Developer Documentation for Two-Dimensional Black-Scholes model Option Pricer

It prices options!

Luke Armitage

introduce software.
list and details of headers, classes, functions and variables.
detail how to add new payoff functions.
detail nonlinear solver method used in 3.
report on test runs of code.

Disclaimer: All code was written by me except for `Project.h`, and the code in `solver.h` is based on the bisection method nonlinear solver found in `Solver03.h` provided as part of the course material.

clever things i've done:
overloading of 'get' functions, so you don't have to enter the same values again and again to test.

1 Analysis of `PriceAmerican` function

Part of the project brief is to analyse the function `PriceAmerican`. Not sure what you mean by that, Alet. Here's the code from `Project.h`.

analyse `PriceAmerican`

```
inline double PriceAmerican (const CorrBinModel& model,
                             const Payoff& payoff,
                             int N)
{
    vector<vector<double> > v,pv;
    vector<double> q = model.Get_q();
    double d = exp(-model.Get_r()*model.Get_h()),
           q00d = d*(1-q[0])*(1-q[1]),
           q01d = d*(1-q[0])*q[1],
           q10d = d*q[0]*(1-q[1]),
           q11d = d*q[0]*q[1];
    double ev,cv;
    v.resize(N+1);
    for(int j0=0; j0<=N; j0++)
    {
        v[j0].resize(N+1);
        for(int j1=0; j1<=N; j1++)
            v[j0][j1] = payoff.Value(model.S(N,j0,j1));
    }
    for(int n=N-1; n>=0; n--)
    {
        pv=v;
        for(int j0=0;j0<=n;j0++)
            for(int j1=0; j1 <= n; j1++)
            {
                ev = payoff.Value(model.S(n,j0,j1));
                cv = q00d*pv[j0][j1]
                   + q01d*pv[j0][j1+1]
                   + q10d*pv[j0+1][j1]
                   + q11d*pv[j0+1][j1+1];
                v[j0][j1] = (ev>cv)?ev:cv;
            }
    }
    return v[0][0];
};
```