5-1-2014

High Precision Rapid Convergence of Asian Options

Mario Y. Harper
Utah State University

High Precision Rapid Convergence of Asian Options

Mario Harper

Economics and Finance Department
Utah State University, Logan, Utah

Thank you!

A special thanks to my mentor Dr. Tyler Brough Luis Gordillo for pushing me in mathematics.

The entire USU Economics/Finance Department and Physics Department

Outline

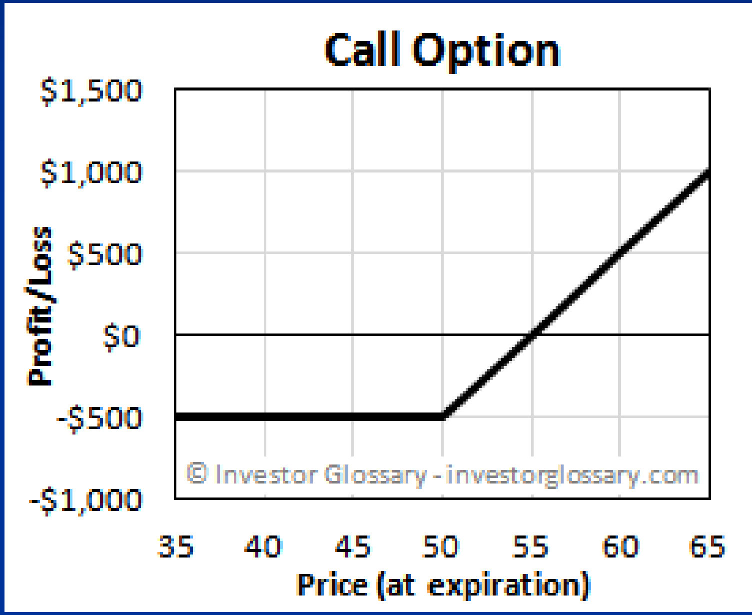
- Asian Options
  - What is it and why do we care?
- What is our problem?
  - Understanding the underlying algorithm.
- Analysis and Modeling
  - Understanding and comparing behavior.
- Future Work
  - Quantitative and computational work.

Motivation

About Me

- Physics/Economics Major: Emphasis in computational methods and mathematics
- Sudden change took me to the field of finance. Has a lot to do with Jamba Juice…

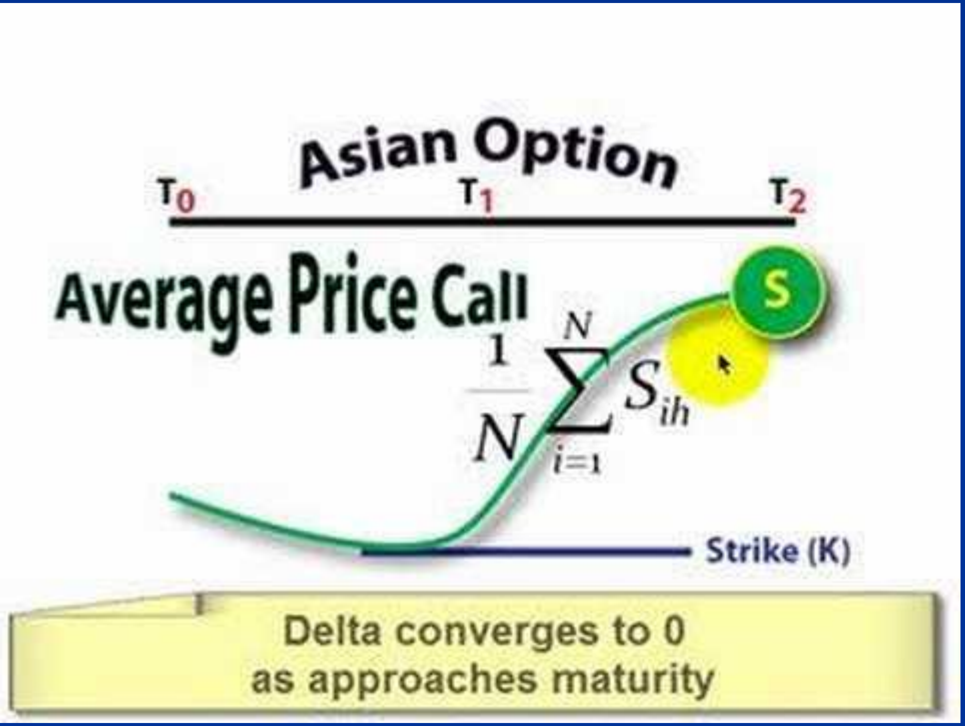
What is a Financial Option?



