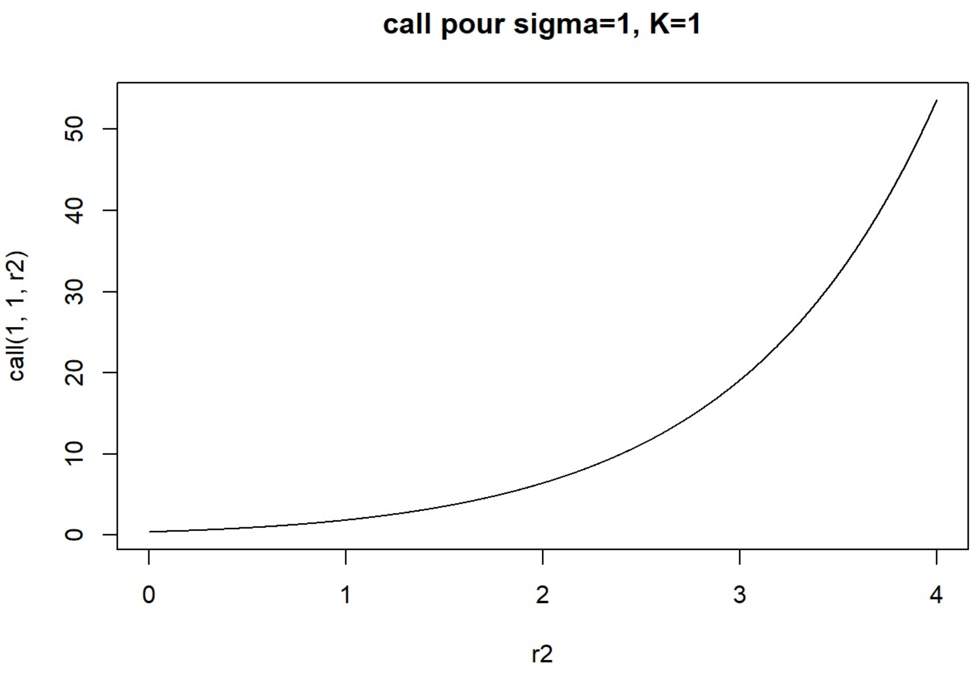
TP6- Dellouve et Gazzo

Exercice 2

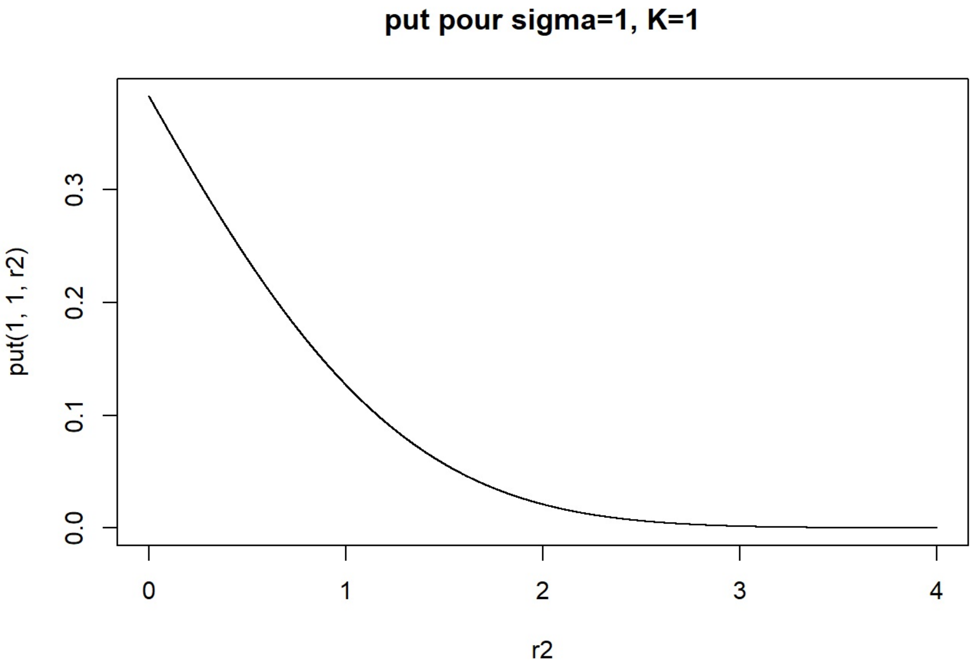
```
#Question 1

call<-function(sigma,K,r=0,S=1) # fonction call
{
    C=S*exp(r)*(1- pnorm(log(K/S)/sigma-r/sigma-sigma/2))-K*(1- pnorm(log(K/S)/sigma-r/sigma+sigma/2))
    return(C)
}

put<-function(sigma,K,r=0,S=1) # fonction put
{
  P=K*(pnorm(log(K/S)/sigma-r/sigma+sigma/2))-S*exp(r)*pnorm(log(K/S)/sigma-r/sigma-sigma/2)
  return(P)
}

call(1,1)-put(1,1) # vérification, ca doit valoir 0
```

```
## [1] -5.551115e-17
```

```
r<-seq(0,1,0.0005) #taux d'intérêt instantanné
r2<-seq(0,4,0.0005) #taux d'intérêt instantanné

plot(r2,call(1,1,r2),type='l',main = "call pour sigma=1, K=1")
```

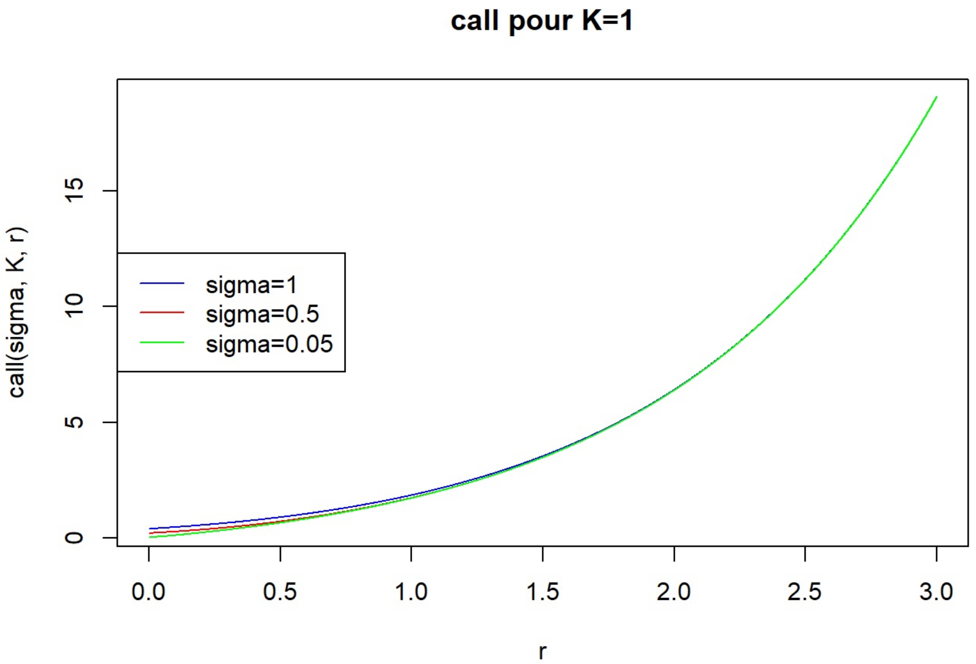



```
plot(r2,put(1,1,r2),type='l',main = "put pour sigma=1, K=1")
```

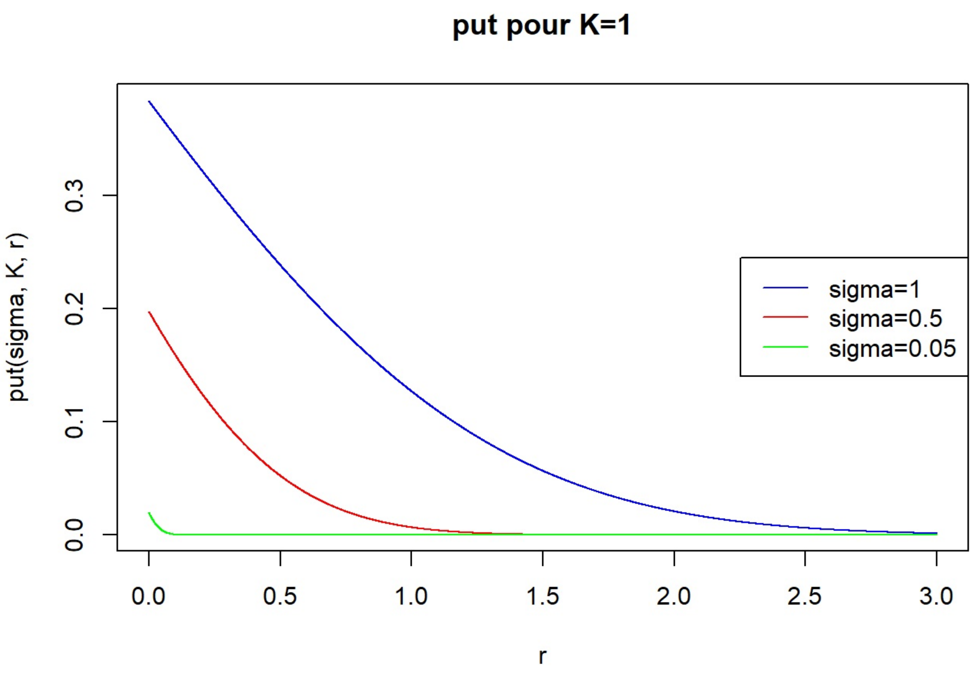


En traçant les graphiques pour K fixé et pour différentes valeurs de sigma, on observe que la fonction call est croissante alors que la fonction put est décroissante et converge vers 0. On trace ensuite les graphiques pour comparer les résultats obtenus pour K fixé et ensuite pour σ fixé.

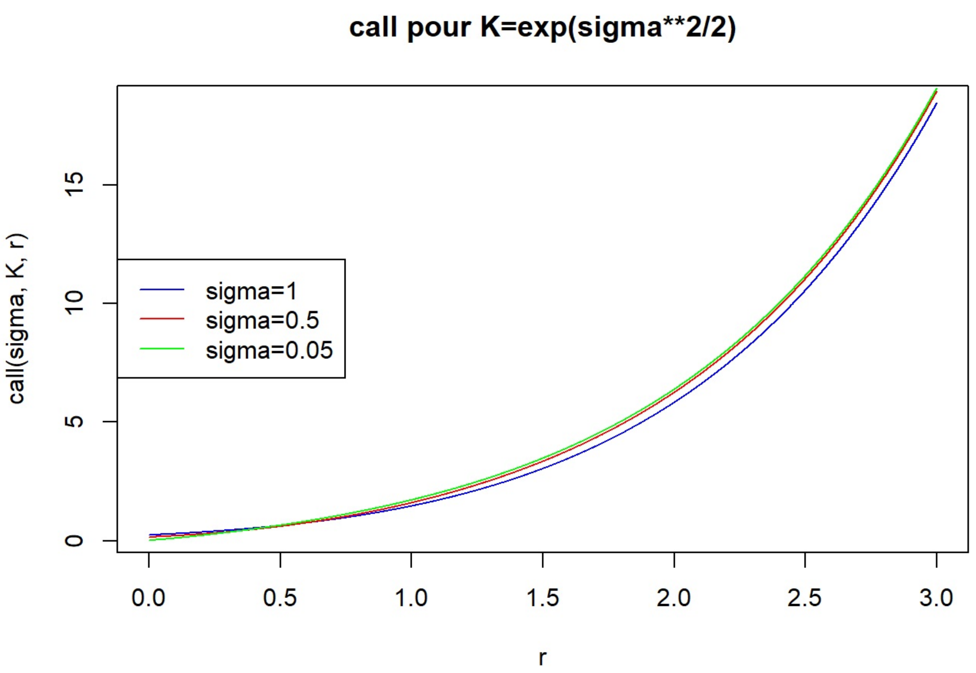
```
r<-seq(0,3,0.0005) #taux d'intérêt instantanné

#cas K=1, pour différents sigma
K<-1
sigma<-1.0
plot(r,call(sigma,K,r),type='l',main = "call pour K=1",col='blue')
sigma<-0.5
lines(r,call(sigma,K,r),type='l',col='red')
sigma<-0.05
lines(r,call(sigma,K,r),type='l',col='green')
legend("left",c("sigma=1","sigma=0.5","sigma=0.05"),col=c('blue','red','green'),
       lwd=1,lty=1)
```



