

Université de Provence

Prépa Agreg . Option Proba.
Correction du TP sur Monte-Carlo

1 Programmes.

Le programme ci-dessous rassemble l'ensemble des exercices.

```
% CALL.M
%
% Differentes methodes de Monte Carlo pour la calcul du call
% pour  $\beta=1$  et  $K=1$ .
%
T=ones(1,5); %T contiendra les temps pris par les differentes methodes.

% Calcul de la valeur theorique du call et du put.
tic
Ct=erfc(1/sqrt(2))/2;
Ct=1-Ct;
Ct=exp(1/2)*Ct-0.5;
t=toc;
T(1)=t;
Pt=Ct+1-exp(0.5);

% Premiere methode d'approximation du call.
N=1000; %Nombre de simulations
rand('state',0) % Remise du generateur de nombres aleatoires a zero en vue de
                % comparaison des differentes methodes.
tic
X=randn(1,N);
X=exp(X)-1; % Simulation du call
C=X.*(X >=0);
I1=cumsum(C)./ [1:N]; % Calcul de l'approximation Monte-Carlo.
t=toc;
T(2)=t;
dev=cumsum(C.^2)./ [1:N]; % Calcul des bornes de l'intervalles de confiance.
dev=dev-I1.^2;
dev=sqrt(dev)./sqrt([1:N]);
dev=1.96*dev;
Bi1= I1 -dev;
Bs1=I1+dev;
plot([1:N],I1,[1 N], [Ct Ct], 'r', [1:N], Bi1,'g', [1:N], Bs1,'g')
```

```

title('Approximation Monte-Carlo du call et du put')
xlabel('Nombre de simulations')
legend('Approximation Monte Carlo' , 'Vraie valeur' , 'Bornes de l''IC a 0.95')

% Comparaison avec le put.
rand('state',0) % Remise du generateur de nombres aleatoires a zero en vue de
                % comparaison des differentes methodes.

tic
X=randn(1,N);
X=exp(X)-1;
P=X.*(X<=0);
P=-P;
Phat=cumsum(P)./ [1:N]; % Calcul de l'approximation Monte-Carlo.
t=toc;
T(4)=toc;
dev=cumsum(P.^2)./ [1:N]; % Calcul des bornes de l'intervalles de confiance.
dev=dev-Phat.^2;
dev=sqrt(dev)./sqrt([1:N]);
dev=1.96*dev;
Bip= Phat -dev;
Bsp=Phat +dev;
hold on
plot([1:N],Phat,'-.',[1 N], [Pt Pt], '-.r', [1:N], Bip,'-.g', [1:N], Bsp,'-.g')
hold off

% Methode 2. Echantillonnage preferentiel.
%
rand('state',0) % Remise du generateur de nombres aleatoires a zero en vue de
                % comparaison des differentes methodes.

tic
X=rand(1,N);
X=-log(X); % Generation de variables exponentielles.
C=sqrt(2*X);
C=(exp(C)-1)./C;
C=C/sqrt(2*pi);
I2=cumsum(C)./ [1:N]; % Calcul de l'approximation Monte-Carlo.
t=toc;
T(3)=t;
dev=cumsum(C.^2)./ [1:N]; % Calcul des bornes de l'intervalles de confiance.
dev=dev-I2.^2;
dev=sqrt(dev)./sqrt([1:N]);
dev=1.96*dev;
Bi2= I2 -dev;
Bs2=I2+dev;

```

```

% Methode 3. Variables de controle.
dif=exp(0.5)-1;
Bi3=Bip+dif;
Bs3=Bsp+dif;

% Methode 4. Variables antithetiques.
rand('state',0) % Remise du generateur de nombres aleatoires a zero en vue de
                % comparaison des differentes methodes.

tic
X=randn(1,N);
C=exp(X);
test=(C>=1);
C=(C-1).*test+(1./C-1).*(1-test);
C=C/2;
I4=cumsum(C)./ [1:N]; % Calcul de l'approximation Monte-Carlo.
t=toc;
T(5)=t;
dev=cumsum(C.^2)./ [1:N]; % Calcul des bornes de l'intervalles de confiance.
dev=dev-I4.^2;
dev=sqrt(dev)./sqrt([1:N]);
dev=1.96*dev;
Bi4= I4 -dev;
Bs4=I4+dev;

% Representation graphique des resultats.
figure(2)
plot([1 N],[Ct Ct], 'r', [1:N], Bi1, 'g', [1:N], Bi2, '-.b', ...
... [1:N], Bi3, 'k--', [1:N], Bi4, 'm:')
hold on
plot([1:N], Bs1, 'g', [1:N], Bs2, '-.b', [1:N], Bs3, 'k--', ...
... [1:N], Bs4, 'm:')
title('Comparaison des approximations Monte-Carlo du call.')
xlabel('Nombre de simulations')
s0='vraie valeur';
s1='IC pour la methode standard';
s2='IC par echantillonnage preferentiel.';
s3='IC par variable de controle';
s4='IC par variable antithetique';
legend(s0,s1,s2,s3,s4)
hold off

% Affichage des temps.

