Classnotes 8

Tobias Boggess

4/4/2022

Section 13.2 Exercises

Exercise 1: Do the following.

a) Use set.seed() to set the "seed" to a value (any positive integer will do). Then use sample() to generate n = 5 random numbers from 1, 2, . . . , 100. Report your R commands.
Code:

```
set.seed(5)

sample(x = 1:100, size = 5)
```

```
## [1] 66 57 79 75 41
```

b) Now set the "seed" again (to the same value), then use sample() again to produce n = 5 random numbers from 1, 2, . . . , 100. Confirm that this regenerates the same five values you got in part a.
Code:

```
set.seed(5)

sample(x = 1:100, size = 5)
```

```
## [1] 66 57 79 75 41
```

c) What would've happened in part b if you hadn't set the "seed" prior to the call to sample()? Try it.
Code:

```
set.seed(23)

sample(x = 1:100, size = 5)
```

```
## [1] 29 28 72 43 45
```

I would have received different numbers since the seed is different.

Exercise 2: Do the following with uniform random variables.

a) Use runif() to generate n = 1,000 random values between 0 and 1. Save them in a vector x. Report your R command(s).
Code:

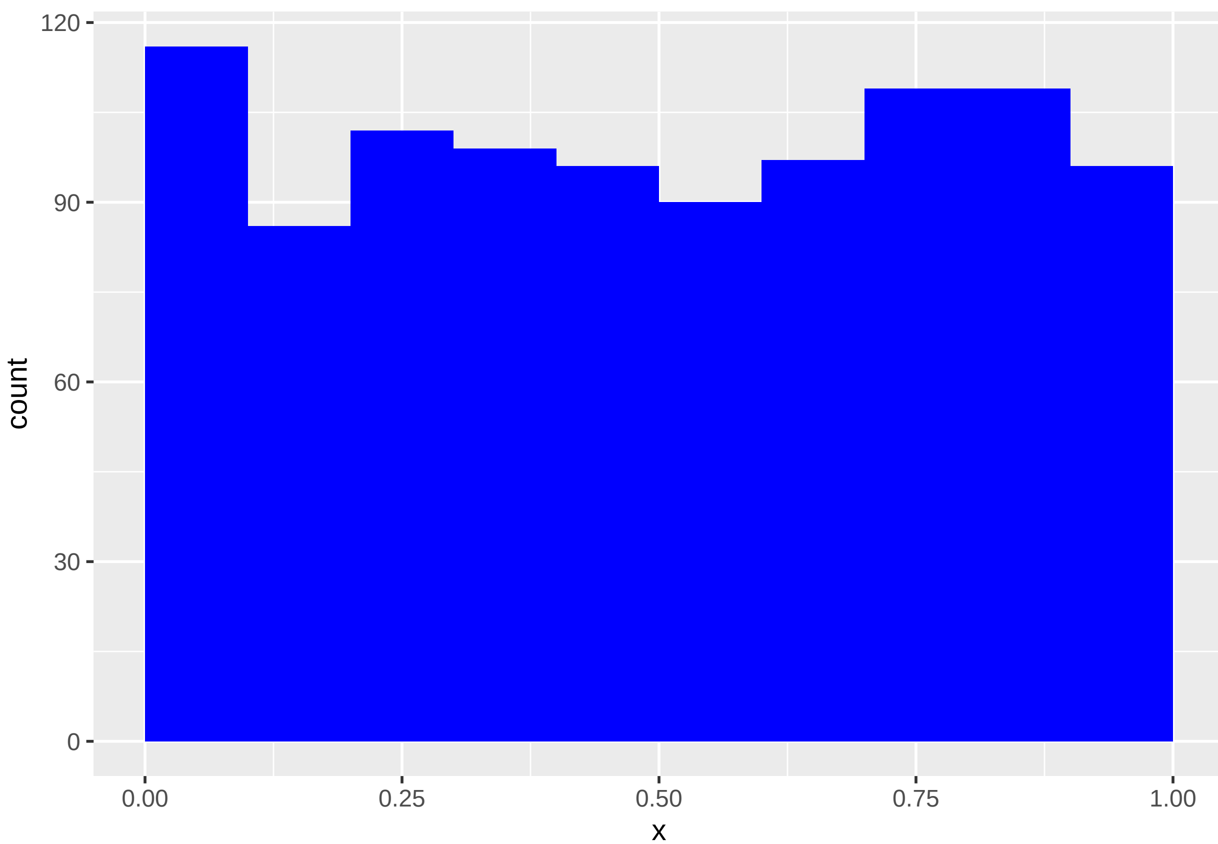
```
set.seed(500)

x <- runif(n = 1000, min = 0, max = 1)
head(x, n = 5)
```

```
## [1] 0.8336000 0.7250118 0.9753142 0.4676038 0.8122781
```

