Homework 3

Daniel Kiser

March 16, 2018

Problem 1

Our implementation of a bivariate Gibbs sampler is shown below. As part of the function, an approximation of the pmf of s is calculated using the formula

$$P(S = s) = \frac{\{sum(s^{(i)}) = s\}}{M}$$

Here is the bivariate Gibbs sampler:

```
set.seed(444)

# Function for bivariate Gibbs sampler
bivar_gibbs <- function(M, n, s, theta) {
  alpha_0 <- 2.0
  beta_0 <- 6.4
  for(i in 2:M) {
    s[i] <- rbinom(1, n, theta[i - 1])
    theta[i] <- rbeta(1, shape1 = alpha_0 + s[i], shape2 = beta_0 + n - s[i])
  }

  # calculate approximate pmf of s
  pmf <- numeric(max(s)+1)
  for(i in 0:max(s)) {
    pmf[i+1] <- sum(s == i) / M
  }
  return(list(s, theta, pmf))
}
```

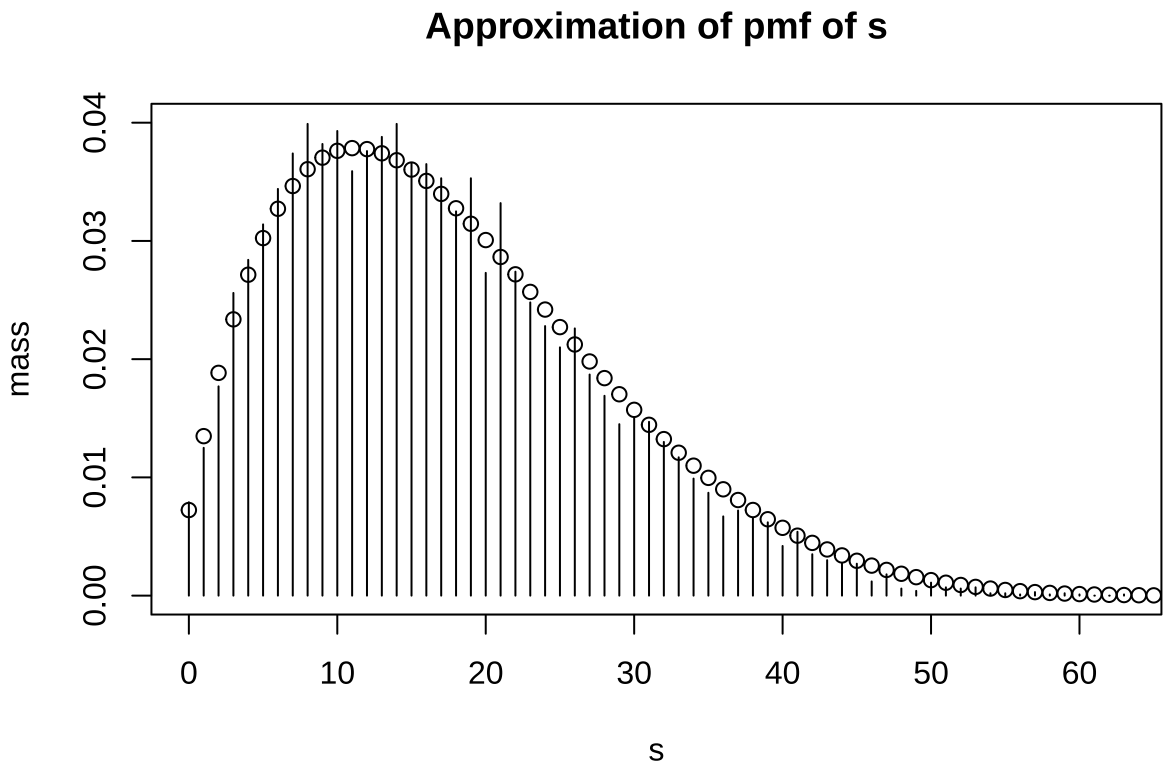
We use our bivariate Gibbs sampler to make 10000 draws of s and θ:

```
M <- 10000
n <- 74

empty_s <- numeric(M)
empty_s[1] <- 16

empty_theta <- numeric(M)
empty_theta[1] <- empty_s[1] / n

params <- bivar_gibbs(M, n, empty_s, empty_theta)
s <- unlist(params[1])
theta <- unlist(params[2])
pmf <- unlist(params[3])
```

Below is a visualization of the approximate pmf of s compared with the actual pmf. They appear to match closely:

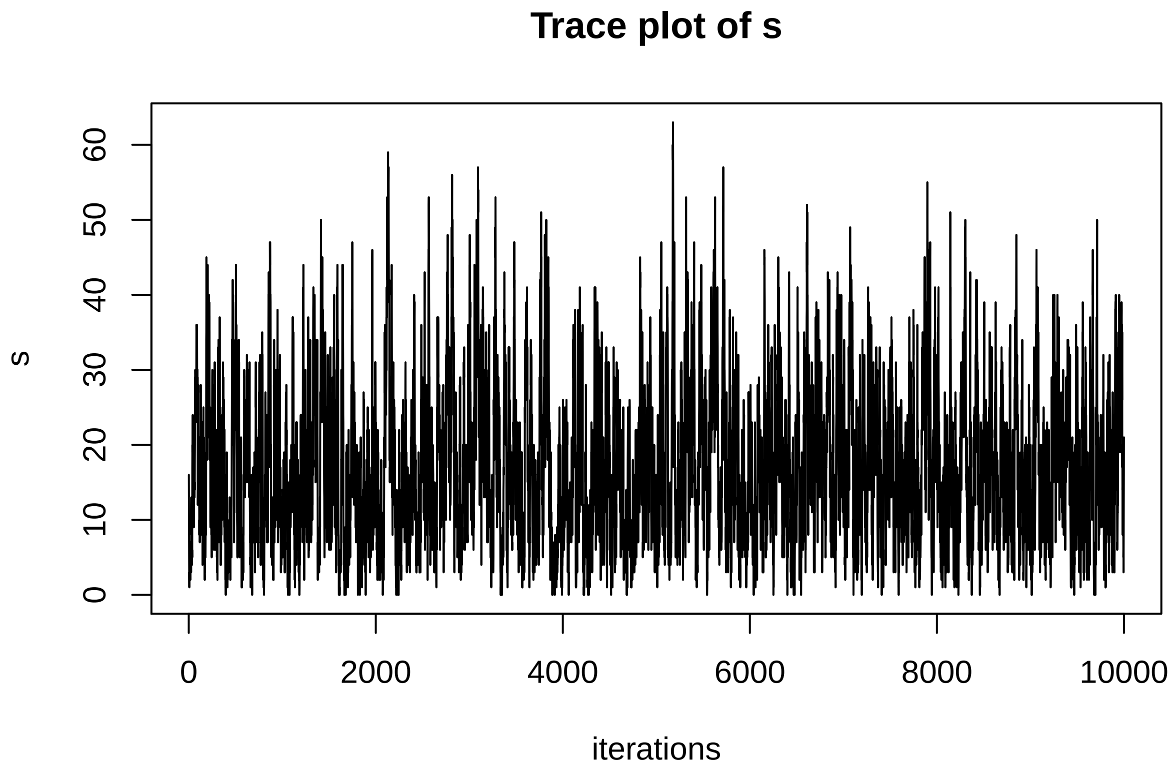
```
# Visualization of approximate pmf of s
s_index <- seq(0, length(pmf)-1)
```

```r
plot(s_index, pmf, main = "Approximation of pmf of s", type = "h", ylim = c(0, 0.04), xlab = "s",
     ylab = "mass")

# actual pmf of s
pmf_s <- function(s) {
  a_0 <- 2.0
  b_0 <- 6.4
  n <- 74
  mass <- choose(n, s) * (gamma(a_0 + b_0)*gamma(a_0 + s)*gamma(b_0 + n - s))/(gamma(a_0)*gamma(b_0)*ga
  return(mass)
}
it <- seq(0,74)
actual_density <- pmf_s(it)
s_index <- seq(0, length(actual_density)-1)
points(s_index, actual_density)
```

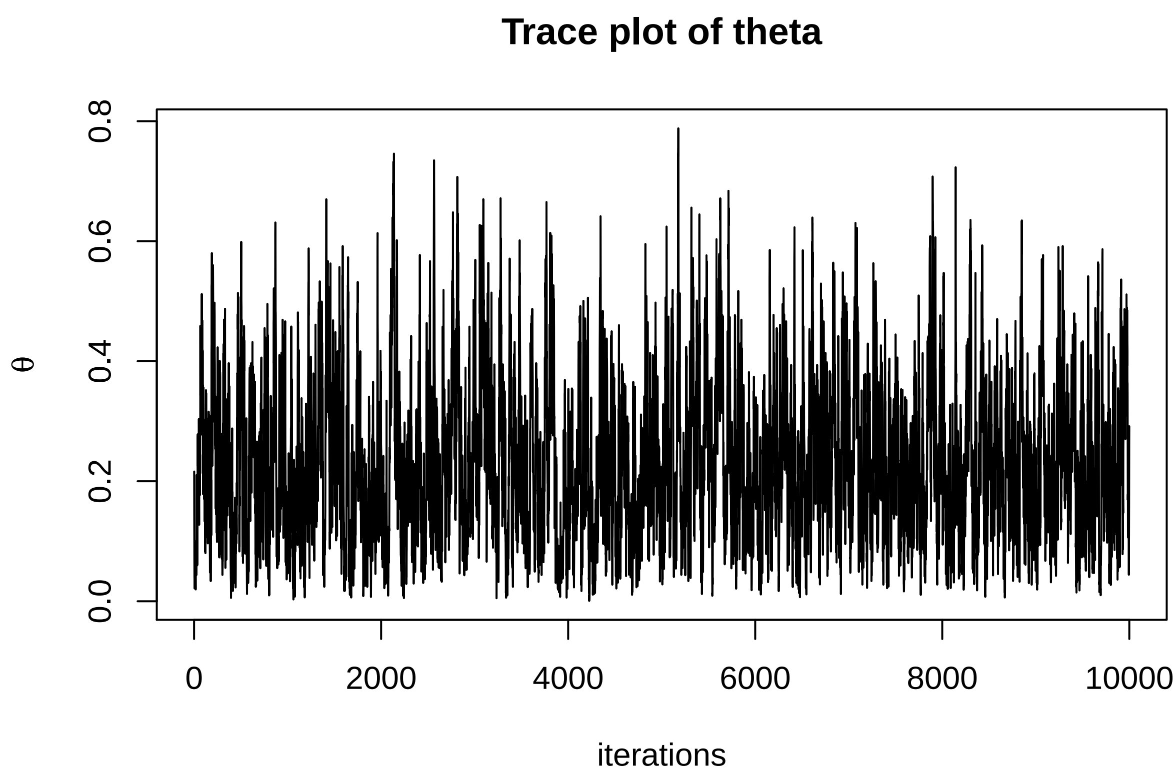


Below are two trace plots, one showing the variability in s, and the other showing the variability in θ. They both appear to converge:

```r
# Trace plot of s
plot.ts(s, type = "l", main = "Trace plot of s", xlab = "iterations")
```

