Martin Luther University of Halle-Wittenberg
Department of Economics
Chair of Econometrics

Econometrics II
2. Bayesian estimation and inference

Christoph Wunder

Goals

- Foundations of Bayesian estimation
- MCMC
- Write simple computer code in R and Stan to draw samples from the posterior distribution
- Summarize samples

Outline

Outline

2.1 Overview: building a simple Bayesian model

Gelman et al. (2013), Ch. 1; McElreath (2020), Ch. 2

1. Model design: setting up a joint probability model for all observable and unobservable quantities in a problem
2. Bayesian updating: calculating the posterior distribution (conditional probability distribution of the parameters given the observed data)
3. Model checking: how well does the model fit the data? Are the conclusions reasonable? Evaluation may lead to model revision.

Outline

2.2 Components of the model

McElreath (2020), Ch. 2.2

Example: nine tosses of a globe

- How much of the surface of planet earth is covered in water?
- Strategy: toss a globe
- Sequence: W L W W W L W L W
- How do the data arise?
  - True proportion of water is π
  - A single toss of the globe is a Bernoulli random variable, where $P(\text{water}) = \pi$ and $P(\text{land}) = 1 - \pi$.
  - Tosses are independent.
- Translate data story into formal probability model.
  - Choose a likelihood (data model)
  - Choose a prior (parameter model)

2.2.1 Likelihood (data model)

- The probability of w water observations (W) in n tosses, with probability π of W on each toss and n tosses total, follows a binomial distribution:

$$P(w|n,\pi) = \binom{n}{w} \pi^{w} (1-\pi)^{n-w} \tag{2.1}$$

- Likelihood function of π: when we regard $P(w|n,\pi)$ as a function of π while holding fixed the data values, then $P(w|n,\pi)$ specifies the probability of the data as a function of π.
- The likelihood function is not a probability distribution, as it does not integrate to 1.

2.2.2 Prior (parameter model)

- Parameters cannot be observed and are estimated from the data.
- A Bayesian model requires an initial plausibility assignment for each possible value of the parameter. The prior is the initial set of plausibilities.
- The prior incorporates any prior knowledge about the model (parameters) before the data are measured.
- In the globe tossing example, we assume equal plausibility for each possible value of π, i.e. the prior distribution of π is uniform over the [0,1] interval:

$$P(\pi) = \frac{1}{1-0} = 1 \tag{2.2}$$

- Reasons for using priors:
  - Flat priors: mimic frequentist methods
  - Weakly informative priors: rule out unreasonable parameter values and make a-priori implausible values unlikely
  - Informative priors: incorporate specific expert information into the model
  - Multilevel priors: represent known data structures (e.g., longitudinal data, spacial units)
- Vast literature:
  - The prior can often only be understood in the context of the likelihood: aspects of the prior distributions persist into the posterior (Gelman et al. 2017).
  - Prior Choice Recommendations: `https://github.com/stan-dev/stan/wiki/Prior-Choice-Recommendations`

2.2.3 Posterior

- Bayesian statistical conclusions are probability statements about parameters given the data.
- The posterior is a compromise of prior and likelihood.
- In the globe tossing model, the posterior is defined as (omitting n):

$$P(\pi|w) = \frac{P(w|\pi)P(\pi)}{P(w)}, \tag{2.3}$$

  where the marginal (or average) likelihood

$$P(w) = \int P(w|\pi)P(\pi)d\pi \tag{2.4}$$

  is used to standardize the posterior so that it integrates to one.
- In general:

$$\text{Posterior} = \frac{\text{Likelihood} \times \text{Prior}}{\text{Average likelihood}}. \tag{2.5}$$

- Likelihood: binomial sampling model

$$P(w|n,\pi) = \binom{n}{w}\pi^{w}(1-\pi)^{n-w} \tag{2.6}$$

- Prior: uniform on the interval $[0,1]$

$$P(\pi) = \frac{1}{1-0} = 1 \tag{2.7}$$

- Applying Bayes' rule gives the posterior density up to a constant of proportionality

$$P(\pi|w) = \frac{\binom{n}{w}\pi^{w}(1-\pi)^{n-w}}{\int P(w|\pi)P(\pi)d\pi} \propto \pi^{w}(1-\pi)^{n-w}. \tag{2.8}$$

