Bayesian methods
Oliver Kirchkamp

\[ \beta_j = \frac{1}{\sigma^2} = \tau \]

\[ \mu = 0 \]

\[ \mu[i] = \beta_0 + \sum_j \beta_j x[i, j] \]

\[ y[i] \]
Contents

1 Introduction 5
   1.1 Preliminaries .................................................. 5
   1.2 Motivation ..................................................... 5
   1.3 Using Bayesian inference .................................... 5
   1.4 The intention of the researcher — p-hacking ............ 9
   1.5 Compare: The Maximum Likelihood estimator ............. 11
   1.6 Terminology ................................................... 14
      1.6.1 Probabilities ........................................... 14
      1.6.2 Prior information ...................................... 15
      1.6.3 Objectivity and subjectivity ......................... 15
      1.6.4 Issues .................................................. 16
   1.7 Decision making ............................................... 16
   1.8 Technical Background ....................................... 18

2 A practical example 19
   2.1 The distribution of the population mean .................. 19
   2.2 Gibbs sampling ............................................... 22
   2.3 Convergence .................................................. 23
   2.4 Distribution of the posterior ................................ 25
   2.5 Accumulating evidence ....................................... 25
   2.6 Priors .......................................................... 26

3 Conjugate Priors 26
   3.1 Accumulating evidence, continued ......................... 26
   3.2 Normal Likelihood ........................................... 27
   3.3 Bernoulli Likelihood ....................................... 29
   3.4 Problems with the analytical approach .................... 29
   3.5 Exercises ..................................................... 30

4 Linear Regression 30
   4.1 Introduction .................................................. 30
   4.2 Demeaning .................................................... 33
   4.3 Correlation ................................................... 37
   4.4 The three steps of the Gibbs sampler ..................... 39
   4.5 Exercises ..................................................... 40

5 Finding posteriors 40
   5.1 Overview ...................................................... 40
   5.2 Example for the exact way: ................................ 40
   5.3 Rejection sampling ......................................... 41
   5.4 Metropolis-Hastings ......................................... 42
## Contents

5.5 Gibbs sampling ......................................................... 43
5.6 Check convergence ..................................................... 46
  5.6.1 Gelman, Rubin (1992): potential scale reduction factor .... 46
5.7 A better vague prior for $\tau$ ........................................... 51
5.8 More on History ....................................................... 53

6 Robust regression ......................................................... 54
  6.1 Robust regression with the Crime data ............................ 54
  6.2 Robust regression with the Engel data ......................... 59
  6.3 Exercises .......................................................... 61

7 Nonparametric ........................................................... 62
  7.1 Preliminaries ........................................................ 62
  7.2 Example: Rank sum based comparison ........................ 62

8 Identification ............................................................ 66

9 Discrete Choice .......................................................... 70
  9.1 Labor force participation ........................................... 70
  9.2 A generalised linear model ...................................... 70
  9.3 Bayesian discrete choice ......................................... 71
  9.4 Exercise ............................................................ 72

10 Count data .............................................................. 72
  10.1 Poisson model .................................................... 72
  10.2 Negative binomial .................................................. 74
  10.3 Exercise .............................................................. 76

11 Multinomial (polymous) logit ........................................... 76
  11.1 Motivation and background ...................................... 76
  11.2 Example ............................................................ 80
  11.3 Bayesian multinomial ............................................. 83
  11.4 Exercise ............................................................ 84

12 Ordered probit .......................................................... 84
  12.1 Model ............................................................... 84
  12.2 Illustration — the Fair data .................................... 86
  12.3 Exercise ............................................................ 91

13 Instrumental variables .................................................. 91
  13.1 Example: Demand and Supply ................................... 93
  13.2 Discrete endogeneous variables .................................. 98
  13.3 Exercises ............................................................ 101
14 Measurement errors
  14.1 Single measures, known error ........................................... 101
  14.2 Multiple measures ......................................................... 105
  14.3 Aggregating evidence .................................................... 106

15 Selection
  15.1 Interval regression ......................................................... 112
  15.2 Bayesian censored model ................................................. 114
  15.3 Heckman correction ........................................................ 116
  15.4 Bayesian Heckman correction ............................................ 119
  15.5 Exercise .................................................................. 120

16 More on initialisation .......................................................... 120

17 Hierarchical Models
  17.1 Mixed effects ................................................................. 121
  17.2 Example: Crime in North Carolina ...................................... 121
  17.3 Bayes and mixed effects .................................................. 123
  17.4 Robust mixed effects ....................................................... 124
  17.5 Exercises ................................................................ 127

18 Model Comparison
  18.1 Example 1 .................................................................. 128
  18.2 Example 2 .................................................................. 129
  18.3 Model 1 .................................................................... 131
  18.4 Model 2 .................................................................... 131
  18.5 A joint model ................................................................ 132
  18.6 Pseudopriors ................................................................ 136
  18.7 Model uncertainty .......................................................... 139
  18.8 Bayes factors ................................................................. 140

19 Mixture Models
  19.1 Example .................................................................. 140
  19.2 Labels and sorting ........................................................... 142
  19.3 More groups ................................................................ 143
  19.4 Ordering, not sorting ...................................................... 145
  19.5 Using dnormmix ............................................................. 146
  19.6 Exercises .................................................................. 147

20 Summary .................................................................. 147

21 Exercises .................................................................. 148
1 Introduction

1.1 Preliminaries

Purpose of this handout  In this handout you find the content of the slides I am using in the lecture. The handout is not supposed to replace a book. I recommend some books on the webpage and in the course.

Homepage: [http://www.kirchkamp.de/](http://www.kirchkamp.de/)

Literature:
- Kruschke, Doing Bayesian Data Analysis
- Hoff, A First Course in Bayesian Statistical Methods.


Aim of the course
- Compare Bayesian with frequentist methods.
  - Two schools of statistical inference: Bayesian / Frequentist
    - Frequentist: Standard hypothesis testing, p-values, confidence intervals. Well known.
    - Bayesian: beliefs conditional on data.
- Learn to apply Bayesian methods.
  - What is the equivalent of frequentist method X in the Bayesian world?
  - How to put Bayesian methods into practice?

1.2 Motivation

1.3 Using Bayesian inference

Pros:
- Prior knowledge
- Model identification is less strict
- Small-sample size
- Non-standard models
- Non-normal distributions
- Categorical data
- Multi-level models
- Missing values
- Latent variables

- Interpretation

Cons:

- Prior knowledge
- Computationally expensive
- Model-fit diagnostic


- $X \leftarrow \theta$, $X$ is random, $\theta$ is fixed.
- Confidence intervals and p-values are easy to calculate.
- Interpretation of confidence intervals and p-values is awkward.
- p-values depend on the intention of the researcher.
- We can test “Null-hypotheses” (but where do these Null-hypotheses come from).
- Not good at accumulating knowledge.
- More restrictive modelling.

Bayesian: (Thomas Bayes, 1702-1761; Metropolis et al., “Equations of State Calculations by Fast Computing Machines”. Journal of Chemical Physics, 1953.)

- $X \rightarrow \theta$, $X$ is fixed, $\theta$ is random.
- Requires more computational effort.
- “Credible intervals” are easier to interpret.
- Can work with “uninformed priors” (similar results as with frequentist statistics)
• Efficient at accumulating knowledge.
• Flexible modelling.

Most people are still used to the frequentist approach. Although the Bayesian approach might have clear advantages it is important that we are able to understand research that is done in the context of the frequentist approach.

\[ \operatorname{Pr}(A \land B) = \operatorname{Pr}(A) \cdot \operatorname{Pr}(B | A) = \operatorname{Pr}(B) \cdot \operatorname{Pr}(A | B) \]

rewrite: \[ \operatorname{Pr}(A) \cdot \frac{\operatorname{Pr}(B | A)}{\operatorname{Pr}(B)} = \operatorname{Pr}(A | B) \]

with \( A = \theta \) \text{ parameter} and \( B = X \) \text{ data}:

\[ \frac{\operatorname{Pr}(\theta) \cdot \operatorname{Pr}(X | \theta)}{\int \operatorname{Pr}(\theta) \operatorname{Pr}(X | \theta) \, d\theta} \]

\[ = \operatorname{Pr}(\theta | X) \]

Before we come to a more formal comparison, let us compare the two approaches, frequentist versus Bayesian, with the help of an example.

I will use an example from the legal profession. Courts have to decide whether a defendant is guilty or innocent. Scientists have to decide whether a hypothesis is correct or not correct. Statistically, in both cases we are talking about the value of a parameter. \( \theta = \text{ guilty or } \theta = \text{ not guilty} \). Alternatively, \( \beta = 0 \) or \( \beta \neq 0 \).

My hope is that the legal context makes it more obvious how the decision process fails or succeeds.

**The prosecutors’ fallacy**
Assuming that the prior probability of a random match is equal to the probability that the defendant is innocent.

Two problems:

• p-values depend on the researcher’s intention. E.g. multiple testing (several suspects, perhaps the entire population, is “tested”, only one suspect is brought to trial)

• Conditional probability (neglecting prior probabilities of the crime)

• Lucia de Berk:
  - \( \operatorname{Pr}(\text{evidence|not guilty}) = 1/342 \text{ million} \)
  - \( \operatorname{Pr}(\text{evidence|not guilty}) = 1/25 \)
• Sally Clark
  – \( \Pr(\text{evidence}|\text{not guilty}) = \frac{1}{73} \text{ million} \)
  – \( \Pr(\text{not guilty}|\text{evidence}) = 78\% \)

The Sally Clark case

• 1996: First child dies from SIDS (sudden infant death syndrome): \( P = \frac{1}{8543} \)
• 1998: Second child dies from SIDS: \( P = \frac{1}{8543} \)
• \( \rightarrow \): \( \Pr(\text{evidence}|\text{not guilty}) = \left(\frac{1}{8543}\right)^2 \approx \frac{1}{73} \text{ million} \)

Problems:

• Correlation of SIDS within a family. \( \Pr(2\text{nd child}) = \left(\frac{1}{8543}\right) \times 5 \ldots 10 \)
• SIDS is actually more likely in this case: \( P = \frac{1}{8543} \rightarrow P = \frac{1}{1300} \)
  \( \Pr(\text{evidence}|1 \text{ not guilty mother}) = \frac{1}{(1300 \cdot 130)} = 0.000592\% \)
• Intention of the researcher/multiple testing: \( \approx 750 \text{ 000 births in England and Wales / year. How likely is it to find two successive SIDS or more among 750 000 mothers.} \)
  \( \Pr(\text{evidence}|750 \text{ 000 not guilty mothers}) = 98.8\% \)

But what is the (posterior) probability of guilt? Here we need prior information.

• What is the prior probability of a mother murdering her child?

\[
\frac{\Pr(\theta)}{\text{prior}} \cdot \frac{\Pr(X|\theta)}{\text{likelihood}} \cdot \frac{1}{\Pr(X)} = \frac{\Pr(\theta|X)}{\text{posterior}}
\]

\[
\frac{\Pr(g)}{\text{prior}} \cdot \frac{\Pr(X|g)}{\text{likelihood}} \cdot \frac{1}{\Pr(g) \cdot \Pr(X|g) + (1 - \Pr(g)) \cdot \Pr(X|\text{not } g)} = \frac{\Pr(g|X)}{\text{posterior}}
\]

Data from the U.S.A. (Miller, Oberman, 2004): per 600 000 mothers 1 killed child, \( \Pr(g) = \frac{1}{600 \text{ 000}} \).
\( \Pr(X|g) = 1, \Pr(X) = \frac{1}{600 \text{ 000}} + \frac{599 \text{ 999}}{600 \text{ 000}} \cdot \frac{1}{1300 \cdot 130} \)

\( \Pr(g|\text{evidence}) = 22\% \)

If \( \Pr(g) = \frac{1}{18800} \) then \( \Pr(g|\text{evidence}) = 90\% \)
If $\Pr(g) = 1/1710$ then $\Pr(g|\text{evidence}) = 99\%$

\[
\Pr(X|\theta) \neq \Pr(\theta|X)
\]

- The interpretation of $\Pr(X|\theta)$ as a p-value is affected by multiple testing (the intention of the researcher)
- $\Pr(\theta|X)$ is not affected by multiple testing (the intention of the researcher)
- $\Pr(\theta|X)$ forces us to think about a (subjective) prior.

**Lessons**

- Since p-values in Null-hypothesis significance testing are derived under the assumption that the Null-hypothesis is true:
  \[
  \rightarrow \text{When the Null-hypothesis is rejected, we can’t make any statement about the effect, except that the Null-hypothesis is not likely.}
  \]
- Since the Null-hypothesis is a point-hypothesis, the statement might be trivial.

**1.4 The intention of the researcher — p-hacking**

$X$ data

$\phi_j$ test procedure
• choice of control variables
• data exclusion
• coding
• analysis
• interactions
• predictors

T(\(X, \phi_j\)) test result

p-hacking

• perform J tests: \{\ldots, T(X, \phi_j), \ldots\}
• report the best result, given the data: T(\(X, \phi_{\text{best}}\))

→ to correct for multiple testing we need to know J ↓
→ robustness checks (for all J ↓)

An example: A researcher uses 60 explanatory variables to explain one dependent variable. Here we assume (for simplicity) that they all have the same standard error \(\sigma = 1\).
smallest $p$-value: no correction $p = 0.011$
Holm’s adjustment $p = 0.69$

A statement about the $p$-value depends on the intention of the researcher. It is affected by multiple testing.

A statement about the posterior odds does not depend on the intention of the researcher. It does, though, depend on a prior.

Above we assumed a flat prior. Is this reasonable? Perhaps, if we have already studied dozens of these variables, and they all seem to be drawn from a distribution with $\mu = 0$ and $\sigma = 1$, it is no longer reasonable to have a flat prior.

Above we pretended to be ignorant. We used a flat prior in each study. Now we use a prior for $\beta$: $\beta_i \sim N(0, 1)$

<table>
<thead>
<tr>
<th>$\beta_i$</th>
<th>$p$-value</th>
<th>Holm’s adj.</th>
<th>$\log_{10}(\text{odds}(\beta &gt; 0))$</th>
</tr>
</thead>
</table>

largest odds: flat prior $\beta_i > 0/\beta_i < 0$ odds=170 : 1
informed prior $\beta_i > 0/\beta_i < 0$ odds=26 : 1

Pretending to be ignorant and assuming a flat prior can be misleading.

- Flat prior in the Sally Clark case:
  $\Pr(\text{guilt}) : \Pr(\text{innocence}) = \frac{1}{2} : \frac{1}{2}$.
  This is absurd.

- Also $H_0 : \forall_i \beta_i = 0$ could be absurd.

1.5 Compare: The Maximum Likelihood estimator

Maximum likelihood has very nice asymptotic properties.
But what if the assumptions for these properties are not fulfilled?

- Consistency
- Asymptotic normality
  - $\theta_0$ must be away from the boundary (not trivial with panel data).
  - the number of nuisance parameters must not increase with the sample size (not trivial with panel data).
- Efficiency when the sample size tends to infinity

**Example**  A wants to estimate the capacity of B’s firm (German tank problem):

- A samples the daily output of the firm.
- The output of the firm follows a uniform distribution over $[0, \theta]$.
- The sample contains the numbers $\{1, 2, 3\}$.

The Maximum Likelihood estimator:

$$L = \begin{cases} 
1/\theta^3 & \text{if } \theta \geq 3 \\
0 & \text{otherwise}
\end{cases} \Rightarrow \theta_{\text{ML}}^* = 3$$

The ML estimator yields a biased estimate.
For a Bayesian estimate we need a prior. Let us assume that \( A \) assumes all capacities between 0 and \( M \) to be equally likely. Then (for \( \theta \geq 3 \)):

\[
\Pr(\theta | X) = \frac{\Pr(\theta) \cdot \Pr(X|\theta)}{\int \Pr(\theta) \Pr(X|\theta) \, d\theta} = \frac{\frac{1}{M} \frac{1}{\theta^3}}{\int_3^M \frac{1}{M} \frac{1}{\theta^3} \, d\theta} = \frac{18M^2}{(M^2 - 9)\theta^3}
\]

Hence

\[
E(\theta) = \int \theta \cdot f(\theta) \, d\theta = \int_3^M \theta \cdot \Pr(\theta | X) \, d\theta = \frac{6M}{M + 3}
\]

E.g. if \( M = 10 \), then \( E(\theta) = 60/13 = 4.615 \). If \( M = 100 \), then \( E(\theta) = 600/103 = 5.825 \).

Remember: density function:

\[
\Pr(\theta | X) = \frac{18M^2}{(M^2 - 9)\theta^3}
\]

Distribution function:

\[
F(q) = \int_3^q \Pr(\theta | X) \, d\theta = \frac{M^2(q^2 - 9)}{(M^2 - 9)q^2}
\]

Quantile function:

\[
\text{Solve } F(q) = p \Rightarrow Q(p) = \frac{3M}{\sqrt{(1-p)M^2 + 9p}}
\]

For \( M = 10 \) we have \( CI_{[2.5\%, 97.5\%]} = \left[ \frac{20 \cdot \sqrt{30}}{\sqrt{1303}}, \frac{6 \cdot 10^{3/2}}{\sqrt{451}} \right] = [3.035, 8.743] \)
1.6 Terminology

1.6.1 Probabilities

Consider the following statements:

**Frequentist probability**

- The probability to throw two times a six is 1/36.
- The probability to win the state lottery is about 1:175 000 000.
- The probability of rainfall on a given day in August is 1/3.
- The probability for a male human to develop lung or bronchus cancer is 7.43%.

**Subjective probability**

- The probability of rainfall tomorrow is 1/3.
- The probability that a Mr. Smith develops lung or bronchus cancer is 7.43%.
- The probability that Ms. X committed a crime is 20%.

**Frequentist**

- $P = \text{objective probability (sampling of the data } X \text{ is infinite)}$.

  $\rightarrow$ but what if the event occurs only once (rainfall tomorrow, Mr. Smith’s health,...)?

  $\rightarrow$ von Mises: event has no probability

  $\rightarrow$ Popper: invent a fictitious population from which the event is a random sample (propensity probability).

- Parameters $\theta$ are unknown but fixed during repeated sampling.

**Bayesian**

- $P = \text{subjective probability of an event (de Finetti/Ramsey/Savage)}$

  $\approx$ betting quotients

- Parameters $\theta$ follow a (subjective) distribution.

**Fixed quantities:**

**Frequentist**
• Parameters $\theta$ are fixed (but unknown).

**Bayesian**
• Data $X$ are fixed.

**Probabilistic statements:**

**Frequentist**
• …about the frequency of errors $p$.
• Data $X$ are a random sample and could potentially be resampled infinitely often.

**Bayesian**
• …about the distribution of parameters $\theta$.

### 1.6.2 Prior information

• Prior research (published / unpublished)
• Intuition (of researcher / audience)
• Convenience (conjugate priors, vague priors).

Prior information is *not* the statistician’s personal opinion. Prior information is the result of and subject to scientific debate.

### 1.6.3 Objectivity and subjectivity

• Bayesian decision making requires assumptions about…
  
  – $\Pr(\theta)$ (prior information)
  
  – $g_0, g_0$ (cost and benefits)

Scientists might disagree about this information.

$\rightarrow$ Bayesian decision making is therefore accused of being “subjective”.

Bayesian’s might “choose” priors, cost and benefits, to subjectively determine the result. E.g. in the Sally Clark case, the researcher might “choose” the prior probability of a mother to kill her child to be $1/1710$ to conclude guilt with $\Pr(g|\text{evidence}) = 99\%$.

The Bayesian’s answer:
• Prior information, cost and benefits are relevant information. Disregarding them (as the frequentists do) is a strange concept of “objectivity”.

• Priors, cost and benefits are subject to scientific debate, like any other assumption. We have to talk about priors, not assume them away.

• Subjectivity exists in both worlds:
  – B.+F. make assumptions about the model → more dangerous than priors.
  – In F. the intention of the researcher has a major influence on p-values and confidence intervals.

1.6.4 Issues

• Probability: frequentist vs. subjective.

• Prior information, how to obtain?

• Results, objective / subjective.

• Flexible modelling: F. has only a limited number of models.
  F: precise method, using a tool which is sometimes not such a good representation of the problem.
  B: approximate method, using a tool which can give a more precise representation of the problem.

• Interpretation: p-values versus posteriors.
  B. predicts (posterior) probability of a hypothesis.
  F. writes carefully worded statements which are wrong 5% of the time (or any other probability) provided $H_0$ is true.

• Quality of decisions: p-values are only a heuristic for a decision rule.
  B.’s decisions are better in expectation.

1.7 Decision making

Which decision rule, Bayesian or frequentist, uses information more efficiently?

$$\Pr(\theta) \cdot \Pr(X|\theta) \cdot \frac{1}{\Pr(X)} = \Pr(\theta|X)$$

Assume $\theta \in \{0, 1\}$. Implement an action $a \in \{0, 1\}$. Payoffs are $\pi_{a\theta}$.

We have $\pi_{11} > \pi_{01}$ and $\pi_{00} > \pi_{10}$, i.e. it is better to choose $a = \theta$. Expected payoffs:

$$E(\pi|a) = \pi_{a1} \cdot \Pr(\theta = 1) + \pi_{a0} \cdot \Pr(\theta = 0)$$
Optimal decision: choose $a = 1$ iff

$$\pi_{11} \cdot \Pr(\theta = 1) + \pi_{10} \cdot \Pr(\theta = 0) > \pi_{01} \cdot \Pr(\theta = 1) + \pi_{00} \cdot \Pr(\theta = 0).$$

Rearrange: choose $a = 1$ iff

$$\Pr(\theta = 1) \underbrace{(\pi_{11} - \pi_{01})} > \Pr(\theta = 0) \underbrace{(\pi_{00} - \pi_{10})}.$$

Here $g_a$ can be seen as the gain from choosing the correct action (or the loss from choosing the wrong action) if $\theta = a$.

If we have some data $X$:

$$\Pr(\theta = 1|X) g_1 > \Pr(\theta = 0|X) g_0.$$ 

Bayes’ rule:

$$\Pr(\theta) \cdot \Pr(X|\theta) \cdot \frac{1}{\Pr(X)} = \Pr(\theta|X)$$

choose $a = 1$ iff

$$\frac{g_1}{g_0} \Pr(\theta = 1) > \frac{\Pr(\theta = 0)}{\Pr(\theta = 1)} \cdot \Pr(X|\theta = 1) > \Pr(X|\theta = 0)$$

Bayesian chooses $a = 1$ iff $\Pr(X|\theta = 0) < \frac{g_1}{g_0} \Pr(\theta = 1) \cdot \Pr(X|\theta = 1)$

Frequentist chooses $a = 1$ iff $\Pr(X|\theta = 0) < 0.05$.

(Here we assume that $H_0$ is $\theta = 0$.)

**When do Bayesians and Frequentists disagree?**

\[ g_1 \frac{\Pr(\theta = 1)}{\Pr(\theta = 0)} \cdot \Pr(X|\theta = 1) < 0.05. \]
For very small and for very large values of \( \Pr(X|\theta = 0) \) both Bayesians and frequentists make the same choice. Only in the range between \( \frac{g_1}{g_0} \frac{\Pr(\theta = 1)}{\Pr(\theta = 0)} \cdot \Pr(X|\theta = 1) \) and 0.05 choices differ. In that range the Bayesian choice maximises expected payoffs while the frequentist does not.

