STAT 491 - Lecture 9: T-tests

Posterior Predictive Distribution

Another valuable tool in Bayesian statistics is the posterior predictive distribution.

- The posterior predictive distribution can be written as:

$$p(y^*|y) = \int p(y^*|\theta)p(\theta|y)d\theta$$

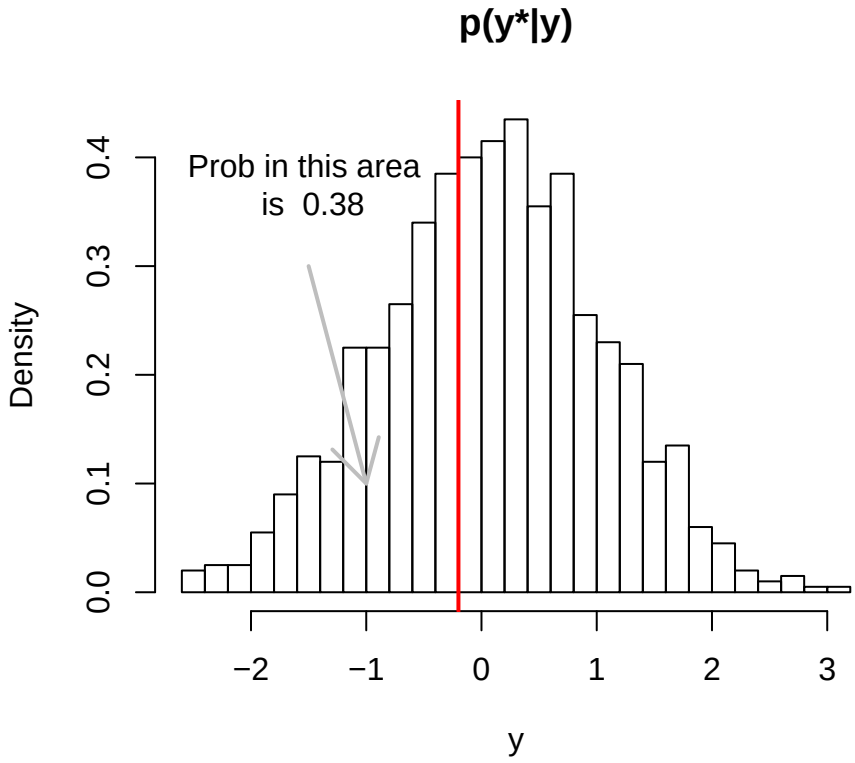
where y^* is interpreted as a new observation and $p(\theta|y)$ is the posterior for the parameter θ given that data y have been observed.

- The posterior predictive distribution allows us to test whether our sampling model and observed data are reasonable. We will talk more about this later.

- The posterior predictive distribution can also be used to make probabilistic statements about the next response, rather than the group mean. In our continuing example, we could calculate the probability of the next observed data point being greater than -0.2.

- When $p(\theta|y)$ does not have a standard form, samples from this distribution can be inserted into the sampling model. This sampling procedure is a Monte Carlo approach for this integration.

```
posterior.mu <- codaSamples[[1]][,'mu']
posterior.sigma <- codaSamples[[1]][,'sigma']
posterior.pred <- rnorm(num.mcmc, mean = posterior.mu, sd = posterior.sigma)
prob.greater <- mean(posterior.pred > -0.2)
```



T - distribution

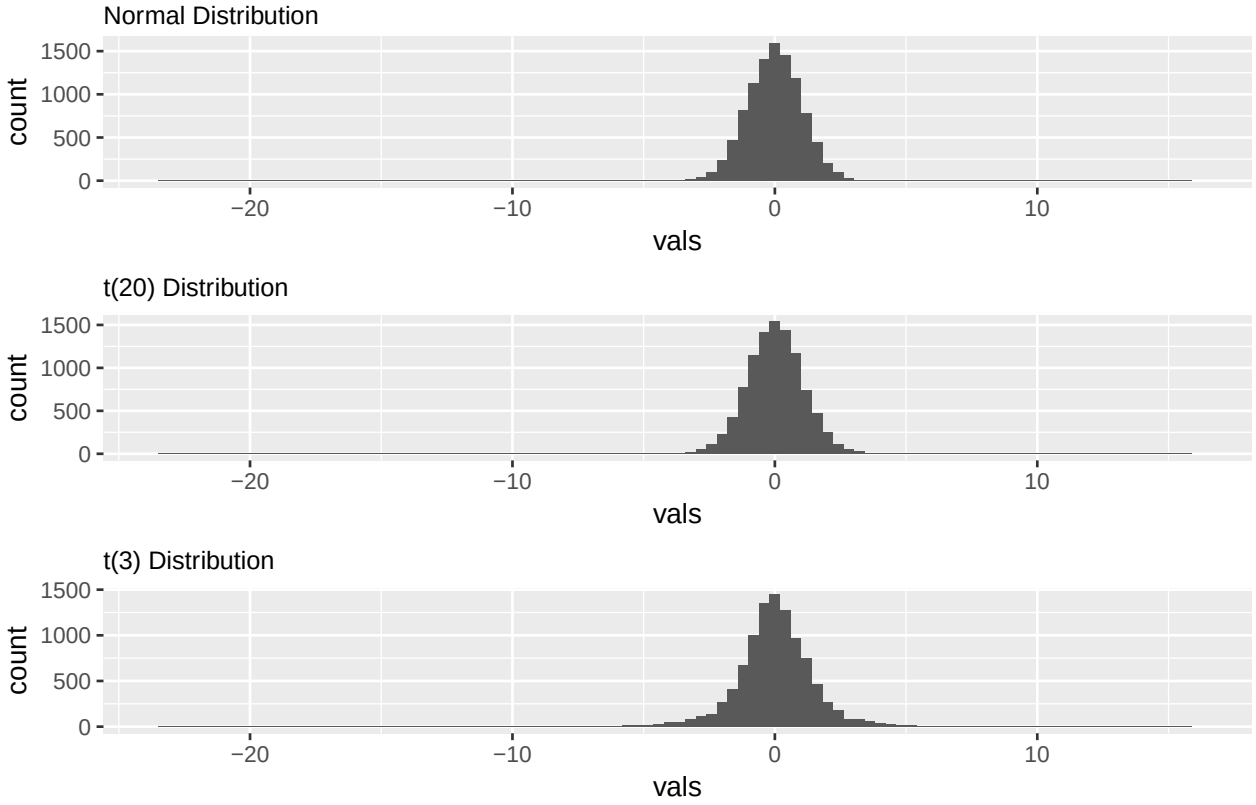
- While the normal distribution is often used for modeling continuous data, an alternative is the t-distribution. *Q:* Where have you seen a t-distribution before and what properties does it have?

