a^*, b^*) \\ & & a^* = a + \frac{n}{2} \\ & & b^* = b + \frac{1}{2} \sum_{i=1}^{n} (y_i - \mu)^2 \end{array}$$

are the basis for the Gibbs sampler updating

$$\begin{array}{rcl} \mu^{(t)}|y, \lambda^{(t-1)} & \sim & N(m^* = l^* \cdot (l \cdot m + \lambda^{(t-1)} \cdot n \cdot \bar{y}), l^* = l + n \cdot \lambda^{(t-1)}) \\ \lambda^{(t)}|\mu^{(t)}, y & \sim & Gamma(a^* = a + \frac{n}{2}, b^* = b + \frac{1}{2} \sum_{i=1}^{n} (y_i - \mu^{(t)})^2) \end{array}$$

Application of Bayesian Simulation Software with R

MCMC based Software/Packages

- Stan
- BUGS (**B**ayesian Inference **U**sing **G**ibbs **S**ampling)
- JAGS (**J**ust **A**nother **G**ibbs **S**ampler)
- Other Packages with pre-implemented Samplers

Other Simulation Methods

- INLA (**I**ntegrated **N**ested **L**aplace **A**pproximation)
- ABC (**A**pproximate **B**ayesian **C**omputation)

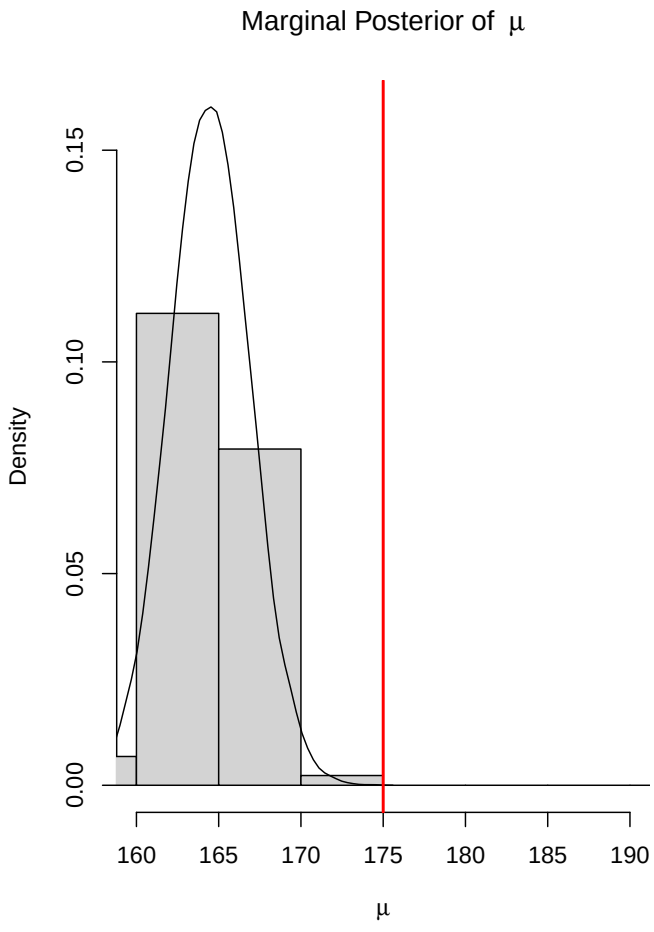
Example - Body Sizes

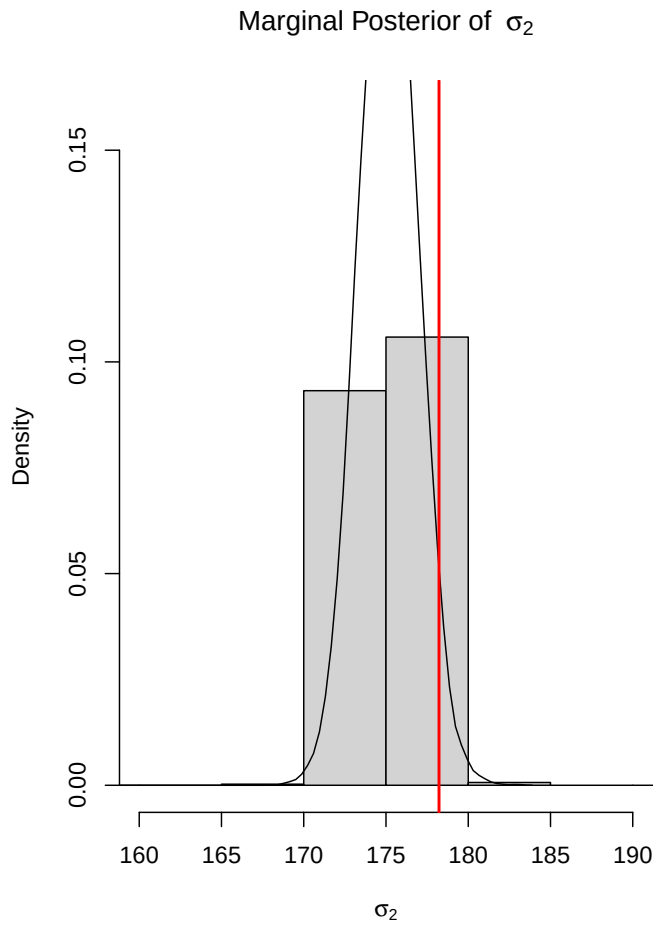
```
## Simulate Data
set.seed(45725)
N = 5000
mu = 175
sigma = 14
sigmasq = sigma^2
y = rnorm(N, mean = mu, sd = sigma)

## set up priors
mu.0 <- 0
tausq.0 <- 100
nu.0 <- .5
sigmasq.0 <- 1
rate.param <- nu.0 * sigmasq.0 / 2
shape.param <- nu.0 / 2
```

Example - Body Sizes - Gibbs Sampler

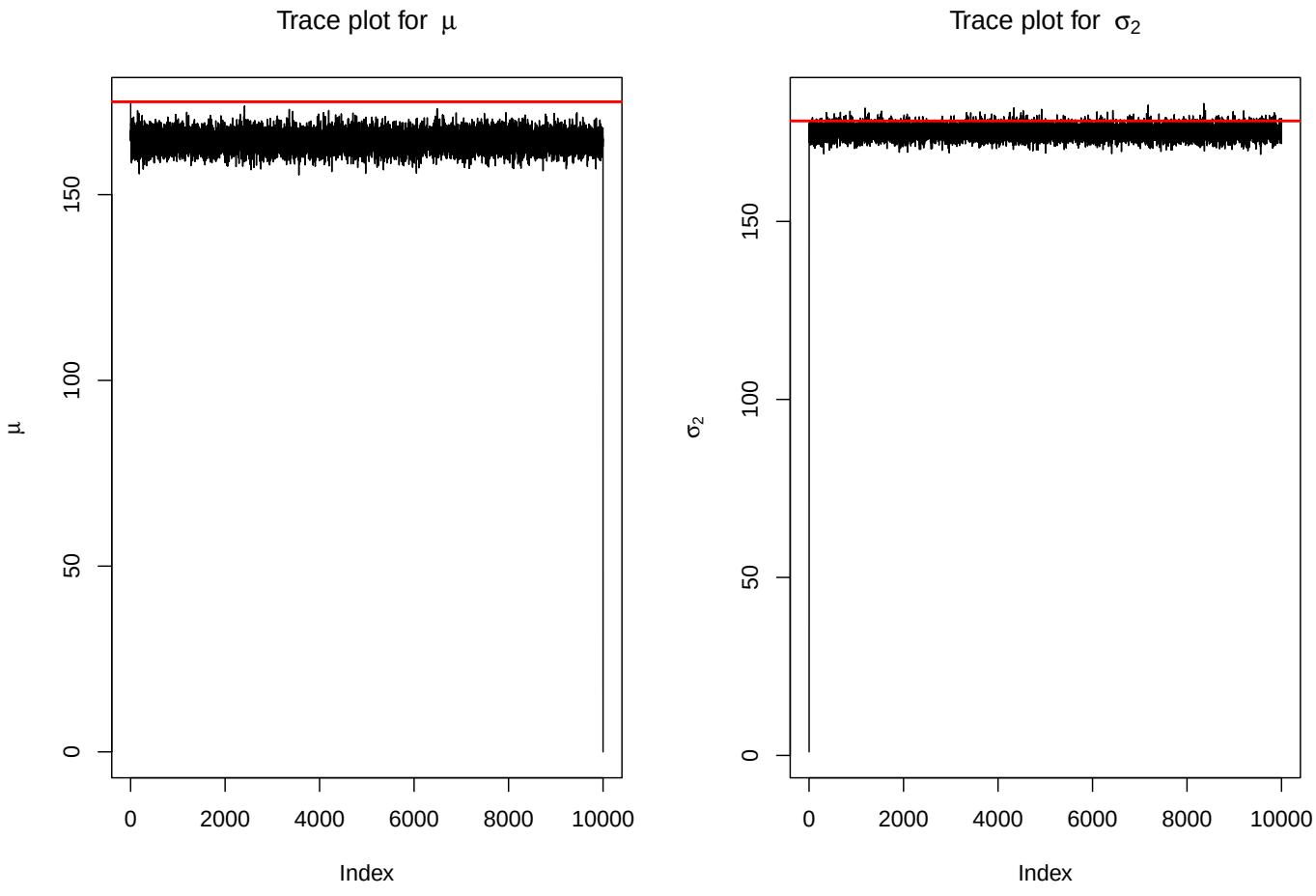
```
### initialize vectors and set starting values
num.sims <- 10000
mu.samples <- rep(0, num.sims)
sigmasq.samples <- rep(1, num.sims)

mean.y <- mean(y)
nu.n <- nu.0 + N

for (iter in 2:num.sims){
  # sample theta from full conditional
  mu.n <- (mu.0 / tausq.0 + N * mean.y / sigmasq.samples[it
  tausq.n <- 1 / (1/tausq.0 + N / sigmasq.samples[iter - 1]
  mu.samples[iter - 1] <- rnorm(1,mu.n,sqrt(tausq.n))

  # sample (1/sigma.sq) from full conditional
  sigmasq.n.theta <- 1/nu.n*(nu.0*sigmasq.0 + sum((y - mu.s
  sigmasq.samples[iter] <- rinvgamma(1,shape = nu.n/2, rate
}
```

Marginal posteriors of mu and sigma





Trace Plots


Trace plot for μ
μ
0
50
100
150
0
2000
4000
6000
8000
10000
Index
Trace plot for σ2
σ2
0
50
100
150
0
2000
4000
6000
8000
10000
Index

Relevance of Priors

```r
## set up priors
mu.0 <- 170
tausq.0 <- 100
nu.0 <- .5
sigmasq.0 <- 1
rate.param <- nu.0 * sigmasq.0 / 2
shape.param <- nu.0 / 2
```

