# RICM4 Probabilités et Simulation – Devoir n°1

Arnaud Legrand, Florence Perronnin

19 octobre 2018

## Informations générales

### Conseils et consignes importantes

- Avant de vous lancer dans la programmation, essayez de répondre aux différentes questions, l'une après l'autre (c'est important), en n'utilisant que votre intuition. Vous rédigerez votre opinion en début de document avant de détailler vos réponses finales. Le fait que cette intuition soit correcte ou pas n'aura aucune impact sur votre note finale. Décrire votre intuition a uniquement pour but que vous commenciez à réfléchir au problème et de réaliser à quel point votre intuition peut être correcte ou pas.

  Vous analyserez, en fin de DM, cette intuition au vu des résultats et des statistiques observés.

- Vous prendrez soin de vous interroger sur la variabilité de vos estimations et de justifier pourquoi/comment vous avez adapté les différents paramètres de vos simulations.

- Nous vous déconseillons fortement de chercher des solutions qui ressembleraient sur Internet ou dans des livres. Ce qu'on trouve sur Internet sur ces sujets est souvent bourré de fautes et les livres qui traitent formellement de ces sujets sont souvent bien au delà de notre programme. N'hésitez cependant pas à nous demander si vous avez un doute sur la signification de l'énoncé ou sur une question.

- Vous pouvez discuter du sujet entre vous mais le codage et la rédaction des analyses doivent être individuels. Vous citerez toute source d'information utilisée pour répondre aux questions.

- Pas la peine de chercher des choses compliquées. Pour la majorité des questions des différents sujets, le code à écrire n'excède pas quelques lignes.

- Votre devoir sera rédigé sous forme d'un document HTML généré à l'aide de `R/Markdown` et publié sur `rpubs` en prenant soin de bien laisser le code apparent et de fixer la graine de votre générateur à l'aide de la fonction `set.seed` au tout début du document afin qu'il soit possible de reproduire vos données avec exactitude. Vous enverrez l'url `rpubs` de votre devoir par mail à `arnaud.legrand@imag.fr` et à `florence.perronnin@imag.fr` en indiquant dans le sujet `[RICM4-PS] DM` avant le 6 décembre à minuit.

## Sujet 1 : Au casino

Vous avez face à vous deux machines à sous $M_A$ et $M_B$ rapportant 0 ou 1€ à chaque fois que vous les utilisez. Ces machines sont différentes et chaque machine a respectivement une probabilité de gain $p_A$ et $p_B$ inconnue a priori. On suppose donc que les gains $X_{M,1}, X_{M,2}, \dots$ de chaque machine $M \in \{A, B\}$ à chaque tirage se modélisent par des variables aléatoires indépendantes et identiquement distribuées (i.e., $\forall M, \forall t, X_{M,t} \sim \mathcal{B}(1, p_M)$).
Votre objectif est d'évaluer différentes stratégies de jeu ayant pour objectif de maximiser votre gain cumulé sur un nombre $T$ de parties. Vous allez jouer successivement, dans l'ordre que vous le souhaitez avec ces deux machines sous et donc à chaque instant $t \leqslant T$, une stratégie $m$ détermine une machine $m(t) \in \{A, B\}$ et votre gain est alors :

$$G_m(T) = \sum_{t=1}^{T} X_{m(t),t}$$

On supposera typiquement que $T$ est grand (de l'ordre de 1 000 ou 10 000) et on va s'intéresser à $\mathbb{E}[G_m(T)]$ pour différentes stratégies $m$.