1.8 Technical Background

\((\Omega, \mathcal{F}, P)\) is a probability space:

- \(\Omega\), a sample space (set of possible outcomes)
- \(\mathcal{F}\), a set of events (\(\mathcal{F}\) is a collection of subsets of \(\Omega\) that is closed under countable-fold set operations. \(\mathcal{F}\) is a \(\sigma\)-algebra, \((\mathcal{F}, \Omega)\) is a measurable space).
- \(P\), a probability measure function.

Axiom 1: \( \forall A \in \Omega : \Pr(A) \geq 0 \)

Axiom 2: \( \Pr(\Omega) = 1 \)

Axiom 3: For pairwise disjoint \( A_i \in \mathcal{F} : \Pr(\sum_i A_i) = \sum_i \Pr(A_i) \)

- \( \Pr(\neg A) = 1 - \Pr(A) \)
- \( \Pr(A \cup B) = \Pr(A) + \Pr(B) - \Pr(A \cap B) \)

Definition: \( \Pr(A|B) := \Pr(A \cap B) / \Pr(B) \)

\[ \rightarrow \Pr(\theta) \cdot \frac{\Pr(X|\theta)}{\int \Pr(\theta) \cdot \Pr(X|\theta) \, d\theta} = \Pr(\theta|X) \]

\( X \sim \text{Exp}(\lambda) \)

\( E(X) = 1/\lambda \)

\( \text{var}(X) = 1/\lambda^2 \)

\( X \sim \text{Gamma}(\alpha, \beta) \)

\( E(X) = \alpha/\beta \)

\( \text{var}(X) = \alpha/\beta^2 \)

\( X \sim \text{Poisson}(\lambda) \)

\( E(X) = \lambda \)

\( \text{var}(X) = \lambda \)

\( X \sim \text{Beta}(\alpha, \beta) \)

\( E(X) = \alpha/(\alpha + \beta) \)

\( \text{var}(X) = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)} \)

\( X \sim N(\mu, \tau) \)

\( E(X) = \mu \)

\( \text{var}(X) = 1/\tau \)

\( X \sim \chi^2(k) \)

\( E(X) = k \)

\( \text{var}(X) = 2k \)

\( X \sim t(k) \)

\( E(X) = 0 \)

\( \text{var}(X) = k/(k - 2) \)

\( X \sim F(k_1, k_2) \)

\( E(X) = k_2/(k_2 - 2) \)

\( \text{var}(X) = \frac{2k_1^2(k_1 + k_2 - 2)}{k_1(k_2 - 2)^2(k_2 - 4)} \)
2 A practical example

2.1 The distribution of the population mean

Here we ask the question: “What is the probability to be arrested in North Carolina in 1981 (conditional on a crime committed)?”

```r
library(Ecdat)
data(Crime)
xyplot(crmrte ~ prbarr, data=Crime, subset=year==81)
```

Example: Crime in North Carolina counties in 1981

![Scatter plot of crmrte vs prbarr]

```r
y <- subset(Crime, year==81)[["prbarr"]]

We can have a look at a part of the data with head:

```r
head(y)
```

```
[1] 0.289696 0.202899 0.406593 0.431095 0.631579 0.369650
```

If we suspect that average rate to be arrested to be 0.3, we use a `t.test`:

```r
t.test(y, mu=.3)
```

One Sample t-test
data:  y
An alternative: The Bayesian Approach  

Required:

- Priors for $\mu$ and $\tau$.
- Likelihood: $y \sim N(\mu, \tau)$ with $\tau = 1/\sigma^2$

$\rightarrow$ Posterior distribution of $\mu$ and $\tau$.

We will here just “use” our software got get a result. Below we will explain what the software actually does.

```r
library(runjags)
X.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu,tau)
  }
  mu ~ dnorm (0,.0001)
  tau ~ dgamma(.01,.01)
  sd <- sqrt(1/tau)
}
',
X.jags<-run.jags(model=X.model,data=list(y=y),monitor=c("mu","sd"))
```

Notation for nodes

- Stochastic nodes (discrete/continuous univariate/multivariate distributed):

  $$y[i] \sim \text{dnorm}(\mu, \tau)$$

  $$\ldots$$

  $$\mu \sim \text{dnorm}(0,.0001)$$

  - ...can be specified by `data` (have always this value)
  - ...can be specified by `inits` (have this value before the first sample)
  - ...can be unspecified

  Note: if `data` or `inits` sets a value to NA, this means “unspecified”.

- Deterministic nodes:
sd <- sqrt(1/tau)

JAGS samples from the posterior of $\mu$ and $\tau$. Here is a distribution for $\mu$:

```
plot(X.jags, var="mu", plot.type=c("trace","density"))
```

![Distribution plot for $\mu$](image)

Here is a summary of our estimation results:

```
summary(X.jags)
```

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>0.272034</td>
<td>0.3001</td>
<td>0.325091</td>
<td>0.298984</td>
<td>0.01346867</td>
<td>0.2986450</td>
<td>0.00009523789</td>
</tr>
<tr>
<td>sd</td>
<td>0.110458</td>
<td>0.128171</td>
<td>0.148288</td>
<td>0.1288285</td>
<td>0.009782551</td>
<td>0.1278104</td>
<td>0.00006969668</td>
</tr>
<tr>
<td>MC%ofSD</td>
<td>0.7</td>
<td>20000</td>
<td>0.002831988</td>
<td>1.0000202</td>
<td>0.9999656</td>
<td></td>
<td></td>
</tr>
<tr>
<td>SSeff</td>
<td>0.7</td>
<td>-0.003087232</td>
<td>0.9999656</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Testing a point prediction in the `t.test`, as in $\mu = 0.3$, is a bit strange, at least from the Bayesian perspective. It might be more interesting to make a statement about the probability of an interval.

First, we convert our jags-object into a dataframe: How probable is $\mu \in (0.29, 0.31)$?

```
X.df <- data.frame(as.mcmc(X.jags))
str(X.df)
```

'data.frame': 20000 obs. of 2 variables:
$ mu: num 0.331 0.289 0.316 0.317 0.303 ... 
$ sd: num 0.126 0.119 0.116 0.122 0.118 ...
We can now say, how probable it is, ex post, that $\mu \in [0.29, .31]$:

```
100*mean(with(X.df,mu > 0.29 & mu < 0.31))
```

[1] 54.755

...or in a more narrow interval:

```
100*mean(with(X.df,mu > 0.299 & mu < 0.301))
```

[1] 5.825

```
100*mean(with(X.df,mu > 0.2999 & mu < 0.3001))
```

[1] 0.645

If, say, a government target is to have an average arrest rate of at least 0.25, we can now calculate the probability that $\mu > 0.25$.

```
100*mean(with(X.df,mu > 0.25))
```

[1] 99.98

Odds for $\mu > 0.25$

```
p<-mean(with(X.df,mu > 0.25))
p/(1-p)
```

[1] 4999

In the following section we will explain how all this works:

### 2.2 Gibbs sampling

The model that we specified above, contained two parts, a likelihood and a prior. Here is a model with only a prior:

We use JAGS notation:

```
dnorm(\mu, \tau)
```

with $\mu =$mean and $\tau = 1/\sigma^2 =$precision.

```
modelPri <- 'model {
  mu ~ dnorm (0,.0001)
}
```

Now we use this model to draw a sample of size 100, so far only given the prior.
Here are the properties of our sample:

```r
summary(pri.jags)
```

<table>
<thead>
<tr>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
<th>SSeff</th>
</tr>
</thead>
<tbody>
<tr>
<td>-195.466</td>
<td>3.00919</td>
<td>169.874</td>
<td>-2.227341</td>
<td>98.9364</td>
<td>8.278946</td>
<td>14.38885</td>
<td>8.4</td>
<td>143</td>
</tr>
<tr>
<td>AC.10</td>
<td>psrf</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>mu 0.1140479</td>
<td>1.005316</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

And here is a plot of the distribution. Since we did not include a likelihood, it is at the same time the distribution of the prior and of the posterior.

```r
plot(pri.jags, var="mu", plot.type=c("trace", "density"))
```

![Plot of distribution](image)

### 2.3 Convergence

In the sample above we saw only observations after round 5000, i.e. we skipped 5000 samples of adaptation and burnin. This was not necessary, since we had only a prior, i.e. the sampler would only sample from the prior.

Things become more interesting when we add a likelihood (which we will do next). Then it is not clear that the sampler will directly start sampling from the posterior distribution. It takes some time. The hope is that after 5000 samples of adaptation and burnin...
the sampler has converged, that it samples from an almost stationary distribution (which is described by our prior and our likelihood).

In the following we add the likelihood to the model. We drop adaptation and burnin and see what happens at the start.

```r
ini<-genInit(2,function(i) list(mu=c(100,-100)[i]))
X2.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu,tau)
  }
  mu ~ dnorm (200,.0001)
  tau ~ dgamma(.01,.01)
  sd <- sqrt(1/tau)
}
'
X100.jags<-run.jags(model=X2.model,data=list(y=y),
  monitor=c("mu","sd"),adapt=0,burnin=0,sample=100,inits=ini)
```

(To obtain reproducible results, I use a custom genInit function in this handout. You find this function in the attachment to this document. You also find a definition in Section 16. For your own calculations you can also drop the inits=ini part.)

```r
plot(X100.jags,var="mu",plot.type=c("trace"))
```

At least here the sampler seems to converge fast. Nevertheless, including a safe number of adaptation and burnin is good practice. Let us look at the posterior with adaptation and burnin:
2.4 Distribution of the posterior

We have now a posterior distribution for \( \mu \) and one for (the nuisance parameter) \( \sigma_d \).

\[
\text{plot(X1001.jags, var=c("mu", "sd"), plot.type=c("density"))}
\]

Since we sampled from two separate “chains”, we actually have two such distributions. Luckily they are quite similar. This enhances our trust in the estimate of the posterior.

2.5 Accumulating evidence

↑ Above we used non-informative priors. \( (\mu \sim N(0, 0.0001)) \)

- Assume that we know something about \( \mu \) (or that we talk to somebody who knows).
  - E.g. we ran a similar study in a different state.
    We found \( \mu = 0.4 \) and \( \sigma_\mu = 0.014 \) (i.e. the same \( \sigma_\mu \) from our data, but a different \( \mu \)).
    \( (\sigma_\mu = 0.014 \text{ is equivalent to } \tau_\mu = 1/\sigma^2_\mu = 5102) \)
  - Now we combine the data, i.e. we use a prior \( \mu \sim N(0.4, 5102) \)
XA.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu,tau)
  }
  mu ~ dnorm (0.4,1/0.014^2)
  tau ~ dgamma(.01,.01)
  sd <- sqrt(1/tau)
}
XA.jags <- run.jags(model=XA.model, data=list(y=y), monitor=c("mu","sd"))

summary(XA.jags)

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>0.329932</td>
<td>0.3514325</td>
<td>0.372681</td>
<td>0.3516260</td>
<td>0.01084617</td>
<td>0.3515811</td>
<td>0.00008563219</td>
</tr>
<tr>
<td>sd</td>
<td>0.117564</td>
<td>0.1381880</td>
<td>0.161305</td>
<td>0.1390088</td>
<td>0.01128897</td>
<td>0.1372491</td>
<td>0.00008957363</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MC%ofSD</th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu</td>
<td>0.8</td>
<td>16043</td>
<td>0.002953220</td>
<td>0.9999910</td>
</tr>
<tr>
<td>sd</td>
<td>0.8</td>
<td>15884</td>
<td>0.006200908</td>
<td>0.9999954</td>
</tr>
</tbody>
</table>

- Prior mean: 0.4
- Sample mean: 0.3
- Posterior mean: 0.35

“A Bayesian is one who, vaguely expecting a horse, and catching a glimpse of a donkey, strongly believes he has seen a mule.”

### 2.6 Priors

- noninformative, flat, vague, diffuse
- weakly informative: intentionally weaker than the available prior knowledge, to keep the parameter within “reasonable bounds”.
- informative: available prior knowledge.

### 3 Conjugate Priors

#### 3.1 Accumulating evidence, continued

**Exchangability**

When we accumulate data $X_1$ and $X_2$ it should not matter, whether we first observe $X_1$ and then add $X_2$ or vice versa.

Call $\mathcal{D}$ the distribution of parameter $\theta$. 
This is easier if $\mathcal{D}_0, \mathcal{D}_1, \mathcal{D}_2, \mathcal{D}_{12}$ belong to one family. For some combinations of prior distributions and likelihoods, we can actually calculate analytically the posterior distribution.

**Conjugate priors for a likelihood function**

<table>
<thead>
<tr>
<th>Likelihood</th>
<th>known</th>
<th>model parameter</th>
</tr>
</thead>
<tbody>
<tr>
<td>$X \sim \mathcal{N}(\mu, \sigma^2)$</td>
<td>$\tau = 1/\sigma^2$</td>
<td>$\mu \sim \mathcal{N}(\mu_0, \sigma_0^2)$</td>
</tr>
<tr>
<td>$X \sim \mathcal{N}(\mu, \tau)$</td>
<td>$\mu$</td>
<td>$\tau \sim \Gamma(\alpha_0, \beta_0)$</td>
</tr>
<tr>
<td>$X \sim \text{bern}(p)$</td>
<td>$p$</td>
<td>$p \sim \text{Beta}(\alpha_0, \beta_0)$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>i.e. $\mu \sim \mathcal{N}(\mu_0, \sigma_0^2)$</td>
</tr>
</tbody>
</table>

If the prior model parameter follows the conjugate prior, then the posterior model parameter is in the same family.

**3.2 Normal Likelihood**

**Conjugate Priors, example: Normal Likelihood $\mu$**

- Likelihood: $X \sim \mathcal{N}(\mu, \sigma^2)$ with known $\tau = 1/\sigma^2$.
- Model parameter: $\mu$
- Conjugate prior distribution: $\mu \sim \mathcal{N}(\mu_0, \sigma_0^2)$
- Prior hyperparameter: $\mu_0, \sigma_0^2$ i.e. prior $\mu \sim \mathcal{N}(\mu_0, \sigma_0^2)$.
- Posterior hyperparameter:

\[
\mu_{\text{post}} = \left( \frac{\mu_0}{\sigma_0^2} + \frac{n \cdot \bar{x}}{\sigma^2} \right) \left( \frac{1}{\sigma_0^2} + \frac{n}{\sigma^2} \right) = \frac{\tau_0 \mu_0 + n \tau \bar{x}}{\tau_0 + n \tau}
\]

\[
\tau_{\text{post}} = 1/\sigma_{\text{post}}^2 = \left( \frac{1}{\sigma_0^2} + \frac{n}{\sigma^2} \right) = \tau_0 + n \tau
\]

i.e. posterior $\mu \sim \mathcal{N}(\mu_{\text{post}}, \sigma_{\text{post}}^2)$.

In other words:

- Prior parameter: $\mu \sim \mathcal{N}(\mu_0, \tau_0)$
- Likelihood: $X \sim \mathcal{N}(\mu, \tau)$
• Posterior parameter: $\mu \sim N(\mu_{\text{post}}, \tau_{\text{post}})$.

Terminology:

• Hyperparameters: $\mu_0, \tau_0$ (they determine the distribution of $\mu$)
• Parameters: $\mu, \tau$
• Posterior hyperparameters: $\mu_{\text{post}}, \tau_{\text{post}}$

**Conjugate Priors, example: Normal Likelihood $\tau$**

• Likelihood: $X \sim N(\mu, \tau)$ with known $\mu$.
• Model parameter: $\tau = 1/\sigma^2$
• Conjugate prior distribution: $\tau \sim \Gamma(\alpha_0, \beta_0)$
• Prior hyperparameter: $\alpha_0, \beta_0$
• Posterior hyperparameter:
  \[
  \begin{align*}
    \text{shape} & \quad \alpha_{\text{post}} = \alpha_0 + \frac{n}{2} \\
    \text{rate} & \quad \beta_{\text{post}} = \beta_0 + \frac{n}{2} \text{var}(x)
  \end{align*}
  \]

In other words:

• Prior parameter: $\tau \sim \Gamma(\alpha_0, \beta_0)$
• Likelihood: $X \sim N(\mu, \tau)$
• Posterior parameter: $\tau \sim \Gamma(\alpha_{\text{post}}, \beta_{\text{post}})$.

Terminology:

• Hyperparameters: $\alpha_0, \beta_0$ (they determine the distribution of $\mu$)
• Parameters: $\mu, \tau$
• Posterior hyperparameters: $\alpha_{\text{post}}, \beta_{\text{post}}$
3.3 Bernoulli Likelihood

Conjugate Priors, example: Bernoulli Likelihood

- Likelihood: $X \sim \text{bern}(p)$.
- Model parameter: $p$
- Conjugate prior distribution: $p \sim \text{Beta}(\alpha_0, \beta_0)$
- Prior hyperparameter: $\alpha_0, \beta_0$
- Posterior hyperparameter:

$$
\alpha_{\text{post}} = \alpha_0 + \sum x_i \\
\beta_{\text{post}} = \beta_0 + n - \sum x_i
$$

In other words:

- Prior parameter: $p \sim \text{Beta}(\alpha_0, \beta_0)$
- Likelihood: $X \sim \text{bern}(p)$
- Posterior parameter: $p \sim \text{Beta}(\alpha_{\text{post}}, \beta_{\text{post}})$

Terminology:

- Hyperparameters: $\alpha_0, \beta_0$ (they determine the distribution of $\mu$)
- Parameters: $\mu, \tau$
- Posterior hyperparameters: $\alpha_{\text{post}}, \beta_{\text{post}}$

3.4 Problems with the analytical approach

- Restrictive for...
  - priors
  - likelihood ("the model" in the frequentist world)
- For many relevant cases we have no analytical solution.
- $\rightarrow$ numerical methods, Markov Chain Monte Carlo (MCMC) methods, Metropolis-Hastings sampling, Gibbs sampling,…

Construct a Markov Chain that has the posterior distribution as its equilibrium distribution.
3.5 Exercises

1. An event can have two possible outcomes, 0 or 1. You are interested in the probability \( p \) of obtaining a 1. You assume that \( p \) follows a Beta distribution. Your prior is that the parameters of the Beta distribution are \( \alpha = \beta = 0 \). You observe three times a 1 and no 0. What is your posterior for \( \alpha \) and \( \beta \)?

2. Later you observe three more times a 1 and four times 0. Given all your observations, what is now your posterior for \( \alpha \) and \( \beta \)?

3. A random variable \( X \) follows a normal distribution with mean \( \mu \) and precision \( \tau \). You want to infer the posterior distribution of \( \mu \). Your prior for \( \mu \) also follows a normal distribution \( \mu \sim N(\mu_0, \tau_0) \) with hyperparameters \( \mu_0 = 10 \) and \( \tau_0 = 2 \). Now you observe a sample of size \( n = 10 \), mean \( \mu = 20 \) and precision \( \tau = 1/5 \). What is your posterior \( \mu_{\text{post}} \)?

4. What is your posterior \( \tau_{\text{post}} \)?

4 Linear Regression

We use linear regression as an example to illustrate some issues of the mechanics behind the MCMC sampling mentioned in the previous section.

4.1 Introduction

Example: Crime in North Carolina in 1981  
Let us have another look at the crime rate and the arrest rate in North Carolina.

```r
library(Ecdat)
data(Crime)
xyplot(crmrte ~ prbarr, data=Crime, subset=year==81, type=c("p", "r"))
```
We suspect that the crime rate is a linear function of the arrest rate. The standard tool would be OLS:

\[
\text{est} \leftarrow \text{lm}(\text{crmrte} \sim \text{prbarr}, \text{data} = \text{Crime}, \text{subset} = \text{year} == 81) \\
\text{summary}(\text{est})
\]

Call:
\text{lm(formula = crmrte ~ prbarr, data = Crime, subset = year == 81)}

Residuals:

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-0.027125</td>
<td>-0.009932</td>
<td>-0.000848</td>
<td>0.007013</td>
<td>0.046819</td>
</tr>
</tbody>
</table>

Coefficients:

|                | Estimate | Std. Error | t value | Pr(>|t|) |
|----------------|----------|------------|---------|---------|
| (Intercept)    | 0.048577 | 0.004261   | 11.400  | < 2e-16 *** |
| prbarr         | -0.052924 | 0.013129 | -4.031  | 0.000118 *** |

---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01571 on 88 degrees of freedom  
Multiple R-squared: 0.1559, Adjusted R-squared: 0.1463  
F-statistic: 16.25 on 1 and 88 DF, p-value: 0.0001177
OLS

\[ Y = \beta_0 + \beta_1 X + u \text{ where } u \sim N(0, \sigma^2) \]
\[ Y \sim N(\beta_0 + \beta_1 X, \sigma^2) \]
\[ Y \sim N(\beta_0 + \beta_1 X, \tau) \]

Both notations are equivalent. The former is more common in the frequentist context, the latter more common in the Bayesian context.

Now we do the same exercise in JAGS:

```r
data<-with(subset(Crime,year==81),list(y=crmrte,x=prbarr))
reg.model<-'
model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(beta0 + beta1*x[i],tau)
  }
  beta0 ~ dnorm (0,.0001)
  beta1 ~ dnorm (0,.0001)
  tau ~ dgamma(.01,.01)
}
',
reg.jags<-run.jags(model=reg.model, data=data, monitor=c("beta0","beta1"))
```

JAGS model summary statistics from 20000 samples (chains = 2; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta0</td>
<td>0.03718</td>
<td>0.048729</td>
<td>0.060465</td>
<td>0.048809</td>
<td>0.0059047</td>
<td>0.048566</td>
<td>0.00014557</td>
</tr>
<tr>
<td>beta1</td>
<td>-0.089838</td>
<td>-0.053553</td>
<td>-0.018249</td>
<td>-0.053655</td>
<td>0.018233</td>
<td>-0.053532</td>
<td>0.00045766</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MC%ofSD</th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta0</td>
<td>2.5</td>
<td>1645</td>
<td>0.20733</td>
<td>1.0007</td>
</tr>
<tr>
<td>beta1</td>
<td>2.5</td>
<td>1587</td>
<td>0.20768</td>
<td>1.0006</td>
</tr>
</tbody>
</table>

Total time taken: 0.4 seconds

```r
summary(est)[["coefficients"]]
```

|            | Estimate | Std. Error | t value | Pr(>|t|)  |
|------------|----------|------------|---------|----------|
| Intercept  | 0.04857749 | 0.004261233 | 11.399865 | 5.087079e-19 |
| prbarr     | -0.05292384 | 0.013128767 | -4.031136 | 1.177237e-04 |

The distribution we get here is very similar to the distribution parameters from the simple OLS.
4.2 Demeaning

This is a technical issue. Demeaning might help improving the performance of our sampler.

Demeaning does not change the estimate of the coefficient of $X$, it does change the constant, though.

\[ Y = \beta_0 + \beta_1 X \]  
\[ Y - \bar{Y} = \beta_0 - \bar{Y} + \beta_1 \bar{X} + \beta_1 (X - \bar{X}) \]

Let us look more closely at the distribution of the sampled posterior:

```r
reg.df <- data.frame(combine.mcmc(reg.jags))
xyp <- xyplot(beta1~beta0, data=head(reg.df, 1000))
```

![Scatter plot of beta0 vs beta1](image.png)

We see that beta0 and beta1 are correlated. As we will see below, this correlation makes the Gibbs sampler slower.

Now we demean the data:

```r
data2 <- with(data, list(y = y - mean(y), x = x - mean(x)))
reg2.jags <- run.jags(model = reg.model, data = data2, monitor = c("beta0", "beta1"))
```
JAGS model summary statistics from 20000 samples (chains = 2; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta0</td>
<td>-0.0054</td>
<td>-6.58e-06</td>
<td>0.0054</td>
<td>-3.52e-06</td>
<td>0.0023</td>
<td>-0.0000159</td>
</tr>
<tr>
<td>beta1</td>
<td>-0.0890</td>
<td>-0.0530</td>
<td>-0.0167</td>
<td>-0.0531</td>
<td>0.0185</td>
<td>-0.052681</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MCerr</th>
<th>MC%ofSD</th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta0</td>
<td>0.00001</td>
<td>0.7</td>
<td>20000</td>
<td>-0.0079</td>
<td>1.0003</td>
</tr>
<tr>
<td>beta1</td>
<td>0.00012</td>
<td>0.7</td>
<td>21291</td>
<td>0.0066</td>
<td>1.0001</td>
</tr>
</tbody>
</table>

Total time taken: 0.4 seconds

The estimate for $\beta_1$ does not change (here we assume that we are mainly interested in the marginal effect, i.e. in $\beta_1$).

Now $\beta_0$ and $\beta_1$ are no longer correlated:

Convergence with raw and demeaned data  To better understand convergence, we look at the first few samples in each case. Let us look at 5 chains with 5 samples each:
In the demeaned case, the Gibbs sampler jumps almost immediately to the center of the distribution. Convergence is reached within a small number of steps. In the not-demeaned case the Gibbs sampler walks slowly along the joint distribution of $\beta_0$ and $\beta_1$. It takes a longer number of steps to reach the center of the distribution and to converge.

Here are 10 samples:
Here are 100 samples:
The Gibbs sampler can only increase the probability of one single posterior parameter in one step. In the posterior distribution the sampler, therefore, can only move parallel to one of the axes. If the posterior distribution is asymmetric (as in the raw data) convergence is slow.

### 4.3 Correlation

A related problem of the Gibbs sampler is that two successive samples may be correlated.
Correlation in the raw case

beta0

beta1

acfplot(as.mcmc(reg2.jags), ylim=c(-1,1), aspect="fill", layout=c(2,1))
A sample of 10,000 can, thus, not be treated as 10,000 independent observations.

Thinning (take only every nth sample) does not lose much information.

### 4.4 The three steps of the Gibbs sampler

**The three steps of the Gibbs sampler**

**adaptation:** optimise the algorithm

**burnin:** converge to the approximate shape of the distribution

**sample:** use a fixed algorithm to sample from posterior

Our problem:

- make sure that the sampler has converged

Solution:

- Demeaning (converge quickly to posterior)
- Good init values (start already from within the posterior)
4.5 Exercises

Consider the year 1979 from the data set LaborSupply from Ecdat.

1. Which variables could explain labor supply?
2. Estimate your model for the year 1979 only.
3. Compare your results with and without demeaning.

5 Finding posteriors

5.1 Overview

\[
\frac{Pr(\theta) \cdot Pr(X|\theta) \cdot \frac{1}{Pr(X)}}{\int Pr(\theta) \cdot Pr(X|\theta) \, d\theta} = Pr(\theta|X)
\]

Find \(Pr(\theta|X)\):

- Exact: but \(\int Pr(\theta) \cdot Pr(X|\theta) \, d\theta\) can be hard (except for specific priors and likelihoods).

- MCMC Sampling
  - Rejection sampling: can be very slow (for a high-dimensional problem, and our problems are high-dimensional).
  - Metropolis–Hastings: quicker, samples are correlated, requires sampling of \(\theta\) from joint distribution \(Pr(X|\theta)\).
  - Gibbs sampling: quicker, samples are correlated, requires sampling of \(\theta_i\) from conditional (on \(\theta_{-i}\)) distribution \(Pr(X|\{\theta_i, \theta_{-i}\})\).

→ this is easy! (at least much easier than \(Pr(X|\theta)\))

5.2 Example for the exact way:

Above we talked about conjugate priors. Consider the case of Normal Likelihood:

- Likelihood: \(N(\mu, \sigma^2)\) with known \(\tau = 1/\sigma^2\).
- Model parameter: \(\mu\)
- Conjugate prior distribution: \(\mu \sim N()\)
- Prior hyperparameter: \(\mu_0, \sigma_0^2\)
• Posterior hyperparameter:

\[
\mu_{\text{post}} = \left( \frac{\mu_0}{\sigma_0^2} + \frac{n \cdot \bar{x}}{\sigma^2} \right) / \left( \frac{1}{\sigma_0^2} + \frac{n}{\sigma^2} \right) = \frac{\tau_0 \mu_0 + n \tau \bar{x}}{\tau_0 + n \tau}
\]

\[
\tau_{\text{post}} = 1/\sigma^2_{\text{post}} = \left( \frac{1}{\sigma_0^2} + \frac{n}{\sigma^2} \right) = \tau_0 + n \tau
\]

5.3 Rejection sampling

\[
\Pr(\theta) \cdot \Pr(X|\theta) \cdot \frac{1}{\Pr(X)} = \Pr(\theta|X)
\]

\[
\int \Pr(\theta) \cdot \Pr(X|\theta) \, d\theta
\]

How it works: Iterate the following:

• Sample a candidate \( \theta \) and a uniformly distributed random number \( r \).

• If \( \Pr(\theta) \cdot \Pr(X|\theta) > r \) then \( \theta \) goes into the sample.

Problems:

• Slow (reject most of the time)

• \( \max(r) > \max(\Pr(\theta) \cdot \Pr(X|\theta)) \)

The more dimensions we have, the more rejections. It would be nice to sample mainly in the posterior.
5.4 Metropolis-Hastings

\[
\frac{\text{Pr}(\theta) \cdot \text{Pr}(X|\theta)}{\int \text{Pr}(\theta) \cdot \text{Pr}(X|\theta) \, d\theta} = \frac{1}{\text{Pr}(X)} = \text{Pr}(\theta|X)
\]

Generates a sample of \(\text{Pr}(\theta|X)\), needs only \(f(\theta) = \text{Pr}(\theta) \cdot \text{Pr}(X|\theta)\) (more generally, MH requires only a function which is proportional to the density function desired).

How it works:

- Starting point \(\eta = \theta_0\), arbitrary symmetric PDF \(Q(\theta|\eta)\), e.g. \(Q = N\).
- Iterate:
  - Sample a candidate \(\theta' \sim Q(\theta'|\theta_t)\).
  - Acceptance ratio is \(\alpha = f(\theta')/f(\theta_t)\).
  - If \(\alpha \geq 1\): \(\theta_{t+1} = \theta'\) \(\text{jump}\).
  - If \(\alpha < 1\): with probability \(\alpha\) we have \(\theta_{t+1} = \theta'\) \(\text{jump}\), otherwise \(\theta_{t+1} = \theta_t\) \(\text{stay}\).

Advantages:

- Faster than rejection sampling (in particular if \(\theta\) is from a higher dimension).
Disadvantages:

- Samples are correlated (depending on Q).
  - If Q makes wide jumps: more rejections but less correlation.
  - If Q makes small jumps: fewer rejections but more correlation.
- Initial samples are from a different distribution. “burn-in” required.
- Finding a “good” jumping distribution $Q(x|y)$ can be tricky.

5.5 Gibbs sampling

Essentially as in Metropolis-Hastings, except that sampling is performed for each component of $\theta$ sequentially.

- determine $\theta_1^{t+1}$ with $f(\theta_1|\theta_2^t, \theta_3^t, \theta_4^t, \ldots, \theta_n^t)$
- determine $\theta_2^{t+1}$ with $f(\theta_2|\theta_1^{t+1}, \theta_3^t, \theta_4^t, \ldots, \theta_n^t)$
- determine $\theta_3^{t+1}$ with $f(\theta_3|\theta_1^{t+1}, \theta_2^{t+1}, \theta_4^t, \ldots, \theta_n^t)$
  
  ... 

- determine $\theta_n^{t+1}$ with $f(\theta_n|\theta_1^{t+1}, \theta_2^{t+1}, \ldots, \theta_{n-1}^{t+1})$

Advantages:

- Requires only conditional distributions. $f(\theta_i|\theta_{-1})$, not joint distributions.
- Finding a “good” jumping distribution $Q(x|y)$ is easier.

Disadvantages:

- Samples are correlated (potentially more than in MH if the number of dimensions is large).
- Initial samples are from a different distribution. “burn-in” required.
- Can get stuck on “unconnected islands”.

In the following example we create on purpose a situation with two (almost) unconnected islands:

```r
x <- rbinom(9, 1, .5)
x
[1] 0 0 0 1 0 1 0 0 0

island.mod <- 'model {
  for (i in 1:length(x)) {
    x[i] ~ dbern(z^2)
  }
  z ~ dunif(-1, 1)
}

island.jags <- run.jags(model=island.mod, data=list(x=x), monitor=c("z"), inits=ini)
```
Now we create more evidence (of the same type). This makes the Gibbs sampler more persistent.

```r
x <- rbinom(50, 1, .5)
x
```

```
[1]  0  0  0  1  0  1  0  0  0  0  1  0  0  0  1  0  1  0  1  0  1  1  1  1  0  1  0  1  0  1  0  0  0  0  1  0  1  1  0
[39]  1  0  1  0  1  0  0  1  1  1  1  1
```

```r
island2.jags <- run.jags(model=island.mod, data=list(x=x), monitor=c("z"), inits=ini)
```
5.6 Check convergence

5.6.1 Gelman, Rubin (1992): potential scale reduction factor

Idea: take $k$ chains, discard “warm-up”, split remaining chains, so that we have $2k$ sequences $\{\psi\}$, each of length $n$.

$$B = \text{between sequence variance}$$
$$W = \text{within sequence variance}$$

Variance of all chains combined:

$$\hat{\sigma}^2 = \frac{n-1}{n} W + \frac{B}{n}$$

Potential scale reduction:

$$\hat{R} = \sqrt{\frac{\hat{\sigma}^2}{W}}$$

Let us first look at the $psrf$ for a “nice” case:

gelman.plot(reg.jags)
And now the island case:
\texttt{gelman.diag(island.jags)}

Potential scale reduction factors:

Point est. Upper C.I.
\begin{verbatim}
z  1.24  1.87
\end{verbatim}

\texttt{summary(island.jags)[, \textit{c("Mean","SD","SSeff","psrf")}]} 

\begin{verbatim}
Mean         SD        SSeff      psrf
-0.08831173  0.47855963  7.00000000  1.06201321
\end{verbatim}

\texttt{gelman.plot(island2.jags)}
```r
gelman.diag(island2.jags)

Potential scale reduction factors:

Point est. Upper C.I.
z 32.1 71.9

summary(island2.jags)[,c("Mean","SD","SSeff","psrf")]

<table>
<thead>
<tr>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>-0.0003617406</td>
<td>0.6761477054</td>
<td>11651.0000000000</td>
<td>32.3159757818</td>
</tr>
</tbody>
</table>

acfplot(as.mcmc(reg.jags), aspect="fill", layout=c(2,1))
```
As a result of autocorrelation, the “effective size” is smaller than the sample size.

```
effectiveSize(as.mcmc.list(reg.jags))
```

\[
\begin{array}{ll}
\text{beta0} & 1789.540 \\
\text{beta1} & 1624.828 \\
\end{array}
\]

```
effectiveSize(as.mcmc.list(reg2.jags))
```

\[
\begin{array}{ll}
\text{beta0} & 20451.02 \\
\text{beta1} & 19314.84 \\
\end{array}
\]

```
effectiveSize(as.mcmc.list(island.jags))
```

\[
\begin{array}{l}
z \\
7.127945 \\
\end{array}
\]

```
effectiveSize(as.mcmc.list(island2.jags))
```

\[
\begin{array}{l}
z \\
11650.9 \\
\end{array}
\]

The effective sample size is also shown in the standard summary:

```
summary(reg.jags)[,c("Mean","SD","SSeff","psrf")]
```

\[
\begin{array}{llll}
\text{Mean} & \text{SD} & \text{SSeff} & \text{psrf} \\
\text{beta0} & 0.04852363 & 0.00601396 & 1790 & 1.004317 \\
\text{beta1} & -0.05281219 & 0.01853645 & 1625 & 1.003977 \\
\end{array}
\]
5.7 A better vague prior for $\tau$

When we specify a regression model we need a precision parameter $\tau$. So far we did this:

```r
reg.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(beta0 + beta1*x[i], tau)
  }
  beta0 ~ dnorm(0,.0001)
  beta1 ~ dnorm(0,.0001)
  tau ~ dgamma(.01,.01)
}'
```
Here is an alternative specification:

```r
reg2.model <- 'model {
for (i in 1:length(y)) {
    y[i] ~ dnorm(beta0 + beta1*x[i],tau)
}
beta0 ~ dnorm (0,.0001)
beta1 ~ dnorm (0,.0001)
tau ~ dgamma(m^2/d^2,m/d^2)
m ~ dgamma(1,1)
d ~ dgamma(1,1)
}
',
```
• $\tau \sim \Gamma(0.01, 0.01)$

Remember:

• If $\tau \sim \Gamma(\alpha, \beta)$ then $E(\tau) = \alpha/\beta$ and $\text{var}(\tau) = \alpha/\beta^2$.
• $\alpha = 0.01$, $\beta = 0.01$ works well if $E(\tau) \approx 1$ and $\text{var}(\tau) \approx 100$.

Alternative:

• $\tau \sim \Gamma \left( \frac{m^2}{d^2}, \frac{m}{d^2} \right)$
• $m \sim \Gamma(1, 1)$
• $d \sim \Gamma(1, 1)$

$\rightarrow E(\tau) = m$, $\text{var}(\tau) = d^2$

• Speed: no substantial loss
• Convergence: often faster

5.8 More on History

To learn more about the development of the field, have a look at the following text:

6 Robust regression

6.1 Robust regression with the Crime data

Crime rate and probability of arrest:

```r
xyplot(crmrte ~ prbarr, data = subset(Crime, year == 81), type = c("p", "r"))
```

Is the linear regression estimate driven by outliers?

**Residuals follow a normal distribution**

\[
\text{crmrte} \sim \mathcal{N}(\beta_0 + \beta_1 \text{prbarr}, \tau)
\]

JAGS model summary statistics from 20000 samples (chains = 2; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta0</td>
<td>0.039913</td>
<td>0.048647</td>
<td>0.05738</td>
<td>0.048602</td>
<td>0.0045036</td>
<td>0.048399</td>
<td>0.00011053</td>
</tr>
<tr>
<td>beta1</td>
<td>-0.080205</td>
<td>-0.053137</td>
<td>-0.0261</td>
<td>-0.053014</td>
<td>0.013874</td>
<td>-0.053153</td>
<td>0.00033847</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MC%ofSD</th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta0</td>
<td>2.5</td>
<td>1660</td>
<td>0.99996</td>
<td></td>
</tr>
<tr>
<td>beta1</td>
<td>2.4</td>
<td>1680</td>
<td>0.99995</td>
<td></td>
</tr>
</tbody>
</table>

Total time taken: 0.6 seconds
Allow fat tails

\[ \text{crmrte} \sim t(\beta_0 + \beta_1 \text{prbarr}, \tau, k) \]

\[ k = \infty \quad \text{green} \quad k = 3 \quad \text{orange dashed} \quad k = 1 \quad \text{blue dotted} \]

\[
\begin{align*}
\text{Density of the } t \text{ distribution} \\
0.0 & \quad 0.1 & \quad 0.2 & \quad 0.3 & \quad 0.4 \\
-4 & \quad -2 & \quad 0 & \quad 2 & \quad 4
\end{align*}
\]

\[ k = \infty \]
\[ k = 3 \]
\[ k = 1 \]

\[
\begin{align*}
\text{t1.model} & \leftarrow \text{'model { for (i in 1:length(y)) { y[i] \sim dt(beta0 + beta1*x[i],tau,1) } beta0 \sim dnorm (0,.0001) beta1 \sim dnorm (0,.0001) tau \sim dgamma(m^2/d^2,m/d^2); m \sim dexp(1); d \sim dexp(1); }'}
\end{align*}
\]

data<-with(subset(Crime,year==81),list(y=crmrte,x=prbarr))
t1.jags<-run.jags(model=t1.model,data=data,monitor=c("beta0","beta1"))
t1.df<-data.frame(as.mcmc(t1.jags))

densityplot(reg.df[["beta1"]],plot.points=FALSE,
auto.key=list(text=c("$k=\infty$","$k=1$"),columns=2),
xlab="$\beta_1$") + 
layer(panel.densityplot(t1.df[["beta1"]],plot.points=FALSE),style=2)
Now make \( k \) endogeneous. We need a prior for \( k \):

\[
\text{xxExp} \leftarrow \text{within(data.frame(list(x=exp(seq(-2,7,.1)))),y=pexp(x,1/30))}
\]

\[
\text{xyplot}(y \sim x, data=xxExp, scales=list(x=list(log=10)), xscale.components = xscale.components.log10)
\]
```
t.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dt(beta0 + beta1*x[i],tau,k)
  }
  beta0 ~ dnorm (0, .0001)
  beta1 ~ dnorm (0, .0001)
  tau ~ dgamma(m^2/d^2,m/d^2); m ~ dexp(1); d ~ dexp(1);
  k ~ dexp(1/30)
}
''

t.jags <- run.jags(model=t.model, data=data, monitor=c("beta0","beta1","k"))
t.df <- data.frame(as.mcmc(t.jags))

densityplot(reg.df[["beta1"]],plot.points=FALSE,
            auto.key=list(text=c("k=\infty","k=1","k\sim dexp"),columns=3),
            xlab="$\beta_1$") +
  layer(panel.densityplot(t1.df[["beta1"]],plot.points=FALSE),style=2) +
  layer(panel.densityplot(t.df[["beta1"]],plot.points=FALSE),style=3)

k = \infty \quad \quad \quad k = 1 \quad \quad k \sim dexp

xxExp <- within(data.frame(list(x=exp(seq(-2,7,.1))),'y=dexp(x,1/30)))
densityplot(t.df[["k"]],xlab="$k$",scales=list(x=list(log=10)),xscale.components = xscale.components)
```
JAGS model summary statistics from 20000 samples (chains = 2; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta0</td>
<td>0.039913</td>
<td>0.0486</td>
<td>0.05738</td>
<td>0.0486</td>
<td>0.0045</td>
<td>0.0484</td>
<td>0.0011053</td>
</tr>
<tr>
<td>beta1</td>
<td>-0.80205</td>
<td>-0.053137</td>
<td>-0.0261</td>
<td>-0.053014</td>
<td>0.013874</td>
<td>-0.053153</td>
<td>0.00033847</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MC%ofSD</th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta0</td>
<td>2.5</td>
<td>1660</td>
<td>0.19726</td>
</tr>
<tr>
<td>beta1</td>
<td>2.4</td>
<td>1680</td>
<td>0.19726</td>
</tr>
</tbody>
</table>

Total time taken: 0.6 seconds
6.2 Robust regression with the Engel data

```r
library(quantreg)
data(engel)
xyplot(foodexp ~ income, data=engel)
```

![Plot showing food expenses and income](image)

```
data<-with(engel,list(y=foodexp,x=income))
engel.reg.jags<-run.jags(model=reg.model,data=data,
                         monitor=c("beta0","beta1"),inits=ini)
engel.t.jags<-run.jags(model=t.model,data=data,
                        monitor=c("beta0","beta1","k"),inits=ini)
```

Here we find a much smaller $k$. We see that fat tails matter.

```r
engel.t.df<-as.mcmc(engel.t.jags)
densityplot(engel.t.df[["k"]],xlab="$k$",scales=list(x=list(log=10)),
xscale.components = xscale.components.log10.3,plot.points=FALSE,
auto.key=list(text=c("posterior $k$","prior $k$"),columns=3),xlim=c(.1,500))+
layer(panel.xyplot(log10(xxExp$x),xxExp$y,t="1"),scales=list(x=list(log=10)),style=2)
```
engel.reg.df <- data.frame(as.mcmc(engel.reg.jags))
all.df <- rbind.fill(within(engel.reg.df, type="$k=\infty$"), within(engel.t.df, type="$k\sim$dexp"))
densityplot(~ beta1, group=type, data=all.df, plot.points=FALSE, xlab="$\beta_1$", auto.key=list)

k \sim \text{dexp} \quad k = \infty


densityplot(~ beta1, group=type, data=all.df, plot.points=FALSE, xlab="$\beta_1$", auto.key=list)

k \sim \text{dexp} \quad k = \infty

JAGS model summary statistics from 20000 samples (chains = 2; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th>Param</th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
<th>SSeff</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta0</td>
<td>112.5</td>
<td>143.34</td>
<td>174.37</td>
<td>143.37</td>
<td>15.917</td>
<td>143.64</td>
<td>0.32063</td>
<td>2</td>
<td>2465</td>
</tr>
<tr>
<td>beta1</td>
<td>0.45943</td>
<td>0.48864</td>
<td>0.51525</td>
<td>0.48848</td>
<td>0.014371</td>
<td>0.48796</td>
<td>0.00028645</td>
<td>2</td>
<td>2517</td>
</tr>
</tbody>
</table>

AC.10 psrf
beta0 0.072199 1.0003
beta1 0.069264 1.0004

Total time taken: 0.7 seconds

JAGS model summary statistics from 20000 samples (chains = 2; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th>Param</th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
<th>SSeff</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta0</td>
<td>49.531</td>
<td>79.555</td>
<td>110.84</td>
<td>79.526</td>
<td>15.465</td>
<td>80.325</td>
<td>0.52445</td>
<td>3.4</td>
<td>869</td>
</tr>
<tr>
<td>beta1</td>
<td>0.52548</td>
<td>0.56206</td>
<td>0.59697</td>
<td>0.5621</td>
<td>0.018052</td>
<td>0.5613</td>
<td>0.00060312</td>
<td>3.3</td>
<td>896</td>
</tr>
<tr>
<td>k</td>
<td>2.0817</td>
<td>3.4211</td>
<td>5.2599</td>
<td>3.5472</td>
<td>0.84881</td>
<td>3.2658</td>
<td>0.013462</td>
<td>1.6</td>
<td>3976</td>
</tr>
</tbody>
</table>

AC.10 psrf
beta0 0.41396 1.0005
beta1 0.41633 1.0006
k 0.053926 1.0001

Total time taken: 33.3 seconds

6.3 Exercises

Consider the data set wages from Ecdat. The data set contains seven observations for each worker. Consider for each worker the first of these observations. You want to study the impact of education on wage.

1. Could there be outliers in lwage?

2. How can you take outliers into account?

3. Estimate your model.
7 Nonparametric

7.1 Preliminaries

- Is the “nonparametric” idea essentially frequentist?
- After all, with “nonparametrics” we avoid a discussion about distributions.
- Perhaps it would be more honest to model what we know about the distribution.

Still...

- Equivalent for “binomial test”: $X \sim \text{dbern}()$.
- Equivalent for “$\chi^2$ test”: $X \sim \text{dpois}()$.

Furthermore...

- As in the frequentist world we can translate one (less known) distribution to another by using ranks.

7.2 Example: Rank sum based comparison

Here we create two variables who both follow an exponential distribution. We might forget this information and use ranks to compare.

