7.1 simulation

Chris Parrish

July 3, 2016

Contents

7.1 simulation of probability models

reference:
- ARM chapter 07, github

```
library(rstan)
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
library(ggplot2)
```

a simple example of discrete predictive simulations

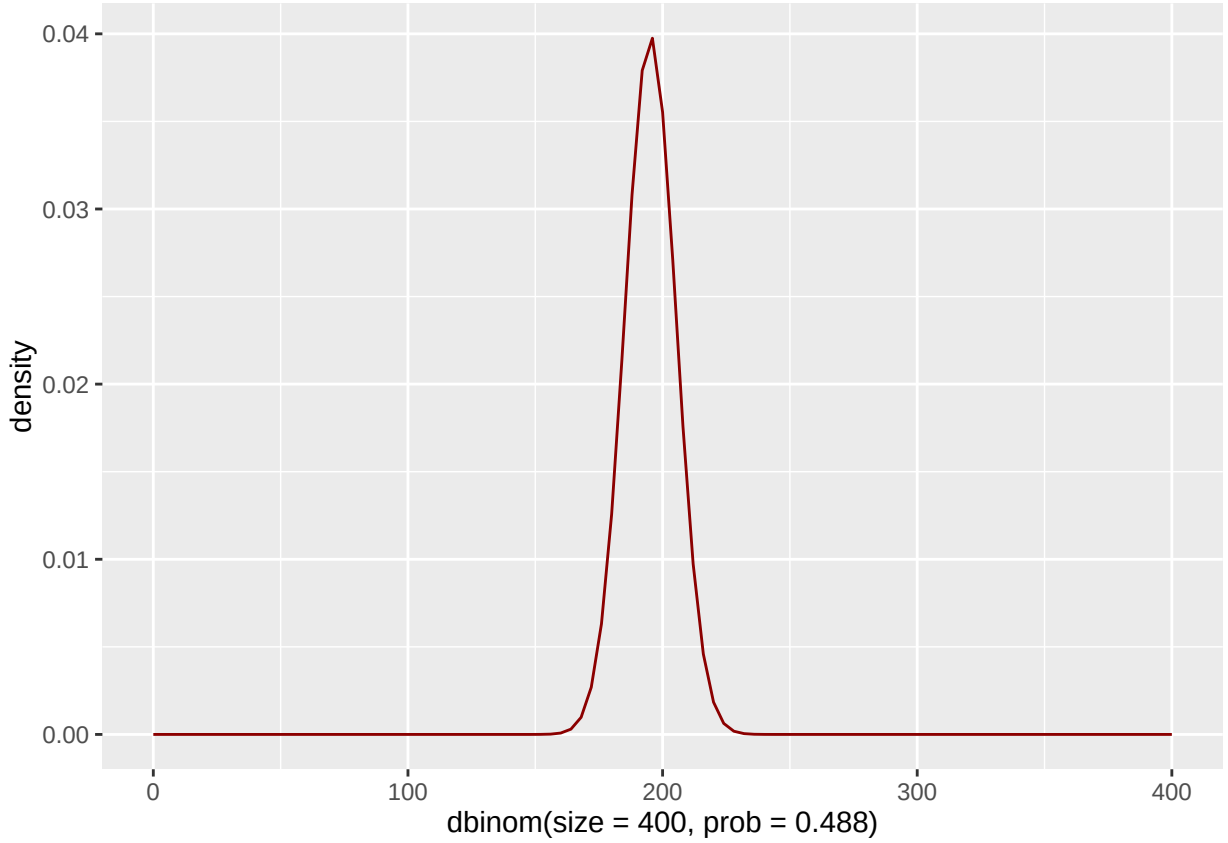
simulate 400 births; report the number of girls born

```
## A simple example of discrete predictive simulations
p.girl <- 0.488
p.boy <- 0.512
n.births <- 400
n.girls <- rbinom (1, 400, .488)
print (n.girls)
```

```
## [1] 177
```