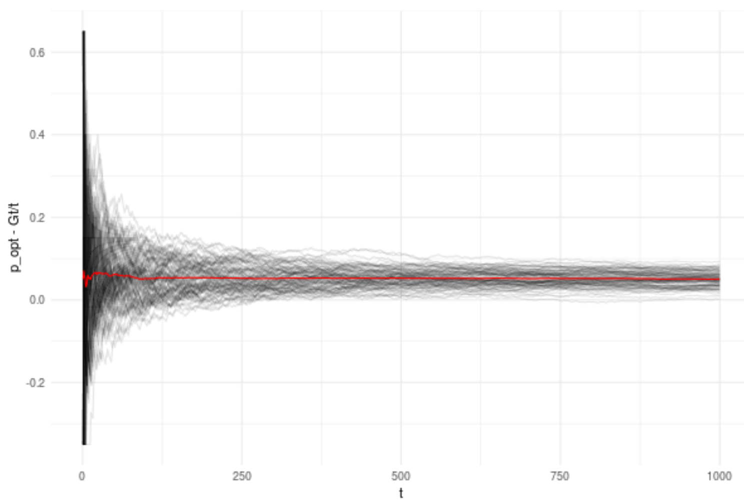
▹ **Q1. Questions préliminaires** Donner une forme close (en fonction de $T$, $p_A$ et $p_B$) pour $\mathbb{E}[G_m(T)]$ quand :

- $m$ est la stratégie qui consiste à jouer systématiquement la machine $M_A$ ;
- $m$ est la stratégie qui consiste à jouer systématiquement la machine $M_B$ ;
- $m$ est la stratégie qui consiste à jouer la machine $M_A$ avec probabilité $1/2$ et la machine $M_B$ avec probabilité $1/2$.

Pour des stratégies $m$ plus adaptatives, il va être très compliqué de calculer explicitement $\mathbb{E}[G_m(T)]$. Dans la suite, on réalisera donc des estimations en R basées sur au moins $N \approx 100$ trajectoires (vous devrez adapter ce $N$ si cela vous semble nécessaire). On étudiera spécifiquement le cas $(p_A, p_B) = (.65, .55)$ mais vous pourrez jouer sur ces paramètres pour étudier la pertinence des stratégies mises en œuvre.

- Justifier pourquoi on s'intéressera à la quantité $R_m(t) = p_A - \frac{\mathbb{E}[G_m(t)]}{t}$, que l'on appellera regret de la stratégie $m$, pour évaluer l'efficacité de nos stratégie. Qu'attendrait-on d'une "bonne" stratégie ?

Conseil : *Dans votre code* `R`*, évitez d'utiliser une variable notée* `T`*. Utilisez plutôt* `Tmax`*.*
Conseil : *Dans la suite, on vous demandera de tracer $R_m(t)$ au cours du temps. Voici typiquement à quoi ce type de graphe peut ressembler, les courbes grises correspondant à 100 trajectoires et la courbe rouge correspondant à la moyenne de ces 100 trajectoires à chaque instant.*




Conseil : *Dans votre code* `R`*, faites une seule fonction de simulation bien paramétrée et qui implémentera les différentes stratégies à l'aide d'un* `switch`*. Faites en sorte qu'elle renvoie l'ensemble des data-frame dans une seule data-frame.*

▹ **Q2. On apprend (un peu) puis on exploite** On se propose d'étudier le regret de la stratégie $m_L$ consistant à consacrer les $\varepsilon.T$ premières parties (avec $\varepsilon = 0.05$ par exemple) à estimer les probabilités $p_A$ et $p_B$ (chaque machine est essayée alternativement) et à ensuite jouer systématiquement la machine la plus prometteuse (celle qui a rapporté le gain le plus élevé sur les étapes $1 \ldots \varepsilon$.T).

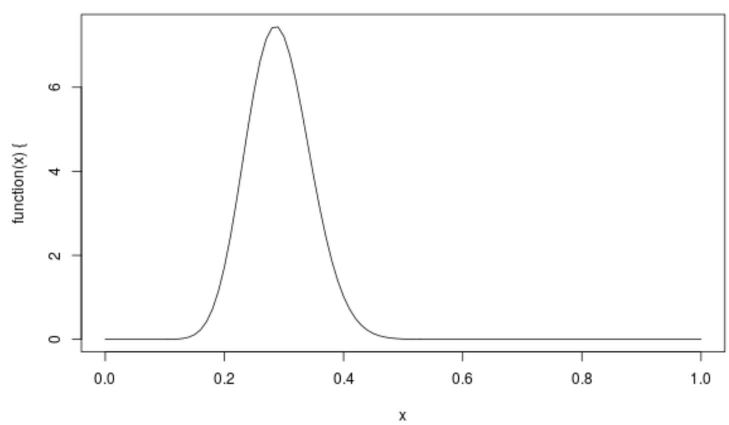
- Tracer une estimation de $R_{m_L}(t)$ au cours du temps $t$.
- La stratégie $m_L$ vous semble-t-elle une bonne stratégie ? Pourquoi ?
  Conseil : *Vous pourrez par exemple regarder certaines trajectoires particulières où cette stratégie se "trompe" et expliquer pourquoi.*

