TP8 Correction

Clement LAROCHE

10 avril 2019

```
f <- function(x)
{
abs(x-log(2))^(0.1)
}
rectangle <- function(n)
{
pas <- 0:n/n
1/n*sum(f(pas))
}
trapeze <- function(n)
{
pas <- 0:n/n
res <- f(pas)
res <- 0.5*(res[-1]+res[-length(res)])
1/n*sum(res)
}
Simpson <- function(n)
{
pas <- 0:n/n
pm <- pas + 1/(2*n)
pm <- pm[-length(pm)]
res <- 1/(6*n)*sum(f(pas[-length(pas)])+4*f(pm)+f(pas[-1]))
res
}
l <- (log(2)^(1.1)+(1-log(2))^(1.1))*1/1.1
l
```

```
## [1] 0.8553304
```

```
mean(f(runif(10^3)))
```

```
## [1] 0.8555562
```

```
Simpson(10^3)
```

```
## [1] 0.8553438
```

```
trapeze(10^3)
```

```
## [1] 0.8553243
```

```
rectangle(10^3)
```

```
## [1] 0.8562506
```

Approximation d'intégrales par la méthode de Monte-Carlo

Attenzione ! L'expression demandée n'est pas $(b-a)\mathbb{E}(X)$ mais $(b-a)\mathbb{E}(f(X))$

Partons de l'expression de I pour arriver au résultat demandé. On a :

$$\begin{aligned} I &= \int_a^b f(t)\mathrm{d}t \\ &= \int_a^b f(t)\frac{(b-a)}{(b-a)}\mathrm{d}t \\ &= \int_{\mathbb{R}} f(t)\frac{(b-a)}{(b-a)}\mathbb{1}_{[a,b]}(t)\mathrm{d}t \\ &= (b-a)\int_{\mathbb{R}} f(t)\frac{1}{(b-a)}\mathbb{1}_{[a,b]}(t)\mathrm{d}t \end{aligned}$$

On reconnait ici que l'expression $\frac{1}{(b-a)}\mathbb{1}_{[a,b]}(t)$ n'est autre que la densité d'une loi uniforme sur $[a,b]$. Donc on peut conclure que :

$$I = (b-a)\mathbb{E}(f(X))$$

avec $X \sim \mathcal{U}_{[a,b]}$.

Maintenant, rappelons le théorème de la limite centrale pour des variable $(X_n)_{n\in\mathbb{N}}$ indépendantes et identiquement distribuées (i.i.d.) dont les moments d'ordre un et deux sont définis (c'est-à-dire leur espérance et leur variance). On a que :

$$\sqrt{\frac{n}{\mathbb{V}(X)}}(\bar{X}_n - \mathbb{E}(X)) \sim \mathcal{N}(0,1)$$

Ou encore :

$$\sqrt{n}(\bar{X}_n - \mathbb{E}(X)) \sim \mathcal{N}(0, \mathbb{V}(X))$$

avec $\bar{X}_n = \frac{1}{n}\sum_n X_n$.
Ici, nous devons appliquer ce théorème non pas aux variables X_n mais aux variables $f(X_n)$. Alors on se lance ! Soit $\overline{f(X_n)} = \frac{1}{n}\sum_n f(X_n)$, on a que :

