Méthodes de Monte Carlo

Travaux dirigés

Pierre Gloaguen

Contents

Références pour la simulation de loi sous `R`

`R` dispose d'un ensemble de fonctions pour générer les lois usuelles (multinomiale avec `sample`, loi uniforme avec `runif`, loi normale avec `rnorm`, etc . . .).

En plus de l'aide de ces fonctions (`help(rnorm)`, par exemple), on pourra se reférer à la partie 5 du polycopié de Christophe Chesneau.

1 Première implémentation

On cherche à évaluer la valeur de l'intégrale suivante:

$$I = \int_{\mathbb{R}^2} \cos^2(x)\sin^2(3y)\exp(-(x^2+y^2))\mathrm{d}x\mathrm{d}y$$

1. Ecrire un estimateur de Monte Carlo, noté $\hat{I}_M$ (où M est l'effort Monte Carlo) pour cette intégrale.

On remarque que:

$$\begin{aligned} I &= \int_{\mathbb{R}^2} \cos^2(x)\sin^2(3y)\exp(-(x^2+y^2))dxdy \\ &= \int_{\mathbb{R}^2} \overbrace{\cos^2(x)\sin^2(3y)\times 2\pi\times\sigma^2}^{:\varphi(z)} \times \frac{1}{\sqrt{2\pi\sigma^2}}\exp(-\frac{x^2}{2\sigma^2}) \times \frac{1}{\sqrt{2\pi\sigma^2}}\exp(-\frac{y^2}{2\sigma^2})\mathrm{d}x\mathrm{d}y \end{aligned}$$

où $\sigma^2 = 0.5$.

Donc

$$I = \int_{\mathbb{R}^2} \varphi(z) f(z) dz = \mathbb{E}[\varphi(Z)],\ Z \sim \mathcal{N}_2(0, \sigma^2)$$

Donc, on a un estimateur Monte Carlo pour M donné:

$$\hat{I}_M = \frac{1}{M} \sum_{k=1}^{M} \varphi(Z_k)$$

où $Z_1, \dots, Z_M$ sont i.i.d. de loi $Z \sim \mathcal{N}_2(0, \sigma^2)$

2. À l'aide du logiciel `R`, donnez une estimation de la valeur de cette intégrale pour un effort de Monte Carlo $M = 10000$. Pour simuler une loi normale sous `R`, vous utiliserez la fonction `rnorm` (voir `help(rnorm)`).

On commence par définir la fonction $\varphi(z)$:

```
phi_function <- function(z){
  x <- z[1]
  y <- z[2]
  cos(x)^2 * sin(3 * y)^2 * 2 * pi * 0.5
}
```

Ensuite, on écrit une fonction tirant des échantillons Monte Carlo et les stockant dans un tableau de type 'tibble', équivalent à un 'data.frame'.

```
get_monte_carlo_samples <- function(M){
  # Rerun is a function from included in the tidyverse packages to
  # perform multiple times the same instruction, it returns a list
  samples <- rerun(M, # Rerun M times
                   phi_function(rnorm(2, 0, sqrt(0.5))) # The same instruction
                   ) %>% # Pass the list to the
    unlist() # unlist function to go from list to vector
  # Return the samples in form of a tibble
  return(tibble(index = 1:M, sample = samples))
}
```

Ainsi, on utilise cette fonction pour $M = 10000$. La moyenne empirique de la colonne 'sample' donne le résultat.

```
library(tidyverse) # Set of packages to handle and visualize data
my_M <- 1e4
my_samples <- get_monte_carlo_samples(my_M) # A tibble containing all the samples
mean(my_samples$sample) # Monte Carlo estimate
```

```
## [1] 1.083177
```

3. Quelle est la variance de $\hat{I}_1$? À l'aide des simulations obtenues précedemment, obtenez une estimation de cette variance. Servez vous de cette estimation pour calculer un intervalle de confiance asymptotique à 95% pour I.

Par définition, la variance de l'estimateur de taille 1 est:

$$\gamma^2 := \mathbb{V}[\hat{I}_1] = \mathbb{E}[\varphi(Z)^2] - I^2$$

Cette variance est inconnue. Elle peut être estimée par un estimateur classique de la variance:

$$\hat{\gamma}^2 = \frac{1}{M} \sum_{k=1}^{M} \left(\varphi(Z_k) - \hat{I}_M\right)^2$$

En ‘R‘, il suffit d’utiliser ‘var‘.

Un intervalle de confiance asymptotique à 95

$$J_M = [\hat{I}_M - z_{0.975}\sqrt{\frac{\hat{\gamma}^2}{M}}, \hat{I}_M + z_{0.975}\sqrt{\frac{\hat{\gamma}^2}{M}}]$$

où $z_{0.975}$ est le quantile d’ordre 0.975 d’une $\mathcal{N}(0, 1)$.

