Math 16:642:623
Computational Finance
Homework 2

NITISH KANABAR
nitish@eden.rutgers.edu

February 11, 2010

Building and Running the Homework

The file HW2_KANABARN.zip contains my implementation and dependencies from Mark Joshi's implementation.

Unzip the submitted file. This will create a directory called HW2_KANABARN which contains my code and the required files from Mark Joshi. Change to that directory and run 'make' to build the homework. This will build the executable 'hw2'. Running 'hw2' will calculate the price of the barrier option as specified in the assignment.

I developed and tested my code on Mac OS X using the GCC Version 4.2.1.

Algorithms

The simulation algorithm is implemented as shown in the following pseudo-code. My algorithm monitors the option for each spot price. If a spot price knocks out the option, then that path is no longer simulated. This prevents unused computations. For antithetic simulation, both options need to be knocked-out. However for control-variates, the full path is always computed. Note that the asset-price is used as the control-variate.

```
for each path
    precompute values to optimize the simulation
    for each step
        compute the new spot price
        if this spot knocks out the option, then stop the path
    end step
    compute and store the option payoff using the ending spot price
end path
compute the average of the stored payoffs
```

The pre-computation optimization technique involves simplifying the approximation and calculating values used in the approximation so that the number of arithmetic operations required per step are minimized. This speeds up the simulation significantly over thousands of simulated paths.

Each simulation type uses the Euler-scheme to simulate movements of the underlying asset.

Results

The simulation resulted in the following values for the specified vanilla-call.

```
Closed-form price = 0.0517846, run time = 5.96046e-06 seconds.
Monte Carlo price = 0.0817877, std error = 0.00658424, run time = 0.115943 seconds.
Antithetic variates price = 0.0765821, std error = 0.00595476, run time = 0.118165 seconds.
Control Variates price = 0.0604525, std error = 0.00588973, run time = 0.119072 seconds.
```

We observe that the price of the price of the variance of the estimators decreases when using Antithetic variates and control-variates estimators.

Estimates for b^* and ρ for the up-and-out barrier when using a Vanilla option variate are as shown below. We observe that using the this variate does not give us accurate results. This is due to the non-linear nature of the option price.

```
Strike = 120, b*: -0.000571995, rho: -0.0238168
Strike = 200, b*: 0.548571, rho: 0.735678
Strike = 1000, b*: 0.951229, rho: 1
```

Estimates for b^* and ρ for a Vanilla call with asset price variate are shown below. We observe that as the strike increases, the correlation and the stderror both decrease.

```
Strike: 20, b*: 0.951229, rho: 1
Strike: 80, b*: 0.841862, rho: 0.981235
Strike: 100, b*: 0.641664, rho: 0.922447
Strike: 120, b*: 0.421125, rho: 0.813318
Strike: 200, b*: 0.0462141, rho: 0.390483
```

We observe that using the variance reduction methods increases the runtime required to compute the option price.