An option is a contract which gives the buyer (the owner) the right, but not the obligation, to buy or sell an underlying asset or instrument at a specified price on or before a specified date.

Why do we want them?
They are essentially a stock insurance.

What is an Asian Option?



An Asian Option is one whose payoff depends on the average price of the underlying asset over a certain period of time as opposed to at maturity. Also known as an average option.

This is an insurance against price changes.

Why Study Asian Options?

Example:

Suppose that you are a power-company:

A major cost for you is fuel to power your plant, but the supply of fuel and its costs are volatile.

You want to charge a fixed rate for electricity, how do we price it?

The Problem

Prices have to be low but without risk
This is a zero sum game, someone is going to lose.

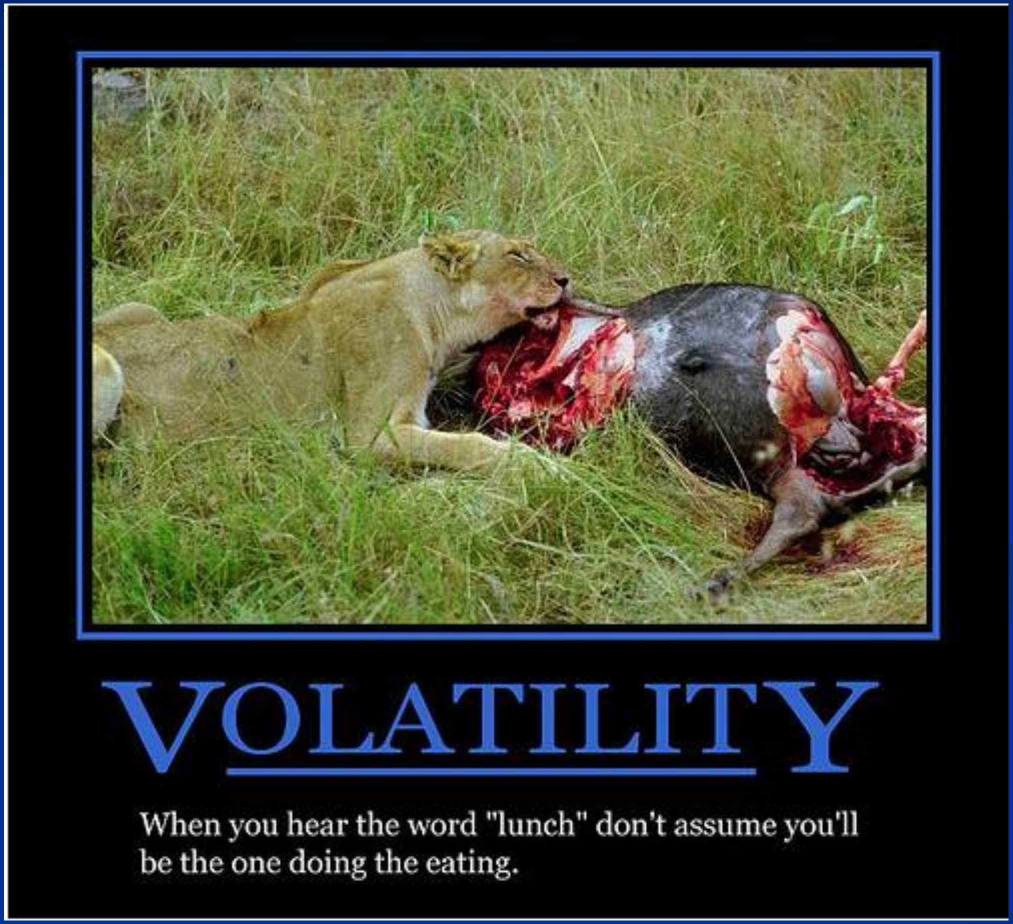
Advantages of selling at a stable price

- Much higher efficiency for the economy

Disadvantages

- Higher stress on those holding the asset.
- Computationally and Mathematically intensive.

The Problem



The bigger problem:

Many stocks and assets need to be priced quickly and accurately. But the future is unknown and poses risks.

