Chapter 2 — Notes

3.1 Sampling from a grid-approximate posterior

- R Code 3.2:

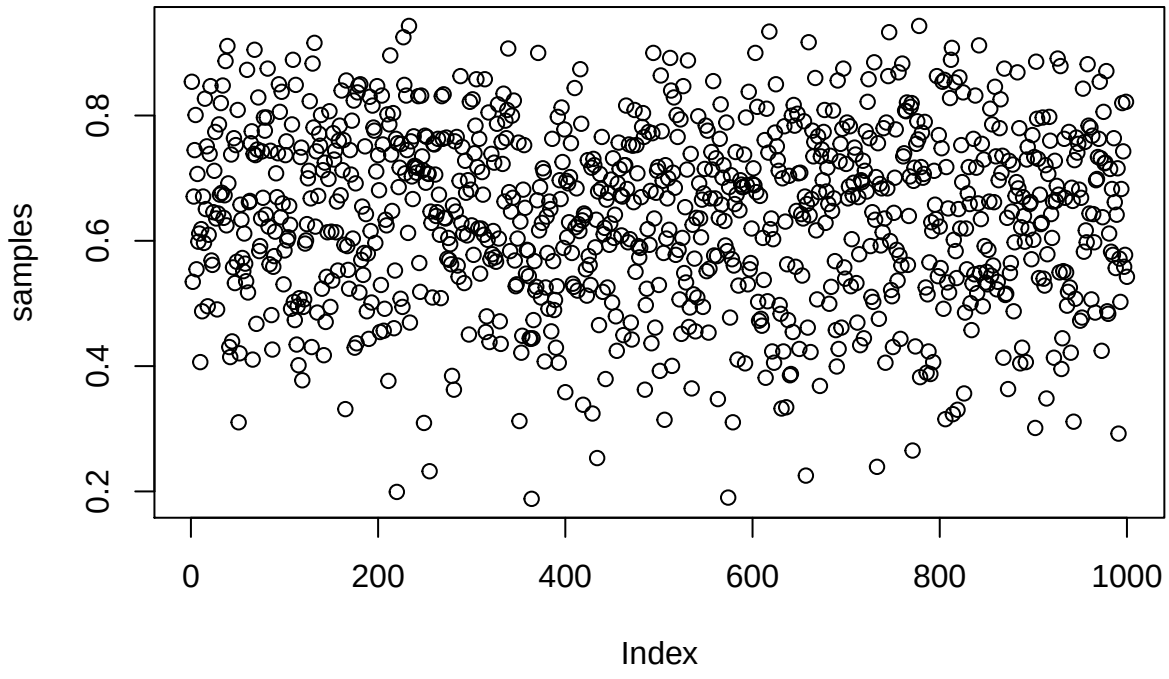
```
n = 1000
p_grid <- seq(from=0, to=1, length.out=n)
prior <- rep(1, n)
likelihood <- dbinom(x=6, size=9, prob=p_grid)
posterior_notnorm <- likelihood * prior
posterior <- posterior_notnorm / sum(posterior_notnorm)
```

Draw 10,000 samples: * R Code 3.3:

```
samples <- sample(p_grid, prob=posterior, size=n, replace=T)
```

