

# Bayesian Estimation Supersedes the t-Test

Mike Meredith and John Kruschke

June 29, 2015

## 1 Introduction

The BEST package provides a Bayesian alternative to a  $t$  test, providing much richer information about the samples and the difference in means than a simple  $p$  value.

Bayesian estimation for two groups provides complete distributions of credible values for the effect size, group means and their difference, standard deviations and their difference, and the normality of the data. For a single group, distributions for the mean, standard deviation and normality are provided. The method handles outliers.

The decision rule can accept the null value (unlike traditional  $t$  tests) when certainty in the estimate is high (unlike Bayesian model comparison using Bayes factors).

The package also provides methods to estimate statistical power for various research goals.

The code used for the computations is the same as that used in Kruschke (2013).

## 2 The Model

To accommodate outliers we describe the data with a distribution that has fatter tails than the normal distribution, namely the  $t$  distribution. (Note that we are using this as a convenient description of the data, not as a sampling distribution from which  $p$  values are derived.) The relative height of the tails of the  $t$  distribution is governed by the shape parameter  $\nu$ : when  $\nu$  is small, the distribution has heavy tails, and when it is large (e.g., 100), it is nearly normal. Here we refer to  $\nu$  as the normality parameter.

The data ( $y$ ) are assumed to be independent and identically distributed (i.i.d.) draws from a  $t$  distribution with different mean ( $\mu$ ) and standard deviation ( $\sigma$ ) for each population, and with a common normality parameter ( $\nu$ ), as indicated in the lower portion of Figure 1.

The priors used are minimally informative: normal priors with large standard deviation for ( $\mu$ ), broad uniform priors for ( $\sigma$ ), and a shifted-exponential prior for ( $\nu$ ), as shown in the upper part of Figure 1. Full details of the priors are given in Kruschke (2013).

For a general discussion see chapters 11 and 12 of Kruschke (2011).

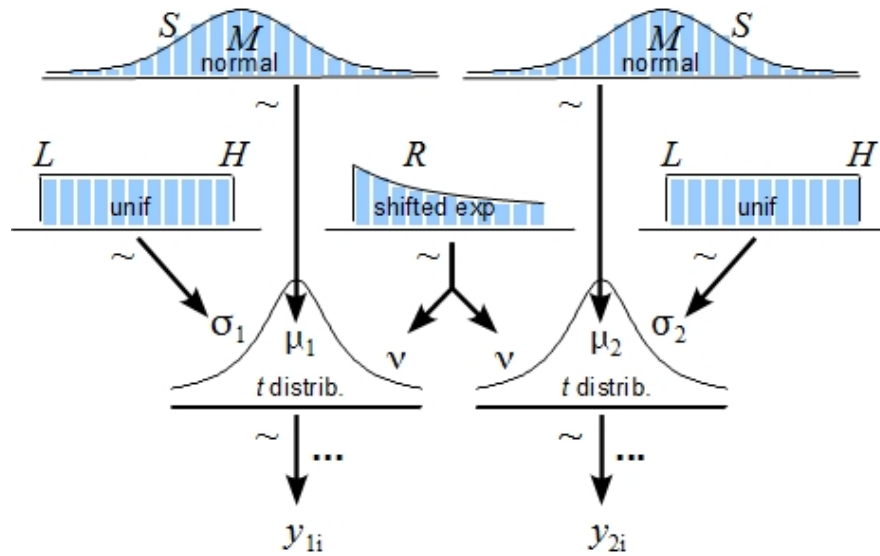


Figure 1: *Hierarchical diagram of the descriptive model for robust Bayesian estimation.*

### 3 Preparing to run BEST

BEST uses the JAGS package (Plummer, 2003) to produce samples from the posterior distribution of each parameter of interest. You will need to download JAGS from <http://sourceforge.net/projects/mcmc-jags/> and install it before running BEST.

BEST also requires the packages `jagsUI` and `coda`, which should normally be installed at the same time as package BEST if you use the `install.packages` function in R.