b) Produce a histogram of the simulated data:Are the simulated values fairly evenly spread over the interval from 0 to 1?
Code:

```
ggplot(data.frame(x = x), mapping = aes(x = x)) +
  geom_histogram(binwidth = 0.1,
                 boundary = 0.0,
                 fill = "blue")
```



Overall, the values are fairly evenly spread across the interval between 0 and 1 according to the histogram.

Exercise 3: Do the following with normal random numbers.

a) Use rnorm() to generate n = 1,000 random values from the normal(,) curve, with with = 0 and = 1. Save them in a vector x. Report your R command(s).
Code:

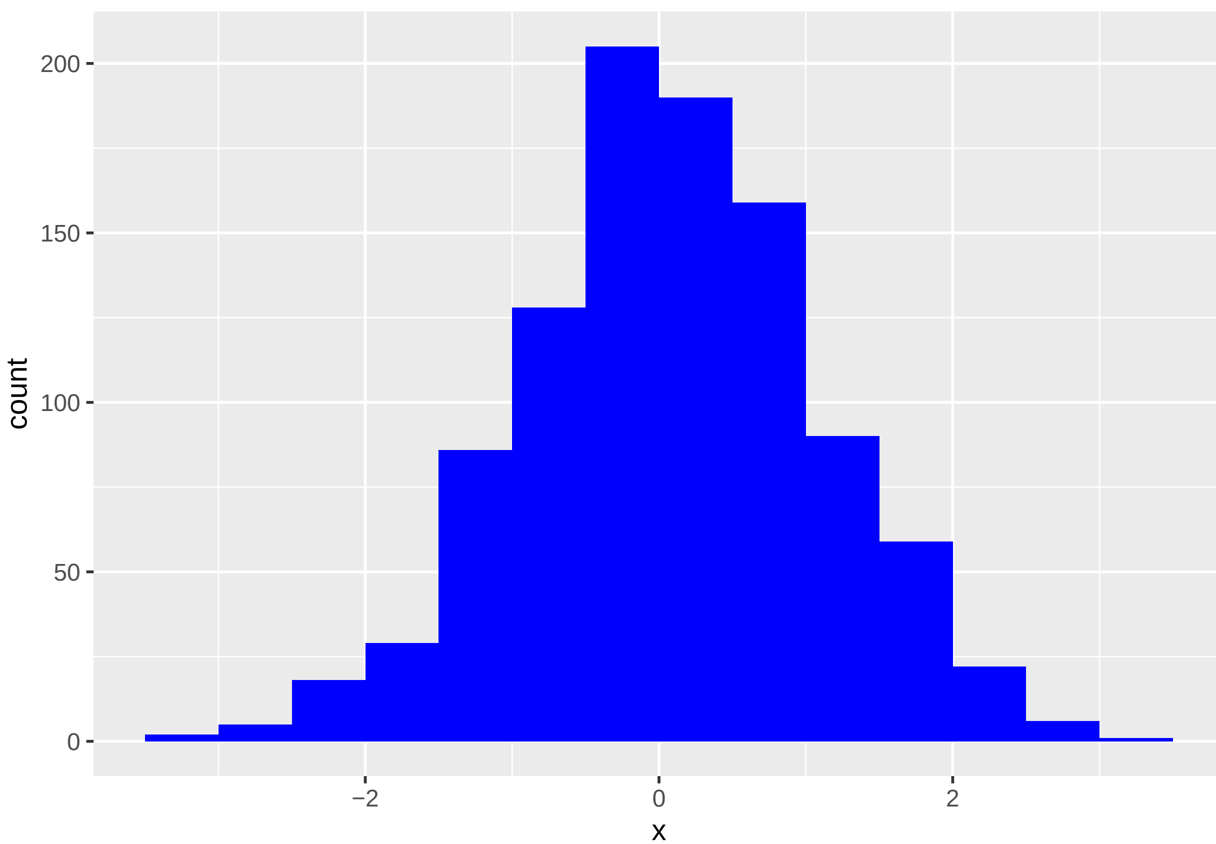
```
set.seed(21)

x <- rnorm(n = 1000, mean = 0, sd = 1)
head(x, n = 5)
```

```
## [1]  0.7930132  0.5222513  1.7462222 -1.2713361  2.1973895
```

