Mini projet 1 : Calcul du prix d'une option asiatique

Valentin DE CRESPIN DE BILLY

Matthias LANG

30.11.2021

N. d'étudiant : 247067 et 313411

Université Catholique de l'Ouest

Mathématiques financières

1 Calculer le prix du sous-jacent

Nous avons essayé d'atteindre une équation qui ne dèpend que des variables connues comme la formule de Black-Scholes. Cela n'a pas fonctionné.

$$dS_t = S_t(rdt + \sigma\sqrt{S_t}dW_t) \tag{1}$$

$$\iff \frac{dS_t}{S_t} = rdt + \sigma\sqrt{S_t}dW_t \tag{2}$$

On prend l'équation 1 :

$$= dS_t \;=\; S_t rdt + \sigma S_t^{1.5} dW_t \quad \text{; Puis}$$

$$d\langle S_t,\ S_t\rangle$$
$$= \langle dS_t,\ dS_t\rangle \;=$$
$$= \langle S_t rdt + \sigma S_t^{1.5} dW_t,\ S_t rdt + \sigma S_t^{1.5} dW_t\rangle \;=$$
$$= \langle \sigma S_t^{1.5} dW_t,\ \sigma S_t^{1.5} dW_t\rangle \;=$$
$$= S_t^3 \sigma^2 \langle dW_t,\ dW_t\rangle \;=$$
$$= S_t^3 \sigma^2 dt$$

On pose : $X_t = ln(S_t)$

$$\text{Formule d'Ito : } dln(S_t) = \frac{dS_t}{S_t} + \frac{1}{2}\frac{-1}{S_t^2} d\langle S_t,\ S_t\rangle \tag{3}$$

$$\tag{4}$$

Avec les équations 2 et 3 :

$$dln(S_t) = rdt + \sigma\sqrt{S_t}dW_t - \frac{1}{2}S_t\sigma^2 dt = (r - \frac{1}{2}S_t\sigma^2)dt + \sigma\sqrt{S_t}dW_t \tag{5}$$

$$\begin{aligned} ln(\frac{S_t}{S_0}) &= ln(S_t) - ln(S_0) = \int_0^t dln(S_u) = \\ &= \int_0^t (r - \frac{1}{2} S_t \sigma^2) du \; + \; \int_0^t \sigma \sqrt{S_t} dW_t \\ &= \dots \end{aligned}$$

Donc on ne peut pas facilement dériver une formule pour le prix comme ça, qui dépend que des variables fixées, mais on peut le simuler pas à pas en utilisant (1) :

$$\begin{aligned} S_0 & \text{ soit connu} \\ dS_0 &= S_0(rdt + \sigma\sqrt{S_0}dW_0) \\ S_1 &\approx S_0 + dS_0 \\ dS_1 &= S_1(rdt + \sigma\sqrt{S_1}dW_1) \\ S_2 &\approx S_1 + dS_1 \\ & \dots \end{aligned} \tag{6}$$

1.1 ecdf

Sieht Gaussian aus -> KI damit probieren, aber keine richtige Normalverteilung, also bootstrap !

1.2 Réduction de la variance du éstimateur

Les éstimateurs ont une variance telle que : $\hat{Var}(C) = \hat{\sigma_i^2}/n_t$, oú n_t est le nombre des observations et $\hat{\sigma_i^2}$ est la variance estimée de la population, qui est égal à la variance de l'échantillon.

Supposons que nous ne connaissions ni les paramètres ni la règle à partir desquels les prix sont établis. Nous ne pouvons donc pas augmenter le nombre d'observations pour améliorer l'estimateur. Quelle autre possibilité existe-t-il pour réduire sa variance ?

Avec les techniques de bootstrap on pourrait répliquer les données. Mais on risque de introduir un biais. Si on utilise une variable de contrôle on n'invente pas des nouvelles données, ni risque-t-on de changer l'ésperance.

2 Réalisation numerique

Les algorithmes sont réalisées avec deux langues de programmation : Matlab et Visual Basic for Applications. Plusieurs graphiques sont y générés, vous les trouverez dans l'annexe **??**. En plus, avec le logiciel Excel nous avons crée un dashboard, voir une capture d'figure refX. Vous trouverez les scriptes et les images dans l'annexe, et avec la fiche de dashboard également dans le repository.

