

Bayesian Specialization Notes

Oren Bochman

2025-01-01

Table of contents

Preface	14
1 About	16
2 Introduction to Bayesian Statistics	17
3 Introduction	18
3.1 Why Inference?	18
3.1.1 Descriptive Statistics	19
3.1.2 Inferential Statistics	19
3.2 How is Inference different from Prediction?	19
3.2.1 Prediction	19
3.2.2 Inference	20
3.3 References	20
4 Probability	21
5 Probability	22
5.1 Rules of Probability, Odds & Expectation	22
5.2 Properties of Probability Measures	24
5.2.1 Odds	26
5.2.2 Expectation	27
5.3 Probability Paradigms	27
5.4 Bayesian Probability and Coherence	30
5.5 Discussions: Objectivity	35
5.6 Expected values	36
5.6.1 Expectation of a discrete random variable	36
5.6.2 Expectation of a continuous random variable . .	37
5.6.3 Properties of Expectation	37
5.7 Variance	38
5.7.1 Properties of Variance	38
5.8 Covariance	39
5.9 Correlation	39

6 Bayes' Theorem	40
6.1 Bayes' Theorem	40
6.2 Conditional Probability	40
6.2.1 Conditional Probability Example - Female CS Student	40
6.2.2 Inverting Conditional Probabilities	42
6.2.3 Conditional Probability Example - ELISA HIV test	42
6.3 Bayes' theorem (Video)	43
6.4 Bayes' Theorem for continuous distributions	45
7 Distributions	46
7.1 Distributions	46
7.2 The Bernoulli & Binomial Distribution	46
7.2.1 The Bernoulli Distribution	46
7.2.2 The Binomial Distribution	49
7.2.3 The Discrete Uniform Distribution	52
7.2.4 The Continuous Uniform Distribution	54
7.3 The Normal, Z, t Distributions	54
7.3.1 The Standard Normal distribution	54
7.3.2 The Normal distribution	55
7.3.3 The t-Distribution	57
7.4 The Exponential Distribution	58
8 Additional Discrete Distributions	61
8.1 The Geometric Distribution	61
8.2 The Multinomial Distribution	62
8.3 The Poisson Distribution	62
8.3.1 Relations	63
8.4 Hypergeometric Distribution	63
9 Frequentist Inference	64
9.1 Confidence Intervals	64
9.2 Likelihood function and MLE	66
9.3 Computing the MLE	69
9.4 Computing the MLE: examples	70
9.4.1 Computing the MLE for Exponential RV	70
9.4.2 Computing the MLE for Uniform RV	71
9.5 Cumulative Distribution Function	72
9.6 Quantile Function	73

10 Introduction to R	74
10.1 Plotting the likelihood function in R	75
11 Probability Distributions in R	77
12 Bayesian Inference	79
12.1 Inference example: frequentist	79
12.2 Bayesian Approach to the Problem	81
12.3 Continuous version of Bayes' theorem	82
12.4 Posterior Intervals	82
12.5 Discussion CIs	82
12.5.1 Focusing on Bayesian / Frequentist paradigms .	83
12.5.2 Focusing on the CI choices	83
13 Priors	85
13.1 Priors and prior predictive distributions	85
13.1.1 Calibration - making priors precise	87
13.2 Prior Predictive: Binomial Example	88
13.3 Posterior Predictive Distribution	90
13.4 Posterior Predictive Distribution	90
14 Binomial Data	93
14.1 Bernoulli/Binomial likelihood with a uniform prior . . .	93
14.2 Conjugate Priors	95
14.3 Posterior mean and effective sample size	96
14.4 Data Analysis Example in R	99
15 Poisson Data	105
15.0.1 Poisson - Chocolate Chip Cookie	105
16 Exponential Data	108
17 Normally distributed Data	111
17.1 Normal Likelihood with known variance	112
17.1.1 Prior (and posterior) predictive distribution .	113
17.2 Normal likelihood with expectation and variance unknown	114
18 Non-Informative Priors	117
18.1 Non-Informative Priors	117
18.1.1 Improper priors	118
18.1.2 Statements about improper priors	119
18.1.3 Normal Case	119

18.1.4	Normal with unknown Variance	120
18.2	Jeffrey's Prior	121
18.2.1	Normal Data	123
18.2.2	Binomial	124
18.2.3	Closing information about priors	124
18.3	Fisher Information	125
18.4	Sensitivity analysis of priors	126
19	Brief Review of Regression	127
19.1	Conjugate Modeling	128
19.1.1	σ^2 known	128
19.1.2	σ^2 Unknown	128
19.2	Linear Regression	129
19.2.1	Single Variable Regression	129
19.2.2	Multivariate Regression	132
20	Statistical Modeling and Monte Carlo estimation	137
20.1	Objectives	137
20.2	A Poll for a political candidate	138
20.3	Modeling Process	139
20.3.1	Process outline:	141
21	Bayesian Modeling	143
22	Notes - Bayesian Modeling	144
22.1	Components of a Bayesian Model	144
22.2	Model Specification	148
22.2.1	Hierarchical representation	148
22.2.2	Graphical representation	149
22.3	Posterior derivation	152
22.4	Non-conjugate models	154
23	Monte Carlo estimation	158
23.1	Monte Carlo Integration	158
23.2	Calculating probabilities	160
23.3	Monte Carlo Error and Marginalization	161
23.4	Marginalization	162
23.5	Computing Examples	162
23.6	Monte Carlo error	165
23.7	Marginalization	166

24	Markov chains	168
24.1	Definition	168
24.2	Examples of Markov chains	169
24.2.1	Discrete Markov chain	169
24.2.2	Random walk (continuous)	169
24.3	Transition matrix	170
24.4	Stationary distribution	172
24.4.1	Continuous example	174
25	Metropolis-Hastings	177
25.1	The Metropolis-Hastings Algorithm	177
25.2	The M-H Algorithm	178
25.3	Proposal distribution q	179
25.4	Acceptance rate α	179
25.5	Demonstration of a Discrete case	180
25.6	Random walk with Normal likelihood, t prior	185
26	Introduction to JAGS	192
26.1	Setup	192
26.1.1	Introduction to JAGS	192
26.1.2	Installation and setup	192
26.2	Modeling in JAGS	193
26.2.1	1. Specify the model	193
26.2.2	2. Set up the model	193
26.2.3	3. Run the MCMC sampler	194
26.2.4	4. Post-processing	195
27	Gibbs sampling	196
27.1	Multiple parameter sampling and full conditional distributions	197
27.1.1	Full conditional distributions	198
27.1.2	Gibbs sampler	199
27.2	Conditionally conjugate prior example with Normal likelihood	200
27.2.1	Normal likelihood, unknown mean and variance	200
27.3	Computing example with Normal likelihood	202
27.3.1	conditionally conjugate priors for the mean	202
27.3.2	conditionally conjugate priors for the variance	203
27.3.3	Gibbs sampler in R	204
28	Assessing Convergence	208

29 Notes - Assessing Convergence	209
29.1 Convergence diagnostics	209
29.1.1 Trace plots	209
29.1.2 Monte Carlo effective sample size	211
29.2 Burn-in	216
29.2.1 Multiple chains, Gelman-Rubin	217
29.2.2 Monte Carlo estimation	220
30 Notes - Linear regression	222
30.1 Introduction to linear regression	222
30.1.1 Priors	225
30.2 R	226
30.3 Python	227
31 Lesson 7.2 Code - Setup & Data	228
31.1 EDA	229
31.1.1 Modeling	233
32 Lesson 7.3 Code - Model in JAGS	235
33 Lesson 7.4 Model Checking- MCMC convergence	237
33.0.1 Convergence diagnostics for the Markov Chains	237
33.1 Residual checks	240
34 Lesson 7.5 - Alternative models	246
34.1 Additional covariates	246
34.1.1 Student-t likelihood	253
34.2 Compare models using Deviance information criterion (DIC)	257
34.3 Regression Diagnostics	258
34.3.1 Residuals vs Fitted	259
34.3.2 QQ plot of the residuals	260
34.3.3 Scale Location plot	261
34.3.4 Cook's Distance	262
34.3.5 Residuals vs Leverage	263
34.3.6 Cook's Distance vs Leverage	264
34.3.7 Python	265
35 ANOVA	266
35.1 Introduction to ANOVA	266
35.2 One way ANOVA model using JAGS	266
35.2.1 Data & EDA	266

35.2.2	Modeling	267
35.2.3	Model checking	272
35.2.4	Results	276
35.3	Two Factor ANOVA	277
35.3.1	Data	277
35.3.2	One-way model	280
35.3.3	Two-way additive model	283
35.3.4	Two-way cell means model	290
35.3.5	Results	296
36	Logistic regression	300
36.1	Introduction to Logistic Regression	300
36.1.1	Data	300
36.1.2	Variable selection	303
36.1.3	Model	304
36.1.4	Results	317
37	Poisson regression	321
37.1	Introduction to Poisson regression	321
37.2	Poisson regression - JAGS model	323
37.2.1	Doctor Visits Model	325
37.2.2	Model checking - Residuals	328
37.3	Predictive distributions	332
37.3.1	Predictive distributions	333
37.4	Prior sensitivity analysis	336
37.4.1	Poisson regression example	337
37.5	Overdispersed model	342
37.5.1	Transformations:	343
38	Hierarchical modeling	355
38.1	Introduction to Hierarchical modeling	355
38.2	Normal hierarchical model	355
38.3	Applications of hierarchical modeling	355
38.4	Prior predictive simulation	356
38.4.1	Data	356
38.4.2	Prior predictive checks	357
38.5	JAGS Model	359
38.5.1	Model checking	362
38.5.2	Results	364
38.6	Posterior predictive simulation	365

38.7	Random intercept linear model	368
38.7.1	Results	375
38.7.2	Other models	377
38.8	Mixture models	377
38.8.1	Bayesian inference for mixture models	380
38.9	Example with JAGS	381
38.9.1	Data	381
38.9.2	Model	382
38.9.3	Results	385
39	Capstone Project	389
40	Appendix: Notation	390
40.1	The argmax function	393
40.2	Indicator Functions	393
40.2.1	Products of Indicator Functions:	394
41	Appendix: Discrete Distributions	395
41.1	Discrete Uniform	395
41.1.1	Stories	395
41.1.2	Moments	396
41.1.3	Probability mass function (PMF)	396
41.1.4	Cumulative distribution function (CDF)	396
41.1.5	Prior	396
41.2	Bernoulli Distribution	396
41.2.1	Stories	396
41.2.2	Parameters	397
41.2.3	Examples	397
41.2.4	Checklist	397
41.2.5	Moments	397
41.2.6	PMF	398
41.2.7	CDF	398
41.2.8	Likelihood	398
41.2.9	Entropy and Information	398
41.2.10	Usage	399
41.2.11	Plots	399
41.3	Binomial distribution	402
41.3.1	Stories	402
41.3.2	Parameters	402
41.3.3	Conditions	402
41.3.4	Examples	402

41.3.5	Usage	403
41.3.6	Relationships	404
41.3.7	Plots	405
41.4	Hypergeometric distribution	407
41.4.1	story 1 - Urn Model	407
41.4.2	Examples	407
41.4.3	Story	407
41.5	Poisson distribution	408
41.5.1	Stories	408
41.5.2	Checklist	408
41.5.3	Examples	408
41.5.4	Moments	409
41.5.5	Probability mass function (PMF)	409
41.5.6	Cumulative distribution function (CDF)	409
41.6	Geometric distribution	410
41.6.1	Stories	410
41.6.2	Conditions	411
41.6.3	Examples	411
41.6.4	Moments	411
41.6.5	PMF	412
41.6.6	CDF	412
41.6.7	Memoryless property	412
41.6.8	Worked out Examples	412
41.6.9	Plots	413
41.7	Negative Binomial Distribution	414
41.7.1	Story	414
41.7.2	Parameters	415
41.7.3	Conditions	415
41.7.4	Examples	415
41.7.5	Moments	416
41.7.6	Probability mass function (PMF)	416
41.7.7	Cumulative distribution function (CDF)	416
41.8	Multinomial Distribution	416
41.8.1	Story	417
41.8.2	Examples	417
41.8.3	Moments	417
41.8.4	Probability Mass Function (PMF)	417
41.9	Beta Binomial	418
41.9.1	Story 1 - Polya Urn Model	418
41.9.2	Story 2 compound distribution	418
41.9.3	Moments	419

41.9.4	Probability mass function (PMF)	419
41.9.5	Cumulative distribution function (CDF)	419
41.9.6	Relations	419
42	Appendix: Continuous Distributions	422
42.1	The Continuous Uniform	422
42.1.1	Stories	422
42.1.2	Moments	422
42.1.3	Probability mass function (PDF)	422
42.1.4	Cumulative distribution function (CDF)	423
42.1.5	Prior	423
42.2	The Beta Distribution	423
42.2.1	Story	423
42.2.2	PDF & CDF	423
42.2.3	Moments	424
42.2.4	Relations	425
42.2.5	As a prior	425
42.3	The Cauchy Distribution	426
42.3.1	PDF	426
42.3.2	CDF	426
42.3.3	As a prior	426
42.4	Double Exponential Distribution (Laplace)	426
42.5	The Gamma Distribution	427
42.5.1	Story	427
42.5.2	PDF	427
42.5.3	Moments	427
42.5.4	Relations	428
42.6	Inverse Gamma Distribution	428
42.6.1	PDF	428
42.6.2	Moments	428
42.7	The Z or Standard normal distribution	428
42.8	The Normal Distribution	429
42.8.1	PDF	429
42.8.2	Moments	430
42.9	The t-Distribution	432
42.10	Location Scale Parametrization t-distribution	433
42.11	The Exponential Distribution	435
42.11.1	Story	435
42.11.2	PDF	435
42.11.3	CDF	436
42.11.4	Moments	436

42.11.5 Special cases:	436
42.11.6 Properties:	436
42.12 LogNormal Distribution	438
42.13 Pareto Distribution	438
42.14 Weibull Distribution	439
42.14.1 PDF	439
42.15 Chi Squared Distribution	439
42.15.1 PDF:	439
42.15.2 CDF:	439
42.15.3 MOMENTS	440
42.16 Logistic Distribution	440
42.17 F Distribution	440
42.17.1 PDF	440
42.17.2 CDF	440
42.17.3 Moments	440
43 Appendix: Exponents & Logarithms	441
43.1 Exponents	441
43.2 Natural Logarithms	441
43.2.1 Log of exponent	443
44 Appendix: The Law of Large Numbers	444
44.1 Law of large numbers	444
45 Appendix: The Central Limit Theorem	445
45.1 Central Limit Theorem	445
46 Appendix: Conjugate Priors	447
46.1 Conjugate Priors	447
47 Appendix: Link Function	449
47.1 Link function	449
48 Bayes by backprop	452
48.1 Introduction	452
48.2 Bayesian learning	454
48.3 Bayesian neural network with weight uncertainty – in principle	455
48.4 Variational Bayes	456
48.5 A backpropagation scheme	458
48.5.1 The idea	458
48.5.2 The reparameterization trick	459

48.5.3	Implementation	460
48.5.4	Minibatches	461
48.5.5	Conclusion	462
48.5.6	Further reading and resources	462
49	Bayesian Books in R & Python	463
49.1	Introduction to Probability	463
49.2	Books in R	463
49.3	Books in Python	464
50	References	465
Index		468

Preface

These are my notes on Bayesian Statistics. They are based on online courses I took after I worked as a Data Scientist for a number of years. Over this time I have often felt that if I was better at statistics I could overcome limitations in the data with mathematics, build even better models, and perhaps most importantly make better inferences using model I have already developed.

I tried to review the main result in statistic and probability theory, and I started to get better results and so I decided to take some classes. I often felt that the material was introductory and simplistic. Real world problems are monsters and the examples are usually trivial in comparison. The issues raised in class are rarely the troubles I saw at work. But I do believe deep down that the two are somehow related. And indeed as I covered more material certain aspects began to connect.

I noticed that like in real classes the teachers made mistakes, their motivation was not always clear and that they skipped steps or referred to certain results. On the other hand each teacher offered different insights into this complicated area of data analysis.

I therefore have a number main areas of Focus:

1. What are the questions one should ask
2. What are the explicit details of each examples or problem.
3. What is the mathematical representation of the model for these
4. What is the code representation for this in R and in Python.
5. Can I find or create diagrams to make interpretation of probabilities clearer.
6. Can I annotate the main equations to break them down.
7. Can I keep track of the most useful Mathematical results in Probability and Statistics?
 1. I tried to keep these handy as appendices to keep the main material less cluttered. However as time goes by these keep expanding.

8. Can I learn to communicate my intuitions of these concepts to laymen and to colleagues
 1. Some courses have *discussion prompts*, but I tried to make the most of these, and included them in these notes.
 2. I realized that while I often get excited about a new paper or technique my colleges are smart and talented people and quickly ask questions that are difficult to answer. These questions often indicate gaps of knowledge which can dampen the enthusiasm for implementing these new ideas.
 3. I found that good communicators can overcome these issues more readily and connect what they already know.

I have often found exercises rather easy to solve and so I often breezed through them with little notice. This time I resolved to make the most of these as opportunities to get better at using and manipulating probabilities and posterior distributions. So although I could get around 75% in each exercises based on intuition I took the extra effort to understand what is really going on here mathematically.

At work one of the main challenges is making the problem conform to a simple model. This can be even more challenging when the goal is a **latent** (unobserved) variable or when you are considering a synergy of **multiple effects** and you have seen and unseen **confounds**. In many cases it is unclear how to proceed based on the simple examples we see in these classes. However I am now able to look at the problems with more critical point of view. Also I see great advantages of a quick expositions to many new simple models. In Bayesian hierarchical framework each can become a link to adding just a little more complexity, integrating new types of prior information and so on. So I view these as jumping boards for overcoming my next challenges.

1 About

About this site

1 + 1

2

2 Introduction to Bayesian Statistics

 Note

Initially I signed up to two different specializations on Bayesian statistics. My notes cover the UoSCSC specialization This intro covers the

3 Introduction

What is covered is:

- The basics of Bayesian probability
- Understanding Bayesian inference and how it works
- The bare-minimum set of tools and a body of knowledge required to perform Bayesian inference in Python, i.e. the PyData stack of NumPy, Pandas, Scipy, Matplotlib, Seaborn and [Plot.ly](#)
- A scalable Python-based framework for performing Bayesian inference, i.e. PyMC3

With this goal in mind, the content is divided into the following three main sections (courses).

- Introduction to Bayesian Statistics
- Introduction to Monte Carlo Methods
- PyMC3 for Bayesian Modeling and **Inference**
- Using the Databricks Community Edition for Data Science
- Using Binder for deploying and interacting with the notebooks
- Setting up the Anaconda distribution for the notebooks
- Introduction to the Databricks Community Edition (CE)
- Databricks CE - Cluster setup
- Databricks CE - Get familiar with the notebook interface
- Databricks CE - Git integration for notebooks

3.1 Why Inference?

The purpose of the set of courses is to focus on **Inferential Statistics** as opposed to **Descriptive Statistics**.

All the samples in the group that we are interested in learning about make up a **population**. Populations can be described by **parameters** such as the mean and variance since they represent all of the data. Often, we do not have access to all the data in our population and have to sample from

the population. The metrics of mean and variance computed from these samples are not called parameters but **statistics** of the data.

3.1.1 Descriptive Statistics

This is used to summarize the data so that we have a quantitative way to understand data. This allows to understand and visualize data qualitatively. We can draw conclusions about the nature of the data. Descriptive statistics is applied to a **population** and hence can provide measures such as the mean and variance of the data. They do not allow us to make predictions about data that we have not analyzed.

3.1.2 Inferential Statistics

Inferential Statistics allow us to make generalizations about the population from the samples. This process of sampling introduces errors as this is never a perfect representation of the underlying data. The *statistics* thus computed are supposed to be an estimate of the true population parameters. It allows you to form a distribution of the population from the sampled data by accounting for the errors in the sampling, thereby allowing you to make predictions about data that is not yet seen or sampled.

3.2 How is Inference different from Prediction?

Reference

3.2.1 Prediction

If you happen to come from a background in Machine Learning, you are probably used to making predictions. This is exactly what it sounds like, you use a model to make predictions on unseen data. The predictive process involves the following steps

- Create the model
- Select the best model using performance metrics such as accuracy, F1 scores on out-of-sample data
- Make predictions on new data

3.2.2 Inference

In Inference, you are trying to model a distribution and understand the process that generates the data. This involves the following steps

- Create the model, usually involves some prior understanding of the data generation process
- Select the model using goodness-of-fit measures such as such as residual analysis, deviance, AIC scores etc.
- Perform inference by generating distributions that describe the data, or the data generation process

3.3 References

- (Casella and Berger 2002) [e-book solutions](#)
- (Spanos 2019)
- (Hobbs and Hooten 2015) [ebook website](#)
- (VanderPlas 2016) [ebook notebooks](#)
- (Bishop 2006) [ebook website](#)

4 Probability

Bayesian Statistics: From Concept to Data Analysis

5 Probability

In these notes I supplement the course material with my own notes for establishing an axiomatic foundation for probability. These were omitted from the course material, but alluded to in the course and their absence was so irksome that I decided to add them to my notes.

5.1 Rules of Probability, Odds & Expectation

- **Background reading:** This reviews the rules of probability, odds, and expectation.

For all its discussion of different paradigms of probability, the course lacks a rigorous definition of probability.

Definition 5.1 (Sample Space and Sample Point).

Sample space Ω

$$\Omega = \{\forall w \mid w \text{ is an outcome of an experiment}\} \neq \emptyset \quad (5.1)$$

then Ω is called a **sample space**.

Since

$$\Omega \neq \emptyset \implies \exists \omega \in \Omega \quad (5.2)$$

then ω is called a **sample point**

Sample point ω

Definition 5.2 (Event).

Event A

$$\Omega \neq \emptyset \implies \exists \mathcal{F} \subset 2^\Omega \implies \exists A \in \mathcal{F} \quad (5.3)$$

Let \mathcal{F} denote a family of subsets of a sample space Ω , and A any such subset. Then A is called an **event**

Definition 5.3 (Elementary Event). An event composed of a single point ω is called an **elementary event**.

Definition 5.4 (Outcome). We say that event A *happened* if when conducting the experiment we got an outcome ω and $\omega \in A$.

Definition 5.5 (Certain Event). Ω is called the **certain event**.

Definition 5.6 (Impossible Event). \emptyset is called the **impossible event**.

Definition 5.7 (σ -Algebra). A family of events \mathcal{F} with the following properties:

1. Ω is the universal set

$$\Omega \in \mathcal{F} \quad (5.4)$$

2. \mathcal{F} is closed under complement operation:

$$\forall A \in \mathcal{F} \implies A^c \in \mathcal{F} \quad (5.5)$$

3. \mathcal{F} is closed under countable unions:

$$\exists A_i \in \mathcal{F} \quad i \in \mathcal{N} \implies \bigcup_{n=1}^{\infty} A_i \in \mathcal{F} \quad (5.6)$$

is called a σ -algebra or a σ -field .

some properties of σ -algebra

- <https://math.stackexchange.com/questions/1330649/difference-between-topology-and-sigma-algebra-axioms>
- [An epsilon of room: pages from year three of a mathematical blog](#)
section 2.7

Definition 5.8 (Probability Measure). if Ω is a *sample space* Definition 5.1 and \mathcal{F} a σ -algebra Definition 5.7 for Ω then a function $P : \mathcal{F} \rightarrow [0, 1]$ with the following properties:

1. Total measure of the sample space is 1:

$$\Pr(\Omega) = 1 \quad (5.7)$$

2. countably additive for pairwise disjoint countable collections of events: is called a probability measure over \mathcal{F} .

$$\forall E_{i \in \mathbb{N}} \quad \Pr\left(\bigcup_{n \in \mathbb{N}} E_n\right) = \sum_{n \in \mathbb{N}} \Pr(E_n) \quad (5.8)$$

then P is a **probability measure** over \mathcal{F}

Definition 5.9 (Probability Space). If Ω is a *sample space* Definition 5.1 and \mathcal{F} a σ -algebra Definition 5.7 for Ω , and P a *probability measure* (Definition 5.8) for \mathcal{F} then the ordered set $\langle \Omega, \mathcal{F}, P \rangle$ is called a **probability space**

💡 Notation

sometimes \mathcal{F} is replaced with Σ . for the σ -algebra like in the figure below

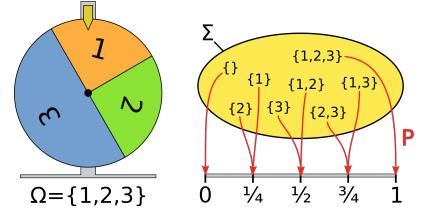


Figure 5.1: Illustration of a probability space by Ziggystar

5.2 Properties of Probability Measures

The probability of the *null event* is 0.

$$\Pr(\emptyset) = 0 \quad (5.9)$$

Probabilities of all possible events (the space of all possible outcomes) must sum to one.

$$\Pr(\Omega) = 1 \quad (5.10)$$

$$A \cap B = \emptyset \implies \Pr(A \cup B) = \Pr(A) + \Pr(B) \quad (5.11)$$

$$\Pr(A^c) = 1 - \Pr(A) \quad \forall A \in \Omega \quad (5.12)$$

if A is an event in Ω then A^C is in Omega and since they are mutually exclusive by Equation 5.10

If S is the certain event in class $C\Omega$ then

For every event X in class Ω

$$1 \geq \mathbb{P}r(X) \geq 0 \quad \forall X \in \Omega \quad (5.13)$$

Probabilities add to one:

$$\sum_{i \in \Omega} \mathbb{P}r(X = i) = 1 \quad (5.14)$$

The complement of an event A is A^c

$$\mathbb{P}r(S) = 1 \quad (5.15)$$

If events A_λ are mutually exclusive (only one event may happen):

$$\mathbb{P}r(A_1 \cup A_2) = \mathbb{P}r(A_1) + \mathbb{P}r(A_2) - \mathbb{P}r(A_1 \cap A_2) \quad (5.16)$$

$$\mathbb{P}r(\bigcup_{\lambda \in \Omega} A_\lambda) = \sum_{\lambda \in \Omega} \mathbb{P}r(A_\lambda) \quad (5.17)$$

if B_i is a finite or countably *infinite partition* of a sample space Ω then

$$\mathbb{P}r(A) = \sum_{i=1}^N \mathbb{P}r(A \cap B_i) = \sum_{i=1}^N \mathbb{P}r(A|B_i)\mathbb{P}r(B_i) \quad (5.18)$$

5.2.1 Odds

C-3PO: Sir, the possibility of successfully navigating an asteroid field is approximately 3,720 to 1! Han Solo: Never tell me the odds! — Star Wars Episode V: The Empire Strikes Back

Another way to think about probabilities is using *odds* Equation 5.19. Odds are more intuitive when we are thinking about the risk of an event happening or not happening, and when we consider the risk associated with uncertainty odds are a handy way of considering the risks.

Definition 5.10 (Odds Definitions). the odds of an event A are:

$$\mathcal{O}(A) = \frac{\Pr(A)}{\Pr(A^c)} = \frac{\Pr(A)}{1 - \Pr(A)} \quad (5.19)$$

It is also possible to convert odds to probabilities Equation 5.20

Theorem 5.1 (Probability from odds).

$$\Pr(A) = \frac{\mathcal{O}(A)}{1 + \mathcal{O}(A)} \quad (5.20)$$

Proof.

$$\begin{aligned} \mathcal{O}(A) &= \frac{\Pr(A)}{1 - \Pr(A)} && \text{(odds definition)} \\ \implies \Pr(A) &= \mathcal{O}(A)(1 - \Pr(A)) && (\times \text{denominator}) \\ \implies \Pr(A) &= \mathcal{O}(A) - \mathcal{O}(A)\Pr(A) && (\text{expand}) \\ \implies \Pr(A)(1 + \mathcal{O}(A)) &= \mathcal{O}(A) && (\text{collect}) \\ \implies \Pr(A) &= \frac{\mathcal{O}(A)}{1 + \mathcal{O}(A)} && \blacksquare \end{aligned}$$

□

If we are at the races and thinking about each horse a horse what we may care about is if it will win or lose. In such a case the odds can summarize the ratio of past successes and failures to win. Odds seem to be in line with a frequentist view summarizing ratios of success to failure. In reality, the other horses have odds as well and we may want to consider the probability

of winning given the other horses in the race, and perhaps other parameters, like the track type, length of the race, jockey, and perhaps some hot tips. So let us not get ahead of ourselves

Data Scientist - insights.

Many of these formulas are rather tedious. But, once you start to work on a data science project you will often discover that there are some problems with the data and because of that you cannot use your favorite algorithm. Or worse when you do the results are not very useful. It is at this point that the ability to think back to first principles will be very fruitful. The more of this material you can recall, the more the dots will connect, and your ability will translate into models of increasing sophistication. Luckily, the rules of probability are *logical*. So it is fairly easy to remember or even derive if you take some time to understand them.

I realize that figuring out which results are more useful is easier in hindsight. And one of the reasons I am taking these courses is to annotate in my note the results I think to be most useful.

5.2.2 Expectation

The *expectation* of a random variable (RV) X is the **weighted average** of the outcomes it can take weighted by their probabilities.

Definition 5.11 (Expectation for a discrete RV).

$$\mathbb{E}(x) = \sum_{i=1}^N x_i \times \mathbb{P}r(X = x_i) \quad (5.21)$$

Definition 5.12 (Expectation for a continuous RV).

$$\mathbb{E}(x) = \int_{\Omega} x \mathbb{P}r(X = x) dx \quad (5.22)$$

5.3 Probability Paradigms

We start by looking at probability as defined or interpreted under three paradigms. Probability is at its root a logical and scientific approach to formalizing and modeling *uncertainty*.

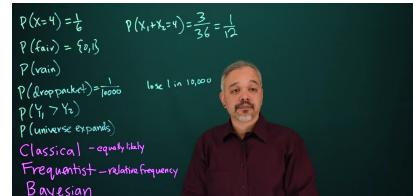


Figure 5.2: Probability Paradigms

The three paradigms are:

Definition 5.13 (Classical Probability). Deals primarily with cases where probabilities are distributed equally, like with dice and cards.

💡 Biographical note on Abraham de Moivre

The Probability of an Event is greater or less, according to the number of chances by which it may happen, compared with the whole number of chances by which it may either happen or fail. — (Moivre 1718)

Abraham de Moivre (1667-1754) was a prominent French mathematician known for his significant contributions to the field of probability and his work on the foundations of Bayesian statistics. His research and writings played a crucial role in establishing the mathematical principles of probability theory and laid the groundwork for future advancements in the field.

De Moivre is best known for his work on the theory of probability. He made significant advancements in understanding the *Binomial distribution* and its application to games of chance and coin tossing. In his influential book, “The Doctrine of Chances” (1718), he presented a comprehensive treatise on probability theory, providing mathematical explanations for various phenomena such as the law of large numbers and the central limit theorem. His book became a standard reference in the field and greatly influenced subsequent research on probability.

Furthermore, de Moivre’s work laid the foundation for Bayesian statistics, although the term “Bayesian” was not coined until many years after his death. He developed a formula known as de Moivre’s theorem, which establishes a connection between the normal distribution and the binomial distribution. This theorem became a fundamental tool in probability theory and enabled the calculation of probabilities for large sample sizes. It provided a bridge between frequentist and Bayesian approaches, allowing for the estimation of parameters and the quantification of uncertainty.

And thus in all cases it will be found, that although Chance produces irregularities, still the Odds will be infinitely great, that in process of Time, those Irregularities will bear no proportion to the recurrency of

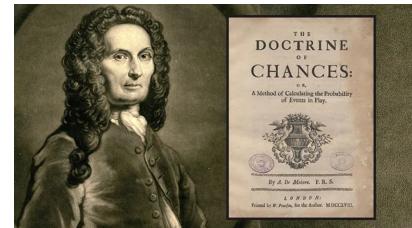


Figure 5.3: Abraham De Moivre

that Order which naturally results from Original Design. (Moivre 1718)

He was an active participant in scientific societies and maintained correspondence with renowned mathematicians of his time, including Isaac Newton and James Stirling. His work played a crucial role in disseminating mathematical knowledge and promoting the study of probability theory across Europe. De Moivre's research and writings laid the groundwork for the development of probability theory and Bayesian statistics. His ideas and formulas continue to be foundational in the field, and his contributions have had a lasting impact on mathematics, statistics, and the broader scientific community.

His work remains an essential reference for researchers and serves as a testament to his profound understanding of probability and statistics.

Further, the same Arguments which explode the Notion of Luck, may, on the other side, be useful in some cases to establish a due comparison between Chance and Design: We may imagine Chance and Design to be, as it were, in Competition with each other, for the production of some sorts of Events, and many calculate what Probability there is, that those Events should be rather be owing to the one than to the other.

(Moivre 1718)

Definition 5.14 (Frequentist Probability). Defines probabilities using long-run limits of frequencies from repeated independent sampling generated by a hypothetical infinite sequence of experiments from a population

Frequentist probability or *frequentism* is an interpretation of probability; it defines an event's probability as the limit of its relative frequency in many trials AKA *long-run probability*. Probabilities can be found, in principle, by a repeatable objective process and are thus ideally devoid of opinion. The continued use of frequentist methods in scientific inference, however, has been called into question.

1. Since in reality we cannot repeat most experiments many times.
2. "by definition, scientific researchers do not possess sufficient knowledge about the relevant and irrelevant aspects of their tests and

populations to be sure that their replications will be equivalent to one another" - Mark Rubin 2020

Definition 5.15 (Bayesian Probability). Defines probability starting with a subjective view of the problem called a prior and updates it as evidence comes in using Bayes Rule.

The lesson and assignments test these views with examples - but the division is rather artificial to me. Not that it does not exist, but rather different authors on the subject treat it differently.

5.4 Bayesian Probability and Coherence

A notion of a **fair bet** - one which we would take either way for the same reward.

- **coherence** following the rules of statistics
- **incoherence** or **Dutch book** one would be guaranteed to lose money.

💡 Biographical note on Bruno de Finetti

From the subjective standpoint, no assertion is possible without a priori opinion, but the variety of possible opinions makes problems depending on different opinions interesting.

Bruno de Finetti 1906-1985 was born in Innsbruck (Austria) to an Italian family. He studied mathematics at the University of Trieste, where he developed a keen interest in probability theory and its applications.

After completing his doctoral studies in 1928, de Finetti embarked on a distinguished academic career. His first research work dealt with mathematical biology and was published, in 1926 when he was still an undergraduate. After graduation and up to 1931, he worked in the mathematical office of the Central Italian Agency for Statistics. From 1931-46, de Finetti worked in Trieste at Assicurazioni Generali, one of the most important insurance companies in Italy. In the same period, he lectured at the University of Trieste and the University of Padua.

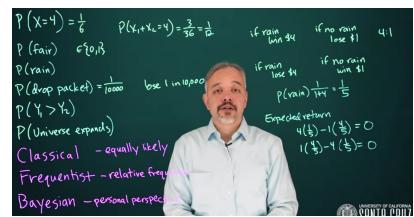


Figure 5.4: Coherence

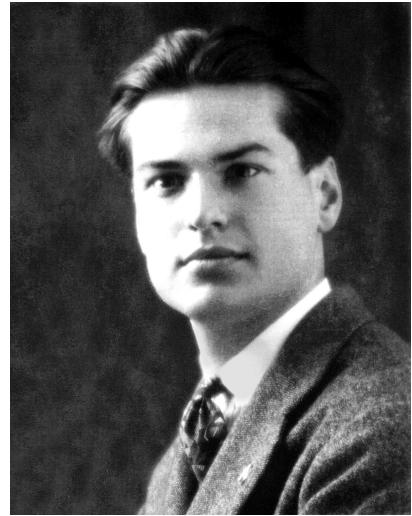


Figure 5.5: Bruno de Finetti

One of de Finetti's most significant contributions was his development of the **theory of subjective probability**, also known as the **Bayesian interpretation of probability**. He developed his ideas independently of [F. P. Ramsey](#) who also published on this (Ramsey 1926)

In his seminal work, (Finetti 1937), he proposed that probability should be interpreted as a personal measure of belief or degree of uncertainty rather than as a frequency or long-run proportion. This subjective approach allowed for the incorporation of prior information and updating of beliefs in light of new data, forming the basis of Bayesian inference.

Probabilistic reasoning – always to be understood as subjective – merely stems from our being uncertain about something. (Finetti 2017 § preface)

It is impossible to summarize in a few paragraphs the scientific activity of de Finetti in the different fields of mathematics (probability), measure theory, analysis, geometry, mathematics of finance, economics, the social sciences, teaching, computer science, and biomathematics or to describe his generous and complex personality as a scientist and a humanitarian. De Finetti discussed his own life in a book edited by Gani (1982). See also the article by Lindley (1989).

My thesis, paradoxically, and a little provocatively, but nonetheless genuinely, is simply this : **PROBABILITY DOES NOT EXIST**. ... Probability, too, if regarded as something endowed with some kind of objective existence, is no less a misleading misconception, an illusory attempt to exteriorize or materialize our true probabilistic beliefs. (Finetti 2017 § preface page x)

de Finetti was a brilliant statistician but his books and papers have garnered a reputation of being challenging to read both in the original Italian, French and English translation. The above quote embodies his radical point of view which he challenged other statisticians to rethink their views.

What I think he meant is that meant primarily was that probabilities unlike physical quantities cannot be measured in the objective sense.

de Finetti was well versed with quantum mechanics, where physical quantities like the position and speed of an electron are interpreted primarily through probabilities in a wave equation, to include a discussion in the start of his second volume.

A large part of this course is that we are inferring parameters - which are often probabilities.

Another milestone result by de Finetti is his theorem

i Question

Representing uncertainty with probability: **Don't use any outside information on this question, just determine probabilities subjectively.** The country of Chile is divided into 15 administrative regions. The size of the country is 756,096 square kilometers. How big do you think the region of Atacama is? Let:

- A_1 be the event: Atacama is less than 10,000 km^2 .
- A_2 be the event: Atacama is between 10,000 and 50,000 km^2
- A_3 be the event: Atacama is between 50,000 and 100,000 km^2
- A_4 be the event: Atacama is more than 100,000 km^2 Assign probabilities to $A_1 \dots A_4$

- What do I know at this point?
 - The expected area for the region is $\frac{750,000}{15} = 50,000 km^2$.
- What Do I believe?
 - I believe that the administrative regions have significantly different sizes
 - from my familiarity with some other countries.
 - As I don't know if Atacama is large or small my best *bet* is to assign equal probabilities to each event.

Event	Min km^2	Max km^2	P
A_1	0	10k	$\frac{1}{4}$
A_2	10k	50k	$\frac{1}{4}$
A_3	50k	100k	$\frac{1}{4}$
A_4	100k		$\frac{1}{4}$

i More information 1

Atacama is the fourth largest of 15 regions. Using this information, I revised my probabilities as follows:

- What do I know?

- The expected area is $\frac{750,000}{15} = 50,000 \text{ km}^2$.
- I know that Atacama is the Fourth largest.

- What Do I believe?

- I believe that the administrative regions have significantly different sizes - from my familiarity with some other countries.

- How do I revise my guesstimate?

- If the region sizes are equally sized I should gamble mostly on A_2 and A_3 .
- But I think there are a few large regions and many smaller ones.
- Also I know that 11 regions are smaller than Atacama and that 3 three are larger.
- None of the events can yet be ruled out.
- A_1 seems extremely unlikely as it necessitates the top three regions account for almost all of the area of the country.
 $\frac{750,000 - 14*10,000}{3} = 203,333.3$ that's about 4 times the average for each state.
- A_2 is fairly unlikely to require the top three regions to account for $\frac{(750,000 - 14*20000)}{3} = 170,000$ each that's more than 3 times the average.

i More information 2

The smallest region is the capital region, Santiago Metropolitan, which has an area of $15,403 \text{ km}^2$. Using this information, I revised my probabilities as follows:

What do I know?

- The expected area is $\frac{750,000}{15} = 50,000 \text{ km}^2$

Event	Min km^2	Max km^2	P
A_1		10k	$\frac{1}{16}$
A_2	10k	50k	$\frac{3}{16}$
A_3	50k	100k	$\frac{6}{16}$
A_4	100k		$\frac{6}{16}$

Event	Min km^2	Max km^2	P
A_1		10k	0
A_2	10k	50k	$\frac{1}{8}$
A_3	50k	100k	$\frac{4}{8}$
A_4	100k		$\frac{3}{8}$

- $\Pr(A_1) = 0$ since the smallest region is \$ 15,403 km².
- I believe that the administrative regions have significantly different sizes - from my familiarity with some other countries.
- I know that Atacama is the Fourth largest.
 - If the region sizes are equally sized I should gamble mostly on A_2 and A_3 .
 - But I think there are a few large regions and many smaller ones.
 - Also I know that 11 regions are smaller than Atacama and that 3 three are larger.
 - None of the events can yet be ruled out. But A_1 and A_2 are now very unlikely as they would require the top three regions to account for almost all of the area of the country.

i More information 3

The third largest region is Aysén del General Carlos Ibáñez del Campo, which has an area of 108,494 km².

Using this information, I revised my probabilities as follows:

- The expected area is $\frac{750,000}{15} = 50,000 \text{ km}^2$
- $\Pr(A_1) = 0$ since the smallest region is \$15,403 km².
- I believe that the administrative regions have significantly different sizes - from my familiarity with some other countries.
- I know that Atacama is the Fourth largest.
 - If the region sizes are equally sized I should gamble mostly on A_2 and A_3 .
 - But I think there are a few large regions and many smaller ones.
 - Also I know that 11 regions are smaller than Atacama and that 3 three are larger.
 - None of the events can yet be ruled out. But A_1 and A_2 are now very unlikely as they would require the top three regions to account for almost all of the area of the country.

Event	Min km ²	Max km ²	P
A_1		10K	0
A_2	10k	50K	$\frac{1}{8}$
A_3	50k	100K	$\frac{1}{8}$
A_4	100k		$\frac{1}{8}$

5.5 Discussions: Objectivity

💡 Discussion: Objectivity

In what ways could the frequentist paradigm be considered objective? In what ways could the Bayesian paradigm be considered objective? Identify ways in which each paradigm might be considered subjective.

- Frequentist:

- The orthodox approach is statisticians should establish an objective statistical methodology and field researchers should then use it to solve their problems. This leads to following flow charts for analysis and tests without fully understanding the model and how it works. At best one makes mistakes due to misunderstanding. But we can see that there is a systematic gaming of this methodology using *p-hacking*, *multiple hypotheses*, and *hiding failed experiments* leading to the publication of outrageously good results, which then cannot be replicated.
- The analysis is done on data that is supposedly sampled from a population. But the same data may belong to different populations (the city, the country, etc) each with different statistics. We should assume the same long-run frequencies would converge to different to each one of these statistics if we repeat the experiment enough times.
- The sample size, or how long we run the experiment is a tricky decision to make in advance and without prior knowledge. And if we do not decide in advance, but periodically as the data comes in. It turns out that this can completely change the outcomes of the experiment - even if both approaches have the same data.
- The choice of H_0 and H_1 is often subjective and each hypothesis can lead to yet another.
- The choice of the confidence level 95%, 99%, etc. used for statistical significance is subjective.
- If an effect size is considered large is subjective and de-

pends on the field one studies.

- Bayesian:

- the prior should be highly informative and therefore **subjective**. But it can be
- uninformative and hence more objective.
- it can be difficult to decide what impact the prior should have on the posterior. Ideally, we can quantify the effective sample size for the prior data and we can understand how much information each contributes to the posterior.

5.6 Expected values

The *expectation* of an RV is a measure of its central tendency.

The expected value, also known as the expectation or mean, of a random variable X is denoted $\mathbb{E}[X]$. It is the *weighted average* of all values X could take, weighted by their probabilities.

💡 Tip

I looked this up and found the following answer, see (<https://math.stackexchange.com/users/25097/autolatry>) (n.d.).

The RV X is a function whereas the Expectation is a Functional (a mapping from a function to a number). Mathematicians adopt the use of square brackets for functionals.

See Wikipedia contributors (2023b) for more information on what a Functional is.

Why Square Brackets for Expectation

5.6.1 Expectation of a discrete random variable

If X is a discrete-valued random variable then its expectation is defined by(?)**@eq-expectation-discrete-RV**

$$\mathbb{E}[X] = \sum_{i=1}^N x_i \cdot \Pr(X = x_i) = \sum_{i=1}^N x_i \cdot f(x) \quad (5.23)$$

where $f(x)$ is the probability mass function (PMF) of X .

5.6.2 Expectation of a continuous random variable

If X is a continuous random variable then its expectation is defined by(?)**expectation-continuous-RV**

$$\mathbb{E}[X] = \int_{-\infty}^{\infty} x \cdot f(x) dx \quad (5.24)$$

while the *mean* is an important descriptive statistic for central tendencies, we often prefer the *median* which is robust to outliers, and pick the *mode* as a representative if we need a value in the data set.

5.6.3 Properties of Expectation

Sum and integral are linear operators so the Expectation is also a linear operator

$$\mathbb{E}[c] = c \quad (5.25)$$

$$\mathbb{E}[aX + bY] = a\mathbb{E}[X] + b\mathbb{E}[Y] \quad (5.26)$$

$$\mathbb{E}[g[X]] = \int g(x)f(x)dx \quad (5.27)$$

where $g[X]$ is a function of the random variable X .

If X & Y are independent

$$\mathbb{E}[XY] = \mathbb{E}[X]\mathbb{E}[Y] \quad (5.28)$$

5.7 Variance

Variance is the dispersion of a distribution about the mean.

Definition 5.16. For a discrete random variable, the Variance is defined using (Equation 5.29)

$$\mathbb{V}ar(X) = \sum_{i=1}^N (x_i - \mu)^2 \mathbb{P}r(X = x_i) \quad (5.29)$$

Definition 5.17. For a continuous random variable, the Variance is defined using (Equation 5.30)

$$\mathbb{V}ar[X] = \int_{-\infty}^{\infty} (x - \mu)^2 f(x) dx \quad (5.30)$$

5.7.1 Properties of Variance

$$\mathbb{V}ar[c] = 0 \quad (5.31)$$

if X and Y are independent then

$$\mathbb{V}ar[aX + by] = a^2 \mathbb{V}ar[X] + b^2 \mathbb{V}ar[Y] \quad (5.32)$$

otherwise

$$\mathbb{V}ar[aX + by] = a^2 \mathbb{V}ar[X] + b^2 \mathbb{V}ar[Y] + 2ab \mathbb{C}ov(X, Y) \quad (5.33)$$

where $\mathbb{C}ov(X, Y)$ is the covariance of X and Y .

Here is one of the most useful identities (Equation 5.34) for wrangling with variance using the expectation of X and X^2 .

$$\begin{aligned} \mathbb{V}ar[X] &= \mathbb{E}[(X - \mathbb{E}[X])^2] \\ &= \mathbb{E}[X^2] - (\mathbb{E}[X])^2 \end{aligned} \quad (5.34)$$

5.8 Covariance

Covariance is a measure of the joint variability of two random variables. It indicates the direction of the linear relationship between the variables.

If X and Y are two random variables, the covariance of X and Y is defined as:

$$\begin{aligned}\text{Cov}(X, Y) &= \mathbb{E}[(X - \mathbb{E}[X])(Y - \mathbb{E}[Y])] \\ &= \mathbb{E}[XY] - \mathbb{E}[X]\mathbb{E}[Y]\end{aligned}\tag{5.35}$$

5.9 Correlation

Correlation is a standardized measure of the linear relationship between two random variables. It is a dimensionless quantity that ranges from -1 to 1.

The correlation coefficient ρ_{XY} is defined as the covariance of X and Y divided by the product of their standard deviations:

$$\rho_{XY} = \frac{\text{Cov}(X, Y)}{\sigma_X \sigma_Y}\tag{5.36}$$

6 Bayes' Theorem

Bayesian Statistics: From Concept to Data Analysis

6.1 Bayes' Theorem

6.2 Conditional Probability

$$\Pr(A | B) = \frac{\Pr(A \cap B)}{\Pr(B)} \quad (6.1)$$

independence

$$\Pr(A | B) = \Pr(A) \implies \Pr(A \cap B) = \Pr(A)\Pr(B) \quad (6.2)$$

6.2.1 Conditional Probability Example - Female CS Student

Suppose there are 30 students, 9 of whom are female. Of the 30 students, 12 are computer science majors. 4 of those 12 computer science majors are female.

We want to estimate what is the probability of a student being female given that she is a computer science major

We start by writing the above in the language of probability by converting frequencies to probabilities. We start with the marginal.

First, the probability of a student being female from the data given above.

$$\Pr(Female) = \frac{9}{30} = \frac{3}{10}$$

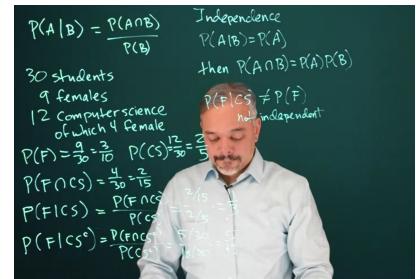


Figure 6.1: conditional probability

Next, we estimate the probability of a student being a computer science major again just using the data given above.

$$\Pr(CS) = \frac{12}{30} = \frac{2}{5}$$

Next, we can estimate the joint probability, i.e. the probability of being female and being a CS major. Again we have been given the numbers in the data above.

$$\Pr(F \cap CS) = \frac{4}{30} = \frac{2}{15}$$

Finally, we can use the definition of conditional probability and substitute the above

$$\Pr(F | CS) = \frac{\Pr(F \cap CS)}{\Pr(CS)} = \frac{2/15}{2/5} = \frac{1}{3}$$

An intuitive way to think about a conditional probability is that we're looking at a sub-segment of the original population, and asking a probability question within that segment

$$\Pr(F | CS^c) = \frac{\Pr(F \cap CS^c)}{\Pr(CS^c)} = \frac{5/30}{18/30} = \frac{5}{18}$$

The concept of **independence** is when one event does not depend on another.

$$A \perp\!\!\!\perp B \iff \Pr(A | B) = \Pr(A)$$

It doesn't matter that B occurred.

If two events are independent then the following is true:

$$A \perp\!\!\!\perp B \iff \Pr(A \cap B) = \Pr(A)\Pr(B) \quad (6.3)$$

This can be derived from the conditional probability equation.

6.2.2 Inverting Conditional Probabilities

If we don't know $\Pr(A | B)$ but we do know the inverse probability $\Pr(B | A)$ is. We can then rewrite $\Pr(A | B)$ in terms of $\Pr(B | A)$

$$\Pr(A | B) = \frac{\Pr(B | A)\Pr(A)}{\Pr(B | A)\Pr(A) + \Pr(B | A^c)\Pr(A^c)} \quad (6.4)$$

6.2.3 Conditional Probability Example - ELISA HIV test

Let's look at an example of an early test for HIV antibodies known as the ELISA test. - The test has a true positive rate of 0.977. - It has a true negative rate of 0.926. - The incidence of HIV in North America is .0026.

Now we want to know the probability of an individual having the disease given that they tested positive $\Pr(HIV|+)$.

This is the inverse probability of the true positive, so we will need to use Bayes' theorem.

We start by encoding the above using mathematical notation, so we know what to substitute into Bayes' theorem.

The true positive rate is:

$$\Pr(+) | HIV = 0.977$$

The true negative rate is:

$$\Pr(- | NO_HIV) = 0.926$$

The probability of someone in North America having this disease was

$$\Pr(HIV) = .0026$$

what we want is: $\Pr(HIV | +)$

$$\begin{aligned}
\Pr(HIV | +) &= \frac{\Pr(+) | HIV) \Pr(HIV)}{\Pr(+) | HIV) \Pr(HIV) + \Pr(+) | NO_HIV) \Pr(NO_HIV)} \\
&= \frac{(.977)(.0026)}{(.977)(.0026) + (1 - .977)(1 - .0026)} \\
&= 0.033
\end{aligned}$$

This is a bit of a **surprise** - although the test has 90% + true and false accuracy - taking it once is only valid 3% of the time. How is this possible?

What happens in Bayes law is that we are updating probabilities. And since we started with such a low probability of .0026, Bayesian updating only brings it up to 0.03.

$$\Pr(A | B) = \frac{\Pr(B | A_1)(A_1)}{\sum_{i=1}^n \Pr(B | A_i)\Pr(A_i)}$$

Note: (McElreath (2015)) discusses how this can be presented less surprisingly.

6.3 Bayes' theorem (Video)

Here are a few formulations of Bayes' theorem. We denote H for our hypothesis and E as our evidence i.e. the data!

We start by using the definition of conditional probability:

$$\Pr(A | B) = \frac{\Pr(A \cap B)}{\Pr(B)} \quad (\text{conditional probability})$$

$$\begin{aligned}
P(A|B) &= \frac{P(B|A)P(A)}{P(B|A)P(A) + P(B|A')P(A')} = \frac{P(A \cap B)}{P(B)} & P(+ | HIV) = 0.977 \\
& & P(- | no HIV) = 0.026 \\
& & P(HIV) = 0.0026 \\
P(CS|F) &= \frac{P(F|CS)P(CS)}{P(F|CS)P(CS) + P(F|CS')P(CS')} & P(HIV|+) = \frac{P(+ | HIV)P(HIV)}{P(+ | HIV)P(HIV) + P(- | HIV)P(HIV)} \\
& & P(+ | HIV) = 0.977 \\
& & P(- | HIV) = 0.026 \\
& & P(HIV|+) = \frac{(0.977)(0.0026)}{(0.977)(0.0026) + (1 - 0.977)(1 - 0.0026)} \\
& & = 0.033
\end{aligned}$$

Figure 6.2: Bayes theorem

$$\begin{aligned}
 \Pr(H|E) &= \frac{\Pr(H \cap E)}{\Pr(E)} \\
 &= \frac{\underbrace{\Pr(H)}_{\text{Marginal Evidence}} \cdot \underbrace{\Pr(E|H)}_{\text{Likelihood}}}{\Pr(E)} \\
 &= \frac{\underbrace{\Pr(H)}_{\text{Prior}} \cdot \underbrace{\Pr(E|H)}_{\text{Likelihood}}}{\underbrace{\Pr(E|H)\Pr(H) + \Pr(E|H^c)\Pr(H^c)}_{\text{Marginal Evidence}}}
 \end{aligned}$$

We can extend Bayes theorem to cases with multiple mutually exclusive events:

if $H_1 \dots H_n$ are mutually exclusive events that sum to 1:

$$\begin{aligned}
 \Pr(H_1 | E) &= \frac{\Pr(E | H)\Pr(H_1)}{\Pr(E | H_1)\Pr(H_1) + \dots + \Pr(E | H_n)\Pr(H_n)} \\
 &= \frac{\Pr(E | H)\Pr(H_1)}{\sum_{i=1}^N \Pr(E | H_i)\Pr(H_i)}
 \end{aligned}$$

where we used the law of total probability in the denominator

if $\{B_i\}$ is a finite or countably finite partition of a sample space then

$$\Pr(A) = \sum_{i=1}^N \Pr(A \cup B_i) = \sum_{i=1}^N \Pr(A | B_i)\Pr(B_i)$$

$$P(H | E) = \frac{\Pr(H) \times \Pr(E | H)}{\Pr(E)}$$

$$\Pr(\text{Unknown} | \text{Data}) = \frac{\Pr(\text{Unknown}) \times \Pr(\text{Data} | \text{Unknown})}{\Pr(E)}$$

Average likelihood

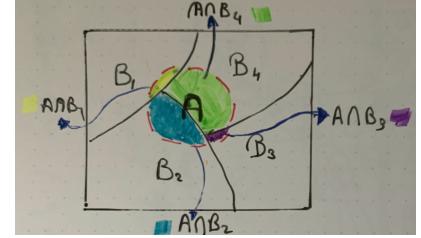


Figure 6.3: mutually exclusive events

The following is a video explaining Bayes law.

6.4 Bayes' Theorem for continuous distributions

When dealing with a continuous random variable θ , we can write the conditional density for θ given y as:

$$f(\theta | y) = \frac{f(y | \theta)f(\theta)}{\int f(y | \theta)f(\theta)d\theta} \quad (6.5)$$

This expression does the same thing that the versions of Bayes' theorem from Lesson 2 do. Because θ is continuous, we integrate over all possible values of θ in the denominator rather than take the sum over these values. The continuous version of Bayes' theorem will play a central role from Lesson 5 on.

💡 Historical Note on The Reverend Thomas Bayes

Bayes Rule is due to Thomas Bayes (1701-1761) who was an English statistician, philosopher and Presbyterian minister. Although Bayes never published what would become his most famous accomplishment; his notes were edited and published posthumously by Richard Price.



Figure 6.4: Rev. Thomas Bayes by Mark Riehl

7 Distributions

Bayesian Statistics: From Concept to Data Analysis

7.1 Distributions

7.2 The Bernoulli & Binomial Distribution

These two distributions are built on a trial of a coin toss (possibly biased).

- We use the Bernoulli distribution to model a random variable for the probability of such a coin toss trial.
- We use the Binomial distribution to model a random variable for the probability of getting k heads in N independent trials.

7.2.1 The Bernoulli Distribution

Arises when modeling events with two possible outcomes, **Success** and **Failure** for a coin toss these can be **Heads** and **Tails**

$$X \sim \text{Bernoulli}(p) = \begin{cases} \Pr(X = 1) = p & \text{success} \\ \Pr(X = 0) = 1 - p & \text{failure} \end{cases} \quad (7.1)$$

Where parameter p is the probability of getting heads.

The probability for the two events is:

Notation:

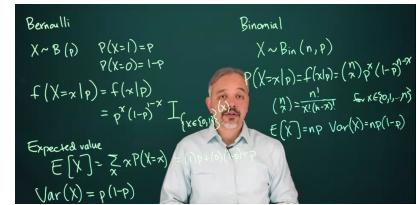


Figure 7.1: Bernoulli and Binomial Distributions

- we use (Roman) p if its value is known.

- we use (Greek) θ when its value is unknown.

This is a probability mass function since it is discrete. But we call it a Probability Density Function (PDF) in the measure-theoretic sense.

$$f(X = x | p) = p^x(1 - p)^{x'} \mathbb{I}_{[0,1]}(x) \quad (7.2)$$

$$\mathbb{E}(x) = p \quad (7.3)$$

$$\text{Var}(x) = \mathbb{P}r(1 - p) \quad (7.4)$$

```
import numpy as np
from scipy.stats import bernoulli
import matplotlib.pyplot as plt

fig, ax = plt.subplots(1, 1)
p = 0.3
mean, var, skew, kurt = bernoulli.stats(p, moments='mvsk')
print(f'{mean=:1.2f}, {var=:1.2f}, {skew=:1.2f}, {kurt=:1.2f}')

mean=0.30, var=0.21, skew=0.87, kurt=-1.24

x = np.arange(bernoulli.ppf(0.01, p),
              bernoulli.ppf(0.99, p))
ax.plot(x, bernoulli.pmf(x, p), 'bo', ms=8, label='bernoulli pmf')
ax.vlines(x, 0, bernoulli.pmf(x, p), colors='b', lw=5, alpha=0.5)

rv = bernoulli(p)
ax.vlines(x, 0, rv.pmf(x), colors='k', linestyles='-', lw=1,
           label='frozen pmf')
ax.legend(loc='best', frameon=False)
plt.show()
```

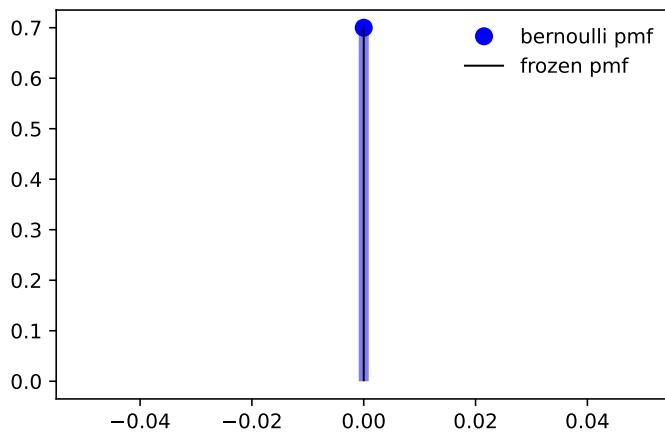


Figure 7.2: Bernoulli distribution

```
## Generate random numbers
r = bernoulli.rvs(p, size=10)
r

array([0, 0, 1, 1, 1, 0, 0, 0, 0])
```

💡 Biographical note on Jacob Bernoulli

It seems that to make a correct conjecture about any event whatever, it is necessary to calculate exactly the number of possible cases and then to determine how much more likely it is that one case will occur than another. (Bernoulli 1713)

The Bernoulli distribution as well as The Binomial distribution are due to Jacob Bernoulli (1655-1705) who was a prominent mathematicians in the Bernoulli family. He discovered the fundamental mathematical constant e. However, his most important contribution was in the field of probability, where he derived the first version of the law of large numbers.

for a fuller [biography see](#)



Figure 7.3: Jacob Bernoulli

7.2.2 The Binomial Distribution

$$\underbrace{[0 \dots 0]}_{N_0} \quad \underbrace{[1 \dots 1]}_{N_1} \quad N \quad (7.5)$$

The Binomial distribution models counts of successes in independent Bernoulli trials . It arises when we need to consider the summing N independent and identically distributed Bernoulli RV with the same probability of success θ .

💡 Conditions

- Discrete data
- Two possible outcomes for each trial
- Each trial is independent
- The probability of success/failure is the same in each trial

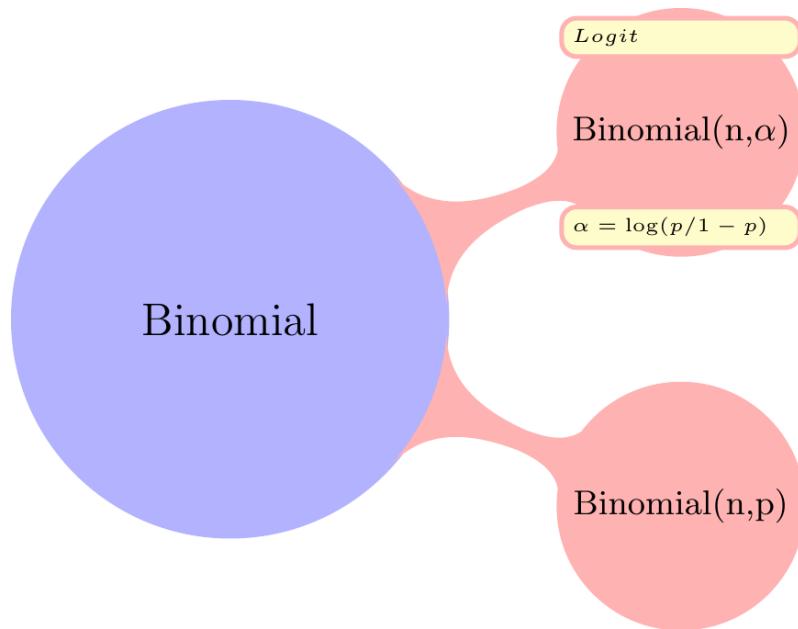


Figure 7.4: Binomial reparams mindmap

$$X \sim Bin[n, p] \quad (7.6)$$

the probability function

$$f(X = x | \theta) = \binom{n}{x} \theta^x (1 - \theta)^{n-x} \quad (7.7)$$

$$\mathcal{L}(\theta) = \prod_{i=1}^n \binom{n}{x_i} \theta^{x_i} (1 - \theta)^{(n-x_i)} \quad (7.8)$$

$$\begin{aligned} \ell(\theta) &= \log \mathcal{L}(\theta) \\ &= \sum_{i=1}^n \left[\log \binom{n}{x_i} + x_i \log \theta + (n - x_i) \log(1 - \theta) \right] \end{aligned} \quad (7.9)$$

$$\mathbb{E}[X] = N \times \theta \quad (7.10)$$

$$\text{Var}[X] = N \cdot \theta \cdot (1 - \theta) \quad (7.11)$$

$$\mathbb{H}(X) = \frac{1}{2} \log_2 (2\pi n \theta (1 - \theta)) + O\left(\frac{1}{n}\right) \quad (7.12)$$

$$\mathcal{I}(\theta) = \frac{n}{\theta \cdot (1 - \theta)} \quad (7.13)$$

7.2.2.1 Relationships

The Binomial Distribution is related to:

- the **Geometric distribution**,
- The **Multinomial distribution** with two categories is the binomial.
- the **Poisson distribution** distribution. If $X \sim \text{Binomial}(n, p)$ rv and $Y \sim \text{Poisson}(np)$ distribution then $\Pr(X = n) \approx \Pr(Y = n)$ for large n and small np .
- the **Bernoulli distribution** If $X \sim \text{Binomial}(n, p)$ RV with $n = 1$, $X \sim \text{Bernoulli}(p)$ RV.

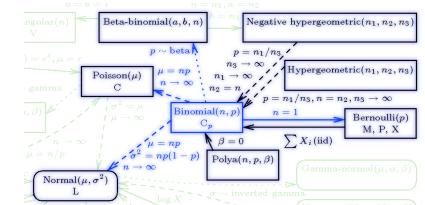


Figure 7.5: binomial distribution relations

- the **Normal distribution** If $X \sim \text{Binomial}(n, p)$ RV and $Y \sim \text{Normal}(\mu = np, \sigma = n\sqrt{p(1-p)})$ then for integers j and k , $\Pr(j \leq X \leq k) \approx \Pr(j - \frac{1}{2} \leq Y \leq k + \frac{1}{2})$. The approximation is better when $p \approx 0.5$ and when n is large. For more information, see normal approximation to binomial
- **Hypergeometric:** The difference between a binomial distribution and a hypergeometric distribution is the difference between sampling with replacement and sampling without replacement. As the population size increases relative to the sample size, the difference becomes negligible. So If $X \sim \text{Binomial}(n, p)$ RV and $Y \sim \text{HyperGeometric}(N, a, b)$ then

$$\lim_{n \rightarrow \infty} X = Y$$

```

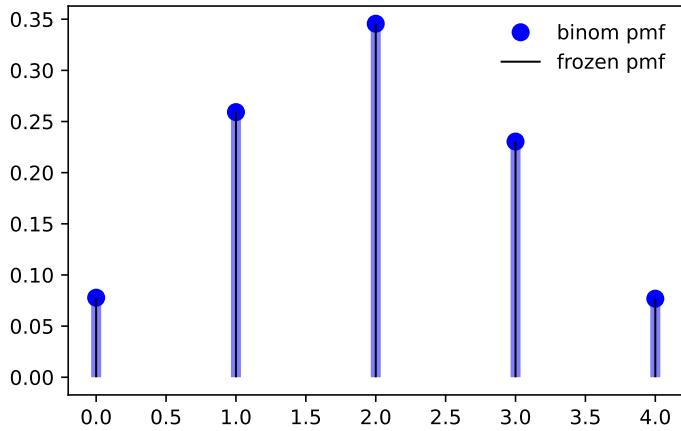
import numpy as np
from scipy.stats import binom
import matplotlib.pyplot as plt

fig, ax = plt.subplots(1, 1)
n, p = 5, 0.4
mean, var, skew, kurt = binom.stats(n, p, moments='mvsk')
print(f'{mean=:1.2f}, {var=:1.2f}, {skew=:1.2f}, {kurt=:1.2f}')

mean=2.00, var=1.20, skew=0.18, kurt=-0.37

x = np.arange(binom.ppf(0.01, n, p), binom.ppf(0.99, n, p))
ax.plot(x, binom.pmf(x, n, p), 'bo', ms=8, label='binom pmf')
ax.vlines(x, 0, binom.pmf(x, n, p), colors='b', lw=5, alpha=0.5)
rv = binom(n, p)
ax.vlines(x, 0, rv.pmf(x), colors='k', linestyles='-', lw=1,
           label='frozen pmf')
ax.legend(loc='best', frameon=False)
plt.show()

```



```
## generate random numbers
r = binom.rvs(n, p, size=10)
r

array([3, 4, 2, 3, 1, 5, 1, 4, 2, 0])
```

7.2.3 The Discrete Uniform Distribution

$$X \sim U[0, 1] \quad (7.14)$$

$$f(x) = \begin{cases} 1, & \text{if } x \in [0, 1] \\ 0, & \text{otherwise} \end{cases} = \mathbb{I}_{\{0 \leq x \leq 1\}}(x) \quad (7.15)$$

```
import numpy as np
from scipy.stats import uniform
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)

n, p = 5, 0.4
mean, var, skew, kurt = uniform.stats(moments='mvsk')
print(f'{mean=:1.2f}, {var=:1.2f}, {skew=:1.2f}, {kurt=:1.2f}')

mean=0.50, var=0.08, skew=0.00, kurt=-1.20
```

```

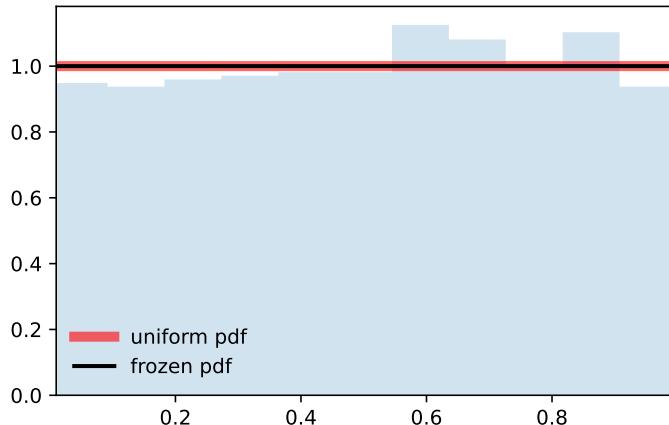
# we use ppf to get the domain from a range of (0.01,0.99)
x = np.linspace(uniform.ppf(0.01), uniform.ppf(0.99), 100)
ax.plot(x, uniform.pdf(x), 'r-', lw=5, alpha=0.6, label='uniform pdf')
rv = uniform()
ax.plot(x, rv.pdf(x), 'k-', lw=2, label='frozen pdf')

## generate random numbers
r = uniform.rvs(size=1000)

# And compare the histogram:
ax.hist(r, density=True, bins='auto', histtype='stepfilled', alpha=0.2)

(array([0.94868592, 0.93765469, 0.95971715, 0.97074838, 0.98177962,
       0.98177962, 1.12518563, 1.0810607 , 1.00384208, 1.10312316,
       0.93765469]), array([0.0011543 , 0.09180601, 0.18245771, 0.27310942, 0.36376113,
       0.45441284, 0.54506455, 0.63571626, 0.72636797, 0.81701968,
       0.90767139, 0.99832309]), [

```



7.2.4 The Continuous Uniform Distribution

$$X \sim \text{Uniform}[\theta_1, \theta_2] \quad (7.16)$$

$$f(x) = \frac{1}{\theta_2 - \theta_1} \mathbb{I}_{\{\theta_1 \leq x \leq \theta_2\}}(x) \quad (7.17)$$

7.3 The Normal, Z, t Distributions

The *normal*, AKA *Gaussian distribution* is one of the most important distributions in statistics.

It arises as the limiting distribution of sums (and averages) of random variables. This is due to the Section 45.1. Because of this property, the normal distribution is often used to model the “errors,” or unexplained variations of individual observations in regression models.

7.3.1 The Standard Normal distribution

The standard normal distribution is given by

$$\mathcal{Z} \sim \mathcal{N}[1, 0] \quad (7.18)$$

$$f(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} \quad (7.19)$$

$$\mathbb{E}[\mathcal{Z}] = 0 \quad (7.20)$$

$$\mathbb{V}ar[\mathcal{Z}] = 1 \quad (7.21)$$

7.3.2 The Normal distribution

Now consider $X = \sigma Z + \mu$ where $\sigma > 0$ and μ is any real constant. Then $\mathbb{E}(X) = \mathbb{E}(\sigma Z + \mu) = \sigma \mathbb{E}(Z) + \mu = \sigma \times 0 + \mu = \mu$ and $Var(X) = Var(\sigma^2 + \mu) = \sigma^2 Var(Z) + 0 = \sigma^2 \cdot 1 = \sigma^2$

Then, X follows a normal distribution with mean μ and variance σ^2 (standard deviation σ) denoted as

$$X \sim \mathcal{N}[\mu, \sigma^2] \quad (7.22)$$

$$f(x | \mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \quad (7.23)$$

$$\mathbb{E}[x] = \mu \quad (7.24)$$

$$\mathbb{V}ar[x] = \sigma^2 \quad (7.25)$$

- The normal distribution is symmetric about the mean μ and is often described as a bell-shaped curve.
- Although X can take on any real value (positive or negative), more than 99% of the probability mass is concentrated within three standard deviations of the mean.

The normal distribution has several desirable properties.

One is that if $X_1 \sim \mathcal{N}(\mu_1, \sigma_1^2)$ and $X_2 \sim \mathcal{N}(\mu_2, \sigma_2^2)$ are independent, then $X_1 + X_2 \sim \mathcal{N}(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)$.

Consequently, if we take the average of n Independent and Identically Distributed (IID) normal random variables we have

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i \sim \mathcal{N}\left(\mu, \frac{\sigma^2}{n}\right) \quad (7.26)$$

```

import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)

n, p = 5, 0.4
mean, var, skew, kurt = norm.stats(moments='mvsk')
print(f'{mean=:1.2f}, {var=:1.2f}, {skew=:1.2f}, {kurt=:1.2f}')

mean=0.00, var=1.00, skew=0.00, kurt=0.00

x = np.linspace(norm.ppf(0.01),
                 norm.ppf(0.99), 100)
ax.plot(x, norm.pdf(x),
         'r-', lw=5, alpha=0.6, label='norm pdf')

rv = norm()
ax.plot(x, rv.pdf(x), 'k-', lw=2, label='frozen pdf')
r = norm.rvs(size=1000)

ax.hist(r, density=True, bins='auto', histtype='stepfilled', alpha=0.2)

(array([0.00438231, 0.01314693, 0.00438231, 0.01752924, 0.03944079,
       0.0438231 , 0.10079314, 0.15776318, 0.21473321, 0.24102707,
       0.34182021, 0.35496714, 0.39440794, 0.5039657 , 0.46014259,
       0.40755487, 0.23226245, 0.21473321, 0.23664476, 0.13146931,
       0.10955776, 0.07011697, 0.03944079, 0.02191155, 0.00876462,
       0.00876462, 0.00438231, 0.00438231]), array([-3.1166646 , -2.88847446, -2.66028433, -2.4320942 ,
       2.20390406,
       -1.97571393, -1.7475238 , -1.51933366, -1.29114353, -
       1.0629534 ,
       -0.83476326, -0.60657313, -0.37838299, -0.15019286, 0.07799727,
       0.30618741, 0.53437754, 0.76256767, 0.99075781, 1.21894794,
       1.44713807, 1.67532821, 1.90351834, 2.13170848, 2.35989861,
       2.58808874, 2.81627888, 3.04446901, 3.27265914]), [

```

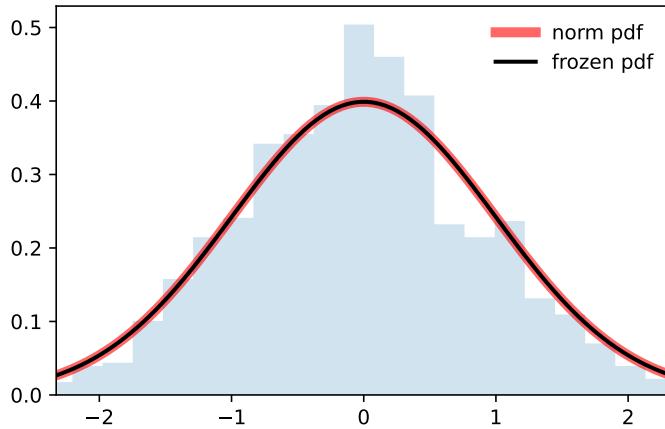
```

ax.set_xlim([x[0], x[-1]])

(-2.3263478740408408, 2.3263478740408408)

ax.legend(loc='best', frameon=False)
plt.show()

```



7.3.3 The t-Distribution

If we have normal data, we can use (Equation 42.32) to help us estimate the mean μ . Reversing the transformation from the previous section, we get

$$\frac{\hat{X} - \mu}{\sigma/\sqrt{n}} \sim N(0, 1) \quad (7.27)$$

However, we may not know the value of σ . If we estimate it from data, we can replace it with $S = \sqrt{\sum_i \frac{(X_i - \hat{X})^2}{n-1}}$, the sample standard deviation. This causes the expression (Equation 42.33) to no longer be distributed as a Standard Normal; but as a standard *t-distribution* with $\nu = n - 1$ degrees of freedom

$$X \sim t[\nu] \quad (7.28)$$

$$f(t | \nu) = \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})\sqrt{\nu\pi}} \left(1 + \frac{t^2}{\nu}\right)^{-(\frac{\nu+1}{2})} \mathbb{I}_{t \in \mathbb{R}} \quad (\text{PDF}) \quad (7.29)$$

where $\Gamma(w) = \int_0^\infty t^{w-1} e^{-t} dt$ is the gamma function

$$f(t | \nu) = \frac{1}{\sqrt{\nu} B(\frac{1}{2}, \frac{\nu}{2})} \left(1 + \frac{t^2}{\nu}\right)^{-(\nu+1)/2} \mathbb{I}_{t \in \mathbb{R}} \quad (\text{PDF}) \quad (7.30)$$

where $B(u, v) = \int_0^1 t^{u-1} (1-t)^{v-1} dt$ is the beta function

$$\mathbb{E}[Y] = 0 \quad \text{if } \nu > 1 \quad (7.31)$$

$$\text{Var}[Y] = \frac{\nu}{\nu - 2} \quad \text{if } \nu > 2 \quad (7.32)$$

The t distribution is symmetric and resembles the Normal Distribution but with thicker tails. As the degrees of freedom increase, the t distribution looks more and more like the standard normal distribution.

7.4 The Exponential Distribution

The **Exponential distribution** models the waiting time between events for events with a rate λ . Those events, typically, come from a **Poisson** process.

The **Exponential distribution** is often used to model the waiting time between random events. Indeed, if the waiting times between successive events are independent then they form an $\exp(r(\lambda))$ distribution. Then for any fixed time window of length t , the number of events occurring in that window will follow a **Poisson distribution** with mean $t\lambda$.

$$X \sim \text{Exp}[\lambda] \quad (7.33)$$

$$f(x | \lambda) = \frac{1}{\lambda} e^{-\frac{x}{\lambda}} (x) \mathbb{I}_{\lambda \in \mathbb{R}^+} \mathbb{I}_{x \in \mathbb{R}_0^+} \quad (\text{PDF}) \quad (7.34)$$

$$\mathbb{E}(x) = \lambda \quad (7.35)$$

$$\mathbb{V}ar[X] = \lambda^2 \quad (7.36)$$

```

import numpy as np
from scipy.stats import expon
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)

n, p = 5, 0.4
mean, var, skew, kurt = expon.stats(moments='mvsk')
print(f'{mean=:1.2f}, {var=:1.2f}, {skew=:1.2f}, {kurt=:1.2f}')


mean=1.00, var=1.00, skew=2.00, kurt=6.00

x = np.linspace(expon.ppf(0.01), expon.ppf(0.99), 100)
ax.plot(x, expon.pdf(x), 'r-', lw=5, alpha=0.6, label='expon pdf')

rv = expon()
ax.plot(x, rv.pdf(x), 'k-', lw=2, label='frozen pdf')

r = expon.rvs(size=1000)

ax.hist(r, density=True, bins='auto', histtype='stepfilled', alpha=0.2)

(array([0.95628696, 0.6652431 , 0.60056669, 0.46659412, 0.38343873,
       0.31876232, 0.24484642, 0.21250821, 0.15707129, 0.11549359,
       0.09239488, 0.06467641, 0.06467641, 0.04619744, 0.05081718,
       0.01847898, 0.04619744, 0.02309872, 0.01385923, 0.01847898,
       0.00461974, 0.00461974, 0.01847898, 0.          , 0.          ,
       0.00461974, 0.          , 0.00461974, 0.00461974, 0.00923949,
       0.          , 0.          , 0.          , 0.          , 0.00461974,
       0.          , 0.00461974]), array([8.79096051e-05, 2.16550131e-01, 4.33012352e-01, 6.49474573e-01,
       8.65936794e-01, 1.08239901e+00, 1.29886124e+00, 1.51532346e+00,
       1.73178568e+00, 1.94824790e+00, 2.16471012e+00, 2.38117234e+00,
       2.59763456e+00, 2.81409678e+00, 3.03055900e+00, 3.24702122e+00,
       3.46388531e+00, 3.68075862e+00, 3.89763193e+00, 4.11450524e+00,
       4.33137855e+00, 4.54825186e+00, 4.76512517e+00, 4.98199848e+00,
       5.19887179e+00, 5.41574510e+00, 5.63261841e+00, 5.84949172e+00,
       6.06636503e+00, 6.28323834e+00, 6.50011165e+00, 6.71698496e+00,
       6.93385827e+00, 7.15073158e+00, 7.36760489e+00, 7.58447820e+00,
       7.80135151e+00, 8.01822482e+00, 8.23509813e+00, 8.45197144e+00,
       8.66884475e+00, 8.88571806e+00, 9.10259137e+00, 9.31946468e+00,
       9.53633799e+00, 9.75321130e+00, 9.97008461e+00, 1.01888587e+01,
       1.04076328e+01, 1.06264069e+01, 1.08451809e+01, 1.10639549e+01,
       1.12827289e+01, 1.15014929e+01, 1.17202569e+01, 1.19390209e+01,
       1.21577849e+01, 1.23765489e+01, 1.25953129e+01, 1.28140769e+01,
       1.30328409e+01, 1.32516049e+01, 1.34703689e+01, 1.36891329e+01,
       1.39078969e+01, 1.41266609e+01, 1.43454249e+01, 1.45641889e+01,
       1.47829529e+01, 1.50017169e+01, 1.52204809e+01, 1.54392449e+01,
       1.56580089e+01, 1.58767729e+01, 1.60955369e+01, 1.63143009e+01,
       1.65330649e+01, 1.67518289e+01, 1.69705929e+01, 1.71893569e+01,
       1.74081209e+01, 1.76268849e+01, 1.78456489e+01, 1.80644129e+01,
       1.82831769e+01, 1.85019409e+01, 1.87207049e+01, 1.89394689e+01,
       1.91582329e+01, 1.93770069e+01, 1.95957709e+01, 1.98145349e+01,
       2.00333089e+01, 2.02520729e+01, 2.04708369e+01, 2.06896009e+01,
       2.09083649e+01, 2.11271289e+01, 2.13458929e+01, 2.15646569e+01,
       2.17834209e+01, 2.20021849e+01, 2.22209489e+01, 2.24397129e+01,
       2.26584769e+01, 2.28772409e+01, 2.30960049e+01, 2.33147689e+01,
       2.35335329e+01, 2.37522969e+01, 2.39710609e+01, 2.41908249e+01,
       2.44095889e+01, 2.46283529e+01, 2.48471169e+01, 2.50658809e+01,
       2.52846449e+01, 2.55034089e+01, 2.57221729e+01, 2.59409369e+01,
       2.61596009e+01, 2.63783649e+01, 2.65971289e+01, 2.68158929e+01,
       2.70346569e+01, 2.72534209e+01, 2.74721849e+01, 2.76909489e+01,
       2.79097129e+01, 2.81284769e+01, 2.83472409e+01, 2.85660049e+01,
       2.87847689e+01, 2.90035329e+01, 2.92223069e+01, 2.94410709e+01,
       2.96598349e+01, 2.98786089e+01, 3.00973729e+01, 3.03161469e+01,
       3.05349109e+01, 3.07536749e+01, 3.09724489e+01, 3.11912129e+01,
       3.14109869e+01, 3.16297509e+01, 3.18485249e+01, 3.20672989e+01,
       3.22860729e+01, 3.25048469e+01, 3.27236209e+01, 3.29423949e+01,
       3.31611689e+01, 3.33800000e+01])

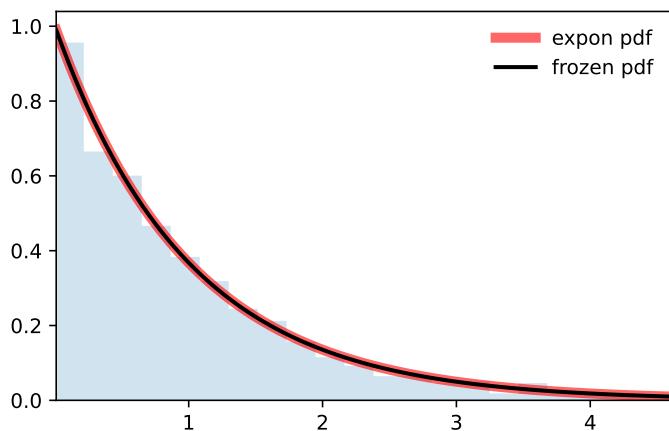
```

```
3.46348345e+00, 3.67994567e+00, 3.89640789e+00, 4.11287011e+00,
4.32933233e+00, 4.54579455e+00, 4.76225677e+00, 4.97871899e+00,
5.19518121e+00, 5.41164343e+00, 5.62810566e+00, 5.84456788e+00,
6.06103010e+00, 6.27749232e+00, 6.49395454e+00, 6.71041676e+00,
6.92687898e+00, 7.14334120e+00, 7.35980342e+00, 7.57626565e+00,
7.79272787e+00, 8.00919009e+00]), [
```

```
ax.set_xlim([x[0], x[-1]])
```

```
(0.010050335853501442, 4.605170185988091)
```

```
ax.legend(loc='best', frameon=False)
plt.show()
```



8 Additional Discrete Distributions

8.1 The Geometric Distribution

The **Geometric distribution** arises when we want to know “What is the number of Bernoulli trials required to get the first success?”, i.e., the number of Bernoulli events until a success is observed, such as the probability of getting the first head when flipping a coin. It takes values on the positive integers starting with one (since at least one trial is needed to observe a success).

$$X \sim \text{Geo}(p) \quad (8.1)$$

$$\Pr(X = x \mid p) = \Pr((1-p)^{x-1} \cdot p) \quad \forall x \in \mathbb{N}; \quad 0 \leq p \leq 1 \quad (8.2)$$

$$\mathbb{E}[X] = \frac{1}{p} \quad (8.3)$$

$$\text{Var}[X] = \frac{1-p}{p^2} \quad (8.4)$$

$$\mathbb{M}_X[t] = \frac{pe^t}{1 - (1-p)e^t} \quad t < -\log(1-p) \quad (8.5)$$

8.2 The Multinomial Distribution

Another generalization of the Bernoulli distribution and the Binomial distribution is the **Multinomial distribution**, which sums the successes of Bernoulli trials when there are n different possible outcomes. Suppose we have n trials and there are k different possible outcomes that occur with probabilities p_1, \dots, p_k . For example, we are rolling a six-sided die that might be loaded so that the sides are not equally likely, then n is the total number of rolls, $k = 6$, p_1 is the probability of rolling a one, and we denote by x_1, \dots, x_6 a possible outcome for the number of times we observe rolls of each of one through six, where

$$X \sim \text{Multinomial}(p_1, \dots, p_k)$$

$$P(X = x | p_1, \dots, p_k) = \frac{n!}{x_1! \cdots x_k!} \prod_i p_i^{x_i}$$

8.3 The Poisson Distribution

The **Poisson distribution** arises when modeling **count** data. The parameter $\lambda > 0$ is the **rate** at which we expect to observe the thing we are counting. We write this as $X \sim \text{Poisson}(\lambda)$

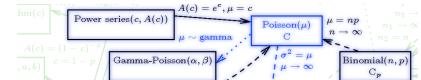
$$\Pr(X = x | \lambda) = \frac{\lambda^x e^{-\lambda}}{x!} \quad \forall x \in \mathbb{N}_0 \quad \text{PDF} \quad (8.6)$$

$$\mathbb{E}[X] = \lambda \quad \text{Expectation} \quad (8.7)$$

$$\text{Var}[X] = \lambda \quad \text{Variance} \quad (8.8)$$

$$\mathbb{M}_X(t) = \exp[\lambda(e^t - 1)] \quad \text{Moment Generating fn.} \quad (8.9)$$

$$\mathcal{I}_X(t) = \frac{1}{\lambda} \quad (8.10)$$



8.3.1 Relations

A *Poisson process* is a process wherein events occur on average at rate \mathbb{E} , events occur one at a time, and events occur independently of each other.

Biographical Note on The Siméon Denis Poisson

The Poisson distribution is due to Baron Siméon Denis Poisson (1781-1840) see (Poisson 2019, 205–7) was a French mathematician and physicist who worked on statistics, complex analysis, partial differential equations, the calculus of variations, analytical mechanics, electricity and magnetism, thermodynamics, elasticity, and fluid mechanics.

for a fuller biography see



Figure 8.2: Siméon Denis Poisson

8.4 Hypergeometric Distribution

Consider an urn with a white balls and b black balls. Draw N balls from this urn without replacement. The number white balls drawn, n is Hypergeometrically distributed.

$$X \sim \text{Hypergeometric}(n \mid N, a, b)$$

$$\text{Hypergeometric}(n \mid N, a, b) = \frac{\binom{a}{n} \binom{b}{N-n}}{\binom{a+b}{N}} \quad (\text{PDF}) \quad (8.11)$$

$$\mathbb{E}[X] = N \frac{a}{a+b} \quad (\text{expectation}) \quad (8.12)$$

$$\text{Var}[X] = N \frac{ab}{(a+b)^2} \frac{a+b-N}{a+b-1} \quad (\text{variance}) \quad (8.13)$$

9 Frequentist Inference

Bayesian Statistics: From Concept to Data Analysis

9.1 Confidence Intervals

A brief review of the frequentist approach to inference will be useful for contrasting with the Bayesian approach. (Kruschke 2011) Chapter 2 suggests that CI provides the basis for a Bayesian workflow and that the rest of the text fills in the missing pieces.

! Frequentist paradigm

Under the **frequentist paradigm**, one views the data as a **random sample** from some larger, potentially **hypothetical population**. We can then make probability statements i.e. **long-run frequency** statements based on this larger population.

Example 9.1 (Coin Flip Example - Central Limit Theorem). Suppose we flip a coin 100 times. And we get **44 heads** and **56 tails**. We can view these 100 flips as a random sample from a much larger infinite hypothetical population of flips from this coin. We can say that each flip is X_i an RV which follows a *Bernoulli distribution* with some probability p . In this case p is unknown, but we can assume it is fixed since we are using a specific physical coin.

$$X_i \sim B(p) \quad (9.1)$$

We ask :

1. What is our best estimate of p the **probability of getting a head?**

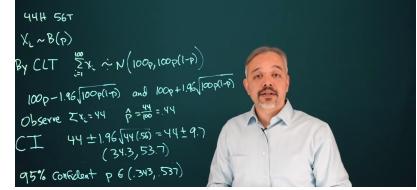


Figure 9.1: frequentist approach to confidence intervals

2. How confident are we in the estimate of p ?

To estimate p we will apply the **Central limit theorem** c.f. Theorem 45.1 which states that the mean of a large number of IID RV with mean μ and variance σ^2 is approximately $N(\mu, \sigma^2)$.

$$\sum_{i=1}^{100} X_i \stackrel{\text{d}}{\sim} N(100 p, 100 \Pr(1-p)) \quad (9.2)$$

Given that this is a **Normal distribution**, we can use the *empirical rule* often called the *68-95-99.7 rule* see (Wikipedia contributors 2023a), that says 95% of the time we will get a result is in within 1.96 standard deviations of the mean. This is referred to as a *Confidence Interval* or (CI).

$$95\% \text{ CI} = 100 \hat{p} \pm 1.96 \sqrt{100 \hat{p}(1 - \hat{p})} \quad (9.3)$$

Since we observed 44 heads we can estimate \hat{p} as

$$\hat{p} = \frac{44}{100} = .44 \quad (9.4)$$

This answers our first questions. Now we want to quantify our uncertainty.

$$\begin{aligned} 95\% \text{ CI for 100 tosses} &= 100 (.44) \pm 1.96 \sqrt{100(0.44)(1 - 0.44)} \\ &= 44 \pm 1.96 \sqrt{(44)(0.56)} \\ &= 44 \pm 1.96 \sqrt{23.64} \\ &= (34.27, 53.73) \end{aligned} \quad (9.5)$$

We can be 95% confident that $100 \times \hat{p} \in [34.3, 53.7]$ We can say that we're 95% confident that the true probability $p \in (.343, .537)$.

If one were to ask **do I think this coin is Fair ?** This is a reasonable hypothesis, since $0.5 \in [.343, .537]$.

But we can also step back and say what does this interval mean when we say we're 95% confident? Under the frequentist paradigm, we have to think back to our infinite hypothetical sequence of events, were we to repeat this trial an arbitrarily large number of times and each time create

a confidence interval, then on average 95% of the intervals we make will contain the true value of p . This makes sense as a long run frequency explanation.

On the other hand, we might want to know something about this particular interval. Does this interval contain the true p ? What's the probability that this interval contains a true p ? Well, we don't know for this particular interval. But under the frequentist paradigm, we're assuming that there is a fixed right answer for p . Either p is in that interval or it's not in that interval. And so technically, from a frequentist perspective, the probability that p is in this interval is either 0 or 1. This is not a particularly satisfying explanation. In the other hand when we get to the Bayesian approach we will be able to compute an interval and actually say there is probably a p is in this interval is 95% based on a random interpretation of an unknown parameter.

Tip

In this example of flipping a coin 100 times, observing 44 heads resulted in the following 95% confidence interval for p : (.343, .537). From this, we concluded that it is plausible that the coin may be fair because $p=.5$ is in the interval.

Suppose instead that we flipped the coin 100,000 times, observing 44,000 heads (the same percentage of heads as before). Then using the method just presented, the 95% confidence interval for p is (.437, .443). **Is it still reasonable to conclude that this is a fair coin with 95% confidence?**

No Because $0.5 \notin (.437, .443)$, we must conclude that $p = .5$ is not a plausible value for the population mean. Observing 100,000 flips increases the power of the experiment, leading to a more precise estimate with a narrower CI, due to the law of large numbers.

