BDA Homework

Juan Lopez Martin

1.9

a.

```
patient_population <- 100
arrival_times <- round(cumsum(rexp(patient_population, rate = 1/10)*60))
consultA_duration <- round((runif(patient_population, 5, 20)*60))
consultB_duration <- round((runif(patient_population, 5, 20)*60))
consultC_duration <- round((runif(patient_population, 5, 20)*60))

seconds <- 7*60*60*2
patients_waiting <- 0
patients_in_consultA <- 0
patients_in_consultB <- 0
patients_in_consultC <- 0
patients_left <- 0

waitime <- matrix(NA, nrow = seconds, ncol = patient_population)

for(s in 1:seconds){
  if(s<=7*60*60){
    patients_waiting <- sum(arrival_times<s) - patients_in_consultA -
      patients_in_consultB - patients_in_consultC - patients_left
  }
  if(patients_in_consultA==1){
      if((consultA_timestamp + consultA_duration[patients_left + 1]) <= s){
        #print('Patient leaves consult')
        patients_in_consultA <- 0
        t <- patients_left + 1
      }
  } else {
    if(patients_waiting>=1){
      #print('Patient enters consult')
      patients_waiting <- patients_waiting - 1
      patients_in_consultA <- 1
      consultA_timestamp <- s
    }
  }
  if(patients_in_consultB==1){
      if((consultB_timestamp + consultB_duration[patients_left + 1]) <= s){
        # Patient leaves consult
        patients_in_consultB <- 0
```

```
        patients_left <- patients_left + 1
      }
  } else {
    if(patients_waiting>=1){
      # Patient enters consult
      patients_waiting <- patients_waiting - 1
      patients_in_consultB <- 1
      consultB_timestamp <- s
    }
  }

  if(patients_in_consultC==1){
      if((consultC_timestamp + consultC_duration[patients_left + 1]) <= s){
        # Patient leaves consult
        patients_in_consultC <- 0
        patients_left <- patients_left + 1
      }
  } else {
    if(patients_waiting>=1){
      # Patient enters consult
      patients_waiting <- patients_waiting - 1
      patients_in_consultC <- 1
      consultC_timestamp <- s
    }
  }
  if(patients_waiting>0){
    waitime[s, (patients_left+1):(patients_left+patients_waiting+1)] <- 1
  }

  if((s>=7*60*60) & (patients_waiting==0) & (patients_in_consultA==0) &
     (patients_in_consultB==0) & (patients_in_consultC==0)){
    clinic_closes <- s
    break
  }
}
```

```
print(paste("Clinic closed after", round((clinic_closes/60)/60, 2), "hours"))
```

```
## [1] "Clinic closed after 7.13 hours"
```

```
print(paste("Attended", patients_left, "patients"))
```

```
## [1] "Attended 43 patients"
```

```
print(paste(sum(colSums(waitime, na.rm = TRUE)[1:patients_left]>1), "patients waited"))
```

```
## [1] "27 patients waited"
```

```
print(paste("The mean wait time was",
            round(mean(colSums(waitime, na.rm = TRUE)[1:patients_left])/60, 2), "minutes"))
```

```
## [1] "The mean wait time was 4.97 minutes"
```

b.

```
# put simulation as a function
simulate_clinic <- function(patient_population = 100, seconds = 7*60*60*2){

  arrival_times <- round(cumsum(rexp(patient_population, rate = 1/10)*60))
  consultA_duration <- round((runif(patient_population, 5, 20)*60))
  consultB_duration <- round((runif(patient_population, 5, 20)*60))
  consultC_duration <- round((runif(patient_population, 5, 20)*60))

  patients_waiting <- 0
  patients_in_consultA <- 0
  patients_in_consultB <- 0
  patients_in_consultC <- 0
  patients_left <- 0

  waitime <- matrix(NA, nrow = seconds, ncol = patient_population)

  for(s in 1:seconds){
    if(s<=7*60*60){
      patients_waiting <- sum(arrival_times<s) - patients_in_consultA - patients_in_consultB - patients
    }
    if(patients_in_consultA==1){
        if((consultA_timestamp + consultA_duration[patients_left + 1]) <= s){
          #print('Patient leaves consult')
          patients_in_consultA <- 0
          t <- patients_left + 1
        }
    } else {
      if(patients_waiting>=1){
        #print('Patient enters consult')
        patients_waiting <- patients_waiting - 1
        patients_in_consultA <- 1
        consultA_timestamp <- s
      }
    }
    if(patients_in_consultB==1){
        if((consultB_timestamp + consultB_duration[patients_left + 1]) <= s){
          # Patient leaves consult
          patients_in_consultB <- 0
          patients_left <- patients_left + 1
        }
    } else {
      if(patients_waiting>=1){
        # Patient enters consult
        patients_waiting <- patients_waiting - 1
        patients_in_consultB <- 1
        consultB_timestamp <- s
      }
    }

    if(patients_in_consultC==1){
```

```
        if((consultC_timestamp + consultC_duration[patients_left + 1]) <= s){
          # Patient leaves consult
          patients_in_consultC <- 0
          patients_left <- patients_left + 1
        }
    } else {
      if(patients_waiting>=1){
        # Patient enters consult
        patients_waiting <- patients_waiting - 1
        patients_in_consultC <- 1
        consultC_timestamp <- s
      }
    }
    if(patients_waiting>0){
      waitime[s, (patients_left+1):(patients_left+patients_waiting+1)] <- 1
    }

    if((s>=7*60*60) & (patients_waiting==0) & (patients_in_consultA==0) & (patients_in_consultB==0) & (
      clinic_closes <- s
      break
    }
  }

  return(c((clinic_closes/60)/60,
         patients_left,
         sum(colSums(waitime, na.rm = TRUE)[1:patients_left]>1),
         mean(colSums(waitime, na.rm = TRUE)[1:patients_left])/60))
}

# repeat the process n_sims times and store the results in res
n_sims <- 100
res <- matrix(NA, nrow = n_sims, ncol = 4)
for(i in 1:n_sims){
  res[i,] <- simulate_clinic()
}

print(paste("50% interval for clinic open (in hours): ",
            paste(round(quantile(res[,1], probs = c(.25, .75)), 2), collapse = " - ")))
```

```
## [1] "50% interval for clinic open (in hours):  7.11 - 7.23"
```

```
print(paste("50% interval for number of patients attended: ",
            paste(round(quantile(res[,2], probs = c(.25, .75)), 2), collapse = " - ")))
```

```
## [1] "50% interval for number of patients attended:  35 - 45"
```

```
print(paste("50% interval for number of patients that had to wait: ",
            paste(round(quantile(res[,3], probs = c(.25, .75)), 2), collapse = " - ")))
```

```
## [1] "50% interval for number of patients that had to wait:  16 - 29"
```

```r
print(paste("50% interval for mean waitime: ",
            paste(round(quantile(res[,4], probs = c(.25, .75)), 2), collapse = " - ")))
```

```
## [1] "50% interval for mean waitime:  2.82 - 6.61"
```