

Chapter 3: Bayesian inference

Big picture

In recent decades Bayesian inference has increasingly been used in ecology. A key difference between Bayesian and maximum likelihood approaches lies in which quantities are treated as random variables. For any likelihood function, the parameters θ are not random variables because there are no probability distributions associated with θ . In contrast, Bayesian approaches treat parameters and future data (actually all unobserved quantities) as random variables, meaning that each unknown is associated with a probability distribution $p(\theta)$. In both Bayesian and maximum likelihood approaches, the probability distribution for observed data conditional on parameters $p(y | \theta)$ plays a central role. See Hobbs and Hooten (Ch. 5) for a more detailed treatment on differences between Bayesian and maximum likelihood based inferential approaches. Some authors also point out differences between Bayesian and frequentist definitions of probability. In a frequentist framework, probabilities are defined in term of long run frequencies of events, often relying on hypothetical realizations. Bayesian probability definitions do not rely on the long-run frequency of events.

Philosophy aside, Bayesian approaches have become more popular because of intuitive appeal and practical advantages. Intuitively, it can seem backwards to focus on the probability of the data given the parameters $p(y | \theta)$. What we really want to know is the probability of the parameters, having observed some data $p(\theta | y)$. As we will see, Bayes' theorem allows us to calculate this probability. Second, Bayesian approaches are often easier to implement than maximum likelihood or frequentist approaches, particularly for complex models. Finally, we find that in many applications, Bayesian approaches facilitate a better understanding of model structure and assumptions.

Learning goals

- Bayes' theorem and Bayesian probability
- relationship between likelihood and Bayesian inference
- priors (generally, informative vs. non-informative)
- proper vs. improper priors
- intro to Bayesian computation and MCMC
- posterior summaries and comparisons
- single parameter models: MLE vs. Bayesian treatments
- Bayesian linear regression: intro to Stan

Bayes' theorem

Bayes' theorem is an incredibly powerful theorem that follows from the rules of probability. To prove the theorem, we need only a few ingredients: 1) the definition of joint probabilities $p(A, B) = p(A | B)p(B)$ or $p(A, B) = p(B | A)p(A)$ (both are valid) and 2) a bit of algebra.

$$p(A, B) = p(A | B)p(B)$$

$$p(B | A)p(A) = p(A | B)p(B)$$

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

This is Bayes' theorem. In modern applications, we typically substitute unknown parameters θ for B , and data y for A :

$$p(\theta | y) = \frac{p(y | \theta)p(\theta)}{p(y)}$$

The terms can verbally be described as follows:

- $p(\theta | y)$: the *posterior* distribution of the parameters. This tells us what the parameters probably are (and are not), conditioned on having observed some data y .
- $p(y | \theta)$: the likelihood of the data y conditioned on the parameters
- $p(\theta)$: the *prior* distributions of the parameters. This should represent our prior knowledge about the values of the parameters, including values we think are likely and values we think are highly unlikely. Prior knowledge comes from similar studies and/or first principles.
- $p(y)$: The marginal distribution of the data (i.e. not conditioned on any particular value of θ). This quantity can be difficult or even impossible to compute, will always be a constant after the data have been observed, and is often ignored.

Because $p(y)$ is a constant, it is valid and common to consider the posterior distribution up to this proportionality constant:

$$p(\theta | y) \propto p(y | \theta)p(\theta)$$

Prior distributions

We have already learned about likelihood, but the introduction of a prior distribution for the parameters requires some attention. From a Bayesian perspective, prior distributions $p(\theta)$ represent our knowledge/beliefs about parameters before having observed the data y , and the posterior distribution represents our updated knowledge/beliefs about parameters after having observed our data. This is similar to the way many scientists operate: we think we know something about a system, and then we do experiments and conduct surveys to update our knowledge about the system. But, the observations generated from the experiments and surveys are not considered in isolation. They are considered in the context of our previous knowledge. In this way, the posterior distribution represents a compromise between our prior beliefs and the likelihood.

The (informal) inclusion of prior information is not unique to Bayesian inference. When selecting study systems, designing experiments, cleaning or subsetting data, and choosing a likelihood function, we inevitably draw upon our previous knowledge of a system to formulate our model. In turn, the form of our model and nature of the data we choose to collect have direct affects on our model output. Consequently our interpretation of scientific evidence has been indirectly affected by our previous knowledge in ways we don't always specify or acknowledge. Bayesian inference simply formalizes the inclusion of some of this information directly into the probability model itself.

Analytical posterior with conjugate priors: Bernoulli case*

** this section is a bit math-heavy for illustration, but most of the time we won't find the posterior analytically*

The Bernoulli distribution is a probability distribution for binary random variables (e.g., those that take one of two values: dead or alive, male or female, heads or tails, 0 or 1, success or failure, and so on). The Bernoulli distribution has one parameter, p : the probability of "success" (or more generally, the probability of one of the two outcomes) in one particular event. A Bernoulli random variable takes one of these two values. The choice of which of the two possible outcomes is considered "success" is often arbitrary - we could consider either "heads" or "tails" to be a success if we wanted to. If p is the probability of success in one

particular trial, then the probability of failure is just the complement of p : $1 - p$, sometimes referred to as q , such that $p + q = 1$. For those familiar with the Binomial distribution, the Bernoulli distribution is a special case of the Binomial, with one trial $k = 1$. We can use the Bernoulli distribution as a likelihood function, where y is either zero or one: $y \in \{0, 1\}$, and p is our only parameter. Because p is a probability, we know that $0 \leq p \leq 1$.

We can express the likelihood for a Bernoulli random variable as follows.

$$p(y | p) = \begin{cases} p & \text{if } y = 1 \\ 1 - p & \text{if } y = 0 \end{cases}$$

Equivalently and more generally:

$$[y | p] = p^y (1 - p)^{1-y}$$

If we have n independent Bernoulli random variables, y_1, \dots, y_n , each with probability p , then the joint likelihood can be written as the product of the point-wise likelihoods:

$$[y_1, \dots, y_n | p] = p^{y_1} (1 - p)^{1-y_1} \dots p^{y_n} (1 - p)^{1-y_n}$$

$$[y | p] = \prod_{i=1}^n p^{y_i} (1 - p)^{1-y_i}$$

Recalling from algebra that $x^a x^b = x^{a+b}$, this implies:

$$[y | p] = p^{\sum_i y_i} (1 - p)^{n - \sum_i y_i}$$

Having obtained the likelihood, we now must specify a prior to complete our specification of the joint distribution of data and parameters $[y | p][p]$, the numerator in Bayes' theorem. In this case we know that p can only take on values on the interval $(0, 1)$, which means that some of the probability distributions we've utilized previously (e.g. normal) won't work because values below zero and above one have non-zero probabilities. Similarly, we wouldn't want to try and use a (non-sensical) discrete probability distribution – for example, giving equal weighting to $p = 0, 0.4, 0.6$, and 0.8 – which would automatically rule out many possible values between zero and one.

A natural choice among common probability distributions is the Beta distribution, which has two parameters α and β , with support on the interval $(0, 1)$. This is a good choice because its bounds are similar to those for probabilities. Additionally, the posterior distribution induced by the Beta prior distribution is also a Beta distribution. When a prior distribution for a parameter induces a posterior distribution that is of the same form (same probability distribution) as the prior, the prior is said to be a “conjugate prior” for the likelihood. The density of the beta distribution for parameter p has two parameters α and β and is defined as:

$$[p] = c p^{\alpha-1} (1 - p)^{\beta-1}$$

Where c is a constant that ensures that $[p | \alpha, \beta]$ integrates to one over the interval $(0, 1)$ (i.e., it is a true probability distribution), with $c = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)}$, and $\Gamma(x) = (x - 1)!$. That's a factorial symbol (!), not a punctuation mark!