- The t-distribution has an interesting history. It was developed by William Gosset, a statistician at Guiness brewery. His results were published in secret under the pseudonym Student. Thus the distribution has become known as Student's t-distribution.

- The t-distribution is more *robust* to observations away from the mean, meaning there is more mass in the tails.

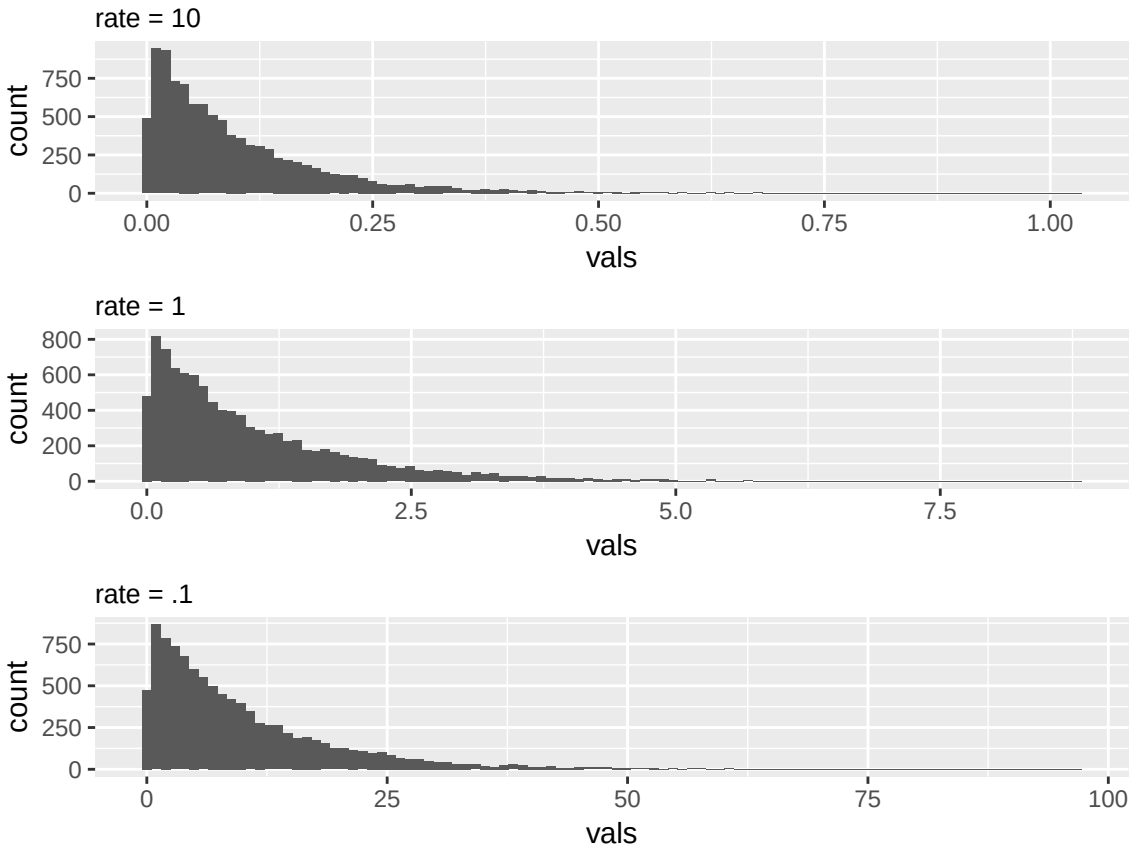
- the wider tails can be illustrated thinking about the 2.5% quantile in terms of standard deviation for a specified degrees of freedom ν.
  - normal = 1.96
  - t(50) = 2.00
  - t(40) = 2.02
  - t(30) = 2.04
  - t(20) = 2.09
  - t(10) = 2.29
  - t(5) = 2.57
  - t(3) = 3.18
  - t(1) (Cauchy) = 12.70

- When the degrees of freedom gets large, the distribution approaches a normal distribution and when the degrees of freedom approach 1 the distribution becomes a Cauchy distribution



Bayesian modeling with t-distribution

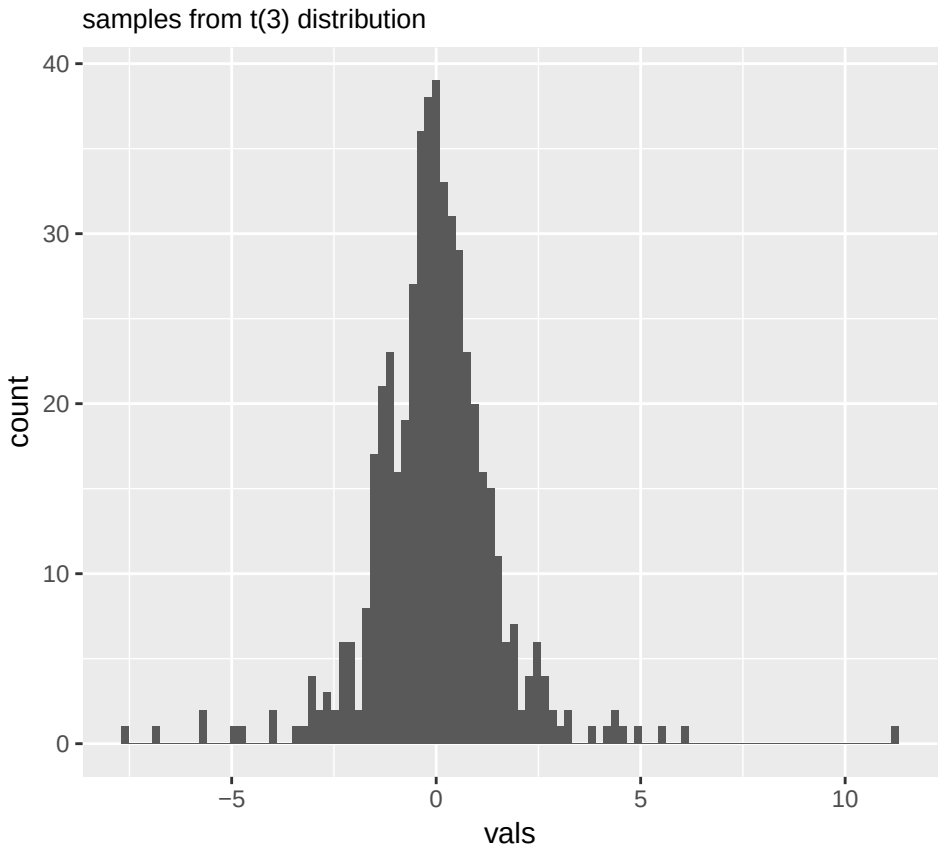
- Sampling model $y \sim t(\mu, \sigma^2, \nu)$
- This requires a prior distribution on:
  - μ: Similiar to the normal sampling model case, we can use a normal distribution with $p(\mu) \sim N(M, S^2)$
  - σ^2: The variance term also has a similar interpretation, so we can use a uniform or inverse-gamma distribution for a prior.
  - ν: The term ν is often called the degrees of freedom, and this controls the tail behavior of the distribution. The restriction is that the degrees of freedom has to be larger than one. A common prior is to use a shifted exponential distribution.



