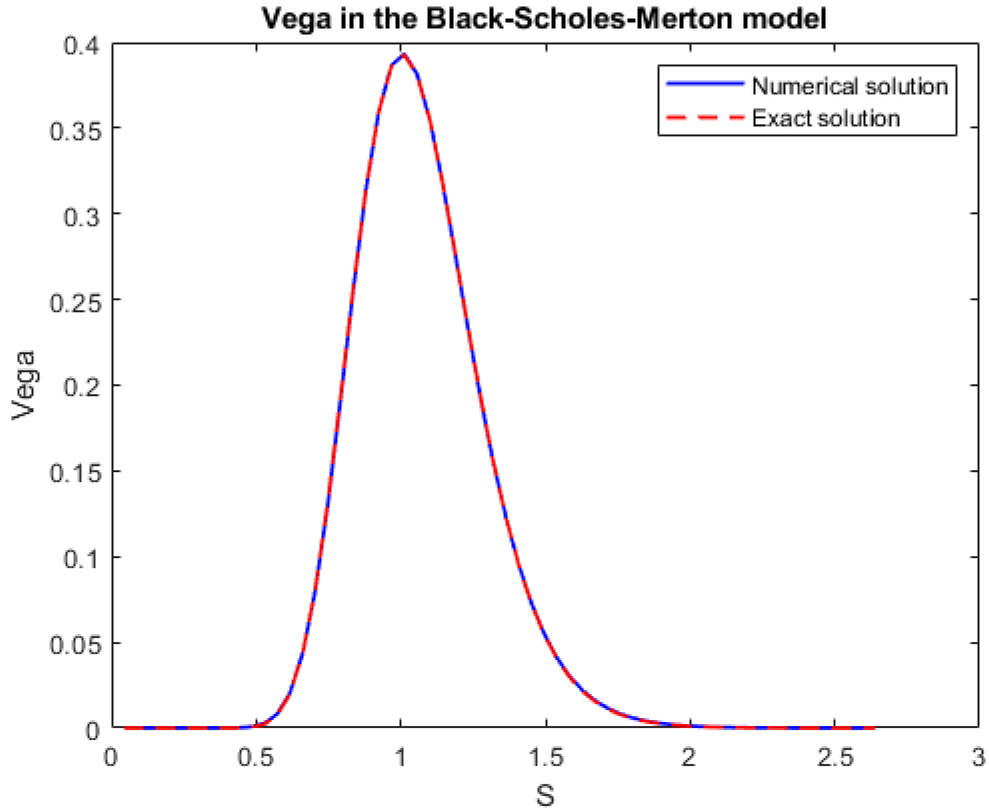
```
% Take Home exam 4 - Anita Mezzetti

clc
clear all
close all

% parameters
sigma = 0.2;
r = 0.015;
K = 1;
T = 1;
S_min = 0;

% S_max is such that the truncation error is less than 10^-6:
max_tol = 1e-6;
S_max = K * exp(0.5 * sigma^2 * T - sigma * sqrt(T) * norminv(max_tol/
K));

%2b
 ------------------------------------------------------------------

%--------------------- i) THETA METHOD: ---------------------------

theta = 0.5;    % => Crank-Nicholson
Nx = 60;        % number of intervals in x
Nt = 100;       % number of time steps

% initialisation functions for bs_timestepping:
forcing = @(S,t) zeros(length(S),length(t)); % right-hand-side of the
 equation
bc_right = @(t) zeros(size(t)); % boundary condition on the right
 border
init_cond = @(S) max(K-S,0); % initial condition of the problem

[U_theta, FD_grid, time_steps] = bs_timestepping...
    (sigma, r, forcing, bc_right, init_cond, S_max, Nx, T, Nt, theta);
% U_theta is the option price (solution of the pde)


%----------- ii) approximate rhs of the PDE solved by vega
 -------------

% approximation of U_SS
vega_rhs = find_rhs(U_theta, sigma, FD_grid, S_max);

% vega boundary and initial conditions
vega_bc_right = @(t) zeros(size(t));
vega_initcond = @(S) zeros(size(S));

% finite-difference solver using theta-method
[vega_approx, FD_grid, ] = vega_timestepping(sigma, r, vega_rhs,
 vega_bc_right,...
```