b) Produce a histogram of the simulated data. Do the simulated values follow a (approximately) bell-shaped pattern from -3 to 3?
Code:

```
ggplot(data.frame(x = x), mapping = aes(x = x)) +
  geom_histogram(binwidth = 0.5,
                 boundary = -3.5,
                 fill = "blue")
```



The histogram does show a bell-shaped pattern from -3 to 3.

Exercise 4: Do the following with dichotomous random numbers.

a) Use rbinom() to generate a dichotomous (0 or 1) sequence of ten (n = 10) ”flips” of a biased ”coin” that lands ”heads” (i.e. results in a 1) with probability 0.7 (prob = 0.7) and ”tails” (0) otherwise. Report your R command(s).
Code:

```
set.seed(64)

rbinom(n = 10, size = 1, prob = 0.7)
```

```
##  [1] 1 0 1 0 1 0 0 1 1 1
```

b) Now use rbinom() to generate a sequence of ten (n = 10) dichotomous outcomes that are decreasingly likely be 1 according to the following probabilities. Report your R command(s).
Code:

```
# set.seed(56)
my.probs <- seq(from = 0.95, to = 0.05, by = -0.1)

rbinom(n = 10, size = 1, prob = my.probs)
```

```
##  [1] 1 1 0 1 1 0 1 1 1 1
```

Section 13.3 Exercises

Exercise 5: This exercise involves simulating data from a logistic regression model to investigate the performance of parameter estimates, the (estimated) intercept b0 and slope b1.

a) generate n = 1,000 dichotomous observations from a logistic regression model, with (true) parameter values intercept $0 = 4$ and slope $1 = -1$, at values of the explanatory variable generated from a uniform(0, 10) distribution. Report your R commands.
Code:

```
set.seed(57)

r <- runif(n = 1000, min = 0, max = 10)

true_probs <- exp(4 + (-1 * r)) / (1 + exp(4 + (-1 * r)))

y <- rbinom(n = 1000, size = 1, prob = true_probs)

sim.data <- data.frame(X = r, Y = y)
head(sim.data)
```

```
##           X Y
## 1 2.4391435 0
## 2 5.1294954 0
## 3 0.3862843 1
## 4 1.6617658 1
## 5 7.3320525 0
## 6 6.6280162 0
```

b) Fit a logistic regression model to your simulated data, and report the resulting parameter estimates b0 and b1 from the output of summary(). Are they close to the true parameter values $0 = 4$ and $1 = -1$?

Code:

```r
logreg <- glm(Y ~ X, data = sim.data, family = "binomial")
summary(logreg)
```

```
##
## Call:
## glm(formula = Y ~ X, family = "binomial", data = sim.data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.6909  -0.3739  -0.1049   0.3467   2.9652
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  4.17255    0.28570   14.61   <2e-16 ***
## X           -1.03260    0.06353  -16.25   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1339.27  on 999  degrees of freedom
## Residual deviance:  584.73  on 998  degrees of freedom
## AIC: 588.73
##
## Number of Fisher Scoring iterations: 6
```

The values from the summary of the logistic model are 4.17255 for β_0 and -1.03260 for β_1.