Parameter estimation (cont.)

Dr. Jarad Niemi

STAT 544 - Iowa State University

January 22, 2018

Outline

- Normal model, unknown mean
  - Jeffreys prior
  - Natural conjugate prior
  - Posterior
- Normal model, unknown variance
  - Jeffreys prior
  - Natural conjugate prior
  - Posterior
- JAGS/Stan
  - Binomial model, unknown probability
  - Normal model, unknown mean
  - Normal model, unknown variance

Jeffreys prior for μ

Theorem

If $Y_i \overset{iid}{\sim} N(\mu, s^2)$ *(s^2 known), Jeffreys prior for μ is* $p(\mu) \propto 1$.

Proof.

Since the normal distribution with unknown mean is an exponential family, use Casella & Berger Lemma 7.3.11

$$
\begin{aligned}
-E_y\left[\frac{\partial^2}{\partial \mu^2}\log p(y|\mu)\right] &= -E_y\left[\frac{\partial^2}{\partial \mu^2}\left(-\log(2\pi s^2)/2 - \frac{1}{2s^2}\sum_{i=1}^n (y_i-\mu)^2\right)\right] \\
&= -E_y\left[\frac{\partial^2}{\partial \mu^2}\left(-\log(2\pi s^2)/2 - \frac{1}{2s^2}\left(\sum_{i=1}^n y_i^2 - 2\mu n\overline{y} + n\mu^2\right)\right)\right] \\
&= -E_y\left[\frac{\partial}{\partial \mu}\left(-\frac{1}{2s^2}(-2n\overline{y}+2n\mu)\right)\right] \\
&= -E_y\left[-\frac{1}{2s^2}(2n)\right] \\
&= n/s^2 \\
p(\mu) &\propto \sqrt{|\mathcal{I}(\mu)|} = \sqrt{n/s^2} \\
&\propto 1
\end{aligned}
$$

So Jeffreys prior for μ is $p(\mu) \propto 1$. □

Posterior propriety

Since $\int_{-\infty}^{\infty} 1 d\mu$ is not finite, we need to check posterior propriety.

Theorem

For $n > 0$, the posterior for a normal mean (known variance) using Jeffreys prior is proper.

Proof.

The posterior is

$$\begin{aligned} p(\mu|y) &\propto p(y|\mu)p(\mu) \\ &\propto \exp\left(-\frac{1}{2s^2}\sum_{i=1}^{n}(y_i-\mu)^2\right) \times 1 \\ &\propto \exp\left(-\frac{1}{2s^2}\left[-2\mu n\overline{y} + n\mu^2\right]\right) \\ &= \exp\left(-\frac{1}{2s^2/n}\left[\mu^2 - 2\mu\overline{y}\right]\right). \end{aligned}$$

This is the kernel of a normal distribution with mean $\overline{y}$ and variance s^2/n which is proper if $n > 0$. □

Natural conjugate prior

Let $Y_i \overset{iid}{\sim} N(\mu, s^2)$ with s^2 known. The likelihood is

$$\begin{aligned} L(\mu) &= \exp\left(-\frac{1}{2s^2/n}\left[\mu^2 - 2\mu\overline{y}\right]\right) \\ &\propto \exp\left(-\frac{1}{2}\left[\frac{n}{s^2}\mu^2 - 2\mu\frac{n}{s^2}\overline{y}\right]\right) \end{aligned}$$

This is the kernel of a normal distribution, so the natural conjugate prior is $\mu \sim N(m, C)$.

$$\begin{aligned} p(\mu|y) &\propto p(y|\mu)p(\mu) = L(\mu)p(\mu) \\ &= \exp\left(-\frac{1}{2}\left[\frac{n}{s^2}\mu^2 - 2\mu\frac{n}{s^2}\overline{y}\right]\right)\exp\left(-\frac{1}{2}\left[\frac{1}{C}\mu^2 - 2\mu\frac{1}{C}m\right]\right) \\ &= \exp\left(-\frac{1}{2}\left[\left(\frac{1}{C} + \frac{n}{s^2}\right)\mu^2 - 2\mu\left(\frac{1}{C}m + \frac{n}{s^2}\overline{y}\right)\right]\right) \\ &= \exp\left(-\frac{1}{2\left(\frac{1}{C}+\frac{n}{s^2}\right)^{-1}}\left[\mu^2 - 2\mu\frac{1}{\left(\frac{1}{C}+\frac{n}{s^2}\right)}\left(\frac{1}{C}m + \frac{n}{s^2}\overline{y}\right)\right]\right) \end{aligned}$$

This is the kernel of a $N(m', C')$ where

$$C' = \left[C^{-1} + n/s^2\right]^{-1} \qquad m' = C'\left[C^{-1}m + n/s^2\overline{y}\right]$$

Normal mean posterior comments

Let $P = 1/C$, $P' = 1/C'$, and $Q = 1/s^2$ be the relevant precisions (inverse variances), then

- The posterior precision is the sum of the prior and observation precisions.

$$P' = P + \sum_{i=1}^{n} Q = P + nQ.$$

- The posterior mean is a precision weighted average of the prior and data.

$$\begin{aligned} m' &= \frac{1}{P'}\left[Pm + nQ\overline{y}\right] \\ &= \frac{P}{P'}m + n\frac{Q}{P'}\overline{y} \\ &= \frac{P}{P'}m + \sum_{i=1}^{n}\frac{Q}{P'}y_i \end{aligned}$$

