PASSONS LE PONT AVEC TOM LE ROBOT D'ANTILLES-GUYANNE
PROBABILITÉS ET ALGORITHMIQUE
Hubert Raymondaud LEGTA Louis Giraud à Carpentras-Serres

- **L'exercice 4 du sujet 2013 de septembre d'Antilles-Guyane** propose un algorithme de simulation d'une marche aléatoire. Alors que l'énoncé précise : "L'objectif de cet exercice est d'estimer la probabilité p de l'événement S [le robot] Tom traverse le pont", on constate que ni l'algorithme de la partie A, ni le calcul de la partie B, basé sur des suites numériques, ne proposent une estimation de cette probabilité. La partie A propose la simulation d'une marche aléatoire, et la partie B permet de trouver une valeur approchée de cette probabilité.
- Dans ce document, je propose la traduction en langage **R** de l'algorithme de l'énoncé, et d'un algorithme répondant à la question A.2. Je propose ensuite un prolongement permettant une représentation graphique du trajet du robot, un prolongement permettant de calculer une estimation de la probabilité de S.
  Je présente enfin un algorithme simulant les résultats des N élèves d'une classe, et illustrant la loi des grands nombres par une juxtaposition de diagrammes en boites et un nuage de points.
- **Tableau 1 : Questions 4.A.1 et 4.A.2**

Le robot Tom doit emprunter un pont sans garde-corps de 10 pas de long et de 2 pas de large. Sa démarche est très particulière :

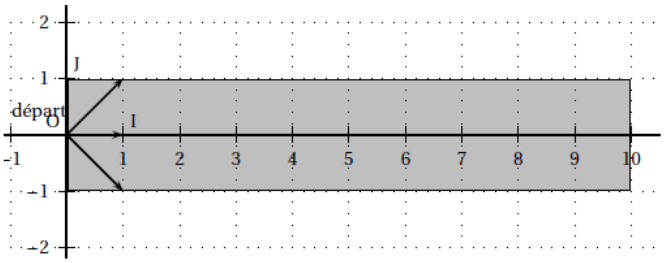
- Soit il avance d'un pas tout droit ;
- Soit il se déplace en diagonale vers la gauche (déplacement équivalent à un pas vers la gauche et un pas tout droit) ;
- Soit il se déplace en diagonale vers la droite (déplacement équivalent à un pas vers la droite et un pas tout droit).

On suppose que ces trois types de déplacement sont aléatoires et équiprobables.

L'objectif de cet exercice est d'estimer la probabilité p de l'évènement S « Tom traverse le pont » c'est-à-dire « Tom n'est pas tombé dans l'eau et se trouve encore sur le pont au bout de 10 déplacements ».

Partie A : modélisation et simulation

On schématise le pont par un rectangle dans le plan muni d'un repère orthonormé (O, I, J) comme l'indique la figure ci-dessous. On suppose que Tom se trouve au point de coordonnées (0 ; 0) au début de la traversée. On note $(x\,;\,y)$ les coordonnées de la position de Tom après x déplacements.



On a écrit l'algorithme suivant qui simule la position de Tom au bout de x déplacements :

```
x, y, n sont des entiers
Affecter à x la valeur 0
Affecter à y la valeur 0
Tant que y ⩾ −1 et y ⩽ 1 et x ⩽ 9
        Affecter à n une valeur choisie au hasard entre −1, 0 et 1
        Affecter à y la valeur y + n
        Affecter à x la valeur x + 1
Fin tant que
Afficher « la position de Tom est » (x ; y)
```

1. On donne les couples suivants : (−1 ; 1) ; (10 ; 0) ; (2 ; 4) ; (10 ; 2).
   Lesquels ont pu être obtenus avec cet algorithme ? Justifier la réponse.
2. Modifier cet algorithme pour qu'à la place de « la position de Tom est $(x\,;\,y)$ », il affiche finalement « Tom a réussi la traversée » ou « Tom est tombé ».

