*Olivier Gimenez*

# *Bayesian Analysis of Capture-Recapture Data with Hidden Markov Models – Theory and Case Studies in R*

# *Contents*

# *List of Tables*

# *List of Figures*

# *Welcome*

Welcome to the website of the book *Bayesian Analysis of Capture-Recapture Data with Hidden Markov Models – Theory and Case Studies in R* by Olivier Gimenez[1]. Note that the book is also available in PDF format[2].

I'm currently writing this book, and I welcome any feedback or requests for content here[3].

Many thanks!

*Last updated: August 26, 2021*

[1] https://oliviergimenez.github.io/

[2] https://github.com/oliviergimenez/banana-book/raw/master/docs/bayesHMMcapturerecapture.pdf

[3] https://github.com/oliviergimenez/banana-book/issues

# *Preface*

The HMM framework has gained much attention in the ecological literature over the last decade, and has been suggested as a general modelling framework for the demography of plant and animal populations. In particular, HMMs are increasingly used to analyse capture-recapture data and estimate key population parameters (e.g., survival, dispersal, recruitment or abundance) with applications all fields of ecology. In parallel, Bayesian statistics is relatively well established and fast growing in ecology and related disciplines, because it resonates with scientific reasoning and allows accommodating uncertainty smoothly. The popularity of Bayesian statistics also comes from the availability of free pieces of software (WinBUGS, OpenBUGS, JAGS, Stan, nimble) that allow practitioners to code their own analyses.

However, to my knowledge, a full Bayesian treatment of HMMs applied to capture-recapture data is yet to be proposed in a book. This is what I propose with this book. Besides, the popular software solutions come with computational limitations when ecologists have to deal with complex models and/or big data. I will use Nimble that is seen by many as the future of ecological data modelling because it extends the BUGS language for writing new functions and distributions, and provides samplers that can deal with discrete latent states in contrast with Stan.

In this book, I will cover both the theory of HMMs for capture-recapture data, and the applications of these models to empower practitioners to fit their models with confidence. An important part of the book will consist in case studies presented in a tutorial style to abide by the "learning by doing" philosophy.

The online version of this book is licensed under the Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License[4].

## Why read this book

## Structure of the book

Blabla.

## Software information and conventions

This book uses primarily the R package **nimble**, so you need to at least install R and the **nimble** package.

The R session information when compiling this book is shown below:

```
sessionInfo()
## R version 4.1.0 (2021-05-18)
## Platform: x86_64-apple-darwin17.0 (64-bit)
## Running under: macOS Catalina 10.15.7
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRblas.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/4.1/Resources/lib/libRlapack.dylib
```

[4] http://creativecommons.org/licenses/by-nc-sa/4.0/

```
##
## locale:
## [1] fr_FR.UTF-8/fr_FR.UTF-8/fr_FR.UTF-8/C/C/fr_FR.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets
## [6] methods   base
##
## other attached packages:
##  [1] pdftools_3.0.1   magick_2.7.3     MCMCvis_0.15.3
##  [4] nimble_0.11.1    forcats_0.5.1    stringr_1.4.0
##  [7] dplyr_1.0.7      purrr_0.3.4.9000 readr_2.0.0
## [10] tidyr_1.1.3      tibble_3.1.3     ggplot2_3.3.5
## [13] tidyverse_1.3.1
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.7         lubridate_1.7.10
##  [3] lattice_0.20-44    assertthat_0.2.1
##  [5] digest_0.6.27      utf8_1.2.2
##  [7] R6_2.5.0           cellranger_1.1.0
##  [9] backports_1.2.1    reprex_2.0.1
## [11] evaluate_0.14      coda_0.19-4
## [13] httr_1.4.2         pillar_1.6.2
## [15] rlang_0.4.11       readxl_1.3.1
## [17] rstudioapi_0.13    rmarkdown_2.10
## [19] qpdf_1.1           igraph_1.2.6
## [21] munsell_0.5.0      broom_0.7.9
## [23] compiler_4.1.0     modelr_0.1.8
## [25] xfun_0.25          askpass_1.1
## [27] pkgconfig_2.0.3    htmltools_0.5.1.1
## [29] tidyselect_1.1.1   bookdown_0.23
## [31] fansi_0.5.0        crayon_1.4.1
## [33] tzdb_0.1.2         dbplyr_2.1.1
## [35] withr_2.4.2        grid_4.1.0
## [37] jsonlite_1.7.2     gtable_0.3.0
## [39] lifecycle_1.0.0    DBI_1.1.1
```

```
## [41] magrittr_2.0.1    scales_1.1.1
## [43] cli_3.0.1         stringi_1.7.3
## [45] fs_1.5.0          xml2_1.3.2
## [47] ellipsis_0.3.2    generics_0.1.0
## [49] vctrs_0.3.8       tools_4.1.0
## [51] glue_1.4.2        hms_1.1.0
## [53] parallel_4.1.0    yaml_2.2.1
## [55] colorspace_2.0-2  rvest_1.0.1
## [57] knitr_1.33        haven_2.4.3
```

We do not add prompts (> and +) to R source code in this book, and we comment out the text output with two hashes ## by default, as you can see from the R session information above. This is for your convenience when you want to copy and run the code (the text output will be ignored since it is commented out). Package names are in bold text (e.g., **nimble**), and inline code and filenames are formatted in a typewriter font (e.g., `knitr::knit('foo.Rmd')`). Function names are followed by parentheses (e.g., `nimble::nimbleCode()`). The double-colon operator :: means accessing an object from a package.

**Acknowledgments**

CNRS. Jean-Do. Roger. Rémi. My students. Chloé, Sarah, Perry, Daniel. Rob Chapman & Hall/CRC. Workshop attendees. Feedback from. FIP radio. Marc Kéry for his support and advice on how to write a book. Proofreading by. My family.

Olivier Gimenez Montpellier, France

# *About the Author*

Je m'appelle Olivier Gimenez (https://oliviergimenez.github.io/). Je suis directeur de recherche au CNRS. Après des études universitaires en mathématiques, j'ai fait une thèse en statistiques pour l'écologie. J'ai passé mon Habilitation à Diriger des Recherches en écologie et évolution. Récemment, je suis retourné sur les bancs de l'université pour m'initier à la sociologie.

J'ai écrit des articles scientifiques[5] faisant appel à la statistique bayésienne, et co-écrit avec des collègues britanniques un livre sur les analyses bayésiennes pour l'écologie des populations[6].

Vous pouvez me retrouver sur Twitter (https://twitter.com/oaggimenez), ou bien me contacter via mon adresse email qui s'écrit olivier suivi d'un point puis gimenez, ensuite arobase, puis cefe, suivi d'un point, puis cnrs, suivi d'un point et pour terminer fr.

Tombé dedans quand j'étais petit. Obélix Roger et Astérix JD.

[5] https://oliviergimenez.github.io/publication/papers/
[6] https://oliviergimenez.github.io/publication/books/

# Part I

# Theory

# 1

# *Bayesian statistics & MCMC*

## 1.1 Bayes' theorem

A theorem about conditional probabilities.

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

I always forget what the letters mean.

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.

Cool because what is doing science after all?

How plausible is some hypothesis given the data?