After looking at the figures, what do you think the mean of the exponential distribution is?
1/rate

JAGS code

```
t.samples <- data.frame(rt(500, df = 3))
colnames(t.samples) <- 'vals'
ggplot(data=t.samples, aes(vals)) + geom_histogram(bins = 100) +
  labs(subtitle = "samples from t(3) distribution")
```

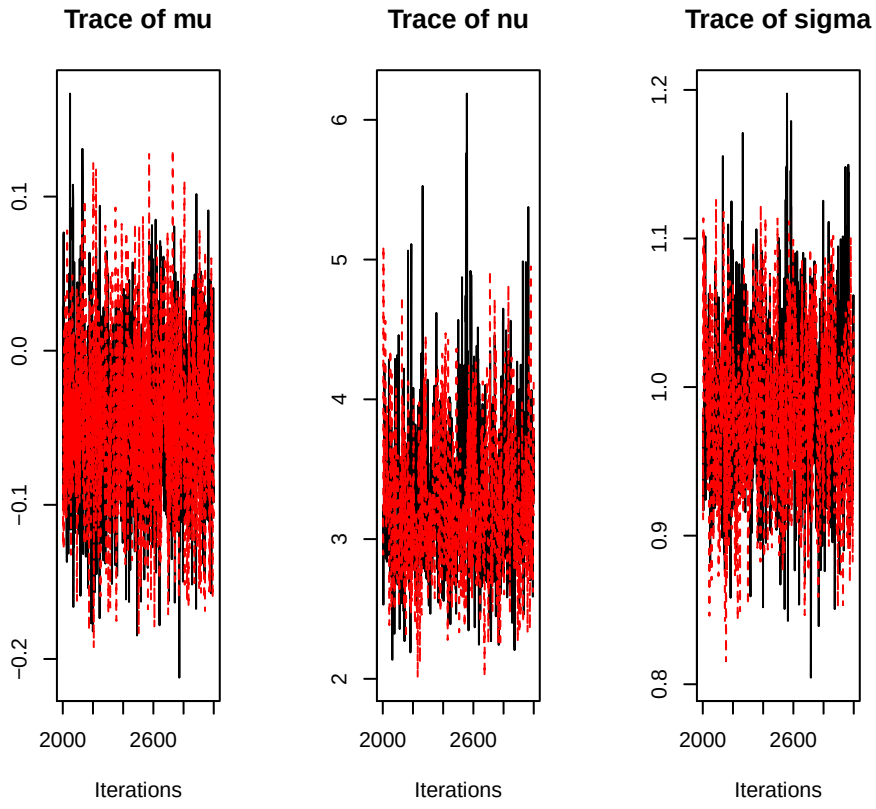



```
#Prior parameters
M <- 0
S <- 100
C <- 10
rate <- .1

# Store data
dataList = list(y = t.samples$vals, Ntotal = nrow(t.samples), M = M, S = S, C = C, rate = rate)

# Model String
modelString = "model {
  for ( i in 1:Ntotal ) {
    y[i] ~ dt(mu, 1/sigma^2, nu) # sampling model
  }
  mu ~ dnorm(M,1/S^2)
  sigma ~ dunif(0,C)
  nu <- nuMinusOne + 1 # transform to guarantee n >= 1
  nuMinusOne ~ dexp(rate)
} "
writeLines( modelString, con='Tmodel.txt')

# initialization
initsList <- function(){
  # function for initializing starting place of theta
  # RETURNS: list with random start point for theta
  return(list(mu = rnorm(1, mean = M, sd = S), sigma = runif(1,0,C),
```

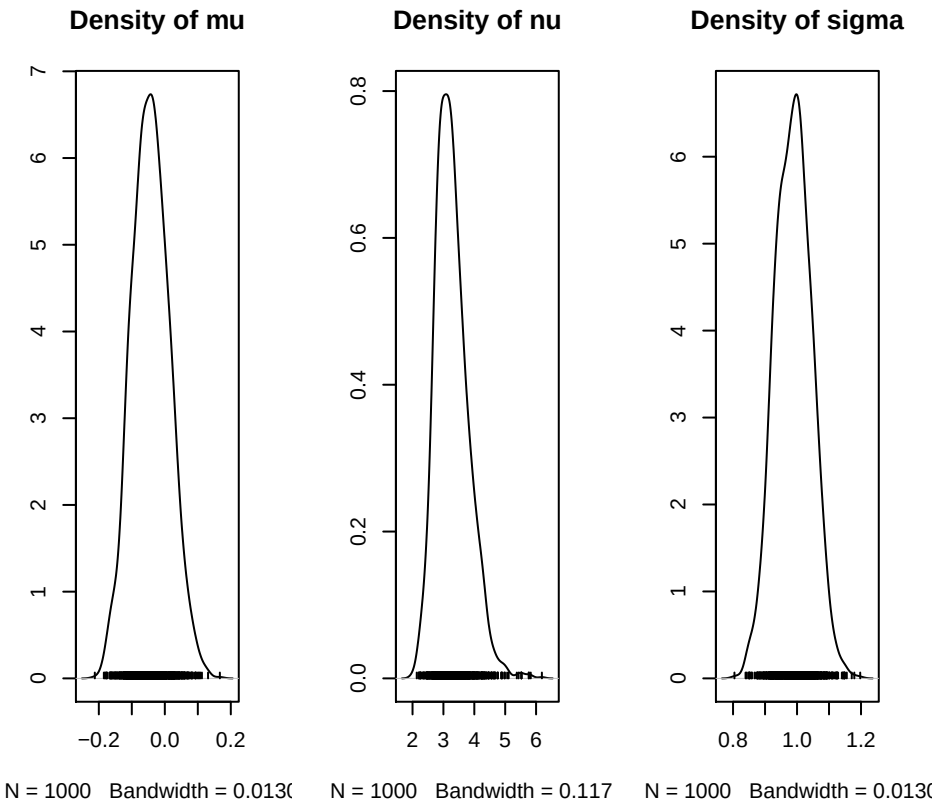
```
              nuMinusOne = rexp(1, rate=rate) ))
}

# Runs JAGS Model
jagsT <- jags.model( file = "Tmodel.txt", data = dataList, inits =initsList,
                     n.chains = 2, n.adapt = 1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 500
##    Unobserved stochastic nodes: 3
##    Total graph size: 516
##
## Initializing model
```

```
update(jagsT, n.iter = 1000)

num.mcmc <- 1000
codaSamples <- coda.samples( jagsT, variable.names = c('mu', 'sigma','nu'), n.iter = num.mcmc)

par(mfcol=c(1,3))
traceplot(codaSamples)
```




```
densplot(codaSamples)
```




```
HPDinterval(codaSamples)
```