Once installed, we need to load the BEST package at the start of each R session, which will also load `jagsUI` and `coda` and link to JAGS:

```
> library(BEST)
```

### 4 An example with two groups

#### 4.1 Some example data

We will use hypothetical data for reaction times for two groups ( $N_1 = N_2 = 6$ ), Group 1 consumes a drug which may increase reaction times while Group 2 is a control group that consumes a placebo.

```
> y1 <- c(5.77, 5.33, 4.59, 4.33, 3.66, 4.48)
> y2 <- c(3.88, 3.55, 3.29, 2.59, 2.33, 3.59)
```

#### 4.2 Running the model

We run `BESTmcmc` and save the result in `BESTout`. We do not use parallel processing here, but if your machine has at least 4 cores, parallel processing cuts the time by 50

```
> BESTout <- BESTmcmc(y1, y2, parallel=FALSE)
```

```
Processing function input.....
```

```
Done.
```

```

Compiling model graph
  Resolving undeclared variables
  Allocating nodes
  Graph Size: 43

Initializing model

Adaptive phase, 500 iterations x 3 chains
If no progress bar appears JAGS has decided not to adapt

|+++++| 100%

Burn-in phase, 1000 iterations x 3 chains

|*****| 100%

Sampling from joint posterior, 33334 iterations x 3 chains

|*****| 100%

MCMC took 0.155 minutes.

```

### 4.3 Basic inferences

The default plot (Figure 2) is a histogram of the posterior distribution of the difference in means.

```
> plot(BESTout)
```

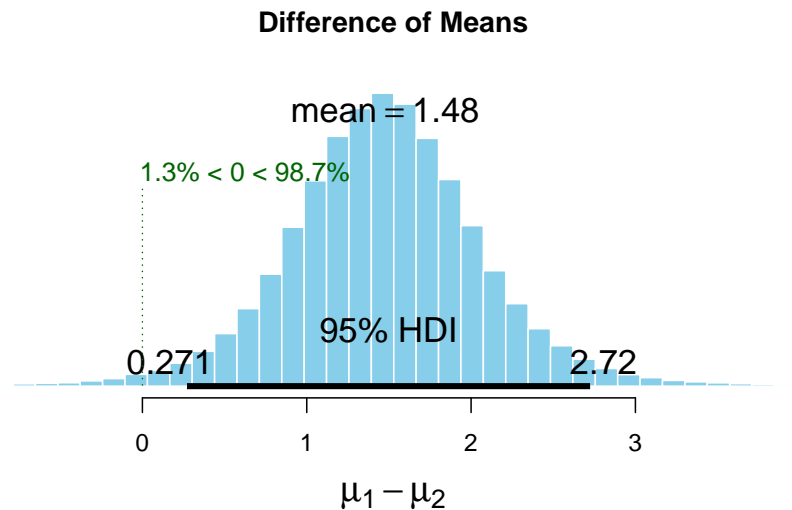


Figure 2: *Default plot: posterior probability of the difference in means.*

Also shown is the mean of the posterior probability, which is an appropriate point estimate of the true difference in means, the 95% Highest Density Interval (HDI), and the posterior

probability that the difference is greater than zero. The 95% HDI does not include zero, and the probability that the true value is greater than zero is shown as 98.7%. Compare this with the output from a  $t$  test:

```
> t.test(y1, y2)

Welch Two Sample t-test

data: y1 and y2
t = 3.7624, df = 9.6093, p-value = 0.003977
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 0.6020466 2.3746201
sample estimates:
mean of x mean of y
 4.693333  3.205000
```

Because we are dealing with a Bayesian posterior probability distribution, we can extract much more information:

- We can estimate the probability that the true difference in means is above (or below) an arbitrary *comparison value*. For example, an increase reaction time of 1 unit may indicate that users of the drug should not drive or operate equipment.
- The probability that the difference in reaction times is precisely zero is zero. More interesting is the probability that the difference may be too small to matter. We can define a *region of practical equivalence* (ROPE) around zero, and obtain the probability that the true value lies therein. For the reaction time example, a difference of  $\pm 0.1$  may be too small to matter.

```
> plot(BESTout, compVal=1, ROPE=c(-0.1,0.1))
```

The annotations in (Figure 3) show a high probability that the reaction time increase is  $> 1$ . In this case it's clear that the effect is large, but if most of the probability mass (say, 95%) lay within the ROPE, we would accept the null value for practical purposes.