- General:

$$\text{posterior} \propto \text{likelihood} \times \text{prior} \tag{2.9}$$

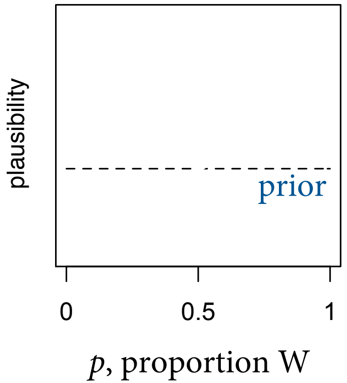
Outline

2.3 Bayesian updating

2.3.1 Idea: converting prior into posterior

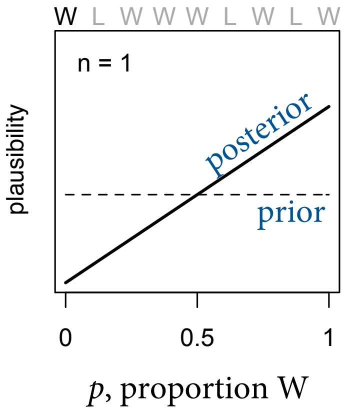
McElreath (2020), Ch. 2.2

The same plausibility is assigned to every value π.



Source: McElreath (2020), Figure 2.5

The Bayesian model updates the prior probabilities given the data to produce the posterior probability.

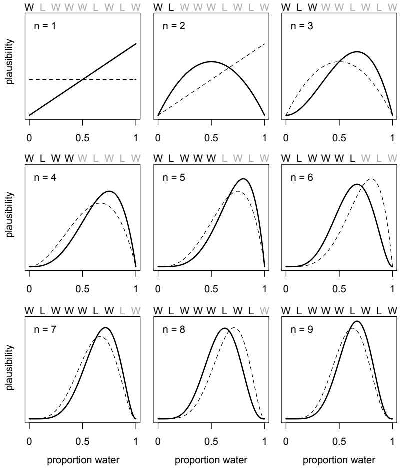


We present the first observation to the model.

$$P(\pi|n=1, w=1) \propto \pi^1(1-\pi)^0 = \pi \tag{2.10}$$

Source: McElreath (2020), Figure 2.5

Data can be presented to the model in any order, or all at once.



Source: McElreath (2020), Figure 2.5

- An analytical approach to compute the posterior distribution is often impossible.
- Typically, numerical techniques are needed to approximate the posterior distribution.
  - Grid approximation (computationally very expensive)
  - Quadratic (or Laplace) approximation (limited): assumes that the posterior distribution is approximately normal
  - Markov chain Monte Carlo (MCMC)
- Choice of numerical technique may influence inferences.

2.3.2 Grid approximation

McElreath (2020), Ch. 3

- We can achieve an approximation of the continuous posterior distribution by considering only a finite grid of parameter values.
- Compute the posterior at each particular value of a parameter π' by multiplying the prior probability of π' by the likelihood at *pip*$'$.
- Procedure:
  1. Define the grid.
  2. Compute the value of the prior at each parameter value on the grid.
  3. Compute the likelihood at each parameter value.
  4. Compute the unstandardized posterior at each parameter value.
  5. Standardize the posterior.

Bayesian updating

1. Define the grid.

```
grid <- seq( from=0, to=1, length.out=5 )
```

```
## [1] 0.00 0.25 0.50 0.75 1.00
```

2. Compute the value of the prior at each parameter value on the grid.

```
prior <- rep( 1, 5 )
```

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

3. Compute the likelihood at each parameter value.

```
likelihood <- dbinom( 6, size=9, prob=grid)
```

```
## [1] 0.000000000 0.008651733 0.164062500 0.233596802 0.000000000
```

4. Compute the unstandardized posterior at each parameter value.

```
unstd.posterior <- likelihood * prior
```

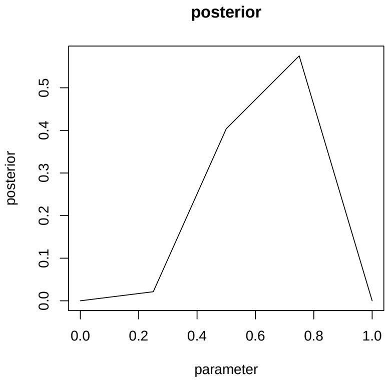
```
## [1] 0.000000000 0.008651733 0.164062500 0.233596802 0.000000000
```

