TPSTAT_CHEN_Zeyu_GROUP_2.1

Zeyu

2018/4/27

<–1–>

<–1.a–>

Rapplle la définition de α : Erreur de première espece. Il correspond la probabilité de rejeter à tort l'hypothèse nulle,i.e.

$\alpha = P(choisir\ H_1|H_0\ est\ vraie)$

<–1.b–>

Comme la variance est inconnue, donc , d'après le cours, on a statistique de test :

$$\Lambda_n = \frac{\sqrt{n}(\bar{X_n} - \mu_0)}{S_n}$$

Donc , la forme de la zone de rejet W pour 5% est :

$$W = \left\{(x_1, ..., x_n); \frac{\sqrt{n}(\bar{X_n}-\mu_0)}{S_n} > K_{0.05}\right\}$$

<–1.c–>

```r
Sn <- rnorm(20,mean=1,sd = sqrt(2))
# Sn est un échantillon, a est erreur de première espèce
# la valeur retourné soit mu0 soit mu1
decision<- function(Sn,a,mu0,mu1)
{
   n <- length(Sn)
   mu_emprique <- sum(Sn)/n
   sn2 <- 0
   for(i in 1:20)
   {
     sn2 <- sn2 + ((Sn[i]-mu_emprique)^2)
   }
   sn2 <- sn2 /(n-1)
   borne <- mu0 + (sqrt(sn2)*qt(1-a,n-1)/sqrt(n))
   if(mu_emprique > borne)
     return(mu1)
   else
     return(mu0)
}
decision(Sn,0.05,1,1.5)
```

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

<–2–>

<–2.a–>

```r
# Simuler N = 100 fois échantillons
# N est la fois d'échantillons
# f est la fonction qu'on va simuler
# n taille d'échantillon,sd est sigma
# Dans R, pour manipuler le array, faut ecrire s[i,j] au lieu de s[i][j]
sim.fun <-function (N,n,mu,sd)
  {
    sample <- array(1:2000,dim=c(100,20))
    for (i in 1:N) {
        s <- rnorm(n,mu,sd)
        for(j in 1 :20)
        {
          sample[i,j]<-s[j]
        }
     }
    return(sample)
}
array <-1:100
S_100 <- array(1:2000,dim=c(100,20))
S_100 <- sim.fun(100,20,1,sqrt(2))

#appelle 100 fois decision en passant alpha
fun_decison <- function(S_100,alpha)
{
array_decison <-1:100
count <- 0
for ( i in 1:100)
{
   Sn <- 1:20
   for(j in 1 :20)
        {
          Sn[j]<-S_100[i,j]
        }
   array_decison[i] <- decision(Sn,alpha,1,1.5)

   if (array_decison[i]==1)
     count=count+1
}
   if(count == 100)
   array_decison[100]=array_decison[100]+0.01;
   cat("le nombre de 1 avec alpha",alpha,"est",count,"\n")
   return(array_decison)
}
array_decison <- fun_decison(S_100,0.05)
```

```
## le nombre de 1 avec alpha 0.05 est 95
```

```r
array_decison
```

```
##   [1] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0
```

```
## [18] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
## [35] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
## [52] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
## [69] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.5 1.0 1.0 1.0 1.0
## [86] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0
```

On peut remarquer que le pourcentage de 1 (i.e la probabilité de choisir H0)est plus très proche que $1-\alpha$. En fait, $\delta(\mathcal{S}_n, \alpha, \mu_0, \mu_1)$ suis une loi bernouille avec paramètre $1-\alpha$

<–2.b–>

plus α est petit, plus K_α est grand, donc ,la zone est plus petite. i.e, la probabilité de rejetter H_0 à tort est plus petite.