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

The Bayesian reasoning echoes the scientific reasoning. You might ask then, why is Bayesian statistics not the default?

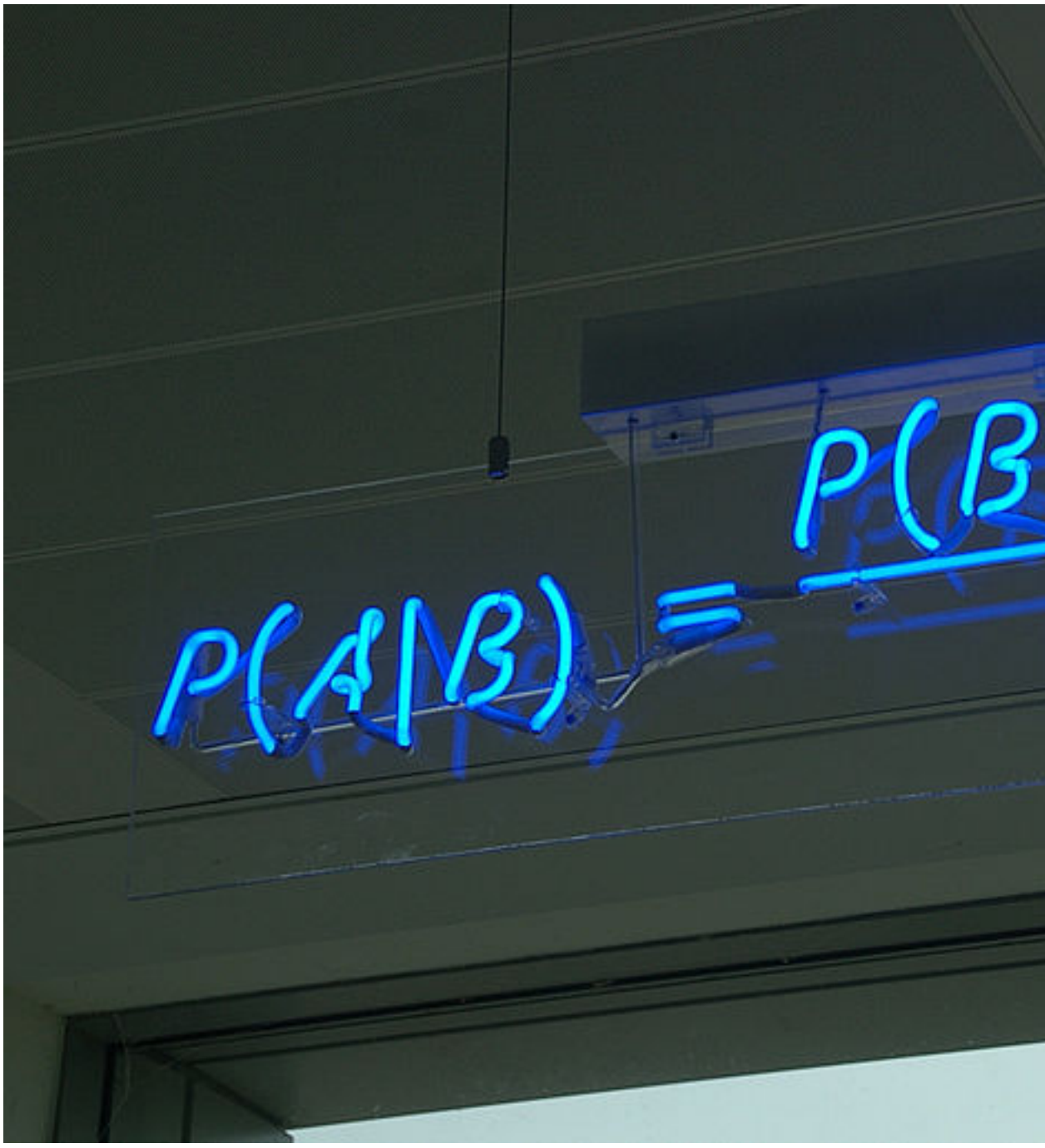



**FIGURE 1.1:** Bayes' theorem spelt out in blue neon at the offices of Autonomy in Cambridge. Source: Wikipedia

You may ask: Why is Bayesian statistics not the default?

Due to practical problems of implementing the Bayesian approach, and futile wars between (male) statisticians, little progress 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.

## 1.2 Frequentist versus Bayesian

Typical stats problems involve estimating parameter $\theta$ 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 are generally point estimates of the parameters of interest.

The Bayesian approach assumes that the parameters are not fixed but have some fixed unknown distribution - a distribution for the parameter.

## 1.3 What is the Bayesian approach?

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

You never start from scratch.

And then updates these beliefs on the basis of observed data.

This updating procedure is based upon the Bayes' Theorem:

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

Schematically if $A = \theta$ and $B =$ data, then

The Bayes' theorem

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

Translates into:

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

## 1.4 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.

Likelihood: This quantity is the 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.

$\Pr(\text{data}) = \int L(\text{data} \mid \theta)\ \Pr(\theta) d\theta$ is a $N$-dimensional integral if $\theta = \theta_1, \ldots, \theta_N$.

Difficult if not impossible to calculate. This is one of the reasons why we need simulation (MCMC) methods.

## 1.5 Brute force via numerical integration

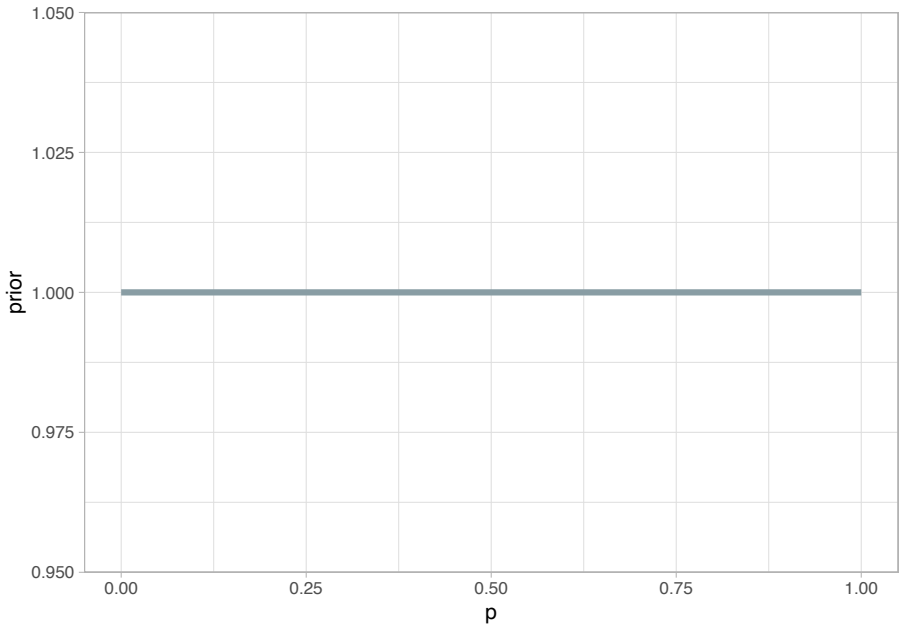
Say we release $n$ animals at the beginning of the winter, out of which $y$ survive, and we'd like to estimate winter survival $\theta$.