Appendices

Toutes les fiches se trouvent dans le repository en ligne :

`https://github.com/matthias-10/UCO_actuariat_mini-projet`

A Graphiques

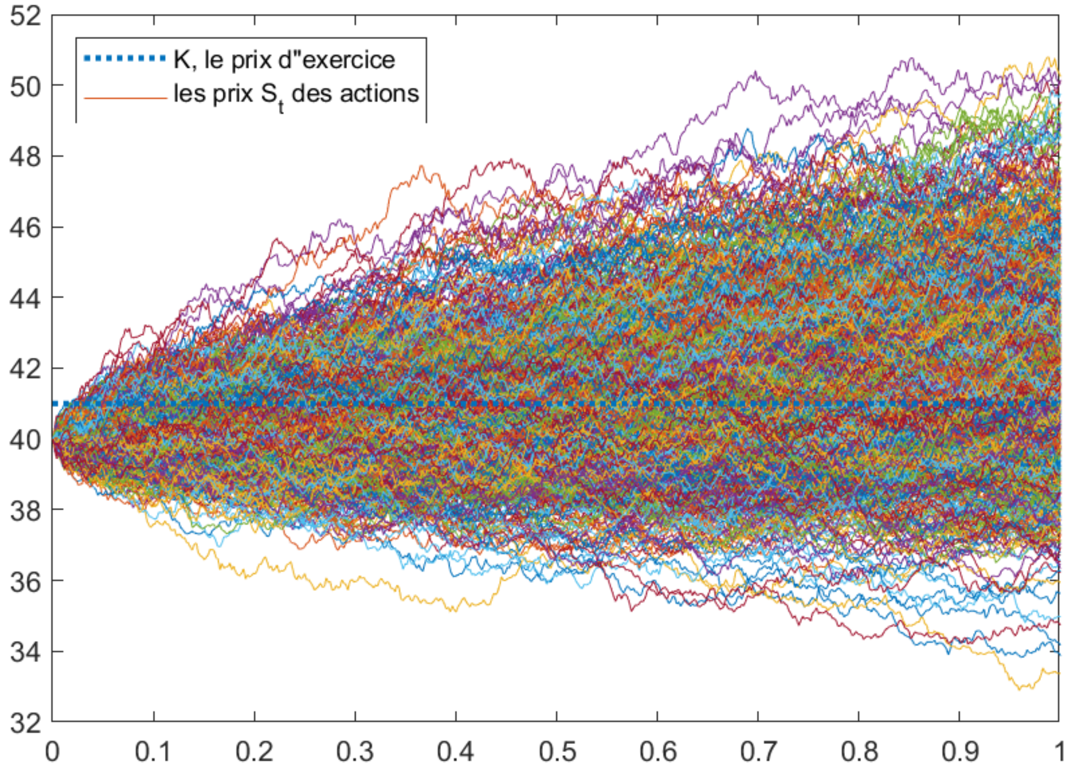

FIGURE 1 – Les graphes de tous trajectoires, plotés avec matlab

B Code Matlab

```
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ %
% Valentin DE CRESPIN DE BILLY                     UTF-8 %
% Matthias LANG                               30.11.2021 %
% requires:                                              %
% - Statistics and Machine Learning Toolbox              %
% - Symbolic Math Toolbox                                %
% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ %
```