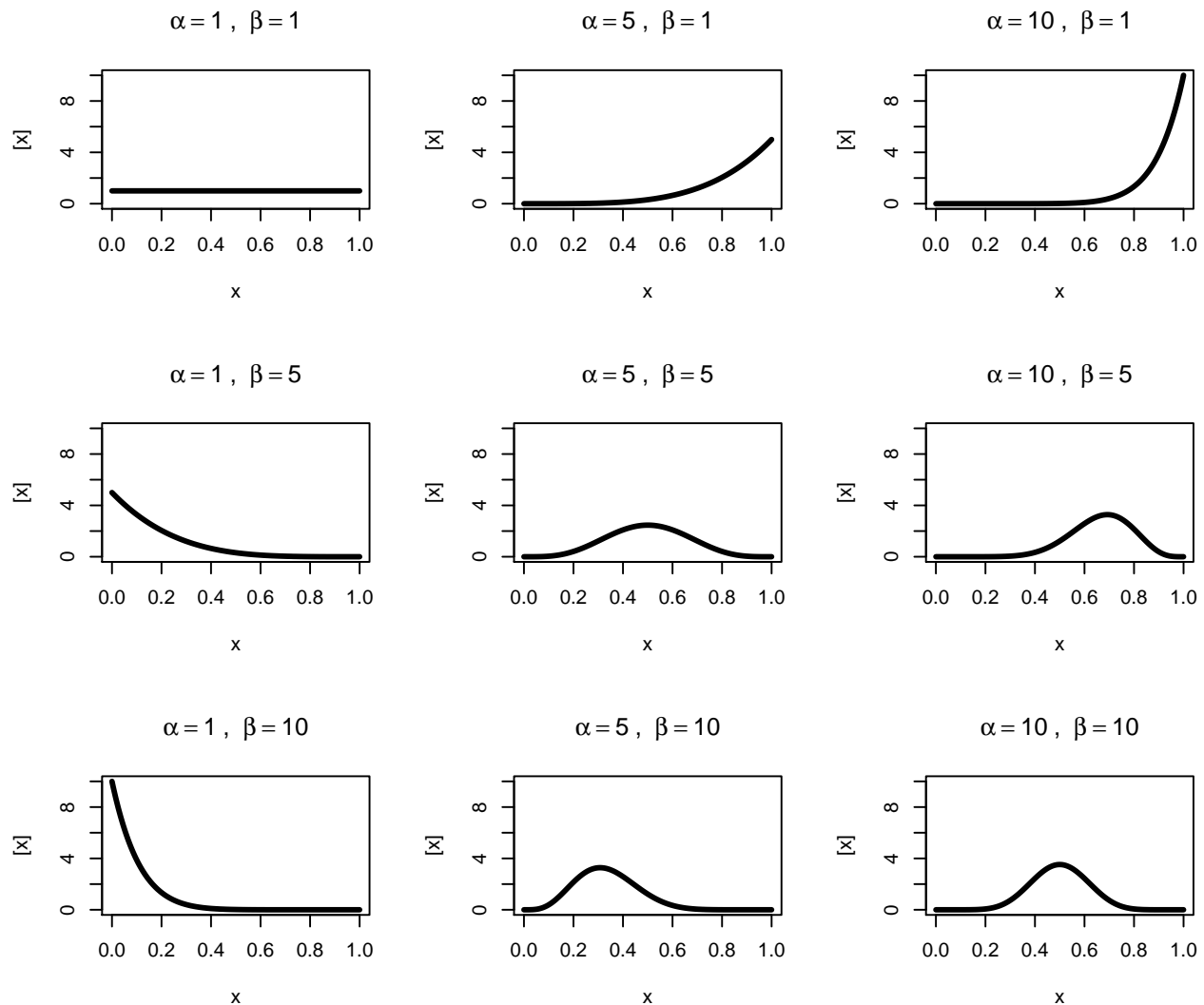
To give a bit of intuition for what the beta distribution looks like, here are some plots of the beta density with different values of α and β :

```

alpha <- rep(c(1, 5, 10))
beta <- rep(c(1, 5, 10))
g <- expand.grid(alpha=alpha, beta=beta)
x <- seq(0, 1, .005)

par(mfrow=c(3, 3))
for (i in 1:nrow(g)){
  plot(x, dbeta(x, g$alpha[i], g$beta[i]),
       type='l', ylim=c(0, 10),
       ylab="[x]", lwd=3)
  title(bquote(alpha == .(g$alpha[i]) ~ ", " ~ beta == .(g$beta[i])))
}

```



One commonly used prior is the $\text{beta}(1, 1)$, because it corresponds to a uniform prior for p (shown in the top left corner). That is, all values between zero and one are given equal weighting. Note that other beta priors put more weight on intermediate values of p (e.g. $\text{beta}(5,5)$), while others weight extremely low or extremely high values of p . Note that it is even possible to construct a beta prior that has high probability of extreme values near zero and one, and low probability elsewhere!

With a likelihood and prior in hand, we now have all of the ingredients to embark on our first Bayesian analysis for a Bernoulli random variable. We'll proceed by finding the posterior distribution up to some proportionality

constant, then we'll use our knowledge of the beta distribution to recover the correct proportionality constant:

$$[p|y] = \frac{[y | p][p]}{[y]}$$

$$[p|y] \propto [y | p][p]$$

Plugging in the likelihood and prior that we described above:

$$[p|y] \propto p^{\sum_i y_i} (1-p)^{n-\sum_i y_i} [p]$$

$$[p|y] \propto p^{\sum_i y_i} (1-p)^{n-\sum_i y_i} c p^{\alpha-1} (1-p)^{\beta-1}$$

Dropping c , because we're only working up to some proportionality constant:

$$[p|y] \propto p^{\sum_i y_i} (1-p)^{n-\sum_i y_i} p^{\alpha-1} (1-p)^{\beta-1}$$

Then, again recalling that $x^a x^b = x^{a+b}$, we find that

$$[p|y] \propto p^{\alpha-1+\sum_i y_i} (1-p)^{\beta-1+n-\sum_i y_i}$$

Notice that this is of the same form as the beta prior for p , with updated parameters: $\alpha_{post} = \alpha + \sum_i y_i$ and $\beta_{post} = \beta + n - \sum_i y_i$. In this sense, the parameters of the beta prior α and β can be interpreted as the previous number of successes and failures, respectively. Future studies can simply use the updated values α_{post} and β_{post} as priors.

In this case, we have found a quantity that is proportional to the posterior distribution, which often is enough, but here we can easily derive the proportionality constant that will ensure that the posterior integrates to one (i.e., it is a true probability distribution).

Recall the proportionality constant c from the beta distribution prior that we used, which is $c = \frac{\Gamma(\alpha + \beta)}{\Gamma(\alpha)\Gamma(\beta)}$.

Updating this proportionality constant gives us the correct value for our posterior distribution, ensuring that it integrates to one, so that we now can write down the posterior distribution in closed form:

$$[p|y] = \frac{\Gamma(\alpha_{post} + \beta_{post})}{\Gamma(\alpha_{post})\Gamma(\beta_{post})} p^{\alpha_{post}-1} (1-p)^{\beta_{post}-1}$$

Now we can explore the effect of our prior distributions on the posterior. Suppose that we have observed $n = 8$ data points, y_1, y_2, \dots, y_8 , with $y_i \in \{0, 1\}$ and 2 successes, $\sum_i y_i = 2$.

Let's graph the posterior distributions (blue) that are implied by the priors (grey) plotted above and the likelihood (red) resulting from these observations.