```r
set.seed(4)
xx<-rbind(data.frame(list(x=rexp(10,1),t=1)),
            data.frame(list(x=rexp(20,3),t=2)))
wilcox.test(x~t,data=xx)
```

Wilcoxon rank sum test

data:  x by t
W = 156, p-value = 0.01273
alternative hypothesis: true location shift is not equal to 0

densityplot(~x,group=t,data=xx)
The parametric approach Here we know that \( x \) follows an exponential distribution:

```
EXP.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dexp(beta[t[i]])
  }
  for (i in 1:2) {
    beta[i] ~ dgamma(m[i]^2/d[i]^2, m[i]/d[i]^2); m[i] ~ dexp(1); d[i] ~ dexp(1);
  }
}
```

\[
\text{data} \leftarrow \text{with(xx, list(y=x, t=t))}
\]

\[
\text{EXP.jags} \leftarrow \text{run.jags(model=EXP.model, data=data, monitor=c("beta", "qDiff"))}
\]

```
EXP.jags

JAGS model summary statistics from 20000 samples (chains = 2; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
<th>SSeff</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>0.52638</td>
<td>1.0664</td>
<td>1.7872</td>
<td>1.0989</td>
<td>0.32894</td>
<td>1.0186</td>
<td>0.0046557</td>
<td>1.4</td>
<td>4992</td>
</tr>
<tr>
<td>beta[2]</td>
<td>1.7014</td>
<td>2.7485</td>
<td>4.0375</td>
<td>2.802</td>
<td>0.60491</td>
<td>2.65</td>
<td>0.0086881</td>
<td>1.4</td>
<td>4848</td>
</tr>
<tr>
<td>qDiff</td>
<td>-3.083</td>
<td>-1.666</td>
<td>-0.39739</td>
<td>-1.703</td>
<td>0.68659</td>
<td>-1.5829</td>
<td>0.0097705</td>
<td>1.4</td>
<td>4938</td>
</tr>
</tbody>
</table>

AC.10 psrf

<p>| | | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>0.036854</td>
<td>1.0002</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>beta[2]</td>
<td>0.042479</td>
<td>1.0004</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```
qDiff 0.042407 0.99998

Total time taken: 0.7 seconds

\begin{verbatim}
plot(EXPR.jags, var="qDiff", plot.type=c("trace", "density"))
\end{verbatim}

\begin{figure}
\includegraphics[width=\textwidth]{plot.png}
\end{figure}

```
EXPR.df<-data.frame(as.mcmc(EXPR.jags))
(p<-mean(EXPR.df["qDiff"]<0))

[1] 0.99655
```

The odds are, hence

```
p/(1-p)

[1] 288.8551
```

**The non-parametric approach**  As with the rank sum test, we normalise the data, using ranks. Regardless what the initial distribution was, if the two samples come from the same distribution, we now have a uniform distribution. Using \texttt{qnorm} we obtain a normal distribution.
\[ \text{xx} <- \text{within}(\text{xx}, \{ r <- \text{rank}(x); n <- \text{qnorm}((r-.5)/\text{max}(r)); \}) \]

\[
\text{densityplot}(\sim n, \text{group}=t, \text{data}=\text{xx})
\]

\[
\text{NP.model} <- '\text{model} \{ \\
\text{for (i in 1:length(y))} \{ \\
\quad y[i] \sim \text{dnorm(beta[t[i]], tau[t[i]])} \\
\}\text{for(i in 1:2)} \{ \\
\quad \text{beta[i] \sim \text{dnorm(0, .0001)}} \\
\quad \text{tau[i] \sim \text{dgamma(m[i]^2/d[i]^2, m[i]/d[i]^2); m[i] \sim \text{dexp(1)}; d[i] \sim \text{dexp(1)}}; \\
\quad \text{qBet[i]<-pnorm(beta[i],0,1)} \\
\}\text{qDiff<-qBet[1]-qBet[2]}'\}
\]

\[
\text{data<-with(\text{xx}, \text{list}(y=n, t=t))} \\
\text{NP.jags<-run.jags(\text{model}=\text{NP.model, data=\text{data}}, \text{monitor}=c("qBet", "qDiff", "tau"))}
\]

\[
\text{JAGS model summary statistics from 20000 samples (chains = 2; adapt+burnin = 5000):}
\]

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>qBet[1]</td>
<td>0.51498</td>
<td>0.72528</td>
<td>0.90258</td>
<td>0.71713</td>
<td>0.10052</td>
<td>0.73647</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>qBet[2]</td>
<td>0.22149</td>
<td>0.38143</td>
<td>0.5428</td>
<td>0.38479</td>
<td>0.081606</td>
<td>0.37161</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>qDiff</td>
<td>0.07169</td>
<td>0.33926</td>
<td>0.57884</td>
<td>0.33234</td>
<td>0.12943</td>
<td>0.33946</td>
<td>0.7</td>
<td></td>
</tr>
<tr>
<td>tau[1]</td>
<td>0.37563</td>
<td>1.2102</td>
<td>2.3619</td>
<td>1.2842</td>
<td>0.53886</td>
<td>1.0736</td>
<td>1.7</td>
<td></td>
</tr>
<tr>
<td>tau[2]</td>
<td>0.51239</td>
<td>1.1427</td>
<td>1.9118</td>
<td>1.1828</td>
<td>0.36586</td>
<td>1.0728</td>
<td>1.5</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>qBet[1]</td>
<td>19587</td>
<td>0.0059906</td>
<td>1.0002</td>
</tr>
</tbody>
</table>
The odds are, hence

\[ p/(1-p) \]

[1] 96.56098

### 8 Identification

**Collinearity**  Regressors which are collinear are (in the linear model) not simultaneously identifiable. Here we create two such regressors.
library(Ecdat)
data(Crime)
dataC<-within(subset(Crime,year==81), c("crmrte","prbarr"), prbarr100<-100*prbarr)
est<-lm(crmrte ~ prbarr + prbarr100, data=dataC)
summary(est)

Call:
lm(formula = crmrte ~ prbarr + prbarr100, data = dataC)

Residuals:
Min 1Q Median 3Q Max
-0.027125 -0.009932 -0.000848 0.007013 0.046819

Coefficients: (1 not defined because of singularities)
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.048577   0.004261 11.400  < 2e-16 ***
prbarr      -0.052924   0.013129  -4.031  0.000118 ***
prbarr100    NA         NA        NA         NA
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.01571 on 88 degrees of freedom
Multiple R-squared: 0.1559, Adjusted R-squared: 0.1463
F-statistic: 16.25 on 1 and 88 DF, p-value: 0.0001177

As we see, OLS makes an identifying assumption and sets the coefficient of prbarr100 to zero.
Now we do the same with JAGS:

regM.model<-'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(inprod(beta,X[i,]),tau)
  }
  for (k in 1:K) {
    beta[k] ~ dnorm (0,.0001)
  }
  tau ~ dgamma(m^2/d^2,m/d^2); m ~ dexp(1); d ~ dexp(1);}
ini <- genInit(4,function(i) list(beta1=rnorm(1,0,0.0001),
                                  beta2=rnorm(1,0,0.0001)))
regM.jags<-run.jags(model=regM.model,
                      data=with(dataC, list(y=crmrte,X=cbind(1,prbarr,prbarr100),K=3)),
                      monitor=c("beta"),
                      inits=ini, sample=100, thin=100)

summary(regM.jags)[,c("Mean","SD","SSeff","psrf")]

Mean     SD  SSeff    psrf
- Standard errors for coefficients are much larger.

- The potential scale reduction factor is larger than 1.1.

\[ y \sim N(\beta_0 + \beta_1 X + \beta_2 \cdot 100 \cdot X, \tau) \]
• The joint distribution of $\beta_1$ and $\beta_2$ shows the dependence of the two regressors.

• But – perhaps we are only interested in $\beta_2 + 100 \cdot \beta_3$?

\[
\text{with(regM.df, plot(density(beta.2 + beta.3 * 100), main="", xlab="$\beta_2 + 100 \cdot \beta_3$")}
\]
Identification Summary

- Many frequentist tools try to obtain point estimates. Hence, they must detect under-identification. Often they make identifying assumptions on their own.

- In the Bayesian world we estimate a joint distribution. Under-identification need not be a problem. It shows up as large standard deviation, lack of convergence, a large `gelman.diag`, etc.

9 Discrete Choice

9.1 Labor force participation

```r
library(Ecdat)
data(Participation)
ecdfplot(~lnnlinc, group=lfp, data=Participation, auto.key=list(title="labour force participation: "
labour force participation:
no ——— yes

9.2 A generalised linear model

\[ P(Y = 1|X) = \Phi(\beta_0 + \beta_1 X) \]

alternative: \[ \Phi^{-1}(P(Y = 1|X)) = \beta_0 + \beta_1 X \]

probit.glm<-glm(lfp="yes" ~ lnnlinc, data=Participation, family=binomial(link=probit))
summary(probit.glm)[["coefficients"]]
```
9.3 Bayesian discrete choice

```r
probit.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dbern(p[i])
    p[i] <- phi(inprod(X[i,],beta))
  }
  for (k in 1:K) {
    beta[k] ~ dnorm (0,.0001)
  }
}
'
Part.data <- with(Participation, list(y = as.numeric(lfp == "yes"), X = cbind(1, lnnlinc), K = 2))
probit.jags <- run.jags(model = probit.model, modules = "glm",
data = Part.data, inits = ini, monitor = c("beta"))
summary(probit.jags)[, c("Mean", "SD", "SSeff", "psrf")]
```

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>5.9838734</td>
<td>1.1902539</td>
<td>29769</td>
<td>1.000072</td>
</tr>
<tr>
<td>beta[2]</td>
<td>-0.5698306</td>
<td>0.1114429</td>
<td>29693</td>
<td>1.000070</td>
</tr>
</tbody>
</table>

(We should use \( \phi(\ldots) \) and not \( \text{pnorm}(\ldots, 0, 1) \). The latter is slower to converge.)

The following specification is equivalent:

```r
probit.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dbern(p[i])
    probit(p[i]) <- inprod(X[i,],beta)
  }
  for (k in 1:K) {
    beta[k] ~ dnorm (0,.0001)
  }
}
'
Part.data <- with(Participation, list(y = as.numeric(lfp == "yes"), X = cbind(1, lnnlinc), K = 2))
probit.jags <- run.jags(model = probit.model, modules = "glm",
data = Part.data, inits = ini, monitor = c("beta"))
summary(probit.jags)[, c("Mean", "SD", "SSeff", "psrf")]
```

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>5.9838734</td>
<td>1.1902539</td>
<td>29769</td>
<td>1.000072</td>
</tr>
<tr>
<td>beta[2]</td>
<td>-0.5698306</td>
<td>0.1114429</td>
<td>29693</td>
<td>1.000070</td>
</tr>
</tbody>
</table>
9.4 Exercise

Consider the dataset *Mroz* from *Ecdat*.

- Which variables could explain work participation?
- Estimate your model.

10 Count data

10.1 Poisson model

The Poisson process:

- During one unit of time you expect \( \lambda \) many events.
- During 1/10 unit of time you expect \( \lambda / 10 \) many events.
- During 1/100 unit of time you expect \( \lambda / 100 \) many events.

Events are stochastically independent of each other (no interaction among events).

(Purely random process)

The Poisson distribution for different values of \( \lambda \):
\[ Y \sim \text{Pois}(\lambda) \]
\[ Y \sim \text{Pois}(\exp(\beta_0 + \beta_1 X)) \]
\[ Y \sim \text{Pois}(\exp(1 + 2X)) \]

Generate some data:

```r
set.seed(123)
N <- 100
x <- rnorm(N)
y <- rpois(N, exp(1 + 2*x))
pois.glm <- glm(y ~ x, family = poisson(link = log))
summary(pois.glm)[["coefficients"]]
```

```
Estimate Std. Error  z value Pr(>|z|)
(Intercept) 0.9611632 0.06330281 15.18358 4.542996e-52
x 2.0389185 0.03799288 53.66581 0.000000e+00
```

We could specify the model like this...

```r
count.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dpois(lambda[i])
    lambda[i] <- exp(inprod(X[i,], beta))
  }
}
```
for (k in 1:K) {
    beta[k] ~ dnorm (0,.0001)
}
}
count.jags <- run.jags(model=count.model, modules="glm",
data=list(y=y, X=cbind(1,x), K=2), inits=ini, monitor=c("beta"))
summary(count.jags)[,c("Mean", "SD", "SSeff", "psrf")]

... or like this...

count.model <- 'model {
for (i in 1:length(y)) {
    y[i] ~ dpois(lambda[i])
    log(lambda[i]) <- inprod(X[i,], beta)
}
for (k in 1:K) {
    beta[k] ~ dnorm (0,.0001)
}
}
count.jags <- run.jags(model=count.model, modules="glm",
data=list(y=y, X=cbind(1,x), K=2), inits=ini, monitor=c("beta"))
summary(count.jags)[,c("Mean", "SD", "SSeff", "psrf")]

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>0.9583643</td>
<td>0.06270963</td>
<td>3032</td>
<td>1.000460</td>
</tr>
<tr>
<td>beta[2]</td>
<td>2.0400627</td>
<td>0.03769276</td>
<td>3648</td>
<td>1.000473</td>
</tr>
</tbody>
</table>

10.2 Negative binomial

Count data and the negative binomial distribution

**Poisson distribution:** Pois(λ)

- Mean: λ
- Variance: λ

**Negative binomial distribution:** NB(μ, r)

- Mean: μ
- Variance: μ + μ²/r

Poisson is a special case of NB:

\[ \lim_{r \to \infty} \text{NB}(\mu, r) = \text{Pois}(\mu) \]

The NB distribution for μ = 4 and for different values of r:
Two notations:

\[ \text{NB}(p, r) \quad (\text{used by JAGS as } \text{dnegbin}(p, r)) \]
\[ \text{NB} (\mu, r) \quad (\text{perhaps easier to interpret}) \]

where \( p = \frac{r}{r + \mu} \) or \( \mu = \frac{r}{p} - r \)

\[
\lim_{r \to \infty} \text{NB}(\mu, r) = \text{Pois}(\mu)
\]

(r is sometimes called \( \theta \))

Let us use the NB model with our Poisson data:

```r
countNB.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnegbin(p[i], r)
    p[i] <- r/(r+mu[i])
    log(mu[i]) <- inprod(X[i,], beta)
  }
  r ~ dgamma(m^2/d^2, m/d^2); m ~ dexp(1); d ~ dexp(1);
  for (k in 1:K) {
    beta[k] ~ dnorm (0, 0.0001)
  }
}
'

countNB.jags <- run.jags(model=countNB.model, modules=c("glm"),
                          data=list(y=y, X=cbind(1, x), K=2), inits=ini, monitor=c("beta", "r"))

summary(countNB.jags)[, c("Mean", "SD", "SSeff", "psrf")]
```
10.3 Exercise

Consider the dataset Doctor from Ecdat. Explain the number of doctor visits as a function of children in the household.

- Use a Poisson model.
- Use a negative binomial model.

11 Multinomial (polytomous) logit

11.1 Motivation and background

Multinomial logit

- choices are mutually exclusive
- choices are exhaustive
- choices are finite
Problems

• one can map problems that do not look mutually exclusive or not exhaustive into a problem that is

E.g.: heating modes: gas / oil / wood / electricity

What about households which use, e.g., gas + electricity →
  – introduce an additional category
  – ask for ‘primary source of heating’

Some households do not use any of the above:
  – introduce an additional category

• Using discrete choice models for metric variables
  – E.g.: consumption of goods which follow a non-linear tariff (telephone, electricity)

Random utility models

Can we tell a story like in the logit/probit case?

A latent variable model (random utility model):

\[
\begin{align*}
\eta_1 &= x' \beta_1 + \xi_1 \\
\eta_2 &= x' \beta_2 + \xi_2 \\
\eta_3 &= x' \beta_3 + \xi_3 \\
&\vdots
\end{align*}
\]

The decision maker chooses alternative \( k \) if \( \eta_k \geq \eta_j \) for all \( j \)

Note: these models are equivalent to their affine transformations.

Normalisations

• We often normalise the constant part of one of the equations to zero.

• If \( \xi_j \) are i.i.d. we often normalise their variance to a convenient value.
  
  (this implies that different distributions for \( \xi \), will lead to different scales for coefficients — logit coefficients will be \( \pi/\sqrt{6} \) times larger than probit.)
Differences

Let us look at the differences between two alternatives:

\[ v_{kj} = \eta_k - \eta_j = \chi'(\beta_k - \beta_j) + \xi_k - \xi_j \]

- \( \xi \sim N(0, 1) \): \( \xi_k - \xi_j \) has variance 2 and covariance 1 (for \( k \neq j \))

```r
dgumbel <- function(x) exp(-exp(-x)-x)
plot(dnorm,-4,4,ylab="$f(\xi)$",xlab="$\xi$")
curve(dgumbel,add=TRUE,lty=2)
legend("topleft",c("Normal","Gumbel"),lty=1:2)
```

- \( \xi \sim \text{Gumbel} \) \( \left( F_{\text{Gumbel}}(\xi) = e^{-e^{-\xi}} \right) \) then
  - the difference \( v_{ki} \) follows a logistic distribution

\[
\Pr(y = k|\xi_k) = \prod_{j \neq k} F_{\text{Gumbel}}(\chi'(\beta_k - \beta_j) + \xi_k) \Pr(\eta_j < \eta_k)
\]

average over \( \xi_k \)

\[
\Pr(y = k) = \int f_{\text{Gumbel}}(\xi_k) \prod_{j \neq k} F_{\text{Gumbel}}(\chi'(\beta_k - \beta_j) + \xi_k) \, d\xi_k
\]

\[
\Pr(y = k) = \frac{e^{\chi'\beta_k}}{\sum_{i=1}^{m} e^{\chi'\beta_i}}
\]

\( \rightarrow \) we get the following multinomial logit (McFadden)
\[ \Pr(y = 1) = \frac{\sum_{k=1}^{m} e^{x' \beta_k}}{e^{x' \beta_2}} \\
\Pr(y = 2) = \frac{\sum_{k=1}^{m} e^{x' \beta_k}}{e^{x' \beta_3}} \\
\Pr(y = 3) = \frac{\sum_{k=1}^{m} e^{x' \beta_k}}{e^{x' \beta_3}} \\
\vdots \\
\]

\[
\cdot \quad 0 < \Pr(y = k) < 1 \\
\cdot \quad \sum_k \Pr(y = k) = 1 \\
\uparrow \beta_k \text{ are not identified}
\]

Normalise:

\[ \Pr(y = 1) = \frac{1}{1 + \sum_{k=2}^{m} e^{x' \beta_k}} \\
\Pr(y = 2) = \frac{1}{1 + \sum_{k=2}^{m} e^{x' \beta_k}} \\
\Pr(y = 3) = \frac{1}{1 + \sum_{k=2}^{m} e^{x' \beta_k}} \\
\vdots \\
\]

the odds ratios are:

\[
\frac{\Pr(y = k)}{\Pr(y = 1)} = e^{x' \beta_k}
\]

This is a strong assumption on the error terms.

**Independence from irrelevant alternatives — IIA**

Example:

- Dependent = choice of travel mode
- unobservable = personal preference for/against means of mass transportation (tube/train).

\[
\frac{\Pr(y = \text{tube})}{\Pr(y = 1)} = e^{x' \beta_{\text{tube}}} \quad \frac{\Pr(y = \text{train})}{\Pr(y = 1)} = e^{x' \beta_{\text{train}}}
\]
→ choices/error terms are correlated.

→ multinomial logit can represent systematic variation of choices (explained by observed characteristics) but not systematic individual (unobserved) variation of choices.

**The log-likelihood:**

With \( I_k(y_i) = \begin{cases} 1 & \text{if } y_i = i \\ 0 & \text{otherwise} \end{cases} \)

\[
\log L = \sum_i \sum_{k=1}^{m} I_k(y_i) \log \Pr(y_i = k) = \sum_i \sum_{k=1}^{m} I_k(y_i) \log \frac{e^{x_i^\prime \beta_k}}{1 + \sum_{k=2}^{m} e^{x_i^\prime \beta_k}}
\]

this function \( \log L \) is globally concave in \( \beta \) (McFadden, 1974)

### 11.2 Example

The purpose of this example is to illustrate an identification problem in the context of multinomial logit. There are different ways to describe the same choices. In the example we see that we use one set of parameters \((mat)\) to generate the choices but the estimator gives us a different set of parameters back \((coef(est))\). We also see how these two sets of parameters are related.

Let us first create individual explanatory variables, \(x1, x2\).

```
N<-100
sd<-10
ex <- cbind(x1=runif(N), x2=runif(N))
head(ex)
```

```
x1   x2
[1,] 0.2875775 0.5999890 
[2,] 0.7883051 0.3328235 
[3,] 0.4089769 0.4886130 
[4,] 0.8830174 0.9544738 
[5,] 0.9404673 0.4829024 
[6,] 0.0455565 0.8903502 
```

The following matrix determines how individual characteristics translate into preferences for three choices:

```
mat<-rbind(c(400,0),
            c(250,200),
            c(100,300))
mat
```
latent <- (ex %*% t(mat)) + sd * cbind(rnorm(N), rnorm(N), rnorm(N))

head(latent)

[,1]  [,2]  [,3]
 [1,] 107.92694 213.8803 201.6020
 [2,] 317.89089 276.7651 171.1507
 [3,] 161.12385 197.3154 178.0962
 [4,] 349.73154 417.0811 364.1188
 [5,] 366.67073 327.5539 234.5459
 [6,] 17.77232 184.6967 274.9725

max.col(latent)

[1] 2 1 2 2 1 3 2 1 2 1 2 1 1 3 3 1 3 3 1 1 1 1 1 1 2 1 1 2 3 1 2 2 2 3 2 3 2 3
[39] 3 3 3 2 1 1 3 2 2 1 1 2 3 2 1 3 1 3 1 1 1 1 3 3 2 1 2 2 1 1 1 2 1 2 3 2 3 2 3
[77] 2 2 3 3

[ reached getOption("max.print") -- omitted 20 entries ]

choice <- max.col(latent)

library(nnet)
est <- multinom(choice ~ x1 + x2, data.frame(ex))

# weights: 12 (6 variable)
initial value 109.861229
iter 10 value 18.323213
iter 20 value 16.923568
iter 30 value 16.881715
iter 40 value 16.880637
iter 50 value 16.880332
iter 60 value 16.880044
iter 70 value 16.879931
final value 16.879896
converged

est

Call:
multinom(formula = choice ~ x1 + x2, data = data.frame(ex))

Coefficients:
 (Intercept) x1 x2
2 0.9444688 -25.80588 31.72050
3 0.1557040 -58.59718 52.66552
Residual Deviance: 33.75979  
AIC: 45.75979

Note that the estimated coefficients are not the matrix of coefficients `mat` that we employed above. However, they are a projection. We are expecting this:

```
mat
```

```
[,1]  [,2]  
[1,] 400 0 
[2,] 250 200 
[3,] 100 300 
```

but we got that:

```
coef(est)
```

```
(Intercept)   x1     x2
2 0.9444688 -25.80588 31.72050
3 0.1557040  -58.59718 52.66552
```

The estimator normalises the first category to zero

```
mat
```

```
[,1]  [,2]  
[1,] 400 0 
[2,] 250 200 
[3,] 100 300 
```

```
mat - cbind(c(1,1,1)) %*% mat[1,]  
```

```
[,1]  [,2]  
[1,]  0 0 
[2,] -150 200 
[3,] -300 300 
```

and sets the variance to one:

```
(mat - cbind(c(1,1,1)) %*% mat[1,]) * pi / sqrt(6) / 10 
```

```
[,1]  [,2]  
[1,] 0.00000 0.00000 
[2,] -19.23825 25.65100 
[3,] -38.47649 38.47649 
```

To access estimation results we have the usual extractor functions:

```
coef(est)
```
11.3 Bayesian multinomial

```r
modelM <- 'model {
  for (i in 1:length(y)) {
    for (j in 1:3) { # three different choices
      exb[i,j] <- exp(inprod(beta[,j],ex[i,]))
    }
    y[i] ~ dcat(exb[i,1:3])
  }
  for (k in 1:K) {
    beta[k,1] <- 0 # identifying restriction
  }
  for (j in 2:3) {
    for (k in 1:K) {
      beta[k,j] ~ dnorm(0,.0001)
    }
  }
}
'
dataList <- list(y=choice, ex=cbind(1,ex), K=dim(ex)[2]+1)
bayesM <- run.jags(model=modelM, data=dataList, monitor=c("beta"))
```

```
bayesM$summary$quantiles[-c(1,2,3),c("2.5%","50%","97.5%")]
```

<table>
<thead>
<tr>
<th></th>
<th>2.5%</th>
<th>50%</th>
<th>97.5%</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1,2]</td>
<td>-2.93526</td>
<td>1.05255000</td>
<td>6.407701</td>
</tr>
<tr>
<td>beta[3,2]</td>
<td>20.81504</td>
<td>37.89850000</td>
<td>67.242305</td>
</tr>
<tr>
<td>beta[1,3]</td>
<td>-5.02241</td>
<td>0.07058615</td>
<td>6.171032</td>
</tr>
</tbody>
</table>
Consider the data set ModeChoice from Ecdat.