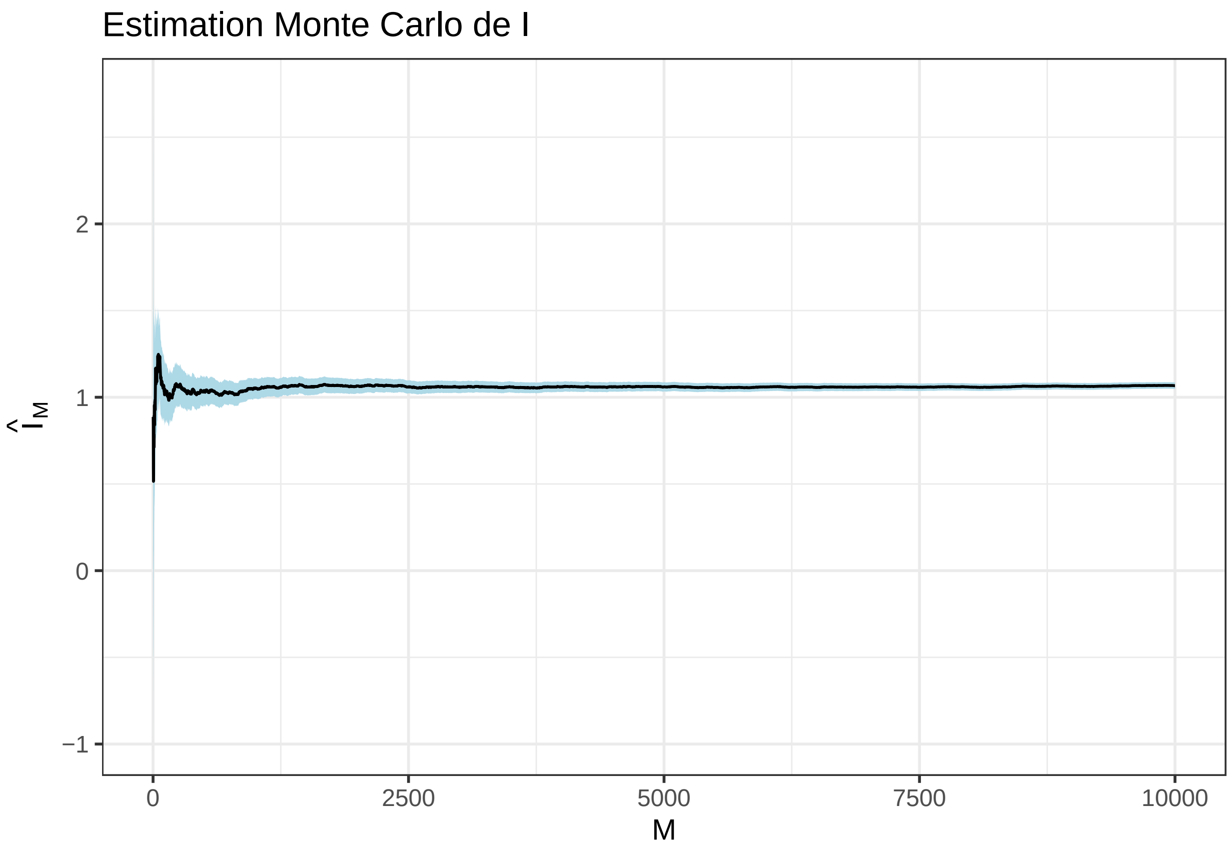
```
I_hat <- mean(my_samples$sample) # Monte Carlo estimate of I
gamma2_hat <- var(my_samples$sample) # Monte Carlo variance estimate
z_975 <- qnorm(0.975, 0, 1) # 0.975 quantile of a normal distribution
I_hat + z_975 * sqrt(gamma2_hat / my_M) * c(-1, 1) # 95% conf. inter.
```

```
## [1] 1.064338 1.102016
```

4. Représentez graphiquement l’évolution de votre estimation en fonction de M ainsi que l’intervalle de confiance associé.

On peut en fait tracer l’évolution de cet intervalle de confiance pour M allant de 1 à 10000. L’estimation de I évolue (ici, on la trace avec le $\hat{\gamma}^2$ final) au fil de l’eau.

```
my_samples %>% # From the initial data
  # We create columns with the mutate function
  mutate(Estimate = cumsum(sample) / index, # On the fly estimates of I
         IC_95_inf = Estimate - z_975 * sqrt(gamma2_hat / index), # Inf IC bound
         IC_95_sup = Estimate + z_975 * sqrt(gamma2_hat / index)) %>%
  # Then we plot the results
  ggplot(aes(x = index)) +
  geom_ribbon(aes(ymin = IC_95_inf, ymax = IC_95_sup), # Columns delimiting the ribbon
              fill = "lightblue") + # Fill the ribbon with blue
  geom_line(aes(y = Estimate)) + # Add the estimate line
  # Now, custom the graph
  labs(x = "M", title = "Estimation Monte Carlo de I",
       y = expression(hat(I)[M])) + # You can add math expressions
  theme_bw() # Black and white background
```



2 Aiguille de Buffon

Au XVIIIe siècle, naturaliste Georges Louis Leclerc de Buffon pose le problème suivant:

On considère un parquet avec une infinité de lattes de longueurs infinies, toutes de largeur 1. On considère ensuite l'expérience suivante: On jette une aiguille de longueur 1 en l'air, qui retombe ensuite sur le parquet. On cherche alors à calculer la probabilité que l'aiguille croise le bord d'une des lattes.

Le centre de l'aiguille tombant toujours entre deux lattes, on notera X la variable aléatoire correspondant à son ordonnée (on visualisera les lattes comme disposées "horizontalement"), comprise entre 0 et 1.

On notera θ l'angle formé par l'aiguille avec l'horizontale. θ est donc compris entre 0 et $\frac{\pi}{2}$.

On suppose que X et θ sont deux variables aléatoires indépendantes distribuées selon des lois uniformes sur $[0, 1]$ et $[0, \frac{\pi}{2}]$ respectivement.

1. Montrer que la probabilité qu'une aiguille croise une latte dans ces conditions est de $\frac{2}{\pi}$.

L'évènement "l'aiguille croise une latte" arrive quand:

- Le centre de l'aiguille est à une distance $< \frac{1}{2}$ de la latte inférieure **et** $X \leq \frac{1}{2} \sin \theta$.