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

Our model:

$$
\begin{aligned}
y &\sim \text{Binomial}(n, \theta) && \text{[likelihood]} \\
\theta &\sim \text{Beta}(1, 1) && \text{[prior for } \theta\text{]}
\end{aligned}
$$

## 1.6 Beta prior




## 1.7 Apply Bayes theorem

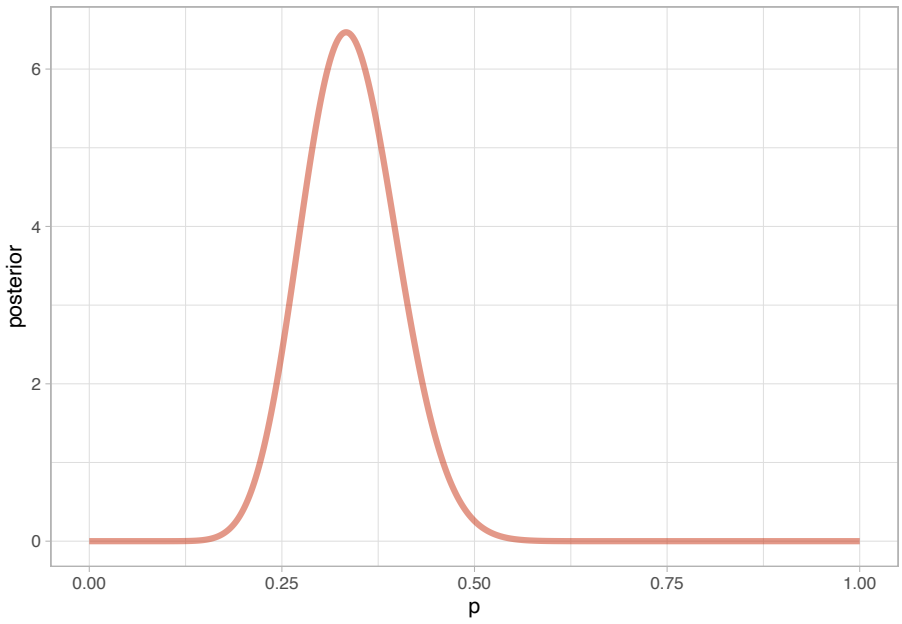
Likelihood times the prior: $\Pr(\text{data} \mid \theta) \ \Pr(\theta)$

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

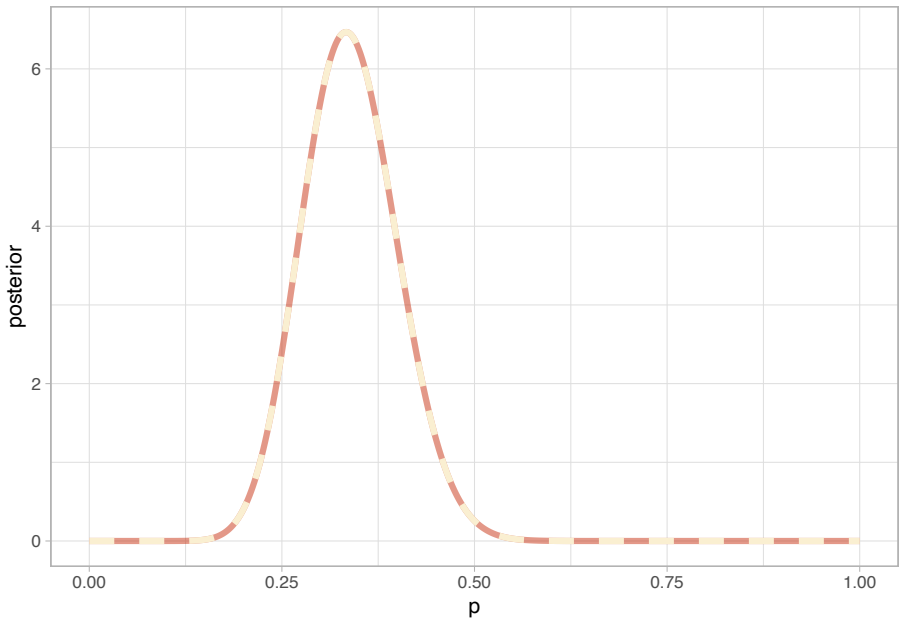
Averaged likelihood: $\Pr(\text{data}) = \int L(\theta \mid \text{data}) \ \Pr(\theta) d\theta$

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

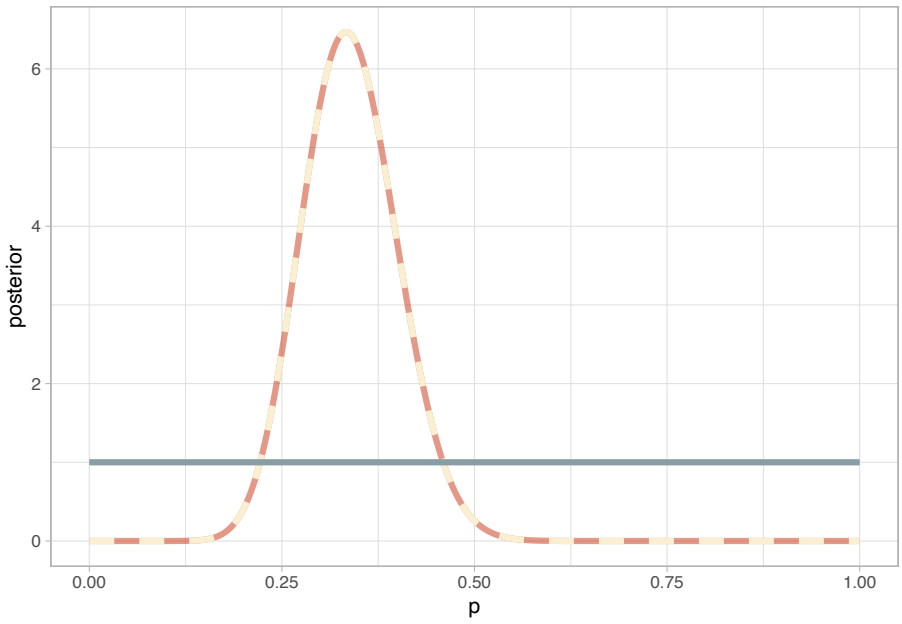
## 1.8 Posterior via numerical integration



## 1.9 Superimpose explicit posterior



## 1.10 And the prior




## 1.11 What if multiple parameters?

Example of a linear regression with parameters $\alpha$, $\beta$ and $\sigma$ to be estimated.