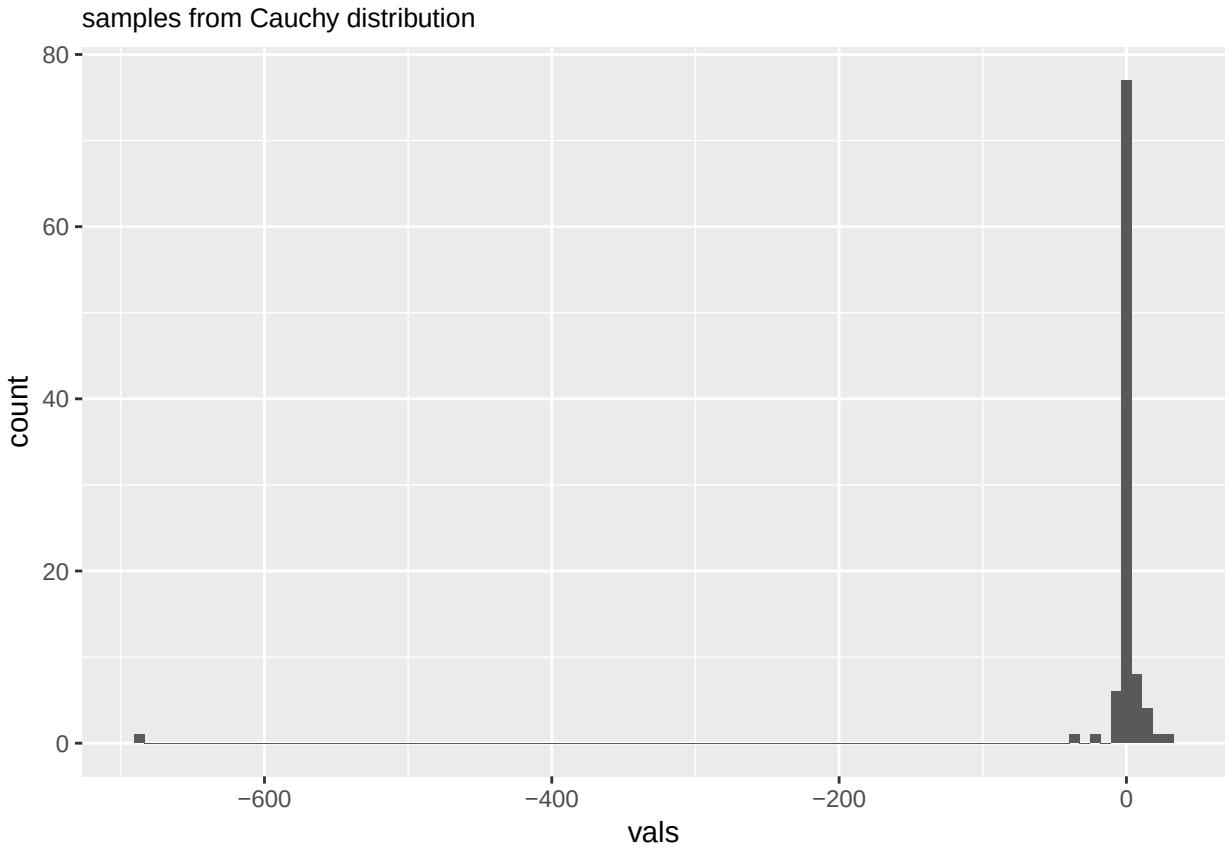
```
## [[1]]
##            lower      upper
## mu    -0.1537257 0.06126606
## nu     2.3688289 4.46911934
## sigma  0.8794691 1.09645366
## attr(,"Probability")
## [1] 0.95
##
## [[2]]
##            lower      upper
## mu    -0.1569754 0.06952619
## nu     2.4633480 4.36374006
## sigma  0.8805058 1.09187628
## attr(,"Probability")
## [1] 0.95
```

Lab Exercise 1

1.

Simulate 100 responses from a Cauchy distribution, t distribution with $\mu = 1$, σ^2=1 and $\nu = 1$, and describe this data with a plot and brief description of the data.

```
set.seed(03132018)
t.samples <- data.frame(rt(100, df = 1))
colnames(t.samples) <- 'vals'
ggplot(data=t.samples, aes(vals)) + geom_histogram(bins = 100) +
  labs(subtitle = "samples from Cauchy distribution")
```



As can be seen from the quantiles of the data there are a few extreme observations, but most of the mass is fairly close to zero.

2.

Use JAGS to fit a normal sampling model and the following priors for this data.

- $p(\mu) \sim N(0, 10^2)$
- $p(\sigma) \sim U(0, 1000)$

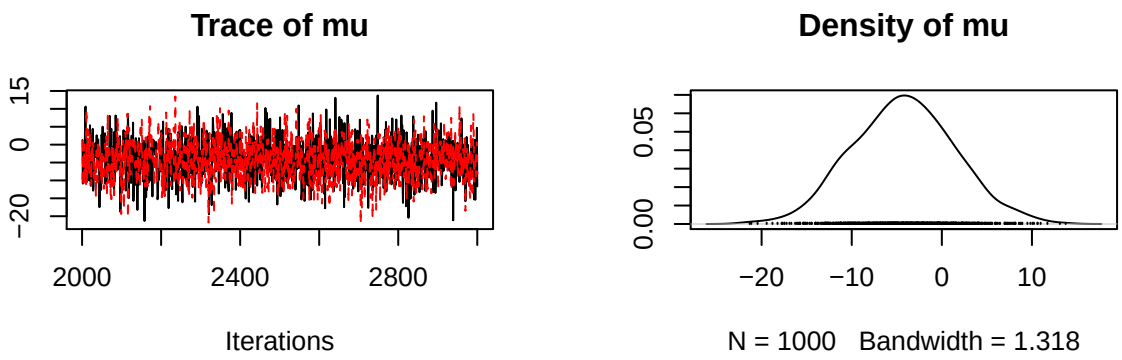
Discuss the posterior HDIs for μ and σ.