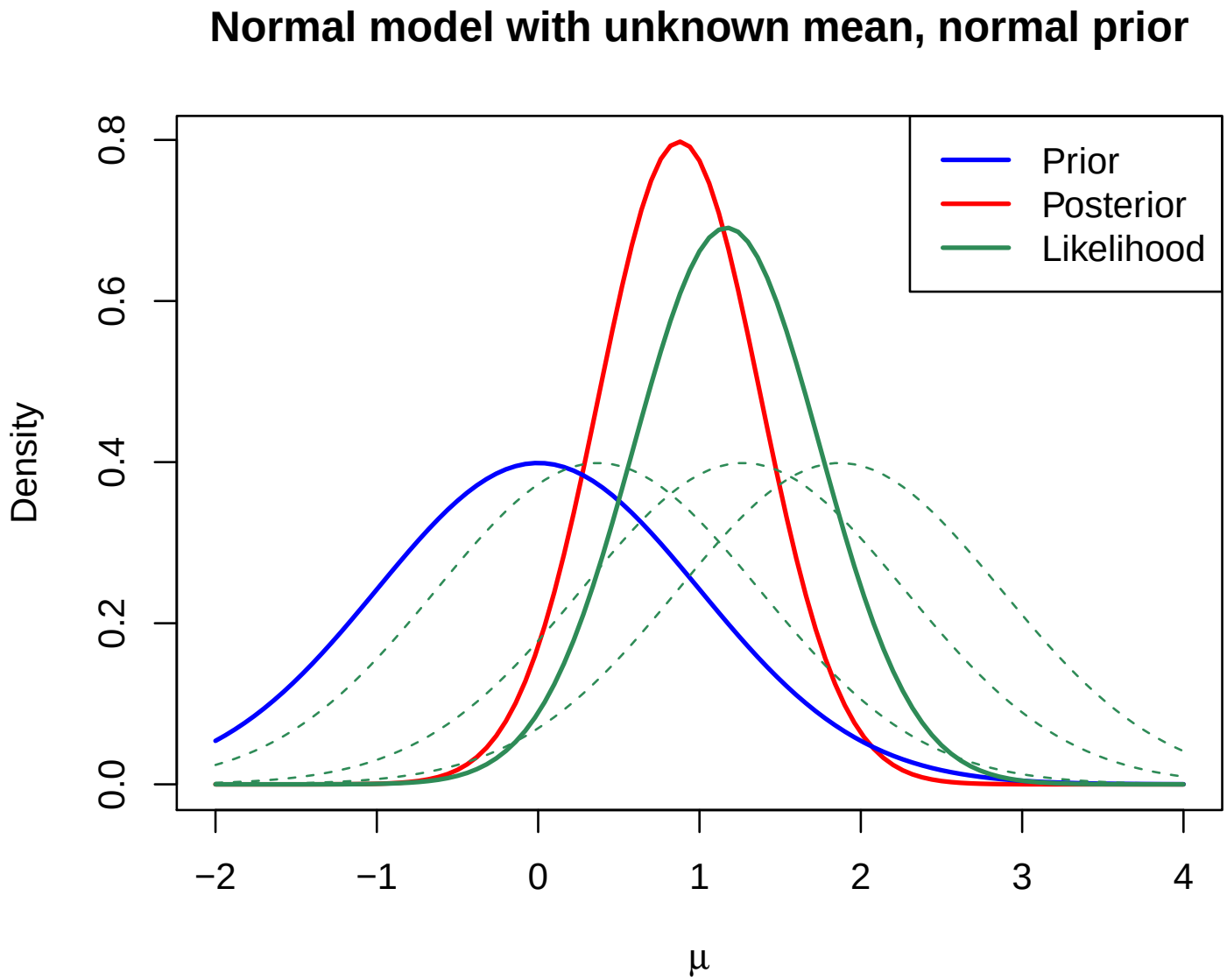
- Jeffreys prior/posterior are the limits of the conjugate prior/posterior as $C \to \infty$, i.e.

$$\lim_{C\to\infty} N(m, C) \xrightarrow{d} \propto 1 \qquad \lim_{C\to\infty} N(m', C') \xrightarrow{d} N(\overline{y}, s^2/n)$$

Example

Consider $Y_i \overset{ind}{\sim} N(\mu, 1)$ and $\mu \sim N(0, 1)$.

```
# Prior
m = 0
C = 1; P = 1/C

# Data
mu = 1
s2 = 1; Q = 1/s2
n = 3
set.seed(6); (y = rnorm(n,mu,sqrt(1/Q)))


[1] 1.2696060 0.3700146 1.8686598


# Posterior
nQ = n*Q
Pp = P+nQ
mp = (P*m+nQ*mean(y))/Pp
```


Normal model with unknown mean, normal prior
Prior
Posterior
Likelihood
Density
0.0
0.2
0.4
0.6
0.8
−2
−1
0
1
2
3
4
μ

Theorem

If $Y_i \overset{iid}{\sim} N(m, \sigma^2)$ *(*m *known), Jeffreys prior for* σ^2 *is* $p(\sigma^2) \propto 1/\sigma^2$.

Proof.

