Data 556 - Homework 8

Will Wright

November 20, 2018

Problem 4b

Use simulations in R to numerically estimate the mean and the variance of the number of movie tickets that will be sold next year assuming that the mean number of movies released each year in the US is 700 and that, on average, 800,000 tickets were sold for each movie.

```
n <- 100000 # number of samples
movies <- rpois(n, 700)
tickets <-rpois(n, 800000)
ticketsSold <- movies*tickets
mean(ticketsSold)
```

```
## [1] 559954353
```

```
var(ticketsSold)
```

```
## [1] 4.464615e+14
```

For comparison, the theoretical mean is simply 700 movies * 800,000 tickets/movie = 560,000,000 total tickets, which is close to the simulated value. The theoretical variance is 700 * 800,000+700 * 800,000^2 + 700^2 * 800,000 = 4.4839 * 10^14, which is also close to the simulated value.

Problem 7

```
# function to return two values: the first is 1 if the roots are different, 0 if not, or NaN
# the second is 1 if both roots are positive, 0 if not, or NaN
quadraticator <- function(input) {
  a_input <- input[1]
  b_input <- input[2]
  c_input <- input[3]
  # assign a value to a
  ifelse(a_input < 0.4, a <- 1, a <- 2)

  # assign a value to b with an ugly, ugly set of ifelses
  ifelse(b_input < 0.25, b <- -3,
         ifelse(0.25 <= b_input & b_input < 0.5, b <- -2,
                ifelse(0.5 <= b_input & b_input < 0.75, b <- -1, b <- 1)))

  # assign a value to c
  ifelse(c_input < 0.5, c <- 1,
         ifelse(0.5 <= c_input & c_input < 0.9, c <- 2, c <- 3))

  # quadratic formula for both positive and negative
  x_1 <- (-b + sqrt(b^2-4*a*c))/(2*a)
  x_2 <- (-b - sqrt(b^2-4*a*c))/(2*a)

  # catch those NaNs
  if(is.nan(x_1) | is.nan(x_2)) {
    partA_result <- NaN
```

```
    partB_result <- NaN
  }

  # if not a NaN, then figure out
  if((is.nan(x_1) | is.nan(x_2))==FALSE){
    # binary for if roots are different
    ifelse(x_1 == x_2, partA_result <- 1, partA_result <- 0)

    # binary for if both roots are positive
    ifelse(x_1 > 0 & x_2 > 0, partB_result <- 1, partB_result <- 0)
  }

  return(c(partA_result, partB_result))
}


# dataframe of random uniforms to apply the quadraticator to
randoms <- data.frame(a_input = runif(10000), b_input <- runif(10000), c_input <- runif(10000))

# transpose and covert the results of applying the quadraticator so we have a nice 2-column data frame
results <- t(data.frame(suppressWarnings(apply(randoms, 1, quadraticator))))

partA <- length(results[which(results[,1]==1)])/dim(results)[1]
partB <- length(results[which(results[,2]==1)])/dim(results)[1]

partA
```

```
## [1] 0.0488
```

```
partB
```

```
## [1] 0.2126
```