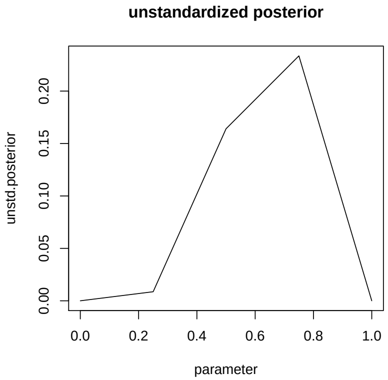
5. Standardize the posterior.

```
posterior <- unstd.posterior / sum(unstd.posterior)
```

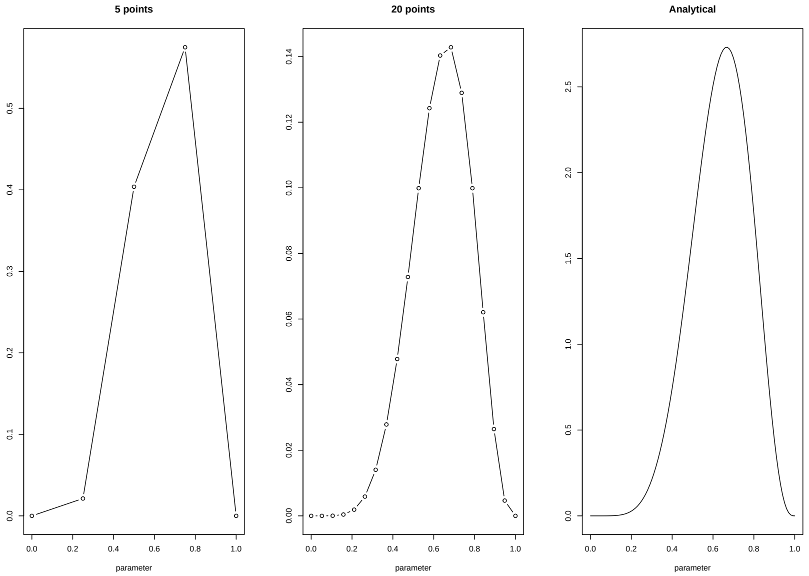
```
## [1] 0.00000000 0.02129338 0.40378549 0.57492114 0.00000000
```

Bayesian updating





Bayesian updating

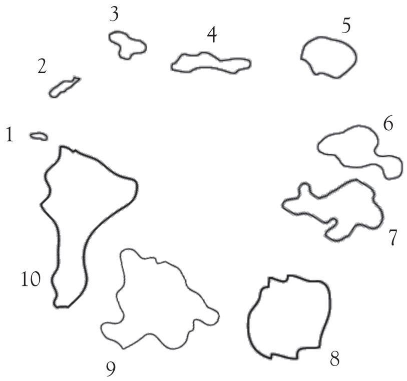


2.3.3 Markov chain Monte Carlo (MCMC)

McElreath (2020), Ch. 9; Kruschke (2015), Ch. 7

- MCMC draws samples of parameter values from the posterior distribution directly.
- The frequencies of the parameter values correspond to the posterior.
- We summarize the samples to describe the posterior.

- King wants to visit his islands. Number of visits should be in proportion to population size.
- Flip coin to choose proposal island on left or right.
- Find population of proposal island: pop_p
- Find population of current island: pop_c
- Move to proposal, with probability $\min\{\frac{\text{pop}_p}{\text{pop}_c}, 1\}$
- Repeat.