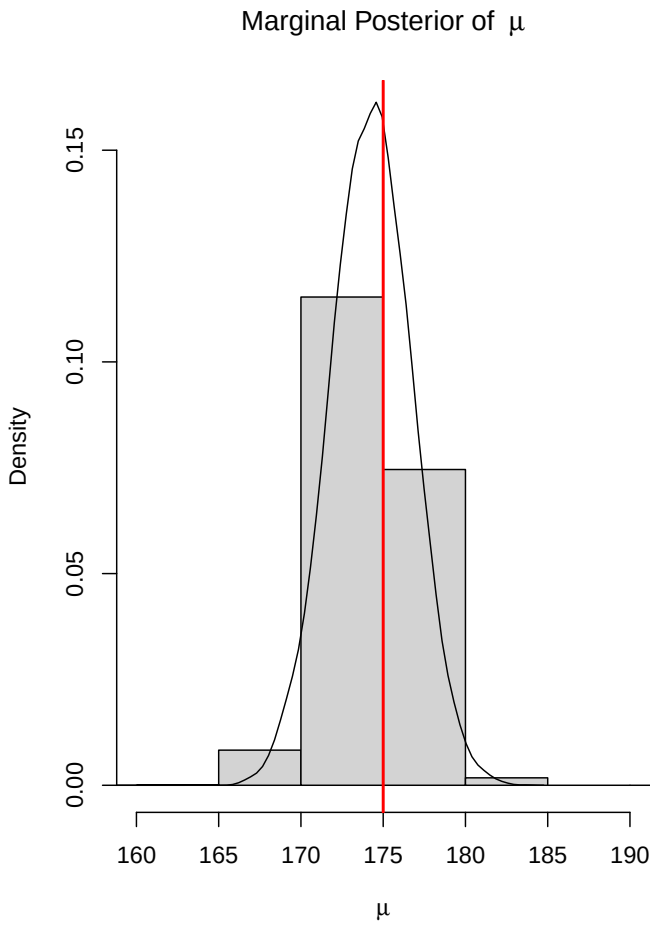
```
## [1] 174.2409
```

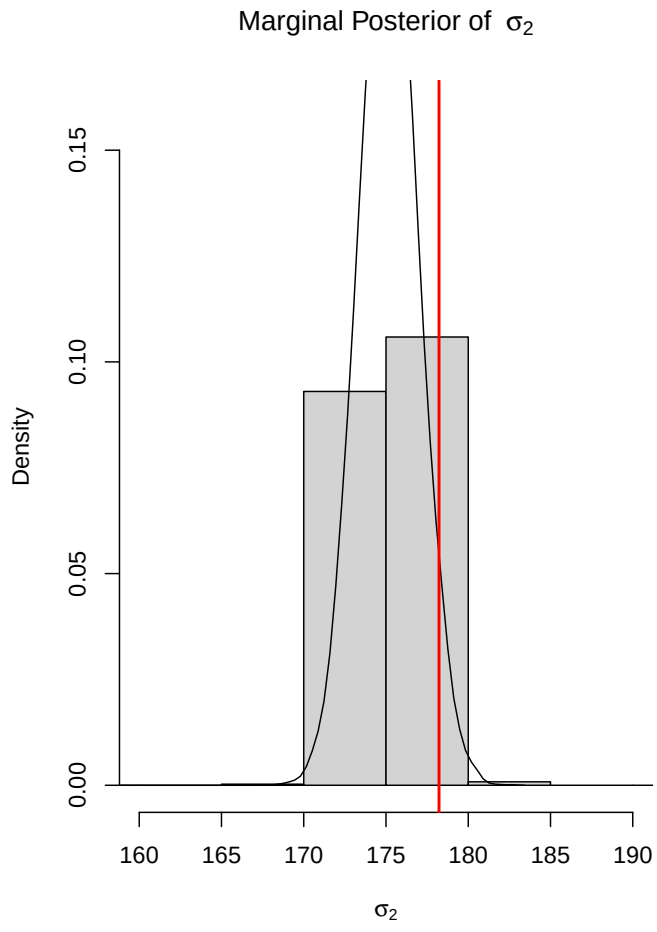
```
##     2.5%    97.5%
## 169.4024 179.0013
```

```
## [1] 30681.14
```

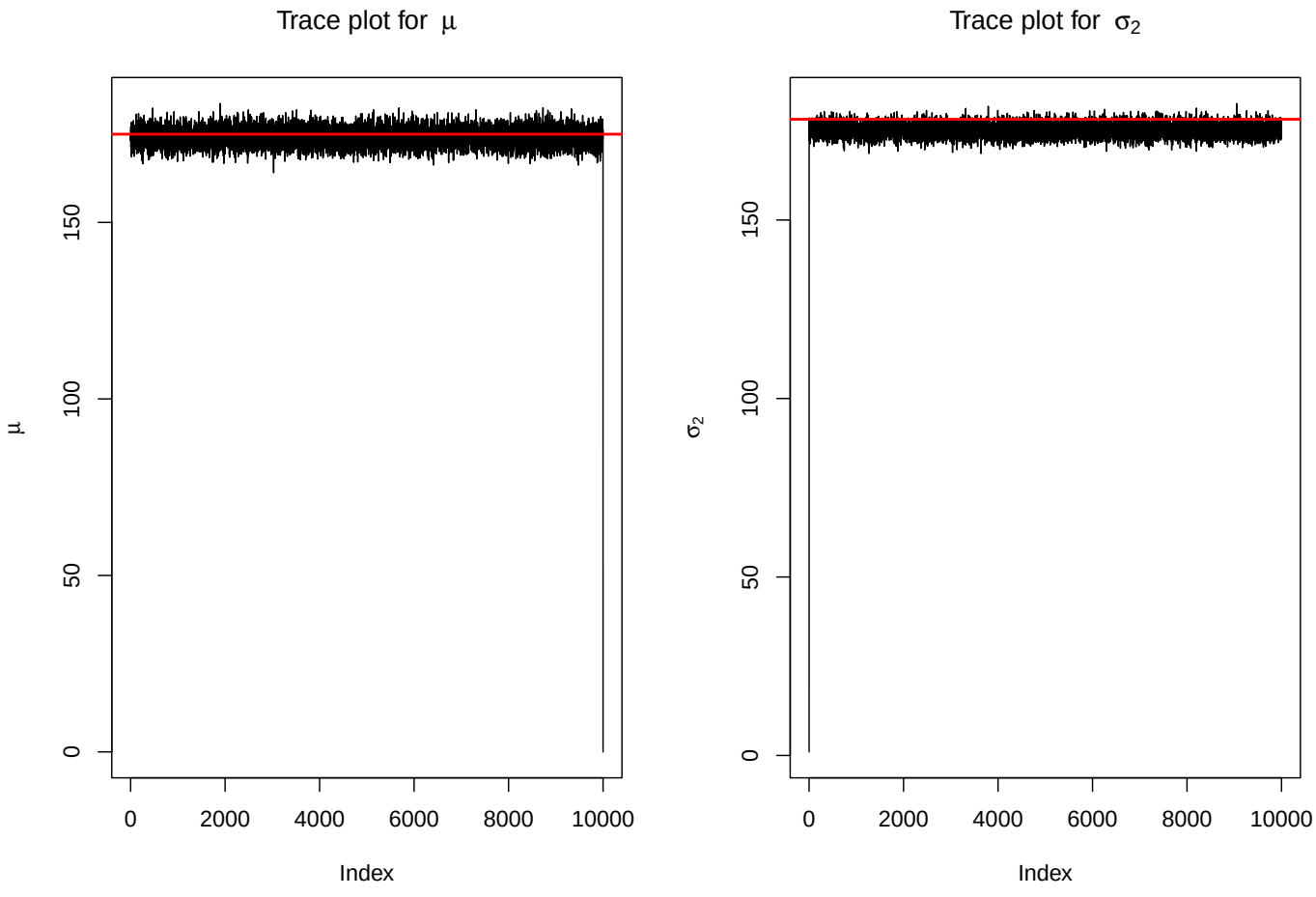
```
##     2.5%    97.5%
## 171.7129 178.6737
```

Marginal posteriors of mu and sigma





Trace Plots


Trace plot for μ
μ
0
50
100
150
0
2000
4000
6000
8000
10000
Index
Trace plot for σ2
σ2
0
50
100
150
0
2000
4000
6000
8000
10000
Index

Metropolis-Hastings Sampler

```
# Run Metropolis Algorithm
num.mcmc <- 15000
stepsize.mu <- 2
stepsize.sigmasq <- .75
acceptratio.mu <- acceptratio.sigmasq <- rep(0,num.mcmc)
mu.samplesmh <- rep(170,num.mcmc)
sigmasq.samplesmh <- sigmasq.star <- rep(3, num.mcmc)
```