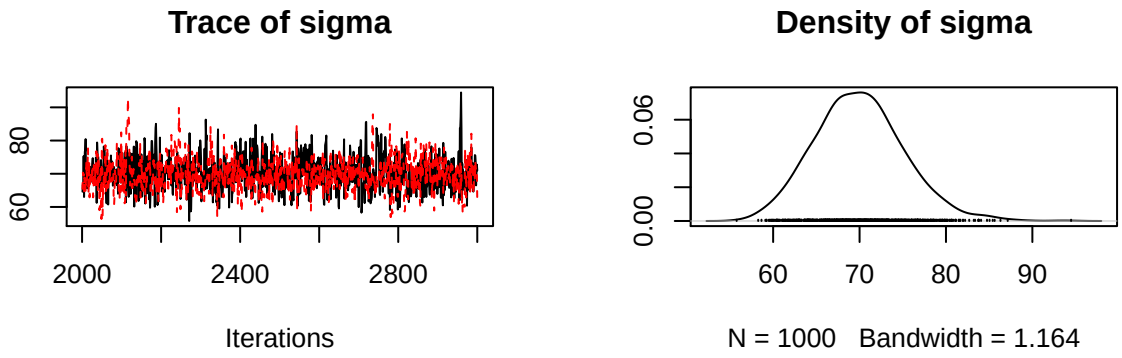
```
#Prior parameters
M <- 0
S <- 10
C <- 1000
```

```r
# Store data
dataList = list(y = t.samples$vals, Ntotal = nrow(t.samples), M = M, S = S, C = C)

# Model String
modelString = "model {
  for ( i in 1:Ntotal ) {
    y[i] ~ dnorm(mu, 1/sigma^2) # sampling model
  }
  mu ~ dnorm(M,1/S^2)
  sigma ~ dunif(0,C)
} "
writeLines( modelString, con='NORMmodel.txt')

# initialization
initsList <- function(){
  # function for initializing starting place of theta
  # RETURNS: list with random start point for theta
  return(list(mu = rnorm(1, mean = M, sd = S), sigma = runif(1,0,C) ))
}

# Runs JAGS Model
jags.norm <- jags.model( file = "NORMmodel.txt", data = dataList, inits =initsList,
                         n.chains = 2, n.adapt = 1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 100
##    Unobserved stochastic nodes: 2
##    Total graph size: 113
##
## Initializing model
```

```r
update(jags.norm, n.iter = 1000)

num.mcmc <- 1000
coda.norm <- coda.samples( jags.norm, variable.names = c('mu', 'sigma'), n.iter = num.mcmc)

par(mfcol=c(2,2))
traceplot(coda.norm)
densplot(coda.norm)
```






```
HPDinterval(coda.norm)
```

```
## [[1]]
##           lower     upper
## mu    -14.04595  8.174991
## sigma  60.37572 80.177694
## attr(,"Probability")
## [1] 0.95
##
## [[2]]
##           lower     upper
## mu    -14.84437  7.240454
## sigma  60.08146 80.284547
## attr(,"Probability")
## [1] 0.95
```

The interval for μ is roughly centered around zero, but is fairly wide with a large degree of uncertainty. The interval for σ is very large, compared to simulated variance of 1 from the t-distribution.

3.

Use JAGS to fit a t sampling model and the following priors for this data.

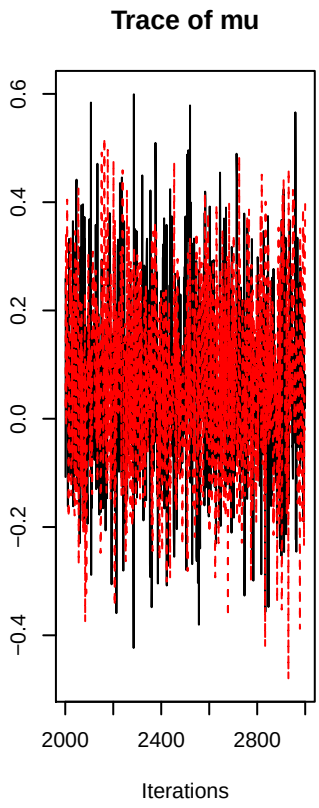
- $p(\mu) \sim N(0, 10^2)$
- $p(\sigma) \sim U(0, 1000)$
- $p(\nu) \sim E_+(.1)$, where $E_+(.1)$ is a shifted exponential with rate = .1.

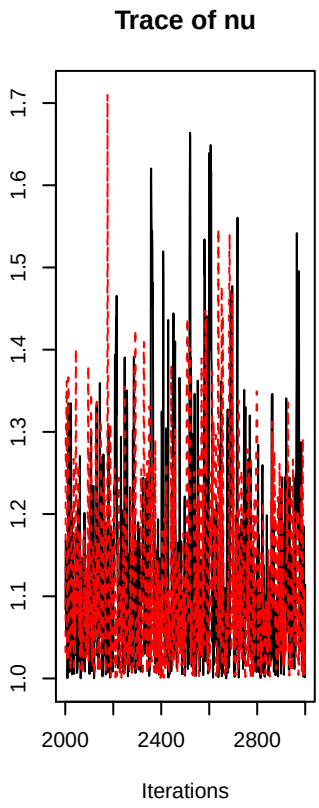
Discuss the posterior HDIs for μ, σ, and ν.

