Projet Quantitative finance

LIN Cécile
LOUIS-LUCAS Théophile

M1 Finance - 2023

Contents

[1]:
```
import numpy as np
import numpy.random as sim # simulate random variable
import matplotlib.pyplot as plt # graphiques
from scipy.stats import norm
```

1 Introduction

Soit une base stochastique $\left(\Omega, F_T, (F_t)_{t\geq 0}, P\right)$. La maturité est $T > 0$. Le marché financier considéré est composé de deux actifs.

- un bond S^0 dont la dynamique est
$$dS_t^0 = r_t S_t^0 dt$$
et de valeur initiale
$$S_0^0 = 1$$
  - r_t est le taux d'intérêt sans risque du marché obligataire dont la dynamique est donnée plus loin.
- un actif risqué S a pour dynamique actualisée:
$$d\tilde{S}_t = \sigma_t \tilde{S}_t dW_t$$
avec
$$\tilde{S}_0 = 25$$
où $t \to \sigma_t$ est une volatilité stochastique dont la dynamique est donnée ci-dessous. Enfin, W est un mouvement Brownien standard.

2 PARTIE 1

On suppose que la volatilité stochastique vérifie $\sigma_t = 0.01(2 + \sin(U_t))$ où U_t vérifie la dynamique:

$$dU_t = 0.1 U_t dB_t$$

et

$$U_0 = 1$$

avec B un mouvement Brownien standard indépendant de W

Sur un intervalle $[0, T]$ on discretise l'intervalle en n intervalles de temps. On obtient n dates $t_i^n = \frac{T}{n} \times i \quad , \quad \forall i \in \{1, 2, \ldots, n\}$

B est un mouvement Brownien standard donc par définition

$$\begin{cases} B_0 = 0 \\ B_{t_i^n} = B_{t_{i-1}^n} + \sqrt{\frac{T}{n}} \times G_i \quad , \quad (G_i)_{i=1}^n \text{ sont iid et suivent une loi } N(0,1) \end{cases}$$

Afin de discrétiser le processus U , on a:

$$dU_t = 0.1 U_t dB_t$$

$$\Longleftrightarrow$$

$$U_{t_i^n} - U_{t_{i-1}^n} = 0.1 \times U_{t_{i-1}^n} \times \left(B_{t_i^n} - B_{t_{i-1}^n}\right)$$

- Euler Scheme:

$$\begin{cases} U_0 = 1 \\ U_{t_i^n} = U_{t_{i-1}^n} + 0.1 \times U_{t_{i-1}^n} \times \sqrt{\frac{T}{n}} \times G_i \end{cases}$$

On obtient alors la volatilité stochastique σ aux dates discrètes:

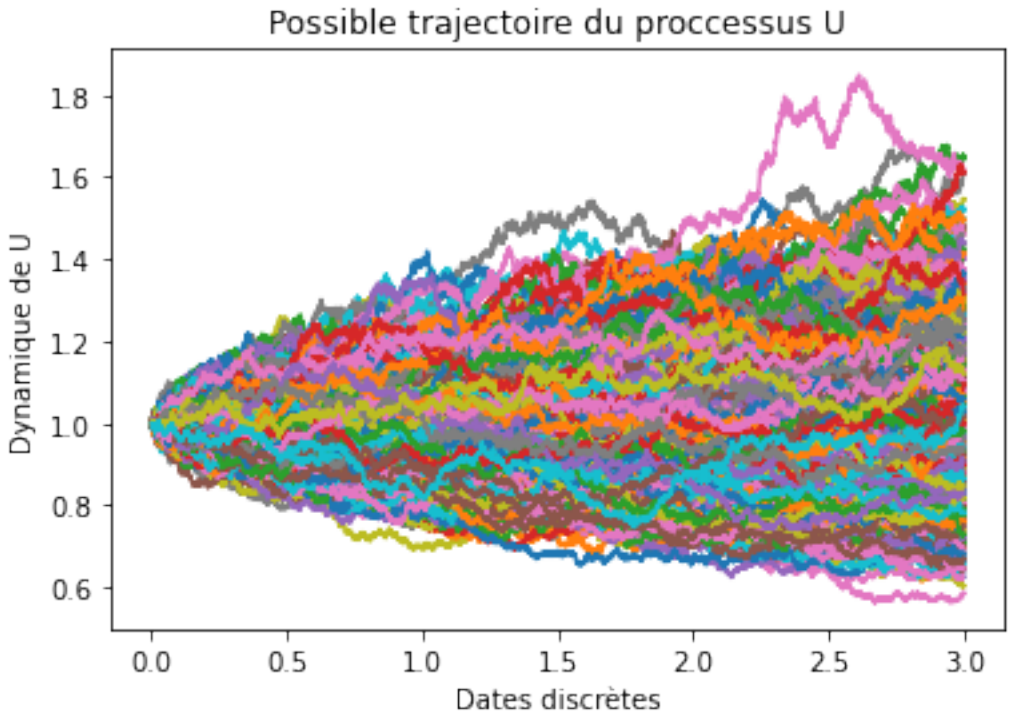
$$\sigma_{t_i^n} = 0.01 \times \left(2 + \sin\left(U_{t_i^n}\right)\right)$$