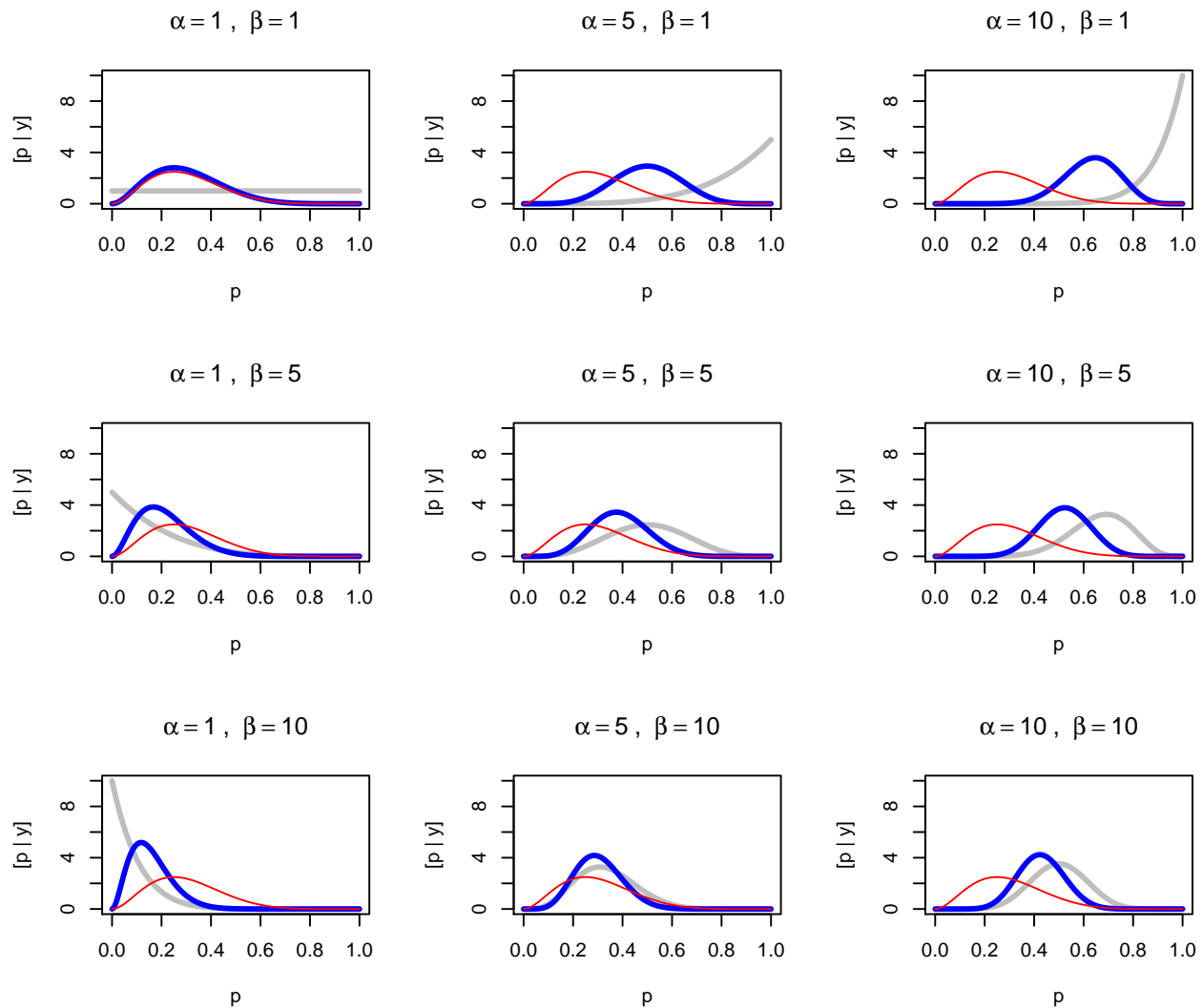
```
g$alpha_post <- g$alpha + 2
g$beta_post <- g$beta + 6

par(mfrow=c(3, 3))
for (i in 1:nrow(g)){
  plot(x, dbeta(x, g$alpha[i], g$beta[i]),
       type='l', ylim=c(0, 10),
       xlab="p", ylab="[p | y]", lwd=3, col='grey')
```

```

lines(x, dbeta(x, g$alpha_post[i], g$beta_post[i]),
      lwd=3, col='blue')
lines(x, 8 * dbinom(x=2, size=8, prob=x), col='red')
title(bquote(alpha == .(g$alpha[i]) ~ ", " ~ beta == .(g$beta[i])))
}

```



A few things to note:

- The uniform prior $\text{beta}(1,1)$ results in a posterior that is identical to the likelihood, meaning that the posterior mode is equal to the maximum likelihood estimate.
- strong priors (i.e. those with a lot of weight on a certain portion of distribution such as $\text{beta}(10,1)$) pull the posterior distribution toward them and away from the likelihood distribution
- weaker, intermediate priors (i.e. $\text{beta}(5,5)$) result in posteriors that are more of a compromise between the prior and the likelihood.

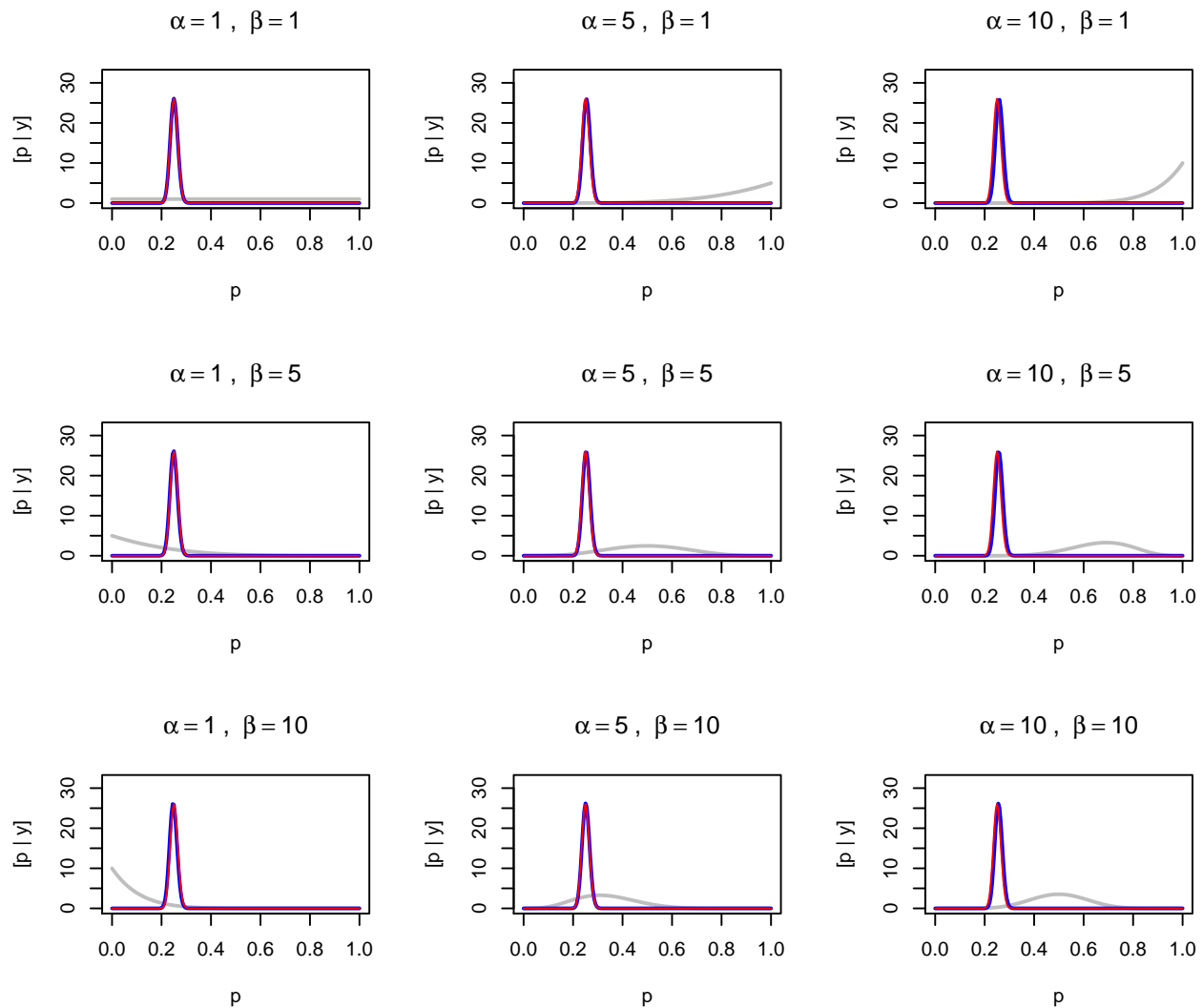
Some statisticians (typically non-Bayesian) will argue that only the uniform prior is appropriate because it assumes complete ignorance about possible values of the parameter (i.e. is unbiased) and allows the data to “speak” for themselves. Note, however, that having a uniform prior is not quite the same as having *no* prior, because even an explicit (or implicit in the case of maximum likelihood estimation) uniform prior represents a our prior belief (whether we want it to or not!)

Consequently, most Bayesian statisticians nowadays advocate for the inclusion of prior distributions that provide a better representation of “reasonable” values of our parameter rather than prior distributions that feign ignorance. For example, if we were trying to estimate the probability that coin flip came up heads (even when we expect that the coin to be biased), we may reasonably use a prior that doesn’t put much weight on probabilities near zero or one. We can do this because we know that even a biased coin will have to land on tails sometimes and head sometimes. We’ll encounter similar examples or reasonable priors for parameters that are more commonly estimated in biological models in later chapters.

Another nice feature of Bayesian approaches is that, given enough data, the prior tends to have less of an influence on the posterior. This is consistent with the notion that when reasonable people are presented with strong evidence, they tend to more or less agree even if they may have disagreed ahead of time (though at times the empirical evidence for this phenomenon may seem somewhat equivocal). To show this, let’s increase the amount of information in our data, so that we have $n = 800$ and $\sum y = 200$ successes.

```
g$alpha_post <- g$alpha + 200
g$beta_post <- g$beta + 600

par(mfrow=c(3, 3))
for (i in 1:nrow(g)){
  plot(x, dbeta(x, g$alpha[i], g$beta[i]),
       type='l', ylim=c(0, 32),
       xlab="p", ylab="[p | y]", col='grey', lwd=2)
  lines(x, dbeta(x, g$alpha_post[i], g$beta_post[i]),
        col='blue', lwd=2)
  lines(x, 800 * dbinom(x=200, size=800, prob=x), col='red')
  title(bquote(alpha == .(g$alpha[i]) ~ ", " ~ beta == .(g$beta[i])))
}
```



Improper priors

Improper priors do not integrate to one (they do not define proper probability distributions). However, improper priors have often been used because they are frequently parameterizations of known probability distributions that approximate uniform priors. For instance, a normal distribution with infinite variance will be essentially flat, but does not actually integrate to one. Sometimes improper priors can still lead to proper posteriors that integrate to one, but this must be checked analytically. Unless you're willing to prove that an improper prior leads to a proper posterior distribution, we recommend using proper priors.