- Le centre de l'aiguille est à une distance $> \frac{1}{2}$ de la latte supérieure **et** $X \geq 1 - \frac{1}{2} \sin \theta$

Donc

$$\begin{aligned}
p^* := \mathbb{P}(\text{Croisement}) &= \mathbb{E}[\mathbf{1}_{X \leq \frac{1}{2}\sin\theta}] + \mathbb{E}[\mathbf{1}_{X \geq 1-\frac{1}{2}\sin\theta}] \\
&= \frac{2}{\pi}\int_0^{\frac{\pi}{2}}\int_0^1 \mathbf{1}_{x \leq \frac{1}{2}\sin z} + \mathbf{1}_{x \geq 1-\frac{1}{2}\sin z} \mathrm{d}x\mathrm{d}z \\
&= \frac{2}{\pi}\int_0^{\frac{\pi}{2}} \left(\frac{1}{2}\sin z + \frac{1}{2}\sin z\right)\mathrm{d}z \\
&= \frac{2}{\pi}
\end{aligned}$$

2. Proposer un estimateur Monte Carlo de cette probabilité.

Ainsi, un estimateur de Monte Carlo de p^* peut être obtenu si on simule $X_1, \ldots, X_n$ un échantillon i.i.d. de V.A. uniformes sur $[0,1]$ et un échantillon $\theta_1, \ldots, \theta_n$ i.i.d. de V.A. uniformes sur $[0, \frac{\pi}{2}]$. On a alors:

$$\hat{p}^*_M = \frac{1}{M}\sum_{k=1}^{M} \mathbf{1}_{X_k \leq \frac{1}{2}\sin\theta_k} + \mathbf{1}_{X_k \geq 1-\frac{1}{2}\sin\theta_k}$$

3. En déduire un estimateur de Monte Carlo de la valeur de π.

On a tout simplement $\hat{\pi}_M = \frac{2}{\hat{p}^*_M}$.

4. Donner un intervalle de confiance asymptotique à 95% pour cet estimateur.

Réponse à σ_p^2 connu:

On note Y_k la variable aléatoire $\mathbf{1}_{X_k \leq \frac{1}{2}\sin\theta_k} + \mathbf{1}_{X_k \geq 1-\frac{1}{2}\sin\theta_k}$. Y_k suit donc une loi de Bernouilli de paramètre $\frac{2}{\pi}$. La variance de $\hat{p}^*_1$ est simplement la variance d'une variable aléatoire de Bernouilli de paramètre $\frac{2}{\pi}$, soit $\frac{2}{\pi}(1-\frac{2}{\pi})$. Donc,

$$\sqrt{M}(\hat{p}^*_M - p^*) \to \mathcal{N}(0, \sigma_p^2 = \frac{2}{\pi}(1-\frac{2}{\pi}))$$

On utilise ensuite la Δ méthode avec $h(x) = \frac{2}{x}$. On a alors, $(h'(p^*))^2 = \frac{4}{(p^*)^4} = \frac{\pi^4}{4}$. Donc, par la Δ méthode, on a la variance théorique

$$\sqrt{M}(\hat{\pi}_M - \pi) \to \mathcal{N}(0, \sigma_\pi^2 = \frac{\pi^4}{4}\sigma_p^2)$$

Réponse à σ_p^2 inconnu:

Dans le cas où on utilise la variance empirique (ce qui est le cas dans les vrais problèmes), on a l'estimateur de la variance empirique suivant pour σ_p^2:

$$\hat{\sigma}^2_{p,M} = \frac{1}{M}\sum_{k=1}^{M}(Y_k - \hat{p}^*_M)^2$$

Cet estimateur est consistant. Par la Δ-méthode, l'estimateur de la variance pour $\hat{pi}$ est donné par:

$$\hat{\sigma}^2_{\pi,M} = (h'(\hat{p}^*_M))^2\hat{\sigma}^2_{p,M} = \frac{4}{(\hat{p}^*_M)^4}\hat{\sigma}^2_{p,M}$$

Ainsi, un intervalle de confiance à 95% est donné par:

$$IC_{\pi,M}(0.95) = \left[\hat{\pi}_M - 1.96\sqrt{\frac{\hat{\sigma}^2_{\pi,M}}{M}}, \hat{\pi}_M + 1.96\sqrt{\frac{\hat{\sigma}^2_{\pi,M}}{M}}\right]$$

Asymptotiquement, on a:

$$\mathbb{P}(IC_{\pi,n}(0.95) \ni \pi) \longrightarrow 0.95$$

5. Sur R, tracez, en fonction du nombres de simulation de Monte Carlo, l'estimation de π trouvée.

On commence par écrire une fonction qui contient toutes les quantités calculées. Dans le code suivant, on stocke en ligne la moyenne empirique et la varianceempirique à l'aide de la fonction `cumsum`, on peut obtenir le vecteur des sommes cumulées.

Les simulations de lois uniformes sont faites avec `runif`.

```
rm(list = ls()) # Clean environment
library(tidyverse)
get_pi_estimate <- function(n_sim){
  x_sample <- runif(n_sim, 0, 1) # Simulation des X
  theta_sample <- runif(n_sim, 0, pi / 2) # Simulation des thetas
  crossing <- (x_sample < 0.5 * sin(theta_sample)) |
    (x_sample > 1 - 0.5 * sin(theta_sample))
  z_975 <- qnorm(0.975) # Quantile de la loi normale
  p_hat <- cumsum(crossing) / (1:n_sim) # Estimation de p
  pi_hat <- 2 / p_hat # Estimation de pi
  # Fast way to compute E[f(x)^2] - E[f(x)]^2 on the fly
  sigma2_p_hat <- cumsum(crossing^2) / (1:n_sim) - p_hat^2
  sigma2_pi_hat <- 4 / p_hat^4 *  sigma2_p_hat
  # On stocke tous dans un data.frame
  tibble(index = 1:n_sim,
         phi_x = crossing,
         pi_hat = pi_hat) %>%
    mutate(sup_IC_emp = pi_hat + z_975 * sqrt(sigma2_pi_hat / index),
           inf_IC_emp = pi_hat - z_975 * sqrt(sigma2_pi_hat / index))
}
```