BEST deals appropriately with differences in standard deviations between the samples and departures from normality due to outliers. We can check the difference in standard deviations or the normality parameter with `plot` (Figure 4).

```
> plot(BESTout, which="sd")
```

The `summary` method gives us more information on the parameters of interest, including derived parameters:

```
> summary(BESTout)
```

	mean	median	mode	HDI%	HDIlo	HDIup	compVal	%>compVal
mu1	4.694	4.690	4.7059	95	3.735	5.63		
mu2	3.217	3.222	3.2145	95	2.457	4.00		
muDiff	1.477	1.470	1.4990	95	0.271	2.72	0	98.7
sigma1	1.029	0.890	0.7430	95	0.368	2.04		
sigma2	0.837	0.728	0.6030	95	0.289	1.64		
sigmaDiff	0.192	0.152	0.0973	95	-1.149	1.64	0	64.1
nu	33.599	24.840	9.5519	95	1.000	92.25		
log10nu	1.362	1.395	1.4943	95	0.566	2.10		
effSz	1.700	1.680	1.5500	95	0.179	3.29	0	98.7

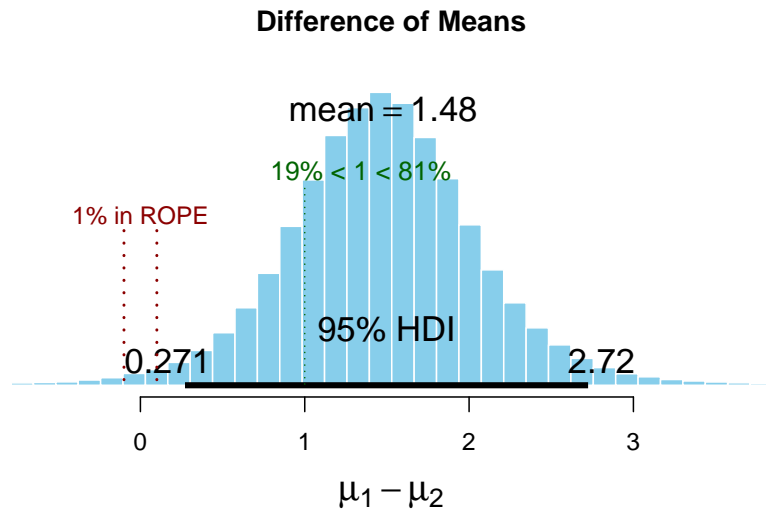


Figure 3: *Posterior probability of the difference in means with compVal=1.0 and ROPE  $\pm 0.1$ .*

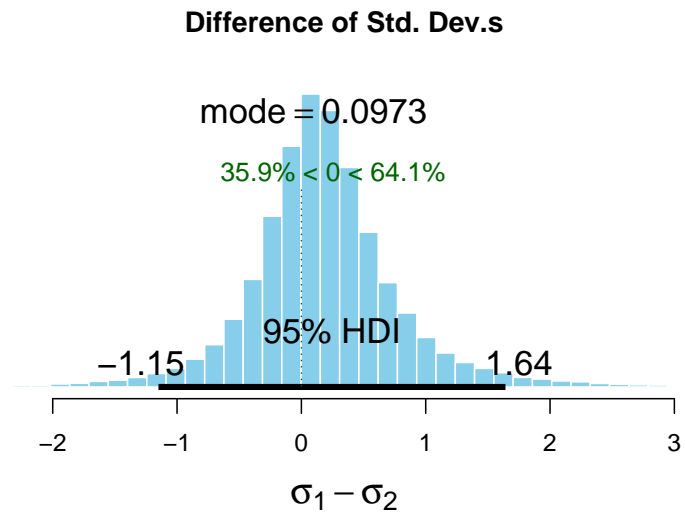


Figure 4: *Posterior plots for difference in standard deviation.*

Here we have summaries of posterior distributions for the derived parameters: difference in means (`muDiff`), difference in standard deviations (`sigmaDiff`) and effect size (`effSz`). As with the `plot` command, we can set values for `compVal` and `ROPE` for each of the parameters of interest:

```
> summary(BESTout, credMass=0.8, ROPEm=c(-0.1,0.1), ROPEsd=c(-0.15,0.15),
          compValeff=1)
```