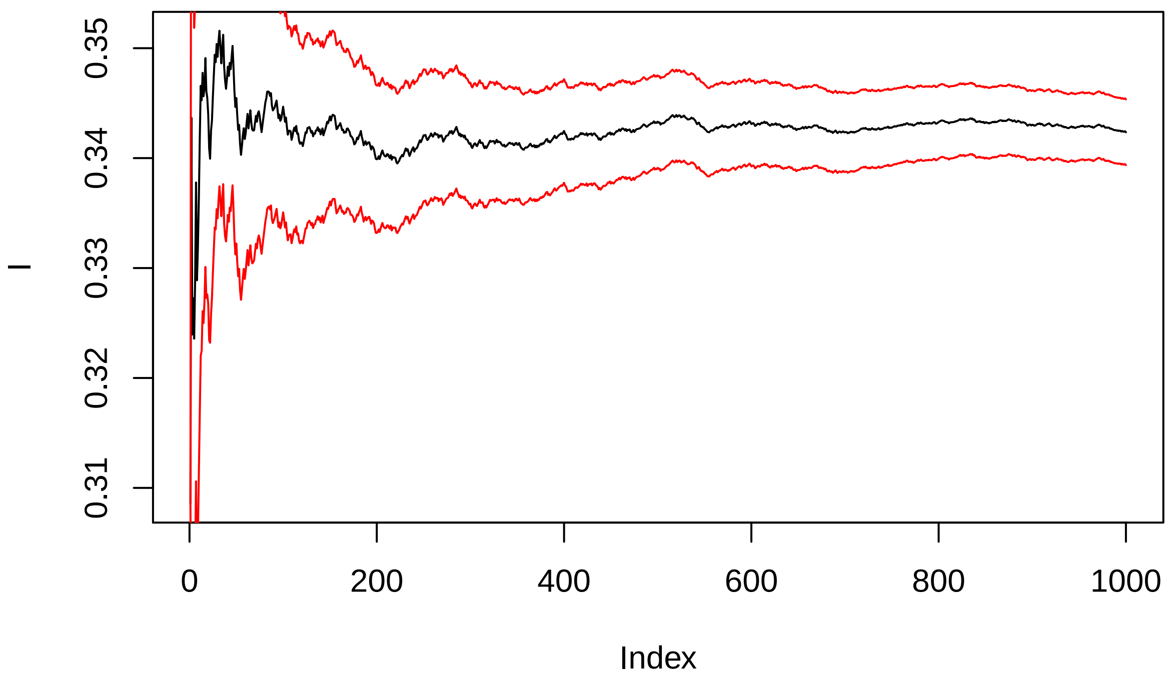
$$\begin{aligned} \sqrt{n}(\overline{f(X_n)} - \mathbb{E}(f(X))) \sim \mathcal{N}(0, \mathbb{V}(f(X))) &\iff \sqrt{n}((b-a)\overline{f(X_n)} - (b-a)\mathbb{E}(f(X))) \sim \mathcal{N}(0, (b-a)^2\mathbb{V}(f(X))) \\ &\iff \sqrt{n}(\hat{I}_n - I) \sim \mathcal{N}(0, (b-a)^2\mathbb{V}(f(X))) \end{aligned}$$

Une fois ce résultat obtenu, étudions $(b-a)^2\mathbb{V}(f(X))$. On a que :

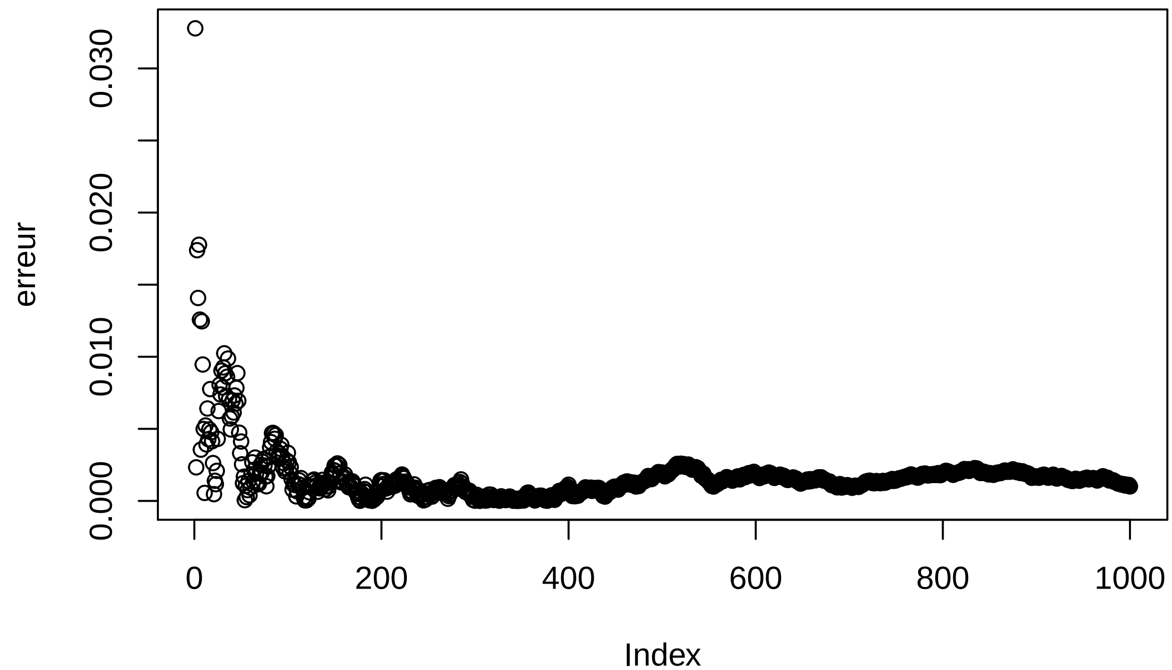
$$(b-a)^2\mathbb{V}(f(X)) = (b-a)^2\mathbb{E}(f(X)^2) - (b-a)^2\mathbb{E}(f(X))^2 = (b-a)^2\mathbb{E}(f(X)^2) - I^2 = (b-a)^2\int_a^b f^2(t)\mathrm{d}t - I^2$$