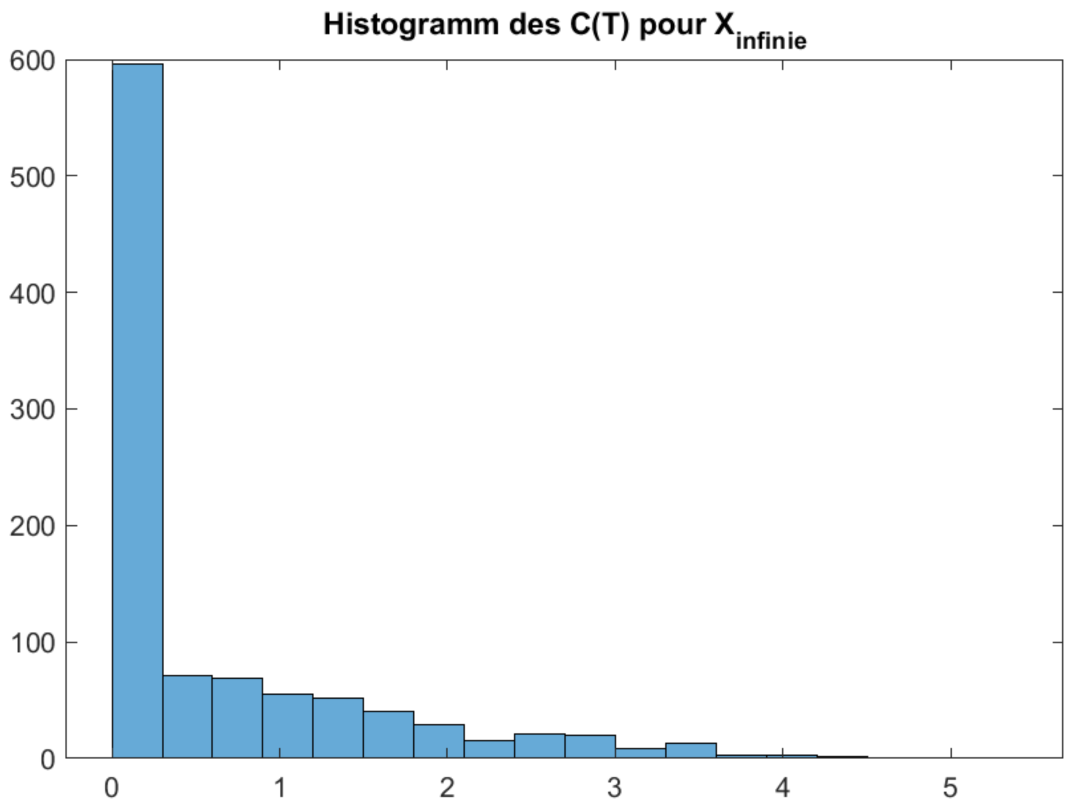

FIGURE 2 – Histogramme des simulations pour C_∞

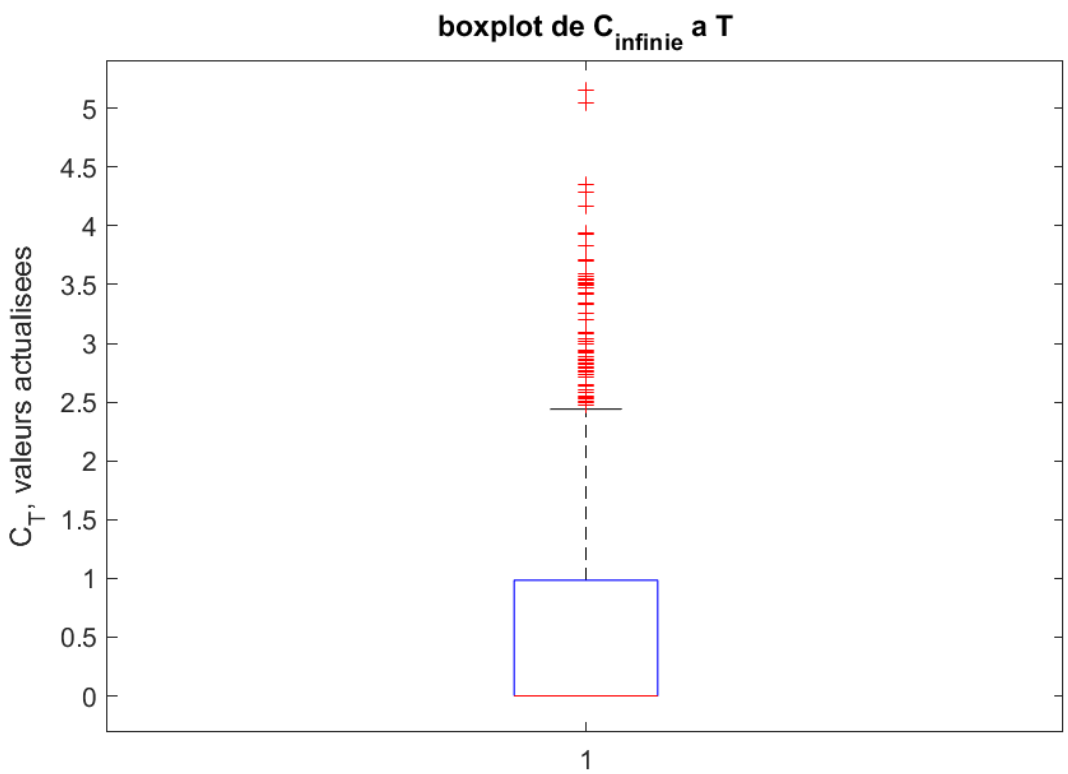

FIGURE 3 – Boxplot des simulations pour C_∞

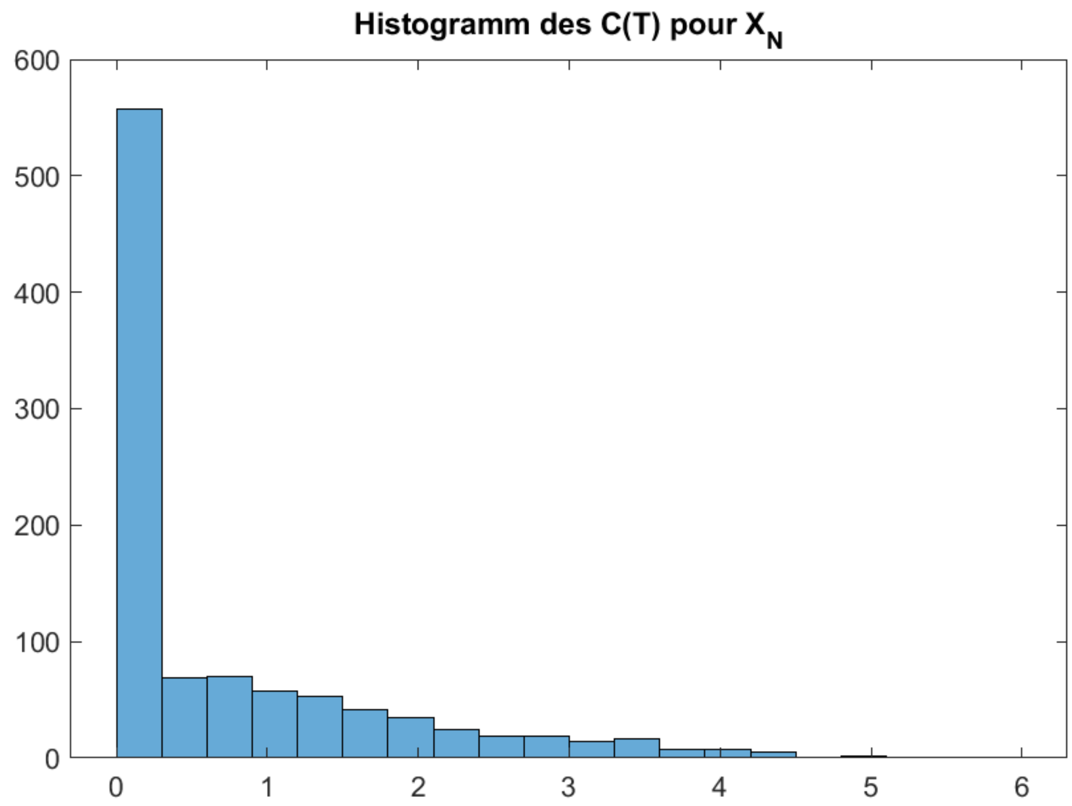

FIGURE 4 – Histogramme des simulations pour C_N

```
% ~~~~~~ Mathematiques financieres: Mini-projet 1 ~~~~~~~ %

%% ~~~~~~~~~~~~~~~~~~~~ Parametres ~~~~~~~~~~~~~~~~~~~~~ %%

S0 = 40;                % Prix initial du sous jacent
K = 41;                 % Prix d'exercice de l'option

r = 0.05;               % Taux d'interet sous risque neutre
sigma = 0.01;           % Variance partie fixe

t0 = 0;                 % Debut de la periode
n = 2^9;                % Nombre de intervalles
T = 1;                  % Fin de la periode
Nd = 8;                 % Nombre des sous-intervalles

nt = 1000;              % Nombre de trajectoires
```