Posterior computation the easy way

In reality, most of the time we don't analytically derive posterior distributions. Mostly, we can express our models in specialized programming languages that are designed to make Bayesian inference easier. Here is a Stan model statement from the above example. Stan model statements are best managed as text files with a `.stan` extension. For instance, we could call this `bern.stan`.

```
data {
  // define the types and names of the data
```



```

  int n; // n is an integer
  int<lower=0, upper=1> y[n]; // y is an integer vector with n elements
}

parameters {
  real<lower=0, upper=1> p; // p is a real number between 0 and 1
}

model {
  // define priors
  p ~ beta(1, 1);

  // define likelihood
  y ~ bernoulli(p);
}

```

The above model statement has three “blocks”. The data block specifies that we have two fixed inputs: the sample size n and a vector of integer values with n elements, and these have to be either zero or one. The parameter block specifies that we have one parameter p , which is a real number between 0 and 1. Last, the model block contains our beta prior for p and our Bernoulli likelihood for the data y .

Stan does not find the analytic expression for the posterior. Rather, Stan translates the above program into a Markov chain Monte Carlo (MCMC) algorithm to simulate samples from the posterior distribution. In practice, this is sufficient to learn about the model parameters.

What is MCMC?

MCMC is used to sample from probability distributions generally, and from posterior probability distributions in the context of Bayesian inference. This is a huge topic, and involves a fair bit of theory that we will not dwell on here, but it is worth reading Chapter 18 in Gelman and Hill for a taste of some of the algorithms that are used. In this class, we will rely on specialized software to conduct MCMC simulations, so that many of the details are left “under the hood”. However, it is still important to know what MCMC is (supposed to be) doing, and how to identify when MCMC algorithms fail.

In a Bayesian context, we often run multiple Markov chain simulations, where we iteratively update our parameter vector θ . Typically, we initialize the chains at different (often random) points in parameter space. If all goes well, then after some number of iterations, every chain has converged to the posterior distribution, and we run the chains a bit longer to generate a representative sample from the posterior, then perform inference on our sample. At this point, usually the early draws from the Markov chains are discarded as “warmup” or “burnin”, as these do not represent draws from the posterior, because the chains had not converged.

It is always necessary to run diagnostics on MCMC output to ensure convergence. Some of these are graphical. Traceplots show the parameter values that a Markov chain has taken on the y-axis, and iteration number on the x-axis. For instance, if we run our Stan model from before:

```

# make up some data
library(rstan)
y <- c(1, 1, 0, 0, 0, 0, 0, 0)
n <- length(y)

# bundle into a list
stan_d <- list(y = y, n = n)

out <- stan('bern.stan', data = stan_d)

```

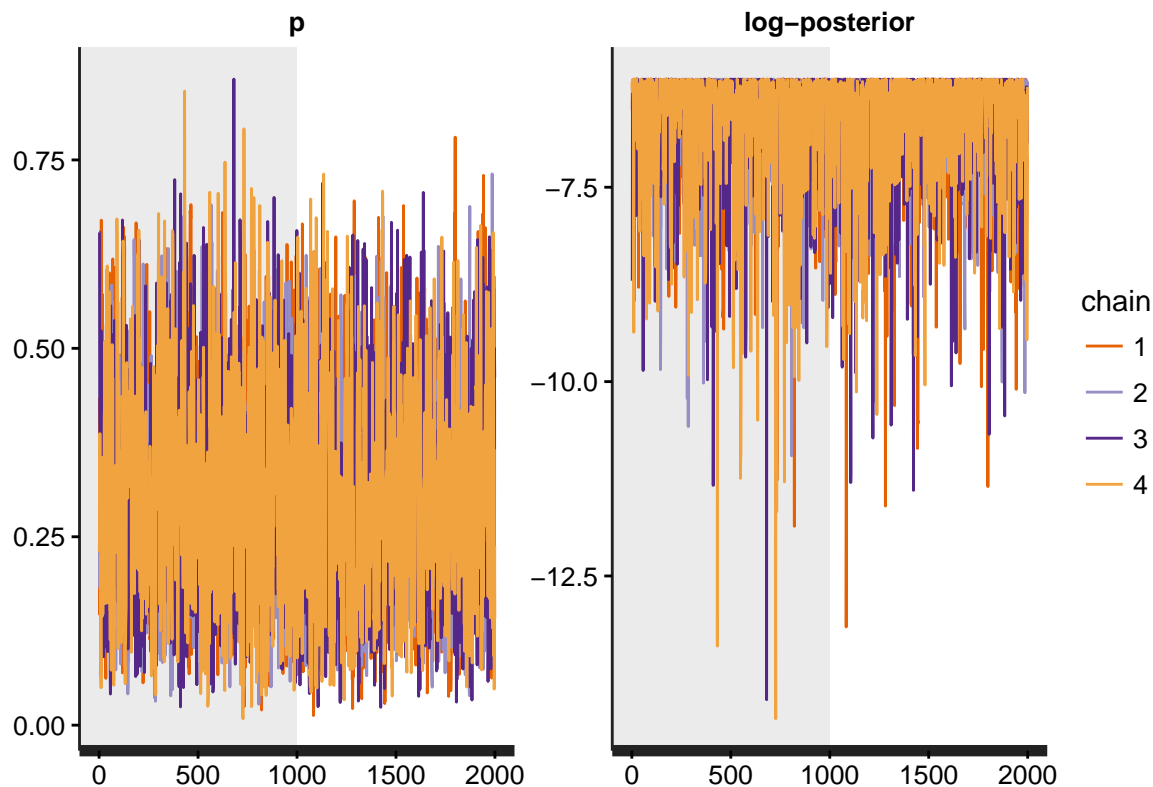
```

##
## SAMPLING FOR MODEL 'bern' NOW (CHAIN 1).
##
## Chain 1, Iteration:    1 / 2000 [  0%] (Warmup)
## Chain 1, Iteration:   200 / 2000 [ 10%] (Warmup)
## Chain 1, Iteration:   400 / 2000 [ 20%] (Warmup)
## Chain 1, Iteration:   600 / 2000 [ 30%] (Warmup)
## Chain 1, Iteration:   800 / 2000 [ 40%] (Warmup)
## Chain 1, Iteration:  1000 / 2000 [ 50%] (Warmup)
## Chain 1, Iteration:  1001 / 2000 [ 50%] (Sampling)
## Chain 1, Iteration:  1200 / 2000 [ 60%] (Sampling)
## Chain 1, Iteration:  1400 / 2000 [ 70%] (Sampling)
## Chain 1, Iteration:  1600 / 2000 [ 80%] (Sampling)
## Chain 1, Iteration:  1800 / 2000 [ 90%] (Sampling)
## Chain 1, Iteration:  2000 / 2000 [100%] (Sampling)#
## # Elapsed Time: 0.00647 seconds (Warm-up)
## #               0.005599 seconds (Sampling)
## #               0.012069 seconds (Total)
## #
##
## SAMPLING FOR MODEL 'bern' NOW (CHAIN 2).
##
## Chain 2, Iteration:    1 / 2000 [  0%] (Warmup)
## Chain 2, Iteration:   200 / 2000 [ 10%] (Warmup)
## Chain 2, Iteration:   400 / 2000 [ 20%] (Warmup)
## Chain 2, Iteration:   600 / 2000 [ 30%] (Warmup)
## Chain 2, Iteration:   800 / 2000 [ 40%] (Warmup)
## Chain 2, Iteration:  1000 / 2000 [ 50%] (Warmup)
## Chain 2, Iteration:  1001 / 2000 [ 50%] (Sampling)
## Chain 2, Iteration:  1200 / 2000 [ 60%] (Sampling)
## Chain 2, Iteration:  1400 / 2000 [ 70%] (Sampling)
## Chain 2, Iteration:  1600 / 2000 [ 80%] (Sampling)
## Chain 2, Iteration:  1800 / 2000 [ 90%] (Sampling)
## Chain 2, Iteration:  2000 / 2000 [100%] (Sampling)#
## # Elapsed Time: 0.006246 seconds (Warm-up)
## #               0.006999 seconds (Sampling)
## #               0.013245 seconds (Total)
## #
##
## SAMPLING FOR MODEL 'bern' NOW (CHAIN 3).
##
## Chain 3, Iteration:    1 / 2000 [  0%] (Warmup)
## Chain 3, Iteration:   200 / 2000 [ 10%] (Warmup)
## Chain 3, Iteration:   400 / 2000 [ 20%] (Warmup)
## Chain 3, Iteration:   600 / 2000 [ 30%] (Warmup)
## Chain 3, Iteration:   800 / 2000 [ 40%] (Warmup)
## Chain 3, Iteration:  1000 / 2000 [ 50%] (Warmup)
## Chain 3, Iteration:  1001 / 2000 [ 50%] (Sampling)
## Chain 3, Iteration:  1200 / 2000 [ 60%] (Sampling)
## Chain 3, Iteration:  1400 / 2000 [ 70%] (Sampling)
## Chain 3, Iteration:  1600 / 2000 [ 80%] (Sampling)
## Chain 3, Iteration:  1800 / 2000 [ 90%] (Sampling)
## Chain 3, Iteration:  2000 / 2000 [100%] (Sampling)#
## # Elapsed Time: 0.006331 seconds (Warm-up)

