

Our online Tutors are available 24*7 to provide Help with MATLAB Homework/Assignment or a long term Graduate/Undergraduate MATLAB Project. Our Tutors being experienced and proficient in MATLAB ensure to provide high quality MATLAB Homework Help. Upload your MATLAB Assignment at 'Submit Your Assignment' button or email it to info@assignmentpedia.com. You can use our 'Live Chat' option to schedule an Online Tutoring session with our MATLAB Tutors.

TASK

Price a vanilla put option and compute its Greeks under the CEV model for the underlying:

$$dS_t = r * S_t dt + \sigma * S_t^\alpha dW_t.$$

Use the Crank-Nicolson scheme.

SOLUTION

The Matlab code is displayed below.

runCEV.m

```
clear;
clc;

dynamics.alpha      = 0.5;
dynamics.volcoeff   = 3;
dynamics.r          = 0.05;
dynamics.S0         = 100;
FD.SMax             = 200;
FD.SMin             = 50;
FD.deltaS           = 0.1;
FD.deltat           = 0.0005;

contract.T           = 0.25;
contract.K           = 100;

displayStart        = dynamics.S0-FD.deltaS*1.5;
displayEnd           = dynamics.S0+FD.deltaS*1.5;
[S0_ALL putprice]   = CEV(contract,dynamics,FD);
indices              = (S0_ALL > displayStart & S0_ALL < displayEnd);
disp([S0_ALL(indices) putprice(indices)])

putprice_0 = putprice(indices);
DELTA = (putprice_0(1) - putprice_0(3)) / (2*FD.deltaS)
GAMMA = (putprice_0(1) - (2*putprice_0(2)) + putprice_0(3)) / (FD.deltaS^2)

for i = 85:1:115
    K(i-84)          = i;
    contract.K        = i;
    [S0_ALL putprice] = CEV(contract,dynamics,FD);
    indices            = (S0_ALL > displayStart & S0_ALL < displayEnd);
```

```

X                = [S0_ALL(indices) putprice(indices)];
putoptionprice_0(i-84) = X(2,2);
vol_imp(i-84)      = blsimpv(dynamics.S0, contract.K, dynamics.r,
contract.T,X(2,2), 50, 0, [], false);
end

result           = [K' putoptionprice_0' vol_imp'];
plot(K,vol_imp,'-+');
title('Implied Volatility vs Strike Price')
xlabel('Strike Price');
ylabel('Implied Volatility');

```

CEV.m

```
function [S,putprice] = CEV(contract,dynamics,FD)
```

```

volcoeff        = dynamics.volcoeff;
alpha           = dynamics.alpha;
r               = dynamics.r;
T               = contract.T;
K               = contract.K;

```

```

SMax            = FD.SMax;
SMin            = FD.SMin;
deltaS          = FD.deltaS;
deltat          = FD.deltat;

```

```

N               = round(T/deltat);
if abs(N-T/deltat) > 1e-12, error('Bad time step'),
end

```

```

numS            = round((SMax-SMin)/deltaS)+1;
if abs(numS-(SMax-SMin)/deltaS-1) > 1e-12, error('Bad space step'),
end

```

```

S               = (SMax:-deltaS:SMin)';
S_lowboundary   = SMin - deltaS;

```

```

putprice        = max(K-S,0);
ratio           = deltat/deltaS;
ratio2          = deltat/deltaS^2;

```

```

f               = (S.^(2*alpha))*(0.5*volcoeff*volcoeff);
g               = r.*S;
h               = -r;

```

