FiniteDifference_BS

Antoine Jacquier

TITLE:	FiniteDifference_BS
AUTHOR:	Antoine Jacquier
NUMBER OF PAGES:	6
FIRST VERSION:	January 19, 2017
CURRENT VERSION:	January 19, 2017
REVISION:	0.0.0

Contents

1 Finite-difference scheme for the Black-Scholes PDE

In [1]:
```
from scipy import interpolate
import numpy as np
```

2 Description of the scheme

2.0.1 Boundary conditions

2.0.1.1 At maturity

In [2]:
```
# Function specifying the boundary condition
def bcTime0(x, args):
    [strike, r, sigma] = args
    a = (r - 0.5 * sigma * sigma) / (sigma * sigma)
    b = 2. * r / (sigma * sigma) + a * a
    # From the Call option payoff
    return np.exp(a * x) * max(np.exp(x) - 1.0, 0.0)
```

2.0.1.2 In the space domain

In [3]:
```
def bcxXMin(x):
    return 0.



def bcxXMax(x, tau, args):
    [strike, r, sigma] = args
    a = (r - 0.5 * sigma * sigma) / (sigma * sigma)
    b = 2. * r / (sigma * sigma) + a * a
    return (np.exp(x) - np.exp(-2. * r * tau / (sigma * sigma))) * np.exp(a * x + b * tau)
```

2.0.2 Explicit Finite Difference Scheme

In [4]:
```
def heatEq_FDExplicit(xMin, xMax, tauMax, nbStepTime, nbStepSpace, *extraArgs):
    # bc represents the function (of x) at the boundary tauMin
    # We follow the conventions in the notes:
    # there are (nbStepSpace+1) points on the x-axis
    # and (nbStepTime+1) points on the tau-axis

    sigma = extraArgs[0][2]
    print "sigma: ", sigma
    # Grid step in space
    dx = (xMax - xMin) / (1. * nbStepSpace)
    dt = tauMax / (1. * nbStepTime)     # Grid step in time
    cfl = dt / (dx * dx)
    if cfl > 1.:
        print '************************************************************'
        print '*****WARNING: THE CFL CONDITION IS NOT SATISFIED!!!!!!*****'
        print "dt = ", dt
```

```
        print "dx2 = ", dx * dx
        print '*********    clf ratio = %s    ***************************' % cfl
        print '************************************************************'
    else:
        print "dt =", dt
        print "dx2 = ", dx * dx
        print "CFL condition satisfied"

    # Boundary conditions
    # set the grid on the x-axis
    xx = np.linspace(xMin, xMax, nbStepSpace + 1)
    uu = [[0.] * (nbStepSpace + 1)] * (nbStepTime + 1)
    timeVector = np.linspace(0., tauMax, nbStepTime + 1)
    uu[0] = np.array([bcTime0(x, extraArgs[0]) for x in xx])

    for i in range(0, nbStepTime):
        uu[i + 1][0] = bcxXMin(xx[0])
        uu[i + 1][-1] = bcxXMax(xx[-1],
                                timeVector[i + 1], extraArgs[0])

        for j in range(1, nbStepSpace):
            uu[i + 1][j] = cfl * \
                (uu[i][j + 1] - 2.0 * uu[i]
                 [j] + uu[i][j - 1]) + uu[i][j]
    return [xx, timeVector, uu]
```

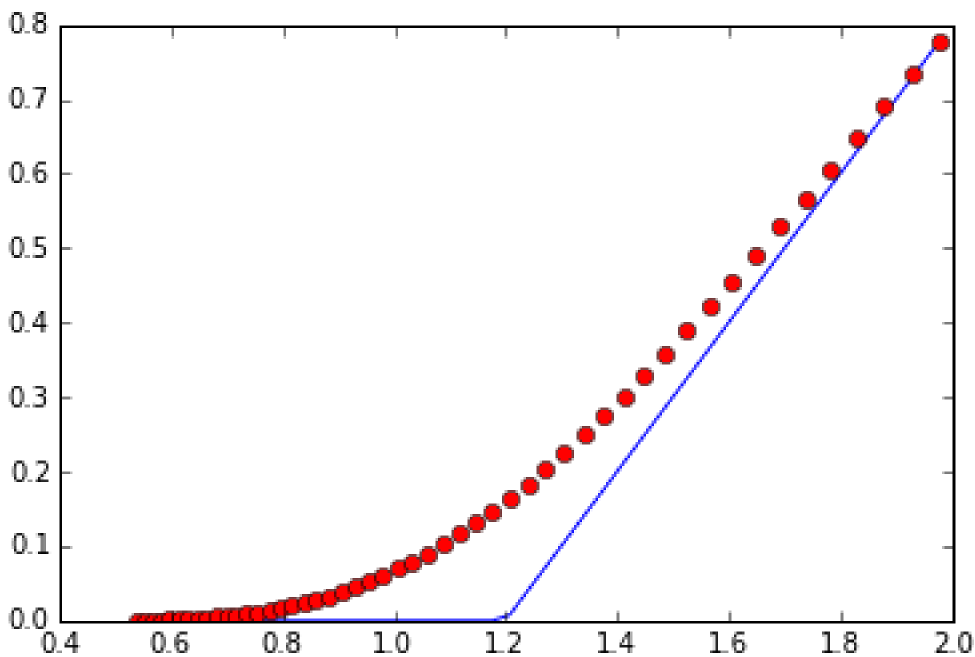
2.0.3 Numerical example

```
In [9]: xMin, xMax, nbStepTime, nbStepSpace = - \
            0.8, 0.5, 200, 50  # (try with 100)
        T, strike, r, sigma = 2., 1.2, 0., 0.2
```

```
In [10]: extraArgs = strike, r, sigma
         [xx, timeVector, uu] = heatEq_FDExplicit(
             xMin, xMax, 0.5 * sigma * sigma * T, nbStepTime, nbStepSpace, extraArgs)
         #plot(xx, uu[0], 'b', xx, uu[-1], 'r')
         # show()



         def mappingXToS(xx, tautau, uu, T, r, sigma, strike):
             # xx is the x coordinate vector; tautau the time-coordinate vector
             # uu the value function on the grid
             a = (r - 0.5 * sigma * sigma) / (sigma * sigma)
             b = 2. * r / (sigma * sigma) + a * a
```

```
    tt = T - tautau / (0.5 * sigma * sigma)
    SS = strike * np.exp(xx)
    output = [0.0] * len(tautau)
    for n in arange(len(tautau)):
        #       for p in arange(len(xx)):
        output[n] = strike * \
            np.exp(-a * xx - b * tautau[n]) * uu[n]
    return [SS, tt, output]

[SS, tt, CC] = mappingXToS(
    xx, timeVector, uu, T, r, sigma, strike)
plot(SS, CC[0], 'b')
plot(SS, CC[-1], 'ro')
show()
m = 99
print "Remaining maturity: ", tt[m]
f = interpolate.interp1d(SS, CC[m], 'cubic')
print "Call option value: ", float(f(1.2))
```

```
sigma:  0.2
dt = 0.0002
dx2 =  0.000676
CFL condition satisfied
```



```
Remaining maturity:  1.01
Call option value:  0.161562738184
```

```
In [6]:
```