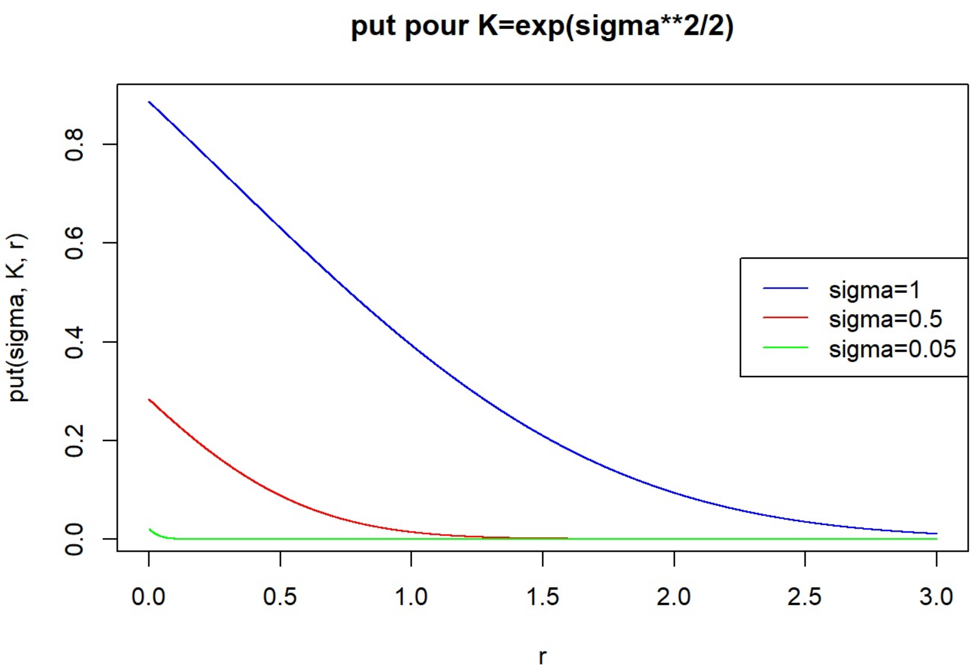
```
sigma<-1.0
plot(r,put(sigma,K,r),type='l',main = "put pour K=1",col='blue')
sigma<-0.5
lines(r,put(sigma,K,r),type='l',col='red')
sigma<-0.05
lines(r,put(sigma,K,r),type='l',col='green')
legend("right",c("sigma=1","sigma=0.5","sigma=0.05"),col=c('blue','red','green'),
       lwd=1,lty=1)
```



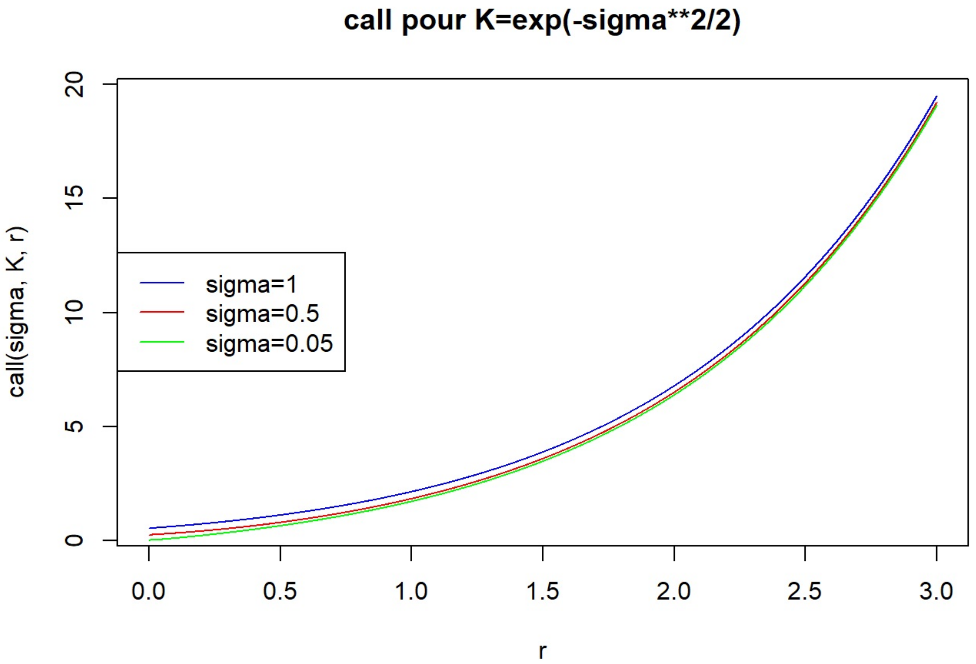

```
#cas K=exp(sigma**2/2), pour différents sigma
sigma<-1.0
K<-exp(sigma**2/2)
plot(r,call(sigma,K,r),type='l',main = "call pour K=exp(sigma**2/2)",col='blue')
sigma<-0.5
K<-exp(sigma**2/2)
lines(r,call(sigma,K,r),type='l',col='red')
sigma<-0.05
K<-exp(sigma**2/2)
lines(r,call(sigma,K,r),type='l',col='green')
legend("left",c("sigma=1","sigma=0.5","sigma=0.05"),col=c('blue','red','green'),
       lwd=1,lty=1)
```



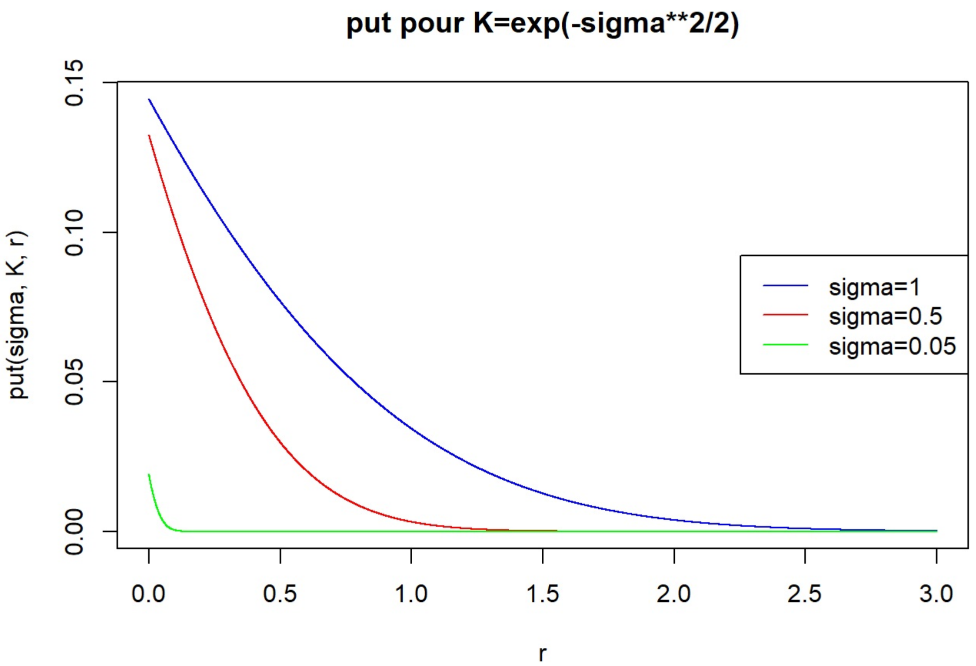
```
sigma<-1.0
K<-exp(sigma**2/2)
plot(r,put(sigma,K,r),type='l',main = "put pour K=exp(sigma**2/2)",col='blue')
sigma<-0.5
K<-exp(sigma**2/2)
lines(r,put(sigma,K,r),type='l',col='red')
sigma<-0.05
K<-exp(sigma**2/2)
lines(r,put(sigma,K,r),type='l',col='green')
legend("right",c("sigma=1","sigma=0.5","sigma=0.05"),col=c('blue','red','green'),
       lwd=1,lty=1)
```




```
#cas K=exp(-sigma**2/2), pour différents sigma
sigma<-1.0
K<-exp(-sigma**2/2)
plot(r,call(sigma,K,r),type='l',main = "call pour K=exp(-sigma**2/2)",col='blue')
sigma<-0.5
K<-exp(-sigma**2/2)
lines(r,call(sigma,K,r),type='l',col='red')
sigma<-0.05
K<-exp(-sigma**2/2)
lines(r,call(sigma,K,r),type='l',col='green')
legend("left",c("sigma=1","sigma=0.5","sigma=0.05"),col=c('blue','red','green'),
       lwd=1,lty=1)
```



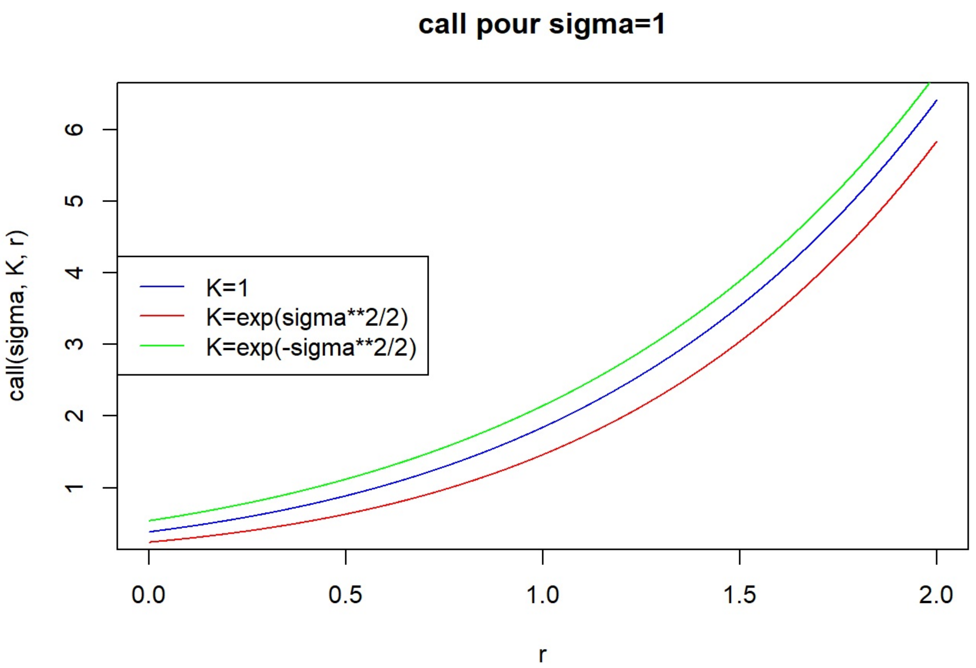

```
sigma<-1.0
K<-exp(-sigma**2/2)
plot(r,put(sigma,K,r),type='l',main = "put pour K=exp(-sigma**2/2)",col='blue')
sigma<-0.5
K<-exp(-sigma**2/2)
lines(r,put(sigma,K,r),type='l',col='red')
sigma<-0.05
K<-exp(-sigma**2/2)
lines(r,put(sigma,K,r),type='l',col='green')
legend("right",c("sigma=1","sigma=0.5","sigma=0.05"),col=c('blue','red','green'),
       lwd=1,lty=1)
```



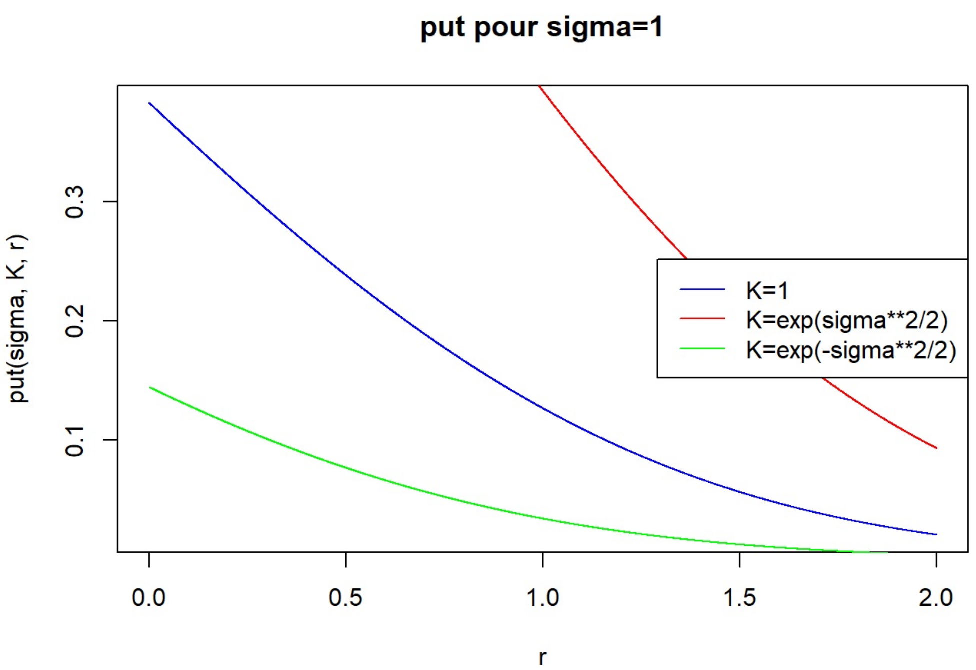
On voit que plus σ est grand, plus la fonction call commence avec une valeur élevée. La courbe croît plus fortement pour σ petit. On voit que la valeur de K influence le phénomène. Pour K petit, les courbes se croisent plus tard comparé au cas K grand.

Pour la fonction put, on observe que plus σ est petit, plus la courbe converge rapidement vers 0 et commence à une valeur faible en $x = 0$. D'autre part, K influence aussi le point de départ de la courbe à l'origine, au plus K est grand, au plus la valeur à l'origine sera grande.