distribution of Binomial(k, n, p)

```
data <- data.frame(x = c(0, 400), y = c(-0.05, 0.4))
ggplot(data, aes(x, y)) +
  stat_function(fun = dbinom, args = list(size = 400, prob = 0.488), color = "darkred") +
  labs(x = "dbinom(size = 400, prob = 0.488)", y = "density")
```




```
round(rbind(mean = n.births * p.girl,
            sd = sqrt(n.births * p.girl * (1 - p.girl))), 3)
```

```
##         [,1]
## mean 195.200
## sd     9.997
```

distribution of values (using a loop)

```
n.sims <- 1000
n.girls <- rep (NA, n.sims)
for (s in 1:n.sims){
  n.girls[s] <- rbinom (1, 400, .488)
}
str(n.girls)
```

```
##  int [1:1000] 193 194 191 192 199 187 197 195 190 193 ...
```

distribution of values (vectorized)

```
n.sims <- 1000
n.girls <- rbinom(n.sims, 400, .488)
str(n.girls)
```

```
##  int [1:1000] 193 189 205 181 197 187 196 211 188 215 ...
```

```
rbind(mean = mean(n.girls), sd = sd(n.girls), proportion.girls = mean(n.girls) / n.births)
```

```
##                         [,1]
## mean              194.730000
## sd                 10.130534
## proportion.girls    0.486825
```

figure

```
frame1 = data.frame(x1 = n.girls)
m <- ggplot(frame1, aes(x = x1))  +
  scale_x_continuous("Number of Girls") +
  geom_histogram(aes(y = ..density..),
                 colour = "seashell", fill = "wheat", binwidth=5) +
  geom_density(color = "darkred") +
  theme_gray()
m
```

accounting for twins

```
## Accounting for twins
birth.type <- sample (c("fraternal twin", "identical twin", "single birth"),
  size=400, replace=TRUE, prob=c(1/125, 1/300, 1 - 1/125 - 1/300))
girls <- rep (NA, 400)
for (i in 1:400){
  if (birth.type[i]=="single birth"){
   girls[i] <- rbinom (1, 1, .488)}
  else if (birth.type[i]=="identical twin"){
   girls[i] <- 2*rbinom (1, 1, .495)}
  else if (birth.type[i]=="fraternal twin"){
   girls[i] <- rbinom (1, 2, .495)}
}
n.girls <- sum (girls)
```

putting in a loop

```
 # putting in a loop
n.sims <- 1000
n.girls <- rep (NA, n.sims)
for (s in 1:n.sims){
 birth.type <- sample (c("fraternal twin", "identical twin", "single birth"),
   size=400, replace=TRUE, prob=c(1/125, 1/300, 1 - 1/125 - 1/300))
 girls <- rep (NA, 400)
 for (i in 1:400){
  if (birth.type[i]=="single birth"){
   girls[i] <- rbinom (1, 1, .488)}
  else if (birth.type[i]=="identical twin"){
   girls[i] <- 2*rbinom (1, 1, .495)}
  else if (birth.type[i]=="fraternal twin"){
   girls[i] <- rbinom (1, 2, .495)}
 }
 n.girls[s] <- sum (girls)
}
```

a simple example of continuous predictive simulations

ifelse

```
two <- rep(2, 10)
five <- rep(5, 10)
choice <- rbinom(10, 1, 0.5)
ifelse(choice == 0, two, five)
```

```
##  [1] 5 2 5 5 2 5 2 2 5 5
```

```
## A simple example of continuous predictive simulations
woman <- rbinom (10, 1, .52)
height <- ifelse (woman==0, rnorm (10, 69.1, 2.9), rnorm (10, 64.5, 2.7))
avg.height <- mean (height)
print(avg.height)
```

```
## [1] 67.31653
```

