Model checking

Dr. Jarad Niemi

STAT 544 - Iowa State University

February 20, 2024

Outline

We assume $p(y|\theta)$ and $p(\theta)$, so it would be prudent to determine if these assumptions are reasonable.

- (Prior) sensitivity analysis
- Posterior predictive checks
  - Graphical checks
  - Posterior predictive pvalues

Prior sensitivity analysis

Since a prior specifies our prior belief, we may want to check to determine whether our conclusions would change if we held different prior beliefs. Suppose a particular scientific question can be boiled down to

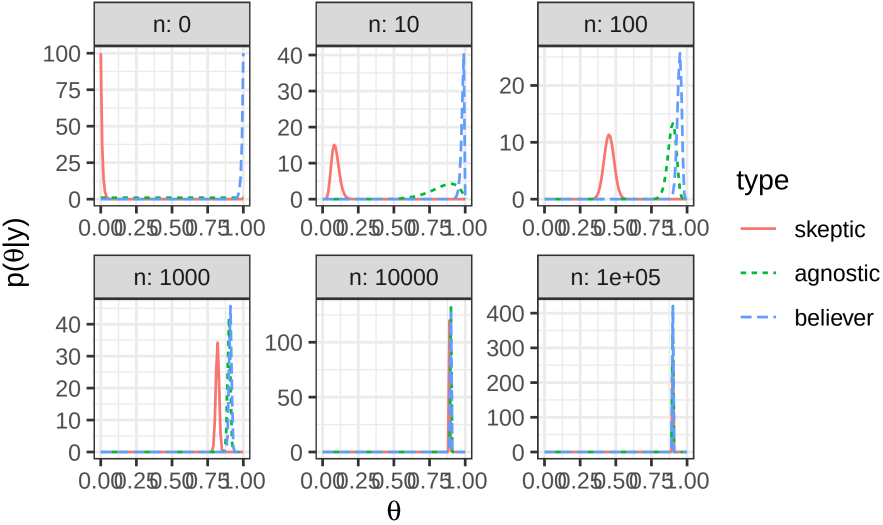
$$Y_i \stackrel{ind}{\sim} Ber(\theta)$$

and that there is wide disagreement about the value for θ such that the following might reasonably characterize different individual beliefs before the experiment is run:

- Skeptic: $\theta \sim Be(1, 100)$
- Agnostic: $\theta \sim Be(1, 1)$
- Believer: $\theta \sim Be(100, 1)$

An experiment is run and the posterior under these different priors are compared.

Posteriors


n: 0
n: 10
n: 100
n: 1000
n: 10000
n: 1e+05
p(θ|y)
θ
type
skeptic
agnostic
believer

Hierarchical variance prior

Recall the normal hierarchical model

$$y_i \stackrel{ind}{\sim} N(\theta_i, s_i^2), \quad \theta_i \stackrel{ind}{\sim} N(\mu, \tau^2)$$

which results in the posterior distribution for τ of

$$p(\tau|y) \propto p(\tau) V_\mu^{1/2} \prod_{i=1}^{I} (s_i^2 + \tau^2)^{-1/2} \exp\left(-\frac{(y_i - \hat{\mu})^2}{2(s_i^2 + \tau^2)} \right)$$

