Lab 2

Yidan Zhang

Urn problem revisited

mark 1 as blue, 0 as yellow

For every experiment, we draw 1 ball from urn1 and combine it in urn2

Essentially it's equivalent to repeat draw 1 ball from urn1 1000 times, the ball in urn2 can be denoted as c(rep(1,6),rep(0,6),x) where x is either 0 or 1

probability = number of blue balls (sum all the draws+6*number of the experiment(blue balls already in urn2 in each experiment))/number of all the balls in urn 2 (13number of the experiment)

Function version 1

```
sol_v1 <- function(nreps=100000, b1=10, y1=8, b2=6, y2=6){
  urn_draw <- as.vector(NA)
  urn1 <- c(rep(1,b1),rep(0,y1))
  urn2 <- c(rep(1,b2),rep(0,y2))
  for (i in 1:nreps){
  urn_draw[i] <- sample(urn1,1)
  }

  #print((sum(urn_draw)+6*nreps)/(13*nreps))
}
```

Function version 2

```r
sol_v2 <- function(nreps=100000, b1=10, y1=8, b2=6, y2=6){
  urn1 <- c(rep(1,b1),rep(0,y1))
  urn2 <- c(rep(1,b2),rep(0,y2))
  urn_draw <- rbinom(nreps,size=1,prob=mean(urn1))
  #print((sum(urn_draw)+b2*nreps)/((b2+y2+1)*nreps))
}
sol_v2()
```

Performance evaluate

```r
bench::mark(sol_v1(),sol_v2(), relative = TRUE, check = FALSE)
```

```
Warning: Some expressions had a GC in every iteration; so filtering is
disabled.
```

```
# A tibble: 2 x 6
  expression   min median `itr/sec` mem_alloc `gc/sec`
  <bch:expr> <dbl>  <dbl>     <dbl>     <dbl>    <dbl>
1 sol_v1()    120.   113.         1      40.6     14.3
2 sol_v2()      1      1        112.       1        1
```

I learned that building an empty vector and write a loop to store the result might not be the most efficient solution although it is straightforward to think, sometimes there are functions that can do things easier (more time and environment efficient) so it is important to think as a big picture.

Estimating the Probability of one treatment arm outperform control arm

```r
simulation <- function(prob){
  result <- do.call(cbind,lapply(1:4, function(i) rbinom(10, 1, prob = prob[i])))
  colnames(result) <- c(paste0("t",0:3))
  nt=rep(nrow(result),4)
  yt=colSums(result)
  posterior <- sapply(1:4, function(i) rbeta(n=1000, shape1=0.35 + yt[i], shape2=0.65 + nt[i]
  colnames(posterior) <- c(paste0("t",0:3))
```

```
  apply(posterior[, 2:4], 2, function(x) mean(x > posterior[, 1]))
}
simulation(prob=rep(0.35,4))
```

```
   t1    t2    t3
0.875 0.946 0.940
```