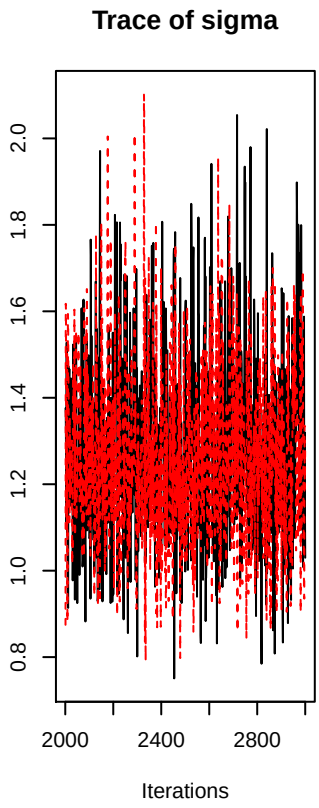
```r
#Prior parameters
M <- 0
S <- 10
C <- 1000
rate <- .1

# Store data
dataList = list(y = t.samples$vals, Ntotal = nrow(t.samples), M = M, S = S, C = C, rate = rate)

# Model String
modelString = "model {
  for ( i in 1:Ntotal ) {
    y[i] ~ dt(mu, 1/sigma^2, nu) # sampling model
  }
  mu ~ dnorm(M,1/S^2)
  sigma ~ dunif(0,C)
  nu <- nuMinusOne + 1 # transform to guarantee n >= 1
  nuMinusOne ~ dexp(rate)
} "
writeLines( modelString, con='Tmodel.txt')

# initialization
initsList <- function(){
  # function for initializing starting place of theta
  # RETURNS: list with random start point for theta
  return(list(mu = rnorm(1, mean = M, sd = S), sigma = runif(1,0,C),
              nuMinusOne = rexp(1, rate=rate) ))
}

# Runs JAGS Model
jagsT <- jags.model( file = "Tmodel.txt", data = dataList, inits =initsList,
                     n.chains = 2, n.adapt = 1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 100
##    Unobserved stochastic nodes: 3
##    Total graph size: 116
##
## Initializing model
```

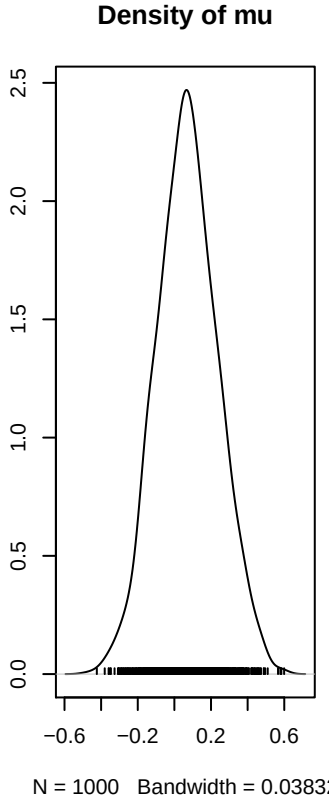
```r
update(jagsT, n.iter = 1000)

coda.t <- coda.samples( jagsT, variable.names = c('mu', 'sigma','nu'), n.iter = num.mcmc)

par(mfcol=c(1,3))
traceplot(coda.t)
```

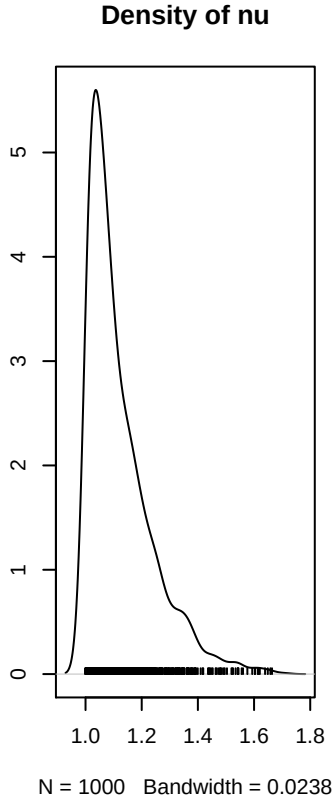


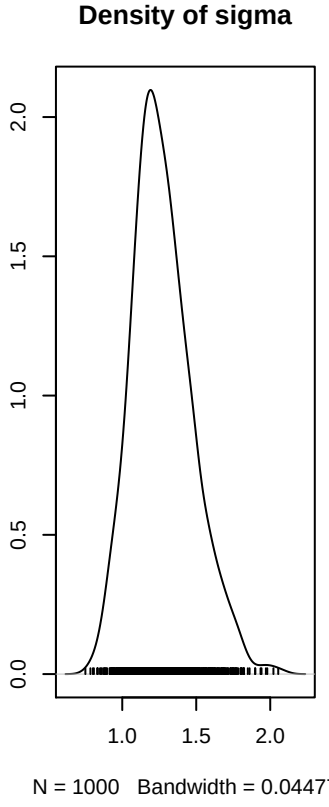





```
densplot(coda.t)
```







```
HPDinterval(coda.t)
```

```
## [[1]]
##            lower     upper
## mu    -0.2684917 0.3992048
## nu     1.0001150 1.3792184
## sigma  0.9250747 1.7502252
## attr(,"Probability")
## [1] 0.95
##
## [[2]]
##            lower     upper
## mu    -0.2146493 0.4128876
## nu     1.0002784 1.3380140
## sigma  0.8854668 1.6427006
## attr(,"Probability")
## [1] 0.95
```