Since the normal distribution with unknown variance is an exponential family, use Casella & Berger Lemma 7.3.11.

$$
\begin{aligned}
-E_y\left[\frac{\partial^2}{\partial(\sigma^2)^2}\log p(y|\sigma^2)\right] &= -E_y\left[\frac{\partial^2}{\partial(\sigma^2)^2} - n\log(2\pi\sigma^2)/2 - \frac{1}{2\sigma^2}\sum_{i=1}^n (y_i - m)^2\right] \\
&= -E_y\left[\frac{\partial}{\partial(\sigma^2)} - \frac{n}{2\sigma^2} + \frac{1}{2(\sigma^2)^2}\sum_{i=1}^n (y_i - m)^2\right] \\
&= -E_y\left[\frac{n}{2(\sigma^2)^2} - \frac{1}{(\sigma^2)^3}\sum_{i=1}^n (y_i - m)^2\right] \\
&= -\frac{n}{2(\sigma^2)^2} + \frac{n}{(\sigma^2)^3}\sigma^2 \\
&= \frac{n}{2}(\sigma^2)^{-2} \\
p(\sigma^2) &\propto \sqrt{|\mathcal{I}(\sigma^2)|} \propto 1/\sigma^2
\end{aligned}
$$

So Jeffreys prior is $p(\sigma^2) \propto 1/\sigma^2$. □

Posterior propriety

Since $\int_0^\infty 1/\sigma^2 d\sigma^2$ is not finite, we need to check posterior propriety.

Theorem

For $n > 0$ and at least one $y_i \neq m$, the posterior for a normal variance (known mean) using Jeffreys prior is proper.

Proof.

The posterior is

$$\begin{aligned} p(\sigma^2|y) &\propto p(y|\sigma^2)p(\sigma^2) \\ &= (2\pi\sigma^2)^{-n/2} \exp\left(-\tfrac{1}{2\sigma^2}\textstyle\sum_{i=1}^n [y_i - m]^2\right)(\sigma^2)^{-1} \\ &\propto (\sigma^2)^{-n/2-1} \exp\left(-\tfrac{1}{2\sigma^2}\textstyle\sum_{i=1}^n [y_i - m]^2\right) \end{aligned}$$

This is the kernel of an inverse gamma distribution with shape $n/2$ and scale $\sum_{i=1}^n [y_i - m]^2/2$ which will be proper so long as $n > 0$ and at least one $y_i \neq m$. □

Natural conjugate prior

Let $Y_i \overset{iid}{\sim} N(m, \sigma^2)$ with m known. The likelihood is

$$L(\sigma^2) \propto (\sigma^2)^{-n/2} \exp\left(-\tfrac{1}{2\sigma^2} \textstyle\sum_{i=1}^n [y_i - m]^2\right)$$

This is the kernel of an inverse gamma distribution, so the natural conjugate prior is $IG(a, b)$.

$$\begin{aligned} p(\sigma^2|y) &\propto p(y|\sigma^2)p(\sigma^2) \\ &= (\sigma^2)^{-n/2} \exp\left(-\tfrac{1}{2\sigma^2} \textstyle\sum_{i=1}^n [y_i - m]^2\right) (\sigma^2)^{-a-1} \exp(-b/\sigma^2) \\ &= (\sigma^2)^{-(a+n/2)-1} \exp\left(-\tfrac{1}{\sigma^2} \left[b + \textstyle\sum_{i=1}^n [y_i - m]^2/2\right]\right) \end{aligned}$$

This is the kernel of an inverse gamma distribution with shape $a + n/2$ and scale $b + \sum_{i=1}^n [y_i - m]^2/2$.

Example

Suppose $Y_i \overset{ind}{\sim} N(1, \sigma^2)$ and $\sigma^2 \sim IG(1, 1)$.

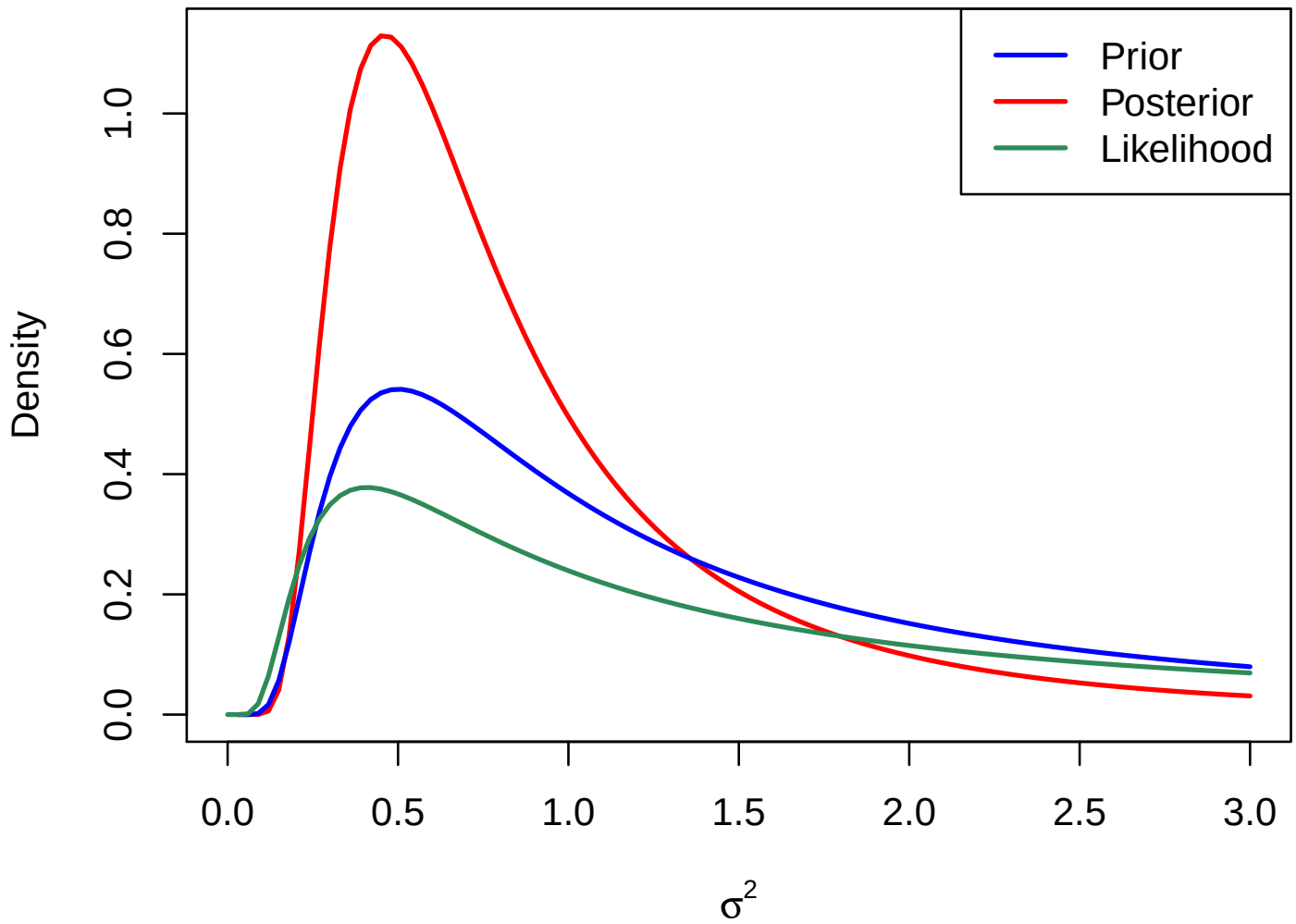
```
# Prior
a = b = 1

# Data
m = 1
n = length(y)
y
```

```
[1] 1.2696060 0.3700146 1.8686598
```

```
# Posterior
ap = a+n/2
(bp = b+sum((y-m)^2)/2)
```

```
[1] 1.612069
```