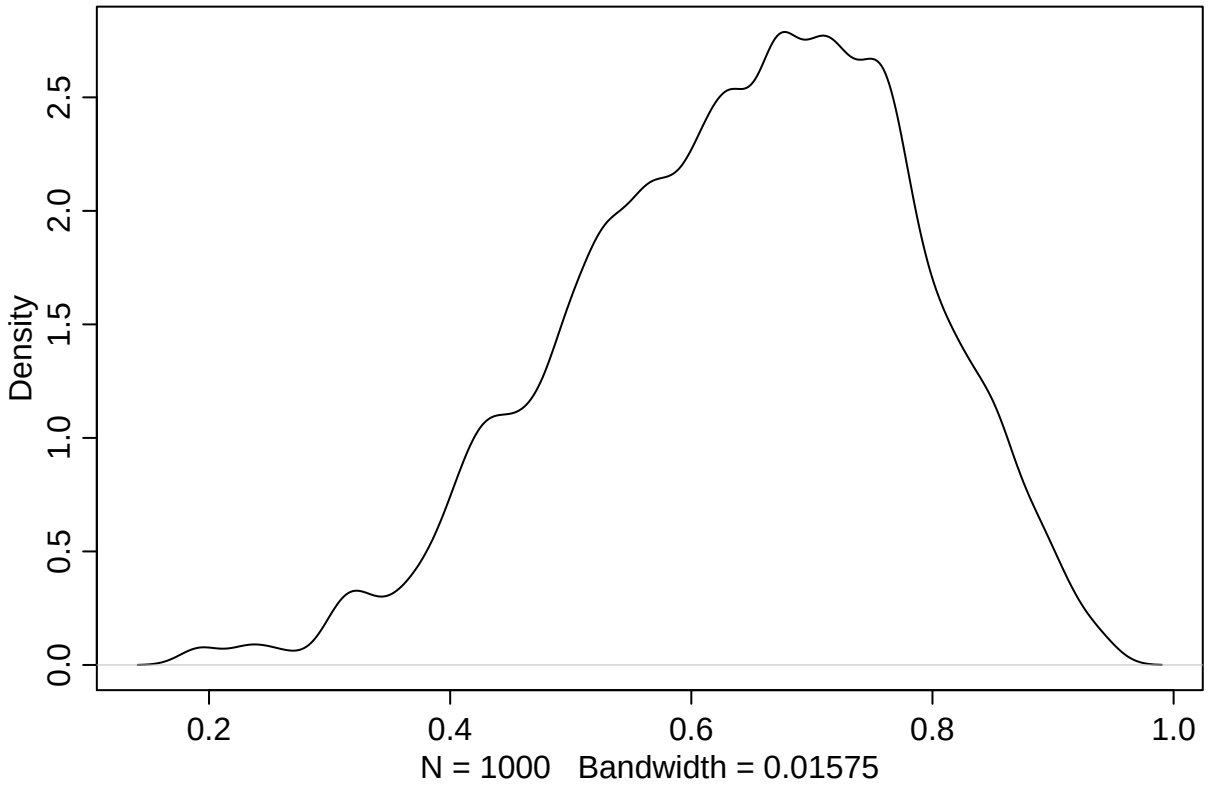
- 3.4:

```
plot(samples)
```



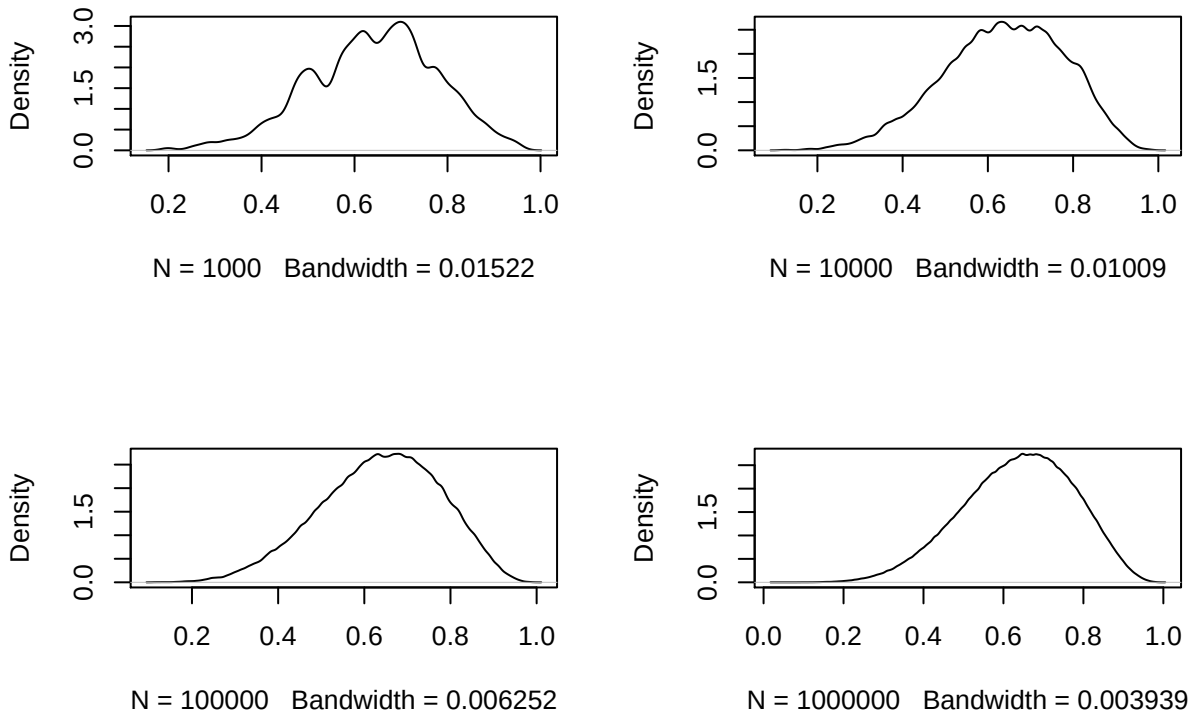
- 3.5:

```
dens(samples)
```



Let's try more samples:

```
par(mfrow=c(2, 2))
dens(sample(p_grid, prob=posterior, size=1e3, replace=T))
dens(sample(p_grid, prob=posterior, size=1e4, replace=T))
dens(sample(p_grid, prob=posterior, size=1e5, replace=T))
dens(sample(p_grid, prob=posterior, size=1e6, replace=T))
```



3.2 Sampling to Summarize

3.2.1. Intervals of defined boundaries.

The posterior probability that the proportion of water is less than 0.5:

- 3.6:

```
p_grid < 0.5
```

```
##   [1]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [12]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [23]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [34]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [45]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [56]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [67]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [78]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [89]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [100]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [111]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [122]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [133]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [144]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [155]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
## [166]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
```

```
##  [177]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [188]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [199]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [210]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [221]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [232]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [243]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [254]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [265]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [276]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [287]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [298]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [309]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [320]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [331]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [342]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [353]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [364]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [375]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [386]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [397]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [408]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [419]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [430]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [441]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [452]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [463]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [474]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [485]  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE  TRUE
##  [496]  TRUE  TRUE  TRUE  TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
##  [507] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [518] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [529] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [540] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [551] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [562] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [573] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [584] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [595] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [606] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [617] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [628] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [639] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [650] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [661] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [672] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [683] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [694] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [705] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [716] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [727] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [738] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [749] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [760] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
##  [771] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [782] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [793] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [804] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [815] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [826] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [837] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [848] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [859] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [870] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [881] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [892] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [903] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [914] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [925] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [936] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [947] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [958] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [969] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [980] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
##  [991] FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
```

```
sum(posterior[p_grid < 0.5])
```

```
## [1] 0.1718746
```

Samples array:

```
head(samples, 100)
```

```
##   [1] 0.8538539 0.5345345 0.6706707 0.7447447 0.8008008 0.5545546 0.7067067
##   [8] 0.5985986 0.6116116 0.4064064 0.6186186 0.4874875 0.6706707 0.5965966
##  [15] 0.8268268 0.6506507 0.7507508 0.4954955 0.6096096 0.7387387 0.8468468
##  [22] 0.5685686 0.5615616 0.6456456 0.7117117 0.7737738 0.6356356 0.4904905
##  [29] 0.6436436 0.7857858 0.6736737 0.8198198 0.6766767 0.8478478 0.6366366
##  [36] 0.6726727 0.8868869 0.6246246 0.9109109 0.6916917 0.4304304 0.4144144
##  [43] 0.7377377 0.4394394 0.5575576 0.7637638 0.5325325 0.7527528 0.5665666
##  [50] 0.8088088 0.3103103 0.4204204 0.6576577 0.6336336 0.5725726 0.5515516
##  [57] 0.6116116 0.5635636 0.5355355 0.8728729 0.5175175 0.6616617 0.6636637
##  [64] 0.7557558 0.7747748 0.4104104 0.7387387 0.9049049 0.7367367 0.4674675
##  [71] 0.7457457 0.8288288 0.5835836 0.5915916 0.6356356 0.7427427 0.6686687
##  [78] 0.7967968 0.7757758 0.6416416 0.7967968 0.8748749 0.6226226 0.5625626
##  [85] 0.7437437 0.4814815 0.4264264 0.5585586 0.5755756 0.6146146 0.7077077
##  [92] 0.6696697 0.7397397 0.8498498 0.8058058 0.6376376 0.8378378 0.6596597
##  [99] 0.5305305 0.5845846
```

The same calculation using samples. Add up all samples that lie in the grid < 0.5, and divide by the total number of samples to get the frequency $\sim$ probability:

- 3.7:

```
n = 1e4
samples = sample(p_grid, prob=posterior, size=n, replace=T)
sum(samples < 0.5) / n
```

```
## [1] 0.174
```

How much probability lies between 0.5 and 0.75: * 3.8:

```
sample_points = sum(samples > 0.5 & samples < 0.75)
sample_points
```

```
## [1] 5982
```

```
sample_points / n
```

```
## [1] 0.5982
```