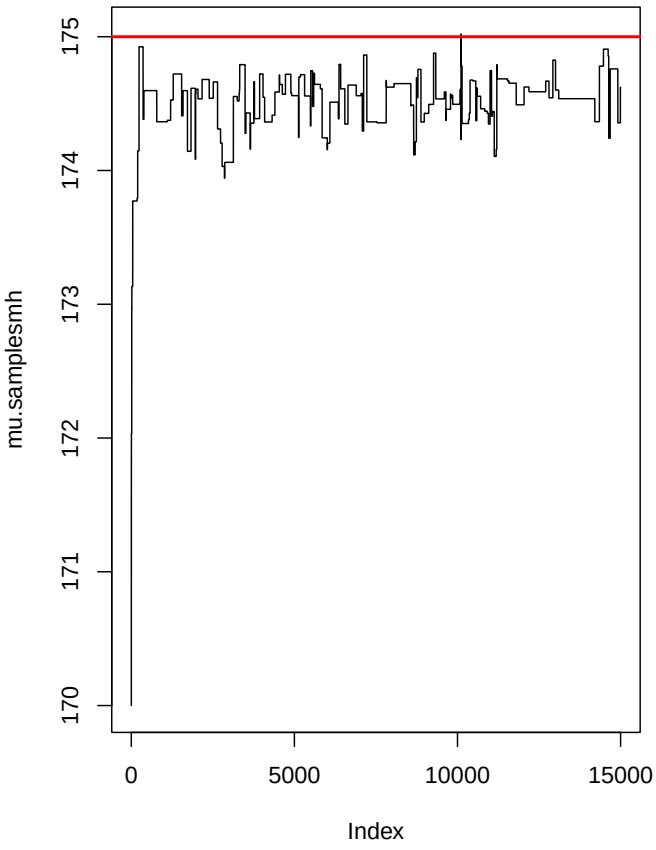
MH Algorithm

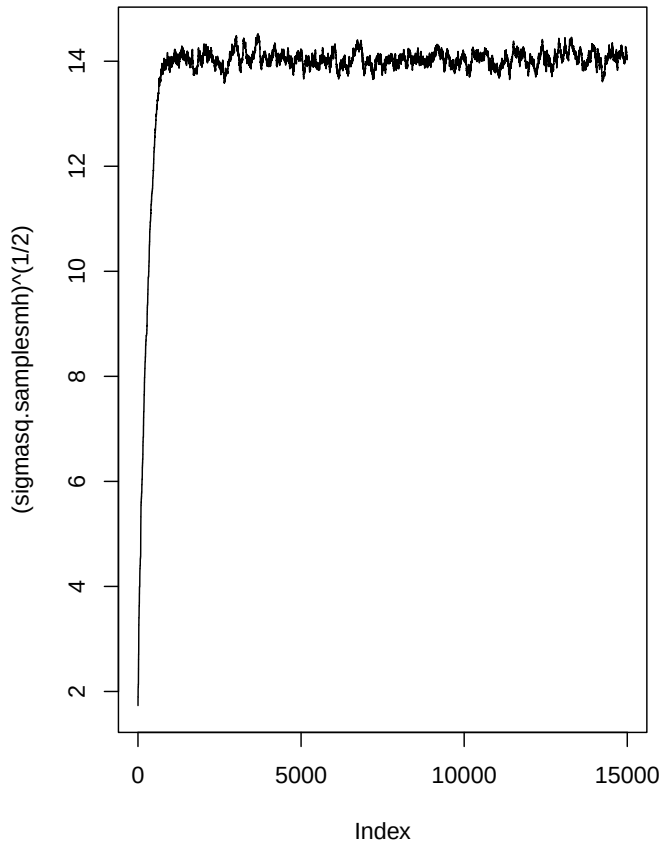
```
for (iter in 2:num.mcmc){
  # mu
  mu.star <- mu.samplesmh[iter] + rnorm(1,0,stepsize.mu)
  log.p.current <- sum(dnorm(y, mean = mu.samplesmh[iter - 1],
                             sd = sqrt(sigmasq.samplesmh[iter - 1]), log=T)) +
  dnorm(mu.samplesmh[iter - 1],mean = mu.0, sd = sqrt(tausq.0), log=T)
  log.p.star <- sum(dnorm(y, mean = mu.star, sd = sqrt(sigmasq.samplesmh[iter - 1]), log=T)) +
    dnorm(mu.star, mean = mu.0, sd = sqrt(tausq.0), log=T)

  log.r <- log.p.star - log.p.current

  if (log(runif(1)) < log.r){
    mu.samplesmh[iter] <- mu.star
    acceptratio.mu[iter] <- 1
  } else{
    mu.samplesmh[iter] <- mu.samplesmh[iter - 1]
  }
  # sigma
  sigmasq.star[iter] <- sigmasq.samplesmh[iter-1] + rnorm(1,0,stepsize.sigmasq)
  log.p.current <- sum(dnorm(y, mean = mu.samplesmh[iter],
                             sd = sqrt(sigmasq.samplesmh[iter - 1]), log=T)) +
     dinvgamma(sigmasq.samplesmh[iter - 1], rate = rate.param, shape = shape.param, log=T)
  log.p.star <- sum(dnorm(y, mean = mu.samplesmh[iter], sd = sqrt(sigmasq.star[iter]), log=T)) +
    dinvgamma(sigmasq.star[iter], rate = rate.param, shape = shape.param, log=T)

  log.r <- log.p.star - log.p.current
  if (log(runif(1)) < log.r){
    sigmasq.samplesmh[iter] <- sigmasq.star[iter]
    acceptratio.sigmasq[iter] <- 1
  } else{
    sigmasq.samplesmh[iter] <- sigmasq.samplesmh[iter - 1]
  }
}
```

Acceptance rates and Convergence

```
## [1] "Mean Acceptance rate of mu: 0.0096"

## [1] "Mean Acceptance rate of sigmasq: 0.923866666666667"

## [1] "Posterior Mean of mu: 174.52"

## [1] "Posterior 95% HPDI of mu: 174.06, 174.86"

## [1] "Posterior Mean of sigmasq: 13.83"

## [1] "Posterior 95% HPDI of sigmasq: 10.89, 14.33"
```

Trace Plots of mu and sigmasq





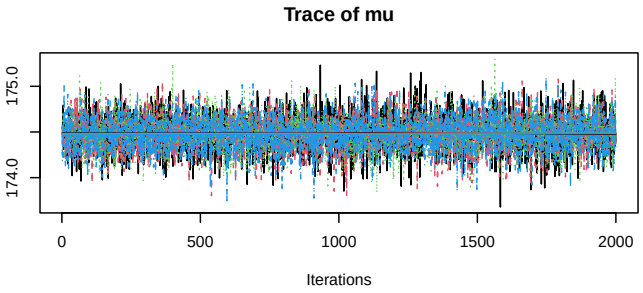
JAGS

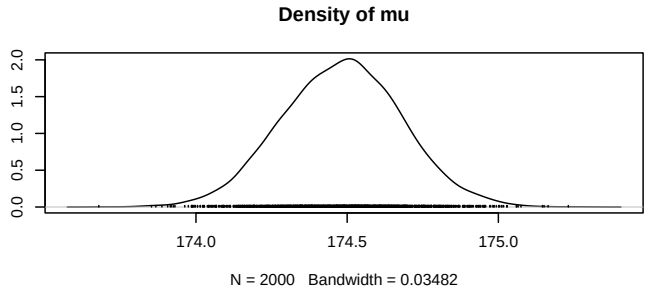
```
library(rjags)
# Define the model:
modelString = "model {
    for (i in 1:N) {
        y[i] ~ dnorm(mu, tau.sq)
    }
    mu ~ dnorm(0, 1/ 100)
    tau.sq  ~ dgamma(.005, .005)
  sigmasq <- 1 / tau.sq
}"
writeLines( modelString , con="TEMPmodel.txt" )
jags <- jags.model('TEMPmodel.txt',
                   data = list('y' = y,
                               'N' = N),
                   n.chains = 4,
                   n.adapt = 100)
 codaSamples = coda.samples( jags , n.iter=1000,
                variable.names=c("mu","tau.sq", 'sigmasq'))
summary(codaSamples)
```

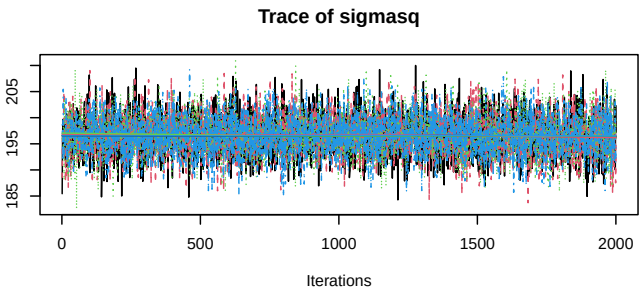
JAGS

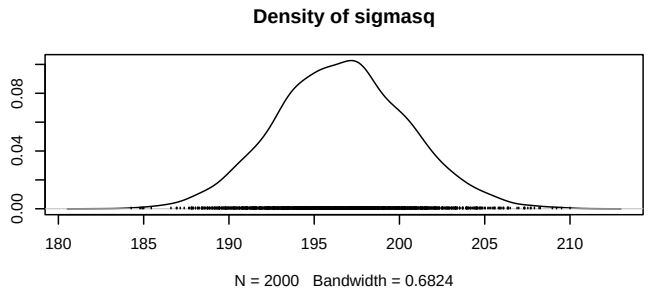
```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 5000
##    Unobserved stochastic nodes: 2
##    Total graph size: 5009
##
## Initializing model

##
## Iterations = 1:2000
## Thinning interval = 1
## Number of chains = 4
## Sample size per chain = 2000
##
## 1. Empirical mean and standard deviation for each variable,
##    plus standard error of the mean:
##
##              Mean        SD  Naive SE Time-series SE
## mu      1.745e+02 0.1982284 2.216e-03      2.283e-03
## sigmasq 1.966e+02 3.8844792 4.343e-02      4.363e-02
## tau.sq  5.088e-03 0.0001005 1.123e-06      1.129e-06
##
## 2. Quantiles for each variable:
##
##              2.5%       25%       50%       75%     97.5%
## mu      1.741e+02 174.34839 1.745e+02 1.746e+02 1.749e+02
## sigmasq 1.891e+02 193.96670 1.966e+02 1.992e+02 2.045e+02
## tau.sq  4.891e-03   0.00502 5.086e-03 5.156e-03 5.287e-03
```

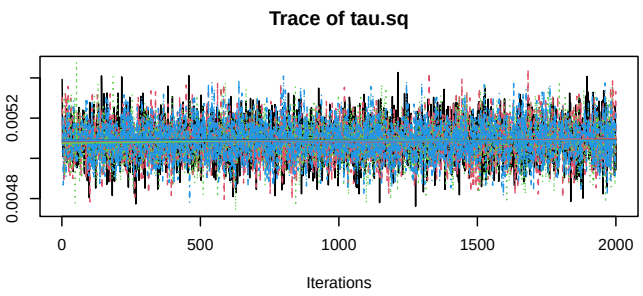
Plot

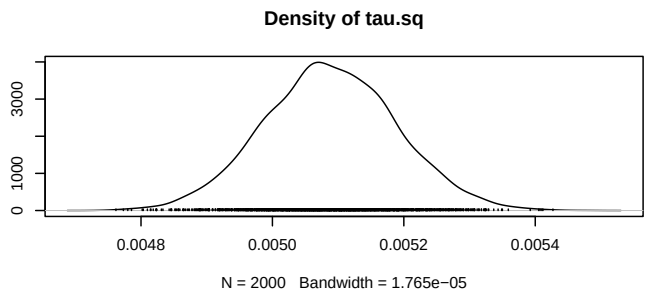

Trace of mu
175.0
174.0
0
500
1000
1500
2000
Iterations


Density of mu
2.0
1.5
1.0
0.5
0.0
174.0
174.5
175.0
N = 2000 Bandwidth = 0.03482


Trace of sigmasq
205
195
185
0
500
1000
1500
2000
Iterations


Density of sigmasq
0.08
0.04
0.00
180
185
190
195
200
205
210
N = 2000 Bandwidth = 0.6824


Trace of tau.sq
0.0052
0.0048
0
500
1000
1500
2000
Iterations


Density of tau.sq
3000
1000
0
0.0048
0.0050
0.0052
0.0054
N = 2000 Bandwidth = 1.765e−05