Normal model with unknown variance, inverse gamma prior



Summary

Suppose $Y_i \sim N(\mu, \sigma^2)$.

- μ unknown (σ^2 known)
  - Jeffreys prior: $p(\mu) \propto 1$ (think of this as $N(0, \infty)$)
  - Natural conjugate prior: $N(m, C)$
  - Posterior $N(m', C')$ with
    - $C' = [1/C + n\sigma^{-2}]^{-1}$
    - $m' = C'[m/C + n\sigma^{-2}\overline{y}]$
- σ^2 unknown (μ known)
  - Jeffreys prior: $p(\sigma^2) \propto 1/\sigma^2$ (think of this as $IG(0, 0)$)
  - Natural conjugate prior $IG(a, b)$
  - Posterior $IG\left(a + n/2, b + \sum_{i=1}^{n}(y_i - \mu)^2/2\right)$

JAGS

Just another Gibbs sampler (JAGS) "is a program for analysis of Bayesian hierarchical models using Markov Chain Monte Carlo (MCMC) simulation not wholly unlike BUGS." We will use JAGS through its R interface `rjags`.

The basic workflow when using `rjags` is

1. Define model and priors in a string
2. Assign data
3. Run JAGS, i.e. simulate from the posterior
4. Summarize as necessary, e.g. mean, median, credible intervals, etc

Binomial model

Let $Y \sim Bin(n, \theta)$ and $\theta \sim Be(1, 1)$ and we observe $y = 3$ successes out of $n = 10$ attempts.

```
model = "
model
{
  y     ~ dbin(theta,n)  # notice p then n
  theta ~ dbeta(a,b)
}
"

dat = list(n=10, y=3, a=1, b=1)

m = jags.model(textConnection(model), dat)


Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 1
   Unobserved stochastic nodes: 1
   Total graph size: 5

Initializing model


r = coda.samples(m, "theta", n.iter=1000)
```