9.2 Likelihood function and MLE

Example 9.2 (Heart Attack Patients - MLE). Consider a hospital where 400 patients are admitted over a month for heart attacks, and a month later 72 of them have died and 328 of them have survived.

$$\begin{aligned}
 Y_i &\sim B(\theta) & L(\theta) &= \log L(\theta) \\
 p(Y_i=y_i) &= \theta^{y_i} (1-\theta)^{1-y_i} & l(\theta) &= \log \left[\prod_{i=1}^n \theta^{y_i} (1-\theta)^{1-y_i} \right] \\
 P(Y=y|\theta) &= P(Y_1=y_1, Y_2=y_2, \dots, Y_n=y_n|\theta) & & \\
 &= P(Y_1=y_1) \cdot P(Y_2=y_2|\theta) \cdots P(Y_n=y_n|\theta) & & \\
 &= \prod_{i=1}^n P(Y_i=y_i|\theta) = \prod_{i=1}^n \theta^{y_i} (1-\theta)^{1-y_i} & & \\
 \text{Likelihood } L(\theta) &= \prod_{i=1}^n \theta^{y_i} (1-\theta)^{1-y_i} & & \\
 &= \theta^{72} (1-\theta)^{328} & & \\
 \text{MLE } \hat{\theta} &= \arg\max L(\theta) & &
 \end{aligned}$$

Figure 9.2: Likelihood fn and MLE

what's our estimate of the mortality rate?

Reference Population

Under the *frequentist paradigm*, we must first establish our **reference population**. This is the cornerstone of our thinking as we are considering how the sample parameter approximates the population statistic. What do we think our reference population is here?

- **Ref Pop 1:** Heart attack patients in the region.
- **Ref Pop 2:** Heart attack patients that are admitted to this hospital, but over a longer period.
- **Ref Pop 3:** The people in the region who might have a heart attack and might get admitted to this hospital.

Both *Ref Pop 1* and *Ref Pop 2* seem like viable options. Unfortunately, in our data is not a random sample drawn from either. We could pretend they are and move on, or we could also try to think harder about what our data is sampled from, perhaps *Ref Pop 3*. This is an odd hypothetical situation, and so there are some *philosophical issues* with the setup of this whole problem within the *frequentist paradigm*

$$Y_i \sim Bernoulli(p) \quad (9.6)$$

Since this is a Bernoulli trial we need to specify what we interpret as the *success*. In this case, the *success* is a mortality.

$$\Pr(Y_i = 1) = \theta \quad (9.7)$$

The PDF for the dataset can be written in vector form. $\Pr(\vec{Y} = \vec{y} | \theta)$ is the Probability of all the Y's take some value little y given a value of theta.

$$\begin{aligned} \Pr(\vec{Y} = \vec{y} | \theta) &= \Pr(Y_1 = y_1, \dots, Y_n = y_n | \theta) && \text{(joint probability)} \\ &= \Pr(Y_1 = y_1 | \theta) \dots \Pr(Y_n = y_n | \theta) && \text{(independence)} \\ &= \prod_{i=1}^n \Pr(Y_i = y_i | \theta) && \text{(product notation)} \\ &= \prod_{i=1}^n \theta^{y_i} (1 - \theta)^{1-y_i} && \text{(Bernoulli PMF)} \end{aligned} \quad (9.8)$$

We now cal the expression for $\Pr(\vec{Y} = \vec{y} \mid \theta)$ above the likelihood function $L(\theta \mid \vec{y})$:

$$\mathcal{L}(\theta \mid \vec{y}) = \prod_{i=1}^n \theta^{y_i} (1 - \theta)^{1-y_i} \quad (9.9)$$

Recall that we want to find the mortality rate parameter θ for our Sample \vec{Y} .

Since it is a probability, it has a range of values from 0 to 1. One way to estimate it is that there should be one value that maximizes (Equation 9.9). It makes the data the most likely to occur for the particular data we observed. This is referred to as the **maximum likelihood estimate** (MLE).

$$\text{MLE}(\hat{\theta}) = \operatorname{argmax} \mathcal{L}(\theta \mid y)$$

Although we are trying to find the θ that maximizes the likelihood, in practice, it's usually easier to maximize the natural logarithm of the likelihood, commonly referred to as the log-likelihood.

$$\begin{aligned} \mathcal{L}(\theta) &= \log(L(\theta \mid \vec{y})) \\ &= \log\left(\prod_{i=1}^n \theta^{y_i} (1 - \theta)^{1-y_i}\right) && \text{subst. liklihood} \\ &= \sum_{i=1}^n \log(\theta^{y_i}) + \log(1 - \theta)^{1-y_i} && \text{log product rule} \\ &= \sum_{i=1}^n y_i \log(\theta) + (1 - y_i) \log(1 - \theta) && \text{log power rule} \\ &= \log(\theta) \sum_{i=1}^n y_i + \log(1 - \theta) \sum_{i=1}^n (1 - y_i) && \text{extracting logs} \end{aligned} \quad (9.10)$$

What is the interpretation of the MLE of θ in the context of the heart attack example?

If $\hat{\theta}$ is the MLE for θ , the 30-day mortality rate, then all possible values of θ produce a lower value of the likelihood than $\hat{\theta}$.

To calculate the MLE one should differentiate $\mathcal{L}(\theta)$ w.r.t. θ and then set it equal to 0.

9.3 Computing the MLE

$$\begin{aligned}
\mathcal{L}'(\theta) &= \frac{1}{\theta} \sum_{i=1}^n y_i - \frac{1}{1-\theta} \sum_{i=1}^n 1-y_i \stackrel{\text{set derivative to } 0}{=} 0 \\
\Rightarrow \quad \frac{1}{\hat{\theta}} \sum_{i=1}^n y_i &= \frac{1}{1-\hat{\theta}} \sum_{i=1}^n 1-y_i \\
\Rightarrow \quad (1-\hat{\theta}) \sum_{i=1}^n y_i &= \hat{\theta} \sum_{i=1}^n 1-y_i \\
\Rightarrow \quad 1 \sum_{i=1}^n y_i - \hat{\theta} \sum_{i=1}^n y_i &= \hat{\theta} \sum_{i=1}^n 1 - \hat{\theta} \sum_{i=1}^n y_i \\
\Rightarrow \quad \sum_{i=1}^n y_i &= \hat{\theta} N \\
\Rightarrow \quad \hat{\theta} &= \frac{1}{N} \sum_{i=1}^n y_i = \hat{p} = \frac{72}{400} = .18
\end{aligned}$$

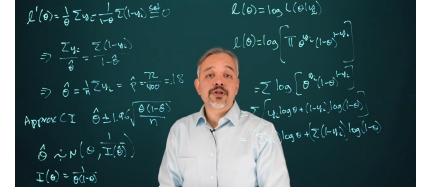


Figure 9.3: Computing the MLE

Maximum Likelihood Estimates (MLEs) possess the following favorable properties:

- **Unbiased** - Thus given sufficient data the MLE will converge to the true value. As a consequence, *MLEs are asymptotically unbiased*. As we will see in the examples they can still be biased in finite samples.
- **consistent** - One important property of maximum likelihood is that it produces consistent estimates.
- **invariant** - The invariance principle states that the *MLE is invariant against reparameterization*.

using the Central Limit theorem (see Theorem 45.1).

$$\hat{\theta} \pm 1.96 \sqrt{\frac{\hat{\theta}(1-\hat{\theta})}{n}}$$

$$\hat{\theta} \simeq \mathcal{N}(\theta, \frac{1}{\mathcal{I}(\hat{\theta})})$$

where \mathcal{I} is the *Fischer information* which for the Bernoulli distribution is:

$$\mathcal{I}(\hat{\theta}) = \frac{1}{\theta(1-\theta)}$$

Note: The *Fischer information* is a measure of how much information about theta is in each data point!

💡 Explainable AI (XAI) & Fischer information

In XAI we use discuss local and global explanations.

- **Global explanations** explain a black box model's predictions based on each feature, via its parameters.
- **Local explanations** explain the prediction of a specific datum from its features.

since *Fischer information* quantifies the information in a data point on a parameter we should be able to use it to produce local and perhaps even global explanations for Bayesian models.

9.4 Computing the MLE: examples

Some more examples of maximum likelihood estimators.

9.4.1 Computing the MLE for Exponential RV

Let's say X_i are exponential distributed

$$X_i \sim \text{Exp}(\lambda)$$

Let's say the data is independent and identically distributed, therefore making the overall density function

$$\begin{aligned} f(x | \lambda) &= \prod_{i=1}^n \lambda e^{-\lambda x_i} \quad (\text{simplifying}) \\ &= \lambda^n e^{-\lambda \sum x_i} \end{aligned} \tag{9.11}$$

Now the likelihood function is

$$L(\lambda | x) = \lambda^n e^{-\lambda \sum x_i} \tag{9.12}$$

the log likelihood is

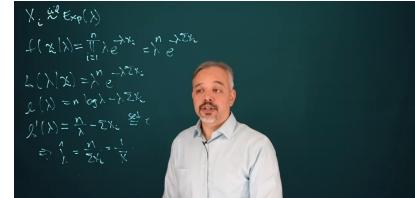


Figure 9.4: computing the MLE for Exponential RV

$$\mathcal{L}(\lambda) = n \ln \lambda - \lambda \sum_i x_i \quad (9.13)$$

Taking the derivative

$$\begin{aligned} \mathcal{L}'(\lambda) &= \frac{n}{\lambda} - \sum_i x_i \stackrel{\text{set}}{=} 0 \quad (\text{set derivative} = 0) \\ \implies \hat{\lambda} &= \sum_i x_i \quad (\text{rearranging}) \end{aligned} \quad (9.14)$$

$$\hat{\lambda} = \frac{n}{\sum_i x_i} = \bar{x} \quad (9.15)$$

9.4.2 Computing the MLE for Uniform RV

$$X_i \sim \text{Uniform}[0, \theta] \quad (9.16)$$

$$f(x | \theta) = \prod_{i=1}^n \frac{1}{\theta} \mathbb{I}_{0 \leq x_i \leq \theta} \quad (9.17)$$

Combining all the indicator functions, for this to be a 1, each of these has to be true. These are going to be true if all the observations are bigger than 0, as in the minimum of the x 's is bigger than or equal to 0. The maximum of the x 's is also less than or equal to θ .

$$\mathcal{L}(\theta | x) = \theta^{-1} \mathbb{I}_{0 \leq \min(x_i) \leq \max(x_i) \leq \theta} \quad (9.18)$$

$$\mathcal{L}'(\theta) = -n\theta^{-(n+1)} \mathbb{I}_{0 \leq \min(x_i) \leq \max(x_i) \leq \theta} \quad (9.19)$$

We ask, can we set this equal to zero and solve for θ ? It turns out, this is not equal to zero for any θ positive value. We need θ to be strictly larger than zero. But for θ positive, this will always be negative. The derivative is negative, that says this is a decreasing function. Therefore this function will be maximized when we pick θ as small as possible. What's the smallest possible value of θ we can pick? Well we need in particular for θ to be larger than all of the X_i . And so, the maximum likelihood estimate is the maximum of X_i

$$\hat{\theta} = \max(x_i) \quad (9.20)$$

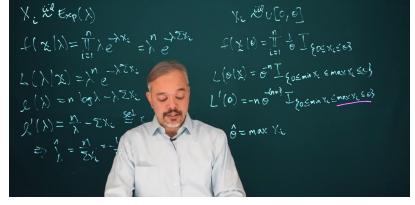


Figure 9.5: computing the MLE for Uniform RV

9.5 Cumulative Distribution Function

The cumulative distribution function (CDF) exists for every distribution. We define it as $F(x) = \Pr(X \leq x)$ for random variable X .

If X is discrete-valued, then the CDF is computed with summation $F(x) = \sum_{t=-\infty}^x f(t)$, where $f(t) = \Pr(X = t)$ is the probability mass function (PMF) which we've already seen.

If X is continuous, the CDF is computed with an integral $F(x) = \int_{-\infty}^x f(t)dt$

The CDF is convenient for calculating probabilities of intervals. Let a and b be any real numbers with $a < b$. Then the probability that X falls between a and b is equal to $\Pr(a < X < b) = \Pr(X \leq b) - \Pr(X \leq a) = F(b) - F(a)$

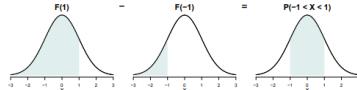


Figure 9.6: Illustration of using the CDF to calculate the probability of an interval for continuous random variable X . Probability values are represented with shaded regions in the graphs.

Example 9.3 (CDF example 1). Suppose $X \sim \text{Binomial}(5, 0.6)$. Then

$$\begin{aligned}
 F(1) &= \Pr(X \leq 1) \\
 &= \sum_{t=-\infty}^1 f(t) \\
 &= \sum_{t=-\infty}^{-1} 0 + \sum_{t=0}^1 \binom{5}{t} 0.6^t (1-0.6)^{5-t} \\
 &= \binom{5}{0} 0.6^0 (1-0.6)^5 - 0 + \binom{5}{1} 0.6^1 (1-0.6)^{5-1} \\
 &= (0.4)^5 + 5(0.6)(0.4)^4 \\
 &\approx 0.087
 \end{aligned} \tag{9.21}$$

Example 9.4 (CDF example 1). Example: Suppose $Y \sim \text{Exp}(1)$. Then

$$\begin{aligned}
F(2) &= \Pr(Y \leq 2) \\
&= \int_{-\infty}^2 e^{-t} \mathbb{I}_{(t \geq 0)} dt \\
&= \int_0^2 e^{-t} dt \\
&= -e^{-t} \Big|_0^2 \\
&= -(e^{-2} - e^0) \\
&= 1 - e^{-2} \\
&\approx 0.865
\end{aligned} \tag{9.22}$$

9.6 Quantile Function

The CDF takes a value for a random variable and returns a probability. Suppose instead we start with a number between 0 and 1, which we call p , and we wish to find a value x so that $\Pr(X \leq x) = p$. The value x which satisfies this equation is called the p quantile. (or $100p$ percentile) of the distribution of X .

Example 9.5 (Quantile Function example 1). In a standardized test, the 97th percentile of scores among all test-takers is 23. Then 23 is the score you must achieve on the test in order to score higher than 97% of all test-takers. We could equivalently call $q = 23$ the .97 quantile of the distribution of test scores.

Example 9.6 (Quantile Function example 2). The middle 50% of probability mass for a continuous random variable is found between the .25 and .75 quantiles of its distribution. If $Z \sim N(0, 1)$, then the .25 quantile is -0.674 and the .75 quantile is 0.674 . Therefore, $\Pr(-0.674 < Z < 0.674) = 0.5$.

10 Introduction to R

R has some nice functions that one can use for analysis

`mean(z)` gives the mean of some row vector z

`var(z)` reports the variance of some row vector

`sqrt(var(z))` gives the standard deviation of some row vector

`seq(from=0.1, to = 0.9, by = 0.1)` creates a vector that starts from 0.1 and goes to 0.9 incrementing by 0.1

```
seq(from=0.1, to = 0.9, by = 0.1)
```

```
[1] 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9
```

```
seq(1, 10)
```

```
[1] 1 2 3 4 5 6 7 8 9 10
```

`names(x)` gives the names of all the columns in the dataset.

```
names(trees)
```

```
[1] "Girth" "Height" "Volume"
```

`hist(x)` provides a histogram based on a vector

The more general `plot` function tries to guess at which type of plot to make. Feeding it two numerical vectors will make a scatter plot.

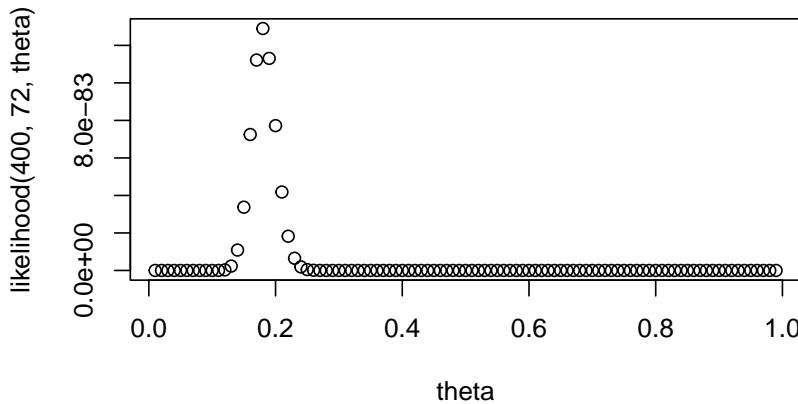
The R function `pairs` takes in a data frame and tries to make all possible Pairwise scatterplots for the dataset.

The `summary` command gives the five/six number summary (minimum, first quartile, median, mean, third quartile, maximum)

10.1 Plotting the likelihood function in R

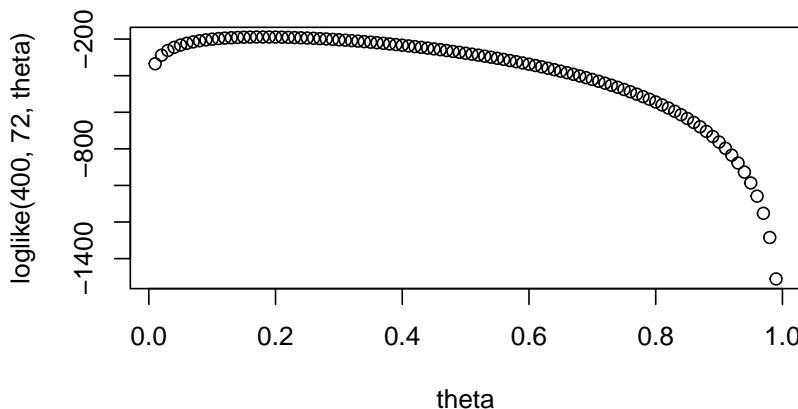
Going back to the hospital example

```
likelihood = function(n, y, theta) {  
  return(theta^y * (1 - theta)^(n - y))  
}  
theta = seq(from = 0.01, to = 0.99, by = 0.01)  
plot(theta, likelihood(400, 72, theta))
```



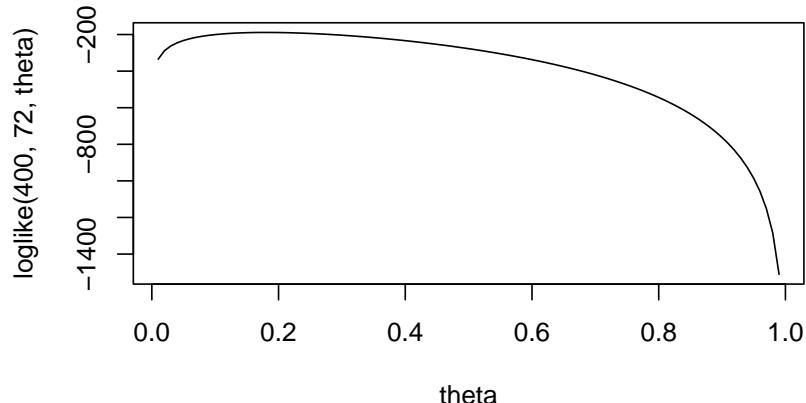
You can also do this with log likelihoods. This is typically more numerically stable to compute

```
loglike = function(n, y, theta) {  
  return(y * log(theta) + (n - y) * log(1 - theta))  
}  
plot(theta, loglike(400, 72, theta))
```



Having these plotted as points makes it difficult to see, let's plot it as lines

```
plot(theta, loglike(400, 72, theta), type = "l")
```



11 Probability Distributions in R

Each of the distributions introduced in Lesson 3 have convenient functions in R which allow you to evaluate the PDF/PMF, CDF, and quantile functions, as well as generate random samples from the distribution. To illustrate, see Table 11.1, which lists these functions for the normal distribution

Table 11.1: R API for the normal distribution

Function	What it does
<code>dnorm(x, mean, sd)</code>	Evaluate the PDF at x ($\text{mean} = \mu$ and $\text{sd} = \sqrt{\sigma^2}$)
<code>pnorm(q, mean, sd)</code>	Evaluate the CDF at q
<code>qnorm(p, mean, sd)</code>	Evaluate the quantile function at p
<code>rnorm(n, mean, sd)</code>	Generate n pseudo-random samples from the normal distribution

These four functions exist for each distribution where `d...` function evaluates the density/mass, `p...` evaluates the CDF, `q...` evaluates the quantile, and `r...` generates a sample. In Table 11.2 which lists the `d...` functions for some of the most popular distributions. The `d` can be replaced with `p`, `q`, or `r` for any of the distributions, depending on what you want to calculate.

For details enter `?dnorm` to view R's documentation page for the Normal distribution. As usual, replace the `norm` with any distribution to read the documentation for that distribution.

Table 11.2: R API for distribution

Distribution	Function	Parameters
$Binomial(n, p)$	<code>dbinom(x, size, prob)</code>	<code>size = n, prob = p</code>
$Poisson(\lambda)$	<code>dpois(x, lambda)</code>	<code>lambda = λ</code>
$Exp(\lambda)$	<code>dexp(x, rate)</code>	<code>rate = λ</code>
$Gamma(\alpha, \beta)$	<code>dgamma(x, shape, rate)</code>	<code>shape = α, rate = β</code>
$Uniform(a, b)$	<code>dunif(x, min, max)</code>	<code>min = a, max = b</code>
$Beta(\alpha, \beta)$	<code>dbeta(x, shape1, shape2)</code>	<code>shape1 = α, shape2 = β</code>
$N(\mu, \sigma^2)$	<code>dnorm(x, mean, sd)</code>	<code>mean = μ, sd = √σ²</code>
t_v	<code>dt(x, df)</code>	<code>df = v</code>

12 Bayesian Inference

Bayesian Statistics: From Concept to Data Analysis

12.1 Inference example: frequentist

Example 12.1 (Two Coin Example). Suppose your brother has a coin that you know to be loaded so that it comes up heads 70% of the time. He then comes to you with some coin, you're not sure which one and he wants to make a bet with you. Betting money that it's going to come up heads.

You're not sure if it's the loaded coin or if it's just a fair one. So he gives you a chance to flip it 5 times to check it out.

You flip it five times and get 2 heads and 3 tails.

We'll start by defining the unknown parameter θ , this is either that the coin is fair or it's a loaded coin.

$$\theta = \{\text{fair, loaded}\} \quad (\text{parameter}) \quad (12.1)$$

we get to flip it five times but we do not know what kind of coin it is

$$X \sim Bin(5, \theta) \quad (\text{model}) \quad (12.2)$$

each value of theta gives us a competing binomial likelihood:

$$\begin{aligned} \Theta &= \{\text{fair, loaded}\} \\ X &\sim Bin(5, ?) \\ f(x|\theta) &= \begin{cases} \binom{5}{x} \left(\frac{1}{2}\right)^5 & \text{if } \theta=\text{fair} \\ \binom{5}{x} (0.7)^x (0.3)^{5-x} & \text{if } \theta=\text{loaded} \end{cases} \\ &= \binom{5}{x} \left(\frac{1}{2}\right)^5 I_{\{\theta=\text{fair}\}} + \binom{5}{x} (0.7)^x (0.3)^{5-x} I_{\{\theta=\text{loaded}\}} \\ X=2 & f(\theta|X=2) = \begin{cases} 0.3125 & \text{if } \theta=\text{fair} \\ 0.1323 & \text{if } \theta=\text{loaded} \end{cases} \end{aligned}$$

Figure 12.1: coin probability inference

Which coin do you think it is and how sure are you about that?

$$f(x \mid \theta) = \begin{cases} \binom{5}{x} \left(\frac{1}{2}\right)^5 & \theta = \text{fair} \\ \binom{5}{x} (.7)^x (.3)^{5-x} & \theta = \text{loaded} \end{cases} \quad (\text{likelihood}) \quad (12.3)$$

We can also rewrite the likelihood $f(x \mid \theta)$ using indicator functions

$$f(x \mid \theta) = \binom{5}{x} (.5)^5 \mathbb{I}_{\{\theta=\text{fair}\}} + \binom{5}{x} (.7)^x (.3)^{5-x} \mathbb{I}_{\{\theta=\text{loaded}\}} \quad (\text{likelihood}) \quad (12.4)$$

In this case, we observed that $x = 2$

$$f(\theta \mid x = 2) = \begin{cases} 0.3125 & \theta = \text{fair} \\ 0.1323 & \theta = \text{loaded} \end{cases} \quad (\text{sub. } x=2) \quad (12.5)$$

$$\therefore \hat{\theta} = \text{fairMLE} \quad (12.6)$$

That's a good point estimate, but then how do we answer the question, *how sure are you?*

This is not a question that's easily answered in the frequentest paradigm. Another question is that we might like to know what is the probability that theta equals fair, given we observe two heads.

$$\Pr(\theta = \text{fair} \mid x = 2) = ? \quad (12.7)$$

In the *frequentest paradigm*, the coin is a physical quantity. It's a fixed coin, and therefore it has a fixed probability of coining up heads. It is either the fair coin, or it's the loaded coin.

$$\Pr(\theta = \text{fair}) = \{0, 1\}$$

Prior $P(\text{loaded}) = 0.6$

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

$$f(\theta|x=2) = \frac{\binom{5}{x} \left[\left(\frac{1}{2}\right)^5 (1-0.6) \mathbb{I}_{(\theta=\text{fair})} + (0.7)^x (0.3)^{5-x} (0.6) \mathbb{I}_{(\theta=\text{loaded})} \right]}{\binom{5}{x} \left[\left(\frac{1}{2}\right)^5 (0.4) + (0.7)^x (0.3)^{5-x} (0.6) \right]}$$

$$= \frac{0.0125 \mathbb{I}_{(\theta=\text{fair})} + 0.0079 \mathbb{I}_{(\theta=\text{loaded})}}{0.0125 + 0.0079}$$

$$= \mathbf{0.612} \mathbb{I}_{(\theta=\text{fair})} + 0.388 \mathbb{I}_{(\theta=\text{loaded})}$$

Figure 12.2: Bayesian coin probability inference

12.2 Bayesian Approach to the Problem

An advantage of the Bayesian approach is that it allows you to easily incorporate prior information when you know something in advance of looking at the data. This is difficult to do under the *frequentist paradigm*.

In this case, we're talking about your brother. You probably know him pretty well. So suppose you think that before you've looked at the coin, there's a 60% probability that this is the loaded coin.

In this case, we put this into our prior. Our prior belief is that the probability the coin is loaded is 0.6. We can update our prior beliefs with the data to get our posterior beliefs, and we can do this using the *Bayes theorem*.

$$\Pr(\text{loaded}) = 0.6 \quad (\text{prior})$$

$$f(\theta|x) = \frac{f(x|\theta)f(\theta)}{\sum_{\theta} f(x|\theta)f(\theta)} \quad (\text{Bayes})$$

$$f(\theta|x=2) = \frac{\binom{5}{x} \left[\left(\frac{1}{2}\right)^5 (1-0.6) \mathbb{I}_{(\theta=\text{fair})} + (0.7)^x (0.3)^{5-x} (0.6) \mathbb{I}_{(\theta=\text{loaded})} \right]}{\binom{5}{x} \left[\left(\frac{1}{2}\right)^5 (0.4) + (0.7)^x (0.3)^{5-x} (0.6) \right]} \quad (\text{sub. } x=2)$$

$$= \frac{0.0125 \mathbb{I}_{(\theta=\text{fair})} + 0.0079 \mathbb{I}_{(\theta=\text{loaded})}}{0.0125 + 0.0079} \quad (\text{normalize})$$

$$= \mathbf{0.612} \mathbb{I}_{(\theta=\text{fair})} + 0.388 \mathbb{I}_{(\theta=\text{loaded})} \quad (\text{MLE})$$

(12.8)

As you can see in the calculation Equation 12.8, we have the *likelihood* times the *prior* in the numerator, and a *normalizing constant* in the denominator. When we divide the two, we'll get an answer that adds up to 1. These numbers match exactly in this case because it's a very simple problem.

This is a concept that we will revisit — what's in the denominator here is always a normalizing constant.

$$\Pr(\theta = \text{loaded} | x = 2) = 0.388$$

This here updates our beliefs after seeing some data about what the probability might be.

We can also examine what would happen under different choices of prior.

$$\Pr(\theta = \text{loaded}) = \frac{1}{2} \implies \Pr(\theta = \text{loaded} | x = 2) = 0.297$$

$$\Pr(\theta = \text{loaded}) = 0.9 \implies \Pr(\theta = \text{loaded} | x = 2) = 0.792$$

In this case, the Bayesian approach is inherently subjective. It represents your perspective, and this is an important part of the paradigm. If you have a different perspective, you will get different answers, and that's okay. It's all done in a mathematically vigorous framework, and it's all mathematically consistent and coherent.

And in the end, we get interpretable results.

12.3 Continuous version of Bayes' theorem

12.4 Posterior Intervals

12.5 Discussion CIs

- **Frequentist confidence intervals** have the interpretation that “If you were to repeat many times the process of collecting data and computing a 95% confidence interval, then on average about 95% of those intervals would contain the true parameter value; however, once you observe data and compute an interval the true value is either in the interval or it is not, but you can't tell which.”
- **Bayesian credible intervals** have the interpretation that “Your posterior probability that the *parameter* is in a 95% credible interval is 95%.”
- Bayesian intervals treat their bounds as fixed and the estimated parameter as a random variable.
- Frequentist confidence intervals treat their bounds as random variables and the parameter as a fixed value.

This is from a video of a lecture by Christian Robert at Paris-Dauphine. When I first posted it here, $f(\theta|y) = 2\theta I_{\theta > 0}$. The terms written in between are just proportional, without the normalizing constant, and the normalizing constant is put back at the end, so that at the end we do have equality with the original left-hand side. And to be completely correct here, I really should write this as $f(\theta|Y=1) = 2\theta I_{\theta > 0, Y=1}$, rather than writing this as $f(\theta|y)$.

Figure 12.3: Continuous version of Bayes' theorem

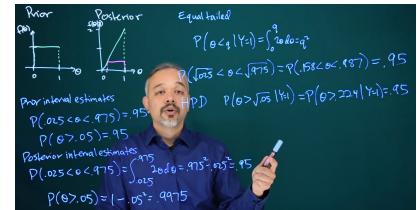


Figure 12.4: Posterior Intervals

i Discussion Under what circumstances would you prefer a frequentist CI or a Bayesian CI?

12.5.1 Focusing on Bayesian / Frequentist paradigms

- A Frequentist CI might be preferred if:
 1. I had plenty of data to support a frequentist construction of frequentist CI and
 2. I was doing research and refining or refuting a result that has been established using frequentist hypothesis testing.
 - I would want to show that for H_1 against some null hypothesis, H_0 the parameters have a certain p-value for some effect.
 - Particularly when we are interested in the inference and are less interested in using the value of the parameter.
 3. I cannot justify introducing some subjective priors.
- A Bayesian CI might be better if:
 1. My dataset is too small.
 2. What I care about is the parameter's value and less about hypothesis testing
 3. I need an estimate of uncertainty for the parameter for the inference it is used in.
 4. I had subjective reasons to introduce a prior:
 - I know about constraints
 - I have access to expert knowledge
 5. I wish to introduce pooling between groups, to share information for reducing uncertainty.
 6. My results are in a Bayesian-oriented domain.

i Discussion: Under what circumstances would you prefer a frequentist CI or a Bayesian CI?

12.5.2 Focusing on the CI choices

Let's point out that this is what we call a loaded question, as it has a bias against the frequentist approach by stating one of its

shortcomings when it is still possible to get a point estimate for the parameter and compare it to the CI. Typically one will have already done it say using regression before considering the CI.

Next, we have all the standard reasons for choosing between the Frequentist and the Bayesian paradigms. I could list them but I don't think that is the real point of this question, but rather what would we prefer if both were viable options and why?

CIs are primarily a tool for understanding uncertainties about parameters that encode effects. In the parametric Bayesian approach we are learning the distribution of our parameters so they have uncertainties baked into them. In the Frequentist approach, we look for the least squares point estimates for our parameters and consider using the CI to approximate the long-run uncertainty due to sampling.

Frequentist CI might be preferable if I am worried about Aletoric uncertainty due to sampling i.e. to what degree can I be certain my experimental outcomes are not due to chance? I would feel this way since I am a classical physicist or a botanist studying a predominately deterministic effect and I see errors in estimating the parameters as artifacts of sampling that can be made smaller till the parameters will converge with the true population statistics and the error will become vanishingly small.

Given that I did my best to get a good data sample I just need to check how sure to decide the cardinal question do I publish or do I perish? I need to decide that the result is due to the effect and not due to some conspiracy bad samples.

Bayesian CIs are just a result of using Bayesian analysis which is a requirement to investigate what are predominately random effects that are the domain of quantum physicists, an ecologist, or a geneticist. Since almost everything I study is predominantly random and I need random variables and Bayes law to get to my results. I also need to report confidence intervals for my work when I publish - but if one is a Bayesian, one will use a Bayesian credible interval?

13 Priors

Bayesian Statistics: From Concept to Data Analysis

In this section, we will delve more deeply into choices of Priors and how they influence Bayesian CI by developing the **prior predictive** (Definition 13.1) and posterior **predictive** (Definition 13.2) intervals.

13.1 Priors and prior predictive distributions

! Choosing a prior

How should we choose a prior?

1. Our prior needs to represent our perspectives, beliefs, and our uncertainties.
2. It should encode any constraints on the data or parameters.
 - age is positive and less than 120
3. It can regularize the data
4. It could encode expert knowledge we have elicited from domain experts.
5. It should prefer informative priors over uninformative ones.

Theoretically, we're defining a cumulative distribution function for the parameter

$$\Pr(\theta \leq c) \quad \forall c \in \mathbb{R}$$

We need to do this for an infinite number of possible sets but it isn't practical to do, and it would be very difficult to do it coherently so that all

the probabilities were consistent. Therefore in practice, we tend to work with a convenient family that is flexible enough for members to represent our beliefs.

Generally if one has enough data, the information in the data will overwhelm the information in the prior. This makes it seem like the prior is less important in terms of the form and substance of the posterior. Once the prior is overwhelmed, any reasonable choice of prior will lead to approximately the same posterior. This is a point where the Bayesian approach should converge to the frequentist and can be shown to be more or less objective.

On the other hand choices of priors can be important because even with masses of data, groups and items can be distributed very sparsely in which case priors can have a lasting impact on the posteriors. Secondly, we can decide to pick priors that have a long-lasting impact on operating as regularizing constraints within our models. In such cases, the impact of the prior can be significant.

One of our guiding questions will be to consider how much information the prior and the data contribute to the posterior. We will consider the effective sample size of different priors.

Finally, a bad choice of priors can lead to specific issues.

Example 13.1 (Example of Bad Prior). Suppose we chose a prior that says the probability of $\Pr(\theta = \frac{1}{2}) = \delta(\frac{1}{2}) = 1$

And thus, the probability of θ equaling any other value is 0. If we do this, our data won't make a difference since we only put a probability of 1 at a single point.

$$f(\theta | y) \propto f(y | \theta) f(\theta) = f(\theta) = \delta(\theta) \quad (13.1)$$

🔥 Avoid priors that assign 0 or 1

- Events with a prior probability of 0 will always have a posterior probability of 0 because $f(\theta) = 0$ in (Equation 13.1) the product will be 0 and therefore the posterior be 0
- Events with a prior probability of 1, will always have a posterior probability of 1. This is a little harder to see. In this case $f(\theta^c) = 0$ in (Equation 13.1) so that the posterior will again

be zero elsewhere.

- It is good practice to avoid assigning a probability of 0 or 1 to any event that has already occurred or is already known not to occur.
- If the priors avoid 0 and 1 values the information within the data will eventually overwhelm the information within the prior.

13.1.1 Calibration - making priors precise

Calibration of predictive intervals is a useful concept in terms of choosing priors. If we make an interval where we're saying we predict 95% of new data points will occur in this interval. It would be good if, in reality, 95% of new data points did fall in that interval. This is a *frequentist* concept but this is important for practical statistical purposes so that our results reflect reality.

We can compute a predictive interval. This is an interval such that 95% of new observations are expected to fall into it. It's an interval for the **data** rather than an interval for θ

Definition 13.1 (Prior Predictive Distribution). The **prior predictive** distribution expresses our uncertainty about a parameter, i.e. the distribution of its possible values **before** we observe any data.

$$\begin{aligned} f(y) &= \int f(y | \theta) f(\theta) d\theta \quad \text{by Bayes theorem} \\ &= \int f(y, \theta) d\theta \quad \text{the joint probability} \end{aligned} \tag{13.2}$$

- $f(y, \theta)$ is the *joint density* of y and θ .
- If we are integrating out θ , we will end up with a marginalized probability distribution of the data.
- However, we may well decide to not integrate out θ completely, so we will end up with a predictive interval.
- But no data y has been observed, so this is the prior predictive before any data is observed.
- It is used in **prior predictive checks** to assess whether the choice of prior distribution captures our prior beliefs.

Q. How do we calibrate our prior

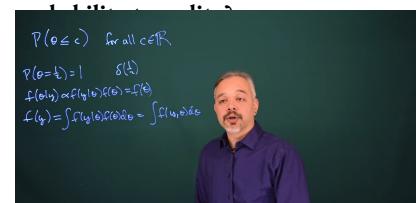


Figure 13.1: Prior Predictive Distribution

13.2 Prior Predictive: Binomial Example

Suppose we're going to flip a coin 10 times and count the number of heads we see. But we are thinking about this in advance of actually doing it, and we are interested in the predictive distribution

So, we'll need to choose a prior.

$$N = 10 \quad \text{number of coin flips}$$

Where Y_i represents individual coin flips. with Head being a success

$$Y \sim \text{Bernoulli}(\theta)$$

Our data is the count of successes (heads) in N flips.

$$X = \sum_{i=0}^N Y_i$$

If we think that all possible coins or all possible probabilities are equally likely, then we can put a prior for θ that's flat over the interval from 0 to 1. That is the Uniform prior (Equation 7.17):

$$f(\theta) = \mathbb{I}_{[0 \leq \theta \leq 1]}$$

The predictive probability is a *binomial likelihood* times the *prior* = 1

$$f(x) = \int f(x | \theta) f(\theta) d\theta = \int_0^1 \frac{10!}{x!(10-x)!} \theta^x (1-\theta)^{10-x} (1) d\theta$$

Note that because we're interested in X at the end, it's important that we distinguish between a Binomial density and a Bernoulli density. Here we just care about the total count rather than the exact ordering which would be Bernoulli.

For most of the analyses, we're doing, where we're interested in θ rather than x , the binomial and the Bernoulli are interchangeable because the part in here that depends on θ is the same.

$$\begin{aligned} X &= \sum_{i=1}^n Y_i \\ f(\theta) &= \mathbb{I}_{\{0 \leq \theta \leq 1\}} \\ f(x) &= \int f(x|\theta) f(\theta) d\theta = \int_0^1 \frac{10!}{x!(10-x)!} \theta^x (1-\theta)^{10-x} (1) d\theta \\ &= \int_0^1 \frac{\Gamma(11)}{\Gamma(x+1)\Gamma(10-x)} \theta^x (1-\theta)^{10-x} d\theta \\ &= \frac{\Gamma(11)}{\Gamma(10)} \left(\frac{\Gamma(10)}{\Gamma(10)} \right)^{10} \theta^{10} (1-\theta)^{10} \\ &= \frac{\Gamma(11)}{\Gamma(10)} (1) = \frac{11}{10} \end{aligned}$$

Figure 13.2: Prior Predictive Distribution
Q. How many heads do we predict we're going to see?

Q. What's the probability that it shows up heads?

To solve this integral let us recall that:

$$n! = \Gamma(n + 1) \quad (13.3)$$

and

$$Z \sim \text{Beta}(\alpha, \beta)$$

The PDF for the beta distribution is given as:

$$f(z) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} z^{\alpha-1} (1-z)^{\beta-1} I_{(0 < z < 1)}$$

where $\alpha > 0$ and $\beta > 0$.

$$\begin{aligned} f(x) &= \int f(x \mid \theta) f(\theta) d\theta && \text{prior predictive dfn} \\ &= \int_0^1 \frac{10!}{x!(10-x)!} \theta^x (1-\theta)^{10-x} (\mathbb{I}_{[0,1]}) d\theta && \text{subst. Binomial, } \mathbb{I}_{[0,1]} \\ &= \int_0^1 \frac{\Gamma(11)}{\Gamma(x+1)\Gamma(11-x)} \theta^{(x+1)-1} (1-\theta)^{(11-x)-1} (1) d\theta && \text{convert to Beta}(x+1, 11-x), \\ &= \frac{\Gamma(11)}{\Gamma(12)} \int_0^1 \frac{\Gamma(12)}{\Gamma(x+1)\Gamma(11-x)} \theta^{(x+1)-1} (1-\theta)^{(11-x)-1} (1) d\theta && \text{integrating PDF=1} \\ &= \frac{\Gamma(11)}{\Gamma(12)} \times 1 = \frac{10!}{11!} = \frac{1}{11} && \forall x \in \{1, 2, \dots, 10\} \end{aligned}$$

Thus we see that if we start with a uniform prior, we then end up with a discrete uniform predictive density for X . If all possible θ probabilities are equally likely, then all possible sums X outcomes are equally likely.

The integral above is a beta density, all integrals of valid beta densities equal one.

$$f(x) = \frac{\Gamma(11)}{\Gamma(12)} = \frac{10!}{11!} = \frac{1}{11}$$

13.3 Posterior Predictive Distribution

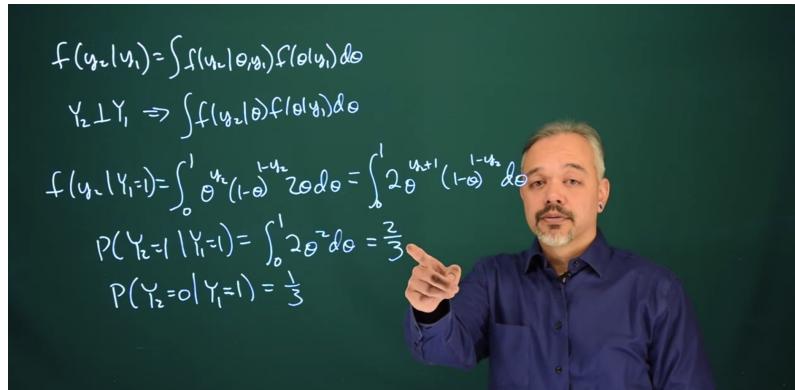


Figure 13.3: Posterior Predictive Distribution

What about after we've observed data? What's our posterior predictive distribution?

Going from the previous example, let us observe after one flip that we got a head.

We want to ask, **what's our predictive distribution for the second flip, given we saw a head on the first flip?**

13.4 Posterior Predictive Distribution

The posterior predictive distribution is produced analogously to the posterior predictive distribution by marginalizing the posterior with respect to the parameter.

Definition 13.2 (Posterior Predictive Distribution).

$$\begin{aligned} f(y_2 | y_1) &= \text{likelihood} \times \text{posterior} \\ &= \int f(y_2 | \theta, y_1) f(\theta | y_1) d\theta \end{aligned} \tag{13.4}$$

💡 Marginalizing distribution

Suppose we have an experiment with events based on two RVs: -

(C) a coin toss - (D) and a dice toss. And we call this event $X = \mathbb{P}r(C, D) = \mathbb{P}r(C) \times \mathbb{P}r(D)$

CD	1	2	3	4	5	6	$\mathbb{P}r(C)$
H	1/12	1/12	1/12	1/12	1/12	1/12	6/12
T	1/12	1/12	1/12	1/12	1/12	1/12	6/12
$\mathbb{P}r(D)$	2/12	2/12	2/12	2/12	2/12	2/12	1

We can recover the $\mathbb{P}r(C)$ coin's distribution or the dice distribution $\mathbb{P}r(D)$ by marginalization. $\mathbb{P}r(X)$ This is done by summing over the row or columns.

The marginal distribution let us subset a joint distribution. The marginal distribution has removed the uncertainty due to a parameter.

we use three terms interchangeably :

- marginalizing the posterior w.r.t. θ
- integrating/summing over θ
- integrating θ out

The first is the real idea, the others are the techniques being used to do it. For a predictive distribution we may want to marginalize all the parameters so we end up with the RV we wish to predict.

We're going to assume that Y_2 is independent of Y_1 . Therefore,

$$f(y_2 | y_1) = \int f(y_2 | \theta) f(\theta | y_1) d\theta$$

Suppose we're thinking of a uniform distribution for θ and we observe the first flip is a "head". What do we predict for the second flip?

This is no longer going to be a uniform distribution like it was before because we have some data. We're going to think it's more likely that we're going to get a second head. We think this because since we observed a head θ is now likely to be at least $\frac{1}{2}$ possibly larger.

$$f(y_2 | Y_1 = 1) = \int_0^1 \theta^{y_2} (1 - \theta)^{1-y_2} 2\theta d\theta$$

$$f(y_2 | Y_1 = 1) = \int_0^1 2\theta^{y_2+1} (1 - \theta)^{1-y_2} d\theta$$

We could work this out in a more general form, but in this case, Y_2 has to take the value 0 or 1. The next flip is either going to be heads or tails so it's easier to just plop in a particular example.

$$\Pr(Y_2 = 1 | Y_1 = 1) = \int_{\theta}^1 2\theta^2 d\theta = \frac{2}{3}$$

$$\Pr(Y_2 = 0 | Y_1 = 1) = 1 - \Pr(Y_2 = 1 | Y_1 = 1) = 1 - \frac{2}{3} = \frac{1}{3}$$

We can see here that the posterior is a combination of the information in the prior and the information in the data. In this case, our prior is like having two data points, one head and one tail.

Saying we have a **uniform prior** for θ is equivalent in an information sense to saying “we have observed one ‘Head’ and one ‘Tail’ ”.

So then when we observe one head, it's like we now have seen two heads and one tail. So our predictive distribution for the second flip says if we have two heads and one tail, then we have a $\frac{2}{3}$ probability of getting another head and a $\frac{1}{3}$ probability of getting another tail.

14 Binomial Data

Bayesian Statistics: From Concept to Data Analysis

14.1 Bernoulli/Binomial likelihood with a uniform prior

When we use a uniform prior for a Bernoulli likelihood, we get a beta posterior.

The Bernoulli likelihood of $\vec{Y} \mid \theta$ is

$$f(\vec{Y} \mid \theta) = \theta^{\sum y_i} (1 - \theta)^{n - \sum y_i} \quad \text{Bernoulli Likelihood}$$

Our prior for θ is just a Uniform distribution

$$f(\theta) = I_{\{0 \leq \theta \leq 1\}} \quad \text{Uniform prior}$$

Thus our posterior for θ is

$$\begin{aligned} f(\theta \mid y) &= \frac{f(y \mid \theta)f(\theta)}{\int f(y \mid \theta)f(\theta) d\theta} && \text{Bayes law} \\ &= \frac{\theta^{\sum y_i} (1 - \theta)^{n - \sum y_i} \mathbb{I}_{\{0 \leq \theta \leq 1\}}}{\int_0^1 \theta^{\sum y_i} (1 - \theta)^{n - \sum y_i} \mathbb{I}_{\{0 \leq \theta \leq 1\}} d\theta} && \text{subst. Likelihood \& Prior} \\ &= \frac{\theta^{\sum y_i} (1 - \theta)^{n - \sum y_i} \mathbb{I}_{\{0 \leq \theta \leq 1\}}}{\frac{\Gamma(\sum y_i + 1)\Gamma(n - \sum y_i + 1)}{\Gamma(n+2)} \int_0^1 \frac{\Gamma(n+2)}{\Gamma(\sum y_i + 1)\Gamma(n - \sum y_i + 1)} \theta^{\sum y_i} (1 - \theta)^{n - \sum y_i} d\theta} && \text{Beta PDF integrates to 1} \\ &= \frac{\Gamma(n+2)}{\Gamma(\sum y_i + 1)\Gamma(n - \sum y_i + 1)} \theta^{\sum y_i} (1 - \theta)^{n - \sum y_i} \mathbb{I}_{\{0 \leq \theta \leq 1\}} && \text{simplifying} \\ &= \text{Beta}(\sum y_i + 1, n - \sum y_i + 1) \end{aligned}$$

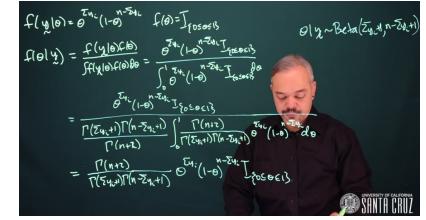


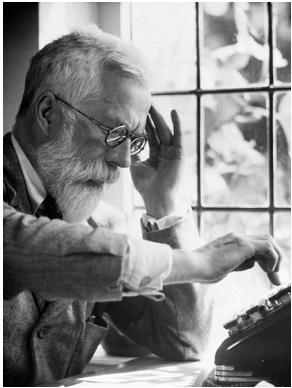
Figure 14.1: Binomial likelihood with a Uniform prior

Where we used a trick of recognizing the denominator as a Beta distribution (Equation 42.7) we then manipulate it to take the exact form of Beta. We can then cancel it since the beta density integrates to 1, we can simplify this as From here we can see that the posterior follows a beta distribution

$$\theta|y \sim Beta(\sum y_i + 1, n - \sum y_i + 1)$$



Historical Note on R.A. Fisher



R.A. Fisher's objection to the Bayesian approach is that "The theory of inverse probability **is founded upon an error, and must be wholly rejected**" (Fisher 1925) was specifically referring to this example of a "Binomial with a Uniform prior". The gist of it is that the posterior depends on the parametrization of the prior.(Aldrich 2008). R.A. Jeffry who corresponded with Fisher went on to develop his eponymous priors which were invariant to reparametrization. Which we will consider in Section 18.2

14.2 Conjugate Priors

The image shows a video thumbnail. A man in a black shirt is standing in front of a chalkboard. The chalkboard contains the following text and equations:

$$f(\theta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1} I_{\{0 \leq \theta \leq 1\}}$$

$$f(\theta|y) \propto f(y|\theta)f(\theta) = \theta^{\sum y_i} (1-\theta)^{n-\sum y_i} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1} I_{\{\theta \leq \theta \leq 1\}}$$

$$\propto \theta^{\alpha + \sum y_i - 1} (1-\theta)^{\beta + n - \sum y_i - 1}$$

Conjugate

© 2014 Kuta Software LLC. All rights reserved.

Figure 14.2: Conjugate Priors

The uniform distribution is $Beta(1, 1)$

Any beta distribution is conjugate for the Bernoulli distribution. Any beta prior will give a beta posterior.

$$f(\theta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1} I_{\{\theta \leq \theta \leq 1\}}$$

$$f(\theta | y) \propto f(y | \theta)f(\theta) = \theta^{\sum y_i} (1-\theta)^{n-\sum y_i} \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1} I_{\{\theta \leq \theta \leq 1\}}$$

$$f(y | \theta)f(\theta) \propto \theta^{\alpha + \sum y_i - 1} (1-\theta)^{\beta + n - \sum y_i - 1}$$

Thus we see that this is a beta distribution

$$\theta | y \sim Beta(\alpha + \sum y_i, \beta + n - \sum y_i)$$

When α and β are one like in the uniform distribution, then we get the same result as earlier.

This whole process where we choose a particular form of prior that works with a likelihood is called using a conjugate family.

A family of distributions is referred to as conjugate if when you use a member of that family as a prior, you get another member of that family as your posterior.

The beta distribution is conjugate for the Bernoulli distribution. It's also conjugate for the binomial distribution. The only difference in the

binomial likelihood is that there is a combinatorics term. Since that does not depend on θ , we get the same posterior.

We often use conjugate priors because they make life much simpler, sticking to conjugate families allows us to get closed-form solutions easily.

If the family is flexible enough, then you can find a member of that family that closely represents your beliefs.

- the Uniform distribution can be written as the Beta(1,1) prior.
- Any Beta prior will give a Beta posterior.
- Beta is conjugate for Binomial and for Bernoulli

14.3 Posterior mean and effective sample size

Figure 14.3: Effective Sample Size

Returning to the beta posterior model it is clear how both the prior and the data contribute to the posterior.

For a prior $Beta(\alpha, \beta)$ we can say that the **effective sample size** of the prior is

$$\alpha + \beta \quad (\text{ESS}) \quad (14.1)$$

Recall that the expected value or mean of a $Beta$ distribution is $\frac{\alpha}{\alpha + \beta}$

Therefore we can derive the posterior mean as

$$\begin{aligned}
posterior_{mean} &= \frac{\alpha + \sum y_i}{\alpha + \sum y_i + \beta + n - \sum y_i} \\
&= \frac{\alpha + \sum y_i}{\alpha + \beta + n} \\
&= \frac{\alpha + \beta}{\alpha + \beta + n} \frac{\alpha}{\alpha + \beta} + \frac{n}{\alpha + \beta + n} \frac{\sum y_i}{n} \\
&= (\text{prior weight} \times \text{prior mean}) + (\text{data weight} \times \text{data mean})
\end{aligned} \tag{14.2}$$

i.e. The **posterior mean** is a weighted average of the **prior mean** and the **data mean**.

This effective sample size gives you an idea of how much data you would need to make sure that your prior does not have much influence on your posterior.

If $\alpha + \beta$ is small compared to n then the posterior will largely just be driven by the data. If $\alpha + \beta$ is large relative to n then the posterior will be largely driven by the prior.

We can make a 95% credible interval using our posterior distribution for θ . We can find an interval that has 95% probability of containing θ .

Using Bayesian Statistics we can do sequential analysis by doing a sequential update every time we get new data. We can get a new posterior, and we just use our previous Posterior as a Prior for doing another update using Bayes' theorem.

- for a Beta prior, its effective sample size is $a + b$
- if $n \gg \alpha + \beta$ the posterior will be predominantly determined by the prior
- if $n \ll \alpha + \beta$ the posterior will be predominantly determined by the data
- the idea of an effective sample size of the prior is a useful concept to work with.
- (Wiesenfarth and Calderazzo 2020)
 - Effective Sample Size (ESS)
 - Effective Current Sample size (ECSS)

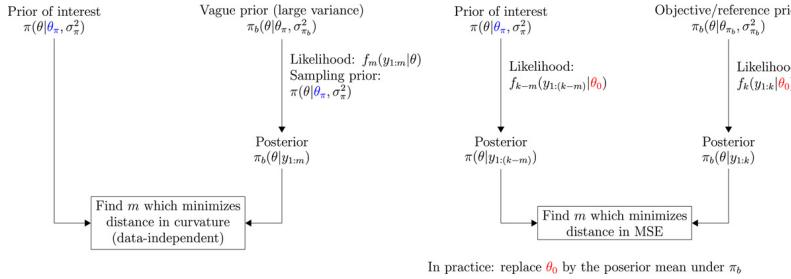


Figure 14.4: ESS algorithms

- with (Morita, Thall, and Müller 2008) on the left and ECSS on the right

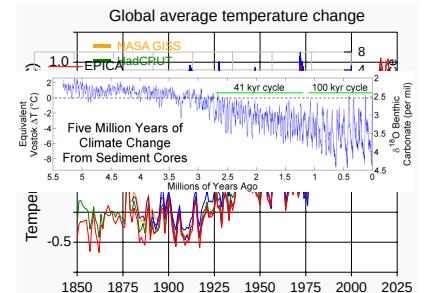
Exercise 14.1 (Discussion on Prior elicitation). Suppose we are interested in global temperatures, and that we have a summary measure that represents the average global temperature for each year. Now we could ask “What is the probability that next year will have a higher average global temperature than this year?” What would be your choice of prior and why? Be specific about the distribution and its parameters. You may use any other information that you want to bring into this problem.

Solution. It is possible to get historical estimates using:

- meteorological and satellites for the last 200 years.
- ice cores for the last 800,000 years
- deep sea sediment oxygen 18 isotope fractionation for the last 5 million years. or yearly temperature data from 1850 till today based on meteorological readings. We can also consider Greenland ice core data covering 800,000 years.

One simple way is to model the yearly temperature as a random walk i.e. Each year is a Bernoulli trial where success is the temperature getting warmer. We can then use the historical data since 1800 to estimate theta the probability that we get warmer.

I suppose we can use a Binomial prior with parameters for alpha the count of years the temperature increased and N for the total number of years and p the probability the a given year is hotter than the previous.



14.4 Data Analysis Example in R

Suppose we're giving two students a multiple-choice exam with 40 questions, where each question has four choices. We don't know how much the students have studied for this exam, but we think that they'll do better than just guessing randomly

1. What are the parameters of interest?

The parameters of interests are $\theta_1 = \text{true}$ the probability that the first student will answer a question correctly, $\theta_2 = \text{true}$ the probability that the second student will answer a question correctly.

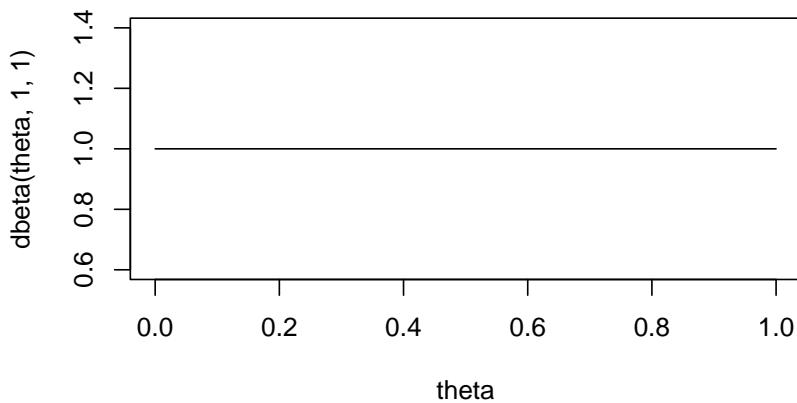
2. What is our likelihood?

The likelihood is $\text{Binomial}(40, \theta)$ if we assume that each question is independent and that the probability a student gets each question right is the same for all questions for that student.

3. What prior should we use?

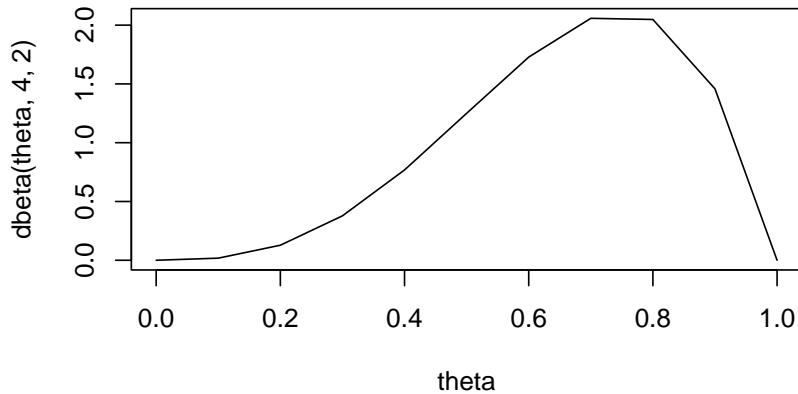
The **Conjugate Prior** is a **Beta Distribution**. We can plot the density with `dbeta`

```
theta = seq(from = 0, to = 1, by = 0.1)
# Uniform
plot(theta, dbeta(theta, 1, 1), type = 'l')
```



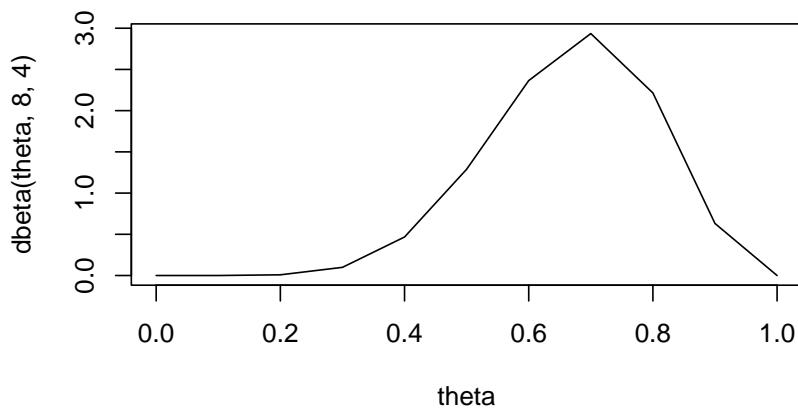
```
# Prior mean 2/3
```

```
plot(theta, dbeta(theta, 4, 2), type = 'l')
```



```
# Prior mean 2/3 but higher effect size (more concentrated at mean)
```

```
plot(theta, dbeta(theta, 8, 4), type = 'l')
```



4. What are the prior probabilities $\Pr(\theta > 0.25)$? $\Pr(\theta > 0.5)$?
 $\Pr(\theta > 0.8)$?

```
1 - pbeta(0.25, 8, 4)
```

```
[1] 0.9988117
```

```
# [1] 0.998117
```

```
1 - pbeta(0.5, 8, 4)
```

```
[1] 0.8867188
#[1] 0.8867188
1 - pbeta(0.8, 8, 4)
```

```
[1] 0.1611392
```

```
# [1] 0.16113392
```

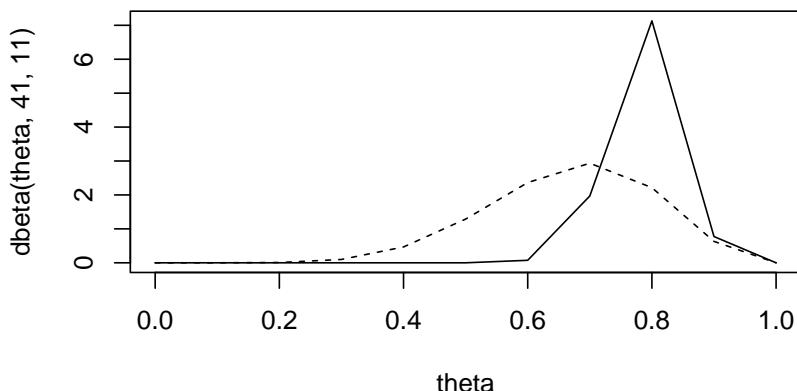
5. Suppose the first student gets 33 questions right. What is the posterior distribution for θ_1 ? $\Pr(\theta > 0.25)$? $\Pr(\theta > 0.5)$? $\Pr(\theta > 0.8)$? What is the 95% posterior credible interval for θ_1 ?

$$\text{Posterior} \sim \text{Beta}(8 + 33, 4 + 40 - 33) = \text{Beta}(41, 11)$$

With a posterior mean of $\frac{41}{41+11} = \frac{41}{52}$

We can plot the posterior distribution with the prior

```
plot(theta, dbeta(theta, 41, 11), type = 'l')
lines(theta, dbeta(theta, 8, 4), lty = 2) #Dashed line for prior
```



Posterior probabilities

```
1 - pbeta(0.25, 41, 11)
```

```
[1] 1
```

```
# [1] 1  
1 - pbeta(0.5, 41, 11)
```

```
[1] 0.9999926
```

```
# [1] 0.9999926  
1 - pbeta(0.8, 41, 11)
```

```
[1] 0.4444044
```

```
# [1] 0.4444044
```

Equal-tailed 95% credible interval

```
qbeta(0.025, 41, 11)
```

```
[1] 0.6688426
```

```
# [1] 0.6688426  
qbeta(0.975, 41, 11)
```

```
[1] 0.8871094
```

```
# [1] 0.8871094
```

95% confidence that θ_1 is between 0.67 and 0.89

6. Suppose the second student gets 24 questions right. What is the posterior distribution for θ_2 ? $\Pr(\theta > 0.25)$? $\Pr(\theta > 0.5)$? $\Pr(\theta > 0.8)$? What is the 95% posterior credible interval for θ_2

Posterior $\sim \text{Beta}(8 + 24, 4 + 40 - 24) = \text{Beta}(32, 20)$

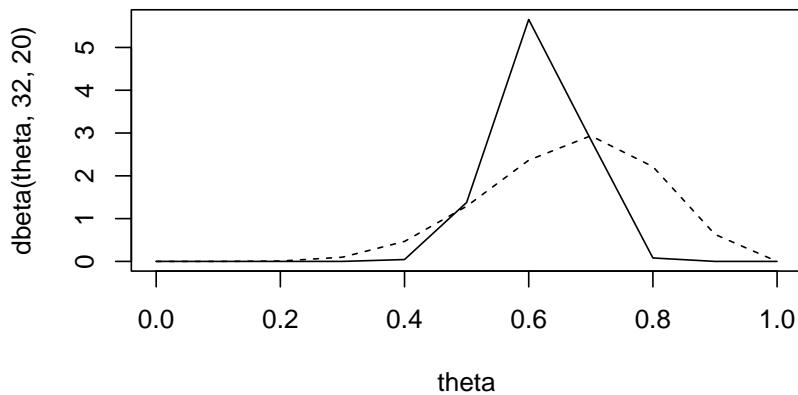
With a posterior mean of $\frac{32}{32+20} = \frac{32}{52}$

We can plot the posterior distribution with the prior

```

plot(theta, dbeta(theta, 32, 20), type = 'l')
lines(theta, dbeta(theta, 8 ,4), lty = 2) #Dashed line for prior

```



Posterior probabilities

```
1 - pbeta(0.25, 32, 20)
```

```
[1] 1
```

```
# [1] 1
1 - pbeta(0.5, 32, 20)
```

```
[1] 0.9540427
```

```
# [1] 0.9540427
1 - pbeta(0.8, 32, 20)
```

```
[1] 0.00124819
```

```
# [1] 0.00124819
```

Equal-tailed 95% credible interval

```
qbeta(0.025, 32, 20)
```

```
[1] 0.4808022
```

```
# [1] 0.4808022  
qbeta(0.975, 32, 20)
```

```
[1] 0.7415564
```

```
# [1] 0.7415564
```

95% confidence that θ_1 is between 0.48 and 0.74

7. What is the posterior probability that $\theta_1 > \theta_2$?

i.e., that the first student has a better chance of getting a question right than the second student?

Estimate by simulation: draw 1,000 samples from each and see how often we observe $\theta_1 > \theta_2$

```
theta1 = rbeta(100000, 41, 11)  
theta2 = rbeta(100000, 32, 20)  
mean(theta1 > theta2)
```

```
[1] 0.97469
```

```
# [1] 0.975
```

15 Poisson Data

Bayesian Statistics: From Concept to Data Analysis

15.0.1 Poisson - Chocolate Chip Cookie

In mass-produced chocolate chip cookies, they make a large amount of dough; mix in a large number of chips; then chunk out the individual cookies. In this process, the number of chips per cookie approximately follows a **Poisson distribution**.

If we were to assume that chips have no volume, then this would be exactly a *Poisson process* and follow exactly a *Poisson distribution*. In practice, since chips are not that small, so they follow approximately a Poisson distribution for the number of chips per cookie.

$$Y_i \sim \text{Poisson}(\lambda) \quad (15.1)$$

The likelihood of the data is given by the Poisson distribution.

$$f(y | \lambda) = \frac{\lambda^{\sum y_i} e^{-n\lambda}}{\prod_{i=1}^n y_i!} \quad \forall (\lambda > 0) \quad \text{Poisson Likelihood}$$

It would be convenient if we could put a *conjugate prior*. What distribution looks like λ raised to a power and e raised to a negative power?

For this, we're going to use a Gamma prior.

$$\begin{aligned} \lambda &\sim \text{Gamma}(\alpha, \beta) \quad \text{Gamma Prior} \\ f(\lambda) &= \frac{\beta^\alpha}{\Gamma(\alpha)} \lambda^{\alpha-1} e^{-\beta\lambda} \quad \text{subst. Gamma PDF} \end{aligned} \quad (15.2)$$

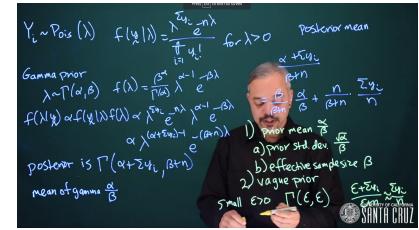


Figure 15.1: Poisson likelihood with a Gamma prior

What is the likelihood of the data?

What type of prior should we put on λ ?

We can use Bayes theorem to find the posterior.

What is the posterior?

$$\begin{aligned}
 f(\lambda | y) &\propto f(y | \lambda) f(\lambda) \\
 &\propto \lambda^{\sum y_i} e^{-n\lambda} \lambda^{\alpha-1} e^{-\beta\lambda} \\
 &\propto \lambda^{\alpha+\sum y_i-1} e^{-(\beta+n)\lambda} \\
 &\propto \text{Gamma}(\alpha + \sum y_i, \beta + n)
 \end{aligned} \tag{15.3}$$

Bayes without the denominator
subst. Likelihood and Prior
collecting terms

The posterior is a Gamma distribution with parameters $\alpha + \sum y_i$ and $\beta + n$.

What is the posterior distribution?

Thus we can see that the posterior is a Gamma Distribution

$$\lambda | y \sim \text{Gamma}(\alpha + \sum y_i, \beta + n) \tag{15.4}$$

The posterior mean of a Gamma distribution is given by

What is the posterior mean?

The mean of Gamma under this parameterization is: $\frac{\alpha}{\beta}$

The posterior mean is going to be

$$\begin{aligned}
 \mu_\lambda &= \frac{\alpha + \sum y_i}{\beta + n} && \text{(Posterior Mean)} \\
 posterior_\mu &= \frac{\alpha + \sum y_i}{\beta + n} \\
 &= \frac{\beta}{\beta + n} \frac{\alpha}{\beta} + \frac{n}{\beta + n} \frac{\sum y_i}{n} \\
 &\propto \beta \cdot \mu_{\text{prior}} + n \cdot \mu_{\text{data}}
 \end{aligned} \tag{15.5}$$

The posterior variance of a Gamma distribution is given by

What is the posterior variance?

As you can see here the posterior mean of the Gamma distribution is also the weighted average of the prior mean and the data mean.

Therefore, the effective sample size (ESS) of the Gamma prior is β

Prior Elicitation of Gamma Hyper-parameters

Here are two strategies for choose the *hyper-parameters* α and β

1. An *informative prior* with a prior mean guess of $\mu = \frac{a}{b}$
e.g. what is the average number of chips per cookie?
 - Next we need another piece of knowledge to pinpoint both parameters.
 - Can you estimate the error for the mean? I.e. what do you think the *standard deviation* is? Since for the Gamma prior
 - What is the effective sample size $ESS = \beta$?
 - How many units of information do you think we have in our prior v.s. our data points? $\sigma = \frac{\sqrt{\alpha}}{\beta}$
2. A *vague prior* refers to one that's relatively flat across much of the space.
 - For a *Gamma prior* we can choose $\Gamma(\epsilon, \epsilon)$ where ϵ is small and strictly positive. This would create a distribution with a $\mu = 1$ and a huge σ stretching across the whole space. And the *effective sample size* will also be ϵ . Hence the posterior will be largely driven by the data and very little by the prior.

The first strategy with a mean and an ESS will be used in numerous models going forward so it is best to remember these two strategies!

16 Exponential Data

Bayesian Statistics: From Concept to Data Analysis

Example 16.1 (Time between buses - Exponential).

Suppose you're waiting for a bus that you think comes on average once every 10 minutes, but you're not sure exactly how often it comes.

$$Y \sim \text{Exp}(\lambda) \quad (16.1)$$

Your waiting time has a prior expectation of $\frac{1}{\lambda}$

It turns out the gamma distribution is conjugate for an exponential likelihood. We need to specify a prior, or a particular gamma in this case. If we think that the buses come on average every ten minutes, that's a rate of one over ten.

$$\text{prior}_\mu = \frac{1}{10}$$

Thus, we'll want to specify a gamma distribution so that the first parameter divided by the second parameter is $\frac{1}{10}$

We can now think about our variability. Perhaps you specify

$$\text{Gamma}(100, 1000)$$

This will indeed have a prior mean of $\frac{1}{10}$ and it'll have a standard deviation of $\frac{1}{100}$. If you want to have a rough estimate of our mean plus or minus two standard deviations then we have the following

Y ~ Exp(λ) Y=12
 prior mean = $\frac{1}{10}$ f(Y|λ) ∝ f(y|λ) f(λ)
 $\Gamma(100, 1000)$ $\propto \lambda^{100-1} e^{-100\lambda}$ posterior mean = $\frac{101}{1012}$
 prior std. dev. = $\frac{1}{10}$ $\propto \lambda^{100+1} e^{-100\lambda}$
 ± 0.2 $\propto \lambda^{101} e^{-100\lambda}$
 $N_{\lambda} \sim \Gamma(101, 1012)$
 $\Rightarrow N_{\lambda} \sim \Gamma(101, 1012)$

Time between buses
 Figure 16.1: Exponential likelihood with a Gamma prior

$$0.1 \pm 0.02$$

Suppose that we wait for 12 minutes and a bus arrives. Now you want to update your posterior for λ about how often this bus will arrive.

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

$$f(\lambda | y) \propto \lambda e^{-\lambda y} \lambda^{\alpha-1} e^{-\beta \lambda}$$

$$f(\lambda | y) \propto \lambda^{(\alpha+1)-1} e^{-(\beta+y)\lambda}$$

$$\lambda | y \sim \text{Gamma}(\alpha + 1, \beta + y)$$

Plugging in our particular prior gives us a posterior for λ which is

$$\lambda | y \sim \text{Gamma}(101, 1012)$$

Thus our posterior mean is going to be $\frac{101}{1012} = 0.0998$.

This one observation does not contain a lot of data under this likelihood. When the bus comes and it takes 12 minutes instead of 10, it barely shifts our posterior mean up.

One data point does not have a big impact here.

Exercise 16.1. We can generalize the result from the lesson to more than one data point.

Suppose

$$Y_1, \dots, Y_n \stackrel{iid}{\sim} \text{Exp}(\lambda) = \lambda e^{-\lambda x} \mathbb{I}_{x \geq 0}$$

with mean

$$\mathbb{E}[Y] = \frac{1}{\lambda}$$

and assume a

$$f(\lambda) = \text{Gamma}(\alpha, \beta) \quad (\text{prior for } \lambda)$$

The likelihood is then:

$$f(y | \lambda) = \prod \lambda e^{-\lambda x_i} \mathbb{I}_{x_i \geq 0} = \lambda^n e^{-\lambda \sum y_i} \cdot 1$$

and we can follow the same steps from the lesson to obtain the posterior distribution (try to derive it yourself):

$$\lambda | y \sim \text{Gamma}(\alpha + n, \beta + \sum y_i)$$

What is the prior effective sample size (ess) in this model?

Solution. The data sample size n is added to α to update the first parameter. Thus α can be interpreted as the sample size equivalent in the prior.

It might be helpful to think about a related problems...

1. We are waiting at a bus stop with 1 bus line, the information at the bus stop say that the bus comes on average every 10 minutes at this time. How long do we expect to wait for the bus?
2. what if we have waited for k minutes and the bus has not arrived yet? How long do we expect to wait for the bus?
3. While we are waiting more people arrive at the bus stop. You notice the bus stop features a digital counter and a display with long term mean E and V variance of the number of people at the bus stop. Can we use this information to get a better estimate of our bus arrival time?
4. If we wait at a bus stop with K different bus lines each with the same lambda, and we see a L people waiting. Can we get a better estimate of our bus arrival time?
5. What if more people come. And we know the mean and variance of the people waiting at the bus stop?
6. What if a different bus line arrives and the number of people waiting is now M?
7. What if each bus line has a different lambda, but we know the mean and variance of the people waiting at the bus stop?

17 Normally distributed Data

Bayesian Statistics: From Concept to Data Analysis

Normally distributed data is not that common. However, modeling using a normal RV is second to none.(Hoff 2009, 75). The CLT is the primary reason that the normal is a good approximation if there are enough IID samples. We will look at two types of *conjugate normal priors*, and in the next unit we will consider two more uninformative priors for Normally distributed data.

[Charles Zaiontz](#) provides pro types of conjugate priors for normally distributed data:

1. Mean unknown and variance known.
2. Variance unknown and mean known and
3. Mean and variance are unknown

In each case, the **unknown** refer to *population statistics*. Since we are able to estimate *sample parameters* such as the mean and variance quite easily. A key question to consider is how well does our posterior distribution of the parameter representative of the unknown population statistic?

Ideally, I will update the notes below with proofs of conjugate, prior and posterior and marginal distribution.

How good is our estimate of the known population statistic?

Some of the proofs are in [here](#) as well

- See (A. Gelman et al. 2013, sec. 3.2) for Normal data with a noninformative prior distribution
- See (A. Gelman et al. 2013, sec. 3.2) for Normal data with a conjugate prior distribution

$X_i \sim N(\mu, \sigma^2)$
 prior $\mu \sim N(m, s_0^2)$
 $f(x|\mu) \propto f(x|\mu)f(\mu)$
 $\mu|x \sim N\left(\frac{n\bar{x} + m_0}{n + \frac{1}{s_0^2}}, \frac{s_0^2}{n + \frac{1}{s_0^2}}\right)$

Figure 17.1: Normal likelihood with variance known

17.1 Normal Likelihood with known variance

See (Hoff 2009, sec. 5.2)

Let's suppose the **standard deviation or variance** σ^2 is known and we're only interested in learning about the mean. This is a situation that often arises in monitoring industrial production processes.

$$X_i \stackrel{iid}{\sim} \mathcal{N}(\mu, \sigma^2) \quad (17.1)$$

It turns out that the Normal distribution is conjugate for itself when looking for the mean parameter

Prior

$$\mu \sim \mathcal{N}(m_0, S_0^2) \quad (17.2)$$

By Bayes rule:

$$f(\mu | x) \propto f(x | \mu) f(\mu)$$

$$\mu | x \sim \mathcal{N}\left(\frac{\frac{n\bar{x}}{\sigma_0^2} + \frac{m_0}{S_0^2}}{\frac{n}{\sigma_0^2} + \frac{1}{S_0^2}}, \frac{1}{\frac{n}{\sigma_0^2} + \frac{1}{S_0^2}}\right) \quad (17.3)$$

where:

- n is the sample size
- $\bar{x} = \frac{1}{n} \sum x_i$ is the sample mean
- $\sigma^2 = \frac{1}{n} \sum (x_i - \bar{x})^2$ is the sample variance
- indexing parameters with 0 seems to be a convention that they are from the prior:
- s_0 is the prior variance
- m_0 is the prior mean

Let's look at the posterior mean

$$\begin{aligned} posterior_{\mu} &= \frac{\frac{n}{\sigma_0^2}}{\frac{n}{\sigma_0^2} + \frac{1}{s_0^2}} \bar{x} + \frac{\frac{1}{s_0^2}}{\frac{n}{\sigma_0^2} + \frac{1}{s_0^2}} m \\ &= \frac{n}{n + \frac{\sigma_0^2}{s_0^2}} \bar{x} + \frac{\frac{\sigma_0^2}{s_0^2}}{n + \frac{\sigma_0^2}{s_0^2}} m \end{aligned} \quad (17.4)$$

⚠️ Warning

should we use $n-1$ in the sample variance?

Thus we see, that the posterior mean is a **weighted average** of the prior mean and the data mean. And indeed that the **effective sample size** for this prior is the ratio of the variance for the data to the variance in the prior.

$$Prior ESS = \frac{\sigma_0^2}{s_0^2} \quad (17.5)$$

This makes sense, because the larger the variance of the prior, the less information that's in it.

The **marginal distribution** for Y is

$$\mathcal{N}(m_0, s_0^2 + \sigma^2) \quad (17.6)$$

17.1.1 Prior (and posterior) predictive distribution

The **prior (and posterior) predictive distribution** for data is particularly simple in the *conjugate normal model*.

If

$$y | \theta \sim \mathcal{N}(\theta, \sigma^2)$$

and

$$\theta \sim \mathcal{N}(m, s_0^2)$$

then the **marginal distribution** for Y , obtained as

$$\int f(y, \theta) d\theta = \mathcal{N}(m_0, s_0^2) \quad (17.7)$$

Example 17.1. Suppose your data are normally distributed with $\mu = \theta$ and $\sigma^2 = 1$.

$$y | \theta \sim \mathcal{N}(\theta, 1)$$

You select a normal prior for θ with mean 0 and variance 2.

$$\theta \sim \mathcal{N}(0, 2)$$

Then the **prior predictive distribution** for one data point would be $N(0, a)$. What is the value of a ?

Since, $m_0 = 0$, and $s_0^2 = 2$ and $\sigma^2 = 1$, the predictive distribution is $N(0, 2 + 1)$.

17.2 Normal likelihood with expectation and variance unknown

💡 Challenging

This section is challenging.

- The updating derivation is skipped,
- the posterior
- updating rule values are introduced without motivations and explanation.
- The model is also the most complicated in the course, the note at the end says this can be extended hierarchically if we want to specify hyper priors for m , w and β
- Other text discuss this case using a inverse chi squared distribution

If we can understand the model the homework is going to make sense. Also this is probably the level needed for the other courses in the specialization.

It can help to review some of the books:

The chalkboard contains the following handwritten notes:

$$Y_i | \mu, \sigma^2 \sim N(\mu, \sigma^2)$$

$$\mu | \sigma^2 \sim N(m, \frac{\sigma^2}{w})$$

$$\sigma^2 \sim \Gamma'(\alpha, \beta)$$

$$\sigma^2 | \mathbf{x}, w \sim \Gamma'(\alpha + \frac{n}{2}, \beta + \frac{1}{2} \sum_{i=1}^n (x_i - \bar{x})^2 + \frac{nw}{2(n+w)} (\bar{x} - m)^2)$$

$$\mu | \sigma^2, \mathbf{x} \sim N\left(\frac{n\bar{x} + w\bar{m}}{n+w}, \frac{\sigma^2}{n+w}\right)$$

$$\mu | \mathbf{x} \sim t$$

where $w = \frac{\sigma^2}{\sigma_x^2}$ is the effective sample size.

Figure 17.2: Normal likelihood with a unknown variance

- See (Hoff 2009, sec. 5.3) which has some R examples.
- See (A. Gelman et al. 2013, sec. 5)

If both μ and σ^2 are unknown, we can specify a **conjugate prior** in a **hierarchical fashion**.

$$X_i \mid \mu, \sigma^2 \stackrel{iid}{\sim} \mathcal{N}(\mu, \sigma^2) \quad (\text{the data given the params})$$

- This is the level 1 hierarchically model - X_i model our observations.
- We state on the left, that the RV X is conditioned on the μ and σ^2 .
- But the variables μ and σ^2 are unknown population statistics which we will need to infer from the data. We can call them latent variables.

Next we add a prior from μ conditional on the value for σ^2

$$\mu \mid \sigma^2 \sim \mathcal{N}\left(m, \frac{\sigma^2}{w}\right) \quad (\text{prior of the mean conditioned on the variance})$$

where:

- w is going to be the ratio of σ^2 and some variance for the Normal distribution. This is the **effective sample size of the prior**.
- Why is the mean conditioned on the variance. We can have a model where they are independent too?
- later on (in the homework) we are told that w can express the confidence in the prior.
- I think this means that Since this is a knowledge of m , i.e. giving w a weight of 1/10 expresses that we value
- Perhaps this is due to CLT ?
- This is level 2 of the model

Finally, the last step is to specify a prior for σ^2 . The conjugate prior here is an inverse gamma distribution with parameters α and β .

$$\sigma^2 \sim \text{Gamma}^{-1}(\alpha, \beta) \quad \text{prior of the variance}$$

After many calculations... we get the posterior distribution

$$\sigma^2 \mid x \sim \text{Gamma}^{-1}\left(\alpha + \frac{n}{2}, \beta + \frac{1}{2} \sum_{i=1}^n (x - \bar{x})^2 + \frac{nw}{2(n+2)} (\bar{x} - m)^2\right) \quad (17.8)$$

$$\mu \mid \sigma^2, x \sim \mathcal{N}\left(\frac{n\bar{x} + wm}{n+w}, \frac{\sigma^2}{n+w}\right) \quad (17.9)$$

Where the posterior mean can be written as the *weighted average* of the *prior mean* and the *data mean*.

$$\frac{n\bar{x} + wm}{n+w} = \frac{w}{n+w}m + \frac{n}{n+w}\bar{x} \quad \text{post. mean} \quad (17.10)$$

In some cases, we only care about μ . We want some inference on μ and we may want it such that it does not depend on σ^2 . We can marginalize that σ^2 integrating it out. The posterior for μ marginally follows a t distribution.

$$\mu \mid x \sim t$$

Similarly, the posterior predictive distribution also is a t distribution.

Finally, note that we can extend this in various directions, this can be extended to the multivariate normal case that requires matrix vector notations and can be extended hierarchically if we want to specify priors for m, w, β

18 Non-Informative Priors

Bayesian Statistics: Techniques and Models

18.1 Non-Informative Priors

We've seen examples of choosing priors that contain a significant amount of information. We've also seen some examples of choosing priors where we're attempting to not put too much information in to keep them vague.

Another approach is referred to as objective Bayesian statistics or inference where we explicitly try to minimize the amount of information that goes into the prior.

This is an attempt to have the data have maximum influence on the posterior

Let's go back to coin flipping

$$Y_i \sim B(\theta)$$

How do we minimize our prior information in θ ? One obvious intuitive approach is to say that all values of θ are equally likely. So we could have a prior for θ which follows a uniform distribution on the interval $[0, 1]$

Saying all values of θ are equally likely **seems** like it would have no information in it.

Recall however, that a $Uniform(0, 1)$ is the same as $Beta(1, 1)$

The effective sample size of a beta prior is the sum of its two parameters. So in this case, it has an effective sample size of 2. This is equivalent to data, with one head and one tail already in it.

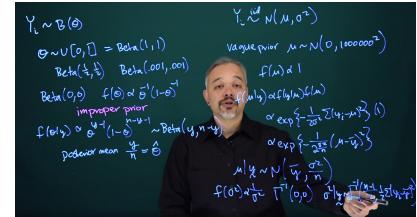


Figure 18.1: Non-Informative Priors

So this is not a completely non-informative prior.

We could think about a prior that has less information. For example $Beta(\frac{1}{2}, \frac{1}{2})$, this would have half as much information with an effective sample size of one.

We can take this even further. Think about something like $Beta(0.001, 0.001)$. This would have much less information, with the effective sample size fairly close to zero. In this case, the data would determine the posterior and there would be very little influence from the prior.

18.1.1 Improper priors

Can we go even further? We can think of the limiting case. Let's think of $Beta(0, 0)$, what would that look like?

$$f(\theta) \propto \theta^{-1}(1 - \theta)^{-1}$$

This is not a proper density. If you integrate this over $(0, 1)$, you'll get an infinite integral, so it's not a true density in the sense of it not integrating to 1.

There's no way to normalize it, since it has an infinite integral. This is what we refer to as an improper prior.

It's improper in the sense that it doesn't have a proper density. But it's not necessarily improper in the sense that we can't use it. If we collect data, we use this prior and as long as we observe one head and one tail, or **at least one success and one failure**. Then we can get a posterior

$$f(\theta | y) \propto \theta^{y-1}(1 - \theta)^{n-y-1} \sim Beta(y, n-y)$$

With a posterior mean of $\frac{y}{n} = \hat{\theta}$, which you should recognize as the maximum likelihood estimate. So by using this improper prior, we get a posterior which gives us point estimates exactly the same as the frequentist approach.

But in this case, we can also think of having a full posterior. From this, we can make interval statements, and probability statements, and we can actually find an interval and say that there's 95% probability that θ is in this

interval. This is not something you can do under the frequentist approach even though we may get the same exact interval.

18.1.2 Statements about improper priors

Improper priors are okay as long as the posterior itself is proper. There may be some mathematical things that need to be checked and you may need to have certain restrictions on the data. In this case, we needed to make sure that we observed at least one head and one tail to get a proper posterior.

But as long as the posterior is proper, we can go forwards and do Bayesian inference even with an improper prior.

The second point is that for many problems there does exist a prior, typically an improper prior that will lead to the same point estimates as you would get under the frequentist paradigm. So we can get very similar results, results that are fully dependent on the data, under the Bayesian approach.

But in this case, we can also continue to have a posterior and make posterior interval estimates and talk about the posterior probabilities of the parameter.

18.1.3 Normal Case

Another example is thinking about the normal case.

$$Y_i \stackrel{iid}{\sim} \mathcal{N}(\mu, \sigma^2)$$

Let's start off by assuming that σ^2 is known and we'll just focus on the mean μ .

We can think about a vague prior like before and say that

$$\mu \sim N(0, 1000000^2)$$

This would just spread things out across the real line. That would be a fairly non-informative prior covering a lot of possibilities. We can then think about taking the limit, what happens if we let the variance go to ∞ . In that case, we're spreading out this distribution across the entire

real number line. We can say that the density is just a constant across the whole real line.

$$f(\mu) \propto 1$$

This is an improper prior because if you integrate the real line you get an infinite answer. However, if we go ahead and find the posterior

$$f(\mu | y) \propto f(y | \mu) f(\mu) \propto \exp\left(-\frac{1}{2\sigma^2} \sum (y_i - \mu)^2\right) \quad (1)$$

$$f(\mu | y) \propto \exp\left(-\frac{1}{2\sigma^2/n} (\mu - \bar{y})^2\right)$$

$$\mu | y \sim N(\bar{y}, \frac{\sigma^2}{n})$$

This should look just like the maximum likelihood estimate.

18.1.4 Normal with unknown Variance

In the case that σ^2 is unknown, the standard non-informative prior is

$$f(\sigma^2) \propto \frac{1}{\sigma^2}$$

$$\sigma^2 \sim \Gamma^{-1}(0, 0)$$

This is an improper prior and it's uniform on the log scale of σ^2 .

In this case, we'll end up with a posterior for σ^2

$$\sigma^2 | y \sim \Gamma^{-1}\left(\frac{n-1}{2}, \frac{1}{2} \sum (y_i - \bar{y})^2\right)$$

This should also look reminiscent of the quantities we get as a frequentist. For example, the samples standard deviation

18.2 Jeffrey's Prior

Choosing a uniform prior depends upon the particular parameterization.

Suppose I used a prior which is uniform on the log scale for σ^2

$$f(\sigma^2) \propto \frac{1}{\sigma^2}$$

Suppose somebody else decides, that they just want to put a uniform prior on σ^2 itself.

$$f(\sigma^2) \propto 1$$

These are both uniform on certain scales or certain parameterizations, but they are different priors. So when we compute the posteriors, they will be different as well.

The key thing is that uniform priors are not invariant with respect to transformation. Depending on how you parameterize the problem, you can get different answers by using a uniform prior

One attempt to round this out is to use **Jeffrey's Prior**

Jeffrey's Prior is defined as the following

$$f(\theta) \propto \sqrt{\mathcal{I}(\theta)}$$

Where $\mathcal{I}(\theta)$ is the fisher information of θ .

In most cases, this will be an **improper prior**.

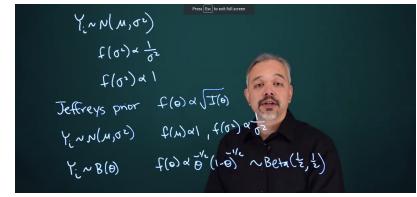


Figure 18.2: Jeffrey's Prior

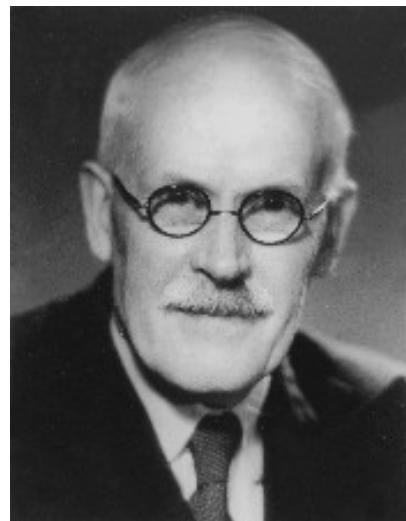


Figure 18.3: Harold Jeffreys

💡 Historical Note on Sir Harold Jeffreys

Jeffreys' Prior is due to Sir Harold Jeffreys (1891-1989) a British geophysicist who used sophisticated mathematical models to study the Earth and solar system. His hypotheses were uncertain, requiring revision in the face of incoming results, Jeffreys tried to construct a formal theory of scientific reasoning based on Bayesian probability. He made significant contributions to mathematics and statistics. His book, *Theory of Probability* (Jeffreys 1983), first published in 1939, played an important role in the revival of the objective Bayesian view of probability.

Inductive and Reductive Inference

“The fundamental problem of scientific progress, and a fundamental one of everyday life, is that of learning from experience. Knowledge obtained in this way is partly merely description of what we have already observed, but part consists of making inferences from past experience to predict future experience. This part may be called generalization or **induction**. ”

JEFFREYS’ RULES FOR A THEORY OF INDUCTIVE INFERENCE

1. All hypotheses used must be explicitly stated and the conclusions must follow from the hypotheses.

2. A theory of induction must be **self-consistent**; that is, it must not be possible to derive contradictory conclusions from the postulates and any given set of observational data.
3. Any rule given must be applicable in practice. A definition is useless unless the thing defined can be recognized in terms of the definition when it occurs. The existence of a thing or the estimate of a quantity must not involve an impossible experiment.
4. A theory of induction must provide explicitly for the possibility that inferences made by it may turn out to be wrong.
5. A theory of induction must not deny any empirical proposition a priori; any precisely stated empirical proposition must be formally capable of being accepted in the sense of the last rule, given a moderate amount of relevant evidence.
6. The number of postulates should be reduced to a minimum. (Occam's Razor)
7. Although we do not regard the human mind as a perfect reasoner, we must accept it as a useful one and the only one available. The theory need not represent actual thought processes in detail but should agree with them in outline.
8. In view of the greater complexity of induction, we cannot hope to develop it more thoroughly than deduction. We therefore take it as a rule that an objection carries no weight if an analogous objection invalidates part of generally accepted pure mathematics.

18.2.1 Normal Data

For the example of Normal Data

$$Y_i \sim N(\mu, \sigma^2)$$

$$f(\mu) \propto 1$$

$$f(\sigma^2) \propto \frac{1}{\sigma^2}$$

Where μ is uniform and σ^2 is uniform on the log scale.

This prior will then be transformation invariant. We will end up putting the same information into the prior even if we use a different parameterization for the Normal.

18.2.2 Binomial

$$Y_i \sim B(\theta)$$

$$f(\theta) \propto \theta^{-\frac{1}{2}}(1-\theta)^{-\frac{1}{2}} \sim \text{Beta}\left(\frac{1}{2}, \frac{1}{2}\right)$$

This is a rare example of where the Jeffrey's prior turns out to be a proper prior.

You'll note that this prior actually does have some information in it. It's equivalent to an effective sample size of one data point. However, this information will be the same, not depending on the parameterization we use.

In this case, we have θ as a probability, but another alternative which is sometimes used is when we model things on a logistics scale.

By using the Jeffreys prior, we'll maintain the exact same information.

18.2.3 Closing information about priors

Other possible approaches to objective Bayesian inference include priors such as reference priors and maximum entropy priors.

A related concept to this is called empirical Bayesian analysis. The idea in empirical Bayes is that you use the data to help inform your prior; such as by using the mean of the data to set the mean of the prior distribution. This approach often leads to reasonable point estimates in your posterior. However, it's sort of cheating since you're using your data twice and as a result may lead to improper uncertainty estimates.