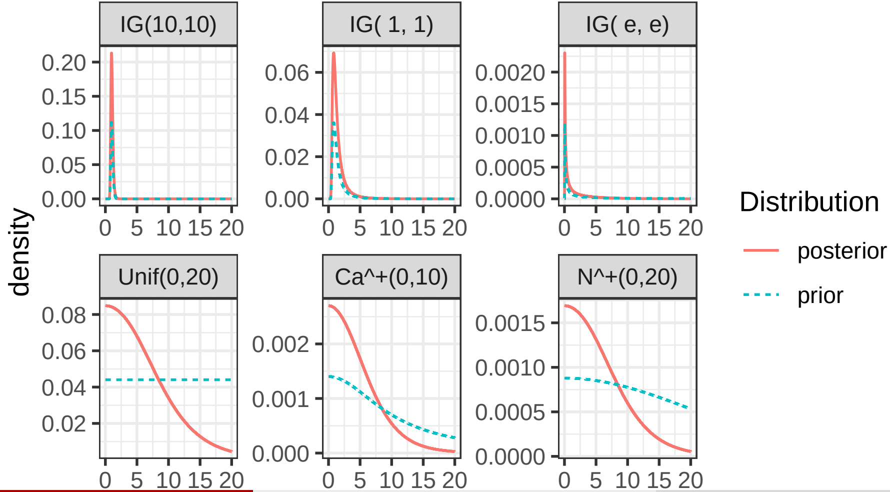
As an attempt to be non-informative, consider an $IG(\epsilon, \epsilon)$ prior for τ^2. As an alternative, consider $\tau \sim Unif(0, C)$ or $\tau \sim Ca^+(0, C)$ where C is problem specific, but is chosen to be relatively large for the particular problem.

The 8-schools example has the following data:

	1	2	3	4	5	6	7	8
y	28	8	-3	7	-1	1	18	12
s	15	10	16	11	9	11	10	18

Posterior for 8-schools example

Reproduction of Gelman 2006 and more:



Summary

For a default prior on a variance (σ^2) or standard deviation (σ), use

1. Easy to remember.
   - Half-Cauchy on the standard deviation ($\sigma \sim Ca^+(0, C)$)
   - Half-normal on the standard deviation ($\sigma \sim N^+(0, C)$)
2. Harder to remember
   - Data-level variance
     - Use default prior ($p(\sigma^2) \propto 1/\sigma^2$)
   - Hierarchical standard deviation
     - Use uniform prior (Unif$(0, C)$) if there are enough reps (5 or more) of that parameter.
     - Use half-Cauchy prior ($Ca^+(0, C)$) otherwise.
     - Use half-normal prior ($N^+(0, C)$) otherwise.

Summary - notes

When assigning the values for C

- For a uniform prior (Unif$(0, C)$) make sure C is large enough to capture any reasonable value for the standard deviation.
- For a half-Cauchy prior $(Ca^+(0, C))$ err on the side of making C too small since the heavy tails will let the data tell you if the standard deviation needs to be larger whereas a value of C that is too large will put too much weight toward large values of the standard deviation and make the prior more informative.
- For a half-normal prior $(Ca^+(0, C))$ err on the side of making C too large since the light tails will not allow the data to tell you if the standard deviation needs to be larger whereas a value of C that is too small will put too much weight toward small values of the standard deviation and make the prior more informative.

Posterior predictive checks

Let y^{rep} be a replication of y, then

$$p(y^{rep}|y) = \int p(y^{rep}|\theta, y)p(\theta|y)d\theta = \int p(y^{rep}|\theta)p(\theta|y)d\theta.$$

where y is the observed data and θ are the model parameters.

To simulate a full replication:

1. Simulate $\theta^{(j)} \sim p(\theta|y)$ and
2. Simulate $y^{rep,j} \sim p(y|\theta^{(j)})$.

To assess model adequacy:

- Compare plots of replicated data to the observed data.
- Calculate posterior predictive pvalues.