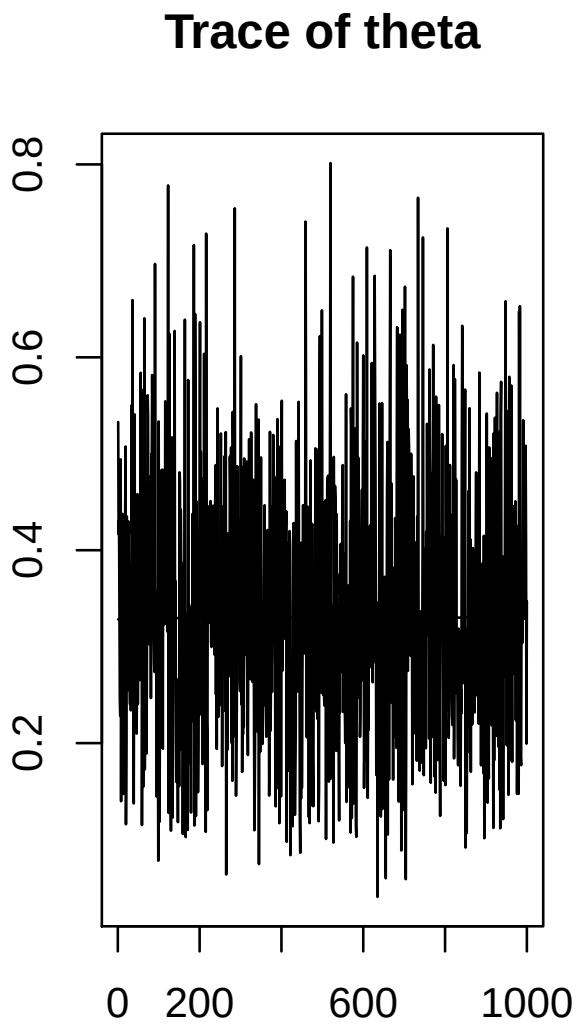
Binomial model

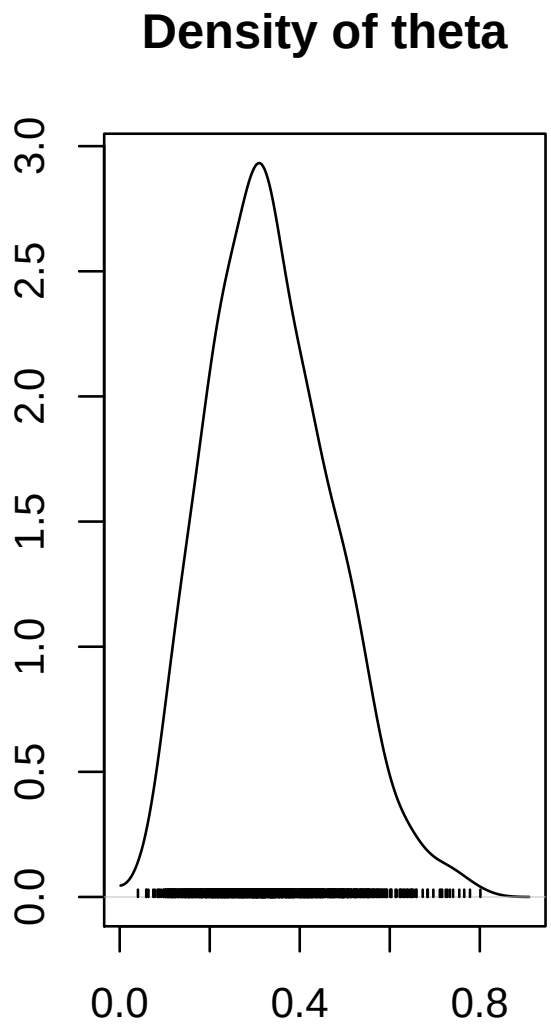
```
summary(r)
```

```
Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
       0.33700        0.13565         0.00429        0.00429

2. Quantiles for each variable:

  2.5%    25%    50%    75%  97.5%
0.1120 0.2353 0.3223 0.4247 0.6328
```

Binomial model

```
plot(r)
```





Normal model, unknown mean

Let $Y \sim N(\mu, s^2)$ and $\mu \sim N(0, 1)$.

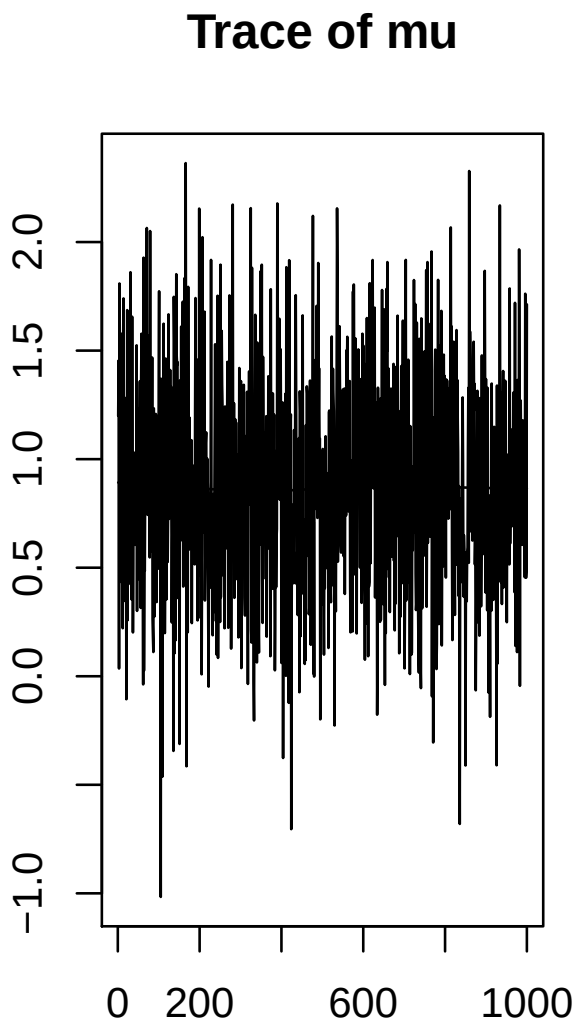
```
model = "
model
{
  for (i in 1:n) {         # iterate over observations
    y[i] ~ dnorm(mu,1/s2) # precision instead of variance
  }
  mu ~ dnorm(m,1/C)        # cannot use improper prior in JAGS
}
"

dat = list(m=0,C=1,s2=1,y=y)
dat$n = length(dat$y)

m = jags.model(textConnection(model), dat)


Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 3
   Unobserved stochastic nodes: 1
   Total graph size: 13

Initializing model


r = coda.samples(m, "mu", n.iter=1000)
```

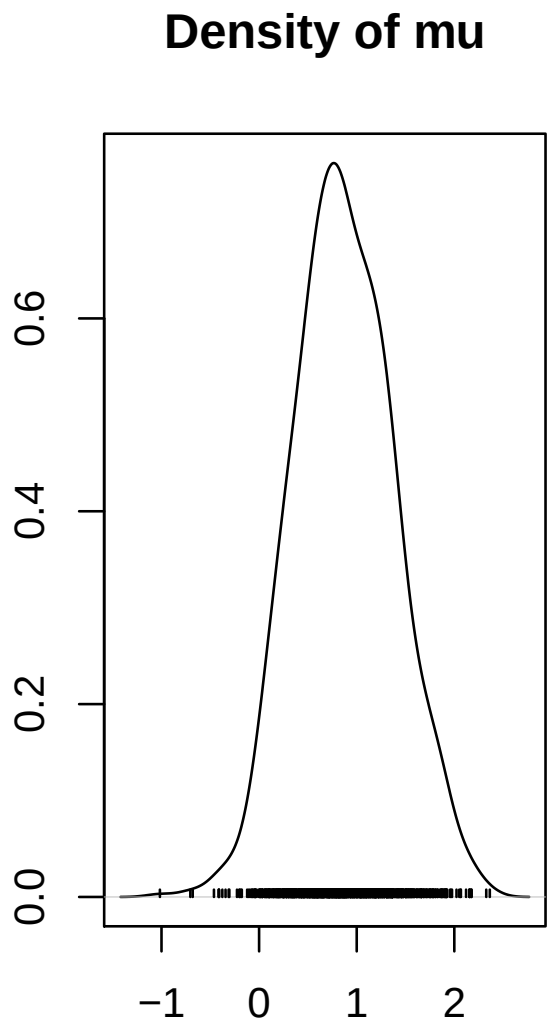
Normal model, unknown mean

```
summary(r)



Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
       0.87275        0.50505         0.01597        0.01672

2. Quantiles for each variable:

    2.5%      25%      50%      75%    97.5%
-0.03984  0.53097  0.84834  1.22078  1.89567
```