18.3 Fisher Information

The Fisher information (for one parameter) is defined as

$$\mathcal{I}(\theta) = E \left[\left(\frac{d}{d\theta} \log(f(X | \theta)) \right)^2 \right]$$

Where the expectation is taken with respect to X which has PDF $f(X | \theta)$. This quantity is useful in obtaining estimators for θ with good properties, such as low variance. It is also the basis for Jeffrey's prior.

💡 Jeffreys prior violates the likelihood principle.

Use of the Jeffreys prior violates the strong version of the [likelihood principle](#). Which proposes that, given a statistical model, all the evidence in a sample relevant to model parameters is contained in the likelihood function. When using the Jeffreys prior, inferences about θ depend not just on the probability of the observed data as a function of θ , but also on the universe of all possible experimental outcomes, as determined by the experimental design, because the Fisher information is computed from an expectation over the chosen universe. Accordingly, the Jeffreys prior, and hence the inferences made using it, may be different for two experiments involving the same θ parameter even when the likelihood functions for the two experiments are the same a violation of the strong likelihood principle.

Example 18.1 (Jeffreys prior). Let

$$X | \theta \sim N(\theta, 1)$$

Then we have

$$f(x | \theta) = \frac{1}{\sqrt{2\pi}} \exp\left[-\frac{1}{2}(x - \theta)^2\right]$$

$$\log(f(x | \theta)) = -\frac{1}{2} \log(2\pi) - \frac{1}{2}(x - \theta)^2$$

$$\left(\frac{d}{d\theta} \log(f(x | \theta)) \right)^2 = (x - \theta)^2$$

and so

$$\mathcal{I}(\theta) = \mathbb{E}[(X - \theta)^2] = \text{Var}[X] = 1$$

18.4 Sensitivity analysis of priors

The general approach to using priors in models is to start with some justification for a prior, run the analysis, then come up with competing priors and reexamine the conclusions under the alternative priors. The results of the final model and the analysis of the sensitivity of the analysis to the choice of prior are written up as a package.

For a discussion of steps and methods to use in a sensitivity analysis, see: (A. Gelman et al. 2013, page: 38) which discusses two approaches:

Many times we choose priors out of convenience. How to judge when assumptions of convenience can be made safely is a central task of Bayesian sensitivity analysis.

1. Analysis using different conjugate prior distributions.
 - Starting with a uniform prior
 - More informative priors are tested and the 95% posterior CI is compared against the posterior mean and the prior mean.
 - A table of prior mean, prior effective sample size , posterior mean and posterior 95 CI is created for the results
 - We are interested primarily to see how well the the posterior CI can excludes the prior mean even for priors with large effective sample size.
2. Analysis using a non-conjugate prior distribution follows the same approach but uses non conjugate prior. The comparisons described in 1. can be carried out using sampling.

(A. Gelman et al. 2013, pages: 141)

19 Brief Review of Regression

Bayesian Statistics: Techniques and Models

Recall that linear regression is a model for predicting a response or dependent variable (Y , also called an output) from one or more covariates or independent variables (X , also called explanatory variables, inputs, or features). For a given value of a single x , the expected value of y is

$$\mathbb{E}[y] = \beta_0 + \beta_1 x$$

or we could say that

$$Y \sim \mathcal{N}(\beta_0 + \beta_1 x, \sigma^2)$$

For data $(x_1, y_1), \dots, (x_n, y_n)$, the fitted values for the coefficients, $\hat{\beta}_0$ and $\hat{\beta}_1$ are those that minimize the sum of squared errors $\sum_{i=1}^n (y_i - \hat{y}_i)^2$, where the predicted values for the response are $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$. We can get these values from R. These fitted coefficients give the least-squares line for the data.

This model extends to multiple covariates, with one β_j for each k covariates

$$\mathbb{E}[y_i] = \beta_0 + \beta_1 x_{i1} + \dots + \beta_k x_{ik}$$

Optionally, we can represent the multivariate case using vector-matrix notation.

19.1 Conjugate Modeling

In the Bayesian framework, we treat the β parameters as unknown, put a prior on them, and then find the posterior. We might treat σ^2 as fixed and known, or we might treat it as an unknown and also put a prior on it. Because the underlying assumption of a regression model is that the errors are independent and identically normally distributed with mean 0 and variance σ^2 , this defines a normal likelihood.

19.1.1 σ^2 known

Sometimes we may know the value of the error variance σ^2 . This simplifies calculations. The conjugate prior for the β is a normal prior. In practice, people typically use a non-informative prior, i.e., the limit as the variance of the normal prior goes to infinity, which has the same mean as the standard least-squares estimates. If we are only estimating β and treating σ^2 as known, then the posterior for β is a (multivariate) normal distribution. If we just have a single covariate, then the posterior for the slope is:

$$\beta_1 | y \sim N\left(\frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}, \frac{\sigma^2}{\sum_{i=1}^n (x_i - \bar{x})^2}\right)$$

If we have multiple covariates, then using a matrix-vector notation, the posterior for the vector of coefficients is

$$\beta | y \sim N((X^t X)^{-1} X^t y, (X^t X)^{-1} \sigma^2)$$

where X denotes the design matrix and X^t is the transpose of X . The intercept is typically included in X as a column of 1's. Using an improper prior requires us to have at least as many data points as we have parameters to ensure that the posterior is proper.

19.1.2 σ^2 Unknown

If we treat both β and σ^2 as unknown, the standard prior is the non-informative Jeffreys prior, $f(\beta, \sigma^2) \propto \frac{1}{\sigma^2}$. Again, the posterior mean for β will be the same as the standard least-squares estimates. The posterior for β conditional on σ^2 is the same normal distributions as when σ^2 is known, but the marginal posterior distribution for β , with σ^2 integrated

out is a t distribution, analogous to the t tests for significance in standard linear regression. The posterior t distribution has mean $(X^t X)^{-1} X^t y$ and scale matrix (related to the variance matrix) $s^2(X^t X)^{-1}$, where $s^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 / (n - k - 1)$. The posterior distribution for σ^2 is an inverse gamma distribution

$$\sigma^2 | y \sim \Gamma^{-1}\left(\frac{n - k - 1}{2}, \frac{n - k - 1}{2}s^2\right)$$

In the simple linear regression case (single variable), the marginal posterior for β is a t distribution with mean $\frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^n (x_i - \bar{x})^2}$ and scale $\frac{s^2}{\sum_{i=1}^n (x_i - \bar{x})^2}$. If we are trying to predict a new observation at a specified input x^* , that predicted value has a marginal posterior predictive distribution that is a t distribution, with mean $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x^*$ and scale $se_r \sqrt{1 + \frac{1}{n} + \frac{(x^* - \bar{x})^2}{(n-1)s_x^2}}$. se_r is the residual standard error of the regression, which can be found easily in R. s_x^2 is the sample variance of x . Recall that the predictive distribution for a new observation has more variability than the posterior distribution for \hat{y} , because individual observations are more variable than the mean.

19.2 Linear Regression

19.2.1 Single Variable Regression

We'll be looking at the Challenger dataset. It contains 23 past launches where it has the temperature at the day of launch and the O-ring damage index

[Challenger dataset](#)

Read in the data <https://pdixon.stat.iastate.edu/stat511/datasets/challenger2.txt>

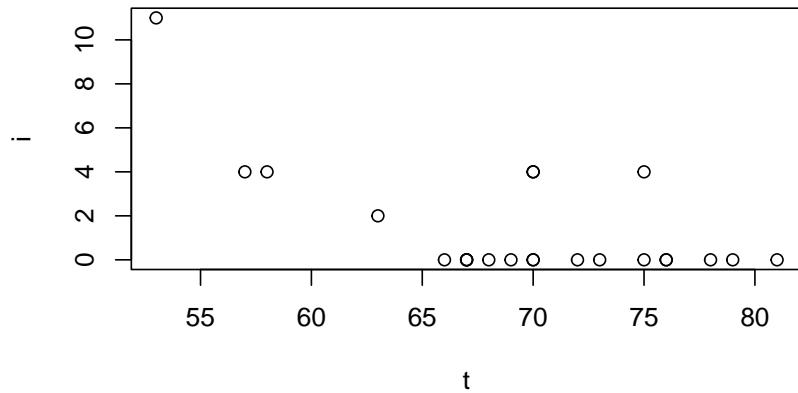
```
oring=read.table("data/challenger.txt", header=T)
# Note that attaching this masks T which is originally TRUE
attach(oring)

head(oring)
```

```
t i
1 53 11
2 57 4
3 58 4
4 63 2
5 66 0
6 67 0
```

Now we'll see the plot

```
plot(t,i)
```



Fit a linear model

```
oring.lm = lm(i ~ t)
summary(oring.lm)
```

```
Call:
lm(formula = i ~ t)
```

Residuals:

Min	1Q	Median	3Q	Max
-2.3025	-1.4507	-0.4928	0.7397	5.5337

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	18.36508	4.43859	4.138	0.000468 ***

```

t           -0.24337    0.06349   -3.833  0.000968 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.102 on 21 degrees of freedom
Multiple R-squared:  0.4116,    Adjusted R-squared:  0.3836
F-statistic: 14.69 on 1 and 21 DF,  p-value: 0.0009677

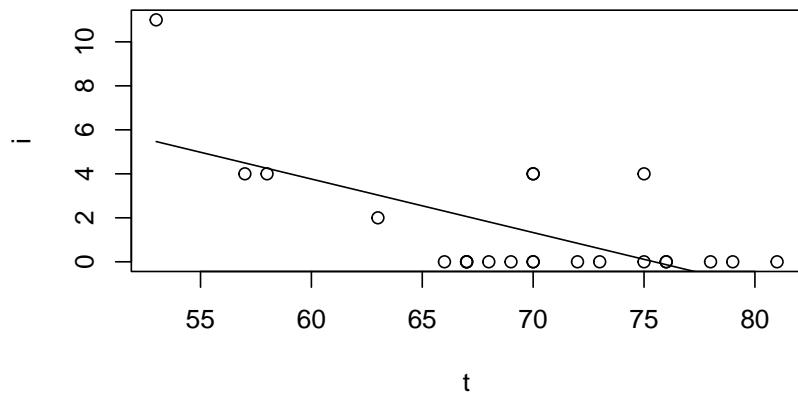
```

Add the fitted line into the scatter plot

```

plot(t,i)
lines(t,fitted(oring.lm))

```



Create a 95% posterior interval for the slope

```

-0.24337 - 0.06349*qt(.975,21)

```

```
[1] -0.3754047
```

```
-0.24337 + 0.06349*qt(.975,21)
```

```
[1] -0.1113353
```

Note: These are the same as the frequentist confidence intervals

If the challenger launch was at 31 degrees Fahrenheit, how much O-Ring damage would we predict?

```
coef(oring.lm)[1] + coef(oring.lm)[2]*31
```

```
(Intercept)  
10.82052
```

```
# [1] 10.82052
```

Let's make our posterior prediction interval

```
predict(oring.lm,data.frame(t=31),interval="predict")
```

	fit	lwr	upr
1	10.82052	4.048269	17.59276

We can calculate the lower bound through the following formula

```
10.82052 - 2.102 * qt(.975, 21) * sqrt(1+1/23 + ((31 - mean(T))^2 / 22 / var(T)))
```

```
[1] 4.850937
```

What's the posterior probability that the damage index is greater than zero?

```
1 - pt((0 - 10.82052) / (2.102 * sqrt(1+1/23 + ((31 - mean(T))^2 / 22 / var(T)))), 21)
```

```
[1] NA
```

19.2.2 Multivariate Regression

We're looking at Galton's seminal data predicting the height of children from the height of the parents.

	Family	Father	Mother	Gender	Height	Kids
1	1	78.5	67.0	M	73.2	4
2	1	78.5	67.0	F	69.2	4
3	1	78.5	67.0	F	69.0	4
4	1	78.5	67.0	F	69.0	4
5	2	75.5	66.5	M	73.5	4
6	2	75.5	66.5	M	72.5	4
7	2	75.5	66.5	F	65.5	4
8	2	75.5	66.5	F	65.5	4

What are the columns in the dataset?

```
names(heights)
```

```
[1] "Family" "Father" "Mother" "Gender" "Height" "Kids"
```

```
# [1] "Family" "Father" "Mother" "Gender" "Height" "Kids"
```

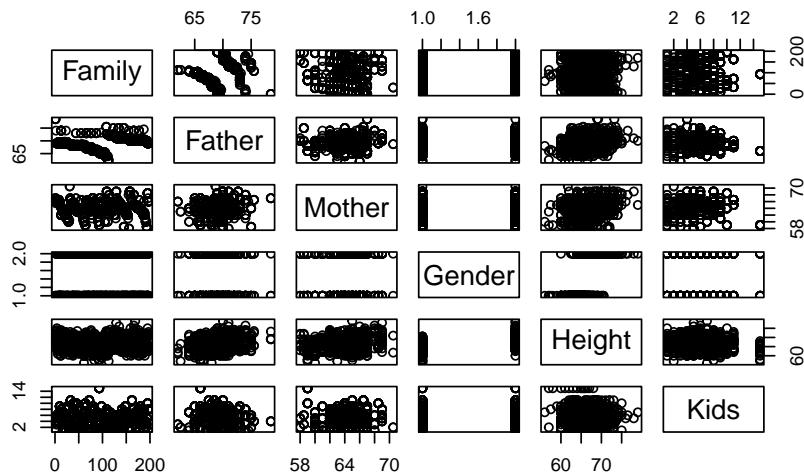
explanation of the columns:

- Family: the family the child is from
- Father: height of the father
- Mother: height of the mother
- Kids: count of children in the family
- Gender: the gender of the child
- Height: the height the child

The Height is our target variable.

Let's look at the relationship between the different variables

```
pairs(heights)
```



Pair plots are a great tool for doing EDA in R. You need to get used read them.

We care primarily about the Height so we can should first consider the row of the height. The other rows can inform us if there is a relation between other variables.

- the Father and Mother are correlated with height.
- Gender male children are generally taller.
- Kids and Family don't seem to have a clear pattern.

First let's start by creating a linear model taking all of the columns into account

```
summary(lm(Height~Father+Mother+Gender+Kids))
```

Call:

```
lm(formula = Height ~ Father + Mother + Gender + Kids)
```

Residuals:

Min	1Q	Median	3Q	Max
-9.4748	-1.4500	0.0889	1.4716	9.1656

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	16.18771	2.79387	5.794	9.52e-09 ***
Father	0.39831	0.02957	13.472	< 2e-16 ***

```

Mother      0.32096   0.03126  10.269 < 2e-16 ***
GenderM     5.20995   0.14422  36.125 < 2e-16 ***
Kids       -0.04382   0.02718  -1.612    0.107
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.152 on 893 degrees of freedom
Multiple R-squared:  0.6407,    Adjusted R-squared:  0.6391
F-statistic: 398.1 on 4 and 893 DF,  p-value: < 2.2e-16

```

As you can see here, the `Kids` column is not statistically significant. Let's look at a model with it removed.

```

heights.lm=lm(Height~Father+Mother+Gender)
summary(heights.lm)

```

```

Call:
lm(formula = Height ~ Father + Mother + Gender)

Residuals:

```

Min	1Q	Median	3Q	Max
-9.523	-1.440	0.117	1.473	9.114

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	15.34476	2.74696	5.586	3.08e-08 ***
Father	0.40598	0.02921	13.900	< 2e-16 ***
Mother	0.32150	0.03128	10.277	< 2e-16 ***
GenderM	5.22595	0.14401	36.289	< 2e-16 ***

```

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

Residual standard error: 2.154 on 894 degrees of freedom
Multiple R-squared:  0.6397,    Adjusted R-squared:  0.6385
F-statistic: 529 on 3 and 894 DF,  p-value: < 2.2e-16

```

This model looks good. We can tell from the summary that:

- each extra inch of the father's height contributes an extra 0.4 inches height of the child.

- each extra inch of the mother's height contributes an extra 0.3 inches height of the child.
- male gender contributes 5.2 inches to the height of the child.

Let's create a 95% posterior interval for the difference in height by gender

```
5.226 - 0.144 * qt(.975, 894)
```

```
[1] 4.943383
```

```
5.226 + 0.144 * qt(.975, 894)
```

```
[1] 5.508617
```

Let's make a **posterior prediction interval** for a male and female with a father whose 68 inches and a mother whose 64 inches.

```
predict(heights.lm, data.frame(Father=68, Mother=64, Gender="M"), interval="predict")
```

	fit	lwr	upr
1	68.75291	64.51971	72.9861

```
predict(heights.lm, data.frame(Father=68, Mother=64, Gender="F"), interval="predict")
```

	fit	lwr	upr
1	63.52695	59.29329	67.76062

20 Statistical Modeling and Monte Carlo estimation

Bayesian Statistics: Techniques and Models

20.1 Objectives

- What are the objectives of statistical models?
- What can they accomplish and where do they fit in the broader field of data science?

This course is about statistical modelling which falls under the analyzing data objective.

- So what is a statistical model?
- A statistical model will be a mathematical structure used to **immitate, And approximate, the data generating process**. It typically describes relationships among variables while accounting for uncertainty and variability in the data.

For what kinds of problems might we use a statistical model?

1. Quantifying uncertainty:
 1. are relationships between variables we cannot measure?
 2. how many people were polled?
 3. how were they chosen?
 4. how would the data change if we repeated the poll.
2. Inference
 1. Extend the result and infer what percentage of the total population supports the candidate?

2. We may also have other demographic information about each person in the poll.

A statistical model might allow us to see how these other variables relate to a person's likelihood of supporting the candidate.

3. Measure support for hypothesis

1. Does the evidence support a hypothesis that the candidate is more popular with men than women?

4. Prediction

1. Given demographic information on a voter we could use the model to predict her vote.
 2. Also important for machine learning.

20.2 A Poll for a political candidate

- 57% for a candidate
- the 99% CI (51,63)
- demographics:
 - 55% women
 - 63% men

Which objective does statistical modeling most share with machine learning?

Solution. Prediction - This objective is vital to both statistics and machine learning, and is paramount in machine learning.

20.3 Modeling Process

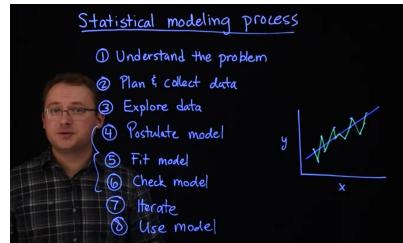


Figure 20.1: statistical modeling process

Building statistical models is a process, and each step should be taken carefully. Here we outline the general process and offer some practical advice. We'll call this the **statistical modeling process**.

The first step in this process is to *understand the problem*. This may seem obvious, but understanding the problem and context is critical to success. A sophisticated model might be useless if it is applied inappropriately.

Example 20.1 (international stores). For example, suppose you have revenue data from several different locations of a store chain at unknown locations.

- It seems reasonable to average these revenue numbers as a summary of how the store is doing.
- Suppose you discover that the stores are located in different countries and reported revenues in different currencies.
- Now that average doesn't seem to have much meaning unless, of course, we get the revenue numbers converted to the same scale.

The second step is to *plan and properly collect relevant data*. There may be multiple quantities that you could potentially measure to help answer your question. In this step, you decide what information will be most useful to solving your problem. How to collect the data and how many data points to collect. The quality of your data collection plan determines the value of your data.

For example, if you conduct a survey of peers in your workplace. Your results would likely not generalize to all workers in the company, especially if there are multiple work sites. If you want generalizable results, a better

understand the problem

plan and collect data

plan would be to select a random sample among all employees to participate in your survey.

The step is addressed in detail in most introductory statistics courses.

The third step in this process is to *explore your data*. In this step, you should ensure that the data collection plan was followed. And that the data were recorded accurately. If there are major surprises in that data, verify that they are not errors. In this step, you'll often want to visualize the data to gain a basic understanding of the relationships among your variables. This can help you decide what kinds of models might be appropriate. Finally, the practice of snooping around or mining your data, looking for interesting hypothesis to test can potentially invalidate your statistical modeling results. If you want to mine your data, and test your findings, it is usually best to randomly split your data into two parts. With one part, you can look for interesting things to test and fit different potential models. With the other, you can fit the model you chose using the first part to validate or see if the results can be replicated on other data.

explore your data

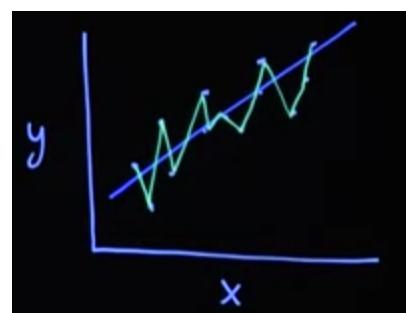
The fourth step in this process is to postulate a model. After gaining an understanding of how your data are structured, choose a model that can appropriately approximate or summarize the interesting content of the data. This might be an off the shelf statistical model like a regression or it could be based on a scientific theory such as a growth model. It could also be multiple models.

postulate a model

Generally, it is desirable to find a model where the parameters we estimate can be interpreted in the context of the original problem. You might also have to strike a balance between model complexity, and model generalizability. This is often referred to as **the bias variance trade-off**. Large complex models, might be able to fit your particular dataset very well. But may fail to generalize to future data.

Example 20.2 (overfitting). Let's look at an example of this. Let's suppose your data looked like where x is your *explanatory variable*, y is your *response variable*. And you have points like these. One possible model you could fit would be just a linear regression going through the points.

Another possibility for model you could fit here would be essentially an interpolator that makes sure it goes through every single point. Now, consider a future scenario where you've got another dataset just like this one with a new cloud of points. You can imagine that perhaps this interpolated



model, which fit the original dataset perfectly, might struggle on a future dataset.

The fifth step in our statistical modeling process is to *fit the model*. In this step we need to estimate the parameters of the model using the data. In this particular class, we're going to take a Bayesian approach to this step.

The sixth step in our statistical modeling process is to *check the model*. Here we want to check to see if the model adequately imitates the data generating process. Are predictions from the model realistic? Does it fit well to your data? Or does it completely miss some of the features? We'll look into some the techniques for doing this, including residual analysis and predictive checks later in the course. In this step, we may also compare a competing models according to some criteria.

The seventh step in our statistical modeling process is to *iterate*. That is, return, possibly, to steps 4 through 6. If the model you have already fit is, for some reason, inadequate, we should return to step 4 and proceed through step 6 again with a new, and hopefully better, model that would address or correct the deficiencies from your previous model.

The eighth and final step in our statistical modeling process is to *use the model*. If we've iterated through these enough times and decided that the model is good, or that we have selected an appropriate model, we can use the results to answer your original research questions and arrive at conclusions. In this course, we are going to focus on steps 4 through 8. But this does not mean steps 1 through 3 should be ignored in any analysis. In fact, the importance of steps 1 through 3 cannot be overstated. The validity of your final results depends on them. That is why most introductory statistics courses will emphasize these steps. If you'll not explore this issues in an introductory statistics course, we highly recommend you do so. We hope you'll refer to this outline, the statistical modeling process often as you begin modeling data.

20.3.1 Process outline:

1. understand the problem.
2. plan and collect data.
3. explore the data.
4. postulate the model.
5. fit the model.

fit the model

check the model

iterate

use the model

6. check the model.
7. iterate by going back to step 4.
8. use the model.

21 Bayesian Modeling

Bayesian Statistics: Techniques and Models

22 Notes - Bayesian Modeling

22.1 Components of a Bayesian Model

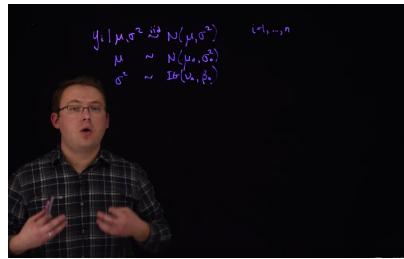


Figure 22.1: a Bayesian Model

In lesson one, we defined a **statistical model** as *a mathematical structure used to imitate or approximate the data generating process*. It incorporates uncertainty and variability using the theory of probability. A model could be very simple, involving only one variable.

Suppose our data consists of the heights of $N = 15$ adult men. . Clearly it would be very expensive or even impossible to collect the genetic information that fully explains the variability in these men's heights. We only have the height measurements available to us. To account for the variability, we might assume that the men's heights follow a normal distribution.

So we could write the model like this: where y_i will represent the height for person i , i will be our index. This will be equal to a constant, a number μ which will represents the mean for all men plus ϵ_i . the individual error term for individual i .

We're going to assume that ϵ_i comes from a normal distribution with mean zero and variance σ^2 . We are also going to assume that these ϵ_i s are independent and identically distributed from this normal distribution. This is also for i equal to 1 up to N which will be 15 in our case. Equivalently we could write this model directly for the y_i themselves.

heights of $N = 15$ men

$$y_i = \mu + \epsilon_i$$

$$\epsilon_i \stackrel{iid}{\sim} N(0, \sigma^2) \quad i \in 1 \dots N$$

So each y_i comes from a normal distribution independent and identically distributed with the normal distribution. With mean μ and variance σ^2 . This specifies a probability distribution and a model for the data.

$$y_i \stackrel{iid}{\sim} N(\mu, \sigma^2) \quad i \in 1 \dots N$$

i heights of men

$$\begin{aligned} y_i &= \mu + \epsilon_i, \\ \epsilon_i &\stackrel{iid}{\sim} N(0, \sigma^2) \end{aligned}$$

another way to write this:

$$y_i \stackrel{iid}{\sim} N(\mu, \sigma^2)$$

If we know the values of μ and σ . It also suggests how we might generate more fake data that behaves similarly to our original data set.

A model can be as simple as the one right here or as complicated and sophisticated as we need to capture the behavior of the data. So far, this model is the same for Frequentists and Bayesians.

As you may recall from the previous course. The frequentist approach to fitting this model right here would be to consider μ and σ to be fixed but unknown constants, and then we would estimate them. To calculate our uncertainty in those estimates a frequentist approach would consider how much the estimates of μ and σ might change if we were to repeat the sampling process and obtain another sample of 15 men, over, and over.

Figure 22.2: Components of a Bayesian Model

The Bayesian approach, the one we're going to take in this class. Tackles our uncertainty in μ and σ^2 with probability directly. By treating them as random variables with their own probability distributions. These are often called **priors**, and they complete a Bayesian model.

In the rest of this segment, we're going to review three key components of Bayesian models. That were used extensively in the previous course. The three primary components of Bayesian models that we often work with are the **likelihood**, the **prior** and the **posterior**.

The likelihood is the probabilistic model for the data. It describes how, given the unknown parameters, the data might be generated. We're going to call unknown parameter theta right here. Also, in this expression, you might recognize this from the previous class, as describing a probability distribution.

$$\Pr(y | \theta) \text{ (likelihood)}$$

The prior, the next step, is the probability distribution that characterizes our uncertainty with the parameter theta. We're going to write it as $\Pr(\theta)$. It's not the same distribution as this one. We're just using this notation p to represent the probability distribution of theta. By specifying a likelihood and a prior.

$$\Pr(\theta) \text{ (prior)}$$

We now have a **joint probability model** for both the knowns, the data, and the unknowns, the parameters. We can see this by using the chain rule of probability. If we wanted the joint distribution of both the data and the parameters theta. Using the chain rule of probability, we could start with the distribution of θ . And multiply that by the probability or the distribution of y given theta. That gives us an expression for the joint distribution. **However if we're going to make inferences about data and we already know the values of y** , we don't need the *joint distribution*, what we need is the *posterior distribution*.

$$\Pr(y, \theta) = \Pr(\theta)\Pr(y | \theta) \text{ (joint probability)}$$

The **posterior distribution** is the distribution of $\Pr(\theta | y)$, i.e. θ given y . We can obtain this expression using the laws of conditional probability and specifically using Bayes' theorem.

$$\Pr(\theta | y) \text{ (posterior)}$$

$$\begin{aligned}\Pr(\theta | y) &= \frac{\Pr(\theta, y)}{\Pr(y)} \\ &= \frac{\Pr(\theta, y)}{\int \Pr(\theta, y)} \\ &= \frac{\Pr(y | \theta) \Pr(\theta)}{\int \Pr(y | \theta) \Pr(\theta) d\theta}\end{aligned}$$

We start with the definition of conditional probability (1). The conditional distribution, $\Pr(\theta | y)$ is the ratio of the *joint distribution* of θ and y , i.e. $\Pr(\theta, y)$; with the *marginal distribution* of y , $\Pr(y)$.

How do we get the marginal distribution of y ?

We start with the **joint distribution** like we have on top, and we *integrate out* or *marginalize* over the values of theta (2)

To make this look like the Bayes theorem that we're familiar with the joint distribution can be rewritten as the product of the prior and the likelihood. We start with the likelihood, because that's how we usually write Bayes' theorem. We have the same thing in the denominator here. But we're going to integrate over the values of theta. These integrals are replaced by summations if we know that θ is a discrete random variable. The *marginal distribution* is another important piece which we may use when we more advanced Bayesian modeling.

The **posterior distribution** is our primary tool for achieving the statistical modeling objectives from lesson one.

i anatomy of a posterior probability

$$\begin{aligned}
 & \mathbb{P}r(y | \theta) && (\textit{likelihood}) \\
 & \mathbb{P}r(\theta) && (\textit{prior}) \\
 & \mathbb{P}r(y, \theta) = \mathbb{P}r(\theta)\mathbb{P}r(y|\theta) && (\textit{joint distribution}) \\
 & \mathbb{P}r(\theta | y) = \frac{\mathbb{P}r(\theta, y)}{\mathbb{P}r(y)} && (\textit{conditional probability}) \\
 & = \frac{\mathbb{P}r(\theta, y)}{\int \mathbb{P}r(\theta, y)} \\
 & = \frac{\mathbb{P}r(y | \theta) \mathbb{P}r(\theta)}{\int \mathbb{P}r(y | \theta) \mathbb{P}r(\theta) d\theta}
 \end{aligned} \tag{22.1}$$

Whereas non-Bayesian approaches consider a probability model for the data only, the hallmark characteristic of Bayesian models is that they specify a joint probability distribution for both data *and* parameters. How does the Bayesian paradigm leverage this additional assumption?

- This allows us to make probabilistic assessments about how likely our particular data outcome is under any parameter setting.
- This allows us to select the most accurate prior distribution.
- This allows us to make probabilistic assessments about hypothetical data outcomes given particular parameter values.

- ☒ This allows us to use the laws of conditional probability to describe our updated information about parameters given the data.

22.2 Model Specification

Before fitting any model we first need to specify all of its components.

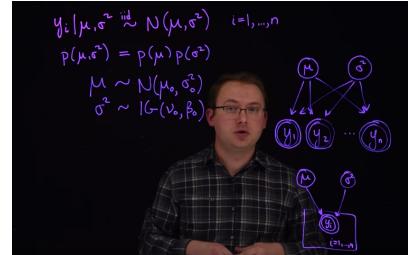
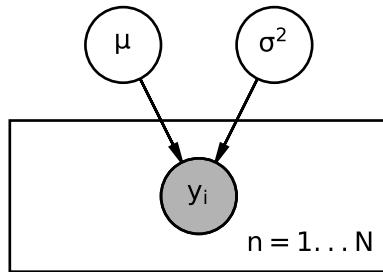


Figure 22.3: The graphical model specification for the height model

22.2.1 Hierarchical representation

One convenient way to do this is to write down the **hierarchical form of the model**. By hierarchy, we mean that the model is specified in steps or in layers. We usually start with the model for the data directly, or the likelihood. Let's write, again, the model from the previous lesson.

We had the height for person i , given our parameters μ and σ^2 , so conditional on those parameters, y_i came from a normal distribution that was independent and identically distributed, where the normal distribution has mean μ and variance σ^2 , and we're doing this for individuals 1 up to N , which was 15 in this example.

The next level that we need is the prior distribution from μ and σ^2 . For now we're going to say that they're independent priors. So that our prior from μ and σ^2 is going to be able to factor into the product of two independent priors. We can assume independents in the prior and still get dependents in the posterior distribution.

$$y_i | \mu, \sigma^2 \stackrel{iid}{\sim} N(\mu, \sigma^2) \text{ for } i \in 1, \dots, 15$$

$$\Pr(\mu, \sigma^2) = \Pr(\mu)\Pr(\sigma^2) \text{ (independence)}$$

In the previous course we learned that *the conjugate prior* for μ , if we know the value of σ^2 , is a *normal distribution*, and that the conjugate prior for σ^2 when μ is known is the *inverse gamma distribution*.

Let's suppose that our prior distribution for μ is a normal distribution where mean will be μ_0 . This is just some number that you're going to fill in here when you decide what the prior should be. Mean μ_0 , and less say σ_0^2 would be the variance of that prior.

The prior for σ^2 will be inverse gamma which has two parameters:

$$\mu \sim N(\mu_0, \sigma_0^2)$$

$$\sigma^2 \sim IG(\nu_0, \beta_0)$$

- It has a *shape parameter*, we're going to call that ν_0 , and
- It has a *scale parameter*, we'll call that β_0 .

We need to choose values for these hyper-parameters here. But we do now have a complete Bayesian model.

We now introduce some new ideas that were not presented in the previous course.

i Hierarchical representation

By hierarchy, we mean that the model is specified in steps or in layers.

- start with the model for the data, or the likelihood.
- write the priors
- add hyper-priors for the parameters of the priors.

More details can be seen on this [wikipedia article](#) and on this [one](#)

22.2.2 Graphical representation

Another useful way to write out this model is using what's called a **graphical representation**. To write a graphical representation, we're going to do the reverse order, we'll start with the priors and finish with the likelihood.

In the graphical representation we draw what are called nodes so this would be a node for μ . The **circle** means that this is a random variable that has its own distribution. So μ with its prior will be represented with that. And then we also have σ^2 . The next part of a graphical model is showing the dependence on other variables. Once we have the parameters, we can generate the data.

For example we have y_1, \dots, y_n . These are also random variables, so we'll create these as nodes. And I'm going to double up the circle here to indicate that these nodes are observed, you see them in the data. So we'll do this for all of the y s here. And to indicate the dependence of the distributions of the y s on μ and σ^2 , we're going to draw arrows. So μ influences the distribution of y for each one of these y s. The same is true for sigma squared, the distribution of each y depends on the distribution of σ^2 . Again, these nodes right here, that are double-circled, mean that they've been observed. If they're shaded, which is the usual case, that also means that they're observed. The arrows indicate the dependence between the random variables and their distributions.

Notice that in this hierarchical representation, I wrote the dependence of the distributions also. We can simplify the graphical model by writing exchangeable random variables and I'll define exchangeable later.

We're going to write this using a representative of the y s here on what's called the **plate**. So I'm going to re draw this hierarchical structure, we have μ and σ^2 . And we don't want to have to write all of these notes again. So I'm going to indicate that there are n of them, And I'm just going to draw one representative, y_i . And they depend on μ and σ^2 . To write a model like this, we must assume that the y s are *exchangeable*. That means that the distribution for the y s does not change if we were to switch the index label like the i on the y there. So, if for some reason, we knew that one of the y s was different from the other y s in its distribution, and if we also know which one it is, then we would need to write a separate node for it and not use a plate like we have here.

i Graphical representation

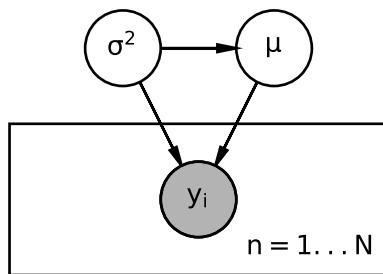


Figure 22.4: pgm-posterior

In the graphical representation we start at the top by drawing:

- circle nodes for the hyper-parameters.
- arrows indicating that they determine the
- nodes for the priors.
- nodes for the RVs (doubled circles)
- plates (rectangles) indicating RVs that are exchangeable. We add an index to the corner of the plate to indicate the amount of replicated RVs

More details can be seen on this [wikipedia article](#)

Both the hierarchical and graphical representations show how you could hypothetically **simulate data** from this model. You start with the variables that don't have any dependence on any other variables. You would simulate those, and then given those draws, you would simulate from the distributions for these other variables further down the chain.

This is also how you might simulate from a prior predictive distribution.

22.3 Posterior derivation

$$\begin{aligned}
 y_i | \mu, \sigma^2 &\sim N(\mu, \sigma^2) \\
 \mu | \sigma^2 &\sim N(\mu_0, \frac{\sigma^2}{\omega_0}) \\
 \sigma^2 &\sim IG(v_0, \beta_0)
 \end{aligned}$$

$$\begin{aligned}
 p(y_1, \dots, y_n | \mu, \sigma^2) &= p(y_1, \dots, y_n | \mu, \sigma^2) p(\mu | \sigma^2) p(\sigma^2) \\
 &= \prod_{i=1}^n [N(y_i | \mu, \sigma^2)] \cdot N(\mu | \mu_0, \frac{\sigma^2}{\omega_0}) IG(\sigma^2 | v_0, \beta_0) \\
 &\propto p(\mu, \sigma^2 | y_1, \dots, y_n) \quad p(\theta | y) = \frac{p(y|\theta)p(\theta)}{\int p(y|\theta)p(\theta)d\theta} \\
 &\propto p(\mu | \sigma^2) p(\sigma^2 | y_1, \dots, y_n) \quad \propto p(y | \theta) p(\theta)
 \end{aligned}$$

Figure 22.5: Posterior derivation

So far, we've only drawn the model with two levels. But in reality, there's nothing that will stop us from adding more layers.

For example, instead of fixing the values for the hyper parameters in the previous segment, those hyper parameters were the μ_0 , the σ_0 , the v_0 and the β_0 .

We could specify just fixed numeric values for those, or we could learn them from the data and model them using additional prior distributions for those variables to make this a hierarchical model.

One reason we might do this is if the data are hierarchically organized so that the observations are naturally grouped together. We will examine these types of hierarchical models in depth later in the course.

Another simple example of a hierarchical model is one you saw already in the previous course.

Let's write it as $y_i | \mu, \sigma^2$, so this is just like the model from the previous lesson, will be independent and identically distributed normal with a mean μ and a variance, σ^2 . The next step, instead of doing independent priors for μ and σ^2 , we're going to have the prior for μ depend on the value of σ^2 . That is given σ^2 , μ follows a normal distribution with mean μ_0 naught, just some hyper parameter that you're going to chose. And the variance of this prior will be σ^2 , this parameter, divided by omega naught. Another hyper parameter that will scale it.

We now have a joint distribution of y and μ given σ^2 . So finally, we need to complete the model with the prior for σ^2 . We'll use our standard inverse gamma with the same hyper parameters as last time. This model has three layers. And μ depends on sigma right here. The graphical representation

for this model looks like this. We start with the variables that don't depend on anything else. So that would be σ^2 and move down the chain.

So here, the next variable is μ which depends on σ^2 . And then dependent on both, we have the y_i 's. We use a double circle because the y_i 's are observed, their data, and we're going to assume that they're exchangeable. So let's put them on a plate here for i in 1 to n . The distribution of y_i depends on both μ and σ^2 , so we'll draw curves connecting those pieces there. To simulate hypothetical data from this model, we would have to first draw from the distribution of the prior for σ^2 . Then the distribution for μ which depends on σ^2 . And once we've drawn both of these, then we can draw random draws from the y_i 's, which of course depends on both of those. With multiple levels, this is an example of a hierarchical model. Once we have a model specification, we can write out what the full posterior distribution for all the parameters given the data looks like. Remember that the numerator in Bayes' theorem is the joint distribution of all random quantities, all the nodes in this graphical representation over here from all of the layers. So for this model that we have right here, we have a joint distribution that'll look like this. We're going to write the joint distribution of everything y_1 up to y_n , μ and σ^2 . Using the chain rule of probability, we're going to multiply all of the distributions in the hierarchy together. So let's start with the likelihood piece. And we'll multiply that by the next layer, the distribution of μ , given σ^2 . And finally, with the prior for σ^2 . So what do these expressions right here look like? The likelihood right here in this level because they're all independent will be a product of normal densities. So we're going to multiply the normal density for each y_i , Given those parameters. This, again, is shorthand right here for the density of a normal distribution. So that represents this piece right here. The conditional prior of μ given σ^2 is also a normal. So we're going to multiply this by a normal distribution of μ , where its parameters are μ_{naught} and σ^2 over ω_{naught} . And finally, we have the prior for σ^2 . We'll multiply by the density of an inverse gamma for σ^2 given the hyper parameters μ_{naught} , sorry, that is given, the hyper parameters μ_{naught} and β_{naught} . What we have right here is the joint distribution of everything. It is the numerator in Bayes theorem. Let's remind ourselves really fast what Bayes theorem looks like again. We have that the posterior distribution of the parameter given the data is equal to the likelihood, Times the prior. Over the same thing again. So this gives us in the numerator the joint distribution of everything which is what we've written right here.

In Bayes theorem, the numerator and the denominator are the exact same

expression accept that we integrate or marginalize over all of the parameters.

Because the denominator is a function of the y 's only, which are known values, the denominator is just a constant number. So we can actually write the posterior distribution as being proportional to, this symbol right here represents proportional to. The joint distribution of the data and parameters, or the likelihood times the prior. The poster distribution is proportional to the joint distribution, or everything we have right here. In other words, what we have already written for this particular model is proportional to the posterior distribution of μ and σ^2 , given all of the data. The only thing missing in this expression right here is just some constant number that causes the expression to integrate to 1. If we can recognize this expression as being proportional to a common distribution, then our work is done, and we know what our posterior distribution is. This was the case for all models in the previous course. However, if we do not use conjugate priors or if the models are more complicated, then the posterior distribution will not have a standard form that we can recognize. We're going to explore a couple of examples of this issue in the next segment.

22.4 Non-conjugate models

$$\begin{aligned}
 & n=10 \\
 & y_i | \mu \stackrel{iid}{\sim} N(\mu, 1) \\
 & \mu \sim f(0, 1, 1) \\
 & P(\mu | y_1, \dots, y_n) \propto \prod_{i=1}^{10} \left[\frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(y_i - \mu)^2\right) \right] \frac{1}{\pi(1+\mu^2)} \\
 & \propto \exp\left[-\frac{1}{2} \sum_{i=1}^{10} (y_i - \mu)^2\right] \frac{1}{1+\mu^2} \\
 & \propto \exp\left[-\frac{1}{2} \left(\frac{\sum y_i}{n} - 2\mu + \sum_{i=1}^n y_i + n\mu^2 \right)\right] \frac{1}{1+\mu^2} \\
 & \propto \frac{\exp\left[n(\bar{y}\mu - \mu^2/2)\right]}{1+\mu^2}
 \end{aligned}$$

Figure 22.6: Non-conjugate models

We'll first look at an example of a one parameter model that is not conjugate.

22.4.0.1 Company Personnel

Suppose we have values that represent the percentage change in total personnel from last year to this year for, we'll say, ten companies. These

companies come from a particular industry. We're going to assume for now, that these are independent measurements from a normal distribution with a known variance equal to one, but an unknown mean.

So we'll say the percentage change in the total personnel for company I, given the unknown mean μ will be distributed normally with mean μ , and we're just going to use variance 1.

In this case, the unknown mean could represent growth for this particular industry.

It's the average of the growth of all the different companies. The small variance between the companies and percentage growth might be appropriate if the industry is stable.

We know that the **conjugate prior** for μ in this location would be a **normal distribution**.

But suppose we decide that our prior beliefs about μ are better reflected using a **standard t distribution** with **one degree of freedom**. So we could write that as the prior for μ is a t distribution with a location parameter 0. That's where the center of the distribution is. A scale parameter of 1 to make it the **standard t-distribution** similar to a standard normal, and 1 degree of freedom.

This particular prior distribution has heavier tails than the conjugate and normal distribution, which can more easily accommodate the possibility of extreme values for mu. It is centered on zero so, that apriori, there is a 50% chance that the growth is positive and a 50% chance that the growth is negative.

Recall that the posterior distribution of μ is proportional to the likelihood times the prior. Let's write the expression for that in this model. That is the posterior distribution for μ given the data $y_1 \dots y_n$ is going to be proportional to the likelihood.

It is a product from i equals 1 to n, in this case that's 10.

Densities from a normal distribution.

Let's write the density from this particular normal distribution.

Is 1 over the square root of 2 pi.

E to the negative one-half.

Y_i minus the mean squared, this is the normal density for each individual Y_i and we multiplied it for likelihood.

The density for this t prior looks like this.

It's 1 over pi times 1 plus μ squared.

This is the likelihood times the prior.

If we do a little algebra here, first of all, we're doing this up to proportionality.

So, constants being multiplied by this expression are not important.

The square root of 2 pi being multiplied n times, is just a constant number, and π creates a constant number. So we will drop them in our next step.

So this is now proportional too, we're removing this piece and now we're going to use properties of exponents.

The product of exponents is the sum of the exponentiated pieces.

So we have the exponent of negative one-half times the sum from i equals 1 to n, of Y_i minus μ squared.

And then we're dropping the pie over here, so times 1 plus μ squared.

We're going to do a few more steps of algebra here to get a nicer expression for this piece.

But we're going to skip ahead to that.

We've now added these last two expressions.

To arrive at this expression here for the posterior, or what's proportional to the posterior distribution.

This expression right here is almost proportional to a normal distribution except we have this 1 plus μ squared term in the denominator.

We know the posterior distribution up to a constant but we don't recognize its form as a standard distribution.

That we can integrate or simulate from, so we'll have to do something else.

Let's move on to our second example. For a two parameter example, we're going to return to the case where we have a normal likelihood.

And we're now going to estimate μ and σ^2 , because they're both unknown.

Recall that if σ^2 were known, the conjugate prior from μ would be a normal distribution.

And if μ were known, the conjugate prior we could choose for σ^2 would be an inverse gamma.

We saw earlier that if you include σ^2 in the prior for μ , and use the hierarchical model that we presented earlier, that model would be conjugate and have a closed form solution. However, in the more general case that we have right here, the posterior distribution does not appear as a distribution that we can simulate or integrate.

Challenging posterior distributions like these ones and most others that we'll encounter in this course kept Bayesian methods from entering the main stream of statistics for many years. Since only the simplest problems were tractable. However, computational methods invented in the 1950's, and implemented by statisticians decades later, revolutionized the field. We do have the ability to simulate from the posterior distributions in this lesson as well as for many other more complicated models.

23 Monte Carlo estimation

Bayesian Statistics: Techniques and Models

23.1 Monte Carlo Integration

$$\begin{aligned}\theta &\sim \text{Ga}(a, b) \\ a = 2, b = \frac{1}{3} \\ E(\theta) &= \int_0^{\infty} \theta p(\theta) d\theta = \int_0^{\infty} \theta \frac{b^a}{\Gamma(a)} \theta^{a-1} e^{-b\theta} d\theta = \frac{a}{b} \\ \theta_i^* &\quad i=1, \dots, m \\ \bar{\theta}^* &= \frac{1}{m} \sum_{i=1}^m \theta_i^* \\ \text{Var}(\theta) &= \int_0^{\infty} (\theta - E(\theta))^2 p(\theta) d\theta\end{aligned}$$

Figure 23.1: Monte Carlo Integration

Before we learn how to simulate from complicated posterior distributions, let's review some of the basics of Monte Carlo estimation.

Monte Carlo estimation refers to simulating hypothetical draws from a probability distribution in order to calculate important quantities. By “important quantities,” we mean things like the *mean*, the *variance*, or the *probability* of some event or distributional property.

All of these calculations involve integration, which except for the simplest distributions, may range from very difficult to impossible :-).

Suppose we have a random variable θ that follows a [Gamma distribution](#)

$$\theta \sim \text{Gamma}(a, b) \tag{23.1}$$

Let's say $a = 2$ and $b = \frac{1}{3}$, where a is the *shape parameter* and b is the *rate parameter*.

$$a = 2 \quad b = 1/3 \quad (23.2)$$

To calculate the mean of this distribution, we would need to compute the following integral. It is possible to compute this integral, and the answer is $\frac{a}{b}$ (6 in this case).

$$\mathbb{E}[\theta] = \int_0^\infty \theta f(\theta) d\theta = \int_0^\infty \theta \frac{b^a}{\Gamma(a)} \theta^{a-1} e^{-b\theta} d\theta = \frac{a}{b} \quad (23.3)$$

However, we could verify this answer through Monte Carlo estimation.

To do so, we would simulate a large number of draws (call them θ_i^* ($i = 1, \dots, m$)) from this gamma distribution and calculate their sample mean.

Why can we do this?

Recall from the previous course that if we have a random sample from a distribution, the average of those samples converges in probability to the true mean of that distribution by the **Law of Large Numbers**.

Furthermore, by the **Central Limit Theorem** (CLT), this sample mean $\bar{\theta}^* = \frac{1}{m} \sum_{i=1}^m \theta_i^*$ approximately follows a normal distribution with mean $\mathbb{E}[\theta]$ and variance $\text{Var}[\theta]/m$.

The theoretical variance of θ is the following integral:

$$\text{Var}[\theta] = \int_0^\infty (\theta - \mathbb{E}(\theta))^2 f(\theta) d\theta \quad (23.4)$$

Just like we did with the mean, we could approximate this variance with the sample variance

$$\text{Var}[\theta^*] = \frac{1}{m} \sum_{i=1}^m (\theta_i^* - \bar{\theta}^*)^2 \quad (23.5)$$

23.2 Calculating probabilities

$$\begin{aligned}
 h(\theta) &= \int h(\theta) p(\theta) d\theta = \mathbb{E}[h(\theta)] \approx \frac{1}{m} \sum_{i=1}^m h(\theta_i^*) \\
 I_A(\theta) & \\
 \text{example: } h(\theta) &= I_{\theta < 5}(\theta) \\
 \mathbb{E}(h(\theta)) &= \int_0^\infty I_{\theta < 5}(\theta) p(\theta) d\theta \\
 &= \int_0^5 1 \cdot p(\theta) d\theta + \int_5^\infty 0 \cdot p(\theta) d\theta \\
 &= \mathbb{P}_{\theta}[\theta < 5] \\
 &\approx \frac{1}{m} \sum_{i=1}^m I_{\theta < 5}(\theta_i^*)
 \end{aligned}$$

Figure 23.2: Monte Carlo Integration

This method can be used to calculate many different integrals. Say $h(\theta)$ is any function and we want to calculate

$$\int h(\theta) \mathbb{P}r(\theta) d\theta = \mathbb{E}(h(\theta)) \approx \frac{1}{m} \sum_{i=1}^m h(\theta_i^*) \quad (23.6)$$

where $\mathbb{P}r(\theta)$ is the probability density function of θ and $h(\theta)$ is any function of θ .

This integral is precisely what is meant by $\mathbb{E}[h(\theta)]$, so we can conveniently approximate it by taking the sample mean of $h(\theta_i^*)$. That is, we apply the function h to each simulated sample from the distribution, and take the average of all the results.

One extremely useful example of an h function is the indicator $I_A(\theta)$ where A is some logical condition about the value of θ . To demonstrate, suppose $h(\theta) = I_{[\theta < 5]}(\theta)$, which will give a 1 if $\theta < 5$ and return a 0 otherwise.

What is $\mathbb{E}(h(\theta))$?

This is the integral:

$$\begin{aligned}
 \mathbb{E}[h(\theta)] &= \int_0^\infty \mathbb{I}_{[\theta < 5]}(\theta) \mathbb{P}r(\theta) d\theta \\
 &= \int_0^5 1 \cdot \mathbb{P}r(\theta) d\theta + \int_5^\infty 0 \cdot \mathbb{P}r(\theta) d\theta \\
 &= \mathbb{P}r(\theta < 5)
 \end{aligned} \quad (23.7)$$

So what does this mean?

It means we can approximate the probability that $\theta < 5$ by drawing many samples θ_i^* , and approximating this integral with $\frac{1}{m} \sum_{i=1}^m I_{\theta_i^* < 5}(\theta_i^*)$. This expression is simply counting how many of those samples come out to be less than 5, and dividing by the total number of simulated samples.

That's convenient!

Likewise, we can approximate quantiles of a distribution. If we are looking for the value z such that $\Pr(\theta < z) = 0.9$, we simply arrange the samples θ_i^* in ascending order and find the smallest drawn value that is greater than 90% of the others.

23.3 Monte Carlo Error and Marginalization

$$h(\theta) = \int h(\theta) p(\theta) d\theta = E[h(\theta)] \approx \frac{1}{m} \sum_{i=1}^m h(\theta_i^*)$$

example: $h(\theta) = I_{\theta < 5}(\theta)$

$$\begin{aligned} E(h(\theta)) &= \int I_{\theta < 5}(\theta) p(\theta) d\theta \\ &= \int_0^5 p(\theta) d\theta + \int_5^\infty p(\theta) d\theta \\ &= \Pr[0 < \theta < 5] \\ &\approx \frac{1}{m} \sum_{i=1}^m I_{\theta < 5}(\theta_i^*) \end{aligned}$$

Figure 23.3: Monte Carlo Error and Marginalization

How good is an approximation by Monte Carlo sampling?

Again we can turn to the CLT, which tells us that the variance of our estimate is controlled in part by m . For a better estimate, we want larger m .

For example, if we seek $E[\theta]$, then the sample mean $\bar{\theta}^*$ approximately follows a normal distribution with mean $E[\theta]$ and variance $Var[\theta]/m$.

The variance tells us how far our estimate might be from the true value.

One way to approximate $Var[\theta]$ is to replace it with the sample variance.

The standard deviation of our Monte Carlo estimate is the square root of that, or the sample standard deviation divided by \sqrt{m} .

If m is large, it is reasonable to assume that the true value will likely be within about two standard deviations of your Monte Carlo estimate.

23.4 Marginalization

We can also obtain Monte Carlo samples from hierarchical models.

As a simple example, let's consider a binomial random variable where $y | \phi \sim \text{Bin}(10, \phi)$ and further suppose ϕ is random (as if it had a prior) and is distributed beta $\phi \sim \text{Beta}(2, 2)$.

Given any hierarchical model, we can always write the joint distribution of y and ϕ as $\Pr(y, \phi) = \Pr(y | \phi)\Pr(\phi)$ using the chain rule of probability.

To simulate from this joint distribution, repeat these steps for a large number m :

1. Simulate ϕ_i^* from its Beta(2,2) distribution.
2. Given the drawn ϕ_i^* , simulate y_i^* from $\text{Bin}(10, \phi_i^*)$.

This will produce m independent pairs of $(y^*, \phi^*)_i$ drawn from their joint distribution.

One major advantage of Monte Carlo simulation is that marginalizing is easy. Calculating the marginal distribution of y , $\Pr(y) = \int_0^1 \Pr(y, \phi) d\phi$, might be challenging. But if we have draws from the joint distribution, we can just discard the ϕ_i^* draws and use the y_i^* as samples from their marginal distribution.

This is also called the prior predictive distribution introduced in the previous course.

In the next segment, we will demonstrate some of these principles.

Remember, we do not yet know how to sample from the complicated posterior distributions introduced in the previous lesson.

But once we learn that, we will be able to use the principles from this lesson to make approximate inferences from those posterior distributions.

23.5 Computing Examples

Monte Carlo simulation from the most common distributions is very straightforward in R.

Let's start with the example from the previous segment, where $\theta \sim \text{Gamma}(a, b)$ with $a = 2, b = 1/3$. This could represent the posterior distribution of θ if our data came from a Poisson distribution with mean θ and we had used a conjugate gamma prior. Let's start with $m = 100$.

```
set.seed(32) # Initializes the random number generator so we can replicate these results. To get different results, change the seed value.
m = 100
a = 2.0
b = 1.0 / 3.0
```

To simulate m independent samples, use the `rgamma` function.

```
theta <- rgamma(n=m, shape = a, rate=b)
```

We can plot a histogram of the generated data, and compare that to the true density.

```
hist(theta, freq=FALSE)
curve(dgamma(x=x, shape=a, rate=b), col="blue", add=TRUE)
```

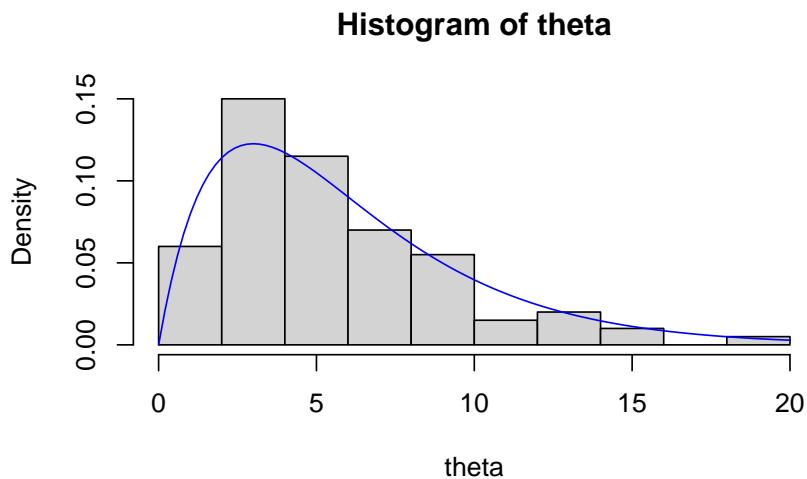


Figure 23.4: Histogram of simulated gamma samples with true density

To find our Monte Carlo approximation to $\mathbb{E}(\theta)$, let's take the average of our sample (and compare it with the truth).

```
sum(theta) / m # sample mean
```

```
[1] 5.514068
```

```
mean(theta) # sample mean
```

```
[1] 5.514068
```

```
a / b # true expected value
```

```
[1] 6
```

Not bad, but we can do better if we increase m to say, 10,000.

```
m = 1e4
theta = rgamma(n=m, shape=a, rate=b)
mean(theta)
```

```
[1] 6.023273
```

How about the variance of θ ?

```
var(theta) # sample variance
```

```
[1] 18.04318
```

```
a / b^2 # true variance of Gamma(a,b)
```

```
[1] 18
```

We can also approximate the probability that $\theta < 5$.

```
ind = theta < 5.0 # set of indicators, TRUE if theta_i < 5
mean(ind)          # automatically converts FALSE/TRUE to 0/1
```

```
[1] 0.497
```

```

pgamma(q=5.0, shape=a, rate=b) # true value of Pr( theta < 5 )

[1] 0.4963317

```

What is the 0.9 quantile (90th percentile) of θ ? We can use the `quantile` function which will order the samples for us and find the appropriate sample quantile.

```

quantile(x=theta, probs=0.9)

90%
11.74338

qgamma(p=0.9, shape=a, rate=b) # true value of 0.9 quantile

[1] 11.66916

```

23.6 Monte Carlo error

We can use the [CLT](#) to approximate how accurate our Monte Carlo estimates are. For example, if we seek $E(\theta)$, then the sample mean $\bar{\theta}^*$ approximately follows a normal distribution with mean $E(\theta)$ and variance $Var(\theta)/m$. We will use the sample standard deviation divided by the square root of m to approximate the Monte Carlo standard deviation.

```

se = sd(theta) / sqrt(m)
2.0 * se # we are reasonably confident that the Monte Carlo estimate is no more than this far from the truth

[1] 0.08495454

```

These numbers give us a reasonable range for the quantity we are estimating with Monte Carlo. The same applies for other Monte Carlo estimates, like the probability that $\theta < 5$.

```

ind = theta < 5.0
se = sd(ind) / sqrt(m)
2.0 * se # we are reasonably confident that the Monte Carlo estimate is no more than this far from the truth

```

```
[1] 0.01000032
```

23.7 Marginalization

Let's also do the second example of simulating a hierarchical model. In our example from the previous segment, we had a binomial random variable where $y | \phi \stackrel{\text{iid}}{\sim} \text{Binomial}(10, \phi)$, and $\phi \sim \text{Beta}(2, 2)$. To simulate from this joint distribution, repeat these steps for a large number m :

1. Simulate ϕ_i from its $\text{Beta}(2, 2)$ distribution.
2. Given the drawn ϕ_i , simulate y_i from $\text{Bin}(10, \phi_i)$.

```

m = 10e4

y = numeric(m) # create the vectors we will fill in with simulations
phi = numeric(m)

for (i in 1:m) {
  phi[i] = rbeta(n=1, shape1=2.0, shape2=2.0)
  y[i] = rbinom(n=1, size=10, prob=phi[i])
}
# which is equivalent to the following 'vectorized' code
phi = rbeta(n=m, shape1=2.0, shape2=2.0)
y = rbinom(n=m, size=10, prob=phi)

```

If we are interested only in the marginal distribution of y , we can just ignore the draws for ϕ and treat the draws of y as a sample from its marginal distribution.

```
mean(y)
```

```
[1] 5.00008
```

```
plot(prop.table(table(y)), ylab="Pr(y)", main="Marginal distribution of y")
```

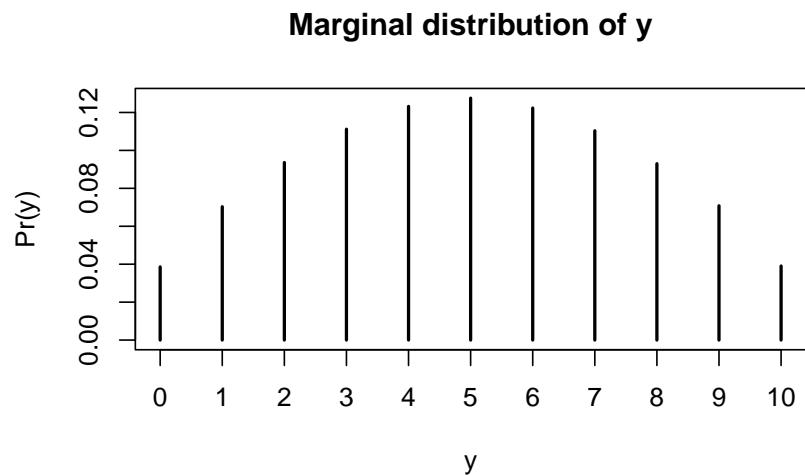


Figure 23.5

24 Markov chains

24.1 Definition

If we have a sequence of random variables X_1, X_2, \dots, X_n where the indices $1, 2, \dots, n$ represent successive points in time, we can use the chain rule of probability to calculate the probability of the entire sequence:

$$\Pr(X_1, X_2, \dots, X_n) = \Pr(X_1) \cdot \Pr(X_2 | X_1) \cdot \Pr(X_3 | X_2, X_1) \cdot \dots \cdot \Pr(X_n | X_{n-1}, X_{n-2}, \dots, X_2, X_1) \quad (24.1)$$

Markov chains simplify this expression by using the Markov assumption. The assumption is that given the entire past history, the probability distribution for the random variable at the next time step only depends on the current variable. Mathematically, the assumption is written like this:

$$\Pr(X_{t+1} | X_t, X_{t-1}, \dots, X_2, X_1) = \Pr(X_{t+1} | X_t) \quad (24.2)$$

for all $t = 2, \dots, n$. Under this assumption, we can write the first expression as

$$\Pr(X_1, X_2, \dots, X_n) = \Pr(X_1) \cdot \Pr(X_2 | X_1) \cdot \Pr(X_3 | X_2) \cdot \Pr(X_4 | X_3) \cdot \dots \cdot \Pr(X_n | X_{n-1}) \quad (24.3)$$

which is much simpler than the original. It consists of an initial distribution for the first variable, $\Pr(X_1)$, and $n-1$ transition probabilities. We usually make one more assumption: that the transition probabilities do not change with time. Hence, the transition from time t to time $t+1$ depends only on the value of X_t .

24.2 Examples of Markov chains

24.2.1 Discrete Markov chain

Suppose you have a secret number (make it an integer) between 1 and 5. We will call it your initial number at step 1. Now for each time step, your secret number will change according to the following rules:

1. Flip a coin.
2.
 - a. If the coin turns up heads, then increase your secret number by one (5 increases to 1).
 - b. If the coin turns up tails, then decrease your secret number by one (1 decreases to 5).
3. Repeat n times, and record the evolving history of your secret number.

Before the experiment, we can think of the sequence of secret numbers as a sequence of random variables, each taking on a value in $\{1, 2, 3, 4, 5\}$. Assume that the coin is fair, so that with each flip, the probability of heads and tails are both 0.5.

Does this game qualify as a true Markov chain? Suppose your secret number is currently 4 and that the history of your secret numbers is $(2, 1, 2, 3)$. What is the probability that on the next step, your secret number will be 5? What about the other four possibilities? Because of the rules of this game, the probability of the next transition will depend only on the fact that your current number is 4. The numbers further back in your history are irrelevant, so this is a Markov chain.

This is an example of a discrete Markov chain, where the possible values of the random variables come from a discrete set. Those possible values (secret numbers in this example) are called states of the chain. The states are usually numbers, as in this example, but they can represent anything. In one common example, the states describe the weather on a particular day, which could be labeled as 1-fair, 2-poor.

24.2.2 Random walk (continuous)

Now let's look at a continuous example of a Markov chain. Say $X_t = 0$ and we have the following transition model:

$$\Pr(X_{t+1} | X_t = x_t) = N(x_t, 1) \quad (24.4)$$

That is, the probability distribution for the next state is Normal with variance 1 and mean equal to the current state. This is often referred to as a “random walk.” Clearly, it is a Markov chain because the transition to the next state X_{t+1} only depends on the current state X_t .

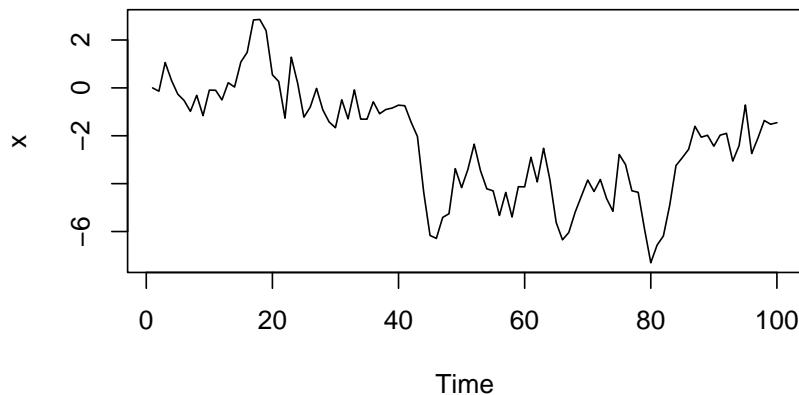
This example is straightforward to code in R:

```
set.seed(34)

n = 100
x = numeric(n)

for (i in 2:n) {
  x[i] = rnorm(1, mean=x[i-1], sd=1.0)
}

plot.ts(x)
```



24.3 Transition matrix

Let’s return to our example of the discrete Markov chain. If we assume that transition probabilities do not change with time, then there are a total of 25 (52) potential transition probabilities. Potential transition probabilities would be from State 1 to State 2, State 1 to State 3, and so forth. These transition probabilities can be arranged into a matrix Q :

$$Q = \begin{pmatrix} 0 & .5 & 0 & 0 & .5 \\ .5 & 0 & .5 & 0 & 0 \\ 0 & .5 & 0 & .5 & 0 \\ 0 & 0 & .5 & 0 & .5 \\ .5 & 0 & 0 & .5 & 0 \end{pmatrix} \quad (24.5)$$

where the transitions from State 1 are in the first row, the transitions from State 2 are in the second row, etc. For example, the probability $\Pr(X_{t+1} = 5 | X_t = 4)$ can be found in the fourth row, fifth column.

The transition matrix is especially useful if we want to find the probabilities associated with multiple steps of the chain. For example, we might want to know $\Pr(X_{t+2} = 3 | X_t = 1)$, the probability of your secret number being 3 two steps from now, given that your number is currently 1. We can calculate this as $\sum_{k=15} \Pr(X_t + 2 = 3 | X_t + 1 = k) \cdot \Pr(X_{t+1} = k | X_t = 1)$, which conveniently is found in the first row and third column of Q_2 .

We can perform this matrix multiplication easily in R:

```
Q = matrix(c(0.0, 0.5, 0.0, 0.0, 0.5,
            0.5, 0.0, 0.5, 0.0, 0.0,
            0.0, 0.5, 0.0, 0.5, 0.0,
            0.0, 0.0, 0.5, 0.0, 0.5,
            0.5, 0.0, 0.0, 0.5, 0.0),
           nrow=5, byrow=TRUE)

Q %*% Q # Matrix multiplication in R. This is Q^2.

[,1] [,2] [,3] [,4] [,5]
[1,] 0.50 0.00 0.25 0.25 0.00
[2,] 0.00 0.50 0.00 0.25 0.25
[3,] 0.25 0.00 0.50 0.00 0.25
[4,] 0.25 0.25 0.00 0.50 0.00
[5,] 0.00 0.25 0.25 0.00 0.50

(Q %*% Q)[1,3]
```

```
[1] 0.25
```

Therefore, if your secret number is currently 1, the probability that the number will be 3 two steps from now is .25.

24.4 Stationary distribution

Suppose we want to know the probability distribution of the your secret number in the distant future, say $\Pr(X_{t+h} | X_t)$ where h is a large number. Let's calculate this for a few different values of h.

```
Q5 = Q %*% Q %*% Q %*% Q %*% Q # h=5 steps in the future
round(Q5, 3)
```

```
[,1] [,2] [,3] [,4] [,5]
[1,] 0.062 0.312 0.156 0.156 0.312
[2,] 0.312 0.062 0.312 0.156 0.156
[3,] 0.156 0.312 0.062 0.312 0.156
[4,] 0.156 0.156 0.312 0.062 0.312
[5,] 0.312 0.156 0.156 0.312 0.062
```

```
Q10 = Q %*% Q # h=10 steps in the future
round(Q10, 3)
```

```
[,1] [,2] [,3] [,4] [,5]
[1,] 0.248 0.161 0.215 0.215 0.161
[2,] 0.161 0.248 0.161 0.215 0.215
[3,] 0.215 0.161 0.248 0.161 0.215
[4,] 0.215 0.215 0.161 0.248 0.161
[5,] 0.161 0.215 0.215 0.161 0.248
```

```
Q30 = Q
for (i in 2:30) {
  Q30 = Q30 %*% Q
}
round(Q30, 3) # h=30 steps in the future
```

```
[,1] [,2] [,3] [,4] [,5]
[1,] 0.201 0.199 0.200 0.200 0.199
```

```
[2,] 0.199 0.201 0.199 0.200 0.200
[3,] 0.200 0.199 0.201 0.199 0.200
[4,] 0.200 0.200 0.199 0.201 0.199
[5,] 0.199 0.200 0.200 0.199 0.201
```

Notice that as the future horizon gets more distant, the transition distributions appear to converge. The state you are currently in becomes less important in determining the more distant future. If we let h get really large, and take it to the limit, all the rows of the long-range transition matrix will become equal to $(.2, .2, .2, .2, .2)$. That is, if you run the Markov chain for a very long time, the probability that you will end up in any particular state is $1/5 = .2$ for each of the five states. These long-range probabilities are equal to what is called the stationary distribution of the Markov chain.

The stationary distribution of a chain is the initial state distribution for which performing a transition will not change the probability of ending up in any given state. That is,

```
c(0.2, 0.2, 0.2, 0.2, 0.2) %*% Q
```

```
[,1] [,2] [,3] [,4] [,5]
[1,] 0.2 0.2 0.2 0.2 0.2
```

One consequence of this property is that once a chain reaches its stationary distribution, the stationary distribution will remain the distribution of the states thereafter.

We can also demonstrate the stationary distribution by simulating a long chain from this example.

```
n = 5000
x = numeric(n)
x[1] = 1 # fix the state as 1 for time 1
for (i in 2:n) {
  x[i] = sample.int(5, size=1, prob=Q[x[i-1],]) # draw the next state from the integers 1 to 5 with probability
}
```

Now that we have simulated the chain, let's look at the distribution of visits to the five states.

Table 24.1

```
table(x) / n

x
1      2      3      4      5
0.1996 0.2020 0.1980 0.1994 0.2010
```

The overall distribution of the visits to the states is approximately equal to the stationary distribution.

As we have just seen, if you simulate a Markov chain for many iterations, the samples can be used as a Monte Carlo sample from the stationary distribution. This is exactly how we are going to use Markov chains for Bayesian inference. In order to simulate from a complicated posterior distribution, we will set up and run a Markov chain whose stationary distribution is the posterior distribution.

It is important to note that the stationary distribution doesn't always exist for any given Markov chain. The Markov chain must have certain properties, which we won't discuss here. However, the Markov chain algorithms we'll use in future lessons for Monte Carlo estimation are guaranteed to produce stationary distributions.

24.4.1 Continuous example

The continuous random walk example we gave earlier does not have a stationary distribution. However, we can modify it so that it does have a stationary distribution.

Let the transition distribution be $\Pr(X_{t+1} \mid X_t = x_t) = N(\phi x_t, 1)$ where $-1 < \phi < 1$. That is, the probability distribution for the next state is Normal with variance 1 and mean equal to ϕ times the current state. As long as ϕ is between -1 and 1 , then the stationary distribution will exist for this model.

Let's simulate this chain for $\phi = -0.6$.

```

set.seed(38)

n = 1500
x = numeric(n)
phi = -0.6

for (i in 2:n) {
  x[i] = rnorm(1, mean=phi*x[i-1], sd=1.0)
}

plot.ts(x)

```

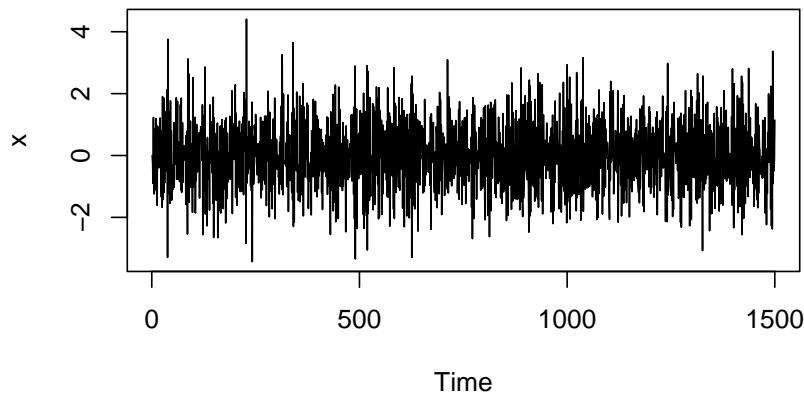


Figure 24.1: Simulated AR(1) process with phi=-0.6

The theoretical stationary distribution for this chain is normal with mean 0 and variance $1/(1 - \phi^2)$, which in our example approximately equals 1.562. Let's look at a histogram of our chain and compare that with the theoretical stationary distribution.

$$\text{Var}_{\text{stationary}} = \frac{1}{1 - \phi^2} \quad (24.6)$$

```

hist(x, freq=FALSE)
curve(dnorm(x, mean=0.0, sd=sqrt(1.0/(1.0-phi^2))), col="red", add=TRUE)
legend("topright", legend="theoretical stationary\ndistribution", col="red", lty=1, bty="n")

```

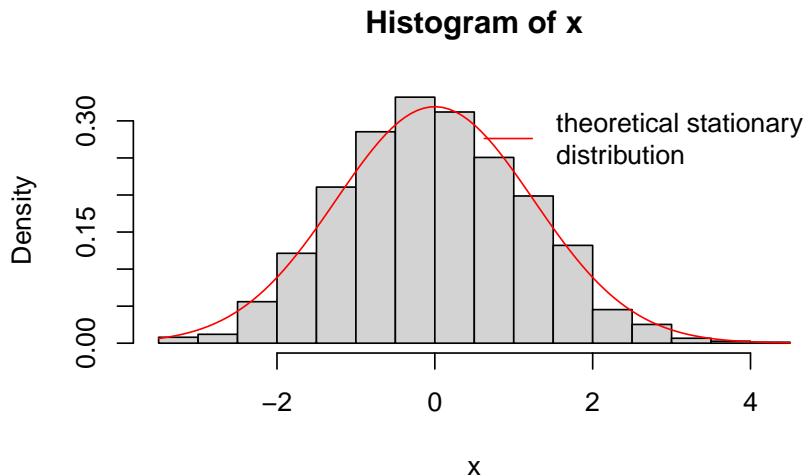


Figure 24.2: Histogram of simulated AR(1) process with theoretical stationary distribution

It appears that the chain has reached the stationary distribution. Therefore, we could treat this simulation from the chain like a Monte Carlo sample from the stationary distribution, a normal with mean 0 and variance 1.562.

Because most posterior distributions we will look at are continuous, our Monte Carlo simulations with Markov chains will be similar to this example.

25 Metropolis-Hastings

Bayesian Statistics: Techniques and Models

25.1 The Metropolis-Hastings Algorithm

Metropolis-Hastings (M-H) is an algorithm that allows us to sample from a generic probability distribution (which we will call the target distribution), even if we do not know the normalizing constant. To do this, we construct and sample from a Markov chain whose stationary distribution is the target distribution. It consists of picking an arbitrary starting value and iteratively accepting or rejecting candidate samples drawn from another distribution, one that is easy to sample.

! Why use M-H or MCMC?

We will use M-H or other MCMC methods if there is no easy way to simulate independent draws from the target distribution. This can be due to non-conjugate priors, challenges in evaluating the normalizing constant or multiple explanatory variables.

25.2 The M-H Algorithm

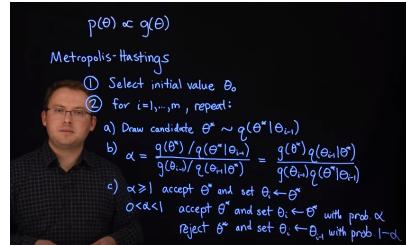


Figure 25.1: The Metropolis-Hastings Algorithm

Let's say we wish to produce samples from a *target distribution* $Pr(\theta) \propto g(\theta)$, where we don't know the normalizing constant (since $\int g(\theta) d\theta$ is hard or impossible to compute), so we only have $g(\theta)$, the *unnormalized joint probability* to work with. The Metropolis-Hastings algorithm proceeds as follows.

1. Select an **initial value** θ_0 .
2. For $i = 1, \dots, m$ repeat the following steps:
 - a. **Draw a candidate sample** θ^* from a **proposal distribution** $q(\theta^* | \theta_{i-1})$.
 - b. **Compute the ratio** $\alpha = \frac{g(\theta^*) / q(\theta^* | \theta_{i-1})}{g(\theta_{i-1}) / q(\theta_{i-1} | \theta^*)} = \frac{g(\theta^*) q(\theta_{i-1} | \theta^*)}{g(\theta_{i-1}) q(\theta^* | \theta_{i-1})}$
 - c. • If $\alpha \geq 1$, then **accept** θ^* and set $\theta_i = \theta^*$.
 • If $0 < \alpha < 1$:
 - **accept** θ^* and set $\theta_i = \theta^*$ with probability α ,
 - **reject** θ^* and set $\theta_i = \theta_{i-1}$ with probability $1 - \alpha$.

! Correction to the proposal distribution

Steps 2.b and 2.c act as a **correction** since the *proposal distribution* is not the *target distribution*. At each step in the chain, we draw a random candidate value of the parameter and decide whether to "move" the chain there or remain where we are. If the proposed move to the candidate is "advantageous," ($\alpha \geq 1$) we "move" there and if it is not "advantageous," we still might move there, but only with probability α . Since our decision to "move" to the candidate only depends on where the chain currently is, this is a *Markov chain*.

proposal distribution q

correction

25.3 Proposal distribution q

One careful choice we must make is the candidate generating distribution $q(\theta^* | \theta_{i-1})$. It may or may not depend on the previous iteration's value of θ .

! Independent Metropolis-Hastings

The simpler case is when the proposal distribution q does not depend on the previous value. We then write it as $q(\theta^*)$. This arises if it is always the same distribution. We call this case **independent Metropolis-Hastings**. If we use independent M-H, $q(\theta)$ **should be as similar as possible to $\mathbb{P}r(\theta)$** .

! Random-Walk Metropolis-Hastings

In the more general case, the proposal distribution takes the form $q(\theta^* | \theta_{i-1})$ with dependence on the previous iteration, is **Random-Walk Metropolis-Hastings**. Here, the proposal distribution is centered on θ_{i-1} .

For instance, it might be a Normal distribution with mean θ_{i-1} . Because the Normal distribution is *symmetric*, this example comes with another advantage: $q(\theta^* | \theta_{i-1}) = q(\theta_{i-1} | \theta^*)$ causing it to cancel out when we calculate α .

Thus, in **Random-Walk M-H** where the candidate is drawn from a Normal with mean θ_{i-1} and constant variance, the acceptance ratio is simply $\alpha = g(\theta^*)/g(\theta_{i-1})$.

25.4 Acceptance rate α

Clearly, not all candidate draws are accepted, so our Markov chain sometimes “stays” where it is, possibly for many iterations. How often you want the chain to accept candidates depends on the type of algorithm you use. If you approximate $\mathbb{P}r(\theta)$ with $q(\theta^*)$ and always draw candidates from that, accepting candidates often is good; it means $q(\theta^*)$ is approximating $\mathbb{P}r(\theta)$ well. However, you still may want q to have a larger variance than p and see some rejection of candidates as an assurance that q is covering the space well.

As we will see in coming examples, a high acceptance rate for the Random-Walk Metropolis-Hastings sampler is not a good thing. If the random walk is taking too small of steps, it will accept often but will take a very long time to fully explore the posterior. If the random walk is taking too large of steps, many of its proposals will have a low probability and the acceptance rate will be low, wasting many draws. Ideally, a random walk sampler should accept somewhere between 23% and 50% of the candidates proposed.

In the next segment, we will see a demonstration of this algorithm used in a discrete case, where we can show mathematically that the Markov chain converges to the target distribution. In the following segment, we will demonstrate coding a Random-Walk Metropolis-Hastings algorithm in R to solve one of the problems from the end of Lesson 2.

25.5 Demonstration of a Discrete case

$$\begin{aligned}
 \Theta &= \{\text{fair, loaded}\} \\
 \text{Prior } P(\theta = \text{loaded}) &= 0.6 \\
 \text{Likelihood } f(x|\theta) &= \binom{x}{2} \left(\frac{1}{2}\right)^x \left(\frac{1}{2}\right)^{2-x} \\
 \text{Posterior } f(\theta|x=2) &= \frac{f(x=2|\theta) f(\theta)}{f(x=2)} \\
 &= \frac{\left(\frac{1}{2}\right)^2 \left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2 \left(\frac{1}{2}\right)^2}{\left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2} \\
 &= \frac{0.0125}{0.0125 + 0.00794} \\
 &\approx 0.612 \underbrace{I_{\{\theta=\text{fair}\}}}_{\text{fair}} + 0.388 \underbrace{I_{\{\theta=\text{loaded}\}}}_{\text{loaded}} \\
 P(\theta = \text{loaded} | x=2) &\approx 0.388
 \end{aligned}$$

Figure 25.2: MCMC Coin Flip Example

The following segment is by Herbert Lee, a professor of statistics and applied mathematics at the University of California, Santa Cruz.

The following is a demonstration of using Markov chain Monte Carlo, used to estimate posterior probabilities in a simplified case, where we can actually work out the correct answer in closed form. We demonstrate that the Metropolis-Hastings algorithm is indeed working, and giving us the right answer.

If you recall from the previous course, the example where your brother or maybe your sister, has a loaded coin that you know will come up heads 70% of the time. But they come to you with some coin, you're not sure if

it's the loaded coin or a fair coin, and they want to make a bet with you. And you have to figure out which coin this is.

Suppose you have a prior probability that it's a 60% probability, that they'll bring a loaded coin to you. They let you flip it five times, and you get two heads and three tails.

And then you need to figure out, what's your posterior probability that this is a loaded coin.

Our unknown parameter θ , can either take the values *fair* or *loaded*.

$$\theta = \{\text{fair, loaded}\} \quad (25.1)$$

Our **prior** for θ is the probability of theta equals loaded, is 0.6.

$$\Pr(\theta = \text{loaded}) = 0.6 \quad (\text{prior}) \quad (25.2)$$

Our likelihood will follow a Binomial distribution, depending upon the value of θ .

$$f(x | \theta) = \binom{5}{x} \frac{1}{2}^5 \mathbb{I}_{\theta=\text{fair}} + \binom{5}{x} (.7)^x (.3)^{5-x} \mathbb{I}_{\theta=\text{loaded}} \quad (\text{likelihood}) \quad (25.3)$$

Our posterior then, we can look at posterior for theta, given that we saw $x = 2$ equals two heads, posterior is the likelihood times the prior, divided by a normalizing constant.

$$\begin{aligned} f(\theta | X = 2) &= \frac{\frac{1}{2}^5 (0.4) \mathbb{I}_{(\theta=\text{fair})} + (.7)^2 (.3)^3 (.6) \mathbb{I}_{(\theta=\text{loaded})}}{\frac{1}{2}^5 (0.4) + (.7)^2 (.3)^3 (.6)} \\ &= \frac{0.0125 \mathbb{I}_{(\theta=\text{fair})} + 0.00794 \mathbb{I}_{(\theta=\text{loaded})}}{0.0125 + 0.00794} \quad (25.4) \\ &= 0.612 \mathbb{I}_{(\theta=\text{fair})} + 0.388 \mathbb{I}_{(\theta=\text{loaded})} \quad (\text{posterior}) \end{aligned}$$

In this case, we can work out the binomial and our prior. And we see that we get these expressions at the end. We get posterior probability of θ is loaded given that we saw two heads, to be 0.388.

$$\therefore \Pr(\theta = \text{loaded} \mid X = 2) = 0.388 \quad (\text{posterior conditional probability}) \quad (25.5)$$

This is all review from the previous course so far.

But suppose we had a more complicated problem, where we couldn't work this all out in closed form? We'll know the likelihood and the prior, but we may not be able to get this normalizing constant. Can we instead do this by simulation? And indeed, yes we can.

We can do this with Markov chain Monte Carlo. In particular, using the Metropolis-Hastings algorithm. What we'll do is, we'll set up a Markov chain whose **equilibrium distribution** has this **posterior distribution**. So we'll consider a Markov chain with two states, theta equals fair and theta equals loaded. And we'll allow the chain to move between those two states, with certain transition probabilities. We set this up using this using the Metropolis-Hastings algorithm.

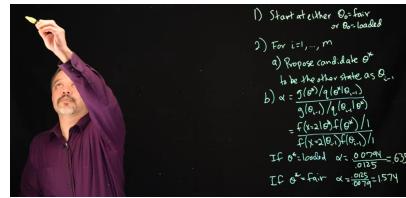


Figure 25.3: The Metropolis-Hastings Algorithm

So under the Metropolis-Hastings algorithm, step one is we start at an arbitrary location. And in this case, we can

1. start at either $\theta \neq \text{fair}$, or $\theta \neq \text{loaded}$.

It does not really matter where we start, we'll be moving back and forth and we're going to look at the long-term running average, the long-term simulations.

So the key is we'll be simulating.

2. Run m simulations and in each iteration, we'll propose a candidate and either accept it or reject it.

- a. So the first part is we're proposing a new candidate. We'll call this candidate θ^* , and we're going to propose it be the other state compared to where we are now. Where we are now is θ_{i-1} , and so we'll propose to move to θ^* .

- If our *current state* is **fair**, we'll propose $\theta^* = \text{loaded}$.
- If our *current state* is **loaded**, we'll propose $\theta^* = \text{fair}$.

The general form for α is:

what's our acceptance probability alpha?

$$\begin{aligned}\alpha &= \frac{g(\theta^*)/q(\theta^* | \theta_{i-1})}{g(\theta_{i-1})/q(\theta_{i-1} | \theta^*)} \\ &= \frac{f(x=2 | \theta^*) f(\theta^*)/1}{f(x=2 | \theta_{i-1}) f(\theta_{i-1})/1} \quad (\text{sub. g,q})\end{aligned}\quad (25.6)$$

In this case,

- $g()$ is our un-normalized likelihood times prior
- $q()$, the *proposal distribution*, is, in this case, since we always accept the opposite state deterministically i.e. $\theta^* = -\theta_{i-1}$ with $P = 1$
- If $\theta^* = \text{loaded} \implies \alpha = \frac{0.00794}{0.0125} = 0.635$
- If $\theta^* = \text{fair} \implies \alpha = \frac{0.0125}{0.00794} = 1.574$

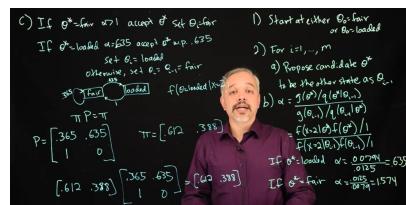


Figure 25.4: The Metropolis-Hastings Algorithm

Given these probabilities, we then can do the acceptance or rejection step.

$$\left\{ \begin{array}{ll} \text{accept } \theta^* \text{ and set } \theta_i = \text{fair} & \text{If } \theta^* = \text{fair}, \alpha > 1 \\ \left\{ \begin{array}{ll} \text{accept } \theta^* \text{ and set } \theta_i = \text{loaded} & \text{With probability 0.635} \\ \text{reject } \theta^* \text{ and set } \theta_i = \text{fair} & \text{Otherwise} \end{array} \right. & \text{If } \theta^* = \text{loaded}, \alpha = .635 \end{array} \right.$$

If the $\theta^* = \text{loaded} \implies \alpha = 0.635$. So we accept theta star with probability 0.635. And if we accept it. Set $\theta_i = \text{loaded}$ Otherwise, set $\theta_i = \theta_{i-1}$, if we do not accept, it stays in that same old fair state.

We can draw this out as a Markov chain with two states, Fair and ‘loaded’. If it’s in the ‘loaded’ state, it will move with probability one to the fair state. If it’s in the fair state, it will move with a probability of 0.635 to the ‘loaded’ state. And with a probability of 0.365 it will stay in the fair state.



Figure 25.5: state diagram

And so here’s a little diagram for this Markov chain with two states. In which case it will move back and forth with certain probabilities.

Thus, if we wanted to find our **posterior probability**, $f(\theta = \text{loaded} | x = 2)$. We can simulate from this Markov chain using these transition probabilities. And observe the fraction of time that it spends in the state theta equals ‘loaded’. And this gives us a good estimate of the posterior probability that it’s the ‘loaded’ coin. In this particular case, we can also show that this gives us the theoretical right answer.

If you’ve seen a little bit of the theory of Markov chains. We can say that a Markov chain with transition probability capital P , has stationary distribution Π .

$$\pi P = \pi \quad (\text{def. stationary distribution}) \quad (25.7)$$

Here we have a transition probability matrix P , where we can think about ‘fair’ and ‘loaded’. Moving from the ‘fair’ state, remaining in the ‘fair’ state happens with a probability of 0.365 and it moves from ‘fair’ to ‘loaded’, with a probability of 0.635. If it’s in the ‘loaded’ state, we’ll move to the ‘fair’ state with probability one, and it will stay in the ‘loaded’ state with probability 0.

$$P = \begin{bmatrix} 0.365 & 0.635 \\ 1 & 0 \end{bmatrix}$$

In this case, we want our stationary distribution to be the posterior probabilities.

$$\Pi = \begin{bmatrix} 0.612 & 0.388 \end{bmatrix}$$

Which you can recall are 0.612 of being ‘fair’ and 0.388 of being ‘loaded’. And so indeed, if you do just the minimal amount of matrix algebra, you can see that 0.612, 0.388 Multiplied by this matrix, 0.365, 0.635, 1, 0, does indeed give you 0.612 and 0.388, at least to within rounding error.

$$\begin{aligned} \Pi P &= \begin{bmatrix} 0.612 & 0.388 \end{bmatrix} \begin{bmatrix} 0.365 & 0.635 \\ 1 & 0 \end{bmatrix} \\ &= \begin{bmatrix} 0.612 & 0.388 \end{bmatrix} \\ &= \Pi \end{aligned} \tag{25.8}$$

Thus in this case we can see, that we do get the correct stationary distribution for the Markov chain using the Metropolis–Hastings algorithm. And that when we simulate it, we do get correct estimates then of the posterior probabilities.

This is a nice simple example where we can work out the posterior probabilities in closed form. We don’t need to run Markov chain Monte Carlo. But this method is very powerful because all we need is to be able to evaluate the likelihood and the prior, we don’t need to evaluate the full posterior and get that normalizing constant. And so this applies to a much broader range of more complicated problems. Where we can use Markov chain Monte Carlo to simulate, to be able to get these probabilities. We’ll make good use of this in the rest of this course.

25.6 Random walk with Normal likelihood, t prior

Recall the model from the last segment of Lesson 2 where the data are the percent change in total personnel from last year to this year for n=10 companies. We used a normal likelihood with known variance and t distribution for the prior on the unknown mean. Suppose the values are $y = (1.2, 1.4, -0.5, 0.3, 0.9, 2.3, 1.0, 0.1, 1.3, 1.9)$. Because this model is not conjugate, the posterior distribution is not in a standard form that we can easily sample. To obtain posterior samples, we will set up a Markov chain whose stationary distribution is this posterior distribution.

Recall that the posterior distribution is

$$\Pr(\mu \mid y_1, \dots, y_n) \propto \frac{\exp[n(\bar{y}\mu - \mu^2/2)]}{1 + \mu^2}$$

The posterior distribution on the left is our target distribution and the expression on the right is our $g(\mu)$.

