Markov chain Monte Carlo

T.J. McKinley (t.mckinley@exeter.ac.uk)

Fitting Bayesian models

Now we return to the problem of how to fit Bayesian models. Recall that we only know the posterior distribution up to a normalising constant:

$$f(\theta \mid \mathbf{y}) \propto f(\mathbf{y} \mid \theta) f(\theta)$$

This means that we cannot write down an analytical form for the **posterior** distribution, so instead we need to appeal to **numerical** methods.

Fitting Bayesian models

VACCINE IMPACT MODELLING CONSORTIUM

It turns out that we can generate **empirical estimates** of probability distributions by **random sampling**.

That is, we take *large numbers of random samples* from a distribution, and then use these to estimate key aspects of the distribution, such as the mean, variance, quantiles, shape etc.

The general concept of estimating distributions (and key summary measures) through random sampling is known generally as **Monte Carlo** estimation[†].

[†] though Monte Carlo methods can be applied to a wider variety of problems than simply those explored here—for example Monte Carlo integration

Example: expected values

Consider a **random variable** X, that comes from a probability distribution with **probability density function** $f(x)^{\dagger}$.

The **expected value** of X is defined as:

$$E(X) = \int_{\mathcal{X}} x f(x) dx,$$

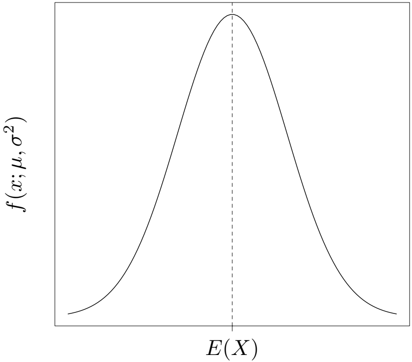
where the integral is over all possible values of X (denoted $\mathcal{X}$).

The **expected value**, $E(X)$, is the formal definition of the **mean** of a probability distribution.

[†] analogous formulations exist for **discrete** X, with probability *density* functions replaced by a probability *mass* functions, and *integration* replaced with *summation*

Aside: expected values

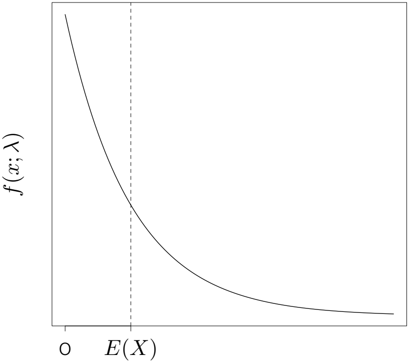
$$
\begin{aligned}
E(X) &= \int_{\mathcal{X}} x f(x) dx \\
&= \int_{-\infty}^{\infty} x \cdot \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{1}{2}\left(\frac{x-\mu}{\sigma}\right)^2} dx \\
&= \mu.
\end{aligned}
$$



One can think about the expected value as being a **weighted average**, where each possible value of x is ***weighted*** by the **probability density function**, $f(x)$, evaluated at x.

Aside: expected values

$$
\begin{aligned}
E(X) &= \int_{\mathcal{X}} x f(x) dx \\
&= \int_0^\infty x \cdot \lambda e^{-\lambda x} dx \\
&= \frac{1}{\lambda}.
\end{aligned}
$$



One can think about the expected value as being a **weighted average**, where each possible value of x is ***weighted*** by the **probability density function**, $f(x)$, evaluated at x.

Aside: expected values

Recall: the **sample mean**:

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i,$$

where the x_i are considered **random samples** from a distribution with a **probability density function** $f(x)^{\dagger}$.

It can be shown that as $n \to \infty$, then $\bar{x} \to E(X)$.