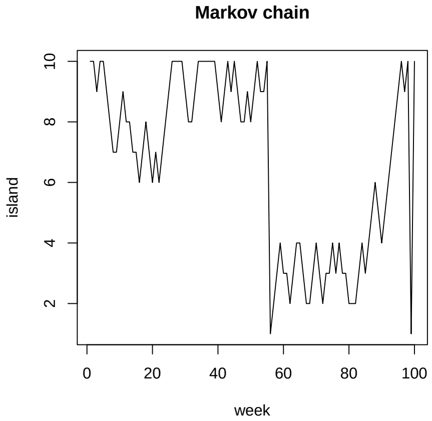
Source: McElreath (2016), Figure 8.1

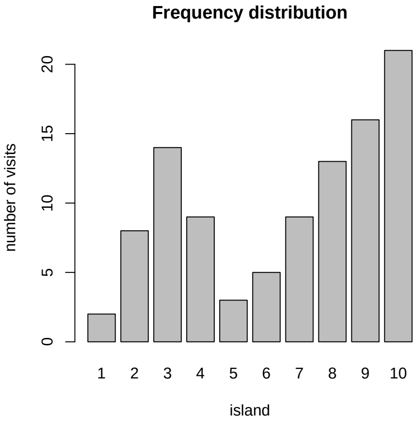
Metropolis algorithm

```
num_weeks <- 1e5
positions <- rep(0,num_weeks)
proposal  <- rep(0,num_weeks)
prob_move <- rep(0,num_weeks)
current <- 10
set.seed(42)
for ( i in 1:num_weeks ) {
    # current position
    positions[i] <- current
    # proposal
    proposal[i] <- current + sample( c(-1,1) , size=1 )
    if ( proposal[i] < 1 ) proposal[i] <- 10
    if ( proposal[i] > 10 ) proposal[i] <- 1
    # move?
    prob_move[i] <- proposal[i]/current
    current <- ifelse( runif(1) < prob_move[i] , proposal[i] , current )
}
```

```
d <- cbind(positions, proposal, prob_move)
d[1:10,]

##       positions proposal prob_move
##  [1,]        10        9 0.9000000
##  [2,]        10        9 0.9000000
##  [3,]         9       10 1.1111111
##  [4,]        10        1 0.1000000
##  [5,]        10        9 0.9000000
##  [6,]         9        8 0.8888889
##  [7,]         8        7 0.8750000
##  [8,]         7        6 0.8571429
##  [9,]         7        8 1.1428571
## [10,]         8        9 1.1250000
```

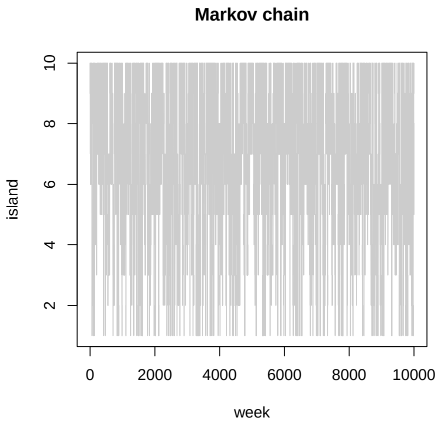
Markov chain after 100 draws

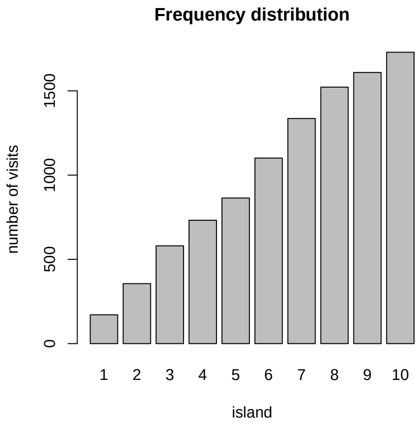




Markov chain after 10000 draws

Samples converge to true proportions, in the long run.





- Metropolis: proposal distribution is symmetric
- Metropolis-Hastings: allows asymmetric proposals
  - Better for parameters that have boundaries (e.g. standard deviation has boundary zero)
  - More efficient (in the sense that fewer samples are required)
- Gibbs sampling: variant of Metropolis-Hastings
  - Adaptive proposals (distribution of proposals is adjusted, depending on current parameter values)
  - Used in BUGS, JAGS
  - Limitations: Models fit efficiently only for conjugate priors, inefficient for big models
- Hamiltonian Monte Carlo (HMC) (e.g. Betancourt 2018)
  - Proposals are even more efficient, needs much fewer samples to describe the posterior distribution
  - Computational costly
  - Requires continuous parameters
- MCMC interactive gallery: `https://chi-feng.github.io/mcmc-demo/`

Example: HMC using Stan

Globe tossing model

Write the Stan code to a file named `Bernoulli.stan`.

```
data {
      int<lower=0> n;
      int<lower=0, upper=1> y[n];
    }
parameters {
      real<lower=0, upper=1> pi;
    }
model {
    pi ~ normal(0.5, 1);
    for (i in 1:n)
            y[i] ~ bernoulli(pi);
    }
```