<–2.c–>

```r
array_decison_0.2 <- fun_decison(S_100,0.2)
```

```
## le nombre de 1 avec alpha 0.2 est 82
```

```r
array_decison_0.2
```

```
##  [1] 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.5 1.0
## [18] 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.5 1.0 1.0 1.0
## [35] 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0
## [52] 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
## [69] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.5 1.5 1.0 1.0 1.0
## [86] 1.5 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.5 1.0 1.0 1.5
```

```r
array_decison_0.1 <- fun_decison(S_100,0.1)
```

```
## le nombre de 1 avec alpha 0.1 est 90
```

```r
array_decison_0.1
```

```
##  [1] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0
## [18] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0
## [35] 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
## [52] 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
## [69] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.5 1.0 1.0 1.0 1.0
## [86] 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.5
```

```r
array_decison_0.05 <- fun_decison(S_100,0.05)
```

```
## le nombre de 1 avec alpha 0.05 est 95
```

```r
array_decison_0.05
```

```
##  [1] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0
## [18] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
## [35] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
## [52] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
## [69] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.5 1.0 1.0 1.0 1.0
## [86] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0
```

```r
array_decison_0.01 <- fun_decison(S_100,0.01)
```

```
## le nombre de 1 avec alpha 0.01 est 98
```

```r
array_decison_0.01
```

```
##   [1] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0
##  [18] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [35] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [52] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [69] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [86] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
```

<–3–>

<–3.a–>

```r
S1_100 <- array(1:2000,dim=c(100,20))
S1_100 <- sim.fun(100,20,1.5,sqrt(2))
fun_decison(S1_100,0.05)
```

```
## le nombre de 1 avec alpha 0.05 est 52
```

```
##   [1] 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.5 1.5 1.5 1.5 1.5 1.5 1.5 1.0 1.5 1.5
##  [18] 1.0 1.5 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.5 1.0 1.5 1.0 1.0 1.0 1.0 1.0
##  [35] 1.5 1.5 1.5 1.5 1.0 1.0 1.5 1.5 1.0 1.5 1.5 1.0 1.0 1.5 1.0 1.0 1.0
##  [52] 1.0 1.5 1.0 1.5 1.5 1.5 1.0 1.0 1.5 1.5 1.5 1.5 1.0 1.0 1.0 1.0 1.5
##  [69] 1.5 1.5 1.5 1.0 1.0 1.5 1.0 1.0 1.5 1.0 1.5 1.0 1.5 1.0 1.5 1.0 1.5
##  [86] 1.0 1.5 1.5 1.0 1.0 1.5 1.0 1.0 1.0 1.5 1.0 1.0 1.5 1.5 1.5
```

On peut observer que dans le cas mu = 1.5 , le pourcentage de 1.5 est plus $1-\alpha$, il plutôt égale à β qui est defini par la puissance d'un test

<–3.b–>

Rapplle la définition de β : La puissance d'un test. Il correspond la probabilité de rejeter H_0 losque H_1 est vraie, i.e.

$\beta = P(choisir\ H_1|H_1\ est\ vraie)$

On sais que

$$\Lambda_n = \frac{\sqrt{n}(\bar{X}_n - \mu_0)}{S_n}$$

donc ,on a :

$$\beta = P_{H_1}(W) = P_{H_1}(\Lambda_n > K_\alpha)$$

$$= P_{H_1}(\frac{\sqrt{n}(\bar{x}_n - \mu_0)}{s_n} > F^{-1}_{T_{n-1}}(1-\alpha))$$

$$= P_{H_1}(\frac{\sqrt{n}(\bar{x}_n - \mu_1)}{s_n} > \frac{\sqrt{n}(\mu_0 - \mu_1)}{s_n} + F^{-1}_{T_{n-1}}(1-\alpha))$$