Bayes' theorem says:

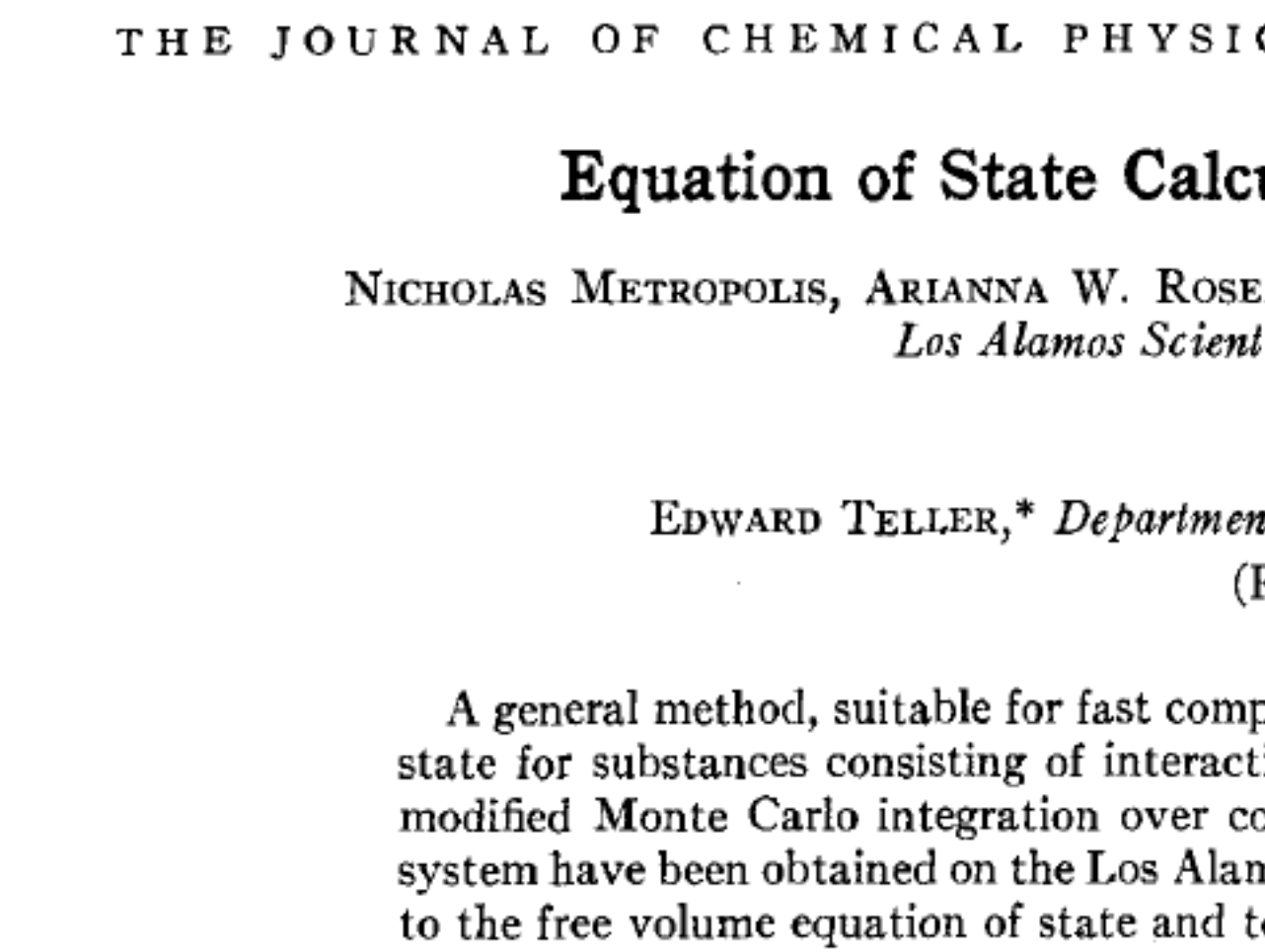
$$P(\alpha, \beta, \sigma \mid \text{data}) = \frac{P(\text{data} \mid \alpha, \beta, \sigma)\, P(\alpha, \beta, \sigma)}{\iiint P(\text{data} \mid \alpha, \beta, \sigma)\, P(\alpha, \beta, \sigma)\, d\alpha\, d\beta\, d\sigma}$$

Do we really wish to calculate a 3D integral?

### 1.12 Bayesian computation

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

```
knitr::include_graphics("images/metropolis.png")
```



THE JOURNAL OF CHEMICAL PHYSIC

Equation of State Calcu

NICHOLAS METROPOLIS, ARIANNA W. ROSE
*Los Alamos Scient*

EDWARD TELLER,* *Departmen*
(I

A general method, suitable for fast comp
state for substances consisting of interacti
modified Monte Carlo integration over co
system have been obtained on the Los Alam
to the free volume equation of state and t

Use stochastic simulation to draw samples from posterior distributions.

Avoid explicit calculation of integrals in Bayes formula.

Instead, approx. posterior w/ some precision by drawing large samples.

Markov chain Monte Carlo (MCMC) gives a boost to Bayesian statistics!

## 1.13 Why are MCMC methods so useful?

- MCMC are stochastic algorithms to produce sequence of dependent random numbers from a Markov chain.
- A Markov chain is a discrete sequence of states, in which the probability of an event depends only on the state in the previous event.
- A Markov chain has an equilibrium (aka stationary) distribution.
- Equilibrium distribution is the desired posterior distribution!
- Several ways of constructing these chains: e.g., Metropolis-Hastings, Gibbs sampler.
- How to implement them in practice?!

## 1.14 The Metropolis algorithm

- Let's go back to animal survival estimation.
- We illustrate sampling from the posterior distribution.
- We write functions in R for the likelihood, the prior and the posterior.

```r
# survival data, 19 "success" out of 57 "attempts"
survived <- 19
released <- 57

# log-likelihood function
loglikelihood <- function(x, p){
  dbinom(x = x, size = released, prob = p, log = TRUE)
}

# prior density
logprior <- function(p){
  dunif(x = p, min = 0, max = 1, log = TRUE)
}

# posterior density function (log scale)
posterior <- function(x, p){
  loglikelihood(x, p) + logprior(p) # - log(Pr(data))
}
```

## 1.15 Metropolis algorithm

1. We start at any possible value of the parameter to be estimated.
2. To decide where to visit next, we propose to move away from the current value of the parameter — this is a **candidate** value. To do so, we add to the current value some random value from say a normal distribution with some variance.
3. We compute the ratio of the probabilities at the candidate and current locations $R$ = posterior(candidate)/posterior(current). This is where the magic of MCMC happens, in that Pr(data), the denominator of the Bayes theorem, cancels out.