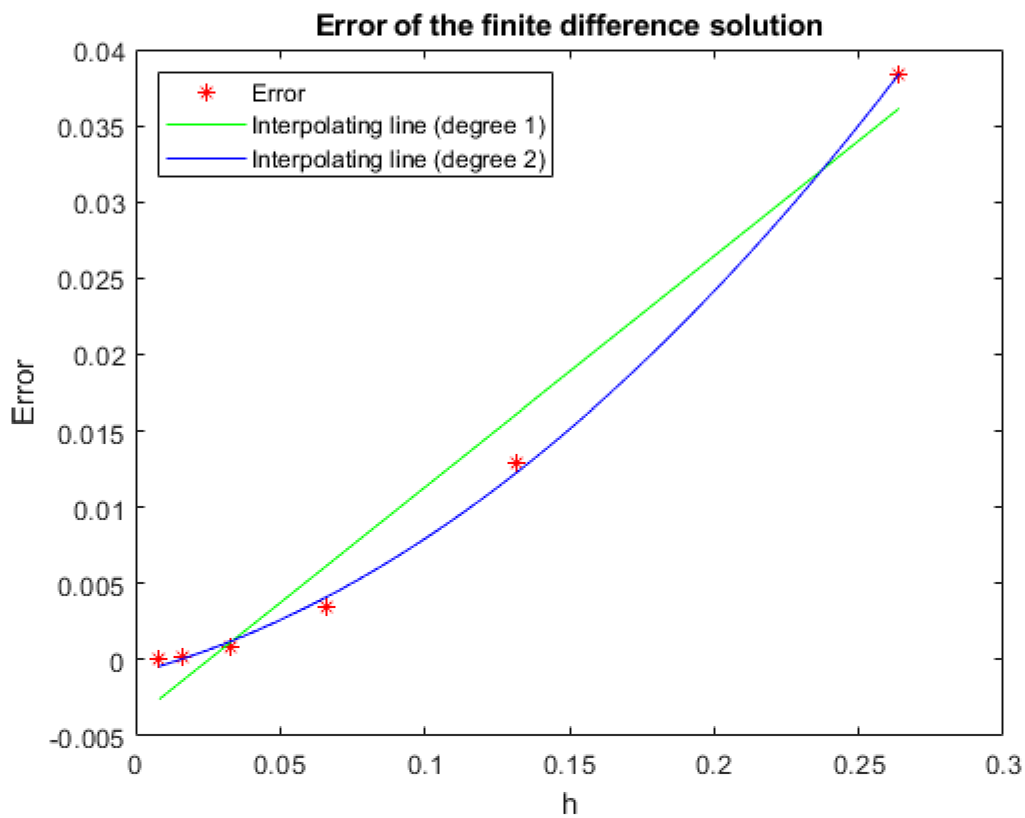
```
    vega_initcond, S_max, Nx, T, Nt, theta);


%------------------- iii) exact vega and plot -----------------------

% exact solution:
vega_exact = blsvega(FD_grid(2:end)',K,r,T,sigma);

% plot 2b iii
figure
plot(FD_grid(2:end),vega_approx(2:end,end),'b', 'LineWidth', 1.3)
hold on
plot(FD_grid(2:end),vega_exact,'r--', 'LineWidth', 1.3)
title('Vega in the Black-Scholes-Merton model')
xlabel('S')
ylabel('Vega')
legend('Numerical solution','Exact solution')

%2c
 ------------------------------------------------------------------


i = [0:5];
Nx = 10 * 2.^i;
Nt = 0.6 * Nx;
h = S_max./Nx;

error = zeros(length(i),1); % initialization error vector

for n = 1:length(i) % loop for getting finer grids
    x = Nx(n); % Nx
    t = Nt(n); % Nt

    % option prices using theta-method:
    [U_theta, FD_grid] = bs_timestepping(...
        sigma, r, forcing, bc_right, init_cond, S_max, x, T, t,
 theta);

    % approximation of U_SS
    vega_rhs = find_rhs( U_theta, sigma, FD_grid, S_max );

    % finite-difference solver using theta-method
    [vega_approx,FD_grid,time_steps] = vega_timestepping(sigma, r,
 vega_rhs,...
        vega_bc_right, vega_initcond, S_max, x, T, t, theta);

    % exact solution
    vega_exact = blsvega(FD_grid(2:end)', K, r, T, sigma);

    % error
    error(n) = max(abs(vega_approx(2:end,end) - vega_exact));
end

% plot 2c
X = linspace(h(1),h(end));
```