RStan

```
## Loading required package: StanHeaders

## Loading required package: ggplot2

## rstan (Version 2.21.8, GitRev: 2e1f913d3ca3)

## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)

##
## Attaching package: 'rstan'

## The following object is masked from 'package:coda':
##
##     traceplot

##
## SAMPLING FOR MODEL '91714379d20b8173ac6060880c3bed73' NOW (CHAIN 1).
## Chain 1:
## Chain 1: Gradient evaluation took 6.6e-05 seconds
## Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.66 seconds.
## Chain 1: Adjust your expectations accordingly!
## Chain 1:
## Chain 1:
## Chain 1: Iteration:    1 / 2000 [  0%]  (Warmup)
## Chain 1: Iteration:  200 / 2000 [ 10%]  (Warmup)
## Chain 1: Iteration:  400 / 2000 [ 20%]  (Warmup)
## Chain 1: Iteration:  600 / 2000 [ 30%]  (Warmup)
## Chain 1: Iteration:  800 / 2000 [ 40%]  (Warmup)
## Chain 1: Iteration: 1000 / 2000 [ 50%]  (Warmup)
## Chain 1: Iteration: 1001 / 2000 [ 50%]  (Sampling)
## Chain 1: Iteration: 1200 / 2000 [ 60%]  (Sampling)
## Chain 1: Iteration: 1400 / 2000 [ 70%]  (Sampling)
## Chain 1: Iteration: 1600 / 2000 [ 80%]  (Sampling)
```

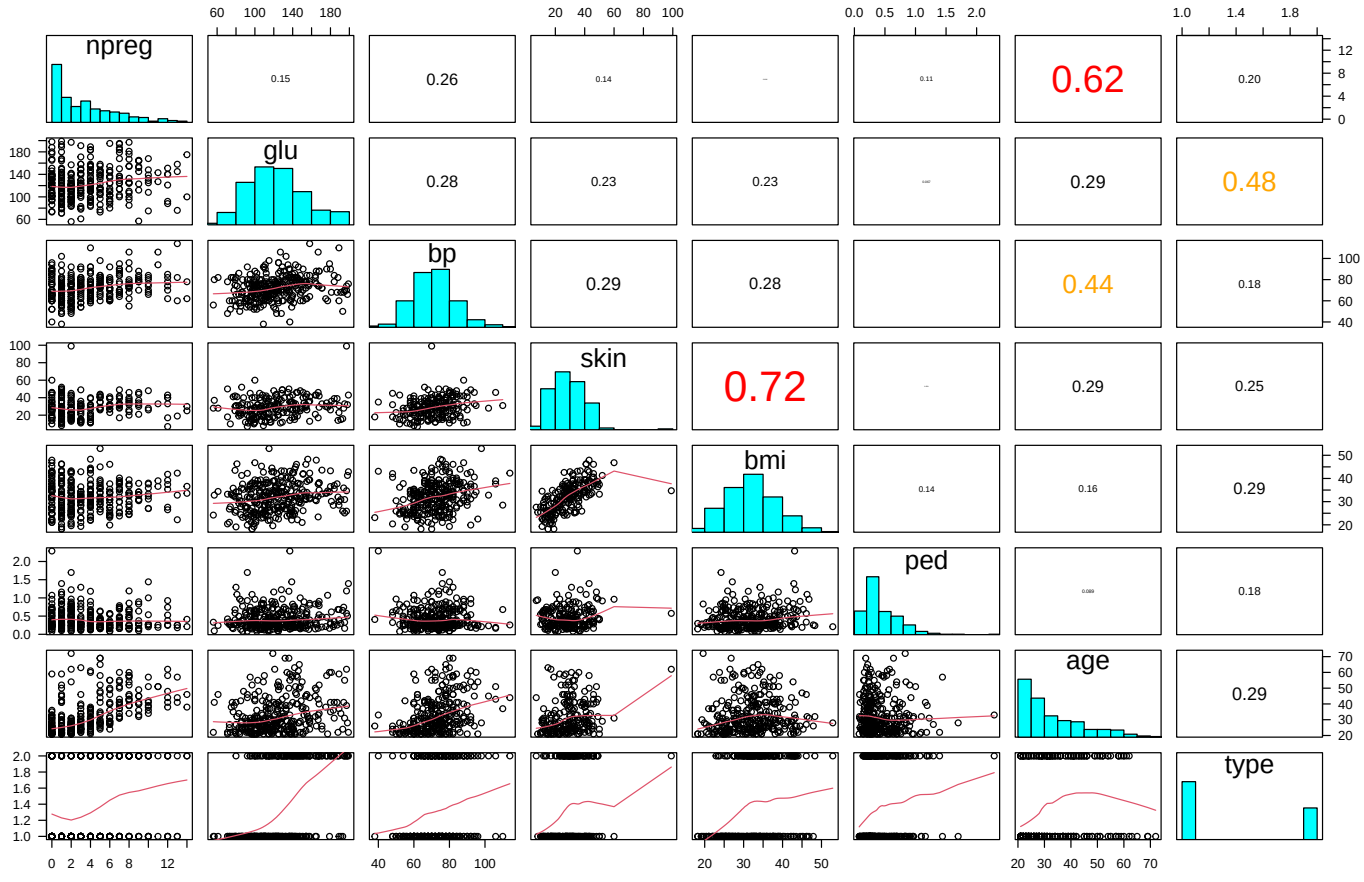
Practical Example for Logistic Regression with Pima Indian Data

Data: Pima.tr2 from MASS

```
##      npreg            glu              bp              skin       
##  Min.   : 0.000   Min.   : 56.0   Min.   : 38.00   Min.   : 7.00  
##  1st Qu.: 1.000   1st Qu.:101.0   1st Qu.: 64.00   1st Qu.:21.00  
##  Median : 3.000   Median :121.0   Median : 72.00   Median :29.00  
##  Mean   : 3.787   Mean   :123.7   Mean   : 72.32   Mean   :29.15  
##  3rd Qu.: 6.000   3rd Qu.:142.0   3rd Qu.: 80.00   3rd Qu.:36.00  
##  Max.   :14.000   Max.   :199.0   Max.   :114.00   Max.   :99.00  
##                                   NA's   :13       NA's   :98     
##       bmi              ped              age         type    
##  Min.   :18.20   Min.   :0.0780   Min.   :21.0   No :194  
##  1st Qu.:27.10   1st Qu.:0.2367   1st Qu.:24.0   Yes:106  
##  Median :32.00   Median :0.3360   Median :29.0            
##  Mean   :32.05   Mean   :0.4357   Mean   :33.1            
##  3rd Qu.:36.50   3rd Qu.:0.5867   3rd Qu.:40.0            
##  Max.   :52.90   Max.   :2.2880   Max.   :72.0            
##  NA's   :3
```

Base Line Reference Logistic Regression of Pima Indian Diabetes Data (Kaggle)

Pima Indian Data



Reference Analysis

```
##
## Call:
## glm(formula = type ~ . - npreg - skin, family = "binomial", data = Pima.tr2)
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -9.762937   1.689986  -5.777 7.61e-09 ***
## glu          0.031584   0.006752   4.677 2.90e-06 ***
## bp          -0.005174   0.018245  -0.284  0.77672
## bmi          0.078722   0.032814   2.399  0.01644 *
## ped          1.729202   0.660093   2.620  0.00880 **
## age          0.060535   0.018901   3.203  0.00136 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 256.41  on 199  degrees of freedom
## Residual deviance: 181.00  on 194  degrees of freedom
##   (100 observations deleted due to missingness)
## AIC: 193
##
## Number of Fisher Scoring iterations: 5

##
## PredictTrain  No Yes
##            0 161  45
##            1  26  52
```

RStan

Stan implements Hamiltonian Monte Carlo Simulation

- approximate Hamiltonian dynamics simulation based on numerical integration.
- instance of the Metropolis–Hastings algorithm, with a Hamiltonian dynamics evolution simulated using a time-reversible and volume-preserving numerical integrator (typically the leapfrog integrator) to propose a move to a new point in the state space.

R packages related to Stan

- rstan: R Interface to Stan
- blmeco: Data Files and Functions Accompanying the Book "Bayesian Data Analysis in Ecology using R, BUGS and Stan"
- breathteststan: Stan-Based Fit to Gastric Emptying Curves
- brms: Bayesian Regression Models using Stan
- edstan: Stan Models for Item Response Theory
- idealstan: Bayesian IRT Ideal Point Models with Stan
- rstanarm: Bayesian Applied Regression Modeling via Stan
- rstansim: Simulation Studies with Stan
- rstantools: Tools for Developing R Packages Interfacing with 'Stan'
- tmbstan: MCMC Sampling from 'TMB' Model Object using 'Stan'

Pima Indians with rstanarm

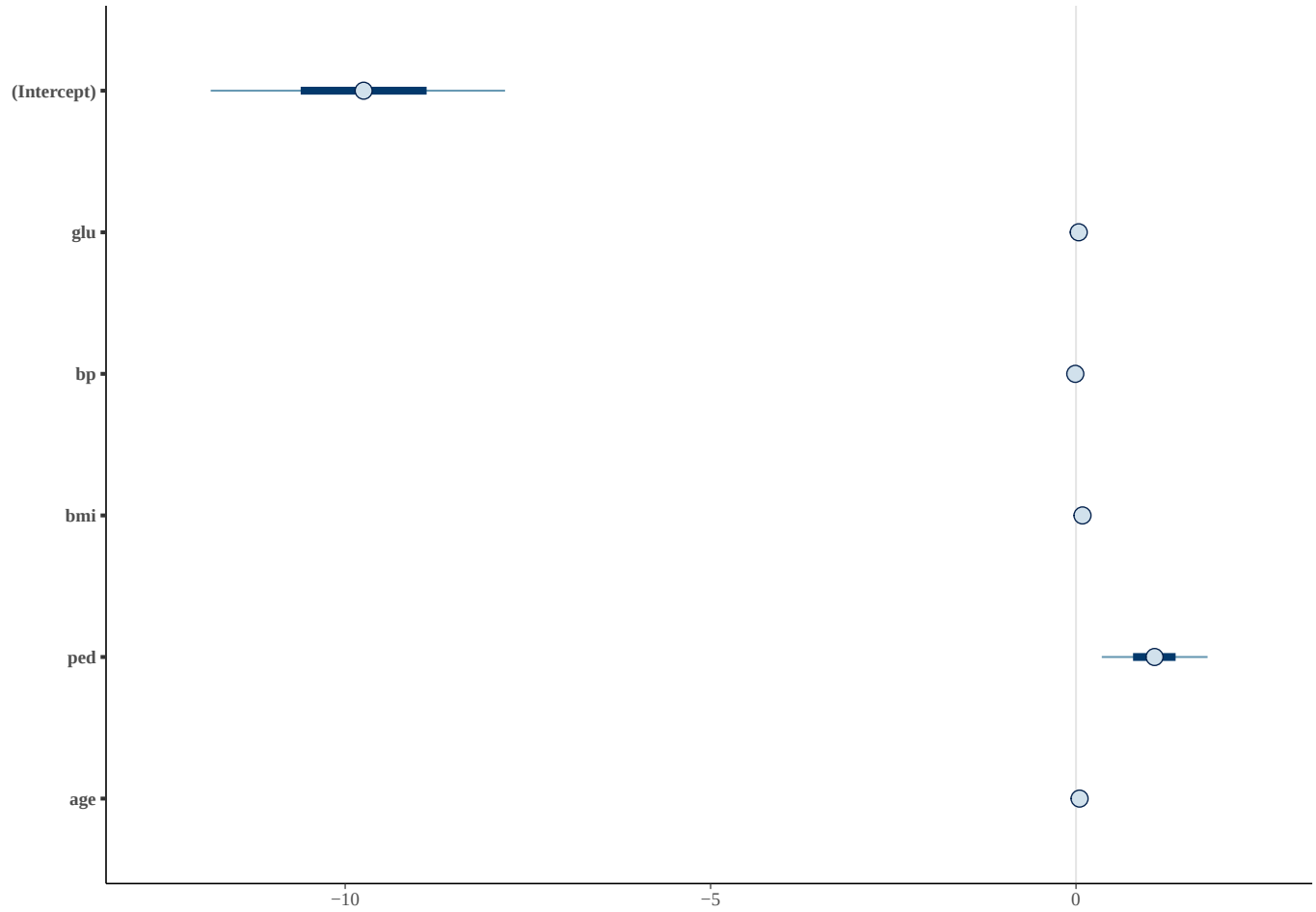
```
library(rstanarm)
fit <- stan_glm(type ~  glu + bp  + bmi + ped + age,
                data = MASS::Pima.te,
                family = binomial(),
                prior_intercept = normal(0, 10),
                prior = normal(0, 2.5),
                prior_aux = cauchy(0, 2.5),
                chains = 4,
                iter = 2000,
                seed = 12345)
print (fit)
```

