R course - Exercice 4

First Name - Last Name

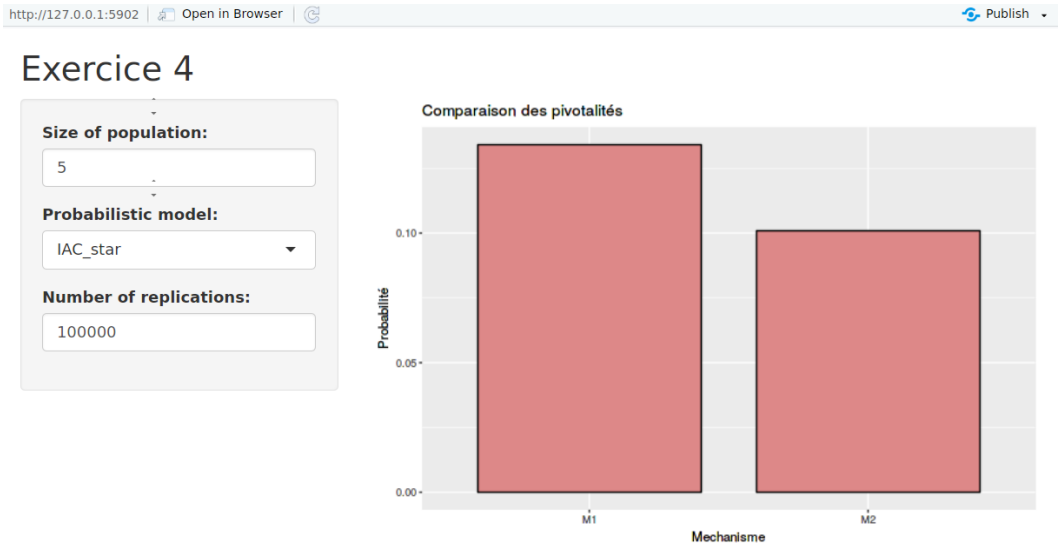
Master 2 Statistics and Econometrics

Presentation of the problem

The problem is the same as that of the exercice 2, except that here we will be interested in a way of restoring the results via **shiny**.

Objective

The goal is to create a **shiny** application that will render below.



The graph represents the result of the function *simul_elec()* whose code is recalled below.

The left menu allows you to define:

- the value of the parameter **n**,
- the value of the parameter **cas** ("IC" or "IAC_star"),
- the value of the parameter **B**.

The graph will be updated when the user changes one of the values above.

Indications

To create this application, you can start from the example given in the course and adapt it to this problem. There are only a few changes to make to achieve the result above ...

Remark : to enter a numeric value, you can use the function *numericInput()*. To create a drop-down menu with "IC" and "IAC_star", you can use the function *selectInput()*.

Notation

You will have to return the exercice in *.pdf* or *.html* format, which would have been done with **R** Markdown if possible. It should contain the lines of code used to answer the questions, but you should also explain what you are doing.

simul_elec() and *processus()*

```
simul_elec <- function(n, cas, B = 1000) {
  # vérification
  stopifnot(n%%2 == 1, cas %in% c("IC", "IAC_star"))
  # initialisation: nombre total d electeurs
  taille_popu <- 3*n
  # réplication de la fonction processus()
  res_simul <- replicate(B, processus(n = n, cas = cas, taille_popu = taille_popu))
  # on retourne les résultats
  return(rowMeans(res_simul)/taille_popu)
}
```

```
processus <- function(n, cas, taille_popu = 3*n) {
  # simulation du choix des électeurs
  vec_etat <- switch(cas, IC = rbinom(3, n, 0.5),
                     IAC_star = rbinom(3, n, runif(3)))
  # vainqueur de l'élection :
  # dans M1, on calcule combien d'électeurs ont vote D
  vote_popu <- sum(vec_etat)
  # on compare ce nombre avec la valeur seuil
  val_seuil_pop <- (taille_popu + 1)/2
  # on regarde si D a gagne oui ou non l'election
  winn_D_M1 <- (vote_popu >= val_seuil_pop)
  # dans M2, on regarde combien d'états ont ete gagne par D
  nb_etat_gagnant <- sum(vec_etat >= (n + 1)/2)
  # sachant qu'il en faut au moins 2 pour remporter l'élection
  # on regarde si D a gagne oui ou non l election
  winn_D_M2 <- (nb_etat_gagnant > 1)
  # proba d'être pivot :
  # dans le cas M1 :
  if ((winn_D_M1 & (vote_popu == val_seuil_pop)) | # soit D gagne avec 1 voix d'avance
   (!winn_D_M1 & (vote_popu == (taille_popu - 1)/2))) { # soit R gagne avec une voix d'avance
    pivot_M1 <- val_seuil_pop # dans ce cas le nb de pivot est tjs le même
  } else {
    pivot_M1 <- 0
  }
  # dans le cas M2,
  if (winn_D_M2 & nb_etat_gagnant == 2) { #  D gagne avec 2 etats
    pivot_M2 <- sum((n + 1)/2 * (vec_etat == (n + 1)/2)) # (n+1)/2*(nb etats avec 1 voix d'écart)
    } else {
      if (!winn_D_M2 & nb_etat_gagnant == 1) { # R gagne avec 2 état
        pivot_M2 <- sum((n + 1)/2 * (vec_etat == (n - 1)/2)) # (n+1)/2*(nb etats avec 1 voix d'écart)
   } else {
   pivot_M2 <- 0
   }
  }
```

```
  return(c(pivot_M1 = pivot_M1, pivot_M2 = pivot_M2))
}
```