3.2.2. Intervals of defined mass.

Boundaries of the lower 80% posterior probability lies:

- 3.9:

```
quantile(samples, probs = .8)
```

```
##       80%
## 0.7617618
```

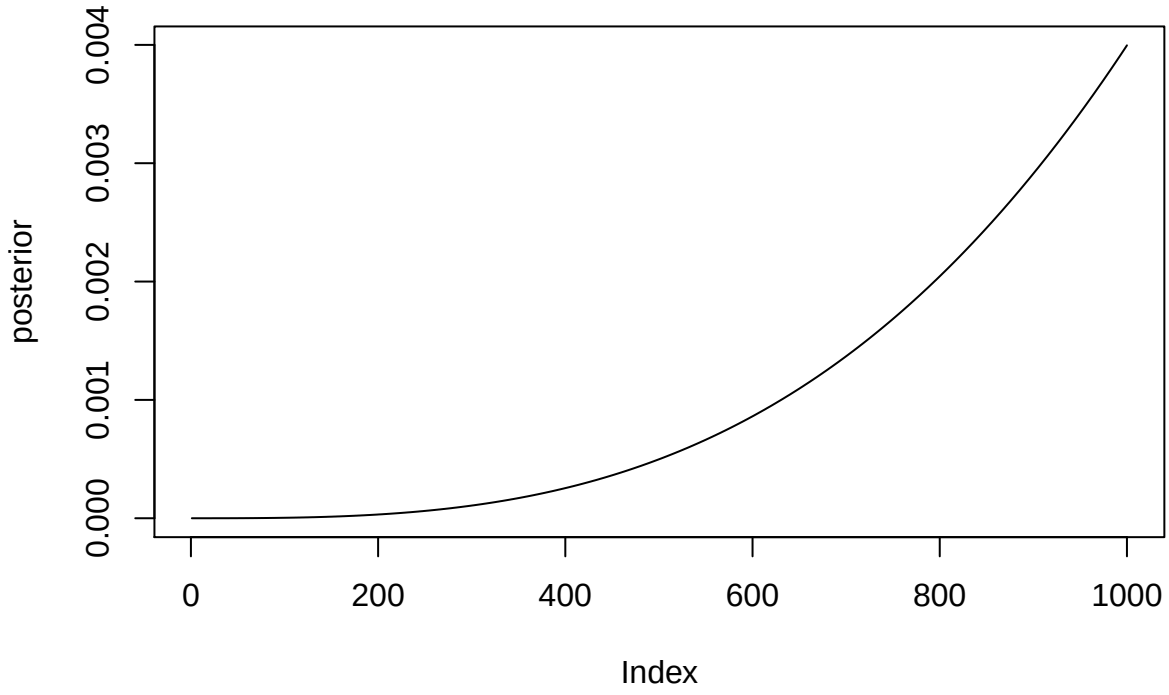
Middle 80%, i.e. lying between 10% and 90%:

```
# 3.10
quantile(samples, probs = c(0.1, 0.9))
```

```
##       10%       90%
## 0.4474474 0.8118118
```

The above are PERCENTILE INTERVALS. Percentiles can be misleading if the distribution is highly skewed.

```
# 3.11
n <- 1000
p_grid <- seq(0, 1, length.out = n)
prior <- rep(1, n)
likelihood <- dbinom(3, size=3, prob=p_grid)
posterior_notnorm <- likelihood * prior
posterior <- posterior_notnorm / sum(posterior_notnorm)
samples <- sample(p_grid, size=1e4, replace=T, prob=posterior)
plot(posterior, type='l')
```




```
# 3.12
PI(samples, prob=0.5)
```

```
##       25%       75%
## 0.7127127 0.9309309
```

Highest Posterior Density Interval described the distribution better. It's the *narrowest* interval containing the specified probability mass, e.g. 50%.

```
# 3.13
HPDI(samples, prob=0.5)
```

```
##      |0.5      0.5|
## 0.8408408 1.0000000
```

3.2.3. Point Estimates

A parameter with the highest posterior probability is called *a maximum a posteriori* estimate, or *MAP*.

```
# 3.14
which.max(posterior)
```

```
## [1] 1000
```

```
p_grid[which.max(posterior)]
```

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

Use samples to get the same (or similar) result:

```r
# 3.15
chainmode(samples, adj=0.01)
```

```
## [1] 0.99704
```

```r
# 3.16
mean(samples)
```

```
## [1] 0.8004087
```

```r
median(samples)
```

```
## [1] 0.8408408
```