> *The **sample mean** $\bar{x}$ is an **unbiased** and **consistent** estimator of $E(X)$.*

†in a slight abuse of notation, we will denote this as $x_i \sim f(x)$

Monte Carlo estimation

Therefore, as long as you can produce **random samples**, $x_i \sim f(x)$, from a distribution, then you can ***estimate*** the **mean** of that distribution ($\mu = E(X)$) as:

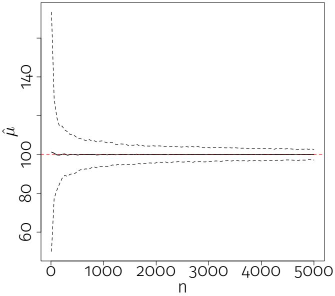
$$\hat{\mu} = \frac{1}{n} \sum_{i=1}^{n} x_i,$$

regardless of the distribution of X.

This is the essence of **Monte Carlo** estimation.

Example: exponential distribution

Assume that we can sample from an **exponential** distribution with mean $\mu = 100$. Take different numbers of samples from $n = 5, \dots, 5000$ and calculate the sample mean each time. Repeat this 1,000 times for each n[†] and plot the mean and 95% quantiles:



Notice:

- always **unbiased**;
- small $n \rightarrow$ large variance;
- large $n \rightarrow$ small variance.

[†] a Monte Carlo estimate of a Monte Carlo estimate, if you like!

The Law of the Unconscious Statistician

In general, if X is a random variable defined on some range of values $\mathcal{X}$, with probability density function $f(x)$, and $g(x)$ is some function of X, then

$$E\left[g(X)\right] = \int_{\mathcal{X}} g(x) f(x) dx.$$

This is known as the **Law of the Unconscious Statistician**[†].

Therefore we can generate Monte Carlo estimates of **general expectations**:

$$\begin{aligned} E\left[g(X)\right] &= \int_{\mathcal{X}} g(x) f(x) dx \\ &\approx \frac{1}{n} \sum_{i=1}^{n} g(x_i), \end{aligned}$$

where $x_i \sim f(x_i)$.

[†] $E(X)$ is a special case where $g(X) = X$

Estimating posterior expectations

Let's think about calculating the **posterior mean** for a parameter θ.

$$f(\theta \mid \mathbf{y}) \propto f(\mathbf{y} \mid \theta) f(\theta).$$

The posterior mean is:

$$E(\theta) = \int_{\Theta} \theta f(\theta \mid \mathbf{y}) d\theta,$$

where Θ corresponds to all possible values of θ. Hence we can approximate $E(\theta)$ as:

$$E(\theta) \approx \frac{1}{n} \sum_{i=1}^{n} \theta_i,$$

where θ_i are samples from the **posterior**, $f(\theta \mid \mathbf{y})$—**as long as we can sample from the posterior**!

Estimating posterior expectations

So, we wish to estimate the **posterior mean** for a parameter θ

$$E(\theta) = \int_\Theta \theta f(\theta \mid \mathbf{y}) d\theta \approx \frac{1}{n} \sum_{i=1}^{n} \theta_i,$$

where $\theta_i \sim f(\theta \mid \mathbf{y})$.

The problem is that we only know the **posterior** up to some normalising constant i.e.

$$f(\theta \mid \mathbf{y}) = \frac{f(\mathbf{y} \mid \theta) f(\theta)}{f(\mathbf{y})},$$

where $f(\mathbf{y})$ is **unknown**. Thus simple methods of sampling, such as inverse transform sampling, do not work.

Instead, we will use a technique called **Markov chain Monte Carlo**.

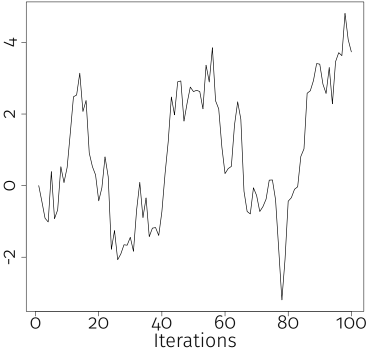
Markov chains

A Markov chain is simply a **sequence** of numbers, where each number in the sequence relates to the ***previous*** number[†]

For example, a simple random-walk:

$$x_i \sim N(x_{i-1}, 1)$$

for $i = 1, \ldots, n$ and $x_0 = 1$ (say).



[†] we will only deal with **first-order** Markov chains here

MCMC

A simple **random-walk** like the one shown previously will just iterate around at random.