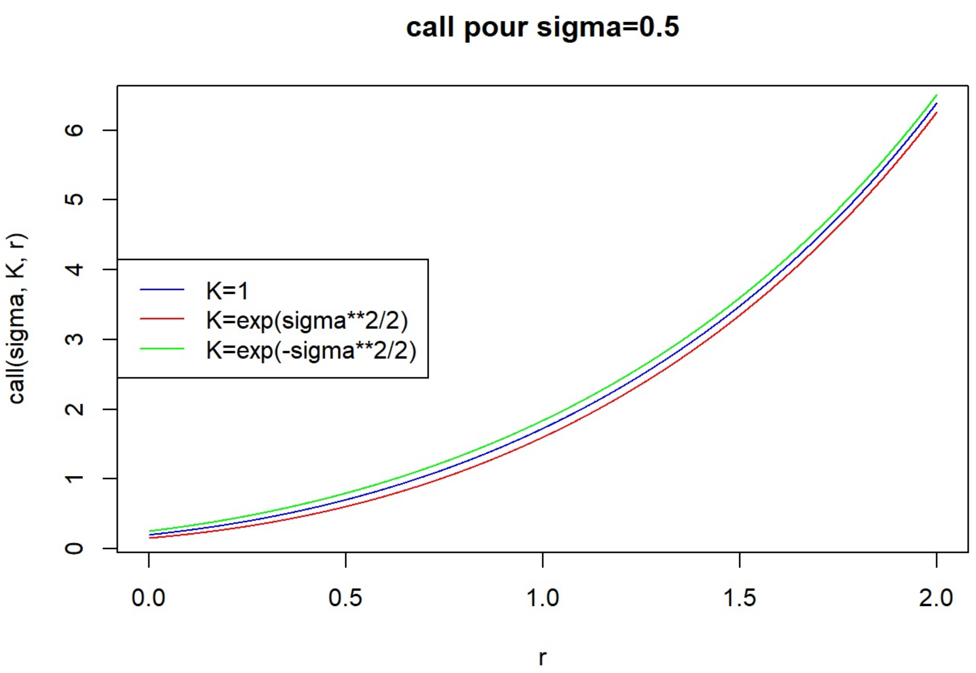
```
r<-seq(0,2,0.0005) #taux d'intérêt instantanné

#cas sigma=1, pour différents sigma
sigma<-1.0
K<-1
plot(r,call(sigma,K,r),type='l',main = "call pour sigma=1",col='blue')
K<-exp(sigma**2/2)
lines(r,call(sigma,K,r),type='l',col='red')
K<-exp(-sigma**2/2)
lines(r,call(sigma,K,r),type='l',col='green')
legend("left",c("K=1","K=exp(sigma**2/2)","K=exp(-sigma**2/2)"),
       col=c('blue','red','green'),lwd=1,lty=1)
```



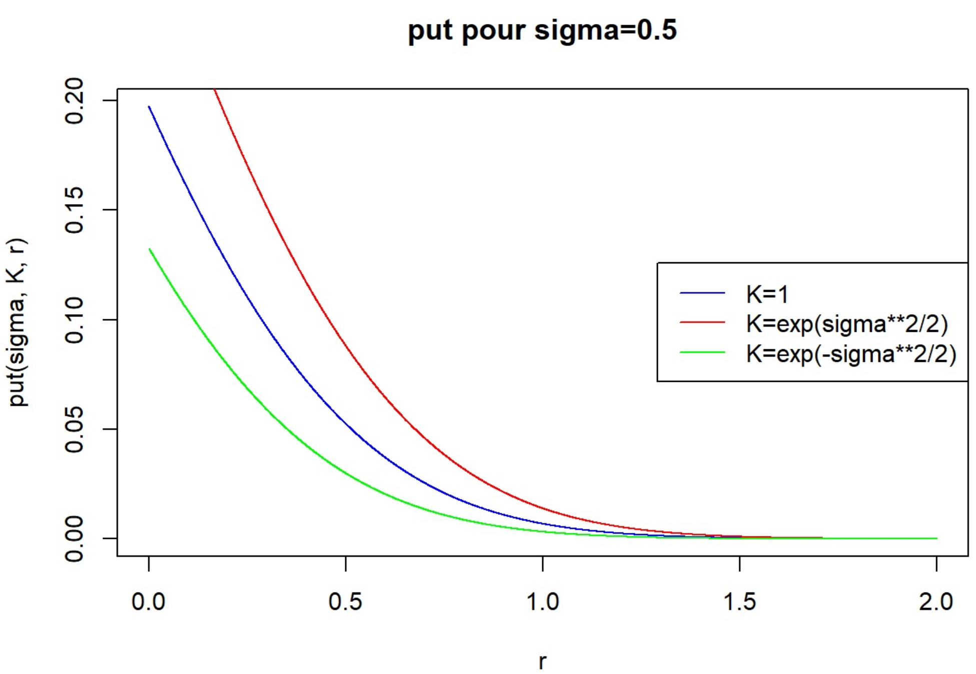

```
K<-1
plot(r,put(sigma,K,r),type='l',main = "put pour sigma=1",col='blue')
K<-exp(sigma**2/2)
lines(r,put(sigma,K,r),type='l',col='red')
K<-exp(-sigma**2/2)
lines(r,put(sigma,K,r),type='l',col='green')
legend("right",c("K=1","K=exp(sigma**2/2)","K=exp(-sigma**2/2)"),
       col=c('blue','red','green'),lwd=1,lty=1)
```



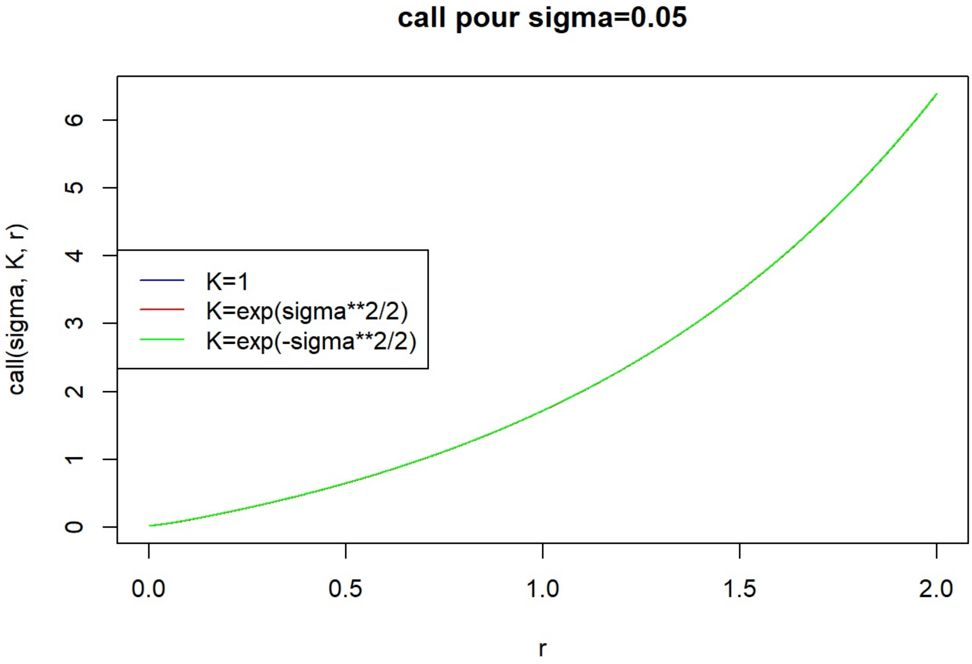
```
#cas sigma=0.5, pour différents sigma
sigma<-0.5
K<-1
plot(r,call(sigma,K,r),type='l',main = "call pour sigma=0.5",col='blue')
K<-exp(sigma**2/2)
lines(r,call(sigma,K,r),type='l',col='red')
K<-exp(-sigma**2/2)
lines(r,call(sigma,K,r),type='l',col='green')
legend("left",c("K=1","K=exp(sigma**2/2)","K=exp(-sigma**2/2)"),
       col=c('blue','red','green'),lwd=1,lty=1)
```



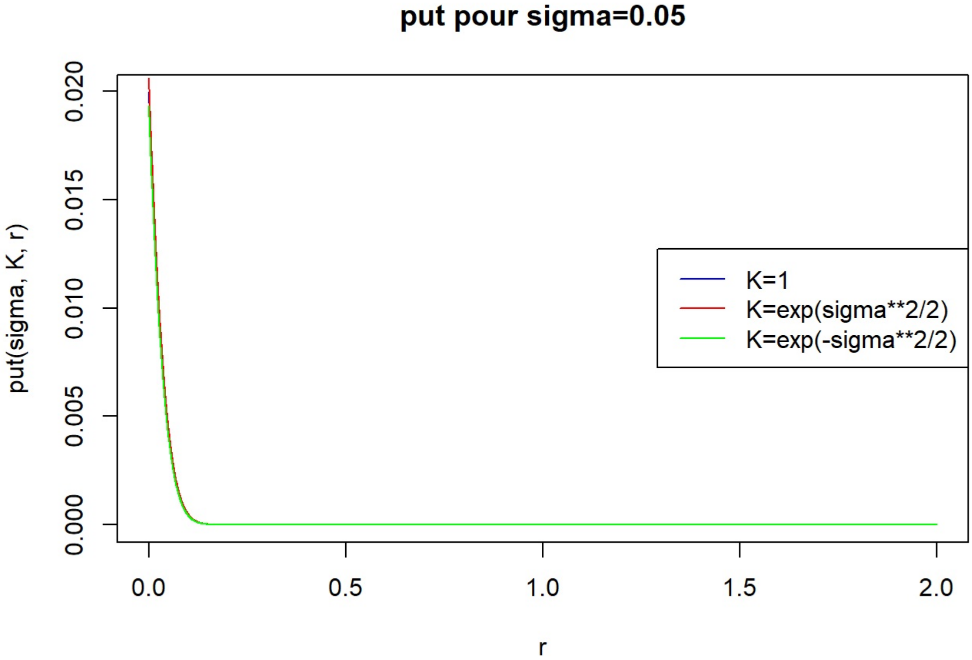

```
K<-1
plot(r,put(sigma,K,r),type='l',main = "put pour sigma=0.5",col='blue')
K<-exp(sigma**2/2)
lines(r,put(sigma,K,r),type='l',col='red')
K<-exp(-sigma**2/2)
lines(r,put(sigma,K,r),type='l',col='green')
legend("right",c("K=1","K=exp(sigma**2/2)","K=exp(-sigma**2/2)"),
       col=c('blue','red','green'),lwd=1,lty=1)
```



```
#cas sigma=0.05, pour différents sigma
sigma<-0.05
K<-1
plot(r,call(sigma,K,r),type='l',main = "call pour sigma=0.05",col='blue')
K<-exp(sigma**2/2)
lines(r,call(sigma,K,r),type='l',col='red')
K<-exp(-sigma**2/2)
lines(r,call(sigma,K,r),type='l',col='green')
legend("left",c("K=1","K=exp(sigma**2/2)","K=exp(-sigma**2/2)"),
       col=c('blue','red','green'),lwd=1,lty=1)
```



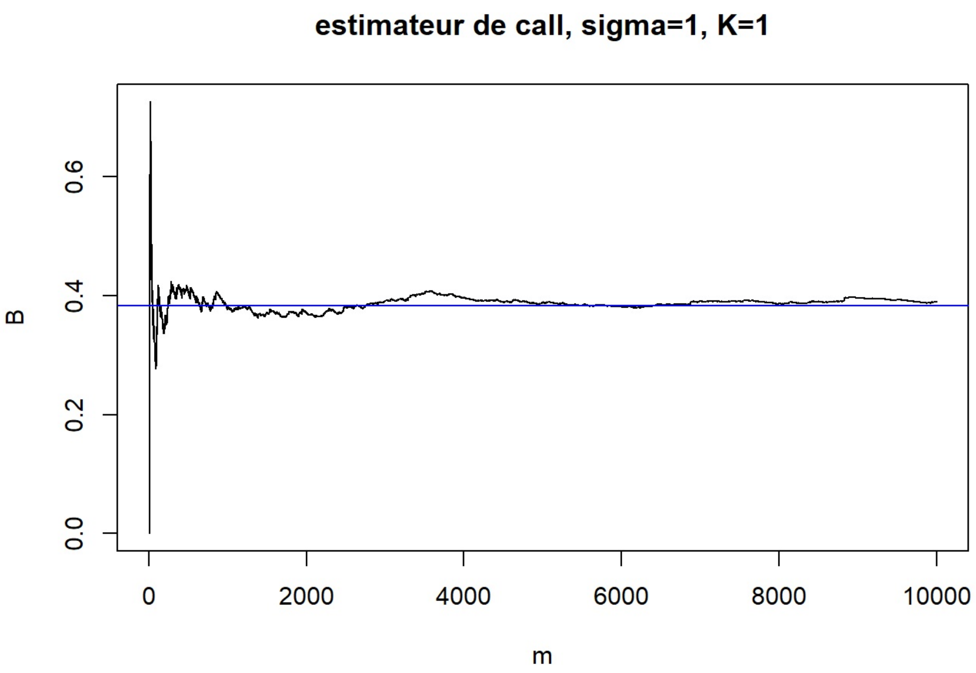

```
K<-1
plot(r,put(sigma,K,r),type='l',main = "put pour sigma=0.05",col='blue')
K<-exp(sigma**2/2)
lines(r,put(sigma,K,r),type='l',col='red')
K<-exp(-sigma**2/2)
lines(r,put(sigma,K,r),type='l',col='green')
legend("right",c("K=1","K=exp(sigma**2/2)","K=exp(-sigma**2/2)"),
       col=c('blue','red','green'),lwd=1,lty=1)
```



On remarque ici que plus σ est petit, plus les courbes se rapprochent.