▷ **Q3. On exploite mais on se force à toujours apprendre (un peu)** On se propose d'étudier le regret de la stratégie $m_G$ consistant à sélectionner avec probabilité $1-\hat{\varepsilon}$ (avec $\hat{\varepsilon} = 0.1$ par exemple) la machine la plus prometteuse (i.e., celle dont le ratio du "nombre de succès" par le "nombre de tentatives plus un", est le plus élevé) jusqu'ici et avec probabilité $\hat{\varepsilon}$ l'autre machine. En cas d'égalité entre les deux machines, on prendra soin de choisir au hasard entre ces deux dernières pour ne pas favoriser l'une plus que l'autre.

- Tracer une estimation de $R_{m_G}(t)$ au cours du temps $t$.
- La stratégie $m_G$ vous semble-t-elle une bonne stratégie ? Pourquoi ?

▷ **Q4. On tire au hasard en biaisant selon nos observations** Soit $M$ une machine. Comme pour la stratégie précédente, si on a gagné $n_1$ fois avec cette machine et perdu $n_2$ fois, sa probabilité de gain la plus vraisemblable est $n_1/(n_1+n_2)$. Mais il est tout à fait possible que cette valeur soit une sur- ou une sous-estimation de la valeur réelle $p_M$. On admettra que la *vraisemblance* de $p_M$ suit une loi bêta de paramètres $(n_1+1, n_2+1)$. Voici un exemple de cette distribution pour $n_1 = 20$ et $n_2 = 50$.

```
n1=20 ; n2=50 ; plot(function(x) {dbeta(x,shape1=n1+1,shape2=n2+1)})
```




On propose d'utiliser la stratégie $m_T$ suivante : à chaque étape $t$, on tire un nombre aléatoire suivant la densité beta correspondant à chacune des machines $A$ et $B$ (on obtient ainsi une valeur vraisemblable de $p_A$ et de $p_B$) et on choisit la machine ayant obtenu la plus grande valeur.

- Tracer une estimation de $R_{m_T}(t)$ au cours du temps $t$.
- La stratégie $m_T$ vous semble-t-elle une bonne stratégie ? Pourquoi ?

## Sujet 2 : Correction

Je commence par écrire une fonction simulant les différentes stratégies. Je stoquerai mes trajectoires dans une data frame.

```
simulation = function(Tmax=1000, N=100, p = c(0.65, 0.55),policy="mixed") {
  df=data.frame();
  for(traj in 1:N) {
    G=0;
    Gt=c();
    Gain=c(0,0);
    Trials=c(0,0);
    for(t in 1:Tmax) {
      switch(policy,
            mixed = { m = sample(size=1,x=c(1,2)); },
            learn = { if(t<.05*Tmax) {
                  m=(t%%2)+1;
                } else {
                  if(Gain[1]>Gain[2]) { m=1; }
                  else { m=2; }
                } },
            greedy = {
              if(Gain[1]/(Trials[1]+1) == Gain[2]/(Trials[2]+1)) {
                  m=sample(size=1,x=c(1,2));
              } else if(Gain[1]/(Trials[1]+1)>Gain[2]/(Trials[2]+1)) { m=1; }
              else { m=2; }
              if(runif(1)<.1) {
                m=3-m; # sample(size=1,x=c(1,2));
              }
            },
            thompson = {
              v1 = rbeta(n=1, shape1=Gain[1]+1, shape2=Trials[1]-Gain[1]+1)
              v2 = rbeta(n=1, shape1=Gain[2]+1, shape2=Trials[2]-Gain[2]+1)
              if(v1>v2) { m=1; }
              else { m=2; }
            },
            stop("Enter something that switches me!")
            );
      outcome=sample(size=1,x=c(1,0),prob=c(p[m],1-p[m]));
      Gain[m]=Gain[m]+outcome;
      Trials[m]=Trials[m]+1;
      G=G+outcome
      Gt[t]=G;
    }
    df=rbind(df,data.frame(Idx=traj,t=1:Tmax,Gt=Gt));
  }
  return(df)
}
set.seed(42)
df = simulation(policy="greedy");
```