However, it is possible to generate Markov chains that **converge** to some distribution, known as the **stationary distribution**.

There are lots of variations of **MCMC**, but the one we will explore here is known as the **Metropolis-Hastings** algorithm† (Metropolis et al. 1953; Hastings 1970).

†technically the **random-walk Metropolis-Hastings** algorithm

Random-walk Metropolis-Hastings

```
**Require:** $\theta^{(0)}$.
  **for** $i = 1, \dots, n$ **do**
    Propose **candidate** $\theta' \sim q\left(\cdot \mid \theta^{(i-1)}\right)$.
    Calculate the **acceptance probability**:

    $$\alpha = \min\left(1, \frac{f\left(\theta'\right) q\left(\theta^{(i-1)} \mid \theta'\right)}{f\left(\theta^{(i-1)}\right) q\left(\theta' \mid \theta^{(i-1)}\right)}\right)$$

    Sample $u \sim U(0,1)$
    **if** $u < \alpha$ **then**
      $\theta^{(i)} = \theta'$
    **else**
      $\theta^{(i)} = \theta^{(i-1)}$
    **end if**
  **end for**
```

$f(\cdot)$ is the **target** distribution.

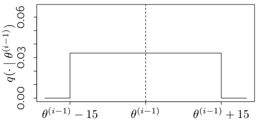
$q\left(\cdot \mid \theta^{(i-1)}\right)$ is a **proposal** distribution.

Example: normal distribution

Let's sample from a **normal distribution** with mean 10 and standard deviation 5, with

$$f(\theta) = \frac{1}{10\sqrt{\pi}} e^{-\frac{1}{2}\left(\frac{\theta-10}{5}\right)^2}$$

Let's start at $\theta^{(0)} = 10$ and use a **uniform** proposal distribution $\theta' \sim U(\theta^{(i-1)} - 15, \theta^{(i-1)} + 15)$.



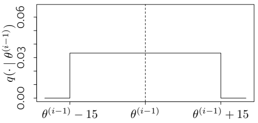
```
Require: $\theta^{(0)}$.
  for $i = 1, \ldots, n$ do
      Propose candidate $\theta' \sim q\left(\cdot \mid \theta^{(i-1)}\right)$.
      Calculate the acceptance probability:
          $\alpha = \min\left(1, \frac{f\left(\theta'\right) q\left(\theta^{(i-1)} \mid \theta'\right)}{f\left(\theta^{(i-1)}\right) q\left(\theta' \mid \theta^{(i-1)}\right)}\right)$
      Sample $u \sim U(0, 1)$
      if $u < \alpha$ then
          $\theta^{(i)} = \theta'$
      else
          $\theta^{(i)} = \theta^{(i-1)}$
      end if
  end for
```

Example: normal distribution

Let's sample from a **normal distribution** with mean 10 and standard deviation 5, with

$$f\left(\theta\right) = \frac{1}{10\sqrt{\pi}} e^{-\frac{1}{2}\left(\frac{\theta-10}{5}\right)^2}$$

Let's start at $\theta^{(0)} = 10$ and use a **uniform** proposal distribution $\theta' \sim U(\theta^{(i-1)} - 15, \theta^{(i-1)} + 15)$.



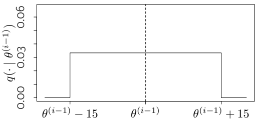
```
**Require:** $\theta^{(0)}$.
  **for** $i = 1, \ldots, n$ **do**
    Propose **candidate** $\theta' \sim q\left(\cdot \mid \theta^{(i-1)}\right)$.
    Calculate the **acceptance probability**:
      $\alpha = \min\left(1, \frac{f\left(\theta'\right) \cancel{q\left(\theta^{(i-1)} \mid \theta'\right)}}{f\left(\theta^{(i-1)}\right) \cancel{q\left(\theta' \mid \theta^{(i-1)}\right)}}\right)$
    Sample $u \sim U(0, 1)$
    **if** $u < \alpha$ **then**
      $\theta^{(i)} = \theta'$
    **else**
      $\theta^{(i)} = \theta^{(i-1)}$
    **end if**
  **end for**
```

Example: normal distribution

Let's sample from a **normal distribution** with mean 10 and standard deviation 5, with

$$f(\theta) = \frac{1}{10\sqrt{\pi}} e^{-\frac{1}{2}\left(\frac{\theta-10}{5}\right)^2}$$

Let's start at $\theta^{(0)} = 10$ and use a **uniform** proposal distribution $\theta' \sim U(\theta^{(i-1)} - 15, \theta^{(i-1)} + 15)$.