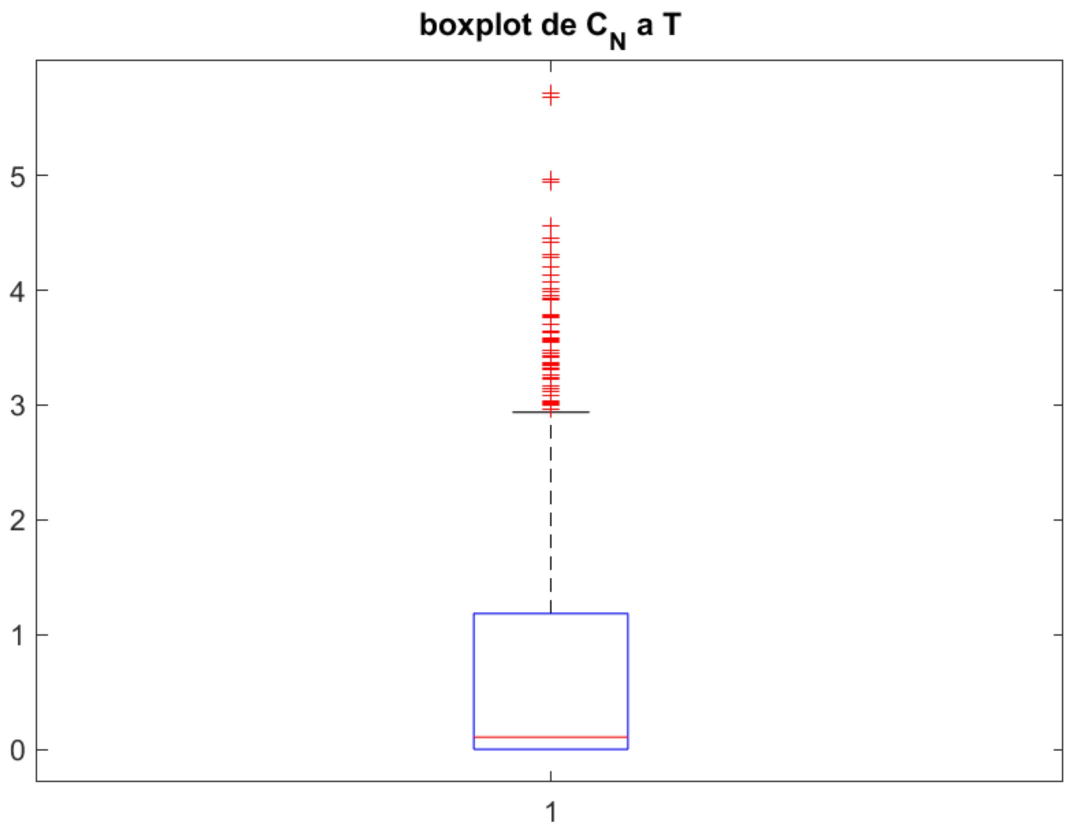

FIGURE 5 – Boxplot des simulations pour C_N

```


%% ~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~~ %%

if Nd > n/2-1
    warning("Le nombre des sous-intervalles est trop petit")
    fprintf('Il fallait Nd << n')
end

starttime = datetime('now');
fprintf('\n ~ ~ ~ ~ ~ ~ ~ ~ ~ ~ \n');
fprintf('La programme a demarre a %s \n', starttime);
fprintf('%d -> Prix initial du sous jacent \n', S0)

%1% syms func(x) %1% requires Symbolic Math Toolbox.
%1% obligation(x) = S0*(1+r)^(x-t0);
```