Normal model, unknown mean

```
plot(r)
```





Normal model, unknown variance

Let $Y \sim N(m, \sigma^2)$ and $\sigma^2 \sim IG(1, 1)$.

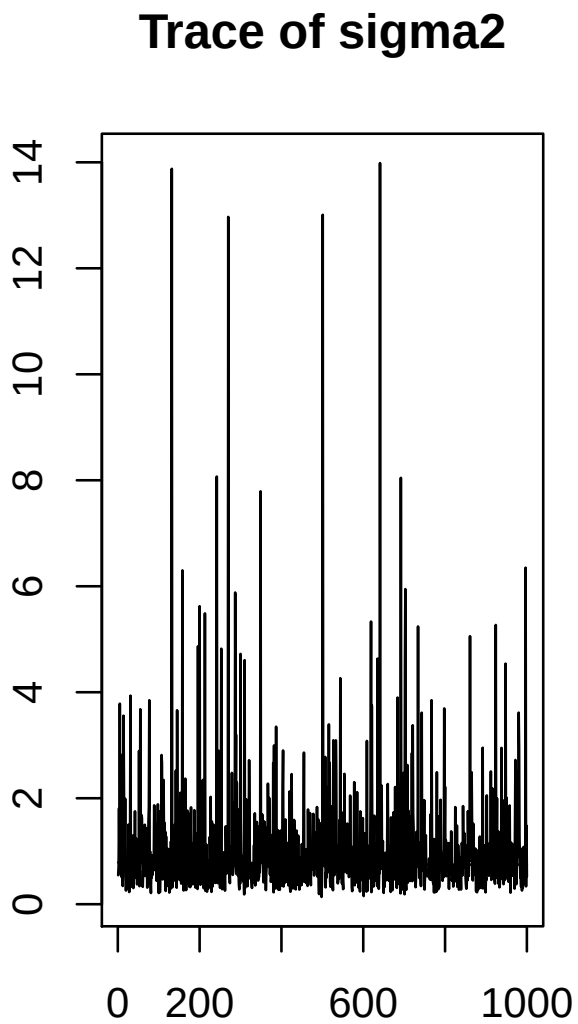
```
model = "
model
{
  for (i in 1:n) {
    y[i] ~ dnorm(m,tau) # precision instead of variance
  }
  tau ~ dgamma(a,b)     # Inverse gamma is not a built in distribution
  sigma2 <- 1/tau       # Functions of parameters
}
"

dat = list(m=1,a=1,b=1,y=y)
dat$n = length(dat$y)

m = jags.model(textConnection(model), dat)


Compiling model graph
   Resolving undeclared variables
   Allocating nodes
Graph information:
   Observed stochastic nodes: 3
   Unobserved stochastic nodes: 1
   Total graph size: 11

Initializing model


r = coda.samples(m, "sigma2", n.iter=1000)
```

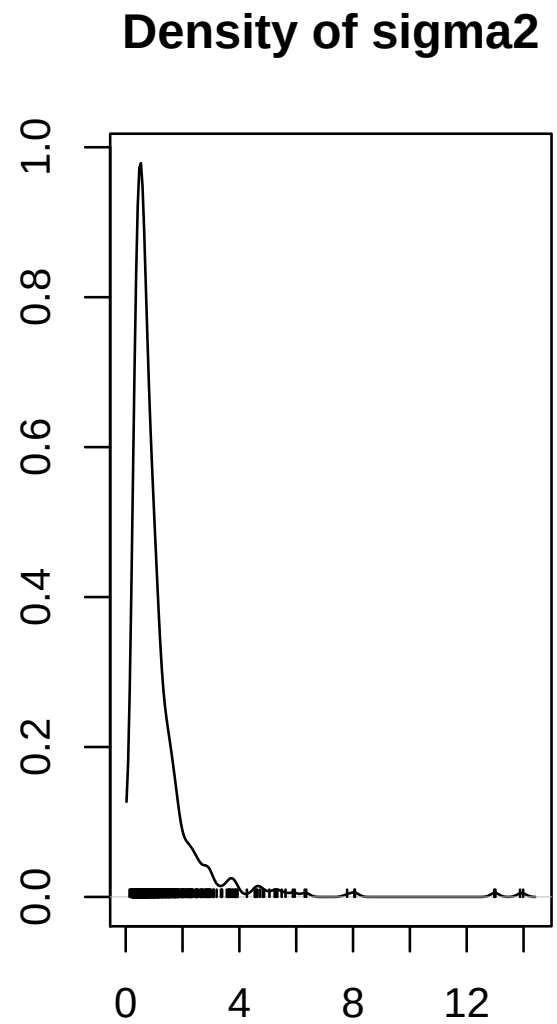
Normal model, unknown variance

```
summary(r)



Iterations = 1:1000
Thinning interval = 1
Number of chains = 1
Sample size per chain = 1000

1. Empirical mean and standard deviation for each variable,
   plus standard error of the mean:

          Mean             SD       Naive SE Time-series SE
       1.07882        1.23931         0.03919        0.03919

2. Quantiles for each variable:

  2.5%    25%    50%    75%  97.5%
0.2387 0.4857 0.7386 1.1968 3.9435
```

Normal model, unknown variance

```
plot(r)
```





Stan

Stan "is a probabilistic programming language implementing full Bayesian statistical inference." We will use Stan through its R interface `rstan`.