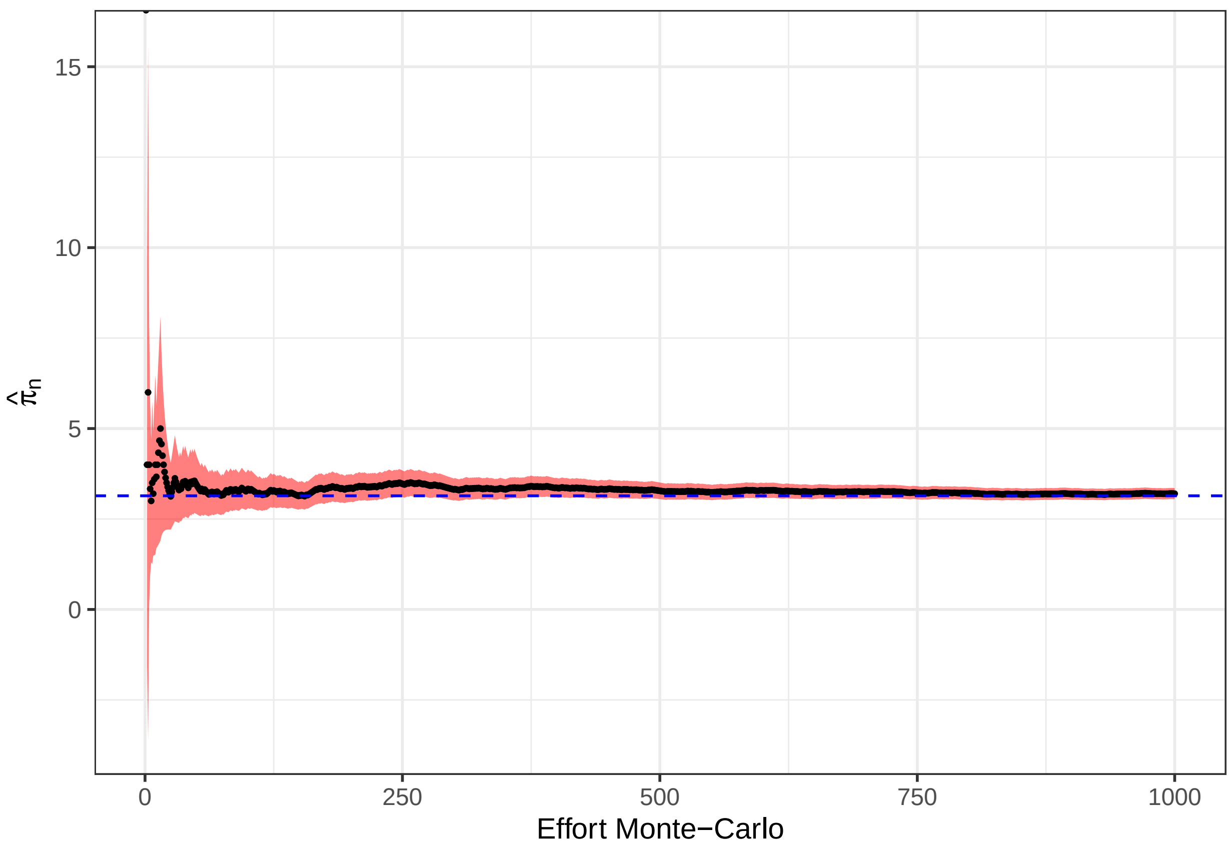
```
# On fait tourner l'algorithme pour M = 1000
set.seed(123)
n_sim <- 1e3
my_pi_estimate <- get_pi_estimate(n_sim = n_sim)# Le tableau complet
# La valeur finale d'estimation de pi est:
tail(my_pi_estimate)
```

```
## # A tibble: 6 x 5
##   index phi_x pi_hat sup_IC_emp inf_IC_emp
##   <int> <lgl>  <dbl>      <dbl>      <dbl>
## 1   995 FALSE   3.20       3.36       3.05
## 2   996 TRUE    3.20       3.36       3.05
## 3   997 FALSE   3.21       3.36       3.05
## 4   998 TRUE    3.20       3.36       3.05
## 5   999 TRUE    3.20       3.36       3.05
## 6  1000 TRUE    3.2        3.35       3.05
```

On peut tracer l'évolution de la valeur de $\hat{\pi}_n$, ainsi que la réalisation de l'intervalle $IC_{\pi,n}(0.95)$.

```
ggplot(my_pi_estimate,
       aes(x = index, y = pi_hat)) +
```

```r
  geom_ribbon(aes(ymin = inf_IC_emp, ymax = sup_IC_emp),
              fill = "red", alpha = 0.5) +
  geom_point(size = 0.5) +
  labs(y = expression(hat(pi)[n]), x = "Effort Monte-Carlo") +
  theme_bw() +
  geom_hline(yintercept = pi, linetype = 2, col = "blue")# Vraie valeur
```



On voit ici clairement que l'estimateur converge vers π. La bande rouge représente la réalisation de l'intervalle de confiance à 95

6. La vitesse d'approximation de π vous semble t'elle bonne?

En pratique, on obtiendra une meilleure précision par d'autres méthodes. Cette méthode est assez lente.

3 Une comparaison avec l'intégration numérique

Cet exercice est une adaptation de l'exercice 1.2 de ce cours en ligne.

On se place dans l'hypercube unitaire de dimension d, autrement dit, l'espace $[0,1]^d$, pour $d \geq 2$.