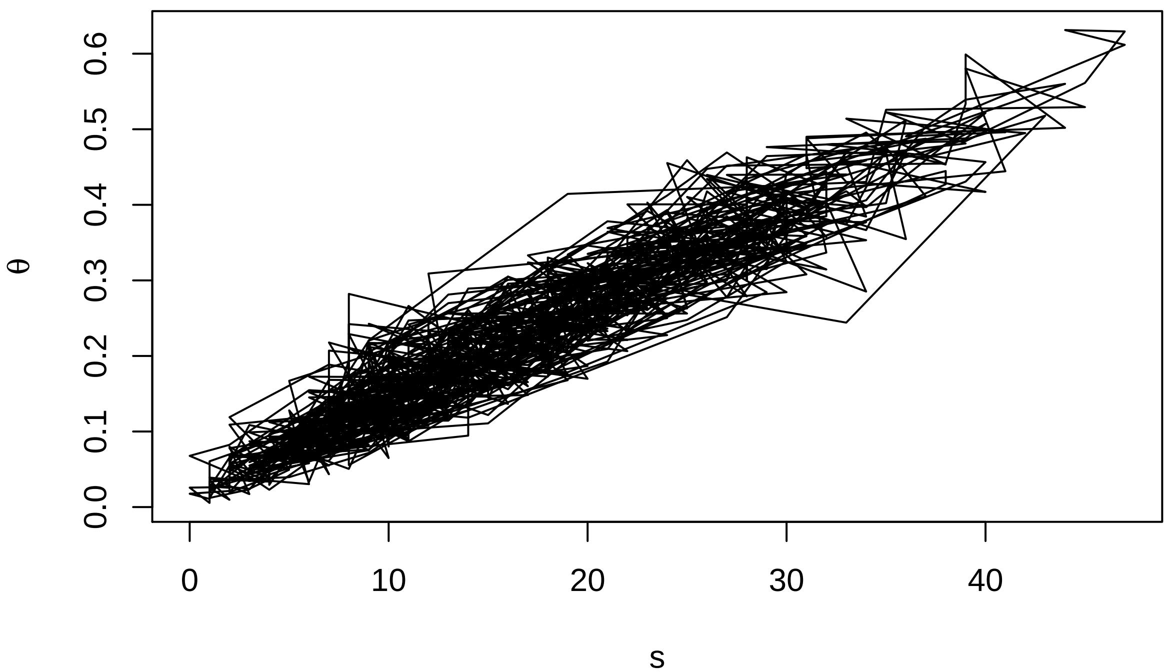



```r
# Trace plot of theta
plot.ts(theta, type = "l", main = "Trace plot of theta", xlab = "iterations",
        ylab = expression(theta))
```



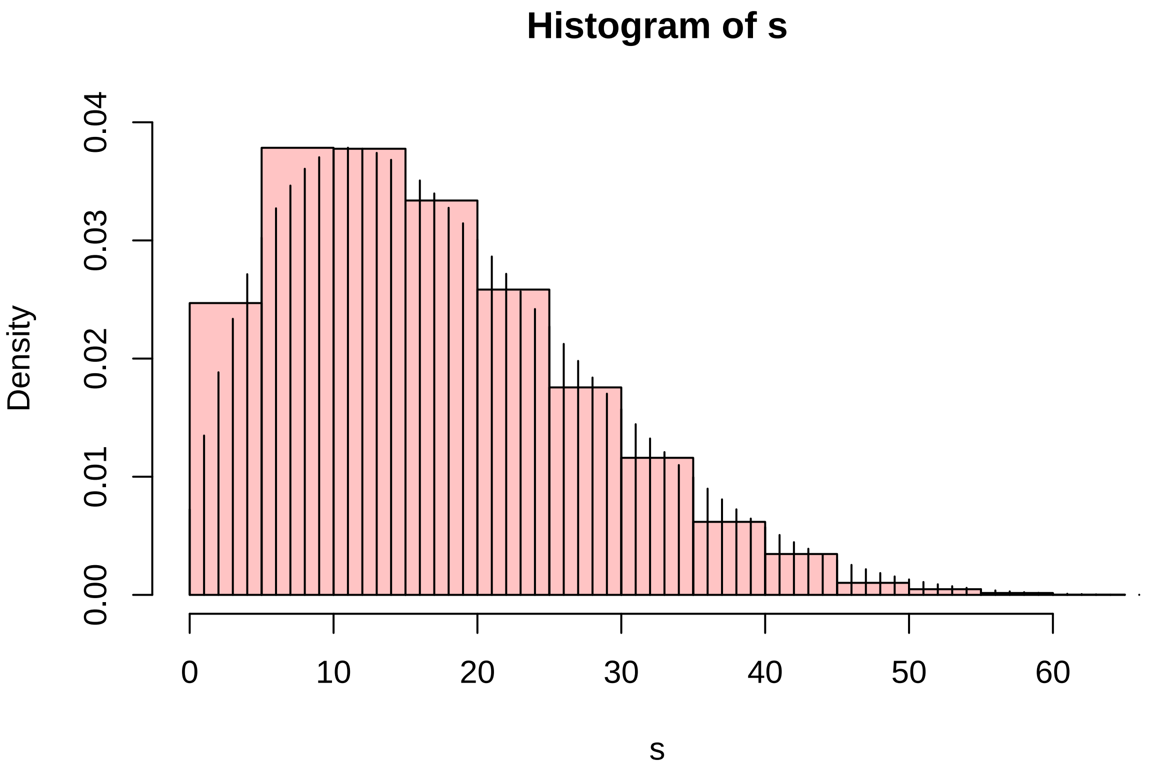
The next trace plot shows how s and θ vary together (only the first 1000 observations were plotted as it makes it easier to distinguish between areas of high density and low density):

```r
# Trace plot for the first 1000 obsevations of s and theta
plot(s[1:1000], theta[1:1000], type = "l", xlab = "s", ylab = expression(theta))
```