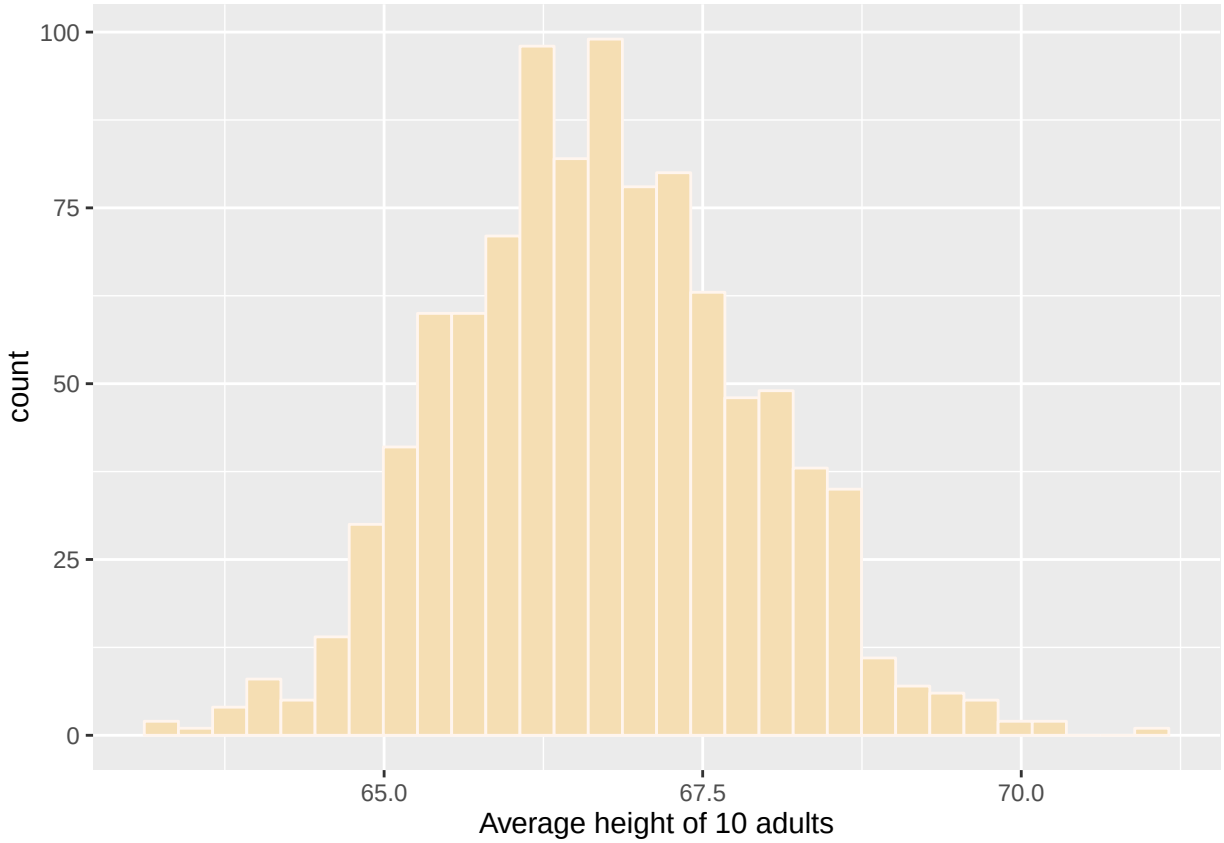
simulation

```
# simulation & Figure 7.1
n.sims <- 1000
avg.height <- rep (NA, n.sims)
for (s in 1:n.sims){
  sex <- rbinom (10, 1, .52)
  height <- ifelse (sex==0, rnorm (10, 69.1, 2.9), rnorm (10, 64.5, 2.7))
  avg.height[s] <- mean (height)
}
```

figure 7.1

```
frame2 = data.frame(x1=avg.height)
p1 <- ggplot(frame2,aes(x=x1))  +
      scale_x_continuous("Average height of 10 adults") +
      geom_histogram(colour = "seashell", fill = "wheat") +
      theme_gray()
print(p1)
```