$$\Longleftrightarrow$$

$$\sigma_{t_i^n} = 0.01 \times \left[2 + \sin\left(U_{t_{i-1}^n} + 0.1 \times U_{t_{i-1}^n} \times \sqrt{\frac{T}{n}} \times G_i\right)\right]$$

[2]:
```
T = 3 # maturité
n = 10000 #precision de l'approximation
N = 1000 # nombre de trajectoire

step = T/n #discretisation

#B0 = 0 # valeur initiale du mouvement Brownien standard
B = np.zeros((n+1,N)) # matrice des valeurs du mouvement Brownien

U0 = 1 # valeur initiale de U_t
U = U0*np.ones((n+1,N)) # matrice n+1 x N rempli de la valeur initiale de  U

#sig = np.zeros((n+1,N)) # matrice des valeurs de la volatilité

#calcul de la volatilité stochastique
def sigma(x):
    return 0.01*(2+np.sin(x))

for j in range(N): # N trajectoires representés par les colonnes de la matrice
    for i in range(1,n+1):
        # euler scheme B
        # B0 = 0 comme la matrice est rempli de 0, on ne change pas à la ligne 0
        # B_ti = B_{ti-1} + sqrt(variance) * G_i
        # ici variance = T/n = step et avec G_i = loi normale
        B[i,j] = B[i-1,j]+np.sqrt(step)*sim.randn()

        # euler scheme U
        # U_0 = 1
        # U_ti = U_(ti-1) + 0.1*U_(ti-1)*[B_ti - B_(ti-1)]
        # avec B_ti - B_(ti-1) = sqrt(variance)*G_i simuler dans la matrice␣
 ↪précédente
        U[i,j] = U[i-1,j]+0.1*U[i-1,j]*(B[i,j]-B[i-1,j])
```

```
        # calcul de la volatilité stochastique
        # sig[i,j] = sigma(U[i,j])

dates=np.linspace(0,T,n+1) # axes des abscisses

plt.plot(dates,U) # possible trajectoire du proccessus U
plt.title("Possible trajectoire du proccessus U")
plt.xlabel("Dates discrètes")
plt.ylabel("Dynamique de U")
plt.show()
```



3 PARTIE 2

On suppose que le taux sans risque vérifie la dynamique

$$dr_t = 0.05\,(0.03 - r_t)\,dt + 0.1 dB_t$$

et sa valeur initiale

$$r_0 = 0.02$$

où B est le mouvement Brownien standard de la partie 1.

On a de nouveau le mouvement Brownien standard de la partie 1:

$$\begin{cases} B_0 = 0 \\ B_{t_i^n} = B_{t_{i-1}^n} + \sqrt{\frac{T}{n}} \times G_i \quad , \quad (G_i)_{i=1}^n \text{ sont iid et suivent une loi } N(0,1) \end{cases}$$

Afin de discrétiser le processus r , on a:

$$dr_t = 0.05 \times (0.03 - r_t)\,dt + 0.1 dB_t$$
$$\Longleftrightarrow$$
$$r_{t_i^n} - r_{t_{i-1}^n} = 0.05 \left(0.03 - r_{t_{i-1}^n}\right) \Delta t_i^n + 0.1 \left(B_{t_i^n} - B_{t_{i-1}^n}\right) \quad , \quad \Delta t_i^n = \frac{T}{n}$$

- Euler Scheme:

$$\begin{cases} r_0 = 0.02 \\ r_{t_i^n} = r_{t_{i-1}^n} + 0.05 \left(0.03 - r_{t_{i-1}^n}\right) \frac{T}{n} + 0.1 \times \sqrt{\frac{T}{n}} \times G_i \end{cases}$$

Pour l'actif sans risque S^0 , on a:

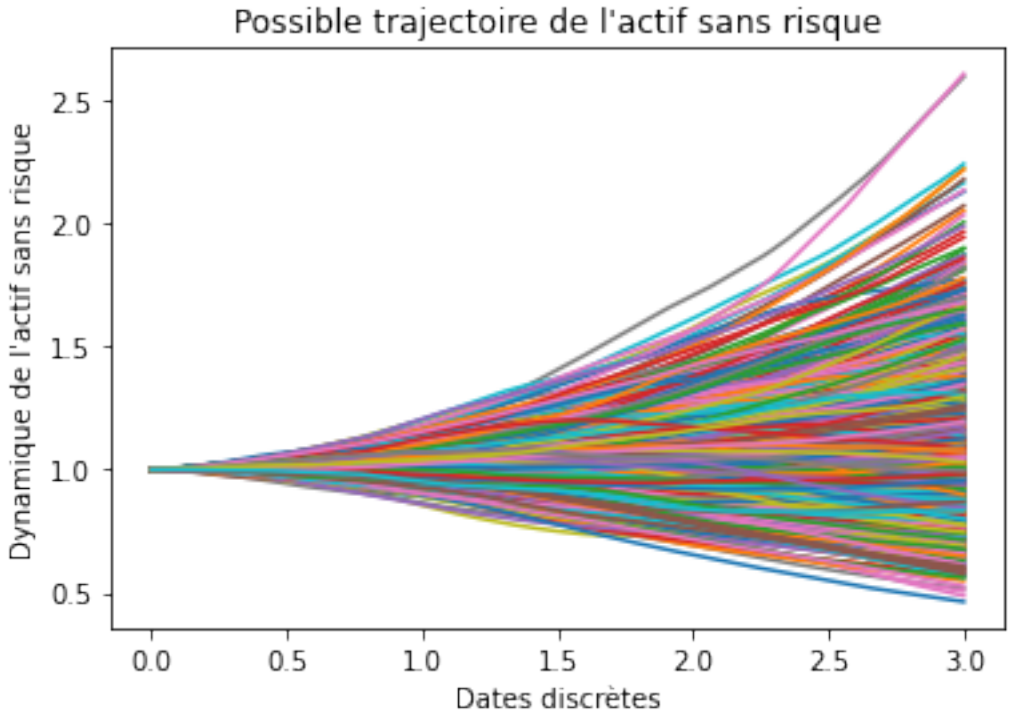
$$dS_t^0 = r_t S_t^0 dt$$
$$\Longleftrightarrow$$
$$S_{t_i^n}^0 - S_{t_{i-1}^n}^0 = r_{t_{i-1}^n} \times S_{t_{i-1}^n}^0 \times \frac{T}{n}$$

- Euler Scheme:

$$\begin{cases} S_0^0 = 1 \\ S_{t_i^n}^0 = S_{t_{i-1}^n}^0 + r_{t_{i-1}^n} \times S_{t_{i-1}^n}^0 \times \frac{T}{n} \end{cases}$$

[3]:
```
r0 = 0.02 # valeur initial du taux sans risque
r = r0*np.ones((n+1,N))

a = 0.05
b = 0.03
gam = 0.1

S00 = 1 # valeur initiale de l'actif sans risque
S0t = S00*np.ones((n+1,N)) # matrice de l'actif sans risque
```

```
for j in range(N):
    for i in range(1,n+1):
        #euler scheme r
        # r0 = 0.02
        # r_ti = r_t(i-1) + 0.05*(0,03 - r_t(i-1))*T/n + 0.1*sqrt(T/n) * Gi avec␣
 ↪G_i iid N(0,1)
        # on a le même mouvement Brownien que la partie 1 donc on n'a pas besoin␣
 ↪de resimuler le mouvement
        r[i,j]=r[i-1,j]+a*(b-r[i-1,j])*step+gam*(B[i,j]-B[i-1,j])

        #euler scheme S^0
        # S^0_0 = 1 = S00
        # S^0_ti = S^0_t(i-1) + r_t(i-1)*S^0_t(i-1)*T/n
        S0t[i,j]=S0t[i-1,j]+r[i-1,j]*S0t[i-1,j]*step


dates=np.linspace(0,T,n+1)
plt.plot(dates,S0t) # possible trajectoire de S0t
plt.title("Possible trajectoire de l'actif sans risque")
plt.xlabel("Dates discrètes")
plt.ylabel("Dynamique de l'actif sans risque")
plt.show()
```