The basic workflow when using `rstan` is (exactly the same as for `rjags`)

1. Define model and priors in a string
2. Assign data
3. Run Stan, i.e. simulate from the posterior
4. Summarize as necessary, e.g. mean, median, credible intervals, etc

But, additional coding is required for Stan.

Stan - Binomial model

Let $Y \sim Bin(n, \theta)$ and $\theta \sim Be(1, 1)$.

```
model = "
data {
  int<lower=0> n;              # define range and type
  int<lower=0> a;              # and notice semicolons
  int<lower=0> b;
  int<lower=0,upper=n> y;
}
parameters {
  real<lower=0,upper=1> theta;
}
model {
  y ~ binomial(n,theta);
  theta ~ beta(a,b);
}
"

dat = list(n=10, y=3, a=1, b=1)

m = stan_model(model_code = model) # Only needs to be done once


In file included from filedaba6c72f03c.cpp:8:
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/src/st
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/stan/m
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/BH/include/boost/math/tool
```

Stan - Binomial model sampling

```
r = sampling(m, data=dat)
```

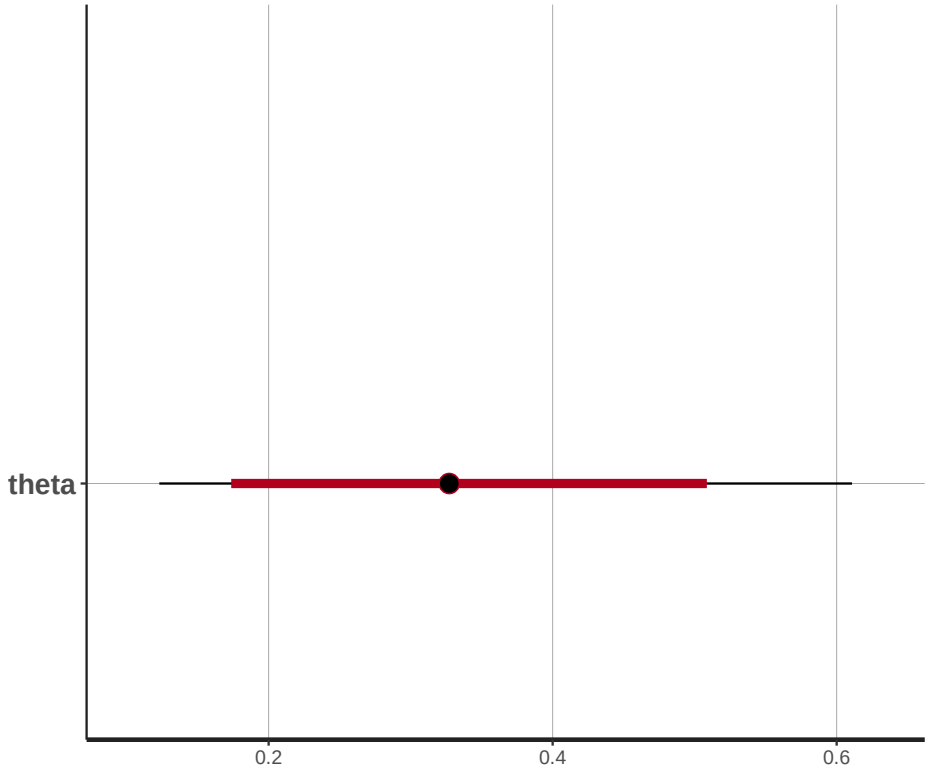
```
SAMPLING FOR MODEL '3f4017b7a8f5171e6969f13bccb0ff3a' NOW (CHAIN 1).

Gradient evaluation took 2e-05 seconds
1000 transitions using 10 leapfrog steps per transition would take 0.2 seconds.
Adjust your expectations accordingly!


Iteration:    1 / 2000 [  0%]  (Warmup)
Iteration:  200 / 2000 [ 10%]  (Warmup)
Iteration:  400 / 2000 [ 20%]  (Warmup)
Iteration:  600 / 2000 [ 30%]  (Warmup)
Iteration:  800 / 2000 [ 40%]  (Warmup)
Iteration: 1000 / 2000 [ 50%]  (Warmup)
Iteration: 1001 / 2000 [ 50%]  (Sampling)
Iteration: 1200 / 2000 [ 60%]  (Sampling)
Iteration: 1400 / 2000 [ 70%]  (Sampling)
Iteration: 1600 / 2000 [ 80%]  (Sampling)
Iteration: 1800 / 2000 [ 90%]  (Sampling)
Iteration: 2000 / 2000 [100%]  (Sampling)

 Elapsed Time: 0.021827 seconds (Warm-up)
               0.025925 seconds (Sampling)
               0.047752 seconds (Total)


SAMPLING FOR MODEL '3f4017b7a8f5171e6969f13bccb0ff3a' NOW (CHAIN 2).
```

Stan - Binomial model results

```
r


Inference for Stan model: 3f4017b7a8f5171e6969f13bccb0ff3a.
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
theta  0.34    0.00 0.13   0.12  0.24  0.33  0.42  0.61  1698    1
lp__  -8.14    0.02 0.69 -10.05 -8.30 -7.88 -7.69 -7.64  1617    1

Samples were drawn using NUTS(diag_e) at Thu Jan 18 15:17:00 2018.
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).
```

Stan - Binomial model

```
plot(r)
```



Normal model, unknown mean

Let $Y \sim N(\mu, s^2)$ and $\mu \sim N(0, 1)$.