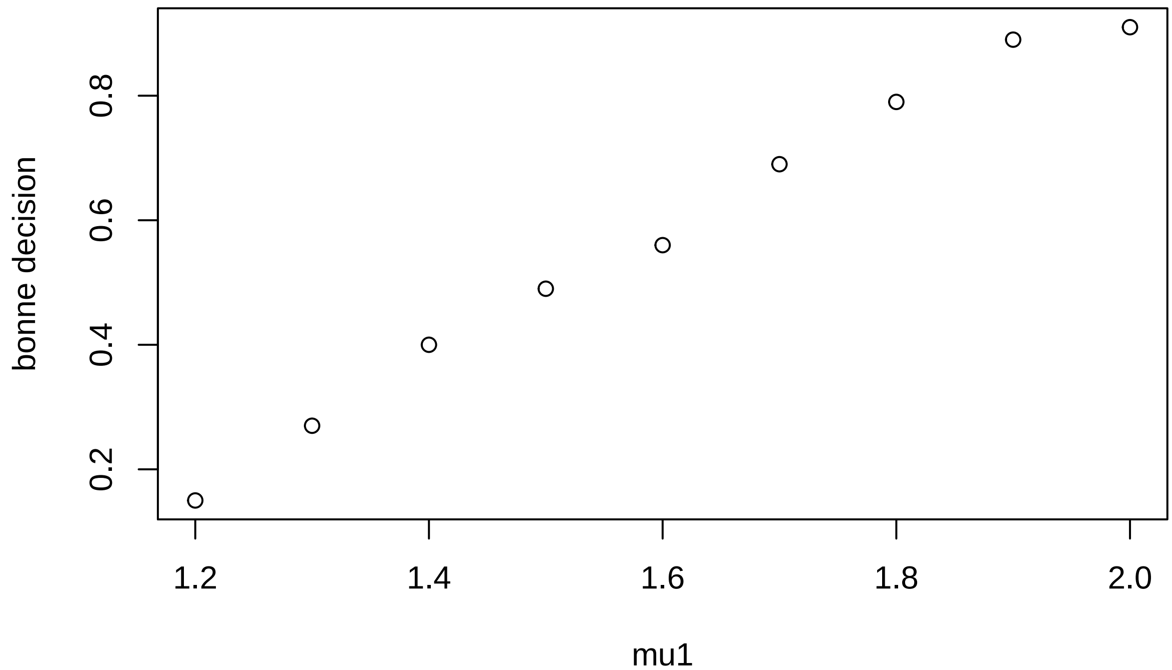
$$= 1 - F_{T_{n-1}}(\frac{\sqrt{n}(\mu_0 - \mu_1)}{s_n} + F^{-1}_{T_{n-1}}(1-\alpha))$$

<–3.c–>

```
#appelle 100 fois decision en passant mu1
fun_decison_mu1 <- function(S_100,mu1)
{
array_decison <-1:100
count <- 0
for ( i in 1:100)
{
   Sn <- 1:20
   for(j in 1 :20)
        {
          Sn[j]<-S_100[i,j]
        }
   array_decison[i] <- decision(Sn,0.05,1,mu1)
   if (array_decison[i]==mu1)
     count=count+1
}
   return (count/100)
}
S1_100 <- array(1:2000,dim=c(100,20))

v <- 1:9
mu1 <- 1.2
for(i in 1:9)
{
  S1_100 <- sim.fun(100,20,mu1,sqrt(2))
  v[i] <- fun_decison_mu1(S1_100,mu1)
  mu1<- mu1+0.1
}
x = seq(1.2,2.0,0.1)

plot(x,v,xlab="mu1",ylab="bonne decision")
```