Airline accident data

Let

- y_i be the number of fatal accidents in year i
- x_i be the number of 100 million miles flown in year i

Consider the model

$$Y_i \stackrel{ind}{\sim} Po(x_i\lambda) \qquad p(\lambda) \propto 1/\sqrt{\lambda}.$$

	year	fatal_accidents	passenger_deaths	death_rate	miles_flown
1	1976	24	734	0	3863
2	1977	25	516	0	4300
3	1978	31	754	0	5027
4	1979	31	877	0	5481
5	1980	22	814	0	5814
6	1981	21	362	0	6033
7	1982	26	764	0	5877
8	1983	20	809	0	6223
9	1984	16	223	0	7433
10	1985	22	1066	0	7107

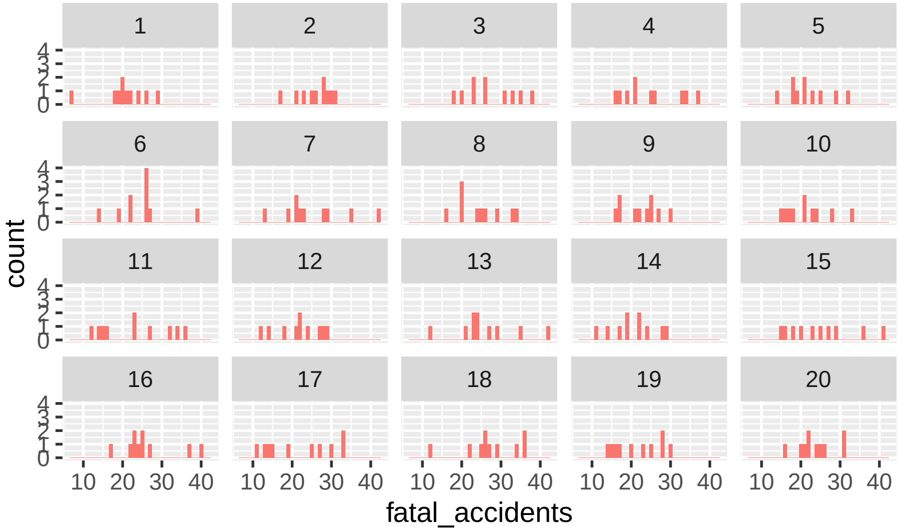
Posterior replications of the data

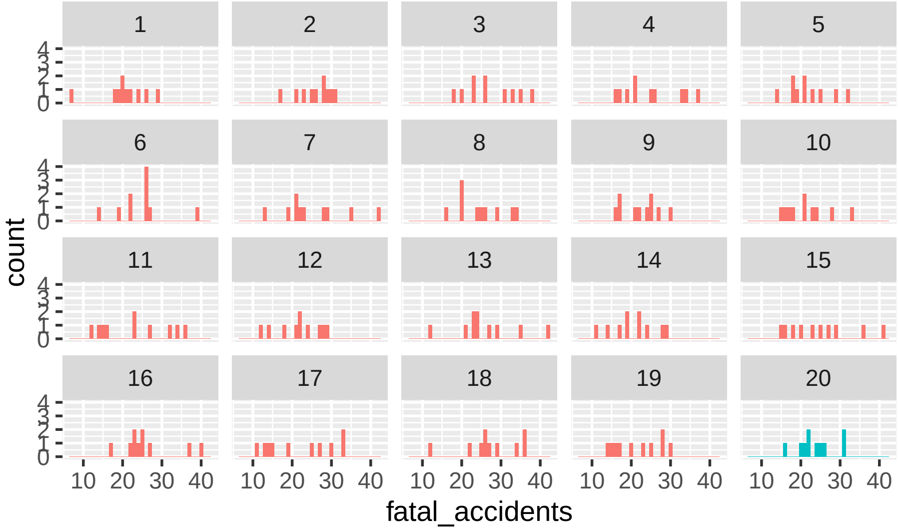
Under Jeffreys prior, the posterior is

$$\lambda|y \sim Ga(0.5 + n\overline{y}, n\overline{x}).$$

So to obtain a replication of the data, do the following

1. $\lambda^{(j)} \sim Ga(0.5 + n\overline{y}, n\overline{x})$ and
2. $y_i^{rep,j} \stackrel{ind}{\sim} Po(x_i\lambda^{(j)})$ for $i = 1, \ldots, n$.