Soit $0 < \varepsilon < 1/2$, on s'intéresse à évaluer le volume d'une sous région de cette hypercube, à savoir:

$$A_{\varepsilon,d} \cap B_{\varepsilon,d}$$

où

- $A_{\varepsilon,d}$ est l'ensemble des points du cube étant à une distance du bord plus petite que ε. Formellement:

$$A_{\varepsilon,d} = \left\{ x \in [0,1]^d, \min_{1 \leq j \leq d} \min(x_j, 1 - x_j) < \epsilon \right\}$$

- $B_{\varepsilon,d}$ est l'ensemble des points du cube étant à une distance de l'hyperplan $\left\{x \in\in [0,1]^d, \sum_{j=1}^d x_j = \frac{d}{2}\right\}$ plus petite que ε. Formellement:

$$B_{\varepsilon,d} = \left\{x \in [0,1]^d, \frac{1}{\sqrt{d}}\left|\sum_{j=1}^d (x_j - \frac{1}{2})\right| < \epsilon\right\}$$

1. Justifier que le volume considéré grandit avec d. *On pourra justifier que le premier volume tende vers 1 quand $d \to \infty$ et que le second se stabilise vers une valeur finie. L'argument pour le premier volume est purement géométrique, l'argument pour le second peut se déduire du TCL.*

On voit immédiatement que le volume de $A_{\varepsilon,d}$ est donné par

$$\overset{\text{Vol. cube total}}{1} - \overset{\text{Vol. cube. interieur}}{(1-2\varepsilon)^d},$$

ainsi, ce volume tend vers 1 quand d grandit.

Concernant $B_{\varepsilon,d}$, on a que:

$$\begin{aligned}
\text{Vol}(B_{\varepsilon,d}) &= \int_{[0,1]^d} \mathbf{1}_{\frac{1}{\sqrt{d}}\left|\sum_{j=1}^d (x_j - \frac{1}{2})\right| < \epsilon} \text{d}x_1 \text{d}x_2 \dots \text{d}x_d \\
&= \mathbb{P}\left(\frac{1}{\sqrt{d}}\left|\sum_{j=1}^d (X_j - \frac{1}{2})\right| < \epsilon\right), \qquad \text{où } X_j \underset{j=1,\dots,d}{\overset{i.i.d}{\sim}} \mathcal{U}[0,1] \\
&= \mathbb{P}\left(\sqrt{d}\left|\frac{1}{d}\sum_{j=1}^d X_j - \frac{1}{2}\right| < \epsilon\right) \\
&= \mathbb{P}\left(-\varepsilon \leq \sqrt{d}(\bar{X} - \mathbb{E}[X]) \leq \varepsilon\right)
\end{aligned}$$

Cette dernière quantité tend vers la probabilité qu'une loi $\mathcal{N}(0, \frac{1}{12})$ soit comprise entre $-\varepsilon$ et ε.

2. Ecrire le volume recherché sous forme d'une intégrale. En déduire un estimateur Monte Carlo de ce volume.

$$I = \mathbb{E}[\mathbf{1}_{X \in A_{\varepsilon,d} \cap B_{\varepsilon,d}}]$$

où $X \sim \mathcal{U}[0,1]^d$.

Ainsi, on simulera un échantillon $X_1, \dots, X_M$ i.i.d. de loi $\mathcal{U}[0,1]^d$ et on aura

$$\hat{I}_M = \frac{1}{M}\sum_{k=1}^M \mathbf{1}_{X_k \in A_{\varepsilon,d} \cap B_{\varepsilon,d}}$$

3. Donner une estimation de ce volume pour $\varepsilon = 0.1$ et $d = 2, 5, 10, 20$. Vous choisirez vous même l'effort de Monte Carlo, en justifiant ce choix. Donnez l'incertitude associée à votre estimation.

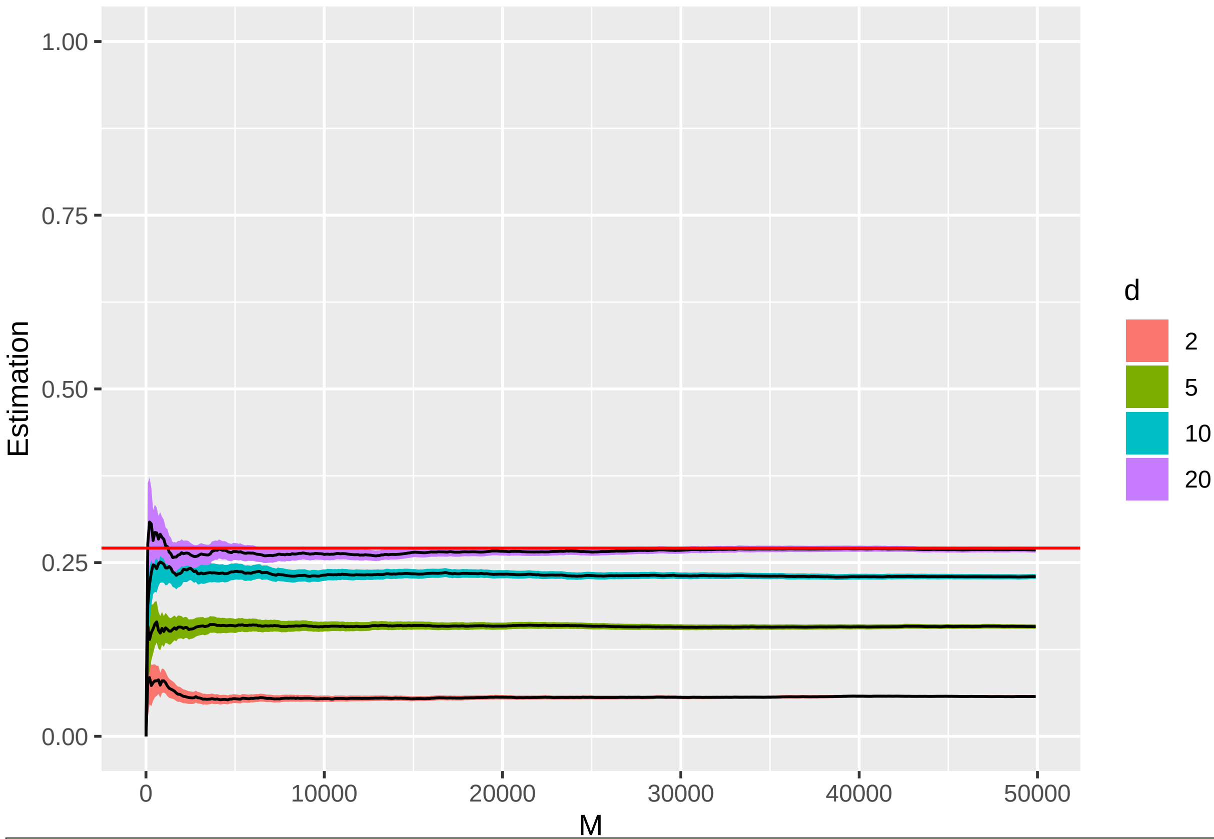
```
# Function to check whether a a d-dimensional vector x is
# in the wanted volume
get_presence <- function(x, d, epsilon){
  # x is a d dimensional vector
  close_to_border <- min(map_dbl(x,
                                 function(x_k){
                                   min(x_k, 1 - x_k)
                                 })) < epsilon
  close_to_diagonal <- abs(sum(x - 0.5)) / sqrt(d) < epsilon
  close_to_border & close_to_diagonal
```

```
}
```

```
# Function outputin monte carlo estimates and estimated variance of estimation
get_volume_estimate <- function(n_sample, d, epsilon){
  presence_sample <- rerun(n_sample,
                          get_presence(runif(d), d, 0.1)) %>% # List of samples
    unlist() # Transform to a vector
  z_975 <- qnorm(0.975)
  tibble(index = 1:n_sample,
         presence = presence_sample) %>%
    mutate(estimate = cumsum(presence) / index,
           variance = cumsum(presence^2) / index - estimate^2,
           borne_inf = estimate - z_975 * sqrt(variance / index),
           borne_sup = estimate + z_975 * sqrt(variance / index),
           d = d)
}
```

```
all_estimates <- map_dfr(c(2, 5, 10, 20), # d arguments
                         get_volume_estimate, # for our function
                         n_sample = 5e4, epsilon = 0.1) # other common arguments
```

```
all_estimates %>%
  slice(seq(1, nrow(.), by = 100)) %>% # Plot one point over 100
  ggplot() +
  aes(x = index, y = estimate) +
  geom_ribbon(aes(ymin = borne_inf, ymax = borne_sup,
                  fill = factor(d))) +
  geom_line(aes(group = factor(d))) +
  labs(x = "M", y = "Estimation",
       fill = "d") +
  coord_cartesian(ylim = c(0, 1)) +
  geom_hline(yintercept = 1 - 2 * pnorm(-0.1, 0, sqrt(1/12)),
             col = "red")
```