4 PARTIE 3

W un mouvement Brownien standard indépendant de B

W est un mouvement Brownien standard donc par définition

$$\begin{cases} W_0 = 0 \\ W_{t_i^n} = W_{t_{i-1}^n} + \sqrt{\frac{T}{n}} \times H_i \quad , \quad (H_i)_{i=1}^n \text{ sont iid et suivent une loi } N(0,1) \end{cases}$$

Afin de discrétiser le processus $\tilde{S}$, on a:

$$d\tilde{S}_t = \sigma_t \tilde{S}_t dW_t$$
$$\Longleftrightarrow$$
$$\tilde{S}_{t_i^n} - \tilde{S}_{t_{i-1}^n} = \sigma_{t_{i-1}^n} \tilde{S}_{t_{i-1}^n} \left(W_{t_i^n} - W_{t_{i-1}^n} \right)$$

- Euler scheme

$$\begin{cases} \tilde{S}_0 = 25 \\ \tilde{S}_{t_i^n} = \tilde{S}_{t_{i-1}^n} + \sigma_{t_{i-1}^n} \tilde{S}_{t_{i-1}^n} \sqrt{\frac{T}{n}} \times H_i \end{cases}$$

Calcul des trajectoires de S aux dates discrètes, on a:

$$\tilde{S}_t = \frac{S_t}{S_t^0}$$
$$\Longrightarrow S_t = \tilde{S}_t \times S_t^0$$
$$\Longrightarrow S_{t_i^n} = \tilde{S_{t_i^n}} \times S_{t_i^n}^0$$