Analysis

A Brief, Brief, very Brief Derivation

$$dS = \mu \cdot S \cdot dt + q \cdot S \cdot dz$$

$$df = \frac{\delta f}{\delta x}(\mu \cdot S \cdot dt + q \cdot S \cdot dz) + \frac{\delta f}{\delta t} dt + \frac{1}{2}\frac{\delta^2 f}{\delta x^2}\left[(\mu \cdot S \cdot dt)^2 + 2 \cdot (\mu \cdot S)(q \cdot S) dt \cdot dz + (q \cdot S \cdot dz)^2\right]$$

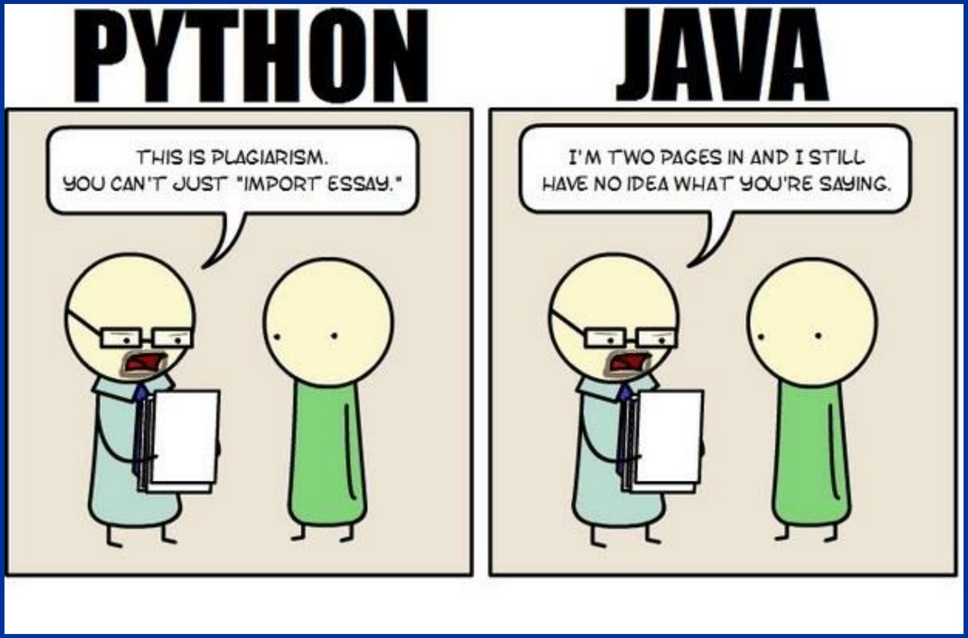
$$dC(S,t) = \left[\frac{\delta C}{\delta t} + \frac{\delta C}{\delta S}\mu \cdot S + \frac{1}{2}(\mu \cdot S)^2 \frac{\delta^2 C}{\delta S^2}\right] dt + \frac{\delta C}{\delta S}\omega \cdot S \cdot dz$$

$$\frac{1}{T} \cdot \int_0^T S(t)\, dt - K$$

$$v(t,x,y) = v_t(t, S_t, Y_t)\, dt + v_x(t, S_t, Y_t)\, dS_t + \frac{1}{2} v_{xx}(t, S_t, Y_t)\, d(S, S_t) + v_y(t, S_t, Y_t)$$

$$v(t,x,y) = \left(v_t(t, S_t, Y_t) + \mu \cdot S_t \cdot v_x(t, S_t, Y_t) + \frac{1}{2}\sigma^2 S_t{}^2 v_{xx}(t, S_t, Y_t) + S_t \cdot v_y(t, S_t, Y_t) \right) \cdot dt + \sigma \cdot S_t \cdot v_x(t, S_t, Y_t)\, dW_t$$