Base Line Reference Logistic Regression of Pima Indian Diabetes Data with rstanarm (Kaggle)

Pima Indians with rstanarm

```
## Inference for Stan model: bernoulli.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##                  mean se_mean   sd    2.5%     25%     50%     75%   97.5%
## (Intercept)     -9.78    0.02 1.25  -12.31  -10.61   -9.75   -8.89   -7.47
## glu              0.04    0.00 0.01    0.03    0.03    0.04    0.04    0.05
## bp              -0.01    0.00 0.01   -0.04   -0.02   -0.01    0.00    0.01
## bmi              0.09    0.00 0.02    0.04    0.07    0.09    0.10    0.14
## ped              1.07    0.01 0.44    0.21    0.78    1.07    1.36    1.94
## age              0.05    0.00 0.01    0.02    0.04    0.05    0.06    0.08
## mean_PPD         0.33    0.00 0.03    0.27    0.31    0.33    0.35    0.39
## log-posterior -157.20    0.04 1.80 -161.59 -158.17 -156.86 -155.88 -154.74
##                n_eff Rhat
## (Intercept)     4747    1
## glu             6415    1
## bp              3567    1
## bmi             3266    1
## ped             3630    1
## age             3692    1
## mean_PPD        4025    1
## log-posterior   1976    1
##
## Samples were drawn using NUTS(diag_e) at Tue Jul  4 18:20:25 2023.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

Plot of Posteriors



Integrated nested Laplace approximation (INLA)

INLA is a method for approximate Bayesian inference

- Meant for latent Gaussian models very useful for models of right type with many hyperparmeters or complex hierarchical structures
- deterministic algorithm that uses numerical integration to approximate the posterior distribution
- much faster than MCMC, particularly for complex models

INLA

```r
library(INLA)
library(data.table)

library(MASS)
# Load the Pima Indian diabetes dataset from the mlbench package
data(Pima.tr2, package = "MASS")

# Split the data into training and testing sets using a 70/30 split
set.seed(1234)
train <- sample(nrow(Pima.tr2), 0.7 * nrow(Pima.tr2))
test <- setdiff(1:nrow(Pima.tr2), train)

y <- as.matrix(as.numeric(Pima.tr2[train, 8])-1)
x <- as.matrix(Pima.tr2[train, -c(1,4,8)])

formula <- y ~ x

model <- inla(formula, family = "binomial", data = list(y = y, x = x), control.compute = list(dic = TRUE))

summary(model)
```

INLA

```
##                mean    sd 0.025quant 0.5quant 0.975quant    mode kld
## (Intercept) -10.101 1.473    -12.987  -10.101     -7.215 -10.101   0
## glu           0.043 0.008      0.028    0.043      0.058   0.043   0
## bp           -0.023 0.011     -0.045   -0.023      0.000  -0.023   0
## bmi           0.104 0.029      0.047    0.104      0.161   0.104   0
## ped           1.532 0.693      0.175    1.532      2.889   1.532   0
## age           0.044 0.017      0.010    0.044      0.077   0.044   0
```

MCMCLogit

Bayesian Inference Logistic Regression of Pima Indian Diabetes Data with MCMCLogit

```
# Empirical Bayes - determine prior from data or pre-step analysis
Pima.tr2$diab<-(as.numeric(Pima.tr2$type)-1)
prior.data <- Pima.tr2[sample(1:dim(Pima.tr2)[1], 40), ]
prior.data.posterior <- MCMCpack::MCMClogit(diab~.-type-skin-npreg, dat
prior.mean.tr <- apply(prior.data.posterior, 2, mean)
prior.mean.tr
```

```
##   (Intercept)           glu            bp           bmi            pe
## -60.621591063   0.190915667   0.219588329   0.359158767  14.57082868
##           age
##   0.004550912
```

```
prior.cov.tr <- cov(prior.data.posterior)
prior.cov.tr
```

```
##               (Intercept)           glu            bp           bmi
## (Intercept)  557.5064113 -1.6999052351 -1.832360314 -4.103133435 -11
## glu           -1.6999052  0.0061655173  0.004522891  0.010172149
## bp            -1.8323603  0.0045228914  0.012391060  0.009300064
## bmi           -4.1031334  0.0101721495  0.009300064  0.052710588
```

Comparison

	(Intercept)	glu	bp	bmi	ped	age
glm(logit)	-9.76	0.03	-0.01	0.08	1.73	0.06
RStan	-9.78	0.04	-0.01	0.09	1.08	0.05
INLA	-10.10	0.04	-0.02	0.10	1.53	0.04
MCMClogit	-10.68	0.03	-0.00	0.09	1.84	0.06

RJags/WinBUGS

JAGS/BUGS is designed specifically with Markov Chain Monte Carlo (MCMC) methods in mind

This requires

- the specification of the likelihood and related models
- the specification of conjugate priors including hyperparameters,
- definition of sampling parameters, such as simulation length (=chain length) and burn-in length and thinning of chains to reduce the Markov Chain induced dependence of samples.

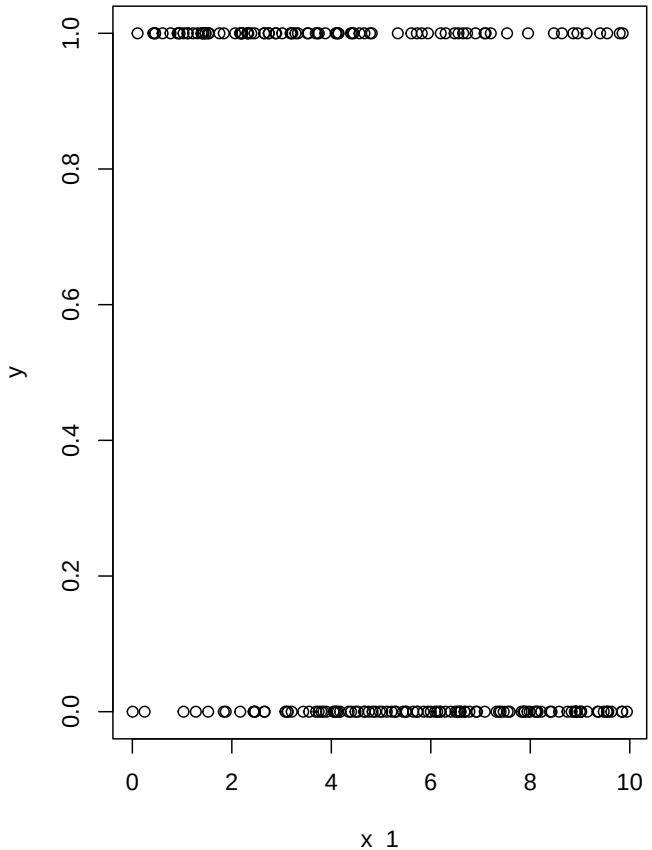
R packages related to JAGS

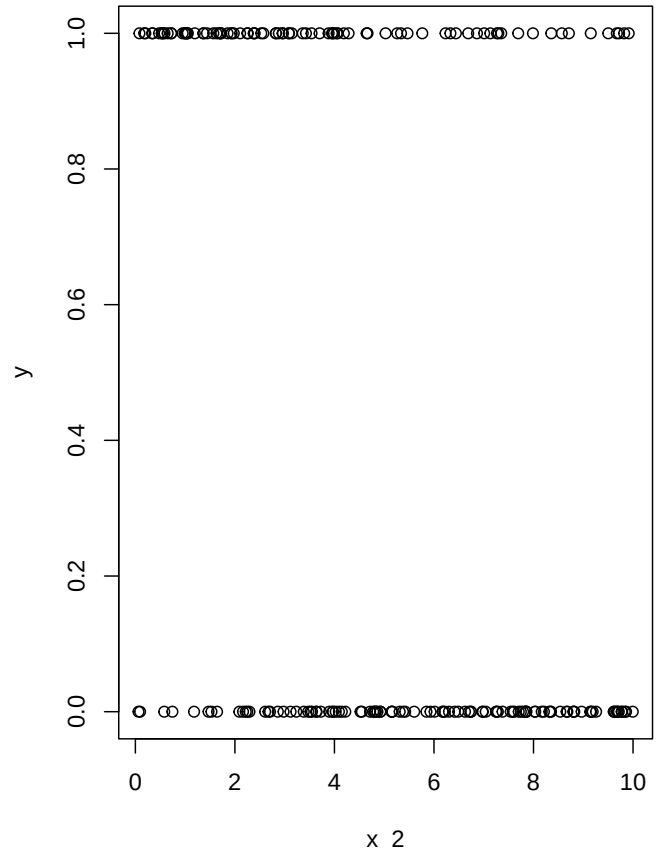
- rjags: R Interface to the JAGS MCMC library
- jagsUI: A Wrapper Around rjags to Streamline JAGS Analyses
- R2Jags: Providing wrapper functions to implement Bayesian analysis in JAGS.
- bayesmix: finite mixture models of univariate Gaussian distributions using JAGS
- dalmatian: Automates fitting of double GLM in JAGS.
- glmmBUGS: Generalized Linear Mixed Models with BUGS and JAGS
- HydeNet: Hybrid Bayesian Networks Using R and JAGS

Fitting a logistic regression in JAGS