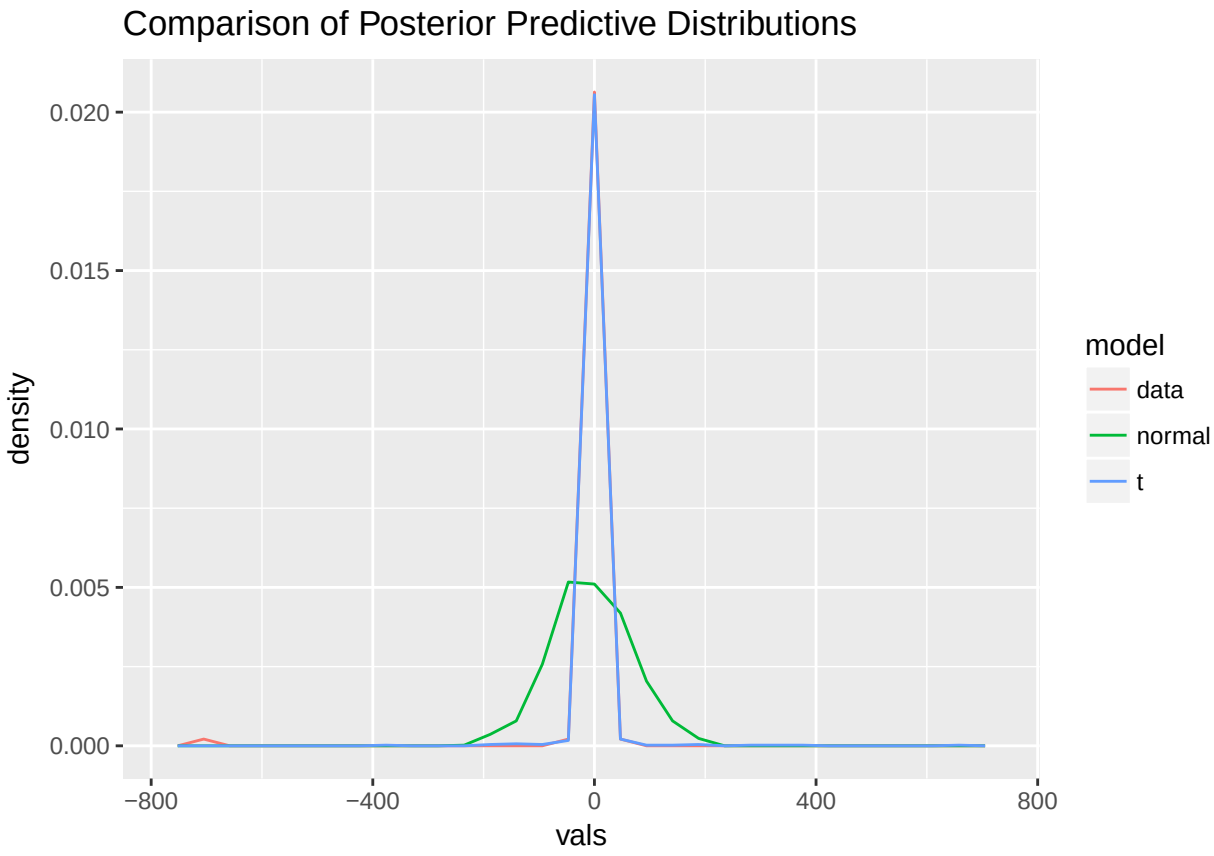
The intervals contain the true values with fairly low uncertainty. The only exception is that the ν value is larger than 1, but this is due to the prior specification and the fact that a t-distribution with $\nu < 1$ in not valid.

4.

Use the following code to create posterior predictive distributions for part 2 and part 3. Note: your data and coda objects may need to be renamed for this to work. Compare the data and the posterior predictive model curves with posterior predictive models. Note this is the final step in Bayesian data analysis: verifying that our model / prior selection is an accurate representation of the data.

```
# Posterior Predictive Normal
post.pred.normal <- rnorm(num.mcmc, coda.norm[[1]][,'mu'], coda.norm[[1]][,'sigma'] )
# Posterior Predictive t
post.pred.t <- rt(num.mcmc, df = coda.t[[1]][,'nu']) * coda.t[[1]][,'sigma'] + coda.t[[1]][,'mu']
data.comb <- data.frame(vals = c(t.samples$vals, post.pred.normal, post.pred.t),
                        model = c(rep('data',100), rep('normal', num.mcmc), rep('t',num.mcmc)))

ggplot(data.comb, aes(vals, ..density.., colour = model)) + geom_freqpoly() +
  ggtitle('Comparison of Posterior Predictive Distributions')
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```


Comparison of Posterior Predictive Distributions
density
0.020
0.015
0.010
0.005
0.000
−800
−400
0
400
800
vals
model
data
normal
t

Estimation with Two Groups

A common use of the t-distribution is to make comparisons between two groups. For instance, we may be interested to determine if the mean height of two groups of OK Cupid users are different.

We can write this model out as

$$y_{ij} \sim t(\mu_j, \sigma_j^2, \nu),$$

where y_{ij} is the height of the i^{th} person in the j^{th} group, μ_j and σ^2 are the mean height and variance of the j^{th} group and ν is a common degree of freedom estimate across the groups. From a Bayesian perspective, this model will require priors on:

-μ_j for all j,

-σ_j^2 for all j, and

- ν.

An aside on Null Hypothesis Significance Testing (NHST) (Ch. 11)

- What is the purpose of NHST? The goal of NHST is to decide whether a particular value of a parameter can be rejected.

- For instance, consider estimating whether a die has a fair probability of rolling a 6 ($\theta = 1/6$).
  - Then if we roll the die several times we'd expect 1/6 of the rolls to return a 6.
  - If the actual number is far greater or less than our expectation, we should reject the hypothesis that the die is fair.
  - To do this, we compute the exact probabilities of getting all outcomes. From this, we can compute the probability of getting an outcome, under the null hypothesis, as extreme or more than the observed outcome. This probability is known as a p-value.
  - The null hypothesis is commonly rejected if the p-value is less than 0.05.

- It is important to note that calculating the probability of all outcomes requires both the sampling and testing procedure.

- We are not going to get into the details, but section 11.1 in the texbook details a situation where a coin is flipped 24 times and results in 7 heads. The goal is determine if the coin is fair. Depending on the sampling procedure used, the p-value can range from .017 to .103 with this dataset.

Bayesian Approach to Testing a Point Hypothesis

Consider the die rolling example. What value for (θ) would be says is meaningfully different than $\theta = 1/6 = 0.167$? If we are in a high-stakes gambling game, we might want θ to be accurate up to 0.001%, however, if we are using the dice in a friendly board game then accuracy of 2% might be sufficient.

- This range around the specified value is known as the Region Of Practical Equivalence (ROPE).

- Given a ROPE and a posterior distribution, the parameter value is declared to be not credible, or rejected, if its entire ROPE lies outside of the 95% HDI of the posterior distribution of that parameter.

- A parameter value is declared to be accepted for practical purposes of that value's ROPE completely contains the 95% HDI of the posterior for that parameter.

- When the HDI and ROPE overlap, with the ROPE not completely containing the HDI, then neither of the above rules is satisfied and we withhold a decision.

- Note that the NHST regime provides no way to confirm a theory, rather just the ability to reject the null hypothesis.

Lab Questions Use the OK Cupid dataset and test the following claim, the mean height OK Cupid respondents reporting their body type as athletic is different than 70.5 inches (this value is arbitrary, but is approximately the mean height of all men in the sample). Interpret the results for each scenario.

```
okc <- read.csv('http://www.math.montana.edu/ahoegh/teaching/stat408/datasets/OKCupid_profiles_clean.csv
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following object is masked from 'package:gridExtra':
##
##     combine
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
okc.athletic <- okc %>% filter(body_type == 'athletic' & sex == 'm')
okc %>% filter(sex == 'm') %>% summarise(mean(height))
```

```
##   mean(height)
## 1     70.44497
```

1. Use `t.test()` with a two-sided procedure.