```

```
## #           0.005588 seconds (Sampling)
## #           0.011919 seconds (Total)
## #
##
## SAMPLING FOR MODEL 'bern' NOW (CHAIN 4).
##
## Chain 4, Iteration:    1 / 2000 [  0%] (Warmup)
## Chain 4, Iteration:   200 / 2000 [ 10%] (Warmup)
## Chain 4, Iteration:   400 / 2000 [ 20%] (Warmup)
## Chain 4, Iteration:   600 / 2000 [ 30%] (Warmup)
## Chain 4, Iteration:   800 / 2000 [ 40%] (Warmup)
## Chain 4, Iteration:  1000 / 2000 [ 50%] (Warmup)
## Chain 4, Iteration: 1001 / 2000 [ 50%] (Sampling)
## Chain 4, Iteration: 1200 / 2000 [ 60%] (Sampling)
## Chain 4, Iteration: 1400 / 2000 [ 70%] (Sampling)
## Chain 4, Iteration: 1600 / 2000 [ 80%] (Sampling)
## Chain 4, Iteration: 1800 / 2000 [ 90%] (Sampling)
## Chain 4, Iteration: 2000 / 2000 [100%] (Sampling)#
## # Elapsed Time: 0.006168 seconds (Warm-up)
## #           0.005618 seconds (Sampling)
## #           0.011786 seconds (Total)
## #
```

```
traceplot(out, pars = c('p', 'lp__'), inc_warmup=TRUE)
```



This traceplot is useful for verifying convergence. In this case, we’ve plotted the “warmup” samples (grey background) and the samples drawn after the “warmup period” (white background). In this case, all four of the chains appear to be sampling from the same region (somewhere between $p = 0.2$ and 0.4). We can also inspect some numerical summaries that are used to detect non-convergence. Specifically, we can look at the

\hat{R} statistic, which is a measure of how similar the chains are to one another. If $\hat{R} > 1.1$, then we ought to be worried about convergence of our chains. In addition, plotting autocorrelation function plots can help to diagnose autocorrelation and inefficient MCMC algorithms, but won't necessarily detect non-convergence. This is important because we need a decent number of independent samples to make reliable inference. Printing our model output returns this statistic as well as some other summary statistics for the posterior draws.

```
out
```

```
## Inference for Stan model: bern.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##      mean se_mean   sd  2.5%   25%   50%   75%  97.5% n_eff Rhat
## p      0.30     0.00 0.14  0.07  0.20  0.29  0.39  0.59  1323    1
## lp__ -6.64     0.02 0.73 -8.70 -6.82 -6.36 -6.17 -6.11  1385    1
##
## Samples were drawn using NUTS(diag_e) at Sat Jan 23 20:18:11 2016.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

This output also tells us that we ran four MCMC chains, each for 2000 iterations, and the first 1000 iterations were discarded as warmup (the shaded region in the traceplot). For each parameter (and for the log probability up to a proportionality constant `lp__`), we get the posterior mean, the MCMC standard error of the mean, the posterior standard deviation, some quantiles, and an estimate for the number of effective samples from the posterior `n_eff`, which should typically be at least in the hundreds.

Example: normal linear models

Recall that all normal linear models can be expressed as:

$$y \sim N(X\beta, \sigma^2)$$

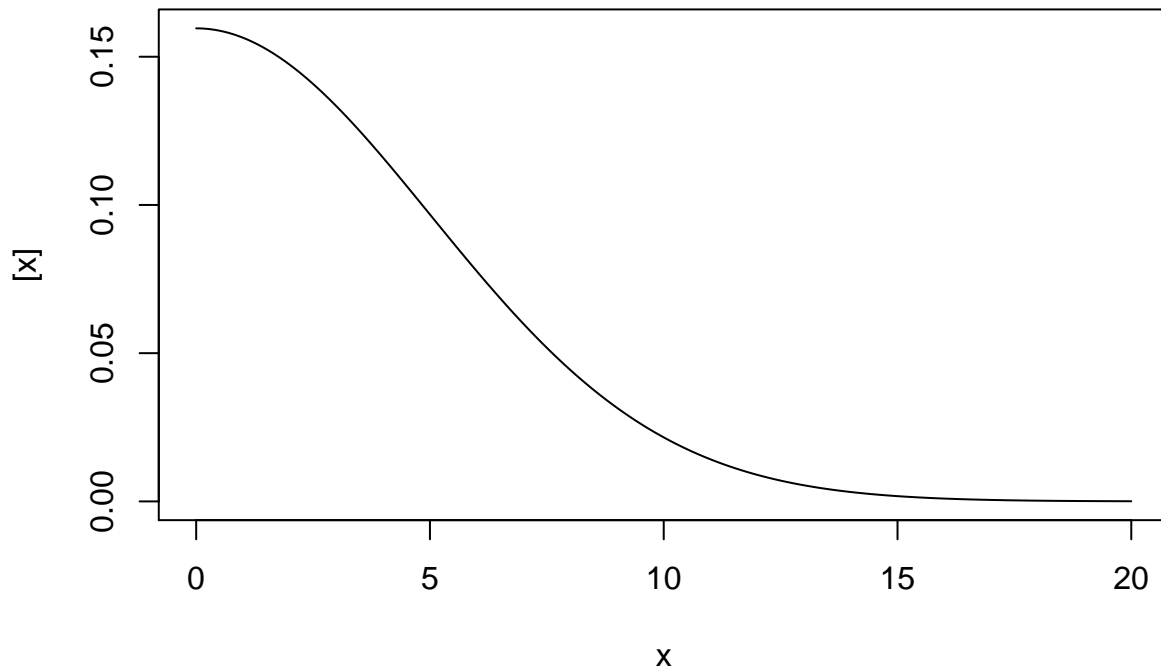
To complete a Bayesian analysis, we need to select prior distributions for the unknown parameters β and σ^2 . For instance:

$$\beta \sim N(0, 5)$$

$$\sigma \sim \text{halfNormal}(0, 5)$$

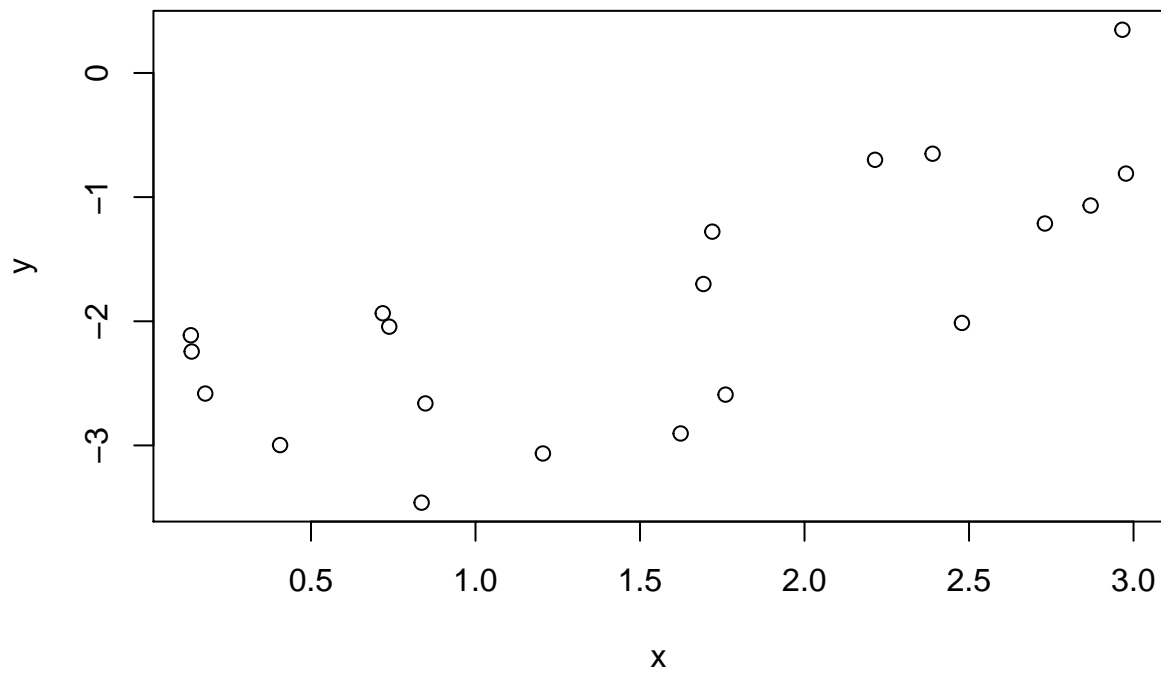
where the half-Normal with mean zero is a folded Gaussian probability density function with only positive mass:

half-Normal density



Let's do a quick linear regression model and summarize/plot the results.