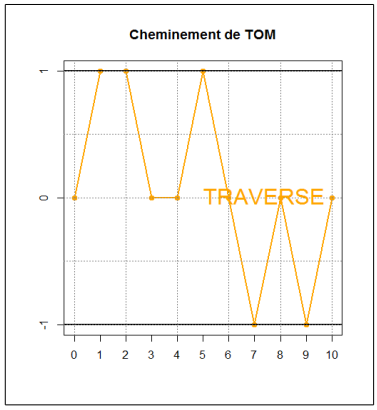
```
# TABLEAU 1 : ANTILLES GUYANE Septembre 2013
# Exercice 4.A.1. L'algorithme de l'énoncé.
robtom1 <- function(){
   x <- 0 ; y <- 0 ; urne <- c(-1, 0, 1)
   while (y >= -1 & y <= 1 & x <= 9) {
      n <- sample(urne, 1)
      y <- y + n ; x <- x + 1
   }
# Affichage des résultats
   cat("La dernière position de TOM est : A(",
      x, ";", y, ")\n\n")
}

> robtom1()
La dernière position de TOM est : A(5 ; 2)

Aucune difficulté pour la traduction R, la
condition d'arrêt(y >= -2 & y <= 1 & x <= 9)
étant donnée dans l'énoncé. & est le connecteur
logique ET.
Cat(...) est la fonction d'affichage.

#-----------------------------------------
# Exercice 4.A.2. Qualification de
# l'arrivée, traversé ou tombé
robtom2 <- function(){
   x <- 0 ; y <- 0 ; urne <- c(-1, 0, 1)
   while (y >= -1 & y <= 1 & x <= 9) {
      n <- sample(urne, 1)
      y <- y + n ; x <- x + 1
   }
# Affichage des résultats
   if (y == -2 | y == 2) {
      cat("TOM est tombé dans l'eau\n")
      cat("Sa dernière position est : A(",
         x, ",", y, ")\n\n")
   } else {
      cat("TOM a réussi la traversée\n")
      cat("Sa dernière position sur le pont : A(",
         x, ";", y, ")\n\n")
   }
}

> robtom2()
TOM est tombé dans l'eau
La dernière position de TOM est : A(3,2)

La principale difficulté se situe dans le test
(y == -2 | y == 2), qui peut aussi se faire par
(x == 10 & y == 1) | (x == 10 & y == - 1) |
(x == 10 & y == 0)
| est le connecteur logique OU.
```

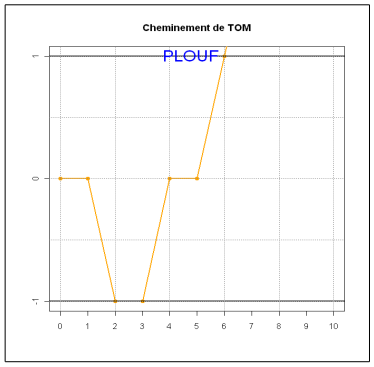
● Tableau 2 : Prolongement graphique.

Il s'agit de représenter graphiquement la marche aléatoire simulée dans la première partie de l'algorithme, en faisant afficher sur le graphique le résultat de la marche, traversée du pont ou tombé à l'eau.

Les suites des coordonnées des points du cheminement sont enregistrées au fur et à mesure par **`suiteX <- c(suiteX, x) ; suiteY <- c(suiteY, y) .`**
Le cadre du graphique est tracé par
`plot(c(0, 10), c(1, -1), type = "n", main = "Cheminement de TOM", xlab = "", ylab = "", xaxp = c(0, 10, 10), yaxp = c(-1, 1, 2))` dans lequel viendra se tracer la ligne brisée du cheminement.
Le tracé de la ligne de cheminement est ensuite effectuée par **`lines(suiteX, suiteY, type = "o", pch = 19, col = "orange", lwd = 2) .`**
L'affichage du résultat à la fin du cheminement se fait par
`text(suiteX[x], suiteY[x], pos = 2, labels = "PLOUF", col = "blue", cex = 2)` à l'intérieur d'un test.

```
# TABLEAU 2 : REPRÉSENTATION GRAPHIQUE DE
# LA MARCHE DE TOM
robtom3 <- function(){
   x <- 0 ; y <- 0 ; urne <- c(-1, 0, 1)
   suiteX <- 0 ; suiteY <- 0
   while (y >= -1 & y <= 1 & x <= 9) {
      n <- sample(urne, 1)
      y <- y + n ; x <- x + 1
      suiteX <- c(suiteX, x) ; suiteY <- c(suiteY, y)
   }
# Représentation graphique du cheminement
   plot(c(0, 10), c(1, -1), type = "n",
         main = "Cheminement de TOM",
         xlab = "", ylab = "",
         xaxp = c(0, 10, 10), yaxp = c(-1, 1, 2))
      lines(suiteX, suiteY, type = "o", pch = 19,
         col = "orange", lwd = 2)
      abline(h = c(-1, 1), lwd = 2)
      grid(col = "grey50")
      if (y == 2 | y == -2) {
         text(suiteX[x], suiteY[x], pos = 2,
            labels = "PLOUF", col = "blue", cex = 2)
      } else {
         text(suiteX[11], suiteY[11], pos = 2,
            labels = "TRAVERSE", col = "orange",
            cex = 2)
      }
}
```