```
# Description of the Bayesian model fitted in this file
# Notation:
# y_t = binomial (often binary) response variable for observation t=1,.
# x_{1t} = first explanatory variable for observation t
# x_{2t} = second " " " " " " " " "
# p_t = probability of y_t being 1 for observation t
# alpha = intercept term
# beta_1 = parameter value for explanatory variable 1
# beta_2 = parameter value for explanatory variable 2

# Likelihood
# y_t ~ Binomial(K,p_t), or Binomial(1,p_t) if binary
# logit(p_t) = alpha + beta_1 * x_1[t] + beta_2 * x_2[t]
# where logit(p_i) = log( p_t / (1 - p_t ))
# Note that p_t has to be between 0 and 1, but logit(p_t) has no limits

# Priors - all vague
# alpha ~ normal(0,100)
# beta_1 ~ normal(0,100)
# beta_2 ~ normal(0,100)
```

Data Simulation





Fitting Jags Model

```
## library(R2jags)
## library(boot) # Package contains the logit transform
##
## # Jags code to fit the model to the simulated data
## model_code <- "
## model
## {
##   # Likelihood
##   for (t in 1:T) {
##     y[t] ~ dbin(p[t], K)
##     logit(p[t]) <- alpha + beta_1 * x_1[t] + beta_2 * x_2[t]
##   }
##   # Priors
##   alpha ~ dnorm(0.0,0.01)
##   beta_1 ~ dnorm(0.0,0.01)
##   beta_2 ~ dnorm(0.0,0.01)
## }
## "
##
## # Set up the data
## model_data <- list(T = T, y = y, x_1 = x_1, x_2 = x_2, K = 1)
##
## # Choose the parameters to watch
## model_parameters <- c("alpha", "beta_1", "beta_2")
##
## # Run the model
## model_run <- jags(
##   data = model_data,
##   parameters.to.save = model_parameters,
##   model.file = textConnection(model_code),
##   n.chains = 4,
##   n.iter = 1000,
##   n.burnin = 200,
##   n.thin = 2
## )
```

Simulated results

