Université Paris-Dauphine
Département MIDO

T.P.4 : Equations différentielles stochastiques-correction

1 EDS et calcul stochastique

1.1 Mouvement Brownien

1. Faire un code qui génère une trajectoire brownienne.

```
%BPATH2  Brownian path simulation: vectorized
close all
randn('state',100)          % set the state of randn
T = 1; N = 500; dt = T/N;

dW = sqrt(dt)*randn(1,N);   % increments
W = cumsum(dW);             % cumulative sum

plot([0:dt:T],[0,W],'r-')   % plot W against t
xlabel('t','FontSize',16)
ylabel('W(t)','FontSize',16,'Rotation',0)
```

2. Calculer les trajectoires correspondant à

$$u(t) = exp(t + 1/2W(t)).$$

Quel est l'effet d'une moyennisation sur un ensemble de ces trajectoires ?

1.2 Equations différentielles stochastiques

On considère l'équation différentielle stocchastique :

$$dX = \lambda X dt + \mu X dW$$

1. Résoudre cette équation en utilisant la méthode d'Euler-Marumaya. On choisira comme paramètres :

$$\lambda = 2, \mu = 1, X(0) = 1, T = 1.$$

2. Etudier numériquement la convergence de la méthode.

```
%EM  Euler-Maruyama method on linear SDE
%
% SDE is  dX = lambda*X dt + mu*X dW,   X(0) = Xzero,
%      where lambda = 2, mu = 1 and Xzero = 1.
%
% Discretized Brownian path over [0,1] has dt = 2^(-8).
% Euler-Maruyama uses timestep R*dt.

randn('state',100)
lambda = 2; mu = 1; Xzero = 1;    % problem parameters
T = 1; N = 2^8; dt = T/N;
dW = sqrt(dt)*randn(1,N);         % Brownian increments
W = cumsum(dW);                   % discretized Brownian path

Xtrue = Xzero*exp((lambda-0.5*mu^2)*([dt:dt:T])+mu*W);
plot([0:dt:T],[Xzero,Xtrue],'m-'), hold on

R = 4; Dt = R*dt; L = N/R;        % L EM steps of size Dt = R*dt
Xem = zeros(1,L);                 % preallocate for efficiency
Xtemp = Xzero;
for j = 1:L
   Winc = sum(dW(R*(j-1)+1:R*j));
   Xtemp = Xtemp + Dt*lambda*Xtemp + mu*Xtemp*Winc;
   Xem(j) = Xtemp;
end

plot([0:Dt:T],[Xzero,Xem],'r--*'), hold off
xlabel('t','FontSize',12)
ylabel('X','FontSize',16,'Rotation',0,'HorizontalAlignment','right')

emerr = abs(Xem(end)-Xtrue(end))

%EMSTRONG  Test strong convergence of Euler-Maruyama
%
% Solves    dX = lambda*X dt + mu*X dW,   X(0) = Xzero,
%         where lambda = 2, mu = 1 and Xzer0 = 1.
%
% Discretized Brownian path over [0,1] has dt = 2^(-9).
% E-M uses 5 different timesteps: 16dt, 8dt, 4dt, 2dt, dt.
```

```
% Examine strong convergence at T=1:  E | X_L - X(T) |.

randn('state',100)
lambda = 2; mu = 1; Xzero = 1;    % problem parameters
T = 1; N = 2^9; dt = T/N;         %
M = 1000;                         % number of paths sampled

Xerr = zeros(M,5);                % preallocate array
for s = 1:M,                      % sample over discrete Brownian paths
    dW = sqrt(dt)*randn(1,N);     % Brownian increments
    W = cumsum(dW);               % discrete Brownian path
    Xtrue = Xzero*exp((lambda-0.5*mu^2)+mu*W(end));
    for p = 1:5
        R = 2^(p-1); Dt = R*dt; L = N/R;     % L Euler steps of size Dt = R*dt
        Xtemp = Xzero;
        for j = 1:L
             Winc = sum(dW(R*(j-1)+1:R*j));
             Xtemp = Xtemp + Dt*lambda*Xtemp + mu*Xtemp*Winc;
        end
        Xerr(s,p) = abs(Xtemp - Xtrue);      % store the error at t = 1
    end
end

Dtvals = dt*(2.^([0:4]));
subplot(221)                                % top LH picture
loglog(Dtvals,mean(Xerr),'b*-'), hold on
loglog(Dtvals,(Dtvals.^(.5)),'r--'), hold off % reference slope of 1/2
axis([1e-3 1e-1 1e-4 1])
xlabel('\Delta t'), ylabel('Sample average of | X(T) - X_L |')
title('emstrong.m','FontSize',10)

%%%% Least squares fit of error = C * Dt^q %%%%
A = [ones(5,1), log(Dtvals)']; rhs = log(mean(Xerr)');
sol = A\rhs; q = sol(2)
resid = norm(A*sol - rhs)

%EMWEAK  Test weak convergence of Euler-Maruyama
%
% Solves    dX = lambda*X dt + mu*X dW,   X(0) = Xzero,
%        where lambda = 2, mu = 1 and Xzer0 = 1.
%
```

```
% E-M uses 5 different timesteps: 2^(p-10),  p = 1,2,3,4,5.
% Examine weak convergence at T=1:   | E (X_L) - E (X(T)) |.
%
% Different paths are used for each E-M timestep.
% Code is vectorized over paths.
%
% Uncommenting the line indicated below gives the weak E-M method.

randn('state',100);
lambda = 2; mu = 0.1; Xzero = 1; T = 1;  % problem parameters
M = 50000;                                % number of paths sampled

Xem = zeros(5,1);                         % preallocate arrays
for p = 1:5                               % take various Euler timesteps
       Dt = 2^(p-10); L = T/Dt;           % L Euler steps of size Dt
       Xtemp = Xzero*ones(M,1);
       for j = 1:L
           Winc = sqrt(Dt)*randn(M,1);
           % Winc = sqrt(Dt)*sign(randn(M,1)); %% use for weak E-M %%
           Xtemp = Xtemp + Dt*lambda*Xtemp + mu*Xtemp.*Winc;
       end
       Xem(p) = mean(Xtemp);
end
Xerr = abs(Xem - exp(lambda));

Dtvals = 2.^([1:5]-10);
subplot(222)                              % top RH picture
loglog(Dtvals,Xerr,'b*-'), hold on
loglog(Dtvals,Dtvals,'r--'), hold off    % reference slope of 1
axis([1e-3 1e-1 1e-4 1])
xlabel('\Delta t'), ylabel('| E(X(T)) - Sample average of X_L |')
title('emweak.m','FontSize',10)

%%%% Least squares fit of error = C * dt^q %%%%
A = [ones(p,1), log(Dtvals)']; rhs = log(Xerr);
sol = A\rhs; q = sol(2)
resid = norm(A*sol - rhs)
```