Exercice 3

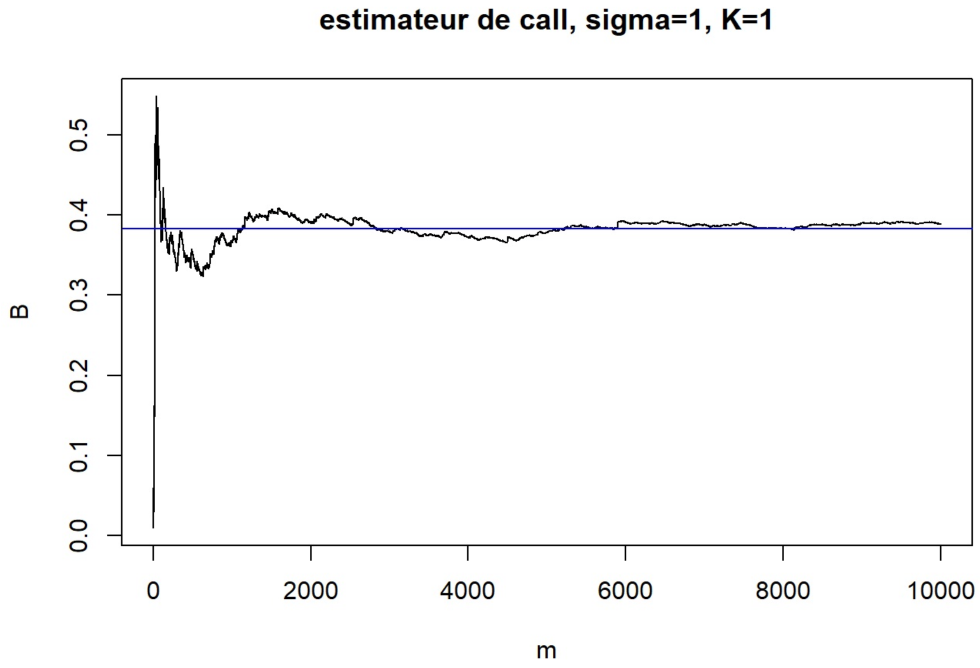
```
rcallput<-function(n, sigma, K)
{
  X<-rnorm(n) #3 On simule n Xi suivant une loi normale centrée réduite
  Y<-exp(sigma*X-sigma**2/2)-K
  Z<-K-exp(sigma*X-sigma**2/2)
  M<-matrix(ncol=2,nrow=n) #matrice de stockage des couples
  M[,1]=Y*(Y>0); M[,2]=Z*(Z>0)
  return(M)
}

A<-rcallput(1000,1,1) # simulation pour n=1000 et K=sigma=1

m<-seq(1,10000,1) # entier de 1 a 10 000



#test1
A<-rcallput(10000,1,1) # n=10 000
B<-cumsum(A[,1])/m # Ym barre

plot(m,B,type='l',main="estimateur de call, sigma=1, K=1") #Ym en fonctio de m, on voit la convergence
call_niveau<-call(1,1)
abline(call_niveau,0,col='blue') # on converge bien vers call
```




```
var(B) #variance de Ym
```

```
## [1] 0.0003869155
```

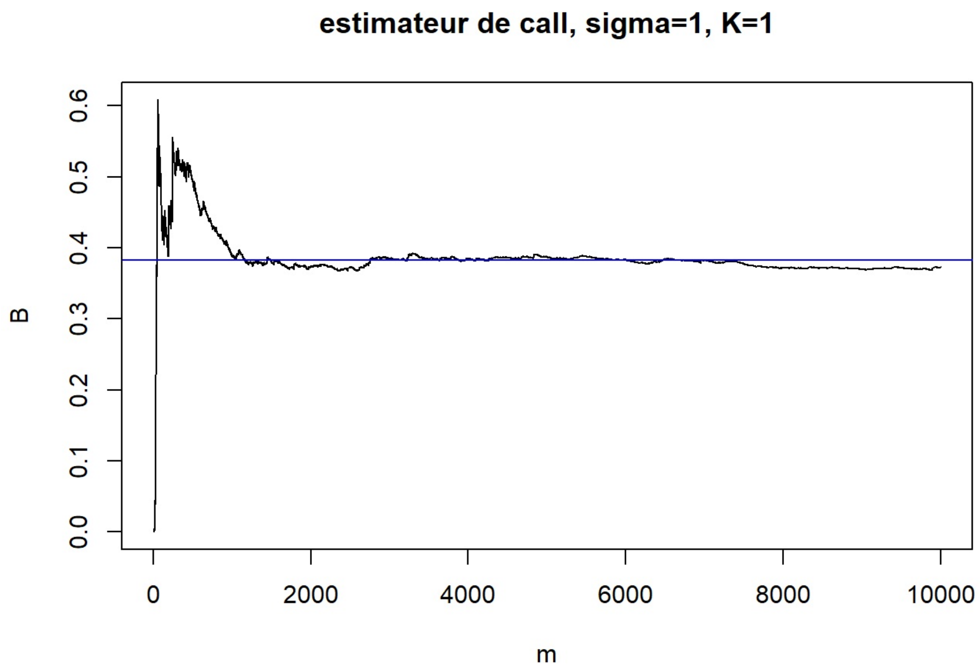
```
#test2
A<-rcallput(10000,1,1) # n=10 000
B<-cumsum(A[,1])/m # Ym barre

plot(m,B,type='l',main="estimateur de call, sigma=1, K=1") #Ym en fonctio de m, on voit la convergence
call_niveau<-call(1,1)
abline(call_niveau,0,col='blue') # on converge bien vers call
```




```
var(B) #variance de Ym
```

```
## [1] 0.000406919
```

```
#test3
A<-rcallput(10000,1,1) # n=10 000
B<-cumsum(A[,1])/m # Ym barre

plot(m,B,type='l',main="estimateur de call, sigma=1, K=1") #Ym en fonctio de m, on voit la convergence
call_niveau<-call(1,1)
abline(call_niveau,0,col='blue') # on converge bien vers call
```

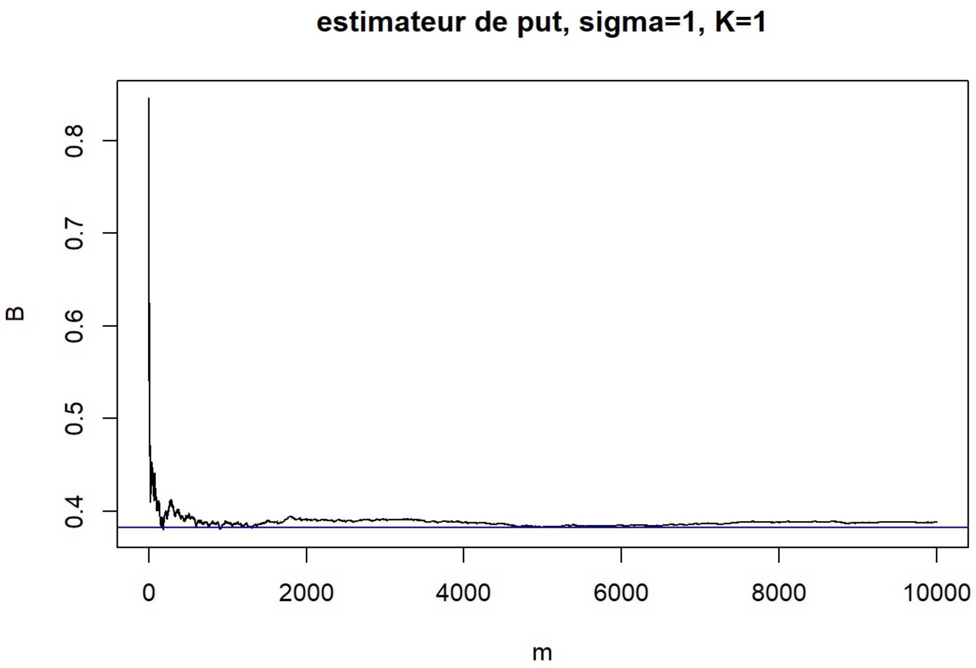



```
var(B) #variance de Ym
```

```
## [1] 0.001269413
```

On observe que les estimateurs convergent vers la valeur calculée par la fonction call.

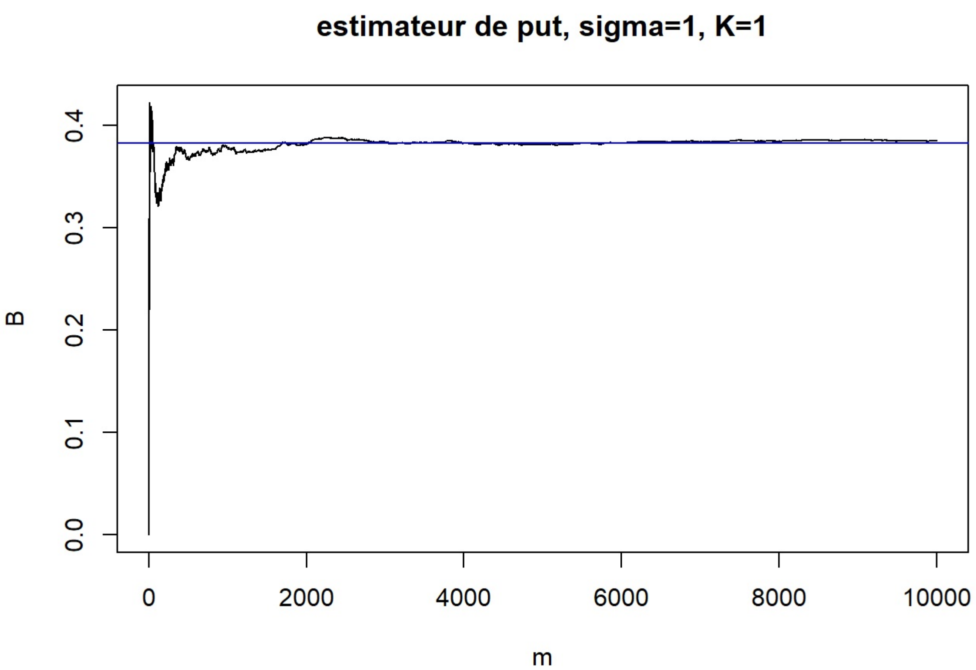
```
#test1
A<-rcallput(10000,1,1) # n=10 000
B<-cumsum(A[,2])/m # Zm barre

plot(m,B,type='l',main="estimateur de put, sigma=1, K=1") #Zm en fonctio de m, on voit la convergence
put_niveau<-put(1,1)
abline(put_niveau,0,col='blue') # on converge bien vers put
```




```
var(B) #variance de Zm
```

```
## [1] 9.131761e-05
```

```
#test2
A<-rcallput(10000,1,1) # n=10 000
B<-cumsum(A[,2])/m # Zm barre

plot(m,B,type='l',main="estimateur de put, sigma=1, K=1") #Zm en fonctio de m, on voit la convergence
put_niveau<-put(1,1)
abline(put_niveau,0,col='blue') # on converge bien vers put
```