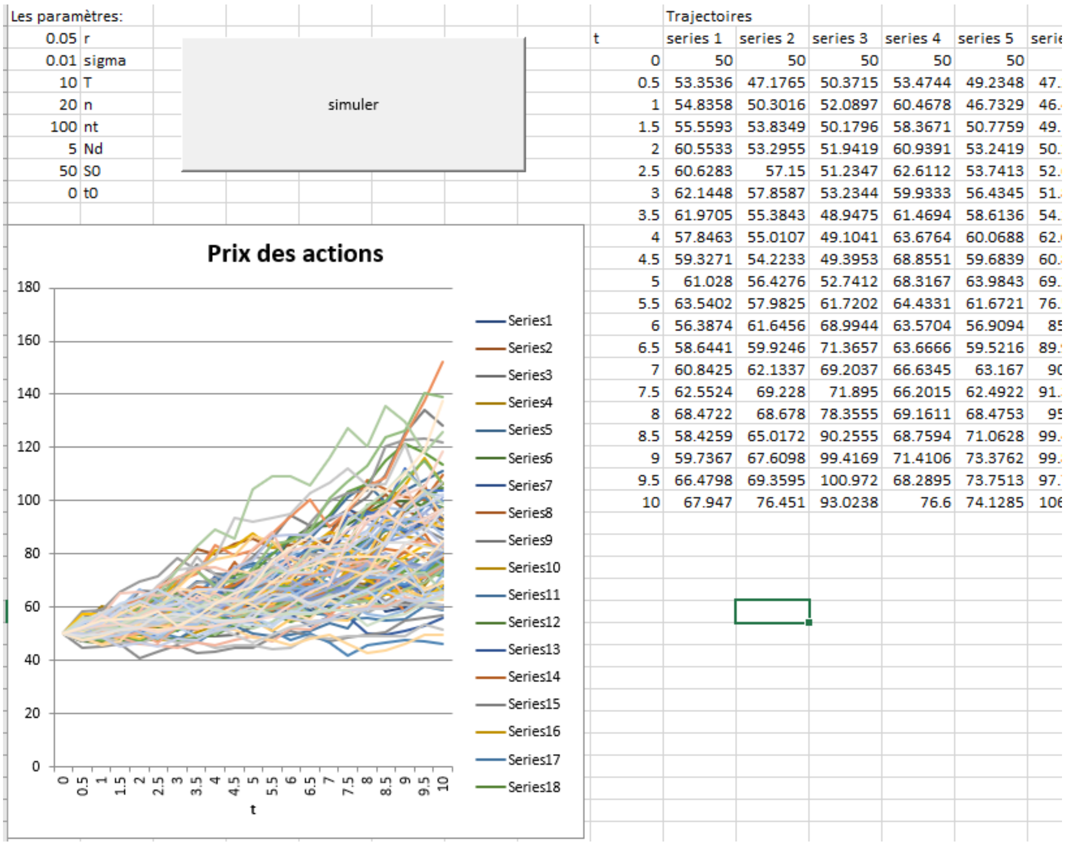

FIGURE 6 – le Excel dashboard

```
%K = int(obligation,t0,T)/(T-t0);
%1% bonds_T = obligation(T);
%1% fprintf('%0.5g -> Prix d''une obligation a T\n',bonds_T)

fprintf('%0.5g -> Prix d''exercice de l''option \n', K);
fprintf(' . . .\n')
tic



%% ~~~~~~~~~~~~~~~~~~~~ Simulation ~~~~~~~~~~~~~~~~~~~~~ %%

dt = (T-t0)/n;
t = t0:dt:T;

% les premieres nt_a valeurs pour l'affichage plus tard
nt_a = 15;
```

```
S = zeros(nt_a, n+1);


%% ~~~~~~~~~~~~~~~~ prix de l'option C ~~~~~~~~~~~~~~~~~ %%

C_inf = zeros(nt,1);
C_N = zeros(nt,1);
X_inf_v = zeros(nt,1);
X_N_v = zeros(nt,1);
for j = 1:nt
    S_vec = zeros(1, n);
    S_vec(:, 1) = S0;


    %% ~~~~~~~~~~~~~ simuler pas a pas ~~~~~~~~~~~~~~~~~ %%

    for i = 2:(n+1)
        dW_t = normrnd(0,sqrt(dt));
        dSi = S_vec(i-1)* ...
              ( r*dt + sigma*sqrt(S_vec(i-1))*dW_t );
        S_vec(i) = S_vec(i-1) + dSi;
    end

    % sauvegarder les premieres nt_a actions
    if j <= nt_a
        S(j,:) = S_vec;
    end


    %% ~~~~~~~~~~~~ C_inf: calcul avec X_T ~~~~~~~~~~~~~ %%

    % integral: l'aire de t0 a T sous S
    X_T = 0.5*S0 + sum(S_vec(2:n)) + 0.5*S_vec(n+1);
    X_T = X_T/n; %ou (n+1)?
```

```
    C_inf_j = (X_T - K) * ( X_T - K >= 0 );
    C_inf_0 = exp(-r*T)*C_inf_j;

    % ~ Estimateur ~
    % C_inf * exp(-rT) est une martingale donc
    % E[exp(-rT)*C_inf]= C_inf(S_0)

    X_inf_v(j)=X_T;
    C_inf(j)=C_inf_0;



    %% ~~~~~~~~~~ C_N: calcul avec X_T_prim ~~~~~~~~~~~~ %%

    %1/N * sum_1^N S_{kT/N}
    % => kT n'est pas un numero entier, il faut arrondir

    index = fliplr(1:n);
    warn_id = 'MATLAB:colon:nonIntegerIndex';
    warning('off', warn_id);
    % ^supprime Warning a cause de arrondir:
    index = index(1:(n/Nd):end);
    X_T_prim = sum(S_vec(index))/Nd;

    C_N_j = (X_T_prim - K) * ( X_T_prim - K >= 0 );

    % C_N * exp(-rT) est une martingale donc
    % E[exp(-rT)*C_N]= C_N(S_0)
    C_N_0 = exp(-r*T)*C_N_j;

    X_N_v(j)=X_T_prim;
    C_N(j)=C_N_0;

end
```

```
%% ~~~~~~~~~~~~~~~~~~~ estimateurs ~~~~~~~~~~~~~~~~~~~~~ %%

% C_inf
C_inf_est = mean(C_inf);
C_inf_est_var = var(C_inf)/nt; %/nt ?

% X_inf
X_inf_mu = mean(X_inf_v);

% C_N
C_N_est = mean(C_N);
C_N_est_var = var(C_N)/nt; %/nt?

% X_N
X_N_mu = mean(X_N_v);


%% ~~~~~~~ estimateurs et intervalle de confiance ~~~~~~ %%
%                  (seulement pour C_inf)                %

alpha = 0.05; % niveau au risque

v = nt/(nt-1)*var(X_inf_v); %variance d'echantillonnage

%%% variable normale
IC_gauss = [X_inf_mu - sqrt(v/nt)*norminv(1-alpha/2) ...
            X_inf_mu - sqrt(v/nt)*norminv(alpha/2) ];

%%% bootstrap
% sims = 10^4
%
% simulate = function(){
```

```
%   return(mean(sample(expenditures, n, replace = T))- muhat)
% }
%
% dat = replicate(sims, simulate())
% KI_bootstrapp = muhat - quantile(dat, c(1-alpha/2, alpha/2))


%% ~~~~~~~~~~~~~~ variable de controle ~~~~~~~~~~~~~~~~~ %%

% a faire


%% ~~~~~~~~~~~~ affichage des estimateurs ~~~~~~~~~~~~~~ %%

duree= toc;
fprintf('\n')
fprintf('%d trajectoires simules\n', nt);
fprintf('Fini en %0.5g\n', duree);
fprintf('\n')

fprintf('Les estimateurs Monte-Carlo:\n')

fprintf('L''estimateur du C_inf a t0 = \n%0.5g\n', ...
 C_inf_est);
fprintf('Son ecart type = %0.5g\n', sqrt(C_inf_est_var));

fprintf(['L''estimateur du C_N a t0, avec ' ...
    '%d sous-intervalles = \n%0.5g\n'], ...
    Nd, C_N_est);
fprintf('Son ecart type = %0.5g\n', sqrt(C_N_est_var));


%% ~~~~~~~~~~~~~~~~~~~~~ graphes ~~~~~~~~~~~~~~~~~~~~~~~ %%