- Which variables could explain the transport mode?
- Estimate your model.

## 12 Ordered probit

### 12.1 Model

We observe whether latent variables $x'\beta$ are in an interval

\[
\begin{align*}
Pr(y_i = 1) &= Pr(\kappa_0 < x_i'\beta + u \leq \kappa_1) \\
Pr(y_i = 2) &= Pr(\kappa_1 < x_i'\beta + u \leq \kappa_2) \\
Pr(y_i = 3) &= Pr(\kappa_2 < x_i'\beta + u \leq \kappa_3) \\
\vdots \\
Pr(y_i = 1) &= Pr(\kappa_0 - x_i'\beta < u \leq \kappa_1 - x_i'\beta) \\
Pr(y_i = 2) &= Pr(\kappa_1 - x_i'\beta < u \leq \kappa_2 - x_i'\beta) \\
Pr(y_i = 3) &= Pr(\kappa_2 - x_i'\beta < u \leq \kappa_3 - x_i'\beta) \\
\vdots
\end{align*}
\]

or (solving for $u$)
The $u$ can follow any (standard) distribution (logistic, normal, ...)

```r
plot(dnorm,-2,2,xaxt="n",xlab=NA)
kappas<-c(-1.2,.2,1)
for(i in 1:length(kappas)) {x<-kappas[i];lines(c(x,x),c(0,dnorm(x)))}
axis(1,kappas,sapply(1:length(kappas),function(d) sprintf("$\kappa_%d - x_1'\beta$",d)))
```

Marginal effects:

```r
plot(dnorm,-2,2,xaxt="n",xlab=NA)
kappas<-c(-1.2,.2,1)
for(i in 1:length(kappas)) {
x<-kappas[i];lines(c(x,x),c(0,dnorm(x)))
y<-kappas[i]-.15;lines(c(y,y),c(0,dnorm(y)))
arrows(x,.05,y,.05,length=.05)
}
axis(1,kappas,sapply(1:length(kappas),function(d) sprintf("$\kappa_%d - x_1'\beta$",d)))
```
The maximum likelihood problem

\[
\begin{align*}
Pr(y_i = 1) & = Pr(\kappa_0 - x_i' \beta < u \leq \kappa_1 - x_i' \beta) \\
Pr(y_i = 2) & = Pr(\kappa_1 - x_i' \beta < u \leq \kappa_2 - x_i' \beta) \\
Pr(y_i = 3) & = Pr(\kappa_2 - x_i' \beta < u \leq \kappa_3 - x_i' \beta) \\
& \vdots \\
\log L & = \sum_i \sum_{k=1}^m I_k(y_i) \log Pr(y_i = k)
\end{align*}
\]

with \( I_k(y_i) = \begin{cases} 
1 & \text{if } y_i = i \\
0 & \text{otherwise}
\end{cases} \)

### 12.2 Illustration — the Fair data

As an illustration, let us look at a dataset on extramarital affairs, collected by Ray Fair. Two variables from the dataset are

- \( ym \) number of years married
- \( rat e \) self rating of marriage (unhappy=1…5=happy)
Does the rating of marriage change over time? A naïve approach would be to use OLS and to explain rate as a linear function of ym.

```r
library(MASS)
library(Ecdat)
data(Fair)
lm(rate ~ ym, data = Fair)
```

```
Call:
lm(formula = rate ~ ym, data = Fair)

Coefficients:
(Intercept) ym
4.32546 -0.04814
```

This approach would assume that all ratings are equidistant. More appropriate is, perhaps, an ordered logistic model...

```r
(estL <- polr(factor(rate) ~ ym, data = Fair))
```

```
Call:
polr(formula = factor(rate) ~ ym, data = Fair)

Coefficients:
 ym
-0.08371391

Intercepts:
1|2 2|3 3|4 4|5
-4.3786529 -2.5996956 -1.6207810 -0.2043441

Residual Deviance: 1597.27
AIC: 1607.27
```

...or an ordered probit:

```r
(estP <- polr(factor(rate) ~ ym, data = Fair, method = "probit"))
```

```
Call:
polr(formula = factor(rate) ~ ym, data = Fair, method = "probit")

Coefficients:
 ym
-0.05110974

Intercepts:
1|2 2|3 3|4 4|5
-2.427247 -1.552900 -0.990142 -0.119791

Residual Deviance: 1594.99
AIC: 1604.99
```
The following graph illustrates the estimated thresholds $\kappa_i$:

```
probFig <- function (est, main) {
    plot(function(x) {x * est$coef}),0,55,ylab="$\kappa$",xlab="years of marriage",main=main)
    for (a in est$zeta) {
        abline(h=a)
        lab=names(est$zeta)[which(est$zeta==a)]
        text(1,a,labels=lab,adj=0)
    }
}
probFig(estL,main="ordered logistic")
probFig(estP,main="ordered probit")
```
• Dependent variable $y[i]$
• Latent variable $t[i]$
• Independent variable $x[i]$
• Parameters $\beta$, $\kappa[j]$

**JAGS notation for intervals**

$$y[i] \sim \text{dinterval}(t[i], \kappa)$$

where

- $t[i]$ realisation of latent variable
- $y[i]$ observable rating
- $\kappa$ thresholds

If $Y \sim \text{dinterval}(t, \kappa)$ then

$$Y = 0 \quad \text{if } t \leq \kappa[1]$$
$$Y = m \quad \text{if } \kappa[m] < t \leq \kappa[m + 1] \quad \text{for } 1 \leq m < M$$
$$Y = M \quad \text{if } \kappa[M] < t$$

Note: We have to give JAGS possible initial values:
dataList <- list(y = Fair$rate - 1, x = Fair$ym, K = max(Fair$rate) - 1)
initList <- with(dataList, list(t = y + 1/2, kappa0 = 1:K))

model0 <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dinterval(t[i], kappa)
    t[i] ~ dnorm(beta*x[i], 1)
  }
  for (j in 1:K) {
    kappa0[j] ~ dnorm(0, .0001)
  }
  kappa[1:4] <- sort(kappa0)
  beta ~ dnorm(0, .0001)
}
'
dataList <- list(y = Fair$rate - 1, x = Fair$ym, K = max(Fair$rate) - 1)
initList <- with(dataList, list(t = y + 1/2, kappa0 = 1:K))
bayesO <- run.jags(model = model0, data = dataList, inits = list(initList, initList),
  monitor = c("beta", "kappa"))

bayesO$summary$quantiles[, c("2.5\%", "50\%", "97.5\%")]

Call:
polr(formula = factor(rate) ~ ym, data = Fair, method = "probit")

Coefficients:
  ym
-0.05110974

Intercepts:
1|2 2|3 3|4 4|5
-2.427247 -1.552900 -0.990142 -0.119791

Residual Deviance: 1594.99
AIC: 1604.99

Convergence is not too exciting
JAGS model summary statistics from 20000 samples (chains = 2; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta</td>
<td>-0.061539</td>
<td>-0.04208</td>
<td>-0.0005693</td>
<td>-0.038078</td>
<td>0.016126</td>
<td>-0.047019</td>
<td>0.0057683</td>
</tr>
<tr>
<td>kappa[1]</td>
<td>-2.6603</td>
<td>-2.3101</td>
<td>-1.7527</td>
<td>-2.2751</td>
<td>0.2228</td>
<td>-2.3448</td>
<td>0.03129</td>
</tr>
<tr>
<td>kappa[2]</td>
<td>-1.6862</td>
<td>-1.4532</td>
<td>-0.8756</td>
<td>-1.3938</td>
<td>0.21054</td>
<td>-1.5099</td>
<td>0.066847</td>
</tr>
<tr>
<td>kappa[3]</td>
<td>-1.0938</td>
<td>-0.87123</td>
<td>-0.38771</td>
<td>-0.82444</td>
<td>0.2017</td>
<td>-0.96236</td>
<td>0.077131</td>
</tr>
<tr>
<td>kappa[4]</td>
<td>-0.24793</td>
<td>-0.013495</td>
<td>0.40045</td>
<td>0.031712</td>
<td>0.17487</td>
<td>-0.060974</td>
<td>0.069547</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MC%ofSD</th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta</td>
<td>35.8</td>
<td>8</td>
<td>0.90535</td>
<td>1.2393</td>
</tr>
<tr>
<td>kappa[1]</td>
<td>14</td>
<td>51</td>
<td>0.94751</td>
<td>1.1735</td>
</tr>
<tr>
<td>kappa[2]</td>
<td>31.8</td>
<td>10</td>
<td>0.99001</td>
<td>1.2485</td>
</tr>
<tr>
<td>kappa[3]</td>
<td>38.2</td>
<td>7</td>
<td>0.99305</td>
<td>1.2836</td>
</tr>
<tr>
<td>kappa[4]</td>
<td>39.8</td>
<td>6</td>
<td>0.99375</td>
<td>1.2465</td>
</tr>
</tbody>
</table>

Total time taken: 6.8 seconds

12.3 Exercise

Consider the data set Math1level from Ecdat.

- Which variables could explain the attained math level.
- Estimate your model.

13 Instrumental variables

The problem:

\[ Y = X\beta + \epsilon \quad \text{but} \quad X \not\perp \epsilon \]

Solution, use instrument \( Z \perp \epsilon \)

1st stage: \( X = Z\gamma + \nu \)

\[ \hat{X} = Z\gamma \]

2nd stage: \( Y = \hat{X}\beta + \epsilon \)
```r
set.seed(123)
N <- 100
eps <- rnorm(N)
Z <- rnorm(N)
X <- -eps + Z + 0.5 * rnorm(N)
Y <- X + eps
summary(lm(Y ~ X))
```

Call:
`lm(formula = Y ~ X)`

Residuals:
```
    Min     1Q Median     3Q    Max
-1.59416 -0.42295 -0.00768  0.45972  1.88043
```

Coefficients:
```
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         0.03257    0.06706  0.486  0.628
X                    0.58001    0.04503 12.881  <2e-16 ***
```

---
Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6678 on 98 degrees of freedom
Multiple R-squared: 0.6287, Adjusted R-squared: 0.6249
F-statistic: 165.9 on 1 and 98 DF, p-value: < 2.2e-16

---

**Naïve model: ignore that** $X \not\perp \epsilon$

```r
instNaive.model <- 'model {
  for(i in 1:length(y)) {
    y[i] ~ dnorm(beta[1]+beta[2]*x[i],tau)
  }
  beta[1] ~ dnorm(0,0.0001)
  beta[2] ~ dnorm(0,0.0001)
  tau ~ dgamma(m^2/d^2,m/d^2); m ~ dexp(1); d ~ dexp(1);
}
ini <- genInit(4)
instNaive.jags <- run.jags(model=instNaive.model,
data=list(y=Y,x=X,z=Z),
monitor=c("beta","tau"),inits=ini)
```

```r
summary(instNaive.jags)[,c("Mean","SD","SSeff","psrf")]
```

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>0.03274741</td>
<td>0.06846955</td>
<td>39339</td>
<td>1.0000189</td>
</tr>
<tr>
<td>beta[2]</td>
<td>0.57971640</td>
<td>0.04581343</td>
<td>38130</td>
<td>0.9999869</td>
</tr>
<tr>
<td>tau</td>
<td>2.20708778</td>
<td>0.30908647</td>
<td>12778</td>
<td>1.0001239</td>
</tr>
</tbody>
</table>
```
**Instrument** $Z \mid \epsilon$

```
inst.model<- "model {
  for(i in 1:length(y)) {
    x[i] ~ dnorm(xHat[i], tau[2])          # 1st stage
    xHat[i]<- gamma[1]+gamma[2]*z[i]
    y[i] ~ dnorm(beta[1]+beta[2]*xHat[i], tau[1])  # 2nd stage
  }
  for(k in 1:2) {
    beta[k] ~ dnorm(0,.0001)
    gamma[k]- dnorm(0,.0001)
    tau[k] - dgamma(m[k]^2/d[k]^2,m[k]/d[k]^2); m[k] ~ dexp(1); d[k] ~ dexp(1);
  }
}
"
```

```
inst.jags<- run.jags(model=inst.model, data=list(y=Y, x=X, z=Z),
  monitor=c("beta","tau"), inits=ini)
```

```
summary(inst.jags)[, c("Mean","SD","SSeff","psrf")]
```

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>0.08822246</td>
<td>0.1224222</td>
<td>3465</td>
<td>1.001245</td>
</tr>
<tr>
<td>beta[2]</td>
<td>0.97934413</td>
<td>0.1204707</td>
<td>3588</td>
<td>1.000527</td>
</tr>
<tr>
<td>tau[1]</td>
<td>4.19849540</td>
<td>0.5962644</td>
<td>11327</td>
<td>1.000033</td>
</tr>
<tr>
<td>tau[2]</td>
<td>0.85500640</td>
<td>0.1194926</td>
<td>18651</td>
<td>1.000030</td>
</tr>
</tbody>
</table>

### 13.1 Example: Demand and Supply

![Graphs showing demand and supply curves](image)

How can we estimate the slope of $D$, if $D$ and $S$ are moving simultaneously?
The Demand for Cigarettes

The dataset

- **state**: Factor indicating state.
- **year**: Factor indicating year.
- **cpi**: Consumer price index.
- **population**: State population.
- **packs**: Number of packs per capita.
- **income**: State personal income (total, nominal).
- **tax**: Average state, federal and average local excise taxes for fiscal year.
- **price**: Average price during fiscal year, including sales tax.
- **taxs**: Average excise taxes for fiscal year, including sales tax.

```r
library(AER)
data("CigarettesSW", package = "AER")
head(CigarettesSW)
```

<table>
<thead>
<tr>
<th>state</th>
<th>year</th>
<th>cpi</th>
<th>population</th>
<th>packs</th>
<th>income</th>
<th>tax</th>
<th>price</th>
<th>taxs</th>
</tr>
</thead>
<tbody>
<tr>
<td>AL</td>
<td>1985</td>
<td>1.076</td>
<td>3973000</td>
<td>116.4863</td>
<td>46014968</td>
<td>32.5</td>
<td>102.1817</td>
<td>33.34834</td>
</tr>
<tr>
<td>AR</td>
<td>1985</td>
<td>1.076</td>
<td>2327000</td>
<td>128.5346</td>
<td>26210736</td>
<td>37.0</td>
<td>101.4750</td>
<td>37.00000</td>
</tr>
<tr>
<td>AZ</td>
<td>1985</td>
<td>1.076</td>
<td>3184000</td>
<td>104.5226</td>
<td>43956936</td>
<td>31.0</td>
<td>108.5788</td>
<td>36.17042</td>
</tr>
<tr>
<td>CA</td>
<td>1985</td>
<td>1.076</td>
<td>26444000</td>
<td>112.9635</td>
<td>49466672</td>
<td>31.0</td>
<td>94.26666</td>
<td>31.00000</td>
</tr>
<tr>
<td>CO</td>
<td>1985</td>
<td>1.076</td>
<td>3209000</td>
<td>112.9635</td>
<td>49466672</td>
<td>31.0</td>
<td>94.26666</td>
<td>31.00000</td>
</tr>
<tr>
<td>CT</td>
<td>1985</td>
<td>1.076</td>
<td>3201000</td>
<td>109.2784</td>
<td>60063368</td>
<td>42.0</td>
<td>128.0249</td>
<td>51.48333</td>
</tr>
</tbody>
</table>

We have to construct some variables:

Clean the data:

```r
Cig <- within(subset(CigarettesSW, year=="1995"),{
  rprice <- price/cpi
  rincome <- income/population/cpi
  tdiff <- (taxs - tax)/cpi
  rtax  <- tax/cpi
})
```

\[
\log(\text{packs}) = Y \\
\log(\text{rprice}) = X \\
\text{tdiff} = Z
\]
with(Cig,{plot(packs ~ rprice); plot(tdiff, rprice)})

We are interested in

\[ \log(\text{packs}) = \beta_0 + \beta_1 \log(\text{rprice}) + u \]

However, \( \text{rprice} \) is endogeneous, correlated with \( u \).
We can use \( \text{tdiff} \), the sales tax on cigarettes, as an instrument for \( \log(\text{rprice}) \).

1st stage: \( \log(\text{rprice}) = \gamma_0 + \gamma_1 \text{tdiff} + \nu \)

\[ \log(\text{rprice}) = \gamma_0 + \gamma_1 \text{tdiff} \]

2nd stage: \( \log(\text{packs}) = \beta_0 + \beta_1 \log(\text{rprice}) + \epsilon \)

The naïve approach

```r
est0.model <- 'model {
for(i in 1:length(y)) {
  y[i] ~ dnorm(beta[1]+beta[2]*x[i],tau)
}
for(k in 1:2) {
  beta[k] ~ dnorm(0,.0001)
}
  tau ~ dgamma(m^2/d^2,m/d^2); m ~ dexp(1); d ~ dexp(1);
}
ini <- genInit(4)
est0.jags<-run.jags(model=est0.model,
data=with(Cig,list(y=log(packs),x=log(rprice))),
monitor=c("beta","tau"),inits=ini)
```
JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
<th>SSeff</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>8.2979</td>
<td>10.166</td>
<td>11.923</td>
<td>10.082</td>
<td>0.95021</td>
<td>10.304</td>
<td>0.17318</td>
<td>18.2</td>
<td>3</td>
</tr>
<tr>
<td>beta[2]</td>
<td>-1.5448</td>
<td>-1.177</td>
<td>-0.78779</td>
<td>-1.1595</td>
<td>0.19867</td>
<td>-1.2082</td>
<td>0.036321</td>
<td>18.3</td>
<td>30</td>
</tr>
<tr>
<td>tau</td>
<td>14.916</td>
<td>24.752</td>
<td>36.272</td>
<td>25.131</td>
<td>5.5338</td>
<td>24.523</td>
<td>0.039646</td>
<td>0.7</td>
<td>19483</td>
</tr>
</tbody>
</table>

AC.10  psrf
beta[1] 0.985 1.2297
beta[2] 0.985 1.2248
tau 0.018035 1.001

Total time taken: 1.1 seconds

2 Stage Least Squares (2SLS)
We use the same model as before:

```r
inst.model<-'model {
for(i in 1:length(y)) {
  x[i] ~ dnorm(xHat[i],tau[2]) # 1st stage
  xHat[i]<- gamma[1]+gamma[2]*z[i]
  y[i] ~ dnorm(beta[1]+beta[2]*xHat[i],tau[1]) # 2nd stage
}
for(k in 1:2) {
  beta[k] ~ dnorm(0,.0001)
  gamma[k]~ dnorm(0,.0001)
  tau[k] ~ dgamma(m[k]^2/d[k]^2,m[k]/d[k]^2); m[k] ~ dexp(1); d[k] ~ dexp(1);
}
}
```

cig.data <- with(Cig,list(y=log(packs)-mean(log(packs)),
                           x=log(rprice)-mean(log(rprice)),
                           z=tdiff-mean(tdiff)))
cig.jags <- run.jags(model=inst.model,
                           data=cig.data,
                           monitor=c("beta","sd"),
                           inits=ini)

cig.jags

JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th>Statistic</th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>-0.074083</td>
<td>0.00047877</td>
<td>0.075199</td>
<td>0.00033328</td>
<td>0.037873</td>
<td>-0.0013349</td>
<td>0.00023624</td>
</tr>
</tbody>
</table>
Extending the 2nd stage

Could there be an omitted variable bias in our second stage equation. Perhaps demand is not only affected by price, but also by income?

→ include log(rincome) in the second stage:

```r
inst2.model <- 'model {
  for(i in 1:length(y)) {
    x[i] ~ dnorm(xHat[i],tau[2]) # 1st stage
    xHat[i]<- gamma[1]+gamma[2]*z[i]
    y[i] ~ dnorm(beta[1]+beta[2]*xHat[i]+beta[3]*x2[i],tau[1]) # 2nd stage
  }
  for(k in 1:3) {
    beta[k] ~ dnorm(0,.0001)
    gamma[k] ~ dnorm(0,0.0001)
    tau[k] ~ dgamma(m[k]^2/d[k]^2,m[k]/d[k]^2); m[k] ~ dexp(1); d[k] ~ dexp(1); 
  }
}'
ini <- genInit(4)
cig2.data <- with(Cig,list(y=log(packs)-mean(log(packs)),
                      x=log(rprice)-mean(log(rprice)),
                      z=tdiff-mean(tdiff),
                      x2=log(rincome)))
cig2.jags <- run.jags(model=inst2.model,
                       data=cig2.data,
                       monitor=c("beta","sd"),
                       inits=ini)
```

```
JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>-0.71018</td>
<td>0.60736</td>
<td>2.1043</td>
<td>0.65172</td>
<td>0.73187</td>
<td>0.62976</td>
<td>0.10195</td>
<td>13.9</td>
</tr>
<tr>
<td>beta[2]</td>
<td>-2.1018</td>
<td>-1.0514</td>
<td>-0.15304</td>
<td>-1.0869</td>
<td>0.49951</td>
<td>-0.99786</td>
<td>0.0041852</td>
<td>0.8</td>
</tr>
<tr>
<td>beta[3]</td>
<td>-0.78136</td>
<td>-0.22658</td>
<td>0.26685</td>
<td>-0.24305</td>
<td>0.27274</td>
<td>-0.23149</td>
<td>0.037999</td>
<td>13.9</td>
</tr>
</tbody>
</table>

SSeff AC.10 psrf
beta[1] 52 0.97441 1.0345
beta[2] 14245 0.023295 1.0005
```
Alternative: \textit{ivreg}

```r
est2.iv <- ivreg(log(packs) ~ log(rprice) + log(rincome) | log(rincome) + tdiff, data = Cig)
summary(est2.iv)
```

Call:
```
ivreg(formula = log(packs) ~ log(rprice) + log(rincome) | log(rincome) + tdiff, data = Cig)
```

Residuals:

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-0.611000</td>
<td>-0.086072</td>
<td>0.009423</td>
<td>0.106912</td>
<td>0.393159</td>
</tr>
</tbody>
</table>

Coefficients:

|               | Estimate | Std. Error | t value | Pr(>|t|) |
|---------------|----------|------------|---------|---------|
| (Intercept)   | 9.4307   | 1.3584     | 6.943   | 0.00000000124 *** |
| log(rprice)   | -1.1434  | 0.3595     | -3.181  | 0.00266 ** |
| log(rincome)  | 0.2145   | 0.2686     | 0.799   | 0.42867 |

---

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.1896 on 45 degrees of freedom
Multiple R-Squared: 0.4189, Adjusted R-squared: 0.3931
Wald test: 6.534 on 2 and 45 DF, p-value: 0.003227

Note that the \textit{ivreg} notation includes \textit{log(rincome)} as an instrument for itself. Technically this means, that \textit{log(rincome)} will be perfectly predicted, i.e. not instrumented.