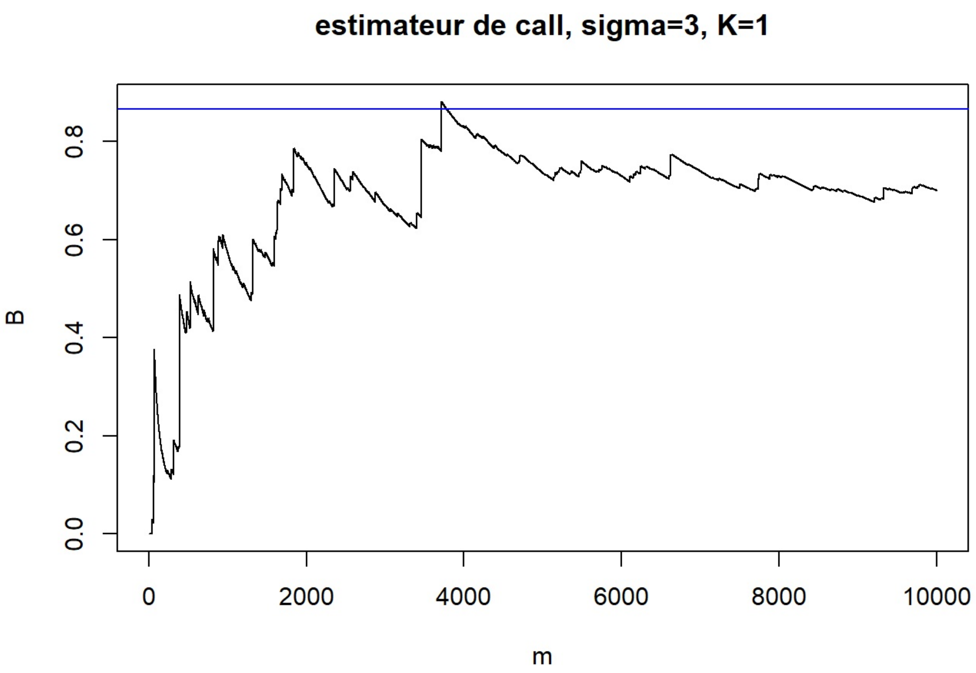


```
var(B) #variance de Zm
```

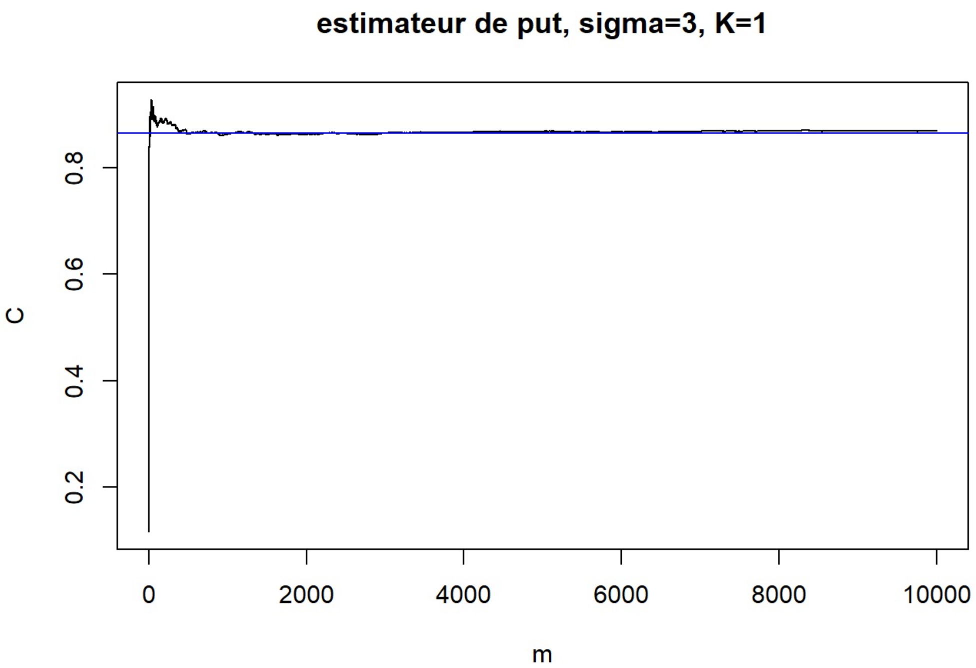
```
## [1] 9.695097e-05
```

On observe que les estimateurs convergent vers la valeur calculée par la fonction put.

```
#test1
A<-rcallput(10000,3,1)
B<-cumsum(A[,1])/m # Ym barre
C<-cumsum(A[,2])/m # Zm barre
call_niveau<-call(3,1)
put_niveau<-put(3,1)

plot(m,B,type='l',main="estimateur de call, sigma=3, K=1")
abline(call_niveau,0,col='blue') # on converge bien vers call
```




```
plot(m,C,type='l',main="estimateur de put, sigma=3, K=1")
abline(put_niveau,0,col='blue') # on converge bien vers call
```



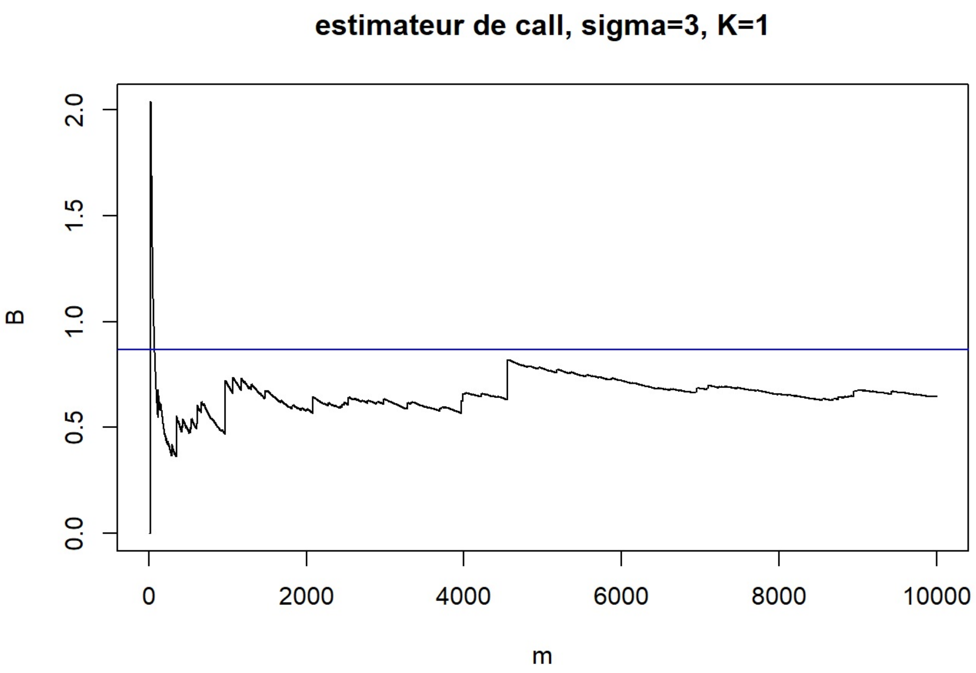

```
var(B) #variance de Ym
```

```
## [1] 0.01855574
```

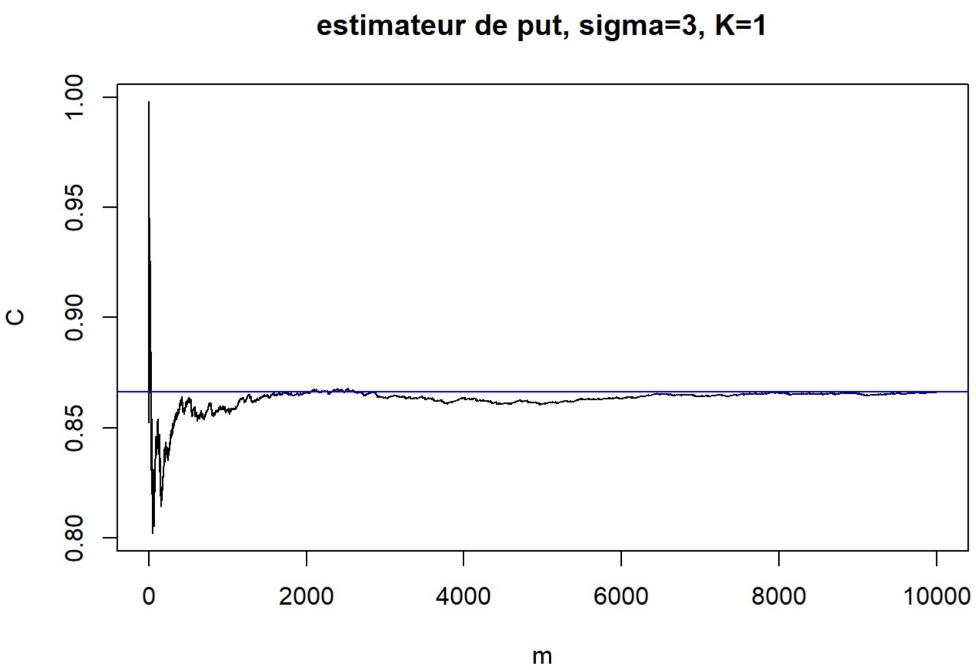
```
var(C) #variance de Zm
```

```
## [1] 9.325699e-05
```

```
#test2
A<-rcallput(10000,3,1)
B<-cumsum(A[,1])/m # Ym barre
C<-cumsum(A[,2])/m # Zm barre
call_niveau<-call(3,1)
put_niveau<-put(3,1)

plot(m,B,type='l',main="estimateur de call, sigma=3, K=1")
abline(call_niveau,0,col='blue') # on converge bien vers call
```




```
plot(m,C,type='l',main="estimateur de put, sigma=3, K=1")
abline(put_niveau,0,col='blue') # on converge bien vers call
```



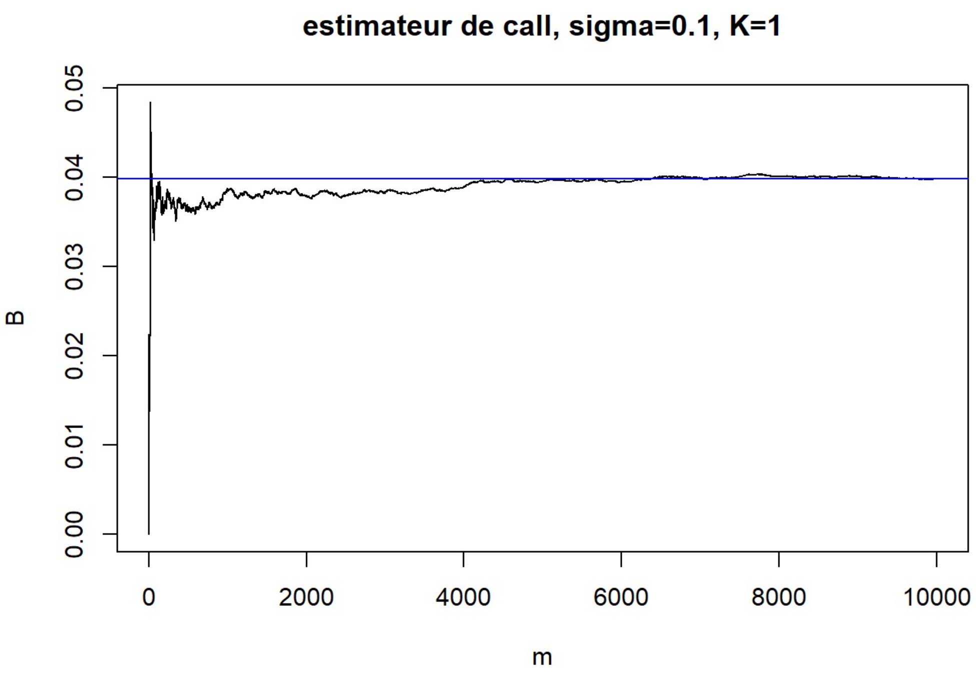

```
var(B) #variance de Ym
```

```
## [1] 0.008801319
```

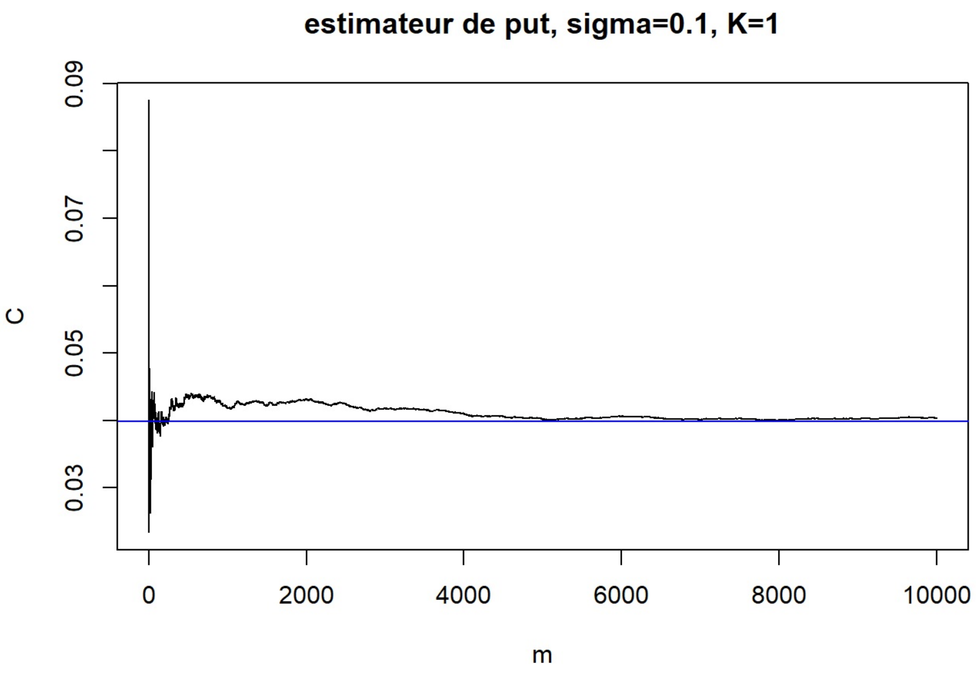
```
var(C) #variance de Zm
```

```
## [1] 3.88932e-05
```

```
#test1
A<-rcallput(10000,0.1,1)
B<-cumsum(A[,1])/m # Ym barre
C<-cumsum(A[,2])/m # Zm barre
call_niveau<-call(0.1,1)
put_niveau<-put(0.1,1)

plot(m,B,type='l',main="estimateur de call, sigma=0.1, K=1")
abline(call_niveau,0,col='blue') # on converge bien vers call
```




```
plot(m,C,type='l',main="estimateur de put, sigma=0.1, K=1")
abline(put_niveau,0,col='blue') # on converge bien vers call
```