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
count
0 1 2 3 4
10 20 30 40
fatal_accidents


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
count
0
1
2
3
4
10 20 30 40
fatal_accidents

How might this model not accurately represent the data?

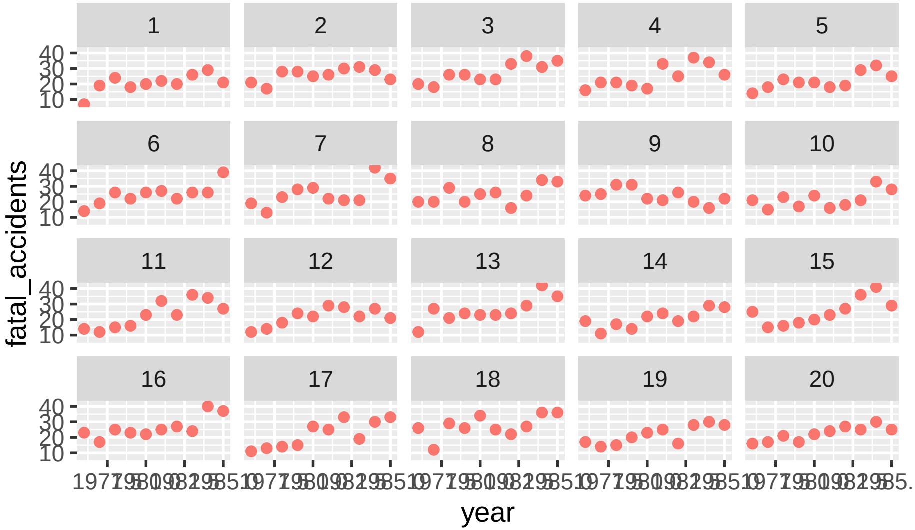
Let

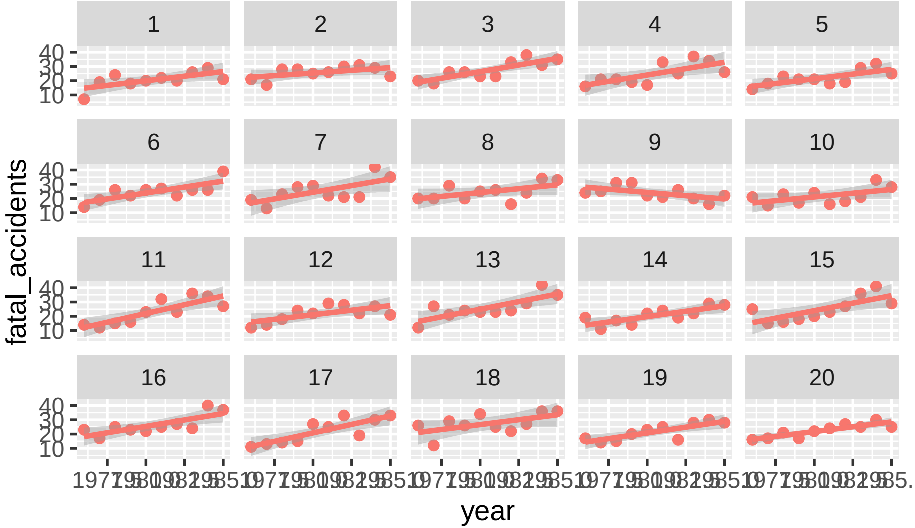
- y_i be the number of fatal accidents in year i
- x_i be the number of 100 million miles flown in year i