On retombe bien sur nos pattes. Notez que comme f est intégrable sur un intervalle borné cela implique que f^2 l'est également donc nous n'avons pas de problème de définition de l'intégrale de f^2. Maintenant, on peut se rendre compte que la formule donnée dans l'énoncé de $\hat{\sigma}^2$ est effectivement un bon estimateur de cette variance.
On obtient donc grâce au théorème de la limite centrale et des quantiles de la loi normale un bel intervalle de confiance à 95% de notre I.
On passe au code !!

```r
n <- 10^3
x <- runif(n)
f <- function(x)
{
  1/sqrt(2*pi)*exp(-0.5*x^2)
}
# on pourrait utiliser dnorm bien sûr
MC <- f(x)
I <- 1/(1:n)*cumsum(MC)
sigma <- 1/(1:n)*cumsum(f(x)^2) -I^2
# on va effectuer un beau tracé de l'intervalle
# de confiance en rouge et dee I_n en noir
{
  plot(I,type="l")
  lines(x = 1:n,y = I - qnorm(0.975)*sqrt(sigma)/sqrt(1:n),col = "red")
  lines(x = 1:n,y = I + qnorm(0.975)*sqrt(sigma)/sqrt(1:n),col = "red")
}#I on utilise qnorm fonction quantile de la loi normale
```




```r
# on regarde l'erreur en fonction de n
erreur <- abs(pnorm(1)-pnorm(0)-I)
plot(erreur)
```



On voit que l'approximation n'est pas terrible ! Comparé au TP6, les méthodes numériques sont beaucoup plus performantes !!

On calcule théoriquement la nouvelle intégrale demandée :

$$
\begin{aligned}
I &= \int_0^1 |t-\ln(2)|^{0.1}\mathrm{d}t \\
&= \int_0^{\ln(2)} (\ln(2)-t)^{0.1}\mathrm{d}t + \int_{\ln(2)}^1 (t-\ln(2))^{0.1}\mathrm{d}t \\
&= \left[-(\ln(2)-t)^{1,1}\right]_0^{\ln 2} + \left[(t-\ln(2))^{1,1}\right]_{\ln(2)}^1 \\
&= \ln(2)^{1.1} + (1-\ln(2))^{1.1}
\end{aligned}
$$

On passe au code !

```
f <- function(x)
{
  abs(x-log(2))^(0.1)
}

rectangle <- function(n)
{
  pas <- 0:n/n
  1/n*sum(f(pas))
}

trapeze <- function(n)
{
```

```
  pas <- 0:n/n
  res <- f(pas)
  res <- 0.5*(res[-1]+res[-length(res)])
  1/n*sum(res)
}

Simpson <- function(n)
{
  pas <- 0:n/n
  pm <- pas + 1/(2*n)
  pm <- pm[-length(pm)]
  res <- 1/(6*n)*sum(f(pas[-length(pas)])+4*f(pm)+f(pas[-1]))
  res
}

l <- log(2)^(1.1)+(1-log(2))^(1.1)
l
```

```
## [1] 0.9408634
```

```
rectangle(10^6)
```

```
## [1] 0.8553313
```

```
trapeze(10^6)
```

```
## [1] 0.8553304
```

```
Simpson(10^6)
```

```
## [1] 0.8553304
```

On voit que les 3 méthodes sont inefficaces !!! Essayons Monte Carlo.

```
x <- mean(f(runif(10^6)))
x
```

```
## [1] 0.8553054
```

Régression linéaire simple par moindres carrés

On nous demande de retrouver la formules des moindres carrés dans le cas d'une régression linéaire simple. Ce résultat se retrouve en dérivant par rapport à a et b la fonction de deux variables suivantes :

$$d_n(a,b) = \sum_{i=1}^{n}(Y_i - (aX_i + b))^2$$

Calculons la dérivée partielle par rapport à a :

$$\frac{d_n(a,b)}{a} = -2\sum_{i=1}^{n} X_i(Y_i - (aX_i + b))$$

Calculons maintenant celle par rapport à b.