	mean	median	mode	HDI%	HDIlo	HDIup	compVal	%>compVal	ROPElow
mu1	4.694	4.690	4.7059	80	4.168	5.215			
mu2	3.217	3.222	3.2145	80	2.792	3.662			
muDiff	1.477	1.470	1.4990	80	0.751	2.179	0	98.7	-0.10
sigma1	1.029	0.890	0.7430	80	0.457	1.340			
sigma2	0.837	0.728	0.6030	80	0.370	1.086			
sigmaDiff	0.192	0.152	0.0973	80	-0.527	0.842	0	64.1	-0.15
nu	33.599	24.840	9.5519	80	1.321	52.151			
log10nu	1.362	1.395	1.4943	80	0.893	1.922			
effSz	1.700	1.680	1.5500	80	0.678	2.698	1	81.0	

```
      ROPEhigh %InROPE
mu1
mu2
muDiff      0.10   0.652
sigma1
sigma2
sigmaDiff    0.15  25.376
nu
log10nu
effSz
```

## 4.4 Checking convergence and fit

The output from `BESTmcmc` has class `BEST`, which has a `print` method:

```
> class(BESTout)

[1] "BEST"          "data.frame"

> print(BESTout)

MCMC fit results for BEST analysis:
100002 simulations saved.
```

	mean	sd	median	HDIlo	HDIup	Rhat	n.eff
mu1	4.6937	0.4938	4.6895	3.7351	5.630	1.001	26660
mu2	3.2170	0.3961	3.2215	2.4575	3.998	1.000	34796
nu	33.5992	29.8155	24.8400	1.0002	92.248	1.000	19028
sigma1	1.0292	0.5712	0.8896	0.3685	2.039	1.004	8877
sigma2	0.8367	0.4387	0.7277	0.2892	1.639	1.002	11695

'HDIlo' and 'HDIup' are the limits of a 95% HDI credible interval.

'Rhat' is the potential scale reduction factor (at convergence, `Rhat=1`).

'n.eff' is a crude measure of effective sample size.

The `print` function displays the mean, standard deviation and median of the posterior distributions of the parameters in the model, together with a 95% Highest Density Interval: see the help page for the `hdi` function for details. Two convergence diagnostic measures are also displayed:

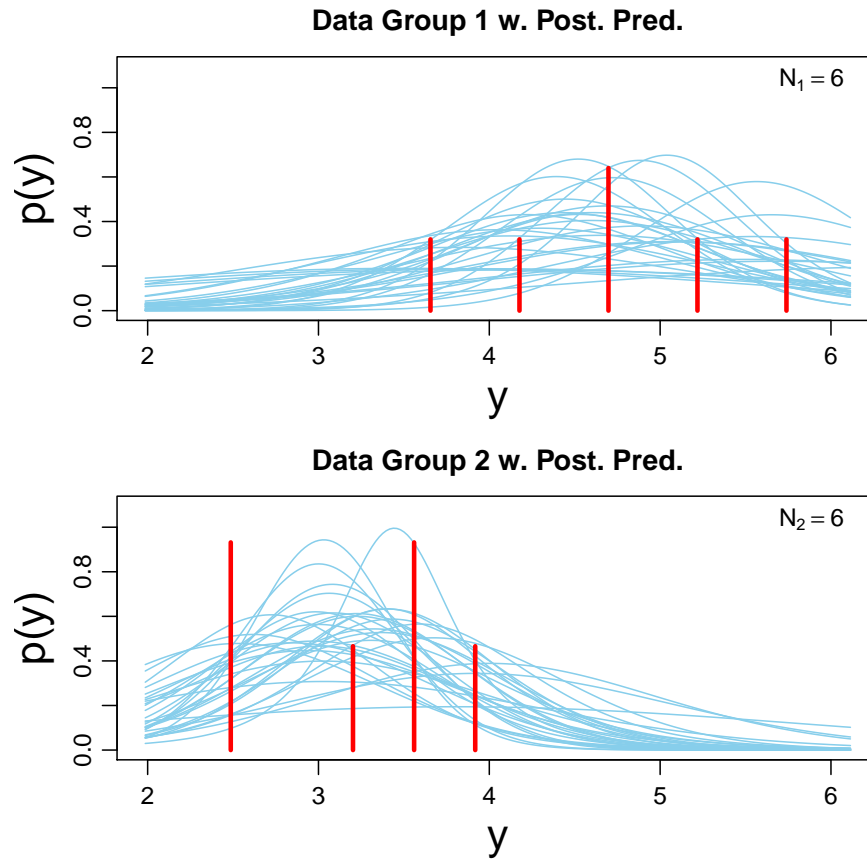


Figure 5: *Posterior predictive plots together with a histogram of the data.*

- **Rhat** is the Brooks-Gelman-Rubin scale reduction factor, which is 1 on convergence. Gelman and Shirley (2011) consider values below 1.1 to be acceptable. Increase the **burnInSteps** argument to **BESTmcmc** if any of the **Rhats** are too big.
- **n.eff** is the effective sample size, which is less than the number of simulations because of autocorrelation between successive values in the sample. Values of **n.eff** around 10,000 are needed for stable estimates of 95% credible intervals.<sup>1</sup> If any of the values is too small, you can increase the **numSavedSteps** or **thinSteps** arguments.

See the help pages for the **coda** package for more information on these measures.

As a further check, we can compare *posterior predictive distributions* with the original data:

```
> plotPostPred(BESTout)
```

Each panel of Figure 5 corresponds to one of the samples, and shows curves produced by selecting 30 random steps in the MCMC chain and plotting the  $t$  distribution with the values of  $\mu$ ,  $\sigma$  and  $\nu$  for that step. Also shown is a histogram of the actual data. We can visually assess whether the model is a reasonably good fit to the sample data (though this is easier for large samples than when  $n = 6$  as here).

<sup>1</sup>See <http://doingbayesiandataanalysis.blogspot.com/2011/07/how-long-should-mcmc-chain-be-to-get.html> for some simulation results.

The function `plotAll` puts histograms of all the posterior distributions and the posterior predictive plots onto a single page (Figure 6).

```
> plotAll(BESTout)
```

## 4.5 Working with individual parameters

Objects of class `BEST` contain long vectors of simulated draws from the posterior distribution of each of the parameters in the model. Since `BEST` objects are also data frames, we can use the `$` operator to extract the columns we want:

```
> names(BESTout)

[1] "mu1"      "mu2"      "nu"       "sigma1"  "sigma2"

> meanDiff <- (BESTout$mu1 - BESTout$mu2)
> meanDiffGTzero <- mean(meanDiff > 0)
> meanDiffGTzero

[1] 0.9869603
```

For example, you may wish to look at the ratio of the variances rather than the difference in the standard deviations. You can calculate a vector of draws from the posterior distribution, calculate summary statistics, and plot the distribution with `plotPost` (Figure 7):

```
> varRatio <- BESTout$sigma1^2 / BESTout$sigma2^2
> median(varRatio)

[1] 1.492709

> hdi(varRatio)

      lower      upper
0.004959893 10.431696320
attr(,"credMass")
[1] 0.95

> mean(varRatio > 1)

[1] 0.6411572

> plotPost(varRatio, xlim=c(0, 30))
```