```
n <- 20  
x <- runif(n, 0, 3)  
y <- rnorm(n, -3 + .75 * x, 1)  
plot(x, y)
```



We'll use Stan again, and we'll save the model file somewhere else with a `.stan` file extension. For instance, maybe we have a general purpose Stan model that can be used for linear models called `lm.stan`:

```

data {
  int n; // sample size
  int p; // number of coefficients
  matrix[n, p] X;
  vector[n] y;
}

parameters {
  vector[p] beta;
  real<lower=0> sigma;
}

model {
  beta ~ normal(0, 5);
  sigma ~ normal(0, 5);
  y ~ normal(X * beta, sigma);
}

```

So we have this file saved somewhere as `lm.stan`, and it can fit any of the linear models that we covered in Chapter 1 by changing the design matrix, but now we can include information to improve our estimates. Fitting the model:

```

library(rstan)
X <- matrix(c(rep(1, n), x), ncol = 2)
stan_d <- list(n = nrow(X), p = ncol(X), X = X, y = y)
out <- stan('lm.stan', data = stan_d)

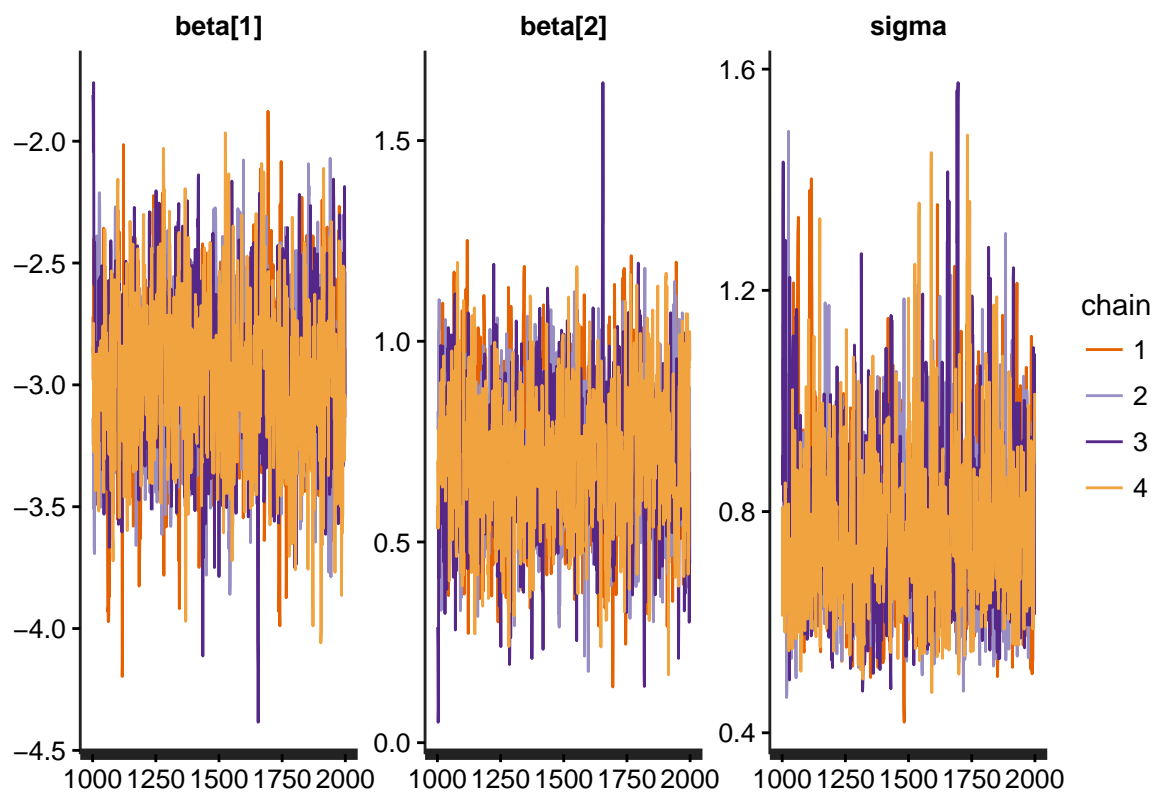
```

There are also some other default plots which are nice:

```

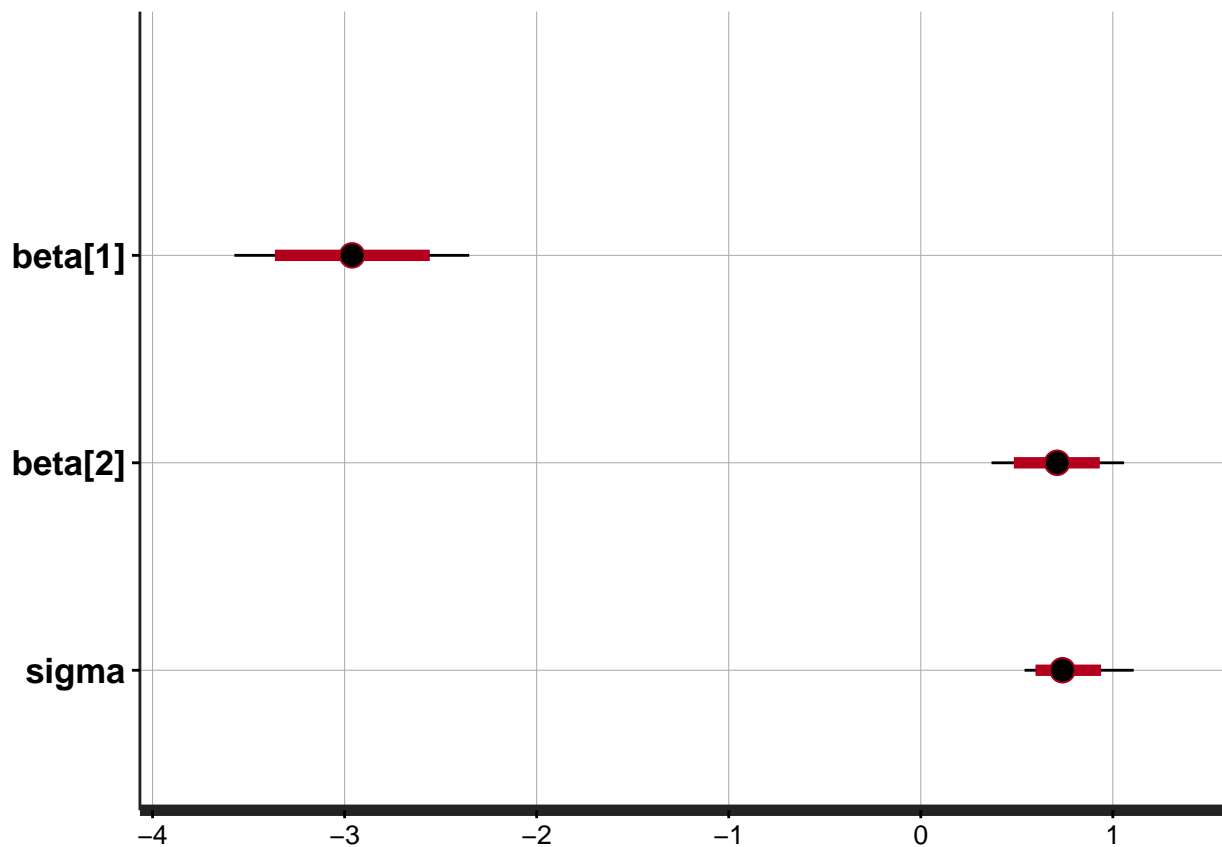
traceplot(out)

```

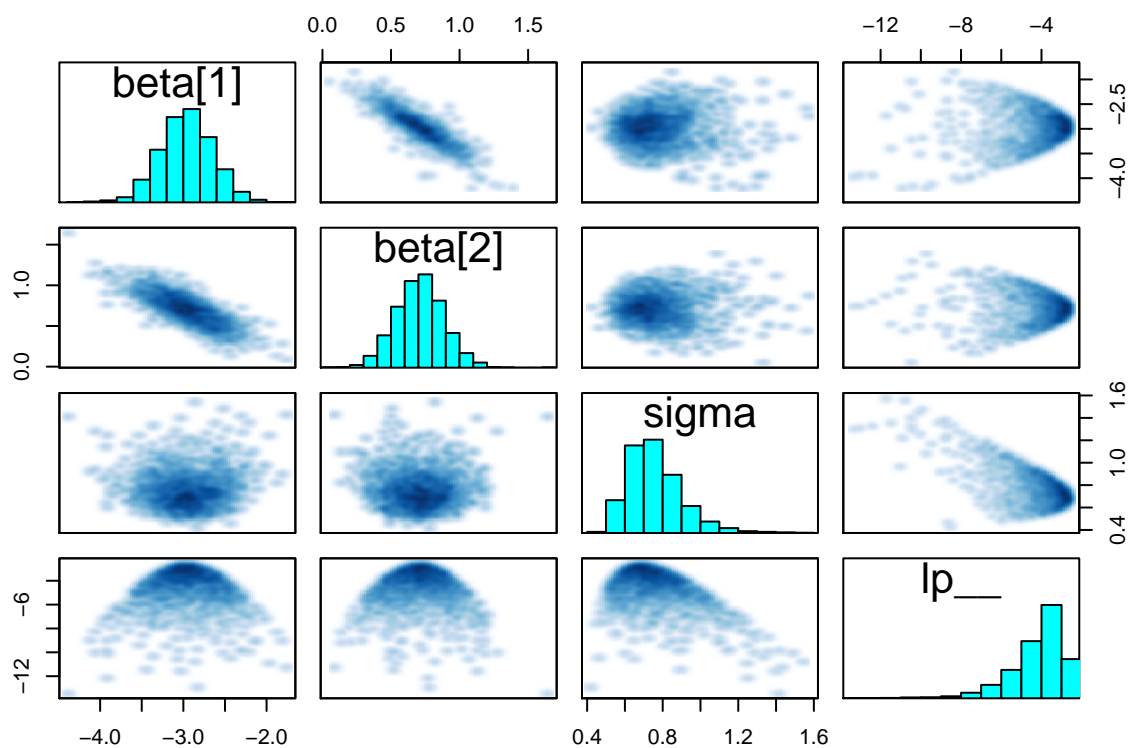