$$\frac{d_n(a,b)}{b} = -2\sum_{i=1}^{n}(Y_i - (aX_i + b))$$

Le minimum de cette fonction est le couple $(\hat{a}, \hat{b})$ qui annule les deux dérivées partielles. On obtient donc le système d'équations dont les solutions sont les suivantes :

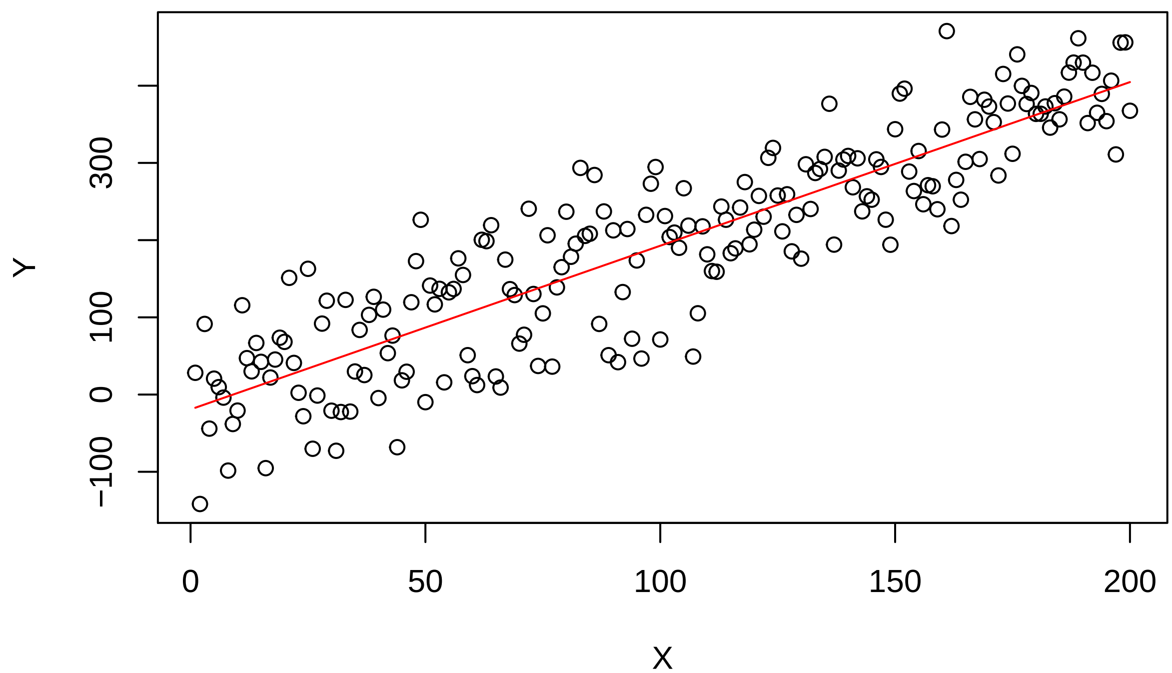
$$\hat{a} = \frac{\sum_{i=1}^{n} X_i Y_i - n\bar{X}_n \bar{Y}_n}{\sum_{i=1}^{n} X_i^2 - n\bar{X}_n^2}$$

et

$$\hat{b} = \bar{Y}_n - a\bar{X}_n$$

On retombe donc sur le résultat demandé en divisant le numérateur et le dénominateur du terme $\hat{a}$ par n.

```
n=200
X=c(1:n)
erreur=rnorm(n,0,64)
Y=-5+2*X+erreur
a=(mean(X*Y)-mean(X)*mean(Y))/(mean(X^2)-mean(X)^2)
b=mean(Y)-a*mean(X)
{
  plot(X,Y)
  lines(X,a*X+b,col='red')
}
```




```
R2=1-sum((Y-a*X-b)^2)/sum((Y-mean(Y))^2)
T=a/sqrt(1-R2)
a # a proche de ce qu'on attendait
```

```
## [1] 2.119441
```

```
b # b pas tellement
```

```
## [1] -19.20031
```

```
R2
```