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



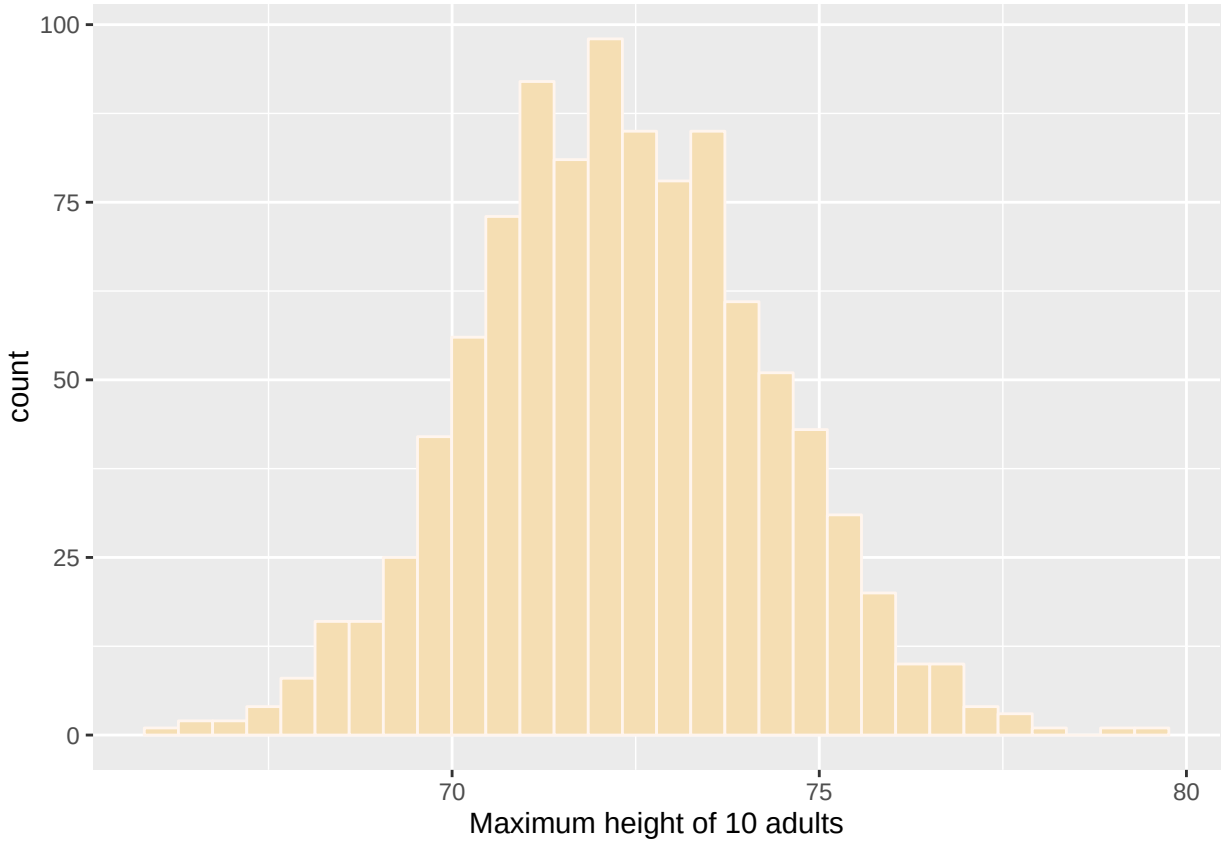
simulation for the maximum height

```
# simulation for the maximum height
n.sims <- 1000
max.height <- rep (NA, n.sims)
for (s in 1:n.sims){
  sex <- rbinom (10, 1, .52)
  height <- ifelse (sex==0, rnorm (10, 69.1, 2.9), rnorm (10, 64.5, 2.7))
  max.height[s] <- max (height)
}
```

figure

```
# dev.new()
frame3 = data.frame(x1=max.height)
p2 <- ggplot(frame3,aes(x=x1))  +
      scale_x_continuous("Maximum height of 10 adults") +
      geom_histogram(colour = "seashell", fill = "wheat") +
      theme_gray()
print(p2)
```