```
plot(out)
```

```
## ci_level: 0.8 (80% intervals)
## outer_level: 0.95 (95% intervals)
```



```
pairs(out)
```



Notice that the slopes and intercepts are correlated in the posterior (do you recall why?). Also, $lp_$ is

tracked automatically, and this is proportional to the log probability of the posterior distribution.

Let's inspect the output in table form:

out

```
## Inference for Stan model: lm.
## 4 chains, each with iter=2000; warmup=1000; thin=1;
## post-warmup draws per chain=1000, total post-warmup draws=4000.
##
##      mean se_mean   sd  2.5%  25%   50%   75%  97.5% n_eff Rhat
## beta[1] -2.96    0.01 0.32 -3.57 -3.17 -2.96 -2.76 -2.35  1189 1.00
## beta[2]  0.71    0.01 0.18  0.37  0.59  0.71  0.83  1.06  1103 1.00
## sigma   0.76    0.00 0.15  0.54  0.66  0.74  0.84  1.11  1074 1.00
## lp__    -4.20    0.05 1.38 -7.61 -4.82 -3.86 -3.20 -2.65   932 1.01
##
## Samples were drawn using NUTS(diag_e) at Sat Jan 23 20:18:13 2016.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

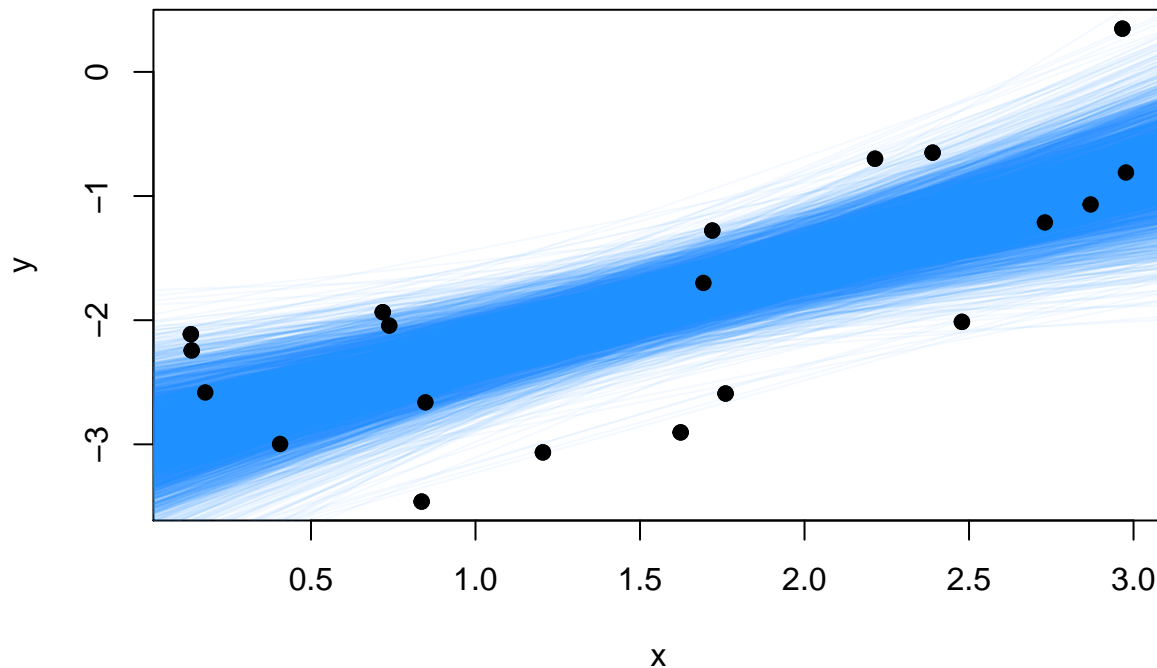
And finally, let's extract our samples from the posterior and plot our estimated line of best fit.

```
library(scales)
post <- rstan::extract(out)

# draw points
plot(x, y)

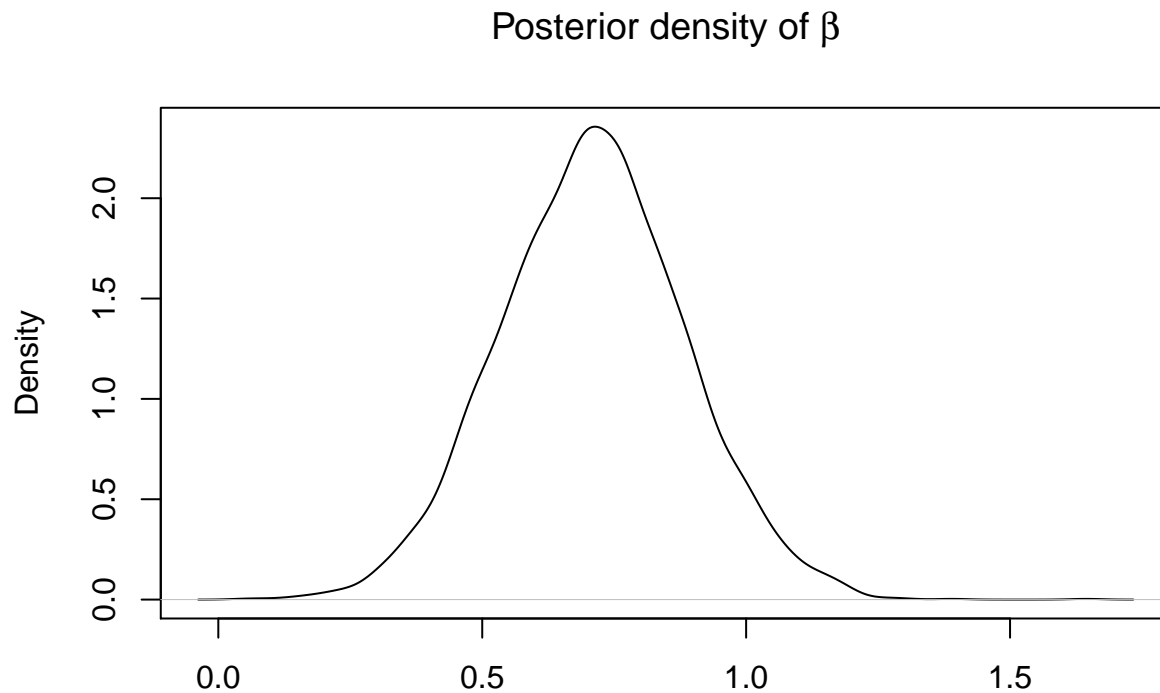
# add a line for each draw from the posterior
n_iter <- length(post$lp__)
for (i in 1:n_iter){
  abline(post$beta[i, 1], post$beta[i, 2], col=alpha('dodgerblue', .05))
}

# add points again so that they are visible over the line
points(x, y, pch=19)
```