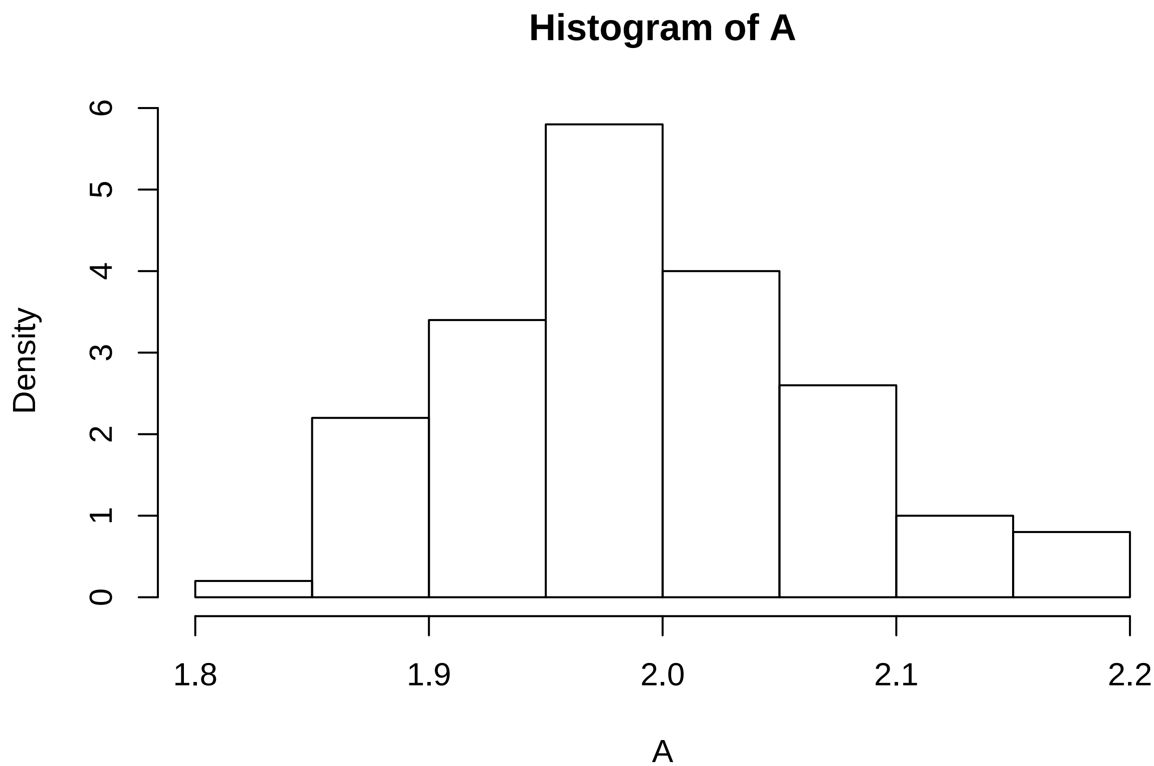
```
## [1] 0.7861302
```

```
T
```

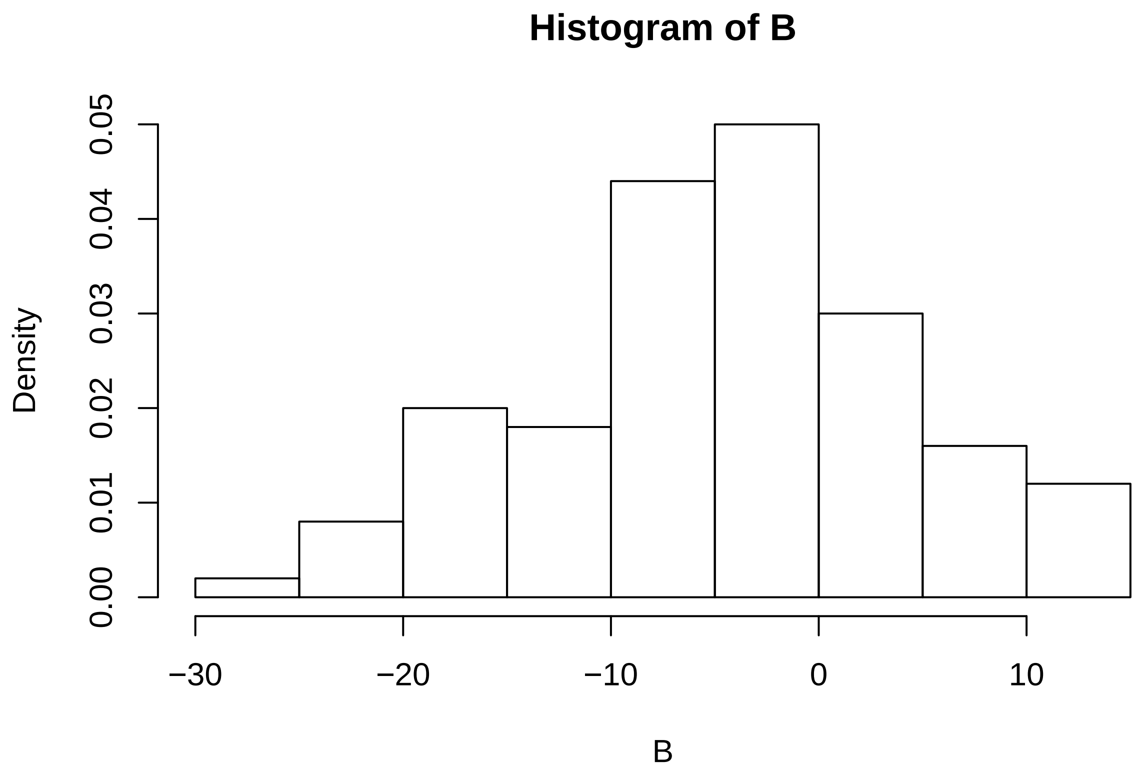
```
## [1] 4.582965
```

On a une acceptation de notre test d'hypothèse de Student. La modélisation par une droite est donc légitime. C'est assez logique car notre modèle sous-jacent est effectivement une droite. On retombe (entre guillemets) sur nos pattes.
On passe au code suivant.