```

```
% 1:   graphes de S;
% 2-3: ecdf de C_inf et C_N;
% 4-5: boxplot des estimateurs

G = "g";
P = input(['\n' ...
    'Pour afficher n''importe quel graphique, tapez ' ...
    'son numero <1-5> ou [Enter]. \n' ...
    'Pour quitter tapez plusieures fois [Enter]:\n'] );

if isstring(P) || isempty(P)
    P = 1;
else
    if ~ismember(P,1:6)
        P = 1;
    end
end

while G~="q"
    disp("[Enter] pour continuer")
    switch P
    case 1
        fprintf('< 1: quelques premiers graphes de S >\n')
        figure(1)
        plot([t0 T],[K K], ':k', 'LineWidth',2)
        hold on
        plot(t, S)
        plot([t0 T],[K K], ':k', 'LineWidth',2)
        hold off
        % pour comparison, si j'epargne pour le taux r:
        %plot([t0 T], [S0 S0*(1+r)^(T-t0)],"--k"); %obl.
        %1% fplot(obligation, [t0 T], "-k");
        legend("K, le prix d''exercice", ...
               "les prix S_t des actions",...
```

```
               "Location","northwest");
        P=P+1; input('\n');

    case 2
        fprintf(['< 2: fonction de distribution ' ...
            'cumulative estime'  ...
            '\n C(T) pour X_{infinie} de C_infinie >\n'])
        figure(1)
        % E_\pi (e^-rT (X_T - K)^+ / F_O) ~ 1/nt \sum{C(T)}
        %histogram( C_inf );
        ecdf( X_inf_v );
        hold on
        plot([K K],[0 1], 'k')
        plot([min(X_inf_v) max(X_inf_v)], [.5 .5],':b')
        x = [min(X_inf_v):.1:max(X_inf_v)];
        nor = normcdf(x,X_inf_mu,v);
        plot(x,nor,':r')
        hold off
        legend("ecdf", "K", "P=50%", "cdf normal")
        title("ecdf X(T) pour X_{infinie}");
        P=P+1; input('\n');

    case 3
        fprintf(['< 3: fonction de distribution ' ...
            'cumulative estime'  ...
            '\n C(T) pour X_{infinie} de C_N >\n'])
        figure(1)
        ecdf( X_N_v );
        hold on
        plot([K K],[0 1], 'k')
        plot([min(X_N_v) max(X_N_v)], [.5 .5],':b')
        hold off
        legend("ecdf", "K", "P=50%")
        title("ecdf X(T) pour X_{N}");
```

```
        P=P+1; input('\n');

    case 4
        fprintf(['< 4: boxplot de l''estimateur ' ...
                 'C_{infinie} >\n'])
        figure(1)
        boxplot( C_inf );
        title('boxplot de C_{infinie} a T')
        ylabel('C_T, valeurs actualisees')
        P=P+1; input('\n');

    case 5
        fprintf('< 5: boxplot de l''estimateur C_{N} >\n\n')
        figure(1)
        boxplot ( C_N );
        title('boxplot de C_{N} a T')
        P=P+1;

    case 6
        P=input([' ~ ~ ~ ~ ~ ~\n ' ...
            'Pour afficher n''importe quel graphique, ' ...
            'tapez son numero <1-5> \n']);
        if ismember(P, 1:5)
            fprintf("Vous avez choisi: ")
        else
            G="q";
        end
    otherwise
        G="q";
    end
end
```