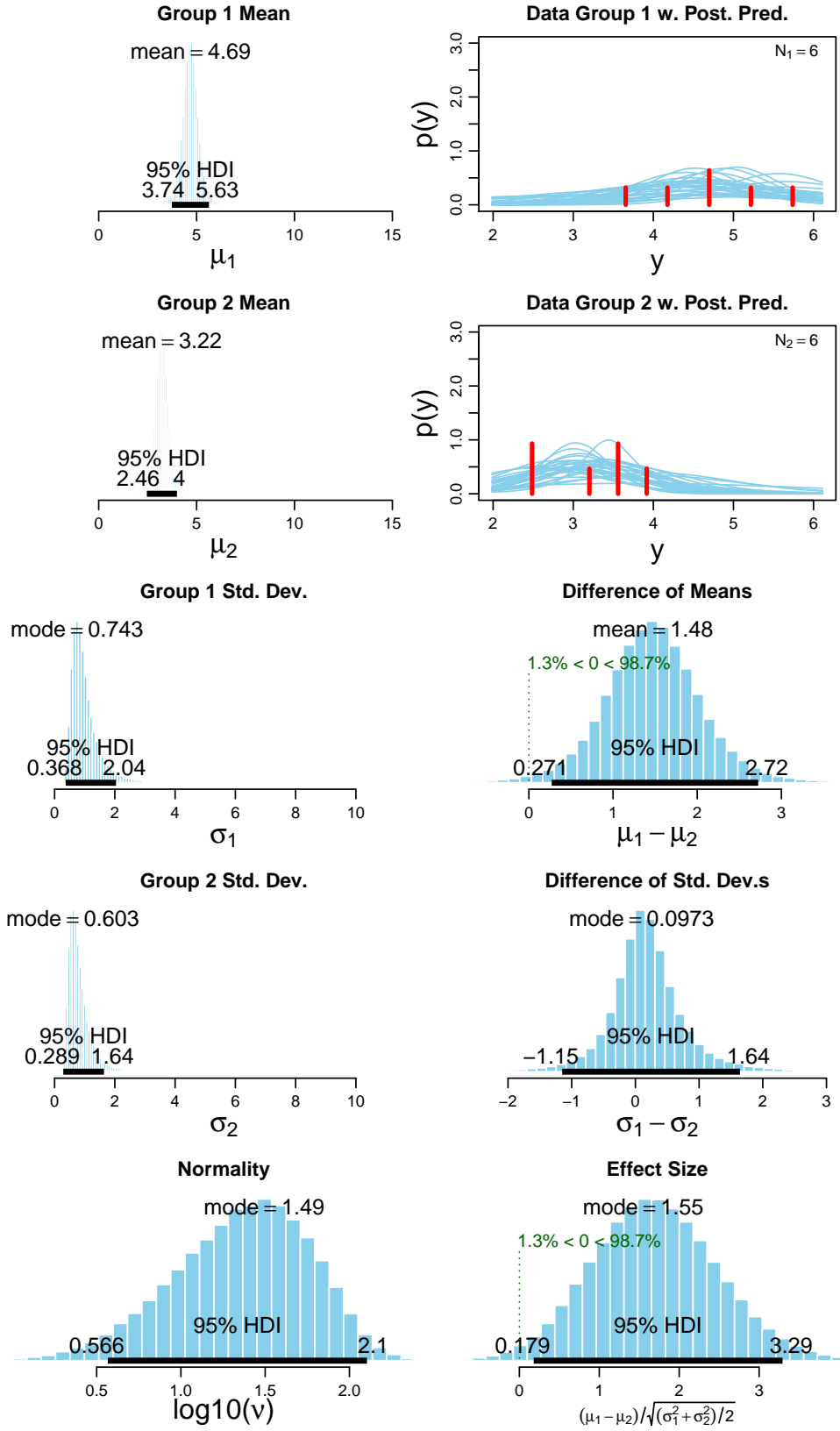


Figure 6: All the posterior distributions and the posterior predictive plots.

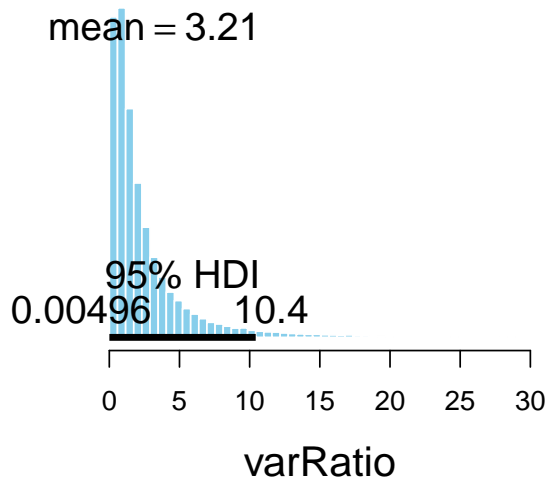


Figure 7: *Posterior distribution of the ratio of the sample variances.*

## 5 An example with a single group

Applying BEST to a single sample, or for differences in paired observations, works in much the same way as the two-sample method and uses the same function calls. To run the model, simply use `BESTmcmc` with only one vector of observations.

```
> y0 <- c(1.89, 1.78, 1.30, 1.74, 1.33, 0.89)
> BESTout1g <- BESTmcmc(y0, parallel=FALSE)
```

Processing function input.....