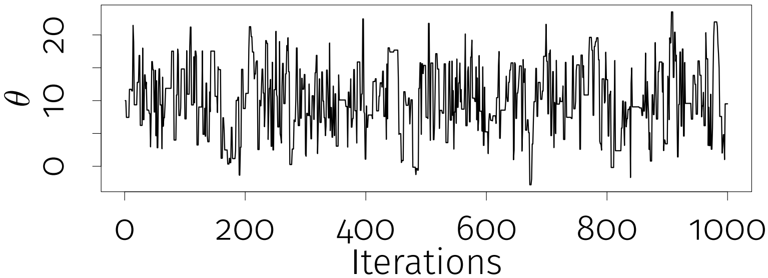


Symmetric proposals gives the **Metropolis** algorithm.

```
Require: $\theta^{(0)}$.
  for $i = 1, \dots, n$ do
      Propose candidate $\theta' \sim q\left(\cdot \mid \theta^{(i-1)}\right)$.
      Calculate the acceptance probability:
          $\alpha = \min\left(1, \frac{f(\theta')}{f\left(\theta^{(i-1)}\right)}\right)$
      Sample $u \sim U(0,1)$
      if $u < \alpha$ then
          $\theta^{(i)} = \theta'$
      else
          $\theta^{(i)} = \theta^{(i-1)}$
      end if
  end for
```

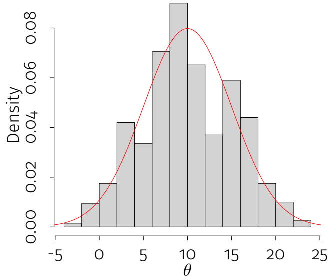
Example: normal distribution

Running this algorithm from $n = 1,000$ iterations gives the following chain:



Example: normal distribution

Plotting a **histogram** of these 1,000 samples provides an **empirical estimate** of the target distribution.



From these samples we can derive a **Monte Carlo** estimate of the mean:

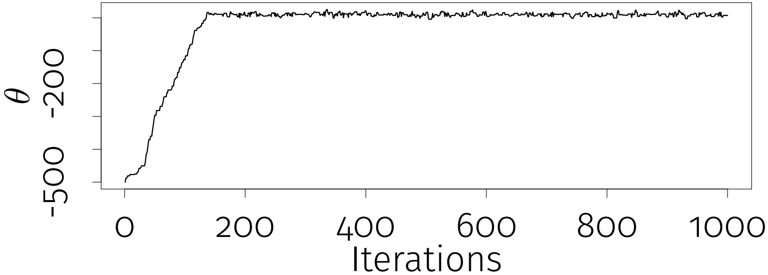
$$E(\theta) \approx \frac{1}{n} \sum_{i=1}^{1000} \theta_i = 10.15,$$

which can be compared to the true value $E(\theta) = 10$.

We can also derive **intervals**, **variances** etc.

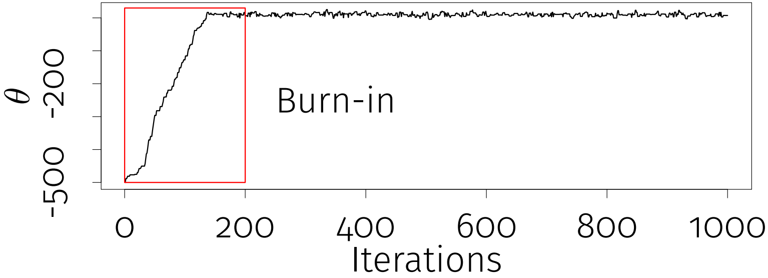
Example: normal distribution

Remarkably, it doesn't matter where we start the chain from. Here $\theta^{(0)} = -500$:



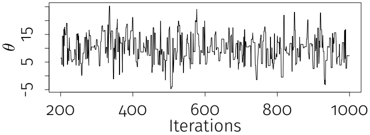
Example: normal distribution

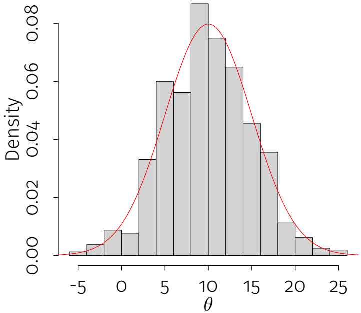
Remarkably, it doesn't matter where we start the chain from. Here $\theta^{(0)} = -500$:



Example: normal distribution

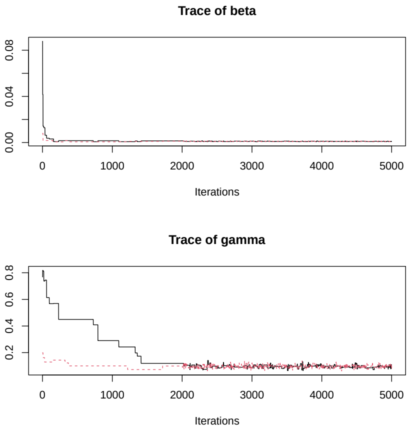
If we discard the **burn-in**, then we are left with samples from the **target** distribution as required.





Convergence

In order for us to make robust inference, we need to be sure that the chain has **converged**. There is no consensus[†], but a common approach is to run **multiple** chains from different **initial values**:



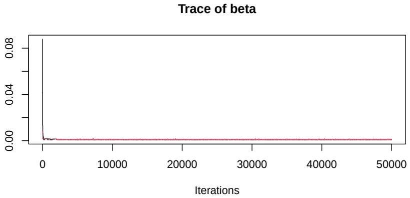
Trace of beta

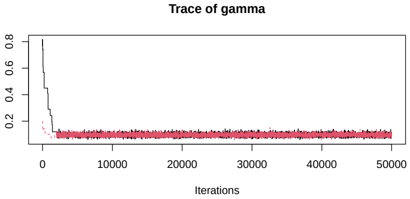
Trace of gamma

[†] or **guarantee**

Mixing

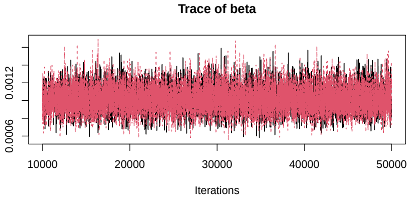
We also want to make sure that the chain is **exploring** the parameter space *efficiently*. This is known as **mixing**. After the **burn-in**, your **trace** plots should look like:

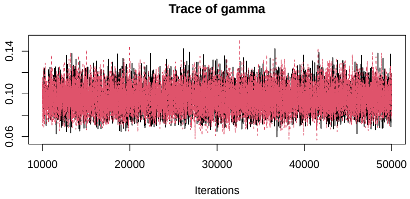




Mixing

We also want to make sure that the chain is **exploring** the parameter space *efficiently*. This is known as **mixing**. After the **burn-in**, your **trace** plots should look like:

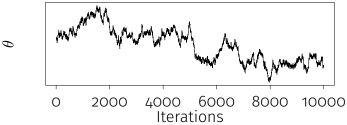




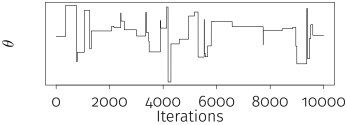
Improving efficiency

The **proposal distribution**, $q(\cdot)$, is key to efficient **mixing** and **convergence**.

In *random-walk* Metropolis-Hastings, the size of the **proposal variances** are important.



Here the proposal variance is **small** → **poor mixing**[a].



