

Bayesian statistics

Olivier Gimenez
July 2020

Credit where credit's due

- Ruth King, Byron Morgan, Steve Brooks (our workshops and Bayesian analysis for population ecology book).
- Richard McElreath (Statistical rethinking book and lecture videos).
- Jim Albert and Jingchen Hu (Probability and Bayesian modelling book).
- Materials shared by Tristan Marh, Jason Matthiopoulos, Francisco Rodriguez Sanchez, Kerrie Mengersen and Mark Lai.

Slides codes and data

- All material prepared with R.
- R Markdown used to write reproducible material.
- Slides available on FigShare [here](#).
- Material available on Github [there](#).

Objectives

- Try and demystify Bayesian statistics, and what we call MCMC.
- Make the difference between Bayesian and Frequentist analyses.
- Understand the Methods section of ecological papers doing Bayesian stuff.
- Run Bayesian analyses, safely hopefully.

BRACE YOURSELF



What is on our plate?

1. Bayesian inference: Motivation and examples.
2. The likelihood.
3. A detour to explore priors.
4. Markov chains Monte Carlo methods (MCMC).
5. Bayesian analyses in R with the Jags software.
6. Contrast ecological hypotheses with model selection.
7. Heterogeneity and multilevel models (aka mixed models).

I want mooooore

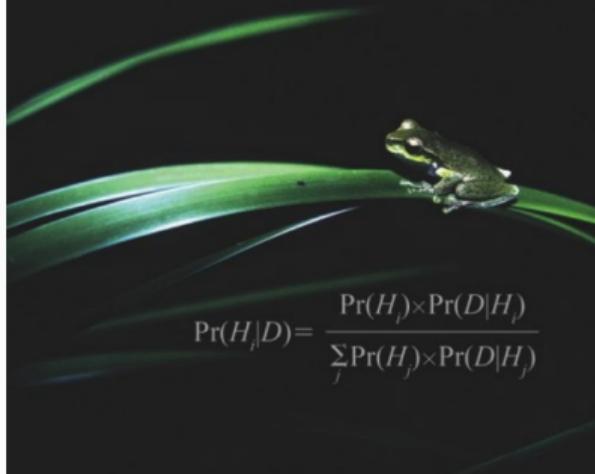
I ONLY LIKE TWO
THINGS:

THEY'RE BOTH BOOKS



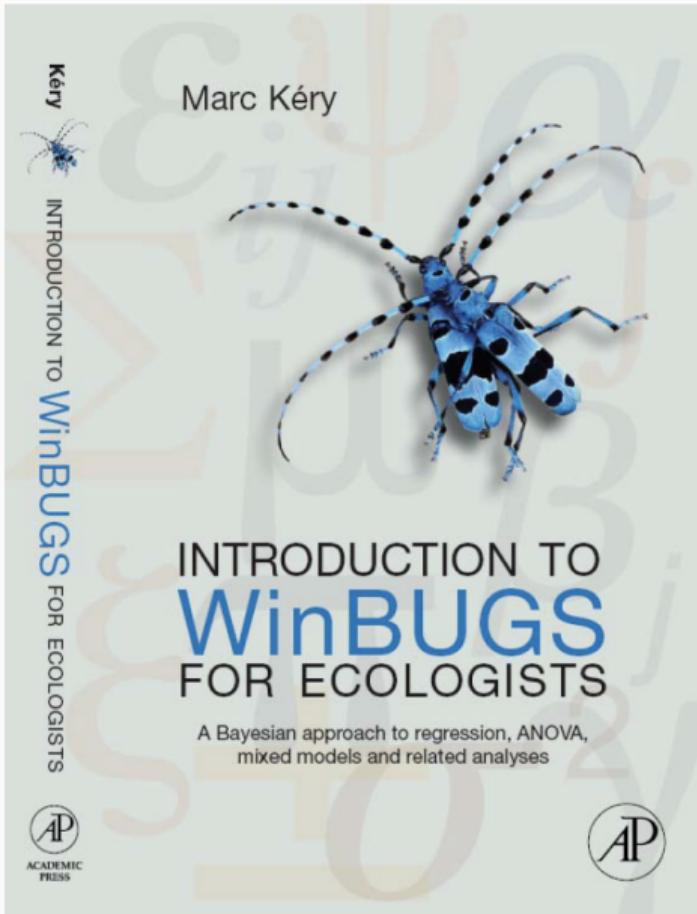
Bayesian Methods for Ecology

Michael A. McCarthy



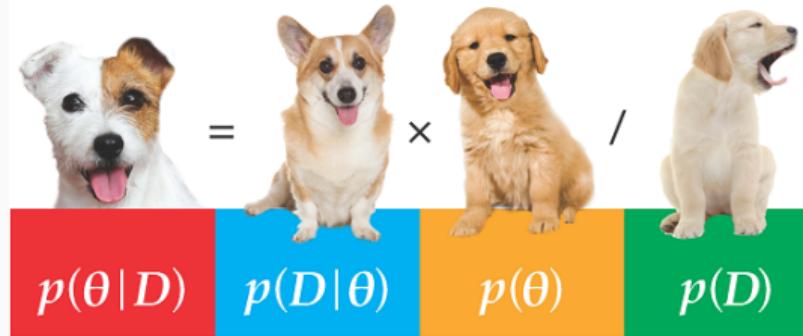
$$\Pr(H_i|D) = \frac{\Pr(H_i) \times \Pr(D|H_i)}{\sum_j \Pr(H_j) \times \Pr(D|H_j)}$$

CAMBRIDGE



Doing Bayesian Data Analysis

A Tutorial with R, JAGS, and Stan



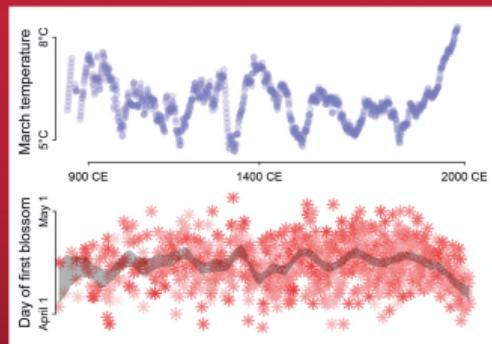
John K. Kruschke



Texts in Statistical Science

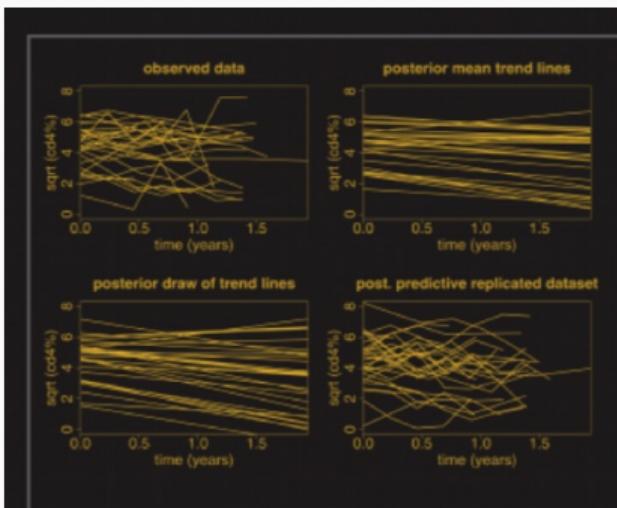
Statistical Rethinking

A Bayesian Course
with Examples in R and Stan
SECOND EDITION



Richard McElreath



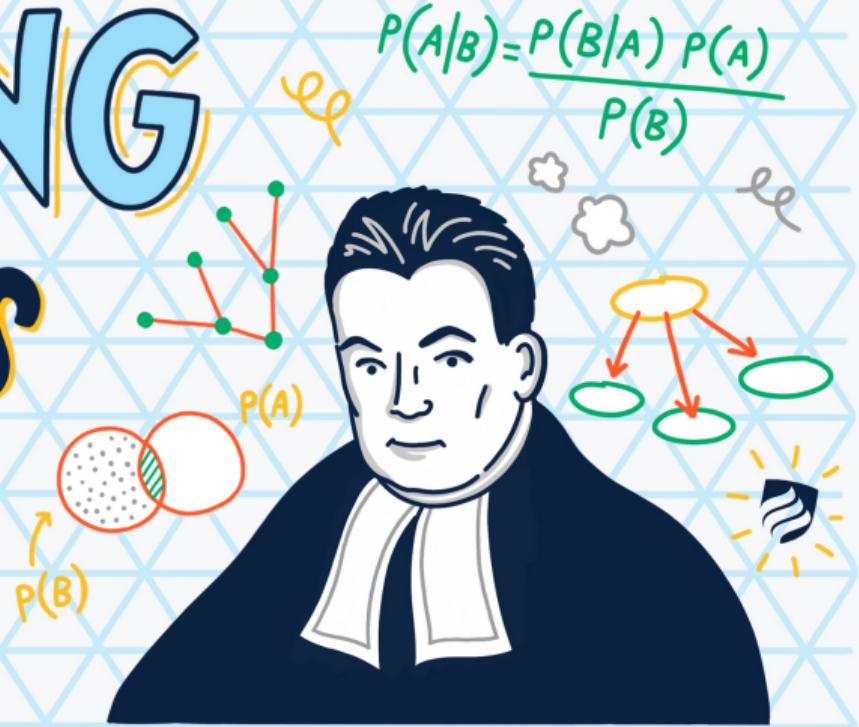


Data Analysis Using Regression and Multilevel/Hierarchical Models

ANDREW GELMAN
JENNIFER HILL

What is Bayesian inference?

THE AMAZING Thomas Bayes



A reminder on conditional probabilities

- $\Pr(A | B)$: Probability of A given B

A reminder on conditional probabilities

- $\Pr(A | B)$: Probability of A given B
- The ordering matters: $\Pr(A | B)$ is not the same as $\Pr(B | A)$.

A reminder on conditional probabilities

- $\Pr(A | B)$: Probability of A given B
- The ordering matters: $\Pr(A | B)$ is not the same as $\Pr(B | A)$.
- $\Pr(A | B) = \frac{\Pr(A \text{ and } B)}{\Pr(B)}$



HOW TO CURE VAMPIRES?

Screening for vampirism

- The chance of the test being positive given you are a vampire is $\Pr(+|\text{vampire}) = 0.90$ (**sensitivity**).

Screening for vampirism

- The chance of the test being positive given you are a vampire is $\Pr(+|\text{vampire}) = 0.90$ (**sensitivity**).
- The chance of a negative test given you are mortal is $\Pr(-|\text{mortal}) = 0.95$ (**specificity**).

What is the question?

- From the perspective of the test: Given a person is a vampire, what is the probability that the test is positive? $\Pr(+|\text{vampire}) = 0.90$.

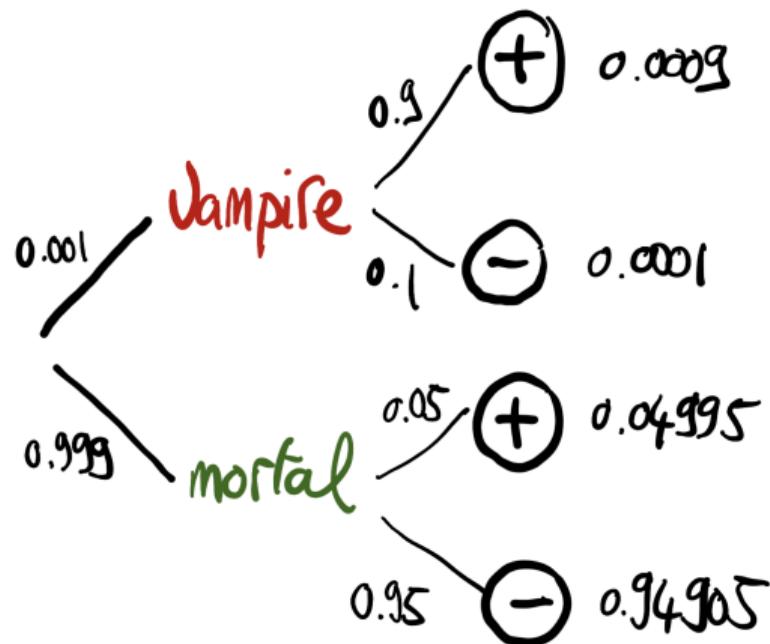
What is the question?

- From the perspective of the test: Given a person is a vampire, what is the probability that the test is positive? $\Pr(+|\text{vampire}) = 0.90$.
- From the perspective of a person: Given that the test is positive, what is the probability that this person is a vampire? $\Pr(\text{vampire}|+) = ?$

What is the question?

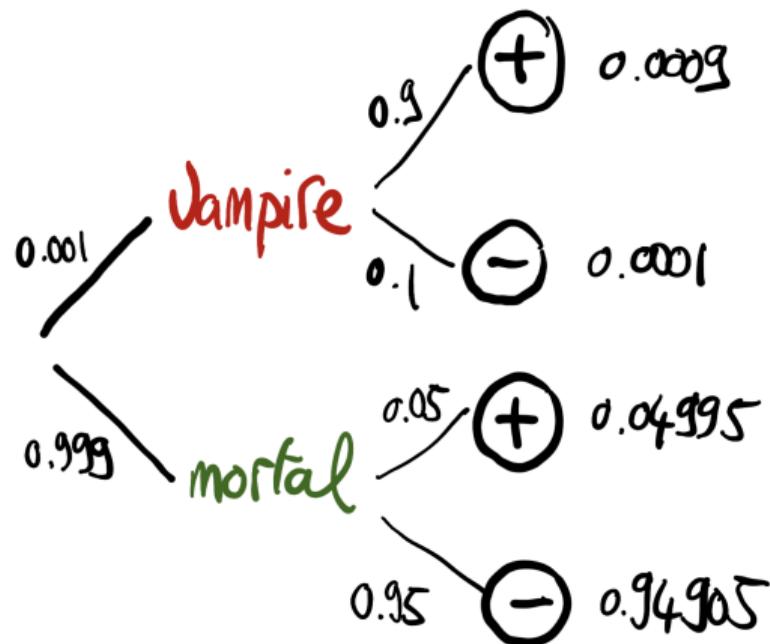
- From the perspective of the test: Given a person is a vampire, what is the probability that the test is positive? $\Pr(+|\text{vampire}) = 0.90$.
- From the perspective of a person: Given that the test is positive, what is the probability that this person is a vampire? $\Pr(\text{vampire}|+) = ?$
- Assume that vampires are rare, and represent only 0.1% of the population. This means that $\Pr(\text{vampire}) = 0.001$.

What is the answer? Bayes' theorem to the rescue!



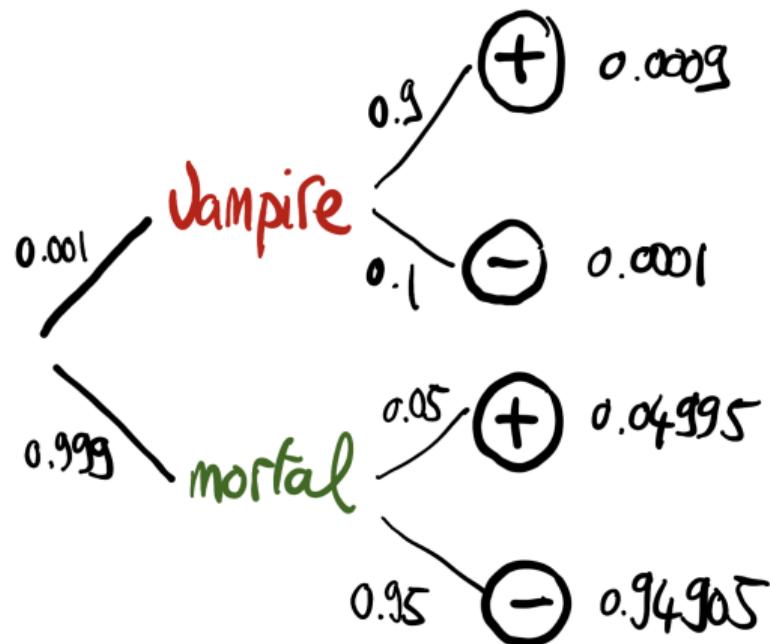
- $\Pr(\text{vampire}|+) = \frac{\Pr(\text{vampire and } +)}{\Pr(+)}$

What is the answer? Bayes' theorem to the rescue!



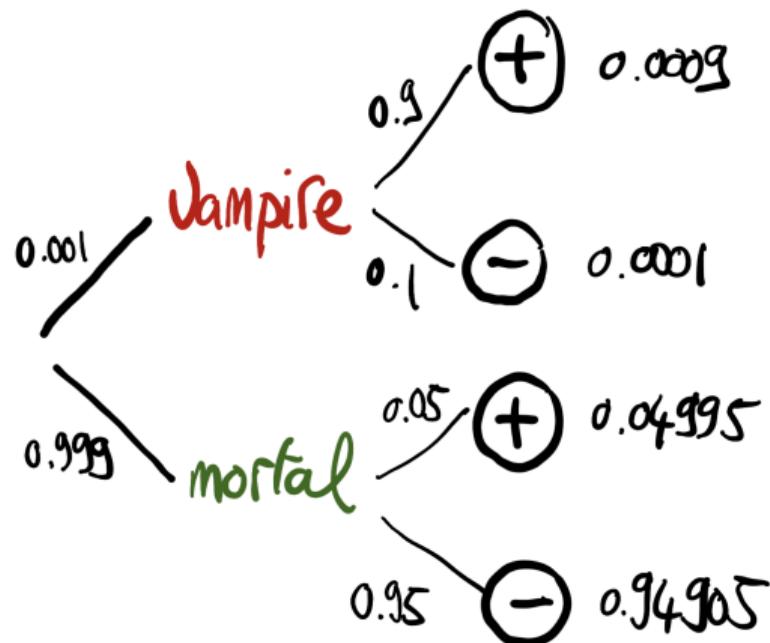
- $\Pr(\text{vampire}|+) = \frac{\Pr(\text{vampire and } +)}{\Pr(+)}$
- $\Pr(\text{vampire and } +) = \Pr(\text{vampire}) \Pr(+|\text{vampire}) = 0.0009$

What is the answer? Bayes' theorem to the rescue!



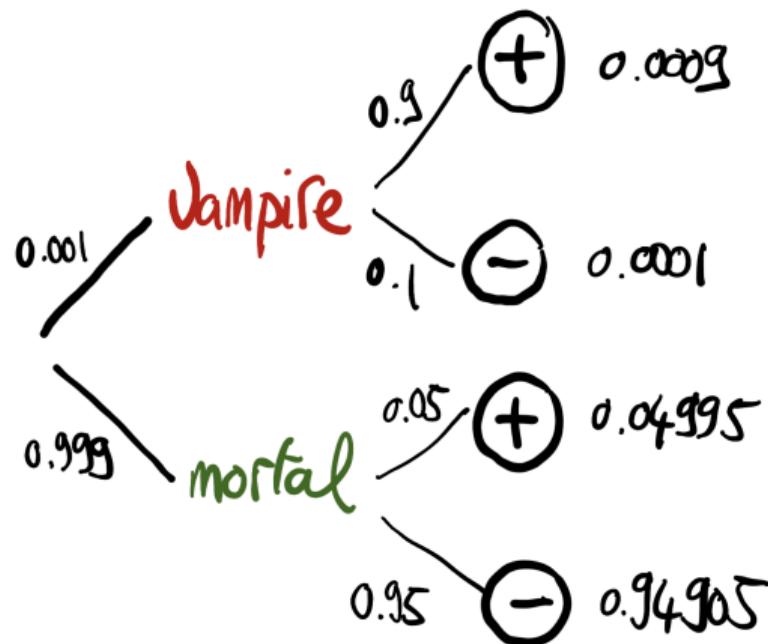
- $\Pr(\text{vampire}|+) = \frac{\Pr(\text{vampire and } +)}{\Pr(+)}$
- $\Pr(\text{vampire and } +) =$
 $\Pr(\text{vampire}) \Pr(+|\text{vampire}) = 0.0009$
- $\Pr(+) = 0.0009 + 0.04995 = 0.05085$

What is the answer? Bayes' theorem to the rescue!



- $\Pr(\text{vampire}|+) = \frac{\Pr(\text{vampire and } +)}{\Pr(+)}$
- $\Pr(\text{vampire and } +) = \Pr(\text{vampire}) \Pr(+|\text{vampire}) = 0.0009$
- $\Pr(+) = 0.0009 + 0.04995 = 0.05085$
- $\Pr(\text{vampire}|+) = 0.0009/0.05085 = 0.02$

What is the answer? Bayes' theorem to the rescue!



- $\Pr(\text{vampire}|+) = \frac{\Pr(\text{vampire and } +)}{\Pr(+)}$
- $\Pr(\text{vampire and } +) = \Pr(\text{vampire}) \Pr(+|\text{vampire}) = 0.0009$
- $\Pr(+) = 0.0009 + 0.04995 = 0.05085$
- $\Pr(\text{vampire}|+) = 0.0009/0.05085 = 0.02$

$$\Pr(\text{vampire}|+) = \frac{\Pr(+|\text{vampire}) \Pr(\text{vampire})}{\Pr(+)}$$

Your turn

Screening for vampirism

- Suppose the diagnostic test has the same sensitivity and specificity but vampirism is more common: 10% of the population is vampire.
- What is the probability that a person is a vampire, given that the test is positive?

Solution

The probability that a person is a vampire, given that the test is positive

- $\Pr(+|\text{vampire}) = 0.9$
- $\Pr(-|\text{mortal}) = 0.95$
- $\Pr(\text{vampire}) = 0.1$

$$\begin{aligned}\Pr(+) &= \Pr(+|\text{vampire}) \Pr(\text{vampire}) + \Pr(+|\text{mortal}) \Pr(\text{mortal}) \\ &= 0.9 * 0.1 + 0.05 * 0.9 \\ &= 0.135\end{aligned}$$

$$\begin{aligned}\Pr(\text{vampire}|+) &= \Pr(+|\text{vampire}) \Pr(\text{vampire}) / \Pr(+) \\ &= 0.9 * 0.1 / 0.135\end{aligned}$$

Bayes' theorem

- A theorem about conditional probabilities.
- $\Pr(B | A) = \frac{\Pr(A | B) \Pr(B)}{\Pr(A)}$

The image shows a chalkboard with the formula for Bayes' theorem written in blue chalk. The formula is:

$$P(A|B) = \frac{P(B|A) P(A)}{P(B)}$$

Bayes' theorem

- Easy to mess up with letters. Might be easier to remember when written like this:

$$\Pr(\text{hypothesis} \mid \text{data}) = \frac{\Pr(\text{data} \mid \text{hypothesis}) \Pr(\text{hypothesis})}{\Pr(\text{data})}$$

Bayes' theorem

- Easy to mess up with letters. Might be easier to remember when written like this:

$$\Pr(\text{hypothesis} \mid \text{data}) = \frac{\Pr(\text{data} \mid \text{hypothesis}) \Pr(\text{hypothesis})}{\Pr(\text{data})}$$

- The “hypothesis” is typically something unobserved or unknown. It’s what you want to learn about using the data.

Bayes' theorem

- Easy to mess up with letters. Might be easier to remember when written like this:

$$\Pr(\text{hypothesis} \mid \text{data}) = \frac{\Pr(\text{data} \mid \text{hypothesis}) \Pr(\text{hypothesis})}{\Pr(\text{data})}$$

- The “hypothesis” is typically something unobserved or unknown. It’s what you want to learn about using the data.
- For regression models, the “hypothesis” is a parameter (intercept, slopes or error terms).

Bayes' theorem

- Easy to mess up with letters. Might be easier to remember when written like this:

$$\Pr(\text{hypothesis} \mid \text{data}) = \frac{\Pr(\text{data} \mid \text{hypothesis}) \Pr(\text{hypothesis})}{\Pr(\text{data})}$$

- The “hypothesis” is typically something unobserved or unknown. It’s what you want to learn about using the data.
- For regression models, the “hypothesis” is a parameter (intercept, slopes or error terms).
- Bayes theorem tells you the probability of the hypothesis given the data.

What is doing science after all?

How plausible is some hypothesis given the data?

$$\Pr(\text{hypothesis} \mid \text{data}) = \frac{\Pr(\text{data} \mid \text{hypothesis}) \Pr(\text{hypothesis})}{\Pr(\text{data})}$$

Why is Bayesian statistics not the default?

- Due to practical problems of implementing the Bayesian approach, and some wars of male statisticians's egos, little advance was made for over two centuries.

Why is Bayesian statistics not the default?

- Due to practical problems of implementing the Bayesian approach, and some wars of male statisticians's egos, little advance was made for over two centuries.
- Recent advances in computational power coupled with the development of new methodology have led to a great increase in the application of Bayesian methods within the last two decades.

Frequentist versus Bayesian

- Typical stats problems involve estimating parameter θ with available data.

Frequentist versus Bayesian

- Typical stats problems involve estimating parameter θ with available data.
- The frequentist approach (**maximum likelihood estimation – MLE**) assumes that the parameters are fixed, but have unknown values to be estimated.

Frequentist versus Bayesian

- Typical stats problems involve estimating parameter θ with available data.
- The frequentist approach (**maximum likelihood estimation – MLE**) assumes that the parameters are fixed, but have unknown values to be estimated.
- Classical estimates generally provide a point estimate of the parameter of interest.

Frequentist versus Bayesian

- Typical stats problems involve estimating parameter θ with available data.
- The frequentist approach (**maximum likelihood estimation – MLE**) assumes that the parameters are fixed, but have unknown values to be estimated.
- Classical estimates generally provide a point estimate of the parameter of interest.
- The Bayesian approach assumes that the parameters are not fixed but have some fixed unknown distribution - a distribution for the parameter.

What is the Bayesian approach?

- The approach is based upon the idea that the experimenter begins with some prior beliefs about the system.

What is the Bayesian approach?

- The approach is based upon the idea that the experimenter begins with some prior beliefs about the system.
- And then updates these beliefs on the basis of observed data.

What is the Bayesian approach?

- The approach is based upon the idea that the experimenter begins with some prior beliefs about the system.
- And then updates these beliefs on the basis of observed data.
- This updating procedure is based upon the Bayes' Theorem:

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

What is the Bayesian approach?

- Schematically if $A = \theta$ and $B = \text{data}$, then

What is the Bayesian approach?

- Schematically if $A = \theta$ and $B = \text{data}$, then
- The Bayes' theorem

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

What is the Bayesian approach?

- Schematically if $A = \theta$ and $B = \text{data}$, then
- The Bayes' theorem

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

- Translates into:

$$\Pr(\theta | \text{data}) = \frac{\Pr(\text{data} | \theta) \ Pr(\theta)}{\Pr(\text{data})}$$

Bayes' theorem

$$\Pr(\theta \mid \text{data}) = \frac{\Pr(\text{data} \mid \theta) \Pr(\theta)}{\Pr(\text{data})}$$

Bayes' theorem

$$\Pr(\theta \mid \text{data}) = \frac{\Pr(\text{data} \mid \theta) \Pr(\theta)}{\Pr(\text{data})}$$

- **Posterior distribution:** Represents what you know after having seen the data. The basis for inference, a distribution, possibly multivariate if more than one parameter (θ).

Bayes' theorem

$$\Pr(\theta \mid \text{data}) = \frac{\Pr(\text{data} \mid \theta) \Pr(\theta)}{\Pr(\text{data})}$$

- **Posterior distribution:** Represents what you know after having seen the data. The basis for inference, a distribution, possibly multivariate if more than one parameter (θ).
- **Likelihood:** We know that guy from before, same as in the MLE approach.

Bayes' theorem

$$\Pr(\theta \mid \text{data}) = \frac{\Pr(\text{data} \mid \theta) \Pr(\theta)}{\Pr(\text{data})}$$

- **Posterior distribution:** Represents what you know after having seen the data. The basis for inference, a distribution, possibly multivariate if more than one parameter (θ).
- **Likelihood:** We know that guy from before, same as in the MLE approach.
- **Prior distribution:** Represents what you know before seeing the data. The source of much discussion about the Bayesian approach.

Bayes' theorem

$$\Pr(\theta \mid \text{data}) = \frac{\Pr(\text{data} \mid \theta) \Pr(\theta)}{\Pr(\text{data})}$$

- **Posterior distribution:** Represents what you know after having seen the data. The basis for inference, a distribution, possibly multivariate if more than one parameter (θ).
- **Likelihood:** We know that guy from before, same as in the MLE approach.
- **Prior distribution:** Represents what you know before seeing the data. The source of much discussion about the Bayesian approach.
- **$\Pr(\text{data}) = \int L(\text{data} \mid \theta) \Pr(\theta) d\theta$:** Possibly high-dimensional integral, difficult if not impossible to calculate. This is one of the reasons why we need simulation (MCMC) methods - more soon.

DID THE SUN JUST EXPLODE? (IT'S NIGHT, SO WE'RE NOT SURE.)

THIS NEUTRINO DETECTOR MEASURES WHETHER THE SUN HAS GONE NOVA.

THEN, IT ROLLS TWO DICE. IF THEY BOTH COME UP SIX, IT LIES TO US. OTHERWISE, IT TELLS THE TRUTH.

LET'S TRY.

DETECTOR! HAS THE SUN GONE NOVA?

(ROLL)

YES.



FREQUENTIST STATISTICIAN:

THE PROBABILITY OF THIS RESULT HAPPENING BY CHANCE IS $\frac{1}{36} = 0.027$. SINCE $P < 0.05$, I CONCLUDE THAT THE SUN HAS EXPLODED.



Bayesian Statistician:

BET YOU \$50 IT HASN'T.



Likelihood

Context

- Usually, when talking about probability distributions, we assume that we know the parameter values.

Context

- Usually, when talking about probability distributions, we assume that we know the parameter values.
- In the real world, it is usually the other way around.

A question of interest might be for example:

We have observed 3 births by a female during her 10 breeding attempts. What does this tell us about the true probability of getting a successful breeding attempt from this female? For the population?