The histogram below shows the approximate distribution of s, with the actual pmf overlaid:

```r
# set colors for plots
col1 <- rgb(0, 0, 255, max = 255, alpha = 60)
col2 <- rgb(255, 0, 0, max = 255, alpha = 60)

# Histogram of s with approximate pmf overlaid
hist(s, freq = F, ylim = c(0, 0.04), col = col2)
points(s_index, actual_density, type = "h")
```




```
# store results for comparison with new sampler with n not fixed
s_n_fixed <- s
pmf_n_fixed <- pmf
theta_n_fixed <- theta

# median of theta
median_n_fixed <- median(theta)
median_n_fixed
```

```
## [1] 0.2110301
```

The Monte Carlo estimate of θ is 0.2110301, as shown above. Since $\hat{\theta}_{MLE} = \frac{s}{n}$, our MLE estimate is:

```
16/74
```

```
## [1] 0.2162162
```

which is very similar to the Monte Carlo estimate of θ.

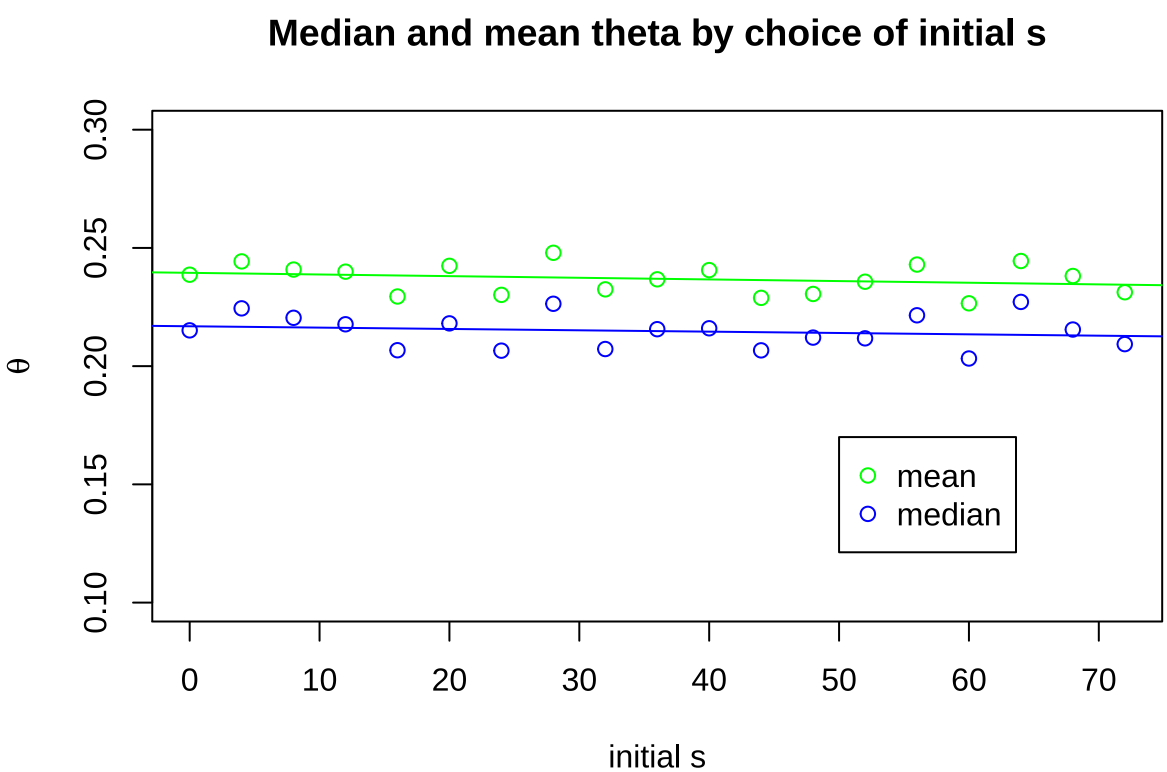
To determine how sensitive the posterior median is to the choice of initial values, we rerun the Gibbs sampler with many different initial values of s (still using $\frac{s}{n}$ as our initial value for θ):

```
# Rerun with different initial values and compare means and medians
M <- 10000
trial_s <- seq(0, 72, 4)
median_theta <- numeric(length(trial_s))
mean_theta <- numeric(length(trial_s))
j = 0
for(i in trial_s) {
  empty_s <- numeric(M)
  empty_s[1] <- i

  empty_theta <- numeric(M)
```

```r
  empty_theta[1] <- empty_s[1] / n

  params <- bivar_gibbs(M, n, empty_s, empty_theta)
  theta <- unlist(params[2])

  j = j + 1
  median_theta[j] <- median(theta)
  mean_theta[j] <- mean(theta)
}
```

The results were fitted with a least-squares line to help discern whether or not the median changed with different initial values of s. The plot below visualizes the results (the mean was also included, for comparison).