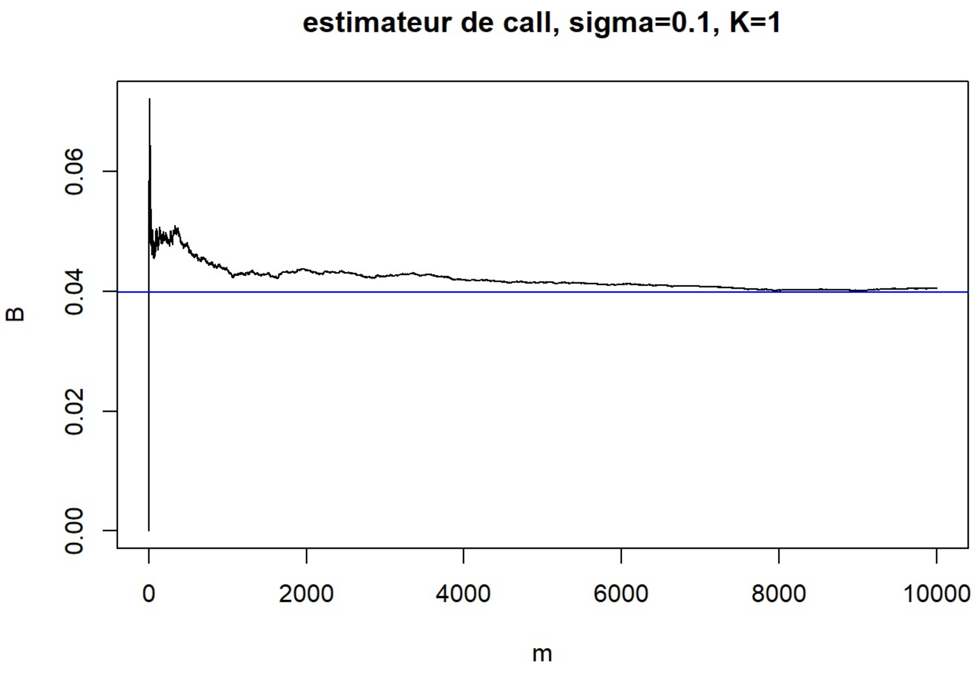
```
var(B) #variance de Ym
```

```
## [1] 2.096543e-06
```

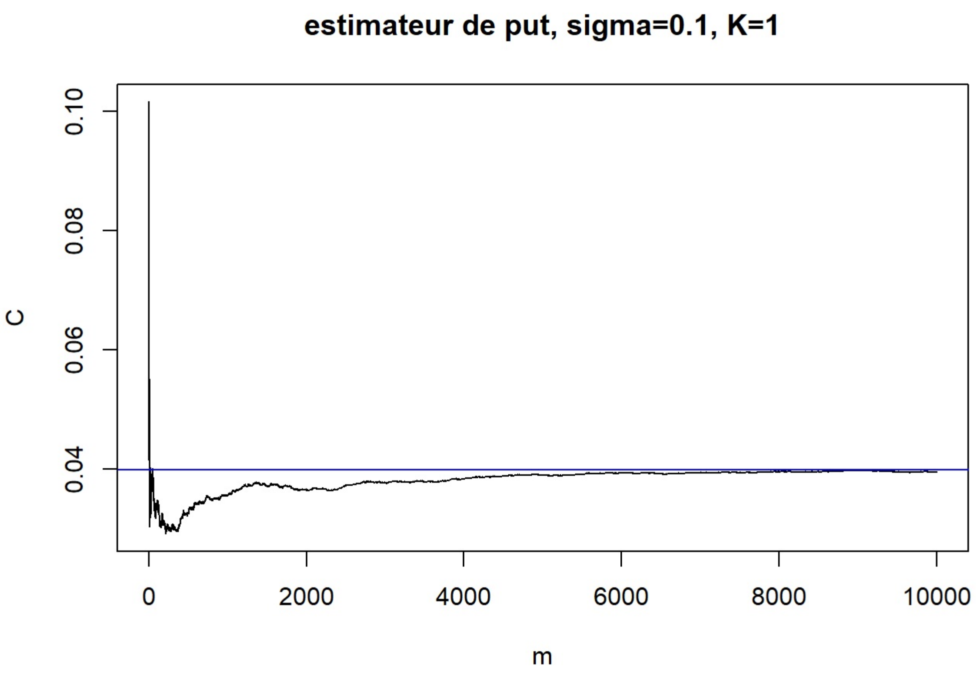
```
var(C) #variance de Zm
```

```
## [1] 1.604615e-06
```

```
#test2
A<-rcallput(10000,0.1,1)
B<-cumsum(A[,1])/m # Ym barre
C<-cumsum(A[,2])/m # Zm barre
call_niveau<-call(0.1,1)
put_niveau<-put(0.1,1)

plot(m,B,type='l',main="estimateur de call, sigma=0.1, K=1")
abline(call_niveau,0,col='blue') # on converge bien vers call
```




```
plot(m,C,type='l',main="estimateur de put, sigma=0.1, K=1")
abline(put_niveau,0,col='blue') # on converge bien vers call
```



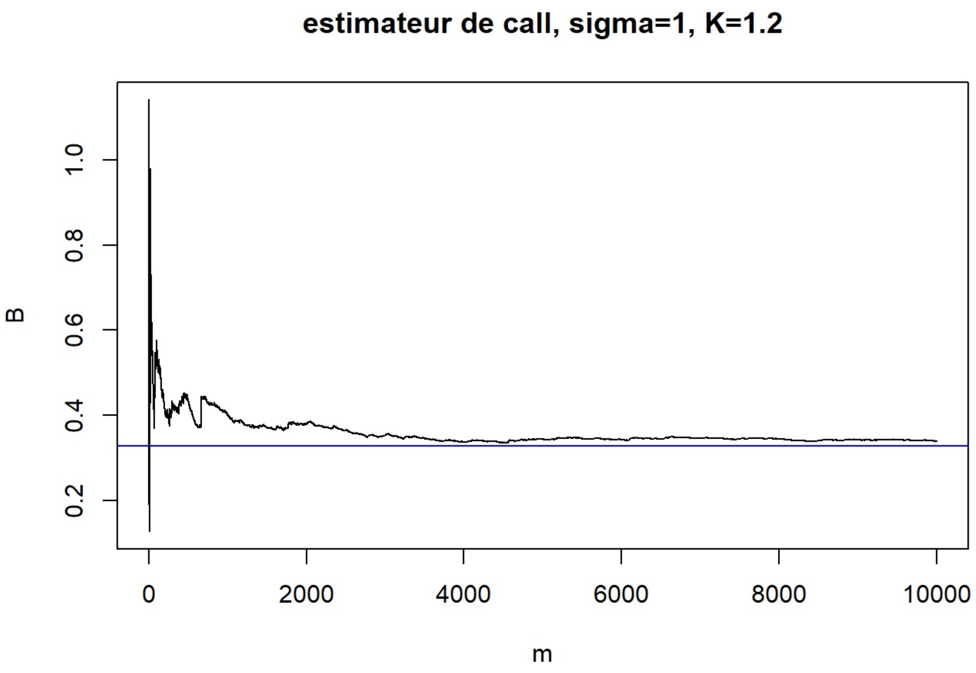
```
var(B) #variance de Ym
```

```
## [1] 5.136028e-06
```

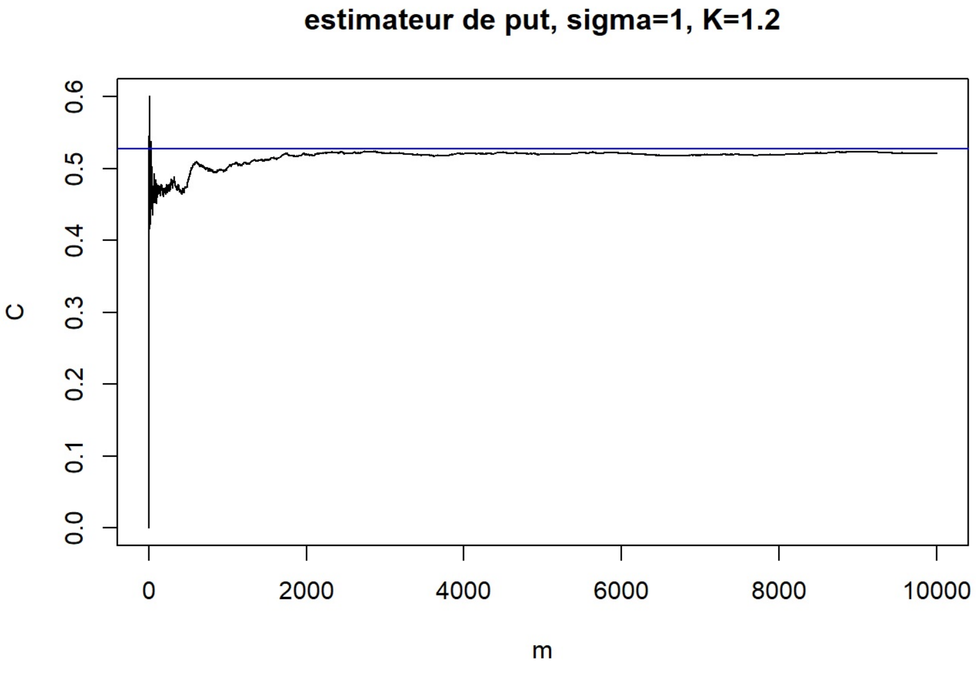
```
var(C) #variance de Zm
```

```
## [1] 4.547919e-06
```

```
#test1
A<-rcallput(10000,1,1.2)
B<-cumsum(A[,1])/m # Ym barre
C<-cumsum(A[,2])/m # Zm barre
call_niveau<-call(1,1.2)
put_niveau<-put(1,1.2)

plot(m,B,type='l',main="estimateur de call, sigma=1, K=1.2")
abline(call_niveau,0,col='blue') # on converge bien vers call
```




```
plot(m,C,type='l',main="estimateur de put, sigma=1, K=1.2")
abline(put_niveau,0,col='blue') # on converge bien vers call
```



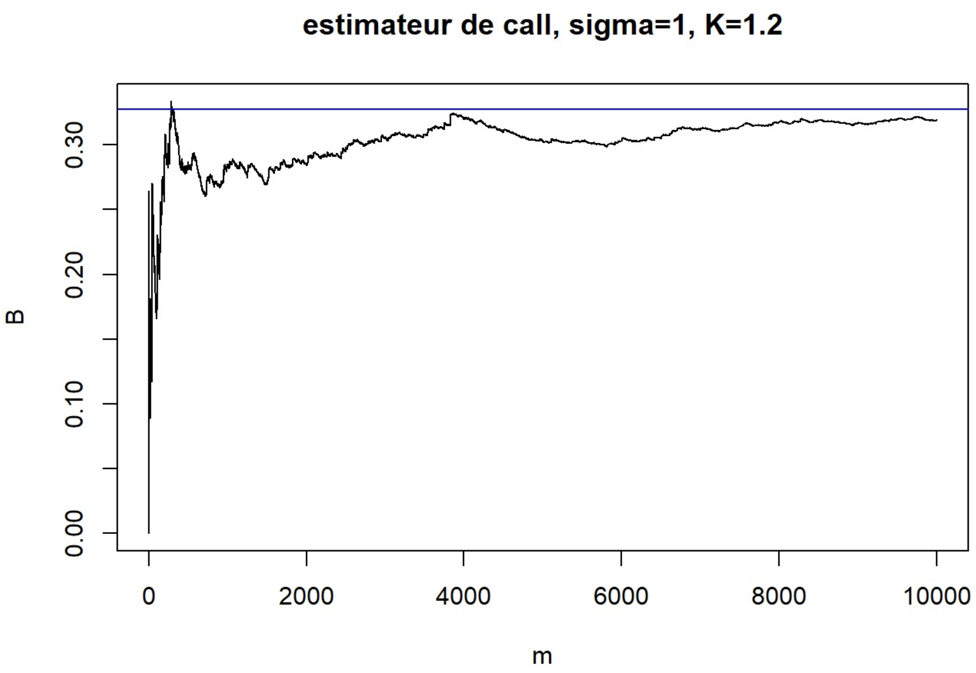

```
var(B) #variance de Ym
```

```
## [1] 0.001244787
```

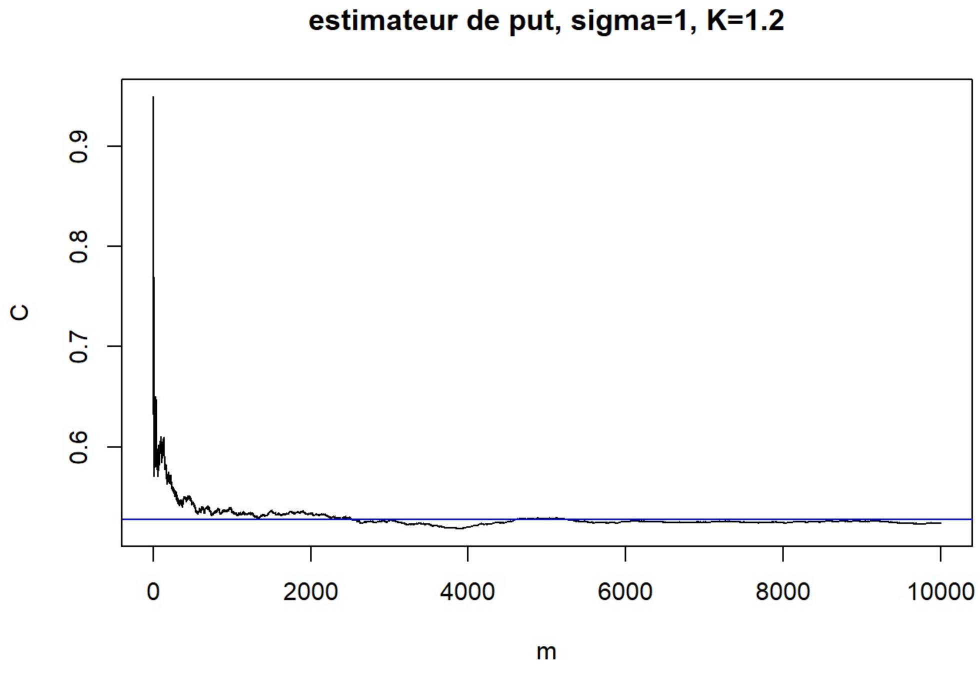
```
var(C) #variance de Zm
```

```
## [1] 0.0001675351
```

```
#test2
A<-rcallput(10000,1,1.2)
B<-cumsum(A[,1])/m # Ym barre
C<-cumsum(A[,2])/m # Zm barre
call_niveau<-call(1,1.2)
put_niveau<-put(1,1.2)

plot(m,B,type='l',main="estimateur de call, sigma=1, K=1.2")
abline(call_niveau,0,col='blue') # on converge bien vers call
```




```
plot(m,C,type='l',main="estimateur de put, sigma=1, K=1.2")
abline(put_niveau,0,col='blue') # on converge bien vers call
```