C Code VBA

```
Sub Macro1()

' parametres
Dim T, n, nt, Nd As Integer
Dim r, sigma, S0, t0 As Double

r = Range("A2").Value
sigma = Range("A3").Value
T = Range("A4").Value
n = Range("A5").Value
nt = Range("A6").Value
Nd = Range("A7").Value
S0 = Range("A8").Value
t0 = Range("A9").Value

Dim dt As Double
dt = ((T - t0) / n)

' premier cellule de la table de trajectoires ~ t0
Dim Srow, Scol As Integer
Dim Scol_abc As String
Srow = 3
Scol_abc = "I"
Scol = 9

' worksheets
Dim sh_dash, sh_calc, sh_s As String
Dim sh_dash_o As Worksheet
sh_s = "Dashboard"
sh_dash = "Dashboard"
Set sh_dash_o = Worksheets(sh_dash)

' iteratives
Dim i, j As Integer
```

```
Dim S() As Double
ReDim S(1 To n + 1, 1 To nt)
Dim dW As Double
Dim dS As Double
Dim x As Double

'effacer t et S() aines
With Worksheets(sh_s)
    Range(.Cells(Srow - 1, Scol), .Cells(Srow + 10000,
       Scol + 10000)).Delete
End With

' afficher t
Dim temps() As Double
ReDim temps(n + 2)
temps(0) = t0
For j = 1 To n + 2
    temps(j) = temps(j - 1) + dt
Next

Range(Scol_abc & Srow & ":" & Scol_abc & UBound(temps) +
   1) = _
    WorksheetFunction.Transpose(temps)

'simuler et afficher S pas a pas
Cells(1 + 1, 9 + 0).Value = "t"
For j = 1 To nt
    x = S0
    i = 1
    Cells(2 + i, 9 + j).Value = x
    Cells(1 + i, 9 + j).Value = "series " & j
    For i = 1 To n + 1
        If i > 1 Then
```

```
            dW = Sqr(-2 * Log(Rnd())) * Cos(6.283185307 * Rnd()) * Sqr(dt)
            'dS = S(i - 1, j) * (r * dt + sigma * Sqr(S(i - 1, j)) * dW)
            'aine = Cells(1 + i, 10 + j).Value
            dS = x * (r * dt + sigma * Sqr(x) * dW)
            'S(i, j) = S(i - 1, j) + dS
            x = x + dS 'S(i - 1, j) + dS
            Cells(2 + i, 9 + j).Value = x
        End If
        S(i, j) = x
    Next
Next
'Range("J21:O100") = S()


' insert Chart
Worksheets(sh_s).Activate
Dim chartrange As Range
Set chartrange = Cells(Srow, Scol + 1)  'sans t
Set chartrange = chartrange.Resize(n + 1, nt)
MsgBox chartrange.Address

Worksheets(sh_dash).Activate
Dim Graphe As Object

'effacer graphes aines
For Each Graphe In ActiveSheet.ChartObjects
  Graphe.Delete
Next Graphe

Set Graphe = sh_dash_o.ChartObjects.Add( _
    Left:=Range("A11").Left, Width:=380, _
    Top:=Range("A11").Top, Height:=400)
```

```
With Graphe.Chart
    .SetSourceData chartrange
    .PlotBy = xlColumns 'echanger x et y axes
    .ChartType = xlLine
    .HasTitle = True
    .ChartTitle.Text = "Prix des actions"
    .FullSeriesCollection(1).XValues = _
        Range(Scol_abc & Srow & ":" & Scol_abc & UBound(
            temps) + 1)
    .Axes(xlCategory).HasTitle = True
    .Axes(xlCategory).AxisTitle.Text = "t"
End With


MsgBox "Simulation finie pour " & nt & " trajectoires."


'Sheets("Dashboard").Activate
'Range("Parametres").Select
'Range("A13").Value = T


End Sub
```