```
## # Check the output - are the true values inside the 95% CI?
## # Also look at the R-hat values - they need to be close to 1 if convergence has been achieved
## print(model_run)
## par(mfrow=c(1,2))
## plot(model_run)
## traceplot(model_run)
##
## # Create a plot of the posterior mean regression line
## post <- print(model_run)
## alpha_mean <- post$mean$alpha
## beta_1_mean <- post$mean$beta_1
## beta_2_mean <- post$mean$beta_2
##
## # As we have two explanatory variables I'm going to create two plots
## # holding one of the variables fixed whilst varying the other
## par(mfrow = c(2, 1))
## plot(x_1, y)
## lines(x_1,
##   inv.logit(alpha_mean + beta_1_mean * x_1 + beta_2_mean * mean(x_2)),
##   col = "red"
## )
## plot(x_2, y)
## lines(x_2,
##   inv.logit(alpha_mean + beta_1_mean * mean(x_1) + beta_2_mean * x_2),
##   col = "red"
## )
##
## # Line for x_1 should be increasing with x_1, and vice versa with x_2
```

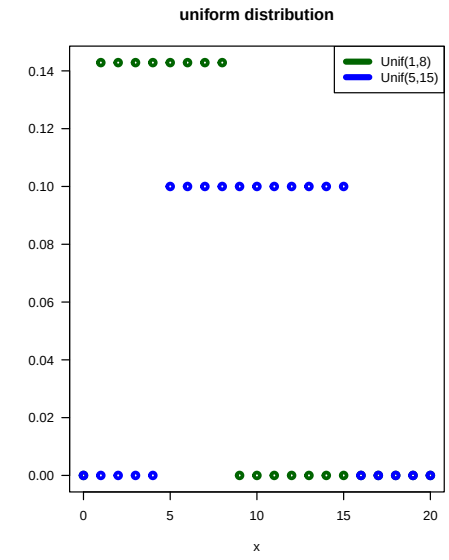
Discrete Distributions

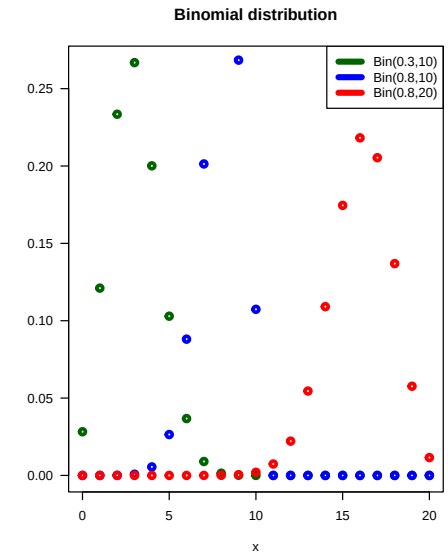
Uniform distribution: $\mathbb{P}(x|x_{min}, x_{min}) = \frac{1}{x_{max} - x_{min}}$ is the probability of the uniform distribution denoted as $U(x_{min}, x_{max})$, where $x_{min} \in \mathbb{R}$ and $x_{max} \in \mathbb{R}$.

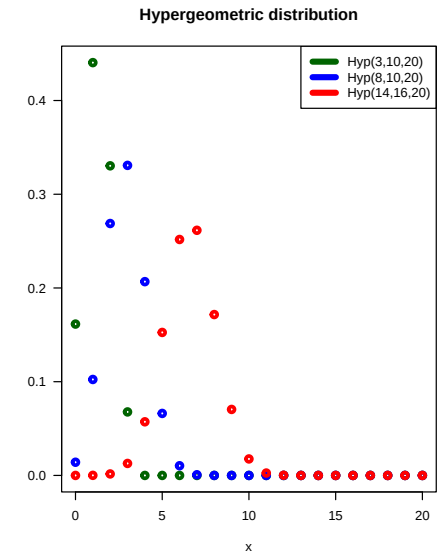
Binomial distribution: $\mathbb{P}(x|p, n) = \binom{n}{x} \cdot p^x \cdot (1-p)^{n-x}$ is the probability of the Binomial distribution denoted as $Bin(n, p)$, where $n \in \mathbb{N}$ is the total number of trials and $p \in [0, 1]$ the probability of success.

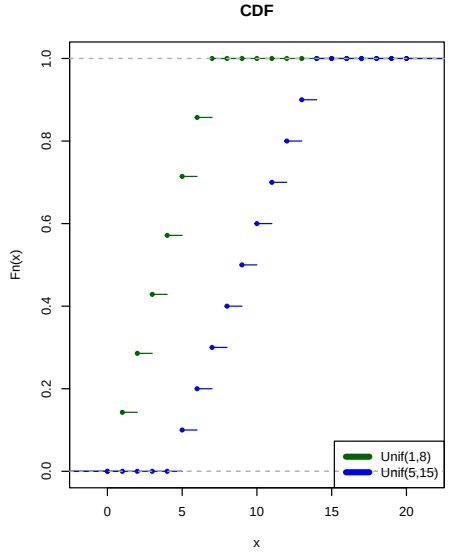
Hypergeometric distribution: $\mathbb{P}(x|N, n, I) = \frac{\binom{I}{x} \cdot \binom{N-I}{n-x}}{\binom{N}{n}}$ is the probability of the Hypergeometric distribution denoted as $Hyp(N, n, I)$, where $N \in \mathbb{N}$ is the total number of observables, $n \in \mathbb{N}$ is the numerb of observables drawn for the sample and $I \in \mathbb{N}$ the total number of interesting units.

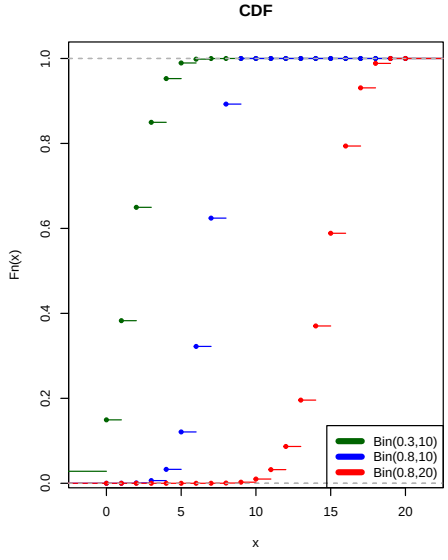
Discrete Distributions I - Visualisation

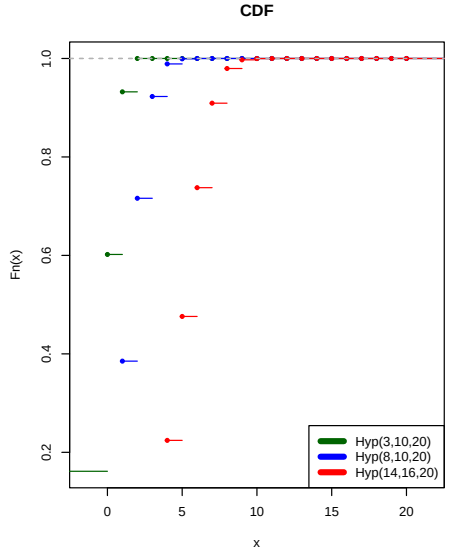












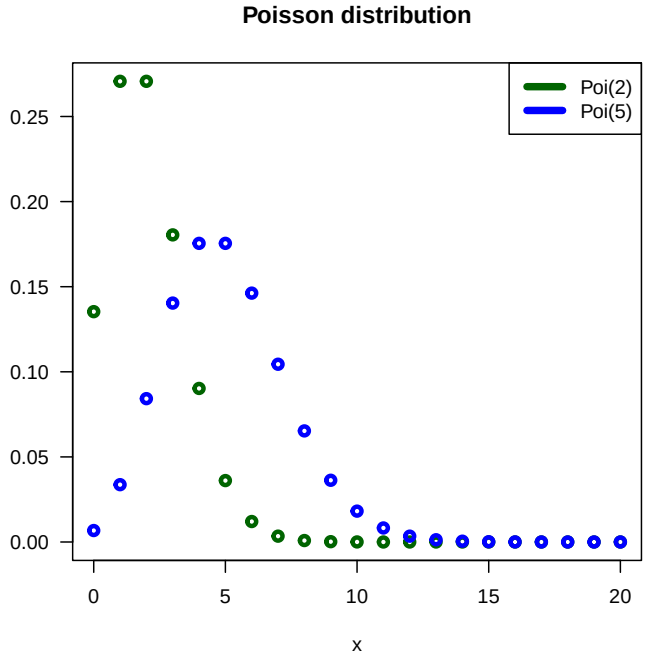
Discrete Distributions

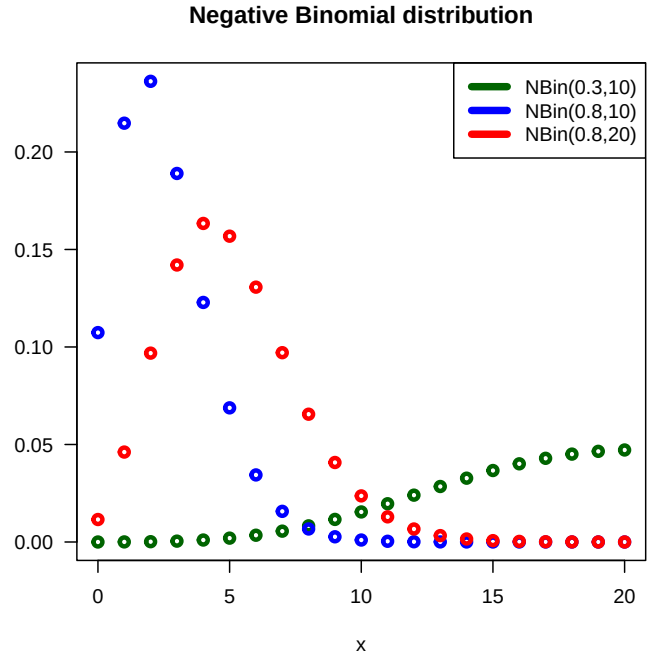
Negative Binomial distribution:

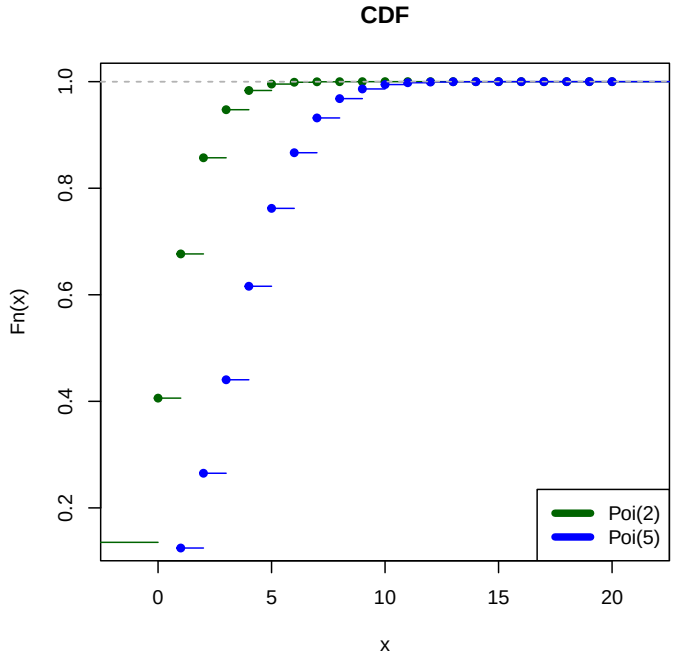
$\mathbb{P}(x|p,n) = \begin{pmatrix} n+x-1 \\ x \end{pmatrix} \cdot p^x \cdot (1-p)^n$ is the probability of the Negative Binomial distribution denoted as $NB(n,p)$, where $n \in \mathbb{N}$ is the total number of failures out of n+x trials until x successes are reached and $p \in [0,1]$ the probability of success.

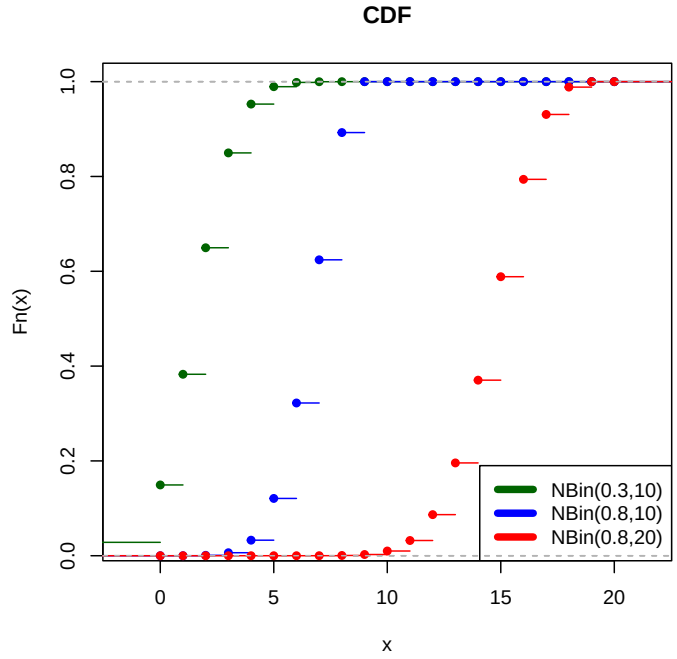