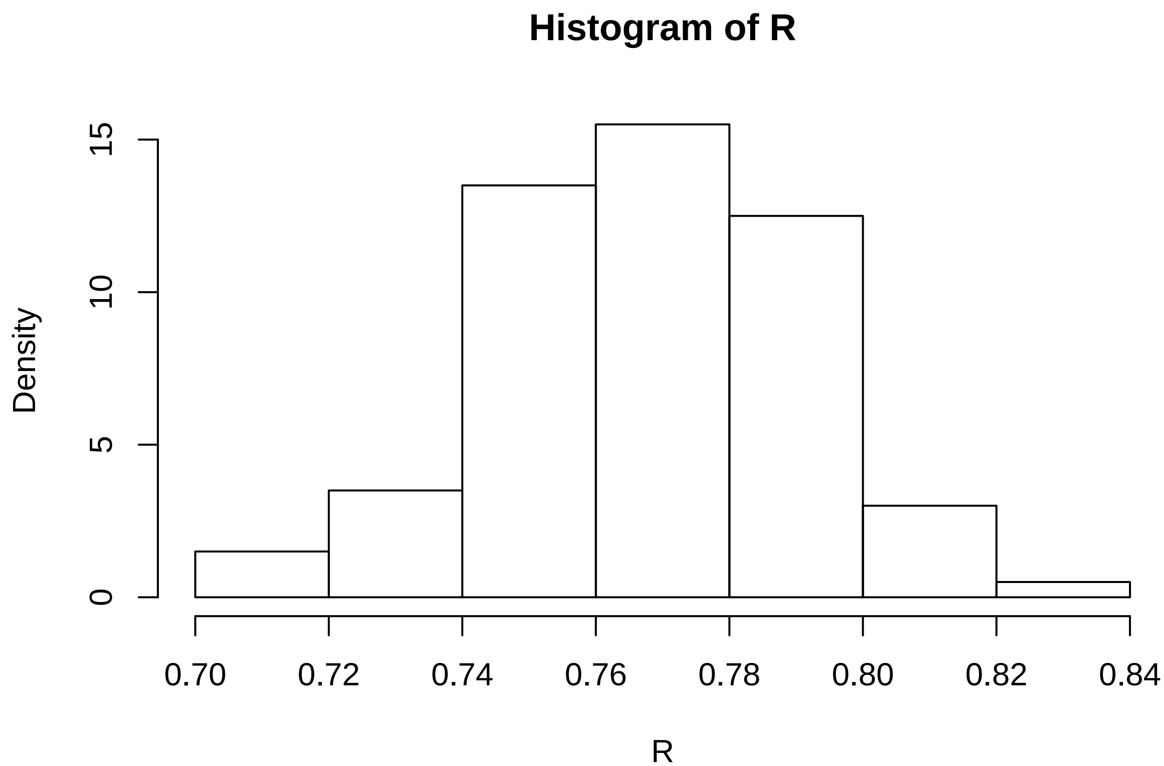
```
A <- c()
B <- c()
R <- c()
TT <- c()
for(i in 1:100)
{
  n=200
  X=c(1:n)
  erreur=rnorm(n,0,64)
  Y=-5+2*X+erreur
  a=(mean(X*Y)-mean(X)*mean(Y))/(mean(X^2)-mean(X)^2)
  b=mean(Y)-a*mean(X)
  R2=1-sum((Y-a*X-b)^2)/sum((Y-mean(Y))^2)
  T=a/sqrt(1-R2)
  A <- c(A,a)
  B <- c(B,b)
  R <- c(R,R2)
  TT <- c(TT,T)
}
hist(A,freq = FALSE)
```