- We don't know what the probability of a birth is.
- But we can calculate the probability of getting our data for different values:

```
dbinom(x=3,size=10,prob=0.1)
#> [1] 0.05739563
```

- We don't know what the probability of a birth is.
- But we can see what the probability of getting our data would be for different values:

```
dbinom(x=3,size=10,prob=0.9)  
#> [1] 8.748e-06
```

- We don't know what the probability of a birth is.
- But we can see what the probability of getting our data would be for different values:

```
dbinom(x=3, size=10, prob=0.25)
#> [1] 0.2502823
```

```
dbinom(x=3,size=10,prob=0.1)
#> [1] 0.05739563
dbinom(x=3,size=10,prob=0.9)
#> [1] 8.748e-06
dbinom(x=3,size=10,prob=0.25)
#> [1] 0.2502823
```

So we would be more likely to observe 3 births if the probability is 0.25 than 0.1 or 0.9.

The likelihood

- This reasoning is so common in statistics that it has a special name:

The likelihood

- This reasoning is so common in statistics that it has a special name:
- **The likelihood** is the probability of observing the data under a certain model.

The likelihood

- This reasoning is so common in statistics that it has a special name:
- **The likelihood** is the probability of observing the data under a certain model.
- The data are known, we usually consider the likelihood as a function of the model parameters $\theta_1, \theta_2, \dots, \theta_p$

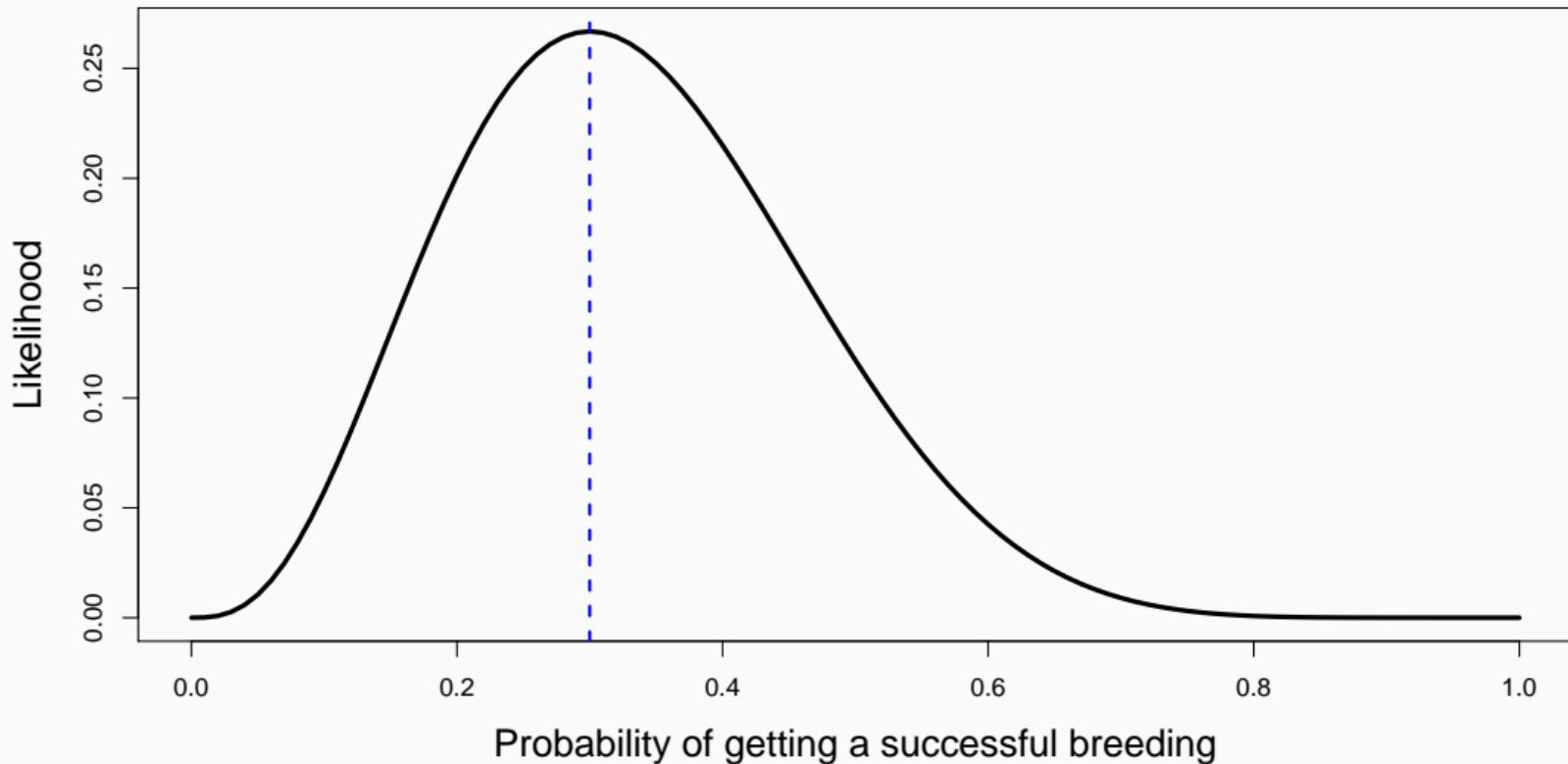
$$L = P(\theta_1, \theta_2, \dots, \theta_p \mid \text{data})$$

Likelihood functions

We may create a function to calculate a likelihood:

```
lik.fun <- function(parameter){  
  ll <- dbinom(x=3, size=10, prob=parameter)  
  return(ll)  
}  
  
lik.fun(0.3)  
#> [1] 0.2668279  
  
lik.fun(0.6)  
#> [1] 0.04246733
```

Maximize the likelihood (3 successes out of 10 attempts)



The *maximum* of the likelihood is at value 0.3

Maximum likelihood estimation

- There is always a set of parameters that gives you the highest likelihood of observing the data, and this is the MLE.

Maximum likelihood estimation

- There is always a set of parameters that gives you the highest likelihood of observing the data, and this is the MLE.
- These can be calculated using:
 - Trial and error (not efficient!).
 - Compute the maximum of a function by hand (rarely doable in practice).
 - An iterative optimization algorithm: `?optim` in R.

By hand: compute MLE of p from $Y \sim \text{Bin}(N = 10, p)$ with $k = 3$ successes

- $P(Y = k) = \binom{k}{N} p^k (1 - p)^{N-k} = L(p).$

By hand: compute MLE of p from $Y \sim \text{Bin}(N = 10, p)$ with $k = 3$ successes

- $P(Y = k) = \binom{k}{N} p^k (1 - p)^{N-k} = L(p).$
- $\log(L(p)) = \text{cte} + k \log(p) + (N - k) \log(1 - p).$

By hand: compute MLE of p from $Y \sim \text{Bin}(N = 10, p)$ with $k = 3$ successes

- $P(Y = k) = \binom{k}{N} p^k (1 - p)^{N-k} = L(p).$
- $\log(L(p)) = \text{cte} + k \log(p) + (N - k) \log(1 - p).$
- We are searching for the maximum of L , or equivalently that of $\log(L)$.

By hand: compute MLE of p from $Y \sim \text{Bin}(N = 10, p)$ with $k = 3$ successes

- $P(Y = k) = \binom{k}{N} p^k (1 - p)^{N-k} = L(p).$
- $\log(L(p)) = \text{cte} + k \log(p) + (N - k) \log(1 - p).$
- We are searching for the maximum of L , or equivalently that of $\log(L)$.
- Compute derivate w.r.t. p : $\frac{d \log(L)}{dp} = \frac{k}{p} - \frac{(N - k)}{(1 - p)}.$

By hand: compute MLE of p from $Y \sim \text{Bin}(N = 10, p)$ with $k = 3$ successes

- $P(Y = k) = \binom{k}{N} p^k (1 - p)^{N-k} = L(p).$
- $\log(L(p)) = \text{cte} + k \log(p) + (N - k) \log(1 - p).$
- We are searching for the maximum of L , or equivalently that of $\log(L)$.
- Compute derivate w.r.t. p : $\frac{d \log(L)}{dp} = \frac{k}{p} - \frac{(N - k)}{(1 - p)}.$
- Then solve $\frac{d \log(L)}{dp} = 0$; the MLE is $\hat{p} = \frac{k}{N} = \frac{3}{10} = 0.3.$

By hand: compute MLE of p from $Y \sim \text{Bin}(N = 10, p)$ with $k = 3$ successes

- $P(Y = k) = \binom{k}{N} p^k (1 - p)^{N-k} = L(p).$
- $\log(L(p)) = \text{cte} + k \log(p) + (N - k) \log(1 - p).$
- We are searching for the maximum of L , or equivalently that of $\log(L)$.
- Compute derivate w.r.t. p : $\frac{d \log(L)}{dp} = \frac{k}{p} - \frac{(N - k)}{(1 - p)}.$
- Then solve $\frac{d \log(L)}{dp} = 0$; the MLE is $\hat{p} = \frac{k}{N} = \frac{3}{10} = 0.3.$
- Here, the MLE is the proportion of observed successes.

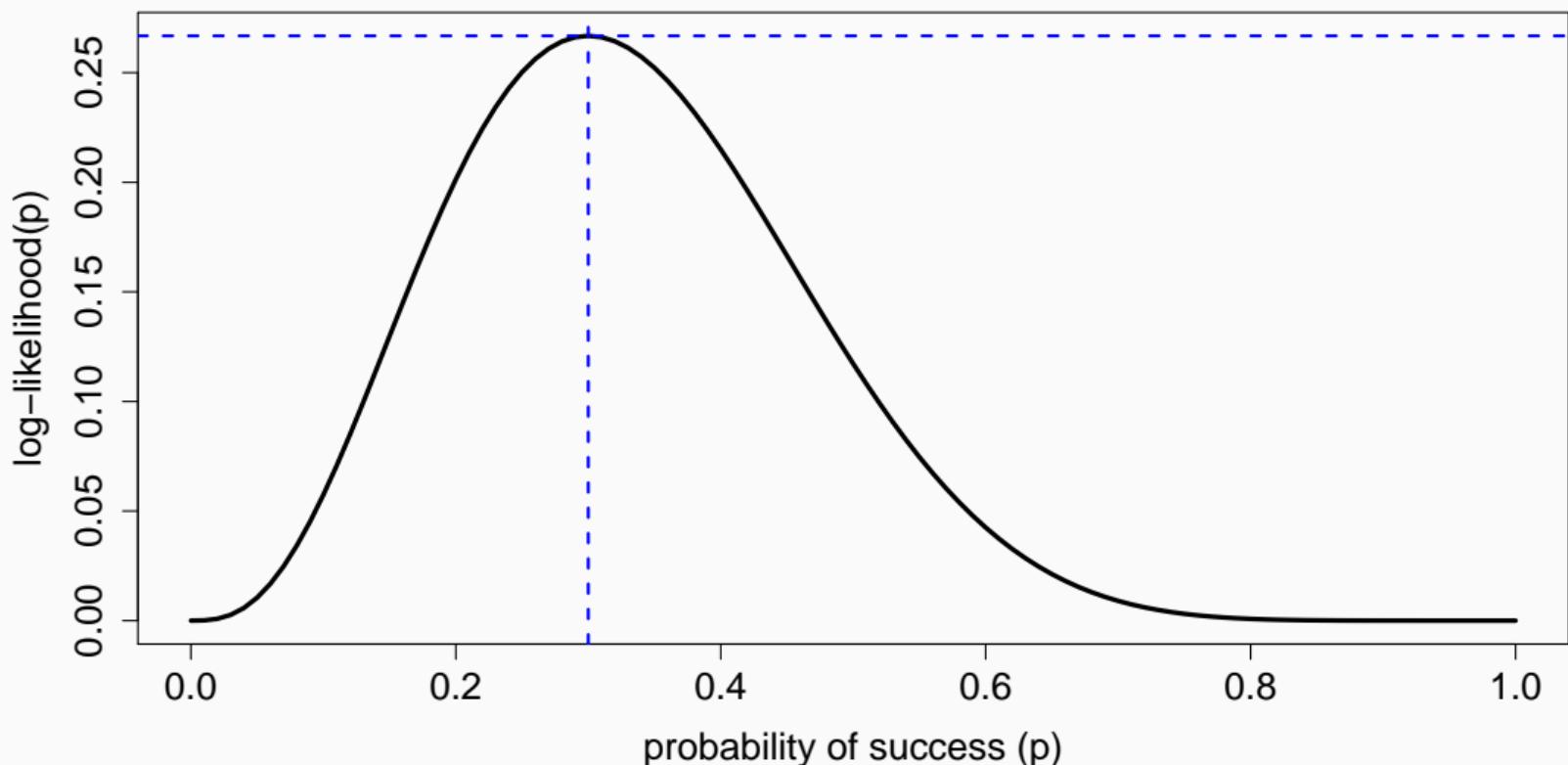
Using a computer: MLE of p from $Y \sim \text{Bin}(N = 10, p)$ with $k = 3$ successes

```
lik.fun <- function(parameter) dbinom(x=3, size=10, prob=parameter)
# ?optimize
optimize(lik.fun,c(0,1),maximum=TRUE)
#> $maximum
#> [1] 0.3000157
#>
#> $objective
#> [1] 0.2668279
```

Use optim when the number of parameters is > 1 .

Using a computer: MLE of p from $Y \sim \text{Bin}(N = 10, p)$ with $k = 3$ successes

Binomial likelihood with 3 successes out of 10 attempts

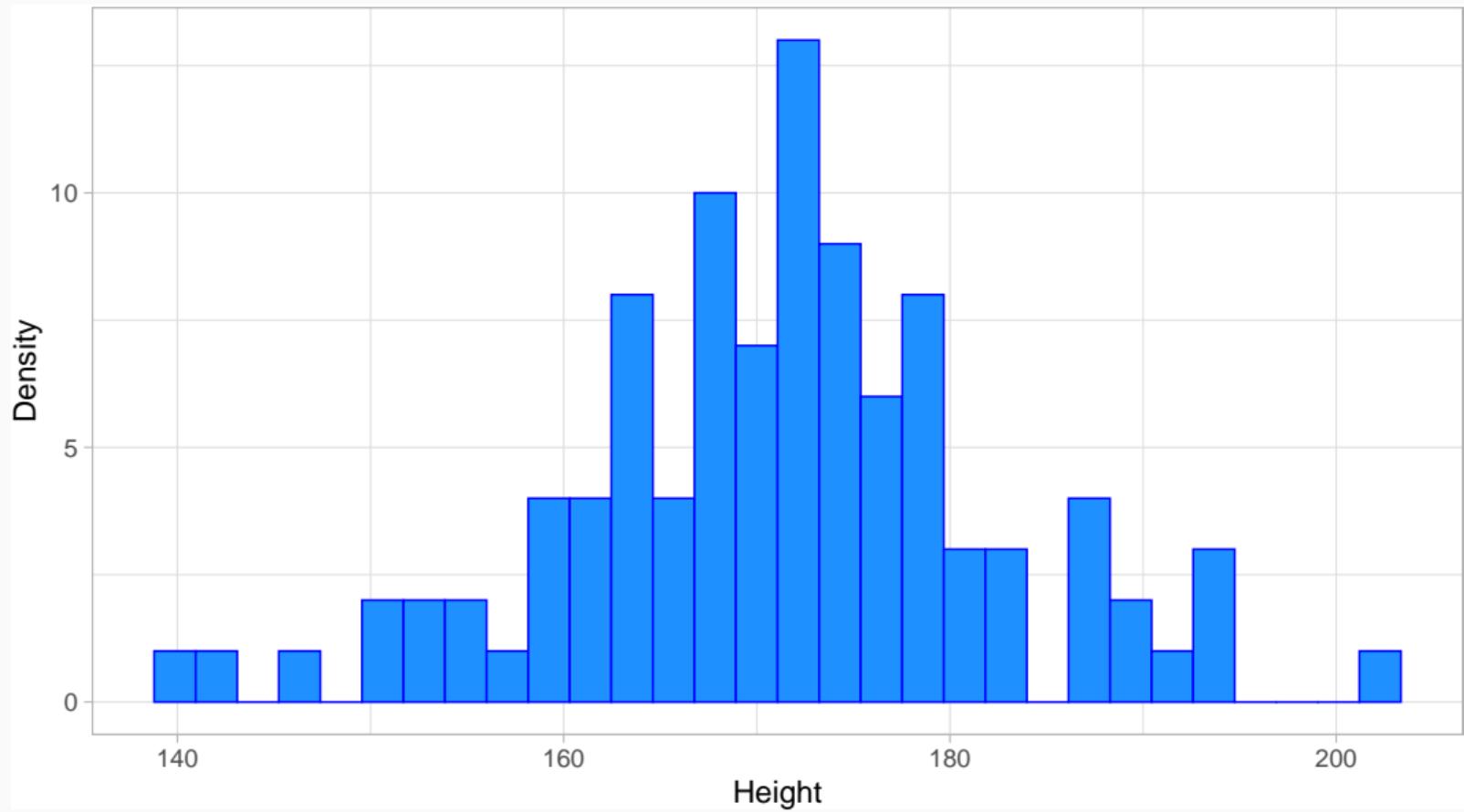


Your turn

MLE of the parameters of a Normal distribution

- Assume we have collected data on the height of 100 people:

```
# set seed for random numbers
set.seed(2020)
# simulate data from Normal distribution
n <- 100
height <- rnorm(n, mean=170, sd=10)
```



- We consider a Normal distribution for the model.
- Compute the MLE of the parameters of the Normal distribution.
- Hint: Use functions `optim()` and `dnorm()`

Solution

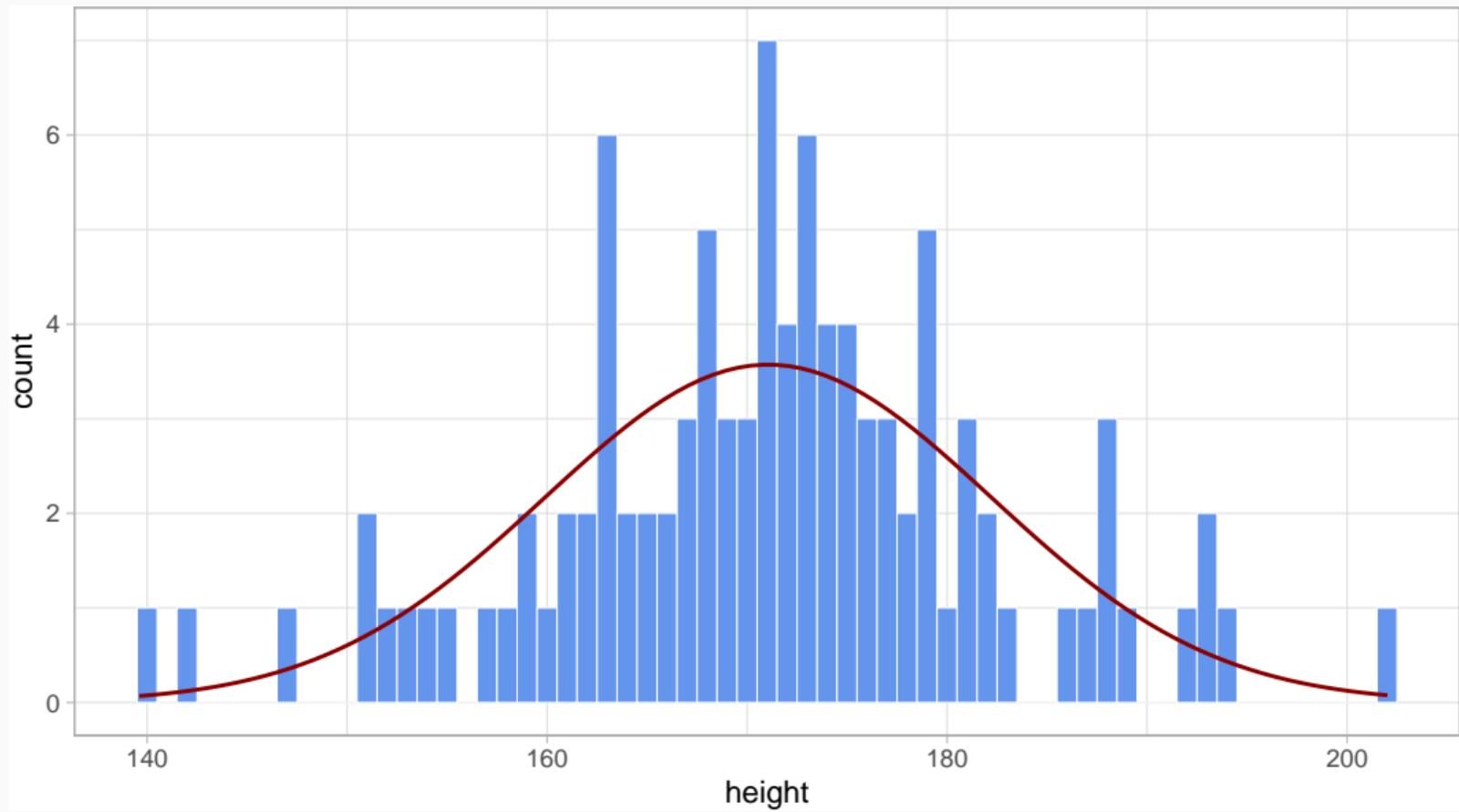
R code

- Write a function for the likelihood of a Normal distribution with parameters mean μ and standard deviation σ :

```
negloglik <- function(theta, data) {  
  mu <- theta[1]  
  sigma <- theta[2]  
  x <- data  
  -sum(dnorm(x, mean = mu, sd = sigma, log = TRUE))  
}  
negloglik(theta = c(150,1), height)  
#> [1] 28530.45
```

- Minimiiiiize

```
fit <- optim(par = c(1,1), fn = negloglik, data = height)
fit
#> $par
#> [1] 171.05358 11.16656
#>
#> $value
#> [1] 382.9207
#>
#> $counts
#> function gradient
#>      135       NA
#>
#> $convergence
#> [1] 0
#>
#> $message
#> NULL
```



Back to Bayes

A simple example

- Let us take a simple example to fix ideas.
- 120 deer were radio-tracked over winter.
- 61 close to a plant, 59 far from any human activity.
- Question: is there a treatment effect on survival?

	Released	Alive	Dead	Other
treatment	61	19	38	4
control	59	21	38	0

- So, $n = 57$ deer were assigned to the treatment group of which $k = 19$ survived the winter.

- So, $n = 57$ deer were assigned to the treatment group of which $k = 19$ survived the winter.
- Of interest is the probability of over-winter survival, call it θ , for the general population within the treatment area.

- So, $n = 57$ deer were assigned to the treatment group of which $k = 19$ survived the winter.
- Of interest is the probability of over-winter survival, call it θ , for the general population within the treatment area.
- The obvious estimate is simply to take the ratio $k/n = 19/57$.

- So, $n = 57$ deer were assigned to the treatment group of which $k = 19$ survived the winter.
- Of interest is the probability of over-winter survival, call it θ , for the general population within the treatment area.
- The obvious estimate is simply to take the ratio $k/n = 19/57$.
- How would the classical statistician justify this estimate?

- Our model is that we have a Binomial experiment (assuming independent and identically distributed draws from the population).

- Our model is that we have a Binomial experiment (assuming independent and identically distributed draws from the population).
- K the number of alive individuals at the end of the winter, so that $P(K = k) = \binom{n}{k} \theta^k (1 - \theta)^{n-k}$.

- Our model is that we have a Binomial experiment (assuming independent and identically distributed draws from the population).
- K the number of alive individuals at the end of the winter, so that $P(K = k) = \binom{n}{k} \theta^k (1 - \theta)^{n-k}$.
- The classical approach is to maximise the corresponding likelihood with respect to θ to obtain the entirely plausible MLE:

$$\hat{\theta} = k/n = 19/57$$

The Bayesian approach

- The Bayesian starts off with a prior.

The Bayesian approach

- The Bayesian starts off with a prior.
- Now, the one thing we know about θ is that it is a continuous random variable and that it lies between zero and one.

The Bayesian approach

- The Bayesian starts off with a prior.
- Now, the one thing we know about θ is that it is a continuous random variable and that it lies between zero and one.
- Thus, a suitable prior distribution might be the Beta defined on $[0, 1]$.

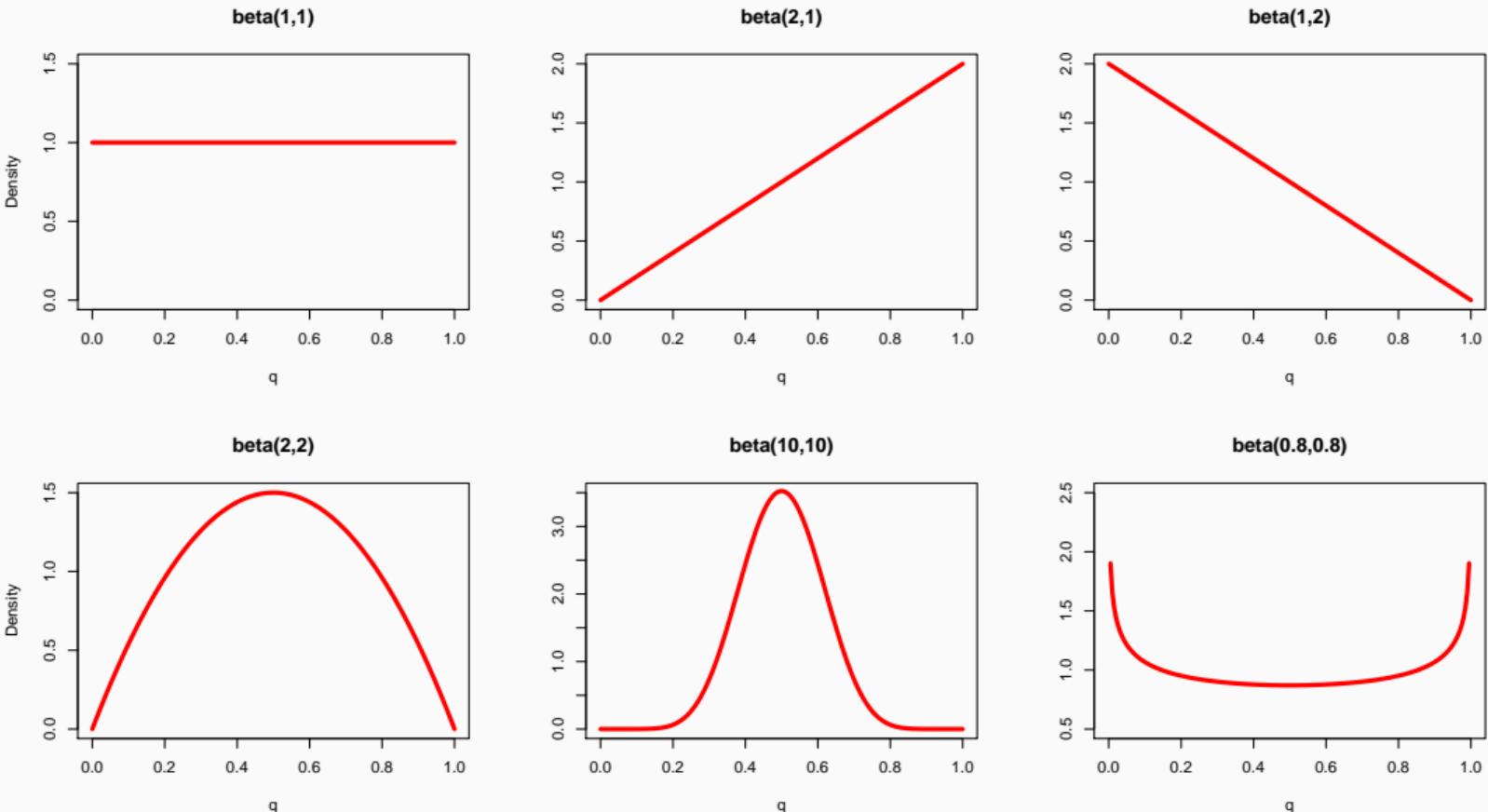
The Bayesian approach

- The Bayesian starts off with a prior.
- Now, the one thing we know about θ is that it is a continuous random variable and that it lies between zero and one.
- Thus, a suitable prior distribution might be the Beta defined on $[0, 1]$.
- What is the Beta distribution?

What is the Beta distribution?

$$q(\theta | \alpha, \beta) = \frac{1}{\text{Beta}(\alpha, \beta)} \theta^{\alpha-1} (1-\theta)^{\beta-1}$$

with $\text{Beta}(\alpha, \beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$ and $\Gamma(n) = (n-1)!$



The Bayesian approach

- We assume a priori that $\theta \sim Beta(a, b)$ so that $Pr(\theta) = \theta^{a-1}(1-\theta)^{b-1}$

The Bayesian approach

- We assume a priori that $\theta \sim Beta(a, b)$ so that $Pr(\theta) = \theta^{a-1}(1-\theta)^{b-1}$
- Then we have:

$$\begin{aligned} Pr(\theta | k) &\propto \binom{n}{k} \theta^k (1-\theta)^{n-k} \theta^{a-1} (1-\theta)^{b-1} \\ &\propto \theta^{(a+k)-1} (1-\theta)^{(b+n-k)-1} \end{aligned}$$

The Bayesian approach

- We assume a priori that $\theta \sim Beta(a, b)$ so that $Pr(\theta) = \theta^{a-1}(1-\theta)^{b-1}$
- Then we have:

$$\begin{aligned} Pr(\theta | k) &\propto \binom{n}{k} \theta^k (1-\theta)^{n-k} \theta^{a-1} (1-\theta)^{b-1} \\ &\propto \theta^{(a+k)-1} (1-\theta)^{(b+n-k)-1} \end{aligned}$$

- That is:

$$\theta | k \sim Beta(a+k, b+n-k)$$

The Bayesian approach

- We assume a priori that $\theta \sim Beta(a, b)$ so that $Pr(\theta) = \theta^{a-1}(1-\theta)^{b-1}$
- Then we have:

$$\begin{aligned} Pr(\theta | k) &\propto \binom{n}{k} \theta^k (1-\theta)^{n-k} \theta^{a-1} (1-\theta)^{b-1} \\ &\propto \theta^{(a+k)-1} (1-\theta)^{(b+n-k)-1} \end{aligned}$$

- That is:

$$\theta | k \sim Beta(a+k, b+n-k)$$

- Take a Beta prior with a Binomial likelihood, you get a Beta posterior (conjugacy)

Application to the deer example

- Posterior distribution of survival is $\theta \sim Beta(a + k, b + n - k)$.