The first thing we can do in R is write a function to evaluate $g(\mu)$. Because posterior distributions include likelihoods (the product of many numbers that are potentially small), $g(\mu)$ might evaluate to such a small number that to the computer, it is effectively zero. This will cause a problem when we evaluate the acceptance ratio α . To avoid this problem, we can work on the log scale, which will be more numerically stable. Thus, we will write a function to evaluate

$$\log(g(\mu)) = n(\bar{y}\mu - \mu^2/2) - \log(1 + \mu^2)$$

This function will require three arguments, μ , \bar{y} , and n .

```
lg = function(mu, n, ybar) {
  mu2 = mu^2
  n * (ybar * mu - mu2 / 2.0) - log(1 + mu2)
}
```

Next, let's write a function to execute the **Random-Walk Metropolis-Hastings** sampler with *Normal* proposals.

```
mh = function(n, ybar, n_iter, mu_init, cand_sd) {
  ## Random-Walk Metropolis-Hastings algorithm

  ## Step 1, initialize
  mu_out = numeric(n_iter)
  accpt = 0
  mu_now = mu_init
  lg_now = lg(mu=mu_now, n=n, ybar=ybar)

  ## Step 2, iterate
  for (i in 1:n_iter) {
    ## step 2a
```

```

mu_cand = rnorm(n=1, mean=mu_now, sd=cand_sd) # draw a candidate

## Step 2b
lg_cand = lg(mu=mu_cand, n=n, ybar=ybar) # evaluate log of g with the candidate
lalpha = lg_cand - lg_now # log of acceptance ratio
alpha = exp(lalpha)

## step 2c
u = runif(1) # draw a uniform variable which will be less than alpha with probability min(1, alpha)
if (u < alpha) { # then accept the candidate
  mu_now = mu_cand
  accpt = accpt + 1 # to keep track of acceptance
  lg_now = lg_cand
}

## collect results
mu_out[i] = mu_now # save this iteration's value of mu
}

## return a list of output
list(mu=mu_out, accpt=accpt/n_iter)
}

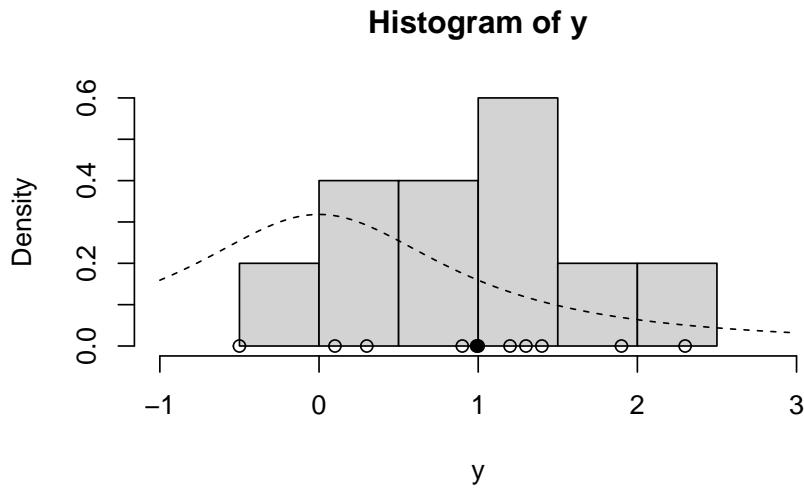
```

Now, let's set up the problem.

```

y = c(1.2, 1.4, -0.5, 0.3, 0.9, 2.3, 1.0, 0.1, 1.3, 1.9)
ybar = mean(y)
n = length(y)
hist(y, freq=FALSE, xlim=c(-1.0, 3.0)) # histogram of the data
curve(dt(x=x, df=1), lty=2, add=TRUE) # prior for mu
points(y, rep(0,n), pch=1) # individual data points
points(ybar, 0, pch=19) # sample mean

```

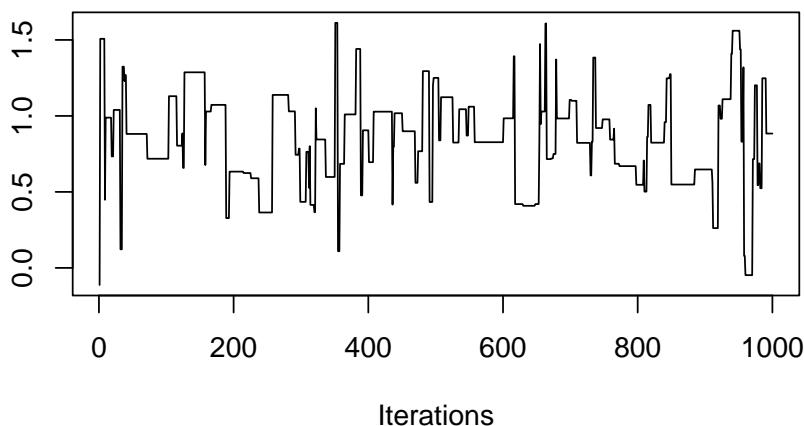


Finally, we're ready to run the sampler! Let's use $m = 1000$ iterations and proposal standard deviation (which controls the proposal step size) 3.0, and initial value at the prior median 0.

```
set.seed(43) # set the random seed for reproducibility
post = mh(n=n, ybar=ybar, n_iter=1e3, mu_init=0.0, cand_sd=3.0)
str(post)
```

```
List of 2
$ mu    : num [1:1000] -0.113 1.507 1.507 1.507 1.507 ...
$ accpt: num 0.122
```

```
library("coda")
traceplot(as.mcmc(post$mu))
```



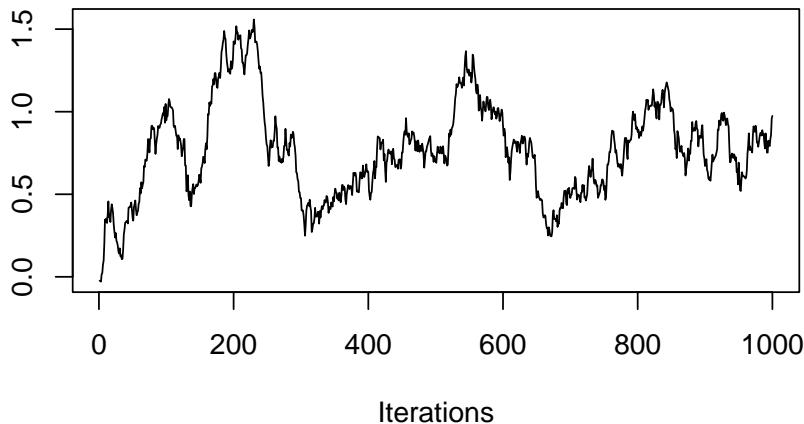
This last plot is called a trace plot. It shows the history of the chain and provides basic feedback about whether the chain has reached its stationary distribution.

It appears our proposal step size was too large (acceptance rate below 23%). Let's try another.

```
post = mh(n=n, ybar=ybar, n_iter=1e3, mu_init=0.0, cand_sd=0.05)
post$accpt
```

```
[1] 0.946
```

```
traceplot(as.mcmc(post$mu))
```

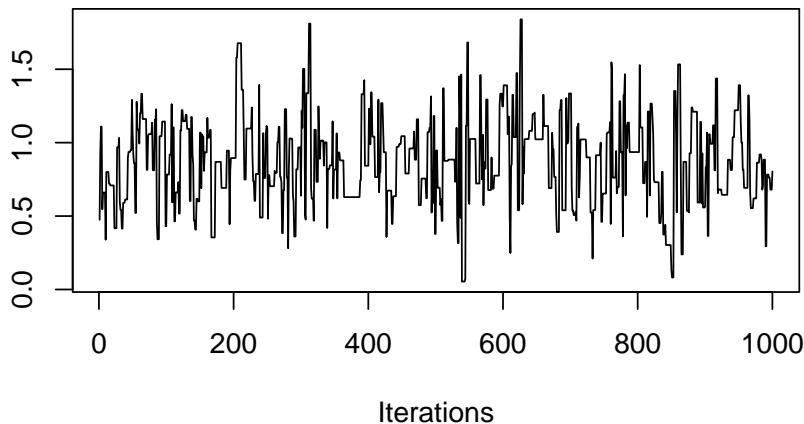


Oops, the acceptance rate is too high (above 50%). Let's try something in between.

```
post = mh(n=n, ybar=ybar, n_iter=1e3, mu_init=0.0, cand_sd=0.9)
post$accpt
```

```
[1] 0.38
```

```
traceplot(as.mcmc(post$mu))
```

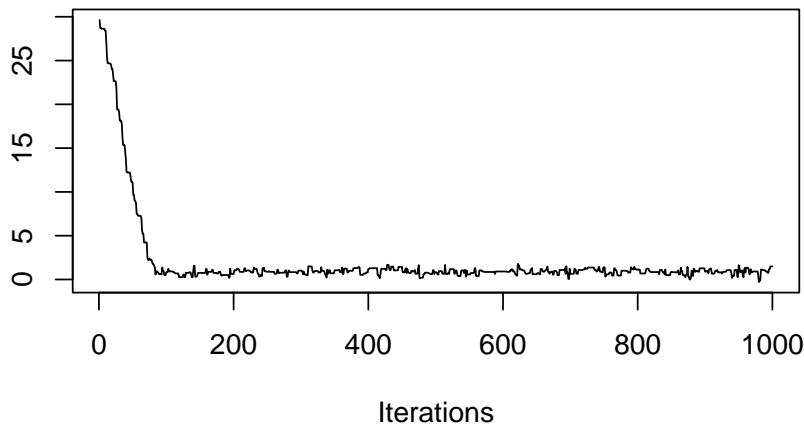


Which looks good. Just for fun, let's see what happens if we initialize the chain at some far-off value.

```
post = mh(n=n, ybar=ybar, n_iter=1e3, mu_init=30.0, cand_sd=0.9)
post$accpt
```

```
[1] 0.387
```

```
traceplot(as.mcmc(post$mu))
```



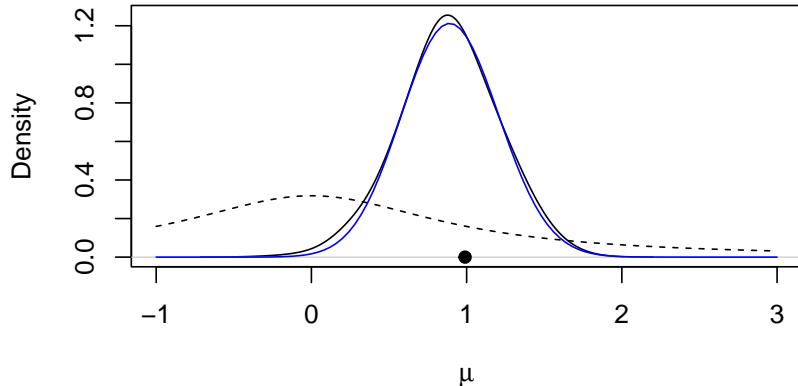
It took awhile to find the stationary distribution, but it looks like we succeeded! If we discard the first 100 or so values, it appears like the rest of the samples come from the stationary distribution, our posterior distribution! Let's plot the posterior density against the prior to see how the data updated our belief about μ .

```

post$mu_keep = post$mu[-c(1:100)] # discard the first 200 samples
plot(density(post$mu_keep, adjust=2.0), main="", xlim=c(-1.0, 3.0), xlab=expression(mu)) # plot density estimate
curve(dt(x=x, df=1), lty=2, add=TRUE) # prior for mu
points(ybar, 0, pch=19) # sample mean

curve(0.017*exp(lg(mu=x, n=n, ybar=ybar)), from=-1.0, to=3.0, add=TRUE, col="blue") # approximation to the true posterior

```



These results are encouraging, but they are preliminary. We still need to investigate more formally whether our Markov chain has converged to the stationary distribution. We will explore this in a future lesson.

Obtaining posterior samples using the Metropolis-Hastings algorithm can be time-consuming and require some fine-tuning, as we've just seen. The good news is that we can rely on software to do most of the work for us. In the next couple of videos, we'll introduce a program that will make posterior sampling easy.

26 Introduction to JAGS

26.1 Setup

26.1.1 Introduction to JAGS

There are several software packages available that will handle the details of MCMC for us. See the supplementary material for a brief overview of options.

The package we will use in this course is **JAGS** (Just Another Gibbs Sampler) by Martyn Plummer. The program is free, and runs on Mac OS, Windows, and Linux. Better yet, the program can be run using R with the **rjags** and **R2jags** packages.

In JAGS, we can specify models and run MCMC samplers in just a few lines of code; JAGS does the rest for us, so we can focus more on the statistical modeling aspect and less on the implementation. It makes powerful Bayesian machinery available to us as we can fit a wide variety of statistical models with relative ease.

26.1.2 Installation and setup

The starting place for JAGS users is mcmc-jags.sourceforge.net. At this site, you can find news about the features of the latest release of JAGS, links to program documentation, as well as instructions for installation.

The documentation is particularly important. It is available under the [files page](#) link in the Manuals folder.

Also under the [files page](#), you will find the JAGS folder where you can download and install the latest version of JAGS. Select the version and operating system, and follow the instructions for download and installation.

Once JAGS is installed, we can immediately run it from R using the **rjags** package. The next segment will show how this is done.

26.2 Modeling in JAGS

There are four steps to implementing a model in JAGS through R:

1. Specify the model.
2. Set up the model.
3. Run the MCMC sampler.
4. Post-processing.

We will demonstrate these steps with our running example with the data are the percent change in total personnel from last year to this year for $n = 10$ companies. We used a normal likelihood with known variance and t distribution for the prior on the unknown mean.

26.2.1 1. Specify the model

In this step, we give JAGS the hierarchical structure of the model, assigning distributions to the data (the likelihood) and parameters (priors). The syntax for this step is very similar to R, but there are some key differences.

```
library("rjags")

mod_string = " model {
  for (i in 1:n) {
    y[i] ~ dnorm(mu, 1.0/sig2)
  }
  mu ~ dt(0.0, 1.0/1.0, 1.0) # location, inverse scale, degrees of freedom
  sig2 = 1.0
}"
```

One of the primary differences between the syntax of JAGS and R is how the distributions are parameterized. Note that the normal distribution uses the mean and precision (instead of variance). When specifying distributions in JAGS, it is always a good idea to check the JAGS user manual [here](#) in the chapter on Distributions.

26.2.2 2. Set up the model

```

set.seed(50)
y = c(1.2, 1.4, -0.5, 0.3, 0.9, 2.3, 1.0, 0.1, 1.3, 1.9)
n = length(y)

data_jags = list(y=y, n=n)
params = c("mu")

inits = function() {
  inits = list("mu"=0.0)
} # optional (and fixed)

mod = jags.model(textConnection(mod_string), data=data_jags, inits=inits)

```

```

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
Observed stochastic nodes: 10
Unobserved stochastic nodes: 1
Total graph size: 15

```

Initializing model

There are multiple ways to specify initial values here. They can be explicitly set, as we did here, or they can be random, i.e., `list("mu"=rnorm(1))`. Also, we can omit the initial values, and JAGS will provide them.

26.2.3 3. Run the MCMC sampler

```

update(mod, 500) # burn-in

mod_sim = coda.samples(model=mod, variable.names=params, n.iter=1000)

```

We will discuss more options to the `coda.samples` function in coming examples.

26.2.4 4. Post-processing

```
summary(mod_sim)
```

```
Iterations = 1501:2500
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000
```

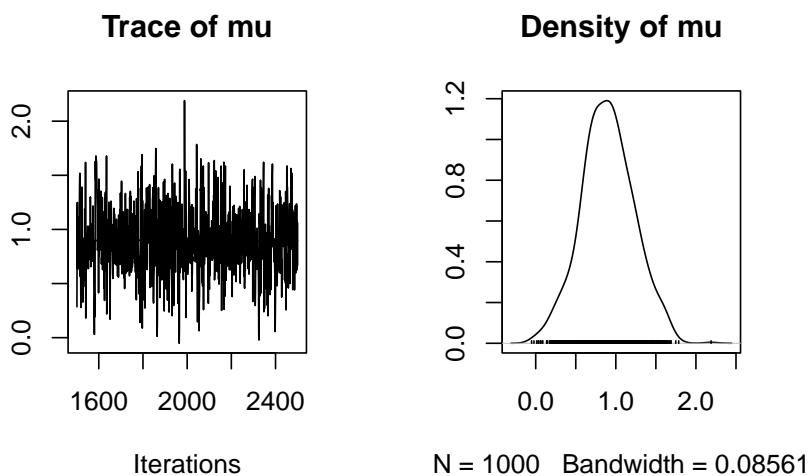
1. Empirical mean and standard deviation for each variable,
plus standard error of the mean:

Mean	SD	Naive SE	Time-series SE
0.89942	0.32918	0.01041	0.01452

2. Quantiles for each variable:

2.5%	25%	50%	75%	97.5%
0.2389	0.6823	0.8963	1.1131	1.5724

```
library("coda")
plot(mod_sim)
```



We will discuss post processing further, including convergence diagnostics, in a coming lesson.

27 Gibbs sampling

Bayesian Statistics: Techniques and Models

Gibbs sampling is a Gibbs sampling is named after the physicist Josiah Willard Gibbs, in reference to an analogy between the sampling algorithm and statistical physics. The algorithm was described in (Geman and Geman 1984) by brothers *Stuart and Donald Geman*, and became popularized in the statistics community for calculating marginal probability distribution, especially the posterior distribution. Gibbs sampling is better suited for sampling from models with many variables by sampling them one at a time from a full conditional distribution.

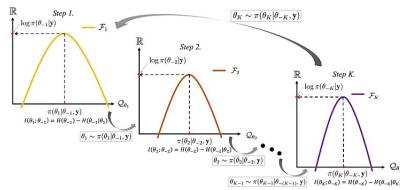


Figure 27.1: Gibbs sampler

💡 How does the Gibbs sampling simplify updating multiple parameters in MCMC?

It divides the process into updating one parameter at a time using (potentially convenient) full conditional distributions. The price we pay for this convenience is

1. the work required to find full conditional distributions and
2. the sampler may require more iterations to fully explore the posterior distribution as the draws tend to be more correlated since they share information in the form of parameters.

It is also possible to run a Gibbs sampler that draws from the “full”

conditional distribution of multiple parameters. We would then cycle through and update blocks of parameters.

27.1 Multiple parameter sampling and full conditional distributions

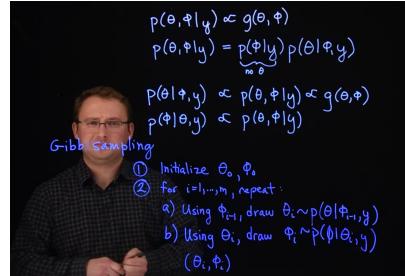


Figure 27.2: Multiple parameter sampling and full conditional distributions

So far, we have demonstrated MCMC for a single parameter.

What if we seek the posterior distribution of multiple parameters, and that posterior distribution does not have a standard form?

One option is to perform Metropolis-Hastings (M-H) by sampling candidates for all parameters at once, and accepting or rejecting all of those candidates together. While this is possible, it can get complicated.

Another (simpler) option is to sample the parameters one at a time.

As a simple example, suppose we have a joint posterior distribution for two parameters θ and ϕ , written $\Pr(\theta, \phi | y) \propto g(\theta, \phi)$. If we knew the value of ϕ , then we would just draw a candidate for θ and use $g(\theta, \phi)$ to compute our Metropolis-Hastings ratio, and possibly accept the candidate. Before moving on to the next iteration, if we don't know ϕ , then we can perform a similar update for it. Draw a candidate for ϕ using some proposal distribution and again use $g(\theta, \phi)$ to compute our Metropolis-Hastings ratio. Here we pretend we know the value of θ by substituting its current iteration from the Markov chain. Once we've drawn for both θ and ϕ , that completes one iteration and we begin the next iteration by drawing a new θ . In other words, we're just going back and forth,

updating the parameters one at a time, plugging the current value of the other parameter into $g(\theta, \phi)$.

This idea of one-at-a-time updates is used in what we call *Gibbs sampling*, which also produces a stationary Markov chain (whose stationary distribution is the posterior). If you recall, this is the namesake of JAGS, “just another Gibbs sampler.”

27.1.1 Full conditional distributions

Before describing the full **Gibbs sampling algorithm**, there’s one more thing we can do. Using the chain rule of probability, we have

$$\Pr(\theta, \phi | y) = \Pr(\theta | \phi, y) \cdot \Pr(\phi | y)$$

Notice that the only difference between $\Pr(\theta, \phi | y)$ and $\Pr(\theta | \phi, y)$ is multiplication by a factor that doesn’t involve θ . Since the $g(\theta, \phi)$ function above, when viewed as a function of θ is proportional to both these expressions, we might as well have replaced it with $\Pr(\theta | \phi, y)$ in our update for θ .

This distribution $\Pr(\theta | \phi, y)$ is called the **full conditional distribution** for θ .

Full conditional distribution

Why use $\Pr(\theta | \phi, y)$ instead of $g(\theta, \phi)$?

In some cases, the full conditional distribution is a standard distribution we know how to sample. If that happens, we no longer need to draw a candidate and decide whether to accept it. In fact, if we treat the full conditional distribution as a candidate proposal distribution, the resulting Metropolis-Hastings acceptance probability becomes exactly 1.

Gibbs samplers require a little more work up front because you need to find the full conditional distribution for each parameter. The good news is that all full conditional distributions have the same starting point: the full joint posterior distribution. Using the example above, we have

$$\Pr(\theta | \phi, y) \propto \Pr(\theta, \phi | y)$$

where we simply now treat ϕ as a known number. Likewise, the other full conditional is $\Pr(\phi | \theta, y) \propto \Pr(\theta, \phi | y)$ where here, we consider θ to be a known number. We always start with the full posterior distribution.

Thus, the process of finding full conditional distributions is the same as finding the posterior distribution of each parameter, pretending that all other parameters are known.

27.1.2 Gibbs sampler

i Note

The idea of Gibbs sampling is that we can **update multiple parameters by sampling just one parameter at a time**, cycling through all parameters and repeating. To perform the update for one particular parameter, we substitute in the current values of all other parameters.

Here is the algorithm. Suppose we have a joint posterior distribution for two parameters θ and ϕ , written $\Pr(\theta, \phi | y)$. If we can find the distribution of each parameter at a time, i.e., $\Pr(\theta | \phi, y)$ and $\Pr(\phi | \theta, y)$, then we can take turns sampling these distributions like so:

1. Using ϕ_{i-1} , draw θ_i from $\Pr(\theta | \phi = \phi_{i-1}, y)$.
2. Using θ_i , draw ϕ_i from $\Pr(\phi | \theta = \theta_i, y)$.

Together, steps 1 and 2 complete one cycle of the Gibbs sampler and produce the draw for (θ_i, ϕ_i) in one iteration of a MCMC sampler. If there are more than two parameters, we can handle that also. One Gibbs cycle would include an update for each of the parameters.

In the following segments, we will provide a concrete example of finding full conditional distributions and constructing a Gibbs sampler.

27.2 Conditionally conjugate prior example with Normal likelihood

27.2.1 Normal likelihood, unknown mean and variance

$$\begin{aligned}
 y_i | \mu, \sigma^2 &\stackrel{\text{iid}}{\sim} N(\mu, \sigma^2) \quad i=1, \dots, n \\
 \mu &\sim N(\mu_0, \sigma_0^2) \\
 \sigma^2 &\sim \text{IG}(\nu_0, \beta_0) \\
 p(\mu, \sigma^2 | y_1, \dots, y_n) &\propto p(y_1, \dots, y_n | \mu, \sigma^2) p(\mu) p(\sigma^2) \\
 &= \prod_{i=1}^n \left[N(y_i | \mu, \sigma^2) \right] N(\mu | \mu_0, \sigma_0^2) \text{IG}(\sigma^2 | \nu_0, \beta_0) \\
 &= \prod_{i=1}^n \left[\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(y_i - \mu)^2\right) \right] \frac{1}{\sqrt{2\pi}\sigma_0} \exp\left(-\frac{1}{2\sigma_0^2}(\mu - \mu_0)^2\right) \frac{\beta_0^{\nu_0}}{\Gamma(\nu_0)} \sigma^{\nu_0} \exp\left(-\frac{\beta_0}{\sigma^2}\right) \\
 &\propto (\sigma^2)^{-\frac{n}{2}} \exp\left[\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mu)^2\right] (\sigma^2)^{(\nu_0+1)} \exp\left[-\frac{\beta_0}{\sigma^2}\right]
 \end{aligned}$$

Figure 27.3: Normal likelihood, unknown mean and variance

$$\begin{aligned}
 p(\mu, \sigma^2 | y_1, \dots, y_n) &\propto (\sigma^2)^{-\frac{n}{2}} \exp\left[-\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mu)^2\right] (\sigma^2)^{(\nu_0+1)} \exp\left[-\frac{\beta_0}{\sigma^2}\right] \\
 p(\mu | \sigma^2, y_1, \dots, y_n) &\propto p(\mu, \sigma^2 | y_1, \dots, y_n) \\
 &\propto \exp\left[-\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mu)^2\right] \exp\left[-\frac{\beta_0}{\sigma^2}\right] \\
 &= \exp\left[-\frac{1}{2} \left(\frac{\sum (y_i - \mu)^2}{\sigma^2} + \frac{(\mu - \mu_0)^2}{\sigma_0^2} \right)\right] \\
 &\propto N(\mu | \frac{\nu_0 \mu_0 + \nu_0 \mu_0}{\nu_0 + \sigma_0^2}, \frac{\sigma_0^2}{\nu_0 + \sigma_0^2}) \\
 p(\sigma^2 | \mu, y_1, \dots, y_n) &\propto p(\mu, \sigma^2 | y_1, \dots, y_n) \\
 &\propto (\sigma^2)^{-\frac{n}{2}} \exp\left[-\frac{1}{2\sigma^2} \sum_{i=1}^n (y_i - \mu)^2\right] (\sigma^2)^{(\nu_0+1)} \exp\left[-\frac{\beta_0}{\sigma^2}\right] \\
 &= (\sigma^2)^{(\nu_0+1)} \exp\left[-\frac{1}{\sigma^2} \left(\beta_0 + \frac{\sum (y_i - \mu)^2}{2} \right)\right] \\
 &\propto \text{IG}(\sigma^2 | \nu_0 + \frac{n}{2}, \beta_0 + \frac{\sum (y_i - \mu)^2}{2})
 \end{aligned}$$

Figure 27.4: Normal likelihood conjugate prior

Let's return to the example at the end of Lesson 2 where we have normal likelihood with unknown mean and unknown variance. The model is:

$$\begin{aligned}
 y_i | \mu, \sigma^2 &\stackrel{\text{iid}}{\sim} N(\mu, \sigma^2), \quad i = 1, \dots, n \\
 \mu &\sim N(\mu_0, \sigma_0^2) \\
 \sigma^2 &\sim \text{IG}(\nu_0, \beta_0).
 \end{aligned}$$

We chose a normal prior for μ because, in the case where σ^2 is known, the normal is the conjugate prior for μ . Likewise, in the case where μ is known, the inverse-gamma is the conjugate prior for σ^2 . This will give us convenient full conditional distributions in a Gibbs sampler.

Let's first work out the form of the full posterior distribution. When we begin analyzing data, the JAGS software will complete this step for us. However, it is extremely valuable to see and understand how this works.

$$\begin{aligned}
\Pr(\mu, \sigma^2 | y_1, y_2, \dots, y_n) &\propto \Pr(y_1, y_2, \dots, y_n | \mu, \sigma^2) \Pr(\mu) \Pr(\sigma^2) \\
&= \prod_{i=1}^n N(y_i | \mu, \sigma^2) \times N(\mu | \mu_0, \sigma_0^2) \times IG(\sigma^2 | \nu_0, \beta_0) \\
&= \prod_{i=1}^n \frac{1}{\sqrt{2\pi\sigma^2}} \exp \left[-\frac{(y_i - \mu)^2}{2\sigma^2} \right] \\
&\quad \times \frac{1}{\sqrt{2\pi\sigma_0^2}} \exp \left[-\frac{(\mu - \mu_0)^2}{2\sigma_0^2} \right] \\
&\quad \times \frac{\beta_0^{\nu_0}}{\Gamma(\nu_0)} (\sigma^2)^{-(\nu_0+1)} \exp \left[-\frac{\beta_0}{\sigma^2} \right] \mathbb{I}_{\sigma^2 > 0}(\sigma^2) \\
&\propto (\sigma^2)^{-n/2} \exp \left[-\frac{\sum_{i=1}^n (y_i - \mu)^2}{2\sigma^2} \right] \\
&\quad \times \exp \left[-\frac{(\mu - \mu_0)^2}{2\sigma_0^2} \right] (\sigma^2)^{-(\nu_0+1)} \\
&\quad \times \exp \left[-\frac{\beta_0}{\sigma^2} \right] \mathbb{I}_{\sigma^2 > 0}(\sigma^2)
\end{aligned}$$

From here, it is easy to continue on to find the two full conditional distributions we need.

First let's look at μ , assuming σ^2 is known (in which case it becomes a constant and is absorbed into the normalizing constant):

$$\begin{aligned}
\Pr(\mu | \sigma^2, y_1, \dots, y_n) &\propto \Pr(\mu, \sigma^2 | y_1, \dots, y_n) \\
&\propto \exp \left[-\frac{\sum_{i=1}^n (y_i - \mu)^2}{2\sigma^2} \right] \exp \left[-\frac{(\mu - \mu_0)^2}{2\sigma_0^2} \right] \\
&\propto \exp \left[-\frac{1}{2} \left(\frac{\sum_{i=1}^n (y_i - \mu)^2}{2\sigma^2} + \frac{(\mu - \mu_0)^2}{2\sigma_0^2} \right) \right] \\
&\propto N \left(\mu | \frac{n\bar{y}/\sigma^2 + \mu_0/\sigma_0^2}{n/\sigma^2 + 1/\sigma_0^2}, \frac{1}{n/\sigma^2 + 1/\sigma_0^2} \right)
\end{aligned} \tag{27.1}$$

which we derived in the supplementary material of the last course. So, given the data and σ^2 , μ follows this normal distribution.

Now let's look at σ^2 , assuming μ is known:

$$\begin{aligned}
 \Pr(\sigma^2 | \mu, y_1, \dots, y_n) &\propto \Pr(\mu, \sigma^2 | y_1, \dots, y_n) \\
 &\propto (\sigma^2)^{-n/2} \exp\left[-\frac{\sum_{i=1}^n (y_i - \mu)^2}{2\sigma^2}\right] (\sigma^2)^{-(\nu_0+1)} \exp\left[-\frac{\beta_0}{\sigma^2}\right] I_{\sigma^2>0}(\sigma^2) \\
 &\propto (\sigma^2)^{-(\nu_0+n/2+1)} \exp\left[-\frac{1}{\sigma^2} \left(\beta_0 + \frac{\sum_{i=1}^n (y_i - \mu)^2}{2}\right)\right] I_{\sigma^2>0}(\sigma^2) \\
 &\propto \text{IG}\left(\sigma^2 | \nu_0 + \frac{n}{2}, \beta_0 + \frac{\sum_{i=1}^n (y_i - \mu)^2}{2}\right)
 \end{aligned} \tag{27.2}$$

These two distributions provide the basis of a Gibbs sampler to simulate from a Markov chain whose stationary distribution is the full posterior of both μ and σ^2 . We simply alternate draws between these two parameters, using the most recent draw of one parameter to update the other.

We will do this in R in the next segment.

27.3 Computing example with Normal likelihood

To implement the Gibbs sampler we just described, let's return to our running example where the data are the percent change in total personnel from last year to this year for $n = 10$ companies. We'll still use a **normal likelihood**, but now we'll *relax the assumption that we know the variance of growth between companies, σ^2* , and estimate that variance. Instead of the t prior from earlier, we will use the **conditionally conjugate priors**, normal for μ and inverse-gamma for σ^2 .

Company personnel

The first step will be to write functions to simulate from the full conditional distributions we derived in the previous segment. The full conditional for μ , given σ^2 and data is

27.3.1 conditionally conjugate priors for the mean

$$N\left(\mu | \frac{n\bar{y}/\sigma^2 + \mu_0/\sigma_0^2}{n/\sigma^2 + 1/\sigma_0^2}, \frac{1}{n/\sigma^2 + 1/\sigma_0^2}\right) \tag{27.3}$$

```

#' update_mu
#
#' @param n - sample size
#' @param ybar - sample mean
#' @param sig2 - current sigma squared
#' @param mu_0 - mean hyper-parameter
#' @param sig2_0 - variance hyper-parameter
#
#' @output - updated value for mu the mean
update_mu = function(n, ybar, sig2, mu_0, sig2_0) {          ①
  sig2_1 = 1.0 / (n / sig2 + 1.0 / sig2_0)                ②
  mu_1 = sig2_1 * (n * ybar / sig2 + mu_0 / sig2_0)       ③
  rnorm(n=1, mean=mu_1, sd=sqrt(sig2_1))                  ④
}

```

- ① we don't need the data y
- ② where: sig2_1 is σ_1^2 the right term in Equation 27.3 sig2 is the current σ_2 which we update in `update_sigma` below using Equation 27.4
 sig2_0 is the hyper parameter for σ_0^2
- ③ mu_1 is σ_1^2 the left term in Equation 27.3 which uses sig2_1 we just computed
- ④ we now draw from the a $N(\mu_1, \sigma_1^2)$ for `update_sig2` and the trace.

27.3.2 conditionally conjugate priors for the variance

The full conditional for σ^2 given μ and data is

$$\text{IG} \left(\sigma^2 \mid \nu_0 + \frac{n}{2}, \beta_0 + \frac{\sum_{i=1}^n (y_i - \mu)^2}{2} \right) \quad (27.4)$$

```

#' update_sig2
#
#' @param n - sample size
#' @param y - the data
#' @param nu_0 - nu hyper-parameter
#' @param beta_0 - beta hyper-parameter
#
#' @output - updated value for sigma2 the variance
update_sig2 = function(n, y, mu, nu_0, beta_0) {           ①
}

```

```

nu_1 = nu_0 + n / 2.0                                ②
sumsq = sum( (y - mu)^2 )                               ③
beta_1 = beta_0 + sumsq / 2.0                           ④
out_gamma = rgamma(n=1, shape=nu_1, rate=beta_1)        ⑤
1.0 / out_gamma                                         ⑥
}

```

- ① we need the data to update beta
- ② nu_1 the left term in Equation 27.4
- ③ vectorized
- ④ beta_1 the right term in Equation 27.4
- ⑤ draw a gamma sample with updated rate for Gamma() is shape for IG() inv-gamma
- ⑥ since there is no rinvgamma in R we use the reciprocal of a gamma random variable which is distributed inv-gamma

With functions for drawing from the full conditionals, we are ready to write a function to perform Gibbs sampling.

27.3.3 Gibbs sampler in R

```

gibbs = function(y, n_iter, init, prior) {
  ybar = mean(y)
  n = length(y)

  ## initialize
  mu_out = numeric(n_iter)
  sig2_out = numeric(n_iter)

  mu_now = init$mu

  ## Gibbs sampler
  for (i in 1:n_iter) {
    sig2_now = update_sig2(n=n, y=y, mu=mu_now, nu_0=prior$nu_0, beta_0=prior$beta_0)
    mu_now = update_mu(n=n, ybar=ybar, sig2=sig2_now, mu_0=prior$mu_0, sig2_0=prior$sig2_0)

    sig2_out[i] = sig2_now
    mu_out[i] = mu_now
  }
}

```

```

cbind(mu=mu_out, sig2=sig2_out) ①
}

```

- ① `cbind` for column bind will take a lists of list and convert them into a matrix of columns.

Now we are ready to set up the problem in R.

$$\begin{aligned}
 y_i | \mu, \sigma &\stackrel{iid}{\sim} N(\mu, \sigma^2), \quad i = 1, \dots, n \\
 \mu &\sim N(\mu_0, \sigma_0^2) \\
 \sigma^2 &\sim IG(\nu, \beta_0)
 \end{aligned} \tag{27.5}$$

We also need to create the prior hyperparameters for σ^2 , ν_0 and β_0 . If we chose these hyperparameters carefully, they are interpretable as a prior guess for sigma squared, as well as a prior effective sample size to go with that guess.

The prior effective sample size. Which we'll call n_0 , is two times this ν_0 parameter. So in other words, the nu parameter will be the prior sample size Divided by 2. We're also going to create an initial guess for sigma squared, let's call it s_0^2 . The relationship between β_0 and these two numbers is the following: It is the prior sample size times the prior guess divided by 2.

This particular parameterization of the *Inverse gamma* distribution is called the **Scaled Inverse Chi Square Distribution**, where the two parameters are n_0 and s_0^2 .

```

y = c(1.2, 1.4, -0.5, 0.3, 0.9, 2.3, 1.0, 0.1, 1.3, 1.9) ①
ybar = mean(y)
n = length(y)

## prior
prior = list()
prior$mu_0 = 0.0
prior$sig2_0 = 1.0
prior$n_0 = 2.0 # prior effective sample size for sig2
prior$s2_0 = 1.0 # prior point estimate for sig2
prior$nu_0 = prior$n_0 / 2.0 # prior parameter for inverse-gamma
prior$beta_0 = prior$n_0 * prior$s2_0 / 2.0 # prior parameter for inverse-gamma

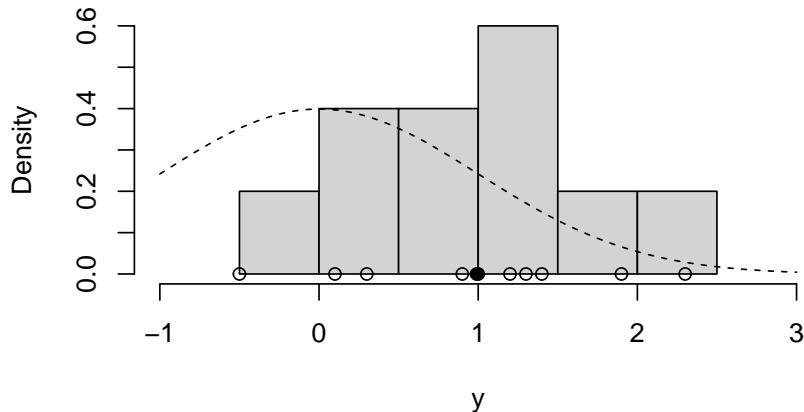
```

```

hist(y, freq=FALSE, xlim=c(-1.0, 3.0)) # histogram of the data
curve(dnorm(x=x, mean = prior$mu_0, sd=sqrt(prior$sig2_0)), lty=2, add=TRUE) # prior for mu
points(y, rep(0,n), pch=1) # individual data points
points(ybar, 0, pch=19) # sample mean

```

Histogram of y



Finally, we can initialize and run the sampler!

```

set.seed(53)

init = list()
init$mu = 0.0

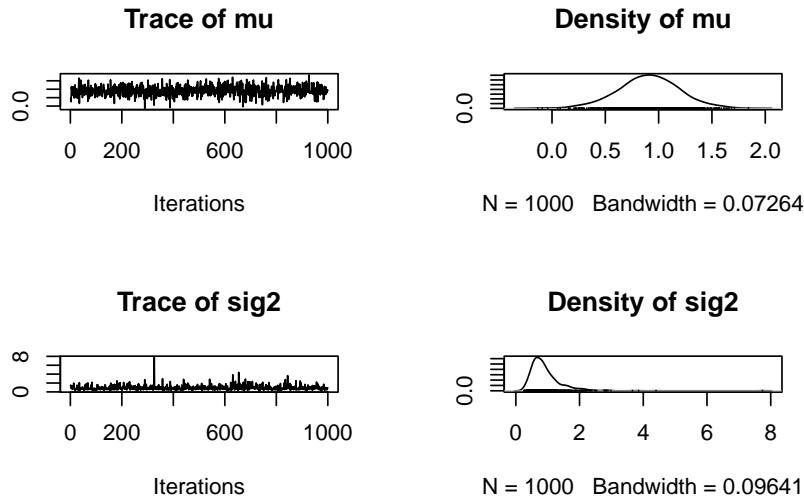
post = gibbs(y=y, n_iter=1e3, init=init, prior=prior)

head(post)

      mu      sig2
[1,] 0.3746992 1.5179144
[2,] 0.4900277 0.8532821
[3,] 0.2536817 1.4325174
[4,] 1.1378504 1.2337821
[5,] 1.0016641 0.8409815
[6,] 1.1576873 0.7926196

```

```
library("coda")
plot(as.mcmc(post))
```



```
summary(as.mcmc(post))
```

```
Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000
```

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

	Mean	SD	Naive SE	Time-series SE
mu	0.9051	0.2868	0.00907	0.00907
sig2	0.9282	0.5177	0.01637	0.01810

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
mu	0.3024	0.7244	0.9089	1.090	1.481
sig2	0.3577	0.6084	0.8188	1.094	2.141

As with the Metropolis-Hastings example, these chains appear to have converged. In the next lesson, we will discuss convergence in more detail.

28 Assessing Convergence

Bayesian Statistics: Techniques and Models

29 Notes - Assessing Convergence

29.1 Convergence diagnostics

In the previous two lessons, we have demonstrated ways you can simulate a Markov chain whose stationary distribution is the target distribution (usually the posterior). Before using the simulated chain to obtain Monte Carlo estimates, we should first ask ourselves: Has our simulated Markov chain converged to its stationary distribution yet? Unfortunately, this is a difficult question to answer, but we can do several things to investigate.

29.1.1 Trace plots

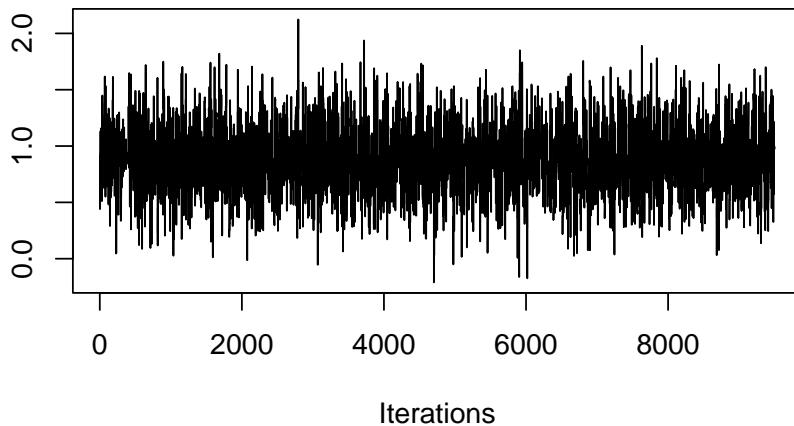
Our first visual tool for assessing chains is the trace plot. A trace plot shows the history of a parameter value across iterations of the chain. It shows you precisely where the chain has been exploring.

First, let's talk about what a chain should look like. Here is an example of a chain that has most likely converged.

```
source('mh.r')

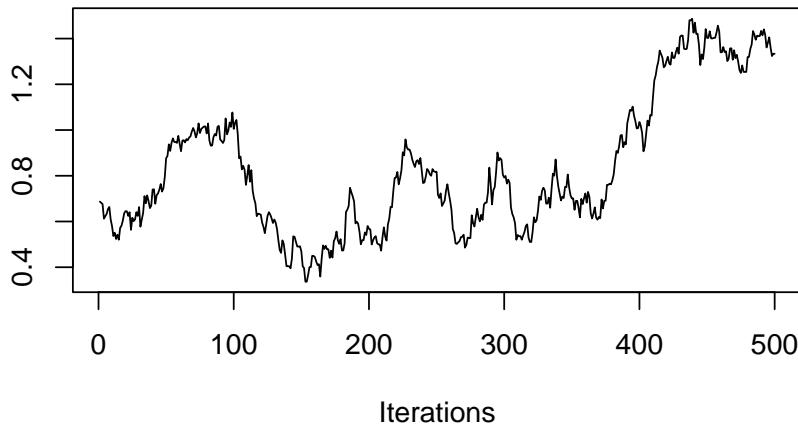
List of 2
$ mu    : num [1:1000] 0.435 0.435 0.435 0.716 0.716 ...
$ accpt: num 0.145

library("coda")
set.seed(61)
post0 = mh(n=n, ybar=ybar, n_iter=10e3, mu_init=0.0, cand_sd=0.9)
coda::traceplot(as.mcmc(post0$mu[-c(1:500)]))
```



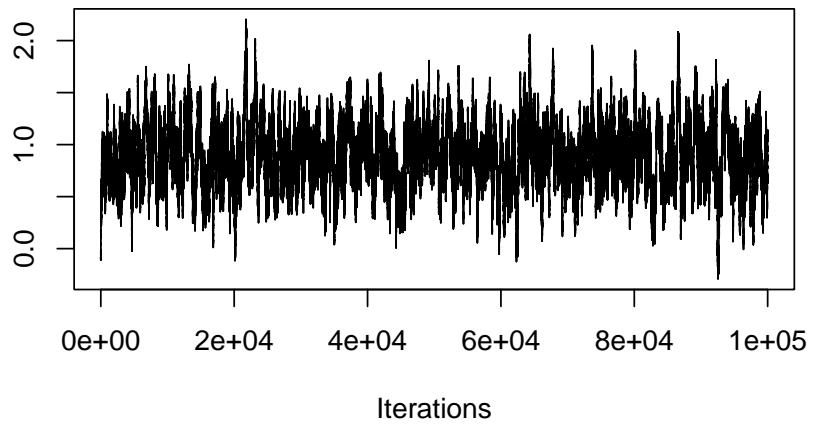
If the chain is stationary, it should not be showing any long-term trends. The average value for the chain should be roughly flat. It should not be wandering as in this example:

```
set.seed(61)
post1 = mh(n=n, ybar=ybar, n_iter=1e3, mu_init=0.0, cand_sd=0.04)
coda::traceplot(as.mcmc(post1$mu[-c(1:500)]))
```



If this is the case, you need to run the chain many more iterations, as seen here:

```
set.seed(61)
post2 = mh(n=n, ybar=ybar, n_iter=100e3, mu_init=0.0, cand_sd=0.04)
coda::traceplot(as.mcmc(post2$mu))
```

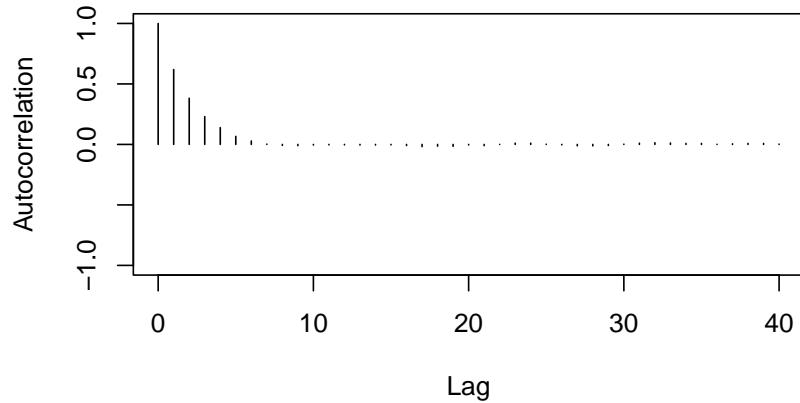


The chain appears to have converged at this much larger time scale.

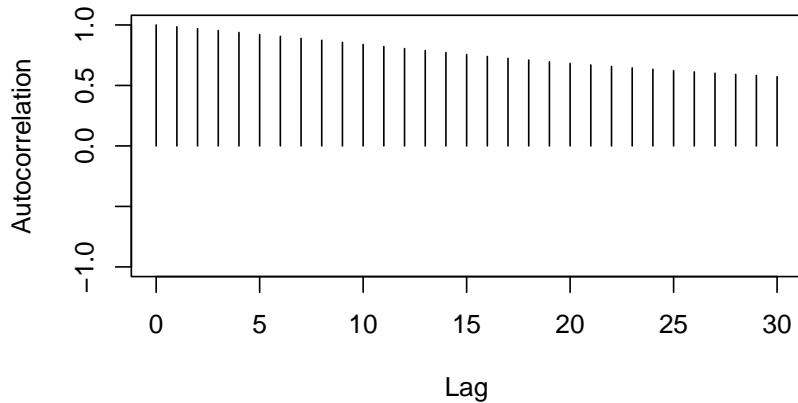
29.1.2 Monte Carlo effective sample size

One major difference between the two chains we've looked at is the level of autocorrelation in each. Autocorrelation is a number between -1 and $+1$ which measures how linearly dependent the current value of the chain is on past values (called lags). We can see this with an autocorrelation plot:

```
coda::autocorr.plot(as.mcmc(post0$mu))
```



```
coda::autocorr.plot(as.mcmc(post1$mu))
```



```
coda::autocorr.diag(as.mcmc(post1$mu))
```

```
[ ,1]
Lag 0  1.0000000
Lag 1  0.9850078
Lag 5  0.9213126
Lag 10 0.8387333
Lag 50 0.3834563
```

Autocorrelation is important because it tells us how much information is available in our Markov chain. Sampling 1000 iterations from a highly correlated Markov chain yields less information about the stationary distribution than we would obtain from 1000 samples *independently* drawn from the stationary distribution.

Autocorrelation is a major component in calculating the Monte Carlo effective sample size of your chain. The Monte Carlo effective sample size is how many *independent* samples from the stationary distribution you would have to draw to have equivalent information in your Markov chain. Essentially it is the m (sample size) we chose in the lesson on Monte Carlo estimation.

```
str(post2) # contains 100,000 iterations
```

```
List of 2
 $ mu    : num [1:100000] -0.0152 -0.1007 -0.0867 -0.1092 -
 0.0811 ...
 $ accpt: num 0.958
```

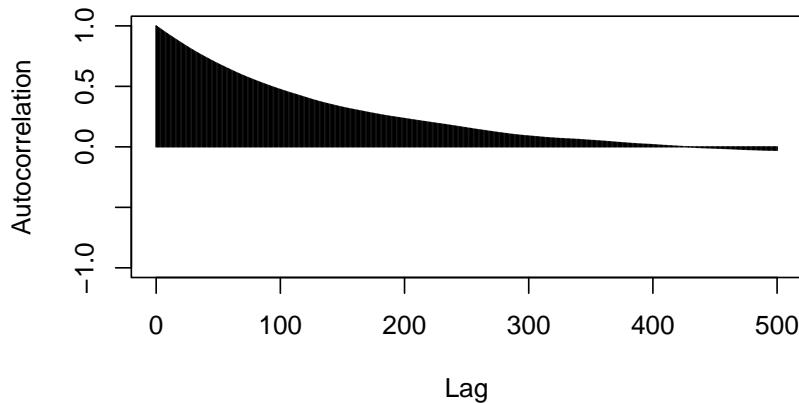
```
coda::effectiveSize(as.mcmc(post2$mu)) # effective sample size of ~350
```

```
var1
```

```
373.858
```

```
## thin out the samples until autocorrelation is essentially 0. This will leave you with approximately independent samples.
```

```
coda::autocorr.plot(as.mcmc(post2$mu), lag.max=500)
```



```
thin_interval = 400 # how far apart the iterations are for autocorrelation to be essentially 0.
```

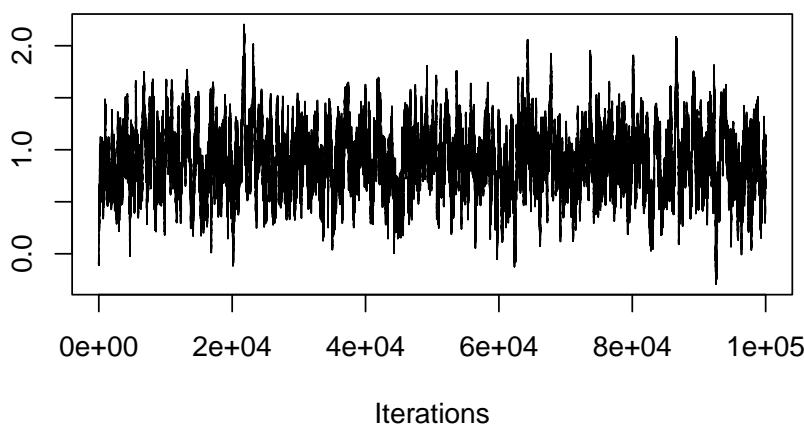
```
thin_idx = seq(from=thin_interval, to=length(post2$mu), by=thin_interval)
```

```
head(thin_idx)
```

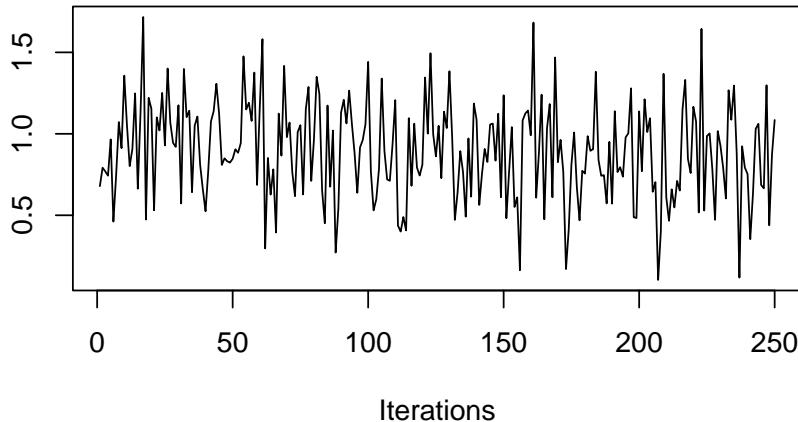
```
[1] 400 800 1200 1600 2000 2400
```

```
post2mu_thin = post2$mu[thin_idx]
```

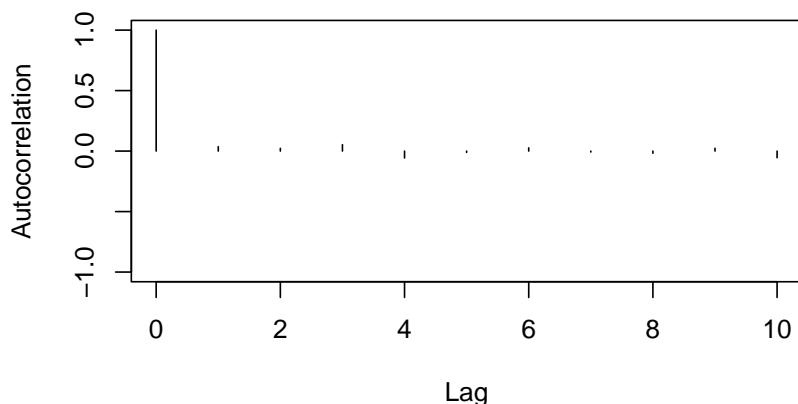
```
traceplot(as.mcmc(post2$mu))
```



```
traceplot(as.mcmc(post2mu_thin))
```



```
coda::autocorr.plot(as.mcmc(post2mu_thin), lag.max=10)
```



```
effectiveSize(as.mcmc(post2mu_thin))
```

```
var1  
250
```

```
length(post2mu_thin)
```

```
[1] 250
```

```

str(post0) # contains 10,000 iterations

List of 2
$ mu    : num [1:10000] 0 0 0.315 0.315 0.949 ...
$ accpt: num 0.382

coda::effectiveSize(as.mcmc(post0$mu)) # effective sample size of ~2,500

var1
2537.924

?effectiveSize

```

The chain from `post0` has 10,000 iterations, but an effective sample size of about 2,500. That is, this chain essentially provides the equivalent of 2,500 independent Monte Carlo samples.

Notice that the chain from `post0` has 10 times fewer iterations than for `post2`, but its Monte Carlo effective sample size is about seven times greater than the longer (more correlated) chain. We would have to run the correlated chain for 700,000+ iterations to get the same amount of information from both chains.

It is usually a good idea to check the Monte Carlo effective sample size of your chain. If all you seek is a posterior mean estimate, then an effective sample size of a few hundred to a few thousand should be enough. However, if you want to create something like a 95% posterior interval, you may need many thousands of effective samples to produce a reliable estimate of the outer edges of the distribution. The number you need can be quickly calculated using the Raftery and Lewis diagnostic.

```

raftery.diag(as.mcmc(post0$mu))

raftery.diag(as.mcmc(post0$mu), q=0.005, r=0.001, s=0.95)

```

```

Quantile (q) = 0.005
Accuracy (r) = +/- 0.001

```

```

Probability (s) = 0.95

You need a sample size of at least 19112 with these values of q, r and s

## 
## Quantile (q) = 0.005
## Accuracy (r) = +/- 0.001
## Probability (s) = 0.95
##
## You need a sample size of at least 19112 with these values of q, r and s

?raftery.diag

```

In the case of the first chain from `post0`, it looks like we would need about 3,700 effective samples to calculate reliable 95% intervals. With the autocorrelation in the chain, that requires about 13,200 total samples. If we wanted to create reliable 99% intervals, we would need at least 19,100 total samples.

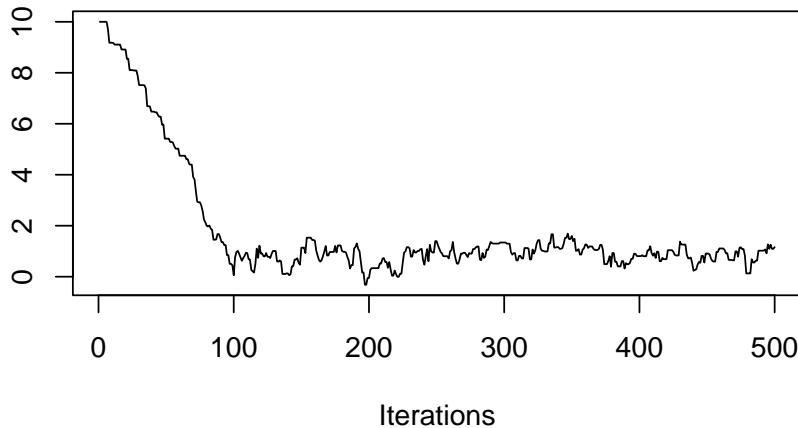
29.2 Burn-in

We have also seen how the initial value of the chain can affect how quickly the chain converges. If our initial value is far from the bulk of the posterior distribution, then it may take a while for the chain to travel there. We saw this in an earlier example.

```

set.seed(62)
post3 = mh(n=n, ybar=ybar, n_iter=500, mu_init=10.0, cand_sd=0.3)
coda::traceplot(as.mcmc(post3$mu))

```



Clearly, the first 100 or so iterations do not reflect draws from the stationary distribution, so they should be discarded before we use this chain for Monte Carlo estimates. This is called the “burn-in” period. You should always discard early iterations that do not appear to be coming from the stationary distribution. Even if the chain appears to have converged early on, it is safer practice to discard an initial burn-in.

29.2.1 Multiple chains, Gelman-Rubin

If we want to be more confident that we have converged to the true stationary distribution, we can simulate multiple chains, each with a different starting value.

```
set.seed(61)

nsim = 500
post1 = mh(n=n, ybar=ybar, n_iter=nsim, mu_init=15.0, cand_sd=0.4)
post1$accpt

[1] 0.616

post2 = mh(n=n, ybar=ybar, n_iter=nsim, mu_init=-5.0, cand_sd=0.4)
post2$accpt

[1] 0.612
```

```
post3 = mh(n=n, ybar=ybar, n_iter=nsim, mu_init=7.0, cand_sd=0.1)
post3$accpt
```

```
[1] 0.844
```

```
post4 = mh(n=n, ybar=ybar, n_iter=nsim, mu_init=23.0, cand_sd=0.5)
post4$accpt
```

```
[1] 0.53
```

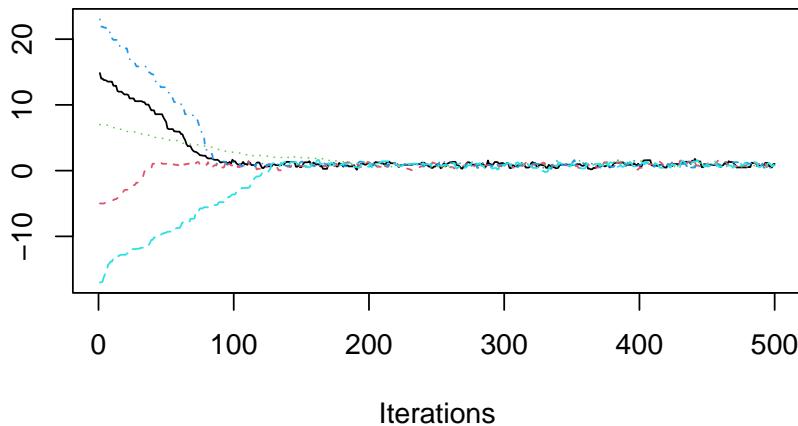
```
post5 = mh(n=n, ybar=ybar, n_iter=nsim, mu_init=-17.0, cand_sd=0.4)
post5$accpt
```

```
[1] 0.618
```

```
pmc = mcmc.list(as.mcmc(post1$mu), as.mcmc(post2$mu),
                 as.mcmc(post3$mu), as.mcmc(post4$mu), as.mcmc(post5$mu))
str(pmc)
```

```
List of 5
$ : 'mcmc' num [1:500] 14.8 14 14 13.8 13.8 ...
..- attr(*, "mcpar")= num [1:3] 1 500 1
$ : 'mcmc' num [1:500] -5 -5 -5 -5 -4.89 ...
..- attr(*, "mcpar")= num [1:3] 1 500 1
$ : 'mcmc' num [1:500] 7 7 7 6.94 6.94 ...
..- attr(*, "mcpar")= num [1:3] 1 500 1
$ : 'mcmc' num [1:500] 23 21.9 21.9 21.8 21.8 ...
..- attr(*, "mcpar")= num [1:3] 1 500 1
$ : 'mcmc' num [1:500] -17 -17 -16.9 -16.2 -15.7 ...
..- attr(*, "mcpar")= num [1:3] 1 500 1
- attr(*, "class")= chr "mcmc.list"
```

```
coda::traceplot(pmc)
```



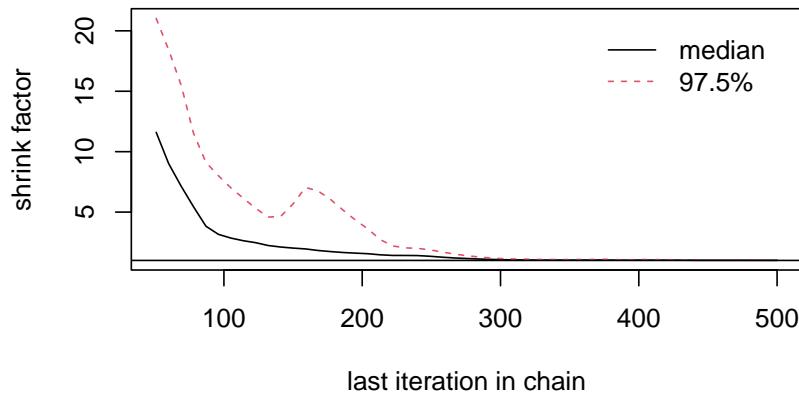
It appears that after about iteration 200, all chains are exploring the stationary (posterior) distribution. We can back up our visual results with the Gelman Rubin diagnostic. This diagnostic statistic calculates the variability within chains, comparing that to the variability between chains. If all chains have converged to the stationary distribution, the variability between chains should be relatively small, and the potential scale reduction factor, reported by the the diagnostic, should be close to one. If the values are much higher than one, then we would conclude that the chains have not yet converged.

```
coda::gelman.diag(pmc)
```

Potential scale reduction factors:

	Point est.	Upper C.I.
[1,]	1.01	1.02

```
coda::gelman.plot(pmc)
```



?gelman.diag

From the plot, we can see that if we only used the first 50 iterations, the potential scale reduction factor or “shrink factor” would be close to 10, indicating that the chains have not converged. But after about iteration 300, the “shrink factor” is essentially one, indicating that by then, we have probably reached convergence. Of course, we shouldn’t stop sampling as soon as we reach convergence. Instead, this is where we should begin saving our samples for Monte Carlo estimation.

29.2.2 Monte Carlo estimation

If we are reasonably confident that our Markov chain has converged, then we can go ahead and treat it as a Monte Carlo sample from the posterior distribution. Thus, we can use the techniques from Lesson 3 to calculate posterior quantities like the posterior mean and posterior intervals from the samples directly.

```
nburn = 1000 # remember to discard early iterations
post0$mu_keep = post0$mu[-c(1:1000)]
summary(as.mcmc(post0$mu_keep))
```

```
Iterations = 1:9000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 9000
```

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

Mean	SD	Naive SE	Time-series SE
0.889449	0.304514	0.003210	0.006295

2. Quantiles for each variable:

2.5%	25%	50%	75%	97.5%
0.2915	0.6825	0.8924	1.0868	1.4890

```
mean(post0$mu_keep > 1.0) # posterior probability that mu > 1.0
```

```
[1] 0.3554444
```

30 Notes - Linear regression

Bayesian Statistics: Techniques and Models

30.1 Introduction to linear regression

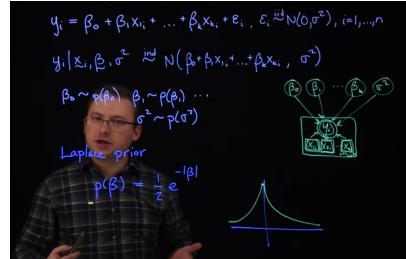


Figure 30.1: Introduction to linear regression

We discussed linear regression briefly in the previous course. And we fit a few models with non-informative priors. Here, we'll provide a brief review, demonstrate fitting linear regression models in JAGS And discuss a few practical skills that are helpful when fitting linear models in general.

This is not meant to be a comprehensive treatment of **linear models**, which you can find in numerous courses and textbooks.

Linear regression is perhaps the simplest way to relate a continuous response variable to multiple explanatory variables.

This may arise from observing several variables together and investigating which variables correlate with the response variable. Or it could arise from conducting an experiment, where we carefully assign values of explanatory

variables to randomly selected subjects. And try to establish a cause-and-effect relationship.

A linear regression model has the following form:

$$y_i = \beta_0 + \beta_1 x_i + \dots + \beta_k x_k + \epsilon_i \epsilon_i \stackrel{iid}{\sim} N(0, \sigma^2) \quad (30.1)$$

This describes the mean, and then we would also add an error, individual term for each observation. We would assume that the errors are IID from a normal distribution means 0 variance σ^2 for observations 1 ... k .

Equivalently we can write this model for y_i directly as y_i given all of the x_i values, betas and a constant variance σ^2 . Again, k is the number of predictor variables.

$$y_i | x_i, \beta_i, \sigma^2 \sim N(\beta_0 + \beta_1 x_i + \dots + \beta_k x_k, \sigma^2) \beta_i \sim Pr(\beta_i) \sigma^2 \sim Pr(\sigma^2) \quad (30.2)$$

This yields the following graphical model structure.

```
findfont: Font family ['STIXGeneral'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXGeneral'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXGeneral'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXGeneral'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXNonUnicode'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXNonUnicode'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXNonUnicode'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXSizeOneSym'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXSizeTwoSym'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXSizeThreeSym'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXSizeFourSym'] not found. Falling back to DejaVu Sans.
findfont: Font family ['STIXSizeFiveSym'] not found. Falling back to DejaVu Sans.
findfont: Font family ['cmtt10'] not found. Falling back to DejaVu Sans.
findfont: Font family ['cmb10'] not found. Falling back to DejaVu Sans.
findfont: Font family ['cmss10'] not found. Falling back to DejaVu Sans.
findfont: Font family ['DejaVu Sans Display'] not found. Falling back to DejaVu Sans.
```

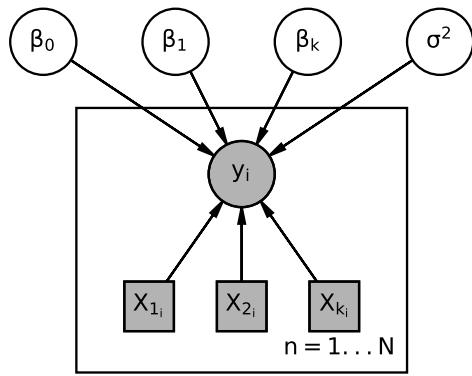


Figure 30.2: The graphical model for linear regression

! Understanding the Graphical Models

- This graphical model uses [plate notation](#)
- We'll start with a *plate* for all of our different y variables,
 - It is repeated $i = 1 \dots N$ times
- y_i , are random variable - (indicated by a circle)
 - they are observed - indicated by a filled shape.
- X_i variables.
 - they are drawn as squares around to indicate that they are constants and not random variables.
 - We're always conditioning on the X s. So they'll just be constants.
 - they are observed, so they are filled in.
- The y_i depend on the values of the x and the values of these parameters. So, we have β_0, \dots, β_k .
- Sigma squared.
- Since the y_i depend on all of these, so this would be the graphical model representation.

The terms of a linear model are always linearly related because of the structure of the model.

But the model does not have to be linear necessarily in the xy relationship.

For example, it may be that y is related linearly to x^2 . Hence we could transform the x and y variables to get new x 's and new y 's but we would still have a linear model. However, in that case, if we transform the variables, we must be careful about how this changes the final interpretation of the results.

! Interpreting Coefficients

The basic interpretation of the β_i coefficients is:

While holding all other X variables constant, if we increases X_i by one then the mean of \bar{y} is expected to increase by β_i .

That is β_i describes how the \bar{y} changes with changes in X_i , while accounting for all the other X variables.

$$\beta \approx \frac{\partial \bar{y}}{\partial x_i} \quad (30.3)$$

That's true for all of the x variables.

⚠ Regression assumptions

We're going to assume that

- The ys are independent of each other, given the xs .
- The y_i s have the same variance.
- The residuals are normally distributed with mean 0 and variance σ^2 .

These are actually strong assumptions that are not often realistic in many situations.

There are many statistical models to address that.

We'll look at some hierarchical methods in the coming lessons.

30.1.1 Priors

The model is not complete until we add the prior distributions.

So we might say β_0 comes from its prior.

β_1 would come from its prior, and so forth for all the β s. And sigma squared would come from its prior.

The most common choice for prior on the β s, is a **Normal distribution**. Or we can do a Multivariate normal for all of the betas at once.

This is conditionally conjugate and allows us to do Gibbs sampling.

If we want to be non-informative, we can choose $Normal(0, \sigma^2 = 1e6)$ priors with very large variance. Which are practically flat for realistic values of beta. The non-informative priors used in the last class are equivalent to using normal priors with infinite variance.

We can also use the *conditionally conjugate* **InverseGamma()** prior for σ^2 that we're familiar with.

Another common prior for the betas is **Double exponential**, or the **Laplace prior**, or **Laplace distribution**.

The Laplace prior has this density:

$$f(x | \mu, \beta) = \frac{1}{2\beta} e^{|x-\mu|} \quad (30.4)$$

where:

- μ is the location parameter and
- β is the scale parameter.

The case where $\mu = 0$ and $\beta = 1$ is called the **standard double exponential distribution**

$$f(x) = \frac{1}{2} e^{|x|} \quad (30.5)$$

And the density looks like this.

30.2 R

```
# Grid of X-axis values
x <- seq(-10, 10, 0.1)
plot(x, ddexp(x, 0, 2), type = "l", ylab = "", lwd = 2, col = "red")
lines(x, ddexp(x, 0, 1.5), type = "l", ylab = "", lwd = 2, col = "green")
lines(x, ddexp(x, 0, 1), type = "l", ylab = "", lwd = 2, col = "blue")
legend("topright",
c(expression(paste(beta)), "1.5", "1", "2"),
```

```

lty = c(0, 1, 1, 1),
col = c("red", "green", "blue"), box.lty = 0, lwd = 2
)

#x <- rdexp(500, location = 2, scale = 1)
#de_sample=ddexp(x, 2, 1)
#CDF <- ecdf(de_sample )
#plot(CDF)

```

30.3 Python

```

loc, scale = 0., 1.
s = np.random.laplace(loc, scale, 1000)

count, bins, ignored = plt.hist(s, 30, density=True)
x = np.arange(-8., 8., .01)
pdf = np.exp(-abs(x-loc)/scale)/(2.*scale)
plt.plot(x, pdf);

g = (1/(scale * np.sqrt(2 * np.pi)) *
     np.exp(-(x - loc)**2 / (2 * scale**2)))
plt.plot(x,g);

```

It's called *double exponential* because it looks like the exponential distribution except it's been reflected over the y axis. It has a sharp peak at x equals 0, or beta equals 0 in this case, which can be useful if we want to do variable selection among our x's. Because it'll favor values in your 0 for these betas.

This is related to the popular regression technique known as the **LASSO**.

More information is available from:

- [NIST](#)
- [Wikipedia](#)

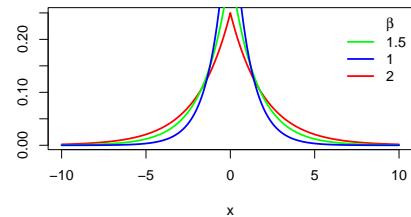


Figure 30.3: The Double Exponential Distribution

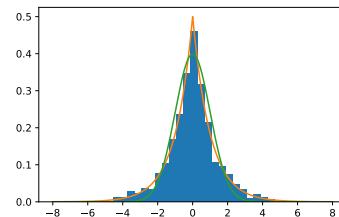


Figure 30.4: The Double Exponential Distribution

31 Lesson 7.2 Code - Setup & Data

As an example of linear regression, we'll look at the line heart data from the `car` package in R.

```
data("Leinhardt")  
?Leinhardt  
head(Leinhardt)
```

- ① use the `data` function and tell it which data set we want.
- ② browse the documentation for the dataset using `? operator`
- ③ look at the head of the data set.

	income	infant	region	oil
Australia	3426	26.7	Asia	no
Austria	3350	23.7	Europe	no
Belgium	3346	17.0	Europe	no
Canada	4751	16.8	Americas	no
Denmark	5029	13.5	Europe	no
Finland	3312	10.1	Europe	no

As an example of linear regression, we'll look at the *Leinhardt* data from the `car` package in R.

The `Leinhardt` dataset contains data on infant mortality for 105 nations of the world around the year 1970. It contains four variables:

- *Per capita income* in USD.
- *Infant mortality* rate per 1,000 live births.
- A factor indicating which *region* of the world this country is located.
- And finally, whether this country exports *oil* as an indicator variable.

```
str(Leinhardt)
```

- ① We can also look at the structure of the data using `str` function

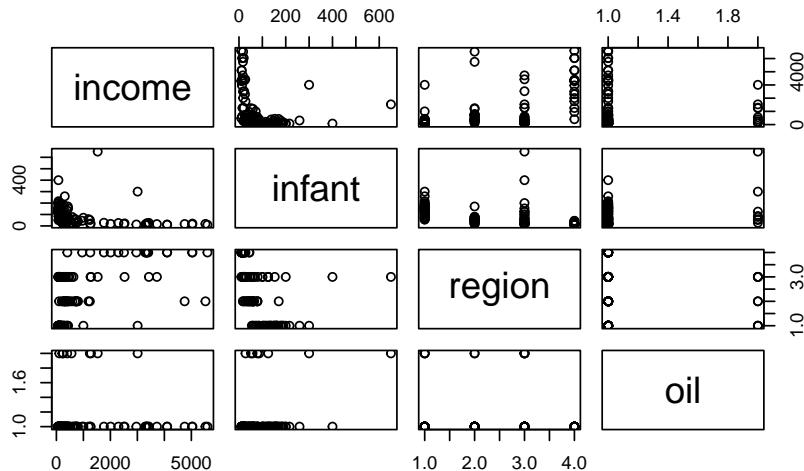
```
'data.frame': 105 obs. of 4 variables:
$ income: int 3426 3350 3346 4751 5029 3312 3403 5040 2009 2298 ...
$ infant: num 26.7 23.7 17 16.8 13.5 10.1 12.9 20.4 17.8 25.7 ...
$ region: Factor w/ 4 levels "Africa","Americas",...: 3 4 4 2 4 4 4 4 4 ...
$ oil    : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

31.1 EDA

`pairs(Leinhardt)`

①

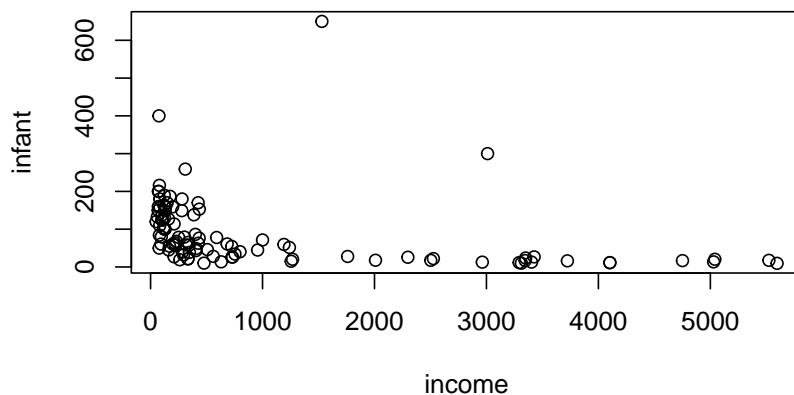
- ① Using `pairs` to investigate the marginal relationships between each of the four variables.



31.1.0.1 Simple linear Model

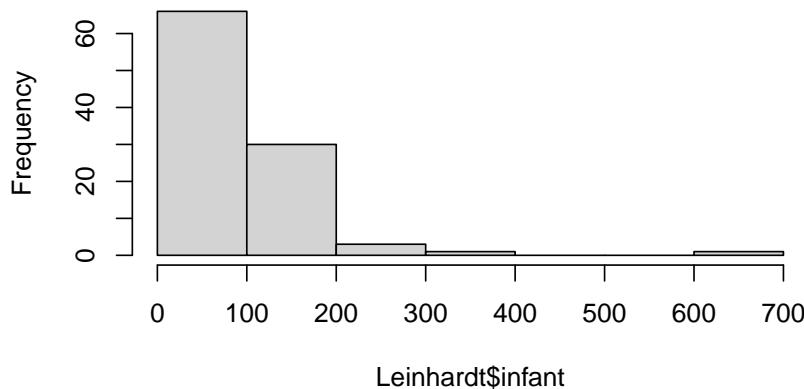
We'll start with a simple linear regression model that relates infant mortality to per capita income.

`plot(infant ~ income, data=Leinhardt)`



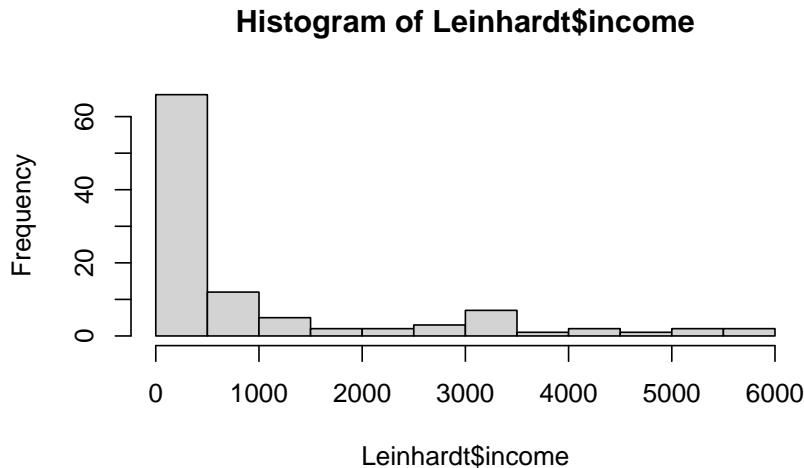
```
hist(Leinhardt$infant)
```

Histogram of Leinhardt\$infant



this is **right-skewed** (many small values and a number of much larger one)

```
hist(Leinhardt$income)
```



also right-skewed.

this indicates that we may do better if we do a **log transform** on these two variables.

31.1.0.2 Log-Log Linear Model

```
Leinhardt$loginfant = log(Leinhardt$infant)          ①
Leinhardt$logincome = log(Leinhardt$income)           ②
```

```
plot(loginfant ~ logincome, data=Leinhardt)          ③
```

- ① log transform *infant* column.
- ② log transform *income* column.
- ③ scatter plot of the log log transformed data.

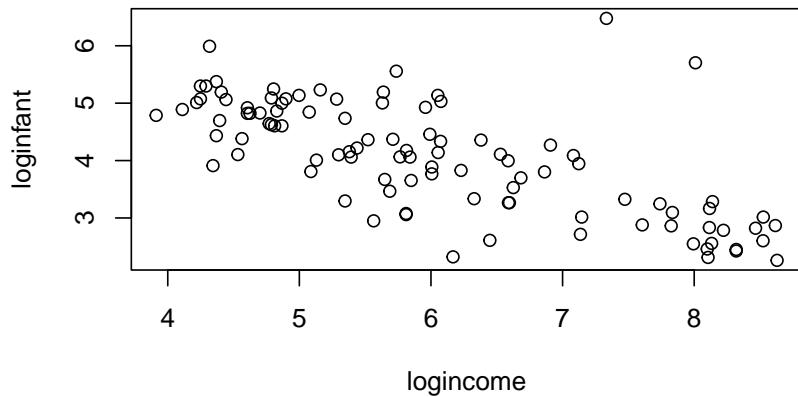


Figure 31.1: log log transformed infant mortality vs income

Since infant mortality and per capita income are positive and right-skewed quantities, we consider modeling them on the logarithmic scale. A linear model appears much more appropriate on this scale.

```
scatterplot(loginfant ~ logincome, data=Leinhardt) ①
```

① scatterplot with a regression fit and uncertainty for the data

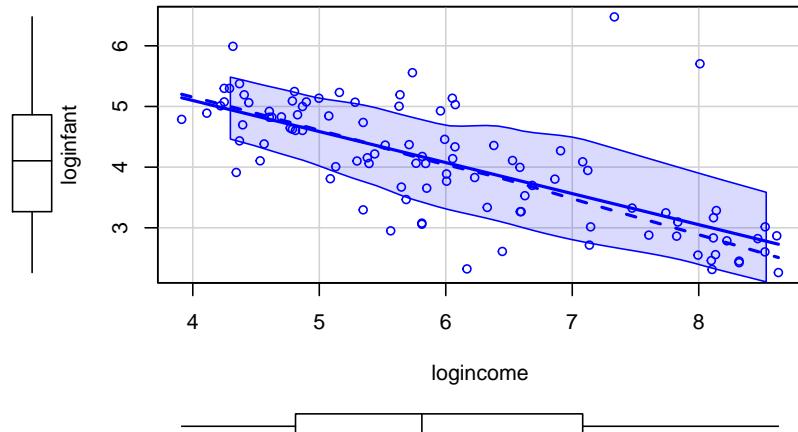
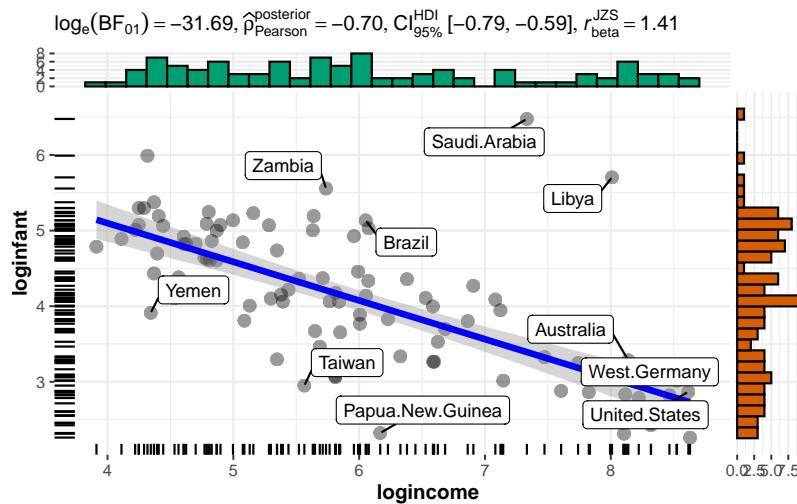


Figure 31.2: log log transformed infant mortality vs income scatterplot



31.1.1 Modeling

The reference Bayesian analysis (with a non-informative prior) is available directly in R.

```
lmod0 = lm(loginfant ~ logincome, data=Leinhardt)
summary(lmod0)
```

```
lm(formula = loginfant ~ logincome, data = Leinhardt)
```

Residuals:

Min	1Q	Median	3Q	Max
-1.66694	-0.42779	-0.02649	0.30441	3.08415

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	7.14582	0.31654	22.575	<2e-16 ***	①
logincome	-0.51179	0.05122	-9.992	<2e-16 ***	②

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

Residual standard error: 0.6867 on 99 degrees of freedom ③

(4 observations deleted due to missingness) ④

Multiple R-squared: 0.5021, Adjusted R-squared: 0.4971 ⑤

F-statistic: 99.84 on 1 and 99 DF, p-value: < 2.2e-16

- ① intercept is \gg its error so it appears statistically significant (***)
- ② posterior mean $\log \text{income}$ too
- ③ Residual standard error gives us an estimate of the left over variance after fitting the model.
- ④ **4 rows were dropped due to missing values**
- ⑤ *Adjusted R-squared* is the explained variance adjusted for degrees of freedom

32 Lesson 7.3 Code - Model in JAGS

Now we'll fit this model in JAGS. A few countries have missing values, and for simplicity, we will omit those.

```
dat = na.omit(Leinhardt)

Loading required package: coda

Linked to JAGS 4.3.2

Loaded modules: basemod, bugs

mod1_string = " model {
  for (i in 1:n) {
    y[i] ~ dnorm(mu[i], prec) # <1>
    mu[i] = b[1] + b[2]*log_income[i] # <2>
  }

  for (i in 1:2) {
    b[i] ~ dnorm(0.0, 1.0/1.0e6) # <3>
  }

  prec ~ dgamma(5/2.0, 5*10.0/2.0) # <4>
  sig2 = 1.0 / prec # <5>
  sig = sqrt(sig2) # <6>
} "

set.seed(72)
data1_jags = list(y=dat$loginfant, n=nrow(dat), log_income=dat$logincome) ⑦

params1 = c("b", "sig")
```

```

inits1 = function() {
  inits = list("b"=rnorm(2,0.0,100.0), "prec"=rgamma(1,1.0,1.0)) ⑧
}

mod1 = jags.model(textConnection(mod1_string), data=data1_jags, inits=inits1, n.chains=3) ⑨
update(mod1, 1000) ⑩

mod1_sim = coda.samples(model=mod1, variable.names=params1, n.iter=5000)

mod1_csim = do.call(rbind, mod1_sim) ⑪

```

- ① likelihood (we are using jags' mean, precision parameterization)
- ② linear part of the model
- ③ prior for the betas
- ④ prior - Inverse Gamma on the variation via a $precision = 1/\sigma^2$
- ⑤ we use the deterministic relationship between the precision and the variance to get to the standard deviation
- ⑥ we wish to monitor the standard deviation is more interpretable than the precision
- ⑦ the data is a list of the log-transformed infant mortality, the number of observations, and the log-transformed income
- ⑧ initial values: two Normals for beta and one from the gamma
- ⑨ putting the parts of the spec together
- ⑩ 1000 iterations to burn in
- ⑪ combine multiple chains by stacking the matrices

```

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
  Observed stochastic nodes: 101
  Unobserved stochastic nodes: 3
  Total graph size: 404

```

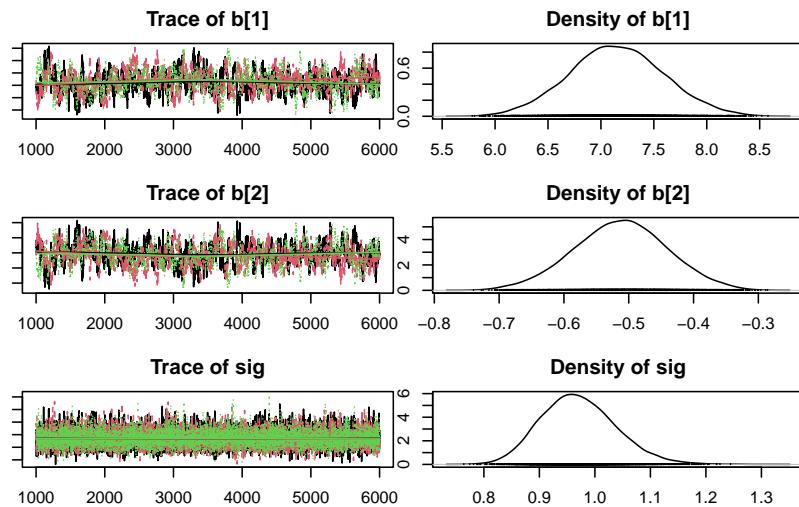
```
Initializing model
```

33 Lesson 7.4 Model Checking- MCMC convergence

33.0.1 Convergence diagnostics for the Markov Chains

Before we check the inferences from the model, we should perform convergence diagnostics for our Markov chains.

```
par(mar = c(2.5, 1, 2.5, 1))
plot(mod1_sim)
```



```
gelman.diag(mod1_sim)
```

①

- ① Gelman Rubin diagnostics scale reduction close to 1 indicated MCMC convergence

Potential scale reduction factors:

Point est. Upper C.I.

```
b[1]      1.01      1.02
b[2]      1.01      1.02
sig       1.00      1.00
```

Multivariate psrf

1

```
autocorr.diag(mod1_sim)
```

①

① high autocorrelation indicates a low sample size

```
          b[1]      b[2]      sig
Lag 0  1.00000000 1.00000000 1.000000000
Lag 1  0.95211343 0.95211854 0.010619708
Lag 5  0.78548466 0.78478942 0.012182901
Lag 10 0.61350414 0.61211527 0.006228351
Lag 50 0.08311278 0.08090856 -0.013502849
```

```
#autocorr.plot(mod1_sim , auto.layout = F)
autocorr.plot(mod1_csim , auto.layout = F)
```

```
effectiveSize(mod1_sim)
```

①

① checking the ESS

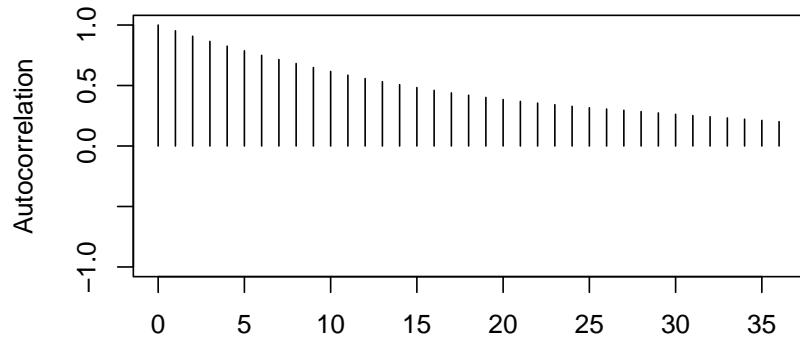
```
          b[1]      b[2]      sig
372.2298  367.8576 14877.9533
```

the ess for sigma is good but for the betas 350 this is too low for estimating quantiles, it is enough to estimate the mean.

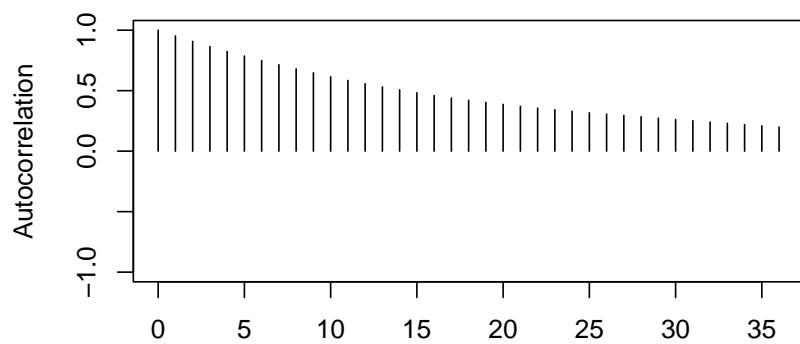
We can get a posterior summary of the parameters in our model.

```
summary(mod1_sim)
```

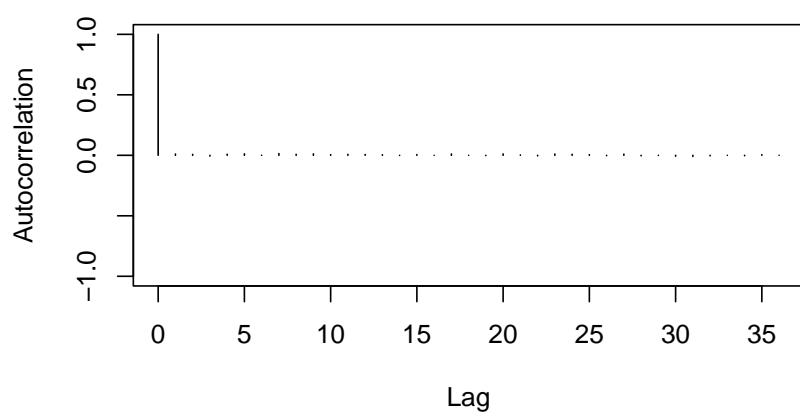
b[1]



b[2]



sig



```
Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000
```

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

	Mean	SD	Naive SE	Time-series SE
b[1]	7.1512	0.44427	0.0036275	0.0229657
b[2]	-0.5127	0.07183	0.0005865	0.0037377
sig	0.9708	0.06742	0.0005505	0.0005527

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
b[1]	6.2808	6.8473	7.1456	7.4496	8.0363
b[2]	-0.6565	-0.5609	-0.5116	-0.4639	-0.3722
sig	0.8508	0.9235	0.9669	1.0138	1.1126

Don't forget that these results are for a regression model relating the logarithm of infant mortality to the logarithm of income.

33.1 Residual checks

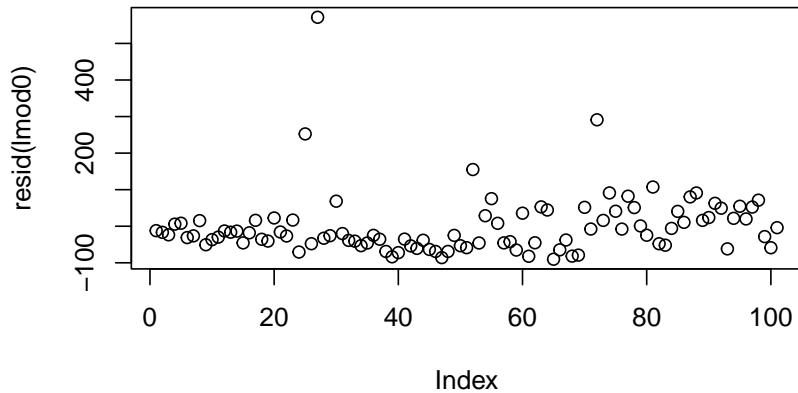
! Important

Analysis gets complicated quickly when we have multiple models. What we shall soon see is how to get residuals from the Bayesian model in Stan so we can compare it visually with the reference model we got using LM.

Checking residuals (the difference between the response and the model's prediction for that value) is important with linear models since residuals can reveal violations of the assumptions we made to specify the model. In particular, we are looking for any sign that the model is *not linear, normally distributed*, or that the *observations are not independent* (conditional on covariates).

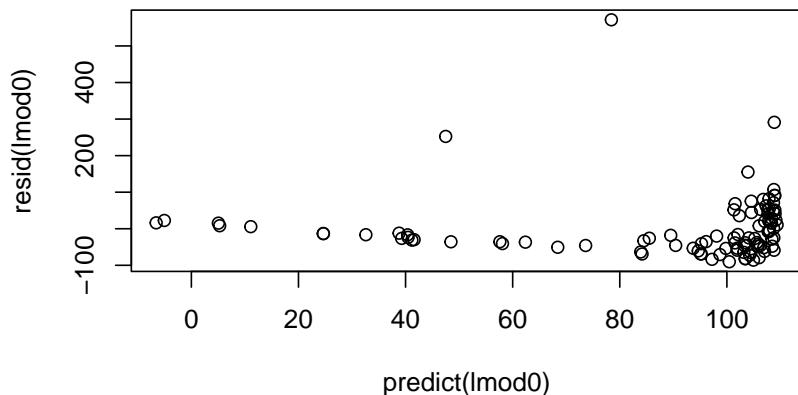
First, let's look at what would have happened if we fit the reference linear model to the un-transformed variables.