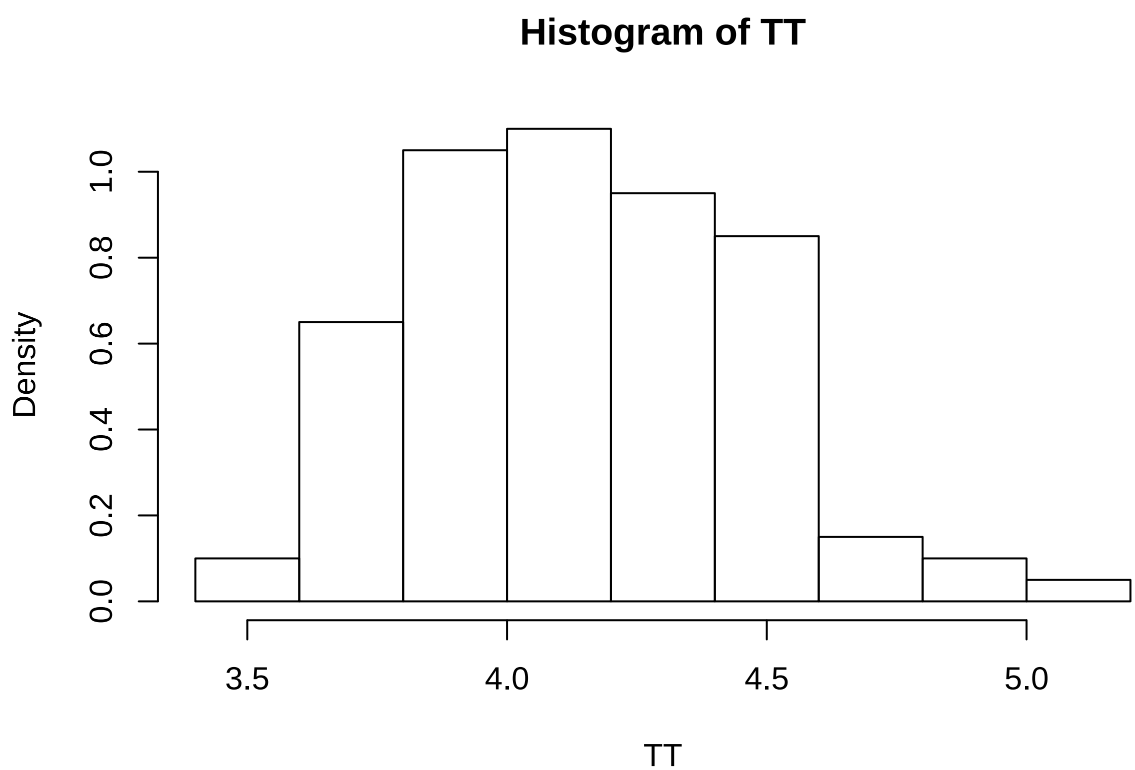

```r
hist(B,freq = FALSE)
```




```r
hist(R,freq = FALSE)
```