Executing the R code below, you get samples from the posterior distribution.

```
library(rstan)

y <- c(1,0,1,1,1,0,1,0,1)
n <- length(y)

fit1 <- stan(file="Bernoulli.stan",
            data = list(n=n, y=y),
            pars = c("pi"),
            iter = 4000,
            warmup = 1000,
            chains = 4)
```

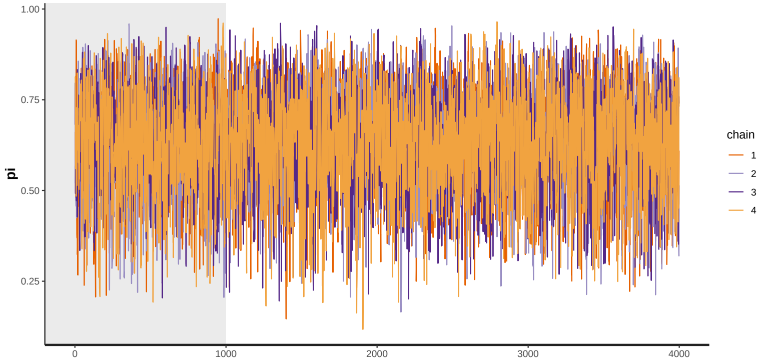
Result:

```
> print(fit1, pars=c("pi"), probs = c(0.10, 0.9))

   mean se_mean   sd  10%  90% n_eff Rhat
pi 0.64       0 0.14 0.45 0.82  4631    1
```

- How many samples (iterations) do we need?
  - Effective number of samples, `n_eff`
    - Estimate of the number of independent samples from the posterior distribution. (Markov chains are typically autocorrelated.)
    - $\texttt{n\_eff}/(\texttt{iter} - \texttt{warmup}) < 0.1$ may indicate problems.
    - In a regression, $\texttt{n\_eff} \geqslant 200$ to estimate posterior mean.
  - Warmup samples:
    - Improve sampling from the posterior by adapting tuning parameters.
    - Not from the posterior.
    - Not used for inference.
    - Not burn-in.
- How many chains do you need?
  - Use more than one chain for checking.
  - All chains should converge.
  - Inspect Gelman-Rubin convergence diagnostic: `Rhat`. Should be ≈ 1.

Check the chain using a traceplot

```
traceplot(fit1, pars = "pi", inc_warmup = TRUE)
```



- Do the chains have a stable mean?
- Does the chain mix well? Are samples highly autocorrelated?

Bad chains

Stan code to estimate mean and standard deviation of a normal outcome:

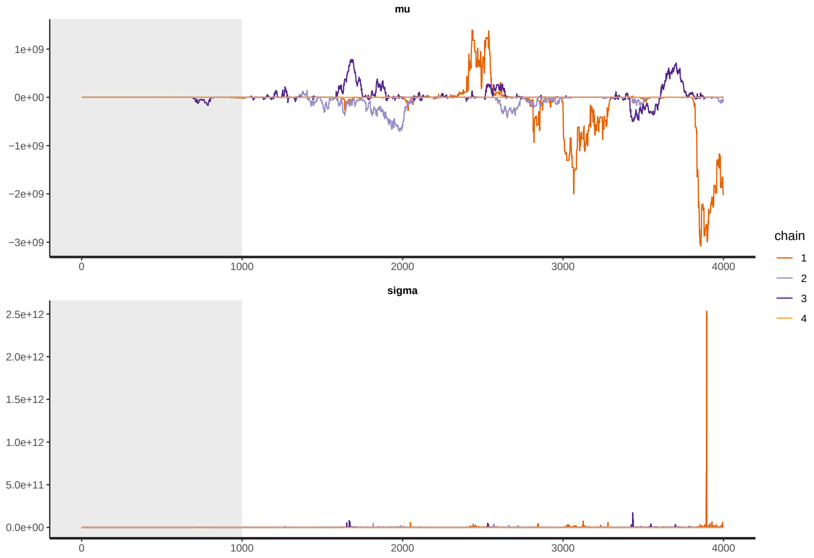
```
data {
    int<lower=1> n;
    vector[n] y;
}
parameters {
    real mu;
    real<lower=0> sigma;
}
model {
    // priors: parameters declared without constraints are
    // given a uniform prior on (- inf, +inf) by default

    // likelihood
    for (i in 1:n) {
        y[i] ~ normal(mu, sigma);
    }
}
```