Application to the deer example

- Posterior distribution of survival is $\theta \sim Beta(a + k, b + n - k)$.
- If we take a Uniform prior, i.e. $Beta(1, 1)$, then we have:

Application to the deer example

- Posterior distribution of survival is $\theta \sim Beta(a + k, b + n - k)$.
- If we take a Uniform prior, i.e. $Beta(1, 1)$, then we have:
- $\theta_{treatment} \sim Beta(1 + 19, 1 + 57 - 19) = Beta(20, 39)$

Application to the deer example

- Posterior distribution of survival is $\theta \sim Beta(a + k, b + n - k)$.
- If we take a Uniform prior, i.e. $Beta(1, 1)$, then we have:
- $\theta_{treatment} \sim Beta(1 + 19, 1 + 57 - 19) = Beta(20, 39)$
- Note that in this specific situation, the posterior has an explicit expression, easy to manipulate.

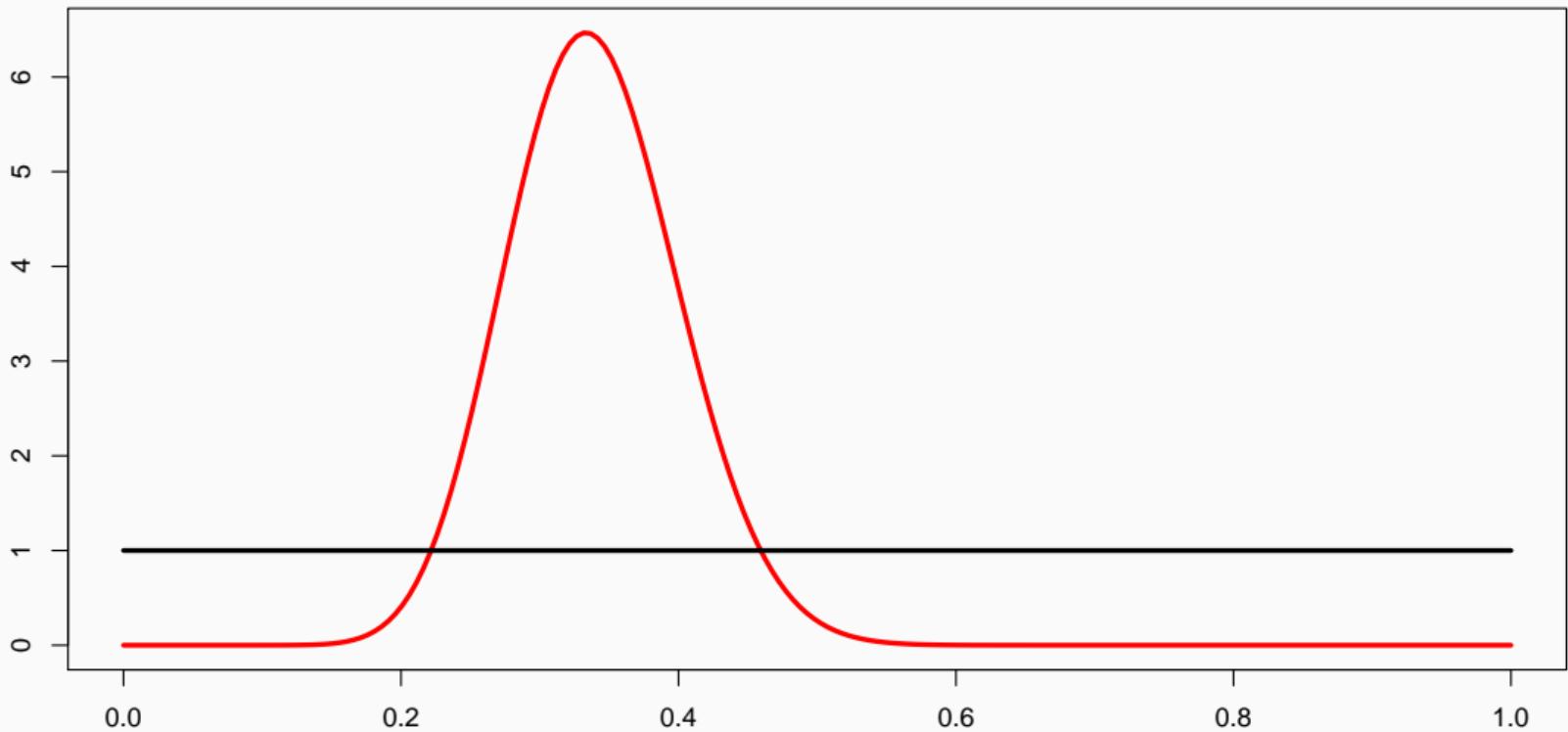
Application to the deer example

- Posterior distribution of survival is $\theta \sim Beta(a + k, b + n - k)$.
- If we take a Uniform prior, i.e. $Beta(1, 1)$, then we have:
- $\theta_{treatment} \sim Beta(1 + 19, 1 + 57 - 19) = Beta(20, 39)$
- Note that in this specific situation, the posterior has an explicit expression, easy to manipulate.
- In particular, $E(Beta(a, b)) = \frac{a}{a+b} = 20/59$ to be compared with the MLE $19/57$.

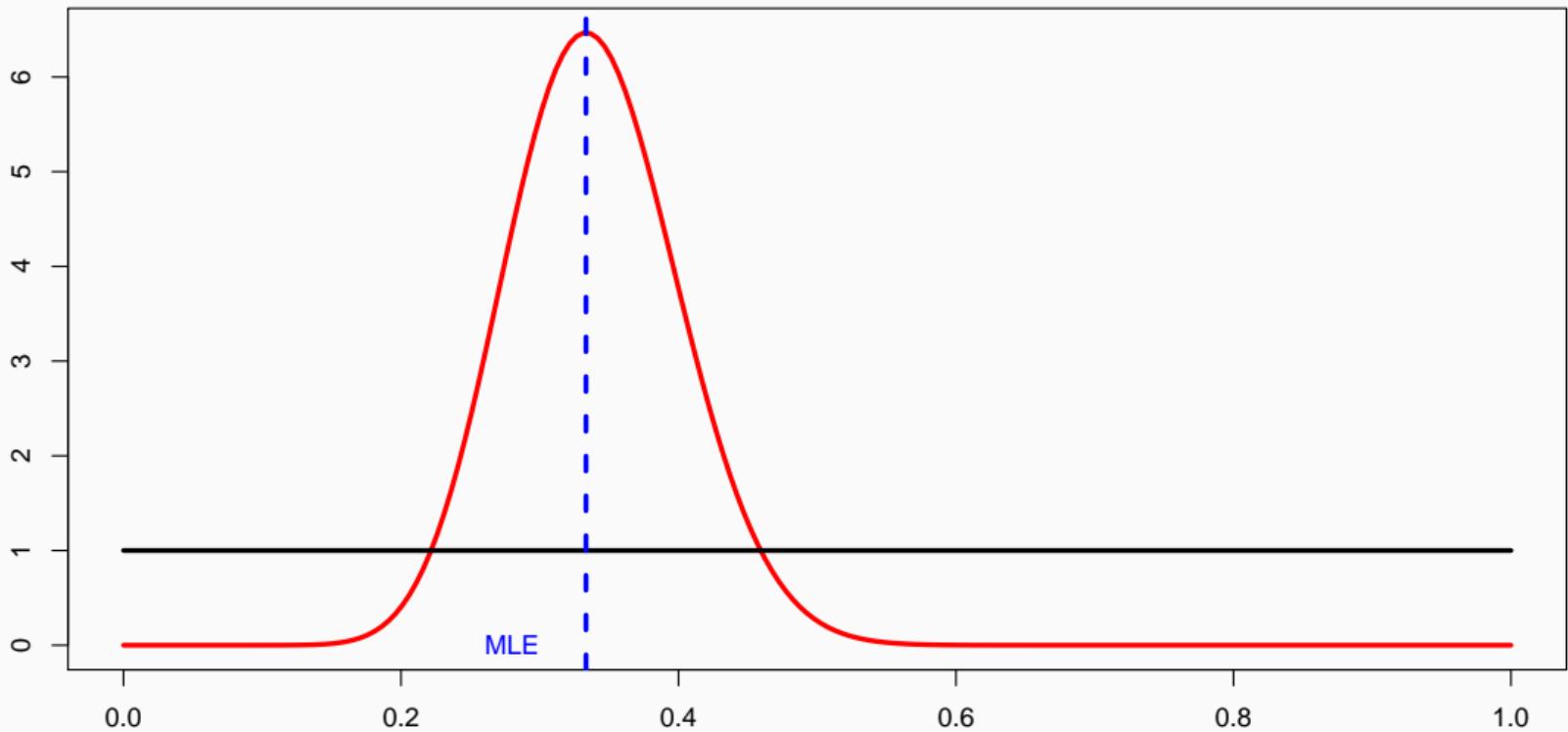
A general result

This is a general result, the Bayesian and frequentist estimates will always agree if there is sufficient data, so long as the likelihood is not explicitly ruled out by the prior.

Prior $\text{Beta}(1, 1)$ and posterior survival $\text{Beta}(20, 39)$



Prior $\text{Beta}(1, 1)$ and posterior survival $\text{Beta}(20, 39)$



Notation

Our model so far

$$y \sim \text{Binomial}(N, \theta)$$

[likelihood]

$$\theta \sim \text{Beta}(1, 1)$$

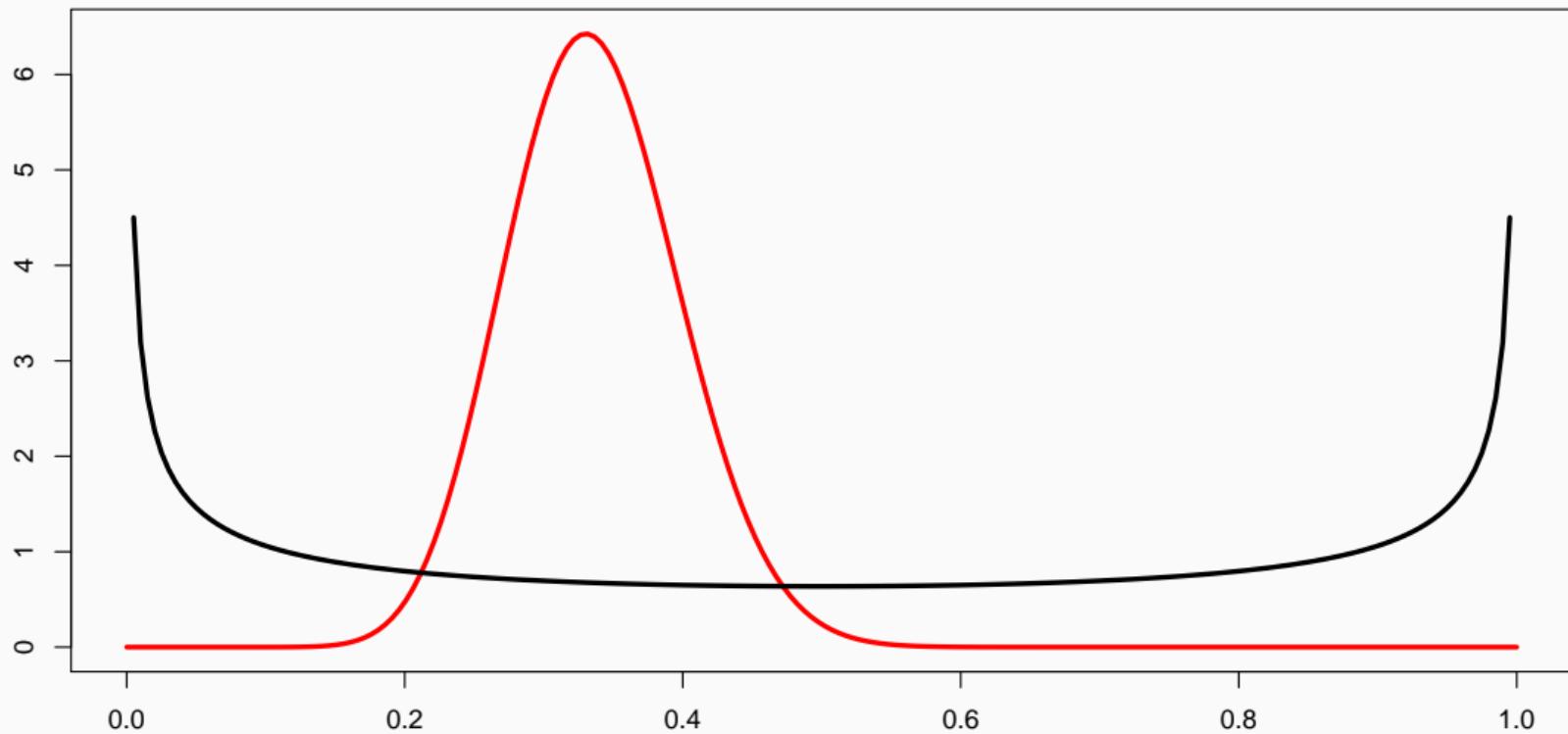
[prior for θ]


$$p(\theta | D) = p(D | \theta) \times p(\theta) / p(D)$$

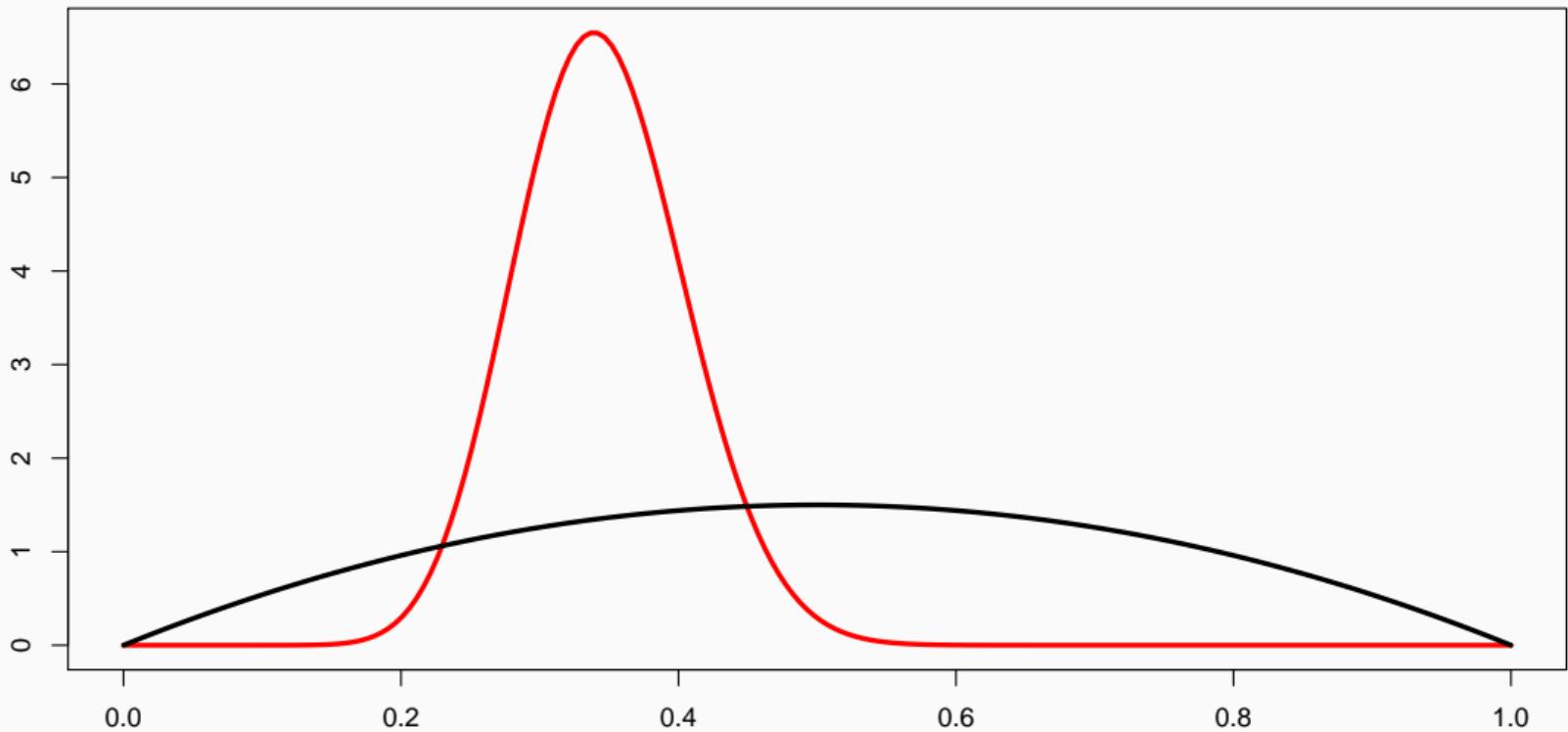
A detour to explore priors

Influence of the prior

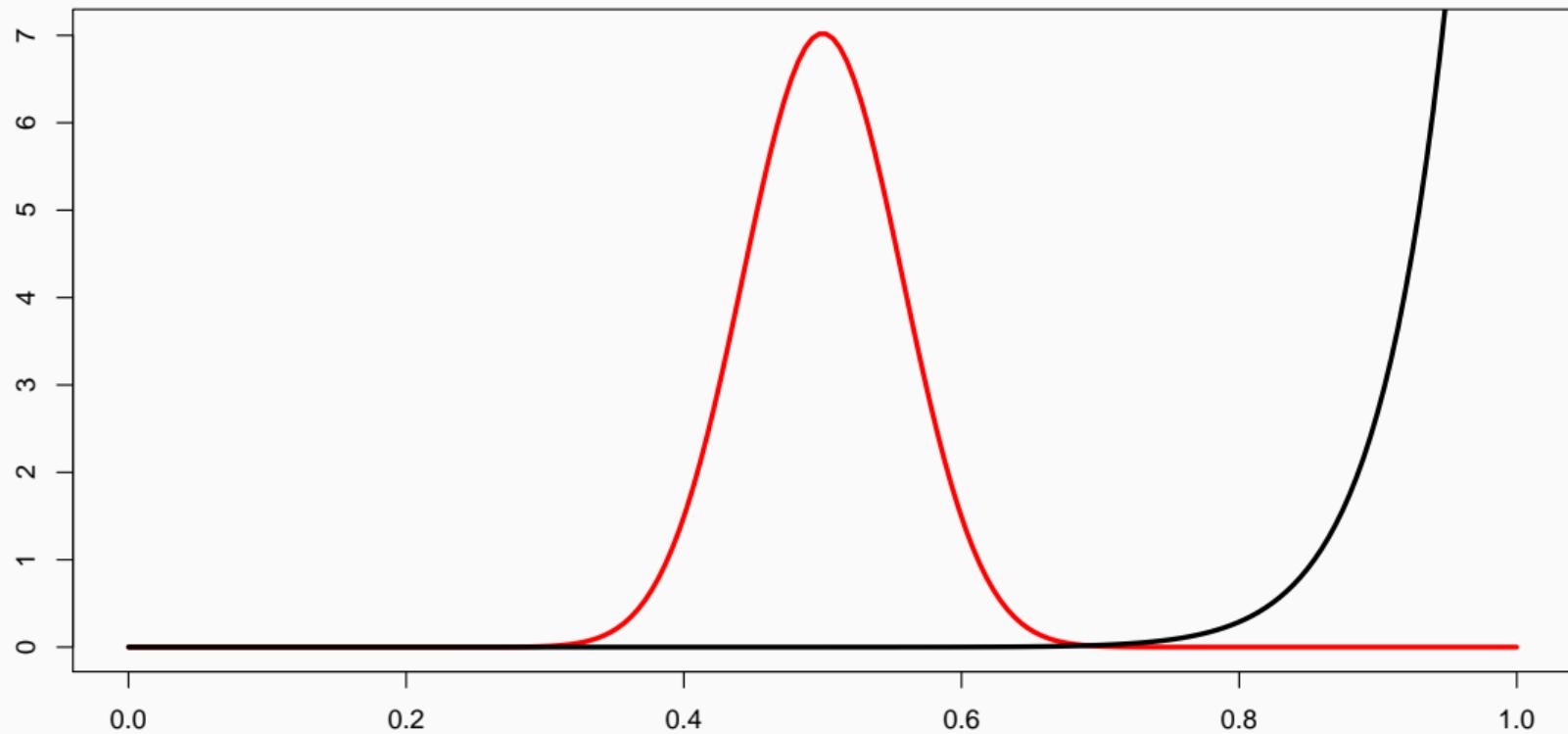
Prior $Beta(0.5, 0.5)$ and posterior survival $Beta(19.5, 38.5)$



Prior $\text{Beta}(2, 2)$ and posterior survival $\text{Beta}(21, 40)$



Prior $\text{Beta}(20, 1)$ and posterior survival $\text{Beta}(39, 49)$



The role of the prior

- In biological applications, the prior is a convenient means of incorporating expert opinion or information from previous or related studies that would otherwise need to be ignored. We'll get back to that.

The role of the prior

- In biological applications, the prior is a convenient means of incorporating expert opinion or information from previous or related studies that would otherwise need to be ignored. We'll get back to that.
- With sparse data, the role of the prior can be to enable inference on key parameters that would otherwise be impossible.

The role of the prior

- In biological applications, the prior is a convenient means of incorporating expert opinion or information from previous or related studies that would otherwise need to be ignored. We'll get back to that.
- With sparse data, the role of the prior can be to enable inference on key parameters that would otherwise be impossible.
- With sufficiently large and informative datasets the prior typically has little effect on the results.

The role of the prior

- In biological applications, the prior is a convenient means of incorporating expert opinion or information from previous or related studies that would otherwise need to be ignored. We'll get back to that.
- With sparse data, the role of the prior can be to enable inference on key parameters that would otherwise be impossible.
- With sufficiently large and informative datasets the prior typically has little effect on the results.
- Always perform a sensitivity analysis.

Informative priors vs. no information

- Informative priors aim to reflect information available to the analyst that is gained independently of the data being studied.

Informative priors vs. no information

- Informative priors aim to reflect information available to the analyst that is gained independently of the data being studied.
- In the absence of any prior information on one or more model parameters we wish to ensure that this lack of knowledge is properly reflected in the prior.

Informative priors vs. no information

- Informative priors aim to reflect information available to the analyst that is gained independently of the data being studied.
- In the absence of any prior information on one or more model parameters we wish to ensure that this lack of knowledge is properly reflected in the prior.
- Always perform a sensitivity analysis.

WE ARE ALL BAYESIANS....

Based on my priors
I will be just fine



How to incorporate prior information?

Estimating survival using capture-recapture data

- A bird might be captured, missed and recaptured; this is coded 101.

Estimating survival using capture-recapture data

- A bird might captured, missed and recaptured; this is coded 101.
- Simplest model relies on constant survival ϕ and detection p probabilities.

Estimating survival using capture-recapture data

- A bird might captured, missed and recaptured; this is coded 101.
- Simplest model relies on constant survival ϕ and detection p probabilities.
- Likelihood for that particular bird:

$$\Pr(101) = \phi(1 - p)\phi p$$

Estimating survival using capture-recapture data

- A bird might captured, missed and recaptured; this is coded 101.
- Simplest model relies on constant survival ϕ and detection p probabilities.
- Likelihood for that particular bird:

$$\Pr(101) = \phi(1 - p)\phi p$$

- We assume a vague prior:

$$\phi_{prior} \sim \text{Beta}(1, 1) = \text{Uniform}(0, 1)$$

Notation

- $y_{i,t} = 1$ if individual i detected at occasion t and 0 otherwise
- $z_{i,t} = 1$ if individual i alive between occasions t and $t + 1$ and 0 otherwise

$$y_{i,t} \mid z_{i,t} \sim \text{Bernoulli}(p \ z_{i,t}) \quad [\text{likelihood (observation eq.)}]$$

$$z_{i,t+1} \mid z_{i,t} \sim \text{Bernoulli}(\phi \ z_{i,t}) \quad [\text{likelihood (state eq.)}]$$

$$\phi \sim \text{Beta}(1, 1) \quad [\text{prior for } \phi]$$

$$p \sim \text{Beta}(1, 1) \quad [\text{prior for } p]$$

European dippers in Eastern France (1981-1987)



How to incorporate prior information?

- If no information, mean posterior survival is $\phi_{posterior} = 0.56$ with credible interval [0.51, 0.61].

How to incorporate prior information?

- If no information, mean posterior survival is $\phi_{posterior} = 0.56$ with credible interval [0.51, 0.61].
- Using information on body mass and annual survival of 27 European passersines, we can predict survival of European dippers using only body mass.

How to incorporate prior information?

- If no information, mean posterior survival is $\phi_{posterior} = 0.56$ with credible interval [0.51, 0.61].
- Using information on body mass and annual survival of 27 European passersines, we can predict survival of European dippers using only body mass.
- For dippers, body mass is 59.8g, therefore $\phi = 0.57$ with $sd = 0.073$.

How to incorporate prior information?

- If no information, mean posterior survival is $\phi_{posterior} = 0.56$ with credible interval [0.51, 0.61].
- Using information on body mass and annual survival of 27 European passersines, we can predict survival of European dippers using only body mass.
- For dippers, body mass is 59.8g, therefore $\phi = 0.57$ with $sd = 0.073$.
- Assuming an informative prior $\phi_{prior} \sim \text{Normal}(0.57, 0.073^2)$.

How to incorporate prior information?

- If no information, mean posterior survival is $\phi_{posterior} = 0.56$ with credible interval [0.51, 0.61].
- Using information on body mass and annual survival of 27 European passersines, we can predict survival of European dippers using only body mass.
- For dippers, body mass is 59.8g, therefore $\phi = 0.57$ with $sd = 0.073$.
- Assuming an informative prior $\phi_{prior} \sim \text{Normal}(0.57, 0.073^2)$.
- Mean posterior $\phi_{posterior} = 0.56$ with credible interval [0.52, 0.60].

How to incorporate prior information?

- If no information, mean posterior survival is $\phi_{posterior} = 0.56$ with credible interval [0.51, 0.61].
- Using information on body mass and annual survival of 27 European passersines, we can predict survival of European dippers using only body mass.
- For dippers, body mass is 59.8g, therefore $\phi = 0.57$ with $sd = 0.073$.
- Assuming an informative prior $\phi_{prior} \sim \text{Normal}(0.57, 0.073^2)$.
- Mean posterior $\phi_{posterior} = 0.56$ with credible interval [0.52, 0.60].
- No increase of precision in posterior inference.

How to incorporate prior information?

- Now if you had only the three first years of data, what would have happened?

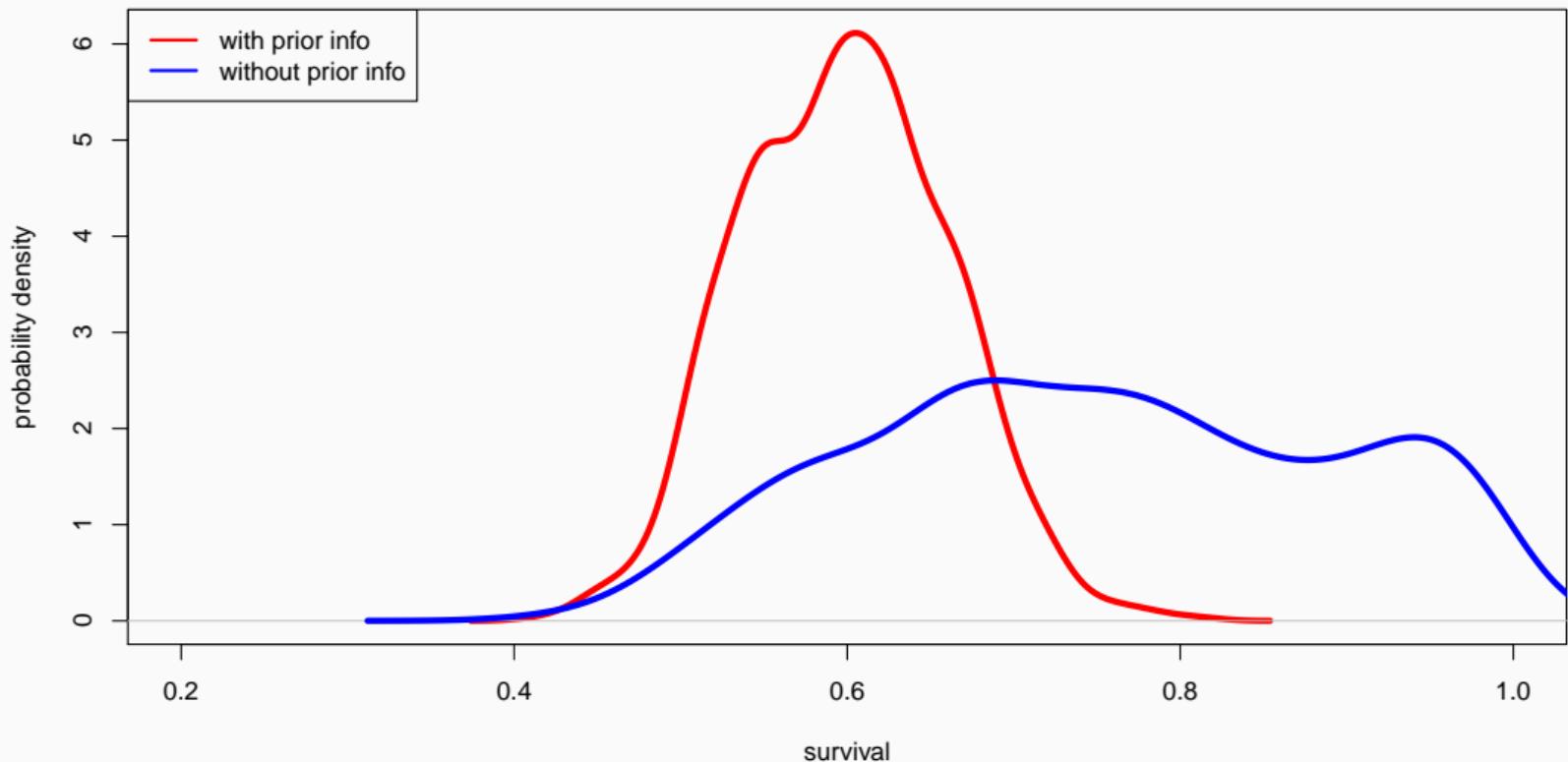
How to incorporate prior information?