4. We spin a continuous spinner that lands anywhere from 0 to 1 — call it the random spin $X$. If $X$ is smaller than $R$, we move to the candidate location, otherwise we remain at the current location.
5. We repeat 2-4 a number of times — or **steps** (many steps).

```
# propose candidate value
move <- function(x, away = .2){
  logitx <- log(x / (1 - x))
  logit_candidate <- logitx + rnorm(1, 0, away)
  candidate <- plogis(logit_candidate)
  return(candidate)
}

# set up the scene
steps <- 100
theta.post <- rep(NA, steps)
set.seed(1234)

# pick starting value (step 1)
inits <- 0.5
theta.post[1] <- inits
```

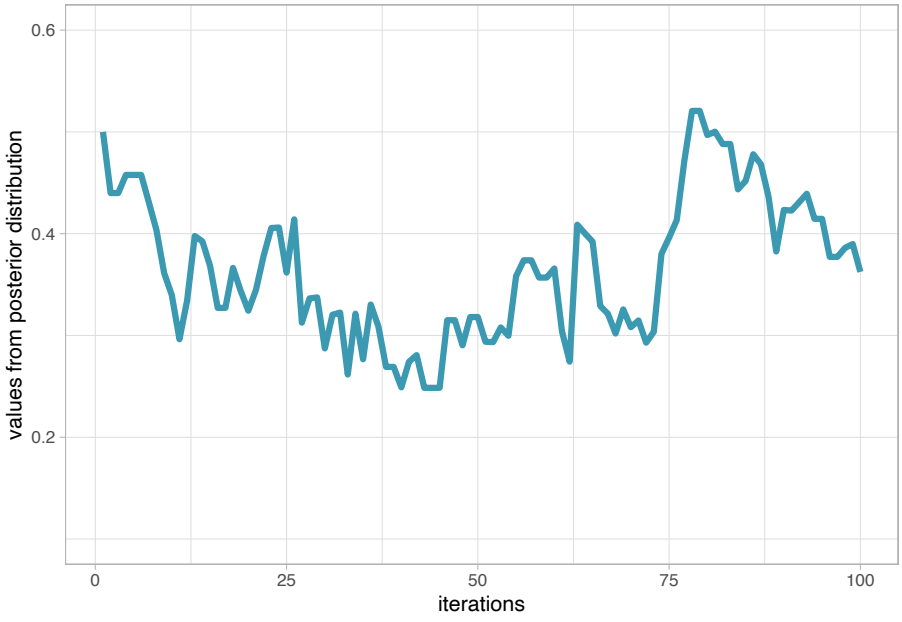
```
for (t in 2:steps){ # repeat steps 2-4 (step 5)

  # propose candidate value for prob of success (step 2)
  theta_star <- move(theta.post[t-1])

  # calculate ratio R (step 3)
  pstar <- posterior(survived, p = theta_star)
  pprev <- posterior(survived, p = theta.post[t-1])
  logR <- pstar - pprev
  R <- exp(logR)
```

```
  # decide to accept candidate value or to keep current value (step 4)
  accept <- rbinom(1, 1, prob = min(R, 1))
  theta.post[t] <- ifelse(accept == 1, theta_star, theta.post[t-1])
}
```

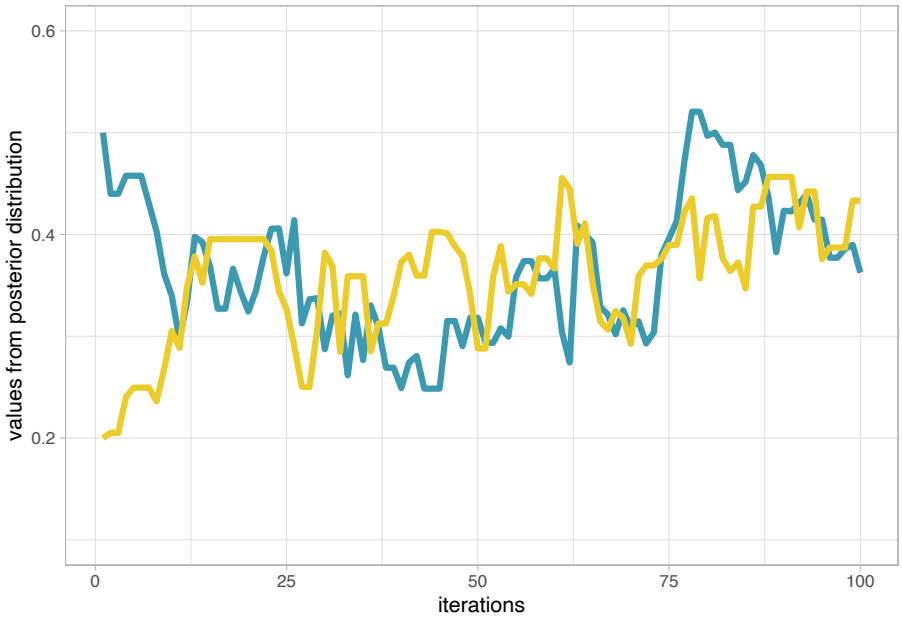
Starting at the value $0.5$ and running the algorithm for $100$ iterations.

```
head(theta.post)
## [1] 0.5000 0.4399 0.4399 0.4577 0.4577 0.4577
tail(theta.post)
## [1] 0.4146 0.3772 0.3772 0.3861 0.3899 0.3624
```

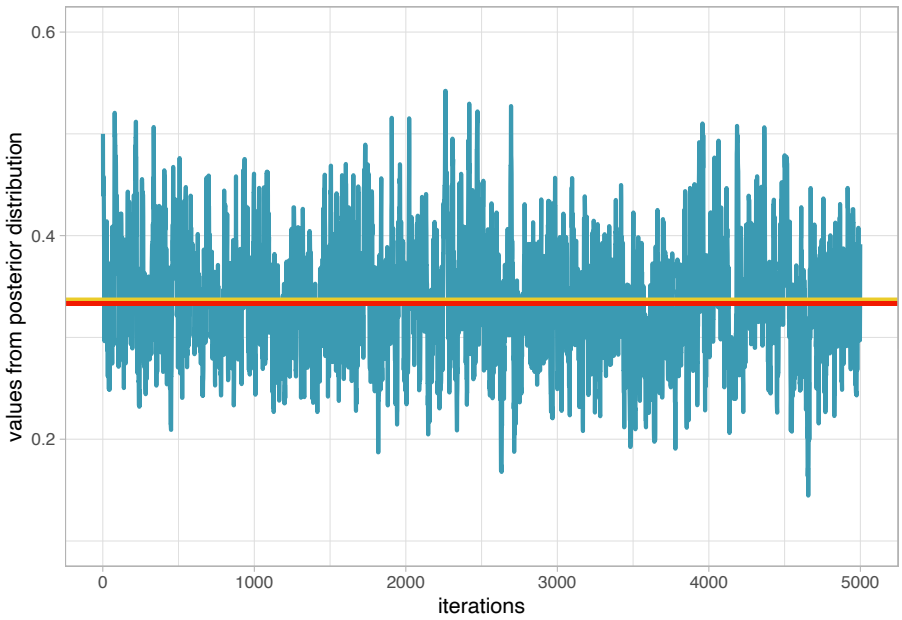
## 1.16 A chain



## 1.17 Another chain



## 1.18 With 5000 steps

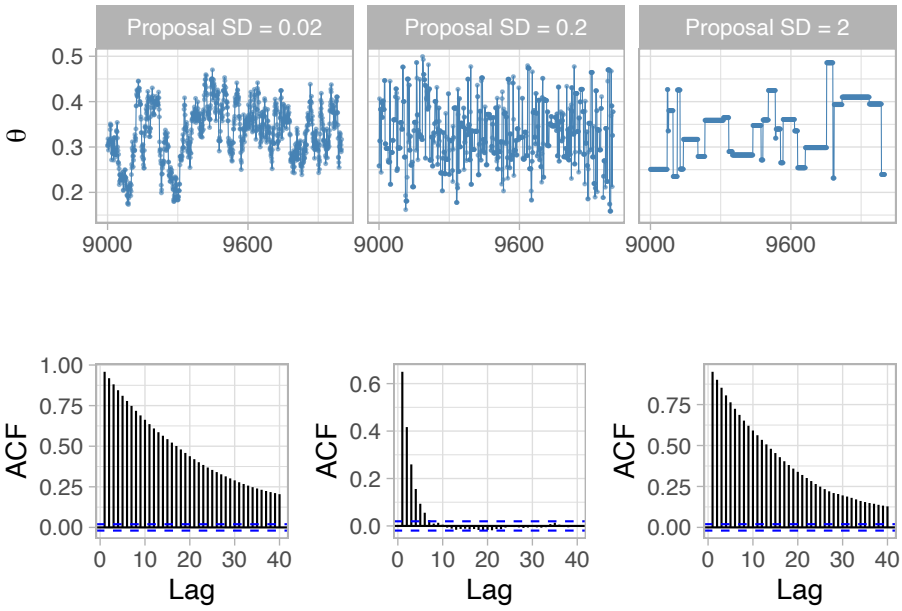



In yellow: posterior mean; in red: maximum likelihood estimate.