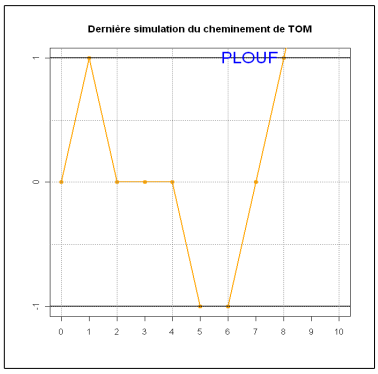


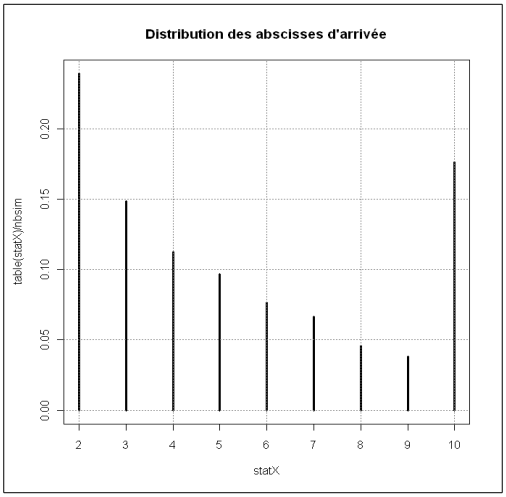


● Tableau 3 : Prolongement pour le calcul d'une estimation de p par simulation.

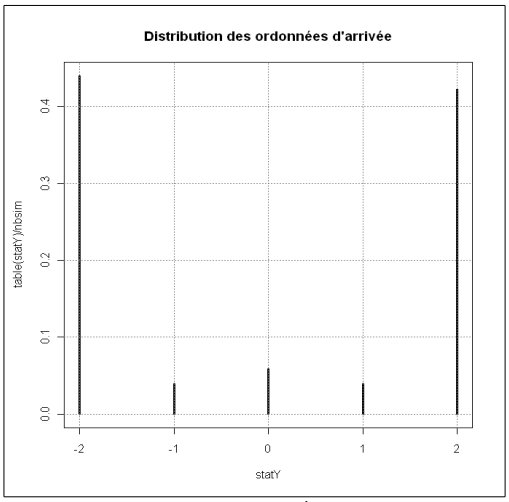
Le tableau ci-dessous présente un algorithme de simulation permettant d'estimer la probabilité de l'événement S. De plus, cet algorithme calcule et représente graphiquement les distributions simulées des abscisses et des ordonnées des positions de la fin de la marche du robot, marche qui se termine soit sur le pont soit dans l'eau (PLOUF).
Le test de traversée du pont est fait après la boucle **`while(...)`** par **`Pont <- (y != -2 & y != 2)`** (**`!=`** code ≠) qui renvoie TRUE ou FALSE qui est transformé en 1 ou 0 dans
`statPont <- statPont + Pont` qui cumule les "succès" c'est à dire les marches qui traversent le pont.
J'ai inclus la représentation graphique de la dernière marche de la simulation, avec son résultat.

```
# TABLEAU 3 : DE LA SIMULATION D'UNE MARCHE ALÉATOIRE AU
# CALCUL D'UNE ESTIMATION DE LA PROBABILITÉ DE TRAVERSER
# LE PONT EN SIMULANT nbsim MARCHES ALÉATOIRES
robtom4 <- function(nbsim = 2000){
   urne <- c(-1, 0, 1)
   statPont <- 0 ; statY <- NULL ; statX <- NULL
   for (k in 1:nbsim) {
      x <- 0 ; y <- 0 ; suiteX <- 0 ; suiteY <- 0
      while (y >= -1 & y <= 1 & x <= 9) {
         n <- sample(urne, 1)
         y <- y + n ; x <- x + 1
         suiteX <- c(suiteX, x) ; suiteY <- c(suiteY, y)
      }
      Pont <- (y != -2 & y != 2)
      statPont <- statPont + Pont
      statX <- c(statX, x) ; statY <- c(statY, y)
   }
   PCPont <- statPont / nbsim
# Affichage des résultats et des graphiques
   cat("Une estimation de la probabilité de passer le pont :",
      PCPont, "\n\n")
# Représentation graphique des distributions des séries
# simulées des abscisses et des ordonnées des positions
# de la fin de la marche du robot sur le pont.
   plot(table(statX) / nbsim,
         main = "Distribution des abscisses d'arrivée")
      grid(col = "grey50")
   plot(table(statY) / nbsim,
         main = "Distribution des ordonnées d'arrivée")
      grid(col = "grey50")
# Représentation graphique du dernier cheminement
   plot(c(0, 10), c(1, -1), type = "n",
         main = "Dernier cheminement de TOM",
         xlab = "", ylab = "",
         xaxp = c(0, 10, 10), yaxp = c(-1, 1, 2))
      lines(suiteX, suiteY, type = "o", pch = 19,
         col = "orange")
      abline(h = c(-1, 1), lwd = 2)
      grid(col = "grey50")
      if (y == 2 | y == -2) {
         text(suiteX[x], suiteY[x], pos = 2,
            labels = "PLOUF", col = "blue")
      } else {
         text(suiteX[11], suiteY[11], pos = 2,
            labels = "TRAVERSE", col = "orange")
      }
}

> robtom4()
Une estimation de la probabilité de passer le pont : 0.135
```