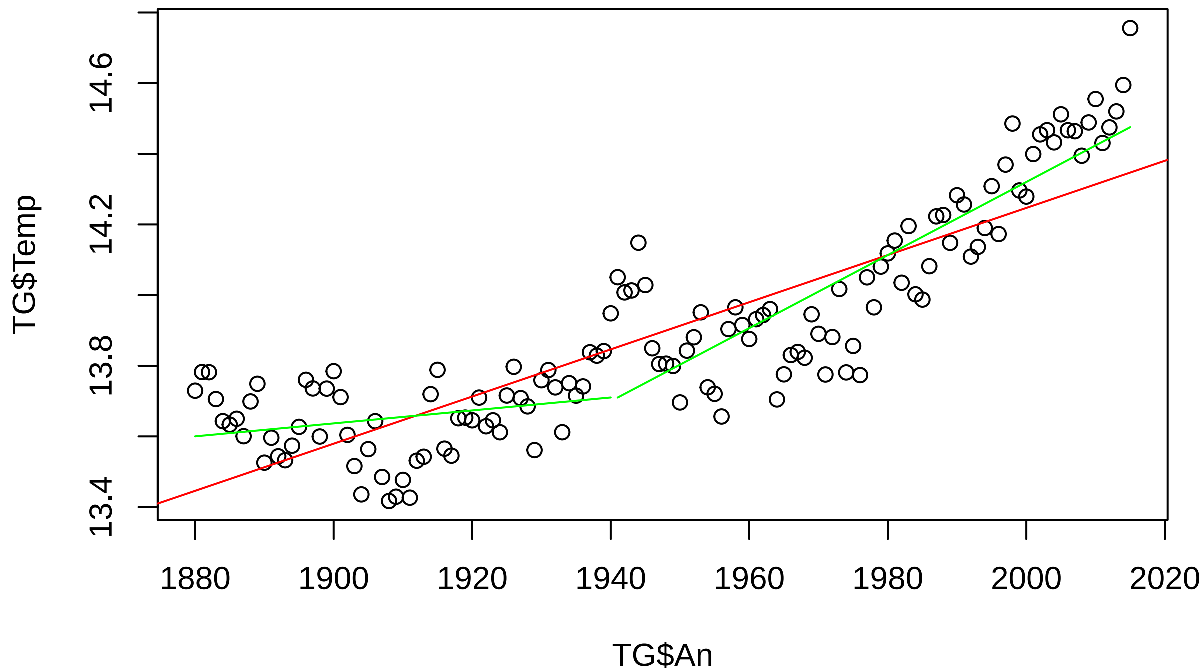

```
hist(TT,freq = FALSE)
```




```
A[length(A)]*(n+20)+B[length(B)] # prediciton avec les paramètres du dernier modèle simulé précédemment
```

```
## [1] 428.3973
```

```
TG=read.table('TGlobe1.txt',header=TRUE,sep = "\t")
#TG; names(TG); TG$An; TG$Temp
An=as.numeric(TG$An); Temp=as.numeric(TG$Temp)
model <- lm(TG$Temp~TG$An)
model2 <- lm(TG$Temp[1:61]~TG$An[1:61])
model2bis <- lm(TG$Temp[62:length(TG$Temp)]~TG$An[62:length(TG$Temp)])
{
  plot(TG$An,TG$Temp)
  abline(a = model$coefficients[1],b = model$coefficients[2],col = "red")
  lines(x = 1880:1940,model2$coefficients[2]*1880:1940+model2$coefficients[1],col = "green")
  lines(x = 1941:2015,model2bis$coefficients[2]*1941:2015+model2bis$coefficients[1],col = "green")
}
```




```
summary(model)
```

```
##
## Call:
## lm(formula = TG$Temp ~ TG$An)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.31309 -0.09756 -0.01870  0.11403  0.40892
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.8985220  0.6536540   1.375    0.172
## TG$An       0.0066741  0.0003356  19.889   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1536 on 134 degrees of freedom
## Multiple R-squared:  0.747,  Adjusted R-squared:  0.7451
## F-statistic: 395.6 on 1 and 134 DF,  p-value: < 2.2e-16
```

```
summary(model2)
```

```
##
## Call:
## lm(formula = TG$Temp[1:61] ~ TG$An[1:61])
##
```

```
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.234348 -0.082045 -0.000603  0.094165  0.238157
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 1.015e+01  1.562e+00   6.500 1.89e-08 ***
## TG$An[1:61] 1.834e-03  8.177e-04   2.243   0.0287 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1124 on 59 degrees of freedom
## Multiple R-squared:  0.07858,    Adjusted R-squared:  0.06296
## F-statistic: 5.031 on 1 and 59 DF,  p-value: 0.02866
```

```r
summary(model2bis)
```

```
##
## Call:
## lm(formula = TG$Temp[62:length(TG$Temp)] ~ TG$An[62:length(TG$Temp)])
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.29841 -0.10643  0.01639  0.07108  0.40735
##
## Coefficients:
##                              Estimate Std. Error t value Pr(>|t|)
## (Intercept)               -6.3651124  1.5083011   -4.22 6.93e-05 ***
## TG$An[62:length(TG$Temp)]  0.0103426  0.0007625   13.56  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.143 on 73 degrees of freedom
## Multiple R-squared:  0.7159, Adjusted R-squared:  0.712
## F-statistic:   184 on 1 and 73 DF,  p-value: < 2.2e-16
```

```r
model$coefficients[1]+model$coefficients[2]*2100 # prédiction modele 1
```

```
## (Intercept)
##    14.91418
```

```r
model2bis$coefficients[1]+model2bis$coefficients[2]*2100# prédiction modele 2 ! Ay caramba !!!!
```

```
## (Intercept)
##     15.3544
```