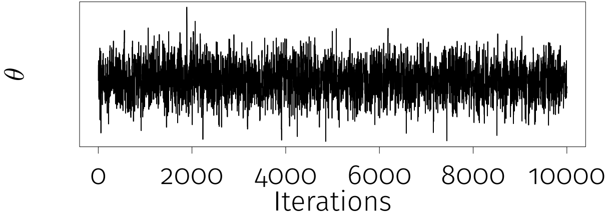
Here the proposal variance is **large** → **poor mixing**.

[a] high **autocorrelation**

Improving efficiency

Poor mixing means that the chain takes a long time to explore the posterior. This means you will have to run the chain for **much longer** to assess **convergence**.

Optimising MCMC is **hard**. In many problems we can use **adaptive proposal distributions** (e.g. Roberts and Rosenthal 2009), or alternative MCMC algorithms such as Hamiltonian Monte Carlo[†].



[†] such as used in the Stan software

Estimating posterior distributions

In Bayesian inference, the **target** distribution is the **posterior**:

$$f(\theta \mid \mathbf{y}) = \frac{f(\mathbf{y} \mid \theta) f(\theta)}{f(\mathbf{y})},$$

and the Metropolis-Hastings algorithm has **acceptance probability**:

$$\alpha = \min\left(1, \frac{f\left(\theta' \mid \mathbf{y}\right)}{f\left(\theta^{(i-1)} \mid \mathbf{y}\right)} \times \frac{q\left(\theta^{(i-1)} \mid \theta'\right)}{q\left(\theta' \mid \theta^{(i-1)}\right)}\right)$$

Estimating posterior distributions

In Bayesian inference, the **target** distribution is the **posterior**:

$$f(\theta \mid \mathbf{y}) = \frac{f(\mathbf{y} \mid \theta) f(\theta)}{f(\mathbf{y})},$$

and the Metropolis-Hastings algorithm has **acceptance probability**:

$$\alpha = \min\left(1, \frac{\frac{f\left(\mathbf{y}|\theta'\right) f(\theta')}{f(\mathbf{y})}}{\frac{f\left(\mathbf{y}|\theta^{(i-1)}\right) f(\theta^{(i-1)})}{f(\mathbf{y})}} \times \frac{q\left(\theta^{(i-1)} \mid \theta'\right)}{q\left(\theta' \mid \theta^{(i-1)}\right)}\right)$$

Estimating posterior distributions

In Bayesian inference, the **target** distribution is the **posterior**:

$$f(\theta \mid \mathbf{y}) = \frac{f(\mathbf{y} \mid \theta) f(\theta)}{f(\mathbf{y})},$$

and the Metropolis-Hastings algorithm has **acceptance probability**:

$$\alpha = \min\left(1, \frac{\frac{f\left(\mathbf{y} | \theta'\right) f(\theta')}{\cancel{f(\mathbf{y})}}}{\frac{f\left(\mathbf{y} | \theta^{(i-1)}\right) f(\theta^{(i-1)})}{\cancel{f(\mathbf{y})}}} \times \frac{q\left(\theta^{(i-1)} \mid \theta'\right)}{q\left(\theta' \mid \theta^{(i-1)}\right)}\right)$$

Estimating posterior distributions

In Bayesian inference, the **target** distribution is the **posterior**:

$$f(\theta \mid \mathbf{y}) = \frac{f(\mathbf{y} \mid \theta) f(\theta)}{f(\mathbf{y})},$$

and the Metropolis-Hastings algorithm has **acceptance probability**:

$$\alpha = \min\left(1, \frac{f\left(\mathbf{y} \mid \theta'\right) f(\theta')}{f\left(\mathbf{y} \mid \theta^{(i-1)}\right) f(\theta^{(i-1)})} \times \frac{q\left(\theta^{(i-1)} \mid \theta'\right)}{q\left(\theta' \mid \theta^{(i-1)}\right)}\right)$$

Hence the **normalising constant** cancels in the ratio!

Posteriors for functions of random variables

One of the beautiful aspects of using MCMC for estimation, is that it is trivial to generate **posterior samples** for any function of the parameters[†].