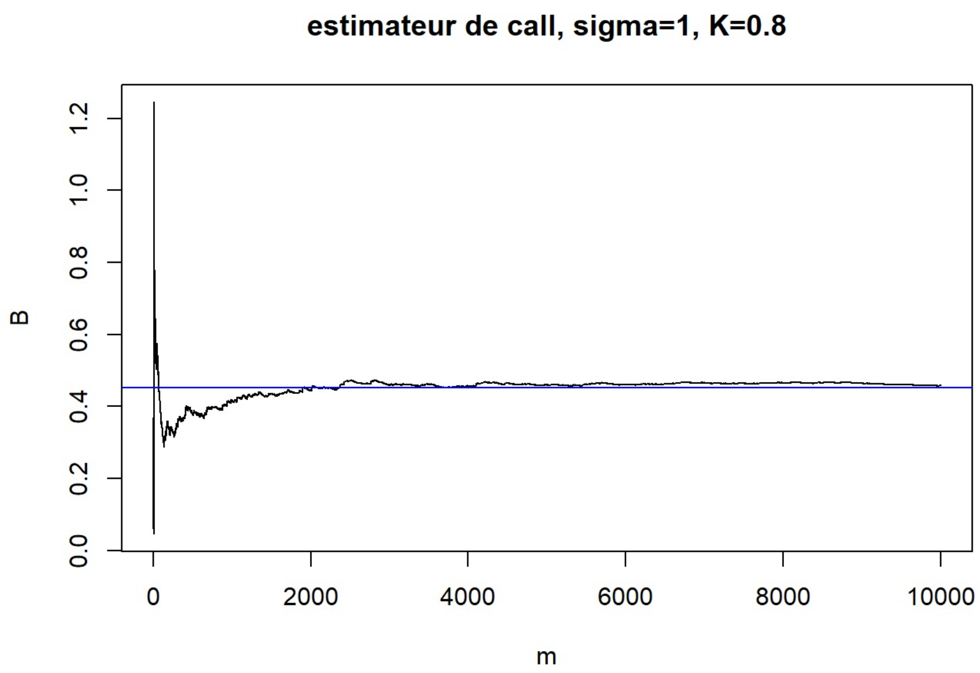

```
var(B) #variance de Ym
```

```
## [1] 0.0004266513
```

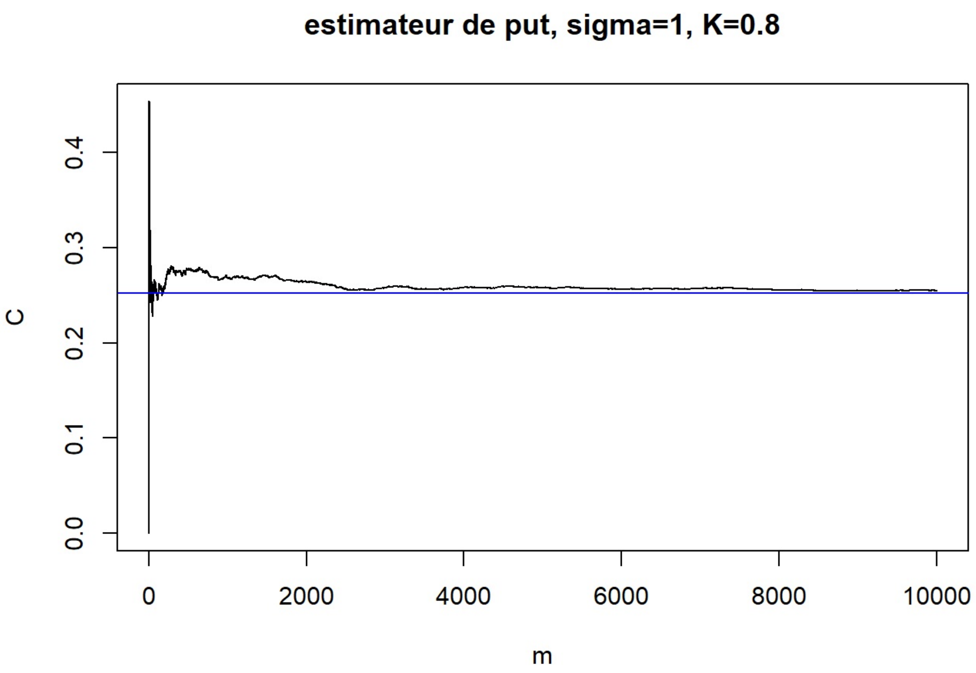
```
var(C) #variance de Zm
```

```
## [1] 0.0001659108
```

```
#test1
A<-rcallput(10000,1,0.8)
B<-cumsum(A[,1])/m # Ym barre
C<-cumsum(A[,2])/m # Zm barre
call_niveau<-call(1,0.8)
put_niveau<-put(1,0.8)

plot(m,B,type='l',main="estimateur de call, sigma=1, K=0.8")
abline(call_niveau,0,col='blue') # on converge bien vers call
```



```
plot(m,C,type='l',main="estimateur de put, sigma=1, K=0.8")
abline(put_niveau,0,col='blue') # on converge bien vers call
```



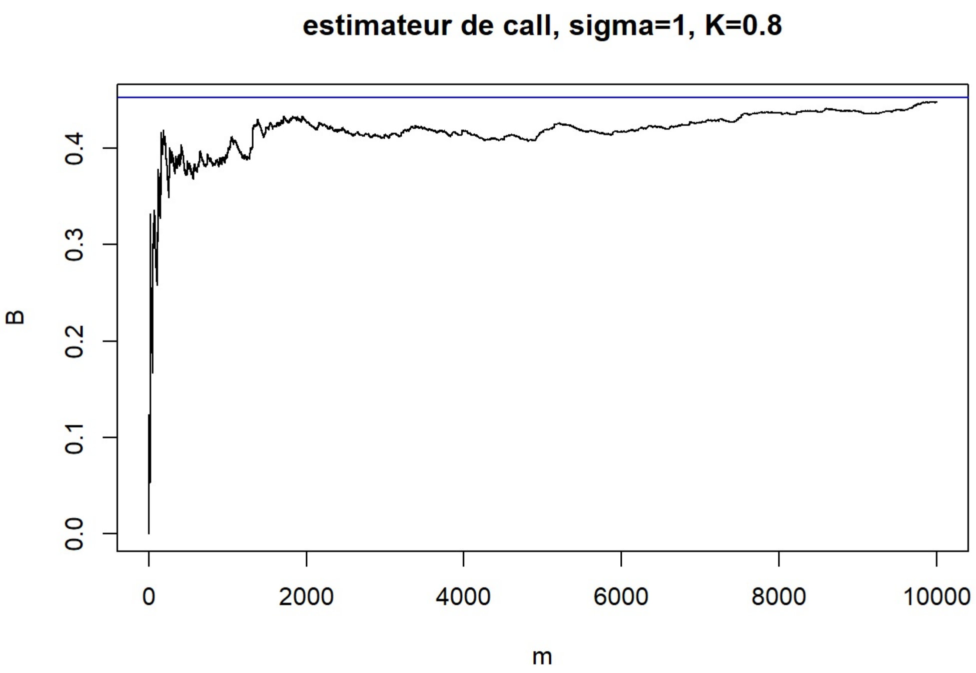

```
var(B) #variance de Ym
```

```
## [1] 0.001151113
```

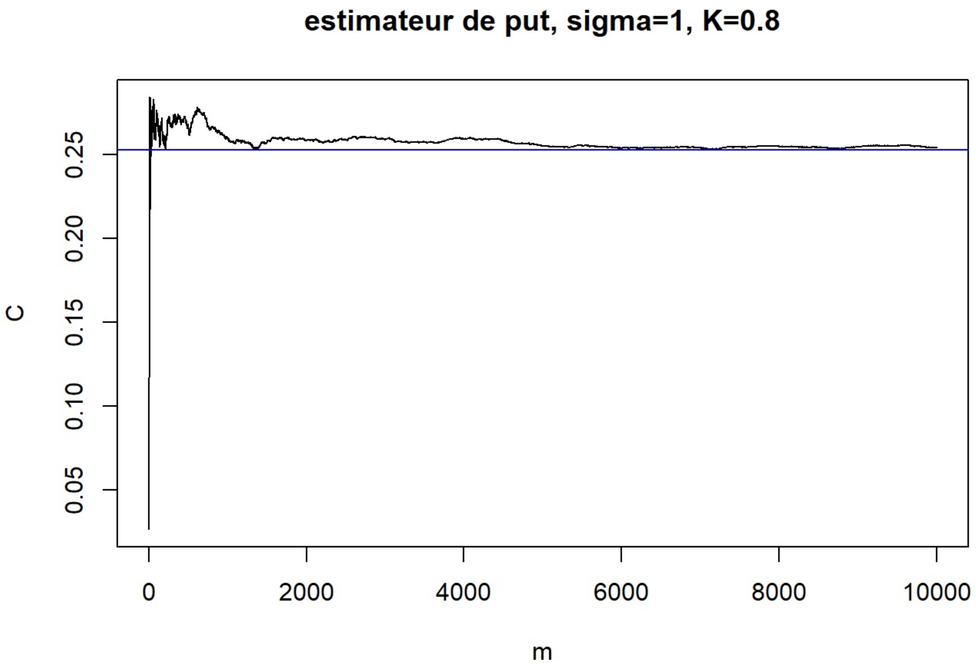
```
var(C) #variance de Zm
```

```
## [1] 6.151431e-05
```

```
#test2
A<-rcallput(10000,1,0.8)
B<-cumsum(A[,1])/m # Ym barre
C<-cumsum(A[,2])/m # Zm barre
call_niveau<-call(1,0.8)
put_niveau<-put(1,0.8)

plot(m,B,type='l',main="estimateur de call, sigma=1, K=0.8")
abline(call_niveau,0,col='blue') # on converge bien vers call
```



```
plot(m,C,type='l',main="estimateur de put, sigma=1, K=0.8")
abline(put_niveau,0,col='blue') # on converge bien vers call
```




```
var(B) #variance de Ym
```

```
## [1] 0.000718675
```

```
var(C) #variance de Zm
```

```
## [1] 4.264427e-05
```

Dans tout les cas, les estimateurs convergent bien vers la valeur attendue.

On ramarque que la variance de $\bar{Y}_m$ est plus élevé que celle de $\bar{Z}_m$, ce qui est assez visible pour le cas $\sigma = 3$. Il vaut mieux calculer la fonction qui donne la plus petite variance pour avoir une meilleur convergence. Ici il vaut donc mieux calculer la fonction put.