- Now if you had only the three first years of data, what would have happened?
- Width of credible interval is 0.47 (vague prior) vs. 0.30 (informative prior).

How to incorporate prior information?

- Now if you had only the three first years of data, what would have happened?
- Width of credible interval is 0.47 (vague prior) vs. 0.30 (informative prior).
- Huge increase of precision in posterior inference (40% gain)!

Compare vague vs. informative prior



Prior elicitation via moment matching

Remember the Beta distribution

- Recall that the Beta distribution is a continuous distribution with values between 0 and 1. Useful for modelling survival or detection probabilities.

Remember the Beta distribution

- Recall that the Beta distribution is a continuous distribution with values between 0 and 1. Useful for modelling survival or detection probabilities.
- If $X \sim Beta(\alpha, \beta)$, then the first and second moments of X are:

$$\mu = E(X) = \frac{\alpha}{\alpha + \beta}$$

$$\sigma^2 = \text{Var}(X) = \frac{\alpha\beta}{(\alpha + \beta)^2(\alpha + \beta + 1)}$$

Moment matching

- In the capture-recapture example, we know a priori that the mean of the probability we're interested in is $\mu = 0.57$ and its variance is $\sigma^2 = 0.073^2$?

Moment matching

- In the capture-recapture example, we know a priori that the mean of the probability we're interested in is $\mu = 0.57$ and its variance is $\sigma^2 = 0.073^2$?
- Parameters μ and σ^2 are seen as the moments of a $Beta(\alpha, \beta)$ distribution.

Moment matching

- In the capture-recapture example, we know a priori that the mean of the probability we're interested in is $\mu = 0.57$ and its variance is $\sigma^2 = 0.073^2$?
- Parameters μ and σ^2 are seen as the moments of a $Beta(\alpha, \beta)$ distribution.
- Now we look for values of α and β that match the observed moments of the Beta distribution (μ and σ^2).

Moment matching

- In the capture-recapture example, we know a priori that the mean of the probability we're interested in is $\mu = 0.57$ and its variance is $\sigma^2 = 0.073^2$?
- Parameters μ and σ^2 are seen as the moments of a $Beta(\alpha, \beta)$ distribution.
- Now we look for values of α and β that match the observed moments of the Beta distribution (μ and σ^2).
- We need another set of equations:

$$\alpha = \left(\frac{1 - \mu}{\sigma^2} - \frac{1}{\mu} \right) \mu^2$$

$$\beta = \alpha \left(\frac{1}{\mu} - 1 \right)$$

- For our model, that means:

```
(alpha <- ( (1 - 0.57)/(0.073*0.073) - (1/0.57) )*0.57^2)
#> [1] 25.64636
(beta <- alpha * ( (1/0.57) - 1))
#> [1] 19.34726
```

- For our model, that means:

```
(alpha <- ( (1 - 0.57)/(0.073*0.073) - (1/0.57) )*0.57^2)  
#> [1] 25.64636  
(beta <- alpha * ( (1/0.57) - 1))  
#> [1] 19.34726
```

- Now use $\phi_{prior} \sim \text{Beta}(\alpha = 25.6, \beta = 19.3)$ instead of $\phi_{prior} \sim \text{Normal}(0.57, 0.073^2)$

Your turn

Question

Use simulations to check that our estimates are correct.

Solution

R code

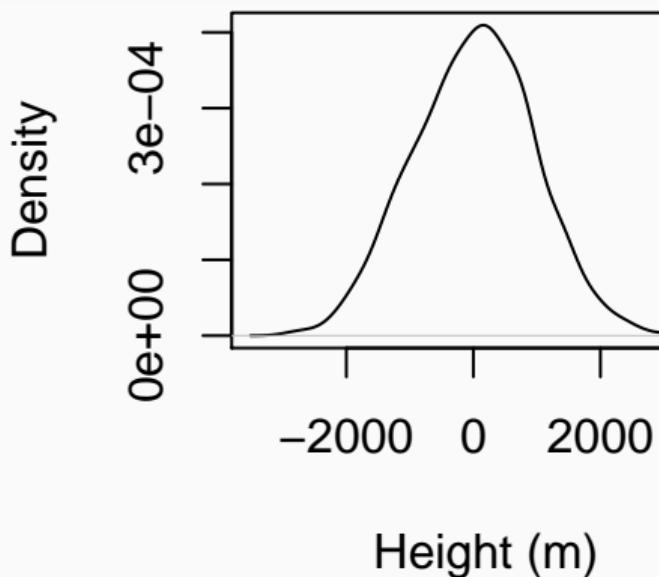
```
alpha <- ( (1 - 0.57)/(0.073*0.073) - (1/0.57) )*0.57^2  
beta <- alpha * ( (1/0.57) - 1)  
n <- 10000  
samp <- rbeta(n, alpha, beta)  
(mu <- mean(samp))  
(sigma <- sqrt(var(samp)))
```

Prior predictive checks

Linear regression

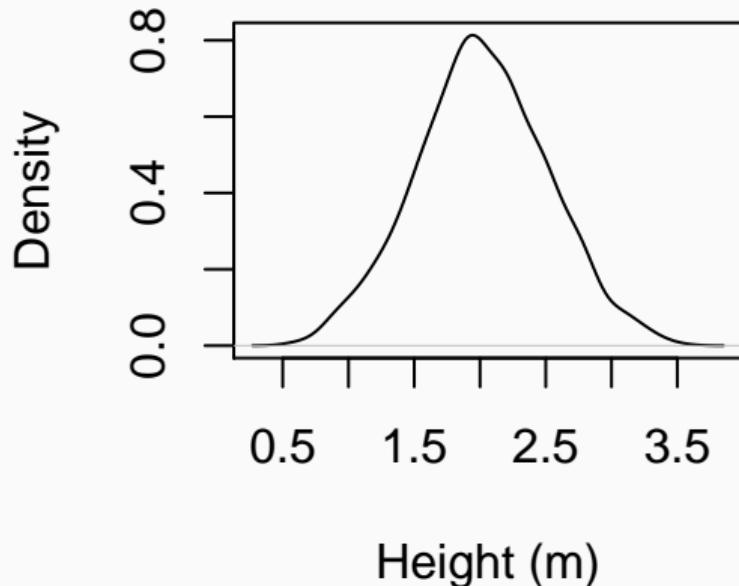
Unreasonable prior $\beta \sim N(0, 1000^2)$

```
plot(density(rnorm(1000, 0, 1000)),  
     main="", xlab="Height (m)")
```



Reasonable prior $\beta \sim N(2, 0.5^2)$

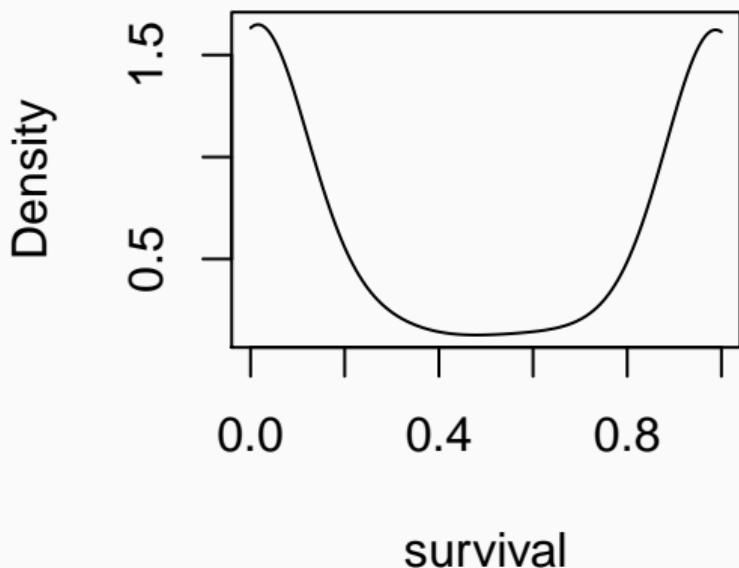
```
plot(density(rnorm(1000, 2, 0.5)),  
     main="", xlab="Height (m)")
```



Logistic regression

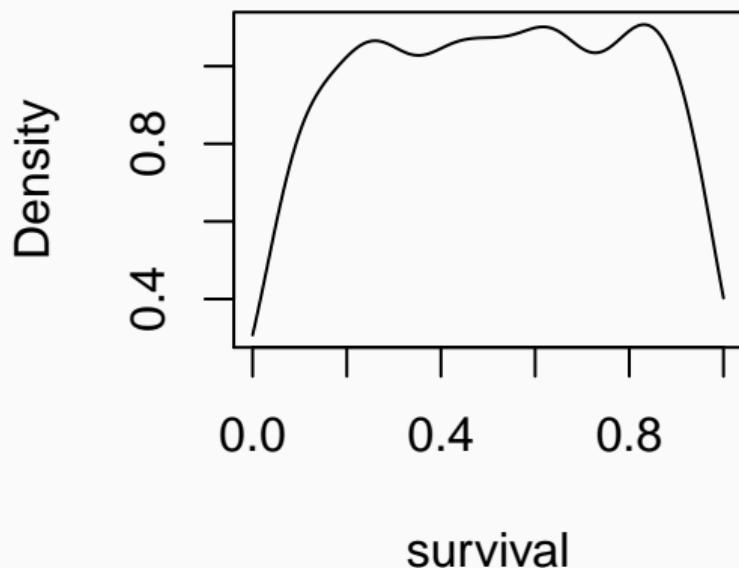
Unreasonable $\text{logit}(\phi) = \beta \sim N(0, 10^2)$

```
plot(density(plogis(rnorm(1000,0,10))),  
from = 0, to = 1), main='', xlab='survival')
```



Reasonable $\text{logit}(\phi) = \beta \sim N(0, 1.5^2)$

```
plot(density(plogis(rnorm(1000,0,1.5))),  
from = 0, to = 1), main='', xlab='survival')
```



Dynamic updating

Today's posterior is tomorrow's prior

If you obtain more data, no need to redo all of the analysis. Your posterior from the first analysis simply becomes your prior for the next analysis (and so on).

Today's posterior is tomorrow's prior

If you obtain more data, no need to redo all of the analysis. Your posterior from the first analysis simply becomes your prior for the next analysis (and so on).

- Stage 0. Prior $p(\theta) \sim \text{Beta}(1, 1)$.

Today's posterior is tomorrow's prior

If you obtain more data, no need to redo all of the analysis. Your posterior from the first analysis simply becomes your prior for the next analysis (and so on).

- Stage 0. Prior $p(\theta) \sim \text{Beta}(1, 1)$.
- Stage 1. Observe $y_1 = 22$ successes from $n_1 = 29$ trials.
 - Likelihood is $p(y_1|\theta) \sim \text{Binomial}(n_1 = 29, \theta)$.
 - Posterior is $p(\theta|y_1) \sim \text{Beta}(23, 8)$ with mean $23/31 = 0.74$.

Today's posterior is tomorrow's prior

If you obtain more data, no need to redo all of the analysis. Your posterior from the first analysis simply becomes your prior for the next analysis (and so on).

- Stage 0. Prior $p(\theta) \sim \text{Beta}(1, 1)$.
- Stage 1. Observe $y_1 = 22$ successes from $n_1 = 29$ trials.
 - Likelihood is $p(y_1|\theta) \sim \text{Binomial}(n_1 = 29, \theta)$.
 - Posterior is $p(\theta|y_1) \sim \text{Beta}(23, 8)$ with mean $23/31 = 0.74$.
- Stage 2. Observe $y_2 = 5$ successes from $n_2 = 10$ new trials.
 - Likelihood is $p(y_2|\theta) \sim \text{Binomial}(n_2 = 10, \theta)$.
 - Prior is $p(\theta) \sim \text{Beta}(23, 8)$ from stage 1.
 - Posterior is $p(\theta|y_1 \text{ and } y_2) \propto p(\theta|y_1)p(y_2|\theta) = \text{Beta}(28, 13)$ with mean $28/41 = 0.68$.

Get posteriors with Markov chains Monte Carlo (MCMC) methods

Back to the Bayes' theorem

- Bayes inference is easy! Well, not so easy in real-life applications.

Back to the Bayes' theorem

- Bayes inference is easy! Well, not so easy in real-life applications.
- The issue is in $\Pr(\theta | \text{data}) = \frac{\Pr(\text{data} | \theta) \Pr(\theta)}{\Pr(\text{data})}$

Back to the Bayes' theorem

- Bayes inference is easy! Well, not so easy in real-life applications.
- The issue is in $\Pr(\theta | \text{data}) = \frac{\Pr(\text{data} | \theta) \Pr(\theta)}{\Pr(\text{data})}$
- $\Pr(\text{data}) = \int L(\text{data} | \theta) \Pr(\theta) d\theta$ is a N -dimensional integral if $\theta = \theta_1, \dots, \theta_N$

Back to the Bayes' theorem

- Bayes inference is easy! Well, not so easy in real-life applications.
- The issue is in $\Pr(\theta | \text{data}) = \frac{\Pr(\text{data} | \theta) \Pr(\theta)}{\Pr(\text{data})}$
- $\Pr(\text{data}) = \int L(\text{data} | \theta) \Pr(\theta) d\theta$ is a N -dimensional integral if $\theta = \theta_1, \dots, \theta_N$
- Difficult, if not impossible to calculate!

Brute force approach via numerical integration

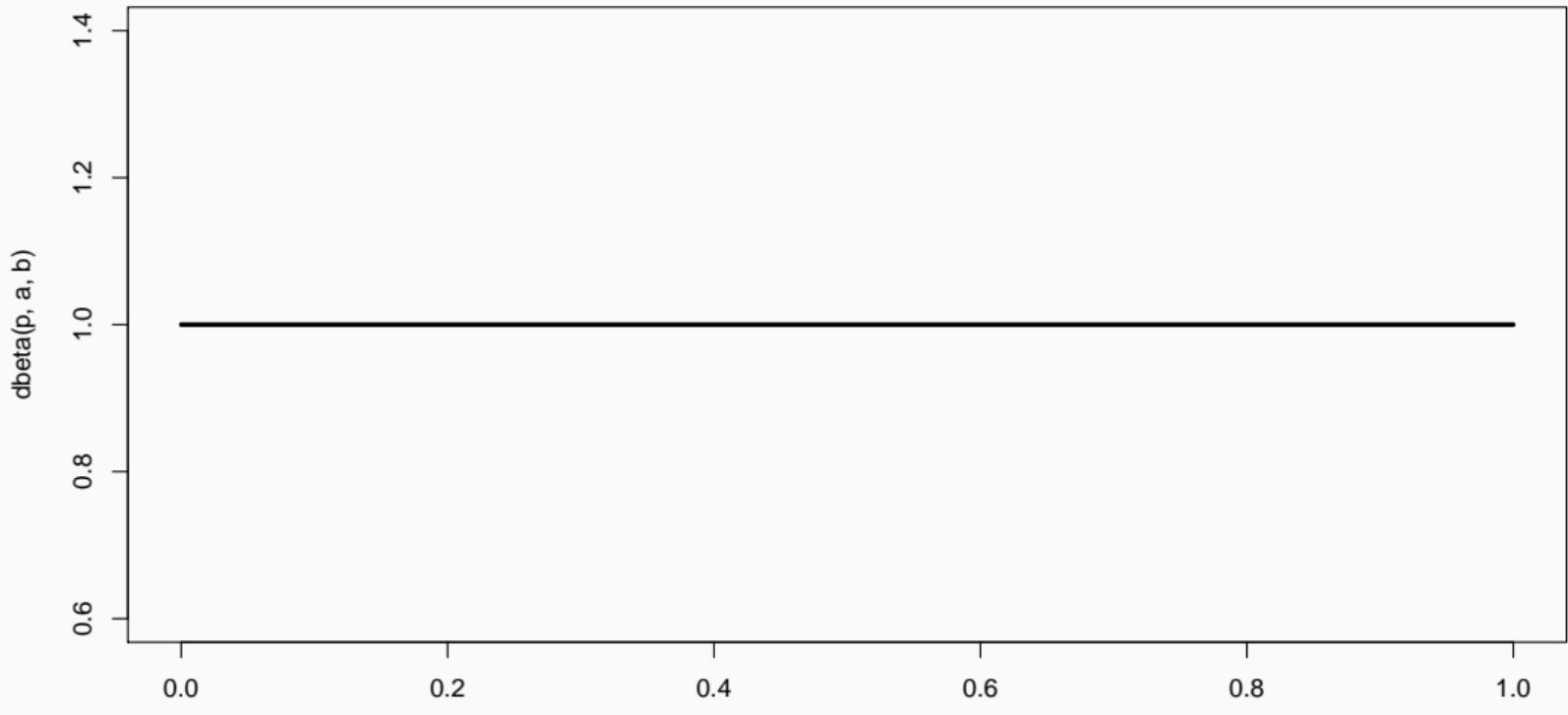
- Deer data

```
y <- 19 # nb of success  
n <- 57 # nb of attempts
```

- Likelihood $\text{Binomial}(57, \theta)$
- Prior $\text{Beta}(a = 1, b = 1)$

Beta prior

```
a <- 1; b <- 1; p <- seq(0,1,.002)
plot(p, dbeta(p,a,b), type='l', lwd=3)
```



Apply Bayes theorem

- Likelihood times the prior: $\Pr(\text{data} \mid \theta) \Pr(\theta)$

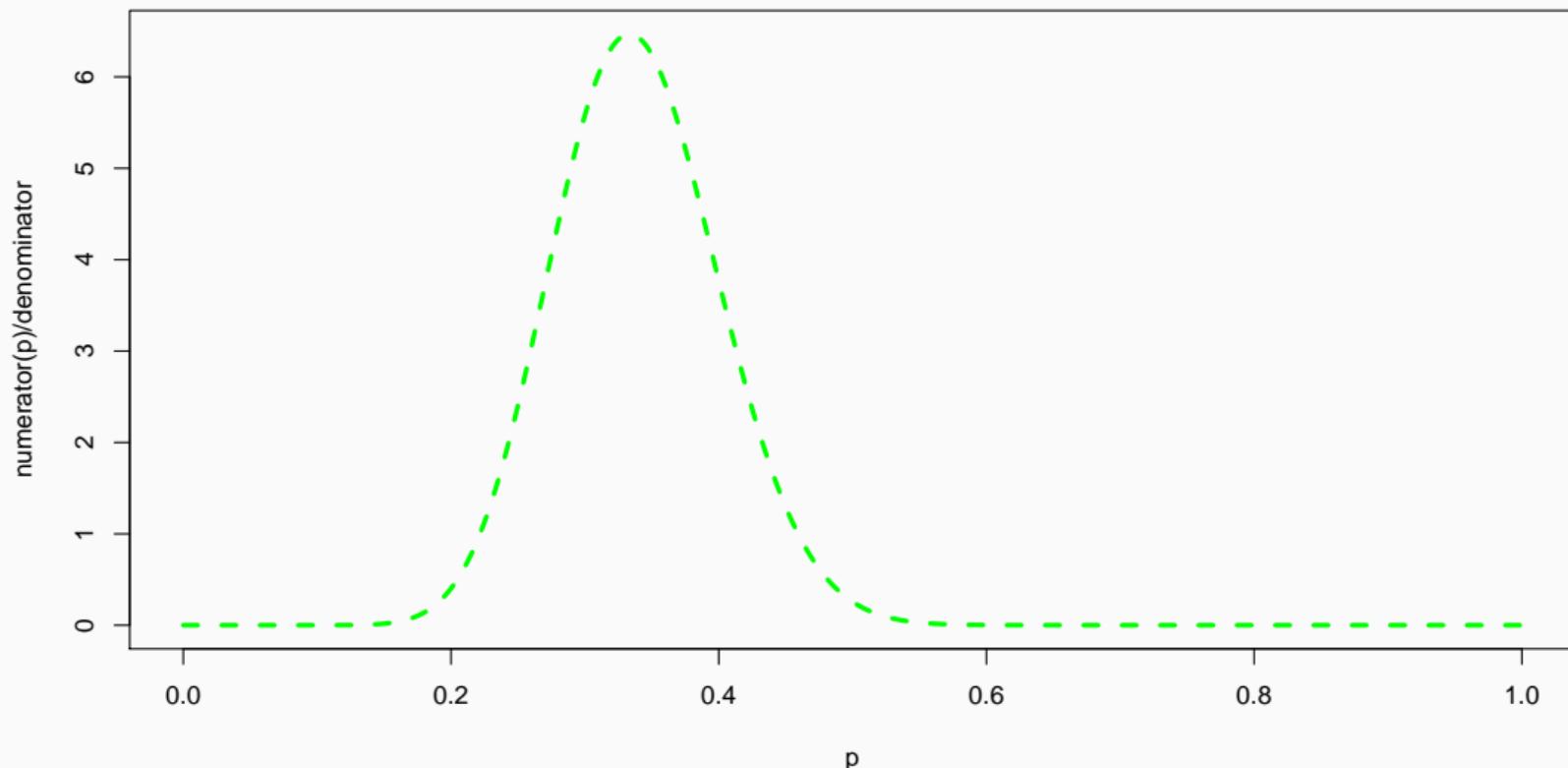
```
numerator <- function(p) dbinom(y,n,p)*dbeta(p,a,b)
```

- Averaged likelihood: $\Pr(\text{data}) = \int L(\theta \mid \text{data}) \Pr(\theta) d\theta$

```
denominator <- integrate(numerator,0,1)$value
```

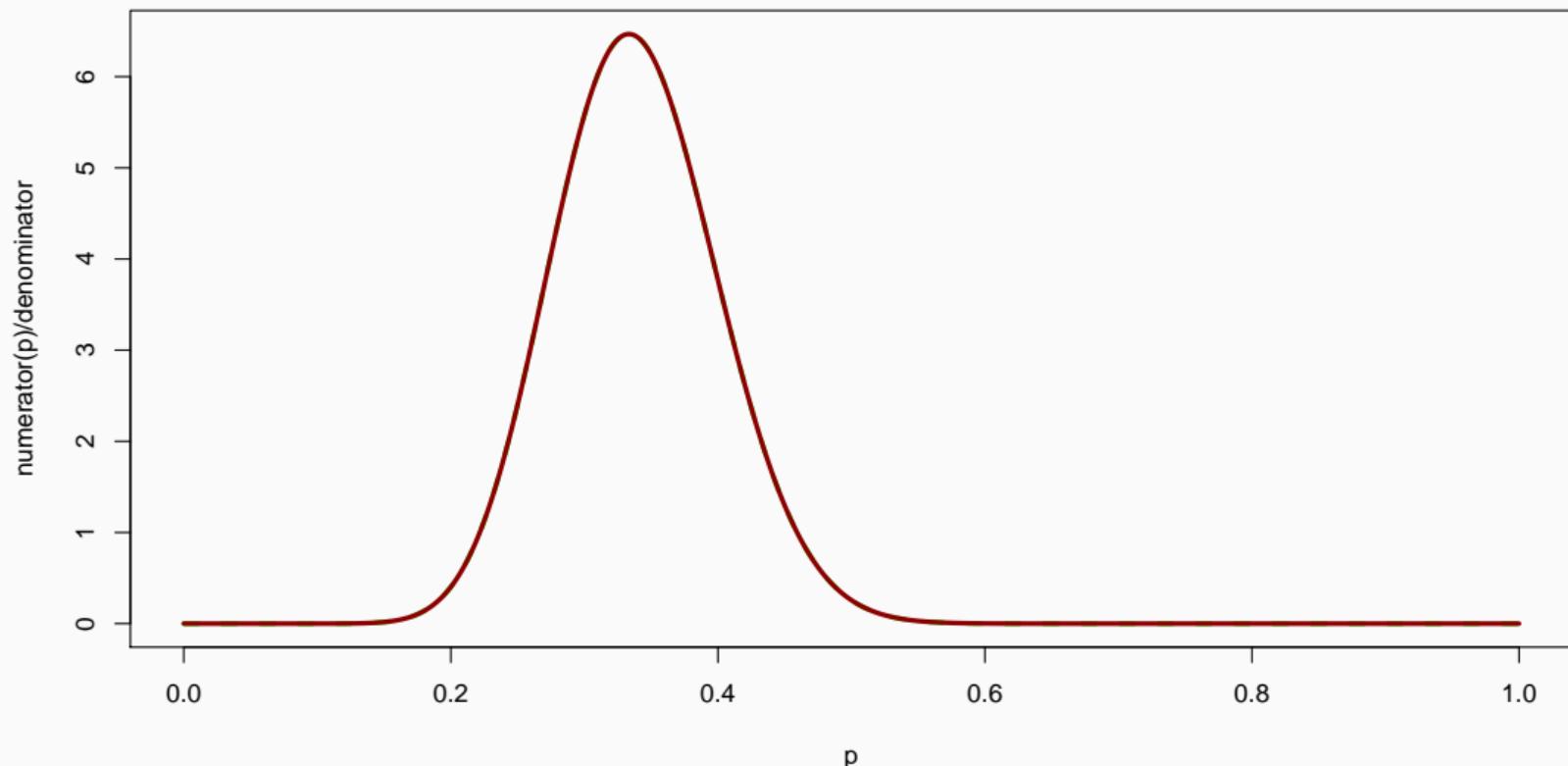
Posterior inference via numerical integration

```
plot(p, numerator(p)/denominator, type="l", lwd=3, col="green", lty=2)
```



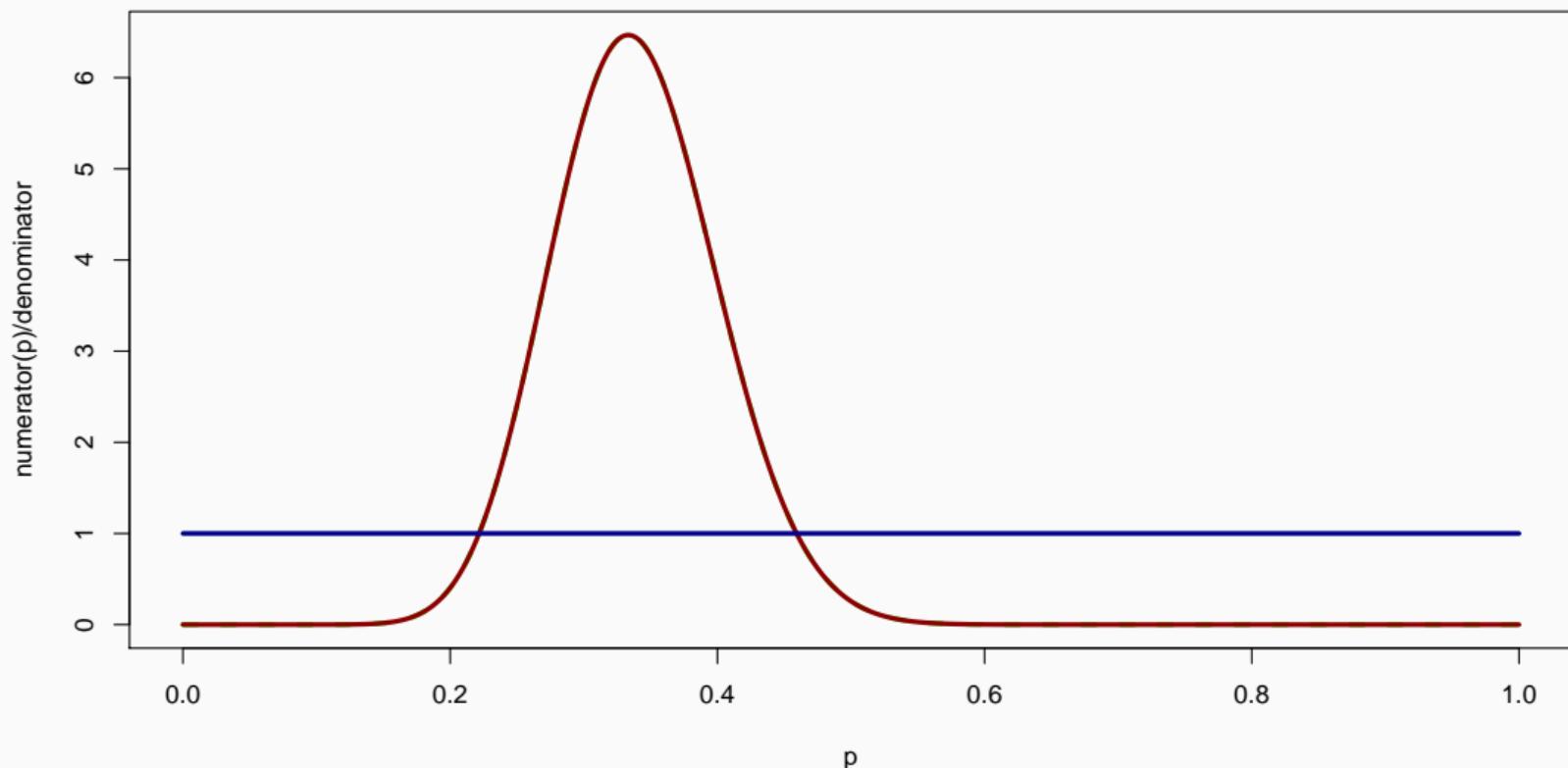
Superimpose explicit posterior distribution (Beta formula)

```
lines(p, dbeta(p,y+a,n-y+b), col='darkred', lwd=3)
```



And the prior

```
lines(p, dbeta(p,a,b), col='darkblue', lwd=3)
```



What if multiple parameters, like in a simple linear regression?

- Example of a linear regression with parameters α , β and σ to be estimated.

What if multiple parameters, like in a simple linear regression?

- Example of a linear regression with parameters α , β and σ to be estimated.
- Bayes' theorem says:

$$P(\alpha, \beta, \sigma \mid \text{data}) = \frac{P(\text{data} \mid \alpha, \beta, \sigma) P(\alpha, \beta, \sigma)}{\iiint P(\text{data} \mid \alpha, \beta, \sigma) P(\alpha, \beta, \sigma) d\alpha d\beta d\sigma}$$

What if multiple parameters, like in a simple linear regression?