Done.

```
Compiling model graph
  Resolving undeclared variables
  Allocating nodes
  Graph Size: 21
```

Initializing model

```
Adaptive phase, 500 iterations x 3 chains
If no progress bar appears JAGS has decided not to adapt
```

```
|+++++| 100%
```

```
Burn-in phase, 1000 iterations x 3 chains
```

```
|*****| 100%
```

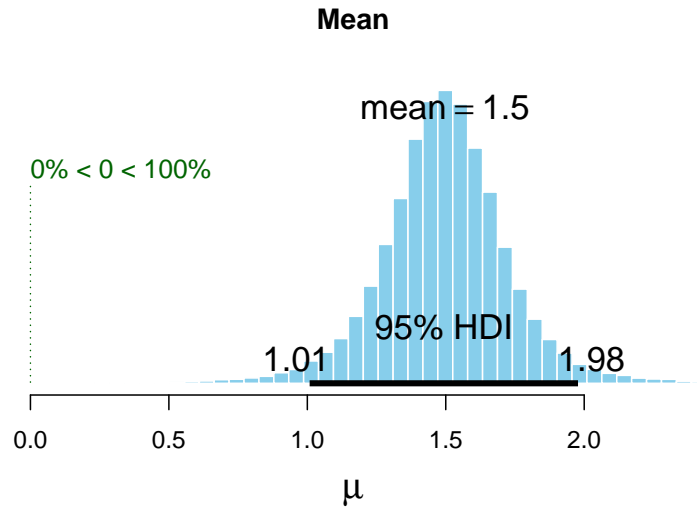


Figure 8: *Default plot: posterior probability distribution for the mean.*

Sampling from joint posterior, 33334 iterations x 3 chains

|\*\*\*\*\*| 100%

MCMC took 0.081 minutes.

This time we have a single mean and standard deviation. The default plot (Figure 8) shows the posterior distribution of the mean.

```
> BESTout1g
```

MCMC fit results for BEST analysis:

100002 simulations saved.

	mean	sd	median	HDIlo	HDIup	Rhat	n.eff
mu	1.497	0.2478	1.497	1.0084	1.978	1.003	35734
nu	32.122	29.1469	23.381	1.0008	90.411	1.000	20547
sigma	0.523	0.2942	0.452	0.1797	1.021	1.014	7533

'HDIlo' and 'HDIup' are the limits of a 95% HDI credible interval.

'Rhat' is the potential scale reduction factor (at convergence, Rhat=1).

'n.eff' is a crude measure of effective sample size.

```
> plot(BESTout1g)
```

Standard deviation, the normality parameter and effect size can be plotted individually, or on a single page with `plotAll` (Figure 9).

```
> plotAll(BESTout1g)
```

And we can access the draws from the posterior distributions with the `$` operator:

```
> names(BESTout1g)
```