Mean	SD	2.5%	97.5%
1.3	0.23	0.89	1.8

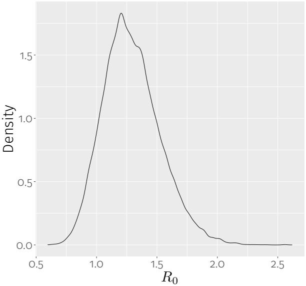
For example, given samples of $\beta^{(i)}$ and $\gamma^{(i)}$ $(i = 1, \ldots, N)$ from an SIR model, we can generate N samples of R_0 as:

$$R_0^{(i)} = \frac{\beta^{(i)} N}{\gamma^{(i)}}.$$

for $i = 1, \ldots, N$.

```
##         beta      gamma          ##        R0
##  0.001101749 0.10350498          ##  1.277329
##  0.001101749 0.10350498          ##  1.277329
##  0.001188174 0.08525523    →     ##  1.672401
##  0.001188174 0.08525523          ##  1.672401
##  0.001188174 0.08525523          ##  1.672401
##  0.001188174 0.08525523          ##  1.672401
```

Marginal for R_0



[†] from the Law of the Unconscious Statistician

Posterior predictive distributions

This idea carries through to **posterior predictive distributions**:

$$f\left(\mathbf{y}^* \mid \mathbf{y}\right) = \int_\Theta f\left(\mathbf{y}^* \mid \theta\right) f\left(\theta \mid \mathbf{y}\right) d\theta,$$

where $f\left(\mathbf{y}^* \mid \mathbf{y}, \theta\right)$ is the predictive distribution for a new observation $\mathbf{y}^*$ (defined by the **model**), given parameters θ; with $f\left(\theta \mid \mathbf{y}\right)$ the posterior density for θ.

Hence we **integrate** (or average) over the **posterior distribution**, and hence propagate uncertainty in the parameters directly through to the predictive density.

We can produce random samples from $f\left(\mathbf{y}^* \mid \mathbf{y}\right)$ by:

- taking N random samples from the **posterior** (via MCMC), $\theta^{(i)}$, then
- simulating $\left(\mathbf{y}^*\right)^{(i)} \sim f\left(\cdot \mid \theta^{(i)}\right)$.

These can be plotted / summarised in the usual way.

Software

There are various **general-purpose** Bayesian modelling packages, most notably:

- WinBUGS
- `jags`
- OpenBUGS
- `MCMCpack`
- `nimble`
- Stan / `rstan`

And various wrapper packages to make fitting simple (e.g. regression models much easier) e.g. the R packages `brms` and `rstanarm`.

Practical

In the first practical we will explore fitting the **catalytic model** for endemic diseases to serology data for ***rubella***.

We will use the R package `MCMCpack` to do this, since we can re-use the **likelihood function** we will derive for the MLEs.

For most use cases I would recommend `nimble` or Stan, which are under active development and very powerful. However, we don't have time to go over how to use these here (especially since the likelihood is non-standard).

References i

Gamerman, Dani, and Hedibert Freitas Lopes. 2006. *Markov Chain Monte Carlo: Stochastic Simulation for Bayesian Inference*. 2nd ed. CRC Press.

Gelman, Andrew, John B. Carlin, Hal S. Stern, and Donald B. Rubin. 2004. *Bayesian Data Analysis*. 2nd ed. Chapman; Hall/CRC.

Gilks, W. R., S. Richardson, and D. J. Spiegelhalter, eds. 1996. *Markov Chain Monte Carlo in Practice*. Chapman; Hall.

Hastings, W. K. 1970. "Monte Carlo Sampling Methods Using Markov Chains and Their Applications." *Biometrika* 57: 97–109.

Metropolis, N., A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. 1953. "Equations of State Calculations by Fast Computing Machine." *Journal of Chemical Physics* 21: 1087–91.

References ii

Roberts, Gareth O., and Jeffrey S. Rosenthal. 2009. “Examples of Adaptive MCMC.” *Journal of Computational and Graphical Statistics* 18 (2): 349–67. https://doi.org/10.1198/jcgs.2009.06134.