```r
lm(log(rincome) ~ I(log(rincome)) + tdiff, data = Cig)
```

Call:
```
lm(formula = log(rincome) ~ I(log(rincome)) + tdiff, data = Cig)
```

Coefficients:

|                | Estimate | Std. Error | t value | Pr(>|t|) |
|----------------|----------|------------|---------|---------|
| (Intercept)    | 0.000e+00 | 1.000e+00  | 0.05332 | 0.95692 |
| I(log(rincome))| 1.000e+00 | 3.043e-19  | 0.05332 | 0.95692 |
| tdiff          | 3.043e-19 | 0.000e+00  | 0.05332 | 0.95692 |

13.2 Discrete endogeneous variables
```r
set.seed(123)
N <- 1000
eps <- rnorm(N)
nu <- rnorm(N)
Z <- rnorm(N)
X <- as.numeric(Z + eps + nu > 0)
Y <- X + eps

summary(lm(Y ~ X))

Call:
  lm(formula = Y ~ X)

Residuals:
   Min     1Q   Median     3Q    Max
-2.45873 -0.59440  0.00077  0.56316  2.74402

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.46093   0.03877  -11.89  <2e-16 ***
   X         1.95795   0.05494   35.64  <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.8687 on 998 degrees of freedom
Multiple R-squared: 0.56, Adjusted R-squared: 0.5595
F-statistic: 1270 on 1 and 998 DF,  p-value: < 2.2e-16

Bayesian inference

discrete.model <- 'model {
  for(i in 1:length(y)) {
    x[i] ~ dbern(xHat[i])  # 1st stage
    probit(xHat[i]) <- gamma[1]+gamma[2]*z[i]
    y[i] ~ dnorm(beta[1]+beta[2]*xHat[i],tau[1]) # 2nd stage
  }
  for (k in 1:2) {
    beta[k] ~ dnorm(0,.0001)
    gamma[k] ~ dnorm(0,.0001)
  }
  tau ~ dgamma(m^2/d^2,m/d^2); m ~ dexp(1); d ~ dexp(1);
}

disc.jags <- run.jags(model=discrete.model,data=list(y=Y,x=X,z=Z),
                      monitor=c("beta"),modules="glm",inits=ini)
```
disc.jags

JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>-0.14957</td>
<td>0.055905</td>
<td>0.26195</td>
<td>0.05475</td>
<td>0.10514</td>
<td>0.060166</td>
<td>0.00057677</td>
<td>0.5</td>
</tr>
<tr>
<td>beta[2]</td>
<td>0.55179</td>
<td>0.91778</td>
<td>1.3055</td>
<td>0.92081</td>
<td>0.19287</td>
<td>0.91839</td>
<td>0.0010584</td>
<td>0.5</td>
</tr>
</tbody>
</table>

SS eff   AC.10    psrf
beta[1]  33230   0.009125 2
beta[2]  33209   0.0080993 0.99997

Total time taken: 2 minutes

2SLS applied to the non-linear case

Without Bayes we could also use the 2SLS:

Caution! We estimate a non-linear process with a linear model.

```r
summary(ivreg(Y ~ X | Z))
```

Call:
```
ivreg(formula = Y ~ X | Z)
```

Residuals:

<table>
<thead>
<tr>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>-2.86937</td>
<td>-0.66214</td>
<td>-0.02271</td>
<td>0.66499</td>
<td>3.26873</td>
</tr>
</tbody>
</table>

Coefficients:

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | 0.0596 | 0.0795 | 0.750 | 0.454 |
| X | 0.9127 | 0.1461 | 6.248 | 6.15e-10 *** |

---

Signif. codes: 0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.014 on 998 degrees of freedom
Multiple R-Squared: 0.4004, Adjusted R-squared: 0.3998
Wald test: 39.04 on 1 and 998 DF, p-value: 6.15e-10

(see Chesher and Rosen, 2015, for a discussion)

Two stage non-linear estimation

(biased standard errors)

```r
step1 <- glm(X ~ Z, family=binomial(link=logit))
Xhat <- plogis(predict(step1))
summary(lm (Y ~ Xhat))
```


13.3 Exercises

Consider the data set `RetSchool` from Ecdat. You want to study the impact of education on wage.

1. Why could education be endogeneous and why could this be problem?

2. Which variables could be used as instruments?

3. Compare a model with and without instruments.

14 Measurement errors

14.1 Single measures, known error

In OLS we assume that $Y$ is measured only with precision $\tau$, however, $X$ is infinitely precise:

$$Y \sim N(\beta_0 + \beta_1 X, \tau_Y)$$

Assume that $X \sim N(\xi, \tau_\eta)$

We are interested in

$$Y \sim N(\beta_0 + \beta_1 \xi, \tau_Y)$$

Pretending that $X$ is an infinitely precise measure for $\xi$ leads to a biased estimate of $\beta_1$. 

Call:

```r
lm(formula = Y ~ Xhat)
```

Residuals:

<table>
<thead>
<tr>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td>-3.3240</td>
<td>-0.9231</td>
<td>-0.0019</td>
<td>0.9277</td>
<td>3.7916</td>
</tr>
</tbody>
</table>

Coefficients:

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | 0.05804 | 0.10057 | 0.577 | 0.564 |
| Xhat | 0.91584 | 0.18448 | 4.964 | 0.00000081 *** |

---

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.294 on 998 degrees of freedom
Multiple R-squared: 0.0241, Adjusted R-squared: 0.02312
F-statistic: 24.65 on 1 and 998 DF, p-value: 0.00000081
If we knew $\xi$, we could do the following:

```r
summary(lm(Y ~ xi))
```

```
Call: lm(formula = Y ~ xi)

Residuals:     Min 1Q Median 3Q Max
-36.48 -13.64 -1.35 10.29 46.62

Coefficients:     Estimate Std. Error t value Pr(>|t|)
(Intercept) -5.0994   2.8280  -1.803  0.0776 .
xi            10.1037   0.5139  19.661  <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```
Residual standard error: 19.98 on 48 degrees of freedom
Multiple R-squared: 0.8895, Adjusted R-squared: 0.8872
F-statistic: 386.6 on 1 and 48 DF, p-value: < 2.2e-16

If we only know $X$, OLS does not work too well:

```r
summary(lm(Y~X))
```

Call:
`lm(formula = Y ~ X)`

Residuals:

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-94.58</td>
<td>-31.51</td>
<td>4.51</td>
<td>42.48</td>
<td>76.88</td>
</tr>
</tbody>
</table>

Coefficients:

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | -8.4834    | 6.2152  | -1.365   | 0.179   |
| X         | 5.8283     | 0.8839  | 6.594    | 0.0000000309 *** |

---

Signif. codes: 0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

Residual standard error: 43.56 on 48 degrees of freedom
Multiple R-squared: 0.4753, Adjusted R-squared: 0.4643
F-statistic: 43.48 on 1 and 48 DF, p-value: 0.00000003089

Possible correction:

\[
\hat{\beta}_1 = \beta_{1,\text{OLS}} \cdot \left(1 + \frac{\tau_x}{\tau_\eta}\right) = \beta_{1,\text{OLS}} \cdot \left(1 + \frac{1}{36} \cdot \frac{1}{25}\right) = \beta_{1,\text{OLS}} \cdot \frac{61}{36}
\]

```r
naive.model <- "model {
  for (i in 1:length(y)) {
  }
  for (i in 1:2) {
    beta[i] ~ dnorm(0, .0001)
    tau[i] ~ dgamma(m[i]^2/d[i]^2,m[i]/d[i]^2); m[i] ~ dexp(1); d[i] ~ dexp(1);
  }
"
}
ini <- genInit(4)

naive.jags <- run.jags(naive.model, data=list(x=X,y=Y),monitor=c("beta"), inits=ini)
```

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
</table>
The following models allows to specify the measurement error as $\tau[1]$

```jags
measure.model <- "model {
  for (i in 1:length(y)) {
    xi[i] ~ dnorm(0, tau[3])
    x[i] ~ dnorm(xi[i], tau[1])
  }
  for (i in 1:2) {
    beta[i] ~ dnorm(0, .0001)
    tau[i] ~ dgamma(m[i]^2/d[i]^2,m[i]/d[i]^2); m[i] ~ dexp(1); d[i] ~ dexp(1);
  }
}
"
```

We replicate the “naïve” model by setting $\tau[1]=100$.

```r
naive2.jags<-run.jags(measure.model,
  data=list(x = X, y = Y, tau=c(100,NA,1/36)),
  monitor=c("beta"),inits=ini)
```

Pretending (as in `naive2.jags`) that there is no measurement error ($\tau[1]=100$) does not help.

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[2]</td>
<td>4.0964</td>
<td>5.8325</td>
<td>7.6587</td>
<td>5.8291</td>
<td>0.046690</td>
<td>5.8060</td>
<td>0.9999</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MC%ofSD</th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>0.000463</td>
<td>1.0000</td>
<td>1.1708</td>
<td></td>
</tr>
<tr>
<td>beta[2]</td>
<td>0.000547</td>
<td>0.9999</td>
<td>1.0342</td>
<td></td>
</tr>
</tbody>
</table>

Here we set the measurement error to the correct value: $\tau[1]=1/25$.

```r
run.jags(measure.model,data=list(x = X, y = Y, tau=c(1/25,NA,1/36)),
  monitor=c("beta"),inits=ini)
```

JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

```
<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
</table>
```

```
AC.10 | psrf
beta[1]| 0.9225  | 0.92962 | 0.9225  |
```

Total time taken: 1.3 seconds
The following example shows that it is also important to set the right prior for the
distribution of $\xi (\tau[3]=1/36)$.
If we misspecify $\xi (\tau[3]=1/10000)$ we get another bias:

```r
run.jags(measure.model, data=list(x=X, y=Y, tau=c(1/25,NA,1/10000)),
          monitor=c("beta"), inits=ini)
```

JAGS model summary statistics from 40000 samples (chains = 4 ; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
<th>SSeff</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>-22.324</td>
<td>-10.37</td>
<td>2.0408</td>
<td>-10.21</td>
<td>6.2505</td>
<td>-10.1</td>
<td>0.39518</td>
<td>6.3</td>
<td>250</td>
</tr>
</tbody>
</table>

AC.10 psrf

| beta[1]| 0.54753 | 1.0646 |
| beta[2]| 0.72566 | 1.0519 |

Total time taken: 1 seconds

### 14.2 Multiple measures

Multiple measures help us to find $\tau_\eta$.

```r
set.seed(123)
N <- 50
xi <- rnorm(N, 0, sd=6)
X1 <- rnorm(N, mean=xi, sd=5)
X2 <- rnorm(N, mean=xi, sd=5)
Y <- rnorm(N, mean=10*xi, sd=20)

measure2.model <- 'model {
  for(i in 1:length(y)) {
    xi[i] ~ dnorm(0,1/36)
    x1[i] ~ dnorm(xi[i],tau[1])
    x2[i] ~ dnorm(xi[i],tau[1])
    y[i] ~ dnorm(beta[1]+beta[2]*xi[i],tau[2])
  }
  for(k in 1:2) {
    beta[k] ~ dnorm(0,0.0001)
    tau[k] ~ dgamma(m[k]^2/d[k]^2,m[k]/d[k]^2); m[k] ~ dexp(1); d[k] ~ dexp(1);
    sd[k] <- 1/sqrt(tau[k])
  }
}
ini <- genInit(4)
measure2.jags<-run.jags(model=measure2.model,
data=list(y=Y,x1=X1,x2=X2),
monitor=c("beta","sd"),inits=ini)
```
JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
<th>SSeff</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>-8.4686</td>
<td>2.3499</td>
<td>10.473</td>
<td>2.2222</td>
<td>4.8426</td>
<td>2.8516</td>
<td>0.49105</td>
<td>10.1</td>
<td>97</td>
</tr>
<tr>
<td>sd[1]</td>
<td>4.5768</td>
<td>5.28</td>
<td>6.0878</td>
<td>5.2982</td>
<td>0.38796</td>
<td>5.2429</td>
<td>0.0031453</td>
<td>0.8</td>
<td>15214</td>
</tr>
<tr>
<td>sd[2]</td>
<td>0.20604</td>
<td>1.0668</td>
<td>11.508</td>
<td>2.3694</td>
<td>3.90982</td>
<td>0.79647</td>
<td>0.30409</td>
<td>7.8</td>
<td>165</td>
</tr>
</tbody>
</table>

AC.10 psrf

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>psrf</td>
<td>0.88184</td>
<td>1.2627</td>
<td>0.032975</td>
<td>1.0028</td>
</tr>
<tr>
<td>beta[2]</td>
<td>0.86929</td>
<td>1.0706</td>
<td>0.90334</td>
<td>1.0313</td>
</tr>
<tr>
<td>sd[2]</td>
<td>0.032975</td>
<td>1.0028</td>
<td>0.90334</td>
<td>1.0313</td>
</tr>
</tbody>
</table>

Total time taken: 1.3 seconds

14.3 Aggregating evidence

Let us have another look at the Crime dataset. Assume that we estimate a regression separately for each county. Here are the first six counties:

```r
xyplot(crmrte ~ prbarr | factor(county),
       data=Crime, subset=as.numeric(factor(county))<7, type=c("p","r"))
```

![Graph of Crime dataset](image)

We use `dplyr` to estimate the regressions for each county. Some renaming of variables is still necessary:
county.reg <- ddply(Crime, .(county), function(x) {
  est <- lm(crmrte ~ prbarr, data=x);
  c(coef(est), diag(vcov(est)))
})
head(county.reg)

head(county.reg)

names(county.reg) <- make.names(names(county.reg), unique=TRUE)
head(county.reg)

county.reg <- rename(county.reg, c("X.Intercept."="Intercept",
                           "X.Intercept..1"="varX", "prbarr.1"="varY"))
head(county.reg)

The following graph illustrates the difference in precision of the estimates:

with(county.reg, plot(prbarr ~ Intercept, pch=3))
library(car)
q <- apply(county.reg, 1, function(x) ellipse(x[2:3], diag(x[4:5]), radius=.2,
                                           center.pch=0, col="black", lwd=1))
Besides: The purpose of this exercise is not necessarily economically meaningful. It is only supposed to illustrate what to do with noisy observations.

So, let us assume that all we have is the 90 observations for the different countries plus their precision.

Can we simply pretend that each observation is infinitely precise?

```r
naiveMeta.model <- "model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(beta[1]+beta[2]*x[i],tau)
  }
  for (i in 1:2) {
    beta[i] ~ dnorm(0,.0001)
  }
  tau ~ dgamma(m^2/d^2,m/d^2); m ~ dexp(1); d ~ dexp(1);
}"
meta.data <- with(county.reg,list(y=prbarr,x=Intercept,
    tauX=1/varX,tauY=1/varY,meanXi=mean(Intercept),tauXi=1/var(Intercept)))
metaNaive.jags<-run.jags(naiveMeta.model,
    data=meta.data,monitor="beta",inits=ini)
```

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>0.0588629</td>
<td>0.0774296</td>
<td>0.0964035</td>
<td>0.07741952</td>
<td>0.009567237</td>
<td>0.07779828</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>MCerr</th>
<th>MC%ofSD</th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>0.0001005751</td>
<td>1.1</td>
<td>9049</td>
<td>-0.0000987161</td>
<td>1.000101</td>
</tr>
<tr>
<td>beta[2]</td>
<td>0.0019988290</td>
<td>1.0</td>
<td>9206</td>
<td>-0.0003562718</td>
<td>1.000205</td>
</tr>
</tbody>
</table>
To illustrate the precision of the naïve estimate, we sample from the estimated coefficients and draw a regression line for each sample.

```r
set.seed(123)
with(county.reg, plot(prbarr ~ Intercept))
metaNaive.df <- data.frame(as.mcmc(metaNaive.jags))
q <- apply(metaNaive.df[sample(1:nrow(metaNaive.df), 100), ,], 1,
function(x) abline(coef=x, col=paste(palette()[1], "22", sep="")))
```
In the following model we describe the precision of each observation explicitly:

```r
meta.model <- "model {
  for (i in 1:length(y)) {
    xi[i] ~ dnorm(meanXi,tauXi)
    x[i] ~ dnorm(xi[i],tauX[i])
    y[i] ~ dnorm(beta[1]+beta[2]*xi[i], tauY[i])
  }
  for (i in 1:2) {
    beta[i] ~ dnorm(0, .0001)
  }
}
"
meta.jags <- run.jags(meta.model,
data=meta.data,monitor=c("beta"),inits=ini)
```

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>0.00644267</td>
<td>0.009803535</td>
<td>0.0136691</td>
<td>0.009875959</td>
<td>0.001839607</td>
</tr>
<tr>
<td>beta[2]</td>
<td>-1.43539000</td>
<td>-1.21847500</td>
<td>-1.0192800</td>
<td>-1.221181582</td>
<td>0.106076059</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>2.1</td>
<td>0.0003809642</td>
<td>2.1</td>
<td>2332</td>
<td>0.3036751</td>
</tr>
<tr>
<td>beta[2]</td>
<td>1.9</td>
<td>0.00197773858</td>
<td>1.9</td>
<td>2877</td>
<td>0.2537141</td>
</tr>
</tbody>
</table>

Here are the two regression lines:

```r
with(county.reg,plot(prbarr ~ Intercept))
abline(coef=summary(metaNaive.jags)[,"Median"],lty=3,col=1)
abline(coef=summary(meta.jags)[,"Median"],lty=2,col=2)
legend("topright",c("naïve","meta"),col=1:2,lty=3:2)
```
And here is an illustration of the precision of both models:

```r
meta_df <- data.frame(as.mcmc(meta.jags))
with(county.reg, plot(prbarr ~ Intercept))
legend("topright", c("naïve", "meta"), col=1:2, lty=1)
q <- apply(metaNaive.df[sample(1:nrow(metaNaive.df), 100),], 1,
  function(x) abline(coef=x, col=paste(palette()[1], "22", sep="")))
q <- apply(meta.df[sample(1:nrow(meta.df), 100),], 1,
  function(x) abline(coef=x, col=paste(palette()[2], "22", sep="")))
```
15 Selection

15.1 Interval regression

Assume the testscore data was censored:

```r
library(Ecdat)
data(Caschool)

testCens <- with(Caschool, ifelse(testscr > 680, 680, testscr))

isCens <- testCens >= 680

xyplot(testscr + testCens ~ str, data = Caschool)
```
The model with the uncensored data:

```R
lm(testscr ~ str, data = Caschool)
```

Call:

```R
lm(formula = testscr ~ str, data = Caschool)
```

Coefficients:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>698.93</td>
<td></td>
</tr>
<tr>
<td>str</td>
<td>-2.28</td>
<td></td>
</tr>
</tbody>
</table>

Two naïve models:

```R
lm(testCens ~ str, data = Caschool)
```

Call:

```R
lm(formula = testCens ~ str, data = Caschool)
```

Coefficients:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>(Intercept)</td>
<td>689.193</td>
<td></td>
</tr>
<tr>
<td>str</td>
<td>-1.826</td>
<td></td>
</tr>
</tbody>
</table>

```R
lm(testCens ~ str, data = Caschool, subset = !isCens)
```

Call:

```R
lm(formula = testCens ~ str, data = Caschool, subset = !isCens)
```
Coefficients:
(Intercept) str
   669.3012   -0.9384

Interval regression:

testMax<-testCens
testMax[isCens]<-NA

library(survival)
survreg(Surv(testCens,testMax,type="interval2") ~ str, dist='gaussian',
       data=Caschool)

Call:
survreg(formula = Surv(testCens, testMax, type = "interval2") ~ str, data = Caschool, dist = "gaussian")

Coefficients:
(Intercept) str
   696.767043  -2.171933

Scale= 18.51073

Loglik(model)= -1704.1  Loglik(intercept only)= -1714
Chisq= 19.8 on 1 degrees of freedom, p= 0.0000086
n= 420

15.2 Bayesian censored model

We need (JAGS) notation for interval-censored data:

\[
Y \sim \text{dinterval}(t, c)
\]

\[
Y = \begin{cases} 
0 & \text{if } t \leq c[1] \\
 m & \text{if } c[m] < t \leq c[m + 1] \quad \text{for } 1 \leq m < M \\
 M & \text{if } c[M] < t 
\end{cases}
\]

Here our data is censored from above, i.e.

\[
Y = \begin{cases} 
0 & \text{if } t \leq t_{\text{max}} = c[1] \\
1 & \text{if } c[1] = t_{\text{max}} < t 
\end{cases}
\]

<table>
<thead>
<tr>
<th>latent (t) ((Y))</th>
<th>observed (c)</th>
<th>(Y) (isCens)</th>
</tr>
</thead>
<tbody>
<tr>
<td>not censored</td>
<td>testCens</td>
<td>testCens</td>
</tr>
<tr>
<td>censored</td>
<td>NA</td>
<td>testCens</td>
</tr>
</tbody>
</table>
One complication with the censored model is that the censored observations are unknown, so JAGS will fit random values. Unless we help JAGS a little, the initial values will be inconsistent, i.e. JAGS will randomise values for \( y \) which are not in the right interval. We must avoid situations like the following:

- \( y[1]=800, \ c[1]=600, \ isCens[1]=0 \)
  
  Problem: according to \( isCens[1] \) we have that \( y[1]<c[1] \) with certainty.

- \( y[1]=400, \ c[1]=600, \ isCens[1]=1 \)
  
  Problem: according to \( isCens[1] \) we have that \( y[1]>c[1] \) with certainty.

Otherwise R would throw the following error:

**Observed node inconsistent with unobserved parents at initialization.**

Try setting appropriate initial values.

Solution: We set the initial \( y \) to “safe” values.

For the unobserved (censored) nodes, \( y=testCens+99 \).