## 1.19 Assessing convergence

- 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.

## 1.20 Mixing and autocorrelation

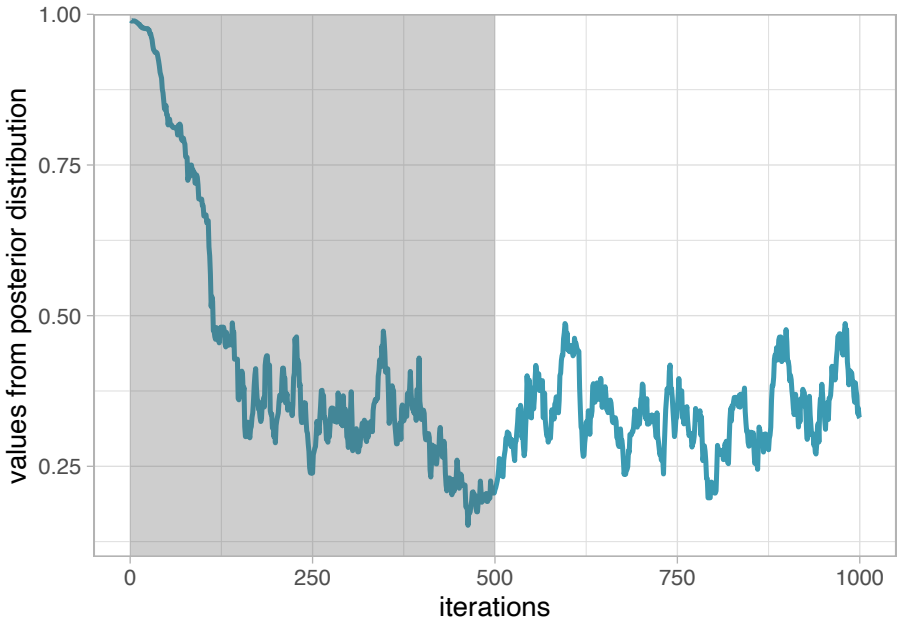



- The movement around the parameter space is often referred to as **mixing**.
- Traceplots of for small and big moves provide (relatively) high correlations (known as autocorrelations) between successive observations of the Markov chain.
- Strongly correlated observations require large sample sizes and 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. cor$(\theta_t, \theta_{t+k})$) for increasing values of $k$.

### 1.21 How do good chains behave?

- Converge to same target distribution; discard some realisations of Markov chain before convergence is achieved.
- Once there, explore efficiently: The post-convergence sample size required for suitable numerical summaries.
- Therefore, 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**.
- Simplest method to determine length of burn-in period is to look at trace plots.

## 1.22 Burn-in




If simulations cheap, be conservative.

## 1.23 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.

### 1.24 Potential scale reduction factor

- Gelman-Rubin statistic $\widehat{R}$
- 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 $\leq 1.1$ 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.

### 1.25 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.
- Proceed with posterior inference.
- Use traceplot, effective sample size and $\widehat{R}$.

### 1.26 What if you have issues of convergence?

- Increase burn-in, sample more.
- Use more informative priors.
- Pick better initial values (good guess), using e.g. estimates from simpler models.
- 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. More later on.

## 1.27 Further reading

- McCarthy, M. (2007). Bayesian Methods for Ecology[1]. Cambridge: Cambridge University Press.
- McElreath, R. (2020). Statistical Rethinking: A Bayesian Course with Examples in R and Stan (2nd ed.)[2]. CRC Press.
- Gelman, A. and Hill, J. (2006). Data Analysis Using Regression and Multilevel/Hierarchical Models (Analytical Methods for Social Research)[3]. Cambridge: Cambridge University Press.

[1] https://www.cambridge.org/core/books/bayesian-methods-for-ecology/9225F65B8A25D69B0B6C50B5A9A78201

[2] https://xcelab.net/rm/statistical-rethinking/

[3] https://www.cambridge.org/core/books/data-analysis-using-regression-and-multilevelhierarchical-models/32A29531C7FD730C3A68951A17C9D983

# 2

# *Introduction to Nimble*

## 2.1 What is Nimble?


**FIGURE 2.1:** (Meme created by Todd Arnold's wonderful students)

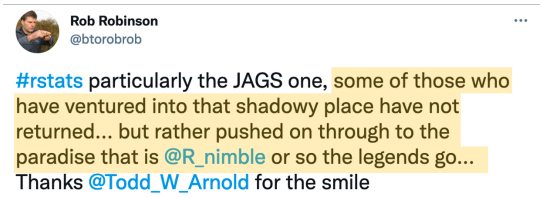



**FIGURE 2.2:** (Meme created by Todd Arnold's wonderful students)

- **N**umerical **I**nference for statistical **M**odels using **B**ayesian and **L**ikelihood **E**stimation.
- A framework for hierarchical statistical models and algorithms.
- Uses almost the same model code as WinBUGS, OpenBUGS, and JAGS.
- An extension of the BUGS language: additional syntax, custom functions and distributions.
- A configurable system for MCMC.
- A library of other methods (SMC, MCEM).
- Sequential Monte Carlo (particle filtering)
- Monte Carlo Expectation Maximization (maximum likelihood)
- A model-generic programming system to write new analysis methods.

## 2.2 Load `nimble` package

```
library(nimble)
```

## 2.3 Build model, made of likelihood and priors

```
naive.survival.model <- nimbleCode({
  # prior
  phi ~ dunif(0, 1)
  # likelihood
  y ~ dbinom(phi, n)
})
```

## 2.4 Syntax: what's new/better/different?

- Vectorization

```
# JAGS (& Nimble)
for(t in 1:Tmax){
  x[t] <- Mu.x + epsilon[t]
}

# Nimble
x[1:Tmax] <- Mu.x + epsilon[1:Tmax]
```

- More flexible specification of distributions

```
# JAGS (& Nimble)
for(t in 1:Tmax){
  epsilon[t] ~ dnorm(0, tau)
}
tau <- pow(sigma, -2)
sigma ~ dunif(0, 5)
```

```
# Nimble
for(t in 1:Tmax){
  epsilon[t] ~ dnorm(0, sd = sigma)
}
sigma ~ dunif(0, 5)
```

- Your own functions and distributions

```
x[1:Tmax] <- myNimbleFunction(a = Mu.x, b = epsilon[1:Tmax])
```

```
sigma ~ dCustomDistr(c = 0.5, z = 10)
```

- The end of empty indices

```
# JAGS
sum.x <- sum(x[])

# Nimble
sum.x <- sum(x[1:Tmax])
```

- & more...