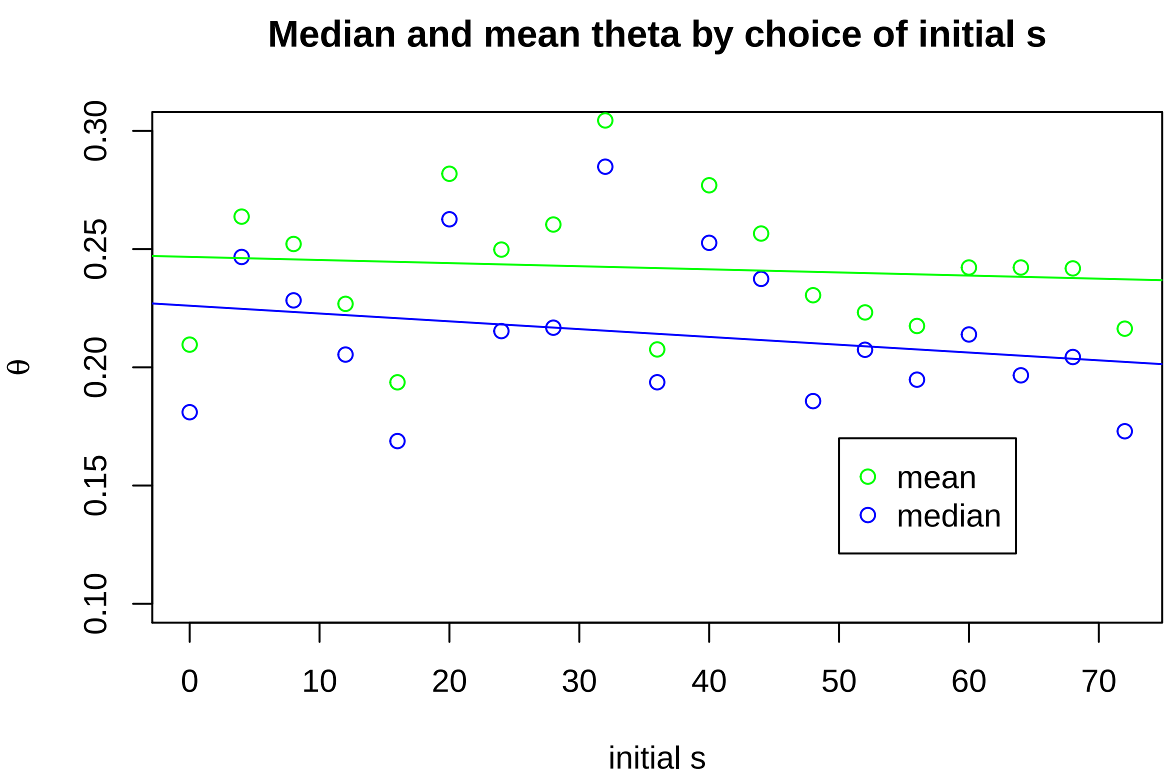
```r
plot(trial_s, median_theta, ylim = c(0.1, 0.3), col = "blue",
     main = "Median and mean theta by choice of initial s",
     xlab = "initial s", ylab = expression(theta))
fit1 <- lm(median_theta ~ trial_s)
abline(fit1, col = "blue")
points(trial_s, mean_theta, col = "green")
fit2 <- lm(mean_theta ~ trial_s)
abline(fit2, col = "green")
legend(50, 0.17, legend = c("mean", "median"), col = c("green", "blue"), pch = 1)
```



As can be seen in the plot, both the mean and median varied only slightly from the fitted least-squares line, and there was no general trend as the initial values of s increased. However, if a smaller sample is taken, say 500 instead of 10000 observations, the median and the mean may change somewhat with different starting values. See the plot below:

```r
# With fewer trials, starting value matters more
M <- 500
trial_s <- seq(0, 72, 4)
```

```r
median_theta <- numeric(length(trial_s))
mean_theta <- numeric(length(trial_s))
j = 0
for(i in trial_s) {
  empty_s <- numeric(M)
  empty_s[1] <- i

  empty_theta <- numeric(M)
  empty_theta[1] <- empty_s[1] / n

  params <- bivar_gibbs(M, n, empty_s, empty_theta)
  theta <- unlist(params[2])

  j = j + 1
  median_theta[j] <- median(theta)
  mean_theta[j] <- mean(theta)
}
plot(trial_s, median_theta, ylim = c(0.1, 0.3), col = "blue",
     main = "Median and mean theta by choice of initial s",
     xlab = "initial s", ylab = expression(theta))
fit1 <- lm(median_theta ~ trial_s)
abline(fit1, col = "blue")
points(trial_s, mean_theta, col = "green")
fit2 <- lm(mean_theta ~ trial_s)
abline(fit2, col = "green")
legend(50, 0.17, legend = c("mean", "median"), col = c("green", "blue"), pch = 1)
```



Here the median of θ appears to get slightly smaller as s increases, but the main difference is that the variability of the estimates increased dramatically. Thus, it seems that as long as a large enough sample is taken, the mean and median of the samples is robust and is an accurate representation of the target distribution.

Problem 2

If we treat n as an unknown parameter, then the posterior distribution of s given θ and n remains the same as before, since $s|\theta$ already depends on n. We can also demonstrate that the conditional distribution of $\theta|s,n$ is the same as $\theta|s$, since

$$p(\theta|s,n) \propto p(s|n,\theta)p(\theta) \propto \theta^s(1-\theta)^{n-s}\theta^{\alpha_0-1}(1-\theta)^{\beta_0-1} \propto \theta^{\alpha_0+s-1}(1-\theta)^{\beta_0+n-s-1}$$

Thus, $\theta|s,n \sim Beta(\alpha_0+s, \beta_0+n-s)$, the same as $\theta|s$.