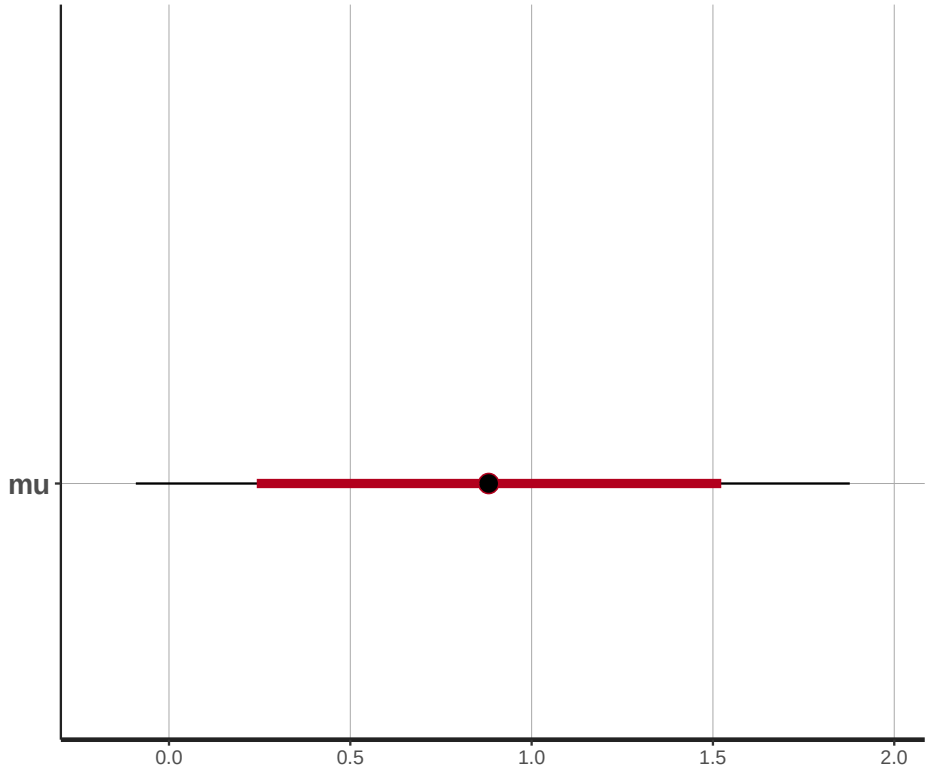
```
model = "
data {
  int<lower=0> n;
  real y[n];            # vector
  real<lower=0> s2;
  real m;
  real<lower=0> C;
}
transformed data {      # run once
  real<lower=0> s;
  real<lower=0> sqrtC;
  s     = sqrt(s2);
  sqrtC = sqrt(C);
}
parameters {
  real mu;              # if used alone, implies a uniform prior
}
model {
  y  ~ normal(mu,s);    # vectorized, i.e. assumed independent
  mu ~ normal(m,sqrtC); # standard deviation
}
"

dat = list(m=0,C=1,s2=1,y=y)
dat$n = length(dat$y)

m = stan_model(model_code = model)
```

Normal model, unknown mean

```
r


Inference for Stan model: f0a514f97c4dfe48a65c8d1fd93db68d.
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
mu     0.89    0.01 0.50 -0.09  0.55  0.88  1.22  1.88  1719    1
lp__ -1.58    0.02 0.71 -3.77 -1.75 -1.31 -1.13 -1.08  1614    1

Samples were drawn using NUTS(diag_e) at Thu Jan 18 15:18:14 2018.
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).
```

Normal model, unknown mean

```
plot(r)
```



Normal model, unknown variance

Let $Y \sim N(m, \sigma^2)$ and $\sigma^2 \sim IG(1, 1)$.

```
model = "
data {
  int<lower=0> n;
  real y[n];
  real m;
  real<lower=0> a;
  real<lower=0> b;
}
parameters {
  real<lower=0> sigma2;     # if used alone, implies a uniform prior on (0,Inf)
}
transformed parameters {    # deterministic function of parameters
  real<lower=0> sigma;
  sigma = sqrt(sigma2);
}
model {
  y  ~ normal(m,sigma);
  sigma2 ~ inv_gamma(a,b);  # built in inverse gamma distribution
}
"

dat = list(a=1,b=1,m=1,y=y)
dat$n = length(dat$y)

m = stan_model(model_code = model)


In file included from filedaba7d94e8c4.cpp:8:
In file included from /Library/Frameworks/R.framework/Versions/3.4/Resources/library/StanHeaders/include/src/st
```

Normal model, unknown variance

```
r


Inference for Stan model: 4039b3757fe245d08771e56962f9306f.
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
sigma2  1.08    0.04 1.17  0.25  0.50  0.74  1.21  3.98  1055    1
sigma   0.96    0.01 0.40  0.50  0.71  0.86  1.10  1.99  1165    1
lp__   -1.94    0.02 0.76 -4.17 -2.12 -1.63 -1.46 -1.40  1145    1

Samples were drawn using NUTS(diag_e) at Thu Jan 18 15:19:28 2018.
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).
```

Normal model, unknown variance

```
plot(r)
```