```
lmod0 = lm(infant ~ income, data=Leinhardt)
plot(resid(lmod0)) # to check independence (looks okay)
```



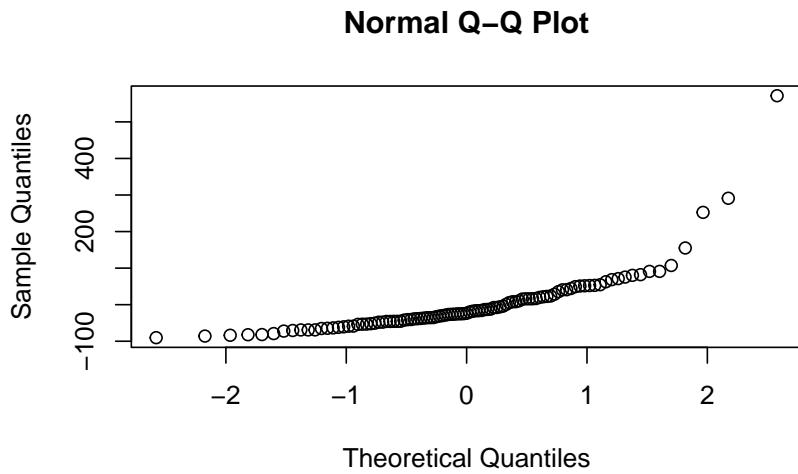
there should not be a pattern - but we can see an increase. This is not an issue and due to the data being presorted.

```
plot(predict(lmod0), resid(lmod0)) # to check for linearity, constant variance (looks bad)
```



after 80 the variance starts increasing

```
qqnorm(resid(lmod0)) # to check Normality assumption (we want this to be a straight line)
```



```
#?qqnorm
```

This looks good except for the last few points.

Now let's return to our model fit to the log-transformed variables. In a Bayesian model, we have distributions for residuals, but we'll simplify and look only at the residuals evaluated at the posterior mean of the parameters.

```
X = cbind(rep(1.0, data1_jags$n), data1_jags$log_income)
head(X)
```

```
[ ,1]      [ ,2]
[1, ]    1 8.139149
[2, ]    1 8.116716
[3, ]    1 8.115521
[4, ]    1 8.466110
[5, ]    1 8.522976
[6, ]    1 8.105308
```

```
(pm_params1 = colMeans(mod1_csim))
```

①

① posterior mean - using (var = expr) forces R to return the value of var

```
b[1]      b[2]      sig
7.1512398 -0.5127002  0.9707862
```

```

yhat1 = drop(X %*% pm_params1[1:2])          ①
resid1 = data1_jags$y - yhat1                  ②
plot(resid1)                                    ③

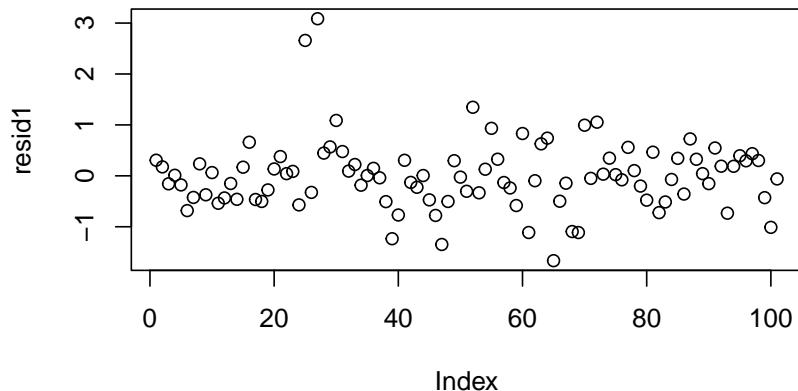
```

① we are evaluating

$$\hat{y}_i = b_0 \times 1 + b_1 \times x_{1,i} \text{ via matrix multiplication of } [1, \\ \text{data1_jags\$log_income}] * [b_0, b_1]$$

② $res_i = y_i - \hat{y}_i = y_i - (b_0 \times 1 + b_1 \times x_{1,i})$

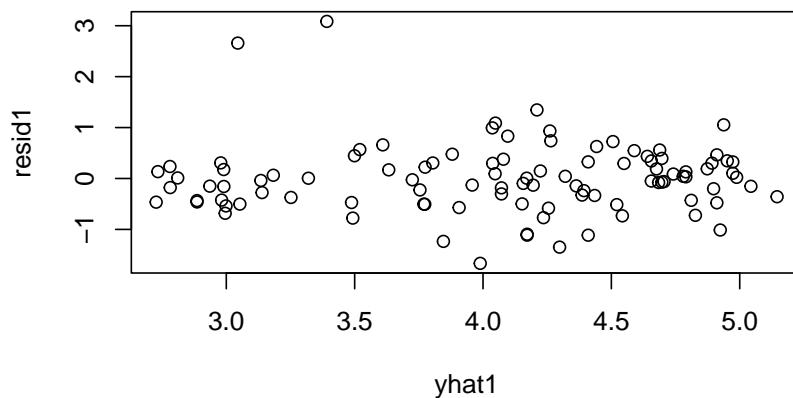
③ plots the residual against the data index



So to get the residuals from Stan we extract the b parameter.

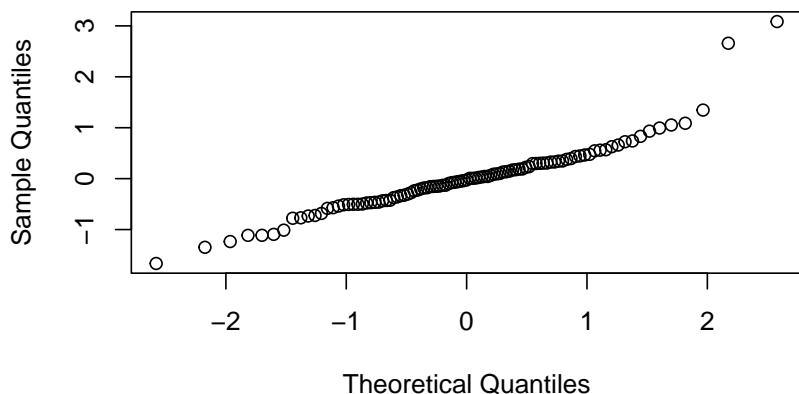
Although we did not discuss it we could estimate \hat{y} by drawing K predictions for each x_i and look at $res_i = \frac{1}{K} \sum_k |y_i - \hat{y}_{i,k}|$ and plot upper and lower bounds as well. Also I'm not sure but I guess we can also do with using the predictive posterior dist. Anyhow here is a link to something like this: [Extracting and visualizing tidy residuals from Bayesian models -jk](#)

```
plot(yhat1, resid1) # against predicted values
```

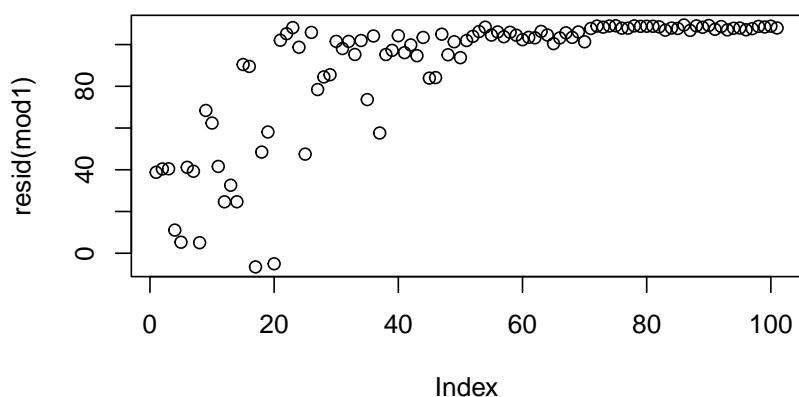


```
qqnorm(resid1) # checking normality of residuals
```

Normal Q–Q Plot



```
plot(predict(lmod0), resid(mod1)) # to compare with reference linear model
```



```
rownames(dat)[order(resid1, decreasing=TRUE)[1:5]] # which countries have the largest positive residuals?  
[1] "Saudi.Arabia" "Libya"           "Zambia"          "Brazil"          "Afganistan"
```

The residuals look pretty good here (no patterns, shapes) except for two strong outliers, Saudi Arabia and Libya. When outliers appear, it is a good idea to double check that they are not just errors in data entry. If the values are correct, you may reconsider whether these data points really are representative of the data you are trying to model. If you conclude that they are not (for example, they were recorded on different years), you may be able to justify dropping these data points from the data set.

If you conclude that the outliers are part of data and should not be removed, we have several modeling options to accommodate them. We will address these in the next segment.

34 Lesson 7.5 - Alternative models

In the previous segment, we saw two outliers in the model relating the logarithm of infant mortality to the logarithm of income. Here we will discuss options for when we conclude that these outliers belong in the data set.

34.1 Additional covariates

The first approach is to look for additional covariates that may be able to explain the outliers. For example, there could be a number of variables that provide information about infant mortality above and beyond what income provides.

Looking back at our data, there are two variables we haven't used yet: region and oil. The oil variable indicates oil-exporting countries. Both Saudi Arabia and Libya are oil-exporting countries, so perhaps this might explain part of the anomaly.

```
library("rjags")

mod2_string = " model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu[i], prec)
    mu[i] = b[1] + b[2]*log_income[i] + b[3]*is_oil[i] # <1>
  }

  for (i in 1:3) { # <2>
    b[i] ~ dnorm(0.0, 1.0/1.0e6)
  }

  prec ~ dgamma(5/2.0, 5*10.0/2.0)
  sig = sqrt( 1.0 / prec )
}
```

```

set.seed(73)
data2_jags = list(y=dat$loginfant, log_income=dat$logincome,
                   is_oil=as.numeric(dat$oil=="yes"))      ③
data2_jags$is_oil

params2 = c("b", "sig")

inits2 = function() {
  inits = list("b"=rnorm(3,0.0,100.0), "prec"=rgamma(1,1.0,1.0)) ④
}
mod2 = jags.model(textConnection(mod2_string), data=data2_jags, inits=inits2, n.chains=3)
update(mod2, 1e3) # burn-in

mod2_sim = coda.samples(model=mod2,
                       variable.names=params2,
                       n.iter=5e3)

mod2_csim = as.mcmc(do.call(rbind, mod2_sim)) # combine multiple chains

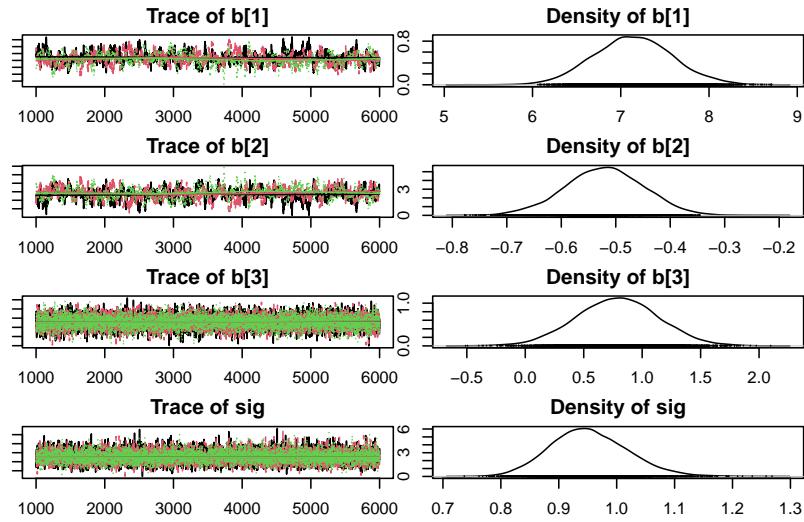
```

- ① we add the `is_oil` indicator parameter
 - ② we increment the number of parameters
 - ③ encode the `is_oil` from text to be binary
 - ④ draw another var for `b`.

Initializing model

As usual, check the convergence diagnostics.

```
par(mar = c(2., 1, 2., 1))
plot(mod2_sim)
```



```
gelman.diag(mod2_sim)
```

Potential scale reduction factors:

	Point est.	Upper C.I.
b[1]	1.03	1.11
b[2]	1.03	1.11
b[3]	1.00	1.00
sig	1.00	1.00

Multivariate psrf

1.03

```
autocorr.diag(mod2_sim)
```

	b[1]	b[2]	b[3]	sig
Lag 0	1.0000000	1.0000000	1.000000000	1.000000000
Lag 1	0.9530637	0.95363048	0.075454274	0.0281499793
Lag 5	0.7911621	0.79076112	0.001590028	0.0071942867
Lag 10	0.6257703	0.62426271	-0.004869111	0.0105098326
Lag 50	0.0743501	0.07466145	-0.017336823	0.0001011004

```
#autocorr.plot(mod2_sim,auto.layout=FALSE )  
autocorr.plot(mod2_csim,auto.layout=FALSE )
```

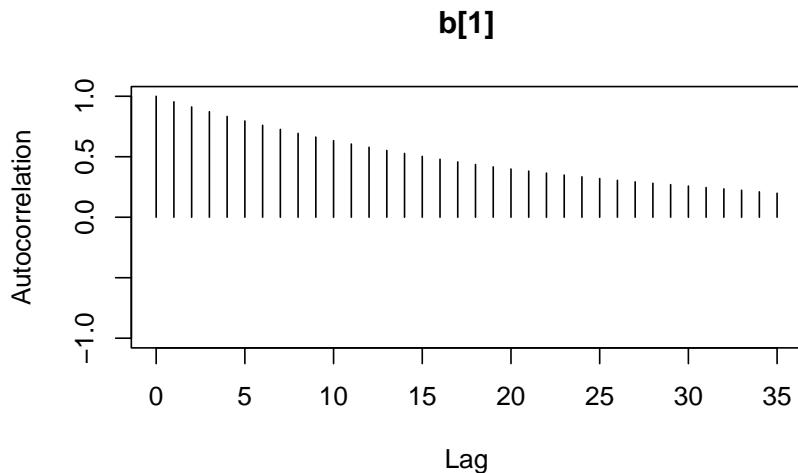


Figure 34.1: auto-correlation plot

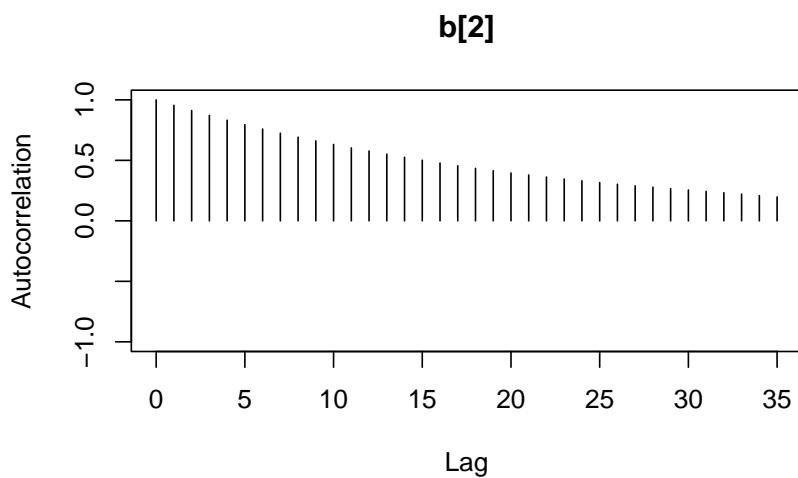


Figure 34.2: auto-correlation plot

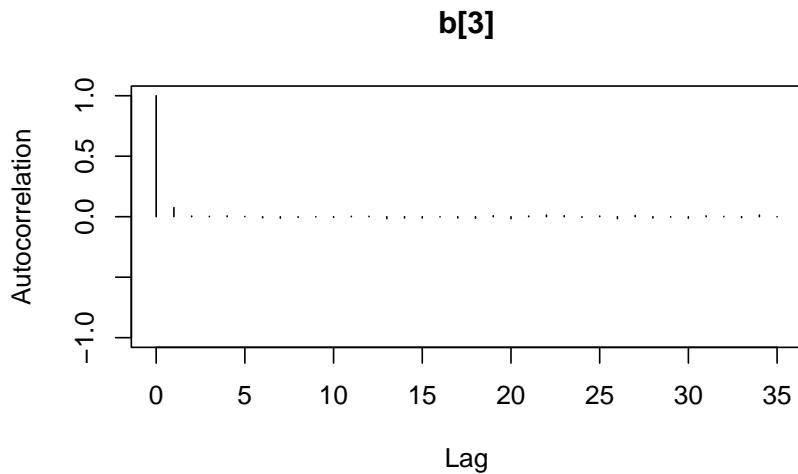


Figure 34.3: auto-correlation plot

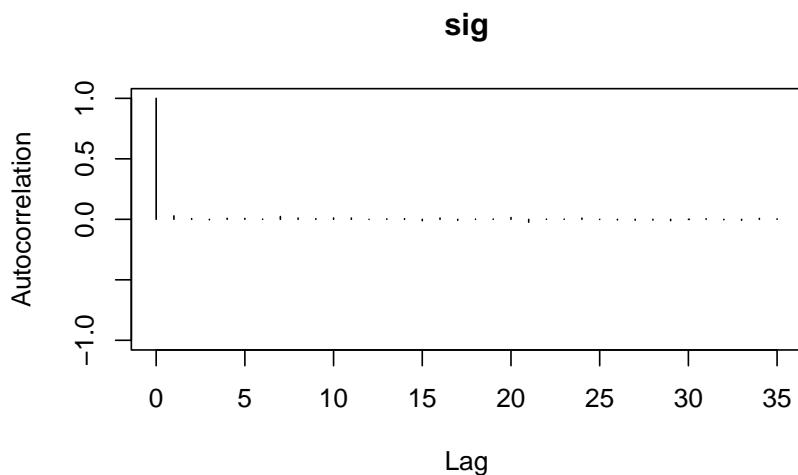


Figure 34.4: auto-correlation plot

```
effectiveSize(mod2_sim)
```

b[1]	b[2]	b[3]	sig
354.1865	350.6111	12895.8995	14140.0784

We can get a posterior summary of the parameters in our model.

```
summary(mod2_sim)
```

```
Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000
```

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

	Mean	SD	Naive SE	Time-series SE
b[1]	7.1391	0.44570	0.0036392	0.0235369
b[2]	-0.5210	0.07235	0.0005907	0.0038366
b[3]	0.7881	0.35310	0.0028831	0.0031110
sig	0.9528	0.06693	0.0005465	0.0005636

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
b[1]	6.27817	6.8375	7.1385	7.4363	8.0342
b[2]	-0.66596	-0.5694	-0.5208	-0.4721	-0.3814
b[3]	0.09832	0.5503	0.7918	1.0252	1.4749
sig	0.83033	0.9065	0.9492	0.9956	1.0940

It looks like there is a positive relationship between oil-production and log-infant mortality. Because these data are merely observational, we cannot say that oil-production causes an increase in infant mortality (indeed that most certainly isn't the case), but we can say that they are positively correlated.

Now let's check the residuals.

```
X2 = cbind(rep(1.0, data1_jags$n), data2_jags$log_income, data2_jags$is_oil)
head(X2)
```

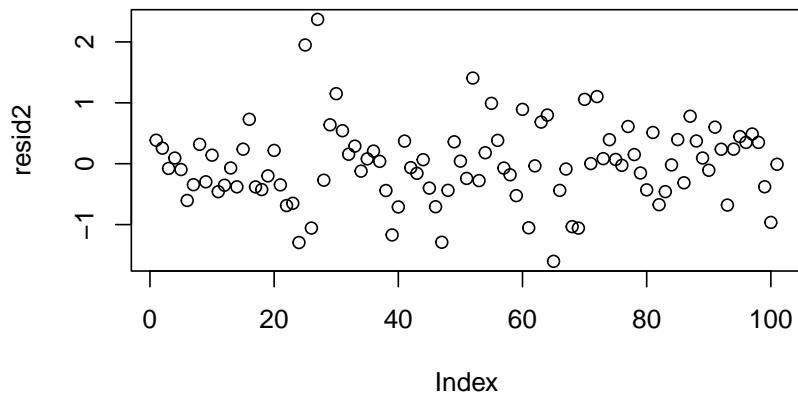
```
[,1]      [,2] [,3]
[1,]      1 8.139149    0
[2,]      1 8.116716    0
```

```
[3,] 1 8.115521 0
[4,] 1 8.466110 0
[5,] 1 8.522976 0
[6,] 1 8.105308 0
```

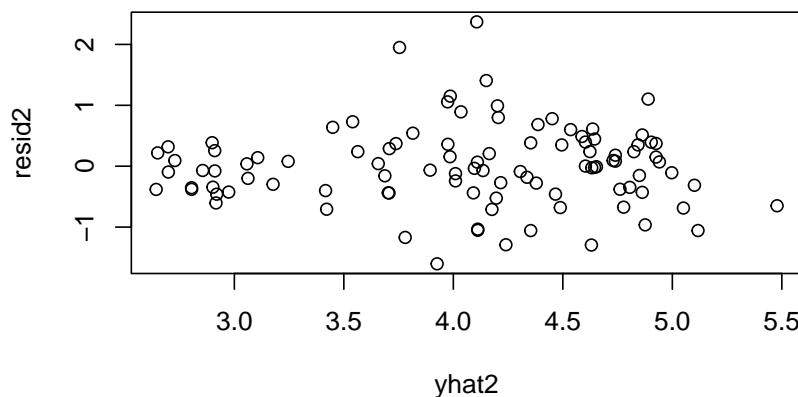
```
(pm_params2 = colMeans(mod2_csim)) # posterior mean
```

```
b[1]      b[2]      b[3]      sig
7.1390540 -0.5209887  0.7881187  0.9528432
```

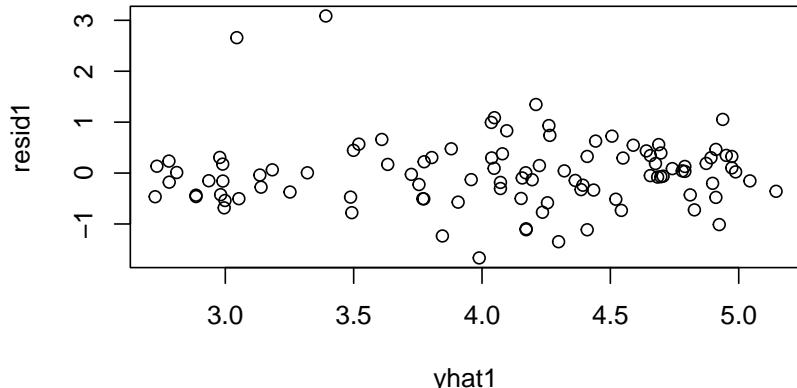
```
yhat2 = drop(X2 %*% pm_params2[1:3])
resid2 = data2_jags$y - yhat2
plot(resid2) # against data index
```



```
plot(yhat2, resid2) # against predicted values
```



```
plot(yhat1, resid1) # residuals from the first model
```



```
sd(resid2) # standard deviation of residuals
```

```
[1] 0.6488698
```

These look much better, although the residuals for Saudi Arabia and Libya are still more than three standard deviations away from the mean of the residuals. We might consider adding the other covariate `region`, but instead let's look at another option when we are faced with strong outliers.

34.1.1 Student-t likelihood

- Let's consider changing the likelihood.
- The normal likelihood has thin tails (almost all of the probability is concentrated within the first few standard deviations from the mean).
- This does not accommodate outliers well.
- Consequently, models with the normal likelihood might be overly-influenced by outliers.
- Recall that the t distribution is similar to the normal distribution, but it has thicker tails which can accommodate outliers.

The t linear model might look something like this. Notice that the t distribution has three parameters, including a positive “degrees of freedom” parameter. The smaller the degrees of freedom, the heavier the tails of the

distribution. We might fix the degrees of freedom to some number, or we can assign it a prior distribution.

```
curve(dnorm(x), from = -5, to = 5)
curve(dt(x,1), from = -5, to = 5,col="red", add = TRUE)
```

```
mod3_string = " model {
  for (i in 1:length(y)) { # <1>
    y[i] ~ dt( mu[i], tau, df )
    mu[i] = b[1] + b[2]*log_income[i] + b[3]*is_oil[i]
  }

  for (i in 1:3) {
    b[i] ~ dnorm(0.0, 1.0/1.0e6)
  }

  nu ~ dexp(1.0) # <2>
  df = nu + 2.0 # <3>

  tau ~ dgamma(5/2.0, 5*10.0/2.0) # <4>
  sig = sqrt( 1.0 / tau * df / (df - 2.0) ) # <5>
}"
```

- ① we replaced normal likelihood with a student t likelihood which has thicker tails
- ② ν nu is the degrees of freedom (dof) but the outcome can be 0 or 1
- ③ we force the degrees of freedom $\text{dof} > 2$ to guarantee the existence of mean and variance in the t dist.
- ④ τ tau is the **inverse scale** is close to, but not equal to the **precision** from above so we use the same prior as we used for precision.
- ⑤ σ sig standard deviation of errors is a deterministic function of tau, and df

We fit this model.

```
set.seed(73)
data3_jags = list(y=dat$loginfant, log_income=dat$logincome,
                  is_oil=as.numeric(dat$oil=="yes"))      ③

params3 = c("b", "sig")
```

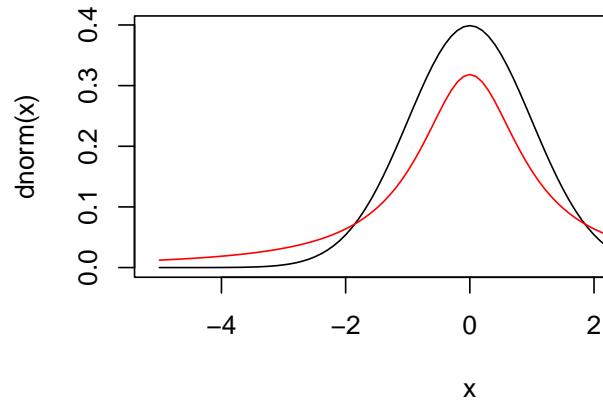


Figure 34.5: normal and t distribution

```

inits3 = function() {
  inits = list("b"=rnorm(3,0.0,100.0), "prec"=rgamma(1,1.0,1.0)) ④
}

mod3 = jags.model(textConnection(mod3_string), data=data3_jags, inits=inits3, n.chains=3)
update(mod3, 1e3) # burn-in

mod3_sim = coda.samples(model=mod3,
                       variable.names=params3,
                       n.iter=5e3)

mod3_csim = as.mcmc(do.call(rbind, mod3_sim)) # combine multiple chains

```

Compiling model graph
 Resolving undeclared variables
 Allocating nodes
 Graph information:
 Observed stochastic nodes: 101
 Unobserved stochastic nodes: 5
 Total graph size: 512

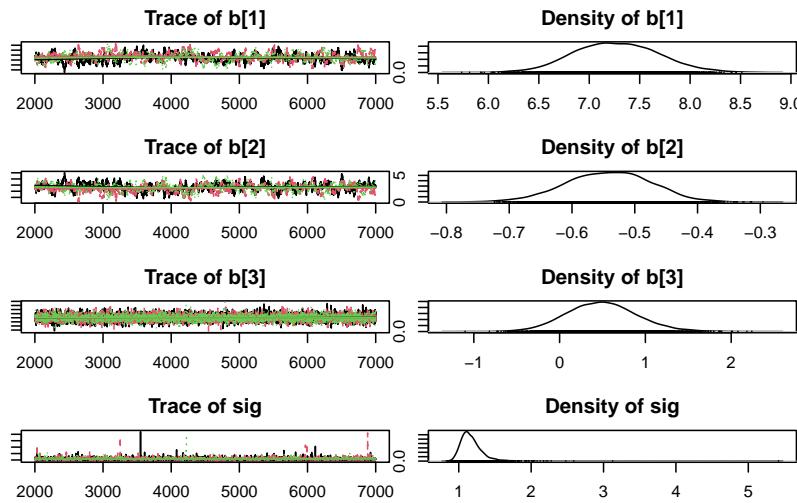
Initializing model

check MCMC convergence visually

```

par(mar = c(2.5, 1, 2.5, 1))
plot(mod3_sim)

```



check MCMC convergence quantitatively using *Rubin Gelman*

```
gelman.diag(mod3_sim)
```

Potential scale reduction factors:

	Point est.	Upper C.I.
$b[1]$	1.01	1.03
$b[2]$	1.01	1.03
$b[3]$	1.00	1.01
sig	1.00	1.00

Multivariate psrf

1.01

```
effectiveSize(mod3_sim)
```

	$b[1]$	$b[2]$	$b[3]$	sig
	271.4934	268.1449	8967.4686	8956.4634

34.2 Compare models using Deviance information criterion (DIC)

We have now proposed three different models. How do we compare their performance on our data? In the previous course, we discussed estimating parameters in models using the maximum likelihood method. Similarly, we can choose between competing models using the same idea.

We will use a quantity known as the deviance information criterion (DIC). It essentially calculates the posterior mean of the log-likelihood and adds a penalty for model complexity.

Let's calculate the DIC for our first two models:

the simple linear regression on log-income,

```
dic.samples(mod1, n.ite=1e3)
```

```
Mean deviance: 231.4  
penalty 2.898  
Penalized deviance: 234.3
```

and the second model where we add oil production.

```
dic.samples(mod2, n.ite=1e3)
```

```
Mean deviance: 225.1  
penalty 4.11  
Penalized deviance: 229.2
```

and the second model where we introduce the Student t likelihood.

```
dic.samples(mod3, n.ite=1e3)
```

```
Mean deviance: 231.1  
penalty 4.042  
Penalized deviance: 235.2
```

The first number is the Monte Carlo estimated posterior mean deviance, which equals -2 times the log-likelihood (plus a constant that will be irrelevant for comparing models). Because of that -2 factor, **a smaller deviance means a higher likelihood**.

Next, we are given a penalty for the complexity of our model. This penalty is necessary because we can always increase the likelihood of the model by making it more complex to fit the data exactly. We don't want to do this because over-fit models generalize poorly. This penalty is roughly equal to the effective number of parameters in your model. You can see this here. With the first model, we had a variance parameter and two betas, for a total of three parameters. In the second model, we added one more beta for the oil effect.

We add these two quantities to get the DIC (the last number). The better-fitting model has a lower DIC value. In this case, the gains we receive in deviance by adding the `is_oil` covariate outweigh the penalty for adding an extra parameter. The final DIC for the second model is lower than for the first, so we would prefer using the second model.

We encourage you to explore different model specifications and compare their fit to the data using DIC. [Wikipedia](#) provides a good introduction to DIC and we can find more details about the JAGS implementation through the `rjags` package documentation by entering `?dic.samples` in the R console.

```
#?dic.samples
```

34.3 Regression Diagnostics

In production we want to flag regression issues in an automated fashion. However while we develop models we should try to examine these issues visually.

Regression diagnostics help identify:

- shortcoming of our model and the preferred ways to improve them
 - transforms of variables
 - different likelihood
 - adding missing covariate relations to remove patterns in the residuals

- increasing interpretability by removing covariates that do not contribute to the fit.
- issues in the data
 - transformation

we should consider the following issues: 1. testing heteroscedasticity with the *Breusch-Pagan* test

Let's try to cover the diagnostic plots which help us validate a regression model.

34.3.1 Residuals vs Fitted

- The “residuals versus fits plot” is the most first diagnostic tool we should look at to determine if the regression is valid. If the regression assumptions are violated we should be able to identify the issues and possibly correct them.
- It is a scatter plot of residuals on the y axis and fitted values (estimated responses) on the x axis.
- The plot can be used to detect:
 - non-linearity,
 - unequal error variances, and
 - outliers.

```
plot(lmod0, 1)
```

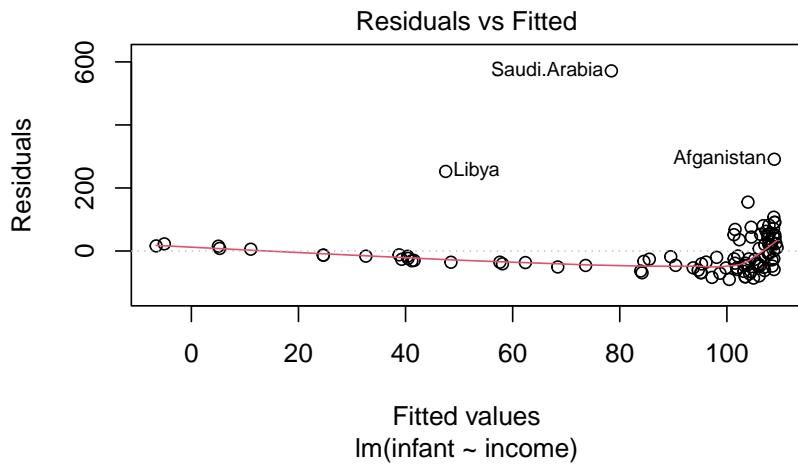


Figure 34.6: Residuals vs Fitted plot

Residuals will enable us to assess visually whether an appropriate model has been fit to the data no matter how many predictor variables are used. We can check the validity of a linear regression model by plotting residuals versus x and look for patterns. - Lack of a discernible pattern is indicative of a valid model. - A pattern is indicative that a function or transformation of X is missing from the model.

! What to look for

Look for *patterns* that can indicate non-linearity,

- that the residuals all are high in some areas and low in others. Change in variability as X changes - U shape missing quadratic term • we can get this plot as follows.

The blue line is there to aid the eye - it should ideally be relatively close to a straight line (in this case, it isn't perfectly straight, which could indicate a mild non-linearity).

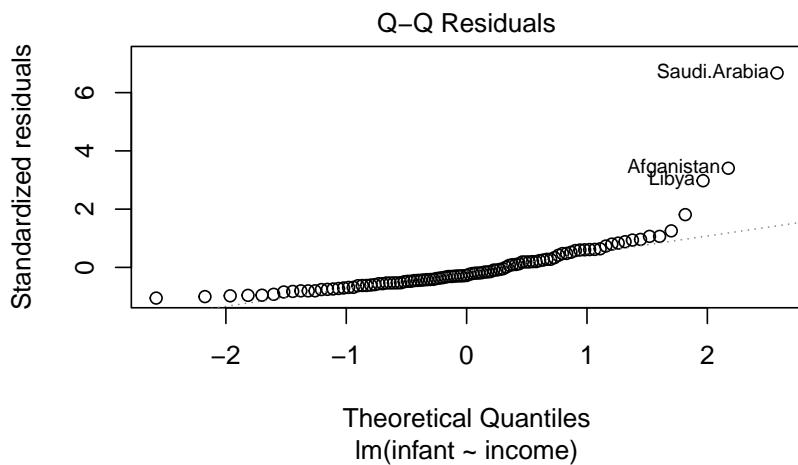
34.3.2 QQ plot of the residuals

This plot shows if residuals are normally distributed. Do residuals follow a straight line well or do they deviate severely?

The regression is valid if the residuals are lined well on the straight dashed line.

we can get this plot as follows

```
plot(lmod0, 2)
```



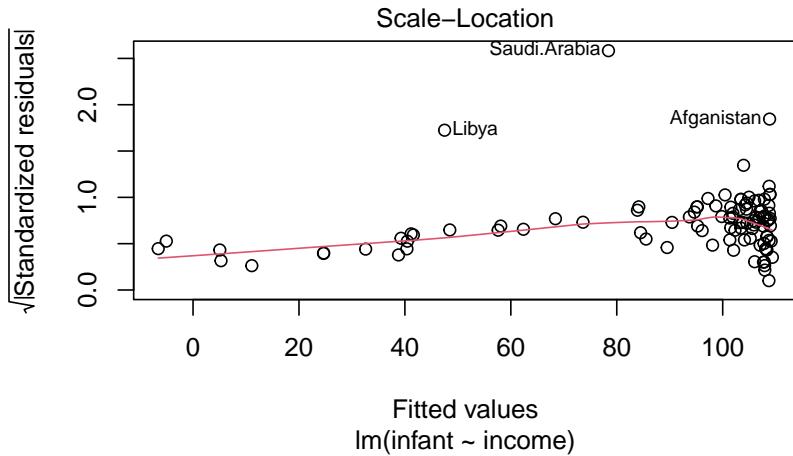
notice that the two outliers are labeled and should be reviewed for - removal
- more robust likelihood

for more info see [understanding QQ plots](#)

34.3.3 Scale Location plot

This plot shows if residuals are spread equally along the ranges of predictors. This is how you can check the assumption of equal variance (homoscedasticity). It's good if you see a horizontal line with equally (randomly) spread points.

```
plot(lmod0, 3)
```

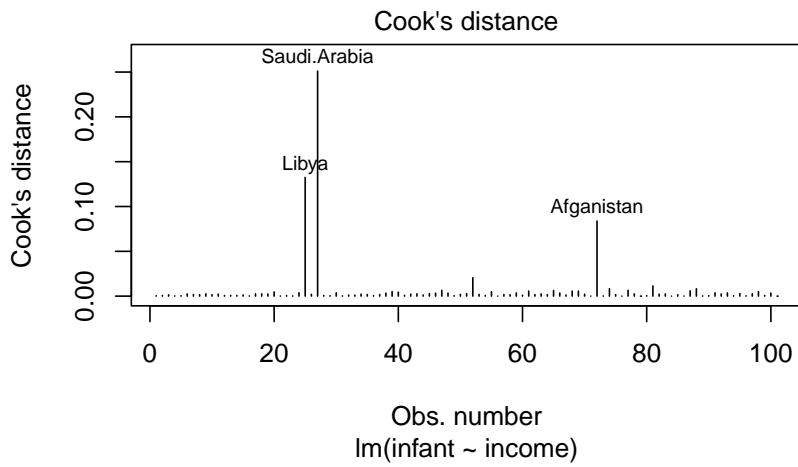


in this case: - most of the points are to the right - the red line is almost flat which is good - there is increasing variance after 80

34.3.4 Cook's Distance

- Originally introduced in (Cook 1977a) [Cook's Distance](#) is an estimate of the influence of a data point.
- It takes into account both the leverage and residual of each observation.
- Cook's Distance is a summary of how much a regression model changes when the i th observation is removed.
- When it comes to outliers we care about outliers that have a high Cook's distance as they can have a large impact on the regression model. by shifting the sample fit from the population fit.
- Another aspect of Cook's distance is it can be used to identify regions of the design space where the model is poorly supported by the data - i.e. where the model is extrapolating and if we can get more data in that region we can improve the model.

```
plot(lmod0, 4)
```

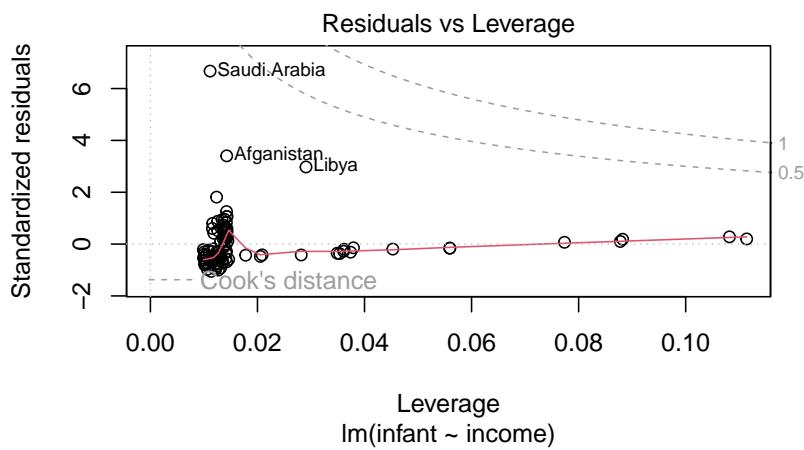


Used to detect highly influential outliers, i.e. points that can shift the sample fit from the population fit. For large sample sizes, a rough guideline is to consider values above $4/(n - p)$, where n is the sample size and p is the number of predictors including the intercept, to indicate highly influential points.

see Williams (1987)

34.3.5 Residuals vs Leverage

```
plot(lmod0, 5)
```



```
#plot(mod3, 5)
```

This plot helps us to sort through the outliers, if there are any. Not all outliers are influential in linear regression analysis. Even though data have extreme values, they might not be influential to determine a regression line. i.e. the fit wouldn't be much different if we choose to omit them from the analysis. If a point is able to exert a influence on the regression line we call it a high leverage point. Even in this case it might not alter the trend. So we want to identify high leverage points that are at a large distance from their predictor's mean.

Unlike the other plots, this time patterns are not relevant. We watch out for outlying values at the upper right corner or at the lower right corner. Those spots are the places where cases can be influential against a regression line. Look for cases outside of the dashed lines. When cases are outside of the dashed lines (meaning they have high "Cook's distance" scores), the cases are influential to the regression results. The regression results will be altered if we exclude those cases.

In this case we see that the pints are within the cook's distance contours so our outliers are not high leverage points.

34.3.6 Cook's Distance vs Leverage

```
plot(lmod0, 6)
```

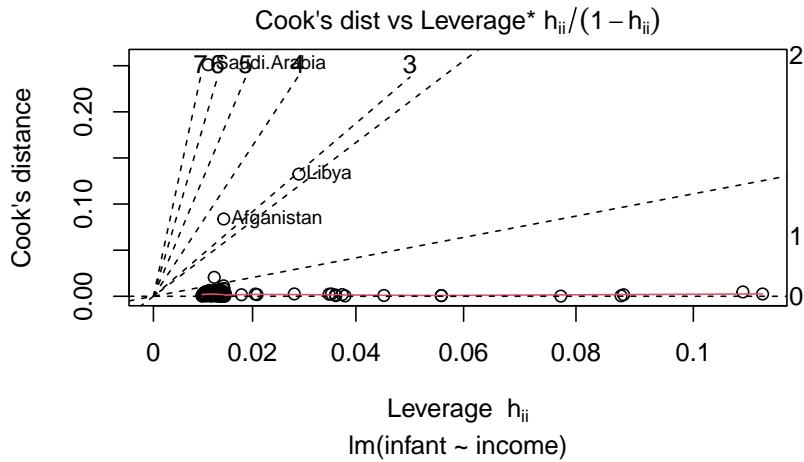


Figure 34.7: Cooks distance v.s. Leverage

Cook's distance and leverage are used to detect highly leverage points, i.e. data points that can shift the sample fit from the population fit.

For large sample sizes, a rough guideline is to consider Cook's distance values above 1 to indicate highly influential points and leverage values greater than 2 times the number of predictors divided by the sample size to indicate high leverage observations. High leverage observations are ones which have predictor values very far from their averages, which can greatly influence the fitted model.

The contours in the scatterplot are standardized residuals labelled with their magnitudes

see Williams (1987)

34.3.7 Python

- <https://emredjan.xyz/blog/2017/07/11/emulating-r-plots-in-python/>
- <https://towardsdatascience.com/going-from-r-to-python-linear-regression-diagnostic-plots-144d1c4aa5a>
- <https://modernstatisticswithr.com/regression.html>

35 ANOVA

Bayesian Statistics: Techniques and Models

35.1 Introduction to ANOVA

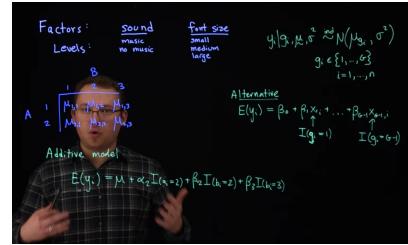


Figure 35.1: Introduction to ANOVA

35.2 One way ANOVA model using JAGS

35.2.1 Data & EDA

As an example of a one-way ANOVA, we'll look at the Plant Growth data in R.

Listing 35.1 Plant Growth Query

```
data("PlantGrowth")
#?PlantGrowth
head(PlantGrowth)
```

```

weight group
1   4.17  ctrl
2   5.58  ctrl
3   5.18  ctrl
4   6.11  ctrl
5   4.50  ctrl
6   4.61  ctrl

```

We first load the dataset (Listing 35.1)

Because the explanatory variable `group` is a factor and not continuous, we choose to visualize the data with box plots rather than scatter plots.

```
boxplot(weight ~ group, data=PlantGrowth)
```

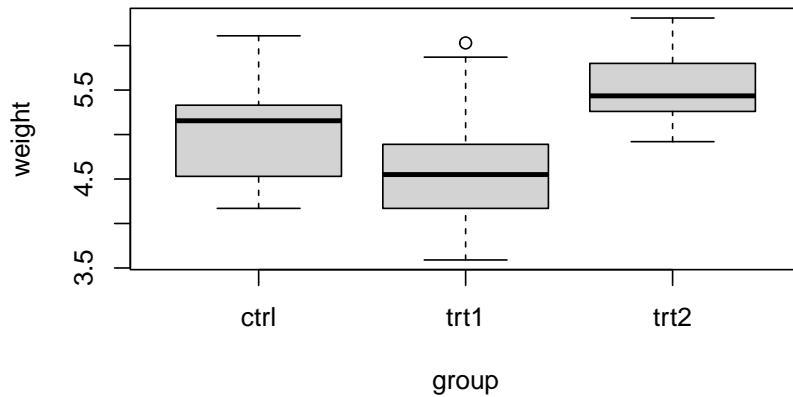


Figure 35.2: PlantGrowth boxplot

The box plots summarize the distribution of the data for each of the three groups. It appears that treatment 2 has the highest mean yield. It might be questionable whether each group has the same variance, but we'll assume that is the case.

35.2.2 Modeling

Again, we can start with the reference analysis (with a noninformative prior) with a linear model in R.

```

lmod = lm(weight ~ group, data=PlantGrowth)
summary(lmod)

Call:
lm(formula = weight ~ group, data = PlantGrowth)

Residuals:
    Min      1Q  Median      3Q     Max 
-1.0710 -0.4180 -0.0060  0.2627  1.3690 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  5.0320    0.1971 25.527 <2e-16 ***
grouptrt1   -0.3710    0.2788 -1.331  0.1944    
grouptrt2    0.4940    0.2788  1.772  0.0877 .  
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6234 on 27 degrees of freedom
Multiple R-squared:  0.2641,    Adjusted R-squared:  0.2096 
F-statistic: 4.846 on 2 and 27 DF,  p-value: 0.01591

anova(lmod)

Analysis of Variance Table

Response: weight
          Df  Sum Sq Mean Sq F value Pr(>F)    
group      2  3.7663  1.8832  4.8461 0.01591 *  
Residuals 27 10.4921  0.3886                
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

plot(lmod) # for graphical residual analysis

```

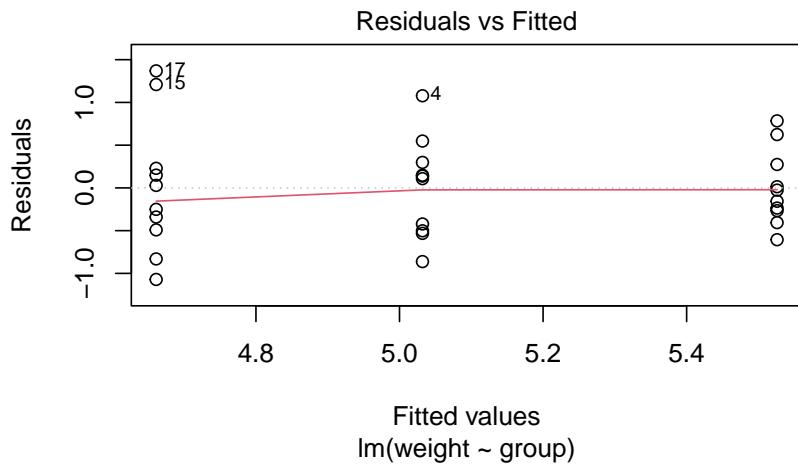


Figure 35.3: Graphical residual analysis

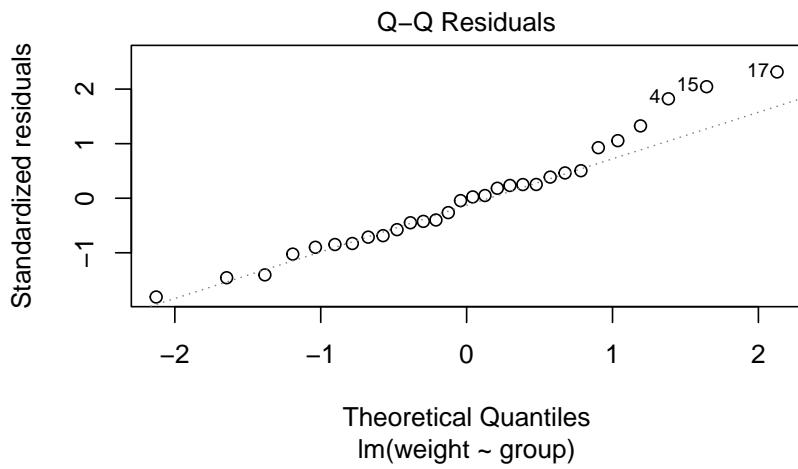


Figure 35.4: Graphical residual analysis

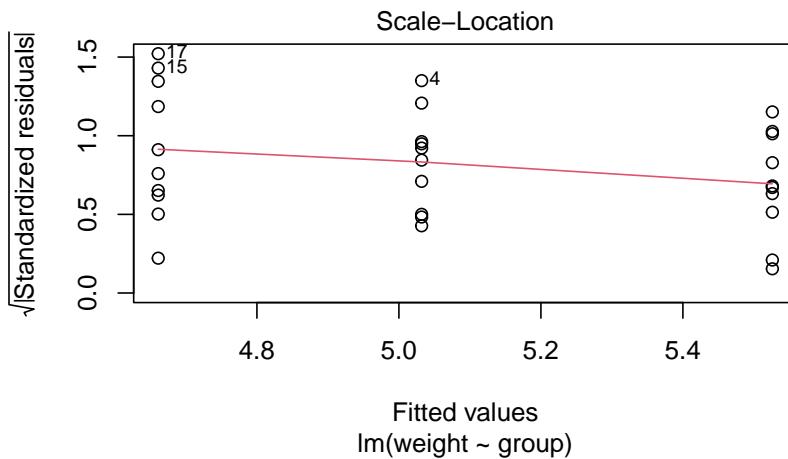


Figure 35.5: Graphical residual analysis

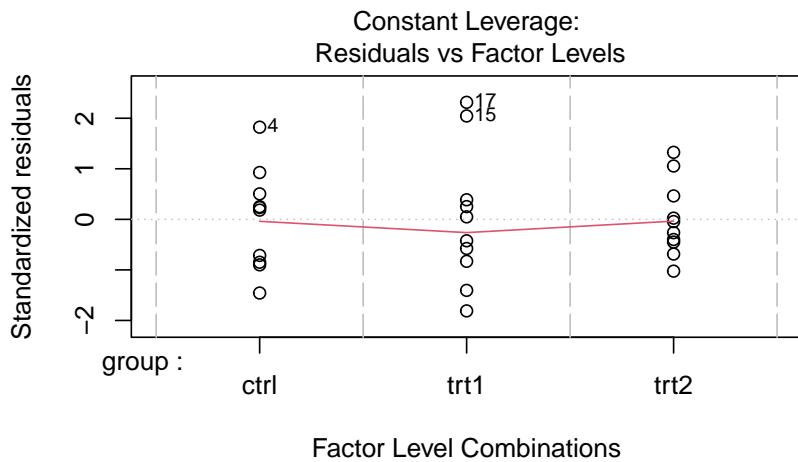


Figure 35.6: Graphical residual analysis

The default model structure in R is the linear model with dummy indicator variables. Hence, the “intercept” in this model is the mean yield for the control group. The two other parameters are the estimated effects of treatments 1 and 2. To recover the mean yield in treatment group 1, you would add the intercept term and the treatment 1 effect. To see how R sets the model up, use the `model.matrix(lmod)` function to extract the X matrix.

The `anova()` function in R compares variability of observations between the treatment groups to variability within the treatment groups to test

whether all means are equal or whether at least one is different. The small p-value here suggests that the means are not all equal.

Let's fit the **cell means** model in JAGS.

```
library("rjags")

mod_string = " model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu[grp[i]], prec)
  }

  for (j in 1:3) {
    mu[j] ~ dnorm(0.0, 1.0/1.0e6)
  }

  prec ~ dgamma(5/2.0, 5*1.0/2.0)
  sig = sqrt( 1.0 / prec )
} "

set.seed(82)
str(PlantGrowth)

'data.frame':   30 obs. of  2 variables:
$ weight: num  4.17 5.58 5.18 6.11 4.5 4.61 5.17 4.53 5.33 5.14 ...
$ group : Factor w/ 3 levels "ctrl","trt1",...: 1 1 1 1 1 1 1 1 1 1 ...

data_jags = list(y=PlantGrowth$weight,
                 grp=as.numeric(PlantGrowth$group))

params = c("mu", "sig")

inits = function() {
  inits = list("mu"=rnorm(3,0.0,100.0), "prec"=rgamma(1,1.0,1.0))
}

mod = jags.model(textConnection(mod_string), data=data_jags, inits=inits, n.chains=3)

Compiling model graph
Resolving undeclared variables
```

```

    Allocating nodes
Graph information:
  Observed stochastic nodes: 30
  Unobserved stochastic nodes: 4
  Total graph size: 74

Initializing model

update(mod, 1e3)

mod_sim = coda.samples(model=mod,
                       variable.names=params,
                       n.iter=5e3)
mod_csim = as.mcmc(do.call(rbind, mod_sim)) # combined chains

```

35.2.3 Model checking

As usual, we check for convergence of our MCMC.

```

par(mar = c(2.5, 1, 2.5, 1))
plot(mod_sim)

gelman.diag(mod_sim)

```

Potential scale reduction factors:

	Point est.	Upper C.I.
mu[1]	1	1
mu[2]	1	1
mu[3]	1	1
sig	1	1

Multivariate psrf

1

```
autocorr.diag(mod_sim)
```

	mu[1]	mu[2]	mu[3]	sig
Lag 0	1.000000000	1.000000000	1.000000000	1.000000000
Lag 1	-0.010270506	-0.005017784	-0.011756653	0.087639192
Lag 5	0.004086335	0.008649091	0.001642567	0.009254555
Lag 10	0.017815477	-0.008984731	-0.003491620	0.019816240
Lag 50	-0.010240810	-0.017259546	-0.012696453	0.002390879

`effectiveSize(mod_sim)`

	mu[1]	mu[2]	mu[3]	sig
	15000.00	14735.99	15204.18	12259.47

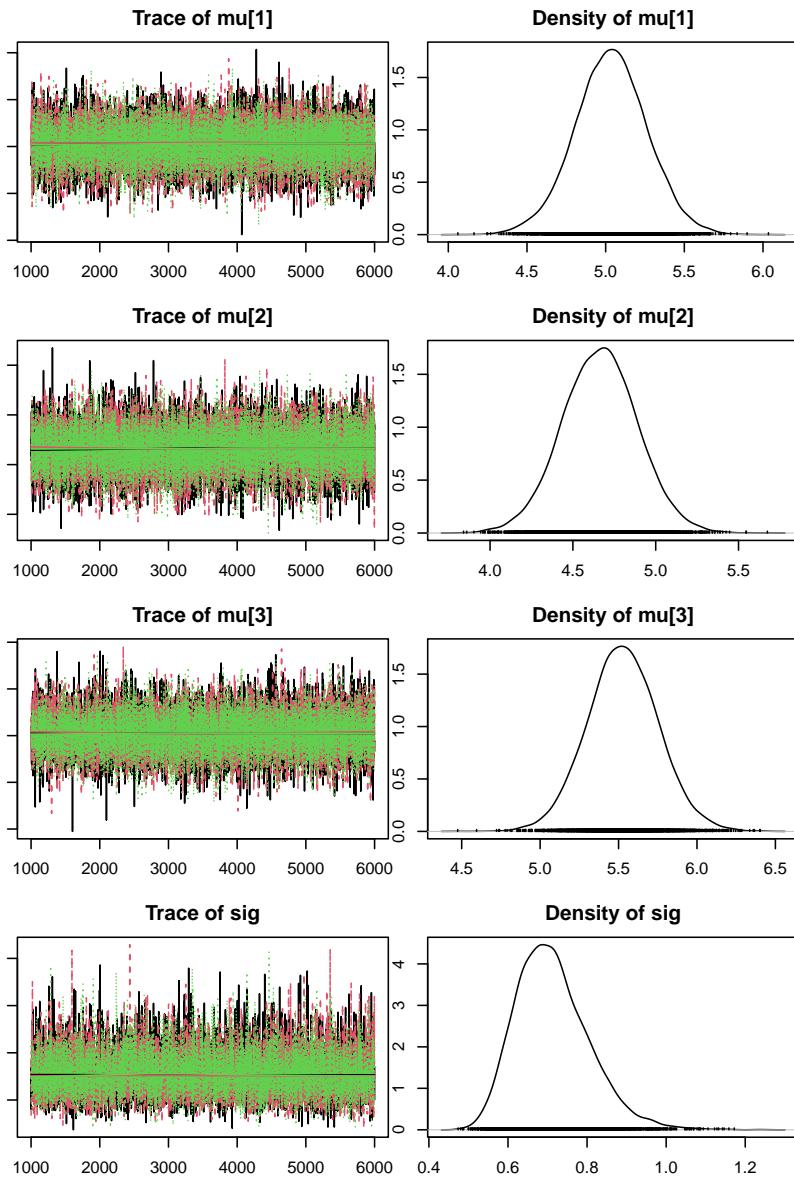


Figure 35.7: MCMC convergence diagnostics

We can also look at the residuals to see if there are any obvious problems with our model choice.

```
(pm_params = colMeans(mod_csim))
```

```
mu[1]      mu[2]      mu[3]      sig
5.0338191 4.6630008 5.5277721 0.7127586
```

```
yhat = pm_params[1:3][data_jags$grp]
resid = data_jags$y - yhat
plot(resid)
```

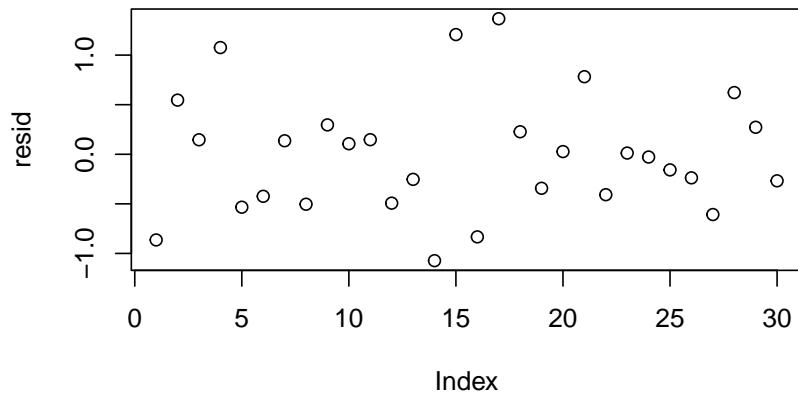


Figure 35.8: Residuals vs Index

```
plot(yhat, resid)
```

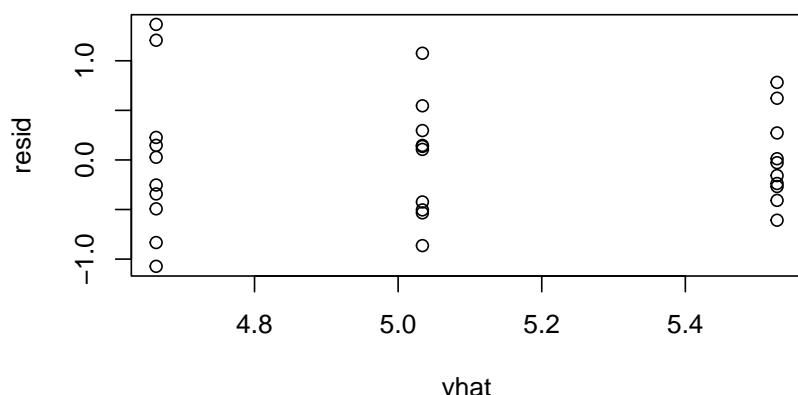


Figure 35.9: Residuals vs Fitted values for PlantGrowth model

Again, it might be appropriate to have a separate variance for each group.
We will have you do that as an exercise.

35.2.4 Results

Let's look at the posterior summary of the parameters.

```
summary(mod_sim)
```

```
Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000
```

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

	Mean	SD	Naive SE	Time-series SE
mu[1]	5.0338	0.22961	0.0018747	0.0018747
mu[2]	4.6630	0.22782	0.0018602	0.0018770
mu[3]	5.5278	0.22693	0.0018529	0.0018414
sig	0.7128	0.09298	0.0007592	0.0008399

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
mu[1]	4.5704	4.8845	5.0352	5.1855	5.4887
mu[2]	4.2136	4.5101	4.6645	4.8141	5.1135
mu[3]	5.0857	5.3780	5.5257	5.6773	5.9824
sig	0.5598	0.6466	0.7037	0.7682	0.9215

```
HPDinterval(mod_csim)
```

	lower	upper
mu[1]	4.5933448	5.5063953
mu[2]	4.2216268	5.1194646
mu[3]	5.0678031	5.9625059
sig	0.5425773	0.8937175
attr(,"Probability")		
[1]	0.95	

The `HPDinterval()` function in the `coda` package calculates intervals of highest posterior density for each parameter.

We are interested to know if one of the treatments increases mean yield. It is clear that treatment 1 does not. What about treatment 2?

```
mean(mod_csim[,3] > mod_csim[,1])
```

```
[1] 0.9375333
```

There is a high posterior probability that the mean yield for treatment 2 is greater than the mean yield for the control group.

It may be the case that treatment 2 would be costly to put into production. Suppose that to be worthwhile, this treatment must increase mean yield by 10%. What is the posterior probability that the increase is at least that?

```
mean(mod_csim[,3] > 1.1*mod_csim[,1])
```

```
[1] 0.4829333
```

We have about 50/50 odds that adopting treatment 2 would increase mean yield by at least 10%.

35.3 Two Factor ANOVA

35.3.1 Data

Let's explore an example with two factors. We'll use the `Warpbreaks` data set in R. Check the documentation for a description of the data by typing `?warpbreaks`.

```
data("warpbreaks")
#?warpbreaks
head(warpbreaks)
```

Table 35.1: Preview of first few rows of warpbreaks data

```
# This chunk is for displaying the output that was previously static.  
# If the static output below is preferred, this chunk can be removed  
# and the static output remains unlabelled as it's not a code cell.  
# For a labeled table, this chunk should generate it.  
# The original file had static output here:  
##   breaks wool tension  
## 1    26    A     L  
## 2    30    A     L  
## 3    54    A     L  
## 4    25    A     L  
## 5    70    A     L  
## 6    52    A     L  
# To make this a labeled table from code:  
head(warpbreaks)
```

```
  breaks wool tension  
1    26    A     L  
2    30    A     L  
3    54    A     L  
4    25    A     L  
5    70    A     L  
6    52    A     L
```

```
  breaks wool tension  
1    26    A     L  
2    30    A     L  
3    54    A     L  
4    25    A     L  
5    70    A     L  
6    52    A     L
```

Again, we visualize the data with box plots.

```
boxplot(breaks ~ wool + tension, data=warpbreaks)
```

Table 35.2: Contingency table of wool type vs tension level

```
table(warpbreaks$wool, warpbreaks$tension)
```

	L	M	H
A	9	9	9
B	9	9	9

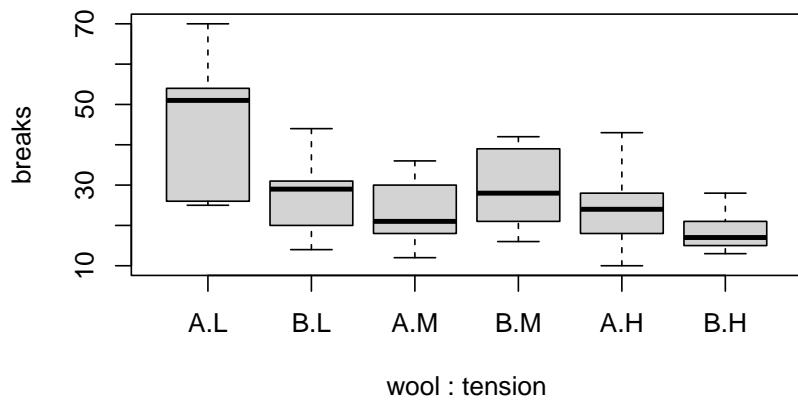


Figure 35.10: Warpbreaks boxplot

```
boxplot(log(breaks) ~ wool + tension, data=warpbreaks)
```

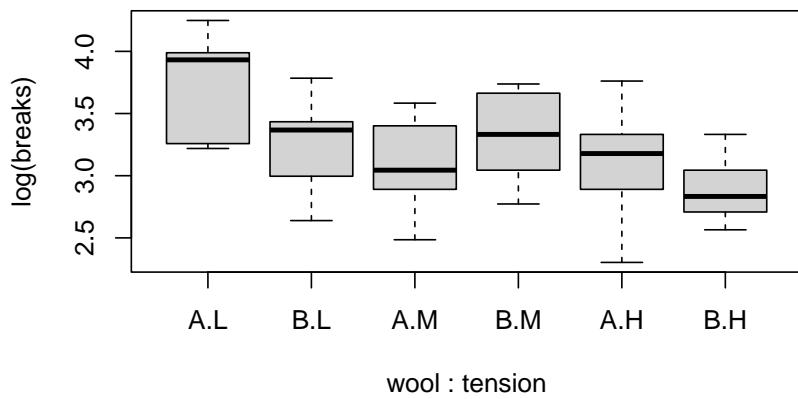


Figure 35.11: Warpbreaks boxplot with log-transformed breaks

The different groups have more similar variance if we use the logarithm of breaks. From this visualization, it looks like both factors may play a role in the number of breaks. It appears that there is a general decrease in breaks as we move from low to medium to high tension. Let's start with a one-way model using tension only.

35.3.2 One-way model

```
mod1_string = " model {
  for( i in 1:length(y)) {
    y[i] ~ dnorm(mu[tensGrp[i]], prec)
  }

  for (j in 1:3) {
    mu[j] ~ dnorm(0.0, 1.0/1.0e6)
  }

  prec ~ dgamma(5/2.0, 5*2.0/2.0)
  sig = sqrt(1.0 / prec)
} "

set.seed(83)
str(warpbreaks)

'data.frame':   54 obs. of  3 variables:
$ breaks : num  26 30 54 25 70 52 51 26 67 18 ...
$ wool   : Factor w/ 2 levels "A","B": 1 1 1 1 1 1 1 1 1 1 ...
$ tension: Factor w/ 3 levels "L","M","H": 1 1 1 1 1 1 1 1 2 ...

data1_jags = list(y=log(warpbreaks$breaks), tensGrp=as.numeric(warpbreaks$tension))

params1 = c("mu", "sig")

mod1 = jags.model(textConnection(mod1_string), data=data1_jags, n.chains=3)

Compiling model graph
Resolving undeclared variables
Allocating nodes
```

```
Graph information:  
  Observed stochastic nodes: 54  
  Unobserved stochastic nodes: 4  
  Total graph size: 123  
  
Initializing model  
  
update(mod1, 1e3)  
  
mod1_sim = coda.samples(model=mod1,  
                        variable.names=params1,  
                        n.iter=5e3)  
  
## convergence diagnostics  
plot(mod1_sim)
```

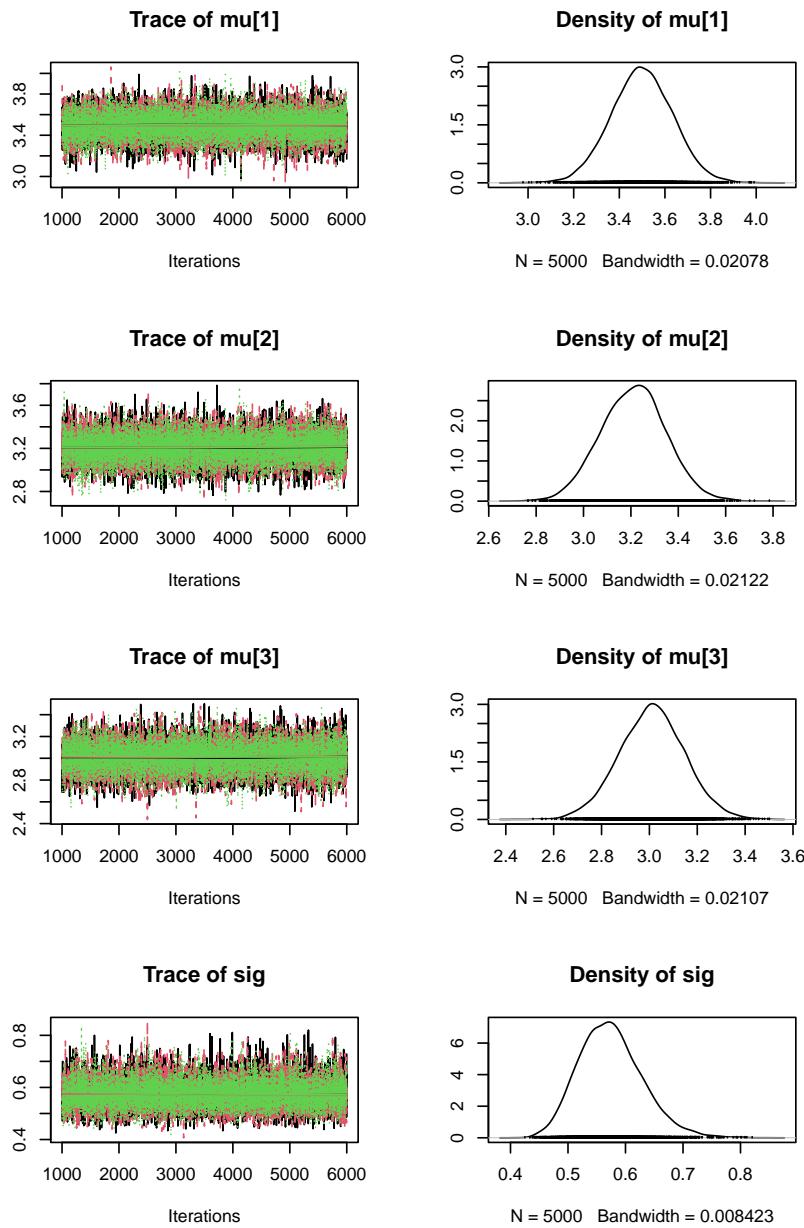


Figure 35.12: MCMC convergence diagnostics for one-way tension model

```
gelman.diag(mod1_sim)
```

Potential scale reduction factors:

```

    Point est. Upper C.I.
mu[1]      1      1
mu[2]      1      1
mu[3]      1      1
sig       1      1

Multivariate psrf

1

autocorr.diag(mod1_sim)

          mu[1]      mu[2]      mu[3]      sig
Lag 0  1.000000000  1.000000000  1.000000000  1.000000000
Lag 1  0.001508516  0.0008293363 0.004948756  0.0570965673
Lag 5  0.010417031  0.0136190542 0.002609665 -0.0018439677
Lag 10 0.005057726 -0.0041693647 -0.002379006 -0.0007024495
Lag 50 0.003828182  0.0009045658 -0.005989350  0.0026076224

effectiveSize(mod1_sim)

          mu[1]      mu[2]      mu[3]      sig
14547.20 14374.73 14801.56 13379.43

```

The 95% posterior interval for the mean of group 2 (medium tension) overlaps with both the low and high groups, but the intervals for low and high group only slightly overlap. That is a pretty strong indication that the means for low and high tension are different. Let's collect the DIC for this model and move on to the two-way model.

```
dic1 = dic.samples(mod1, n.iter=1e3)
```

35.3.3 Two-way additive model

With two factors, one with two levels and the other with three, we have six treatment groups, which is the same situation we discussed when introducing multiple factor ANOVA. We will first fit the additive model which

Table 35.3: Head of the design matrix for the additive model

```
X = model.matrix(~ wool + tension, data=warpbreaks)
head(X)
```

	(Intercept)	woolB	tensionM	tensionH
1	1	0	0	0
2	1	0	0	0
3	1	0	0	0
4	1	0	0	0
5	1	0	0	0
6	1	0	0	0

Table 35.4: Tail of the design matrix for the additive model

```
tail(X)
```

	(Intercept)	woolB	tensionM	tensionH
49	1	1	0	1
50	1	1	0	1
51	1	1	0	1
52	1	1	0	1
53	1	1	0	1
54	1	1	0	1

treats the two factors separately with no interaction. To get the X matrix (or design matrix) for this model, we can create it in R.

By default, R has chosen the mean for wool A and low tension to be the intercept. Then, there is an effect for wool B, and effects for medium tension and high tension, each associated with dummy indicator variables.

```
mod2_string = " model {
  for( i in 1:length(y)) {
    y[i] ~ dnorm(mu[i], prec)
    mu[i] = int + alpha*isWoolB[i] + beta[1]*isTensionM[i] + beta[2]*isTensionH[i]
  }

  int ~ dnorm(0.0, 1.0/1.0e6)
  alpha ~ dnorm(0.0, 1.0/1.0e6)
  for (j in 1:2) {
    beta[j] ~ dnorm(0.0, 1.0/1.0e6)
  }

  prec ~ dgamma(3/2.0, 3*1.0/2.0)
  sig = sqrt(1.0 / prec)
}

data2_jags = list(y=log(warpbreaks$breaks), isWoolB=X[, "woolB"], isTensionM=X[, "tensionM"], isTensionH=X[, "te

params2 = c("int", "alpha", "beta", "sig")

mod2 = jags.model(textConnection(mod2_string), data=data2_jags, n.chains=3)

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
  Observed stochastic nodes: 54
  Unobserved stochastic nodes: 5
  Total graph size: 243

Initializing model
```

```

update(mod2, 1e3)

mod2_sim = coda.samples(model=mod2,
                       variable.names=params2,
                       n.iter=5e3)

## convergence diagnostics
plot(mod2_sim)

gelman.diag(mod2_sim)      # Corrected from mod1_sim

```

Potential scale reduction factors:

	Point est.	Upper C.I.
alpha	1	1
beta[1]	1	1
beta[2]	1	1
int	1	1
sig	1	1

Multivariate psrf

1

```
autocorr.diag(mod2_sim) # Corrected from mod1_sim
```

	alpha	beta[1]	beta[2]	int	sig
Lag 0	1.000000000	1.000000000	1.000000000	1.000000000	1.000000000
Lag 1	0.4976282979	0.5022011526	0.49360025	0.74434697	0.071368512
Lag 5	0.0256529340	0.1101414630	0.09088576	0.17008957	0.010523031
Lag 10	0.0049814426	0.0172071037	0.01148580	0.02033475	-0.008321242
Lag 50	0.0003516398	0.0005358724	0.00589442	-0.00723292	0.007493060

```
effectiveSize(mod2_sim) # Corrected from mod1_sim
```

	alpha	beta[1]	beta[2]	int	sig
	5117.643	3833.527	3985.188	2482.042	12592.094

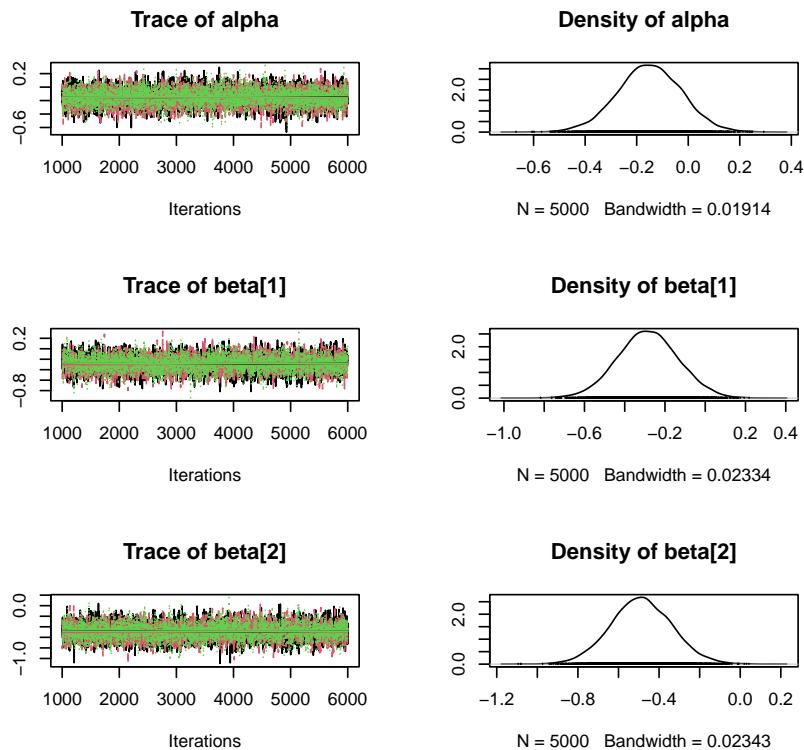


Figure 35.13: Convergence and diagnostics for the additive two-way ANOVA model

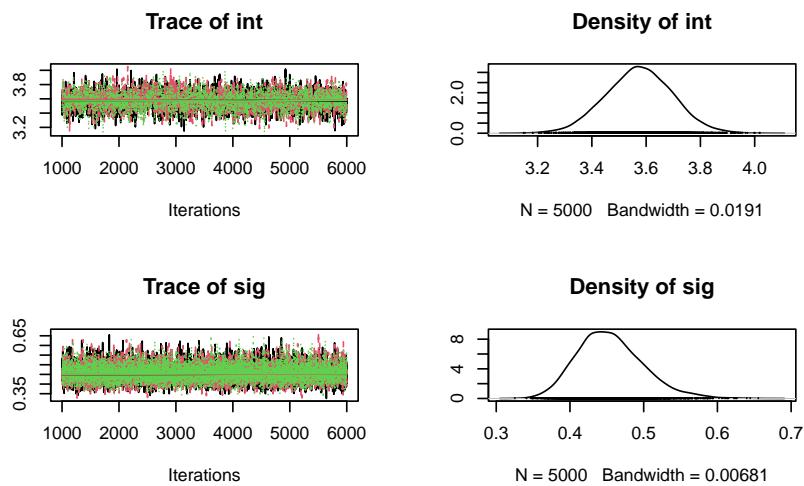


Figure 35.14: Convergence and diagnostics for the additive two-way ANOVA model

Let's summarize the results, collect the DIC for this model, and compare it to the first one-way model.

```
summary(mod2_sim)
```

```
Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000
```

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

	Mean	SD	Naive SE	Time-series SE
alpha	-0.1503	0.12420	0.0010141	0.0017366
beta[1]	-0.2852	0.15213	0.0012421	0.0024574
beta[2]	-0.4879	0.15132	0.0012356	0.0023986
int	3.5739	0.12327	0.0010065	0.0024805
sig	0.4546	0.04503	0.0003676	0.0004012

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
alpha	-0.3981	-0.2319	-0.1503	-0.06633	0.08973
beta[1]	-0.5838	-0.3863	-0.2859	-0.18445	0.01572
beta[2]	-0.7828	-0.5886	-0.4887	-0.38596	-0.18514
int	3.3325	3.4915	3.5744	3.65705	3.81252
sig	0.3770	0.4232	0.4511	0.48215	0.55548

```
(dic2 = dic.samples(mod2, n.iter=1e3))
```

```
Mean deviance: 55.41
penalty 5.133
Penalized deviance: 60.54
```

```
dic1
```

```
Mean deviance: 66.62
```

```

penalty 3.995
Penalized deviance: 70.62

```

This suggests there is much to be gained adding the wool factor to the model. Before we settle on this model however, we should consider whether there is an interaction. Let's look again at the box plot with all six treatment groups.

```
boxplot(log(breaks) ~ wool + tension, data=warpbreaks)
```

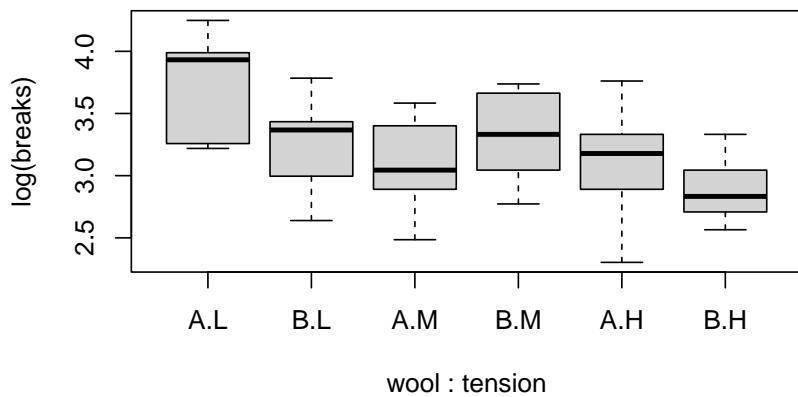


Figure 35.15: Re-examining boxplot of $\log(\text{breaks})$ by wool and tension for interaction effects

Our two-way model has a single effect for wool B and the estimate is negative. If this is true, then we would expect wool B to be associated with fewer breaks than its wool A counterpart on average. This is true for low and high tension, but it appears that breaks are *higher* for wool B when there is medium tension. That is, the effect for wool B is not consistent across tension levels, so it may appropriate to add an interaction term. In R, this would look like:

```

lmod2 = lm(log(breaks) ~ .^2, data=warpbreaks)
summary(lmod2)

```

```

Call:
lm(formula = log(breaks) ~ .^2, data = warpbreaks)

```

```

Residuals:
    Min      1Q  Median      3Q     Max 
-0.81504 -0.27885  0.04042  0.27319  0.64358 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  3.7179    0.1247  29.824 < 2e-16 ***
woolB        -0.4356    0.1763  -2.471  0.01709 *  
tensionM     -0.6012    0.1763  -3.410  0.00133 ** 
tensionH     -0.6003    0.1763  -3.405  0.00134 ** 
woolB:tensionM  0.6281    0.2493   2.519  0.01514 *  
woolB:tensionH  0.2221    0.2493   0.891  0.37749 
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.374 on 48 degrees of freedom
Multiple R-squared:  0.3363,    Adjusted R-squared:  0.2672 
F-statistic: 4.864 on 5 and 48 DF,  p-value: 0.001116

```

Adding the interaction, we get an effect for being in wool B and medium tension, as well as for being in wool B and high tension. There are now six parameters for the mean, one for each treatment group, so this model is equivalent to the full cell means model. Let's use that.

35.3.4 Two-way cell means model

In this new model, μ will be a matrix with six entries, each corresponding to a treatment group.

```

mod3_string = " model {  
  for( i in 1:length(y)) {  
    y[i] ~ dnorm(mu[woolGrp[i], tensGrp[i]], prec)  
  }  
  
  for (j in 1:max(woolGrp)) {  
    for (k in 1:max(tensGrp)) {  
      mu[j,k] ~ dnorm(0.0, 1.0/1.0e6)  
    }  
  }  
}
```

```

prec ~ dgamma(3/2.0, 3*1.0/2.0)
sig = sqrt(1.0 / prec)
} "


str(warpbreaks)

'data.frame':   54 obs. of  3 variables:
$ breaks : num  26 30 54 25 70 52 51 26 67 18 ...
$ wool   : Factor w/ 2 levels "A","B": 1 1 1 1 1 1 1 1 1 1 ...
$ tension: Factor w/ 3 levels "L","M","H": 1 1 1 1 1 1 1 1 2 ...

data3_jags = list(y=log(warpbreaks$breaks), woolGrp=as.numeric(warpbreaks$wool), tensGrp=as.numeric(warpbreaks$tension))

params3 = c("mu", "sig")

mod3 = jags.model(textConnection(mod3_string), data=data3_jags, n.chains=3)

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
Observed stochastic nodes: 54
Unobserved stochastic nodes: 7
Total graph size: 179

Initializing model

update(mod3, 1e3)

mod3_sim = coda.samples(model=mod3,
                        variable.names=params3,
                        n.iter=5e3)
mod3_csim = as.mcmc(do.call(rbind, mod3_sim))

plot(mod3_sim)

```

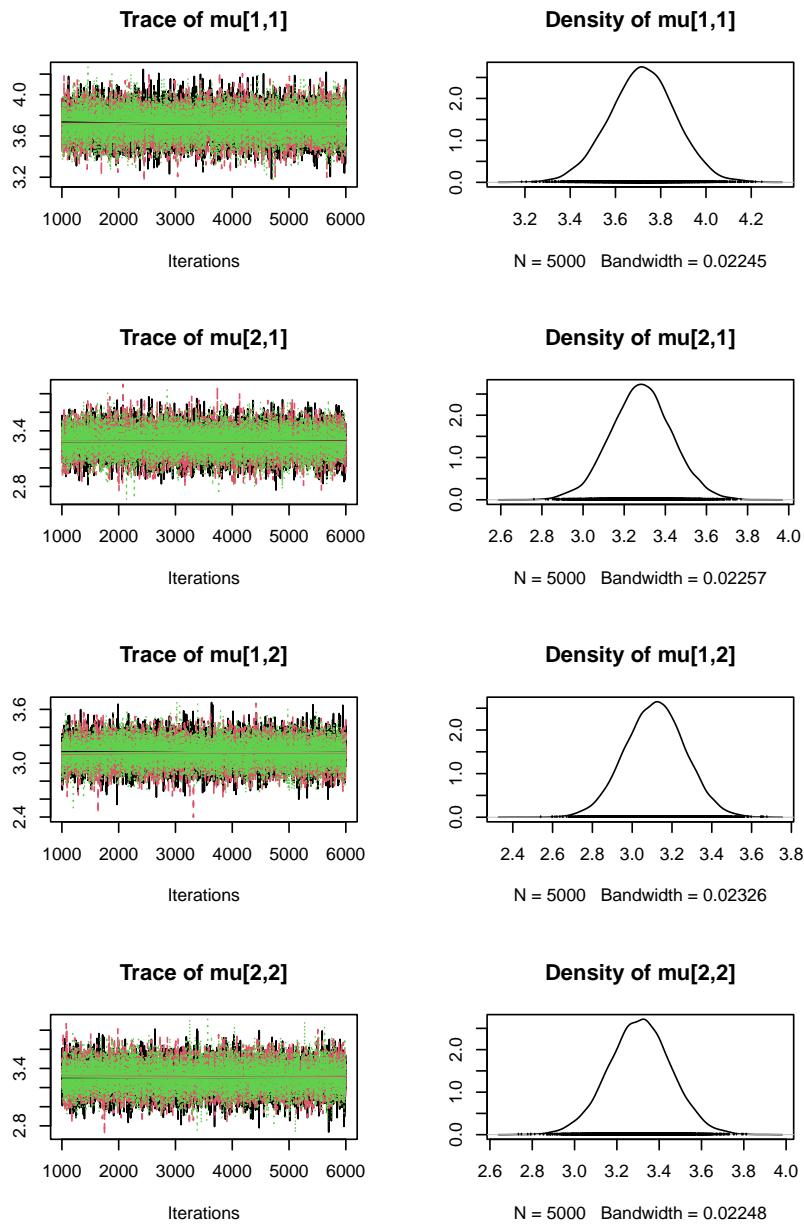


Figure 35.16: Traceplots for the cell means model

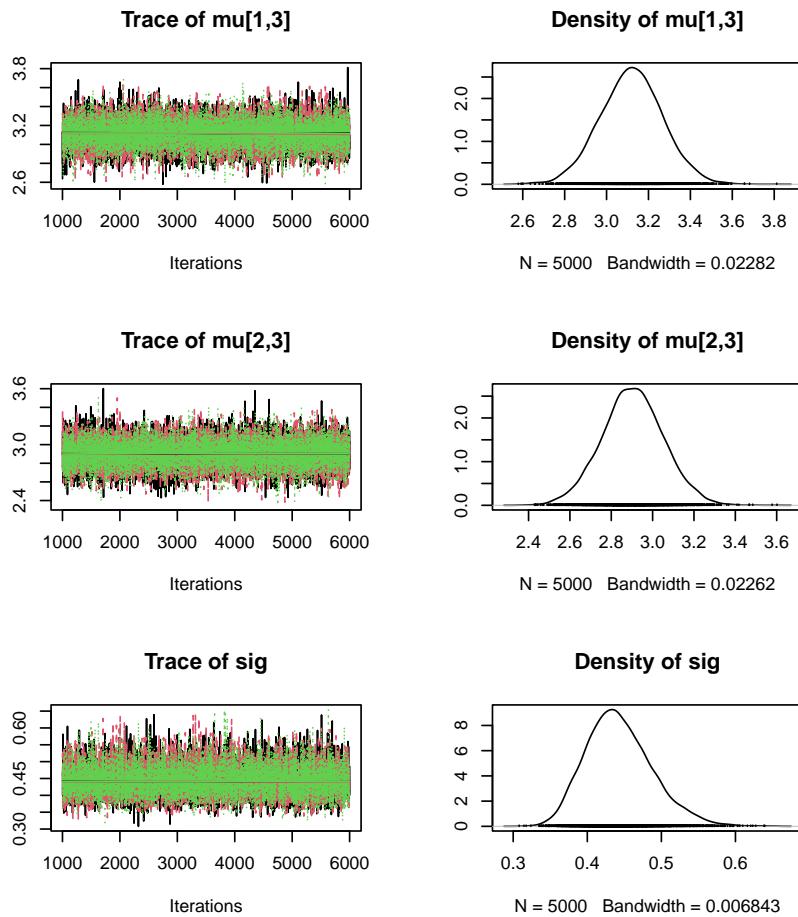


Figure 35.17: Traceplots for the cell means model

```
## convergence diagnostics
gelman.diag(mod3_sim)
```

Potential scale reduction factors:

	Point est.	Upper C.I.
mu[1,1]	1	1
mu[2,1]	1	1
mu[1,2]	1	1
mu[2,2]	1	1
mu[1,3]	1	1
mu[2,3]	1	1

```

sig           1           1

Multivariate psrf

1

autocorr.diag(mod3_sim)

      mu[1,1]      mu[2,1]      mu[1,2]      mu[2,2]      mu[1,3]
Lag 0  1.000000000  1.000000000  1.000000000  1.000000000  1.000000000
Lag 1  0.001192465 -0.007415187 -0.0006307488  0.002358313 -
0.005844319
Lag 5   0.003768745  0.006084217 -0.0041591200 -0.004064950 -
0.005247436
Lag 10  -0.001826478 -0.010495778 -0.0099325928  0.002505943  0.008730558
Lag 50   0.001847142 -0.008064034  0.0118463562 -0.005531318  0.011301083
      mu[2,3]      sig
Lag 0   1.000000000  1.000000000
Lag 1   -0.006785244  0.101042664
Lag 5   -0.014142350  0.015390560
Lag 10   0.002298853  0.007130286
Lag 50   0.000339505  0.006329967

effectiveSize(mod3_sim)

mu[1,1]  mu[2,1]  mu[1,2]  mu[2,2]  mu[1,3]  mu[2,3]      sig
15000.00 15231.51 15000.00 15607.01 15315.64 14915.77 11817.56

raftery.diag(mod3_sim)

[[1]]

Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

      Burn-in Total Lower bound Dependence
      (M)       (N)    (Nmin)      factor (I)

```

mu[1,1] 2	3561	3746	0.951
mu[2,1] 2	3803	3746	1.020
mu[1,2] 2	3803	3746	1.020
mu[2,2] 2	3741	3746	0.999
mu[1,3] 1	3712	3746	0.991
mu[2,3] 2	3866	3746	1.030
sig 3	4062	3746	1.080

[[2]]

Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

	Burn-in (M)	Total (N)	Lower bound (Nmin)	Dependence factor (I)
mu[1,1] 2	3866	3746	1.030	
mu[2,1] 2	3741	3746	0.999	
mu[1,2] 2	3866	3746	1.030	
mu[2,2] 2	3741	3746	0.999	
mu[1,3] 2	3741	3746	0.999	
mu[2,3] 2	3741	3746	0.999	
sig 3	4062	3746	1.080	

[[3]]

Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95

	Burn-in (M)	Total (N)	Lower bound (Nmin)	Dependence factor (I)
mu[1,1] 2	3803	3746	1.020	
mu[2,1] 2	3741	3746	0.999	
mu[1,2] 1	3712	3746	0.991	
mu[2,2] 2	3680	3746	0.982	
mu[1,3] 2	3680	3746	0.982	
mu[2,3] 2	3803	3746	1.020	
sig 2	3930	3746	1.050	

Let's compute the DIC and compare with our previous models.