- Example of a linear regression with parameters α , β and σ to be estimated.
- Bayes' theorem says:

$$P(\alpha, \beta, \sigma \mid \text{data}) = \frac{P(\text{data} \mid \alpha, \beta, \sigma) P(\alpha, \beta, \sigma)}{\iiint P(\text{data} \mid \alpha, \beta, \sigma) P(\alpha, \beta, \sigma) d\alpha d\beta d\sigma}$$

- Do we really wish to calculate a 3D integral?

Bayesian computation

- In the early 1990s, statisticians rediscovered work from the 1950's in physics.

THE JOURNAL OF CHEMICAL PHYSICS

VOLUME 21, NUMBER 6

JUNE, 1953

Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*
(Received March 6, 1953)

A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

Bayesian computation

- In the early 1990s, statisticians rediscovered work from the 1950's in physics.

THE JOURNAL OF CHEMICAL PHYSICS

VOLUME 21, NUMBER 6

JUNE, 1953

Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*
(Received March 6, 1953)

A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

- Use stochastic simulation to draw samples from posterior distributions.

Bayesian computation

- In the early 1990s, statisticians rediscovered work from the 1950's in physics.

THE JOURNAL OF CHEMICAL PHYSICS

VOLUME 21, NUMBER 6

JUNE, 1953

Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*

(Received March 6, 1953)

A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

- Use stochastic simulation to draw samples from posterior distributions.
- Avoid explicit calculation of integrals in Bayes formula.

Bayesian computation

- In the early 1990s, statisticians rediscovered work from the 1950's in physics.

THE JOURNAL OF CHEMICAL PHYSICS

VOLUME 21, NUMBER 6

JUNE, 1953

Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*

(Received March 6, 1953)

A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

- Use stochastic simulation to draw samples from posterior distributions.
- Avoid explicit calculation of integrals in Bayes formula.
- Instead, approximate posterior to arbitrary degree of precision by drawing large sample.

Bayesian computation

- In the early 1990s, statisticians rediscovered work from the 1950's in physics.

THE JOURNAL OF CHEMICAL PHYSICS

VOLUME 21, NUMBER 6

JUNE, 1953

Equation of State Calculations by Fast Computing Machines

NICHOLAS METROPOLIS, ARIANNA W. ROSENBLUTH, MARSHALL N. ROSENBLUTH, AND AUGUSTA H. TELLER,
Los Alamos Scientific Laboratory, Los Alamos, New Mexico

AND

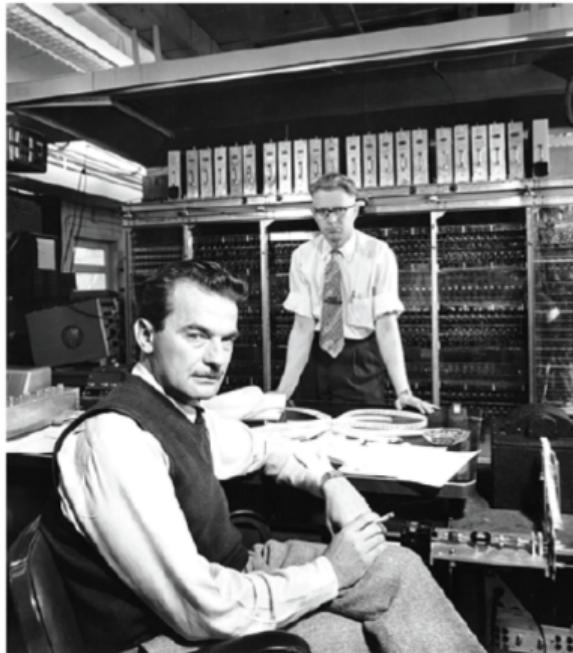
EDWARD TELLER,* *Department of Physics, University of Chicago, Chicago, Illinois*

(Received March 6, 1953)

A general method, suitable for fast computing machines, for investigating such properties as equations of state for substances consisting of interacting individual molecules is described. The method consists of a modified Monte Carlo integration over configuration space. Results for the two-dimensional rigid-sphere system have been obtained on the Los Alamos MANIAC and are presented here. These results are compared to the free volume equation of state and to a four-term virial coefficient expansion.

- Use stochastic simulation to draw samples from posterior distributions.
- Avoid explicit calculation of integrals in Bayes formula.
- Instead, approximate posterior to arbitrary degree of precision by drawing large sample.
- Markov chain Monte Carlo = MCMC; boost to Bayesian statistics!

MANIAC:
Mathematical Analyzer, Numerical Integrator, and Computer



MANIAC:
1000 pounds
5 kilobytes of memory
70k multiplications/sec

Your laptop:
4–7 pounds
2–8 million kilobytes
Billions of multiplications/sec

Why are MCMC methods so useful?

- MCMC: stochastic algorithm to produce sequence of dependent random numbers (from Markov chain).

Why are MCMC methods so useful?

- MCMC: stochastic algorithm to produce sequence of dependent random numbers (from Markov chain).
- Converge to equilibrium (aka stationary) distribution.

Why are MCMC methods so useful?

- MCMC: stochastic algorithm to produce sequence of dependent random numbers (from Markov chain).
- Converge to equilibrium (aka stationary) distribution.
- Equilibrium distribution is the desired posterior distribution!

Why are MCMC methods so useful?

- MCMC: stochastic algorithm to produce sequence of dependent random numbers (from Markov chain).
- Converge to equilibrium (aka stationary) distribution.
- Equilibrium distribution is the desired posterior distribution!
- Several ways of constructing these chains: e.g., Metropolis-Hastings, Gibbs sampler, Metropolis-within-Gibbs.

Why are MCMC methods so useful?

- MCMC: stochastic algorithm to produce sequence of dependent random numbers (from Markov chain).
- Converge to equilibrium (aka stationary) distribution.
- Equilibrium distribution is the desired posterior distribution!
- Several ways of constructing these chains: e.g., Metropolis-Hastings, Gibbs sampler, Metropolis-within-Gibbs.
- How to implement them in practice?!

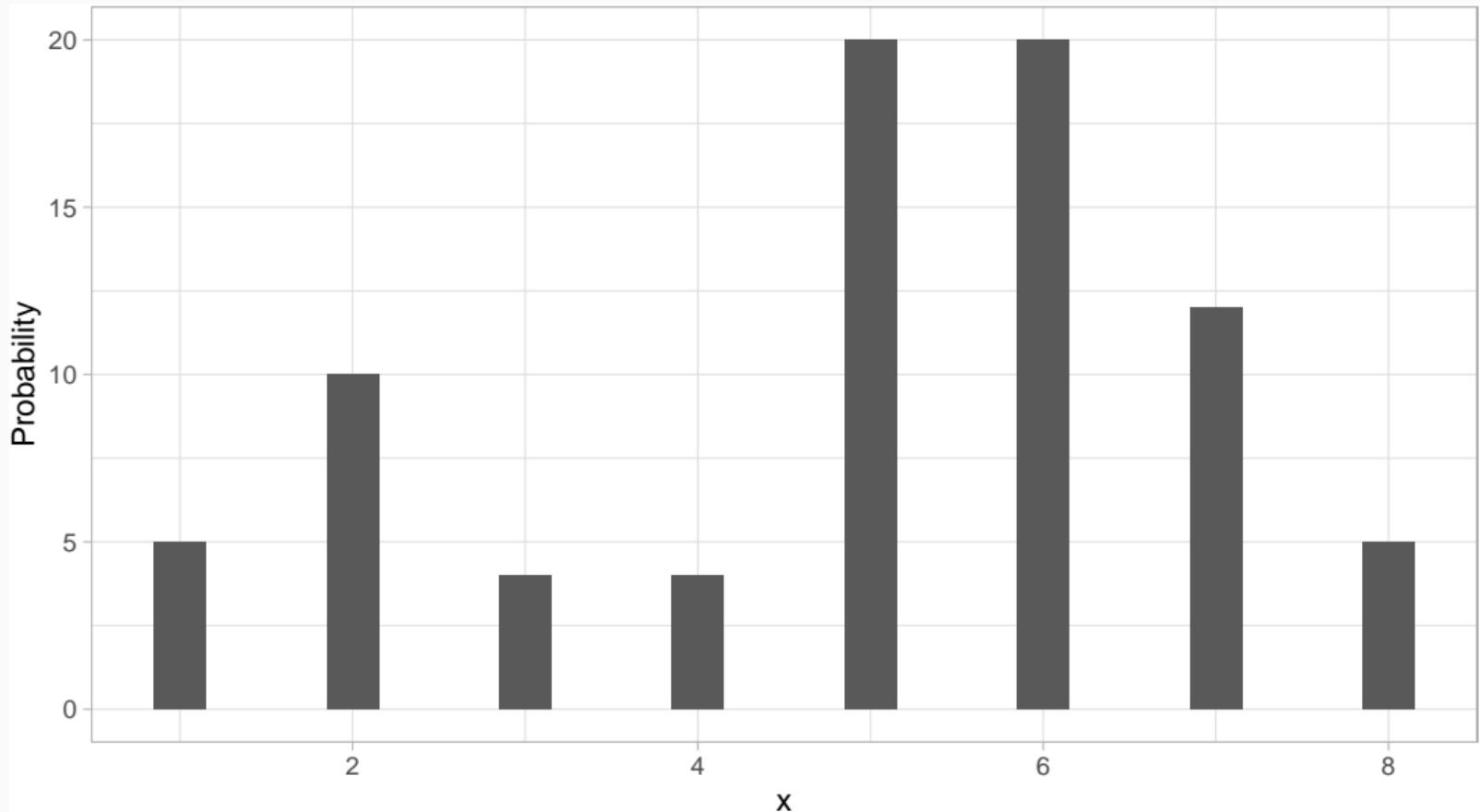
The Metropolis algorithm

- We illustrate sampling from a discrete distribution. Suppose we define a discrete probability distribution on the integers $1, \dots, K$.

The Metropolis algorithm

- We illustrate sampling from a discrete distribution. Suppose we define a discrete probability distribution on the integers $1, \dots, K$.
- We write a short function `pd()` in R taking on the values $1, \dots, 8$ with probabilities proportional to the values 5, 10, 4, 4, 20, 20, 12, and 5.

```
pd <- function(x){  
  values <- c(5, 10, 4, 4, 20, 20, 12, 5)  
  ifelse(x %in% 1:length(values), values[x], 0)  
}  
prob_dist <- data.frame(x = 1:8, prob = pd(1:8))  
prob_dist  
#>   x prob  
#> 1 1    5  
#> 2 2   10  
#> 3 3    4  
#> 4 4    4  
#> 5 5   20  
#> 6 6   20  
#> 7 7   12  
#> 8 8    5
```



To simulate from this probability distribution, we take a **random walk** described as follows.

To simulate from this probability distribution, we take a **random walk** described as follows.

1. We start at any possible location of our random variable from 1 to $K = 8$

To simulate from this probability distribution, we take a **random walk** described as follows.

1. We start at any possible location of our random variable from 1 to $K = 8$
2. To decide where to visit next, a fair coin is flipped. If the coin lands heads, we think about visiting the location one value to the left, and if coin lands tails, we consider visiting the location one value to right. We call this location the **candidate** location.

To simulate from this probability distribution, we take a **random walk** described as follows.

1. We start at any possible location of our random variable from 1 to $K = 8$
2. To decide where to visit next, a fair coin is flipped. If the coin lands heads, we think about visiting the location one value to the left, and if coin lands tails, we consider visiting the location one value to right. We call this location the **candidate** location.
3. We compute the ratio of the probabilities at the candidate and current locations
$$R = pd(candidate)/pd(current)$$

To simulate from this probability distribution, we take a **random walk** described as follows.

1. We start at any possible location of our random variable from 1 to $K = 8$
2. To decide where to visit next, a fair coin is flipped. If the coin lands heads, we think about visiting the location one value to the left, and if coin lands tails, we consider visiting the location one value to right. We call this location the **candidate** location.
3. We compute the ratio of the probabilities at the candidate and current locations
 $R = pd(candidate)/pd(current)$
4. We spin a continuous spinner that lands anywhere from 0 to 1 – call the random spin X . If X is smaller than R , we move to the candidate location, and otherwise we remain at the current location.

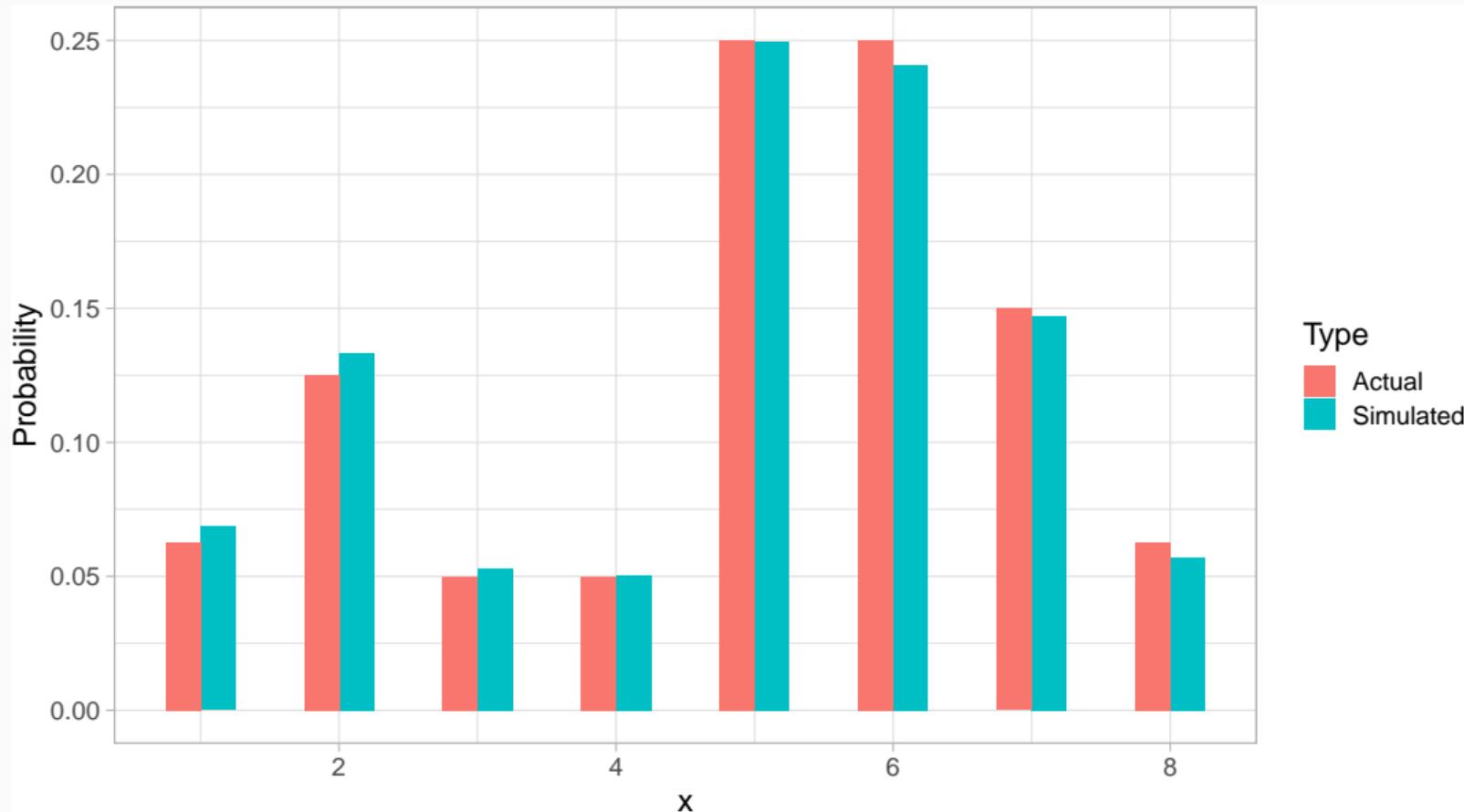
To simulate from this probability distribution, we take a **random walk** described as follows.

1. We start at any possible location of our random variable from 1 to $K = 8$
2. To decide where to visit next, a fair coin is flipped. If the coin lands heads, we think about visiting the location one value to the left, and if coin lands tails, we consider visiting the location one value to right. We call this location the **candidate** location.
3. We compute the ratio of the probabilities at the candidate and current locations
 $R = pd(candidate)/pd(current)$
4. We spin a continuous spinner that lands anywhere from 0 to 1 – call the random spin X . If X is smaller than R , we move to the candidate location, and otherwise we remain at the current location.
5. We repeat 2-4 a number of times called **steps** (many steps).

```
random_walk <- function(pd, start, num_steps){  
  y <- rep(0, num_steps)  
  current <- start  
  for (j in 1:num_steps){  
    candidate <- current + sample(c(-1, 1), 1)  
    prob <- pd(candidate) / pd(current)  
    if (runif(1) < prob) current <- candidate  
    y[j] <- current  
  }  
  return(y)  
}
```

Starting at the value $X = 4$ and running the algorithm for $s = 10,000$ iterations.

```
out <- random_walk(pd, 4, 10000)
head(out)
#> [1] 3 2 2 2 3 4
tail(out)
#> [1] 2 2 3 4 3 2
```



Animating the Metropolis algorithm - 2D example

<https://mbjoseph.github.io/posts/2018-12-25-animating-the-metropolis-algorithm/>

The Markov-chain Monte Carlo Interactive Gallery

<https://chi-feng.github.io/mcmc-demo/>

Bayes in practice

Software implementation (R compatible)

Oldies but goodies:

- WinBUGS, OpenBUGS: Where it all began.
- Jags: What we will use in this course.

Software implementation (R compatible)

Oldies but goodies:

- WinBUGS, OpenBUGS: Where it all began.
- Jags: What we will use in this course.

The new kids on the block:

- Nimble: What I'm going for these days.
- Stan: Entirely different algorithmic approach.
- Greta: Dunno anything about it.

Introduction to JAGS (Just Another Gibbs Sampler)

Martyn Plummer



Real example

Impact of climatic conditions on white stork breeding success



Let's do a logistic regression on some White stork data

- Assess effects of temperature and rainfall on productivity.

Let's do a logistic regression on some White stork data

- Assess effects of temperature and rainfall on productivity.
- We have collected data.

Let's do a logistic regression on some White stork data

- Assess effects of temperature and rainfall on productivity.
- We have collected data.
- We need to build a model - write down the likelihood.

Let's do a logistic regression on some White stork data

- Assess effects of temperature and rainfall on productivity.
- We have collected data.
- We need to build a model - write down the likelihood.
- We need to specify priors for parameters.

Read in the data

```
nbchicks <- c(151,105,73,107,113,87,77,108,118,122,112,120,122,89,69,71,  
53,41,53,31,35,14,18)  
  
nbpairs <- c(173,164,103,113,122,112,98,121,132,136,133,137,145,117,90,80,  
67,54,58,39,42,23,23)  
  
temp <- c(15.1,13.3,15.3,13.3,14.6,15.6,13.1,13.1,15.0,11.7,15.3,14.4,14.4,  
12.7,11.7,11.9,15.9,13.4,14.0,13.9,12.9,15.1,13.0)  
  
rain <- c(67,52,88,61,32,36,72,43,92,32,86,28,57,55,66,26,28,96,48,90,86,  
78,87)  
  
datax <- list(N = 23, nbchicks = nbchicks, npairs = npairs,  
temp = (temp - mean(temp))/sd(temp),  
rain = (rain - mean(rain))/sd(rain))
```

Write down the model

$$\text{nbchicks}_i \sim \text{Binomial}(\text{nbpairs}_i, p_i) \quad [\text{likelihood}]$$

$$\text{logit}(p_i) = a + b_{\text{temp}} \text{ temp}_i + b_{\text{rain}} \text{ rain}_i \quad [\text{linear model}]$$

$$a \sim \text{Normal}(0, 1000) \quad [\text{prior for } a]$$

$$b_{\text{temp}} \sim \text{Normal}(0, 1000) \quad [\text{prior for } b_{\text{temp}}]$$

$$b_{\text{rain}} \sim \text{Normal}(0, 1000) \quad [\text{prior for } b_{\text{rain}}]$$

Build the model

```
{  
# Likelihood  
for( i in 1 : N){  
    nbchicks[i] ~ dbin(p[i],nbpairs[i])  
    logit(p[i]) <- a + b.temp * temp[i] + b.rain * rain[i]  
}  
# ...
```

Specify priors

```
# Priors  
a ~ dnorm(0,0.001)  
b.temp ~ dnorm(0,0.001)  
b.rain ~ dnorm(0,0.001)  
}
```

Warning: Jags uses precision for Normal distributions (1 / variance)

You need to write everything in a file

```
model <-
paste(
model
{
  for( i in 1 : N)
  {
    nbchicks[i] ~ dbin(p[i],nbpairs[i])
    logit(p[i]) <- a + b.temp * temp[i] + b.rain * rain[i]
  }
a ~ dnorm(0,0.001)
b.temp ~ dnorm(0,0.001)
b.rain ~ dnorm(0,0.001)
}
")
writeLines(model,"code/logistic.txt")
```

Alternatively, you may write a R function

```
logistic <- function() {  
  for( i in 1 : N)  
  {  
    nbchicks[i] ~ dbin(p[i],nbpairs[i])  
    logit(p[i]) <- a + b.temp * temp[i] + b.rain * rain[i]  
  }  
  
  # priors for regression parameters  
  a ~ dnorm(0,0.001)  
  b.temp ~ dnorm(0,0.001)  
  b.rain ~ dnorm(0,0.001)  
}
```

Let us specify a few additional things

```
# list of lists of initial values (one for each MCMC chain)
init1 <- list(a = -0.5, b.temp = -0.5, b.rain = -0.5)
init2 <- list(a = 0.5, b.temp = 0.5, b.rain = 0.5)
inits <- list(init1,init2)

# specify parameters that need to be estimated
parameters <- c("a","b.temp","b.rain")

# specify nb iterations for burn-in and final inference
nb.burnin <- 1000
nb.iterations <- 2000
```

Run Jags

```
# load R2jags
library(R2jags)
# run Jags
storks <- jags(data = datax,
                 inits = inits,
                 parameters.to.save = parameters,
                 model.file = "code/logistic.txt",
                 # model.file = logistic, # if a function was written
                 n.chains = 2,
                 n.iter = nb.iterations,
                 n.burnin = nb.burnin)

storks
```

Inspect parameter estimates

```
#> Compiling model graph
#> Resolving undeclared variables
#> Allocating nodes
#> Graph information:
#>   Observed stochastic nodes: 23
#>   Unobserved stochastic nodes: 3
#>   Total graph size: 181
#>
#> Initializing model
#> Inference for Bugs model at "code/logistic.txt", fit using jags,
#> 2 chains, each with 2000 iterations (first 1000 discarded)
#> n.sims = 2000 iterations saved
#>      mu.vect sd.vect    2.5%     25%     50%     75%   97.5%   Rhat n.eff
#> a        1.550   0.084   1.430   1.519   1.554   1.589   1.667 1.183   170
#> b.rain   -0.161   0.063  -0.285  -0.201  -0.162  -0.120  -0.047 1.002   1300
#> b.temp    0.026   0.061  -0.087  -0.016   0.026   0.064   0.151 1.001   2000
#> deviance 206.339 30.006 201.776 202.735 203.849 205.676 212.060 1.078   2000
```

Your turn

Practical

- Run the stork analysis yourself.
- Does it seem like there is an effect of rainfall or temperature on breeding success?

Assess convergence

Reminder – MCMC Algorithm

- MCMC algorithms can be used to construct a Markov chain with a given stationary distribution (set to be the posterior distribution).

Reminder – MCMC Algorithm

- MCMC algorithms can be used to construct a Markov chain with a given stationary distribution (set to be the posterior distribution).
- For the MCMC algorithm, the posterior distribution is only needed to be known up to proportionality.

Reminder – MCMC Algorithm

- MCMC algorithms can be used to construct a Markov chain with a given stationary distribution (set to be the posterior distribution).
- For the MCMC algorithm, the posterior distribution is only needed to be known up to proportionality.
- Once the stationary distribution is reached we can regard the realisations of the chain as a (dependent) sample from the posterior distribution (and obtain Monte Carlo estimates).

Reminder – MCMC Algorithm

- MCMC algorithms can be used to construct a Markov chain with a given stationary distribution (set to be the posterior distribution).
- For the MCMC algorithm, the posterior distribution is only needed to be known up to proportionality.
- Once the stationary distribution is reached we can regard the realisations of the chain as a (dependent) sample from the posterior distribution (and obtain Monte Carlo estimates).
- We consider some important implementation issues.

MCMC – Proposal Distribution

- To implement a MCMC algorithm, we often need to specify a proposal distribution from which we generate candidate value then accept/reject.

MCMC – Proposal Distribution

- To implement a MCMC algorithm, we often need to specify a proposal distribution from which we generate candidate value then accept/reject.
- This typically involves
 - specifying a given distribution family (e.g. normal, uniform), and then,
 - setting the parameters of the given distribution.

MCMC – Proposal Distribution

- To implement a MCMC algorithm, we often need to specify a proposal distribution from which we generate candidate value then accept/reject.
- This typically involves
 - specifying a given distribution family (e.g. normal, uniform), and then,
 - setting the parameters of the given distribution.
- Although the exact distribution specified is essentially arbitrary – it will have a significant effect on the performance of the MCMC algorithm.

Why is the proposal distribution so important?

- If only small moves can be proposed, the acceptance probability is high, but it will take a long time to explore the posterior distribution.

Why is the proposal distribution so important?

- If only small moves can be proposed, the acceptance probability is high, but it will take a long time to explore the posterior distribution.
- Proposing large jumps has the potential to move further, but generally have smaller acceptance probabilities.

Why is the proposal distribution so important?

- If only small moves can be proposed, the acceptance probability is high, but it will take a long time to explore the posterior distribution.
- Proposing large jumps has the potential to move further, but generally have smaller acceptance probabilities.
- In order to balance the size of the proposed moves with the chance of accepting them the proposal variance is often tuned to obtain a mean acceptance probability of 20 – 40%.

Why is the proposal distribution so important?

- If only small moves can be proposed, the acceptance probability is high, but it will take a long time to explore the posterior distribution.
- Proposing large jumps has the potential to move further, but generally have smaller acceptance probabilities.
- In order to balance the size of the proposed moves with the chance of accepting them the proposal variance is often tuned to obtain a mean acceptance probability of 20 – 40%.
- Automatic in Jags – ouf!

Why is the proposal distribution so important?

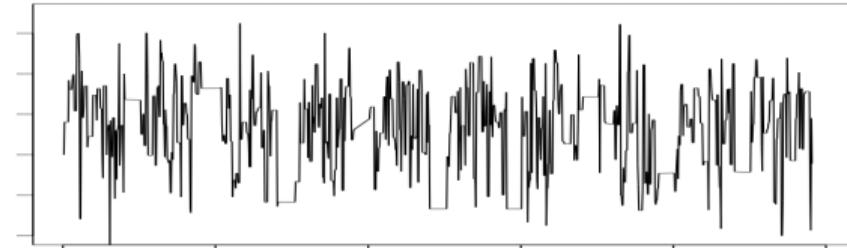
- If only small moves can be proposed, the acceptance probability is high, but it will take a long time to explore the posterior distribution.
- Proposing large jumps has the potential to move further, but generally have smaller acceptance probabilities.
- In order to balance the size of the proposed moves with the chance of accepting them the proposal variance is often tuned to obtain a mean acceptance probability of 20 – 40%.
- Automatic in Jags – ouf!
- The movement around the parameter space is often referred to as **mixing**.

Good/Bad Traces

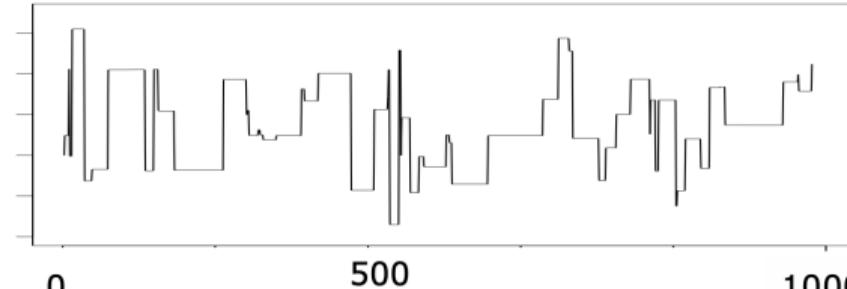
*Small
moves -
bad*



good



*Large
moves -
bad*



Autocorrelation functions

- Traceplots of small and big moves provide (relatively) high correlations (known as autocorrelations) between successive observations of the Markov chain.

Autocorrelation functions

- Traceplots of small and big moves provide (relatively) high correlations (known as autocorrelations) between successive observations of the Markov chain.
- Strongly correlated observations require large sample sizes and therefore longer simulations.

Autocorrelation functions

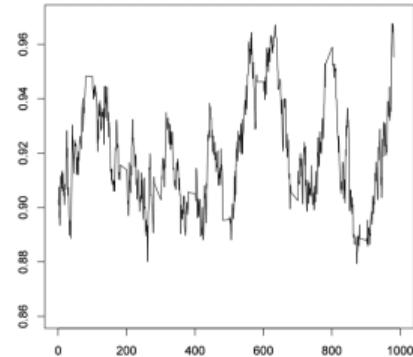
- Traceplots of small and big moves provide (relatively) high correlations (known as autocorrelations) between successive observations of the Markov chain.
- Strongly correlated observations require large sample sizes and therefore longer simulations.
- Autocorrelation function (ACF) plots are a convenient way of displaying the strength of autocorrelation in the given sample values.