Bayesian updating

```
y <- c(-1,1)
n <- length(y)
fit <- stan(file="normal_flat.stan",
            data = list(n=n, y=y),
            pars = c("mu", "sigma"),
            iter = 4000,
            warmup = 1000,
            chains = 4
)
```

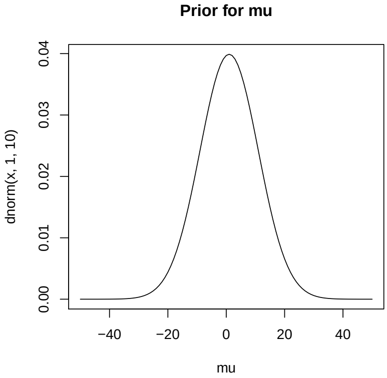
```
           mean   se_mean          sd           10%       90% n_eff Rhat
mu    -48321698  43559404   329153232 -147910120.92  18834677    57 1.11
sigma 900818432 461857074 25035255420         31.55 761216686  2938 1.00
```

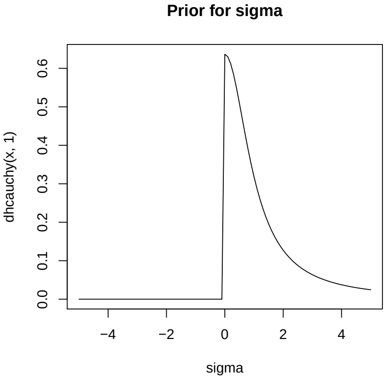
Bayesian updating



- Problem results from flat prior (and insufficient data).
- Solution: use (weakly) informative priors that exclude infinity

$$\mu \sim N(1, 10) \tag{2.11}$$
$$\sigma \sim \text{half Cauchy}(0, 1) \tag{2.12}$$



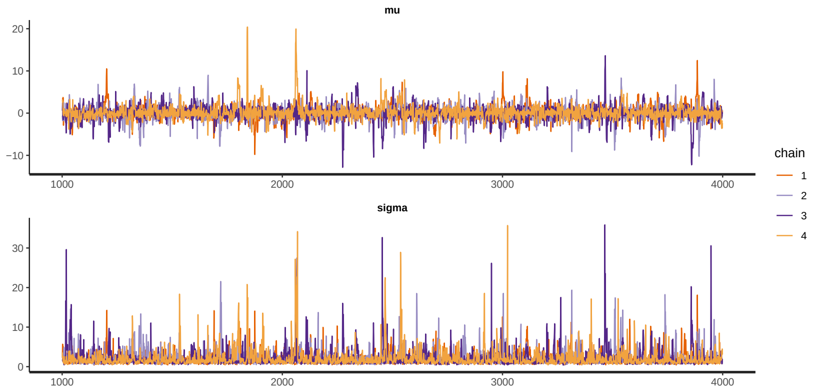


Bayesian updating

```
data {
    int<lower=1> n;
    vector[n] y;
}
parameters {
    real mu;
    real<lower=0> sigma;
}
model {
    // priors
^^Imu ~ normal(1, 10);
^^Isigma ~ cauchy(0, 1);

    // likelihood
    for (i in 1:n) {
        y[i] ~ normal(mu, sigma);
    }
}
```

Bayesian updating

```
       mean se_mean   sd   10%  90% n_eff Rhat
mu     0.04    0.04 1.69 -1.56 1.68  2299    1
sigma  2.07    0.04 2.05  0.79 3.74  2382    1
```



Outline

2.4 Interpreting the posterior distribution

We analyze the samples from the globe tossing example (object `fit1`).

Extract post-warmup samples:

```
> draws <- as.array(fit1)
> dim(draws) # (iter - warmup) x chains x parameters/generated-quantities array
[1] 3000    4    2
> dimnames(draws)
$chains
[1] "chain:1" "chain:2" "chain:3" "chain:4"

$parameters
[1] "pi"   "lp__"
```

Inspect post-warmup samples in R object `draws`:

```
> draws[1:10,1:4,1]
          chains
iterations   chain:1   chain:2   chain:3   chain:4
      [1,] 0.6031528 0.7451515 0.6668081 0.8274667
      [2,] 0.5381914 0.7799567 0.7169043 0.6100482
      [3,] 0.5701217 0.8165593 0.6842442 0.6100482
      [4,] 0.8151679 0.7759657 0.8350839 0.6817880
      [5,] 0.8151919 0.8078835 0.6947183 0.4548109
      [6,] 0.6231729 0.7892111 0.7968371 0.4766859
      [7,] 0.5603192 0.7548966 0.8114312 0.4508476
      [8,] 0.4914767 0.7200964 0.7486141 0.6481601
      [9,] 0.6447773 0.6907242 0.6498707 0.4323452
     [10,] 0.8251227 0.6778853 0.6622768 0.7079519
```

Intervals of defined boundaries

We use the samples (in the R object `draws`) to describe the posterior.