```
(dic3 = dic.samples(mod3, n.iter=1e3))
```

```
Mean deviance: 52.02
penalty 7.198
Penalized deviance: 59.22
```

```
dic2
```

```
Mean deviance: 55.41
penalty 5.133
Penalized deviance: 60.54
```

```
dic1
```

```
Mean deviance: 66.62
penalty 3.995
Penalized deviance: 70.62
```

This suggests that the full model with interaction between wool and tension (which is equivalent to the cell means model) is the best for explaining/predicting warp breaks.

35.3.5 Results

```
summary(mod3_sim)
```

```
Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000
```

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

	Mean	SD	Naive SE	Time-series SE
mu[1,1]	3.7196	0.14689	0.0011993	0.0011994
mu[2,1]	3.2830	0.14824	0.0012104	0.0012013
mu[1,2]	3.1153	0.15012	0.0012258	0.0012256
mu[2,2]	3.3087	0.14789	0.0012075	0.0011850
mu[1,3]	3.1187	0.14931	0.0012191	0.0012071
mu[2,3]	2.9038	0.14913	0.0012177	0.0012231
sig	0.4432	0.04503	0.0003677	0.0004153

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
mu[1,1]	3.4269	3.6224	3.7205	3.8167	4.005
mu[2,1]	2.9873	3.1846	3.2830	3.3798	3.574
mu[1,2]	2.8202	3.0146	3.1170	3.2162	3.409
mu[2,2]	3.0135	3.2112	3.3096	3.4057	3.598
mu[1,3]	2.8231	3.0199	3.1199	3.2173	3.414
mu[2,3]	2.6095	2.8063	2.9036	3.0020	3.201
sig	0.3668	0.4115	0.4392	0.4707	0.543

```
HPDinterval(mod3_csim)
```

	lower	upper
mu[1,1]	3.4277243	4.0054398
mu[2,1]	2.9997340	3.5828905
mu[1,2]	2.8287553	3.4147201
mu[2,2]	3.0201607	3.6029681
mu[1,3]	2.8350241	3.4242514
mu[2,3]	2.6118653	3.2027065
sig	0.3598126	0.5331316

```
attr(),"Probability")
```

```
[1] 0.95
```

```
par(mfrow=c(3,2)) # arrange frame for plots
densplot(mod3_csim[,1:6], xlim=c(2.0, 4.5))
```

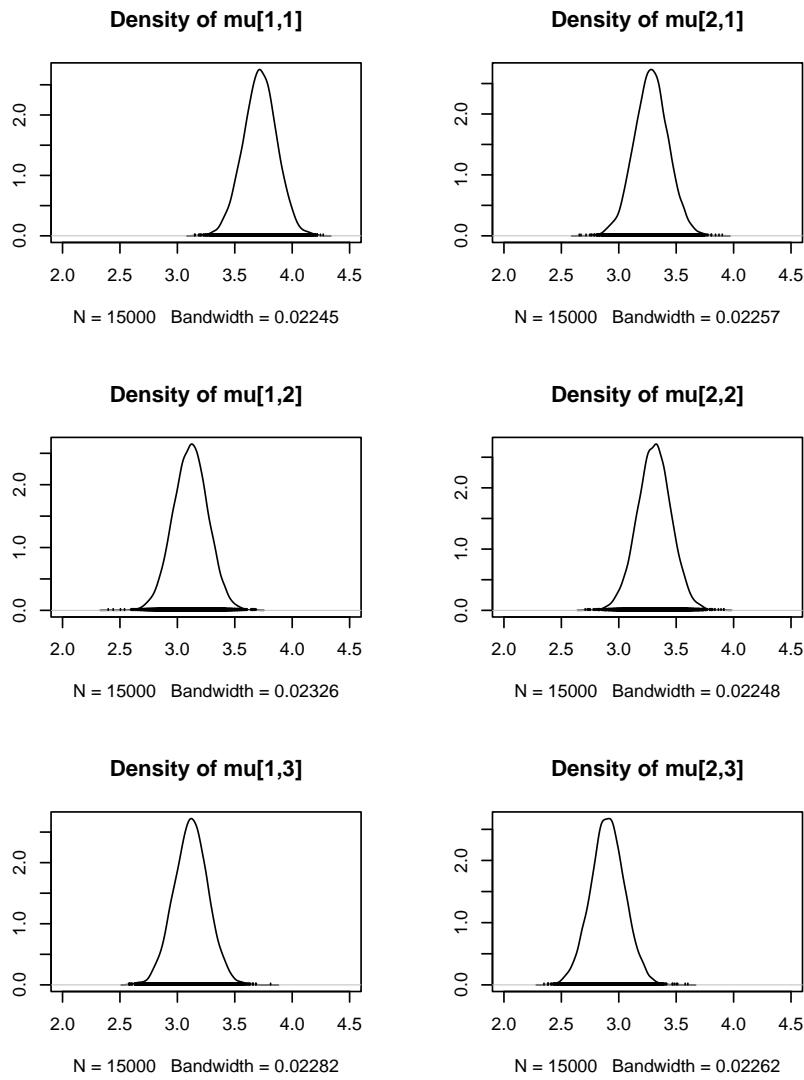


Figure 35.18: Posterior densities for cell means

It might be tempting to look at comparisons between each combination of treatments, but we warn that this could yield spurious results. When we discussed the statistical modeling cycle, we said it is best not to search your results for interesting hypotheses, because if there are many hypotheses, some will appear to show “effects” or “associations” simply due to chance. Results are most reliable when we determine a relatively small number of hypotheses we are interested in beforehand, collect the data, and statistically evaluate the evidence for them.

Table 35.5: Posterior probabilities of each treatment group having the smallest mean break rate

```
prop.table( table( apply(mod3_csim[,1:6], 1, which.min) ) )
```

	2	3	4	5	6
	0.01513333	0.12326667	0.01033333	0.11433333	0.73693333

One question we might be interested in with these data is finding the treatment combination that produces the fewest breaks. To calculate this, we can go through our posterior samples and for each sample, find out which group has the smallest mean. These counts help us determine the posterior probability that each of the treatment groups has the smallest mean.

The evidence supports wool B with high tension as the treatment that produces the fewest breaks.

36 Logistic regression

Bayesian Statistics: Techniques and Models

Logistic regression is the preferred model when modelling a problem where the response variable is binary such as a classification or the outcome of a Bernoulli trial. In such the traditional least square fit suffers from a number of shortcomings. The main idea here is a log transform. However a naive approach this transform imposes issues with 0 valued inputs since $\log(0) = -\infty$

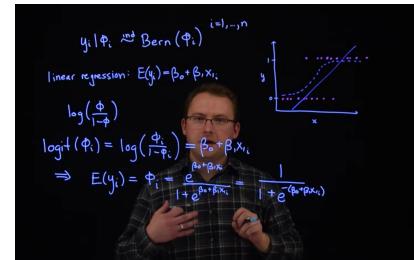
36.1 Introduction to Logistic Regression

36.1.1 Data

For an example of logistic regression , we'll use the urine data set from the boot package in R. The response variable is r, which takes on values of 0 or 1. We will remove some rows from the data set which contain missing values.

```
library("boot")
data("urine")
?urine
head(urine)

r    gravity    ph  osmo cond urea calc
1 0    1.021  4.91   725   NA  443 2.45
2 0    1.017  5.74   577 20.0  296 4.49
3 0    1.008  7.20   321 14.9  101 2.36
4 0    1.011  5.51   408 12.6  224 2.15
5 0    1.005  6.52   187  7.5   91 1.16
6 0    1.020  5.27   668 25.3  252 3.34
```



logistic regression
Figure 36.1.1 Introduction to logistic regression

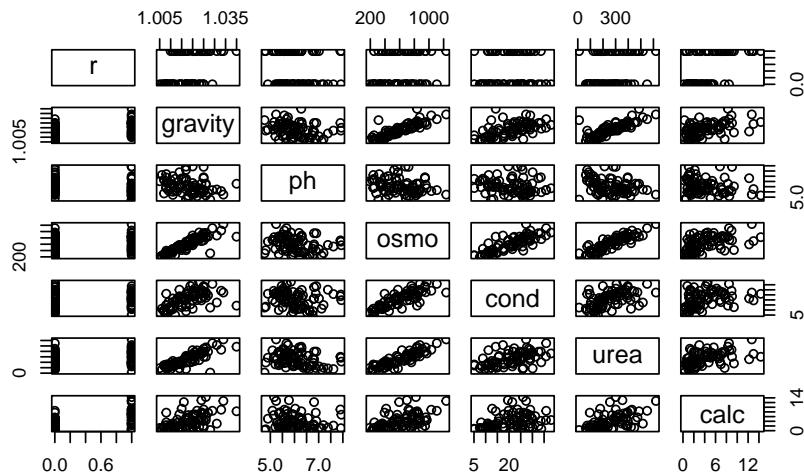
```
dat = na.omit(urine)
```

①

① drop missing values

Let's look at pairwise scatter plots of the seven variables.

```
pairs(dat)
```



One thing that stands out is that several of these variables are strongly correlated with one another. For example `gravity` and `osmo` appear to have a very close linear relationship. Collinearity between x variables in linear regression models can cause trouble for statistical inference. Two correlated variables will compete for the ability to predict the response variable, leading to unstable estimates. This is not a problem for prediction of the response, if prediction is the end goal of the model. But if our objective is to discover *how* the variables relate to the response, we should avoid collinearity.

! Collinearity and Multicollinearity

When two covariates are highly correlated we call this relation **collinearity**. When one covariate in a multiple regression model can be linearly predicted from the others with a substantial degree of accuracy we call this relation **multicollinearity**. It is possible that no two pairs of a such a group of covariates are correlated.

In both cases this will lead to the *design matrix* being almost *singular*. Near singular matrices are a strong cause of instability in

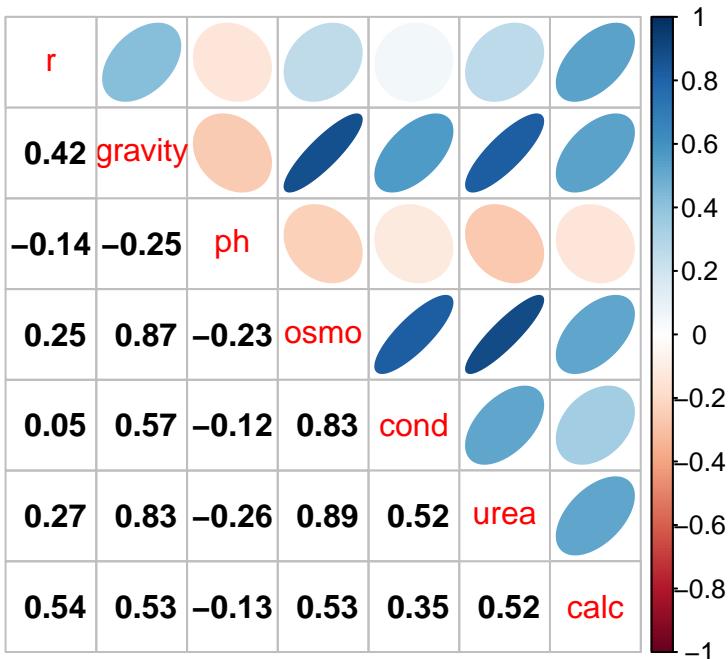
numerical calculations. Statistical this tends to lead to a model with inflated *standard errors* compared to models where we only keep the a subset where variables are neither **collinear** nor **multicollinear**. A consequence of this is that we will see a drop in *statistical significance* for these variables, which will make interpreting the model harder.

We have seen a few strategies ways to deal with these issues:

1. include `pair plot` in the exploratory data analysis phase.
2. picking subsets and checking DIC or,
3. variable selection using double exponential priors.
4. PCA creates independent covariates with a lower dimension with a trade of losing interpretability. See (Johnson and Wichern 2001, 386) (Belsley, Kuh, and Welsch 1980, 85–191) (Härdle and Simar 2019)
5. Feature elimination based on combination of Variance inflation factors (VIF) (Sheather 2009, 203)

We can more formally estimate the correlation among these variables using the `corrplot` package.

```
library("corrplot")  
  
corrplot 0.95 loaded  
  
Cor = cor(dat)  
corrplot(Cor, type="upper", method="ellipse", tl.pos="d")  
corrplot(Cor, type="lower", method="number", col="black",  
        add=TRUE, diag=FALSE, tl.pos="n", cl.pos="n")
```



36.1.2 Variable selection

One primary goal of this analysis is to find out which variables are related to the presence of calcium oxalate crystals. This objective is often called “variable selection.” We have already seen one way to do this: fit several models that include different sets of variables and see which one has the best DIC. Another way to do this is to use a linear model where the priors for the β coefficients favor values near 0 (indicating a weak relationship). This way, the burden of establishing association lies with the data. If there is not a strong signal, we assume it doesn’t exist.

Rather than tailoring a prior for each individual β based on the scale its covariate takes values on, it is customary to subtract the mean and divide by the standard deviation for each variable.

```
X = scale(dat[,-1], center=TRUE, scale=TRUE)
head(X[, "gravity"])
```

```
2           3           4           5           6           7
-0.1403037 -1.3710690 -0.9608139 -1.7813240  0.2699514 -
0.8240622
```

```

colMeans(X)

      gravity          ph          osmo         cond        urea
-9.861143e-15 8.511409e-17 1.515743e-16 -1.829852e-
16 7.335402e-17
      calc
-1.689666e-18

apply(X, 2, sd)

gravity      ph      osmo      cond      urea      calc
1           1       1       1       1       1

```

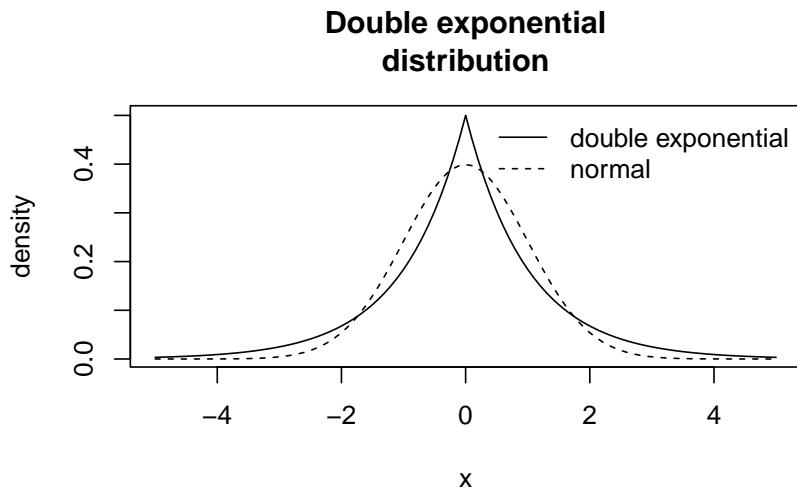
36.1.3 Model

Our prior for the β (which we'll call b in the model) coefficients will be the double exponential (or Laplace) distribution, which as the name implies, is the exponential distribution with tails extending in the positive direction as well as the negative direction, with a sharp peak at 0. We can read more about it in the JAGS manual. The distribution looks like:

```

ddexp = function(x, mu, tau) {
  0.5*tau*exp(-tau*abs(x-mu))
}
curve(ddexp(x, mu=0.0, tau=1.0), from=-5.0, to=5.0, ylab="density", main="Double exponential\\ndistribution")
curve(dnorm(x, mean=0.0, sd=1.0), from=-5.0, to=5.0, lty=2, add=TRUE) # normal distribution
legend("topright", legend=c("double exponential", "normal"), lty=c(1,2), bty="n")

```



```
library("rjags")
```

Loading required package: coda

Linked to JAGS 4.3.2

Loaded modules: basemod, bugs

```
mod1_string = " model {
  for (i in 1:length(y)) {
    y[i] ~ dbern(p[i])
    logit(p[i]) = int + b[1]*gravity[i] + b[2]*ph[i] + b[3]*osmo[i] + b[4]*cond[i] + b[5]*urea[i] + b[6]*calc[i]
  }
  int ~ dnorm(0.0, 1.0/25.0)
  for (j in 1:6) {
    b[j] ~ ddexp(0.0, sqrt(2.0)) # has variance 1.0
  }
}

set.seed(92)
head(X)
```

	gravity	ph	osmo	cond	urea	calc
2	-0.1403037	-0.4163725	-0.1528785	-0.1130908	0.25747827	0.09997564

```

3 -1.3710690  1.6055972 -1.2218894 -0.7502609 -1.23693077 -
0.54608444
4 -0.9608139 -0.7349020 -0.8585927 -1.0376121 -0.29430353 -
0.60978050
5 -1.7813240  0.6638579 -1.7814497 -1.6747822 -1.31356713 -
0.91006194
6  0.2699514 -1.0672806  0.2271214  0.5490664 -0.07972172 -
0.24883614
7 -0.8240622 -0.5825618 -0.6372741 -0.4379226 -0.51654898 -
0.83726644

data_jags = list(y=dat$r, gravity=X[, "gravity"], ph=X[, "ph"], osmo=X[, "osmo"], cond=X[, "cond"], urea=X[, "urea"])

params = c("int", "b")

mod1 = jags.model(textConnection(mod1_string), data=data_jags, n.chains=3)

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
Observed stochastic nodes: 77
Unobserved stochastic nodes: 7
Total graph size: 1085

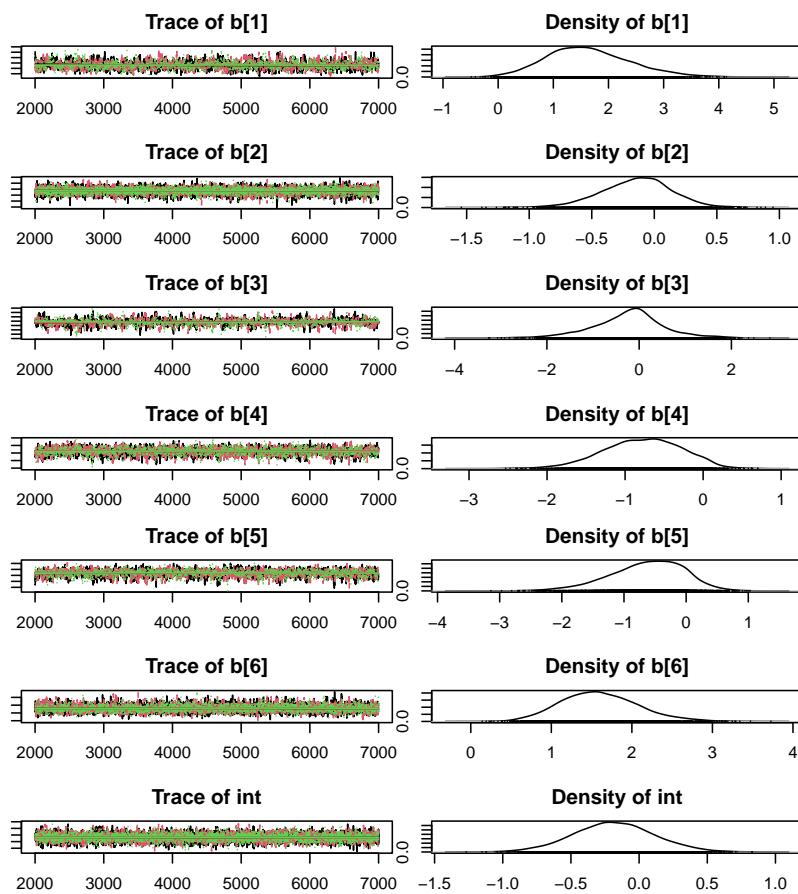
Initializing model

update(mod1, 1e3)

mod1_sim = coda.samples(model=mod1,
                        variable.names=params,
                        n.iter=5e3)
mod1_csim = as.mcmc(do.call(rbind, mod1_sim))

## convergence diagnostics
par(mar = c(2.5, 1, 2.5, 1))
plot(mod1_sim, ask=TRUE)

```



```
gelman.diag(mod1_sim)
```

Potential scale reduction factors:

	Point est.	Upper C.I.
b[1]	1.01	1.02
b[2]	1.00	1.00
b[3]	1.00	1.01
b[4]	1.00	1.00
b[5]	1.00	1.01
b[6]	1.00	1.00
int	1.00	1.00

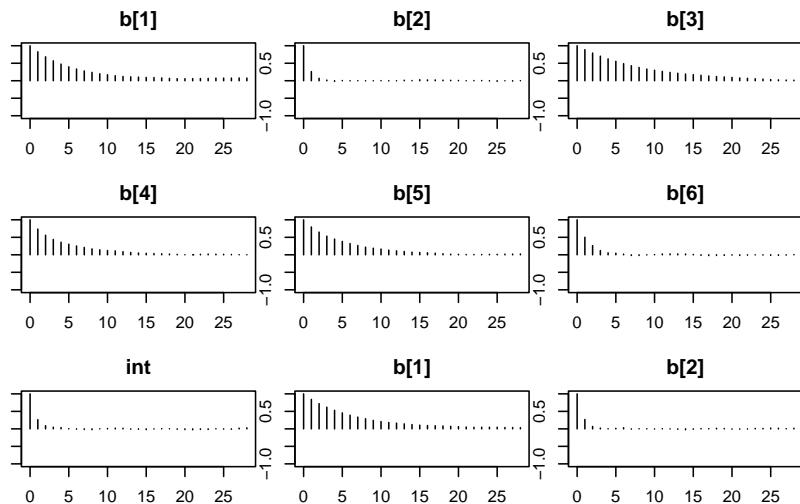
Multivariate psrf

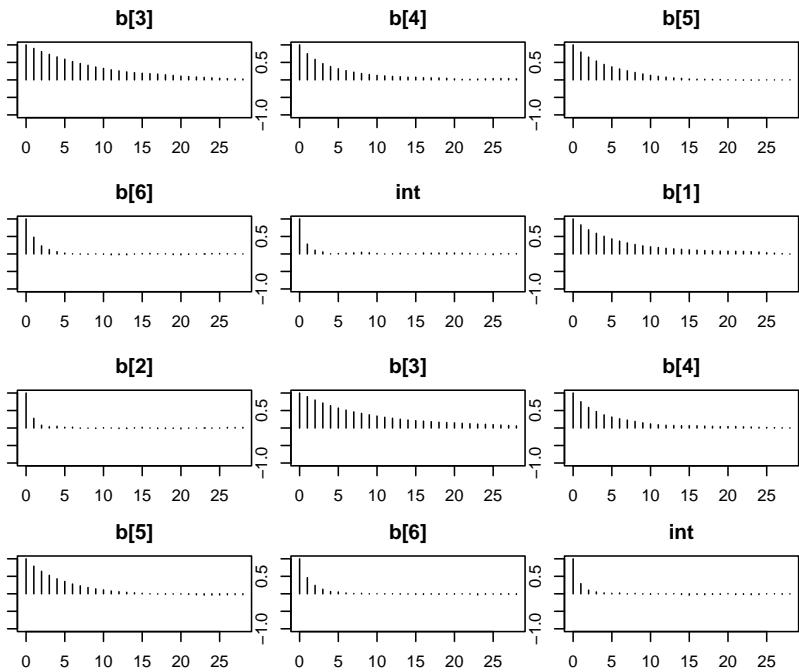
1

```
autocorr.diag(mod1_sim)
```

	b[1]	b[2]	b[3]	b[4]	b[5]	
Lag 0	1.00000000	1.00000000	1.00000000	1.00000000	1.00000000	1.0000000000
Lag 1	0.83333971	0.268680198	0.89398527	0.74501958	0.793376592	
Lag 5	0.42839700	0.014655332	0.57191639	0.31170317	0.368968624	
Lag 10	0.19530612	0.003473139	0.32275307	0.12275465	0.135011942	
Lag 50	0.05425269	-0.012637238	-0.02129934	-0.02554117	0.008644485	
	b[6]	int				
Lag 0	1.0000000000	1.0000000000				
Lag 1	0.4804842272	0.278120975				
Lag 5	0.0400654593	0.009934782				
Lag 10	0.0003389167	0.003054410				
Lag 50	-0.0084813557	-0.018285056				

```
autocorr.plot(mod1_sim)
```





```
effectiveSize(mod1_sim)
```

	b[1]	b[2]	b[3]	b[4]	b[5]	b[6]	int
1292.3237	8332.3081	839.5348	1697.1928	1525.4536	5255.7450	7648.3931	

```
## calculate DIC
dic1 = dic.samples(mod1, n.itер=1e3)
```

Let's look at the results.

```
summary(mod1_sim)
```

```
Iterations = 2001:7000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000
```

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

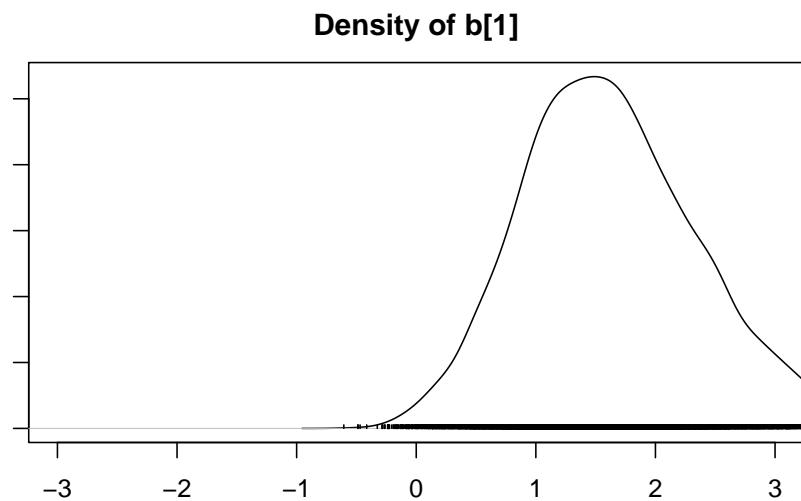
	Mean	SD	Naive SE	Time-series SE	SE
b[1]	1.6363	0.7540	0.006157		0.021070
b[2]	-0.1438	0.2847	0.002325		0.003128
b[3]	-0.2360	0.7908	0.006457		0.027327
b[4]	-0.7771	0.5041	0.004116		0.012242
b[5]	-0.6295	0.5956	0.004863		0.015266
b[6]	1.5998	0.4800	0.003919		0.006621
int	-0.1849	0.2983	0.002436		0.003429

2. Quantiles for each variable:

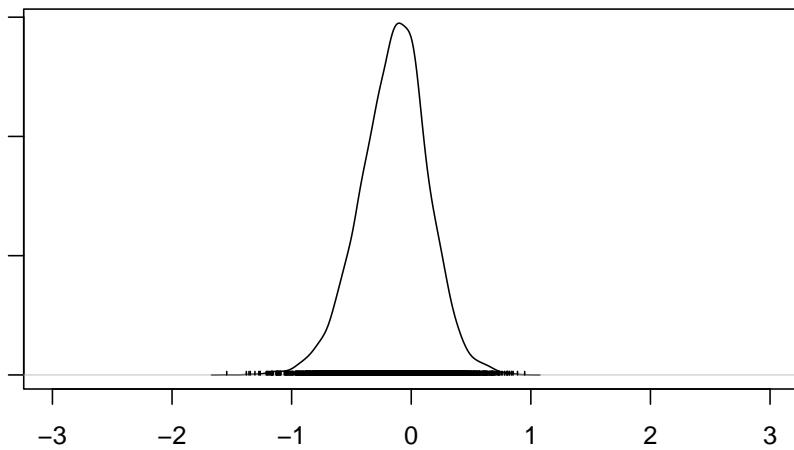
	2.5%	25%	50%	75%	97.5%
b[1]	0.2999	1.1033	1.5765	2.11485	3.2628
b[2]	-0.7366	-0.3250	-0.1287	0.04150	0.3886
b[3]	-1.9347	-0.6774	-0.1863	0.20637	1.4195
b[4]	-1.8027	-1.1122	-0.7578	-0.41747	0.1104
b[5]	-1.9323	-1.0081	-0.5775	-0.19679	0.3941
b[6]	0.7316	1.2620	1.5738	1.90818	2.6318
int	-0.7750	-0.3851	-0.1876	0.01112	0.4069

```
#par(mfrow=c(3,2))
par(mar = c(2.5, 1, 2.5, 1))

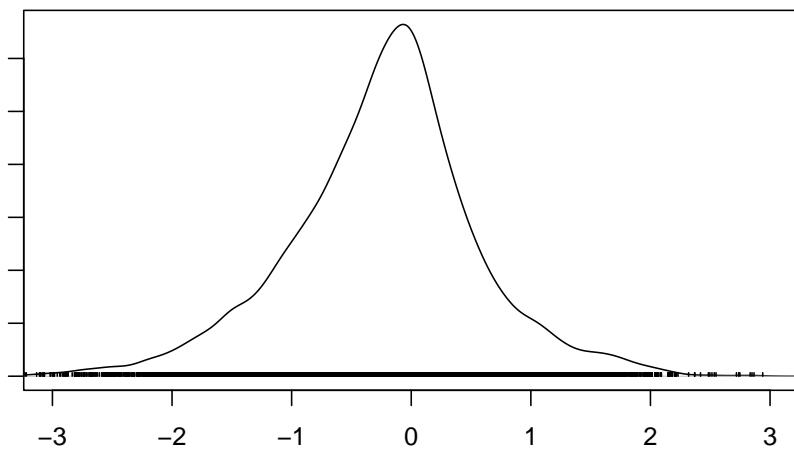
densplot(mod1_csim[,1:6], xlim=c(-3.0, 3.0))
```



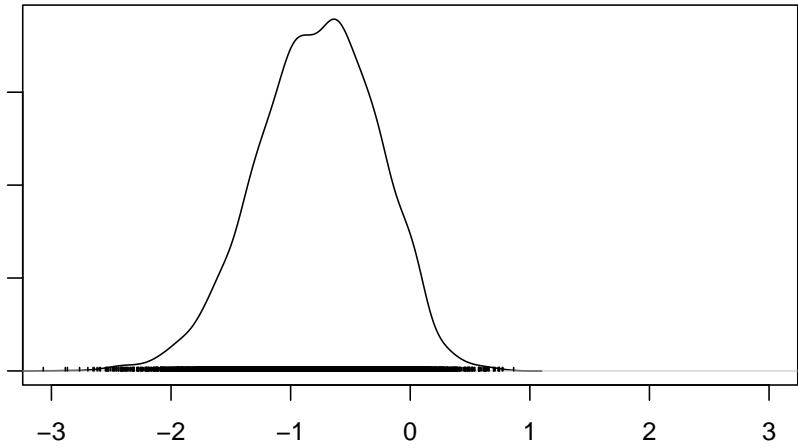
Density of $b[2]$



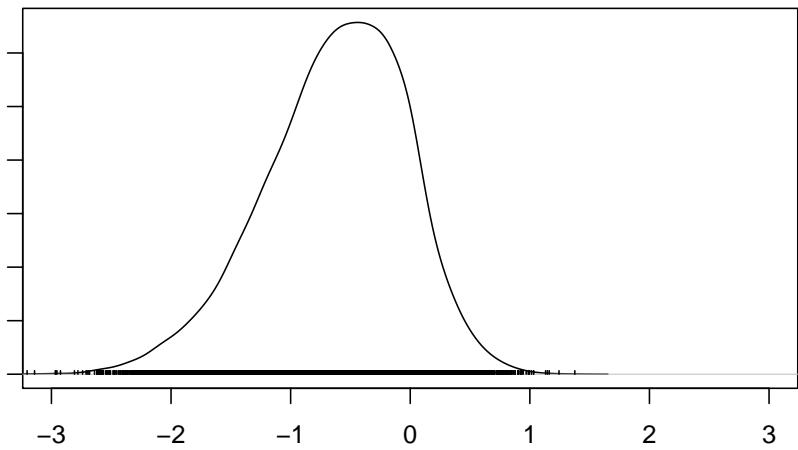
Density of $b[3]$

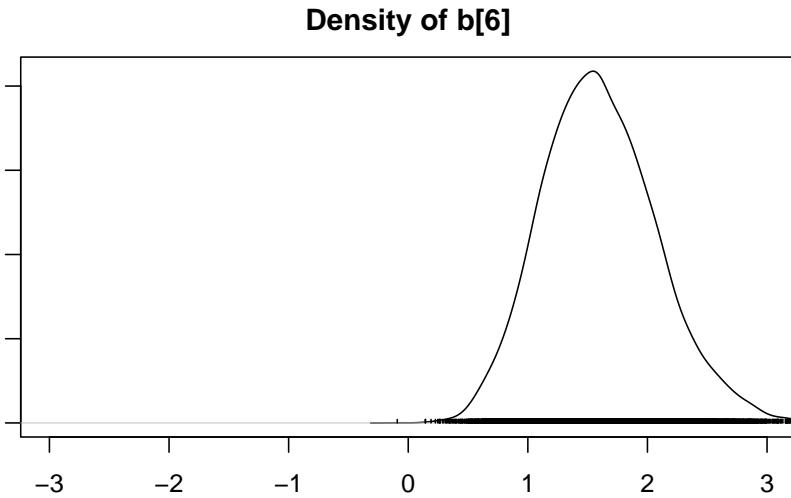


Density of $b[4]$



Density of $b[5]$





```
colnames(X) # variable names
```

```
[1] "gravity" "ph"      "osmo"     "cond"     "urea"     "calc"
```

It is clear that the coefficients for variables `gravity`, `cond` (conductivity), and `calc` (calcium concentration) are not 0. The posterior distribution for the coefficient of `osmo` (osmolarity) looks like the prior, and is almost centered on 0 still, so we'll conclude that `osmo` is not a strong predictor of calcium oxalate crystals. The same goes for `ph`.

`urea` (urea concentration) appears to be a borderline case. However, if we refer back to our correlations among the variables, we see that `urea` is highly correlated with `gravity`, so we opt to remove it.

Our second model looks like this:

```
mod2_string = " model {
  for (i in 1:length(y)) {
    y[i] ~ dbern(p[i])
    logit(p[i]) = int + b[1]*gravity[i] + b[2]*cond[i] + b[3]*calc[i]
  }
  int ~ dnorm(0.0, 1.0/25.0)
  for (j in 1:3) {
    b[j] ~ dnorm(0.0, 1.0/25.0) # noninformative for logistic regression
  }
}
```

```

mod2 = jags.model(textConnection(mod2_string), data=data_jags, n.chains=3)

Warning in jags.model(textConnection(mod2_string), data = data_jags, n.chains =
3): Unused variable "ph" in data

Warning in jags.model(textConnection(mod2_string), data = data_jags, n.chains =
3): Unused variable "osmo" in data

Warning in jags.model(textConnection(mod2_string), data = data_jags, n.chains =
3): Unused variable "urea" in data

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
Observed stochastic nodes: 77
Unobserved stochastic nodes: 4
Total graph size: 635

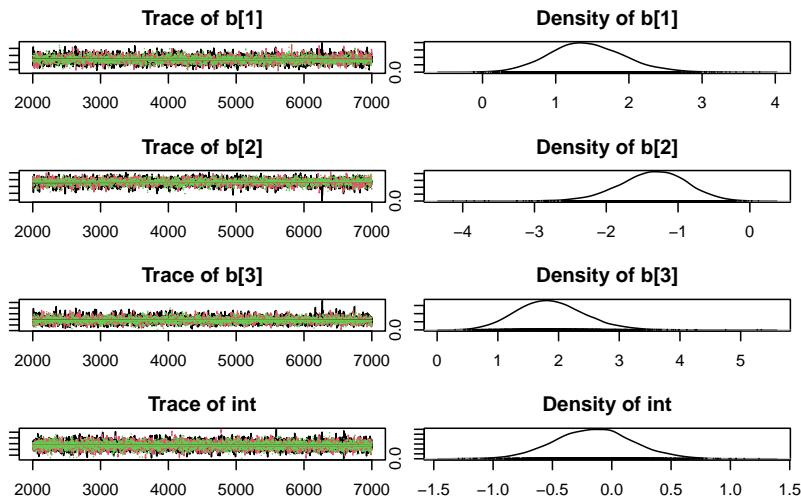
Initializing model

update(mod2, 1e3)

mod2_sim = coda.samples(model=mod2,
                       variable.names=params,
                       n.iter=5e3)
mod2_csim = as.mcmc(do.call(rbind, mod2_sim))

par(mar = c(2.5, 1, 2.5, 1))
#plot(mod2_sim, ask=TRUE)
plot(mod2_sim)

```



```
gelman.diag(mod2_sim)
```

Potential scale reduction factors:

	Point est.	Upper C.I.
b[1]	1	1.00
b[2]	1	1.01
b[3]	1	1.00
int	1	1.00

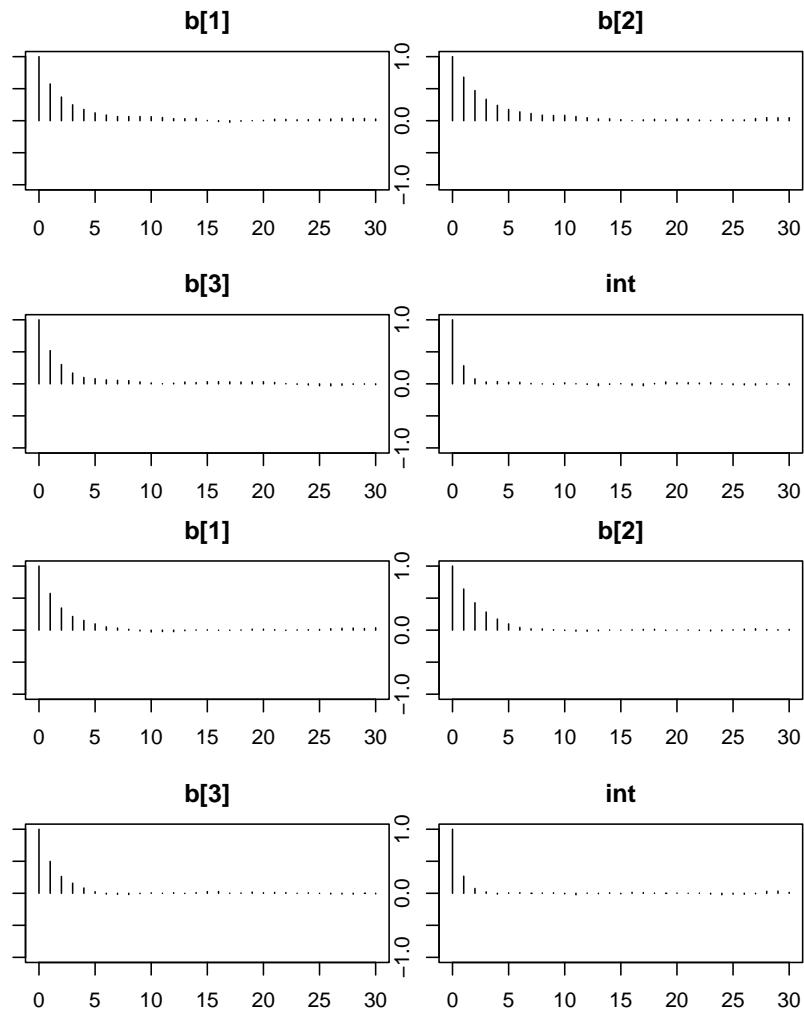
Multivariate psrf

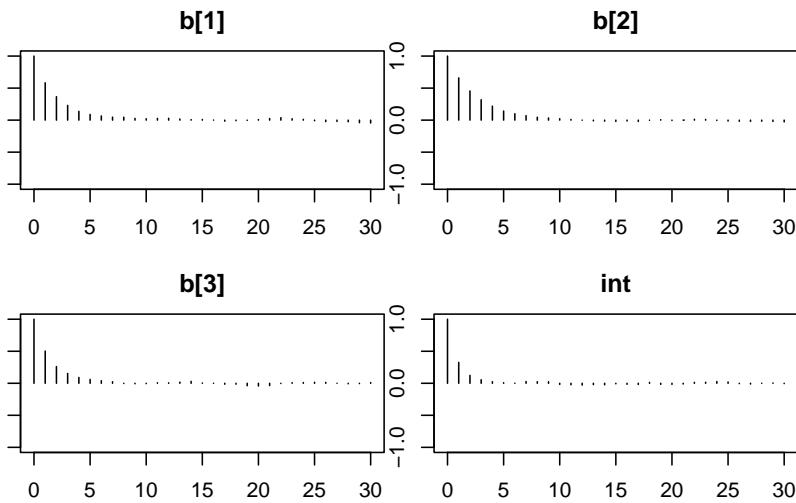
1

```
autocorr.diag(mod2_sim)
```

	b[1]	b[2]	b[3]	int
Lag 0	1.000000000	1.000000000	1.000000000	1.000000000
Lag 1	0.576813396	0.660650056	0.506176769	0.292122676
Lag 5	0.103047112	0.138054989	0.054868797	0.012707173
Lag 10	0.018403496	0.033676464	0.004801542	-0.003187347
Lag 50	-0.008449617	0.001492202	0.001927898	0.009934440

```
autocorr.plot(mod2_sim)
```





```
effectiveSize(mod2_sim)
```

	b[1]	b[2]	b[3]	int
3556.347	2724.732	4829.071	8224.933	

```
dic2 = dic.samples(mod2, n.iter=1e3)
```

36.1.4 Results

```
dic1
```

```
Mean deviance: 68.19
penalty 5.529
Penalized deviance: 73.72
```

```
dic2
```

```
Mean deviance: 71.23
penalty 3.963
Penalized deviance: 75.19
```

```
summary(mod2_sim)
```

```
Iterations = 2001:7000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000
```

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

	Mean	SD	Naive SE	Time-series SE
b[1]	1.4253	0.5079	0.004147	0.008521
b[2]	-1.3509	0.4659	0.003804	0.009055
b[3]	1.8714	0.5549	0.004531	0.008015
int	-0.1517	0.3264	0.002665	0.003604

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
b[1]	0.4829	1.0793	1.4021	1.75538	2.4890
b[2]	-2.3252	-1.6478	-1.3322	-1.03081	-0.4918
b[3]	0.8600	1.4853	1.8444	2.22288	3.0361
int	-0.7896	-0.3728	-0.1516	0.06151	0.4969

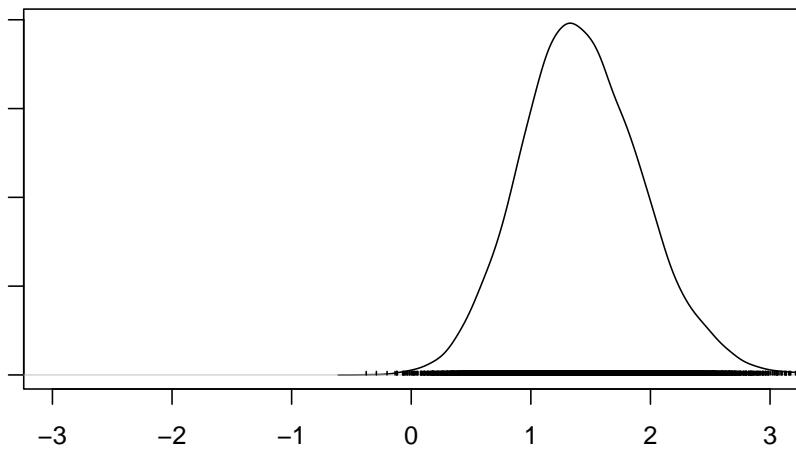
```
HPDinterval(mod2_csim)
```

	lower	upper
b[1]	0.4232802	2.4207008
b[2]	-2.2929131	-0.4701245
b[3]	0.8219038	2.9831269
int	-0.7925645	0.4917614

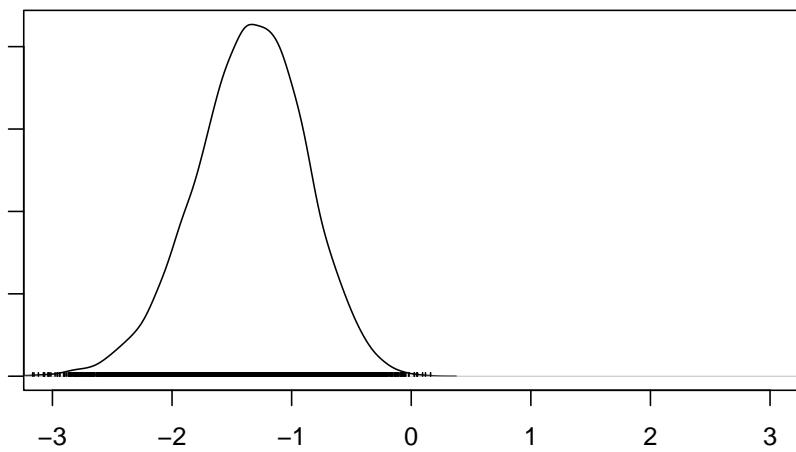
```
attr(),"Probability")
[1] 0.95
```

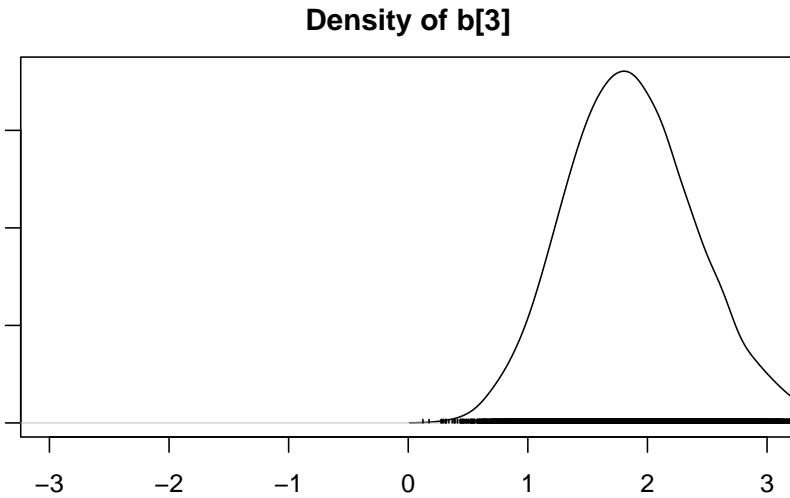
```
#par(mfrow=c(3,1))
par(mar = c(2.5, 1, 2.5, 1))
densplot(mod2_csim[,1:3], xlim=c(-3.0, 3.0))
```

Density of $b[1]$



Density of $b[2]$





```
colnames(X)[c(1,4,6)] # variable names
```

```
[1] "gravity" "cond"      "calc"
```

The DIC is actually better for the first model. Note that we did change the prior between models, and generally we should not use the DIC to choose between priors. Hence comparing DIC between these two models may not be a fair comparison. Nevertheless, they both yield essentially the same conclusions. Higher values of `gravity` and `calc` (calcium concentration) are associated with higher probabilities of *calcium oxalate crystals*, while higher values of `cond` (conductivity) are associated with lower probabilities of *calcium oxalate crystals*.

There are more modeling options in this scenario, perhaps including transformations of variables, different priors, and interactions between the predictors, but we'll leave it to you to see if you can improve the model.

37 Poisson regression

Bayesian Statistics: Techniques and Models

Poisson regression is the preferred method to handle count data where the response is positive values but includes zeroes. The gist of this method is that we use a log transform link function on the regressors but not on the response which allows the response to be zero valued which correspond to zero counts. One limit of this approach mentioned below is that the Poisson takes one parameter λ for both its expected value and its variance. We will give a deeper solution to this problem in the next course on mixture models, however in this course we will consider more restricted cases where we extend the Poisson regression with the **Negative Binomial Distribution** which allows us to model over-dispersed data.

i Overdispersion

In cases where the variance needs to be a separate feature of the model we can use an alternative model with additional free parameters may provide a better fit. In this case of count data, a Poisson mixture model like the Negative Binomial Distribution can be posited instead, in which the mean of the Poisson distribution can itself be thought of as a random variable drawn – from the Gamma distribution thereby introducing an additional free parameter. We don't see the case of over-dispersion in this course but the next course is on mixture models.

37.1 Introduction to Poisson regression

We now have experience fitting regression models when the response is continuous, and when it is binary. What about when we have count data?



Figure 37.1: Introduction to Poisson regression

We could fit a linear normal regression, but here we have a couple of drawbacks. First of all, counts usually aren't negative. And the variances might not be constant. The Poisson distribution provides a natural likelihood for count data.

$$y_i \mid \lambda + i \stackrel{iid}{\sim} \text{Pois}(\lambda_i) \quad i = 1, \dots, n$$

Here, λ conveniently represents the expected value of y $\mathbb{E}[y]$. It turns out that λ is also the variance of y $\text{Var}[y]$. So if we expect a count to be higher, we also expect the variability in counts to go up.

We saw this earlier with the *warp breaks* data.

If we model the mean directly, like we did with linear regression. That is, we had the expected value y_i was directly modeled with this linear form.

$$\mathbb{E}[y] = \beta_0 + \beta_1 x_i \quad (\text{linear regression})$$

We would run into the same problem we did with logistic regression. The expected value has to be greater than zero in the Poisson distribution. To naturally deal with that restriction, we're going to use the logarithmic link function.

So, the log link. That is, that the log of λ_i is equal to this linear piece.

log link:

$$\log(\lambda_i) = \beta_0 + \beta_1 x_i \quad (\text{log link}) \quad (37.1)$$

$$\mathbb{E}[y] = \beta_0 + \beta_1 x_i \quad (\text{linear regression})$$

From this, we can easily recover the expression for the mean itself. That is, we can invert this link function to get the expected value of y_i ,

$$\implies \mathbb{E}[y] = \lambda_i = e^{(\beta_0 + \beta_1 x_i)} \quad (37.2)$$

It might seem like this model is equivalent to fitting a normal linear regression to the log of y . But there are a few key differences. In the normal regression, we're modeling the mean of the response directly. So we would be fitting a model to the $\log(y)$. Where we're modeling the expected value

of the $\log(y)$. This is different from what we're modeling here, here we're doing the log of the expected value of y .

$$\mathbb{E}[\log(y)] \neq \log(\mathbb{E}[y])$$

These are not equal, they're usually similar, but they're not the same. Another difference is that we have a separate independent parameter for the variants in a normal regression. In Poisson regression, the variance is automatically the same as λ , which may not always be appropriate, as we'll see in an upcoming example.

As usual, we can add more explanatory x variables to the Poisson regression framework. They can be continuous, categorical, or they could be counts themselves.

If we have three predictor variables $x_i = (x_{1,i}, x_{2,i}, x_{3,i})$, what would the likelihood part of the hierarchical representation of a Poisson regression with logarithmic link look like?

- $y_i | x_i, \beta \stackrel{\text{ind}}{\sim} \text{Pois}\left(e^{-(\beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \beta_3 x_{3,i})}\right)$
- $y_i | x_i, \beta \stackrel{\text{ind}}{\sim} \text{Pois}\left(\beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \beta_3 x_{3,i}\right)$
- $y_i | x_i, \beta \stackrel{\text{ind}}{\sim} \text{Pois}\left(e^{\beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \beta_3 x_{3,i}}\right)$
- $y_i | x_i, \beta \stackrel{\text{ind}}{\sim} \text{Pois}\left(\log[\beta_0 + \beta_1 x_{1,i} + \beta_2 x_{2,i} + \beta_3 x_{3,i}]\right)$

Here we incorporated the (inverse) link function directly into the likelihood rather than writing it with two lines.

37.2 Poisson regression - JAGS model

For an example of Poisson regression, we'll use the `badhealth` data set from the `COUNT` package in R.

doctor visits

```
library("COUNT")
```

```
Loading required package: msme
```

```
Loading required package: MASS
```

```
Loading required package: lattice
```

```
Loading required package: sandwich
```

```
data("badhealth")
#?badhealth
head(badhealth)
```

	numvisit	badh	age
1	30	0	58
2	20	0	54
3	16	0	44
4	20	0	57
5	15	0	33
6	15	0	28

according to the description:

i Data Card for badhealth

1,127 observations from a 1998 German survey with 3 variables:

1. numvisit - number of visits to the doctor in 1998 (**response**)
2. badh - $\begin{cases} 1 & \text{patient claims to be in bad health} \\ 0 & \text{patient does not claim to be in bad health} \end{cases}$
3. age - age of patient

```
any(is.na(badhealth))
```

```
[1] FALSE
```

1. remove na

As usual, let's visualize these data.

```
hist(badhealth$numvisit, breaks=20)
```

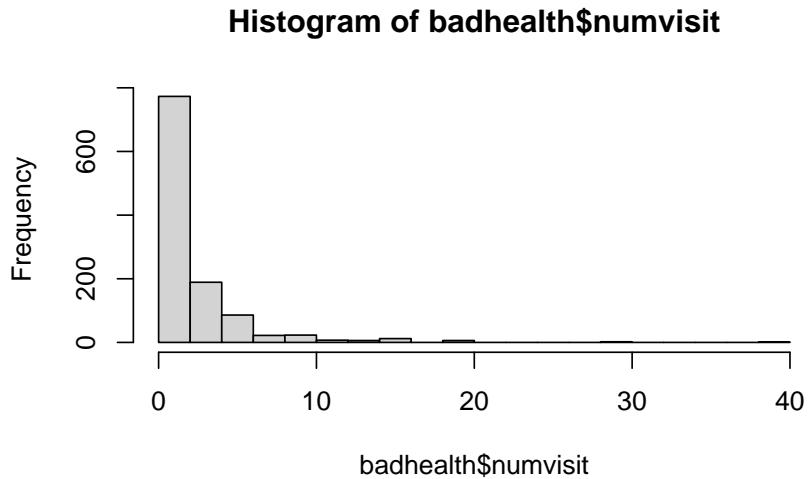


Figure 37.2: Histogram of number of doctor visits

```
plot(jitter(log(numvisit)) ~ jitter(age), data=badhealth, subset=badh==0, xlab="age", ylab="log(visits)")
points(jitter(log(numvisit)) ~ jitter(age), data=badhealth, subset=badh==1, col="red")
```

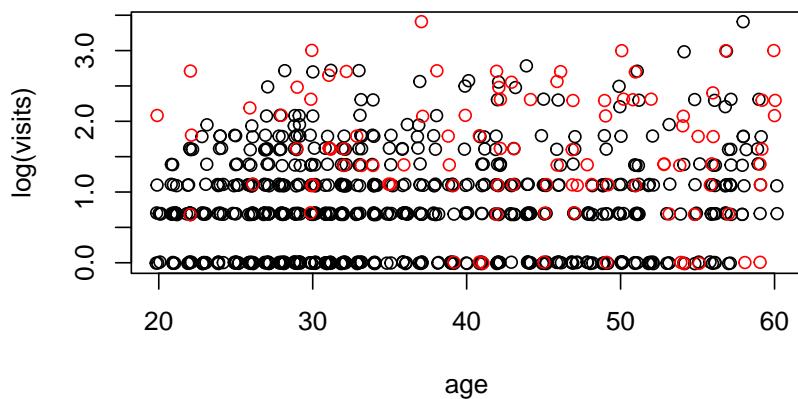


Figure 37.3

37.2.1 Doctor Visits Model

It appears that both age and bad health are related to the number of doctor visits. We should include model terms for both variables. If we believe the age/visits relationship is different between healthy and non-healthy populations, we should also include an interaction term. We will fit the

doctor visits

full model here and leave it to you to compare it with the simpler additive model.

```
library("rjags")

Loading required package: coda

Linked to JAGS 4.3.2

Loaded modules: basemod,bugs

mod_string = " model {
  for (i in 1:length(numvisit)) {
    numvisit[i] ~ dpois(lam[i])
    log(lam[i]) = int + b_badh*badh[i] + b_age*age[i] + b_intx*age[i]*badh[i]
  }

  int ~ dnorm(0.0, 1.0/1e6)
  b_badh ~ dnorm(0.0, 1.0/1e4)
  b_age ~ dnorm(0.0, 1.0/1e4)
  b_intx ~ dnorm(0.0, 1.0/1e4)
} "

set.seed(102)

data_jags = as.list(badhealth)

params = c("int", "b_badh", "b_age", "b_intx")

mod = jags.model(textConnection(mod_string), data=data_jags, n.chains=3)

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
  Observed stochastic nodes: 1127
  Unobserved stochastic nodes: 4
  Total graph size: 3665

Initializing model
```

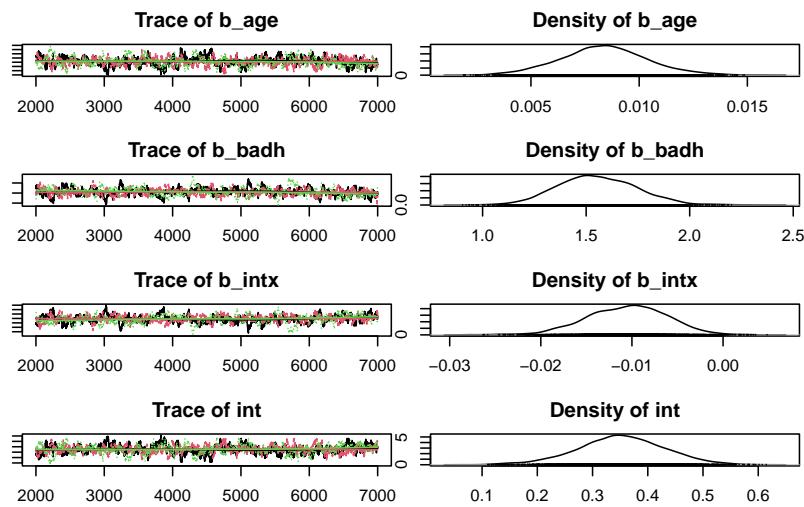
```

update(mod, 1e3)

mod_sim = coda.samples(model=mod, variable.names=params, n.iter=5e3)
mod_csim = as.mcmc(do.call(rbind, mod_sim))

## convergence diagnostics
par(mar = c(2.5, 1, 2.5, 1))
plot(mod_sim)

```



```
gelman.diag(mod_sim)
```

Potential scale reduction factors:

	Point est.	Upper C.I.
b_age	1.01	1.02
b_badh	1.02	1.03
b_intx	1.03	1.04
int	1.01	1.02

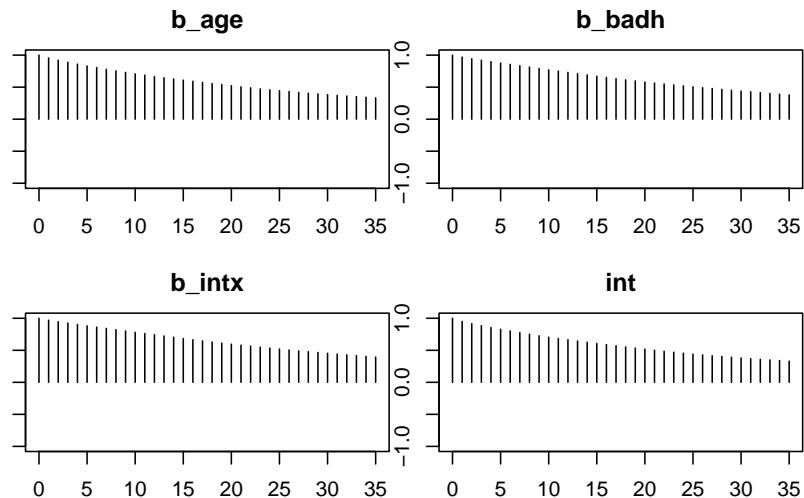
Multivariate psrf

1

```
autocorr.diag(mod_sim)
```

	b_age	b_badh	b_intx	int
Lag 0	1.0000000	1.0000000	1.0000000	1.0000000
Lag 1	0.9544594	0.9670165	0.9687405	0.9500387
Lag 5	0.8261804	0.8686127	0.8747054	0.8227263
Lag 10	0.6991612	0.7557883	0.7663685	0.6961377
Lag 50	0.1979447	0.2232933	0.2368599	0.1954060

```
autocorr.plot(mod_csim)
```



```
effectiveSize(mod_sim)
```

	b_age	b_badh	b_intx	int
272.9714	207.0261	191.3766	272.7331	

```
## compute DIC
dic = dic.samples(mod, n.iter=1e3)
```

37.2.2 Model checking - Residuals

“While inexact models may mislead, attempting to allow for every contingency a priori is impractical. Thus models

must be built by an iterative feedback process in which an initial parsimonious model may be modified when diagnostic checks applied to residuals indicate the need.” —G. E. P. Box

To get a general idea of the model’s performance, we can look at predicted values and residuals as usual. Don’t forget that we must apply the inverse of the link function to get predictions for λ .

```
X = as.matrix(badhealth[, -1])  
X = cbind(X, with(badhealth, badh*age))  
head(X)
```

- ① we drop the first column since it is the column for our y .
- ② we add a third column with $\text{badh} \times \text{age}$

```
badh age  
[1,] 0 58 0  
[2,] 0 54 0  
[3,] 0 44 0  
[4,] 0 57 0  
[5,] 0 33 0  
[6,] 0 28 0
```

```
(pmed_coef = apply(mod_csim, 2, median))
```

①

- ① this are the column medians of the coefficients.

```
b_age b_badh b_intx int  
0.008362897 1.546582440 -0.010363081 0.351975234
```

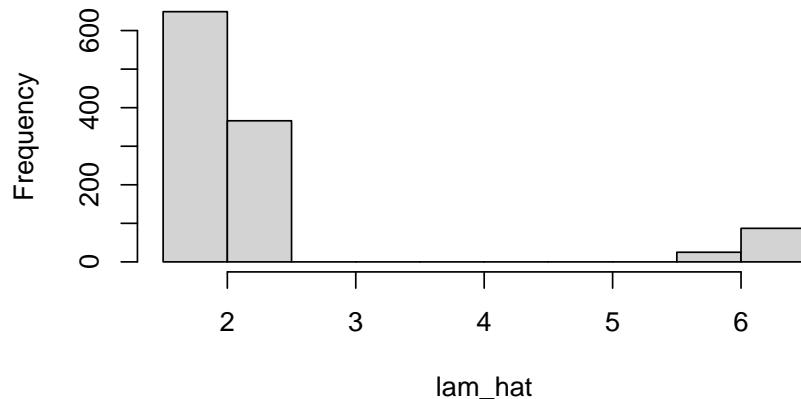
```
llam_hat = pmed_coef["int"] + X %*% pmed_coef[c("b_badh", "b_age", "b_intx")]  
lam_hat = exp(llam_hat)
```

②

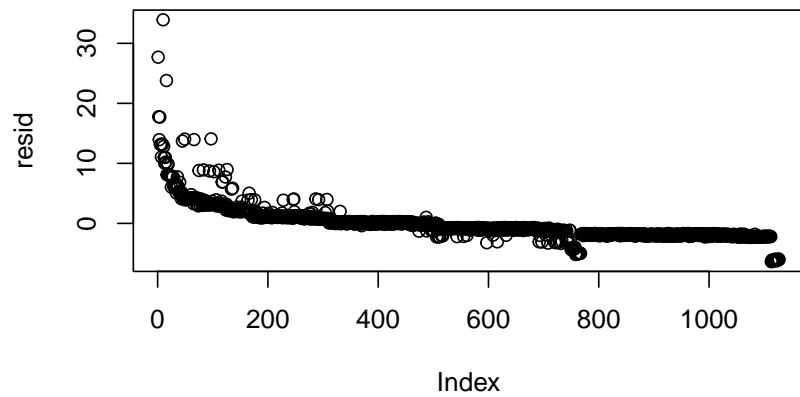
`hist(lam_hat)`

- ① $X \cdot \vec{b}_i$ gives the linear part.
- ② $\hat{\lambda}_i = e^{X \cdot \vec{b}_i}$ we need to apply the inverse link function

Histogram of lam_hat



```
resid = badhealth$numvisit - lam_hat  
plot(resid) # the data were ordered
```



this plot looks bad, it might not be iid but we can ignore the issue since the data is presorted/

```
plot(lam_hat, badhealth$numvisit)  
abline(0.0, 1.0)
```

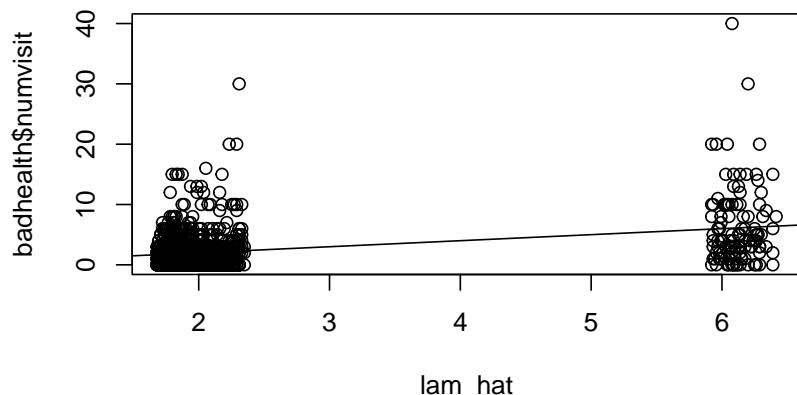


Figure 37.4

```
plot(lam_hat[which(badhealth$badh==0)], resid[which(badhealth$badh==0)], xlim=c(0, 8), ylab="residuals", xlab="lambda-hat")
points(lam_hat[which(badhealth$badh==1)], resid[which(badhealth$badh==1)], col="red")
```

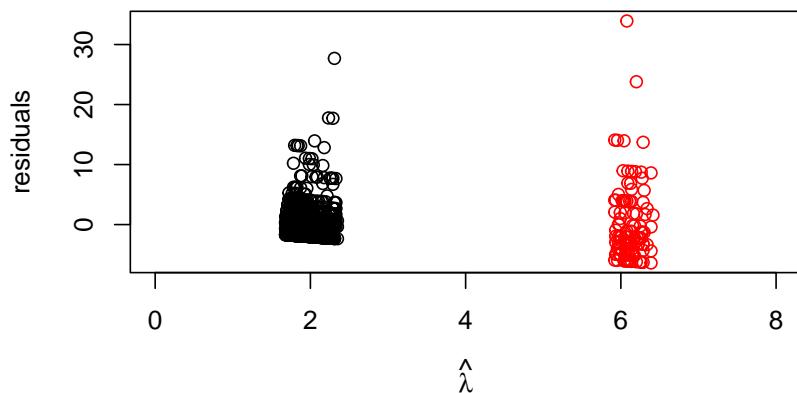


Figure 37.5

It is not surprising that the variability increases for values predicted at higher values since the mean is also the variance in the Poisson distribution. However, observations predicted to have about two visits should have variance about two, and observations predicted to have about six visits should have variance about six.

```
var(resid[which(badhealth$badh==0)])
```

```
[1] 7.022601
```

```
var(resid[which(badhealth$badh==1)])
```

```
[1] 41.1969
```

For this data the variance is much bigger this is not the case with these data. This indicates that either the model fits poorly (meaning the covariates don't explain enough of the variability in the data), or the data are “**overdispersed**” for the Poisson likelihood we have chosen. This is a common issue with count data. If the data are more variable than the Poisson likelihood would suggest, a good alternative is the negative binomial distribution, which we will not pursue here.

37.3 Predictive distributions

Assuming the model fit is adequate, we can interpret the results.

```
summary(mod_sim)
```

```
Iterations = 2001:7000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000
```

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

	Mean	SD	Naive SE	Time-series SE
b_age	0.008364	0.002011	1.642e-05	0.0001268
b_badh	1.553735	0.192126	1.569e-03	0.0143542
b_intx	-0.010581	0.004426	3.614e-05	0.0003408
int	0.351642	0.078480	6.408e-04	0.0048974

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
b_age	0.004392	0.007062	0.008363	0.009650	0.012492
b_badh	1.198005	1.421405	1.546582	1.682886	1.937184

```
b_intx -0.019361 -0.013569 -0.010363 -0.007514 -0.002441
int      0.191461  0.301380  0.351975  0.403701  0.503140
```

The **intercept** is not necessarily *interpretable* here because it corresponds to the number of doctor visits for a healthy 0-year-old. While the number of visits for a newborn baby sounds like interesting information, the youngest person in the data set is 20 years old. In such cases we should avoid making such projections and say that the intercept is an artifact of the model.

- For healthy individuals, it appears that **age** is associated with an *increase* in the Expected *number of doctor visits*.
- **Bad health** is associated with an *increase in expected number of visits*.
- The interaction coefficient is interpreted as an adjustment to the age coefficient for people in bad health. Hence, for people with bad health, age is essentially unassociated with number of visits.

37.3.1 Predictive distributions

Let's say we have two people aged 35, one in good health and the other in poor health.

This goes beyond the posterior probabilities we have calculated comparing expected responses in previous lessons. Here we will create Monte Carlo samples for the **responses themselves**. This is done by taking the Monte Carlo samples of the model parameters, and for each of those, drawing a sample from the likelihood.

Q. What is the posterior probability that the individual with poor health will have more doctor visits?

Let's walk through this.

First, we need the x values for each individual. We'll say the healthy one is Person 1 and the unhealthy one is Person 2. Their x values are:

```
x1 = c(0, 35, 0)                                ①
x2 = c(1, 35, 35)                                ②
```

- ① good health person's data (bad_health_indicator=0,age=35,age*indicator=0)
- ② bad health person's (bad_health_indicator=1,age=35,age*indicator=35)

The posterior samples of the model parameters are stored in `mod_csim`:

```
head(mod_csim)
```

```
Markov Chain Monte Carlo (MCMC) output:  
Start = 1  
End = 7  
Thinning interval = 1  
      b_age    b_badh     b_intx      int  
[1,] 0.01101356 1.766364 -0.01663365 0.2669958  
[2,] 0.01062432 1.741655 -0.01495074 0.2844927  
[3,] 0.01101064 1.777060 -0.01583831 0.2471388  
[4,] 0.01114886 1.882064 -0.01625016 0.2397804  
[5,] 0.01047565 1.816620 -0.01642297 0.2425705  
[6,] 0.01024267 1.786422 -0.01452622 0.2630138  
[7,] 0.01047055 1.694525 -0.01436676 0.2642218
```

First, we'll compute the linear part of the predictor:

```
loglam1 = mod_csim[, "int"] + mod_csim[, c(2, 1, 3)] %*% x1  
loglam2 = mod_csim[, "int"] + mod_csim[, c(2, 1, 3)] %*% x2
```

Next we'll apply the inverse link:

```
lam1 = exp(loglam1)  
lam2 = exp(loglam2)
```

The final step is to use these samples for the λ parameter for each individual and simulate actual number of doctor visits using the likelihood:

```
(n_sim = length(lam1))
```

```
[1] 15000
```

we have distribution of 15000 samples of λ for each person.

```

plot(table(factor(y1, levels=0:18))/n_sim, pch=2, ylab="posterior prob.", xlab="visits")
points(table(y2+0.1)/n_sim, col="red")

```

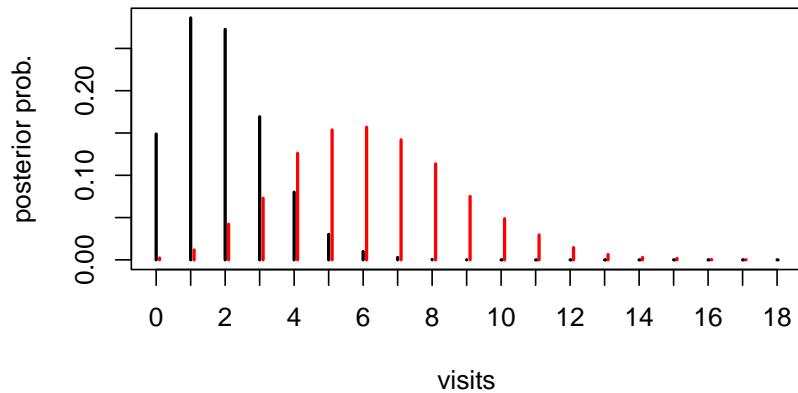


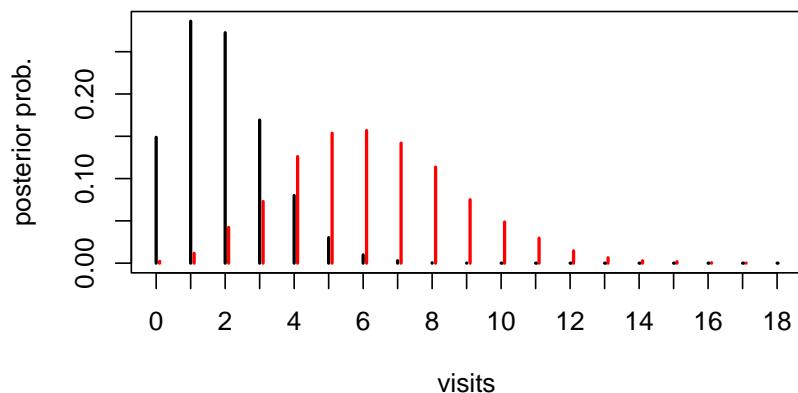
Figure 37.6

```

y1 = rpois(n=n_sim, lambda=lam1)                                     ①
y2 = rpois(n=n_sim, lambda=lam2)

plot(table(factor(y1, levels=0:18))/n_sim, pch=2, ylab="posterior prob.", xlab="visits")
points(table(y2+0.1)/n_sim, col="red")

```



Finally, we can answer the original question: What is the probability that the person with poor health will have more doctor visits than the person with good health?

```
mean(y2 > y1)
```

```
[1] 0.9186667
```

Because we used our posterior samples for the model parameters in our simulation (the `loglam1` and `loglam2` step above), this posterior predictive distribution on the number of visits for these two new individuals naturally account for our uncertainty in the model estimates. This is a more honest/realistic distribution than we would get if we had fixed the model parameters at their MLE or posterior means and simulated data for the new individuals.

37.4 Prior sensitivity analysis

When communicating results from any analysis, a responsible statistician will report and justify *modeling decisions*, especially assumptions. In a Bayesian analysis, there is an additional assumption that is open to scrutiny: the *choices of prior distributions*. In the models considered so far in this course, there are an infinite number of prior distributions we could have chosen from. When communicating results from any analysis, a responsible statistician will report and justify *modeling decisions*, especially assumptions. In a Bayesian analysis, there is another assumption that is open to scrutiny: the *choices of prior distributions*. In the models considered so far in this course, there are an infinite number of prior distributions we could have chosen from.

If they truly represent your beliefs about the parameters before analysis and the model is appropriate, then the posterior distribution truly represents your updated beliefs. If you don't have any strong beliefs beforehand, there are often default, reference, or non-informative prior options, and you will have to select one. However, a collaborator or a boss (indeed, somebody somewhere) may not agree with your choice of prior. One way to increase the credibility of your results is to repeat the analysis under a variety of priors, and report how the results differ as a result. This process is called *prior sensitivity analysis*.

At a minimum you should always report your choice of model and prior. If you include a sensitivity analysis, select one or more alternative priors and describe how the results of the analysis change. If they are sensitive to

Q. How do you justify the model you choose?

Q. How do you justify the priors you choose?

the choice of prior, you will likely have to explain both sets of results, or at least explain why you favor one prior over another. If the results are not sensitive to the choice of prior, this is evidence that the data are strongly driving the results. It suggests that different investigators coming from different backgrounds should come to the same conclusions.

If the purpose of your analysis is to establish a hypothesis, it is often prudent to include a “skeptical” prior which does not favor the hypothesis. Then, if the posterior distribution still favors the hypothesis despite the unfavorable prior, you will be able to say that the data substantially favor the hypothesis. This is the approach we will take in the following example, continued from the previous lesson.

37.4.1 Poisson regression example

Let’s return to the example of number of doctor visits. We concluded from our previous analysis of these data that both bad health and increased age are associated with more visits. Suppose the burden of proof that bad health is actually associated with more visits rests with us, and we need to convince a skeptic.

First, let’s re-run the original analysis and remind ourselves of the posterior distribution for the `badh` (bad health) indicator.

```
library("COUNT")
library("rjags")

data("badhealth")

mod_string = " model {
  for (i in 1:length(numvisit)) {
    numvisit[i] ~ dpois(lam[i])
    log(lam[i]) = int + b_badh*badh[i] + b_age*age[i] + b_intx*age[i]*badh[i]
  }

  int ~ dnorm(0.0, 1.0/1e6)
  b_badh ~ dnorm(0.0, 1.0/1e4)
  b_age ~ dnorm(0.0, 1.0/1e4)
  b_intx ~ dnorm(0.0, 1.0/1e4)
}
```

doctor visits

```

set.seed(102)

data_jags = as.list(badhealth)

params = c("int", "b_badh", "b_age", "b_intx")

mod = jags.model(textConnection(mod_string), data=data_jags, n.chains=3)

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
Observed stochastic nodes: 1127
Unobserved stochastic nodes: 4
Total graph size: 3665

Initializing model

update(mod, 1e3)

mod_sim = coda.samples(model=mod,
                       variable.names=params,
                       n.iter=5e3)
mod_csim = as.mcmc(do.call(rbind, mod_sim))

plot(density(mod_csim[, "b_badh"]))

```

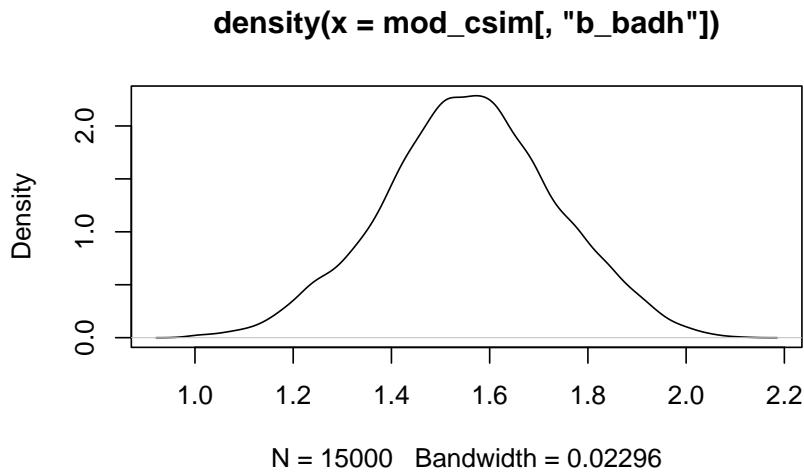


Figure 37.7

Essentially all of the posterior probability mass is above 0, suggesting that this coefficient is positive (and consequently that bad health is associated with more visits). We obtained this result using a relatively noninformative prior. What if we use a prior that strongly favors values near 0? Let's repeat the analysis with a normal prior on the badh coefficient that has mean 0 and standard deviation 0.2, so that the prior probability that the coefficient is less than 0.6 is > 0.998 . We'll also use a small variance on the prior for the interaction term involving badh (standard deviation 0.01 because this coefficient is on a much smaller scale).

```
mod2_string = " model {
  for (i in 1:length(numvisit)) {
    numvisit[i] ~ dpois(lam[i])
    log(lam[i]) = int + b_badh*badh[i] + b_age*age[i] + b_intx*age[i]*badh[i]
  }

  int ~ dnorm(0.0, 1.0/1e6)
  b_badh ~ dnorm(0.0, 1.0/0.2^2)
  b_age ~ dnorm(0.0, 1.0/1e4)
  b_intx ~ dnorm(0.0, 1.0/0.01^2)
}

mod2 = jags.model(textConnection(mod2_string), data=data_jags, n.chains=3)
```

Compiling model graph

```

Resolving undeclared variables
Allocating nodes
Graph information:
  Observed stochastic nodes: 1127
  Unobserved stochastic nodes: 4
  Total graph size: 3672

Initializing model

update(mod2, 1e3)

mod2_sim = coda.samples(model=mod2,
                        variable.names=params,
                        n.iter=5e3)
mod2_csim = as.mcmc(do.call(rbind, mod2_sim))

```

How did the posterior distribution for the coefficient of `bbadh` change?

```

curve(dnorm(x, mean=0.0, sd=sqrt(1e4)), from=-3.0, to=3.0, ylim=c(0.0, 3.0), lty=2,
      main="bbadh", ylab="density", xlab="bbadh")
curve(dnorm(x, mean=0.0, sd=0.2), from=-3.0, to=3.0, col="red", lty=2, add=TRUE)
lines(density(mod_csim[, "bbadh"]))
lines(density(mod2_csim[, "bbadh"]), col="red")
legend("topleft", legend=c("noninformative prior", "posterior", "skeptical prior", "posterior"),
       lty=c(2,1,2,1), col=rep(c("black", "red"), each=2), bty="n")

```

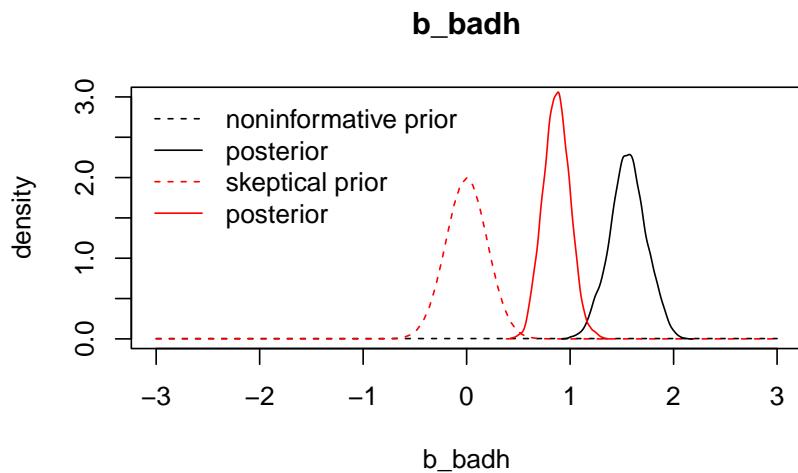


Figure 37.8

Under the skeptical prior, our posterior distribution for b_{badh} has significantly dropped to between about 0.6 and 1.1. Although the strong prior influenced our inference on the magnitude of the bad health effect on visits, it did not change the fact that the coefficient is significantly above 0. In other words: even under the skeptical prior, bad health is associated with more visits, with posterior probability near 1.

We should also check the effect of our skeptical prior on the interaction term involving both age and health.

```
curve(dnorm(x, mean=0.0, sd=sqrt(1e4)), from=-0.05, to=0.05, ylim=c(0.0, 140.0), lty=2,
      main="b_intx", ylab="density", xlab="b_intx")
curve(dnorm(x, mean=0.0, sd=0.01), from=-0.05, to=0.05, col="red", lty=2, add=TRUE)
lines(density(mod_csim[, "b_intx"]))
lines(density(mod2_csim[, "b_intx"]), col="red")
legend("topleft", legend=c("noninformative prior", "posterior", "skeptical prior", "posterior"),
       lty=c(2,1,2,1), col=rep(c("black", "red"), each=2), bty="n")
```

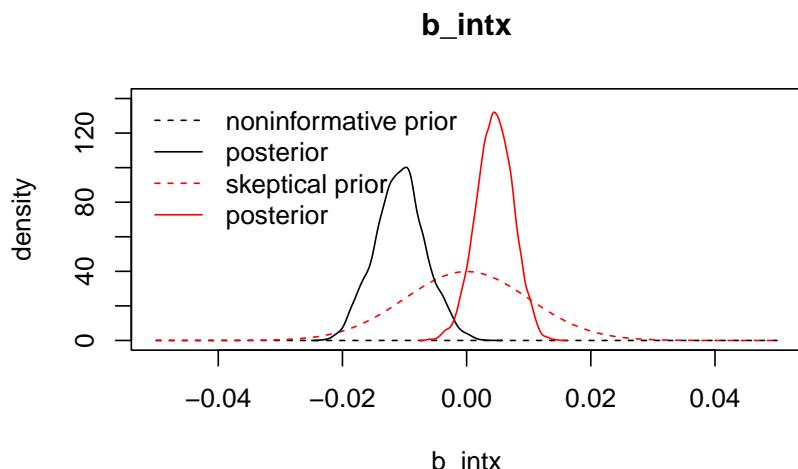


Figure 37.9

```
mean(mod2_csim[, "b_intx"] > 0) # posterior probability that b_intx is positive
```

```
[1] 0.9315333
```

The result here is interesting. Our estimate for the interaction coefficient has gone from negative under the non-informative prior to positive under

the skeptical prior, so the result is sensitive. In this case, because the skeptical prior shrinks away much of the bad health main effect, it is likely that this interaction effect attempts to restore some of the positive effect of bad health on visits. Thus, despite some observed prior sensitivity, our conclusion that bad health positively associates with more visits remains unchanged.

37.5 Overdispersed model

Recall that the **Negative Binomial** can be used to model overdispersed count data.

stan has three parameterizations for the **Negative Binomial**.

The first looks like similar to a binomial parameterization:

$$\text{NegBinomial}(y | \alpha, \beta) = \binom{y + \alpha - 1}{\alpha - 1} \left(\frac{\beta}{\beta + 1}\right)^{\alpha} \left(\frac{1}{\beta + 1}\right)^y.$$

$$\mathbb{E}[y] = \frac{\alpha}{\beta} \quad \text{and} \quad \text{Var}[Y] = \frac{\alpha}{\beta^2}(\beta + 1).$$

we can sample from this using the following statement

```
n ~ neg_binomial(alpha, beta)
```

But this parameterization if not a match to the Poisson model, so we move on

The second parametrization $\mu \in \mathbb{R}^+$ and $\phi \in \mathbb{R}^+$:

$$\text{NegBinomial2}(n | \mu, \phi) = \binom{n + \phi - 1}{n} \left(\frac{\mu}{\mu + \phi}\right)^n \left(\frac{\phi}{\mu + \phi}\right)^{\phi}$$

$$\mathbb{E}[n] = \mu \quad \text{and} \quad \text{Var}[n] = \mu + \frac{\mu^2}{\phi}$$

we can sample from this using the following statement

```
n ~ neg_binomial_2(mu, phi)
```

And there is a third parametrization

$$NegBinomial2Log(y | \mu, \phi) = NegBinomial2(y | \exp(\eta), \phi).$$

we can sample from this using the following statement:

```
y \sim **neg_binomial_2\_log**(y\|mu, phi)
```

jags has just one parameterization:

$$f(y | r, p) = \frac{\Gamma(y + r)}{\Gamma(r)\Gamma(y + 1)} p^r (1 - p)^y$$

We think of the Negative Binomial Distribution as the probability of completing y successful trials allowing for r failures in a sequence of $(y+r)$ Bernoulli trials where success is defined as drawing (with replacement) a white ball from an urn of white and black balls with a probability p of success.

$$\mathbb{E}[Y] = \mu = \frac{r(1-p)}{p} \quad \text{and} \quad \text{Var}[Y] = \mu + \frac{\mu^2}{r}$$

37.5.1 Transformations:

Since we want to have a model corresponding to a poisson regression we will transform the model as follows:

If we set $p = \frac{r}{r+\lambda}$ then the mean becomes : λ

and if we also set $r = \frac{\lambda^2}{\omega}$ then the variance becomes a sum of $\lambda + \omega$ where ω is our **over dispersion** term.

$$\omega = \lambda^2/r$$

$$\begin{aligned}
\mathbb{E}[Y] &= \frac{r(1-p)}{p} \\
&= rp^{-1} - r \\
&\stackrel{\text{sub } p}{=} \frac{r(\lambda + \lambda)}{r} - \lambda \\
&= \lambda \mathbb{V}\text{ar}[Y] \\
&= \frac{(1-p)r}{p^2} \\
&\stackrel{\text{sub } \lambda}{=} \frac{1}{p} \lambda \\
&\stackrel{\text{sub } p}{=} \lambda \frac{(r + \lambda)}{r} \\
&= \frac{\lambda r + \lambda^2}{r} \\
&= 1\lambda + \frac{\lambda^2}{r} \\
&\stackrel{\text{sub } \omega}{=} \lambda + \omega
\end{aligned}$$

Where we interpret λ as the mean and ω as the overdispersion

```

library("rjags")
library("COUNT")
data("badhealth")

mod3_string = "
model {
  for (i in 1:length(numvisit)) {
    mu[i]      = b0 + b_badh*badh[i] + b_age*age[i] + b_intx*age[i]*badh[i] # <1>
    lambda[i]  = exp(mu[i]) # <2>
    p[i]       = r / (r + lambda[i]) # <3>
    numvisit[i] ~ dnegbin(p[i], r) # <4>
    resid[i]   = numvisit[i] - p[i] # <5>
  }
  ## Priors
  b0        ~ dnorm(0.0, 1.0/1e6) # <6>
  b_badh   ~ dnorm(0.0, 1.0/0.2^2) # <7>
  b_age    ~ dnorm(0.0, 1.0/1e4) # <8>
  b_intx   ~ dnorm(0.0, 1.0/0.01^2) # <9>
  r ~ dunif(0,50) # <10>
}

```

```

## extra deterministic parameters
omega      <-  pow(mean(lambda),2)/2
#theta     <-  pow(1/mean(p),2) # <11>
#scale     <-  mean((1-p)/p) # <12>
}"
data3_jags = as.list(badhealth)
mod3 = jags.model(textConnection(mod3_string), data=data3_jags, n.chains=3)
update(mod3, 1e3)                                     (13)
params3 = c("b_intx", "b_badh", "b_age", 'over_disp', 'b0', 'omega', 'r')
mod3_sim = coda.samples(model=mod3, variable.names=params3, n.iter=5e3) (14)
mod3_csim = as.mcmc(do.call(rbind, mod3_sim))          (15)
(dic3 = dic.samples(mod3, n.iter=1e3))                (16)

```

- ① the linear part
- ② lambda corresponds to the parameter used in the Poisson regression
- ③ p is the success parameter
- ④ we draw from the negative binomial distribution
- ⑤ sampling using the parametrization of the Negative Binomial distribution.
- ⑥ normal prior for intercept b0
- ⑦ normal prior for b_badh
- ⑧ normal prior for b_age
- ⑨ normal prior for b_intx
- ⑩ uniform prior for over_disp - at the upper limit of 50 **NegBin** converges to **Poisson** see (Jackman 2009, 280)
- ⑪ theta param
- ⑫ scale param
- ⑬ burn in
- ⑭ sample
- ⑮ stack samples from the chains
- ⑯ estimate the DIC

```

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
  Observed stochastic nodes: 1127
  Unobserved stochastic nodes: 5
  Total graph size: 4204

```

```
Initializing model
```

```
Mean deviance: 4478  
penalty 3.896  
Penalized deviance: 4481
```

```
gelman.diag(mod3_sim )
```

```
Potential scale reduction factors:
```

	Point est.	Upper C.I.
b0	1.01	1.04
b_age	1.01	1.05
b_badh	1.00	1.01
b_intx	1.00	1.01
omega	1.00	1.00
r	1.00	1.00

```
Multivariate psrf
```

```
1.01
```

```
raftery.diag(mod3_sim)
```

```
[[1]]
```

```
Quantile (q) = 0.025  
Accuracy (r) = +/- 0.005  
Probability (s) = 0.95
```

	Burn-in (M)	Total (N)	Lower bound (Nmin)	Dependence factor (I)
b0	26	29146	3746	7.78
b_age	36	35298	3746	9.42
b_badh	12	12836	3746	3.43
b_intx	16	16894	3746	4.51
omega	2	3930	3746	1.05
r	5	6078	3746	1.62

[[2]]

```
Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95
```

	Burn-in (M)	Total (N)	Lower bound (Nmin)	Dependence factor (I)
b0	22	23947	3746	6.390
b_age	22	24038	3746	6.420
b_badh	14	13860	3746	3.700
b_intx	12	12362	3746	3.300
omega	2	3741	3746	0.999
r	5	5483	3746	1.460

[[3]]

```
Quantile (q) = 0.025
Accuracy (r) = +/- 0.005
Probability (s) = 0.95
```

	Burn-in (M)	Total (N)	Lower bound (Nmin)	Dependence factor (I)
b0	30	32809	3746	8.76
b_age	22	21906	3746	5.85
b_badh	14	15032	3746	4.01
b_intx	10	10032	3746	2.68
omega	2	3803	3746	1.02
r	4	5124	3746	1.37

```
autocorr.diag(mod3_sim)
```

	b0	b_age	b_badh	b_intx	omega	r
Lag 0	1.00000000	1.00000000	1.00000000	1.00000000	1.0000000000	1.0000000000
Lag 1	0.94000519	0.94265467	0.7868563	0.79541227	0.005291599	0.246809631
Lag 5	0.77352274	0.77672711	0.3635436	0.37874669	-	
	0.013409039	0.012200623				
Lag 10	0.61208995	0.61294603	0.1377969	0.14948495	0.004102791	0.004487142

```
Lag 50 0.08500864 0.09212251 -0.0282855 -0.02202781 0.007499163 -
0.007277463
```

```
effectiveSize(mod3_sim)
```

b0	b_age	b_badh	b_intx	omega	r
379.5981	375.5226	1528.4688	1416.3193	15690.7158	8885.4942

```
summary(mod3_sim)
```

```
Iterations = 2001:7000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000
```

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

	Mean	SD	Naive SE	Time-series SE
b0	0.456436	0.129394	1.056e-03	0.0066848
b_age	0.005926	0.003378	2.758e-05	0.0001754
b_badh	0.380068	0.169786	1.386e-03	0.0043674
b_intx	0.014829	0.004389	3.584e-05	0.0001167
omega	2.777371	0.219787	1.795e-03	0.0017572
r	0.987866	0.069564	5.680e-04	0.0007384

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
b0	0.2032633	0.369479	0.453904	0.544838	0.71066
b_age	-0.0006621	0.003653	0.005958	0.008211	0.01256
b_badh	0.0497521	0.263962	0.379141	0.492251	0.71030
b_intx	0.0063279	0.011884	0.014797	0.017808	0.02341
omega	2.3780145	2.622336	2.768973	2.920945	3.23699
r	0.8588172	0.939975	0.984258	1.033251	1.13059

```

par(mar = c(2.5, 1, 2.5, 1))
plot(mod3_sim, auto.layout = FALSE)

autocorr.plot(mod3_csim,auto.layout = FALSE)

X = as.matrix(badhealth[,-1])
X = cbind(X, with(badhealth, badh*age))
(pmed_coef = apply(mod3_csim, 2, median))

      b0        b_age       b_badh       b_intx       omega         r
0.453904434 0.005958067 0.379140544 0.014796628 2.768972527 0.984258005

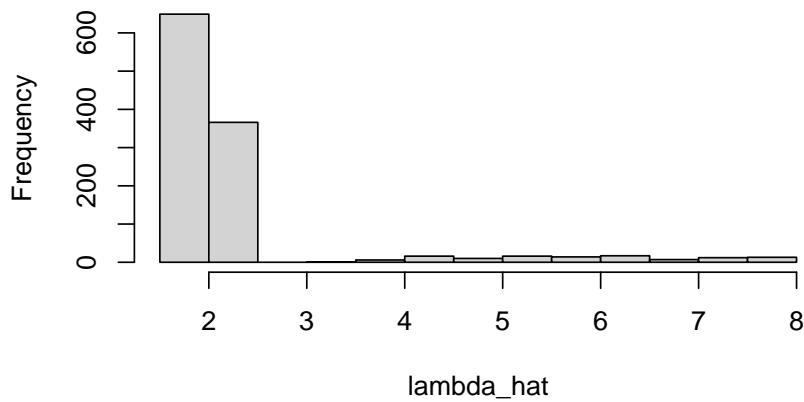
(r = pmed_coef["r"] )

r
0.984258

mu_hat = pmed_coef["b0"] + X %*% pmed_coef[c("b_badh", "b_age", "b_intx")]
lambda_hat = exp(mu_hat)
p_hat = r / (r + lambda_hat)
hist(lambda_hat)

```

Histogram of lambda_hat



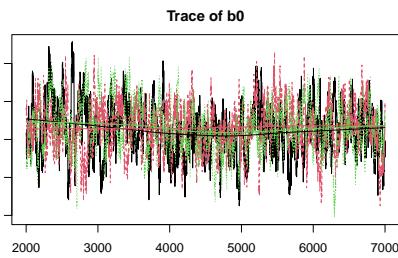


Figure 37.10

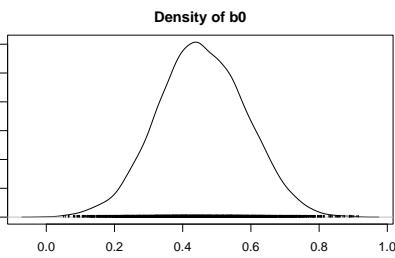


Figure 37.11

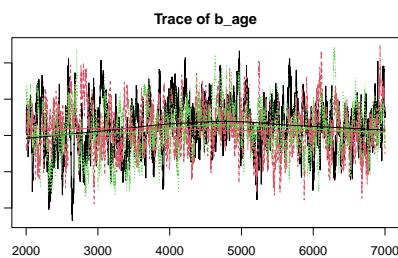


Figure 37.12

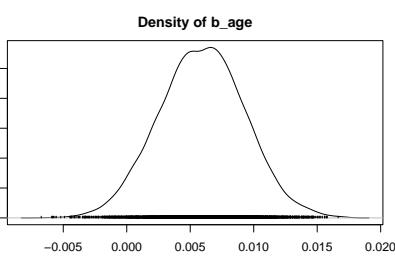


Figure 37.13

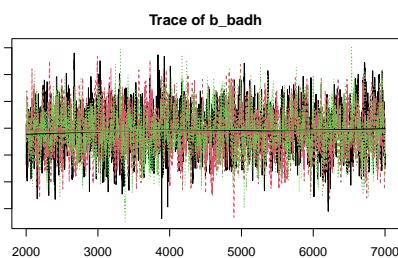


Figure 37.14

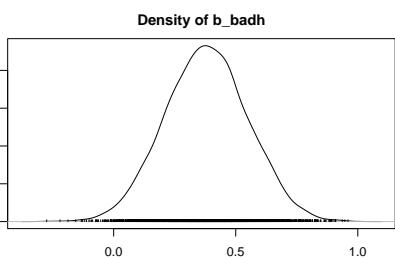


Figure 37.15

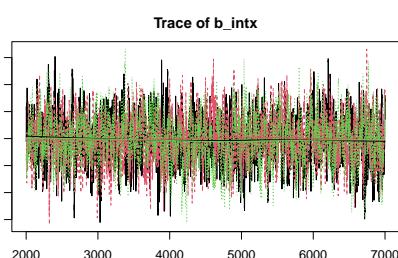


Figure 37.16

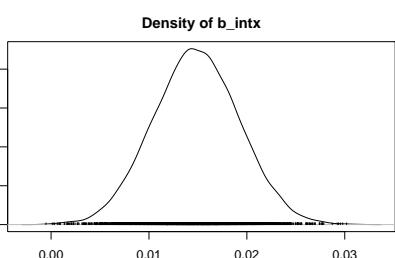


Figure 37.17

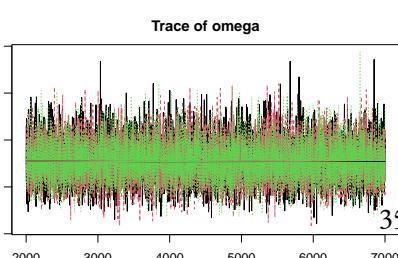


Figure 37.18

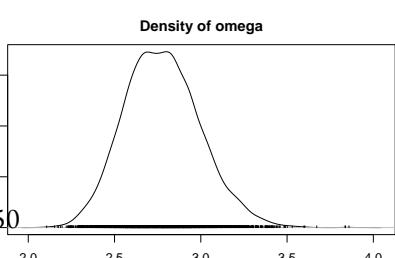
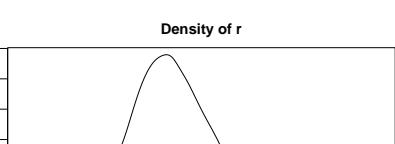
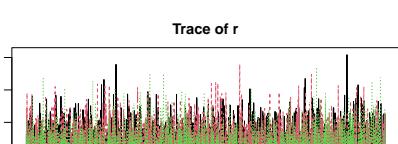


Figure 37.19



b0

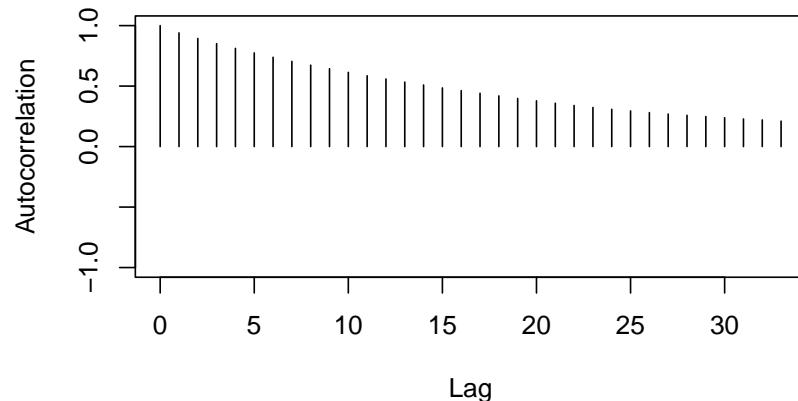


Figure 37.22

b_age

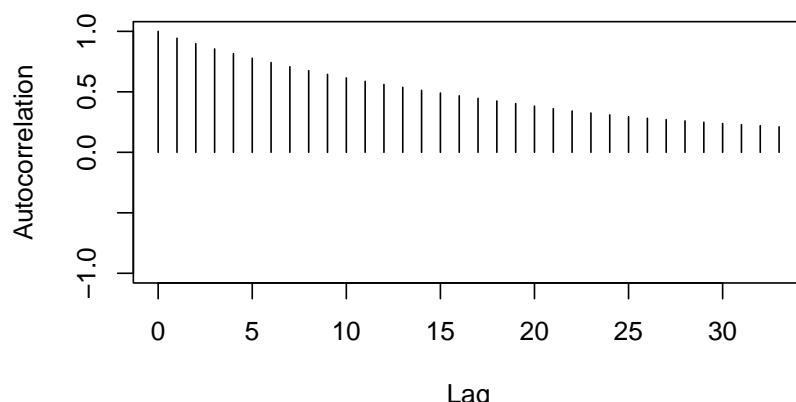
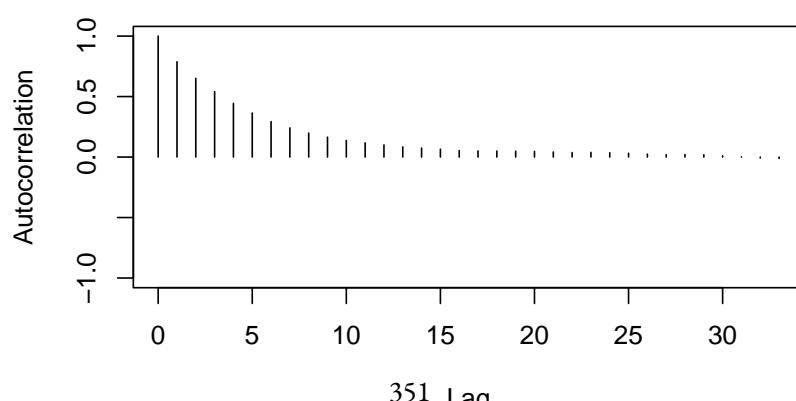


Figure 37.23

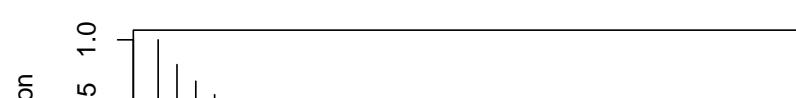
b_badh



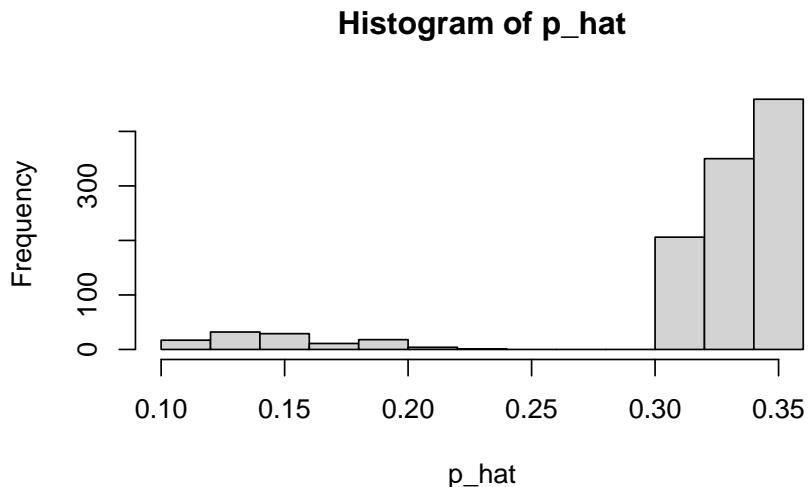
351 Lag

Figure 37.24

b_intx



```
hist(p_hat)
```



residuals

```
resid = badhealth$numvisit - p_hat  
head(resid)
```

```
[ ,1]  
[1,] 29.69325  
[2,] 19.68815  
[3,] 15.67523  
[4,] 19.69198  
[5,] 14.66069  
[6,] 14.65398
```

```
plot(resid) # the data were ordered
```

Table 37.1: First few rows of mod3_csim

```
head(mod3_csim)
```

Markov Chain Monte Carlo (MCMC) output:

```
Start = 1
End = 7
Thinning interval = 1
      b0      b_age     b_badh     b_intx     omega         r
[1,] 0.4349737 0.005312512 0.4311368 0.01437445 2.556055 0.9614930
[2,] 0.4227126 0.005330513 0.3119540 0.02044381 2.720177 0.9470283
[3,] 0.4241562 0.008260342 0.1266961 0.02088917 3.151760 0.9007646
[4,] 0.3203589 0.008707598 0.2078028 0.02033820 2.732001 1.0649393
[5,] 0.3956062 0.008783977 0.1700476 0.02155593 3.228081 0.9801937
[6,] 0.2934651 0.009195927 0.1492259 0.02102659 2.654456 0.9533749
[7,] 0.3387074 0.008299897 0.1485969 0.01824748 2.535815 1.0248507
```

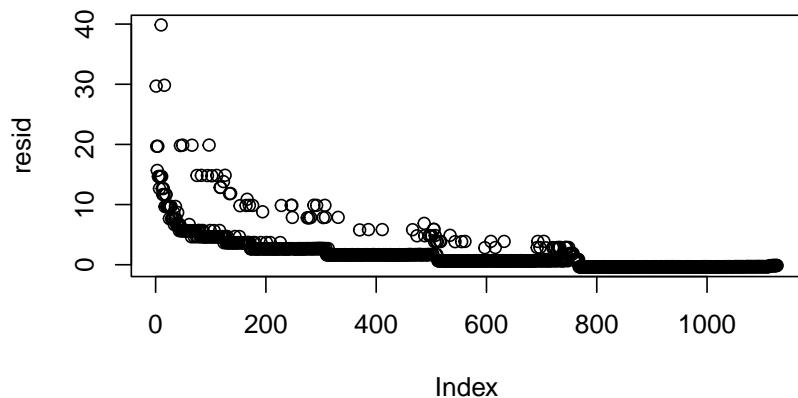


Figure 37.28: Plot of residuals

```
plot(p_hat, badhealth$numvisit)
abline(0.0, 1.0)
```

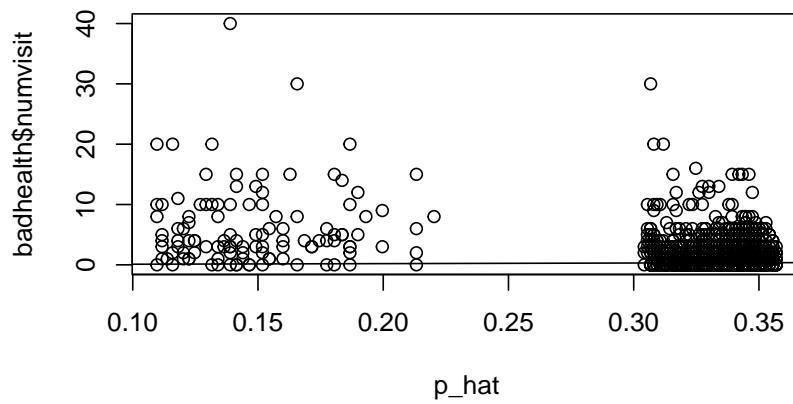


Figure 37.29: Plot of p_{hat} vs numvisit

```
plot(p_hat[which(badhealth$badh==0)], resid[which(badhealth$badh==0)], xlim=range(p_hat), ylab="residuals", xlab="p_hat")
points(p_hat[which(badhealth$badh==1)], resid[which(badhealth$badh==1)], col="red")
```

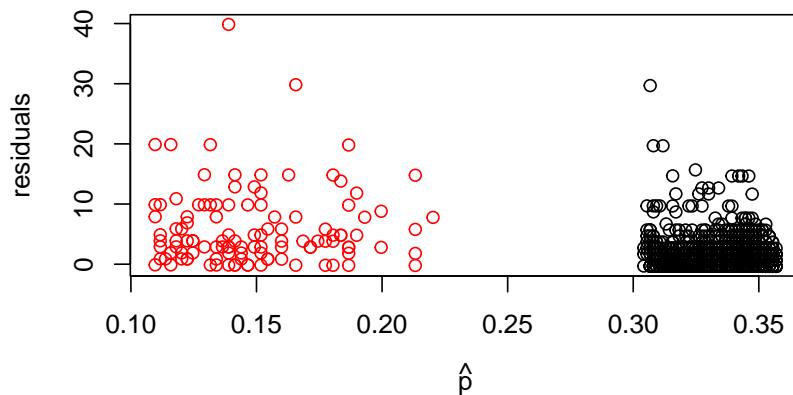


Figure 37.30: Plot of p_{hat} vs residuals, colored by health status

```
var(resid[which(badhealth$badh==0)])
```

```
[1] 7.061473
```

```
var(resid[which(badhealth$badh==1)])
```

```
[1] 41.21235
```

38 Hierarchical modeling

Bayesian Statistics: Techniques and Models

38.1 Introduction to Hierarchical modeling

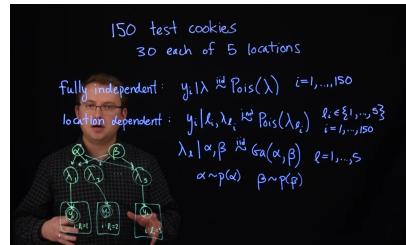


Figure 38.1: Introduction to Hierarchical modeling

38.2 Normal hierarchical model

Handout: [Normal hierarchical model](#)

38.3 Applications of hierarchical modeling

Handout: [Common applications of Bayesian hierarchical models](#)

38.4 Prior predictive simulation

38.4.1 Data

Let's fit our hierarchical model for counts of chocolate chips. The data can be found in `cookies.dat`.