On peut regarder l'estimation finale ainsi que l'erreur relative (ecart type Monte Carlo / vraie valeur).

```
all_estimates %>%
  group_by(d) %>%
  summarise(estimate = estimate[n()],
            relative_error = sqrt(variance[n()] / n()) / estimate)
```

```
# A tibble: 4 x 3
      d estimate relative_error
  <dbl>    <dbl>          <dbl>
1     2   0.0572        0.0182
2     5   0.158         0.0103
3    10   0.230         0.00818
4    20   0.268         0.00738
```

4. À l'aide de la fonction `hcubature` du package `cubature`, donnez une valeur du volume obtenue par approximation numérique pour les mêmes valeurs de d.

```
library(cubature) # install.packages(cubature) if necessary
res_integration_numerique <- map_dfr(c(2, 5, 10, 20),
        function(my_dim){
          integration_result <- hcubature(f = get_presence,
                                          lowerLimit = rep(0, my_dim),
                                          upperLimit = rep(1, my_dim),
                                          d = my_dim, epsilon = 0.1,
                                          maxEval = 5e4) # Same effort
          tibble(d = my_dim,
                 estimate = integration_result$integral,
                 relative_error = integration_result$error)
```

```
    })
```

On peut ainsi voir les résultats et comparer avec l'intégration Monte Carlo:

```
res_integration_numerique
```

```
# A tibble: 4 x 3
      d estimate relative_error
  <dbl>    <dbl>          <dbl>
1     2   0.0565      0.0000676
2     5   0.136       0.173
3    10   0.223       1.42
4    20   3.86        9.17
```

5. Comparez les résultats et commentez.

On peut remarquer que dès $d = 5$, l'incertitude est relativement grande pour l'intégration numérique. Cette incertitude explose et la valeur estimée devient aberrant pour $d = 20$.

4 Cas des évènements rares

On se propose d'étudier l'erreur relative de l'estimateur de Monte Carlo de la probabilité p d'un événement E ($0 < p \leq 1$), en fonction de la valeur de p.

On se place dans le cas où pour estimer p, on simule M variables aléatoires indépendantes $X_1, \dots, X_M$ de loi de Bernouilli de paramètre p.

L'estimateur de Monte Carlo de p est donné par

$$\hat{p} = \bar{X}_M = \frac{1}{M} \sum_{k=1}^{M} X_k$$

On s'intéresse à l'erreur relative de $\hat{p}$, à savoir la quantité:

$$\Delta_p = \frac{\hat{p} - p}{p}$$

1. Calculer la variance de Δ_p.

$$\mathbb{V}[\Delta_p] = \frac{1}{p^2} \mathbb{V}[\hat{p}] = \frac{1-p}{Mp}$$

2. Pour $0 < \alpha < 1$, exprimer $\mathbb{P}(|\Delta_p| > \alpha)$ exactement en fonction de la loi d'une variable aléatoire binomiale de paramètres (M, p). Que pouvez vous conjecturer sur cette probabilité quand p devient petit?

$$\begin{aligned}
\mathbb{P}(|\Delta_p| > \alpha) &= \mathbb{P}(\Delta_p > \alpha) + \mathbb{P}(\Delta_p < -\alpha) \\
&= \mathbb{P}(\hat{p} > p(1+\alpha)) + \mathbb{P}(\hat{p} < p(1-\alpha)) \\
&= \mathbb{P}(\bar{X}_M > p(1+\alpha)) + \mathbb{P}(\bar{X}_M < p(1-\alpha)) \\
&= \mathbb{P}(Y > Mp(1+\alpha)) + \mathbb{P}(Y < Mp(1-\alpha)) \text{ où } Y \sim Bin(M,p) \\
&= \mathbb{P}(Y \geq \lfloor Mp(1+\alpha) \rfloor + 1) + \mathbb{P}(Y \leq \lceil Mp(1-\alpha) \rceil - 1)
\end{aligned}$$

Quand p est petit (et Mp proche de 0), ce terme se comporte comme $\mathbb{P}(Y \geq 1) + \mathbb{P}(Y \leq 0) = 1$. Donc l'erreur relative devient dure à contrôler.