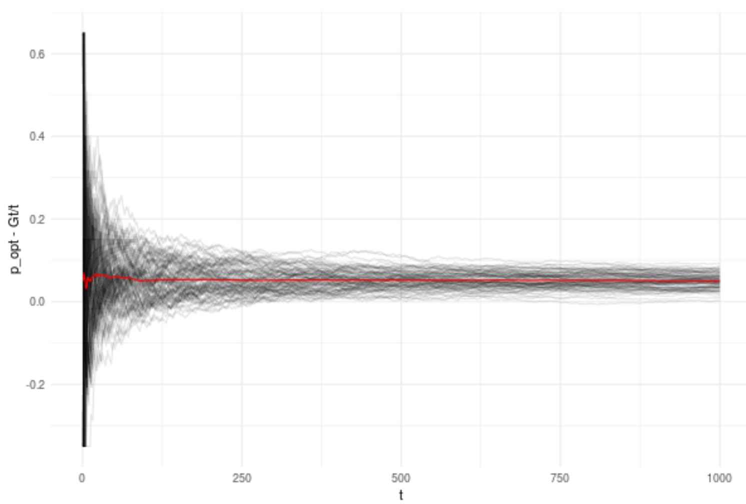
```
head(df)
```

```
  Idx t Gt
1   1 1  1
2   1 2  1
3   1 3  1
4   1 4  1
5   1 5  2
6   1 6  2
```

Je peux ensuite générer une figure me montrant l'ensemble de mes trajectoires ainsi que l'estimation (en rouge) de l'espérance à chaque pas de temps.

```
library(ggplot2)
library(dplyr)

set.seed(42)
df = simulation(policy="mixed");
p_opt=0.65;
df %>% group_by(t) %>% summarize(Gtm=mean(Gt)) -> df_summarized
ggplot(data=df,aes(x=t,y=p_opt - Gt/t)) + theme_minimal() +
    geom_line(aes(group=Idx),alpha=.1) +
    geom_line(data=df_summarized,aes(y=p_opt - Gtm/t),color="red")
```




Comme je réutilisera régulièrement ce type de visulisation, je factorise.

```
plot_bandit = function(df) {
    p_opt=0.65;
    df %>% group_by(t) %>% summarize(Gtm=mean(Gt)) -> df_summarized
    ggplot(data=df,aes(x=t,y=p_opt - Gt/t)) + theme_minimal() +
        geom_line(aes(group=Idx),alpha=.1) +
        geom_line(data=df_summarized,aes(y=p_opt - Gtm/t),color="red")
}
```