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



simulation using custom-made functions

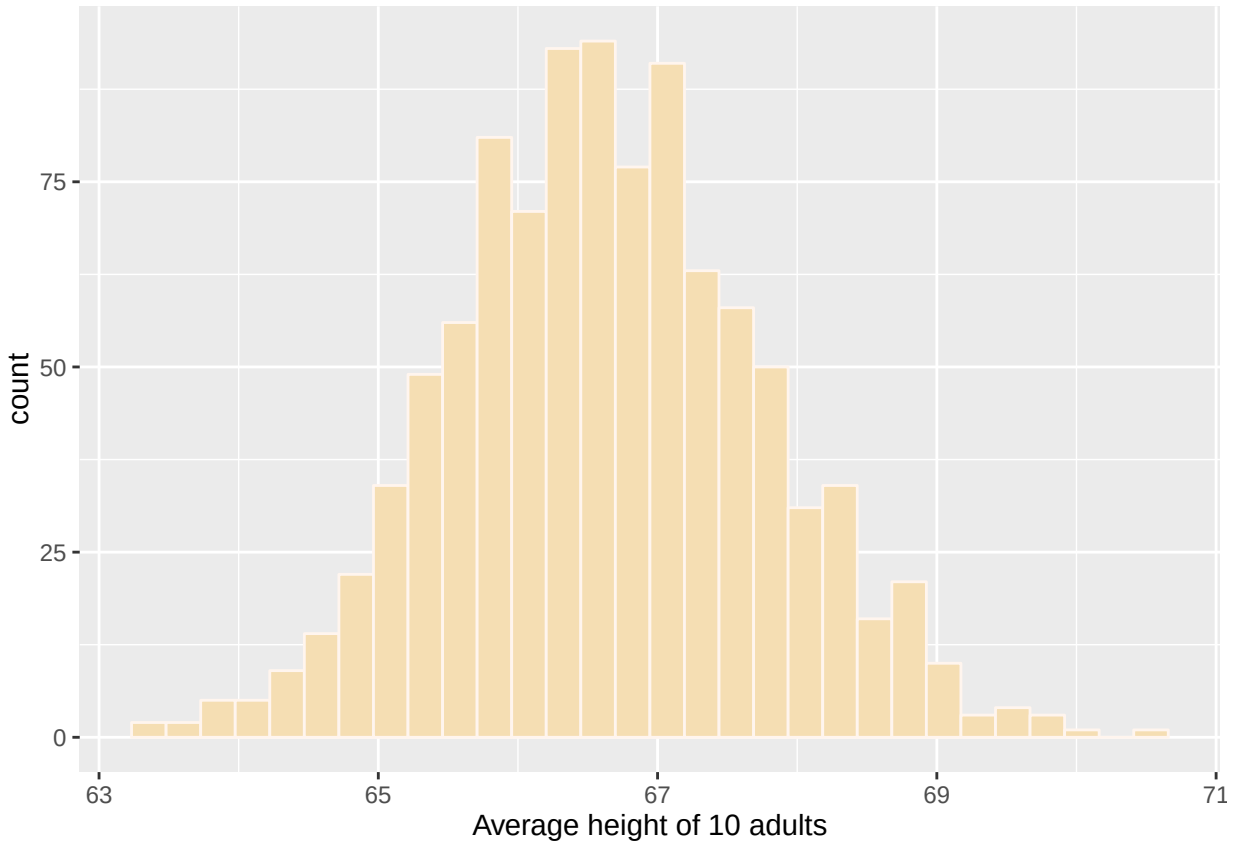
```
## Simulation using custom-made functions
Height.sim <- function (n.adults){
  sex <- rbinom (n.adults, 1, .52)
  height <- ifelse (sex==0, rnorm (10, 69.1, 2.9), rnorm (10, 64.5, 2.7))
  return (mean(height))
}
n.sims <- 1000
avg.height <- rep (NA, n.sims)
for (s in 1:n.sims){
  avg.height[s] <- Height.sim(n.adults=10)
}
```

```
avg.height <- replicate(1000, Height.sim(n.adults=10))
```

figure

```
frame4 = data.frame(x1=avg.height)
p3 <- ggplot(frame4,aes(x=x1))  +
      scale_x_continuous("Average height of 10 adults") +
      geom_histogram(colour = "seashell", fill = "wheat") +
      theme_gray()
print(p3)
```

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


count
75
50
25
0
63
65
67
69
71
Average height of 10 adults