## 2.5 Read in data

Back to our naive survival model:

```
naive.survival.model <- nimbleCode({
  # prior
  phi ~ dunif(0, 1)
  # likelihood
  y ~ dbinom(phi, n)
})
```

```
my.data <- list(n = 57, y = 19)
```

## 2.6 Distinguish constants and data

To Nimble, not all "data" is data...

```
my.constants <- list(n = 57)
my.data <- list(y = 19)
```

**Constants**: + Can never be changed + Must be provided when a model is defined (part of the model structure) + E.g. vector of known index values, variables used to define for-loops, etc.

To Nimble, not all "data" is data...

```
my.constants <- list(n = 57)
my.data <- list(y = 19)
```

**Data**: + Can be changed without re-building the model + Can be (re-)simulated within a model + E.g. stuff that *only* appears to the left of a "~"

For computational efficiency, better to specify as much as possible as constants.

Nimble will help you with this!

## 2.7 Specify initial values

```
initial.values <- function() list(phi = runif(1,0,1))
```

```
initial.values()
## $phi
## [1] 0.5332
```

## 2.8 Which parameters to save?

```
parameters.to.save <- c("phi")
```

## 2.9 MCMC details

```
n.iter <- 5000
n.burnin <- 1000
n.chains <- 2
n.thin <- 1
```

Number of posterior samples per chain:

$$n.posterior = \frac{n.iter - n.burnin}{n.thin}$$

## 2.10 Run model, tadaa!

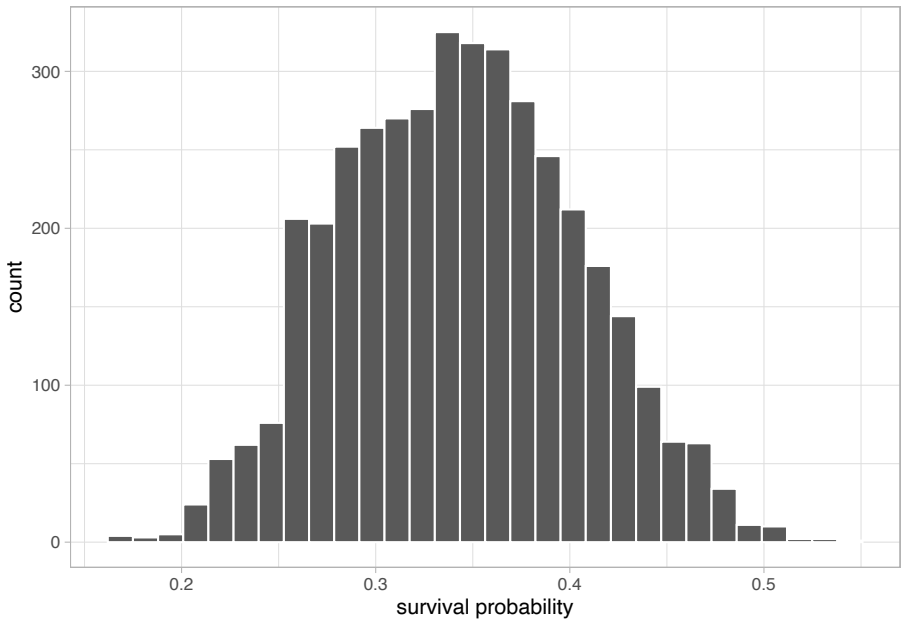
```
mcmc.output <- nimbleMCMC(code = naive.survival.model,
                          data = my.data,
                          constants = my.constants,
                          inits = initial.values,
                          monitors = parameters.to.save,
                          thin = n.thin,
                          niter = n.iter,
                          nburnin = n.burnin,
                          nchains = n.chains)
```

## 2.11 Explore MCMC outputs

```
str(mcmc.output)
## List of 2
```

```
##  $ chain1: num [1:4000, 1] 0.305 0.305 0.392 0.422 0.422 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr "phi"
##  $ chain2: num [1:4000, 1] 0.352 0.416 0.416 0.416 0.416 ...
##   ..- attr(*, "dimnames")=List of 2
##   .. ..$ : NULL
##   .. ..$ : chr "phi"
```

```
head(mcmc.output$chain1)
##         phi
## [1,] 0.3046
## [2,] 0.3046
## [3,] 0.3916
## [4,] 0.4221
## [5,] 0.4221
## [6,] 0.4449
```



## 2.12 Numerical summaries

```
library(MCMCvis)
MCMCsummary(mcmc.output, round = 2)
##     mean   sd 2.5%  50% 97.5% Rhat n.eff
## phi 0.34 0.06 0.23 0.34  0.46    1  1931
```

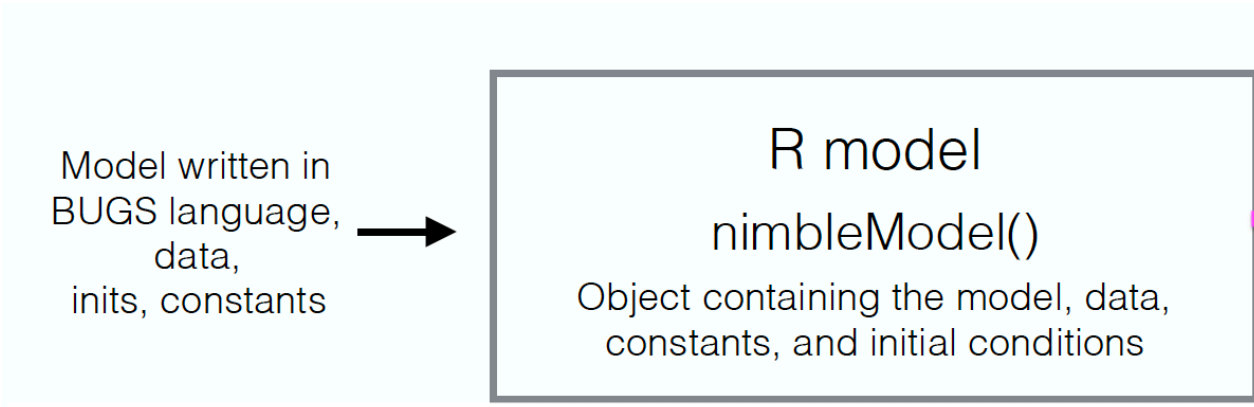
## 2.13 Trace and posterior density

```
MCMCtrace(mcmc.output,
          pdf = FALSE)
```

```
MCMCtrace(mcmc.output,
          pdf = FALSE,
          ind = TRUE,
          Rhat = TRUE,
          n.eff = TRUE)
```

## 2.14 Our `nimble` workflow so far

```
knitr::include_graphics("images/nimble_workflow_sofar.png")
```




Adapted from L. Ponisio

## 2.15 Useful resources

- Official website https://r-nimble.org
- User Manual https://r-nimble.org/html_manual/cha-welcome-nimble.html and cheatsheet[1].
- Users mailing list https://groups.google.com/forum/#!forum/nimble-users
- Training material https://github.com/nimble-training
- Reference to cite when using nimble in a publication:

> de Valpine, P., D. Turek, C. J. Paciorek, C. Anderson-Bergman, D. Temple Lang, and R. Bodik (2017). Programming With Models: Writing Statistical Algorithms for General Model Structures With NIMBLE[2]. *Journal of Computational and Graphical Statistics* **26** (2): 403–13.