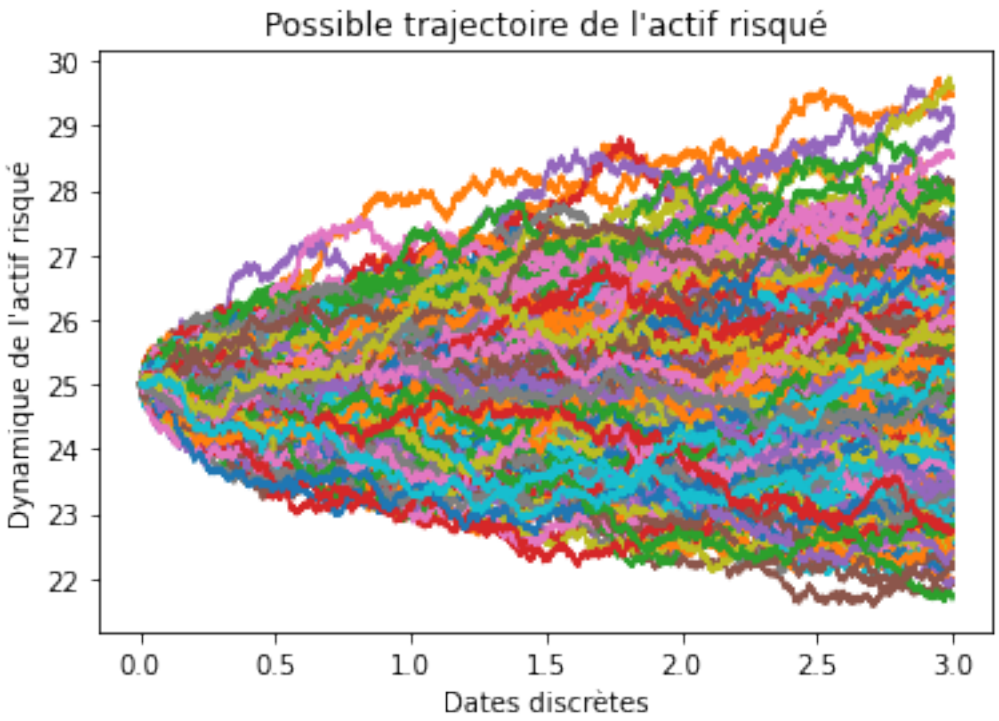
donc

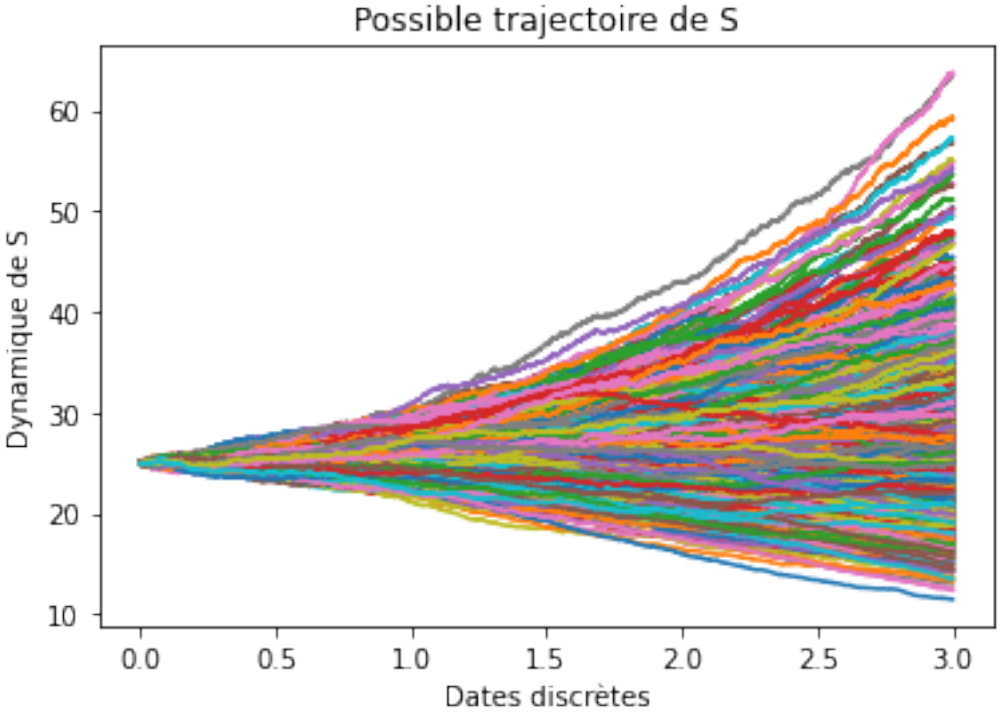
$$S_{t_i^n} = \left(\tilde{S}_{t_{i-1}^n} + \sigma_{t_{i-1}^n} \tilde{S}_{t_{i-1}^n} \sqrt{\frac{T}{n}} \times H_i \right) \times \left(S_{t_{i-1}^n}^0 + r_{t_{i-1}^n} \times S_{t_{i-1}^n}^0 \times \frac{T}{n} \right)$$

Code en Python et représentations graphiques de S

[4]:
```
W = np.zeros((n+1,N))

tildeS0=25 # valeur initiale de l'actif risqué

tildeS=tildeS0*np.ones((n+1,N))
S=tildeS0*np.ones((n+1,N))

for j in range(N):
    for i in range(1,n+1):
        # euler scheme W
        # W0 = 0
        # W_ti = W_{ti-1} + sqrt(variance) * H_i
```

```
        # ici variance = T/n = step et avec H_i = loi normale
        W[i,j]=W[i-1,j]+np.sqrt(step)*sim.randn()

        #euler scheme
        # tildeS0 = S0 given
        # tildeS_ti = tildeS_t(i-1) + sigma(U_t(i-1))*tildeS_t(i-1)) * sqrt(T/n)
 ↪* H_i avec H_i iid N(0,1)
        tildeS[i,j]=tildeS[i-1,j]+sigma(U[i-1,j])*tildeS[i-1,j]*(W[i,j]-W[i-1,j])

        # tildeS_t = S_t / S^0_t par definition
        # S_t = S0_t*tildeS_t
        S[i,j]=S0t[i,j]*tildeS[i,j]


dates=np.linspace(0,T,n+1)

plt.plot(dates,tildeS) # possible trajectoires de tildeS
plt.title("Possible trajectoire de l'actif risqué")
plt.xlabel("Dates discrètes")
plt.ylabel("Dynamique de l'actif risqué")
plt.show()

plt.plot(dates,S) # possible trajectoire de S
plt.title("Possible trajectoire de S")
plt.xlabel("Dates discrètes")
plt.ylabel("Dynamique de S")
plt.show()
```


Possible trajectoire de l'actif risqué
Dynamique de l'actif risqué
30
29
28
27
26
25
24
23
22
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Dates discrètes


Possible trajectoire de S
Dynamique de S
60
50
40
30
20
10
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Dates discrètes

5 PARTIE 4

On considère les options de maturité T et de payoffs terminaux respectifs:

$$\xi_T^1 = (K_1 - S_T)^+ \quad , \quad K_1 = 30$$

$$\xi_T^2 = \frac{1}{T}\left(\int_0^T S_u du - K_2\right)^+ \quad , \quad K_2 = 15$$

5.0.1 Principe de pricing sous probabilité de risque neutre et en marché complet

On dit que le marché est complet si tout produit dérivé est réplicable par une stratégie de portefeuille auto-finançante.