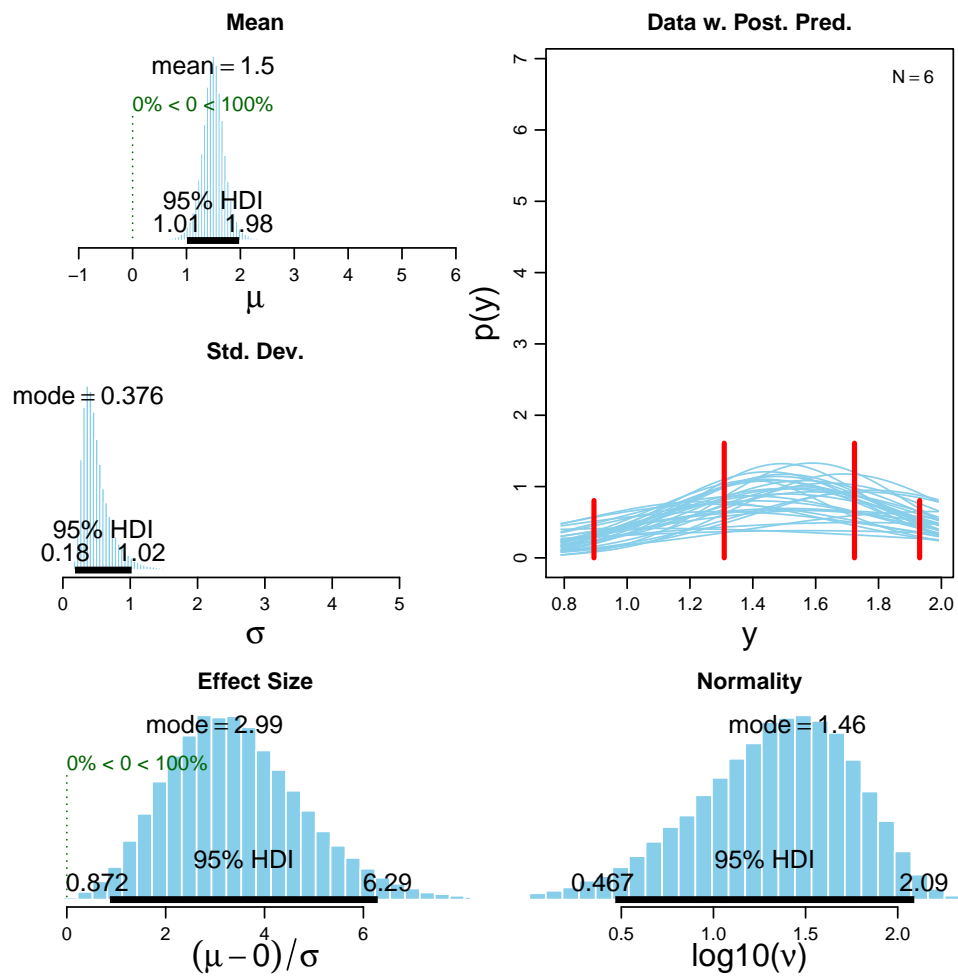


Figure 9: All the posterior distributions and the posterior predictive plots.

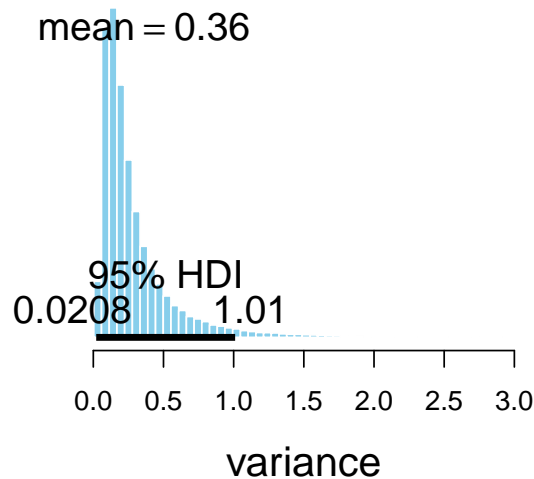


Figure 10: *Posterior distribution of the sample variance.*

```
[1] "mu"      "nu"      "sigma"

> length(BESTout1g$nu)

[1] 100002

> variance <- BESTout1g$sigma^2
> plotPost(variance, xlim=c(0, 3))
```

## 6 What next?

The default is to use uninformative priors as described in Kruschke (2013), but `BESTmcmc` allows you to specify your own informative priors. See the help page.

The package includes functions to estimate the power of experimental designs: see the help pages for `BESTpower` and `makeData` for details on implementation and Kruschke (2013) for background.

If you want to know how the functions in the `BEST` package work, you can download the R source code from CRAN or from GitHub <https://github.com/mikemeredith/BEST> or find almost the same code at <http://www.indiana.edu/~kruschke/BEST/> together with links to articles, videos, and the blog.

Bayesian analysis with computations performed by JAGS is a powerful approach to analysis. For a practical introduction see Kruschke (2011).

## 7 References

- Gelman A, Shirley K (2011). “Inference from simulations and monitoring convergence.” In S Brooks, A Gelman, G Jones, XL Meng (eds.), *Handbook of Markov chain Monte Carlo*, pp. 163–174. Chapman & Hall.
- Kruschke JK (2011). *Doing Bayesian data analysis: a tutorial with R and BUGS*. Elsevier, Amsterdam etc.
- Kruschke JK (2013). “Bayesian estimation supersedes the  $t$  test.” *Journal of Experimental Psychology: General*, **142**(2), 573–603.
- Plummer M (2003). “JAGS: A Program for Analysis of Bayesian Graphical Models Using Gibbs Sampling.” In *3rd International Workshop on Distributed Statistical Computing (DSC 2003)*. Vienna, Austria.