- Example: what is the posterior probability that $\pi < 0.5$?

```
sum( draws[,,1] < 0.5 ) / 12000
[1] 0.1699167
```

- Example: what is the posterior probability that $\pi > 0.5$ and $\pi < 0.8$?

```
> sum( draws[,,1] > 0.5 & draws[,,1] < 0.8 ) / 12000
[1] 0.7075
```

Intervals of defined mass (credible interval)

- The credible interval reports two parameter values that contain between them a specified amount of posterior probability.
- Percentile intervals assign equal probability mass to each tail.
- Example: Which range of parameter values contains 89% of the posterior probability?

```
> quantile( draws[,,1] , c( 0.055 , 0.945 ) )
     5.5%     94.5%
0.4052391 0.8458696
```

- Example: Which range of parameter values contains 99% of the posterior probability?

```
quantile( draws[,,1] , c( 0.005 , 0.995 ) )
     0.5%     99.5%
0.2764785 0.9157665
```

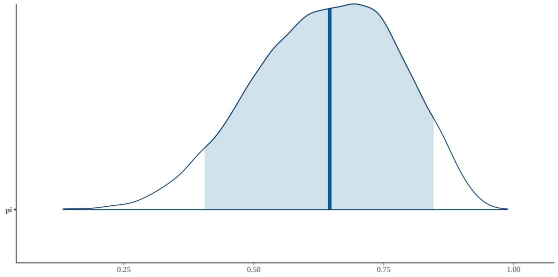
Interpreting the posterior distribution

- Stan output:

```
> print(fit1, pars=c("pi"), probs = c(0.005, 0.055, 0.945, 0.995))
   mean se_mean   sd 0.5% 5.5% 94.5% 99.5% n_eff Rhat
pi 0.64       0 0.14 0.28 0.41  0.85  0.92  4235    1
```

- Using R package `bayesplot` to visualize the posterior 89% interval estimates from MCMC draws:

```
mcmc_areas( draws, regex_pars = "pi", prob = 0.89, prob_outer = 1 )
```



Point estimates

- Maximum a posteriori (MAP) estimate: parameter value with highest posterior probability

```
chainmode(draws[,,1]) # requires: library(rethinking)
[1] 0.6924287
```

- Mean and median also summarize the posterior:

```
> print(fit1, pars=c("pi"), probs = c(0.005, 0.055, 0.5, 0.945, 0.995))

   mean se_mean   sd 0.5% 5.5%  50% 94.5% 99.5% n_eff Rhat
pi 0.64       0 0.14 0.28 0.41 0.65  0.85  0.92  4235    1
```

Outline

2.5 Bayesian workflow

- Define a joint probability model of all data and parameters.
- Prior predictive checking: What do priors imply about the data our model can generate?
- Use fake data to estimate the model: can you recover the parameters from simulated data?
- Use real data to fit the model and check the algorithm diagnostics.
- Run posterior predictive checking.
- Model comparison (e.g., via cross-validation)

Model checking

- Necessary to supervise and criticize model.
- Do the inferences from the model make sense?
- Do different chains converge to a common posterior distribution?
- Do chains have stable mean and variance?
- Are draws highly autocorrelated?
- Are inferences very sensitive to changes in assumptions?
- Posterior predictive checking: data generated under the model should look similar to the observed data.

Prior predictive checks

- **Prior predictive check**: We use the prior distributions to simulate hypothetical data from the model. Then, we check whether the simulated data are plausible and consistent with domain expertise.