On appelle probabilité risque-neutre toute probabilité équivalente $\mathbb{Q}$ telle que $\tilde{S} = (\tilde{S}\_t)\_\{t\{0,1\}\}$ soit une martingale sous $\mathbb{Q}$.

En l'absence d'opportunité d'arbitrage, le marché est complet. Autrement dit, tout produit dérivé de payoff ξ est réplicable par une stratégie auto-finançante.

Soit ξ_T une variable aléatoire F_T -mesurable, interpreté comme un payoff d'une certaine option. Un prix pour ξ_T est la valeur initiale V_0 d'un processus de portefeuille auto-finançant V tel que $V_T \geq \xi_T$ presque surement. V_T est un prix de sur-réplication.

Si nous avons l'absence d'opportunité d'arbitrage alors

$$\text{le marché est complet} \iff \text{la probabiité risque neutre est unique}$$

Dans ce cas, pour tout $\xi_T, \mathbb{Q}$ - integrable il existe un unique processus de portefeuille auto-finançante V tel que $V_T = \xi_T$.

5.0.2 Principe mathématique par lequel on peut approximer un prix : Loi des grands nombres

Soit $\left(\frac{\xi_T^i}{S_T^0}\right)_{i=1...N}$ une suite de variable aléatoire intégrable et iid sous la loi $\mathbb{Q}$ alors $\frac{1}{N}\left(\sum_{i=1}^N \frac{\xi_T^i}{S_T^0}\right) \underset{N\to+\infty}{\longrightarrow} \mathbb{E}_{\mathbb{Q}}\left(\frac{\xi_T}{S_T^0}\right)$

5.0.3 Code Python pour évaluer les prix des options considérées

On a:

$$\xi_T^2 = \frac{1}{T}\left(\int_0^T S_u du - K_2\right)^+$$

et

$$\int_0^T S_u du = \sum_{i=1}^n \int_{t_{i-1}^n}^{t_i^n} S_u du$$

et

$$\int_{t_{i-1}^n}^{t_i^n} S_u du \simeq \int_{t_{i-1}^n}^{t_i^n} S_{t_{i-1}^n} du \tag{1}$$

$$= S_{t_{i-1}^n} \int_{t_{i-1}^n}^{t_i^n} du \tag{2}$$

$$= S_{t_{i-1}^n} \left(t_i^n - t_{i-1}^n\right) \tag{3}$$

$$= S_{t_{i-1}^n} \frac{T}{n} \tag{4}$$

$$\Longrightarrow \int_0^T S_u du = \lim_{n\to+\infty} \sum_{i=1}^{n} S_{t_{i-1}^n} \frac{T}{n} \tag{5}$$

$$= \lim_{n\to+\infty} \frac{T}{n} \sum_{i=1}^{n} S_{t_{i-1}^n} \tag{6}$$

$$\Longrightarrow \frac{1}{T} \int_0^T S_u du = \lim_{n\to+\infty} \frac{1}{n} \sum_{i=1}^{n} S_{t_{i-1}^n}$$

$$\Longrightarrow \xi_T^2 = \frac{1}{T} \left(\int_0^T S_u du - K_2 \right)^+ \tag{7}$$

$$\simeq \left(\frac{1}{n} \sum_{i=1}^{n} S_{t_{i-1}^n} - K_2 \right)^+ \tag{8}$$

[5]:
```
K1 = 30 # strike
K2 = 15

payoff=[]
payoff1=[]

#valeur intrinsèque du put et du call
def put(x,k):
    return (max(k-x,0))

def call(x,k):
    return (max(x-k,0))

for j in range(N):
    # pour xi 1
    payoff.append(put(S[n,j],K1)/S0t[n,j])

    # pour xi 2
    # on veut mtn la moyenne de la trajectoire j pour appliquer
```

```
    # (moyenne colonne j - K2)+ ce qui donne xi_T de la colonne j
    averagetraj=np.mean(S[:,j])
    payoff1.append(call(averagetraj,K2)/S0t[n,j])

V1=np.mean(payoff) #payoff du put
V2=np.mean(payoff1) #payoff du call

print("payoff du put =",V1)
print("payoff du call asiatique =",V2)
```

```
payoff du put = 5.551476256643226
payoff du call asiatique = 10.074239386421711
```