###<–4–>

<–4.a–>

```
r= rnorm(20,1,sqrt(20))
t.test(r,mu=1)
```

```
##
##  One Sample t-test
##
## data:  r
## t = -1.9645, df = 19, p-value = 0.06427
## alternative hypothesis: true mean is not equal to 1
## 95 percent confidence interval:
##  -3.543963  1.143974
## sample estimates:
## mean of x
## -1.199994
```

t est la valeur des statistiques du t-test

on peut l'obtenir par :

$$t = \frac{\sqrt{n}(\bar{X_n} - \mu_0)}{S_n}$$

Et df est le degree de liberté , il égale à n-1

<–4.b–>

```
la zone de rejet $W$ est $$W = {\begin{Bmatrix} (x_1,...,x_n) ;  \frac{\sqrt{n}(\bar{X_n}-\mu_0)}{S_n} 
```

et on sais que

$$t = \frac{\sqrt{n}(\bar{X_n} - \mu_0)}{S_n}$$

et

$$K_\alpha = F^{-1}_{T_{n-1}}(1-\alpha)$$

Donc, on a

$$t > F^{-1}_{T_{n-1}}(1-\alpha) => F(t) > 1-\alpha => \alpha > 1 - F(t) = p$$

Finalement , on a trouvé que quand $p < \alpha$, on rejette l'hypothèse H_0

<–4.c–>

```
array_decison_0.2 <- fun_decison(S_100,0.2)
```

```
## le nombre de 1 avec alpha 0.2 est 82
```

```
t.test(array_decison_0.2,mu=1)
```

```
##
##  One Sample t-test
##
## data:  array_decison_0.2
## t = 4.6617, df = 99, p-value = 9.804e-06
## alternative hypothesis: true mean is not equal to 1
## 95 percent confidence interval:
##  1.051692 1.128308
## sample estimates:
## mean of x
##      1.09
```

```
array_decison_0.1 <- fun_decison(S_100,0.1)
```

```
## le nombre de 1 avec alpha 0.1 est 90
```

```
t.test(array_decison_0.1,mu=1)
```

```
##
##  One Sample t-test
##
## data:  array_decison_0.1
## t = 3.3166, df = 99, p-value = 0.001275
## alternative hypothesis: true mean is not equal to 1
## 95 percent confidence interval:
##  1.020087 1.079913
## sample estimates:
## mean of x
##      1.05
```

```
array_decison_0.05 <- fun_decison(S_100,0.05)
```

```
## le nombre de 1 avec alpha 0.05 est 95
```

```r
t.test(array_decison_0.05,mu=1)
```

```
##
##  One Sample t-test
##
## data:  array_decison_0.05
## t = 2.2827, df = 99, p-value = 0.02459
## alternative hypothesis: true mean is not equal to 1
## 95 percent confidence interval:
##  1.003269 1.046731
## sample estimates:
## mean of x
##     1.025
```

```r
array_decison_0.01 <- fun_decison(S_100,0.01)
```

```
## le nombre de 1 avec alpha 0.01 est 98
```

```r
t.test(array_decison_0.01,mu=1)
```

```
##
##  One Sample t-test
##
## data:  array_decison_0.01
## t = 1.4214, df = 99, p-value = 0.1583
## alternative hypothesis: true mean is not equal to 1
## 95 percent confidence interval:
##  0.9960405 1.0239595
## sample estimates:
## mean of x
##      1.01
```

On peut remarquer que plus le α est petit, plus p-value est grand. Surtout, quand $\alpha = 0.01$, on a $p - value > \alpha$, c'est à dire que H_0 est vrai et on le choisi