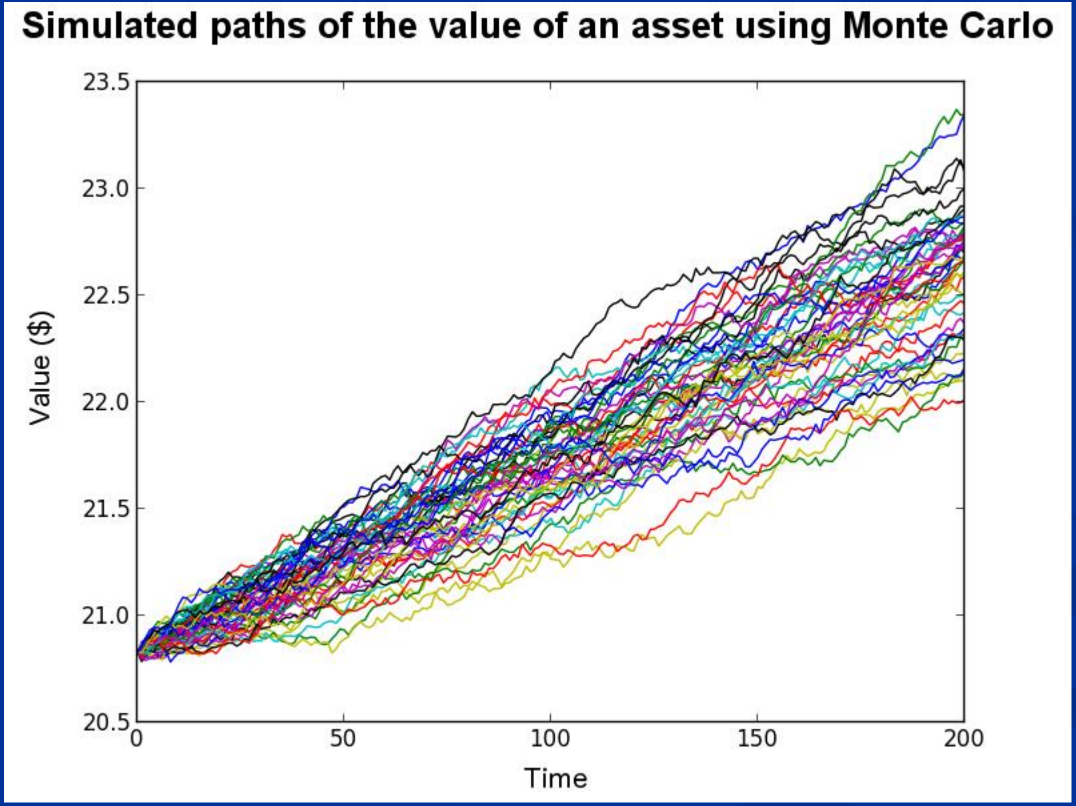
Python



- Very simple language compared to Java
- Easy to import libraries and operators
- Somewhat slow compared to C++
- Easily able to show comparisons

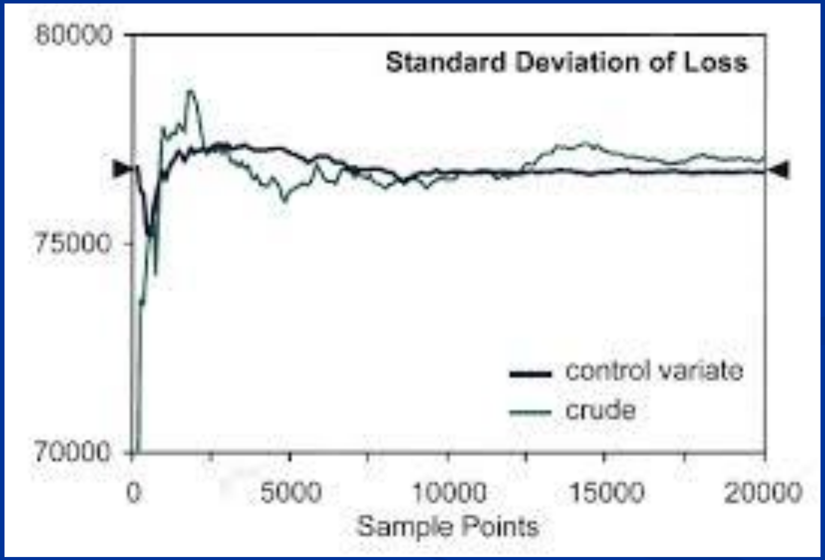
Monte Carlo Simulations

- We simulate the random path of the price by Monte Carlo simulations. Some drawbacks…



Control Variate Theory

- As the name sounds, this is a method to control the variance in the simulations.
- Unfortunately, this uses more computing power and time.