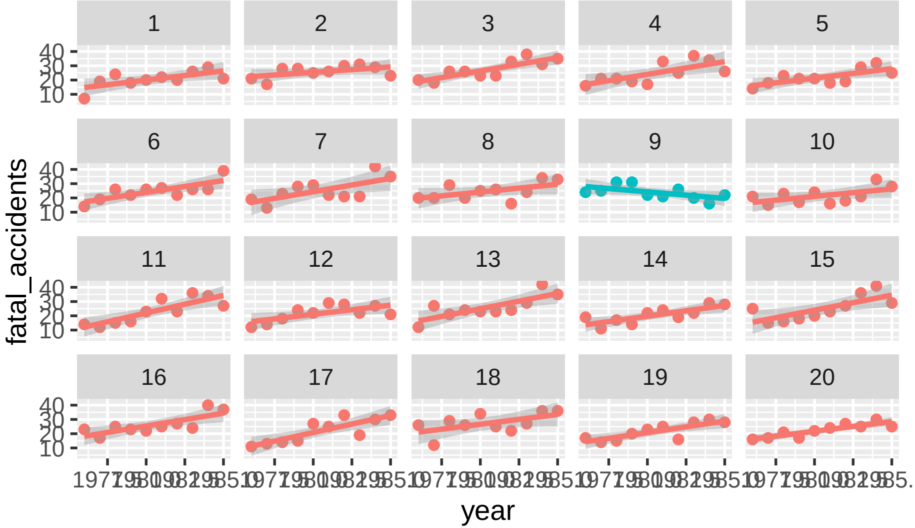
Consider the model

$$Y_i \stackrel{ind}{\sim} Po(x_i\lambda) \qquad p(\lambda) \propto 1/\sqrt{\lambda}.$$

	year	fatal_accidents	passenger_deaths	death_rate	miles_flown	.n
1	1976	24	734	0	3863	20
2	1977	25	516	0	4300	20
3	1978	31	754	0	5027	20
4	1979	31	877	0	5481	20
5	1980	22	814	0	5814	20
6	1981	21	362	0	6033	20
7	1982	26	764	0	5877	20
8	1983	20	809	0	6223	20
9	1984	16	223	0	7433	20
10	1985	22	1066	0	7107	20


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
40
30
20
10
fatal_accidents
year


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
40
30
20
10
fatal_accidents
year


1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
40
30
20
10
fatal_accidents
year

Posterior predictive pvalues

To quantify the discrepancy between observed and replicated data:

1. Define a test statistic $T(y,\theta)$.
2. Define the posterior predictive pvalue

$$p_B = P(T(y^{rep},\theta) \ge T(y,\theta)|y)$$

where y^{rep} and θ are random. Typically this pvalue is calculated via simulation, i.e.

$$\begin{array}{rl} p_B & = E_{y^{rep},\theta}[\mathrm{I}(T(y^{rep},\theta) \ge T(y,\theta))|y] \\ & = \int\int \mathrm{I}(T(y^{rep},\theta) \ge T(y,\theta))p(y^{rep}|\theta)p(\theta|y)dy^{rep}d\theta \\ & \approx \frac{1}{J}\sum_{j=1}^{J} \mathrm{I}(T(y^{rep,j},\theta^{(j)}) \ge T(y,\theta^{(j)})) \end{array}$$

where $\theta^{(j)} \sim p(\theta|y)$ and $y^{rep,j} \sim p(y|\theta^{(j)})$.

Small or large pvalues are (possible) cause for concern.

Posterior predictive pvalue for slope

Let

$$Y_i^{obs} = \beta_0^{obs} + \beta_1^{obs}\, i$$

where

- Y_i^{obs} is the observed number of fatal accidents in year i and
- β_1^{obs} be the test statistic.

Now, generate replicate data y^{rep} and fit

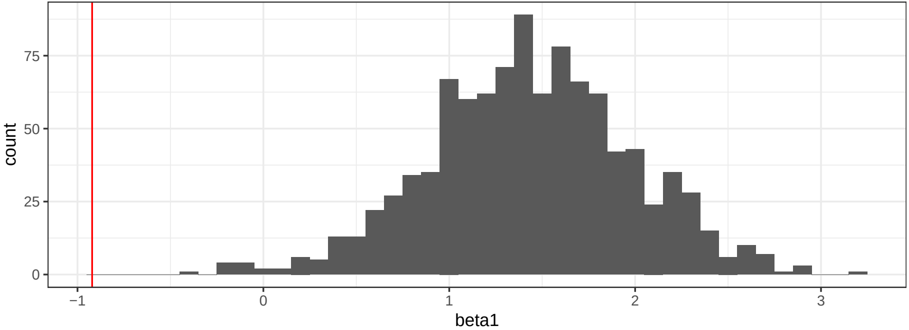
$$Y_i^{rep} = \beta_0^{rep} + \beta_1^{rep} i.$$