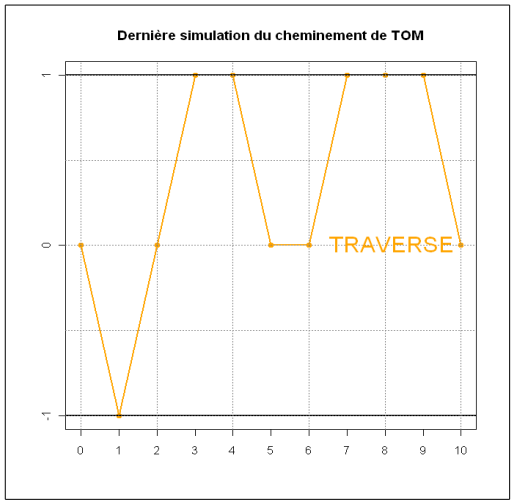




La fréquence du 10 est à commenter.



Comment retrouver la fréquence de succès ?



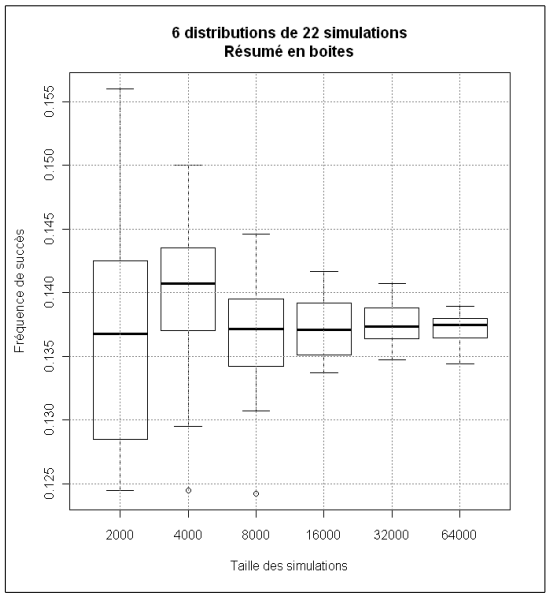
- **Tableau 4 : Prolongement simulant les résultats d'une classe de N élèves et faisant l'illustration graphique de la loi des grands nombres.**

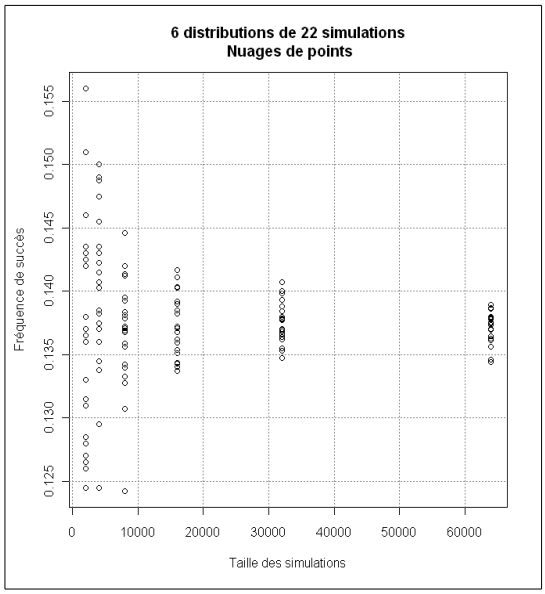
Dans le tableau ci dessous, je propose un algorithme permettant d'obtenir les séries de simulations effectuées dans une classe de **N (22)** élèves, chacun effectuant une série de simulations de différentes **Tailles**, 2000, 4000, 8000, 16000, 32000, 64000 …