```
dat = read.table(file="data/cookies.dat", header=TRUE)
head(dat)
```

```
  chips location
1     12       1
2     12       1
3      6       1
4     13       1
5     12       1
6     12       1
```

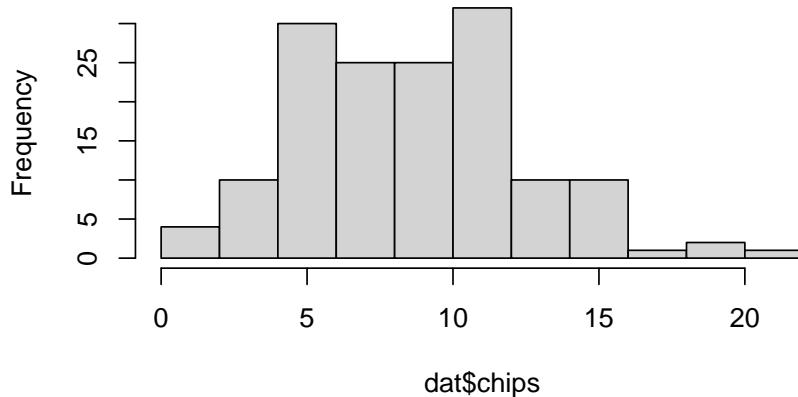
```
table(dat$location)
```

```
1 2 3 4 5
30 30 30 30 30
```

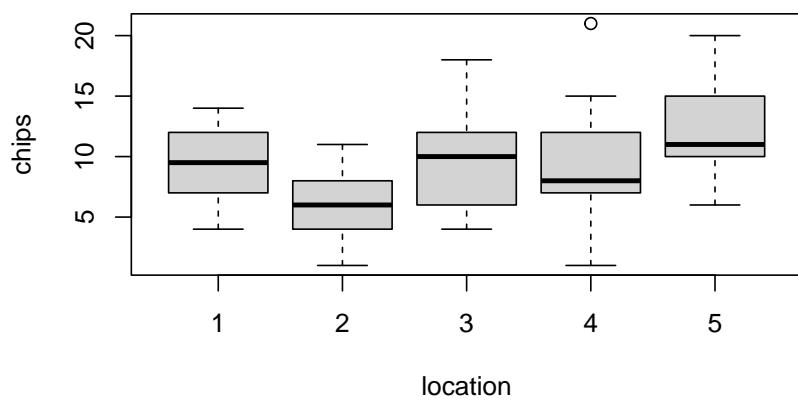
We can also visualize the distribution of chips by location.

```
hist(dat$chips)
```

Histogram of dat\$chips



```
boxplot(chips ~ location, data=dat)
```



38.4.2 Prior predictive checks

Before implementing the model, we need to select prior distributions for α and β , the hyperparameters governing the gamma distribution for the λ parameters. First, think about what the λ 's represent. For location j , λ_j is the expected number of chocolate chips per cookie. Hence, α and β control the distribution of these means between locations. The mean of this gamma distribution will represent the overall mean of number of chips for all cookies. The variance of this gamma distribution controls the variability between locations. If this is high, the mean number of chips will vary widely from location to location. If it is small, the mean number of chips will be nearly the same from location to location.

To see the effects of different priors on the distribution of λ 's, we can simulate. Suppose we try independent exponential priors for α and β .

```
set.seed(112)
n_sim = 500
alpha_pri = rexp(n_sim, rate=1.0/2.0)
beta_pri = rexp(n_sim, rate=5.0)
mu_pri = alpha_pri/beta_pri
sig_pri = sqrt(alpha_pri/beta_pri^2)

summary(mu_pri)

   Min. 1st Qu. Median     Mean 3rd Qu.    Max.
0.0213  2.9829  9.8522  61.1271 29.9801 4858.7861

summary(sig_pri)

   Min. 1st Qu. Median     Mean 3rd Qu.    Max.
0.1834  3.3663  8.5488  41.8137 22.2219 2865.6461
```

After simulating from the priors for α and β , we can use those samples to simulate further down the hierarchy:

```
lam_pri = rgamma(n=n_sim, shape=alpha_pri, rate=beta_pri)
summary(lam_pri)

   Min. 1st Qu. Median     Mean 3rd Qu.    Max.
0.000  1.171  7.668  83.062 28.621 11005.331

Or for a prior predictive reconstruction of the original data set:

(lam_pri = rgamma(n=5, shape=alpha_pri[1:5], rate=beta_pri[1:5]))
```

[1]	66.444084	9.946688	6.028319	15.922568	47.978587
-----	-----------	----------	----------	-----------	-----------

```
(y_pri = rpois(n=150, lambda=rep(lam_pri, each=30)))

[1] 63 58 64 63 70 62 61 48 71 73 70 77 66 60 72 77 69 62 66 71 49 80 66 75 74
[26] 55 62 90 65 57 12 9 7 10 12 10 11 7 14 13 9 6 6 13 7 10 12 9 9 10
[51] 7 8 6 9 7 10 13 13 8 12 6 10 3 6 7 4 6 7 5 5 4 3 6 2 8
[76] 4 8 4 5 7 1 4 5 3 8 8 3 1 7 3 16 14 13 17 17 12 13 13 16 16
[101] 15 14 11 10 13 17 16 19 16 17 15 16 7 17 21 16 12 15 14 13 52 44 51 46 39
[126] 40 40 44 46 59 45 49 58 42 31 52 43 47 53 41 48 57 35 60 51 58 36 34 41 59
```

Because these priors have high variance and are somewhat noninformative, they produce unrealistic predictive distributions. Still, enough data would overwhelm the prior, resulting in useful posterior distributions. Alternatively, we could tweak and simulate from these prior distributions until they adequately represent our prior beliefs. Yet another approach would be to re-parameterize the gamma prior, which we'll demonstrate as we fit the model.

38.5 JAGS Model

```
library("rjags")

Loading required package: coda

Linked to JAGS 4.3.2

Loaded modules: basemod, bugs

mod_string = " model {
  for (i in 1:length(chips)) {
    chips[i] ~ dpois(lam[location[i]])
  }

  for (j in 1:max(location)) {
    lam[j] ~ dgamma(alpha, beta)
  }
}
```

```

alpha = mu^2 / sig^2
beta = mu / sig^2

mu ~ dgamma(2.0, 1.0/5.0)
sig ~ dexp(1.0)

} "


set.seed(113)

data_jags = as.list(dat)

params = c("lam", "mu", "sig")

mod = jags.model(textConnection(mod_string), data=data_jags, n.chains=3)

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
Observed stochastic nodes: 150
Unobserved stochastic nodes: 7
Total graph size: 315

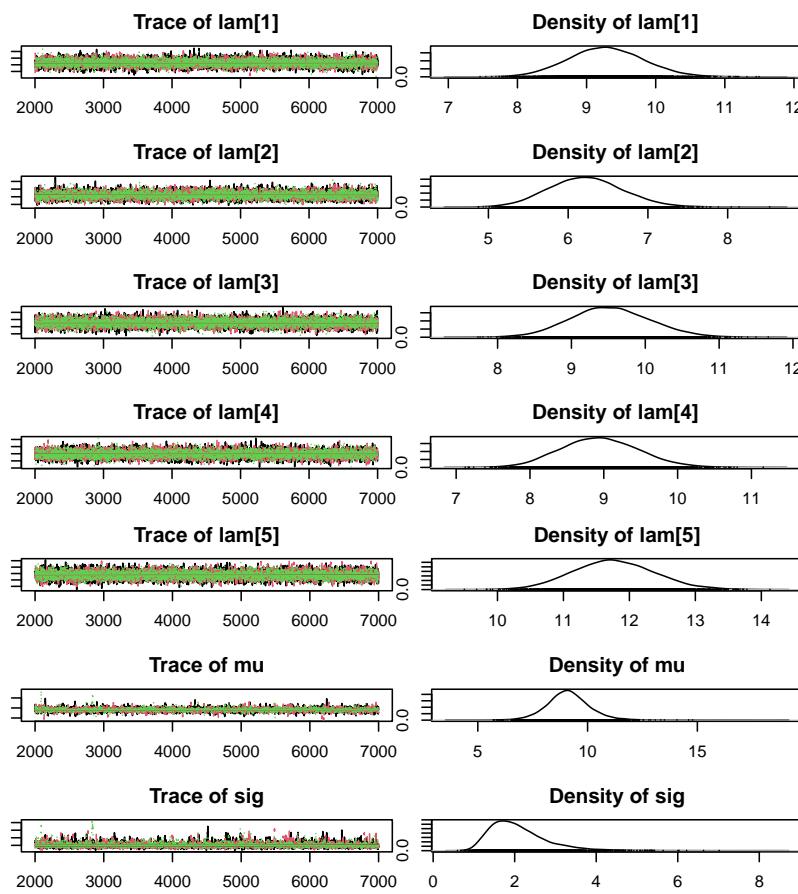
Initializing model

update(mod, 1e3)

mod_sim = coda.samples(model=mod,
                      variable.names=params,
                      n.iter=5e3)
mod_csim = as.mcmc(do.call(rbind, mod_sim))

## convergence diagnostics
par(mar = c(2.5, 1, 2.5, 1))
plot(mod_sim)

```



```
gelman.diag(mod_sim)
```

Potential scale reduction factors:

	Point est.	Upper C.I.
lam[1]	1	1
lam[2]	1	1
lam[3]	1	1
lam[4]	1	1
lam[5]	1	1
mu	1	1
sig	1	1

Multivariate psrf

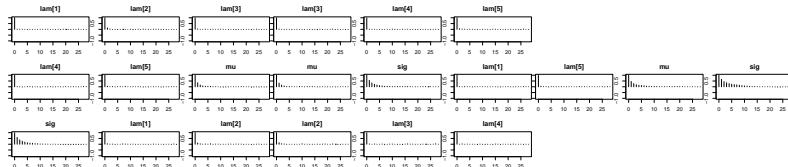
1

```
autocorr.diag(mod_sim)
```

	lam[1]	lam[2]	lam[3]	lam[4]	lam[5]
Lag 0	1.000000e+00	1.000000000	1.000000000	1.000000000	1.000000000
Lag 1	2.718360e-02	0.117943107	0.032116426	0.014305226	0.053618064
Lag 5	-5.059400e-03	-0.001026777	0.007701007	0.005318567	0.015405970
Lag 10	7.935742e-03	0.017488288	0.011526414	-0.004791243	0.007835478
Lag 50	-8.476924e-05	-0.009524519	-0.002533869	0.006072110	-0.008374683

	mu	sig
Lag 0	1.000000000	1.000000000
Lag 1	0.376559473	0.59942921
Lag 5	0.060267970	0.15608056
Lag 10	0.009051459	0.04120084
Lag 50	-0.017243718	-0.02962639

```
par(mar = c(2.5, 1, 2.5, 1))
autocorr.plot(mod_sim)
```



```
effectiveSize(mod_sim)
```

lam[1]	lam[2]	lam[3]	lam[4]	lam[5]	mu	sig
14381.268	10868.917	13743.919	14600.954	13471.183	5916.463	2979.507

```
## compute DIC
dic = dic.samples(mod, n.iter=1e3)
```

38.5.1 Model checking

After assessing convergence, we can check the fit via residuals. With a hierarchical model, there are now two levels of residuals: the observation

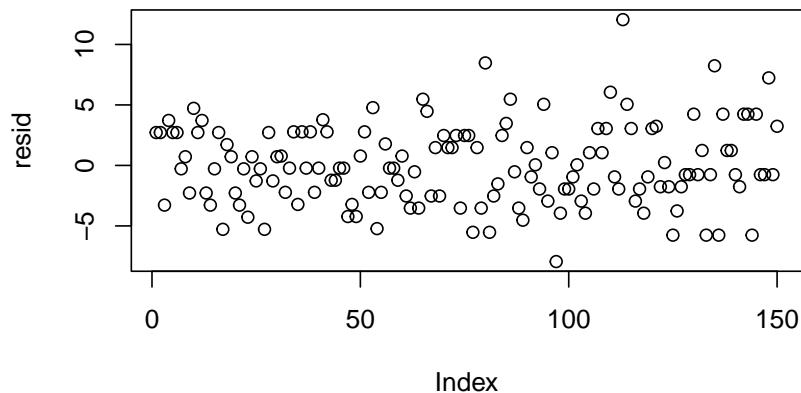
level and the location mean level. To simplify, we'll look at the residuals associated with the posterior means of the parameters.

First, we have observation residuals, based on the estimates of location means.

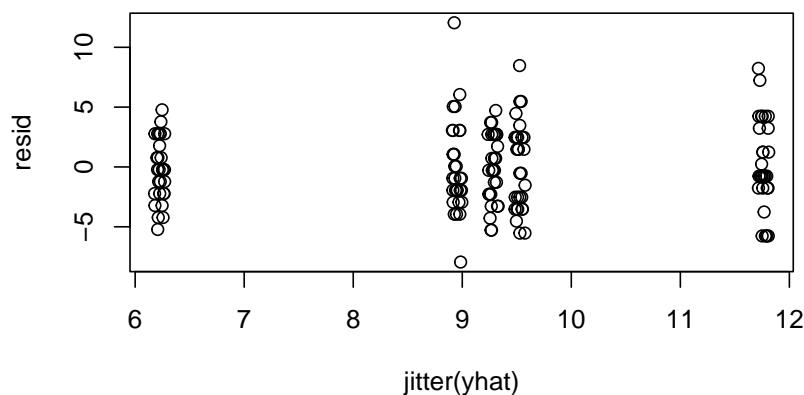
```
## observation level residuals
(pm_params = colMeans(mod_csim))

  lam[1]      lam[2]      lam[3]      lam[4]      lam[5]          mu        sig
 9.282293   6.224896   9.527672   8.949026  11.764913  9.119918  2.091156

yhat = rep(pm_params[1:5], each=30)
resid = dat$chips - yhat
plot(resid)
```



```
plot(jitter(yhat), resid)
```



```
var(resid[yhat<7])
```

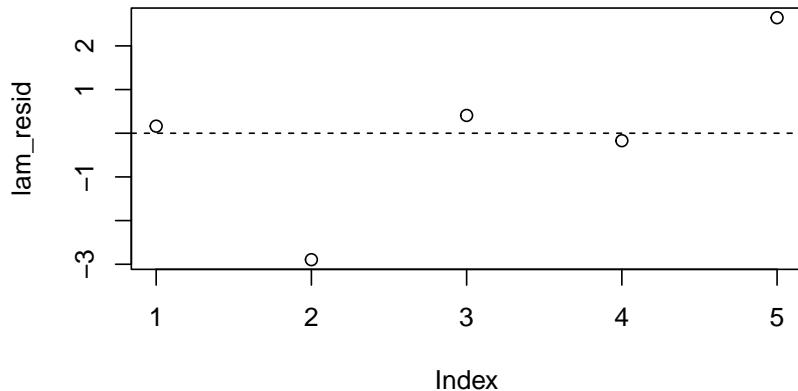
```
[1] 6.447126
```

```
var(resid[yhat>11])
```

```
[1] 13.72414
```

Also, we can look at how the location means differ from the overall mean μ .

```
## location level residuals
lam_resid = pm_params[1:5] - pm_params["mu"]
plot(lam_resid)
abline(h=0, lty=2)
```



We don't see any obvious violations of our model assumptions.

38.5.2 Results

```
summary(mod_sim)
```

```
Iterations = 2001:7000
Thinning interval = 1
```

```
Number of chains = 3
Sample size per chain = 5000
```

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

	Mean	SD	Naive SE	Time-series SE
lam[1]	9.282	0.5395	0.004405	0.004501
lam[2]	6.225	0.4623	0.003774	0.004440
lam[3]	9.528	0.5412	0.004419	0.004618
lam[4]	8.949	0.5294	0.004322	0.004383
lam[5]	11.765	0.6161	0.005030	0.005308
mu	9.120	1.0027	0.008187	0.013648
sig	2.091	0.7342	0.005995	0.014037

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
lam[1]	8.253	8.916	9.274	9.641	10.364
lam[2]	5.355	5.908	6.214	6.527	7.160
lam[3]	8.502	9.161	9.514	9.885	10.616
lam[4]	7.950	8.585	8.938	9.302	10.023
lam[5]	10.594	11.345	11.750	12.174	13.025
mu	7.204	8.507	9.089	9.694	11.181
sig	1.110	1.581	1.955	2.434	3.859

38.6 Posterior predictive simulation

Just as we did with the prior distribution, we can use these posterior samples to get Monte Carlo estimates that interest us from the posterior predictive distribution.

For example, we can use draws from the posterior distribution of μ and σ to simulate the posterior predictive distribution of the mean for a new location.

```
(n_sim = nrow(mod_csim))
```

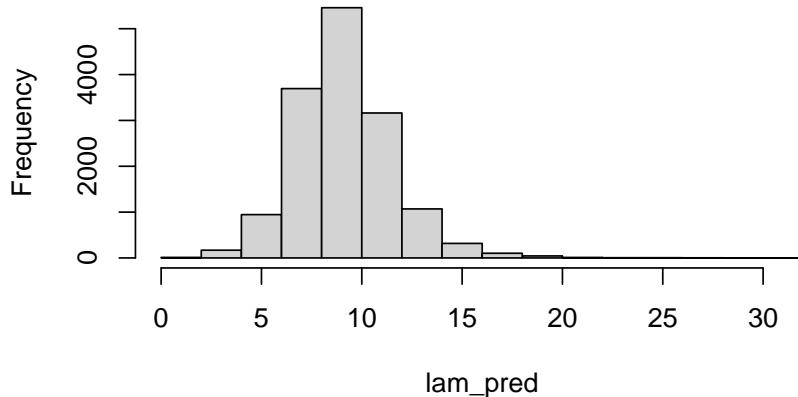
```
[1] 15000
```

```

lam_pred = rgamma(n=n_sim, shape=mod_csim[, "mu"]^2/mod_csim[, "sig"]^2,
                  rate=mod_csim[, "mu"]/mod_csim[, "sig"]^2)
hist(lam_pred)

```

Histogram of lam_pred



```
mean(lam_pred > 15)
```

```
[1] 0.01973333
```

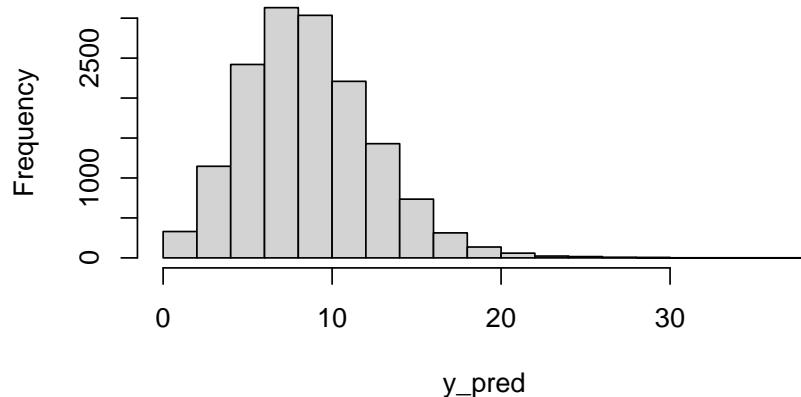
Using these λ draws, we can go to the observation level and simulate the number of chips per cookie, which takes into account the uncertainty in λ :

```

y_pred = rpois(n=n_sim, lambda=lam_pred)
hist(y_pred)

```

Histogram of y_pred

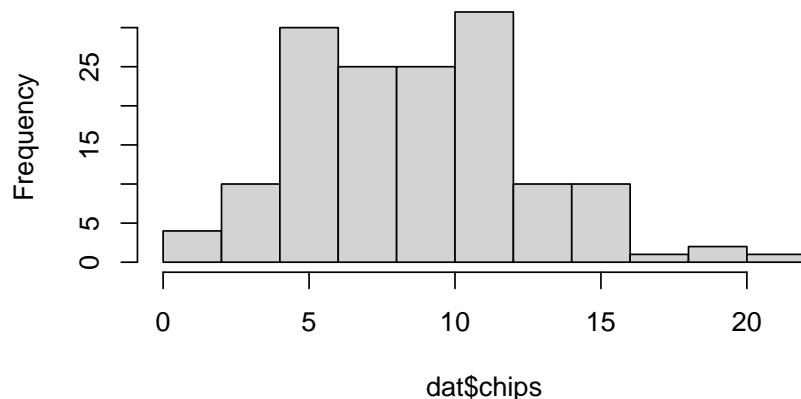


```
mean(y_pred > 15)
```

```
[1] 0.0574
```

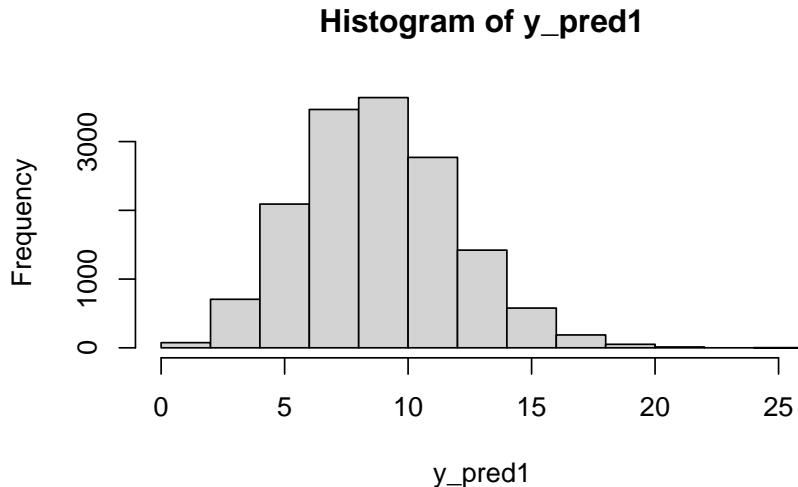
```
hist(dat$chips)
```

Histogram of dat\$chips



Finally, we could answer questions like: what is the posterior probability that the next cookie produced in Location 1 will have fewer than seven chips?

```
y_pred1 = rpois(n=n_sim, lambda=mod_csim[, "lam[1]"])
hist(y_pred1)
```



```
mean(y_pred1 < 7)
```

```
[1] 0.1914667
```

38.7 Random intercept linear model

We can extend the linear model for the Leinhardt data on infant mortality by incorporating the region variable. We'll do this with a hierarchical model, where each region has its own intercept.

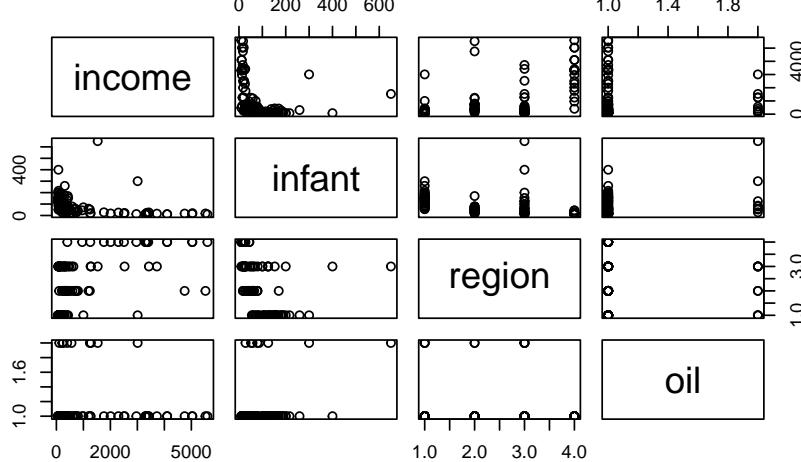
```
library("car")
```

```
Loading required package: carData
```

```
data("Leinhardt")
?Leinhardt
str(Leinhardt)
```

```
'data.frame': 105 obs. of 4 variables:
$ income: int 3426 3350 3346 4751 5029 3312 3403 5040 2009 2298 ...
$ infant: num 26.7 23.7 17 16.8 13.5 10.1 12.9 20.4 17.8 25.7 ...
$ region: Factor w/ 4 levels "Africa","Americas",...: 3 4 4 2 4 4 4 4 4 ...
$ oil    : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 1 ...
```

```
pairs(Leinhardt)
```



```
head(Leinhardt)
```

	income	infant	region	oil
Australia	3426	26.7	Asia	no
Austria	3350	23.7	Europe	no
Belgium	3346	17.0	Europe	no
Canada	4751	16.8	Americas	no
Denmark	5029	13.5	Europe	no
Finland	3312	10.1	Europe	no

Previously, we worked with infant mortality and income on the logarithmic scale. Recall also that we had to remove some missing data.

```
dat = na.omit(Leinhardt)
dat$logincome = log(dat$income)
dat$loginfant = log(dat$infant)
str(dat)
```

```
'data.frame': 101 obs. of 6 variables:
$ income   : int 3426 3350 3346 4751 5029 3312 3403 5040 2009 2298 ...
$ infant    : num 26.7 23.7 17 16.8 13.5 10.1 12.9 20.4 17.8 25.7 ...
$ region    : Factor w/ 4 levels "Africa","Americas",..: 3 4 4 2 4 4 4 4 4 ...
$ oil        : Factor w/ 2 levels "no","yes": 1 1 1 1 1 1 1 1 1 ...
$ logincome: num 8.14 8.12 8.12 8.47 8.52 ...
$ loginfant: num 3.28 3.17 2.83 2.82 2.6 ...
- attr(*, "na.action")= 'omit' Named int [1:4] 24 83 86 91
..- attr(*, "names")= chr [1:4] "Iran" "Haiti" "Laos" "Nepal"
```

Now we can fit the proposed model:

```
library("rjags")

mod_string = " model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu[i], prec)
    mu[i] = a[region[i]] + b[1]*log_income[i] + b[2]*is_oil[i]
  }

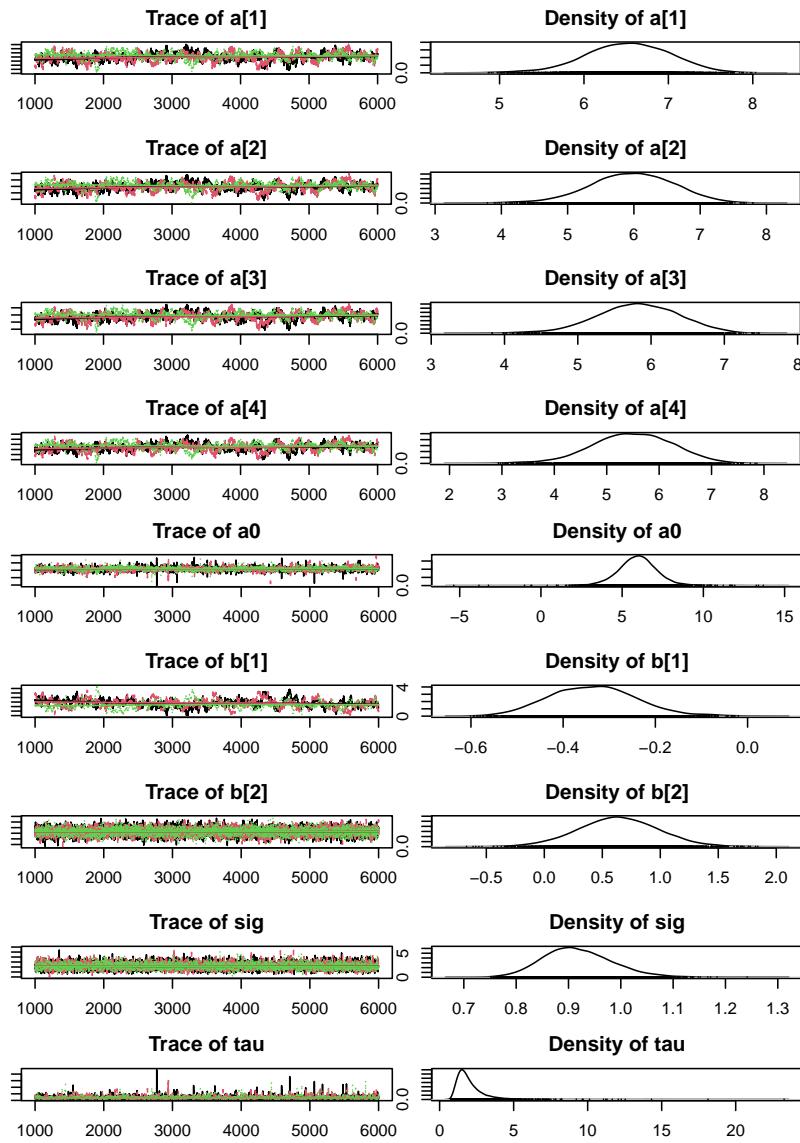
  for (j in 1:max(region)) {
    a[j] ~ dnorm(a0, prec_a)
  }

  a0 ~ dnorm(0.0, 1.0/1.0e6)
  prec_a ~ dgamma(1/2.0, 1*10.0/2.0)
  tau = sqrt( 1.0 / prec_a )

  for (j in 1:2) {
    b[j] ~ dnorm(0.0, 1.0/1.0e6)
  }

  prec ~ dgamma(5/2.0, 5*10.0/2.0)
  sig = sqrt( 1.0 / prec )
} "

set.seed(116)
data_jags = list(y=dat$loginfant, log_income=dat$logincome,
                  is_oil=as.numeric(dat$oil=="yes"), region=as.numeric(dat$region))
data_jags$is_oil
```

```
gelman.diag(mod_sim)
```

Potential scale reduction factors:

	Point est.	Upper C.I.
$a[1]$	1.01	1.03
$a[2]$	1.01	1.02
$a[3]$	1.01	1.02

```
a[4]      1.01      1.02
a0       1.00      1.00
b[1]      1.01      1.03
b[2]      1.00      1.00
sig       1.00      1.00
tau       1.00      1.00
```

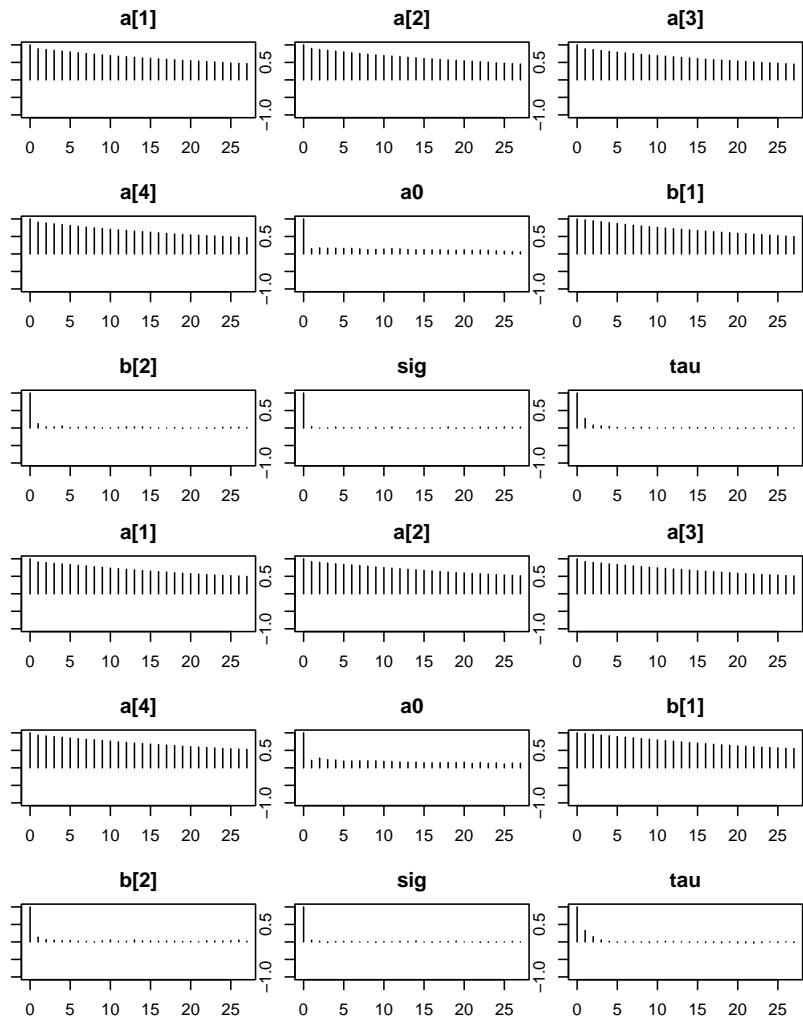
Multivariate psrf

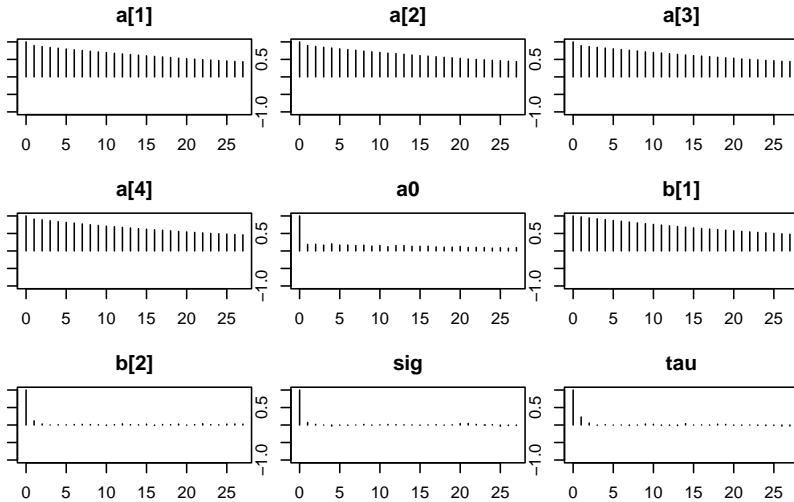
1.01

```
autocorr.diag(mod_sim)
```

	a[1]	a[2]	a[3]	a[4]	a0	b[1]	b[2]
Lag 0	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000
Lag 1	0.8979228	0.9024408	0.8992793	0.9168758	0.18448647	0.9744659	0.12653280
Lag 5	0.8094857	0.8121086	0.8090285	0.8263848	0.17172413	0.8781236	0.01880542
Lag 10	0.7077581	0.7144681	0.7102967	0.7259236	0.15974494	0.7709403	0.01885124
Lag 50	0.2689329	0.2768163	0.2667309	0.2879667	0.06517909	0.2984249	0.01447915
	sig	tau					
Lag 0	1.000000000	1.000000000					
Lag 1	0.053762960	0.2761627397					
Lag 5	0.008076585	0.0025534594					
Lag 10	0.006956894	0.0109173671					
Lag 50	0.001453875	-0.0006687977					

```
autocorr.plot(mod_sim)
```





```
effectiveSize(mod_sim)
```

a[1]	a[2]	a[3]	a[4]	a0	b[1]	b[2]
222.8324	217.4593	214.7296	212.2778	1177.0929	193.9683	9353.9871
sig	tau					
13471.2759	8051.4769					

38.7.1 Results

Convergence looks okay, so let's compare this with the old model from Lesson 7 using DIC:

```
dic.samples(mod, n.ite=1e3)
```

```
Mean deviance: 213.7
penalty 6.979
Penalized deviance: 220.6
```

```
### nonhierarchical model: 230.1
```

It appears that this model is an improvement over the non-hierarchical one we fit earlier. Notice that the penalty term, which can be interpreted as the “effective” number of parameters, is less than the actual number of parameters (nine). There are fewer “effective” parameters because they are

“sharing” information or “borrowing strength” from each other in the hierarchical structure. If we had skipped the hierarchy and fit one intercept, there would have been four parameters. If we had fit separate, independent intercepts for each region, there would have been seven parameters (which is close to what we ended up with).

Finally, let’s look at the posterior summary.

```
summary(mod_sim)
```

```
Iterations = 1001:6000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000
```

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

	Mean	SD	Naive SE	Time-series SE
a[1]	6.5200	0.50313	0.0041080	0.0336774
a[2]	5.9691	0.63158	0.0051568	0.0432297
a[3]	5.8152	0.56259	0.0045935	0.0382712
a[4]	5.4881	0.76956	0.0062835	0.0527488
a0	5.9553	1.26198	0.0103040	0.0387046
b[1]	-0.3350	0.09482	0.0007742	0.0067916
b[2]	0.6376	0.34756	0.0028378	0.0036998
sig	0.9179	0.06603	0.0005391	0.0005691
tau	2.0428	1.06771	0.0087178	0.0119966

2. Quantiles for each variable:

	2.5%	25%	50%	75%	97.5%
a[1]	5.47631	6.1893	6.5309	6.8718	7.4590
a[2]	4.67430	5.5546	5.9818	6.4079	7.1517
a[3]	4.65978	5.4500	5.8241	6.2064	6.8698
a[4]	3.88723	4.9825	5.5014	6.80208	6.9237
a0	3.49372	5.2125	5.9629	6.6887	8.4333
b[1]	-0.51029	-0.4023	-0.3364	-0.2732	-0.1398
b[2]	-0.04946	0.4082	0.6359	0.8689	1.3219
sig	0.80036	0.8719	0.9127	0.9598	1.0598

```
tau    0.97325 1.4104 1.7737 2.3424 4.6572
```

In this particular model, the intercepts do not have a real interpretation because they correspond to the mean response for a country that does not produce oil and has \$0 log-income per capita (which is \$1 income per capita). We can interpret a_0 as the overall mean intercept and τ as the standard deviation of intercepts across regions.

38.7.2 Other models

We have not investigated adding interaction terms, which might be appropriate. We only considered adding hierarchy on the intercepts, but in reality nothing prevents us from doing the same for other terms in the model, such as the coefficients for income and oil. We could try any or all of these alternatives and see how the DIC changes for those models. This, together with other model checking techniques we have discussed could be used to identify your *best* model that you can use to make inferences and predictions.

38.8 Mixture models

Histograms of data often reveal that they do not follow any standard probability distribution. Sometimes we have explanatory variables (or covariates) to account for the different values, and normally distributed errors are adequate, as in normal regression. However, if we only have the data values themselves and no covariates, we might have to fit a non-standard distribution to the data. One way to do this is by mixing standard distributions.

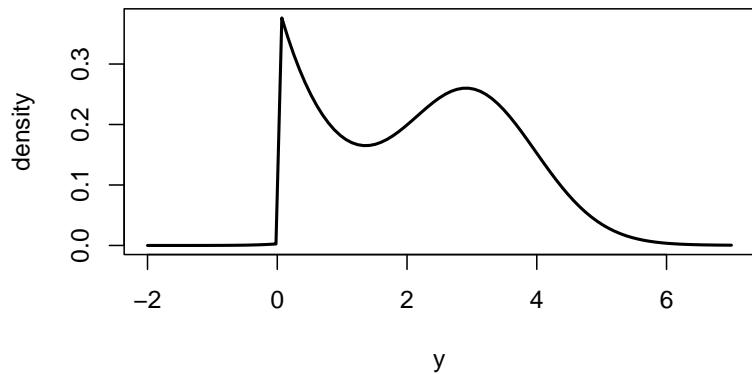
Mixture distributions are just a weighted combination of probability distributions. For example, we could take an exponential distribution with mean 1 and normal distribution with mean 3 and variance 1 (although typically the two mixture components would have the same support; here the exponential component has to be non-negative and the normal component can be positive or negative). Suppose we give them weights: 0.4 for the exponential distribution and 0.6 for the normal distribution. We could write the PDF for this distribution as

$$\Pr(y) = 0.4 \cdot \exp(-y) \cdot \mathbb{I}_{(y \geq 0)} + 0.6 \cdot \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}(y-3)^2\right)$$

The PDF of this mixture distribution would look like this:

```
curve( 0.4*dexp(x, 1.0) + 0.6*dnorm(x, 3.0, 1.0), from=-2.0, to=7.0, ylab="density", xlab="y", main="40/60 mixture")
```

40/60 mixture of exponential and normal distributions

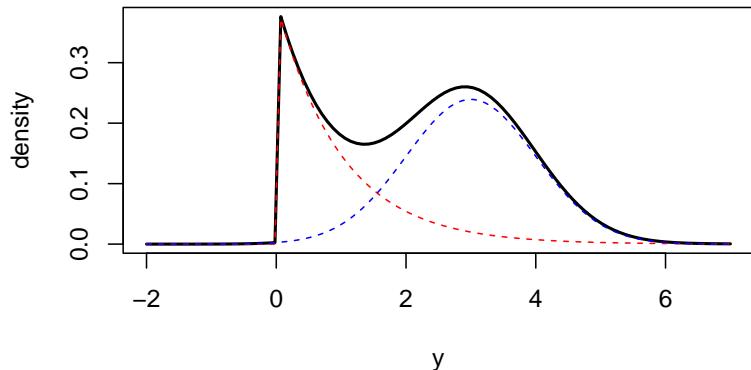


We could think of these two distributions as governing two distinct populations, one following the exponential distribution and the other following the normal distribution.

Let's draw the weighted PDFs for each population.

```
curve( 0.4*dexp(x, 1.0) + 0.6*dnorm(x, 3.0, 1.0), from=-2.0, to=7.0, ylab="density", xlab="y", main="40/60 mixture")
curve( 0.4*dexp(x, 1.0), from=-2.0, to=7.0, col="red", lty=2, add=TRUE)
curve( 0.6*dnorm(x, 3.0, 1.0), from=-2.0, to=7.0, col="blue", lty=2, add=TRUE)
```

40/60 mixture of exponential and normal distributions



The general form for a discrete mixture of distributions is as follows:

$$\Pr(y) = \sum_{j=1}^J \omega_j \cdot f_j(y)$$

where the ω 's are positive weights that add up to 1 (they are probabilities) and each of the $J f_j(y)$ functions is a PDF for some distribution. In the example above, the weights were 0.4 and 0.6, f_1 was an exponential PDF and f_2 was a normal PDF.

One way to simulate from a mixture distribution is with a hierarchical model. We first simulate an indicator for which “population” the next observation will come from using the weights ω . Let's call this z_i . In the example above, z_i would take the value 1 (indicating the exponential distribution) with probability 0.4 and 2 (indicating the normal distribution) with probability 0.6. Next, simulate the observation y_i from the distribution corresponding to z_i .

Let's simulate from our example mixture distribution.

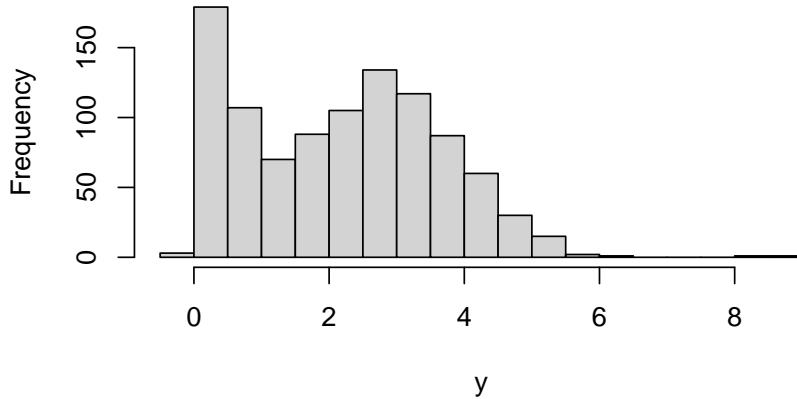
```
set.seed(117)
n = 1000
z = numeric(n)
y = numeric(n)
for (i in 1:n) {
  z[i] = sample.int(2, 1, prob=c(0.4, 0.6)) # returns a 1 with probability 0.4, or a 2 with probability 0.6
  if (z[i] == 1) {
    y[i] = rexp(1, rate=1.0)
  } else if (z[i] == 2) {
```

```

y[i] = rnorm(1, mean=3.0, sd=1.0)
}
}
hist(y, breaks=30)

```

Histogram of y



If we keep only the y values and throw away the z values, we have a sample from the mixture model above. To see that they are equivalent, we can marginalize the joint distribution of y and z :

$$\Pr(y) = \sum_{j=1}^2 \Pr(y, z=j) = \sum_{j=1}^2 \Pr(z=j) \cdot \Pr(y | z=j) = \sum_{j=1}^2 \omega_j \cdot f_j(y)$$

38.8.1 Bayesian inference for mixture models

When we fit a mixture model to data, we usually only have the y values and do not know which “population” they belong to. Because the z variables are unobserved, they are called *latent* variables. We can treat them as parameters in a hierarchical model and perform Bayesian inference for them. The hierarchical model might look like this:

$$\begin{aligned}
y_i | z_i, \theta &\stackrel{\text{ind}}{\sim} f_{z_i}(y | \theta), \quad i = 1, \dots, n \\
\Pr(z_i = j | \omega) &= \omega_j, \quad j = 1, \dots, J \\
\omega &\sim \Pr(\omega) \\
\theta &\sim \Pr(\theta)
\end{aligned}$$

where we might use a Dirichlet prior (see the review of distributions in the supplementary material) for the weight vector ω and conjugate priors for the population-specific parameters in θ . With this model, we could obtain posterior distributions for z (population membership of the observations), ω (population weights), and θ (population-specific parameters in f_j). Next, we will look at how to fit a mixture of two normal distributions in JAGS.

38.9 Example with JAGS

38.9.1 Data

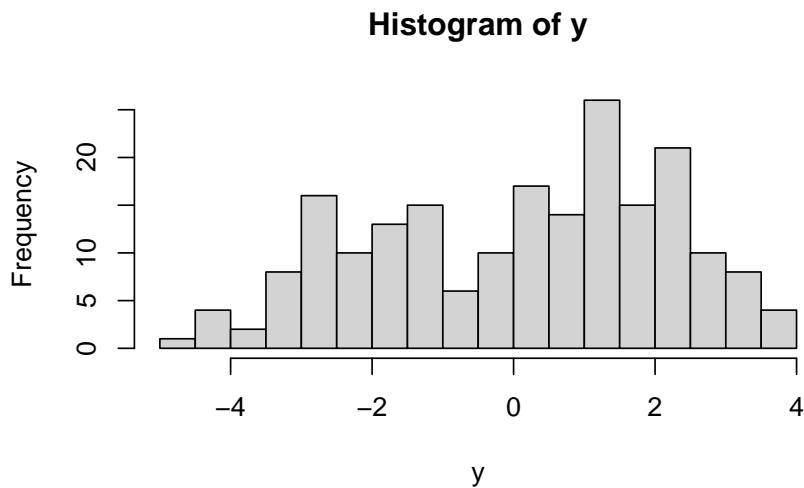
For this example, we will use the data in the attached file `mixture.csv`.

```
dat = read.csv("data/mixture.csv", header=FALSE)
y = dat$V1
n = length(y)
```

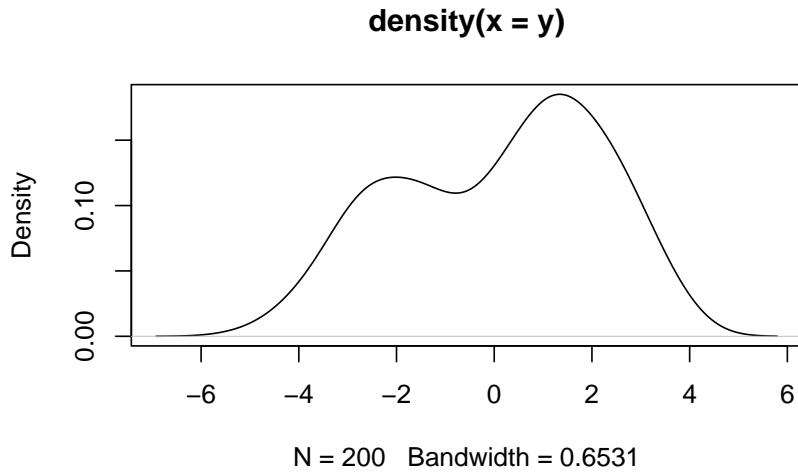
```
[1] 200
```

Let's visualize these data.

```
hist(y, breaks=20)
```



```
plot(density(y))
```



It appears that we have two populations, but we do not know which population each observation belongs to. We can learn this, along with the mixture weights and population-specific parameters with a Bayesian hierarchical model.

We will use a mixture of two normal distributions with variance 1 and different (and unknown) means.

38.9.2 Model

```
library("rjags")

mod_string = " model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu[z[i]], prec)
    z[i] ~ dcat(omega)
  }

  mu[1] ~ dnorm(-1.0, 1.0/100.0)
  mu[2] ~ dnorm(1.0, 1.0/100.0) T(mu[1],) # ensures mu[1] < mu[2]

  prec ~ dgamma(1.0/2.0, 1.0*1.0/2.0)
  sig = sqrt(1.0/prec)
```

```

    omega ~ ddirich(c(1.0, 1.0))
}

set.seed(11)

data_jags = list(y=y)

params = c("mu", "sig", "omega", "z[1]", "z[31]", "z[49]", "z[6]") # Select some z's to monitor

mod = jags.model(textConnection(mod_string), data=data_jags, n.chains=3)

Compiling model graph
Resolving undeclared variables
Allocating nodes
Graph information:
Observed stochastic nodes: 200
Unobserved stochastic nodes: 204
Total graph size: 614

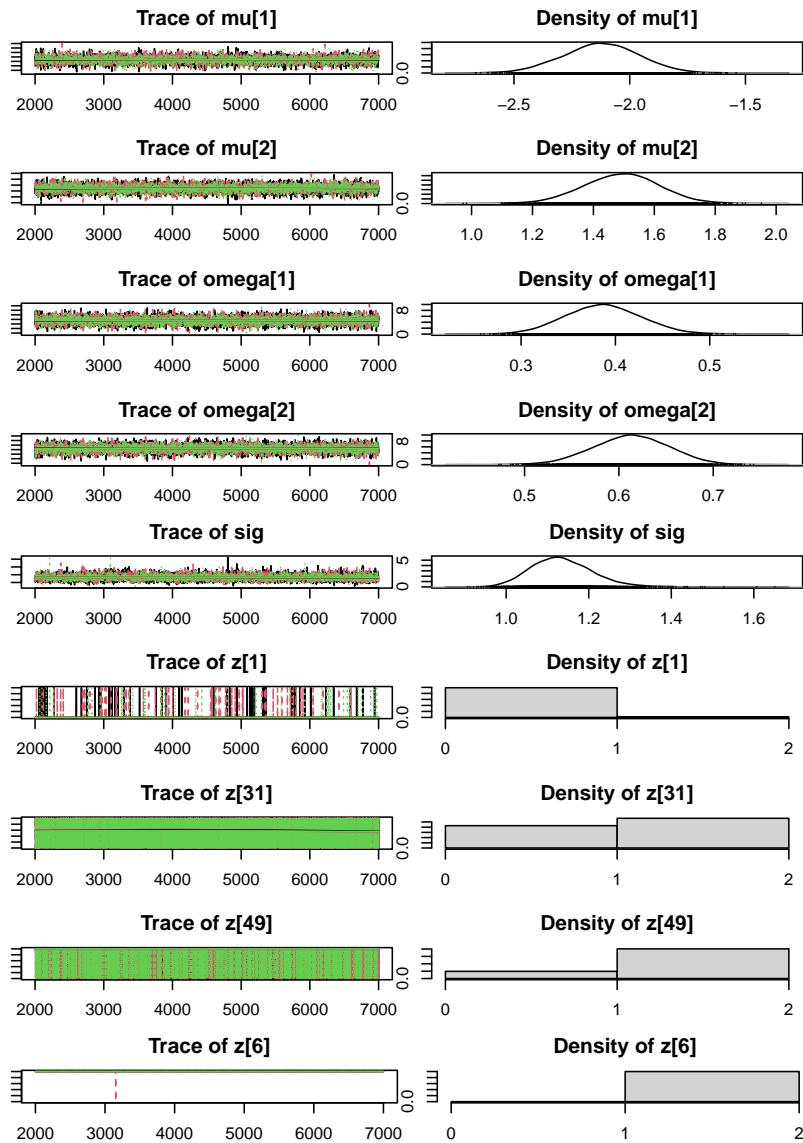
Initializing model

update(mod, 1e3)

mod_sim = coda.samples(model=mod,
                      variable.names=params,
                      n.iter=5e3)
mod_csim = as.mcmc(do.call(rbind, mod_sim))

## convergence diagnostics
par(mar = c(2.5, 1, 2.5, 1))
plot(mod_sim, ask=TRUE)

```



```
autocorr.diag(mod_sim)
```

	mu[1]	mu[2]	omega[1]	omega[2]	sig	z[1]
Lag 0	1.000000000	1.000000000	1.000000000	1.000000000	1.000000000	1.000000000
Lag 1	0.524354358	0.332347143	0.29056781	0.29056781	0.403158077	0.0182965930
Lag 5	0.069088955	0.049092548	0.05810839	0.05810839	0.043615287	0.0023334686
Lag 10	0.004923340	0.005268958	0.01561434	0.01561434	0.001786669	0.0006548686

```

Lag 50 0.003560756 0.006088961 0.01567735 0.01567735 0.004393704 -
0.0056559641
      z[31]      z[49]  z[6]
Lag 0  1.000000000 1.000000000  NaN
Lag 1  0.040816494 0.032691402  NaN
Lag 5  0.008354672 0.008919817  NaN
Lag 10 0.008882272 0.004775162  NaN
Lag 50 -0.001290924 -0.006732659  NaN

effectiveSize(mod_sim)

  mu[1]      mu[2]  omega[1]  omega[2]      sig      z[1]      z[31]      z[49]
4184.564  5595.064 6250.590 6250.590  5589.005 14647.515 12353.384 13700.920
      z[6]
  5000.000

```

38.9.3 Results

```
summary(mod_sim)
```

```

Iterations = 2001:7000
Thinning interval = 1
Number of chains = 3
Sample size per chain = 5000

```

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

	Mean	SD	Naive SE	Time-series SE
mu[1]	-2.1238	0.164904	1.346e-03	2.550e-03
mu[2]	1.4919	0.124769	1.019e-03	1.670e-03
omega[1]	0.3871	0.040148	3.278e-04	5.109e-04
omega[2]	0.6129	0.040148	3.278e-04	5.109e-04
sig	1.1351	0.074418	6.076e-04	9.969e-04
z[1]	1.0110	0.104306	8.517e-04	8.637e-04
z[31]	1.5711	0.494940	4.041e-03	4.455e-03
z[49]	1.8053	0.396008	3.233e-03	3.385e-03

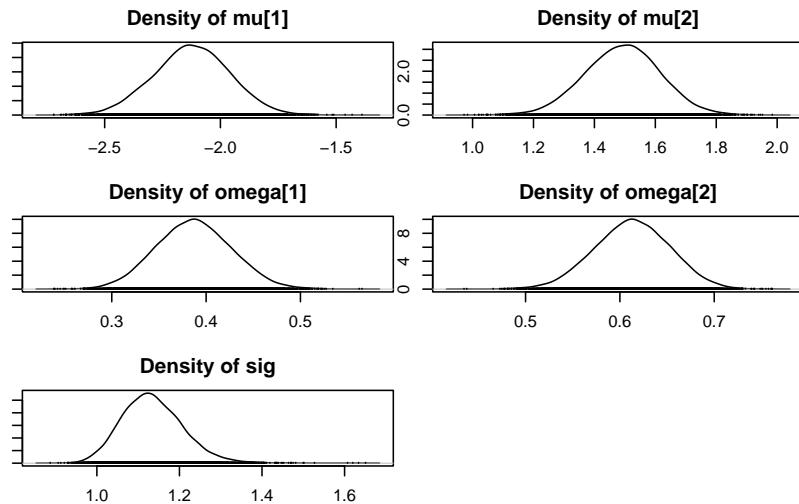
```
z[6]      1.9999 0.008165 6.667e-05      6.667e-05
```

2. Quantiles for each variable:

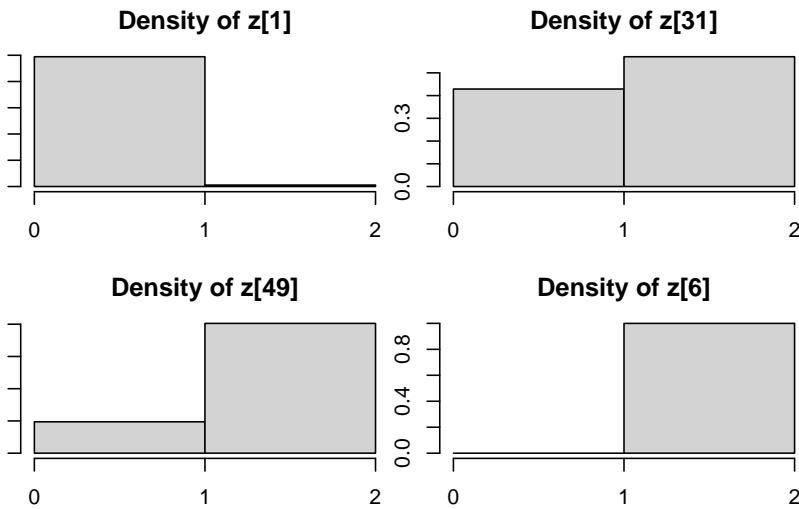
	2.5%	25%	50%	75%	97.5%
mu[1]	-2.4512	-2.2326	-2.1225	-2.0133	-1.8042
mu[2]	1.2448	1.4087	1.4942	1.5756	1.7340
omega[1]	0.3101	0.3600	0.3866	0.4138	0.4665
omega[2]	0.5335	0.5862	0.6134	0.6400	0.6899
sig	1.0039	1.0832	1.1303	1.1813	1.2949
z[1]	1.0000	1.0000	1.0000	1.0000	1.0000
z[31]	1.0000	1.0000	2.0000	2.0000	2.0000
z[49]	1.0000	2.0000	2.0000	2.0000	2.0000
z[6]	2.0000	2.0000	2.0000	2.0000	2.0000

```
## for the population parameters and the mixing weights
```

```
par(mar = c(2.5, 1, 2.5, 1))
par(mfrow=c(3,2))
densplot(mod_csim[,c("mu[1]", "mu[2]", "omega[1]", "omega[2]", "sig")])
```



```
## for the z's
par(mfrow=c(2,2))
par(mar = c(2.5, 1, 2.5, 1))
densplot(mod_csim[,c("z[1]", "z[31]", "z[49]", "z[6]")])
```



```
table(mod_csim[, "z[1]"]) / nrow(mod_csim) ## posterior probabilities for z[1], the membership of y[1]
```

1	2
0.989	0.011

```
table(mod_csim[, "z[31]"]) / nrow(mod_csim) ## posterior probabilities for z[31], the membership of y[31]
```

1	2
0.4289333	0.5710667

```
table(mod_csim[, "z[49]"]) / nrow(mod_csim) ## posterior probabilities for z[49], the membership of y[49]
```

1	2
0.1947333	0.8052667

```
table(mod_csim[, "z[6]"]) / nrow(mod_csim) ## posterior probabilities for z[6], the membership of y[6]
```

1	2
6.666667e-05	9.999333e-01

```
y[c(1, 31, 49, 6)]
```

```
[1] -2.2661749 -0.3702666  0.0365564  3.7548080
```

If we look back to the y values associated with these z variables we monitored, we see that y_1 is clearly in Population 1's territory, y_{31} is ambiguous, y_{49} is ambiguous but is closer to Population 2's territory, and y_6 is clearly in Population 2's territory. The posterior distributions for the z variables closely reflect our assessment.

39 Capstone Project

40 Appendix: Notation

parameters are with Greek letters

$$\theta$$

estimate

$$\hat{p}$$

the Certain event

$$\Omega$$

probability of RV x taking value x

$$\Pr(X = x)$$

odds

$$\mathcal{O}(X)$$

random variables

$$X$$

X is distributed as

$$X \sim N(\mu, \sigma^2)$$

X is proportional to

$$X \propto N(\mu, \sigma^2)$$

Probability of A and B

$$\mathbb{P}r(X \cap Y)$$

Conditional probability

$$\mathbb{P}r(X | Y)$$

Joint probability

$$\mathbb{P}r(X, Y)$$

$$Y_i \stackrel{iid}{\sim} N(\mu, \sigma^2)$$

Approximately distributed as (say using the CLT)

$$Y_i \sim N(\mu, \sigma^2)$$

$$\mathbb{E}[X_i]$$

The **expected value** of an RV X set to 0 (A.K.A. a fair bet)

$$\mathbb{E}[X_i] \stackrel{set}{=} 0$$

The variance of an RV

$$\mathbb{V}ar[X_i]$$

logical implication

$$\implies$$

if and only if

$$\iff$$

therefore

∴

independence

$$A \perp\!\!\!\perp B \iff \Pr(A | B) = \Pr(A)$$

Indicator function

$$\mathbb{I}_{\{A\}} = \begin{cases} 1 & \text{if } A \text{ is true} \\ 0 & \text{otherwise} \end{cases}$$

Dirichlet function

This is a continuous version of the indicator function, defined as a limit.

$$\delta(x) = \lim_{\epsilon \rightarrow 0} \frac{1}{2\epsilon} \mathbb{I}_{\{|x|<\epsilon\}}$$

The Dirichlet function is used to represent a point mass at a point, often used as a component for zero inflated mixtures.

The following are from course 4

- $\{y_t\}$ - the time series process, where each y_t is a univariate random variable and t are the time points that are equally spaced.
- $y_{1:T}$ or y_1, y_2, \dots, y_T - the observed data.
- You will see the use of ' to denote the transpose of a matrix,
- and the use of \sim to denote a distribution.
- under tildes \tilde{y} are used to denote estimates of the true values y .
- $\tilde{\Sigma}$ matrix of eigenvalues
- $\Lambda = \text{diagonal}(\alpha_1, \alpha_2, \dots, \alpha_p)$ is a diagonal matrix with the eigenvalues of Σ on the diagonal.
- $J_p(1)$ = a p by p Jordan form matrix with 1 on the superdiagonal

i Todo

- F the event space
- Likelihood

- LogLikelihood
- MLE Maximum likelihood estimator
- MAP Maximum a posteriori estimate

40.1 The argmax function

When we want to maximize a function $f(x)$, there are two things we may be interested in:

1. The value $f(x)$ achieves when it is maximized, which we denote $\max_x f(x)$.
2. The x -value that results in maximizing $f(x)$, which we denote $\hat{x} = \arg \max_x f(x)$. Thus $\max_x f(x) = f(\hat{x})$.

40.2 Indicator Functions

The concept of an indicator function is a really useful one. This is a function that takes the value one if its argument is true, and the value zero if its argument is false. Sometimes these functions are called Heaviside functions or unit step functions. I write an indicator function as $\mathbb{I}_A(x)$, although sometimes they are written $\mathbb{1}_A(x)$. If the context is obvious, we can also simply write IA .

Example:

$$\mathbb{I}_{x>3}(x) = \begin{cases} 0, & \text{if } x \leq 3 \\ 1, & \text{otherwise} \end{cases}$$

Note: Indicator functions are easy to implement in code using a `lambda` function. They can be combined using a dictionary.

Because $0 \cdot 1 = 0$, the product of indicator functions can be combined into a single indicator function with a modified condition.

40.2.1 Products of Indicator Functions:

$$\mathbb{I}_{x < 5} \cdot \mathbb{I}_{x \geq 0} = \mathbb{I}_{0 \leq x < 5}$$

$$\prod_{i=1}^N \mathbb{I}_{(x_i < 2)} = \mathbb{I}_{(x_i < 2) \forall i} = \mathbb{I}_{\max_i x_i < 2}$$

41 Appendix: Discrete Distributions

41.1 Discrete Uniform

41.1.1 Stories

i Discrete Uniform Finite set Parametrization

Let C be a finite, nonempty set of numbers and X random variable associated with the event of *choosing one of these numbers uniformly at random*, that is all values being equally likely $X(x = c)$. Then X is said to have the Discrete Uniform distribution with parameter C .

We denote this by $X \sim DUnif(C)$.

i Discrete Uniform with Lower and Upper bound Parametrization

When the set C above is $C = \{c \in \mathbb{Z} \mid a \leq c \leq b\}$. Then X is said to have the Discrete Uniform distribution with lower bound parameter a and upper bound parameter b . We denote this by $X \sim DUnif(a, b)$.

i Urn Model

Suppose we have an urn with n balls labeled with the numbers a_1, \dots, a_n . One drawing from the urn produces a discrete uniform random variable on the set $\{a_1, \dots, a_n\}$.

41.1.2 Moments

$$\begin{aligned}\phi_X(t) &= \frac{e^{at} - e^{(b+1)t}}{n(1 - e^t)} && (\text{MGF}) \\ \mathbb{E}[X] &= \frac{a+b}{2} && (\text{Expectation}) \\ \text{Var}[X] &= \frac{(b-a+1)^2 - 1}{12} && (\text{Variance})\end{aligned}\quad (41.1)$$

41.1.3 Probability mass function (PMF)

$$f(x | a, b) = \frac{1}{b-a+1}$$

41.1.4 Cumulative distribution function (CDF)

$$F(x | a, b) = \frac{\lfloor x \rfloor - a - 1}{b - a + 1} \text{ where } \lfloor x \rfloor \text{ is the floor function (rounds down reals to nearest smaller integer)}$$

41.1.5 Prior

Since a number of families have the uniform as a special case we can use them as priors when we want a uniform prior:

41.2 Bernoulli Distribution

41.2.1 Stories

i Note

The Bernoulli distribution arises when modeling the outcome of a binary event called a **Bernoulli trial**.

Let X be the indicator variable corresponding to the success of getting “heads” in a “coin toss”, with a coin that has probability of success p for getting “heads”.

Then X has a *Bernoulli Distribution* with parameter p

We denote this as $X \sim \text{Bern}(p)$

41.2.2 Parameters

Because of this story, the *parameter* p is often called the **success** probability of the $\text{Bern}(p)$ distribution.

41.2.3 Examples

Note

- fair coin toss
- unfair coin toss
- ad click
- web site conversion
- death or survival of a patient in a medical trial
- indicator random variable

41.2.4 Checklist

Note

- Discrete data
- A single trial
- Only two trial outcomes: **success** and **failure** (These do not need to literally represent successes and failures, but this shorthand is typically used.)

$$\begin{aligned} X &\sim \text{Bernoulli}(p) \\ &\sim \text{Bern}(p) \\ &\sim B(p) \end{aligned} \tag{41.2}$$

41.2.5 Moments

$$M_X(t) = q + pe^t \quad (\text{MGF}) \tag{41.3}$$

$$\mathbb{E}[X] = p \quad (\text{Expectation}) \tag{41.4}$$

$$\mathbb{V}ar[x] = \mathbb{P}r(1 - p) \quad (\text{Variance}) \quad (41.5)$$

41.2.6 PMF

Where parameter p is the probability of getting heads.

The probability for the two events is:

$$\mathbb{P}r(X = 1) = p \quad \mathbb{P}r(X = 0) = 1 - p$$

$$\begin{cases} 1 - p & \text{if } k = 0 \\ p & \text{if } k = 1 \end{cases} \quad (\text{PMF}) \quad (41.6)$$

41.2.7 CDF

$$\begin{cases} 0 & \text{if } k < 0 \\ 1 - p & \text{if } 0 \leq k < 1 \\ 1 & \text{if } k \geq 1 \end{cases} \quad (\text{CDF}) \quad (41.7)$$

41.2.8 Likelihood

$$L(\theta) = \prod p^x (1 - p)^{1-x} \mathbb{I}_{[0,1]}(x) \quad (\text{Likelihood}) \quad (41.8)$$

$$\mathcal{L}(\theta) = \log(p) \sum x + \log(1 - p) \sum (1 - x) \quad (\text{Log Likelihood}) \quad (41.9)$$

41.2.9 Entropy and Information

$$\mathbb{H}(x) = -q \ln(q) - p \ln(p) \quad (\text{Entropy}) \quad (41.10)$$

$$\mathcal{I}[X] \frac{1}{\mathbb{P}r(1 - p)} \quad (\text{Fisher Information}) \quad (41.11)$$

$$\text{Beta}(x) \quad (\text{Conjugate Prior}) \quad (41.12)$$

41.2.10 Usage

Table 41.1: Usage of Bernoulli

Package	Syntax
NumPy	<code>rg.choice([0, 1], p=[1-theta, theta])</code>
SciPy	<code>scipy.stats.bernoulli(theta)</code>
Stan	<code>bernoulli(theta)</code>

41.2.11 Plots

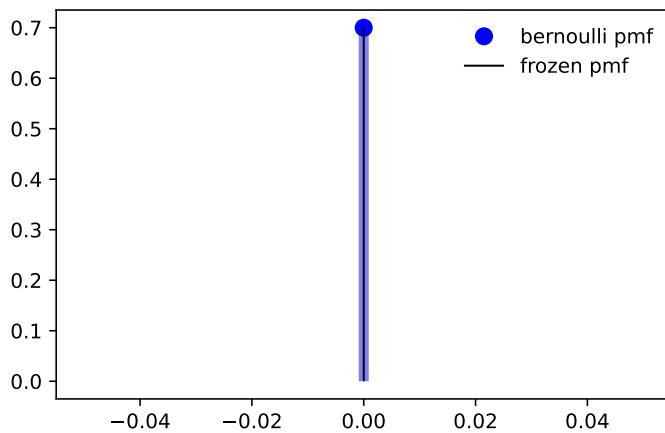
```
import numpy as np
from scipy.stats import bernoulli
import matplotlib.pyplot as plt

fig, ax = plt.subplots(1, 1)
p = 0.3
mean, var, skew, kurt = bernoulli.stats(p, moments='mvsk')
print(f'{mean=:1.2f}, {var=:1.2f}, {skew=:1.2f}, {kurt=:1.2f}')

mean=0.30, var=0.21, skew=0.87, kurt=-1.24

x = np.arange(bernoulli.ppf(0.01, p),
              bernoulli.ppf(0.99, p))
ax.plot(x, bernoulli.pmf(x, p), 'bo', ms=8, label='bernoulli pmf')
ax.vlines(x, 0, bernoulli.pmf(x, p), colors='b', lw=5, alpha=0.5)

rv = bernoulli(p)
ax.vlines(x, 0, rv.pmf(x), colors='k', linestyles='-', lw=1,
          label='frozen pmf')
ax.legend(loc='best', frameon=False)
plt.show()
```



```
## Generate random numbers
r = bernoulli.rvs(p, size=10)
r

array([1, 0, 0, 0, 0, 0, 1, 0, 0, 0])
```



Figure 41.1: A Swiss stamp issued in 1994 depicting Mathematician Jakob Bernoulli and the formula and diagram of the law of large numbers

Biographical note on Jacob Bernoulli

It seems that to make a correct conjecture about any event whatever, it is necessary to calculate exactly the number of possible cases and then to determine how

much more likely it is that one case will occur than another. (Bernoulli 1713)

The Bernoulli distribution as well as The Binomial distribution are due to Jacob Bernoulli (1655-1705) who was a prominent mathematician in the Bernoulli family. He discovered the fundamental mathematical constant e. With his brother Johann, he was among the first to develop Leibniz's calculus, introducing the word integral and applying it to polar coordinates and the study of curves such as the catenary, the logarithmic spiral and the cycloid

His most important contribution was in the field of probability, where he derived the first version of the law of large numbers (LLN). The LLN is a theorem that describes the result of performing the same experiment a large number of times. According to the law, the average of the results obtained from a large number of trials should be close to the expected value and will tend to become closer to the expected value as more trials are performed. A special form of the LLN (for a binary random variable) was first proved by Jacob Bernoulli. It took him over 20 years to develop sufficiently rigorous mathematical proof.

For a more extensive biography visit the following [link](#)

The Bernoulli distribution is built on a trial of a coin toss (possibly biased).

- We use the Bernoulli distribution to model a random variable for the probability of such a coin toss trial.
- We use the Binomial distribution to model a random variable for the probability of getting k heads in N independent trials.

41.3 Binomial distribution

41.3.1 Stories

Note

$$\underbrace{\boxed{0} \dots \boxed{0}}_{N_0} \underbrace{\boxed{1} \dots \boxed{1}}_{N_1} \quad (41.13)$$

The Binomial distribution arises when we conduct multiple independent Bernoulli trials and wish to model X the number of successes in $Y_i \mid \theta$ identically distributed Bernoulli trials with the same probability of success θ . If n independent *Bernoulli trials* are performed, each with the same success probability p . The distribution of X is called the Binomial distribution with parameters n and p . We write $X \sim \text{Bin}(n, p)$ to mean that X has the Binomial distribution with parameters n and p , where n is a positive integer and $0 < p < 1$.

41.3.2 Parameters

- θ - the probability of success in the Bernoulli trials
- N - the total number of trials being conducted

41.3.3 Conditions

Tip

- Discrete data
- Two possible outcomes for each trial
- Each trial is independent and
- The probability of success/failure is the same in each trial
- The outcome is the aggregate number of successes

41.3.4 Examples

- to model the aggregate outcome of clinical drug trials,

- to estimate the proportion of the population voting for each political party using exit poll data (where there are only two political parties).

$$X \sim Bin[n, p] \quad (41.14)$$

$$f(X = x | \theta) = \binom{n}{x} \theta^x (1 - \theta)^{n-x} \quad (41.15)$$

$$L(\theta) = \prod_{i=1}^n \binom{n}{x_i} \theta^{x_i} (1 - \theta)^{(n-x_i)} \quad (41.16)$$

$$\begin{aligned} \ell(\theta) &= \log \mathcal{L}(\theta) \\ &= \sum_{i=1}^n \left[\log \binom{n}{x_i} + x_i \log \theta + (n - x_i) \log(1 - \theta) \right]. \end{aligned} \quad (41.17)$$

$$\mathbb{E}[X] = N \times \theta \quad (41.18)$$

$$\mathbb{V}ar[X] = N \cdot \theta \cdot (1 - \theta) \quad (41.19)$$

$$\mathbb{H}(X) = \frac{1}{2} \log_2 (2\pi n \theta (1 - \theta)) + O\left(\frac{1}{n}\right) \quad (41.20)$$

$$\mathcal{I}(\theta) = \frac{n}{\theta \cdot (1 - \theta)} \quad (41.21)$$

41.3.5 Usage

Table 41.2: Usage of Binomial

Package	Syntax
NumPy	<code>rg.binomial(N, theta)</code>
SciPy	<code>scipy.stats.binom(N, theta)</code>
Stan	<code>binomial(N, theta)</code>

41.3.6 Relationships

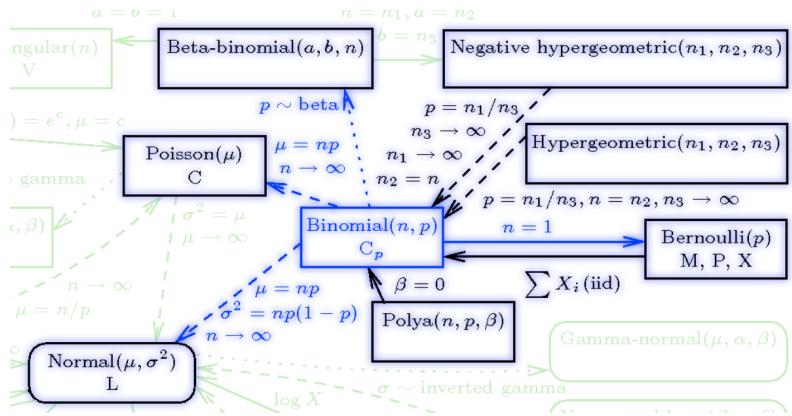


Figure 41.2: binomial distribution relations

The Binomial Distribution is related to

- The **Binomial** is a *special case* of the **Multinomial distribution** with K=2 (two categories).
- the **Poisson distribution** distribution. If $X \sim \text{Binomial}(n, p)$ RV and $Y \sim \text{Poisson}(np)$ distribution then $\Pr(X = n) \approx \Pr(Y = n)$ for large n and small np .
- The **Bernoulli distribution** is a *special case* of the the **Binomial distribution** $X \sim \text{Binomial}(n = 1, p)$
 $\Rightarrow X \sim \text{Bernoulli}(p)$
- the **Normal distribution** If $X \sim \text{Binomial}(n, p)$ RV and $Y \sim \text{Normal}(\mu = np, \sigma = n\Pr(1-p))$ then for integers j and k, $\Pr(j \leq X \leq k) \approx \Pr(j - 1/2 \leq Y \leq k + 1/2)$. The approximation is better when $p \approx 0.5$ and when n is large. For more information, see normal approximation to the Binomial
- The **Binomial** is a *limit* of the **Hypergeometric**. The difference between a binomial distribution and a hypergeometric distribution is the difference between sampling with replacement and sampling without replacement. As the population size increases relative to the sample size, the difference becomes negligible. So If $X \sim \text{Binomial}(n, p)$ RV and $Y \sim \text{HyperGeometric}(N, a, b)$ then

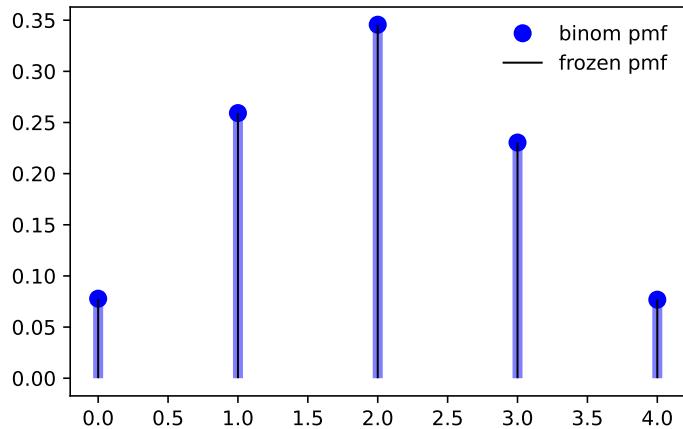
$$\lim_{n \rightarrow \infty} X = Y$$

41.3.7 Plots

```
import numpy as np
from scipy.stats import binom
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)
n, p = 5, 0.4
mean, var, skew, kurt = binom.stats(n, p, moments='mvsk')
print(f'{mean=:1.2f}, {var=:1.2f}, {skew=:1.2f}, {kurt=:1.2f}')

mean=2.00, var=1.20, skew=0.18, kurt=-0.37

x = np.arange(binom.ppf(0.01, n, p),
               binom.ppf(0.99, n, p))
ax.plot(x, binom.pmf(x, n, p), 'bo', ms=8, label='binom pmf')
ax.vlines(x, 0, binom.pmf(x, n, p), colors='b', lw=5, alpha=0.5)
rv = binom(n, p)
ax.vlines(x, 0, rv.pmf(x), colors='k', linestyles='-', lw=1,
           label='frozen pmf')
ax.legend(loc='best', frameon=False)
plt.show()
```