Poisson distribution: $\mathbb{P}(x|\lambda) = \frac{\lambda^x}{x!} e^{-\lambda \cdot x}$ is the probability of the Poisson distribution denoted as $Poi(\lambda)$, where $\lambda \in \mathbb{R}^+$ is the mean number of occurrences of an interesting event per reference unit.

Discrete Distributions II - Visualisation









Continuous Distributions I

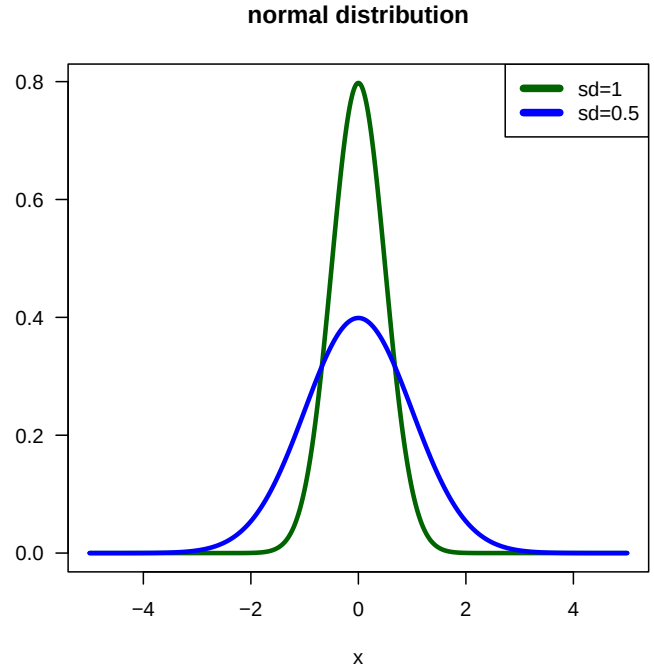
Normal distribution: $f(x|\mu,\sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2}$ is the density of the normal distribution denoted as $N(\mu,\sigma^2)$, where $\mu \in \mathbb{R}$ and $\sigma \in \mathbb{R}^+$.

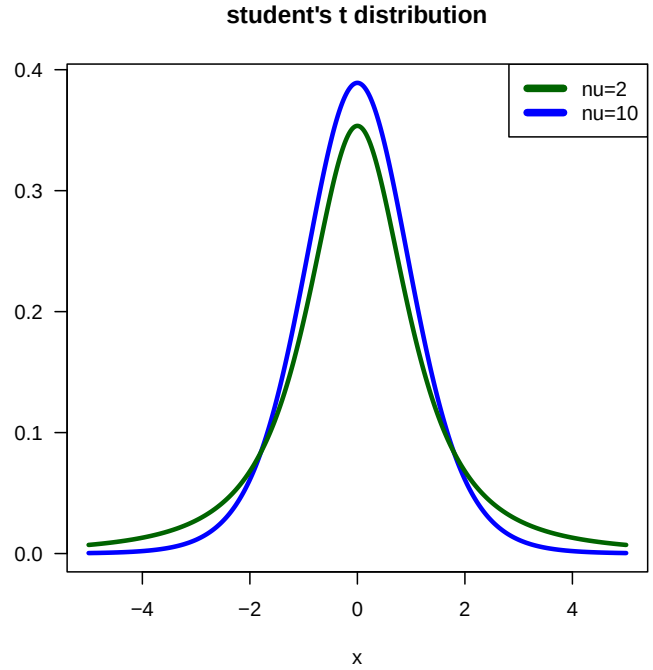
student's t-distribution: $f(x|\nu) = \frac{\Gamma((\nu+1)/2)}{\sqrt{2\nu}\Gamma(\nu/2)} \left(1+\frac{x^2}{\nu}\right)^{-\frac{\nu+1}{2}}$ is the density of the t-distribution with ν degrees of freedom, denoted as t_ν, where $\nu \in \mathbb{N}^+$. For $\nu = 1$ the distribution is called Cauchy distribution.

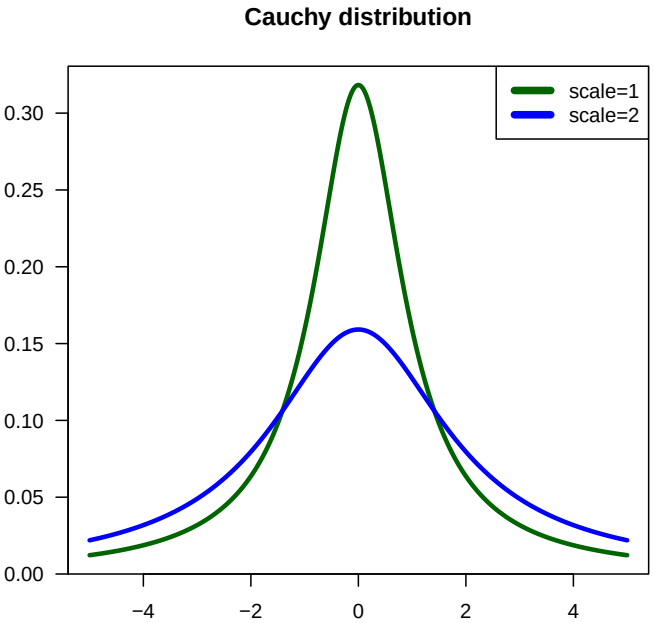
Cauchy distribution: $f(x|\gamma) = \frac{1}{\pi\gamma\left(1+\frac{x^2}{\gamma^2}\right)}$ is the density of the Cauchy-distribution with scale parameter γ.

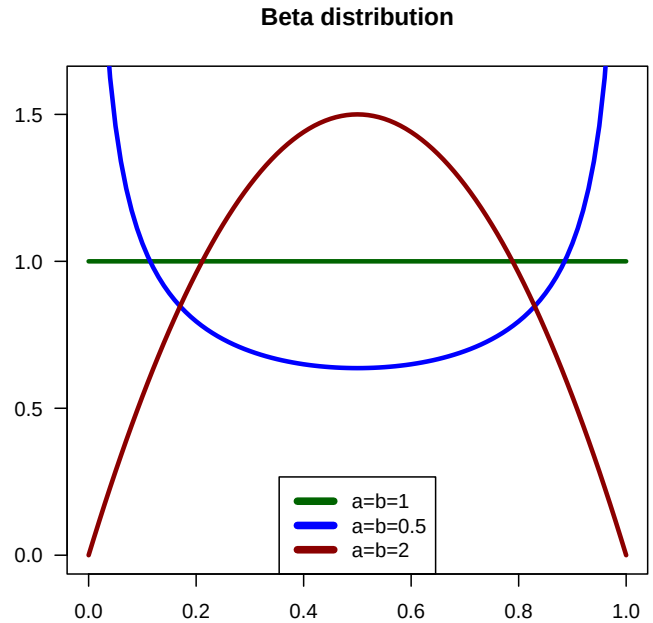
Beta distribution: $f(x|\alpha,\beta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}$ is the density of the Beta distribution denoted as $Be(\alpha,\beta)$, where $x \in [0,1]$, $\alpha \in \mathbb{R}$ and $\beta \in \mathbb{R}^+$.

Continuous Distributions I - Visualisation









Continuous Distributions II

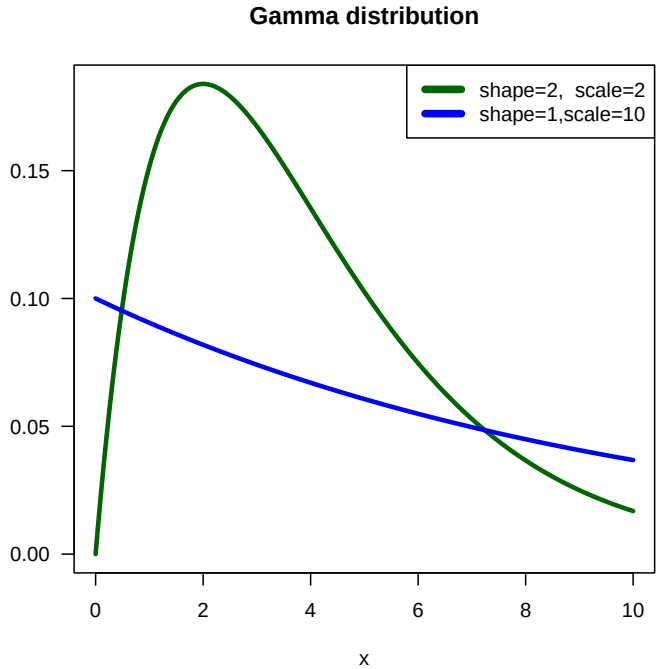
Gamma distribution: $f(x|\alpha,\beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$ is the density of the Gamma distribution denoted as $Ga(\alpha,\beta)$, where $x \geq 0$, the shape parameter $\alpha \in \mathbb{R}$ and the scale parameter $\beta \in \mathbb{R}^+$.

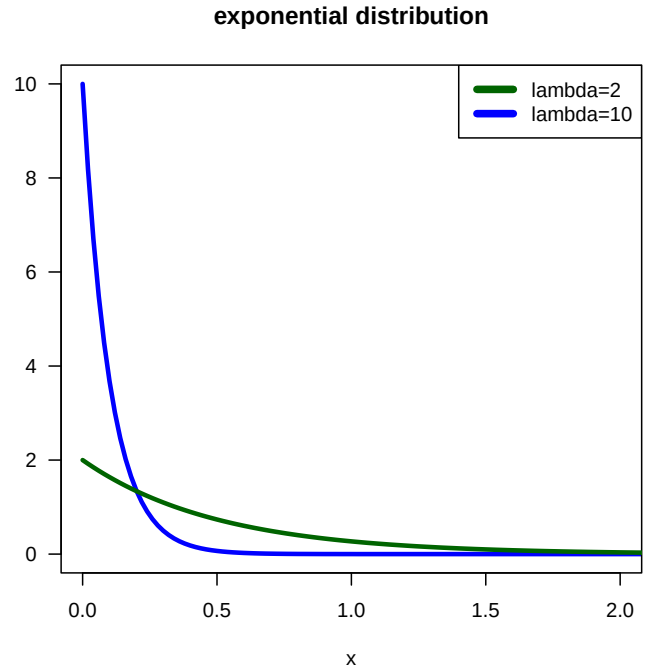
Exponential distribution: $f(x|\lambda) = \lambda e^{-\lambda x}$ is the density of the exponential distribution denoted as $Exp(\lambda)$, where $x \geq 0$ and $\lambda \in \mathbb{R}^+$.

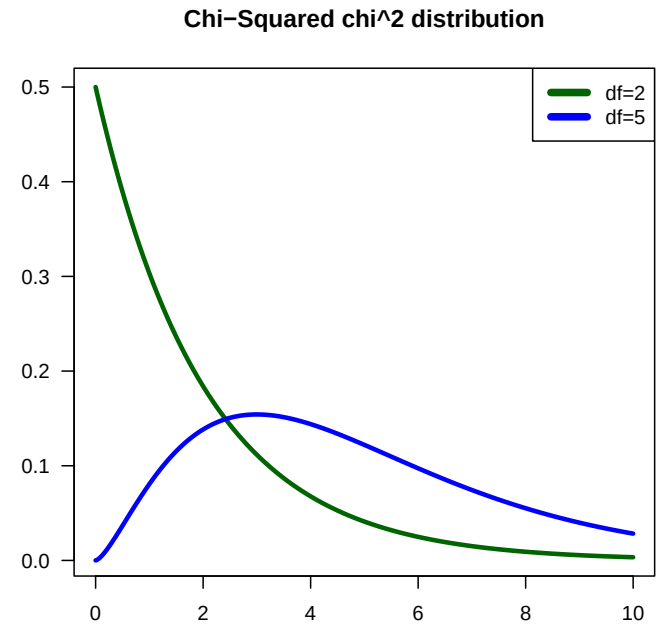
Chi-square distribution: $f(x|k) = \frac{x^{k/2-1} e^{-x/2}}{2^{k/2}\Gamma(k/2)}$ is the density of the chi-square distribution with k degrees of freedom, denoted as χ^2_k, where $x \geq 0$ and $k \in \mathbb{N}^+$.

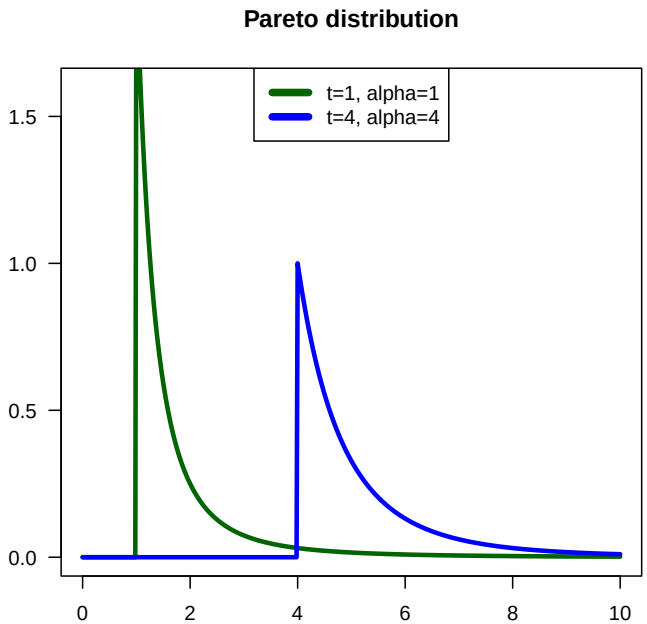
Pareto distribution: $f(x|\alpha,x_0) = \alpha x_0^\alpha x^{-(\alpha+1)}$ is the density of the Pareto (Type I) distribution with tail index α, denoted as $Pa(\alpha,x_0)$, where $x \geq x_0$ and $\alpha \in \mathbb{R}^+$.

Continuous Distributions II - Visualisation









Relations between Distributions visualised

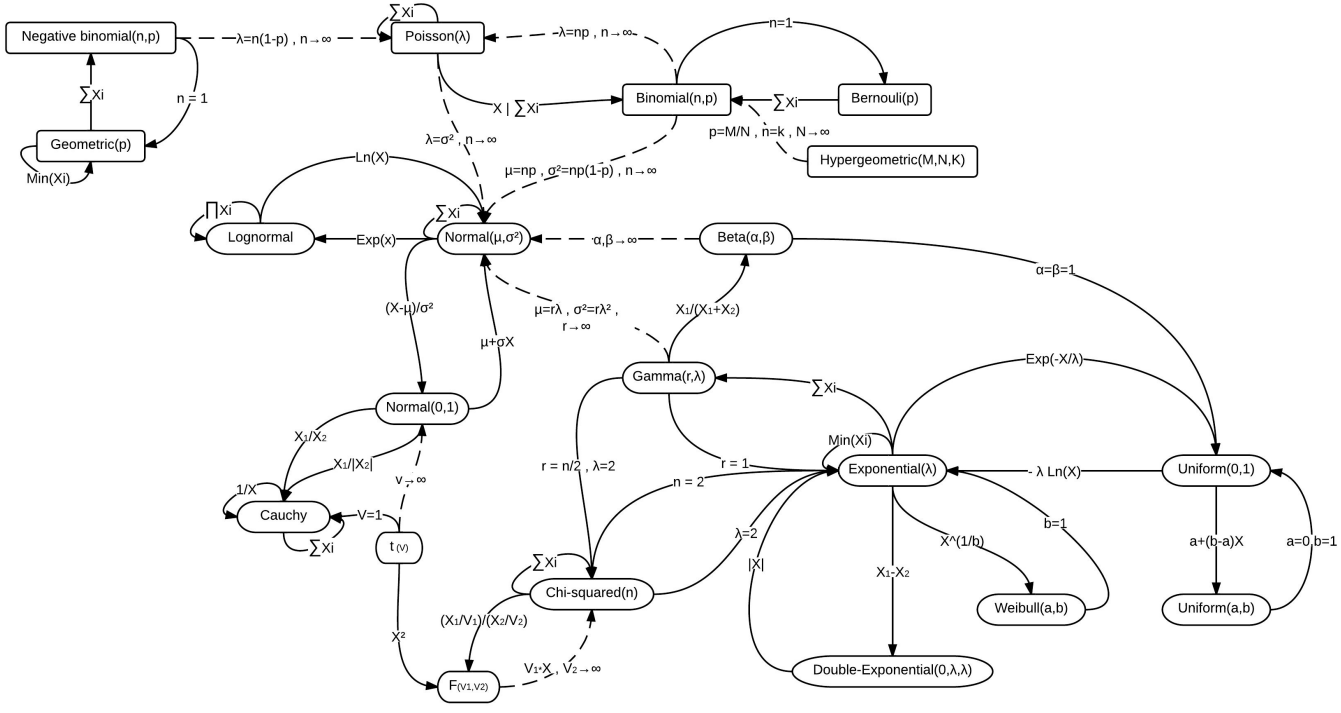


Abbildung 1: Source: www.math.wm.edu/~leemis/2008amstat.pdf