<–4.d–>

```r
array_decison_0.2 <- fun_decison(S_100,0.2)
```

```
## le nombre de 1 avec alpha 0.2 est 82
```

```r
array_decison_0.2
```

```
##   [1] 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.5 1.0
##  [18] 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.5 1.0 1.0 1.0
##  [35] 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0
##  [52] 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [69] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.5 1.5 1.0 1.0 1.0
##  [86] 1.5 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.5 1.0 1.0 1.5
```

```r
t.test(array_decison_0.2,mu=1)
```

```
##
##  One Sample t-test
##
## data:  array_decison_0.2
```

```
## t = 4.6617, df = 99, p-value = 9.804e-06
## alternative hypothesis: true mean is not equal to 1
## 95 percent confidence interval:
##  1.051692 1.128308
## sample estimates:
## mean of x
##      1.09
```

```r
array_decison_0.1 <- fun_decison(S_100,0.1)
```

```
## le nombre de 1 avec alpha 0.1 est 90
```

```r
array_decison_0.1
```

```
##   [1] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0
##  [18] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [35] 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [52] 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [69] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.5 1.0 1.0 1.0 1.0
##  [86] 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.5
```

```r
t.test(array_decison_0.1,mu=1)
```

```
##
##  One Sample t-test
##
## data:  array_decison_0.1
## t = 3.3166, df = 99, p-value = 0.001275
## alternative hypothesis: true mean is not equal to 1
## 95 percent confidence interval:
##  1.020087 1.079913
## sample estimates:
## mean of x
##      1.05
```

```r
array_decison_0.05 <- fun_decison(S_100,0.05)
```

```
## le nombre de 1 avec alpha 0.05 est 95
```

```r
array_decison_0.05
```

```
##   [1] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0
##  [18] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [35] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [52] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [69] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.5 1.0 1.0 1.0 1.0
##  [86] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0
```

```r
t.test(array_decison_0.05,mu=1)
```

```
##
##  One Sample t-test
##
## data:  array_decison_0.05
## t = 2.2827, df = 99, p-value = 0.02459
## alternative hypothesis: true mean is not equal to 1
## 95 percent confidence interval:
##  1.003269 1.046731
```

```
## sample estimates:
## mean of x
##     1.025
```

```r
array_decison_0.01 <- fun_decison(S_100,0.01)
```

```
## le nombre de 1 avec alpha 0.01 est 98
```

```r
array_decison_0.01
```

```
##   [1] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0
##  [18] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [35] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [52] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [69] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
##  [86] 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0
```

```r
t.test(array_decison_0.01,mu=1)
```

```
##
##  One Sample t-test
##
## data:  array_decison_0.01
## t = 1.4214, df = 99, p-value = 0.1583
## alternative hypothesis: true mean is not equal to 1
## 95 percent confidence interval:
##  0.9960405 1.0239595
## sample estimates:
## mean of x
##      1.01
```

On peut remarquer que quand on fait varier α, le pourcentage de 1 dans l'intervalle de confiance est toujour égal à $1 - \alpha$. Ce qui est normal.

on rappelle le cours :

$C(X1, ..., Xn)$ **est un ensemble (typiquement un intervalle de R) qui est aléatoire (car sa définition dépend des observations), et telle que quelque soit θ,il y ait une probabilité au moins supérieur à 1-α que le vrai paremetre θ soit dedans.**

\

Le TP est une étude des tests statistiques. **1) Test de Student**\ Simuler un échantillon i.i.d $\mathcal{S}_n = (x_1, \ldots, x_n)$ de taille $n = 20$, et dont la loi commune est une loi normale $\mathcal{N}(\mu, \sigma^2)$ avec $\mu = 1$ et $\sigma^2 = 2$ (on rappelle que la fonction R pour simuler *rnorm*). \

1. Nous voulons tester si la moyenne de l'échantillon μ est égale à $\mu_0 = 1$, ou plutôt égale ? $\mu_1 = 1.5$. On suppose que la variance σ^2 est inconnue. Pour répondre ? cette question, on va faire un test statistique avec un niveau de significativité $\alpha = 5\%$ (appelé encore risque de 1ère espece).
   (a) Les hypothèses du test sont $H_0 : \mu = \mu_0$ et $H_1 : \mu = \mu_1$. Rappeler la définition de α et à quoi il correspond.
   (b) Donner la forme de la zone de rejet W, pour $\alpha = 5\%$ (on pourra utiliser le lemme de Neyman-Pearson, vu en cours).
   (c) Programmer la règle de décision associée $\delta(\mathcal{S}_n, \alpha, \mu_0, \mu_1)$ (écrire une fonction R paramètre par les moyennes, α, et $\mathcal{S}_n$).