Prior predictive checks require the following steps

1. Randomly draw a parameter set $\tilde{\theta}$ from the prior:

$$\tilde{\theta} \sim p(\theta).$$

2. Use the parameter set $\tilde{\theta}$ to simulate hypothetical data $\tilde{y}$ from the model:

$$\tilde{y} \sim p(y|\tilde{\theta}).$$

3. Compute summary statistics of the simulated data.

Example: a Gaussian model of height

McElreath (2020), Ch. 4.3 and 4.4

- Goal is to model the relationship between height (y) and weight (x).

$$
\begin{aligned}
y_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha + \beta(x_i - \bar{x}) \\
\alpha &\sim \text{Normal}(178, 20) \\
\beta &\sim \text{Normal}(0, 10) \\
\sigma &\sim \text{Uniform}(0, 50)
\end{aligned}
$$

- The intercept α is the expected height when $x_i = \bar{x}$.
- The slope β is the change in expected height when x_i changes by 1 unit.

Simulation using only the priors

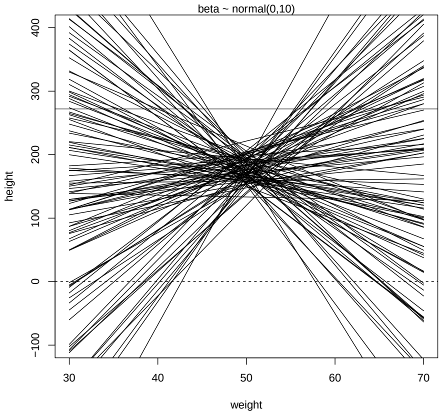
- We simulate heights from the model using only the priors.
- Randomly draw parameters $\tilde{\alpha}$ and $\tilde{\beta}$ from the priors:

```
set.seed(2971)
N <-  100 # number of simulations
a <- rnorm( N, 178, 20)
b <- rnorm( N, 0, 10 )
```

- Simulate hypothetical regression lines:

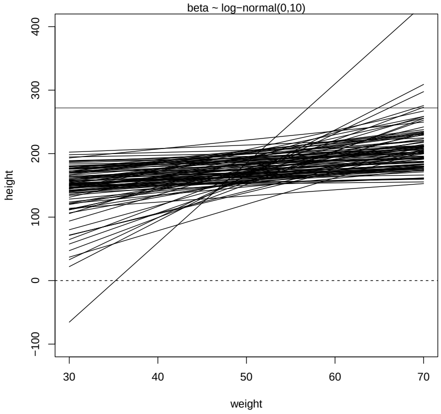
```
pdf( file =  "fig/ppc_1.pdf" )
plot( NULL , xlim=c(30, 70) , ylim=c(-100,400) ,
    xlab="weight" , ylab="height" )
abline( h=0 , lty=2 )
abline( h=272 , lty=1 , lwd=0.5 )
mtext( "beta ~ normal(0,10)" )
xbar <- 50
for ( i in 1:N ) curve( a[i] + b[i]*(x - xbar) ,
    from=30 , to=70 , add=TRUE )
dev.off()
```

Bayesian workflow



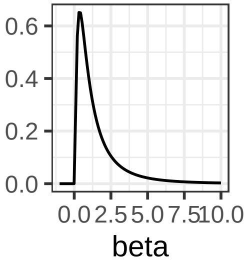
- Negative relationships are not reasonable.
- Magnitudes are partly inconsistent with domain knowledge.

Bayesian workflow



- Here, we use a log-normal prior for β:

$$\beta \sim \text{Log-Normal}(0,\ 1)$$



```r
set.seed(2971)
N <- 100
a <- rnorm( N, 178, 20)
b <- rlnorm( N, 0, 1)
```

References

Betancourt, M. (2018). A conceptual introduction to Hamiltonian Monte Carlo. ArXiv e-prints 1701.02434.

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian Data Analysis. Third Edition*. CRC Press, Boca Raton.

Gelman, A., Simpson, D., and Betancourt, M. (2017). The Prior Can Often Only Be Understood in the Context of the Likelihood. *Entropy*, 19(10):555.

Kruschke, J. K. (2015). *Doing Bayesian Data Analysis. A Tutorial with R, JAGS, and Stan*. Elsevier, Amsterdam.

McElreath, R. (2016). *Statistical Rethinking. A Bayesian Course with Examples in R and Stan*. Chapman & Hall/CRC Press, Boca Raton.

McElreath, R. (2020). *Statistical Rethinking. A Bayesian Course with Examples in R and Stan*. Chapman & Hall/CRC Press, Boca Raton.