To find the posterior distribution $n|\theta,s$ we use Bayes Theorem:

$$p(n|s,\theta) = \frac{p(s|n,\theta)p(n)}{p(s)}$$

Because the support of S depends on the unknown parameter n, we must find the normalizing constant, which is given by:

$$p(s) = \sum_{n=s}^{\infty} p(s|n)p(n) = \sum_{n=s}^{\infty} \binom{n}{s}\theta^s(1-\theta)^{n-s}e^{-\lambda}\frac{\lambda^n}{n!} = \sum_{n=s}^{\infty} \frac{n!}{s!(n-s)!}\theta^s(1-\theta)^{n-s}e^{-\lambda}\frac{\lambda^n}{n!}$$

$$= \sum_{n=s}^{\infty} \frac{1}{s!(n-s)!}\theta^s(1-\theta)^{n-s}e^{-\lambda}\lambda^n = \frac{e^{-\lambda}\theta^s}{s!}\sum_{n=s}^{\infty}\frac{1}{(n-s)!}(1-\theta)^{n-s}\lambda^n$$

$$= \frac{e^{-\lambda}\theta^s}{s!}\sum_{n=s}^{\infty}\frac{1}{(n-s)!}(1-\theta)^{n-s}\lambda^{n-s}\lambda^s$$

$$= \frac{e^{-\lambda}\theta^s\lambda^s}{s!}\sum_{n=s}^{\infty}\frac{1}{(n-s)!}(1-\theta)^{n-s}\lambda^{n-s} = \frac{e^{-\lambda}\theta^s\lambda^s}{s!}\sum_{n=s}^{\infty}\frac{(\lambda(1-\theta))^{n-s}}{(n-s)!} = \frac{e^{-\lambda}\theta^s\lambda^s}{s!}\frac{1}{e^{-\lambda(1-\theta)}}$$

The last step is possible because we recognize the summation as the sum over the support of a Poisson density without the normalizing factor, $e^{-\lambda(1-\theta)}$. We then simplify:

$$p(s) = \frac{e^{-\lambda+\lambda(1-\theta)}\theta^s\lambda^s}{s!}$$

Now that we have the normalizing factor, we can find the posterior distribution of n given s and θ:

$$p(n|s,\theta) = \frac{p(s|n,\theta)p(n)}{p(s)} = \frac{\binom{n}{s}\theta^s(1-\theta)^{n-s}\frac{e^{-\lambda_0}\lambda_0^n}{n!}}{\frac{e^{-\lambda_0+\lambda_0(1-\theta)}\theta^s\lambda_0^s}{s!}} = \frac{\frac{n!}{s!(n-s)!}\theta^s(1-\theta)^{n-s}e^{-\lambda_0}\lambda_0^n s!}{n!e^{-\lambda_0+\lambda_0(1-\theta)}\lambda_0^s\theta^s}$$

$$= \frac{(1-\theta)^{n-s}e^{-\lambda_0}\lambda_0^{n-s}}{(n-s)!e^{-\lambda_0+\lambda_0(1-\theta)}} = \frac{e^{-\lambda_0(1-\theta)}(\lambda_0(1-\theta))^{n-s}}{(n-s)!}$$

Thus, $n|s,\theta \sim Poi(\lambda)$, where $\lambda = \lambda_0(1-\theta)$. However, the above density is the distribution of $n-s$. If we want to draw a sample of n using this density, we must take the values we draw from this distribution and sum them with s:

$$n_{sample} = s + N$$

where N is a random value drawn from $Poi(\lambda_0(1-\theta))$. Below is the implementation of the new Gibbs sampler:

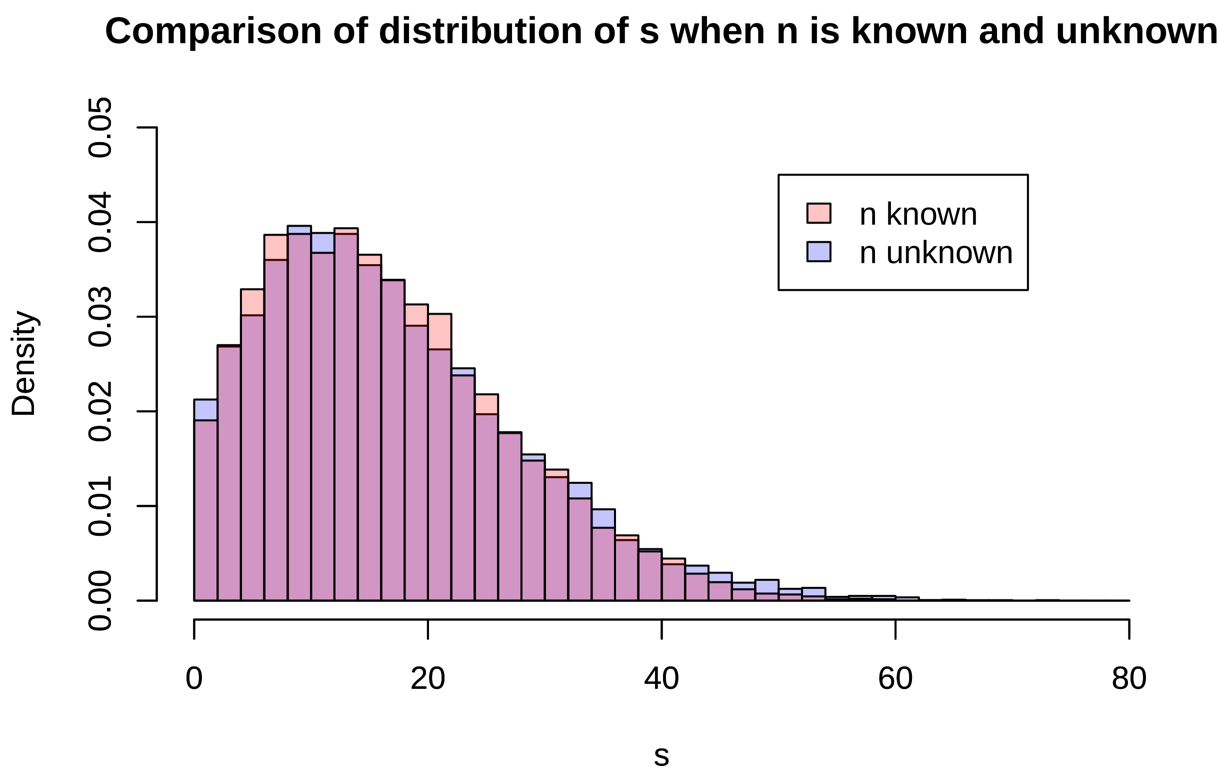
```r
# Function for trivariate Gibbs sampler
trivar_gibbs <- function(M, n, s, theta, lambda) {
  alpha_0 <- 2.0
  beta_0 <- 6.4
  lambda <- 74
  for(i in 2:M) {
    s[i] <- rbinom(1, n[i-1], theta[i - 1])
    theta[i] <- rbeta(1, shape1 = alpha_0 + s[i], shape2 = beta_0 + n[i-1] - s[i])
    n[i] <- s[i] + rpois(1, lambda*(1 - theta[i]))
  }

  # calculate approximate pmf of s
  pmf <- numeric(max(s))
  for(i in 0:max(s)) {
    pmf[i] <- sum(s == i) / M
  }
  return(list(s, theta, n, pmf, lambda))
}
```

In order to confirm that we have calculated the conditional distributions of s, θ, and n correctly, we use the trivariate Gibbs sampler to draw samples of s, θ, and n, and we compare the distributions of s and θ with the distributions of the samples drawn before.

```r
M <- 10000

empty_s <- numeric(M)
empty_s[1] <- 16

empty_n <- numeric(M)
empty_n[1] <- 74

empty_theta <- numeric(M)
empty_theta[1] <- empty_s[1] / empty_n[1]

params <- trivar_gibbs(M, empty_n, empty_s, empty_theta, empty_lambda)
s <- unlist(params[1])
theta <- unlist(params[2])
pmf <- unlist(params[3])
```

We compare our samples of s by overlaying the histograms of both samples. We see that their distributions are practically identical:

```r
# Histogram of s with n known and unknown
hist(s, freq = F, breaks = pretty(0:80, n = 50), ylim = c(0, 0.05), main = "", xlab = "s",
     col = col1)
opar <- par(no.readonly = TRUE)
par(new = TRUE)
hist(s_n_fixed, freq = F, breaks = pretty(0:80, n = 50), ylim = c(0, 0.05), xlab = "s",
     col = col2, main = "Comparison of distribution of s when n is known and unknown")
legend(x = 50, y = 0.045, legend = c("n known", "n unknown"), fill = c(col2, col1))
```




```
par(opar)
```

We also compare the medians of our samples of θ. They are very similar to each other, and they are also both similar to the MLE of θ, which is $\frac{s}{n}$.