```

## generate random numbers
r = binom.rvs(n, p, size=10)
r

array([3, 2, 1, 4, 3, 2, 2, 3, 2, 1])

``` {{python}}
from __future__ import print_function
from ipywidgets import interact, interactive, fixed, interact_manual
import ipywidgets as widgets
import numpy as np
import scipy
from scipy.special import gamma, factorial, comb
import plotly.express as px
import plotly.offline as pyo
import plotly.graph_objs as go
#pyo.init_notebook_mode()
INTERACT_FLAG=False
def binomial_vector_over_y(theta, n):
 total_events = n
 y = np.linspace(0, total_events , total_events + 1)
 p_y = [comb(int(total_events), int(yelem)) * theta** yelem * (1 - theta)**(total_events - yelem) for yelem in y]
 fig = px.line(x=y, y=p_y, color_discrete_sequence=["steelblue"],
 height=600, width=800, title=" Binomial distribution for theta = %lf, n = %d" %(theta, n))
 fig.data[0].line['width'] = 4
 fig.layout.xaxis.title.text = "y"
 fig.layout.yaxis.title.text = "P(y)"
 fig.show()

if(INTERACT_FLAG):
 interact(binomial_vector_over_y, theta=0.5, n=15)
else:
 binomial_vector_over_y(theta=0.5, n=10)
```

```

https://youtu.be/sn-mp_ESSMc

41.4 Hypergeometric distribution

41.4.1 story 1 - Urn Model

The beta-binomial distribution with parameters α success rate and β failure and n the number of trials can be motivated by an Pólya urn model.

Imagine a trial in which a ball is drawn without replacement from urn containing α white balls and β black balls. If this is repeated n times, then the probability of observing x white balls follows a hypergeometric distribution with parameters n, α and β .

Note: is we used a

If the random draws are with simple replacement (no balls over and above the observed ball are added to the urn), then the distribution follows a binomial distribution and if the random draws are made without replacement, the distribution follows a hypergeometric distribution.

<https://youtu.be/UgqQc6epZnc>

41.4.2 Examples

- k white balls from an urn without replacement
- capture-recapture
- Aces in a poker hand

41.4.3 Story

Consider an urn with w white balls and b black balls. We draw n balls out of the urn at random without replacement, such that all $w+b$ samples are equally likely. Let X be the number of white balls in n the sample. Then X is said to have the Hypergeometric distribution with parameters w, b , and n ; we denote this by $X \sim HGeom(w, b, n)$

41.5 Poisson distribution

41.5.1 Stories

Poisson Parametrization

The Poisson distribution arises when modeling the number of successes of independent and identically distributed (IID) events in a fixed interval of time or space, occurring at a constant rate λ . Let X represent the count of the number of phone calls received at a call center in a given interval, such as an hour, with the parameter λ corresponding to the average rate at which events occur in that interval. Then X is said to have the Poisson distribution with parameter λ , and we denote this as $X \sim \text{Pois}(\lambda)$.

$$X \sim \text{Pois}(\lambda) \quad (41.22)$$

<https://youtu.be/yIPFA1sk5NA>

41.5.2 Checklist

- Count of discrete events
- Individual events occur at a given rate and independently of other events
- Fixed amount of time or space in which the events can occur

41.5.3 Examples

- The number of emails you receive in an hour. There are a lot of people who could potentially email you at that hour, but it is unlikely that any specific person will actually email you at that hour. Alternatively, imagine subdividing the hour into milliseconds. There are 3.6×10^6 seconds in an hour, but in any specific millisecond, it is unlikely that you will get an email.
- The number of chips in a chocolate chip cookie. Imagine subdividing the cookie into small cubes; the probability of getting a chocolate chip in a single cube is small, but the number of cubes is large.

- The number of earthquakes in a year in some regions of the world.
At any given time and location, the probability of an earthquake is small, but there are a large number of possible times and locations for earthquakes to occur over the course of the year.
- Count of component failures per week
- estimating the failure rate of artificial heart valves,
- estimating the prevalence of violent crimes in different districts,
- approximating the binomial which is, itself, being used to explain the prevalence of autism in the UK.

41.5.4 Moments

$$\mathbb{E}(X) = \lambda$$

$$\text{Var}(X) = \lambda$$

41.5.5 Probability mass function (PMF)

$$f(x | \lambda) = \frac{\lambda^x e^{-\lambda}}{x!} \quad (41.23)$$

41.5.6 Cumulative distribution function (CDF)

$$F(x | \lambda) = \frac{\Gamma(\lfloor x + 1 \rfloor, \lambda)}{\lfloor x \rfloor!} \quad \text{CDF} \quad (41.24)$$

where $\Gamma(u, v) = \int_v^\infty t^{u-1} e^{-t} dt$ is the upper incomplete gamma function
(41.25)

and $\lfloor x \rfloor$ is the floor function (rounds down reals to nearest smaller integer)
(41.26)

41.6 Geometric distribution

41.6.1 Stories

i Geometric Distribution Failures before success

Consider a sequence of independent Bernoulli trials, each with the same success probability $p \in (0, 1)$, with trials performed until a success occurs. Let X be the number of failures before the first successful trial. Then X has the Geometric distribution with parameter p ; we denote this by $X \sim \text{Geom}(p)$.

For example, if we flip a fair coin until it lands Heads for the first time, then the number of Tails before the first occurrence of Heads is distributed as $\text{Geom}(1/2)$.

To get the Geometric PMF from the story, imagine the Bernoulli trials as a string of 0's (failures) ending in a single 1 (success). Each 0 has probability $q = 1 - p$ and the final 1 has probability p , so a string of k failures followed by one success has probability $q^k p$.

i Geometric distribution Failures and success

Consider a sequence of independent Bernoulli trials, each with the same success probability $p \in (0, 1)$, with trials performed until a success occurs. Let X be the number of failures before the first successful trial. Then X has the Geometric distribution with parameter p ; we denote this by $X \sim \text{Geom}(p)$.

For example, if we flip a fair coin until it lands Heads for the first time, then the number of Tails before the first occurrence of Heads is distributed as $\text{Geom}(1/2)$.

To get the Geometric PMF from the story, imagine the Bernoulli trials as a string of 0's (failures) ending in a single 1 (success). Each 0 has probability $q = 1 - p$ and the final 1 has probability p , so a string of k failures followed by one success has probability $q^k p$.

<https://youtu.be/-vvtrsS4rkA>

41.6.2 Conditions



Tip

- Discrete data
- Two possible outcomes for each trial
- Each trial is independent and
- The probability of success/failure is the same in each trial
- The outcome is the count of failures before the first success

41.6.3 Examples

- Consider polymerization of an actin filament. At each time step, an actin monomer may add to the end of the filament (“failure”), or an actin monomer may fall off the end (“success”) with (usually very low) probability θ . The length of actin filaments, measured in a number of constitutive monomers, is Geometrically distributed.

The **Geometric distribution** arises when we want to know “What is the number of Bernoulli trials required to get the first success?”, i.e., the number of Bernoulli events until a success is observed, such as the probability of getting the first head when flipping a coin. It takes values on the positive integers starting with one (since at least one trial is needed to observe a success).

<https://youtu.be/-vvtrsS4rkA>

$$X \sim Geo(p) \quad (41.27)$$

41.6.4 Moments

$$\mathbb{M}_X[t] = \frac{pe^t}{1 - (1-p)e^t} \quad t < -\ln(1-p) \quad (41.28)$$

$$\mathbb{E}[X] = \frac{1}{p} \quad (41.29)$$

$$\mathbb{V}ar[X] = \frac{1-p}{p^2} \quad (41.30)$$

41.6.5 PMF

$$\Pr(X = x \mid p) = \Pr(1 - p)^{x-1} \quad \forall x \in N; \quad 0 \leq p \leq 1 \quad (41.31)$$

41.6.6 CDF

$$1 - (1 - p)^{\lfloor x \rfloor} \quad x < 1 \quad (41.32)$$

41.6.7 Memoryless property

The geometric distribution is based on geometric series.

The geometric distribution has the memoryless property:

$$P(X > s \mid X > t) = P(X > s - t)$$

One can say that the distribution “forgets” what has occurred, so that The probability of getting an additional $s - t$ failures, having already observed t failures, is the same as the probability of observing $s - t$ failures at the start of the sequence. In other words, the probability of getting a run of failures depends only on the length of the run, not on its position.

$Y = X - 1$ is the negative binomial($1, p$)

41.6.8 Worked out Examples

Example 41.1 (Geometric Distribution). The Geometric distribution arises when we consider how long we will have to “wait for a success” during repeated Bernoulli trials.

What is the probability that we flip a fair coin four times and don’t see any heads?

This is the same as asking what is $\Pr(X > 4)$ where $X \sim Geo(1/2)$.

$$\begin{aligned} \Pr(X > 4) &= 1 - \Pr(X = 1) - \Pr(X = 2) - \Pr(X = 3) - \Pr(X = 4) \\ &= 1 - \left(\frac{1}{2}\right) - \left(\frac{1}{2}\right)\left(\frac{1}{2}\right) - \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)^2 - \left(\frac{1}{2}\right)\left(\frac{1}{2}\right)^3 \\ &= \frac{1}{16} \end{aligned}$$

Of course, we could also have just computed it directly, but here we see an example of using the geometric distribution and we can also see that we got the right answer.

41.6.9 Plots

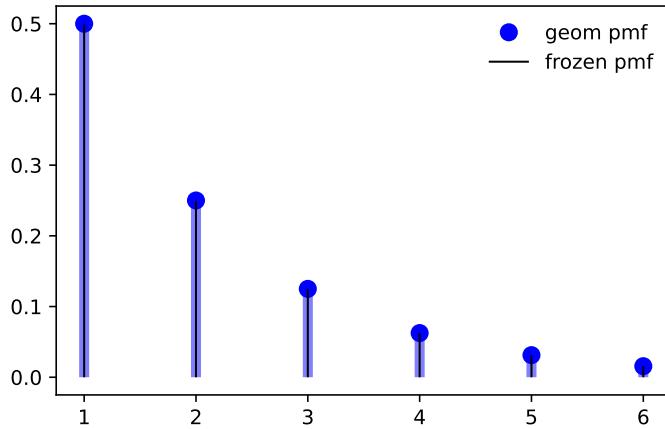
```
import numpy as np
from scipy.stats import geom
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)

p = 0.5
mean, var, skew, kurt = geom.stats(p, moments='mvsk')
print(f'{mean=:1.2f}, {var=:1.2f}, {skew=:1.2f}, {kurt=:1.2f}')

mean=2.00, var=2.00, skew=2.12, kurt=6.50

x = np.arange(geom.ppf(0.01, p),
              geom.ppf(0.99, p))
ax.plot(x, geom.pmf(x, p), 'bo', ms=8, label='geom pmf')
ax.vlines(x, 0, geom.pmf(x, p), colors='b', lw=5, alpha=0.5)

rv = geom(p)
ax.vlines(x, 0, rv.pmf(x), colors='k', linestyles='-', lw=1,
           label='frozen pmf')
ax.legend(loc='best', frameon=False)
plt.show()
```



```
r = geom.rvs(p, size=10)
r

array([1, 2, 1, 3, 1, 1, 1, 4, 6, 1])
```

41.7 Negative Binomial Distribution

41.7.1 Story

i Note

In a sequence of independent Bernoulli trials with success probability p , if X is the number of failures before the r^{th} success, then X is said to have the *Negative Binomial* distribution with parameters r and p , denoted $X \sim NBin(r, p)$.

Both the Binomial and the *Negative Binomial* distributions are based on independent Bernoulli trials; they differ in the stopping rule and in what they are counting.

The Binomial counts the number of successes in a fixed number of trials; the *Negative Binomial* counts the number of failures until a fixed number of successes.

In light of these similarities, it comes as no surprise that the derivation of the *Negative Binomial* PMF bears a resemblance to the corresponding derivation for the Binomial.

41.7.2 Parameters

- r the number of successes.
- p the probability of the Bernoulli trial.

41.7.3 Conditions

 Tip

- Count of discrete events
- Non-independent events; it is sometimes said that the events can exhibit contagion, meaning that if one event occurs, it is more likely that another will also occur
- Can model a data-generating process where the variance exceeds the mean
- Fixed amount of time or space in which the events can occur

41.7.4 Examples

- Stamp collection - Suppose there are n types of stamps, which you are collecting one by one, with the goal of getting a complete set. When collecting stamps, the stamp types are random. Assume that each time you collect a stamp, it is equally likely to be any of the n types. What is the expected number of stamps needed until you have a complete set?
- everything the Poisson can do and more,
- to model the number of measles cases that occur on an island,
- the number of banks that collapse in a financial crisis.
- the length of a hospital stay
- the probability you will have to visit Y houses if you must sell r cookies before returning home

41.7.5 Moments

$$\text{E}(X) = \lambda$$

$$\text{var}(X) = \lambda + \frac{\lambda^2}{\kappa}$$

41.7.6 Probability mass function (PMF)

$$f(x | \lambda, \kappa) = \frac{\Gamma(x + \kappa)}{x! \Gamma(\kappa + 1)} \left(\frac{\lambda}{\lambda + \kappa} \right)^x \left(\frac{\kappa}{\lambda + \kappa} \right)^\kappa$$

41.7.7 Cumulative distribution function (CDF)

$$F(x | \lambda, \kappa) = \begin{cases} I_{\frac{\kappa}{\kappa+\lambda}}(\kappa, 1 + \lfloor x \rfloor), & x \geq q0 \\ 0, & \text{Otherwise} \end{cases}$$

where $I_w(u, v)$ is the regularised incomplete beta function: $I_w(u, v) = \frac{B(w; u, v)}{B(u, v)}$

where $B(w; u, v) = \int_0^w t^{u-1} (1-t)^{v-1} dt$ is the incomplete beta function and $B(u, v) = \int_0^1 t^{u-1} (1-t)^{v-1} dt$ is the complete beta function.

41.8 Multinomial Distribution

The Multinomial distribution is a generalization of the Binomial. Whereas the Binomial distribution counts the successes in a fixed number of trials that can only be categorized as success or failure, the Multinomial distribution keeps track of trials whose outcomes can fall into multiple categories, such as excellent, adequate, poor; or red, yellow, green, blue.

41.8.1 Story

Multinomial distribution. Each of N objects is independently placed into one of k categories. An object is placed into category j with probability p_j , P where the p_j are non-negative and $\sum_{j=1}^k p_j = 1$. Let X_1 be the number of objects in category 1, X_2 the number of objects in category 2, etc., so that $X_1 + \dots + X_k = n$. Then $X = (X_1, \dots, X_k)$ is said to have the Multinomial distribution with parameters n and $p = (p_1, \dots, p_k)$. We write this as $X \sim Mult_k(n, p)$.

We call X a *random vector* because it is a vector of random variables. The joint PMF of X can be derived from the story.

41.8.2 Examples

- Blood type counts across n individuals
- Numbers of people voting for each party in a sample

41.8.3 Moments

$$E(X_i) = np_i, \forall i$$

$$var(X_i) = np_i(1 - p_i), \forall i$$

$$cov(X_i, X_j) = -np_i p_j, \forall i \neq j$$

41.8.4 Probability Mass Function (PMF)

$$f(x_1, x_2, \dots, x_d | n, p_1, p_2, \dots, p_d) = \frac{n!}{x_1! x_2! \dots x_d!} p_1^{x_1} p_2^{x_2} \dots p_d^{x_d}$$

41.9 Beta Binomial

41.9.1 Story 1 - Polya Urn Model

The beta-binomial distribution with parameters α success rate and β failure and n the number of trials can be motivated by an Pólya urn model.

Imagine an urn containing α red balls and β black balls, where random draws are made. If a red ball is observed, then two red balls are returned to the urn. Likewise, if a black ball is drawn, then two black balls are returned to the urn. If this is repeated n times, then the probability of observing x red balls follows a beta-binomial distribution with parameters n , α and β .

If the random draws are with simple replacement (no balls over and above the observed ball are added to the urn), then the distribution follows a binomial distribution and if the random draws are made without replacement, the distribution follows a hypergeometric distribution.

41.9.2 Story 2 compound distribution

The Beta distribution is a conjugate distribution of the binomial distribution. This fact leads to an analytically tractable compound distribution constructed in a hierarchical fashion where one can think of the p parameter in the *binomial* distribution as being randomly drawn from a *beta* distribution.

Suppose we were interested in predicting the number of heads, x in n future trials. This is given by

$$\begin{aligned} f(x \mid n, \alpha, \beta) &= \int_0^1 \text{Bin}(x \mid n, p) \text{Beta}(p \mid \alpha, \beta) dp \\ &= \binom{n}{x} \frac{1}{B(\alpha, \beta)} \int_0^1 p^{x+\alpha-1} (1-p)^{n-x+\beta-1} dp \\ &= \binom{n}{x} \frac{B(x+\alpha, n-x+\beta)}{B(\alpha, \beta)}. \end{aligned}$$

$$f(x \mid n, \alpha, \beta) = \frac{\Gamma(n+1)}{\Gamma(x+1)\Gamma(n-x+1)} \frac{\Gamma(x+\alpha)\Gamma(n-x+\beta)}{\Gamma(n+\alpha+\beta)} \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)}.$$

41.9.3 Moments

$$\begin{aligned}\mathbb{E}(X) &= \frac{n\alpha}{\alpha + \beta} \\ \text{var}(X) &= \frac{n\alpha\beta(\alpha + \beta + n)}{(\alpha + \beta)^2(\alpha + \beta + 1)}\end{aligned}$$

41.9.4 Probability mass function (PMF)

$$f(x \mid n, \alpha, \beta) = \binom{n}{x} \frac{B(x + \alpha, n - x + \beta)}{B(\alpha, \beta)}$$

where $B(u, v) = \int_0^1 t^{u-1}(1-t)^{v-1} dt$ is the (complete) beta function

41.9.5 Cumulative distribution function (CDF)

$$F(x \mid n, \alpha, \beta) = \begin{cases} 0, & x < 0 \\ \binom{n}{x} \frac{B(x + \alpha, n - x + \beta)}{B(\alpha, \beta)} {}_3F_2(1, -x, n - x + \beta; n - x - 1, 1 - x - \alpha; 1), & 0 \leq x \leq n \\ 1, & x > n \end{cases}$$

where ${}_3F_2(a, b, c; d, e; x)$ is the generalised hypergeometric function

41.9.6 Relations

- The **Pascal distribution** (after Blaise Pascal) is special cases of the negative binomial distribution. Used with an integer-valued stopping-time parameter r
- The **Pólya distribution** (for George Pólya) is special cases of the negative binomial distribution. Used with a real-valued-valued stopping-time parameter r



Figure 41.3: A photo of Hungarian Mathematician George Pólya

💡 Biographical note on George Pólya

The cookbook gives a detailed description of ingredients and procedures but no proofs for its prescriptions or reasons for its recipes; the proof of the pudding is in the eating ... Mathematics cannot be tested in exactly the same manner as a pudding; if all sorts of reasoning are debarred, a course of calculus may easily become an incoherent inventory of indigestible information.
(Polya 1945)

Pólya was arguably the most influential mathematician of the 20th century. His basic research contributions span complex analysis, mathematical physics, probability theory, geometry, and combinatorics. He was a teacher par excellence who maintained a strong interest in pedagogical matters throughout his long career.

He was awarded a doctorate in mathematics having studied, essentially without supervision, a problem in the theory of geometric probability. Later Pólya looked at the Fourier transform of a probability measure, showing in 1923 that it was a characteristic function. He wrote on the normal distribution and coined the term “central limit theorem” in 1920 which is now standard usage.

In 1921 he proved his famous theorem on *random walks* on an

integer lattice. He considered a d-dimensional array of lattice points where a point moves to any of its neighbors with equal probability. He asked whether given an arbitrary point A in the lattice, a point executing a random walk starting from the origin would reach A with probability 1. Pólya's surprising answer was that it would for $d = 1$ and for $d = 2$, but it would not for $d \geq 3$. In later work he looked at two points executing independent random walks and also at random walks satisfying the condition that the moving point never passed through the same lattice point twice.

One of Pólya's notable achievements was his collaboration with the economist Abraham Wald during World War II. They developed statistical techniques to solve military problems, including estimating enemy troop movements and predicting the effectiveness of bombing missions. These contributions played a vital role in aiding the Allies during the war.

His book "How to Solve It," published in 1945, presented problem-solving heuristics applicable to various mathematical domains, including probability and statistics. This influential work emphasized the importance of understanding the problem, devising a plan, executing the plan, and reflecting on the results. Pólya's problem-solving strategies continue to be widely taught and practiced.

For a more extensive biography visit the following [link](#)

42 Appendix: Continuous Distributions

Following a subjective view of distribution, which is more amenable to reinterpretation I use an indicator function to place restrictions on the range of parameter of the PDF.

42.1 The Continuous Uniform

42.1.1 Stories

 Discrete Uniform Finite set Parametrization

$$X \sim U[\alpha, \beta] \quad (42.1)$$

42.1.2 Moments

$$\mathbb{E}[X] = \frac{(\alpha + \beta)}{2} \quad (42.2)$$

$$\text{Var}[X] = \frac{(\beta - \alpha)^2}{12} \quad (42.3)$$

42.1.3 Probability mass function (PDF)

$$f(x) = \frac{1}{\alpha - \beta} \mathbb{I}_{\{\alpha \leq x \leq \beta\}}(x) \quad (42.4)$$

42.1.4 Cumulative distribution function (CDF)

$$F(x | \alpha, \beta) = \begin{cases} 0, & \text{if } x < \alpha \\ \frac{x-\alpha}{\beta-\alpha}, & \text{if } x \in [\alpha, \beta] \\ 1, & \text{if } x > \beta \end{cases} \quad (42.5)$$

42.1.5 Prior

Since a number of families have the uniform as a special case we can use them as priors when we want a uniform prior:

$$\text{Normal}(0, 1) = \text{Beta}(1, 1)$$

42.2 The Beta Distribution

42.2.1 Story

The Beta distribution is used for random variables which take on values between 0 and 1. For this reason (and other reasons we will see later in the course), the Beta distribution is commonly used to model probabilities.

$$X \sim \text{Beta}(\alpha, \beta) \quad (42.6)$$

42.2.2 PDF & CDF

$$f(x | \alpha, \beta) = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1} (1-x)^{\beta-1} \mathbb{I}_{x \in (0,1)} \mathbb{I}_{\alpha \in \mathbb{R}^+} \mathbb{I}_{\beta \in \mathbb{R}^+} \quad (\text{PDF}) \quad (42.7)$$

$$\begin{aligned} F(x | \alpha, \beta) &= I_x(\alpha, \beta) && \text{(CDF)} \\ \text{where } I_w(u, v) & && \text{is the regularized beta function:} \\ I_w(u, v) &= \frac{B(w; u, v)}{B(u, v)} \\ \text{where } B(w; u, v) &= \int_0^w t^{u-1} (1-t)^{v-1} dt && \text{is the incomplete beta function} \\ \text{and } B(u, v) & && \text{is the (complete) beta function} \end{aligned} \quad (42.8)$$

42.2.3 Moments

$$\mathbb{E}[X] = \frac{\alpha}{\alpha + \beta} \quad (\text{expectation}) \quad (42.9)$$

$$\mathbb{V}ar[X] = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)} \quad (\text{variance}) \quad (42.10)$$

$$\mathbb{M}_X(t) = 1 + \sum_{i=1}^{\infty} \left(\prod_{j=0}^{\infty} \frac{\alpha + j}{\alpha + \beta + j} \right) \frac{t^i}{i!} \quad (42.11)$$

where $\Gamma(\cdot)$ is the **Gamma function** introduced with the gamma distribution.

Note also that $\alpha > 0$ and $\beta > 0$.

42.2.4 Relations

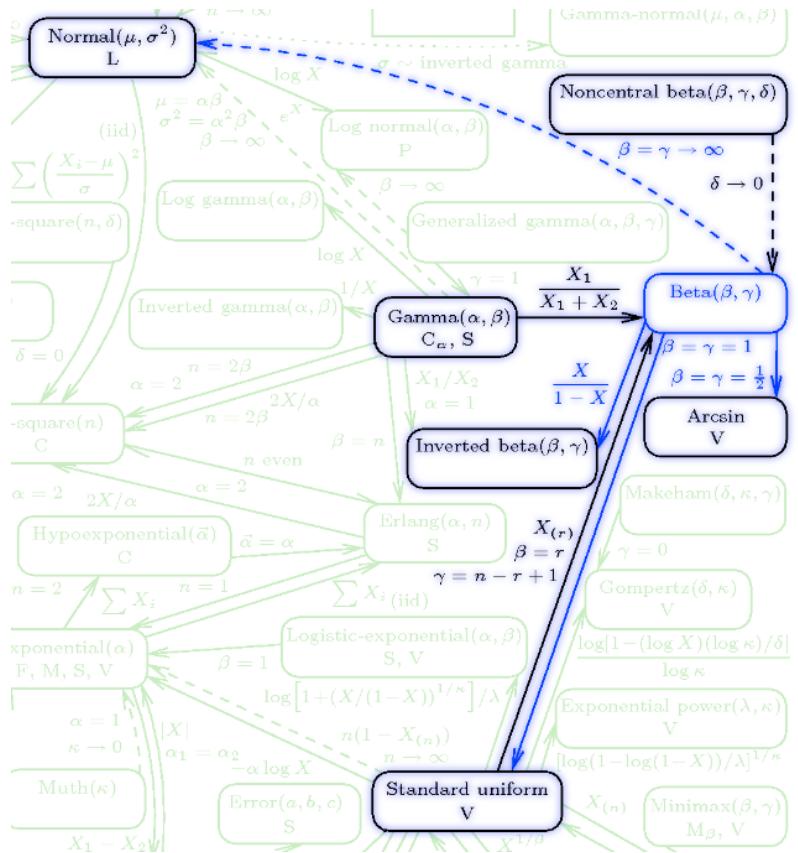


Figure 42.1: Relations of the Beta distribution

The standard $Uniform(0, 1)$ distribution is a special case of the beta distribution with $\alpha = \beta = 1$.

$$Uniform(0, 1) = Beta(1, 1) \quad (42.12)$$

42.2.5 As a prior

The Beta distribution is often used as a prior for parameters that are **probabilities**, since it takes values from 0 and 1.

During prior elicitation the parameters can be set using

1. the mean: $\frac{\alpha}{\alpha+\beta}$ which I would interpret here as count of successes over trials prior to seeing the data.
2. variance: Equation 42.10 or
3. The effective sample size which is $\alpha + \beta$ (see course 1 lesson 7.3 for the derivation).

42.3 The Cauchy Distribution

42.3.1 PDF

$$\text{Cauchy}(y | \mu, \sigma) = \frac{1}{\pi\sigma} \frac{1}{1 + ((y - \mu)/\sigma)^2} \mathbb{I}_{\mu \in \mathbb{R}} \mathbb{I}_{\sigma \in \mathbb{R}^+} \mathbb{I}_{y \in \mathbb{R}} \quad (\text{PDF}) \quad (42.13)$$

42.3.2 CDF

$$F(x | \mu, \sigma) = \frac{1}{2} + \frac{1}{\pi} \arctan\left(\frac{x - \mu}{\sigma}\right) \quad (\text{CDF}) \quad (42.14)$$

$$\mathbb{E}(X) = \text{undefined}$$

$$\mathbb{V}ar[X] = \text{undefined}$$

42.3.3 As a prior

The Cauchy despite having no mean or variance is recommended as a prior for regression coefficients in Logistic regression. see (Andrew Gelman et al. 2008) this is analyzed and discussed in (Ghosh, Li, and Mitra 2018)

42.4 Double Exponential Distribution (Laplace)

$$\text{DoubleExponential}(y | \mu, \sigma) = \frac{1}{2\sigma} \exp\left(-\frac{|y - \mu|}{\sigma}\right) \quad (\text{PDF}) \quad (42.15)$$

42.5 The Gamma Distribution

42.5.1 Story

If X_1, X_2, \dots, X_n are independent (and identically distributed $\text{Exp}(\lambda)$) **waiting times** between successive events, then **the total waiting time for all n events** to occur $Y = \sum X_i$ will follow a gamma distribution with shape parameter $\alpha = n$ and rate parameter $\beta = \lambda$:

We denote this as:

$$Y = \sum_{i=0}^N \text{Exp}_i(\lambda) \sim \text{Gamma}(\alpha = N, \beta = \lambda) \quad (42.16)$$

42.5.2 PDF

$$f(y | \alpha, \beta) = \frac{\beta^\alpha}{\Gamma(\alpha)} y^{\alpha-1} e^{-\beta y} \mathbb{I}_{y \geq 0}(y) \quad (42.17)$$

42.5.3 Moments

$$\mathbb{E}[Y] = \frac{\alpha}{\beta} \quad (42.18)$$

$$\mathbb{V}ar[Y] = \frac{\alpha}{\beta^2} \quad (42.19)$$

where $\Gamma(\cdot)$ is the gamma function, a generalization of the factorial function which can accept non-integer arguments. If n is a positive integer, then $\Gamma(n) = (n - 1)!$.

Note also that $\alpha > 0$ and $\beta > 0$.

42.5.4 Relations

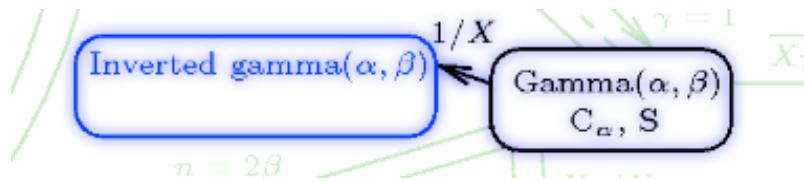


Figure 42.2: Relations of the Gamma Distribution

The exponential distribution is a special case of the Gamma distribution with $\alpha = 1$. The gamma distribution commonly appears in statistical problems, as we will see in this course. It is used to model positive-valued, continuous quantities whose distribution is right-skewed. As α increases, the gamma distribution more closely resembles the normal distribution.

42.6 Inverse Gamma Distribution

42.6.1 PDF

$$\text{InvGamma}(y|\alpha, \beta) = \frac{1}{\Gamma(\alpha)} \frac{\beta^\alpha}{y^{\alpha+1}} e^{-\frac{\beta}{y}} \mathbb{I}_{\alpha \in \mathbb{R}^+} \mathbb{I}_{\beta \in \mathbb{R}^+} \mathbb{I}_{y \in \mathbb{R}^+} \quad (\text{PDF}) \quad (42.20)$$

42.6.2 Moments

$$\mathbb{E}[X] = \frac{\beta}{\alpha - 1} \quad \text{Expectation} \quad (42.21)$$

$$\text{Var}[X] = \frac{\beta^2}{(\alpha - 1)^2(\alpha - 2)} \quad \text{Variance} \quad (42.22)$$

42.7 The Z or Standard normal distribution

- The Standard normal distribution is given by:

$$Z \sim \mathcal{N}[1, 0] \quad (42.23)$$

$$f(z) = \frac{1}{\sqrt{2\pi}} e^{-\frac{z^2}{2}} \quad (42.24)$$

$$\mathcal{L}(\mu, \sigma) = \prod_{i=1}^n \frac{1}{2\pi\sigma} e^{-\frac{(x_i - \mu)^2}{2\sigma^2}} \quad (42.25)$$

$$\begin{aligned} \ell(\mu, \sigma) &= \log \mathcal{L}(\mu, \sigma) \\ &= -\frac{n}{2} \log(2\pi) - n \log \sigma - \sum_{i=1}^n \frac{(x_i - \mu)^2}{2\sigma^2} \end{aligned} \quad (42.26)$$

$$\mathbb{E}(Z) = 0 \quad (\text{Expectation}) \quad \mathbb{V}ar(Z) = 1 \quad (\text{Variance}) \quad (42.27)$$

42.8 The Normal Distribution

The **normal**, or **Gaussian distribution** is one of the most important distributions in statistics.

It arises as the limiting distribution of sums (and averages) of random variables. This is due to the Section 45.1. Because of this property, the normal distribution is often used to model the “errors,” or unexplained variations of individual observations in regression models.

Now consider $X = \sigma Z + \mu$ where $\sigma > 0$ and μ is any real constant. Then $E(X) = E(\sigma Z + \mu) = \sigma E(Z) + \mu = \sigma_0 + \mu = \mu$ and $\text{Var}(X) = \text{Var}(\sigma Z + \mu) = \sigma^2 \text{Var}(Z) + 0 = \sigma^2 \cdot 1 = \sigma^2$

Then, X follows a normal distribution with mean μ and variance σ^2 (standard deviation σ) denoted as

$$X \sim N[\mu, \sigma^2] \quad (42.28)$$

42.8.1 PDF

$$f(x|\mu, \sigma^2) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2\sigma^2}(x-\mu)^2} \quad (42.29)$$

42.8.2 Moments

$$\mathbb{E}(x) = \mu \quad (42.30)$$

$$Var(x) = \sigma^2 \quad (42.31)$$

- The normal distribution is symmetric about the mean μ and is often described as a bell-shaped curve.
- Although X can take on any real value (positive or negative), more than 99% of the probability mass is concentrated within three standard deviations of the mean.

The normal distribution has several desirable properties.

One is that if $X_1 \sim N(\mu_1, \sigma_1^2)$ and $X_2 \sim N(\mu_2, \sigma_2^2)$ are independent, then $X_1 + X_2 \sim N(\mu_1 + \mu_2, \sigma_1^2 + \sigma_2^2)$.

Consequently, if we take the average of n Independent and Identically Distributed (IID) Normal random variables we have:

$$\bar{X} = \frac{1}{n} \sum_{i=1}^n X_i \sim N\left(\mu, \frac{\sigma^2}{n}\right) \quad (42.32)$$

```
import numpy as np
from scipy.stats import norm
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)

n, p = 5, 0.4
mean, var, skew, kurt = norm.stats(moments='mvsk')
print(f'{mean:.2f}, {var:.2f}, {skew:.2f}, {kurt:.2f}')

mean=0.00, var=1.00, skew=0.00, kurt=0.00

x = np.linspace(norm.ppf(0.01),
                 norm.ppf(0.99), 100)
ax.plot(x, norm.pdf(x),
         'r-', lw=5, alpha=0.6, label='norm pdf')

rv = norm()
```

```

ax.plot(x, rv.pdf(x), 'k-', lw=2, label='frozen pdf')
r = norm.rvs(size=1000)

ax.hist(r, density=True, bins='auto', histtype='stepfilled', alpha=0.2)

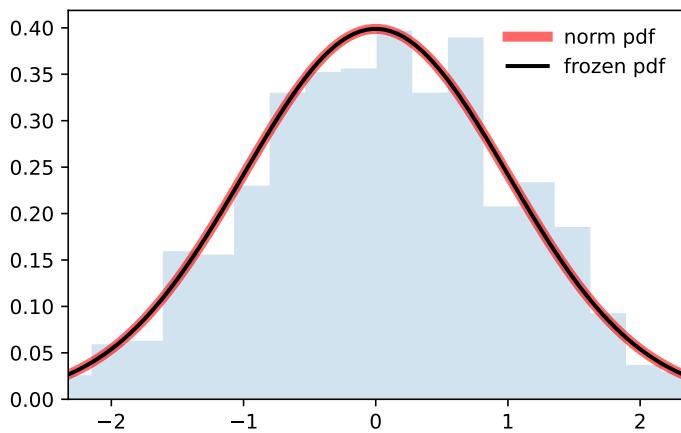
(array([0.00371025, 0.01113076, 0.01855127, 0.02597177, 0.05936405,
       0.06307431, 0.1595409 , 0.15583064, 0.23003571, 0.33021255,
       0.35247407, 0.35618432, 0.39699711, 0.33021255, 0.38957661,
       0.20777419, 0.23374596, 0.18551267, 0.09275633, 0.03710253,
       0.02968203, 0.02597177, 0.          , 0.01484101]), array([-3.2277657 , -2.95824233, -2.68871896, -2.41919559, -2.14967222,
      -1.88014884, -1.61062547, -1.3411021 , -1.07157873, -0.80205536,
      -0.53253199, -0.26300862,  0.00651475,  0.27603813,  0.5455615 ,
      0.81508487,  1.08460824,  1.35413161,  1.62365498,  1.89317835,
      2.16270173,  2.4322251 ,  2.70174847,  2.97127184,  3.24079521]), [])

ax.set_xlim([x[0], x[-1]])

(-2.3263478740408408, 2.3263478740408408)

ax.legend(loc='best', frameon=False)
plt.show()

```



42.9 The t-Distribution

If we have normal data, we can use (Equation 42.32) to help us estimate the mean μ . Reversing the transformation from the previous section, we get:

$$\frac{\hat{X} - \mu}{\sigma/\sqrt{n}} \sim N(0, 1) \quad (42.33)$$

However, we may not know the value of σ . If we estimate it from data, we can replace it with $S = \sqrt{\sum_i \frac{(X_i - \hat{X})^2}{n-1}}$, the sample standard deviation. This causes the expression (Equation 42.33) to no longer be distributed as a Standard Normal; but as a standard *t-distribution* with $\nu = n - 1$ degrees of freedom

$$X \sim t[\nu] \quad (42.34)$$

$$f(t | \nu) = \frac{\Gamma(\frac{\nu+1}{2})}{\Gamma(\frac{\nu}{2})\sqrt{\nu\pi}} \left(1 + \frac{t^2}{\nu}\right)^{-(\frac{\nu+1}{2})} \mathbb{I}_{t \in \mathbb{R}} \quad (\text{PDF}) \quad (42.35)$$

where $\Gamma(w) = \int_0^\infty t^{w-1} e^{-t} dt$ is the gamma function

$$f(t | \nu) = \frac{1}{\sqrt{\nu} B(\frac{1}{2}, \frac{\nu}{2})} \left(1 + \frac{t^2}{\nu}\right)^{-(\nu+1)/2} \mathbb{I}_{t \in \mathbb{R}} \quad (\text{PDF}) \quad (42.36)$$

where $B(u, v) = \int_0^1 t^{u-1} (1-t)^{v-1} dt$ is the beta function

$$F(t) = \int_{-\infty}^t f(u) du = 1 - \frac{1}{2} I_{x(t)}\left(\frac{\nu}{2}, \frac{1}{2}\right) \quad (\text{CDF})$$

where $I_{x(t)} = \frac{B(x; u, v)}{B(u, v)}$ is the regularized Beta function

where $B(w; u, v) = \int_0^w t^{u-1} (1-t)^{v-1} dt$ is the incomplete Beta function

and $B(u, v) = \int_0^1 t^{u-1} (1-t)^{v-1} dt$ is the (complete) beta function
(42.37)

$$\int_{-\infty}^t f(u) du = \frac{1}{2} + t \frac{\Gamma\left(\frac{1}{2}(\nu+1)\right)}{\sqrt{\pi\nu}\Gamma\left(\frac{\nu}{2}\right)} {}_2F_1\left(\frac{1}{2}, \frac{1}{2}(\nu+1); \frac{3}{2}; -\frac{t^2}{\nu}\right)$$

$$\mathcal{L}(\mu, \sigma, \nu) = \prod_{i=1}^n \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\Gamma\left(\frac{\nu}{2}\right)} \left(1 + \frac{(x_i - \mu)^2}{\sigma^2\nu}\right)^{-\frac{\nu+1}{2}} \quad (42.38)$$

$$\begin{aligned} \ell(\mu, \sigma, \nu) &= \log \mathcal{L}(\mu, \sigma, \nu) \\ &= \sum_{i=1}^n \left[\log \Gamma\left(\frac{\nu+1}{2}\right) - \log \sqrt{\nu\pi} - \log \Gamma\left(\frac{\nu}{2}\right) - \frac{\nu+1}{2} \log \left(1 + \frac{(x_i - \mu)^2}{\sigma^2\nu}\right) \right] \\ &= n \log \Gamma\left(\frac{\nu+1}{2}\right) - n \log \sqrt{\nu\pi} - n \log \Gamma\left(\frac{\nu}{2}\right) - \frac{\nu+1}{2} \sum_{i=1}^n \log \left(1 + \frac{(x_i - \mu)^2}{\sigma^2\nu}\right). \end{aligned} \quad (42.39)$$

$$\mathbb{E}[Y] = 0 \quad \text{if } \nu > 1 \quad (42.40)$$

$$\text{Var}[Y] = \frac{\nu}{\nu-2} \quad \text{if } \nu > 2 \quad (42.41)$$

42.10 Location Scale Parametrization t-distribution

$$X = \mu + \sigma T$$

The resulting distribution is also called the non-standardized Student's t-distribution.

this is another parameterization of the student-t with:

- location $\mu \in \mathbb{R}^+$
- scale $\sigma \in \mathbb{R}^+$
- degrees of freedom $\nu \in \mathbb{R}^+$

$$f(x | \mu, \sigma, \nu) = \frac{\left(\frac{\nu}{\nu + \frac{(x-\mu)^2}{\sigma^2}}\right)^{\frac{\nu+1}{2}}}{\sqrt{\nu}\sigma B\left(\frac{\nu}{2}, \frac{1}{2}\right)} \quad (42.42)$$

where $B(u, v) = \int_0^1 t^{u-1}(1-t)^{v-1} dt$ is the beta function

$$F(\mu, \sigma, \nu) = \begin{cases} \frac{1}{2} I_{\frac{\nu\sigma^2}{(x-\mu)^2+\nu\sigma^2}} \left(\frac{\nu}{2}, \frac{1}{2}\right), & x \leq \mu \\ \frac{1}{2} \left(I_{\frac{(x-\mu)^2}{(x-\mu)^2+\nu\sigma^2}} \left(\frac{1}{2}, \frac{\nu}{2}\right) + 1\right), & \text{Otherwise} \end{cases} \quad (42.43)$$

where $I_w(u, v)$ is the regularized incomplete beta function:

$$I_w(u, v) = \frac{B(w; u, v)}{B(u, v)}$$

where $B(w; u, v)$ is the *incomplete beta function*:

$$B(w; u, v) = \int_0^w t^{u-1}(1-t)^{v-1} dt$$

And $B(u, v)$ is the (complete) beta function

$$\mathbb{E}[X] = \begin{cases} \mu, & \text{if } \nu > 1 \\ \text{undefined} & \text{otherwise} \end{cases} \quad (42.44)$$

$$\text{Var}[X] = \frac{\nu\sigma^2}{\nu - 2} \quad (42.45)$$

The *t distribution* is symmetric and resembles the Normal Distribution but with thicker tails. As the degrees of freedom increase, the t distribution looks more and more like the standard normal distribution.



Figure 42.3: William Sealy Gosset AKA Student



Historical Note on The William Sealy Gosset A.K.A Student

The student-t distribution is due to Gosset, William Sealy (1876-1937) who was an English statistician, chemist and brewer who served as Head Brewer of Guinness and Head Experimental Brewer of Guinness and was a pioneer of modern statistics. He is known for his pioneering work on small **sample experimental designs**. Gosset published under the pseudonym "Student" and developed most famously Student's t-distribution – originally called Student's "z" – and "Student's test of statistical significance".

He was told to use a Pseudonym and choose 'Student' after a predecessor at Guinness published a paper that leaked trade secrets. Gosset was a friend of both Karl Pearson and Ronald Fisher. Fisher suggested a correction to the student-t using the degrees of freedom rather than the sample size. Fisher is also credited with helping to publicize its use.

for a full biography see (Pearson et al. 1990)

42.11 The Exponential Distribution

42.11.1 Story

The *Exponential distribution* models the waiting time between events for events with a rate lambda. Those events, typically, come from a *Poisson process*

The *exponential distribution* is often used to model the waiting time between random events. Indeed, if the waiting times between successive events are independent then they form an $Exp(\lambda)$ distribution. Then for any fixed time window of length t , the number of events occurring in that window will follow a *Poisson distribution* with mean $t\lambda$.

$$X \sim Exp[\lambda] \quad (42.46)$$

42.11.2 PDF

$$f(x | \lambda) = \frac{1}{\lambda} e^{-\frac{x}{\lambda}} (x) \mathbb{I}_{\lambda \in \mathbb{R}^+} \mathbb{I}_{x \in \mathbb{R}_0^+} \quad (\text{PDF}) \quad (42.47)$$

42.11.3 CDF

$$F(x | \lambda) = 1 - e^{-\lambda x} \quad (\text{CDF})$$

$$\mathcal{L}(\lambda) = \prod_{i=1}^n \lambda e^{-\lambda x_i} \quad (42.48)$$

$$\begin{aligned} \ell(\lambda) &= \log \mathcal{L}(\lambda) \\ &= \sum_{i=1}^n \log(\lambda) - \lambda x_i \\ &= n \log(\lambda) - \lambda \sum_{i=1}^n x_i \end{aligned} \quad (42.49)$$

42.11.4 Moments

$$\mathbb{E}(x) = \lambda \quad (42.50)$$

$$\mathbb{V}ar[X] = \lambda^2 \quad (42.51)$$

$$\mathbb{M}_X(t) = \frac{1}{1 - \lambda t} \quad t < \frac{1}{\gamma} \quad (42.52)$$

42.11.5 Special cases:

- **Weibull** $Y = X^{\frac{1}{\gamma}}$
- **Rayleigh** $Y = \sqrt{\frac{2X}{\lambda}}$
- **Gumbel** $Y = \alpha - \gamma \log(\frac{X}{\lambda})$

42.11.6 Properties:

- memoryless

```

import numpy as np
from scipy.stats import expon
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)

n, p = 5, 0.4
mean, var, skew, kurt = expon.stats(moments='mvsk')
print(f'{mean=:1.2f}, {var=:1.2f}, {skew=:1.2f}, {kurt=:1.2f}')

mean=1.00, var=1.00, skew=2.00, kurt=6.00

x = np.linspace(expon.ppf(0.01), expon.ppf(0.99), 100)
ax.plot(x, expon.pdf(x), 'r-', lw=5, alpha=0.6, label='expon pdf')

rv = expon()
ax.plot(x, rv.pdf(x), 'k-', lw=2, label='frozen pdf')

r = expon.rvs(size=1000)

ax.hist(r, density=True, bins='auto', histtype='stepfilled', alpha=0.2)

(array([0.88043953, 0.56698462, 0.61308093, 0.51166905, 0.37338011,
       0.31806454, 0.32267417, 0.19821413, 0.18438524, 0.08758299,
       0.15672746, 0.08758299, 0.03687705, 0.02765779, 0.05531557,
       0.05531557, 0.01843852, 0.03687705, 0.01382889, 0.00460963,
       0.00921926, 0.00921926, 0.00460963, 0.          , 0.00460963,
       0.          , 0.00921926, 0.          , 0.00460963, 0.00921926,
       0.00460963, 0.          , 0.00460963]), array([2.27809067e-03, 2.19215193e-01, 4.36152295e-01, 6.53089397e-01,
       8.70026499e-01, 1.08696360e+00, 1.30390070e+00, 1.52083780e+00,
       1.73777491e+00, 1.95471201e+00, 2.17164911e+00, 2.38858621e+00,
       2.60552332e+00, 2.82246042e+00, 3.03939752e+00, 3.25633462e+00,
       3.47327172e+00, 3.69020883e+00, 3.90714593e+00, 4.12408303e+00,
       4.34102013e+00, 4.55795723e+00, 4.77489434e+00, 4.99183144e+00,
       5.20876854e+00, 5.42570564e+00, 5.64264274e+00, 5.85957985e+00,
       6.07651695e+00, 6.29345405e+00, 6.51039115e+00, 6.72732825e+00,
       6.94426536e+00, 7.16120246e+00]), [

```

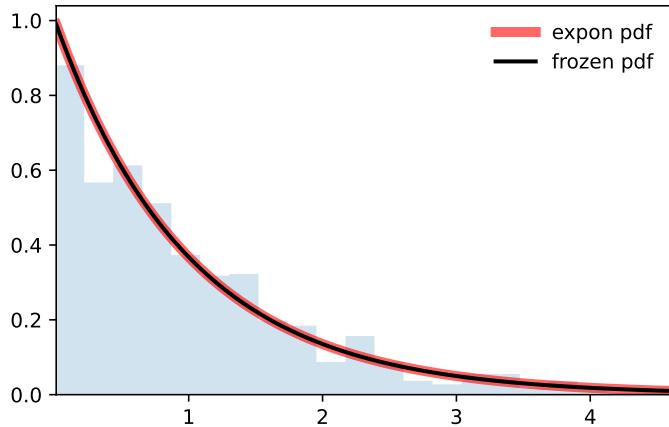
```

ax.set_xlim([x[0], x[-1]])

(0.010050335853501442, 4.605170185988091)

ax.legend(loc='best', frameon=False)
plt.show()

```



42.12 LogNormal Distribution

The long normal arises when a log transform is applied to the normal distribution.

$$\text{LogNormal}(y \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi} \sigma} \frac{1}{y} \exp\left(-\frac{1}{2} \left(\frac{\log y - \mu}{\sigma}\right)^2\right) \mathbb{I}_{\mu \in \mathbb{R}} \mathbb{I}_{\sigma \in \mathbb{R}^+} \mathbb{I}_{y \in \mathbb{R}^+} \quad (\text{PDF}) \quad (42.53)$$

42.13 Pareto Distribution

$$\text{Pareto}(y|y_{\min}, \alpha) = \frac{\alpha y_{\min}^\alpha}{y^{\alpha+1}} \mathbb{I}_{\alpha \in \mathbb{R}^+} \mathbb{I}_{y_{\min} \in \mathbb{R}^+} \mathbb{I}_{y \geq y_{\min} \in \mathbb{R}^+} \quad (\text{PDF}) \quad (42.54)$$

$$\text{Pareto_Type_2}(y|\mu, \lambda, \alpha) = \frac{\alpha}{\lambda} \left(1 + \frac{y - \mu}{\lambda}\right)^{-(\alpha+1)} \mathbb{I}_{\mu \in \mathbb{R}} \mathbb{I}_{\lambda \in \mathbb{R}^+} \mathbb{I}_{\alpha \in \mathbb{R}^+} \mathbb{I}_{y \geq \mu \in \mathbb{R}} \quad (\text{PDF}) \quad (42.55)$$

$$\mathbb{E}[X] = \frac{\alpha y_{\min}}{\alpha - 1} \mathbb{I}_{\alpha > 1} \quad (\text{expectation}) \quad (42.56)$$

$$\mathbb{V}ar[X] = \frac{\alpha y_{\min}^2}{(\alpha - 1)^2(\alpha - 2)} \mathbb{I}_{\alpha > 2} \quad (\text{variance}) \quad (42.57)$$

42.14 Weibull Distribution

42.14.1 PDF

$$\text{Weibull}(y|\alpha, \sigma) = \frac{\alpha}{\sigma} \left(\frac{y}{\sigma}\right)^{\alpha-1} e^{-(\frac{y}{\sigma})^\alpha} \mathbb{I}_{\alpha \in \mathbb{R}^+} \mathbb{I}_{\sigma \in \mathbb{R}^+} \mathbb{I}_{y \in \mathbb{R}^+} \quad (\text{PDF})$$

42.15 Chi Squared Distribution

The chi squared distribution is a special case of the gamma. It is widely used in hypothesis testing and the construction of confidence intervals. It is parameterized using parameter ν for the degrees of freedom

42.15.1 PDF:

$$\text{ChiSquare}(y | \nu) = \frac{2^{-\nu/2}}{\Gamma(\nu/2)} y^{\nu/2-1} \exp\left(-\frac{1}{2} y\right) \mathbb{I}_{\nu \in \mathbb{R}^+} \mathbb{I}_{y \in \mathbb{R}} \quad (\text{PDF}) \quad (42.58)$$

42.15.2 CDF:

$$\frac{1}{\Gamma(\nu/2)} \gamma\left(\frac{\nu}{2}, \frac{x}{2}\right) \quad (\text{CDF}) \quad (42.59)$$

42.15.3 MOMENTS

$$\mathbb{E}[X] = \nu \quad (42.60)$$

$$\mathbb{V}ar[X] = 2\nu \quad (42.61)$$

42.16 Logistic Distribution

$$\text{Logistic}(y|\mu, \sigma) = \frac{1}{\sigma} \exp\left(-\frac{y-\mu}{\sigma}\right) \left(1 + \exp\left(-\frac{y-\mu}{\sigma}\right)\right)^{-2} \mathbb{I}_{\mu \in \mathbb{R}} \mathbb{I}_{\sigma \in \mathbb{R}^+} \mathbb{I}_{y \in \mathbb{R}} \quad (\text{PDF}) \quad (42.62)$$

42.17 F Distribution

The F-distribution or F-ratio, arises frequently as the null distribution of a test statistic, in the analysis of variance (ANOVA) and other F-tests.

F-Distribution

42.17.1 PDF

$$\frac{\sqrt{\frac{(d_1 x)^{d_1} d_2^{d_2}}{(d_1 x + d_2)^{d_1 + d_2}}}}{x \text{B}\left(\frac{d_1}{2}, \frac{d_2}{2}\right)} \quad (42.63)$$

42.17.2 CDF

$$\mathbb{I}_{\frac{d_1 x}{d_1 x + d_2}} \left(\frac{d_1}{2}, \frac{d_2}{2}\right) \quad (42.64)$$

42.17.3 Moments

$$\mathbb{E}[X] = \frac{d_2}{d_2 - 2} \quad (42.65)$$

$$\mathbb{V}ar[X] = \frac{2 d_2^2 (d_1 + d_2 - 2)}{d_1 (d_2 - 2)^2 (d_2 - 4)} \quad (42.66)$$

43 Appendix: Exponents & Logarithms

43.1 Exponents

Exponents are of the form a^x where:

- a (called the base) and
- x (called the exponent) is any real number.

Recall that $a^0 = 1$. Exponents have the following useful properties

1. $a^x \cdot a^y = a^{x+y}$
2. $(a^x)^y = a^{x \cdot y}$

Note: that the first property requires that both terms have the same base a.

We cannot simplify $a^x \cdot b^y$ if $a \neq b$.

- One common base is the number e which is approximately equal to 2.7183.
- The function e^x is so common in mathematics and has its own symbol $e^x = \exp(x)$.
- Because $e > 0$ we have $e^x > 0$ for all real numbers x
- $\lim_{x \rightarrow \infty} x = e^{-x} = 0$.

43.2 Natural Logarithms

We will need to manipulate long products of probabilities. Since there often comprise small fractions, their calculation on computers can be problematic due to the underflow of floats. We will therefore prefer to convert these products into sums of logarithms.

Definition 43.1 (The Logarithm). A log is the inverse of a power. We can use (Equation 43.1).

$$y = a^x \implies \log_a(y) = x \quad (43.1)$$

Definition 43.2 (The Natural log). The natural logarithm function has base e and is written without the subscript

$$\log_e(y) = \log(y) \quad (43.2)$$

Theorem 43.1 (Logs take positive values). *logs only exist for values greater than 0*

$$\forall x (e^x > 0) \implies \exists \log(y) \iff y > 0$$

We can use the properties of exponents from the previous section to obtain some important properties of logarithms:

Definition 43.3 (Log of a product). we can use Equation 43.3 to convert a log of a product to a sum of logs.

$$\log(x \cdot y) = \log(x) + \log(y) \quad (43.3)$$

Definition 43.4 (Log of a quotient). we can use Equation 43.4 to convert a log of a quotient to a difference of logs.

$$\log\left(\frac{x}{y}\right) = \log(x) - \log(y) \quad (43.4)$$

Definition 43.5 (Log of a power). we can use Equation 43.5 to convert a log of a variable raised to a power into the product.

$$\log(x^b) = b \cdot \log(x) \quad (43.5)$$

Definition 43.6 (Log of one). we can use (Equation 43.6) to replace a log of 1 with zero since $\log(1) = 0$

$$\log(1) = 0 \quad (43.6)$$

43.2.1 Log of exponent

we can use (Equation 43.7) to cancel a log of an exponent since the log is the inverse function of the exponent.

$$\exp(\log(y)) = \log(\exp(y)) = y \quad (43.7)$$

Example 43.1 (Logarithm).

$$\log \frac{5^2}{10} = 2\log(5) - \log(10) \approx 0.916.$$

Definition 43.7 (Change of base for a log). we can use (Equation 43.8) to change the base of a logarithm.

$$\log_b(a) = \frac{\log_c(a)}{\log_c(n)} \quad (43.8)$$

Definition 43.8 (Derivative of a Log). we can use (Equation 43.9) to differentiate a log.

$$\frac{d}{dx} \log_b(x) = \frac{1}{x} \quad (43.9)$$

- Because the natural logarithm is a monotonically increasing one-to-one function, finding the x which maximizes any (positive-valued function) $f(x)$ is equivalent to maximizing $\log(f(x))$.
- This is useful because we often take derivatives to maximize functions.
- If $f(x)$ has product terms, then $\log(f(x))$ will have summation terms, which are usually simpler when taking derivatives.

44 Appendix: The Law of Large Numbers

44.1 Law of large numbers

Suppose we observe data $D = \{x_1, \dots, x_n\}$ with each $x_i \sim F$.

By the strong law of large numbers the empirical distribution \hat{F}_n based on data $D = \{x_1, \dots, x_n\}$ converges to the true underlying distribution F as $n \rightarrow \infty$ almost surely:

$$\hat{F}_n \xrightarrow{a.s.} F$$

The [Glivenko–Cantelli](#) asserts that the convergence is uniform. Since the strong law implies the weak law we also have convergence in probability:

$$\hat{F}_n \xrightarrow{P} F$$

Correspondingly, for $n \rightarrow \infty$ the average $E_{\hat{F}_n}(h(x)) = \frac{1}{n} \sum_{i=1}^n h(x_i)$ converges to the expectation $E_F(h(x))$.

45 Appendix: The Central Limit Theorem

45.1 Central Limit Theorem

The Central Limit Theorem is one of the most important results in statistics, stating that with sufficiently large sample sizes, the sample average approximately follows a normal distribution. This underscores the importance of the normal distribution, as well as most of the methods commonly used which make assumptions about the data being normally distributed.

Let's first stop and think about what it means for the sample average to have a distribution. Imagine going to the store and buying a bag of your favorite brand of chocolate chip cookies. Suppose the bag has 24 cookies in it. Will each cookie have the exact same number of chocolate chips in it? It turns out that if you make a batch of cookies by adding chips to dough and mixing it really well, then putting the same amount of dough onto a baking sheet, the number of chips per cookie closely follows a Poisson distribution. (In the limiting case of chips having zero volume, this is exactly a Poisson process.) Thus we expect there to be a lot of variability in the number of chips per cookie. We can model the number of chips per cookie with a Poisson distribution. We can also compute the average number of chips per cookie in the bag. For the bag we have, that will be a particular number. But there may be more bags of cookies in the store. Will each of those bags have the same average number of chips? If all of the cookies in the store are from the same industrial-sized batch, each cookie will individually have a Poisson number of chips. So the average number of chips in one bag may be different from the average number of chips in another bag. Thus we could hypothetically find out the average number of chips for each bag in the store. And we could think about what the distribution of these averages is, across the bags in the store, or all the bags of cookies in the world. It is this distribution of averages that the central limit theorem says is approximately a normal distribution, with the same

mean as the distribution for the individual cookies, but with a standard deviation that is divided by the square root of the number of samples in each average (i.e., the number of cookies per bag).

Theorem 45.1 (Central Limit Theorem). *Let X_1, \dots, X_n be independent and identically distributed (IID) with $\mathbb{E}(X_i) = \mu$ and $\text{Var}(X_i) = \sigma^2 < \infty$*

Then:

$$\lim_{n \rightarrow \infty} \sqrt{n} \sum_{i=0}^n \frac{1}{n} \frac{(X_i - \mu)}{\sigma} = \sum_{i=0}^n \frac{X_i - \mu}{\sqrt{n}\sigma} = N(0, 1)$$

That is, \hat{X}_n is approximately normally distributed with mean and variance $\frac{\sigma}{\sqrt{n}}$ or standard deviation $\frac{\sigma}{\sqrt{n}}$.

46 Appendix: Conjugate Priors

46.1 Conjugate Priors

Table 46.1: Conjugate prior

| Likeli- | Con- | | | |
|-------------|------------------------------|-------|---|--|
| hood | jugate | prior | Posterior | Posterior predictive |
| Bernoulli | $\text{Beta}(\alpha, \beta)$ | | $\text{Beta}\left(\alpha + \sum_{i=1}^n x_i, \beta + n \mathbb{P}_r \sum_{i=1}^n \mathbf{x}_i \mathbf{1}\right)$ | $= \frac{\alpha'}{\alpha' + \beta'}$ |
| Binomial | $\text{Beta}(m, p)$ | | $\text{Beta}\left(\alpha + \sum_{i=1}^n x_i, \beta + \sum_{i=1}^n \text{BetaBin}(\tilde{x} \alpha'_i, \beta')\right)$ | |
| NegBin | $\text{Beta}(rn, r)$ | | $\text{Beta}\left(\alpha + rn, \beta + \sum_{i=1}^n x_i\right)$ | $\text{NegBin}(\tilde{x} \alpha', \beta')$ |
| Poisson | λ | | $\text{Gamma}(k, \theta)$ | $\text{NB}\left(\tilde{x} k', \frac{1}{\theta' + 1}\right)$ |
| Poisson | λ | | $\text{Gamma}(k, \theta)$ | $\text{NB}\left(\tilde{x} \alpha', \frac{\beta'}{1 + \beta'}\right)$ |
| Categorical | $p, cats = k$ | | $\text{Dir}((p)_k)$ | $\Pr(\tilde{x} = i) = \frac{\alpha'_i}{\sum_i \alpha'_i} = \frac{\alpha_i + c_i}{\sum_i \alpha_i + n}$ |
| Multinomial | $p, cats = k$ | | $\text{Dir}((p)_k)$ | $\text{DirMult}(\tilde{\mathbf{x}} \alpha')$ |

| Likeli- | Con- | |
|----------------|--------|---|
| hood | jugate | |
| | prior | Posterior |
| Hypergeometric | Beta | $\text{Beta}(\alpha + \sum_{i=1}^n x_i, \beta + \sum_{i=1}^n N_i - \sum_{i=1}^n x_i)$ |
| n) | N) | |
| Geometric | Beta | $\text{Beta}(\alpha + n, \beta + \sum_{i=1}^n x_i)$ |

47 Appendix: Link Function

47.1 Link function

In statistics, a generalized linear model (GLM) is a flexible generalization of ordinary linear regression. The GLM generalizes linear regression by allowing the linear model to be related to the response variable via a link function and by allowing the magnitude of the variance of each measurement to be a function of its predicted value.

The link function provides the relationship between the linear predictor and the [mean](#) of the distribution function. There are many commonly used link functions, and their choice is informed by several considerations. There is always a well-defined *canonical* link function which is derived from the exponential of the response's [density function](#). However, in some cases, it makes sense to try to match the [domain](#) of the link function to the [range](#) of the distribution function's mean, or use a non-canonical link function for algorithmic purposes, for example [Bayesian probit regression](#).

When using a distribution function with a canonical parameter θ , the canonical link function is the function that expresses θ in terms of μ i.e. $\theta = b(\mu)$. For the most common distributions, the mean μ is one of the parameters in the standard form of the distribution's Density function, and then $b(\mu)$ is the function as defined above that maps the density function into its canonical form. When using the canonical link function, $b(\mu) = \theta = \mathbf{X}^T \boldsymbol{\beta}$, which allows $\mathbf{X}^T \mathbf{Y}$ to be a *sufficient statistic* for $\boldsymbol{\beta}$.

Following is a table of several exponential-family distributions in common use and the data they are typically used for, along with the canonical link functions and their inverses (sometimes referred to as the mean function, as done here).

Table 47.1: Common distributions with typical uses and canonical link functions

| Distri-
bution | Support | Link
name | Link fn | Mean
fn |
|-------------------|------------------|---------------------|--|------------|
| Normal | \mathbb{R} | Identity | $\mathbf{X}\beta = \mu$ | |
| Exponential | \mathbb{R}_0^+ | Negative
inverse | $\mathbf{X}\beta = -\mu^{-1}$ | |
| Gamma | \mathbb{R}_0^+ | Negative
inverse | $\mathbf{X}\beta = -\mu^{-1}$ | |
| Inverse | \mathbb{R}_0^+ | Inverse
squared | $\mathbf{X}\beta = \mu^{-2}$ | |
| Gamma | | | | |
| Poisson | \mathbb{N}_0 | Log | $\mathbf{X}\beta = \ln(\mu)$ | |
| Bernoulli | $\{0, 1\}$ | Logit | $\mathbf{X}\beta = \ln(\frac{\mu}{1-\mu})$ | |
| Binomial | | Logit | $\mathbf{X}\beta = \ln(\frac{\mu}{n-\mu})$ | |
| Categorical | | Logit | $\mathbf{X}\beta = \ln(\frac{\mu}{1-\mu})$ | |
| Multinomial | | Logit | $\mathbf{X}\beta = \ln(\frac{\mu}{1-\mu})$ | |

In the cases of the exponential and gamma distributions, the domain of the canonical link function is not the same as the permitted range of the mean. In particular, the linear predictor may be positive, which would give an impossible negative mean. When maximizing the likelihood, precautions must be taken to avoid this. An alternative is to use a non-canonical link function.

In the case of the Bernoulli, binomial, categorical and multinomial distributions, the support of the distributions is not the same type of data as the parameter being predicted. In all of these cases, the predicted parameter is one or more probabilities, i.e. real numbers in the range $[0, 1]$. The resulting model is known as Logistic regression (or Multinomial lo-

gistic regression in the case that K-way rather than binary values are being predicted).

For the Bernoulli and binomial distributions, the parameter is a single probability, indicating the likelihood of occurrence of a single event. The Bernoulli still satisfies the basic condition of the generalized linear model in that, even though a single outcome will always be either 0 or 1, the expected value will nonetheless be a real-valued probability, i.e. the probability of occurrence of a “yes” (or 1) outcome. Similarly, in a binomial distribution, the expected value is Np , i.e. the expected proportion of “yes” outcomes will be the probability to be predicted.

For categorical and multinomial distributions, the parameter to be predicted is a K -vector of probabilities, with the further restriction that all probabilities must add up to 1. Each probability indicates the likelihood of occurrence of one of the K possible values. For the multinomial distribution, and for the vector form of the categorical distribution, the expected values of the elements of the vector can be related to the predicted probabilities similarly to the binomial and Bernoulli distributions.

Credits:

This page is based on the [Generalized linear model](#) article on Wikipedia, which is licensed under the [Creative Commons Attribution-ShareAlike 3.0 Unported License](#). By Wikimedia contributors, available under [CC BY-SA 3.0](#).

The text has been modified for clarity and conciseness.

48 Bayes by backprop

48.1 Introduction

This appendix reviews of a method to introduce weight uncertainty into *neural networks* called the “*Bayes by Backprop*” method introduced in (Blundell et al. 2015), where, the main question is how to determine the parameters of the distribution for each network weight. I learned about it from Probabilistic Deep Learning with TensorFlow 2 by Dr Kevin Webster, S reading from based this on

The authors note that prior work which considered uncertainty at the hidden unit (H_i) an approach that allows to state the uncertainty with respect to a particular observation and which is an easier problem since the number of weight is greater by two orders of magnitude. But considering the uncertainty in the weights is complementary, in the sense that it captures uncertainty about which neural network is appropriate, leading to regularization of the weights and model averaging. This weight uncertainty can be used to drive the exploration/exploitation in contextual bandit problems using Thompson sampling . Weights with greater uncertainty introduce more variability into the decisions made by the network, naturally leading to **exploration**. As more data are observed, the uncertainty can decrease, allowing the decisions made by the network to become more deterministic as the environment is better understood.

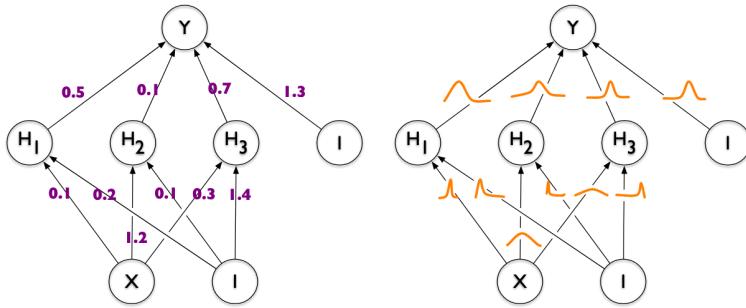


Figure 1. Left: each weight has a fixed value, as provided by classical backpropagation. Right: each weight is assigned a distribution, as provided by Bayes by Backprop.

Figure 48.1: Fig. 1 from (Blundell et al. 2015) contrasting traditional and Bayesian neural networks

In a traditional neural network, as shown in the upper left, each weight has a single value. But in the true value for these weights is not certain. Much of this uncertainty comes from imperfect training data, which is an approximation of the the distribution of the data generating process from which the data were sampled. Recall that this is called *epistemic* uncertainty, which we expect to decrease as the amount of training data increases.

In this method, we want to include such uncertainty in deep learning models. This is done by changing each weight from a single deterministic value to a *probability distribution*. We then learn the parameters of this distribution. Consider a neural network weight w_i . In a standard (deterministic) neural network, this has a single value \hat{w}_i , learnt via backpropagation. In a neural network with weight uncertainty, each weight is represented by a probability distribution, and the *parameters* of this distribution are learned via backpropagation. Suppose, for example, that each weight has a normal distribution. This has two parameters: a mean μ_i and a standard deviation σ_i .

- Classic deterministic NN: $w_i = \hat{w}_i$
- NN with weight uncertainty represented by normal distribution:
 $w_i \sim N(\hat{\mu}_i, \hat{\sigma}_i)$.

Since the weights are uncertain, the feedforward value of some input x_i is not constant. A single feedforward value is determined in two steps: 1. Sample each network weight from their respective distributions – this gives a single set of network weights. 2. Use these weights to determine a feedforward value \hat{y}_i .

Hence, the key question is how to determine the parameters of the distribution for each network weight. The paper introduces exactly such a scheme, called *Bayes by Backprop*.

48.2 Bayesian learning

Note: We use the notation P to refer to a probability density. For simplicity, we'll only consider continuous distributions (which have a density). In the case of discrete distributions, P would represent a probability mass and integrals should be changed to sums. However, the formulae are the same.

What you need to know now is that Bayesian methods can be used to calculate the distribution of a model parameter given some data. In the context of weight uncertainty in neural networks, this is convenient, since we are looking for the distribution of weights (model parameters) given some (training) data. The key step relies on Bayes' theorem. This theorem states, in mathematical notation, that

$$\Pr(w \mid D) = \frac{\Pr(D \mid w)\Pr(w)}{\int \Pr(D \mid w')\Pr(w')dw'}$$

where the terms mean the following:

- D is some data, e.g. x and y value pairs: $D = \{(x_1, y_1), \dots, (x_n, y_n)\}$. This is sometimes called the *evidence*.
- w is the value of a model weight.
- $\Pr(w)$ is called the *prior*. This is our “prior” belief on the probability density of a model weight, i.e. the distribution that we postulate before seeing any data.
- $\Pr(D \mid w)$ is the *likelihood* of having observed data D given weight w . It is precisely the same likelihood used to calculate the negative log-likelihood.

- $\Pr(w | D)$ is the *posterior* density of the distribution of the model weight at value w , given our training data. It is called *posterior* since it represents the distribution of our model weight *after* taking the training data into account.

Note that the term $\int \Pr(D | w') \Pr(w') dw' = \Pr(D)$ does not depend on w (as the w' is an integration variable). It is only a normalization term. For this reason, we will from this point on write Bayes' theorem as

$$\Pr(w | D) = \frac{\Pr(D | w) \Pr(w)}{\Pr(D)}.$$

Bayes' theorem gives us a way of combining data with some “prior belief” on model parameters to obtain a distribution for these model parameters that considers the data, called the *posterior distribution*.

48.3 Bayesian neural network with weight uncertainty – in principle

The above formula gives a way to determine the distribution of each weight in the neural network:

1. Pick a prior density $\Pr(w)$.
2. Using training data D , determine the likelihood $\Pr(D | w)$.
3. Determine the posterior density $\Pr(w | D)$ using Bayes' theorem.

This is the distribution of the NN weight.

While this works in principle, in many practical settings it is difficult to implement. The main reason is that the normalization constant $\int \Pr(D | w') \Pr(w') dw' = \Pr(D)$ may be very difficult to calculate, as it involves solving or approximating a complicated integral. For this reason, approximate methods, such as *Variational Bayes* described below, are often employed.

48.4 Variational Bayes

Variational Bayes methods approximate the posterior distribution with a second function, called a *variational posterior*. This function has a known functional form, and hence avoids the need to determine the posterior $\Pr(w | D)$ exactly. Of course, approximating a function with another one has some risks, since the approximation may be very bad, leading to a posterior that is highly inaccurate. In order to mediate this, the variational posterior usually has a number of parameters, denoted by θ , that are tuned so that the function approximates the posterior as well as possible. Let's see how this works below.

Instead of $\Pr(w | D)$, we assume the network weight has density $q(w | \theta)$, parameterized by θ . $q(w | \theta)$ is known as the *variational posterior*. We want $q(w | \theta)$ to approximate $\Pr(w | D)$, so we want the “difference” between $q(w | \theta)$ and $\Pr(w | D)$ to be as small as possible. This “difference” between the two distributions is measured by the **Kullback-Leibler divergence** D_{KL} (note that this is **unrelated** to the D we use to denote the data). The Kullback-Leibler divergence between two distributions with densities $f(x)$ and $g(x)$ respectively is defined as

$$D_{KL}(f(x) \| g(x)) = \int f(x) \log \left(\frac{f(x)}{g(x)} \right) dx$$

Note that this function has value 0 (indicating no difference) when $f(x) \equiv g(x)$, which is the result we expect. We use the convention that $\frac{0}{0} = 1$ here.

Viewing the data D as a constant, the Kullback-Leibler divergence between $q(w | \theta)$ and $\Pr(w | D)$ is hence:

$$\begin{aligned} D_{KL}(q(w | \theta) \| \Pr(w | D)) &= \int q(w | \theta) \log \left(\frac{q(w | \theta)}{\Pr(w | D)} \right) dw \\ &= \int q(w | \theta) \log \left(\frac{q(w | \theta)\Pr(D)}{\Pr(D | w)\Pr(w)} \right) dw \\ &= \int q(w | \theta) \log \Pr(D) dw + \int q(w | \theta) \log \left(\frac{q(w | \theta)}{\Pr(w)} \right) dw - \int q(w | \theta) \log \Pr(D | w) dw \\ &= \log \Pr(D) + D_{KL}(q(w | \theta) \| \Pr(w)) - \mathbb{E}_{q(w | \theta)}(\log \Pr(D | w)) \end{aligned}$$

where, in the last line, we have used

$$\int q(w | \theta) \log \mathbb{P}r(D) dw = \log \mathbb{P}r(D) \int q(w | \theta) dw = \log \mathbb{P}r(D)$$

since $q(w | \theta)$ is a probability distribution and hence integrates to 1. If we consider the data D to be constant, the first term is a constant also, and we may ignore it when minimizing the above. Hence, we are left with the function

$$L(\theta | D) = D_{KL}(q(w | \theta) || \mathbb{P}r(w)) - \mathbb{E}_{q(w|\theta)}(\log \mathbb{P}r(D | w))$$

Note that this function depends only on θ and D , since w is an integration variable. This function has a nice interpretation as the sum of: - The Kullback-Leibler divergence between the variational posterior $q(w | \theta)$ and the prior $\mathbb{P}r(w)$. This is called the *complexity cost*, and it depends on θ and the prior but not the data D . - The expectation of the negative log likelihood $\log \mathbb{P}r(D | w)$ under the variational posterior $q(w | \theta)$. This is called the *likelihood cost* and it depends on θ and the data but not the prior.

$L(\theta | D)$ is the loss function that we minimize to determine the parameter θ . Note also from the above derivation, that we have

$$\begin{aligned} \log \mathbb{P}r(D) &= \mathbb{E}_{q(w|\theta)}(\log \mathbb{P}r(D | w)) - D_{KL}(q(w | \theta) || \mathbb{P}r(w)) + D_{KL}(q(w | \theta) || \mathbb{P}r(w | D)) \\ &\geq \mathbb{E}_{q(w|\theta)}(\log \mathbb{P}r(D | w)) - D_{KL}(q(w | \theta) || \mathbb{P}r(w)) =: ELBO \end{aligned}$$

which follows because $D_{KL}(q(w | \theta) || \mathbb{P}r(w | D))$ is non negative. The final expression on the right hand side is therefore a lower bound on the log-evidence, and is called the *evidence lower bound*, often shortened to *ELBO*. The {ELBO} is the negative of our loss function, so minimizing the loss function is equivalent to maximizing the ELBO.

Maximizing the ELBO requires a trade off between the KL term and expected log-likelihood term. On the one hand, the divergence between $q(w | \theta)$ and $\mathbb{P}r(w)$ should be kept small, meaning the variational posterior shouldn't be too different to the prior. On the other, the variational posterior parameters should maximize the expectation of the log-likelihood $\log \mathbb{P}r(D | w)$, meaning the model assigns a high likelihood to the data.

48.5 A backpropagation scheme

48.5.1 The idea

We can use the above ideas to create a neural network with weight uncertainty, which we will call a *Bayesian neural network*. From a high level, this works as follows. Suppose we want to determine the distribution of a particular neural network weight w .

1. Assign the weight a prior distribution with density $\mathbb{P}r(w)$, which represents our beliefs on the possible values of this network before any training data. This may be something simple, like a unit Gaussian. Furthermore, this prior distribution will usually not have any trainable parameters.
2. Assign the weight a variational posterior with density $q(w | \theta)$ with some trainable parameter θ .
3. $q(w | \theta)$ is the approximation for the weight's posterior distribution. Tune θ to make this approximation as accurate as possible as measured by the ELBO.

The remaining question is then how to determine θ . Recall that neural networks are typically trained via a backpropagation algorithm, in which the weights are updated by perturbing them in a direction that reduces the loss function. We aim to do the same here, by updating θ in a direction that reduces $L(\theta | D)$.

Hence, the function we want to minimise is

$$\begin{aligned} L(\theta | D) &= D_{KL}(q(w | \theta) \| \mathbb{P}r(w)) - \mathbb{E}_{q(w|\theta)}(\log \mathbb{P}r(D | w)) \\ &= \int q(w | \theta)(\log q(w | \theta) - \log \mathbb{P}r(D | w) - \log \mathbb{P}r(w))dw. \end{aligned}$$

In principle, we could take derivatives of $L(\theta | D)$ with respect to θ and use this to update its value. However, this involves doing an integral over w , and this is a calculation that may be impossible or very computationally expensive. Instead, we want to write this function as an expectation and use a Monte Carlo approximation to calculate derivatives. At present, we can write this function as

$$L(\theta | D) = \mathbb{E}_{q(w|\theta)}(\log q(w | \theta) - \log \mathbb{P}r(D | w) - \log \mathbb{P}r(w))$$

However, taking derivatives with respect to θ is difficult because the underlying distribution the expectation is taken with respect to depends on θ . One way we can handle this is with the *reparameterization trick*.

48.5.2 The reparameterization trick

The reparameterization trick is a way to move the dependence on θ around so that an expectation may be taken independently of it. It's easiest to see how this works with an example. Suppose $q(w | \theta)$ is a Gaussian, so that $\theta = (\mu, \sigma)$. Then, for some arbitrary $f(w; \mu, \sigma)$, we have

$$\begin{aligned}\mathbb{E}_{q(w|\mu,\sigma)}(f(w; \mu, \sigma)) &= \int q(w | \mu, \sigma) f(w; \mu, \sigma) dw \\ &= \int \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(w - \mu)^2\right) f(w; \mu, \sigma) dw \\ &= \int \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{1}{2}\epsilon^2\right) f(\mu + \sigma\epsilon; \mu, \sigma) d\epsilon \\ &= \mathbb{E}_{\epsilon \sim N(0,1)}(f(\mu + \sigma\epsilon; \mu, \sigma))\end{aligned}$$

where we used the change of variable $w = \mu + \sigma\epsilon$. Note that the dependence on $\theta = (\mu, \sigma)$ is now only in the integrand and we can take derivatives with respect to μ and σ :

$$\frac{\partial}{\partial \mu} \mathbb{E}_{q(w|\mu,\sigma)}(f(w; \mu, \sigma)) = \frac{\partial}{\partial \mu} \mathbb{E}_{\epsilon \sim N(0,1)}(f(\mu + \sigma\epsilon; \mu, \sigma)) = \mathbb{E}_{\epsilon \sim N(0,1)} \frac{\partial}{\partial \mu} f(\mu + \sigma\epsilon; \mu, \sigma)$$

$$\frac{\partial}{\partial \sigma} \mathbb{E}_{q(w|\mu,\sigma)}(f(w; \mu, \sigma)) = \frac{\partial}{\partial \sigma} \mathbb{E}_{\epsilon \sim N(0,1)}(f(\mu + \sigma\epsilon; \mu, \sigma)) = \mathbb{E}_{\epsilon \sim N(0,1)} \frac{\partial}{\partial \sigma} f(\mu + \sigma\epsilon; \mu, \sigma)$$

Finally, note that we can approximate the expectation by its Monte Carlo estimate:

$$\mathbb{E}_{\epsilon \sim N(0,1)} \frac{\partial}{\partial \theta} f(\mu + \sigma\epsilon; \mu, \sigma) \approx \sum_i \frac{\partial}{\partial \theta} f(\mu + \sigma\epsilon_i; \mu, \sigma), \quad \epsilon_i \sim N(0, 1).$$

The above reparameterization trick works in cases where we can write the $w = g(\epsilon, \theta)$, where the distribution of the random variable ϵ is independent of θ .

48.5.3 Implementation

Putting this all together, for our loss function $L(\theta \mid D) \equiv L(\mu, \sigma \mid D)$, we have

$$f(w; \mu, \sigma) = \log q(w \mid \mu, \sigma) - \log \mathbb{P}r(D \mid w) - \log \mathbb{P}r(w)$$

$$\frac{\partial}{\partial \mu} L(\mu, \sigma \mid D) \approx \sum_i \left(\frac{\partial f(w_i; \mu, \sigma)}{\partial w_i} + \frac{\partial f(w_i; \mu, \sigma)}{\partial \mu} \right)$$

$$\frac{\partial}{\partial \sigma} L(\mu, \sigma \mid D) \approx \sum_i \left(\frac{\partial f(w_i; \mu, \sigma)}{\partial w_i} \epsilon_i + \frac{\partial f(w_i; \mu, \sigma)}{\partial \sigma} \right)$$

$$f(w; \mu, \sigma) = \log q(w \mid \mu, \sigma) - \log \mathbb{P}r(D \mid w) - \log \mathbb{P}r(w)$$

where $w_i = \mu + \sigma \epsilon_i$, $\epsilon_i \sim N(0, 1)$. In practice, we often only take a single sample ϵ_1 for each training point. This leads to the following backpropagation scheme:

1. Sample $\epsilon_i \sim N(0, 1)$.
2. Let $w_i = \mu + \sigma \epsilon_i$
3. Calculate

$$\nabla_{\mu} f = \frac{\partial f(w_i; \mu, \sigma)}{\partial w_i} + \frac{\partial f(w_i; \mu, \sigma)}{\partial \mu} \quad \nabla_{\sigma} f = \frac{\partial f(w_i; \mu, \sigma)}{\partial w_i} \epsilon_i + \frac{\partial f(w_i; \mu, \sigma)}{\partial \sigma}$$

4. Update the parameters with some gradient-based optimizer using the above gradients.

This is how we learn the parameters of the distribution for each neural network weight.

48.5.4 Minibatches

Note that the loss function (or negative of the ELBO) is

$$\begin{aligned} L(\theta | D) &= D_{KL}(q(w | \theta) \| \mathbb{P}r(w)) - \mathbb{E}_{q(w|\theta)}(\log \mathbb{P}r(D | w)) \\ &= D_{KL}(q(w | \theta) \| \mathbb{P}r(w)) - \sum_{j=1}^N \log \mathbb{P}r(y_j, x_j | w_j) \end{aligned}$$

where j runs over all the data points in the training data (N in total) and $w_j = \mu + \sigma\epsilon_j$ is sampled using $\epsilon_j \sim N(0, 1)$ (we assume a single sample from the approximate posterior per data point for simplicity).

If training occurs in minibatches of size B , typically much smaller than N , we instead have a loss function

$$D_{KL}(q(w | \theta) \| \mathbb{P}r(w)) - \sum_{j=1}^B \log \mathbb{P}r(y_j, x_j | w_j).$$

Note that the scaling factors between the first and second terms have changed, since before the sum ran from 1 to N , but it now runs from 1 to B . To correct for this, we should add a correction factor $\frac{N}{B}$ to the second term to ensure that its expectation is the same as before. This leads to the loss function, after dividing by N to take the average per training value, of

$$\frac{1}{N} D_{KL}(q(w | \theta) \| \mathbb{P}r(w)) - \frac{1}{B} \sum_{j=1}^B \log \mathbb{P}r(y_j, x_j | w_j).$$

By default, when Tensorflow calculates the loss function, it calculates the average across the minibatch. Hence, it already uses the factor $\frac{1}{B}$ present on the second term. However, it does not, by default, divide the first term by N . In an implementation, we will have to specify this. You'll see in the next lectures and coding tutorials how to do this.

48.5.5 Conclusion

We introduced the *Bayes by Backpropagation* method, which can be used to embed weight uncertainty into neural networks. Good job getting through it, as the topic is rather advanced. This approach allows the modelling of *epistemic* uncertainty on the model weights. We expect that, as the number of training points increases, the uncertainty on the model weights decreases. This can be shown to be the case in many settings. In the next few lectures and coding tutorials, you'll learn how to apply these methods to your own models, which will make the idea much clearer.

48.5.6 Further reading and resources

- [Bayes by backprop paper](#) (Blundell et al. 2015)
- [Wikipedia article on Bayesian inference](#)

49 Bayesian Books in R & Python

There are many books in R and Python that can help you learn more about these languages and how to use them for data analysis.

Here are some of the most popular books on R and Python:

49.1 Introduction to Probability

- [Introduction to Probability](#) by Dennis L. Sun

49.2 Books in R

- [R for Data Science](#) by Hadley Wickham & Garrett Grolemund
- [Advanced R](#) by Hadley Wickham
- [ggplot2: Elegant Graphics for Data Analysis \(3e\)](#)
- [R Graphics Cookbook, 2nd edition](#)
- [An Introduction to Statistical Learning](#)
- [Engineering Production-Grade Shiny Apps](#)
- [Forecasting: Principles and Practice \(3rd ed\)](#)
- [Exploratory Data Analysis with R](#) Roger D. Peng
- [Modern R with the tidyverse](#) by Bruno Rodrigues
- [Modern Statistics with R](#) by Benjamin S. Baumer, Daniel T. Kaplan, and Nicholas J. Horton
- [Mastering Shiny](#) by Hadley Wickham, Winston Chang, and Joe Cheng
- [Learning Statistics with R](#) by Danielle Navarro
- [Text Mining with R](#) by Julia Silge and David Robinson

49.3 Books in Python

- [An Introduction to Statistical Learning with python](#) by Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani
- [Python for Data Analysis](#) by Wes McKinney of Pandas infamy parquet and Apache Arrow
- [Python Data Science Handbook](#) by Jake VanderPlas
- [Think Stats](#) by Allen B. Downey
- [Think Bayes](#) by Allen B. Downey
- [Probabilistic Programming & Bayesian Methods for Hackers](#) by Cameron Davidson-Pilon

50 References

- Aldrich, John. 2008. “R. A. Fisher on Bayes and Bayes’ Theorem.” *Bayesian Analysis* 3 (March). <https://doi.org/10.1214/08-BA306>.
- Belsley, David A., Edwin Kuh, and Roy E. Welsch. 1980. *Regression Diagnostics*. John Wiley & Sons, Inc. <https://doi.org/10.1002/0471725153>.
- Bernoulli, J. 1713. *Ars Conjectandi [the Art of Conjecturing]*. Impensis Thurnisiorum. <https://books.google.co.il/books?id=Ba5DAAAAcA AJ>.
- Bishop, C. M. 2006. *Pattern Recognition and Machine Learning*. Information Science and Statistics. Springer (India) Private Limited. <https://www.microsoft.com/en-us/research/uploads/prod/2006/01/Bishop-Pattern-Recognition-and-Machine-Learning-2006.pdf>.
- Blundell, Charles, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. 2015. “Weight Uncertainty in Neural Networks.” <https://doi.org/10.48550/ARXIV.1505.05424>.
- Casella, G., and R. L. Berger. 2002. *Statistical Inference*. Duxbury Advanced Series in Statistics and Decision Sciences. Thomson Learning. <http://home.ustc.edu.cn/~zt001062/MaStmaterials/George%20Casella&Roger%20L.Berger--Statistical%20Inference.pdf>.
- Cook, R. Dennis. 1977b. “Detection of Influential Observation in Linear Regression.” *Technometrics* 19 (1): 15. <https://doi.org/10.2307/1268249>.
- . 1977a. “Detection of Influential Observation in Linear Regression.” *Technometrics* 19 (1): 15. <https://doi.org/10.2307/1268249>.
- Finetti, Bruno de. 1937. “La Prévision: Ses Lois Logiques, Ses Sources Subjectives.” *Annales de l’Institut Henri Poincaré* 7 (1): 1–68.
- . 2017. “Theory of Probability.” Edited by Antonio Machí and Adrian Smith. *Wiley Series in Probability and Statistics*, January. <https://doi.org/10.1002/9781119286387>.
- Fisher, R. A. 1925. *Statistical Methods for Research Workers*. 1st ed. Edinburgh Oliver & Boyd.
- Gelman, A., J. B. Carlin, H. S. Stern, D. B. Dunson, A. Vehtari, and D. B. Rubin. 2013. *Bayesian Data Analysis, Third Edition*. Chapman

- & Hall/CRC Texts in Statistical Science. Taylor & Francis. <https://books.google.co.il/books?id=ZXL6AQAAQBAJ>.
- Gelman, Andrew, Aleks Jakulin, Maria Grazia Pittau, and Yu-Sung Su. 2008. “A Weakly Informative Default Prior Distribution for Logistic and Other Regression Models.” *The Annals of Applied Statistics* 2 (4). <https://doi.org/10.1214/08-aos191>.
- Geman, Stuart, and Donald Geman. 1984. “Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images.” *IEEE Transactions on Pattern Analysis and Machine Intelligence* PAMI-6 (6): 721–41. <https://doi.org/10.1109/tpami.1984.4767596>.
- Ghosh, Joyee, Yingbo Li, and Robin Mitra. 2018. “On the Use of Cauchy Prior Distributions for Bayesian Logistic Regression.” *Bayesian Analysis* 13 (2). <https://doi.org/10.1214/17-ba1051>.
- Härdle, Wolfgang Karl, and Léopold Simar. 2019. *Applied Multivariate Statistical Analysis*. Springer International Publishing. <https://doi.org/10.1007/978-3-030-26006-4>.
- Hobbs, N. Thompson, and Mevin B. Hooten. 2015. *Bayesian Models: A Statistical Primer for Ecologists*. STU - Student edition. Princeton University Press. <http://www.jstor.org/stable/j.ctt1dr36kz>.
- Hoff, Peter D. 2009. *A First Course in Bayesian Statistical Methods*. Springer New York. <https://doi.org/10.1007/978-0-387-92407-6>.
- (<https://math.stackexchange.com/users/25097/autolatry>), Autolatry. n.d. “Why Square Brackets for Expectation.” Mathematics Stack Exchange. <https://math.stackexchange.com/q/1302543>.
- Jackman, Simon. 2009. “Bayesian Analysis for the Social Sciences.” *Wiley Series in Probability and Statistics*, October. <https://doi.org/10.1002/9780470686621>.
- Jeffreys, H. 1983. *Theory of Probability*. International Series of Monographs on Physics. Clarendon Press.
- Johnson, R. A., and D. W. Wichern. 2001. *Applied Multivariate Statistical Analysis*. Pearson Modern Classics for Advanced Statistics Series. Prentice Hall. <https://books.google.co.il/books?id=QBqlswEACA AJ>.
- Kruschke, John K. 2011. *Doing Bayesian Data Analysis: A Tutorial with R and BUGS*. Burlington, MA: Academic Press. <http://www.amazon.com/Doing-Bayesian-Data-Analysis-Tutorial/dp/0123814855>.
- McElreath, Richard. 2015. *Statistical Rethinking, a Course in r and Stan*.
- Moivre, Abraham De. 1718. *The Doctrine of Chances*. H. Woodfall. <https://tellingstorieswithdata.com>.
- Morita, Satoshi, Peter F Thall, and Peter Müller. 2008. “Determining the Effective Sample Size of a Parametric Prior.” *Biometrics* 64 (2):

- 595–602.
- Pearson, E. S., W. S. Gosset, R. L. Plackett, and G. A. Barnard. 1990. *Student: A Statistical Biography of William Sealy Gosset*. Clarendon Press. <https://books.google.co.il/books?id=LBDvAAAAMAAJ>.
- Poisson, S. -D. 2019. “English Translation of Poisson’s ”Recherches Sur La Probabilité Des Jugements En Matière Criminelle Et En Matière Civile” / ”Researches into the Probabilities of Judgements in Criminal and Civil Cases”.” <https://arxiv.org/abs/1902.02782>.
- Polya, G. 1945. *How to Solve It*. Princeton University Press. <https://doi.org/10.1515/9781400828678>.
- Ramsey, Frank P. 1926. “Truth and Probability.” In *The Foundations of Mathematics and Other Logical Essays*, edited by R. B. Braithwaite, 156–98. McMaster University Archive for the History of Economic Thought. <https://EconPapers.repec.org/RePEc:hay:hetcha:ramsey1926>.
- Sheather, Simon. 2009. *A Modern Approach to Regression with r*. Springer New York. <https://doi.org/10.1007/978-0-387-09608-7>.
- Spanos, A. 2019. *Probability Theory and Statistical Inference*. Cambridge University Press. <https://books.google.co.il/books?id=9nCiDwAAQBAJ>.
- VanderPlas, J. 2016. *Python Data Science Handbook: Essential Tools for Working with Data*. O’Reilly Media. <https://jakevdp.github.io/PythonDataScienceHandbook/>.
- Wiesenfarth, Manuel, and Silvia Calderazzo. 2020. “Quantification of Prior Impact in Terms of Effective Current Sample Size.” *Biometrics* 76 (1): 326–36. <https://doi.org/10.1111/biom.13124>.
- Wikipedia contributors. 2023a. “68–95–99.7 Rule — Wikipedia.” https://en.wikipedia.org/w/index.php?title=68%E2%80%9395%E2%80%9399.7_rule.
- . 2023b. “Functional (Mathematics) — Wikipedia, the Free Encyclopedia.” [https://en.wikipedia.org/w/index.php?title=Functional_\(mathematics\)&oldid=1148699341](https://en.wikipedia.org/w/index.php?title=Functional_(mathematics)&oldid=1148699341).
- Williams, D. A. 1987. “Generalized Linear Model Diagnostics Using the Deviance and Single Case Deletions.” *Applied Statistics* 36 (2): 181. <https://doi.org/10.2307/2347550>.

Index

| | | |
|--|--|--|
| Bernoulli trial, 49,
396,
402 | 426 | Gelman Rubin di-
agnostic, 219 |
| Beta distribution, 423 | effective sample
size, 110 | Geometric distri-
bution, 61,
412 |
| Cauchy Distribu-
tion, 426 | ELBO, 457 | Gibbs sampling al-
gorithm, 198 |
| Chi Squared
Distri-
bution, 439 | Empirical Bayes,
124 | Gosset, William
Sealy, 435 |
| collinearity, 301 | Event, 22 | Gumbel, 436 |
| conjugate normal
model, 113 | Exponential distri-
bution, 58,
435 | Hypergeometric
Distri-
bution, 63 |
| Deviance informa-
tion
crite-
rion, 257 | F-Distribution,
440 | Inverse Gamma
Distri-
bution, 428 |
| DIC,
see Deviance
information
criterion 257 | F-ratio, 440 | Jeffreys prior, 125 |
| Double exponen-
tial, 226 | full conditional
distribu-
tion, 198 | Kullback-Leibler,
456 |
| Double Exponen-
tial
Distri-
bution, | Gamma Distribu-
tion, 427 | Laplace Distribu-
tion, see
Double
Expo-
nential |
| | Gaussian Distribu-
tion, <i>see</i>
Normal
Distri-
bu-
tion | |
| | Gaussian distribu-
tion,
<i>see</i> Normal
distribution 54 | |

| | | |
|--------------------------|----------------------|---------------------------------------|
| Distri- | 62 | regression, linear,
222 |
| bu- | | residual analysis,
329 |
| tion | | |
| Laplace distribu- | | Sample point,
22 |
| tion, | | Sample space,
22 |
| 226 | | scale reduction |
| Laplace prior, | | factor,
220 |
| 226 | | sensitivity analysis,
126 , |
| LASSO, 227 | | 336 |
| linear model, | | shrink factor,
220 |
| 222 | | Standard normal |
| Logistic Distribu- | | distribu- |
| tion, | | 54 , |
| 440 | | 428 |
| logistic regression, | | Student, <i>see</i> |
| 300 | | Gosset, |
| LogNormal Distribu- | | William |
| tion, | | Sealy |
| 438 | | Student's t- |
| memoryless prop- | | Distribution, |
| erty, | | <i>see</i> t- |
| 412 | | distribution |
| Metropolis- | | t-Distribution, |
| Hastings, | | 432 |
| 177 | | t-distribution, |
| modeling deci- | | 57 |
| sions, | | target distribution, |
| 336 | | 178 |
| Monte Carlo, | | vague prior, |
| 158 | | 107 |
| Monte Carlo inte- | | Variational Bayes, |
| gration, | | 456 |
| 158 | | |
| multicollinearity, | | |
| 301 | | |
| Multinomial, 50 , | | |
| 404 | | |
| Multinomial | | |
| distribu- | | |
| tion, | | |
| | Rayleigh, 436 | |

| | | |
|------------------------------|------------------------------|---------------------|
| Variational Poste- | Weibull, 436 | |
| rior,
456 | Weibull Distribu- | 439 |