Now compare β_1^{obs} to the distribution of β_1^{rep}.

```r
mean(rep$beta1 > observed_slope)
```

```
[1] 1
```

```r
ggplot(rep, aes(x=beta1)) + geom_histogram(binwidth=0.1) +
  geom_vline(xintercept=observed_slope, color="red") +
  theme_bw()
```



Consider a linear model for the λ_i

Consider the model

$$\begin{aligned} Y_i &\stackrel{ind}{\sim} Po(x_i\lambda_i) \\ \log(\lambda_i) &= \beta_0 + \beta_1 i \end{aligned}$$

where

- Y_i is the number of fatal accidents in year i
- x_i is the number of 100 million miles flown in year i
- λ_i is the accident rate in year i

Here the λ_i are a deterministic function of year, but (possibly) different each year.

Stan linear model for accident rate

```
model = "
data {
  int<lower=0> n;
  int<lower=0> y[n];
  vector<lower=0>[n] x;
}
transformed data {
  vector[n] log_x;
  log_x = log(x); # both x and logx need to be vectors
}
parameters {
  real beta[2];
}
transformed parameters {
  vector[n] log_lambda;
  for (i in 1:n) log_lambda[i] = beta[1] + beta[2]*i;
}
model {
  y ~ poisson_log(log_x + log_lambda); # _log indicates mean on log scale
}
"
```

Stan run

```
m <- stan_model(model_code = model) # compile model
r <- sampling(m,
              list(n = nrow(d),
                   y = d$fatal_accidents,
                   x = d$miles_flown),
              iter = 10000)
```

```
SAMPLING FOR MODEL 'anon_model' NOW (CHAIN 1).
Chain 1:
Chain 1: Gradient evaluation took 1.8e-05 seconds
Chain 1: 1000 transitions using 10 leapfrog steps per transition would take 0.18 seconds.
Chain 1: Adjust your expectations accordingly!
Chain 1:
Chain 1:
Chain 1: Iteration:    1 / 10000 [  0%]  (Warmup)
Chain 1: Iteration: 1000 / 10000 [ 10%]  (Warmup)
Chain 1: Iteration: 2000 / 10000 [ 20%]  (Warmup)
Chain 1: Iteration: 3000 / 10000 [ 30%]  (Warmup)
Chain 1: Iteration: 4000 / 10000 [ 40%]  (Warmup)
Chain 1: Iteration: 5000 / 10000 [ 50%]  (Warmup)
Chain 1: Iteration: 5001 / 10000 [ 50%]  (Sampling)
Chain 1: Iteration: 6000 / 10000 [ 60%]  (Sampling)
Chain 1: Iteration: 7000 / 10000 [ 70%]  (Sampling)
Chain 1: Iteration: 8000 / 10000 [ 80%]  (Sampling)
Chain 1: Iteration: 9000 / 10000 [ 90%]  (Sampling)
Chain 1: Iteration: 10000 / 10000 [100%]  (Sampling)
Chain 1:
```

```
r


Inference for Stan model: anon_model.
4 chains, each with iter=10000; warmup=5000; thin=1;
post-warmup draws per chain=5000, total post-warmup draws=20000.

                  mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
beta[1]          -4.90    0.00 0.14  -5.17  -4.99  -4.90  -4.81  -4.64  4566    1
beta[2]          -0.10    0.00 0.02  -0.15  -0.12  -0.10  -0.09  -0.06  4615    1
log_lambda[1]    -5.01    0.00 0.12  -5.24  -5.08  -5.01  -4.93  -4.78  4767    1
log_lambda[2]    -5.11    0.00 0.10  -5.31  -5.18  -5.11  -5.04  -4.92  5189    1
log_lambda[3]    -5.21    0.00 0.08  -5.38  -5.27  -5.21  -5.16  -5.06  6165    1
log_lambda[4]    -5.32    0.00 0.07  -5.46  -5.37  -5.32  -5.27  -5.18  8724    1
log_lambda[5]    -5.42    0.00 0.07  -5.55  -5.47  -5.42  -5.38  -5.30 15547    1
log_lambda[6]    -5.53    0.00 0.07  -5.66  -5.57  -5.53  -5.48  -5.40 20770    1
log_lambda[7]    -5.63    0.00 0.08  -5.79  -5.68  -5.63  -5.58  -5.48 15734    1
log_lambda[8]    -5.74    0.00 0.09  -5.92  -5.80  -5.73  -5.67  -5.56 10744    1
log_lambda[9]    -5.84    0.00 0.11  -6.06  -5.91  -5.84  -5.76  -5.63  8451    1
log_lambda[10]   -5.94    0.00 0.13  -6.20  -6.03  -5.94  -5.86  -5.70  7271    1
lp__            516.84    0.01 1.01 514.16 516.44 517.15 517.56 517.83  6193    1

Samples were drawn using NUTS(diag_e) at Tue Feb 20 15:22:24 2024.
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).
```