Median when n is unknown:

```
median(theta)
```

```
## [1] 0.2137272
```

Median when n is known:

```
median_n_fixed
```

```
## [1] 0.2110301
```

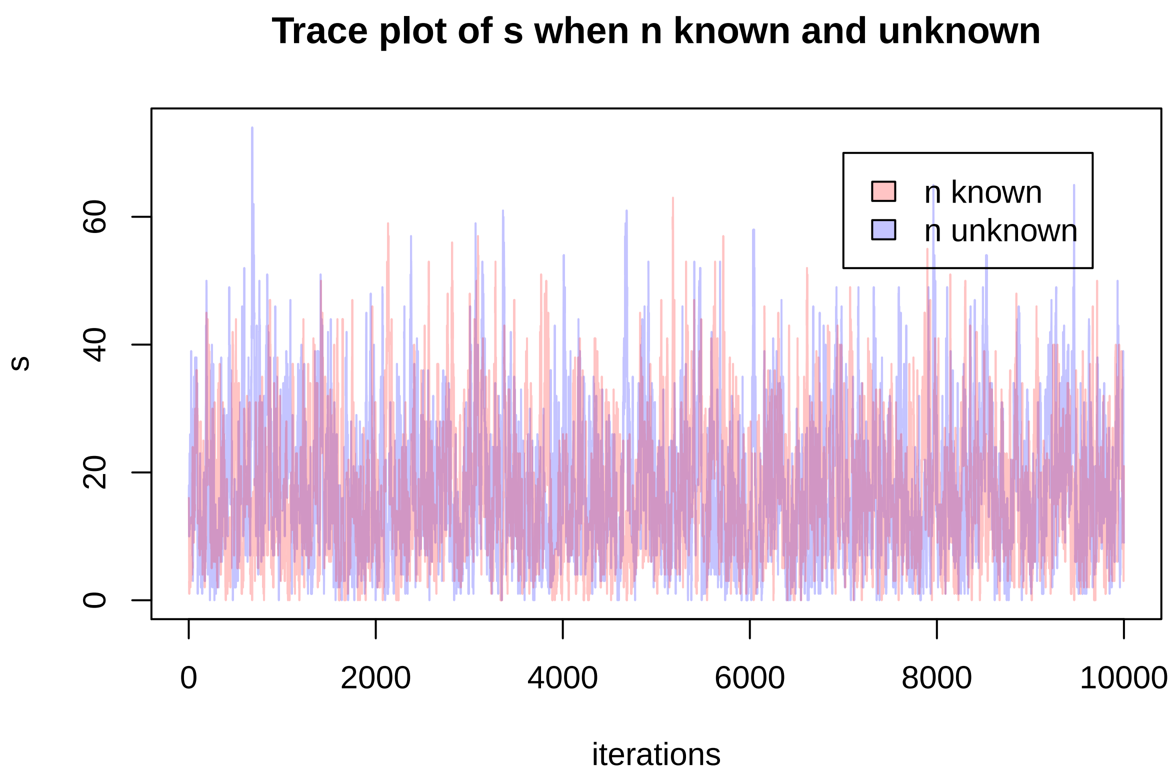
MLE of θ:

```
16/74
```

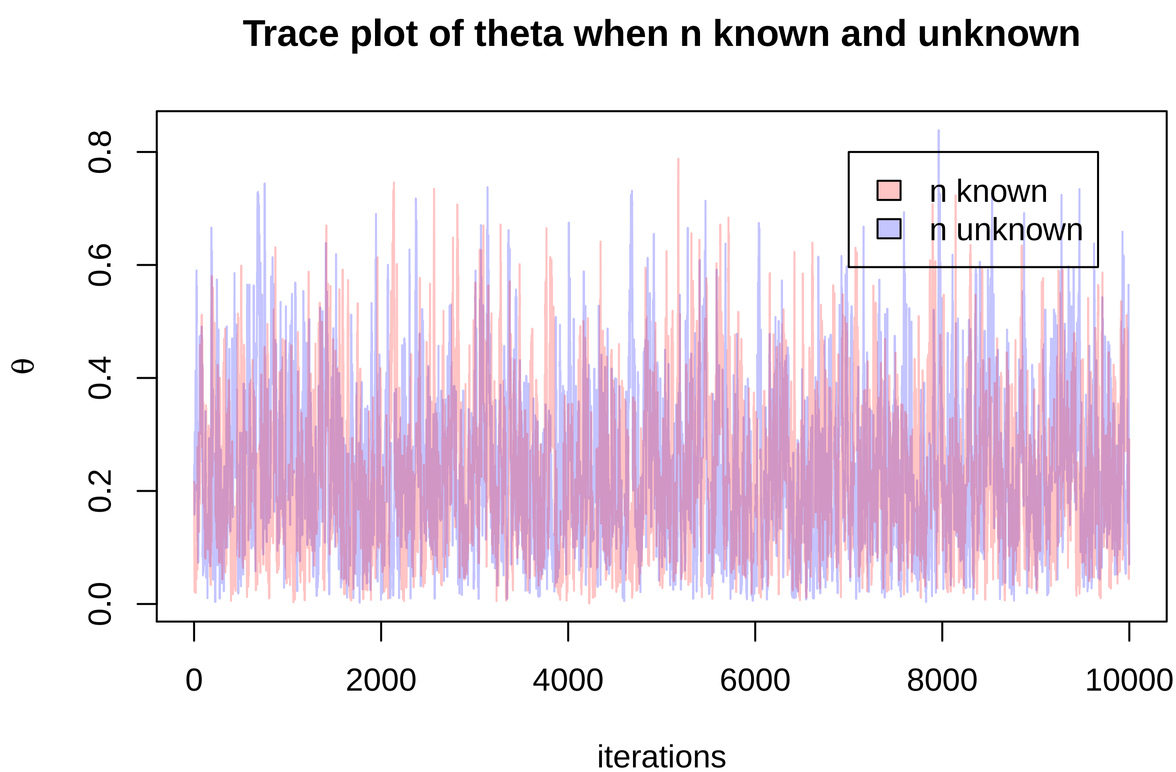
```
## [1] 0.2162162
```

To compare the convergence of our too samplers, we overlay the trace plots produced for s and θ when n is known and unknown. We see that the plots are practically indistinguishable from each other, indicating that we get similar convergence results with both samplers:

```
# Compare convergence results using trace plots
# Trace plot of s
plot(s, type = "l", main = "Trace plot of s when n known and unknown", col = col1,
     xlab = "iterations")
lines(s_n_fixed, col = col2)
legend(x = 7000, y = 70, legend = c("n known", "n unknown"), fill = c(col2, col1))
```




```r
# Trace plot of theta
plot(theta, type = "l", main = "Trace plot of theta when n known and unknown", col = col1,
     xlab = "iterations", ylab = expression(theta))
lines(theta_n_fixed, col = col2)
legend(x = 7000, y = 0.8, legend = c("n known", "n unknown"), fill = c(col2, col1))
```



We also make a plot for both n-known and n-unknown showing how s and θ vary together. Here we can see that the range of s and θ is slightly larger when n is not fixed, but their values still converge to the same center. This makes sense, considering that three parameters are allowed to vary instead of only two.

```
# Trace plot for both s and theta
plot(s[1:10000], theta[1:10000], type = "l", col = col1,
     main = "Trace plot of s and theta when n known and unknown", ylab = expression(theta),
     xlab = "s")
lines(s_n_fixed[1:10000], theta_n_fixed[1:10000], col = col2, ylab = "", xlab = "")
legend(x = 50, y = 0.3, legend = c("n known", "n unknown"), fill = c(col2, col1))
```