For the observed (uncensored) nodes, we can not overwrite the observed nodes. Hence the init value is \( y=NA \).

```r
y <- ifelse(isCens,NA,testCens)
yInit <- ifelse(isCens,testCens+99,NA)  #<- must resolve initial uncertainty about censored y
dataList<-list(y=y,c=testCens,x=Caschool$str,isCens=as.numeric(isCens))
ini <- genInit(4,function(i)
  list(beta0=rnorm(1,0,.001),beta1=rnorm(1,0,.001),y=yInit))
```

dataList<-
initJags<-list()
initJags[[1]]<-list(.RNG.seed=1,.RNG.name="base::Mersenne-Twister")
initJags[[2]]<-list(.RNG.seed=2,.RNG.name="base::Super-Duper")
initJags[[3]]<-list(.RNG.seed=3,.RNG.name="base::Wichmann-Hill")
initJags[[4]]<-list(.RNG.seed=4,.RNG.name="base::Marsaglia-Multicarry")

```r
genInit <- function(nChains,fn=NULL) {
  x<-list()
  for (i in 1:nChains) {
    x[[i]]<-initJags[[i]]
    if(!is.null(fn)) {
      vals<-fn(i)
      lapply(1:length(vals),function(j)
        x[[i]][[names(vals)[j]]]<-vals[[j]])
    }
  }
  x
}
```
### Heckman correction

What, if selection is determined by a different, but correlated process?

```r
set.seed(123)
N <- 100
```
x <- runif(N)
u <- rnorm(N)
u2 <- u + .1*rnorm(N)
cor(u, u2)

[1] 0.9952951

y <- 2+2*x+u
S <- 4*x+u2<3
df <- within(data.frame(list(y=y, x=x, S=S)), y[y[S]<0] = 0)

plot(y ~ x, col=S+1, pch=2*S+1)
abline(lm(y ~ x, data=subset(df, y!=0)), col=2, lty=2)
abline(lm(y ~ x), col=1)

summary(lm(y ~ x, data=subset(df, y!=0)))

Call:
  lm(formula = y ~ x, data = subset(df, y != 0))

Residuals:
  Min     1Q Median     3Q    Max
-2.1329 -0.5530  0.0135  0.5771  1.5563

Coefficients:
   Estimate Std. Error t value Pr(>|t|)
(Intercept)  2.03687   0.30385   6.704 0.000357 **
x           2.03242   0.02382  85.773 < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
We want to explain

\[ Y = X\beta + u \]

We observe \( Y \) only when \( S = 1 \):

\[ \Pr(S = 1|Z) = \Phi(Z\gamma) \]

Hence, we are interested in

\[ \mathbb{E}[Y|X, S = 1] = X\beta + \mathbb{E}[u|X, S = 1] \]

\[ \mathbb{E}[Y|X, S = 1] = X\beta + \rho\sigma_u\lambda(Z\gamma) \]

where

\[ \lambda(Z\gamma) = \frac{\Phi(Z\gamma)}{\Phi(Z\gamma)} \quad \text{(inverse Mills ratio)} \]

This model can be estimated with ML:
library(sampleSelection)

summary(heckit(selection = S ~ x, outcome = y ~ x, data = df))

Tobit 2 model (sample selection model)
2-step Heckman / heckit estimation
100 observations (26 censored and 74 observed)
7 free parameters (df = 94)
Probit selection equation:

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | 3.0335 | 0.5356 | 5.664 | 0.000000161 *** |
| x | -4.0255 | 0.7795 | -5.164 | 0.000001345 *** |

Outcome equation:

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| (Intercept) | 2.0274 | 0.2683 | 7.557 | 2.69e-11 *** |
| x | 1.7366 | 1.0678 | 1.626 | 0.107 |

Multiple R-Squared: 0.0545, Adjusted R-Squared: 0.0278

Error terms:

| Estimate | Std. Error | t value | Pr(>|t|) |
|----------|------------|---------|----------|
| invMillsRatio | -0.8932 | 0.7570 | -1.18 | 0.241 |
| sigma | 0.9110 | NA | NA |
| rho | -0.9804 | NA | NA |

15.4 Bayesian Heckman correction

s.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(ifelse(S[i], Xbeta[i] + rhoSigma*lambda[i], 0),
      ifelse(S[i], tau, 0.0001))
    S[i] ~ dbern(p[i])
    p[i] <- phi(Zgamma[i])
    Zgamma[i] <- gamma[1] + gamma[2]*x[i]
    lambda[i] <- dnorm(Zgamma[i], 0,1)/phi(Zgamma[i])
  }
  for (i in 1:2) {
    beta[i] ~ dnorm(0, 0.0001)
    gamma[i] ~ dnorm(0, 0.0001)
  }
  rhoSigma ~ dnorm(0, 0.0001)
  tau ~ dgamma(m^2/d^2, m/d^2)
  m ~ dgamma(1,1)
  d ~ dgamma(1,1)
}
'

data <- with(df, list(y = y, x = x, S = as.numeric(S)))
run.jags(s.model, data = data, monitor = c("beta", "gamma", "rhoSigma"), inits = genInit(4))
JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>1.5597</td>
<td>2.0398</td>
<td>2.5262</td>
<td>2.0415</td>
<td>0.24686</td>
<td>2.0216</td>
<td>0.0068395</td>
<td>2.8</td>
</tr>
<tr>
<td>beta[2]</td>
<td>-0.57032</td>
<td>1.829</td>
<td>4.4669</td>
<td>1.8987</td>
<td>1.2834</td>
<td>1.7863</td>
<td>0.053792</td>
<td>4.2</td>
</tr>
<tr>
<td>gamma[1]</td>
<td>1.9511</td>
<td>2.886</td>
<td>3.9597</td>
<td>2.9079</td>
<td>0.52496</td>
<td>2.8471</td>
<td>0.019446</td>
<td>3.7</td>
</tr>
<tr>
<td>rhoSigma</td>
<td>-3.4739</td>
<td>-0.99067</td>
<td>0.99711</td>
<td>-1.1132</td>
<td>0.87304</td>
<td>1.1601</td>
<td>0.048647</td>
<td>4.2</td>
</tr>
</tbody>
</table>

SSeff     AC.10     psrf
beta[1]   1303   0.48573  1.0019
beta[2]   569   0.75123  1.0123
gamma[1]  729   0.68522  1.0032
gamma[2]  741   0.6909  1.0022
rhoSigma  569   0.74052  1.0186

Total time taken: 8.8 seconds

15.5 Exercise

1. Consider the data set Workinghours from Ecdat.
   - Which variables could explain the labour supply of the wife?
   - In which way could the labour supply be censored?
   - Estimate your model.

2. Consider the data set Mroz87 from sampleSelection.
   - Explain wage as a function of experience and education.
     Assume that selection into the labour force is determined by age, family income and education.

16 More on initialisation

We use different random number generators to make sure that each chain will take a different turn.

```
str(initJags)
```

List of 4
$ :List of 2
  ..$ .RNG.seed: num 1
  ..$ .RNG.name: chr "base::Mersenne-Twister"
$ :List of 2
  ..$ .RNG.seed: num 2
The `genInit()` function helps adding more variables to the init.

```r
genInit <- function(nChains, fn=NULL) {
  x<-list()
  for (i in 1:nChains) {
    x[[i]]<-initJags[[i]]
    if(!is.null(fn)) {
      vals<-fn(i)
      lapply(1:length(vals),function(j)
        x[[i]][[names(vals)[j]]] <<- vals[[j]])
    }
  }
  x
}
```

now `genInit()` can be called like this:

```r
ini <- genInit(4, function(i) list(beta0=rnorm(1,0,0.0001),
                                betal=rnorm(1,0,0.0001)))
run.jags(..., inits=ini,...)
```

## 17 Hierarchical Models

### 17.1 Mixed effects

**OLS:**

\[ Y_i = X_i \beta + \epsilon_i \text{ with } \epsilon \sim N(0, \sigma) \]

Mixed effects (with groups k):

\[ Y_{ik} = X_{ik} \beta + Z_{ik} \nu + \epsilon_{ik} \text{ with } \nu \sim N(0, \Sigma), \epsilon \sim N(0, \sigma) \]

### 17.2 Example: Crime in North Carolina

The dataset `Crime` contains information about several years. Perhaps the relationship between `crmrte` and `prbarr` is not the same in each year?
We are estimating the following equation:

\[
\text{crmrte}_{ik} = (\beta_1 + \nu_{1,k}) + (\beta_2 + \nu_{2,k})prbarr + \epsilon_{ik}
\]
or, equivalently,

\[
\text{crmrte}_{ik} \sim N((\beta_1 + \nu_{1,k}) + (\beta_2 + \nu_{2,k})prbarr, \tau_3)
\]

First we use Maximum Likelihood (and the \texttt{lmer} function).

```
summary(lmer(crmrte ~ prbarr + (prbarr+1|year), data=Crime))
```

Linear mixed model fit by REML ['lmerMod']
Formula: crmrte ~ prbarr + (prbarr + 1 | year)
Data: Crime

REML criterion at convergence: -3339

Scaled residuals:

<table>
<thead>
<tr>
<th></th>
<th>Min</th>
<th>1Q</th>
<th>Median</th>
<th>3Q</th>
<th>Max</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>-1.7122</td>
<td>-0.6575</td>
<td>0.1795</td>
<td>0.4460</td>
<td>7.3139</td>
</tr>
</tbody>
</table>

Random effects:

<table>
<thead>
<tr>
<th>Groups</th>
<th>Name</th>
<th>Variance</th>
<th>Std.Dev.</th>
<th>Corr</th>
</tr>
</thead>
<tbody>
<tr>
<td>year</td>
<td>(Intercept)</td>
<td>0.00001702</td>
<td>0.004125</td>
<td></td>
</tr>
<tr>
<td></td>
<td>prbarr</td>
<td>0.00016957</td>
<td>0.013022</td>
<td>-1.00</td>
</tr>
<tr>
<td></td>
<td>Residual</td>
<td>0.00028084</td>
<td>0.016758</td>
<td></td>
</tr>
</tbody>
</table>

Number of obs: 630, groups: year, 7
**17.3 Bayes and mixed effects**

Next we specify the mixed model as a JAGS model (since we are mainly interested in the marginal effect, we de-mean `crmrte` and `prbarr`).

```r
mer.model <- 'model {
for (i in 1:length(y)) {
  y[i] ~ dnorm(mu[i], tau[3])
}
for (k in 1:2) {
  beta[k] ~ dnorm (0,.0001)
  for (j in 1:max(group)) {
    nu[k,j] ~ dnorm(0,tau[k])
  }
}
for (k in 1:3) {
  tau[k] ~ dgamma(.01,.01)
  sd[k] <- sqrt(1/tau[k])
}
}
}'
dataList<-with(Crime, list(y=crmrte-mean(crmrte), x=prbarr-mean(prbarr),
                           group=as.numeric(year)))
mer.jags<-run.jags(model=mer.model, data=dataList, inits=ini,
                   monitor=c("beta","sd"))

plot(mer.jags, vars="beta[2]", plot.type=c("trace","density"))
```


17.4 Robust mixed effects

Looking at the graph we might find that some observations look like outliers. As in section 6 we can use the t-distribution with endogenous degrees of freedom to allow for more robustness. In the model we replace $dnorm$ with $dt$ and add a prior for the degrees of freedom.

We are conservative here and make all random effects follow a t-distribution.

```r
merT.model <- 
  model {
    for (i in 1:length(y)) {
      y[i] ~ dt(mu[i], tau[3], df[3])
    }
  }
```
\[
\text{mu}[i] \leftarrow \beta_1 + \nu_1,\text{group}[i] + (\beta_2 + \nu_2,\text{group}[i]) \times x[i]
\]

for (k in 1:2) {
  \beta[k] \sim \text{dnorm}(0, .0001)
  for (j in 1:max(group)) {
    \nu[k,j] \sim \text{dt}(0, \tau[k], df[k])
  }
}

for (k in 1:3) {
  \tau[k] \sim \text{dgamma}(0.01, .01)
  sd[k] \leftarrow \sqrt{1/\tau[k]}
  df[k] \sim \text{dexp}(1/30)
}

}\n
\text{merT.jags} \leftarrow \text{run.jags}(\text{model} = \text{merT.model}, \text{data} = \text{dataList}, \text{inits} = \text{ini}, \text{monitor} = \text{c(} \text{"beta"}, \text{"sd"}, \text{"df"})\text{)}

\text{summary(merT.jags)}

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>\beta[1]</td>
<td>-0.0493</td>
<td>0.0048</td>
<td>0.0509</td>
<td>0.0037</td>
<td>0.025</td>
</tr>
<tr>
<td>\beta[2]</td>
<td>-0.0961</td>
<td>-0.0401</td>
<td>0.0178</td>
<td>-0.0414</td>
<td>0.031</td>
</tr>
<tr>
<td>\sigma[1]</td>
<td>0.0332</td>
<td>0.0609</td>
<td>0.1111</td>
<td>0.0657</td>
<td>0.022</td>
</tr>
<tr>
<td>\sigma[2]</td>
<td>0.0329</td>
<td>0.0639</td>
<td>0.1197</td>
<td>0.0695</td>
<td>0.027</td>
</tr>
<tr>
<td>\sigma[3]</td>
<td>0.0145</td>
<td>0.0157</td>
<td>0.0169</td>
<td>0.0157</td>
<td>0.001</td>
</tr>
<tr>
<td>\eta[1]</td>
<td>0.4972</td>
<td>26.123</td>
<td>94.289</td>
<td>34.57</td>
<td>30.25</td>
</tr>
<tr>
<td>\eta[2]</td>
<td>0.7022</td>
<td>25.94</td>
<td>97.16</td>
<td>35.09</td>
<td>31.23</td>
</tr>
</tbody>
</table>

\text{Here is the posterior density of the degrees of freedom.}

\text{merT.df} \leftarrow \text{data.frame(as.mcmc(merT.jags))}
\text{xx} \leftarrow \text{melt(merT.df[, grep("^df", names(merT.df))])}
\text{xx} \leftarrow \text{rbind.fill(xx, data.frame(list(\text{variable="prior"}, value=qexp(((1:100)-1/2)/100,1/30)))))}
\text{densityplot(~value, group=variable, data=xx, plot.points=FALSE, auto.key=list(space="top", columns=4), xlim=c(-10,100))}
We see that, in particular, the residual \((df[3])\) needs some robustness.

For \(df[1]\) and \(df[2]\), i.e. for the random effects for intercept and slope, the posterior is very similar to the prior, i.e. the data does not provide a lot of extra information in this respect.

On the other hand, the estimate for \(\beta[2]\) does not seem to change much with the robust model:

```r
mer.df<-data.frame(as.mcmc(mer.jags))
densityplot(~mer$beta.2+merT$beta.2,plot.points=FALSE,
xlab="\(\beta_2\)",auto.key=list(space="top",columns=2,text=c("normal","robust")))
```
17.5 Exercises

Consider the data set LaborSupply from Ecdat.

1. Which variables could explain labor supply?
2. Estimate your model, taking into account a random effect for the intercept?
3. Include a random effect for the slope.

18 Model Comparison

- Several models (at least two).
- Discrete variable selects among the models.
- \( \rightarrow \) posterior probability for each model.

Preliminaries

- F: Null-Hypothesis-Testing: \( \Pr(X|H_0) \).
- B: Model comparison: \( \Pr(H_0|X) \) versus \( \Pr(H_1|X) \) versus \( \Pr(H_2|X) \) …
  - Models can be nested, they need not be nested.
– Models can be of different complexity (automatic penalty).

Are these the only plausible models? (similar to a very strong prior)

18.1 Example 1

• Data:

```r
set.seed(123)
x <- rnorm(5, 5, 1)
```

• \( x \sim N(\mu, 1) \) where \( N \) is the normal distribution.

  We compare three models:
  \[ \mu_1 = 3.8, \mu_2 = 4, \mu_3 = 6. \]

```r
c.model <- 'model {
  for (i in 1:length(x)) {
    x[i] ~ dnorm(mu[m], 1)
  }
  m ~ dcat(modelProb)
}
'  
c.data <- list(x=x, modelProb=c(1, 1, 1), mu=c(3.8, 4, 6))  
c.jags <- run.jags(c.model, c.data, monitor=c("m"))  
with(data.frame(as.mcmc(c.jags)), table(m)/length(m))
```

```
m
1 2 3
0.03415 0.12255 0.84330
```

What is, actually, a vague prior in this context? Can we give more flexibility to the prior for \( m \)?

```r
c.model2 <- 'model {
  for (i in 1:length(x)) {
    x[i] ~ dnorm(mu[m], 1)
  }
  m ~ dcat(mP)
  mP ~ ddirch(modelProb)
}
'  
c.jags <- run.jags(c.model2, c.data, monitor=c("m"))  
with(data.frame(as.mcmc(c.jags)), table(m)/length(m))
```

```
m
1 2 3
0.03390 0.12455 0.84155
```
18.2 Example 2

Should we better use a polynomial or a fractional polynomial model to describe the relation in the following simulated data:

```r
set.seed(123)
N <- 100
x <- runif(N)
y <- 5 - 4.3 * x + 3.1 * x^2 + rnorm(N)
myData <- data.frame(y = y, x = x)
quad <- predict(lm(y ~ x + I(x^2), data = myData))
hyp <- predict(lm(y ~ x + I(1/(x+1)), data = myData))
```

- Model 1: \( y = \beta_0 + \beta_1 x + \beta_2 x^2 + u \)
- Model 2: \( y = \beta_0 + \beta_1 x + \beta_2 \frac{1}{x+1} + u \)
AIC

\[
AIC = -2 \log(L) + 2k
\]

- AIC is a measure of information loss of a model (Akaike, 1973).
- AIC is asymptotically equivalent to leave one out cross-validation (Stone, 1977; Fang, Yixin, 2011).

\[
\text{extractAIC(lm}(y \sim x + I(1/(x+1)), \text{data}=mData))
\]
[1] 3.000000 -1.199884

\[
\text{extractAIC(lm}(y \sim x + I(x^2), \text{data}=mData))
\]
[1] 3.000000 -2.757703

DIC  Deviance information criterion:

\[
\begin{align*}
\text{Deviance} & \quad D(\theta) = -2 \log(P(X|\theta)) + C \\
\bar{D} & \quad = E[D(\theta)], \quad \bar{\theta} = E[\theta] \\
\text{eff. # parameters} & \quad p_D = \bar{D} - D(\bar{\theta}) \quad (\text{Spiegelhalter et al., 2002}) \\
\text{DIC} & \quad = D(\bar{\theta}) + 2p_D
\end{align*}
\]
### 18.3 Model 1

```r
modell.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(inprod(beta,X[i,]),tau)
  }
  for (k in 1:K) {
    beta[k] ~ dnorm (0, .0001)
  }
  tau ~ dgamma(m^2/d^2,m/d^2); m ~ dexp(1); d ~ dexp(1);
}
'
modell.data <- with(mData, list(y=y, K=4, X=cbind(1,x,x^2,1/(x+1))))
modell.data <- within(modell.data, {X<-sweep(X,2,apply(X,2,mean));X[,1]<-1})
ini<-genInit(4)
modell.jags <- run.jags(model=modell.model, data=within(modell.data,beta<-c(NA,NA,0,NA)), inits=ini, monitor=c("beta", "tau", "dic", "popt"))
```

**JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):**

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>3.6282</td>
<td>3.8227</td>
<td>4.0158</td>
<td>3.822</td>
<td>0.098466</td>
<td>3.8199</td>
<td>0.00049776</td>
<td>0.5</td>
</tr>
<tr>
<td>beta[2]</td>
<td>-0.058831</td>
<td>3.9502</td>
<td>7.9823</td>
<td>3.9875</td>
<td>2.0602</td>
<td>3.9981</td>
<td>0.085904</td>
<td>4.2</td>
</tr>
<tr>
<td>beta[3]</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>tau</td>
<td>0.7631</td>
<td>1.0402</td>
<td>1.343</td>
<td>1.0472</td>
<td>0.149</td>
<td>1.0325</td>
<td>0.0011955</td>
<td>0.8</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>SSeff</th>
<th>AC.10</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>39132</td>
<td>-0.0033029</td>
<td>0.99999</td>
</tr>
<tr>
<td>beta[2]</td>
<td>575</td>
<td>0.74862</td>
<td>1.0046</td>
</tr>
<tr>
<td>beta[3]</td>
<td>--</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>beta[4]</td>
<td>589</td>
<td>0.74929</td>
<td>1.0045</td>
</tr>
<tr>
<td>tau</td>
<td>15534</td>
<td>0.021715</td>
<td>1</td>
</tr>
</tbody>
</table>

**Model fit assessment:**

- DIC = 295.7388
- PED = 299.9781

Estimated effective number of parameters: pD = 4.10993, pOpt = 8.34924

Total time taken: 3.3 seconds

### 18.4 Model 2

```r
model2.jags <- run.jags(model=modell.model, data=within(modell.data,beta<-c(NA,NA,0)), inits=ini, monitor=c("beta", "tau", "dic", "popt"))
```
JAGS model summary statistics from 40000 samples (chains = 4; adapt+burnin = 5000):

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
<th>MC%ofSD</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>3.6304</td>
<td>3.822</td>
<td>4.0128</td>
<td>3.8225</td>
<td>0.097745</td>
<td>3.8181</td>
<td>0.00049109</td>
<td>0.5</td>
</tr>
<tr>
<td>beta[2]</td>
<td>-8.1456</td>
<td>-5.3772</td>
<td>-2.5942</td>
<td>-5.3861</td>
<td>1.4185</td>
<td>-5.3932</td>
<td>0.040147</td>
<td>2.8</td>
</tr>
<tr>
<td>beta[4]</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>--</td>
<td>--</td>
</tr>
<tr>
<td>tau</td>
<td>0.78073</td>
<td>1.0592</td>
<td>1.3684</td>
<td>1.0647</td>
<td>0.15076</td>
<td>1.0485</td>
<td>0.0011949</td>
<td>0.8</td>
</tr>
</tbody>
</table>

SSeff   AC.10  psrf
beta[1] 39616  0.00046172 | 1
beta[2] 1248   0.53816  1.004
beta[3] 1238   0.53725  1.0039
beta[4]  --     --     --
tau     15919  0.024424  1.0001

Model fit assessment:
DIC = 294.1368
PED = 298.4162
Estimated effective number of parameters: pD = 4.07012, pOpt = 8.34955

Total time taken: 3.3 seconds

18.5 A joint model

Different from the models so far we use parameters (priBetaMean, priBetaTau, priTM, priTD) to describe priors. These priors are defined as part of the data argument to run.jags.

We also restrict the matrix beta to describe the different models.

```r
modelSel0.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm((yy[i,mod]),tau[mod])
    for (m in 1:2) {
      yy[i,m] <- inprod(beta[,m],X[i,])
    }
  }
  for (m in 1:2) {
    for (k in 1:K) {
      beta[k,m] ~ dnorm(priBetaMean[k,m],priBetaTau[k,m])
    }
    tau[m] ~ dgamma(priTauAlpha[m],priTauBeta[m])
  }
  mod ~ dcat(modelProb)
}
modelSel0.data<within(model1.data, {
  beta <- matrix(NA,K,2); beta[3,1]<-0; beta[4,2]<-0; 
  priBetaMean <- matrix(0,K,2); priBetaTau <- matrix(.0001,K,2) 
  priTauAlpha <- c(.01,.01); priTauBeta <- c(.01,.01)
})
```
Convergence in model comparison  Unfortunately, in the above specification the model (even if we guide `partTauAlpha` and `priTauBeta`) does not converge too well:

```r
modelSel0.jags <- run.jags(model=modelSel0.model,
data=modelSel0.data, inits=ini, monitor=c("beta", "tau", "mod"))
```

```r
summary(modelSel0.jags)[,c("Mean","SD","SSeff","psrf")]
```

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1,1]</td>
<td>3.8217968</td>
<td>0.09815767</td>
<td>40186</td>
<td>1.0000236</td>
</tr>
<tr>
<td>beta[2,1]</td>
<td>3.9348496</td>
<td>1.98008024</td>
<td>619</td>
<td>1.0020974</td>
</tr>
<tr>
<td>beta[3,1]</td>
<td>0.0000000</td>
<td>0.00000000</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>beta[4,1]</td>
<td>11.0058461</td>
<td>4.12504235</td>
<td>628</td>
<td>1.0021217</td>
</tr>
<tr>
<td>beta[1,2]</td>
<td>-0.3627482</td>
<td>100.25173235</td>
<td>40260</td>
<td>1.0002285</td>
</tr>
<tr>
<td>beta[2,2]</td>
<td>0.4437219</td>
<td>99.88351859</td>
<td>41634</td>
<td>1.0002128</td>
</tr>
<tr>
<td>beta[3,2]</td>
<td>-0.1120987</td>
<td>99.46382220</td>
<td>40591</td>
<td>0.9999999</td>
</tr>
<tr>
<td>beta[4,2]</td>
<td>0.0000000</td>
<td>0.00000000</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>tau[1]</td>
<td>1.0417498</td>
<td>0.14984201</td>
<td>36746</td>
<td>1.0000176</td>
</tr>
<tr>
<td>tau[2]</td>
<td>0.9247774</td>
<td>9.62548329</td>
<td>40000</td>
<td>1.0045433</td>
</tr>
<tr>
<td>mod</td>
<td>1.0000000</td>
<td>0.00000000</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

(if `partTauAlpha` and `priTauBeta` are left to `NA`, then simulations crash with Error in node `priTauAlpha[2]`. Slicer stuck at value with infinite density).