Autocorrelation functions

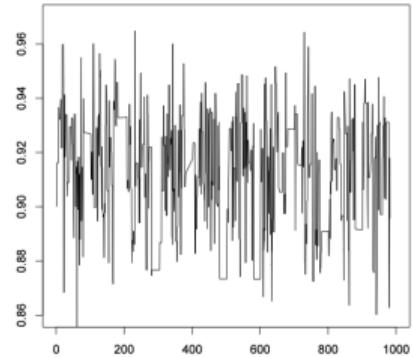
- Traceplots of small and big moves provide (relatively) high correlations (known as autocorrelations) between successive observations of the Markov chain.
- Strongly correlated observations require large sample sizes and therefore longer simulations.
- Autocorrelation function (ACF) plots are a convenient way of displaying the strength of autocorrelation in the given sample values.
- ACF plots provide the autocorrelation between successively sampled values separated by k iterations, referred to as lag, (i.e. $\text{cor}(\theta_t, \theta_{t+k})$) for increasing values of k .

ACFs

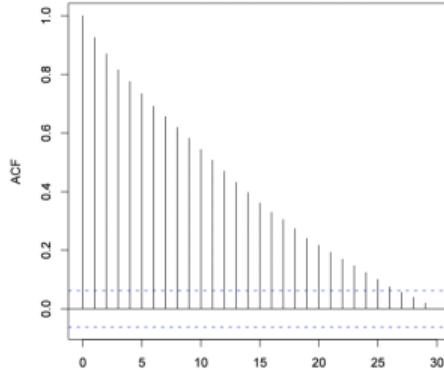
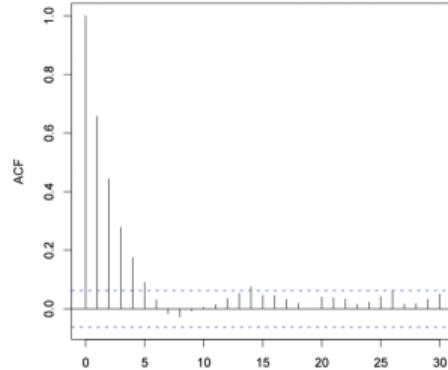
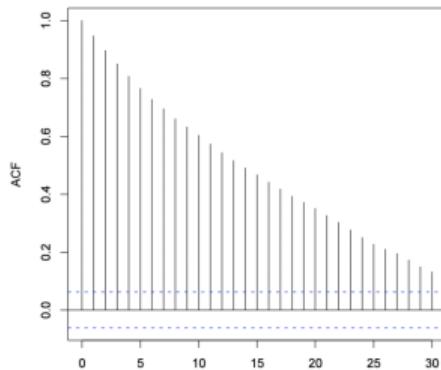
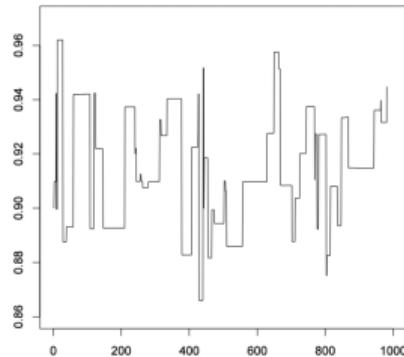
Small moves



OK

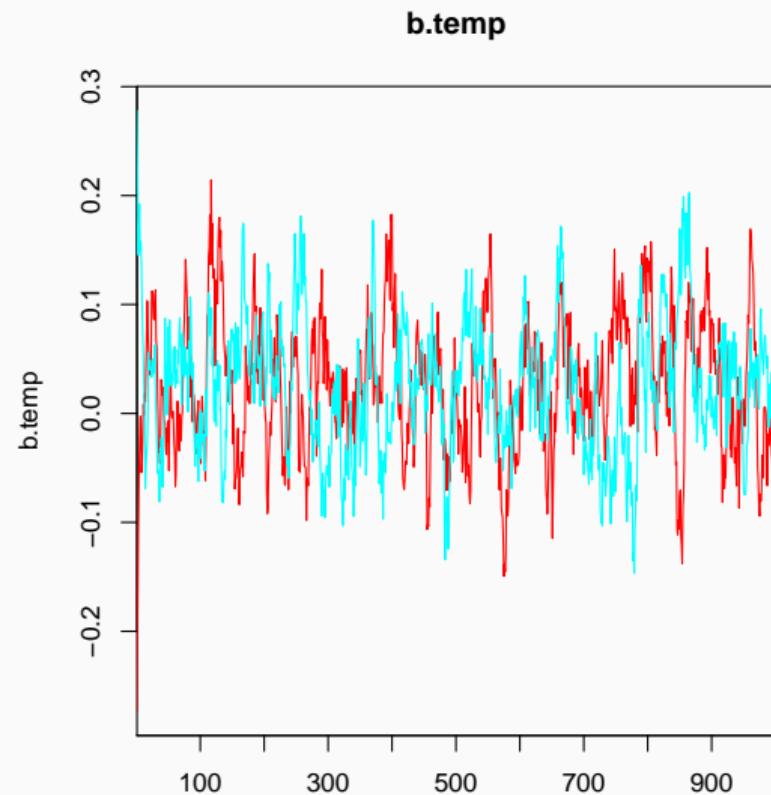
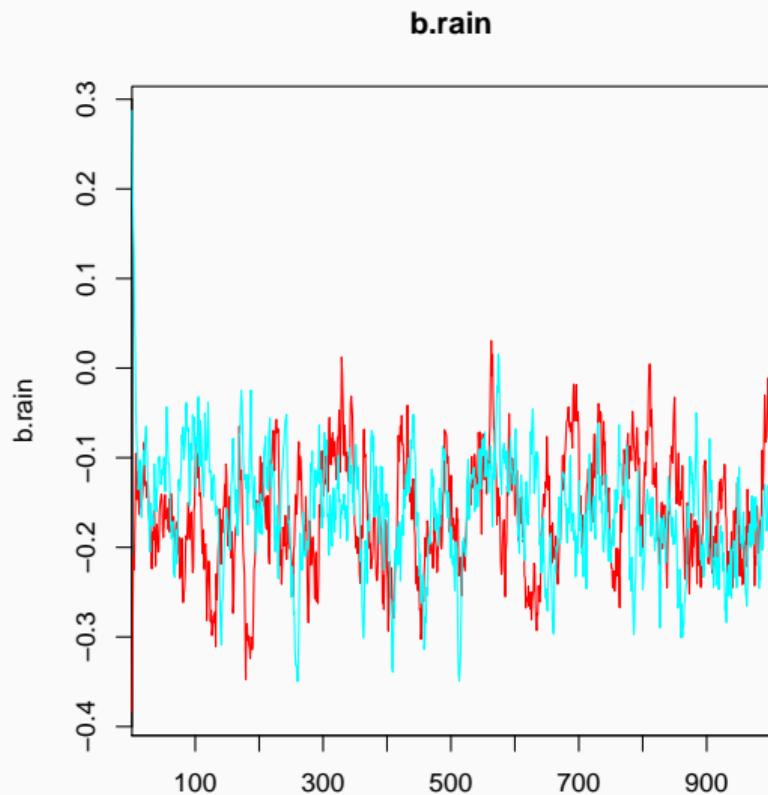


Big moves



Traceplots for the storks

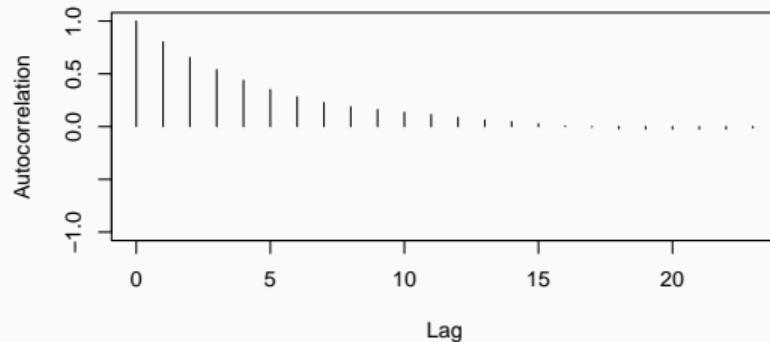
```
traceplot(storks,mfrow = c(1, 2), varname = c('b.rain','b.temp'), ask = FALSE)
```



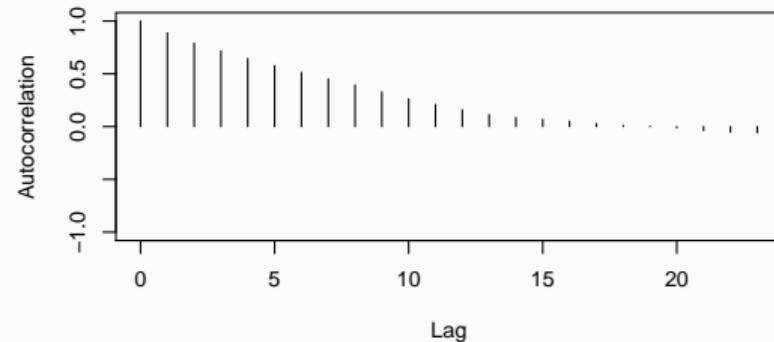
Autocorrelation for the storks

```
autocorr.plot(as.mcmc(storks), ask = FALSE)
```

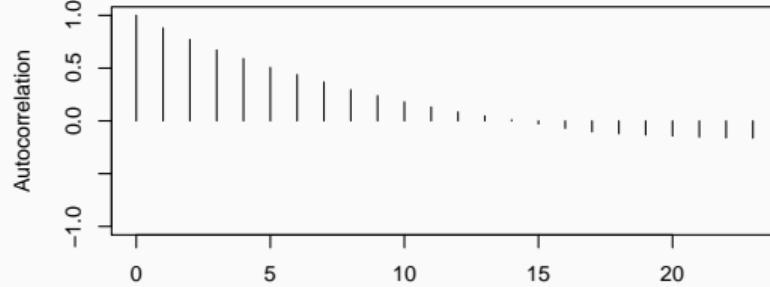
a



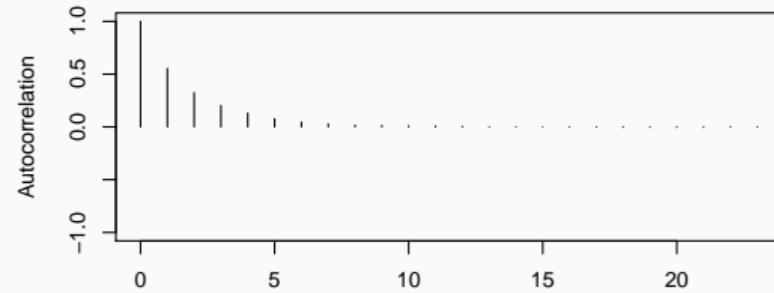
b.rain



b.temp



deviance



How do good chains behave?

- Converge to same target distribution: We need to think of the time required for convergence (realisations of the Markov chain have to be discarded before this is achieved).

How do good chains behave?

- Converge to same target distribution: We need to think of the time required for convergence (realisations of the Markov chain have to be discarded before this is achieved).
- Once there, explore efficiently: The post-convergence sample size required for suitable numerical summaries.

Convergence assessment

- Here, we are looking to determine how long it takes for the Markov chain to converge to the stationary distribution.

Convergence assessment

- Here, we are looking to determine how long it takes for the Markov chain to converge to the stationary distribution.
- In practice, we must discard observations from the start of the chain and just use observations from the chain once it has converged.

Convergence assessment

- Here, we are looking to determine how long it takes for the Markov chain to converge to the stationary distribution.
- In practice, we must discard observations from the start of the chain and just use observations from the chain once it has converged.
- The initial observations that we discard are referred to as the **burn-in**.

Convergence assessment

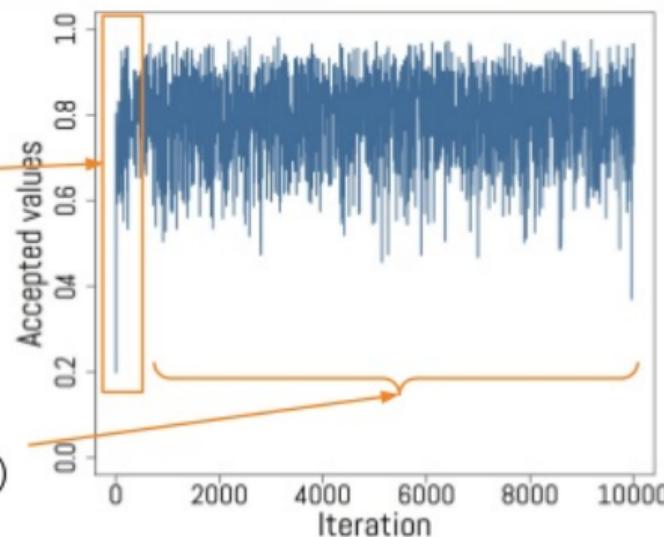
- Here, we are looking to determine how long it takes for the Markov chain to converge to the stationary distribution.
- In practice, we must discard observations from the start of the chain and just use observations from the chain once it has converged.
- The initial observations that we discard are referred to as the **burn-in**.
- The simplest method to determine the length of the burn-in period is to look at trace plots.

Burn-in (if simulations cheap, be conservative)

Discard initial guesses that are still far from optimum: the

BURN-IN

These numbers should be a good sample of the Posterior $P(\phi | \text{data})$



Effective sample size n.eff

- How long of a chain is needed to produce stable estimates ?

Effective sample size n.eff

- How long of a chain is needed to produce stable estimates ?
- Most MCMC chains are strongly autocorrelated.

Effective sample size n.eff

- How long of a chain is needed to produce stable estimates ?
- Most MCMC chains are strongly autocorrelated.
- Successive steps are near each other, and are not independent.

Effective sample size n.eff

- How long of a chain is needed to produce stable estimates ?
- Most MCMC chains are strongly autocorrelated.
- Successive steps are near each other, and are not independent.
- The effective sample size (n.eff) measures chain length while taking into account the autocorrelation of the chain.
 - n.eff is less than the number of MCMC iterations.
 - Check the n.eff of every parameter of interest.
 - Check the n.eff of any interesting parameter combinations.

Effective sample size n.eff

- How long of a chain is needed to produce stable estimates ?
- Most MCMC chains are strongly autocorrelated.
- Successive steps are near each other, and are not independent.
- The effective sample size (n.eff) measures chain length while taking into account the autocorrelation of the chain.
 - n.eff is less than the number of MCMC iterations.
 - Check the n.eff of every parameter of interest.
 - Check the n.eff of any interesting parameter combinations.
- We need $n.eff \geq 100$ independent steps.

Potential scale reduction factor

- Gelman-Rubin statistic \hat{R}

Potential scale reduction factor

- Gelman-Rubin statistic \hat{R}
- Measures the ratio of the total variability combining multiple chains (between-chain plus within-chain) to the within-chain variability. Asks the question is there a chain effect? Very much alike the F test in an ANOVA.

Potential scale reduction factor

- Gelman-Rubin statistic \hat{R}
- Measures the ratio of the total variability combining multiple chains (between-chain plus within-chain) to the within-chain variability. Asks the question is there a chain effect? Very much alike the F test in an ANOVA.
- Values near 1 indicates likely convergence, a value of ≤ 1.1 is considered acceptable.

Potential scale reduction factor

- Gelman-Rubin statistic \hat{R}
- Measures the ratio of the total variability combining multiple chains (between-chain plus within-chain) to the within-chain variability. Asks the question is there a chain effect? Very much alike the F test in an ANOVA.
- Values near 1 indicates likely convergence, a value of ≤ 1.1 is considered acceptable.
- Necessary condition, not sufficient; In other words, these diagnostics cannot tell you that you have converged for sure, only that you have not.

`n.eff` and \hat{R} for the storks

storks

```
#> Inference for Bugs model at "code/logistic.txt", fit using jags,
#> 2 chains, each with 2000 iterations (first 1000 discarded)
#> n.sims = 2000 iterations saved
#>          mu.vect sd.vect    2.5%     25%     50%     75%   97.5% Rhat n.eff
#> a        1.550  0.084   1.430   1.519   1.554   1.589   1.667 1.183   170
#> b.rain   -0.161  0.063  -0.285  -0.201  -0.162  -0.120  -0.047 1.002  1300
#> b.temp    0.026  0.061  -0.087  -0.016   0.026   0.064   0.151 1.001  2000
#> deviance 206.339 30.006 201.776 202.735 203.849 205.676 212.060 1.078  2000
#>
#> For each parameter, n.eff is a crude measure of effective sample size,
#> and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
#>
#> DIC info (using the rule, pD = var(deviance)/2)
#> pD = 450.2 and DIC = 656.6
#> DIC is an estimate of expected predictive error (lower deviance is better).
```

To sum up

- Run multiple chains from arbitrary starting places (initial values).

To sum up

- Run multiple chains from arbitrary starting places (initial values).
- Assume convergence when all chains reach same regime.

To sum up

- Run multiple chains from arbitrary starting places (initial values).
- Assume convergence when all chains reach same regime.
- Discard initial burn-in phase.

To sum up

- Run multiple chains from arbitrary starting places (initial values).
- Assume convergence when all chains reach same regime.
- Discard initial burn-in phase.
- Check autocorrelation, effective sample size and \hat{R} .

What if you have issues of convergence?

- Increase burn-in, sample more.

What if you have issues of convergence?

- Increase burn-in, sample more.
- Use more informative priors.
- Pick better initial values (good guess).

What if you have issues of convergence?

- Increase burn-in, sample more.
- Use more informative priors.
- Pick better initial values (good guess).
- Reparameterize:
 - Standardize covariates.
 - Non-centering: $\alpha \sim N(0, \sigma)$ becomes $\alpha = z\sigma$ with $z \sim N(0, 1)$.

What if you have issues of convergence?

- Increase burn-in, sample more.
- Use more informative priors.
- Pick better initial values (good guess).
- Reparameterize:
 - Standardize covariates.
 - Non-centering: $\alpha \sim N(0, \sigma)$ becomes $\alpha = z\sigma$ with $z \sim N(0, 1)$.
- Something wrong with your model?
 - Start with a simpler model (remove complexities).
 - Use simulations.

What if you have issues of convergence?

- Increase burn-in, sample more.
- Use more informative priors.
- Pick better initial values (good guess).
- Reparameterize:
 - Standardize covariates.
 - Non-centering: $\alpha \sim N(0, \sigma)$ becomes $\alpha = z\sigma$ with $z \sim N(0, 1)$.
- Something wrong with your model?
 - Start with a simpler model (remove complexities).
 - Use simulations.
- Change your sampler. Upgrade to Nimble or Stan.

**MCMC makes you queens and kings
of the stats world**

Get all values sampled from posteriors

```
res <- as.mcmc(storks) # convert outputs in a list
res <- rbind(res[[1]],res[[2]]) # put two MCMC lists on top of each other
head(res)

#>           a      b.rain      b.temp deviance
#> [1,] 0.05606234 -0.3831903 -0.273593867 1233.3921
#> [2,] 0.41740637 -0.3036765 -0.186492758  760.3424
#> [3,] 0.67028274 -0.2089572 -0.114716329  517.6488
#> [4,] 0.83857082 -0.2247830 -0.061526055  399.1947
#> [5,] 0.95558592 -0.2016857 -0.030715601  334.9337
#> [6,] 1.08709297 -0.1557893 -0.002744679  279.9401
```

Compute a posteriori $\Pr(\text{rain} < 0)$

```
# probability that the effect of rainfall is negative  
mean(res[, 'b.rain'] < 0)  
#> [1] 0.994
```

Compute a posteriori $\Pr(\text{temp} < 0)$

```
# probability that the effect of temperature is negative  
mean(res[, 'b.temp'] < 0)  
#> [1] 0.334
```

Get credible interval for the rain effect

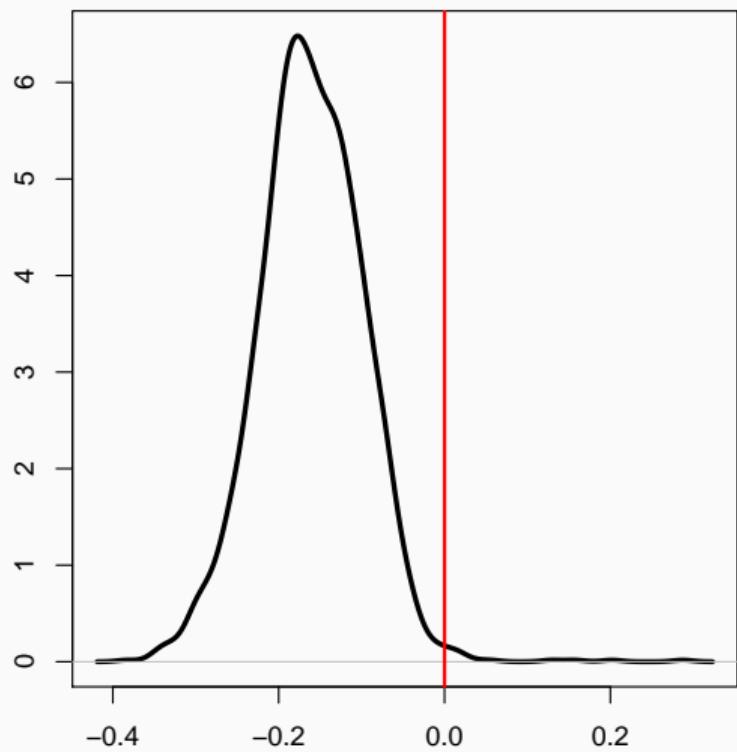
```
quantile(res[, 'b.rain'], c(0.025, 0.975))  
#>      2.5%     97.5%  
#> -0.28547618 -0.04708845
```

Get credible interval for the temperature effect

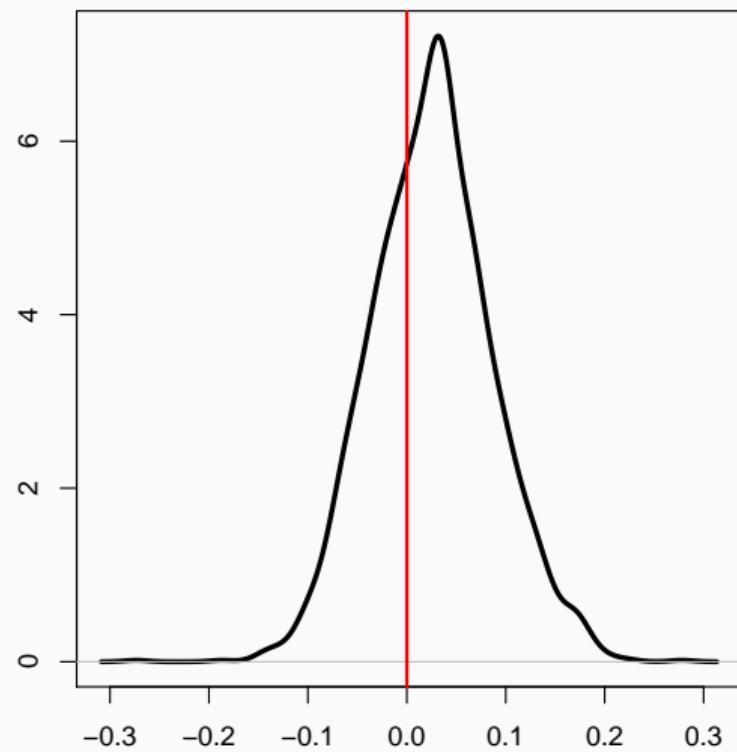
```
quantile(res[, 'b.temp'], c(0.025, 0.975))
#>      2.5%    97.5%
#> -0.08688747  0.15064221
```

Graphical summaries

Rainfall



Temperature



Your turn

A stupid question

- Get the posterior distribution of $b_{rain}^2 + \cos(b_{temp})$

Solution

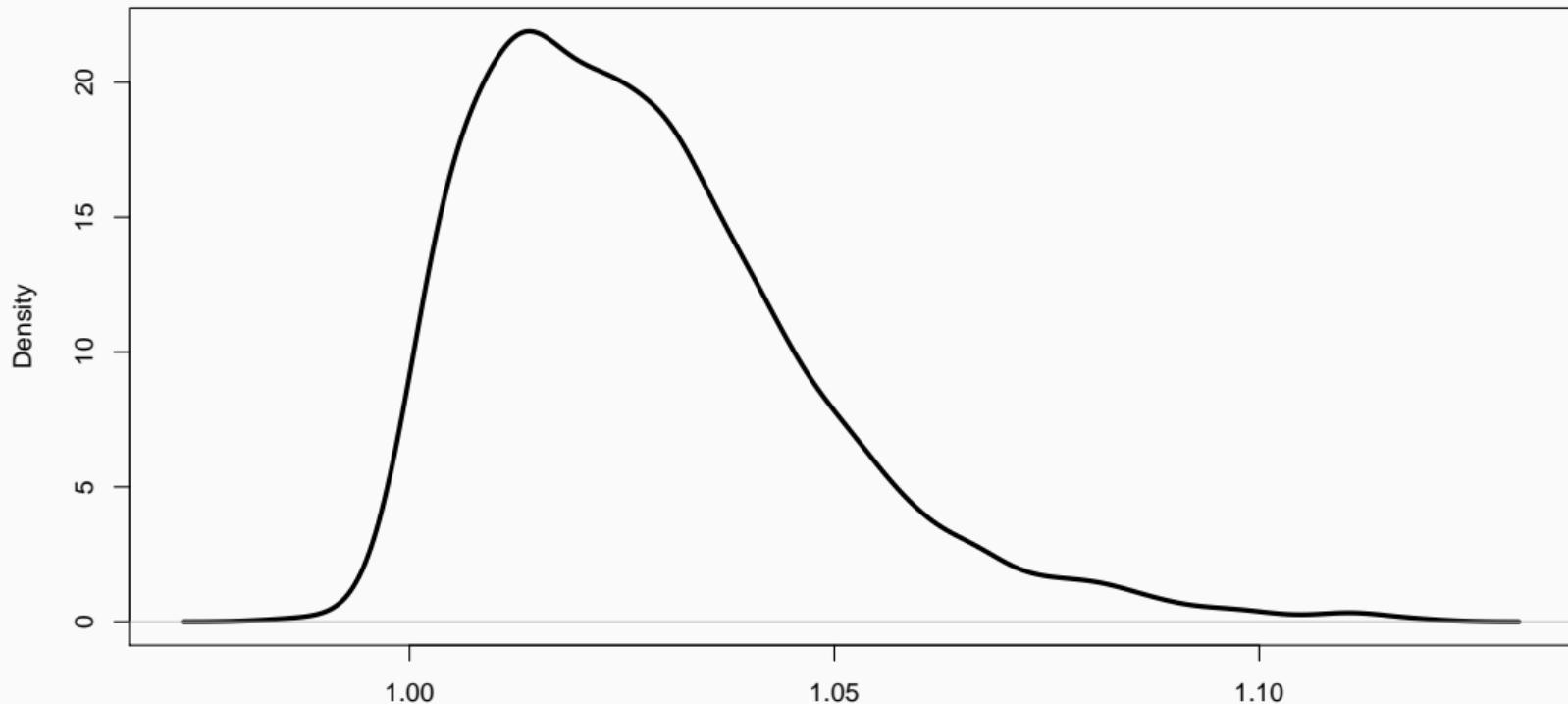
R code

- Evaluate the function for each MCMC iteration

```
stupid_pd <- res[, 'b.rain']^2 + cos(res[, 'b.temp'])  
head(stupid_pd)  
#> [1] 1.109641 1.074880 1.037090 1.048635 1.040205 1.024267
```

- Plot the distribution

```
plot(density(stupid_pd), xlab = "", main = "", lwd = 3)
```



Model selection

How to select a best model?

- Is there any effect of rain or temperature or both on breeding success?

How to select a best model?

- Is there any effect of rain or temperature or both on breeding success?
- The proportion of explained variance R^2 is problematic, because the more variables you have, the bigger R^2 is.

How to select a best model?

- Is there any effect of rain or temperature or both on breeding success?
- The proportion of explained variance R^2 is problematic, because the more variables you have, the bigger R^2 is.
- Idea: **penalize models with too many parameters.**

Akaike information criterion (AIC)

$$AIC = -2 \log(L(\hat{\theta}_1, \dots, \hat{\theta}_K)) + 2K$$

with L the likelihood and K the number of parameters θ_i .

Akaike information criterion (AIC)

$$\text{AIC} = -2 \log(L(\hat{\theta}_1, \dots, \hat{\theta}_K)) + 2K$$

A measure of goodness-of-fit of the model to the data: the more parameters you have, the smaller the deviance is (or the bigger the likelihood is).

Akaike information criterion (AIC)

$$\text{AIC} = -2 \log(L(\hat{\theta}_1, \dots, \hat{\theta}_K)) + 2K$$

A penalty: twice the number of parameters K

Akaike information criterion (AIC)

- AIC makes the balance between *quality of fit* and *complexity* of a model.

Akaike information criterion (AIC)

- AIC makes the balance between *quality of fit* and *complexity* of a model.
- Best model is the one with lowest AIC value.

Akaike information criterion (AIC)

- AIC makes the balance between *quality of fit* and *complexity* of a model.
- Best model is the one with lowest AIC value.
- Two models are difficult to distinguish if $\Delta\text{AIC} < 2$.

Bayesian version

- Watanabe-Akaike Information Criteria or WAIC:

$$\text{WAIC} = -2 \sum_{i=1}^n \log E[p(y_i | \theta)] + 2p_{\text{WAIC}}$$

- where $E[p(y_i | \theta)]$ is the posterior mean of the likelihood of the i th observation and
- p_{WAIC} is the effective number of parameters computed using the posterior variance of the likelihood.
- Relatively new and not yet available in Jags in routine.

WAIC in Jags

```
# calculate wAIC with JAGS
# https://sourceforge.net/p/mcmc-jags/discussion/610036/thread/8211df61/#ea5c
samples <- jags.samples(storks$model,c("WAIC","deviance"), type = "mean",
                        n.iter = 2000,
                        n.burnin = 1000,
                        n.thin = 1)
```

WAIC in Jags

```
samples$p_waic <- samples$WAIC
samples$waic <- samples$deviance + samples$p_waic
tmp <- sapply(samples, sum)
waic <- round(c(waic = tmp[["waic"]], p_waic = tmp[["p_waic"]]), 1)
waic
#>   waic p_waic
#> 216.3 11.9
```

Your turn

Model selection with WAIC

- Fit models with rainfall effect, temperature effect and without any covariate.
- Rank them with WAIC.