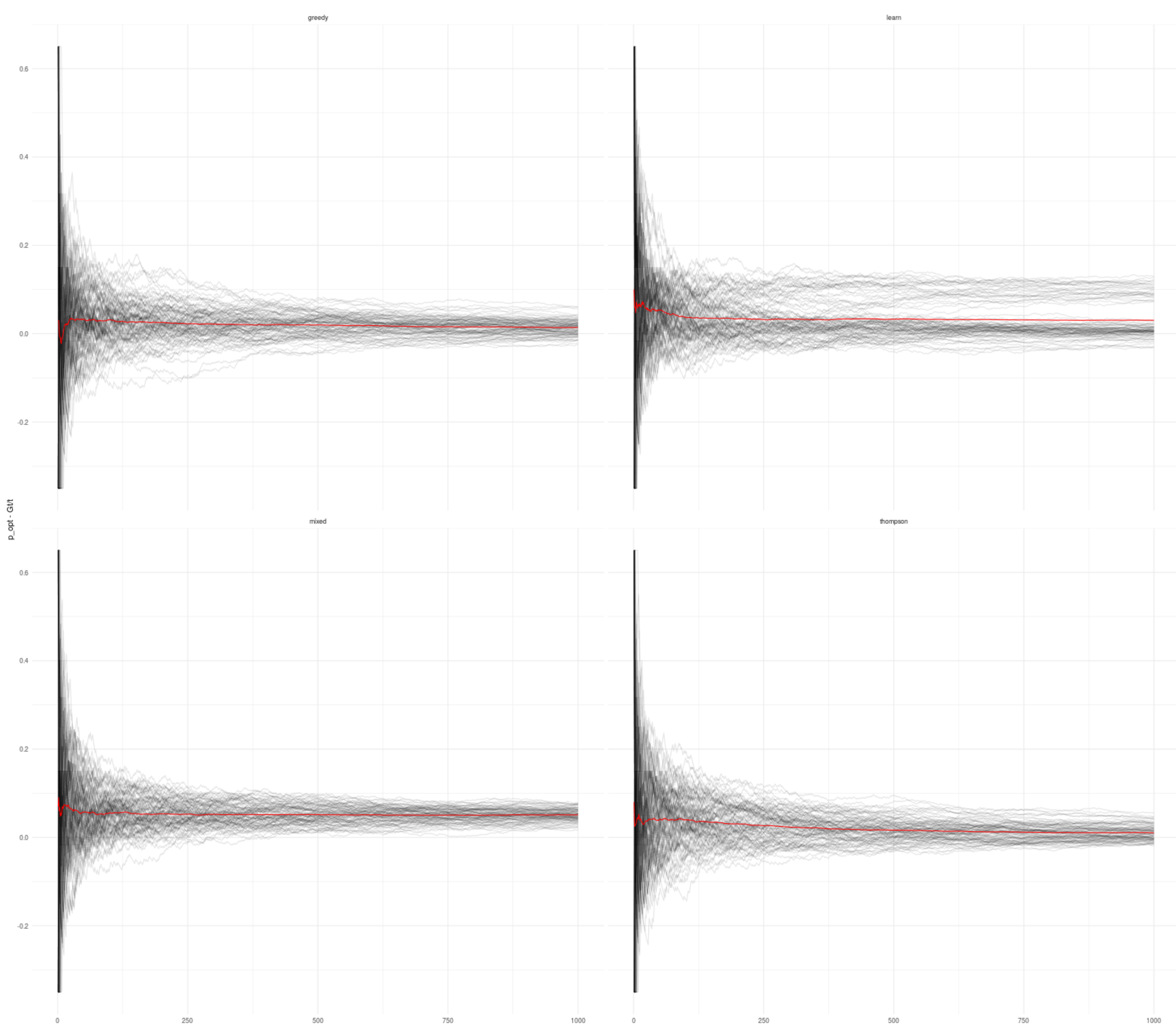
Je peux alor lancer une simulation pour les quatre stratégies et les regrouper dans une unique data-frame afin de pouvoir les comparer :

```
df_mixed = simulation(policy="mixed");df_mixed$policy="mixed";
df_learn = simulation(policy="learn");df_learn$policy="learn";
df_greedy = simulation(policy="greedy");df_greedy$policy="greedy";
df_thompson = simulation(policy="thompson");df_thompson$policy="thompson";
df=rbind(df_mixed, df_learn, df_greedy, df_thompson);
```

Je peux alors visualiser l'ensemble des trajectoires des quatre stratégies, même si la comparaison risque de ne pas etre très lisible

```
df %>% group_by(t,policy) %>% summarize(Gtm=mean(Gt)) -> df_summarized

ggplot(data=df,aes(x=t,y=p_opt - Gt/t)) + theme_minimal() +
    geom_line(aes(group=Idx),alpha=.1) +
    geom_line(data=df_summarized,aes(y=p_opt - Gtm/t),color="red") +
    facet_wrap(~policy)
```




Voici le comportement moyen de l'ensemble des stratégies qui permettra de mieux les comparer :

```
ggplot(data=df_summarized,aes(x=t,y=p_opt - Gtm/t,color=policy)) +
    theme_minimal() + geom_line() + geom_hline(yintercept=0)
```