3. En utilisant le théorème central limite, donner une expression asymptotique de cette probabilité basée sur la fonction de répartition de la loi normale centrée réduite.

Le TCL nous garantit que:

$$\sqrt{M}(\hat{p} - p) \longrightarrow \mathcal{N}(0, p(1-p))$$

Ainsi on a

$$\sqrt{M}\Delta_p \longrightarrow \mathcal{N}(0, \frac{(1-p)}{p})$$

Donc

$$\mathbb{P}(|\Delta_p| > \alpha) \simeq 2\mathbb{P}\left(Z < -\alpha\sqrt{\frac{Mp}{1-p}}\right) \text{ où } Z \sim \mathcal{N}(0,1)$$

Encore une fois, on conclut au même problème quand p est très proche de 0, cette probabilité tend vers 1 à M fixé.

4.1 Attention à la dimension!

4. On peut montrer que le volume d'une sphère de rayon 1 en dimension $d \geq 2$ est donné par la fonction:

$$V(d) = \frac{\pi^{d/2}}{\Gamma(\frac{d}{2} + 1)}$$

où, pour $z > 0$

$$\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}dt$$

On se propose d'estimer la valeur de π en tirant, en dimension d, une U variable uniforme dans l'hypercube $[-1, 1]^d$. On pose alors $X = \mathbf{1}_{\|U\|^2 \leq 1}$, sur un échantillon de taille $M = 10000$.

a. Quelle est la valeur de p, le paramètre de la loi de Bernouilli de X?

$$p = \frac{V(d)}{2^d}$$

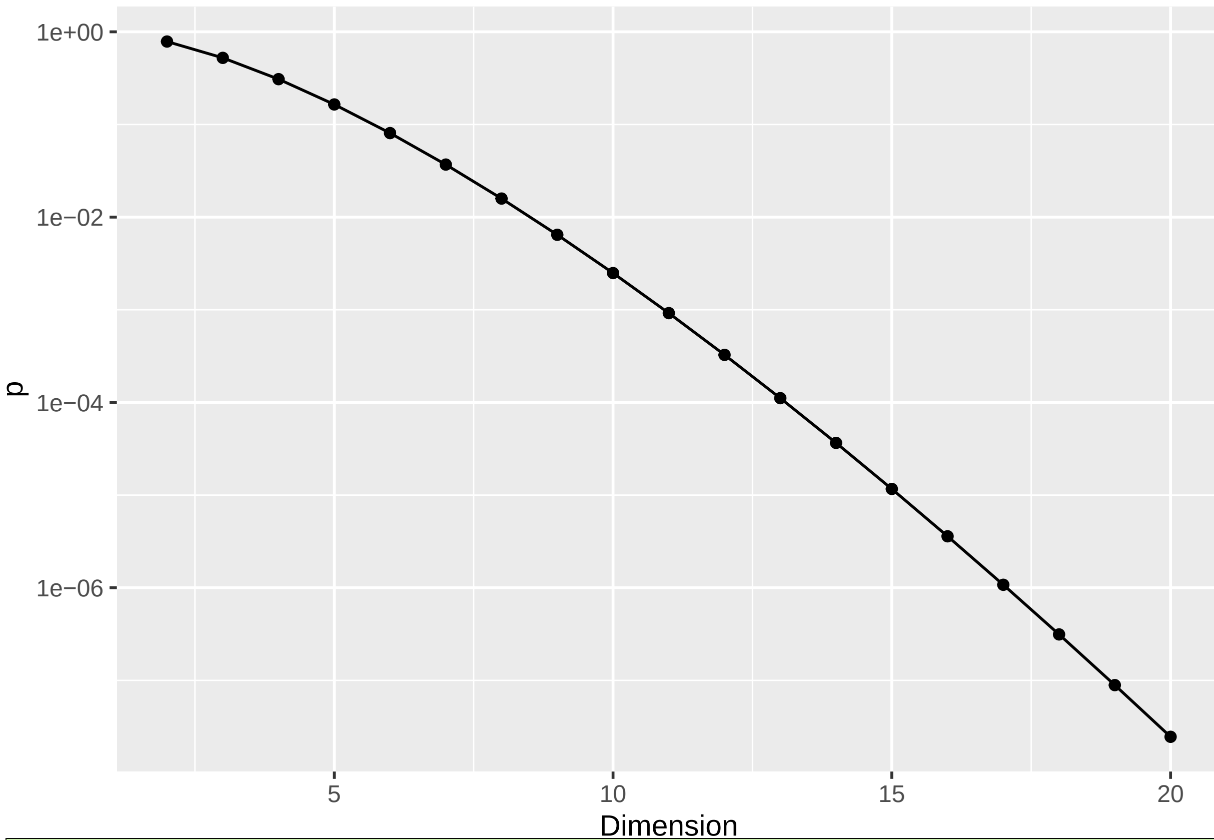
```
b. Donner alors l'estimateur de $\pi$ en fonction de l'estimateur de $p$.
```

Si on prend l'estimateur de p écrit plus haut, on a $\hat{\pi} = \left(\Gamma(\frac{d}{2} + 1) \times \hat{p}\right)^{\frac{2}{d}}$

```
c. Discutez la qualité de l'estimateur quand $d$ grandit. Vous pourrez vous aidez de R pour voir le comp
```