2. Simuler $N = 100$?chantillons $\mathcal{S}_n^1, \dots, \mathcal{S}_n^N$ (toujours tel que $\mathcal{N}(\mu, \sigma^2)$, avec $\mu = 1$ et $\sigma^2 = 2$).

   (a) On rappellera la loi de la variable al?atoire $\delta(\mathcal{S}_n, \alpha, \mu_0, \mu_1)$. Appliquer la r?gle de d?cision du test de Student sur $S_n^i, i = 1, \dots, 100$. Qu'observez vous? .

   (b) Faire varier $\alpha = 0.2, 0.1, 0.05, 0.01$: comment la zone de rejet est-elle modifi?e ?

   (c) Pour $\alpha = 0.2, 0.1, 0.05, 0.01$, appliquer la r?gle de d?cision $\delta(\mathcal{S}_n^i, \alpha, \mu_0, \mu_1)$, $i = 1, \dots, N$.

3. On va simuler $N = 100$?chantillons $\mathcal{S}_n^{'1}, \dots, \mathcal{S}_n^{'N}$, mais qui suivent maintenant une loi $\mathcal{N}(\mu, \sigma^2)$, avec $\mu = 1.5$ et $\sigma^2 = 2$.

   (a) On rappellera la loi de la variable al?atoire $\delta(\mathcal{S}_n^{'i}, \alpha, \mu_0, \mu_1)$. Appliquer la r?gle de d?cision du test de Student sur $\mathcal{S}_n^{'i}, i = 1, \dots, 100$. Qu'observez vous?

   (b) Rappeler la d?finition et calculer th?oriquement la puissance du test β, en fonction de α, μ_0, μ_1.

   (c) On fixe $\alpha = 0.05$, et on fait varier l'hypoth?se alternative $H_1 : \mu = \mu_1$. Simuler $N = 100$?chantillons $\mathcal{S}_n^{'1}, \dots, \mathcal{S}_n^{'N}$ en faisant varier la moyenne $\mu = \mu_1 \in \{1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.0\}$ et appliquer la r?gle de d?cision $\delta(\mathcal{S}_n^{'i}, \alpha, \mu_0, \mu_1)$, $i = 1, \dots, N$. Tracer en fonction de μ_1 le pourcentage de bonne d?cision et comparer avec les r?sultats de la question pr?c?dente.

4. On va utiliser la fonction R *t.test* qui permet de faire le test d'une hypoth?se simple $H_0 : \mu = \mu_0$, contre une hypoth?se multiple (ou composite) $H_1 : \mu > \mu_0$ (ou $\mu \neq \mu_0$).

   (a) Pour un ?chantillon $\mathcal{S}_n = (x_1, \dots, x_n)$ de la question 2, utiliser la fonction *t.test* pour faire le test vu ci-dessus. On lira attentivement l'aide de la fonction pour comprendre les inputs et outputs : à quoi correspond la valeur "t". A quoi correspond df ?

   (b) Si on note $x \mapsto F_{n-1}^T(x)$, la fonction de répartition d'une loi de Student à $n-1$ degrés de libertés, alors la p-value donnée par la fonction *t.test* est égale ? $p - value = 1 - F_{n-1}^T(t)$, où t est la valeur donnée pr?c?demment. En se rappelant la forme de la zone de rejet W, et le caractère monotone d'une fonction de répartition, expliquer comment la p-value permet de prendre une décision au niveau $\alpha = 0.05$.

   (c) Reprendre les N échantillons de la question 2, et utiliser la p-value pour ?tudier l'impact de α variant dans $\alpha = 0.2, 0.1, 0.05, 0.01$.

   (d) La fonction *t.test* permet de calculer l'intervalle de confiance au niveau $1 - \alpha$. Rappeler comment l'intervalle de confiance. Sur les N ?chantillons de la question 2, dans combien de cas 1 est dans l'intervalle de confiance. Est ce normal ?

\end{document}