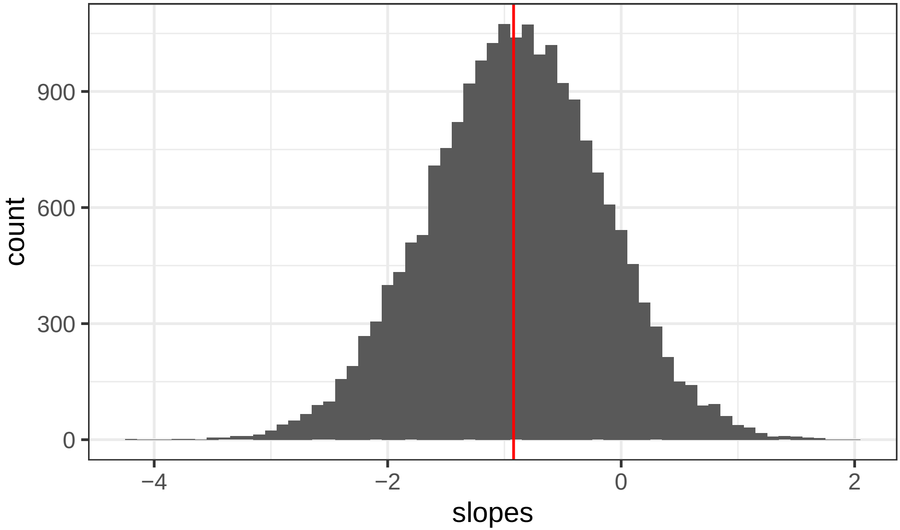
Posterior predictive pvalue: slope

```r
# Need to update to eliminate plyr dependency
library("plyr")
log_lambda <- extract(r, "log_lambda")[["log_lambda"]]
reps <- plyr::adply(log_lambda, 1, function(a) {
  d <- data.frame(
    yrep = rpois(nrow(d), d$miles_flown * exp(a)),
    year = 1:10)
})

rep_slopes = plyr::daply(reps, .(iterations), function(d) {
  slopes = coefficients(lm(yrep ~ year, d))[2]
})
```

```r
# Posterior predictive pvalue: slope
mean(rep_slopes > observed_slope)
```

```
[1] 0.50915
```


count
900
600
300
0
−4
−2
0
2
slopes