`rep(Tailles, each = N)` répète N fois chaque taille d'échantillon. Les deux boucles for imbriquées créent les N séries de simulations pour chacune des tailles prises dans Tailles. Les 22(N) fois 6 (nombre de tailles) résultats (fréquences de succès) sont stockés dans le vecteur **`vectSucces`**.

```
# TABLEAU 4 : SIMULATION DES RÉSULTAT D'UNE CLASSE
# ET ILLUSTRATION DE LA LOI DES GRANDS NOMBRES
# La fonction effectuant nbsim simulations de la marche
RobTomSim <- function(nbsim){
   urne <- c(-1, 0, 1)
   statPont <- 0
   for (k in 1:nbsim) {
      x <- 0 ; y <- 0
      while (y >= -1 & y <= 1 & x <= 9) {
         n <- sample(urne, 1)
         y <- y + n ; x <- x + 1
      }
      Pont <- (y != -2 & y != 2)
      statPont <- statPont + Pont
   }
   PCPont <- statPont / nbsim
   return(PCPont)
}

# SIMULATION DES RÉSULTATS D'UNE CLASSE DE N ÉLÈVES
#
# LES N TAILLES D'ÉCHANTILLONS SONT SAISIES EN LIGNE DE
# COMMANDE (c'est plus pratique que de les saisir lors de
# l'utilisation de la fonction)
N <- 22
Tailles <- c(2000, 4000, 8000, 16000, 32000, 64000)

# LA FONCTION GÉNÉRANT LES DISTRIBUTIONS OBTENUES
# DANS UNE CLASSE DE N ÉLÈVES. ILLUSTRATION GRAPHIQUES
# EN BOITES À PATTES ET NUAGES DE POINTS
SimSucces <- function(Tailles, N) {
   vecTailles <- rep(Tailles, each = N)
   vectSucces <- NULL
   for (k in 1:length(Tailles)) {
      vectSuccesT <- NULL
      for (i in 1:N) {
         vectSuccesT <- c(vectSuccesT, RobTomSim(Tailles[k]))
      }
      vectSucces <- c(vectSucces, vectSuccesT)
   }
# REPRÉSENTATION GRAPHIQUE DES DISTRIBUTIONS SIMULÉES
   plot(as.factor(vecTailles), vectSucces,
         main = paste(length(Tailles), "distributions de", N,
            "simulations\nRésumé en boites"),
         xlab = "Taille des simulations",
         ylab = "Fréquence de succès")
      grid(col = "grey50")
   plot(vecTailles, vectSucces,
         main = paste(length(Tailles), "distributions de", N,
            "simulations\n Nuages de points"),
         xlab = "Taille des simulations",
         ylab = "Fréquence de succès")
      grid(col = "grey50")
}
```

ATTENTION À L'ÉCHELLE DES "ABSCISSES"





● EN GUISE DE CONCLUSION

Dans la pratique je commence toujours mon introduction aux probabilité en BTS par une expérience concrète de simulation d'un contrôle de qualité par attribut, avec des bouteilles représentant des lots de semences dans lesquelles on effectue un échantillonnage selon un plan de contrôle inspiré des normes AFNOR.

Dans ce plan figurent des grilles de saisie manuelle des résultats, étape qui me semble indispensable à la bonne compréhension des structures de données que l'on utilise par la suite. J'en ai fait une rapide présentation au séminaire de juin 2012 de l'IREM de Lyon (page 20 du document joint dans http://math.univ-lyon1.fr/irem/spip.php?article506). Cet exemple a été repris (avec des biberons à la place des bouteilles) dans un document de formation de l'IREM de Clermont-Ferrand (http://www.irem.univ-bpclermont.fr/spip.php?article366).

Je ne fais intervenir la simulation numérique qu'après avoir fait manipuler les simulations et leurs données manuellement. Cela permet de donner du sens aux différentes étapes nécessaires à la simulation et aux types de données que l'on obtient. Les élèves comprennent ensuite plus facilement les algorithmes de simulation qu'on leur présente.