$Y = 0$ si on a :

$$e^{\sigma X - \frac{\sigma^2}{2}} - K \leq 0 \Leftrightarrow e^{\sigma X - \frac{\sigma^2}{2}} \leq K \Leftrightarrow \sigma X - \frac{\sigma^2}{2} \leq ln(K) = 0, \text{ car } K = 1,$$

$$X - \frac{\sigma}{2} \leq 0 \Leftrightarrow X \leq \frac{\sigma}{2}.$$

Et $Z = 0$ si on a:

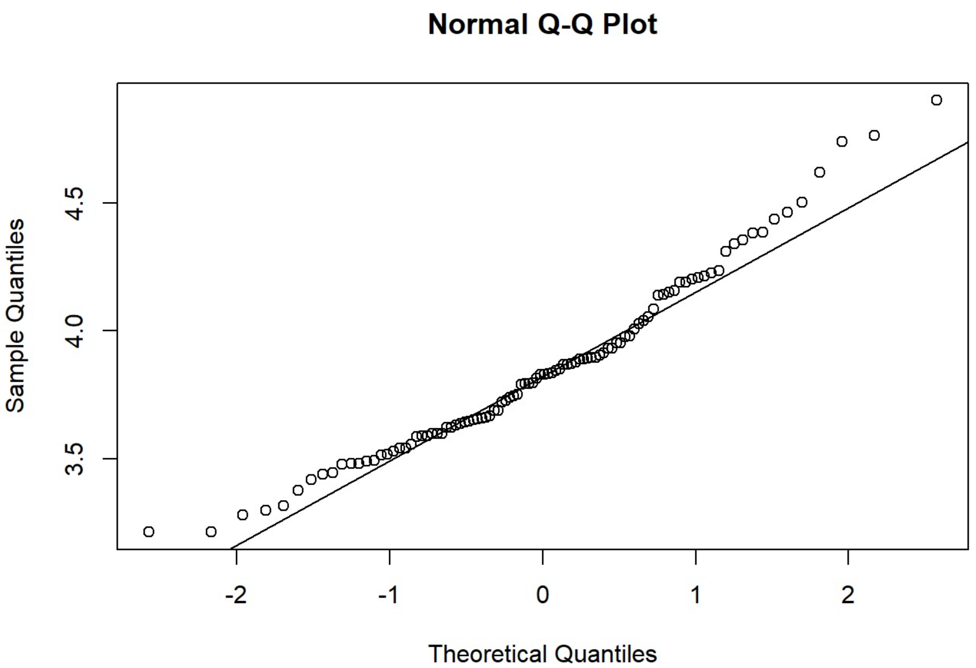
$$K - e^{\sigma X - \frac{\sigma^2}{2}} \leq 0 \Leftrightarrow K \leq e^{\sigma X - \frac{\sigma^2}{2}} \Leftrightarrow 0 = ln(K) \leq \sigma X - \frac{\sigma^2}{2}, \text{ car } K = 1,$$

$$0 \leq \sigma X - \frac{\sigma^2}{2} \Leftrightarrow X \geq \frac{\sigma}{2}.$$

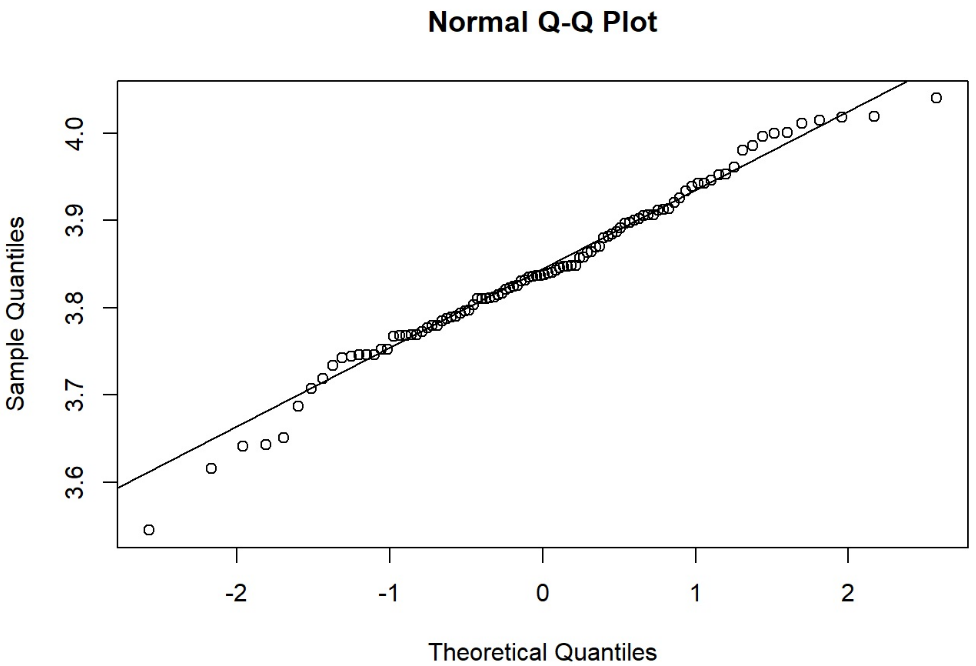
On peut expliquer les observatons sur la variance par le fait que X est centrée est suit une loi normale, on a donc plus de termes proche de 0. Ainsi, sachant que chaque terme est susceptible d'augmenter la variance, on trouve que $\bar{Y}_m$ a une plus grande variance.

Exercice 4

```
#Q1 Q2 Q3
n<-100
m<-1000
Y<-matrix(ncol=n,nrow=m)
Z<-matrix(ncol=n,nrow=m)
for (i in 1:n)
{
  A<-rcallput(m,1,1)
  Y[,i]=A[,1]
  Z[,i]=A[,2]
}

vect_y<-colSums(Y)/n
vect_z<-colSums(Z)/n

#Q4
qqnorm(vect_y)
qqline(vect_y)
```




```
qqnorm(vect_z)
qqline(vect_z)
```



qqplot et qqnorm donne le rapport quantile-quantile entre l'échantillon et la loi normale adaptée.

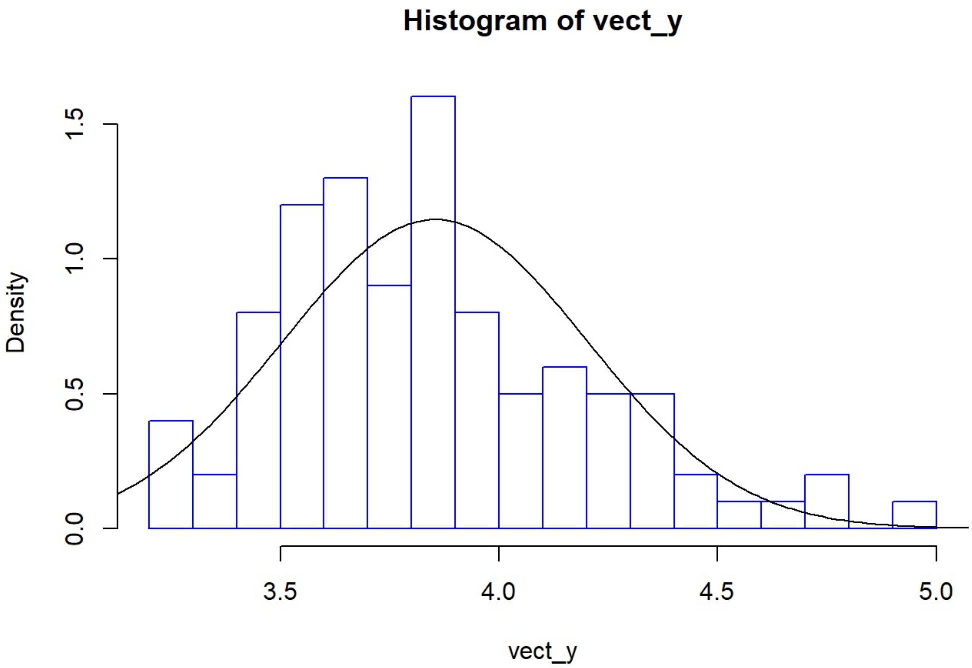
Plus la pente est raide, plus la variance est grande. Ici, on observe que la variance de Y est plus grande que celle de Z. C'était prévisible, car théoriquement $Var(Y) > Var(Z)$.

L'intersection à l'origine correspond à la moyenne.

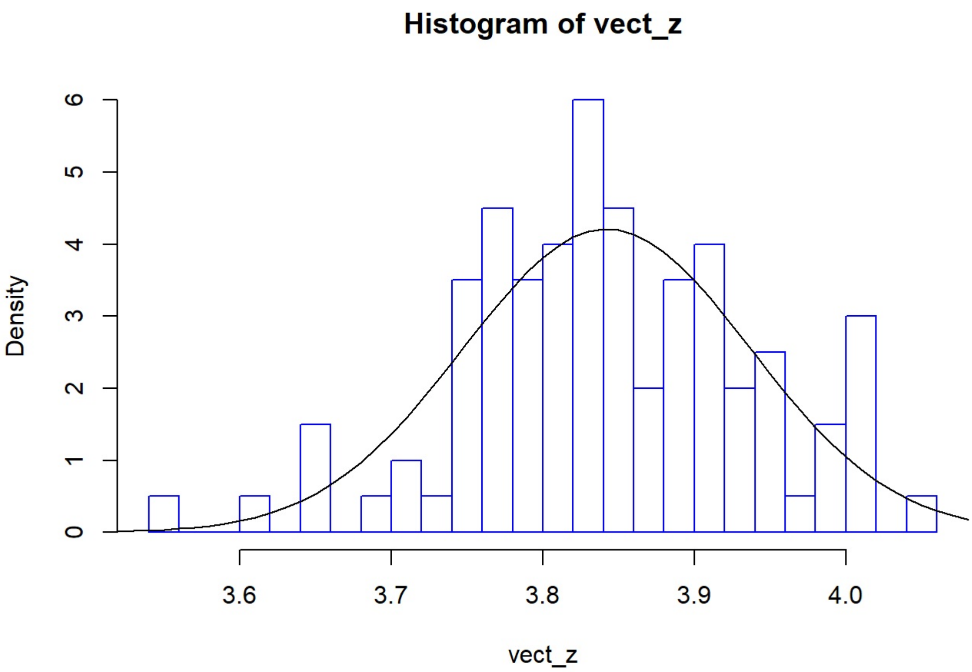
On remarque que les échantillons semblent proches d'une loi normale adaptée.

On trace donc les histogrammes avec la densité d'une loi normale adaptée à titre de comparaison:

```
#Q5-6
I<-seq(2,6,0.01)

hist(vect_y,20,freq=FALSE,border='blue')
lines(I,dnorm(I,mean=mean(vect_y),sd=sqrt(var(vect_y))))
```




```
#Q7

hist(vect_z,20,freq=FALSE,border='blue')
lines(I,dnorm(I,mean=mean(vect_z),sd=sqrt(var(vect_z))))
```



On observe une similarité.

```
#Q8

n<-100
m<-100
Y<-matrix(ncol=n,nrow=m)
Z<-matrix(ncol=n,nrow=m)
for (i in 1:n)
{
  A<-rcallput(m,1,1)
  Y[,i]=A[,1]
  Z[,i]=A[,2]
}


vect_y<-colSums(Y)/n
vect_z<-colSums(Z)/n
```

On obtient la moyenne des deux estimateurs :

```
mean(vect_y)
```

```
## [1] 0.3643599
```

```
mean(vect_z)
```

```
## [1] 0.3863631
```

On obtient l'intervalle de confiance suivant :

```
#IC y
int_conf_call<-c(mean(vect_y)-1.96*(sqrt(var(vect_y)))/sqrt(100),mean(vect_y)+1.96*(sqrt(var(vect_y)))/sqrt(100))
int_conf_call
```

```
## [1] 0.3429758 0.3857441
```

```
#IC z
int_conf_put<-c(mean(vect_z)-1.96*(sqrt(var(vect_z)))/sqrt(100),mean(vect_z)+1.96*(sqrt(var(vect_z)))/sqrt(100))
int_conf_put
```

```
## [1] 0.3799764 0.3927499
```

On a bien un intervalle de confiance de taille inférieure pour put.

Exercice 5

```
#Q1

I<-function(X)
{
  n<-length(X)
  int_conf<-c(mean(X)-1.96*(sqrt(var(X)))/sqrt(n),mean(X)+1.96*(sqrt(var(X)))/sqrt(n))
  return(int_conf)
}
```

Question 2

On obtient l'intervalle de confiance suivant :

```
A<-rcallput(1000,1,1)
vect_y<-A[,1]
I(vect_y)
```

```
## [1] 0.3446225 0.5305586
```

Question 3

On utilise la relation liant call et put donné a l'exercice 1, qui pour notre cas donne $C = P$ (car $C - P = 0$). On va donc calculer l'intervalle de confiance de l'estimateur de put, qui a une variance inférieure à l'estimateur de call. Grâce à la relation citée, on aura également l'intervalle de confiance de call. On trouve l'intervalle de confiance suivant :

```
A<-rcallput(1000,1,1)
vect_z<-A[,2]
I(vect_z)
```

```
## [1] 0.3498384 0.3909446
```

On obtient bien un intervalle de confiance de taille inférieure.

Question 4

On trouve l'intervalle de confiance suivant :

```
#Q4

V<-rexp(1000,1./2) #1000 réalsation de V

K=1
sigma=1

q1<-K-exp(sigma*sqrt(V)-sigma**2/2)
q2<-K-exp(-sigma*sqrt(V)-sigma**2/2)
q<-q1*(q1>0)+q2*(q2>0)
I(q/sqrt(2*pi*V)) #meilleur que q1
```

```
## [1] 0.3350805 0.3784684
```

La taille de l'intervalle de confiance ne diminue pas, il n'est pas meilleur.

Question 5

On dérive la fonction $f(x) = K - e^{\sigma X - \frac{\sigma^2}{2}}$ pour étudier sa monotonie.

$f'(x) = -\sigma e^{\sigma X - \frac{\sigma^2}{2}} < 0$, ainsi f est décroissante.

Question 6

On peut prendre $T(X) = -X$ car X est symétrique.

Question 7

On utilise l'estimateur $e_2 = \frac{1}{2n}\sum_{i=1}^{n}\left[g(X_i) + g(T(X_i))\right]$

Question 8

On code e_2 et on obtient l'intervalle de confiance suivant :

```
X<-rnorm(1000)
a<-K-exp(sigma*X-sigma^2/2)
a<-a*(a>0)
b<-K-exp(sigma*(-X)-sigma^2/2)
b<-b*(b>0)
e2<-(1/2000)*sum(a+b)
e2
```

```
## [1] 0.3808537
```

```
I((1/2)*(a+b)) # meilleur qu'on a
```

```
## [1] 0.3783939 0.3833135
```

La taille de l'intervalle de confiance est plus petit que ce qu'on a obtenu précedemment, il est donc meilleur.