We might be particularly interested in the effect of x on y . In the frequentist world, our inference is often centered around a null hypothesis that x has no effect on y . However, we have left the world of frequentism, so what kinds of inference can we make about the relationship between x and y given our posterior. For starters, it doesn't make sense to ask what the probability is that the slope is exactly equal to zero (i.e. the probability of a frequentist null hypothesis). If we think of a posterior probability density of our slope, then the true probability of any one particular value is zero because this is a probability density function, not a probability mass function (this is true for $\beta = 0$ and for $\beta = .0112351351$, for instance).

```
plot(density(post$beta[, 2]),
     main=expression(paste('Posterior density of ', beta)))
```



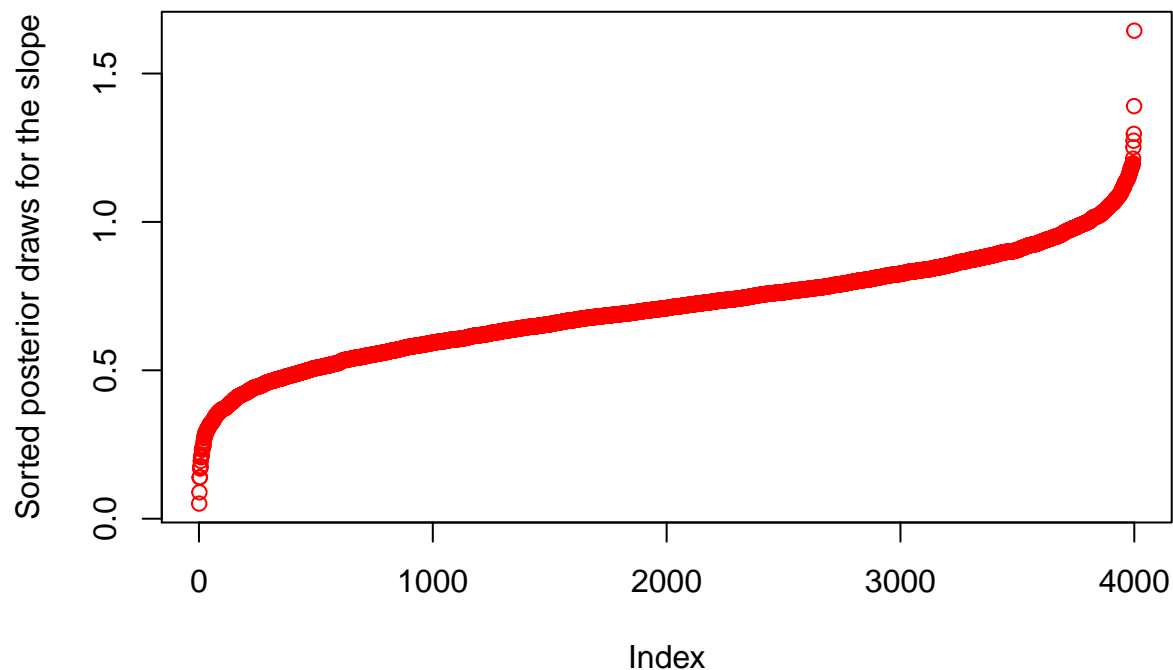
N = 4000 Bandwidth = 0.02971

A better question (answerable with a probability density function) might be: given the data (and, implicitly, our prior information and the model we used), what is the probability that x has a positive effect on y ? This is equivalent to asking about the area to the right of zero in the posterior distribution of β , and the number is approximated simply the proportion of posterior draws greater than zero.

```
sum(post$beta[, 2] > 0)/4000 # we have 4000 draws from the posterior
```

```
## [1] 1
```

```
ord <- order(post$beta[, 2])
plot(post$beta[ord, 2],
     col=ifelse(post$beta[ord, 2] > 0, 'red', 'black'),
     ylab='Sorted posterior draws for the slope')
```



Alternatively, we might believe that small positive values of the slope are not biologically relevant, and we might instead ask about the probability that the slope is above a biologically meaningful cutoff depending on the units of x and y (say $\beta > 0.5$).

```
sum(post$beta[, 2] > 0.5)/4000
```

```
## [1] 0.884
```

We might also construct a 95% credible interval for our estimate to communicate uncertainty in our estimate of β .

```
quantile(post$beta[, 2], probs=c(0.025, 0.975))
```

```
##      2.5%      97.5%
## 0.3685008 1.0586027
```

Conditional on the data (and, again, the prior and the model), there is a 95% probability that the true parameter value is in the credible interval. This is different from the interpretation of frequentist confidence intervals, which relies on long-run frequencies for imaginary realizations of the data collection and interval estimation procedure. In other words, our confidence in frequentist confidence intervals is in the procedure of creating confidence intervals - in the hypothetical long run, 95% of the intervals that we construct will contain the true value. As pointed out [here](#), this is the right answer to the wrong question.

Additionally, the Bayesian credible interval gives the the quantity that we often we often want to ascribe frequentist confidence intervals. That is, many practitioners treat confidence intervals as though the most likely value for the parameter falls in the middle, with decreasing probability towards the bounds of the confidence interval. In fact, we can ascribe no such distribution to a confidence interval, as each value contained therein is equally likely to be the “true” value of the parameter. Only by actually calculating many such confidence intervals on many sets of randomly sampled data could we begin to narrow down where the “true” value lies.

The credible interval constructed above has equal probability density on either side of the interval because we based our interval end-points on quantiles. Sometimes, we may wish to instead construct the narrowest

possible interval that encapsulates some proportion of the probability mass (e.g. 95%). These intervals are called highest density posterior intervals, and can be constructed using the following function:

```
HDI <- function(values, percent=0.95){
  sorted <- sort(values)
  index <- floor(percent * length(sorted))
  nCI <- length(sorted) - index

  width <- rep(0, nCI)
  for (i in 1:nCI){
    width[i] <- sorted[i + index] - sorted[i]
  }

  HDImin <- sorted[which.min(width)]
  HDImax <- sorted[which.min(width) + index]
  HDIlim <- c(HDImin, HDImax)
  return(HDIlim)
}

rbind(quantile(post$beta[, 2], probs=c(0.025, 0.975)),HDI(post$beta[, 2]))

##           2.5%    97.5%
## [1,] 0.3685008 1.058603
## [2,] 0.3579826 1.044092
```

In this instance, our posterior is fairly symmetric and the two types of credible intervals will not be much different. However, if we were working with posterior distributions that were highly asymmetric, then the highest density posterior intervals and the quantile based credible intervals will tend to differ. The main consideration is that the intervals or other summary statistics reported align with the research question of interest, rather than blindly picking one summary statistic that we assume will work well all of the time.

Last, we mention that we are not limited to working with the posterior distributions of the parameters defined explicitly in the model. We can also calculate the posterior distributions of any derived parameters immediately from the simulated posterior draws. For instance, if we were interested in the posterior distribution of the correlation between the intercept and slope, we can evaluate the correlation for each posterior draw to derive the full posterior distribution for this correlation. We could do the same if we were interested in the posterior distribution for the ratio between the slope and intercept, the difference between the slope and intercept, and so forth. This is one major advantage of the Bayesian framework in many situations, because often the targets of inference are functions of other parameters.

Further reading

Gelman and Hill. 2009. *Data analysis using regression and multilevel/hierarchical models*. Chapter 18.

Hobbs and Hooten. 2015. *Bayesian models: a statistical primer for ecologists*. Chapter 7.

Gelman et al. 2014. *Bayesian data analysis. Third edition*. Chapter 1-3.

Ellison AM. 2004. Bayesian inference in Ecology. *Ecology Letters* 7: 509-520.