```

```

disp([' Temps pris par les differentes methodes pour ' num2str(N) ..
..' simulations'])
disp(['Methode deterministe:      ' num2str(T(1))])
disp(['MC standard:              ' num2str(T(2))])
disp(['MC echantillonnage:       ' num2str(T(3))])
disp(['MC variable de controle:  ' num2str(T(4))])
disp(['MC variable antithetique: ' num2str(T(5))])

% Affichage des variances.
disp([' Precision pour les differentes methodes pour ' num2str(N) ..
..' simulations'])
disp(['MC standard:              ' num2str(Bs1(N)-Bi1(N))])
disp(['MC echantillonnage:       ' num2str(Bs2(N)-Bi2(N))])
disp(['MC variable de controle:  ' num2str(Bs3(N)-Bi3(N))])
disp(['MC variable antithetique: ' num2str(Bs4(N)-Bi4(N))])

```

2 Résultats.

```

>> call
    Temps pris par les differentes methodes pour 1000 simulations
Methode deterministe:      0.002179
MC standard:              0.002052
MC echantillonnage:       0.003355
MC variable de controle:  0.00313
MC variable antithetique: 0.002768
    Precision pour les differentes methodes pour 1000 simulations
MC standard:              0.23456
MC echantillonnage:       0.060099
MC variable de controle:  0.037418
MC variable antithetique: 0.13428

```

On compare ici les différentes méthodes en terme de temps calcul, et de précision de l'approximation.

La méthode dite déterministe correspond au calcul du “call” utilisant la fonction `erfc` de Matlab. Pour 1000 simulations, les temps calcul sont comparables pour les différentes méthodes. Evidemment, si on augmente le nombre de simulations, le temps-calcul augmente pour les méthodes de Monte-Carlo. Ainsi, pour 100000 simulations, on obtient

```

>> call
    Temps pris par les differentes methodes pour 100000 simulations
Methode deterministe:      0.003099
MC standard:              0.15386
MC echantillonnage:       0.29897

```

MC variable de controle: 0.18159
 MC variable antithetique: 0.42058

Precision pour les differentes methodes pour 100000 simulations
 MC standard: 0.025061
 MC echantillonnage: 0.0056724
 MC variable de controle: 0.0036819
 MC variable antithetique: 0.016047

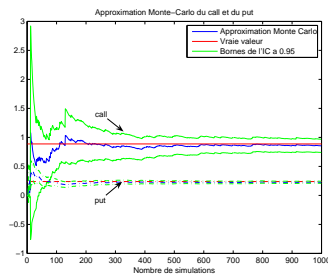


Figure 1: Comparaison des méthodes de Monte-Carlo pour le call et le put.

La figure 1 donne les approximations Monte-Carlo du call et du put par la méthode de l'exercice 1. On voit sur cette figure que l'intervalle de confiance pour le put est plus précis que pour le call. Ce qui signifie que la variance de $((K - e^X)_+)$ est plus faible que celle de $(e^X - K)_+$.

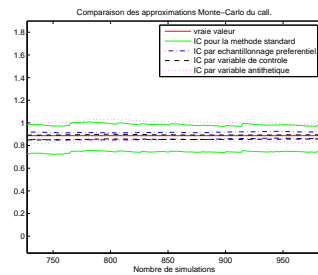


Figure 2: Comparaison des méthodes de Monte-Carlo pour le call et le put.

La figure 2 compare les différentes méthodes de Monte Carlo pour le calcul approché du call. Si on classe les différentes méthodes de la plus précise à la moins précise, on obtient

1. méthode par contrôle avec le put.
2. méthode par échantillonnage préférentiel de l'exercice 2.
3. méthode par variable antithétique.
4. méthode de l'exercice 1.

On voit que les méthodes de réduction peuvent ici faire gagner un facteur 10 sur la précision, ce qui signifie que pour une même précision souhaitée, on peut suivant la méthode choisie, diviser le nombre de simulations nécessaires par 100, et gagner autant en temps calcul.