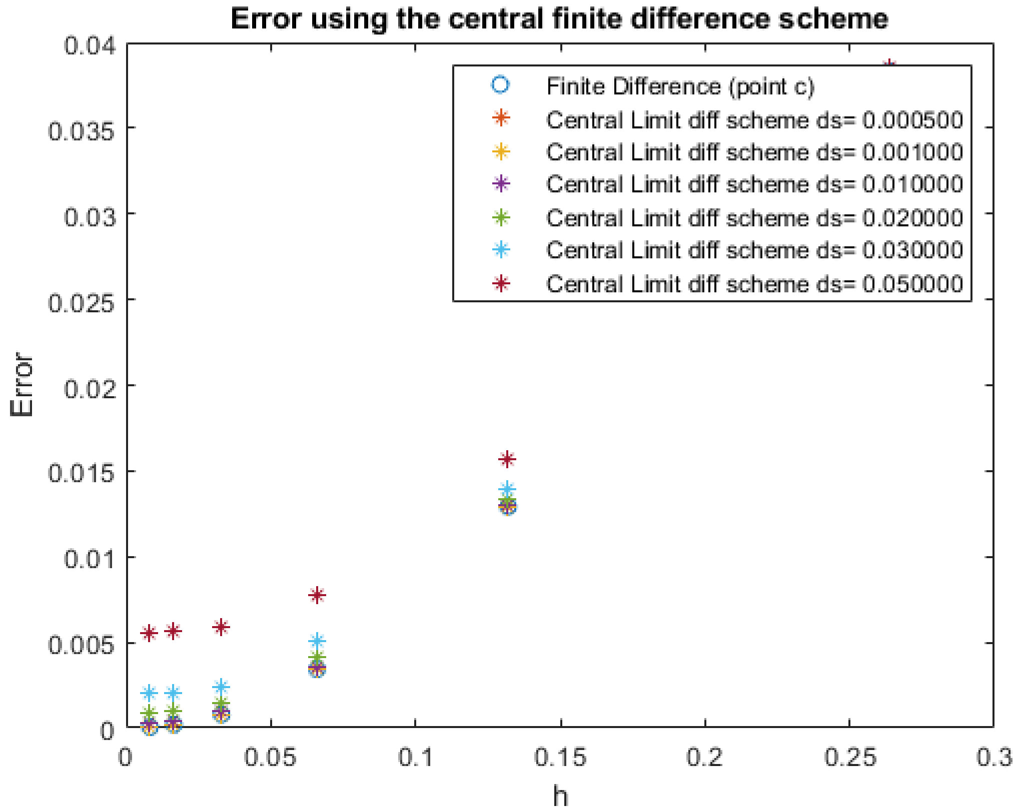
```
P1 = polyfit(h,error,1);
Y1 = polyval(P1,X);
P2 = polyfit(h,error,2);
Y2 = polyval(P2,X);

figure
plot(h, error', 'r*')
hold on
plot(X, Y1, 'g')
plot(X, Y2, 'b')
title('Error of the finite difference solution')
xlabel('h')
ylabel('Error')
legend('Error','Interpolating line (degree 1)',...
    'Interpolating line (degree 2)','Location', 'northwest')

% order of convergence:
log_error = log(error);


%2d
 -------------------------------------------------------------------
ds = [0.0005 0.001 0.01 0.02 0.03 0.05];
%ds = [1e-5 1e-6 1e-7];

%------------ i) vega with the central limit diff scheme:
 ----------------
%------------ ii) approximation error and plot:
 --------------------------

error2 = zeros (length(Nx), length(ds)); % error matrix initialisation

for n = 1:length(Nx)

    x = Nx(n); % Nx
    t = Nt(n); % Nt

    for i = 1:length(ds)

        % option prices using theta-method:
        [U_plus, ] = bs_timestepping(sigma+ds(i), r, forcing, ...
            bc_right, init_cond, S_max, x, T, t, theta);

        [U_minus,FD_grid] = bs_timestepping(sigma-ds(i), r,
 forcing,...
            bc_right, init_cond, S_max, x, T, t, theta);

        % approximated vega using the central limit difference scheme:
        vega_sigma = (U_plus(:,end) - U_minus(:,end)) / (2 * ds(i));

        % exact vega
        vega_exact = blsvega(FD_grid(2:end)',K,r,T,sigma);

        % error
```

```
        error2(n,i) = max(abs(vega_sigma(2:end) - vega_exact));
    end
end

% plot

% replot the results of c:
figure
plot(h, error, 'o', 'DisplayName', 'Finite Difference (point c)')

% new plot
for i = 1:length(ds)
    hold on
    tt = sprintf('Central Limit diff scheme ds= %f ', ds(i));
    plot(h, error2(:,i), '*', 'DisplayName', tt);
end
title('Error using the central finite difference scheme')
xlabel('h')
ylabel('Error')
legend show
hold off




%
 functions:-----------------------------------------------------------

function [rhs] = find_rhs ( U, sigma, FD_grid, S_max )

[Nx,Nt] = size(U);
h = S_max/(Nx - 1);      % space step
rhs = zeros(Nx, Nt);     % solution initialisation

for x = 2 : Nx-1
    approx_second_der = ( U(x+1,:) - 2*U(x,:) + U(x-1,:)) / h^2;
    rhs(x,:) = sigma * FD_grid(x)^2 * approx_second_der;
end

end

ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
```

ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao
ciao


Vega in the Black-Scholes-Merton model
Numerical solution
Exact solution
Vega
S
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
0
0.5
1
1.5
2
2.5
3


Error of the finite difference solution
Error
Interpolating line (degree 1)
Interpolating line (degree 2)
Error
h
0.04
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
-0.005
0
0.05
0.1
0.15
0.2
0.25
0.3



Published with MATLAB® R2020a