Cet estimateur deviendra très mauvais quand d grandit, en effet, p diminue de manière drastique vers 0!

```r
library(tidyverse)
proba_boule <- function(dimension){# Fonction de calcul de la proba
  pi^(0.5 * dimension) / gamma(0.5 * dimension + 1) / (2^dimension)
}
# On trace cette proba pour les dimensions allant de 2 à 10
data.frame(dimension = 2:20) %>%
  mutate(proba_boule = proba_boule(dimension)) %>%
  ggplot(aes(x = dimension, y = proba_boule)) +
  geom_point() + geom_line() + labs(x = "Dimension", y = "p") +
  scale_y_continuous(trans = "log10") # Echelle ordonnée en log base 10 (non linéaire)
```



La boule unité occupe un volume de plus en plus négligeable dans l'hypercube.

D'après la première partie, on aura beaucoup de mal à estimer π de cette manière pour un grand d!

5 Détection d'aggrégats dans une série temporelle

5.1 Présentation du problème

On s'intéresse à une série temporelle à valeurs dans $\mathbb{R}$. Ainsi, les données consistent en un vecteur $X_{1:n} = (X_1, \dots X_n)$ de valeurs ordonnées dans le temps.

La question est la suivante: *Existe-t-il une fenêtre temporelle de valeurs anormalement élevées?*.

Pour cela, on se propose de faire le test

- H_0: Les variables aléatoires $X_1, \dots, X_n$ sont indépendantes et identiquement distribuées.
- H_1: Il existe une fenêtre temporelle où les valeurs de la série sont plus importantes.

Pour tester cette hypothèse, pour une série temporelle $X_{1:n} = (X_1, \dots, X_n)$, on va définir une statistique de test $T(X_{1:n})$.

Pour l'échantillon aléatoire $X_1, \dots X_n$, on note R_k le rang de X_k parmi les valeurs de l'échantillon (il est égal à 1 si X_k est la valeur la plus faible, à n si X_k est la valeur la plus élevée). Comme on considère des variables aléatoires continues, on considère dans la suite que deux rangs ne peuvent pas être égaux. **Vous remarquerez que l'hypothèse H0 ne fait pas d'hypothèse sur la distribution des valeurs observées, en effet, H0 fait une hypothèse sur la distribution jointe des rangs**.

1. Justifier que, sous H_0, la loi de R_k est une loi uniforme discrète sur $\{1, \dots, n\}$. Quelle est la loi de R_k sachant R_ℓ ($\ell \neq k$)?

Pour tout couple (i,j) tel que $1 \leq i \leq j \leq n$ on considère la variable aléatoire suivante:

$$S(i,j) = \sum_{k=i}^{j} R_k.$$

2. Que représente cette variable aléatoire? Dans quel cas prendra t'elle des grandes valeurs?
3. Montrer que, sous H_0, $m_{ij} := \mathbb{E}[S(i,j)] = \frac{1}{2}(n+1)(j-i+1)$ pour tout couple (i,j).
4. Calculer, sous H_0, $v_{ij} := \mathbb{V}[S(i,j)] = \frac{1}{12}(n+1)(j-i+1)(n-j+i-1)$ pour tout couple (i,j).

On définit maintenant la variable aléatoire centrée et réduite, pour tout couple d'entiers (i,j) tel que $1 \leq i \leq j \leq n$.

$$T(i,j) = \begin{cases} 0 & \text{si } i = 1 \text{ et } j = n \\ \frac{S(i,j) - m_{ij}}{\sqrt{v_{ij}}} & \text{sinon.} \end{cases}$$

Notre statistique de test $T_n(X_{1:n})$ sera donc donnée par

$$T_n(X_{1:n}) = \max_{1 \leq i \leq j \leq n} T(i,j). \tag{1}$$

5.2 Principe du test et prise de décision par méthode de Monte Carlo.

Le principe du test est le suivant: pour un échantillon observé $\mathbf{x}$ un risque α, on rejette H_0 si $T_n(\mathbf{x}) > t_{1-\alpha}$ où $t_{1-\alpha}$ est le quantile d'ordre α de la loi de $T_n(\mathbf{X})$. On concluera que la fenêtre temporelle pour laquelle la statistique est calculée (soit $(i_{\max}, j_{\max}) = \text{argmax}_{i,j}\ T(i,j)$) est anormalement loin de 0 sous H_0. On rejettera alors H_0 pour conclure a un aggrégat de valeurs élevées sur cette fenêtre.

5. La loi de T_n sous H_0 étant inconnue, on se propose d'approcher ses quantiles sous H_0 par méthode de Monte Carlo. Donner un algorithme simple de simulation de T_n sous H_0.
6. Proposer une méthode de Monte Carlo pour répondre à la question initiale à un risque α fixé, pour n'importe quelle série temporelle observée $x_{1:n}$.

5.3 Implémentation sous R pour les températures à Hobart, Tasmanie.

7. Ecrire une fonction `get_tn`, qui pour une série temporelle $x_{1:n}$ donnée, calcule $T_n(x_{1:n})$ et, si on le demande, renvoit les indices temporels de la fenêtre sur laquelle cette statistique est obtenue. Calculer cette statistique de test pour la série des températures à Hobart. On notera cette valeur t^*
8. Ecrire une fonction `get_h0_sample` qui, pour un entier n et un entier M permet d'obtenir M réalisations de T_n sous H_0.
9. Simuler un M échantillon de T_n sous H_0 pour une valeur de n correspondant à celles des données d'Hobart. Vous prendrez $M = 5000$. Représenter l'estimation obtenue de $\mathbb{P}(T_n > t^*)$ ainsi que son intervalle de confiance asymptotique à 95%.
10. Répondre à la question initiale sur les températures à Hobart