```
t.test(okc.athletic$height, mu = 70.5, alternative = 'two.sided')
```

```
##
##  One Sample t-test
##
## data:  okc.athletic$height
## t = 3.8933, df = 3783, p-value = 0.0001006
## alternative hypothesis: true mean is not equal to 70.5
## 95 percent confidence interval:
##  70.58947 70.77099
## sample estimates:
## mean of x
##  70.68023
```

2. Fit a Bayesian model for μ with a ROPE of $\pm$ .5 inch. Use the following priors: $p(\mu) \sim N(70.5, 10^2)$, $p(\sigma) \sim Unif(0, 20)$, $p(\nu) \sim E_+(.1)$ and a t-sampling model.

```
M <- 70.5
S <- 10
C <- 20
rate <- .1

# Store data
dataList = list(y = okc.athletic$height, Ntotal = nrow(okc.athletic), M = M, S = S, C = C, rate = rate)

# Model String
modelString = "model {
  for ( i in 1:Ntotal ) {
    y[i] ~ dt(mu, 1/sigma^2, nu) # sampling model
```

```
  }
  mu ~ dnorm(M,1/S^2)
  sigma ~ dunif(0,C)
  nu <- nuMinusOne + 1 # transform to guarantee n >= 1
  nuMinusOne ~ dexp(rate)
} "
writeLines( modelString, con='Tmodel.txt')

# initialization
initsList <- function(){
  # function for initializing starting place of theta
  # RETURNS: list with random start point for theta
  return(list(mu = rnorm(1, mean = M, sd = S), sigma = runif(1,0,C),
              nuMinusOne = rexp(1, rate=rate) ))
}

# Runs JAGS Model
jagsT <- jags.model( file = "Tmodel.txt", data = dataList, inits =initsList,
                     n.chains = 1, n.adapt = 1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 3784
##    Unobserved stochastic nodes: 3
##    Total graph size: 3800
##
## Initializing model
```

```
update(jagsT, n.iter = 500)

coda.t <- coda.samples( jagsT, variable.names = c('mu', 'sigma','nu'), n.iter = num.mcmc)

HPDinterval(coda.t)
```

```
## [[1]]
##          lower     upper
## mu    70.59702 70.781734
## nu    12.57096 31.935671
## sigma  2.62483  2.791543
## attr(,"Probability")
## [1] 0.95
```

3. Fit a Bayesian model for μ with a ROPE of $\pm$ .05 inch

Back to the two-sample case

- Now consider whether their is a significant height difference between male OK Cupid respondents self-reporting their body type as "athletic" and those self-reporting their body type as "fit"
- From the frequentist paradigm, this can be accomplished using the `t.test()` function.

```
okc.fit <- okc %>% filter(sex == 'm' & body_type == 'fit')
t.test(x= okc.athletic$height, y = okc.fit$height)
```

```
##
##  Welch Two Sample t-test
##
## data:  okc.athletic$height and okc.fit$height
## t = 4.5651, df = 6878.8, p-value = 5.08e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  0.1771015 0.4436713
## sample estimates:
## mean of x mean of y
##  70.68023  70.36985
```

- It is important to note there is no analog to ROPE with the t-test. If you have ever heard that statistical significance does not imply practical significance this is why.
- Here is the Bayesian attempt, using JAGS. We want the posterior of $\mu_{ath} - \mu_{fit}$ for inferences.

```
M <- 70.5
S <- 10
C <- 20
rate <- .1

# Store data
dataList = list(x = okc.athletic$height, y = okc.fit$height, M = M, S = S, C = C, rate = rate)

# Model String
modelString <- "model {
  for(i in 1:length(x)) {
    x[i] ~ dt( mu_x , 1/sigma_x^2 , nu )

  }
  x_pred ~ dt( mu_x , 1/sigma_x^2 , nu ) # posterior predictive for x

  for(i in 1:length(y)) {
    y[i] ~ dt( mu_y , 1/sigma_y^2 , nu )
  }
  y_pred ~ dt( mu_y , 1/sigma_y^2 , nu ) # posterior predictive for y

  mu_diff <- mu_x - mu_y

  # The priors
  mu_x ~ dnorm( M , 1/S^2 )
  sigma_x ~ dunif( 0 , C )

  mu_y ~ dnorm( M , 1/S^2  )
  sigma_y ~ dunif( 0 , C )

  nu <- nuMinusOne+1
  nuMinusOne ~ dexp(rate)
}"
writeLines( modelString, con='TwoSampleT.txt')

# initialization
initsList <- function(){
  # function for initializing starting place of theta
```

```
  # RETURNS: list with random start point for theta
  return(list(mu_x = rnorm(1, mean = M, sd = S), sigma_x = runif(1,0,C),
              mu_y = rnorm(1, mean = M, sd = S), sigma_y = runif(1,0,C),
              nuMinusOne = rexp(1, rate=rate) ))
}

# Runs JAGS Model
jagsT <- jags.model( file = "TwoSampleT.txt", data = dataList, inits =initsList,
                     n.chains = 3, n.adapt = 1000)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 7034
##    Unobserved stochastic nodes: 7
##    Total graph size: 7056
##
## Initializing model
```

```
update(jagsT, n.iter = 1000)

coda.t <- coda.samples( jagsT, variable.names = c('mu_x', 'sigma_x','nu', 'mu_y', 'sigma_y', 'mu_diff')

HPDinterval(coda.t)
```

```
## [[1]]
##             lower      upper
## mu_diff  0.200548  0.4626475
## mu_x    70.606391 70.7772089
## mu_y    70.268110 70.4625723
## nu      16.132123 35.1433740
## sigma_x  2.652301  2.8063901
## sigma_y  2.642607  2.7954993
## attr(,"Probability")
## [1] 0.95
##
## [[2]]
##              lower      upper
## mu_diff  0.1992145  0.4665636
## mu_x    70.5962098 70.7764058
## mu_y    70.2684131 70.4547490
## nu      16.2284436 32.4390980
## sigma_x  2.6451101  2.7965831
## sigma_y  2.6440730  2.7961013
## attr(,"Probability")
## [1] 0.95
##
## [[3]]
##             lower     upper
## mu_diff  0.1958494  0.455966
## mu_x    70.5982568 70.777961
## mu_y    70.2659701 70.463711
## nu      16.5028022 34.136358
## sigma_x  2.6580491  2.799120
```

```
## sigma_y  2.6478446  2.805216
## attr(,"Probability")
## [1] 0.95
```

Given that the HDI for the difference in mean heights is 0.200548, 0.4626475 the interpretation here depends on our ROPE.