Solution

Model with temperature only

```
# model specification
model <-
paste(
model
{
  for( i in 1 : N)
  {
    nbchicks[i] ~ dbin(p[i],nbpairs[i])
    logit(p[i]) <- a + b * cov[i]
  }

# priors for regression parameters
a ~ dnorm(0,0.001)
b ~ dnorm(0,0.001)
}
")
writeLines(model,"code/logtemp.txt")
```

```
# list of lists of initial values (one for each MCMC chain)
init1 <- list(a = -0.5, b = -0.5)
init2 <- list(a = 0.5, b = 0.5)
inits <- list(init1,init2)

# specify parameters that need to be estimated
parameters <- c("a","b")

# specify nb iterations for burn-in and final inference
nb.burnin <- 1000
nb.iterations <- 2000

# read in data
datax <- list(N = 23, nbchicks = nbchicks, nbpairs = nbpairs,
               cov = (temp - mean(temp))/sd(temp))
```

```
# load R2jags to run Jags through R
storks_temp <- jags(data = datax,
                      inits = inits,
                      parameters.to.save = parameters,
                      model.file = "code/logtemp.txt",
                      n.chains = 2,
                      n.iter = nb.iterations,
                      n.burnin = nb.burnin)

#> Compiling model graph
#> Resolving undeclared variables
#> Allocating nodes
#> Graph information:
#>   Observed stochastic nodes: 23
#>   Unobserved stochastic nodes: 2
#>   Total graph size: 125
#>
#> Initializing model
```

```
# compute WAIC
samples <- jags.samples(storks_temp$model,c("WAIC","deviance"), type = "mean",
                        n.iter = 2000,
                        n.burnin = 1000,
                        n.thin = 1)
samples$p_waic <- samples$WAIC
samples$waic <- samples$deviance + samples$p_waic
tmp <- sapply(samples, sum)
waic_temp <- round(c(waic = tmp[["waic"]], p_waic = tmp[["p_waic"]]),1)
```

Model with rainfall only

```
# read in data
datax <- list(N = 23, nbchicks = nbchicks, nbpairs = nbpairs,
               cov = (rain - mean(rain))/sd(rain))
```

```
# load R2jags to run Jags through R
storks_temp <- jags(data = datax,
                      inits = inits,
                      parameters.to.save = parameters,
                      model.file = "code/logtemp.txt",
                      n.chains = 2,
                      n.iter = nb.iterations,
                      n.burnin = nb.burnin)

#> Compiling model graph
#> Resolving undeclared variables
#> Allocating nodes
#> Graph information:
#>   Observed stochastic nodes: 23
#>   Unobserved stochastic nodes: 2
#>   Total graph size: 134
#>
#> Initializing model
```

```
# compute WAIC
samples <- jags.samples(storks_temp$model,c("WAIC","deviance"), type = "mean",
                        n.iter = 2000,
                        n.burnin = 1000,
                        n.thin = 1)
samples$p_waic <- samples$WAIC
samples$waic <- samples$deviance + samples$p_waic
tmp <- sapply(samples, sum)
waic_rain <- round(c(waic = tmp[["waic"]], p_waic = tmp[["p_waic"]]),1)
```

Model with no effect of covariates

```
# model specification
model <-
paste(
model
{
  for( i in 1 : N)
  {
    nbchicks[i] ~ dbin(p[i],nbpairs[i])
    logit(p[i]) <- a
  }

# priors for regression parameters
a ~ dnorm(0,0.001)
}
")
writeLines(model,"code/lognull.txt")
```

```
# list of lists of initial values (one for each MCMC chain)
init1 <- list(a = -0.5)
init2 <- list(a = 0.5)
inits <- list(init1,init2)

# specify parameters that need to be estimated
parameters <- c("a")

# specify nb iterations for burn-in and final inference
nb.burnin <- 1000
nb.iterations <- 2000

# read in data
datax <- list(N = 23, nbchicks = nbchicks, npairs = npairs)
```

```
# load R2jags to run Jags through R
storks_temp <- jags(data = datax,
                     inits = inits,
                     parameters.to.save = parameters,
                     model.file = "code/lognull.txt",
                     n.chains = 2,
                     n.iter = nb.iterations,
                     n.burnin = nb.burnin)

#> Compiling model graph
#> Resolving undeclared variables
#> Allocating nodes
#> Graph information:
#>   Observed stochastic nodes: 23
#>   Unobserved stochastic nodes: 1
#>   Total graph size: 51
#>
#> Initializing model
```

```
# compute WAIC
samples <- jags.samples(storks_temp$model,c("WAIC","deviance"), type = "mean",
                        n.iter = 2000,
                        n.burnin = 1000,
                        n.thin = 1)
samples$p_waic <- samples$WAIC
samples$waic <- samples$deviance + samples$p_waic
tmp <- sapply(samples, sum)
waic_null <- round(c(waic = tmp[["waic"]], p_waic = tmp[["p_waic"]]),1)
```

Compare WAIC

```
data.frame(model = c('both_covariates', 'temp', 'rain', 'none'),
           waic = c(waic[1],waic_temp[1],waic_rain[1],waic_null[1]),
           p_waic = c(waic[2],waic_temp[2],waic_rain[2],waic_null[2])) %>%
  arrange(waic)

#>          model   waic   p_waic
#> 1         rain 213.0    9.2
#> 2        none 215.3    6.3
#> 3 both_covariates 216.3   11.9
#> 4       temp 220.3   10.2
```

Model with rainfall only seems to be better supported by the data. In case models have similar WAIC values, model-averaging might be useful.

Multilevel (aka mixed-effect) models

What are multilevel models?

- Multilevel models include both fixed and random effects.

What are multilevel models?

- Multilevel models include both fixed and random effects.
- Random effects are statistical parameters that attempt to **explain noise caused by clusters** of the population you are trying to model.

What are multilevel models?

- Multilevel models include both fixed and random effects.
- Random effects are statistical parameters that attempt to **explain noise caused by clusters** of the population you are trying to model.
- A multilevel model assumes that the dataset being analysed consists of **a hierarchy of different populations** whose differences relate to that hierarchy.

What are multilevel models?

- Multilevel models include both fixed and random effects.
- Random effects are statistical parameters that attempt to **explain noise caused by clusters** of the population you are trying to model.
- A multilevel model assumes that the dataset being analysed consists of **a hierarchy of different populations** whose differences relate to that hierarchy.
- Measurement that come **in clusters** or groups.

Your turn

Question

- Come up with examples of clusters or groups

Solution

Clusters might be:

- Classrooms within schools
- Students within classrooms
- Chapters within books
- Individuals within populations
- Populations within species
- Trajectories within individuals
- Fishes within tanks
- Frogs within ponds
- PhD applicants in doctoral schools
- Nations in continents
- Sex or age are not clusters per se (if we were to sample again, we would take the same levels, e.g. male/female and young/old)

Why do we need multilevel models?

- Model the clustering itself.

Why do we need multilevel models?

- Model the clustering itself.
- Interested in variance components (environmental vs. genetic variance).

Why do we need multilevel models?

- Model the clustering itself.
- Interested in variance components (environmental vs. genetic variance).
- Control for bias due to pseudoreplication (time, space, individual).

McElreath's explanation of multilevel models

- Fixed-effect models have amnesia.

McElreath's explanation of multilevel models

- Fixed-effect models have amnesia.
- Every new cluster (individual, species, classroom) is a new world.

McElreath's explanation of multilevel models

- Fixed-effect models have amnesia.
- Every new cluster (individual, species, classroom) is a new world.
- No information passed among clusters.

McElreath's explanation of multilevel models

- Fixed-effect models have amnesia.
- Every new cluster (individual, species, classroom) is a new world.
- No information passed among clusters.
- Multilevel models remember and pool information. They have memory.

McElreath's explanation of multilevel models

- Fixed-effect models have amnesia.
- Every new cluster (individual, species, classroom) is a new world.
- No information passed among clusters.
- Multilevel models remember and pool information. They have memory.
- Properties of clusters come from a population.

McElreath's explanation of multilevel models

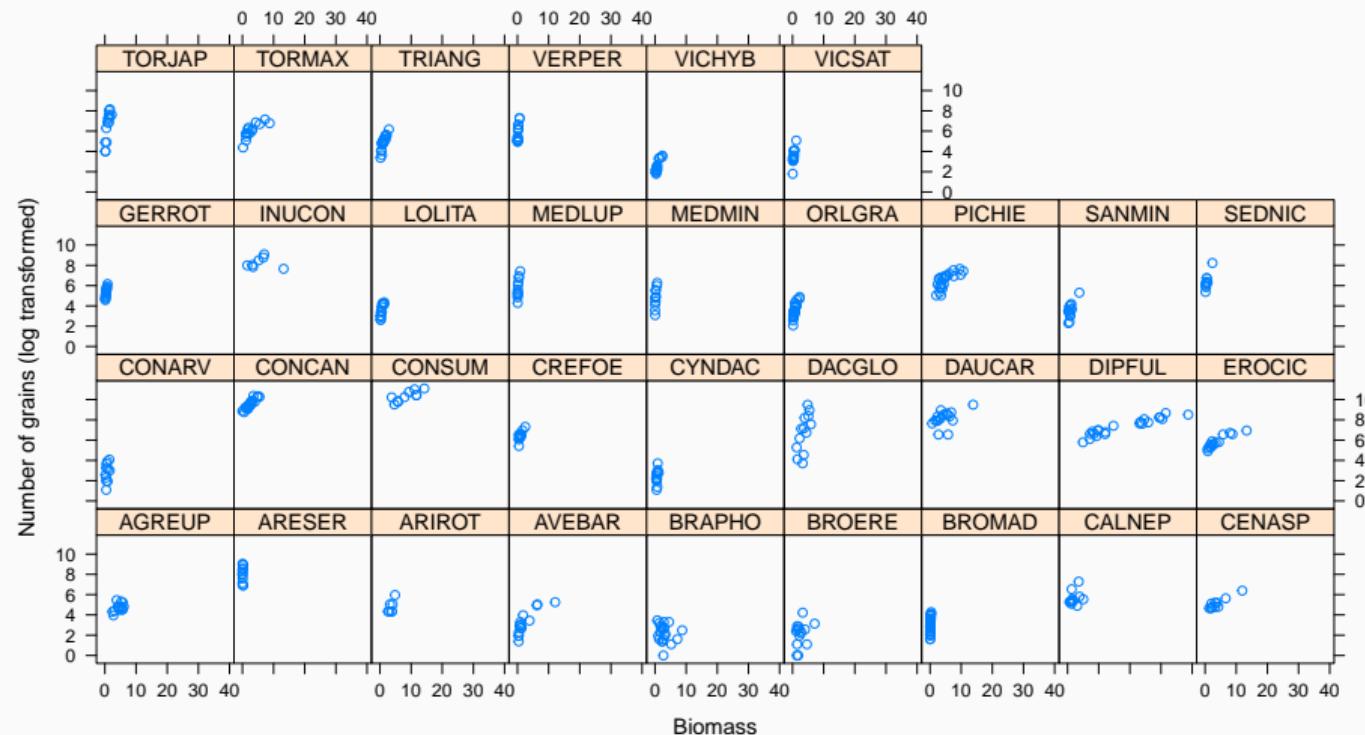
- Fixed-effect models have amnesia.
- Every new cluster (individual, species, classroom) is a new world.
- No information passed among clusters.
- Multilevel models remember and pool information. They have memory.
- Properties of clusters come from a population.
- If previous clusters improve your guess about a new cluster, you want to use pooling.

Plant experiment in the field at CEFE



Courtesy of Pr Eleni Kazakou

Number of grains per species (cluster) as a function of biomass



GLM with complete pooling

$$Y_i \sim \text{Distribution}(\text{mean}_i) \quad [\text{likelihood}]$$

$$\text{link}(\text{mean})_i = \alpha + \beta x_i \quad [\text{linear model}]$$

$\alpha \sim$ to be determined [prior for intercept]

$\beta \sim$ to be determined [prior for slope]

Model with complete pooling. All clusters the same.

GLM with no pooling

$$Y_i \sim \text{Distribution}(\text{mean}_i) \quad [\text{likelihood}]$$

$$\text{link}(\text{mean})_i = \alpha_{\text{CLUSTER}[i]} + \beta x_i \quad [\text{linear model}]$$

$$\alpha_j \sim \text{to be determined} \quad [\text{prior for intercept}]$$

$$\beta \sim \text{to be determined} \quad [\text{prior for slope}]$$

Model with no pooling. All clusters unrelated (fixed effect).

GLMM or GLM with partial pooling

$Y_i \sim \text{Distribution}(\text{mean}_i)$	[likelihood]
$\text{link}(\text{mean})_i = \alpha \text{CLUSTER}[i] + \beta x_i$	[linear model]
$\alpha_j \sim \text{Normal}(\bar{\alpha}, \sigma)$	[prior for varying intercepts]
$\bar{\alpha} \sim \text{to be determined}$	[prior for population mean]
$\sigma \sim \text{to be determined}$	[prior for standard deviation]
$\beta \sim \text{to be determined}$	[prior for slope]

Model with partial pooling. Clusters are somehow related (random effect).

Back to the plant example

Model with complete pooling (all species are the same)

$$\text{nseeds}_i \sim \text{Normal}(\mu_i, \sigma^2) \quad [\text{likelihood}]$$

$$\mu_i = \alpha + \beta \text{ biomass}_i \quad [\text{linear model}]$$

$$\alpha \sim \text{Normal}(0, 1000) \quad [\text{prior for intercept}]$$

$$\beta \sim \text{Normal}(0, 1000) \quad [\text{prior for slope}]$$

$$\sigma \sim \text{Uniform}(0, 100) \quad [\text{prior for standard deviation}]$$

Read in and manipulate data

```
# read in data
VMG <- read.table("dat/VMG.csv", header=TRUE, dec= ".", sep =";")
# nb of seeds (log)
y <- VMG$NGrTotest
# biomass
x <- VMG$Vm
x <- (x - mean(x))/sd(x)
# species name
Sp <- VMG$Sp
# species label
species <- as.numeric(Sp)
# species name
nbspecies <- length(levels(Sp))
# total nb of measurements
n <- length(y)
```

Specify the model in Jags

```
model <-
paste(
model{
for(i in 1:n){
  y[i] ~ dnorm(mu[i],tau.y)
  mu[i] <- a + b*x[i]
}
tau.y <- 1/(sigma.y*sigma.y)
sigma.y ~ dunif(0,100)
a ~ dnorm(0,0.001)
b ~ dnorm(0,0.001)
}
")
writeLines(model,"code/completelpooling.bug")
```

Prepare ingredients for running Jags

```
# data
allom.data <- list(y=y,n=n,x=x)

# initial values
init1 <- list(a=rnorm(1), b=rnorm(1),sigma.y=runif(1))
init2 <- list(a=rnorm(1), b=rnorm(1),sigma.y=runif(1))
inits <- list(init1,init2)

# parameters to be estimated
allom.parameters <- c("a", "b", "sigma.y")
```

Run Jags

```
allom.1 <- R2jags::jags(allom.data,inits,allom.parameters,
                         n.iter = 2500,model.file="code/completelpooling.bug",
                         n.chains = 2, n.burn = 1000)

#> Compiling model graph
#> Resolving undeclared variables
#> Allocating nodes
#> Graph information:
#>   Observed stochastic nodes: 488
#>   Unobserved stochastic nodes: 3
#>   Total graph size: 1956
#>
#> Initializing model
```

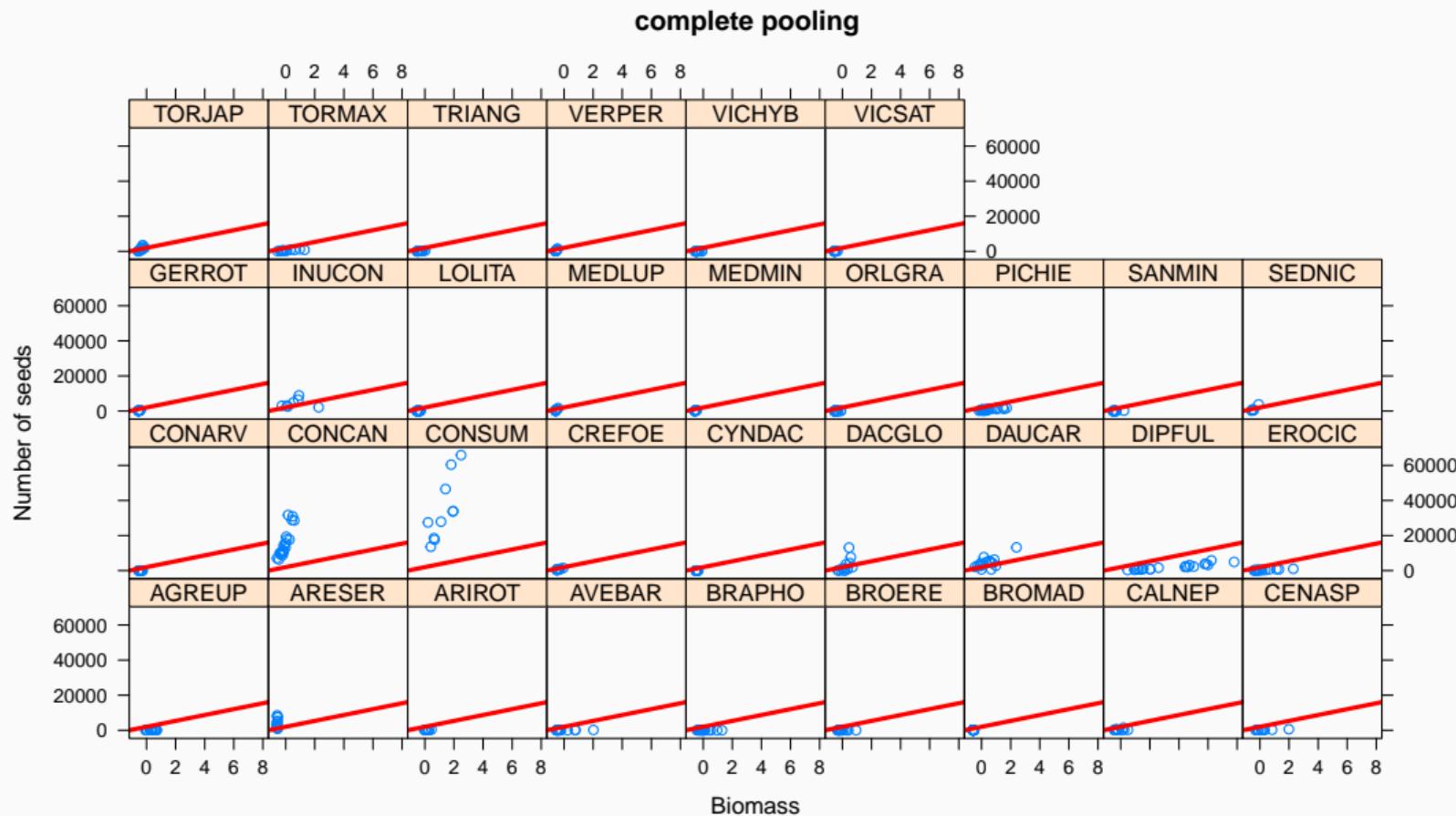
Display results

allom.1

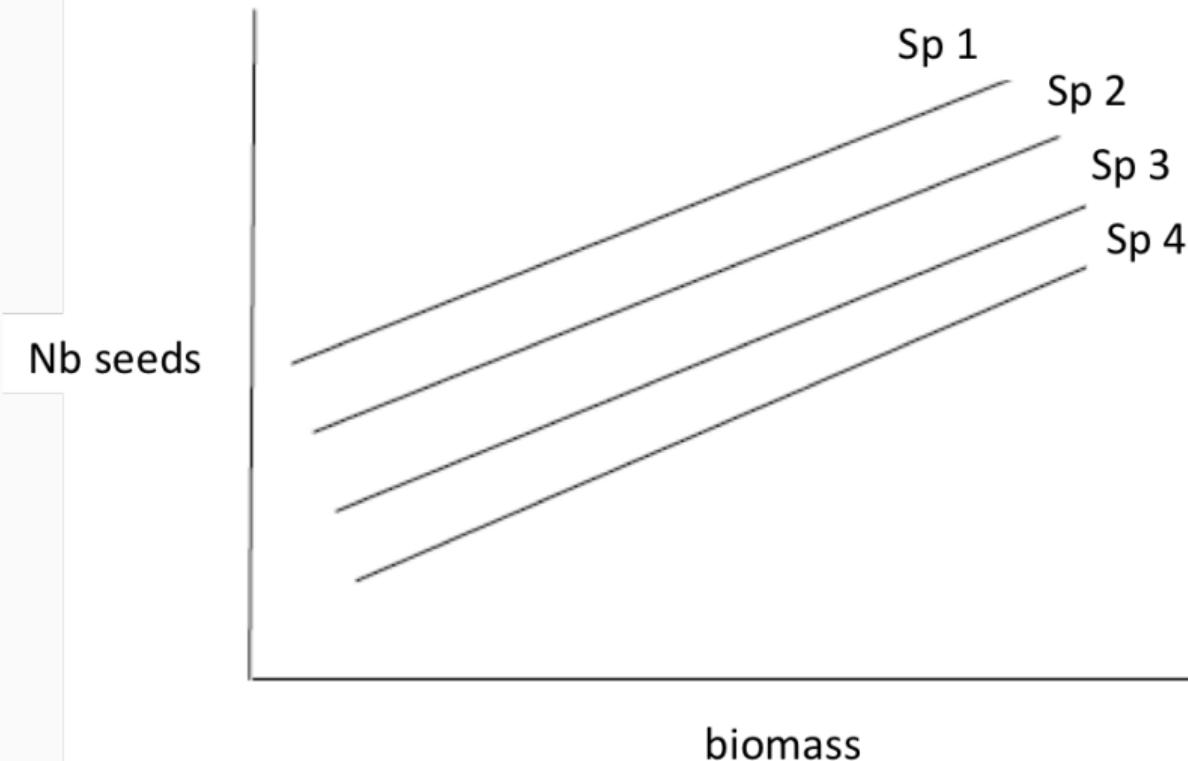
```
#> Inference for Bugs model at "code/completelpooling.bug", fit using jags,
#> 2 chains, each with 2500 iterations (first 1000 discarded)
#> n.sims = 3000 iterations saved

#>          mu.vect sd.vect      2.5%      25%      50%      75%
#> a        1991.179  4.512  1982.553  1988.155  1991.163  1994.231
#> b        1677.244  4.432  1668.553  1674.221  1677.355  1680.264
#> sigma.y   100.000  0.000   100.000   100.000   100.000   100.000
#> deviance 1905427.533 23.596 1905384.594 1905411.183 1905426.348 1905442.954
#>          97.5%  Rhat n.eff
#> a        1999.909 1.001  3000
#> b        1685.535 1.001  3000
#> sigma.y   100.000 1.000     1
#> deviance 1905478.429 1.000     1
#>
#> For each parameter, n.eff is a crude measure of effective sample size,
#> and Rhat is the potential scale reduction factor (at convergence, Rhat=1).
```

Output



Model with partial pooling (species random effect)



Model with partial pooling (all species related in some way)

$$nseeds_i \sim \text{Normal}(\mu_i, \sigma^2) \quad [\text{likelihood}]$$

$$\mu_i = \alpha_{\text{species}[i]} + \beta \text{ biomass}_i \quad [\text{linear model}]$$

$$\alpha_j \sim \text{Normal}(\bar{\alpha}, \sigma_\alpha) \quad [\text{prior for varying intercepts}]$$

$$\bar{\alpha} \sim \text{Normal}(0, 1000) \quad [\text{prior for population mean}]$$

$$\sigma_\alpha \sim \text{Uniform}(0, 100) \quad [\text{prior for } \sigma_\alpha]$$

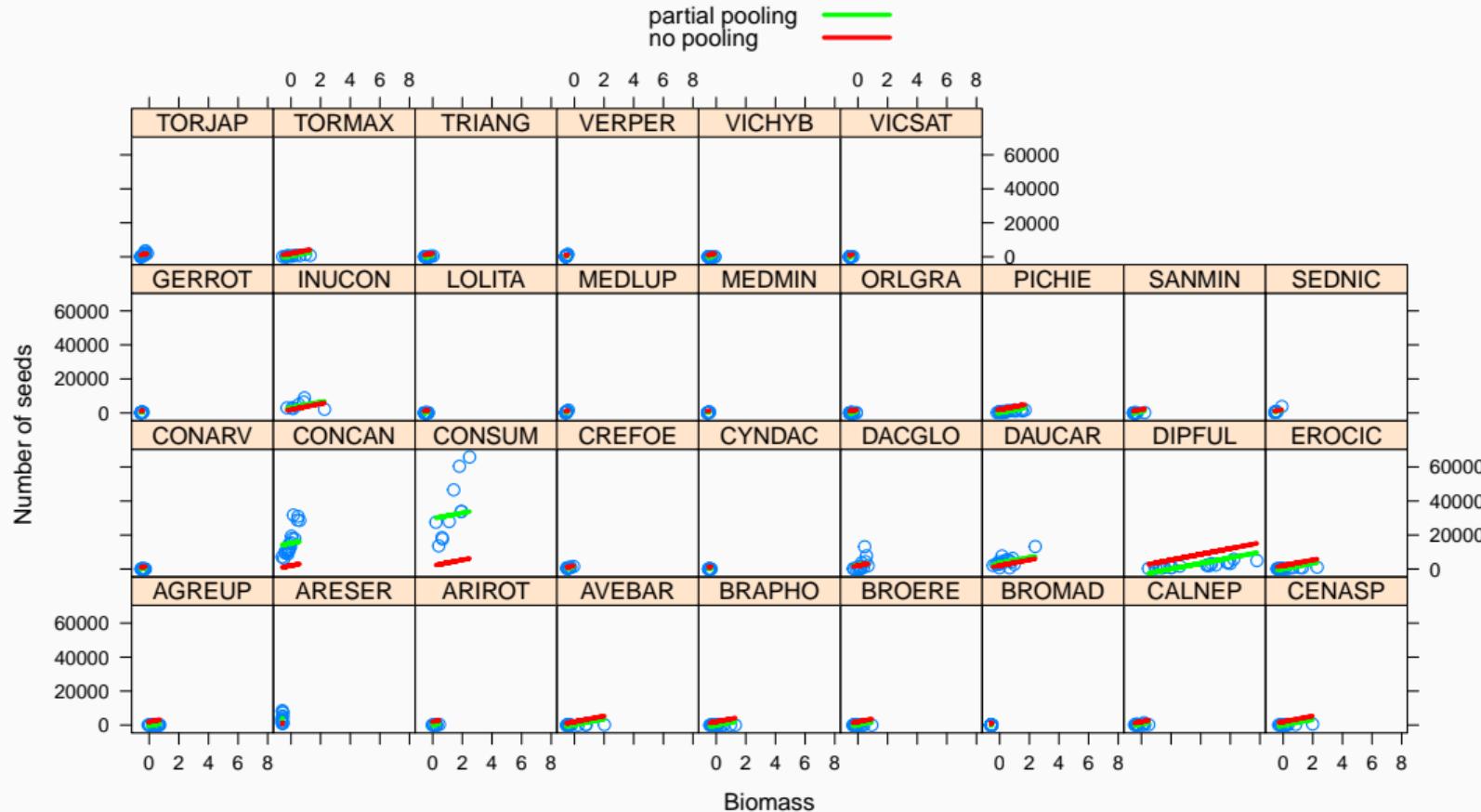
$$\beta \sim \text{Normal}(0, 1000) \quad [\text{prior for slope}]$$

$$\sigma \sim \text{Uniform}(0, 100) \quad [\text{prior for } \sigma]$$

Implementation in Jags

```
model <- paste("
model {
  for (i in 1:n){
    y[i] ~ dnorm (mu[i], tau.y)
    mu[i] <- a[species[i]] + b *x[i]}
  tau.y <- pow(sigma.y, -2)
  sigma.y ~ dunif (0, 100)
  for (j in 1:nbspecies){ a[j] ~ dnorm (mu.a, tau.a)}
  mu.a ~ dnorm (0, 0.001)
  tau.a <- pow(sigma.a, -2)
  sigma.a ~ dunif (0, 100)
  b ~ dnorm (0, 0.001)    }")
writeLines(model,"code/varint.bug")
```

Compare **complete pooling** vs **partial pooling**



Model with no pooling (all species unrelated)

$$\text{nseeds}_i \sim \text{Normal}(\mu_i, \sigma^2) \quad [\text{likelihood}]$$

$$\mu_i = \alpha_{\text{species}[i]} + \beta \text{ biomass}_i \quad [\text{linear model}]$$

$$\alpha_j \sim \text{Normal}(0, 1000) \quad [\text{prior for intercepts}]$$

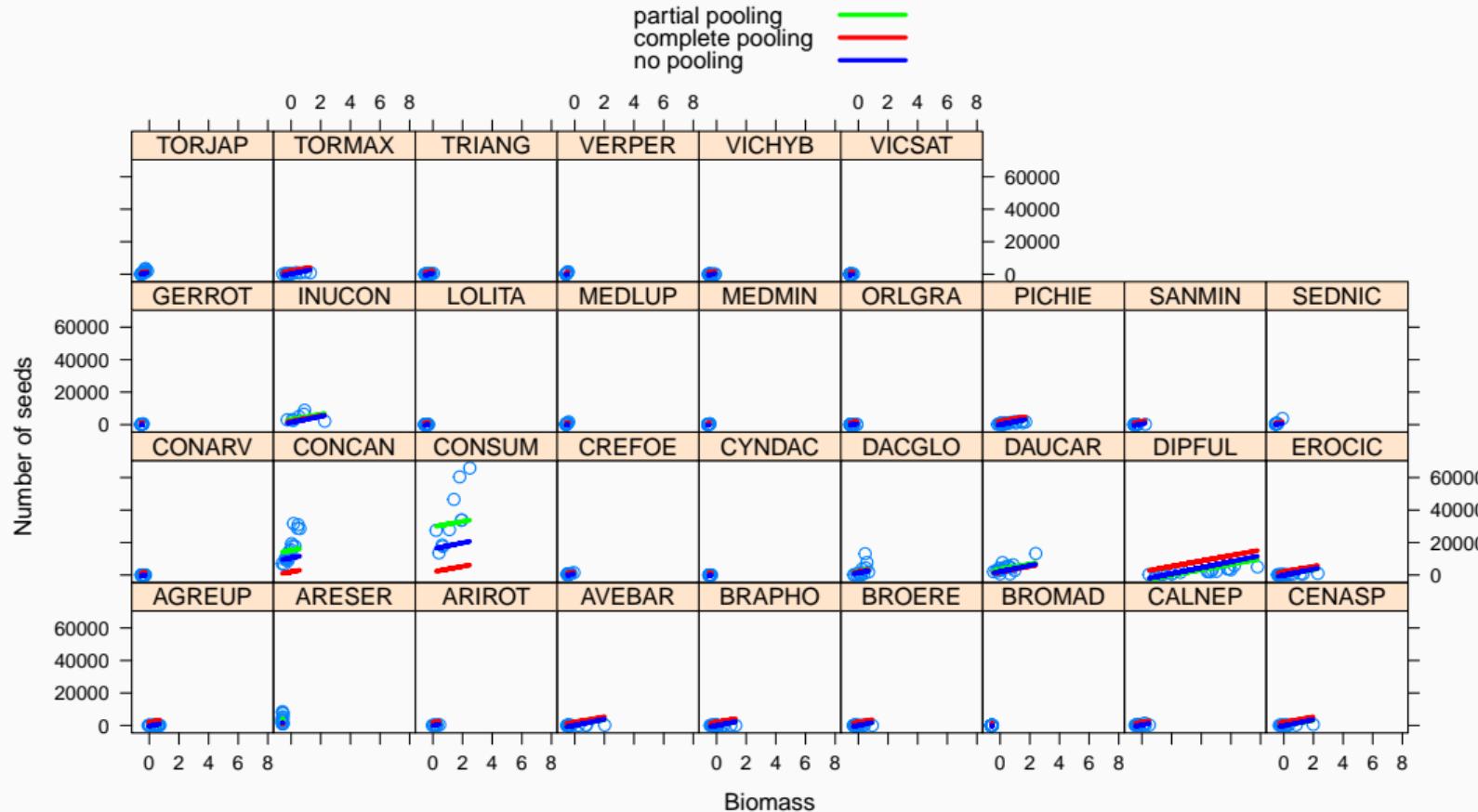
$$\beta \sim \text{Normal}(0, 1000) \quad [\text{prior for slope}]$$

$$\sigma \sim \text{Uniform}(0, 100) \quad [\text{prior for } \sigma]$$

Implementation in Jags

```
model <- paste("
model {
  for (i in 1:n){
    y[i] ~ dnorm (mu[i], tau.y)
    mu[i] <- a[species[i]] + b *x[i]}
  tau.y <- pow(sigma.y, -2)
  sigma.y ~ dunif(0, 100)
  for (j in 1:nbspecies){ a[j] ~ dnorm (0, 0.001)}
  b ~ dnorm (0,0.001)    }")
writeLines(model,"code/nopooling.bug")
```

Compare complete pooling vs partial pooling vs no pooling



Shrinkage results from pooling of information

- Varying effect estimates shrink towards mean ($\bar{\alpha}$).