The main reason for the lack of convergence is: chains do not mix:

```r
plot(as.mcmc(modelSel0.jags)[,"mod"])
```
What happens if the chain selects one model:

- coefficients for this model adjust.
  - likelihood for this model is good.

- coefficients of the other model still follow prior distribution.
  - likelihood of the other model is bad.

→ the sampler will almost never switch (model selection is correlated).

Convergence is very slow, we have to help the sampler.

→ Pseudopriors (Carlin, Chib, 1995).

- When model is selected: use vague priors (as before).
- When model is not selected: use pseudopriors (posteriors from previous estimation).

```r
modelSel.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm((yy[i,mod]),tau[mod])
    for (m in 1:2) { yy[i,m] <- inprod(beta[,m],X[i,]) }
  }
  for (m in 1:2) {
    for (k in 1:K) {
    
```
beta[k,m] ~ dnorm (priBetaMean[k,m,mod], priBetaTau[k,m,mod])
}
tau[m] ~ dgamma(priTM[m,mod]^2/priTD[m,mod]^2, priTM[m,mod]/priTD[m,mod]^2)
for (modI in 1:2) {
    priTM[m,modI] ~ dgamma(1,1); priTD[m,modI] ~ dgamma(1,1);
}
mod ~ dcat(modelProb)
}

modelSel.data <- within(modelSel.data,
    beta <- matrix(NA,K,2); beta[3,1]<-0; beta[4,2]<-0;
    priBetaMean <- array(0,c(K,2,2)); priBetaTau <- array(.0001,c(K,2,2))
    priTM <- matrix(NA,2,2); priTD <- matrix(NA,2,2)
    modelProb <- c(1,1))
ini<-genInit(4)

Digression: We can use the above (flexible) model to estimate the previous (specific) models:

model1B.jags <- run.jags(model=modelSel.model, within(modelSel.data,mod<1), inits=ini, monitor=c("beta","tau","mod"))

summary(model1B.jags)[c(1:4,9),c("Mean","SD","SSeff","psrf")]

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1,1]</td>
<td>3.822</td>
<td>0.098</td>
<td>40000</td>
<td>1.000074</td>
</tr>
<tr>
<td>beta[2,1]</td>
<td>3.833</td>
<td>2.011</td>
<td>634</td>
<td>1.004135</td>
</tr>
<tr>
<td>beta[3,1]</td>
<td>0.000</td>
<td>0.000</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>beta[4,1]</td>
<td>10.793</td>
<td>4.189</td>
<td>637</td>
<td>1.004153</td>
</tr>
<tr>
<td>tau[1]</td>
<td>1.047</td>
<td>0.149</td>
<td>14775</td>
<td>1.000149</td>
</tr>
</tbody>
</table>

summary(model1.jags)[c("Mean","SD","SSeff","psrf")]

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1]</td>
<td>3.822</td>
<td>0.098</td>
<td>39132</td>
<td>0.9999892</td>
</tr>
<tr>
<td>beta[3]</td>
<td>0.000</td>
<td>0.000</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>tau</td>
<td>1.047</td>
<td>0.149</td>
<td>15534</td>
<td>0.9999992</td>
</tr>
</tbody>
</table>

model2B.jags <- run.jags(model=modelSel.model, within(modelSel.data,mod<2), inits=ini, monitor=c("beta","tau","mod"))

summary(model2B.jags)[c(5:8,10),c("Mean","SD","SSeff","psrf")]

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1,2]</td>
<td>3.822</td>
<td>0.098</td>
<td>39853</td>
<td>1.000000</td>
</tr>
<tr>
<td>beta[2,2]</td>
<td>-5.304</td>
<td>1.458</td>
<td>1152</td>
<td>1.000958</td>
</tr>
<tr>
<td>beta[3,2]</td>
<td>4.009</td>
<td>1.404</td>
<td>1163</td>
<td>1.000993</td>
</tr>
<tr>
<td>beta[4,2]</td>
<td>0.000</td>
<td>0.000</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>tau[2]</td>
<td>1.062</td>
<td>0.149</td>
<td>16640</td>
<td>1.000211</td>
</tr>
</tbody>
</table>

summary(model2.jags)[c("Mean","SD","SSeff","psrf")]

To avoid coding mistakes it might be better to use one (single) flexible model for all calculations.

### 18.6 Pseudopriors

Extract pseudopriors from previous estimates:

\[
\beta \sim \mathcal{N}(\mu, \tau) \quad \mu = \text{Mean}, \quad \tau = 1/\text{SD}^2
\]

```r
sum2prior <- function(jags, pattern, var) {
  x <- summary(jags)
  x[grep(pattern, rownames(x)), var]
}
sum2prior(model1B.jags, "beta[,1]", "Mean")

3.822052 3.833818 0.000000 10.793066
sum2prior(model1B.jags, "beta[,1]", "SD")

0.09835104 2.01108758 0.00000000 4.18976062
```

We construct the pseudopriors here one by one, so that we can see each step:

```r
within(modelSel.data,{
  priBetaMean[,1,2]<-sum2prior(model1B.jags,"beta[,1]","Mean")
  priBetaMean[,2,1]<-sum2prior(model2B.jags,"beta[,2]",&quot;Mean&quot;)
})[["priBetaMean"]]

, , 1

[,1]   [,2]
[1,] 0  3.822396
[2,] 0 -5.304239
[3,] 0  4.009177
[4,] 0  0.000000

, , 2

[,1]   [,2]
[1,] 3.822052 0
```

Now we do all the pseudopriors in one step:
pseudo.data <- within(modelSel1.data,

  priBetaMean[,1,2] <- sum2prior(model1B.jags, "beta\[.,1\]", "Mean")
  priBetaMean[,2,1] <- sum2prior(model2B.jags, "beta\[.,2\]", "Mean")
  priBetaTau[,1,2] <- 1/sum2prior(model1B.jags, "beta\[.,1\]", "SD")^2
  priBetaTau[,2,1] <- 1/sum2prior(model2B.jags, "beta\[.,2\]", "SD")^2
  priBetaTau[3,1] <- -100; priBetaTau[4,2] <- -100
  priTM[1,2] <- sum2prior(model1B.jags, "tau\[1\]", "Mean")
  priTM[2,1] <- sum2prior(model2B.jags, "tau\[2\]", "Mean")
  priTD[1,2] <- sum2prior(model1B.jags, "tau\[1\]", "SD")^2
  priTD[2,1] <- sum2prior(model2B.jags, "tau\[2\]", "SD")^2

)

modelSelPP.jags <- run.jags(model=modelSel1.model, data=pseudo.data, inits=ini,
  monitor=c("beta", "tau", "mod"))

summary(modelSelPP.jags)[,c("Mean", "SD", "SSeff", "psrf")]

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1,1]</td>
<td>3.821997</td>
<td>0.09871367</td>
<td>38950.0000</td>
<td>1.000209</td>
</tr>
<tr>
<td>beta[2,1]</td>
<td>4.002512</td>
<td>1.98414304</td>
<td>1377.0000</td>
<td>1.000341</td>
</tr>
<tr>
<td>beta[3,1]</td>
<td>0.000000</td>
<td>0.00000000</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>beta[4,1]</td>
<td>11.193314</td>
<td>4.12112634</td>
<td>1298.0000</td>
<td>1.000679</td>
</tr>
<tr>
<td>beta[1,2]</td>
<td>3.822861</td>
<td>0.09832232</td>
<td>41560.0000</td>
<td>0.999965</td>
</tr>
<tr>
<td>beta[2,2]</td>
<td>-5.298587</td>
<td>1.44274116</td>
<td>3594.0000</td>
<td>1.001714</td>
</tr>
<tr>
<td>beta[3,2]</td>
<td>4.006381</td>
<td>1.38483622</td>
<td>3343.0000</td>
<td>1.001336</td>
</tr>
<tr>
<td>beta[4,2]</td>
<td>0.000000</td>
<td>0.00000000</td>
<td>NA</td>
<td>NA</td>
</tr>
<tr>
<td>tau[1]</td>
<td>1.047751</td>
<td>0.11587482</td>
<td>15748.0000</td>
<td>1.001371</td>
</tr>
<tr>
<td>tau[2]</td>
<td>1.063313</td>
<td>0.09624414</td>
<td>16621.0000</td>
<td>1.003209</td>
</tr>
<tr>
<td>mod</td>
<td>1.402250</td>
<td>0.49035798</td>
<td>517.0000</td>
<td>1.010387</td>
</tr>
</tbody>
</table>

Compare with, e.g., model 2:

summary(model2B.jags)[5:8, c("Mean", "SD", "SSeff", "psrf")]

<table>
<thead>
<tr>
<th></th>
<th>Mean</th>
<th>SD</th>
<th>SSeff</th>
<th>psrf</th>
</tr>
</thead>
<tbody>
<tr>
<td>beta[1,2]</td>
<td>3.822396</td>
<td>0.09806754</td>
<td>39853.0000</td>
<td>1.000000</td>
</tr>
<tr>
<td>beta[2,2]</td>
<td>-5.304239</td>
<td>1.45780548</td>
<td>1152.0000</td>
<td>1.000958</td>
</tr>
<tr>
<td>beta[3,2]</td>
<td>4.009177</td>
<td>1.40370489</td>
<td>1163.0000</td>
<td>1.000993</td>
</tr>
<tr>
<td>beta[4,2]</td>
<td>0.000000</td>
<td>0.00000000</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>

With pseudopriors convergence is good and the chains mix well:

plot(as.mcmc(modelSelPP.jags)[,"mod"])
The mean coefficient of \textit{mod} is 1.402.

→ the second (polynomial) model has a posterior probability of 40.2%.

- The first (fractional polynomial) model has a posterior probability of 59.8%.
- The fractional polynomial model is 1.49 times more probable than the polynomial one.

Are these two models the only relevant models?

### 18.7 Model uncertainty

- F: inference is based on one model.
- B: composite inference from model posteriors.

What if there is a large number of possible models?

- Occam’s window: consider a subset of plausible and not too complex models.
- Markov Chain Monte Carlo Model Composition (MC³).
18.8 Bayes factors

Posterior probability of Model $H_1$:

$$\Pr(H_1 | X) = \Pr(H_1) \cdot \Pr(X | H_1) \frac{1}{\Pr(X)}$$

Hence

$$\frac{\Pr(H_1 | X)}{\Pr(H_2 | X)} = \frac{\Pr(H_1) \cdot \Pr(X | H_1)}{\Pr(H_2) \cdot \Pr(X | H_2)}$$

For uninformed priors $\Pr(H_1) = \Pr(H_2)$ we have the Bayes factor

$$K = \frac{\Pr(X | H_1)}{\Pr(X | H_2)} = \frac{\int \Pr(\theta_1 | H_1) \Pr(X | \theta_1, H_1) \, d\theta_1}{\int \Pr(\theta_2 | H_2) \Pr(X | \theta_2, H_2) \, d\theta_2} \neq \frac{\Pr(X | \theta^*_1, H_1)}{\Pr(X | \theta^*_2, H_2)}$$

**Interpreting $K$:** Harold Jeffreys (1961):

<table>
<thead>
<tr>
<th>$K$</th>
<th>10^0</th>
<th>10^0.5</th>
<th>10^1</th>
<th>10^1.5</th>
<th>10^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>barely worth mentioning</td>
<td>substantial</td>
<td>strong</td>
<td>very strong</td>
<td>decisive</td>
<td></td>
</tr>
</tbody>
</table>


$$e^0 \quad e^1 \approx 2.7 \quad e^3 \approx 20 \quad e^5 \approx 150$$

<table>
<thead>
<tr>
<th>$K$</th>
<th>10^0</th>
<th>10^0.5</th>
<th>10^1</th>
<th>10^1.5</th>
<th>10^2</th>
</tr>
</thead>
<tbody>
<tr>
<td>barely worth mentioning</td>
<td>positive</td>
<td>strong</td>
<td>very strong</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

19 Mixture Models

19.1 Example

Sometimes we assume that our population can be described as a mixture of two distributions. Here we construct such a mixture with means $\mu = 2$ and $\mu = 3$ respectively:

```r
set.seed(123)
N <- 100
group <- rbinom(N, 1, .3)
y <- rnorm(N, mean = 2 + group, sd = .3)
```
We first consider a model with exactly 2 groups:

```r
mix0.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu[group[i]+1],tau)
    group[i] ~ dbern(p)
  }
  for (g in 1:2) {
    mu[g] ~ dnorm(0,.0001)
  }
  p ~ dbeta(1,1)
  tau ~ dgamma(m^2/d^2,m/d^2)
  m ~ dgamma(1,1)
  d ~ dgamma(1,1)
  sd <- 1/sqrt(tau)
}

mix0.jags <- run.jags(mix0.model, data=list(y=y),
  inits=genInit(4,function(i) list(mu0=rnorm(2,0,100))),
  monitor=c("mu","p","sd"))
```

The model does not seem to converge well:

```r
summary(mix0.jags)
```

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu[1]</td>
<td>1.877390</td>
<td>1.987180</td>
<td>3.059580</td>
<td>2.2209454</td>
<td>0.44060046</td>
<td>1.9662915</td>
<td>0.0044379631</td>
</tr>
</tbody>
</table>
19.2 Labels and sorting

The problem has to do with *labels*. We have two *mus*, a large one and a small one. But which is which? We need a convention, e.g. that the smaller one is always *mu[1]*. There are different ways to implement this convention. One is *sort*.
19.3 More groups

For a potentially larger number of groups we replace `dbern` with `dcat` and `dbeta` with `ddirch` (the Dirichlet distribution).

```r
mixGen.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu[group[i]], tau)
    group[i] ~ dcat(p)
  }
  for (g in 1:G) {
    mu0[g] ~ dnorm(0, 0.0001)
    alpha[g] <- alphaD # concentration parameter
  }
  p[1:G] - ddirch(alpha)
  mu[1:G] <- sort(mu0)
  tau ~ dgamma(m^2/d^2,m/d^2)
  m ~ dgamma(1,1)
  d ~ dgamma(1,1)
  sd <- 1/sqrt(tau)
}'}
```
Convergence is not too exciting. Let us have a look at $p$:

We see that sometimes $p$ reaches extreme values. As a consequence, if, e.g. $p[2]=0$, there is no pressure on $mu[2]$, so $mu[2]$ starts drifting. Increasing the concentration parameter for the Dirichlet distribution helps:

Here is the symmetric Dirichlet distribution for different values of $\alpha$: 
mixGen3.jags <- run.jags(mixGen.model, data=list(y=y, G=2, alphaD=1.2), inits=genInit(4, function(i) list(mu0=rnorm(2, 0, 100))), monitor=c("mu", "p", "sd"))

summary(mixGen3.jags)

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu[1]</td>
<td>1.877330</td>
<td>1.967910</td>
<td>2.062510</td>
<td>1.969537</td>
<td>0.0477852</td>
<td>1.966130</td>
<td>0.00 04995502</td>
</tr>
<tr>
<td>mu[2]</td>
<td>2.829810</td>
<td>2.979145</td>
<td>3.130150</td>
<td>2.9784262</td>
<td>0.07777156</td>
<td>2.9811662</td>
<td>0.0008417142</td>
</tr>
<tr>
<td>p[1]</td>
<td>0.586327</td>
<td>0.6946925</td>
<td>0.799249</td>
<td>0.6932889</td>
<td>0.05462998</td>
<td>0.6982235</td>
<td>0.0004613692</td>
</tr>
<tr>
<td>p[2]</td>
<td>0.200751</td>
<td>0.3053075</td>
<td>0.413673</td>
<td>0.3067111</td>
<td>0.05462998</td>
<td>0.3017765</td>
<td>0.0004613692</td>
</tr>
<tr>
<td>sd</td>
<td>0.255730</td>
<td>0.3115115</td>
<td>0.384091</td>
<td>0.3160652</td>
<td>0.03533687</td>
<td>0.3057062</td>
<td>0.0005375619</td>
</tr>
</tbody>
</table>

MC%ofSD SSeff AC.10 psrf
mu[1] 1.0 9150 0.06431488 1.001063
mu[2] 1.1 8537 0.05781594 1.002620
p[1] 0.8 14021 0.01870416 1.000145
p[2] 0.8 14021 0.01870416 1.000145
sd 1.5 4321 0.16758452 1.002422

19.4 Ordering, not sorting

Above we made sure that the sampler used the sorted mus. This is not necessary. An alternative way to make sure that the mu[1] we observe is always the smaller one is to only sort the variables we monitor. We have to make sure that we sort p and mu in the same way.
mixord.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu0[group[i]],tau)  
group[i] ~ dcat(p1)
  }
  for (g in 1:G) {
    mu0[g] ~ dnorm(0,.0001)
    alpha[g] <- alphaD # concentration parameter
  }
  p1[1:G] ~ ddirch(alpha)
  oo <- order(mu0)
  for (g in 1:G) {
    mu[g] <- mu0[oo[g]]
    p[g] <- p1[oo[g]]
  }
  tau ~ dgamma(m^2/d^2,m/d^2)
  m ~ dgamma(1,1)
  d ~ dgamma(1,1)
  sd <- 1/sqrt(tau)
}'

mixord.jags <- run.jags(mixord.model, data=list(y=y, G=2, alphaD=1.2), 
  inits=genInit(4, function(i) list(mu0=rnorm(2,0,1))), 
  monitor=c("mu", "p", "sd"))
summary(mixord.jags)

19.5 Using `dnormmix`

We can also use a special “mixture” distribution: `dnormmix`.

mixmix.model <- 'model {
  for (i in 1:length(y)) {
    y[i] ~ dnormmix(mu0,tau0,p1)
  }
  for (g in 1:G) {
    mu0[g] ~ dnorm(0,.0001)
  }
}'
alpha[g] <- alphaD
tau0[g] <- tau
mu[g] <- mu0[oo[g]]
p[g] <- p1[oo[g]]
}
p1[1:G] ~ ddirch(alpha)
oo <- order(mu0)
tau ~ dgamma(m^-2/d^-2,m/d^-2)
m ~ dgamma(1,1)
d ~ dgamma(1,1)
sd <- 1/sqrt(tau)
)

mixmix.jags <- run.jags(mixmix.model, data=list(y=y, G=2, alphaD=1),
  inits=genInit(4),
  monitor=c("mu", "p", "sd"), modules="mix",
  factories="mix::TemperedMix sampler off")
summary(mixmix.jags)

<table>
<thead>
<tr>
<th></th>
<th>Lower95</th>
<th>Median</th>
<th>Upper95</th>
<th>Mean</th>
<th>SD</th>
<th>Mode</th>
<th>MCerr</th>
</tr>
</thead>
<tbody>
<tr>
<td>mu[1]</td>
<td>1.87886</td>
<td>1.96877</td>
<td>2.06829</td>
<td>1.97095</td>
<td>0.0487</td>
<td>1.9652</td>
<td>0.0005</td>
</tr>
<tr>
<td>mu[2]</td>
<td>2.82944</td>
<td>2.98069</td>
<td>3.13551</td>
<td>2.97983</td>
<td>0.0809</td>
<td>2.9801</td>
<td>0.0007</td>
</tr>
<tr>
<td>p[1]</td>
<td>0.58295</td>
<td>0.69665</td>
<td>0.79718</td>
<td>0.6947</td>
<td>0.0571</td>
<td>0.7015</td>
<td>0.0008</td>
</tr>
<tr>
<td>p[2]</td>
<td>0.20281</td>
<td>0.30334</td>
<td>0.41705</td>
<td>0.3053</td>
<td>0.0571</td>
<td>0.2984</td>
<td>0.0008</td>
</tr>
<tr>
<td>sd</td>
<td>0.25653</td>
<td>0.3122</td>
<td>0.38856</td>
<td>0.3172</td>
<td>0.0366</td>
<td>0.3060</td>
<td>0.0005</td>
</tr>
</tbody>
</table>

MC%ofSD SSeff AC.10 psrf
mu[1] 1.0 9448 0.06863096 1.000358
mu[2] 1.0 10278 0.06463824 1.002544
p[1] 1.5 4648 0.11796077 1.002673
p[2] 1.5 4648 0.11796078 1.002673
sd 1.5 4572 0.17138468 1.001610

19.6 Exercises
Consider the following data:

N<-100
set.seed(123)
group <- rbinom(N,1,.3)
x <- rnorm(N)
y <- rnorm(N,mean=x*group-group, sd=.3)

group can not be observed. Estimate a mixture model where \( Y = X'\beta_g + \epsilon \) and where \( \beta_g \) is a group specific coefficient.

20 Summary

- Probability: objective vs. subjective.
• Priors, how to get them?

• Results: F: depend on intention of the experimenter  
  B: depend on prior.

• Flexible modelling: F. has only a limited number of models.  
  F: precise models which are sometimes not such a good representation of the problem.  
  B: approximate models which can be a more precise representation of the problem.

• Interpretation: p-values versus posteriors.  
  B. predicts (posterior) probability of a hypothesis.  
  F. writes carefully worded statements which are wrong 5% of the time (or any other probability) provided \( H_0 \) is true.

• Quality of decisions: p-values are only a heuristics for a decision rule.  
  B.’s decisions are better in expectation.

21 Exercises

Exercise 21.1 You assume that \( X \sim N(\mu, \tau) \).

Your prior is \( \mu \sim N(10, 1), \tau = 2. \)

Your sample is \( X = \{8, 9, 10\}. \)

What is your posterior for \( \mu \) and \( \tau \)?

Exercise 21.2 (Female labour supply) Have a look at the dataset Mroz in Ecdat. Estimate the following model

\[
\text{hours}_w = \beta_0 + \beta_1 \text{educ}_w + \beta_2 \text{income} + \epsilon_i
\]

Compare two specifications:

• First assume that \( \text{hours}_w \) is not censored. Use standard OLS and a Bayesian specification.

• Then assume that \( \text{hours}_w \) is censored at zero. Use a standard ML model and a Bayesian specification.

• Next make this a robust regression, taking into account the censoring.

Exercise 21.3 (Young males) Have a look at the dataset Males in Ecdat.
1. Estimate a mixed effects model that explains wage as a function of exper and school. Include a random effect on the intercept for the identity of the male. Use Maximum Likelihood and a Bayesian Model. Check convergence.

2. Use a robust model.