[1] https://r-nimble.org/cheatsheets/NimbleCheatSheet.pdf
[2] https://arxiv.org/pdf/1505.05093.pdf

# 3

# *Hidden Markov models*

## 3.1 Back to our survival example

- We have $z$ survivors out of $n$ released animals with winter survival probability $\phi$
- Let's get back to our survival example.
- Our model so far:

$$\begin{aligned} z &\sim \text{Binomial}(n, \phi) && \text{[likelihood]} \\ \phi &\sim \text{Beta}(1, 1) && \text{[prior for } \phi\text{]} \end{aligned}$$

- Our model so far has been a combination
- Of a binomial likelihood
- And a Beta prior with param 1 and 1, which is a uniform between 0 and 1.
- This is also:

$$\begin{aligned} z_i &\sim \text{Bernoulli}(\phi),\ i = 1, \dots, N && \text{[likelihood]} \\ \phi &\sim \text{Beta}(1, 1) && \text{[prior for } \phi\text{]} \end{aligned}$$

- The binomial is just a sum of Bernoulli outcomes

- Like flipping a coin for each individual and get a survivor with prob phi.
- What if we had several winters? Say $T = 5$ winters.
- In this design, we have a single winter. But for many species, we'll need to collect data on the long term to get a representative estimate of survival.
- Therefore what if we had say big T five winters?

## 3.2 Longitudinal data

- $z_{i,t} = 1$ if individual $i$ alive at winter $t$, and $z_{i,t} = 2$ if dead.

| id | winter 1 | winter 2 | winter 3 | winter 4 | winter 5 |
|---|---|---|---|---|---|
| 1 | 1 | 2 | 2 | 2 | 2 |
| 2 | 1 | 1 | 1 | 1 | 1 |
| 3 | 1 | 2 | 2 | 2 | 2 |
| 4 | 1 | 1 | 1 | 1 | 1 |
| 5 | 1 | 2 | 2 | 2 | 2 |
| 6 | 1 | 1 | 1 | 1 | 1 |
| 7 | 1 | 1 | 1 | 1 | 2 |
| 8 | 1 | 1 | 1 | 1 | 1 |
| 9 | 1 | 1 | 1 | 1 | 1 |
| 10 | 1 | 1 | 2 | 2 | 2 |
| 11 | 1 | 1 | 1 | 1 | 1 |
| 12 | 1 | 1 | 1 | 1 | 1 |
| 13 | 1 | 1 | 2 | 2 | 2 |
| 14 | 1 | 1 | 1 | 1 | 1 |
| 15 | 1 | 1 | 1 | 1 | 1 |
| 16 | 1 | 1 | 1 | 1 | 1 |
| 17 | 1 | 1 | 2 | 2 | 2 |
| 18 | 1 | 1 | 1 | 1 | 1 |
| 19 | 1 | 1 | 1 | 2 | 2 |
| 20 | 1 | 1 | 1 | 1 | 1 |
| 21 | 1 | 2 | 2 | 2 | 2 |
| 22 | 1 | 1 | 2 | 2 | 2 |
| 23 | 1 | 1 | 2 | 2 | 2 |
| 24 | 1 | 1 | 1 | 1 | 2 |
| 25 | 1 | 1 | 1 | 1 | 1 |
| 26 | 1 | 1 | 1 | 1 | 2 |
| 27 | 1 | 1 | 1 | 1 | 1 |
| 28 | 1 | 2 | 2 | 2 | 2 |
| 29 | 1 | 1 | 2 | 2 | 2 |
| 30 | 1 | 1 | 1 | 1 | 1 |
| 31 | 1 | 1 | 2 | 2 | 2 |
| 32 | 1 | 1 | 1 | 2 | 2 |
| 33 | 1 | 1 | 1 | 1 | 1 |
| 34 | 1 | 1 | 1 | 2 | 2 |
| 35 | 1 | 1 | 1 | 2 | 2 |
| 36 | 1 | 1 | 2 | 2 | 2 |
| 37 | 1 | 2 | 2 | 2 | 2 |
| 38 | 1 | 1 | 2 | 2 | 2 |
| 39 | 1 | 1 | 1 | 1 | 1 |
| 40 | 1 | 1 | 2 | 2 | 2 |
| 41 | 1 | 2 | 2 | 2 | 2 |
| 42 | 1 | 1 | 1 | 1 | 2 |
| 43 | 1 | 1 | 2 | 2 | 2 |

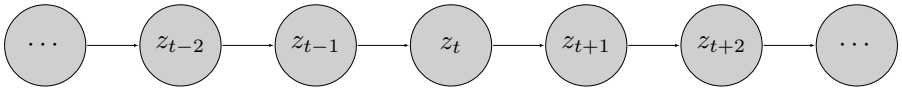
- This is what we call longitudinal data.
- Each row is an individual i, and columns are for winters t, or sampling occasions.
- z is indexed by both i and t, and takes value 1 if ind i is alive in winter t, and 2 otherwise.

## 3.3 A model for longitudinal survival data

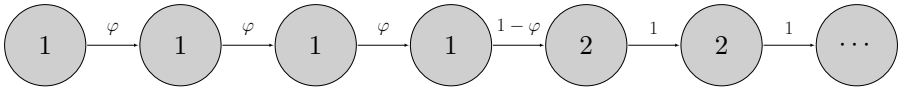
- A model relies on assumptions.
- Let's think of a model for these data.
- The objective remains the same, estimating survival.
- To build this model, we'll make assumptions.
- The state of an animal at a given winter, alive or dead, is only dependent on its state the winter before.
- First, we assume that the state of an animal in a given winter, alive or dead, is only dependent on its state the winter before.
- The future depends only on the present, not the past: Markov process.
- In others words, he future depends only on the present, not the past
- This is a Markov process.
- If an animal is alive in a given winter, the probability it survives to the next winter is $\phi$.
- If an animal is alive in a given winter, the probability it survives to the next winter is $\phi$.
- The probability it dies is $1 - \phi$.
- The probability it dies is $1 - \phi$.

- If an animal is dead a winter, it remains dead, unless you believe in zombies.
- If an animal is dead a winter, it remains dead, unless you believe in zombies.

## 3.4 Markov process




- A markov process can be represented this way.
- The state at t+1 only depends on the state at t.




- In our model, going from a winter to the next is driven by survival and mortality processes.
- The probability of going from alive or 1 to alive or 1 is phi.
- Then from alive 1 to dead 2 is 1 - phi.
- And the probability to remain dead is 1, that is to go from state 2 dead to state 2 for dead.

## 3.5 Transition matrix

- The core of the Markov process is made of the transition probabilities.
- The engine of a Markov model is the transition matrix.