Shrinkage results from pooling of information

- Varying effect estimates shrink towards mean ($\bar{\alpha}$).
- Avoids underfitting as in complete pooling model (null variance) or overfitting as in no pooling model (infinite variance).

Shrinkage results from pooling of information

- Varying effect estimates shrink towards mean ($\bar{\alpha}$).
- Avoids underfitting as in complete pooling model (null variance) or overfitting as in no pooling model (infinite variance).
- Varying effects: adaptive regularization through cluster variance estimation.

Shrinkage results from pooling of information

- Varying effect estimates shrink towards mean ($\bar{\alpha}$).
- Avoids underfitting as in complete pooling model (null variance) or overfitting as in no pooling model (infinite variance).
- Varying effects: adaptive regularization through cluster variance estimation.
- Further from mean, more shrinkage.

Shrinkage results from pooling of information

- Varying effect estimates shrink towards mean ($\bar{\alpha}$).
- Avoids underfitting as in complete pooling model (null variance) or overfitting as in no pooling model (infinite variance).
- Varying effects: adaptive regularization through cluster variance estimation.
- Further from mean, more shrinkage.
- Fewer data in cluster, more shrinkage.

Multilevel models are awesome!

Multilevel models in a nutshell

- **Shrinkage via pooling is desirable.** The no-pooling model overstates variation among clusters and makes the individual clusters look more different than they are (overfitting). The complete-pooling model simply ignores the variation among clusters (underfitting).

Multilevel models in a nutshell

- **Shrinkage via pooling is desirable.** The no-pooling model overstates variation among clusters and makes the individual clusters look more different than they are (overfitting). The complete-pooling model simply ignores the variation among clusters (underfitting).
- We can **generalize to a wider population.** Is there an allometry relationship between number of seeds and biomass?

Multilevel models in a nutshell

- **Shrinkage via pooling is desirable.** The no-pooling model overstates variation among clusters and makes the individual clusters look more different than they are (overfitting). The complete-pooling model simply ignores the variation among clusters (underfitting).
- We can **generalize to a wider population.** Is there an allometry relationship between number of seeds and biomass?
- We may consider **varying slopes.** We'd need to deal with correlations between intercept and slope random effects. Open a whole new world with spatial (or time) autocorrelation, phylogenetic regressions, quantitative genetics, network models.

Multilevel models in a nutshell

- **Shrinkage via pooling is desirable.** The no-pooling model overstates variation among clusters and makes the individual clusters look more different than they are (overfitting). The complete-pooling model simply ignores the variation among clusters (underfitting).
- We can **generalize to a wider population.** Is there an allometry relationship between number of seeds and biomass?
- We may consider **varying slopes**. We'd need to deal with correlations between intercept and slope random effects. Open a whole new world with spatial (or time) autocorrelation, phylogenetic regressions, quantitative genetics, network models.
- We may **include predictors at the cluster level.** Imagine we know something about functional traits, and wish to determine whether some species-to-species variation in the allometry relationship is explained by these traits.

Your turn

Model selection with WAIC

- Consider the plant example. Compare the three models (no, partial and complete pooling) with WAIC.

Solution

R code

WAIC of model with no pooling

```
samples <- jags.samples(allom.1$model,c("WAIC","deviance"), type = "mean",
                        n.iter = 2000, n.burnin = 1000, n.thin = 1)
samples$p_waic <- samples$WAIC; samples$waic <- samples$deviance + samples$p_waic
tmp <- sapply(samples, sum)
waic_completenesspooling <- round(c(waic = tmp[["waic"]], p_waic = tmp[["p_waic"]]),1)
```

WAIC of model with partial pooling

```
samples <- jags.samples(allom.2$model,c("WAIC","deviance"), type = "mean",
                        n.iter = 2000, n.burnin = 1000, n.thin = 1)
samples$p_waic <- samples$WAIC; samples$waic <- samples$deviance + samples$p_waic
tmp <- sapply(samples, sum)
waic_partialpooling <- round(c(waic = tmp[["waic"]], p_waic = tmp[["p_waic"]]),1)
```

WAIC of model with complete pooling

```
samples <- jags.samples(allom.3$model,c("WAIC","deviance"), type = "mean",
                        n.iter = 2000, n.burnin = 1000, n.thin = 1)
samples$p_waic <- samples$WAIC; samples$waic <- samples$deviance + samples$p_waic
tmp <- sapply(samples, sum)
waic_nopooling <- round(c(waic = tmp[["waic"]], p_waic = tmp[["p_waic"]]),1)
```

```
data.frame(model = c('no pooling', 'partial pooling', 'complete pooling'),
           waic = c(waic_nopooling[1],
                     waic_partialpooling[1],
                     waic_completelpooling[1]),
           p_waic = c(waic_nopooling[2],
                      waic_partialpooling[2],
                      waic_completelpooling[2])) %>%
  arrange(waic)
#> #>          model      waic   p_waic
#> #> 1  partial pooling  473128.1 35758.6
#> #> 2      no pooling  807530.1 40056.3
#> #> 3 complete pooling 1922799.7 17372.6
```

Conclusions

Take-home messages about Bayesian statistics

- Frees the modeler in you (M. Kéry)
 - Uses probability to quantify uncertainty for everything (propagation of uncertainty).
 - Allows use of prior information ('better' estimates).
 - Can fit complex (hierarchical) models with same MCMC algorithms.

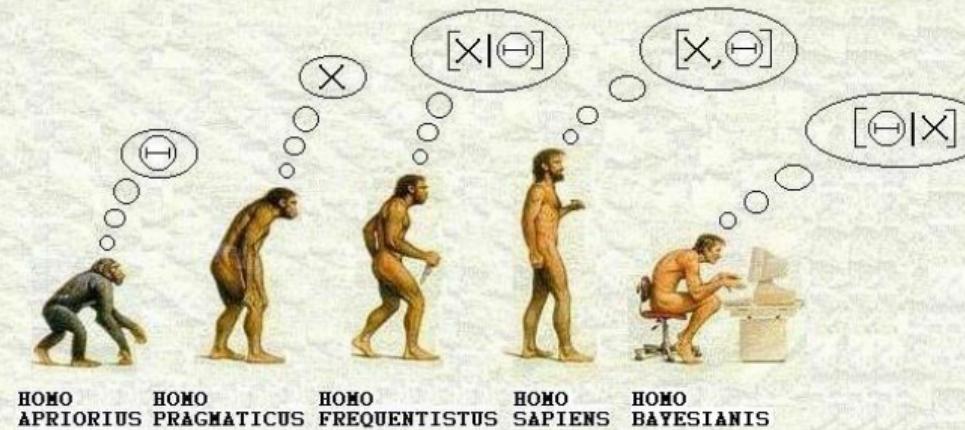
Take-home messages about Bayesian statistics

- Frees the modeler in you (M. Kéry)
 - Uses probability to quantify uncertainty for everything (propagation of uncertainty).
 - Allows use of prior information ('better' estimates).
 - Can fit complex (hierarchical) models with same MCMC algorithms.
- With great tools come great responsibilities
 - Checking convergence is painful.
 - Specifying priors might be tricky.
 - Model adequacy should be checked (posterior predictive checks - not covered).
 - Computational burden can be high (see function `R2jags::jags.parallel()` and package '[jagsUI](#)'.

Take-home messages about Bayesian statistics

- Frees the modeler in you (M. Kéry)
 - Uses probability to quantify uncertainty for everything (propagation of uncertainty).
 - Allows use of prior information ('better' estimates).
 - Can fit complex (hierarchical) models with same MCMC algorithms.
- With great tools come great responsibilities
 - Checking convergence is painful.
 - Specifying priors might be tricky.
 - Model adequacy should be checked (posterior predictive checks - not covered).
 - Computational burden can be high (see function `R2jags::jags.parallel()` and package '[jagsUI](#)'.
- So what?
 - Make an informed and pragmatic choice.
 - Are you after complexity, speed, uncertainties, etc?
 - Talk to colleagues.

(YET ANOTHER) HISTORY OF LIFE AS WE KNOW IT...



Why become a bayesian? Ask twitter!

 Chelsea Parlett-Pelleriti
@ChelseaParlett

Why did you become a Bayesian, wrong answers only

[Traduire le Tweet](#)



A GIF showing a close-up of a deer's head from a front-on perspective. The deer has large, white-tipped ears and is looking down at a white bowl filled with cereal. The background is a soft-focus green and brown, suggesting a natural setting. A small "GIF" label is visible in the bottom left corner of the image frame.

Bonus

Longitudinal study on coral reef and GLMM (Poisson)

A survey of a coral reef uses 10 predefined linear transects covered by divers once every week. The response variable of interest is the abundance of a particular species of anemone as a function of water temperature. Counts of anemones are recorded at 20 regular line segments along the transect. The following piece of code will generate a data set with realistic properties according to the above design. Make sure you understand what it is doing. You might want to explain the script to the colleague next to you. Also, to try and make sense of the code of others, it is always good to plot and/or run small sections of the code.

From Jason Matthiopoulos' book.

```
transects <- 10
data <- NULL
for (tr in 1:transects){
  ref <- rnorm(1,0,.5) # random effect (intercept)
  t <- runif(1, 18,22) + runif(1,-.2,0.2)*1:20 # water temperature gradient
  ans <- exp(ref -14 + 1.8 * t - 0.045 * t^2) # Anemone gradient (expected response)
  an <- rpois(20, ans) # actual counts on 20 segments of the current transect
  data <- rbind(data,cbind(rep(tr, 20), t, an))
}
```

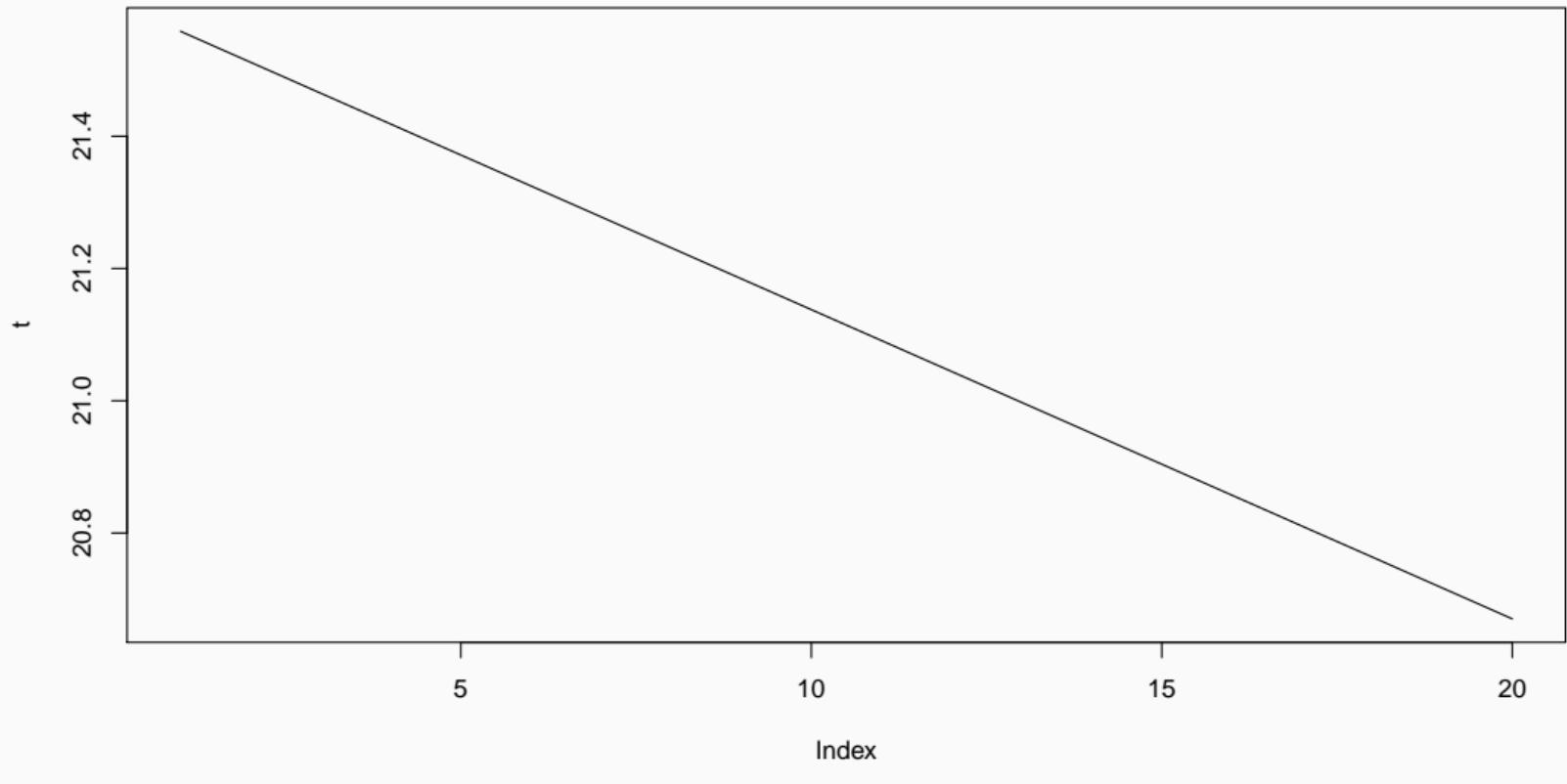
- Generate a data set using the anemone code and fit a GLMM with quadratic effect of temperature and a random intercept.
- Fit the same model to the same data in a Frequentist framework using function `lme4::glmer()`.
- Compare the estimates.

Solution

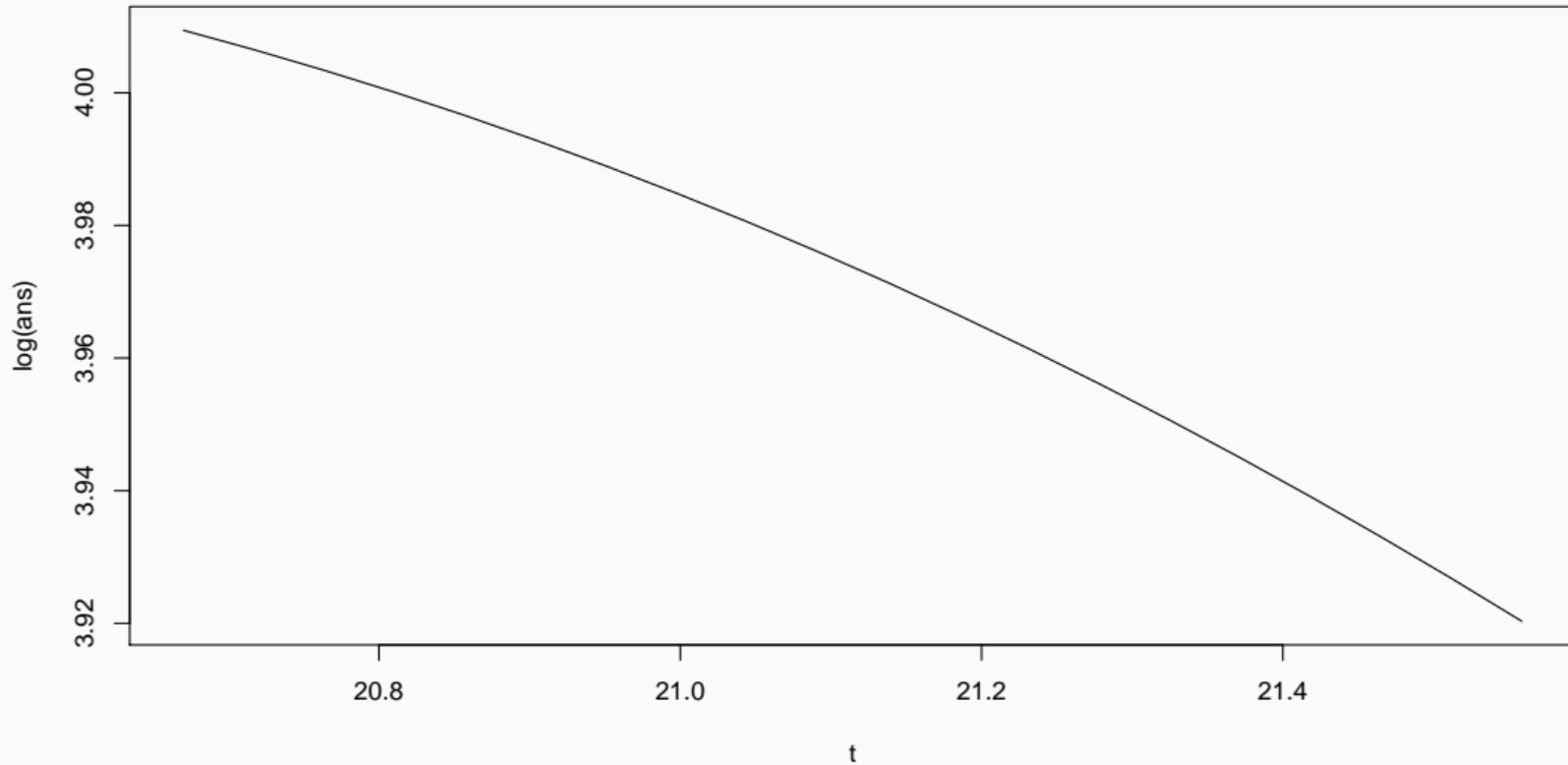
Make sense of the code

- Always difficult to make sense of the code of others.
- Good to plot and/or run small sections of the code.

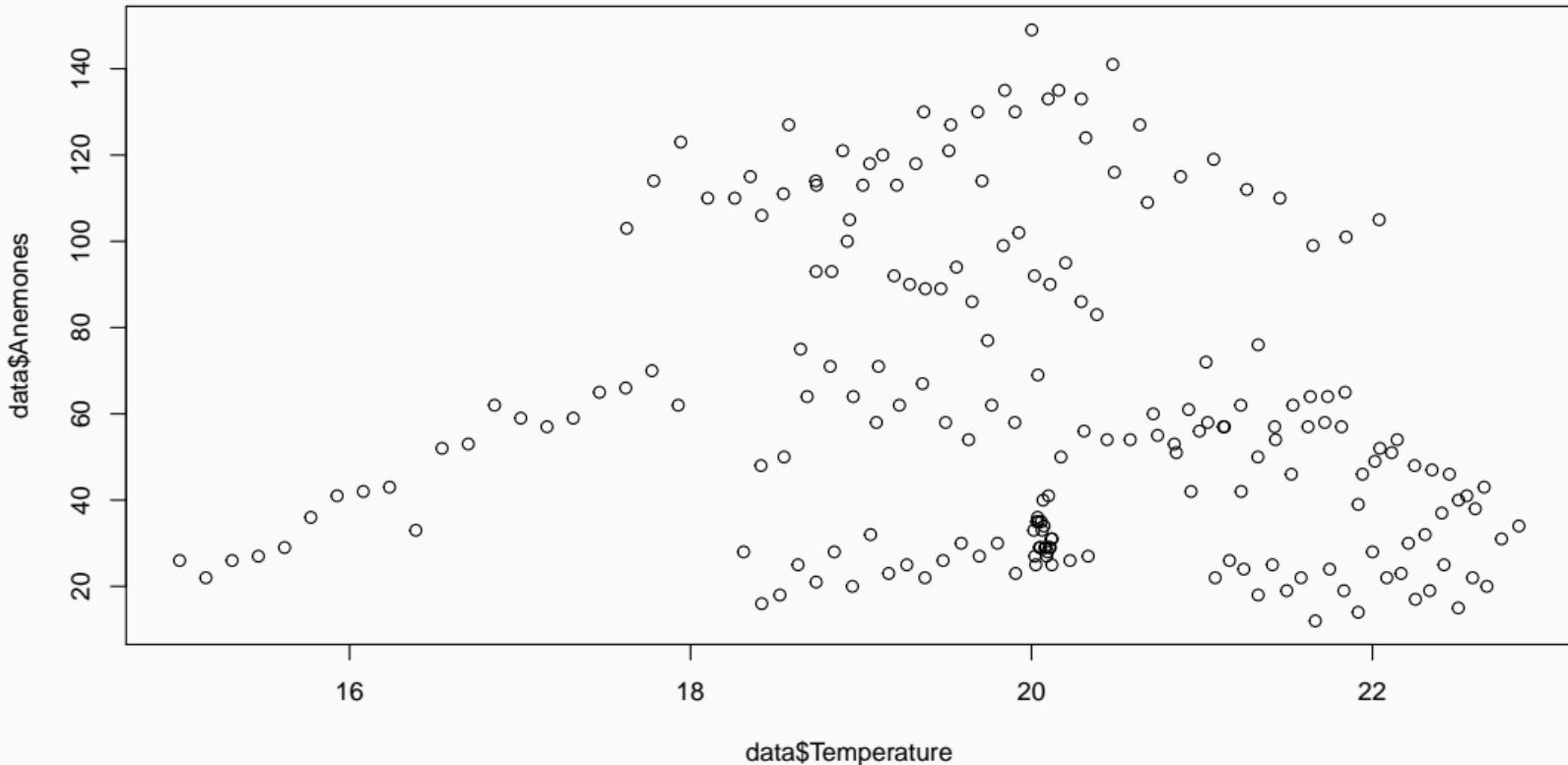
```
ref <- rnorm(1,0,.5) # random effect (intercept)
t <- runif(1, 18,22) + runif(1,-.2,0.2)*1:20 # water temperature gradient
plot(t,type='l')
```



```
ans <- exp(ref -14 + 1.8 * t - 0.045 * t^2) # Anemone gradient (expected response)
plot(t,log(ans),type='l')
```



```
data <- data.frame(Transect=data[,1],Temperature=data[,2],Anemones=data[,3])
plot(data$Temperature, data$Anemones)
```



Write down model

$\text{Count}_i \sim \text{Poisson}(\lambda_i)$ [likelihood]

$\log(\lambda_i) = a_{\text{TRANSECT}[i]} + b_1 \text{ temp}_i + b_2 \text{ temp}_i^2$ [linear model]

$a_j \sim \text{Normal}(\bar{a}, \sigma)$ [prior for varying intercepts]

$\bar{a} \sim \text{Normal}(0, 1000)$ [prior for population mean]

$\sigma \sim \text{Uniform}(0, 100)$ [prior for standard deviation]

$b_1, b_2 \sim \text{Normal}(0, 1000)$ [prior for slopes]

Standardize Temperature covariate.

```
data$Temp <- (data$Temperature - mean(data$Temperature))/sd(data$Temperature)
head(data)

#>   Transect Temperature Anemones      Temp
#> 1      1    20.63519     127  0.38182300
#> 2      1    20.47682     141  0.28904349
#> 3      1    20.31845     124  0.19626397
#> 4      1    20.16008     135  0.10348446
#> 5      1    20.00171     149  0.01070494
#> 6      1    19.84334     135 -0.08207458
```

```
model <-
paste(
model {
  for (i in 1:n){
    count[i] ~ dpois(lambda[i])
    log(lambda[i]) <- a[transect[i]] + b[1] * x[i] + b[2] * pow(x[i],2)
  }
  for (j in 1:nbtransects){
    a[j] ~ dnorm (mu.a, tau.a)
  }
  mu.a ~ dnorm (0, 0.001)
  tau.a <- pow(sigma.a, -2)
  sigma.a ~ dunif (0, 100)
  b[1] ~ dnorm (0, 0.001)
  b[2] ~ dnorm (0, 0.001)
}
")
writeLines(model,"code/GLMMpoisson.bug")
```

```
dat <- list(n = nrow(data),  
            nbtransects = transects,  
            x = data$Temp,  
            count = data$Anemones,  
            transect = data$Transect)  
  
init1 <- list(a=rnorm(transects), b=rnorm(2), mu.a=rnorm(1), sigma.a=runif(1))  
init2 <- list(a=rnorm(transects), b=rnorm(2), mu.a=rnorm(1), sigma.a=runif(1))  
inits <- list(init1,init2)  
par <- c ("a", "b", "mu.a", "sigma.a")
```

```
fit <- jags(dat, inits, par, n.iter = 2500, model.file="code/GLMMpoisson.bug", n.chains = 2)
#> Compiling model graph
#> Resolving undeclared variables
#> Allocating nodes
#> Graph information:
#>   Observed stochastic nodes: 200
#>   Unobserved stochastic nodes: 14
#>   Total graph size: 1622
#>
#> Initializing model
```

fit

```
#> Inference for Bugs model at "code/GLMMpoisson.bug", fit using jags,  
#> 2 chains, each with 2500 iterations (first 1000 discarded)  
#> n.sims = 3000 iterations saved
```

#>	mu.vect	sd.vect	2.5%	25%	50%	75%	97.5%	Rhat
#> a[1]	4.879	0.022	4.836	4.864	4.878	4.894	4.923	1.001
#> a[2]	4.098	0.029	4.042	4.078	4.098	4.118	4.157	1.001
#> a[3]	3.278	0.058	3.166	3.239	3.278	3.316	3.391	1.002
#> a[4]	4.457	0.064	4.333	4.414	4.459	4.501	4.582	1.001
#> a[5]	4.235	0.044	4.151	4.206	4.235	4.265	4.320	1.002
#> a[6]	4.818	0.022	4.773	4.803	4.818	4.832	4.862	1.001
#> a[7]	3.265	0.044	3.176	3.236	3.266	3.295	3.350	1.001
#> a[8]	4.059	0.042	3.976	4.031	4.058	4.087	4.142	1.004
#> a[9]	4.523	0.024	4.475	4.507	4.523	4.540	4.570	1.001
#> a[10]	3.454	0.041	3.373	3.427	3.454	3.481	3.530	1.003
#> b[1]	-0.031	0.020	-0.072	-0.045	-0.031	-0.017	0.008	1.001
#> b[2]	-0.164	0.014	-0.190	-0.173	-0.164	-0.155	-0.137	1.001
#> mu.a	4.105	0.235	3.626	3.962	4.105	4.248	4.565	1.001
#> sigma.a	0.707	0.209	0.430	0.567	0.660	0.802	1.201	1.001
#> deviance	1310.607	5.108	1302.941	1306.916	1309.862	1313.340	1322.743	1.002

```

library(lme4)
fit_lme4 <- glmer(Anemones ~ Temp + I(Temp^2) + (1 | Transect), data=data, family=poisson)
fit_lme4
#> Generalized linear mixed model fit by maximum likelihood (Laplace
#> Approximation) [glmerMod]
#> Family: poisson ( log )
#> Formula: Anemones ~ Temp + I(Temp^2) + (1 / Transect)
#> Data: data
#>      AIC      BIC      logLik  deviance df.resid
#> 1374.7947 1387.9879 -683.3973 1366.7947      196
#> Random effects:
#> Groups   Name        Std.Dev.
#> Transect (Intercept) 0.5687
#> Number of obs: 200, groups: Transect, 10
#> Fixed Effects:
#> (Intercept)          Temp      I(Temp^2)
#>     4.10758     -0.03174    -0.16345

```

Parameter estimates obtained with both approaches are very similar.

```
visreg:::visreg(fit_lme4,xvar='Temp')
```