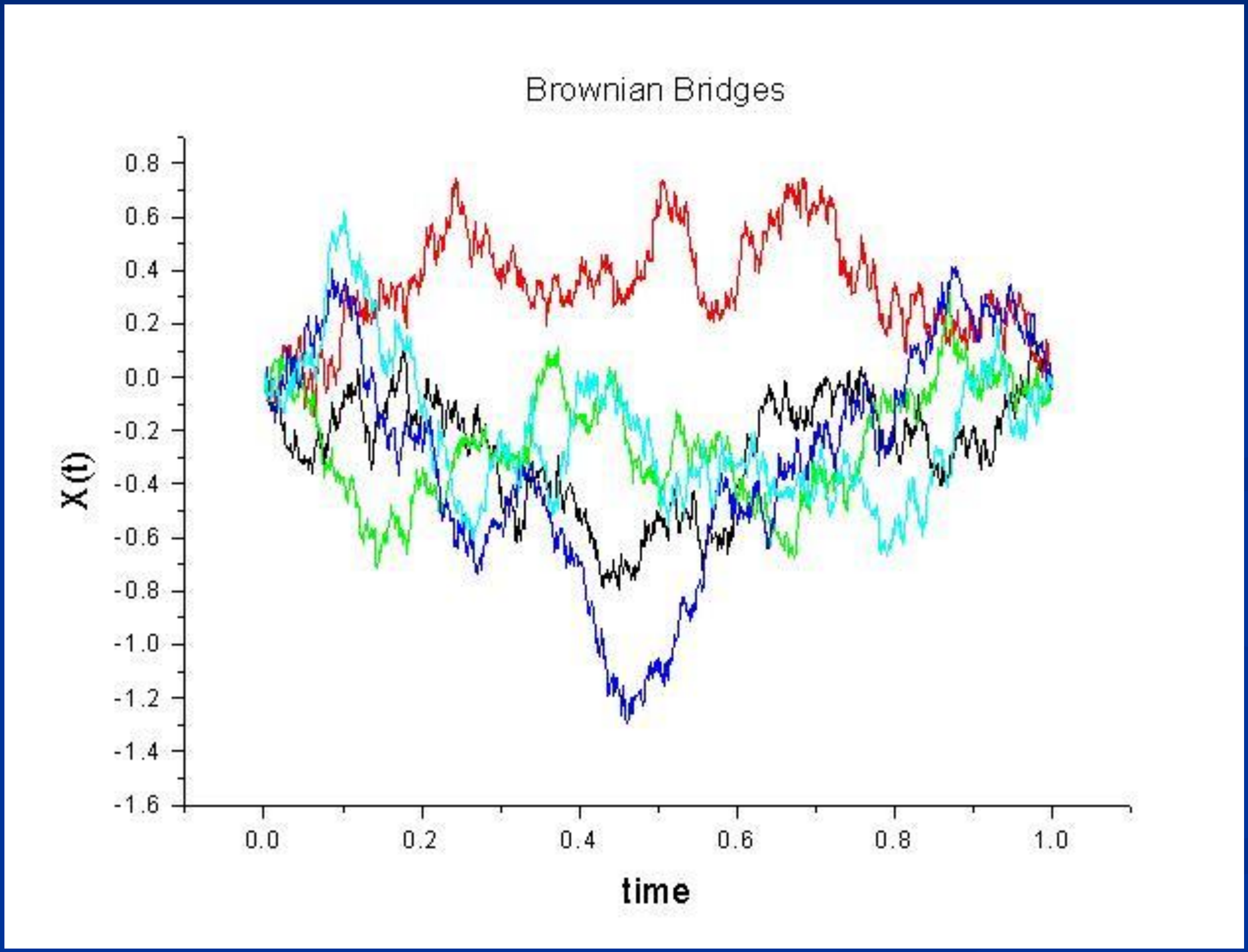


Control Variate Code

We can approximate the true value around which we choose to center our distribution around by decomposing the PDE from before into:

```
def BlackScholesCall(S0, X, r, sigma, T, N, delta):
        d1 = (log(S0/X)+(r - deltas + .5*sigmas*sigmas)*T) / (sqrt(T)*sigmas)
        d2 = d1-sigmas*sqrt(T)
        Gtrue = ( S0*exp(- deltas * T)* norm.cdf(d1)) - (X*exp(-r*T)*norm.cdf(d2))
return Gtrue
Gtrue = BlackScholesCall(S0, X, r, sigma, T, N, delta)
```

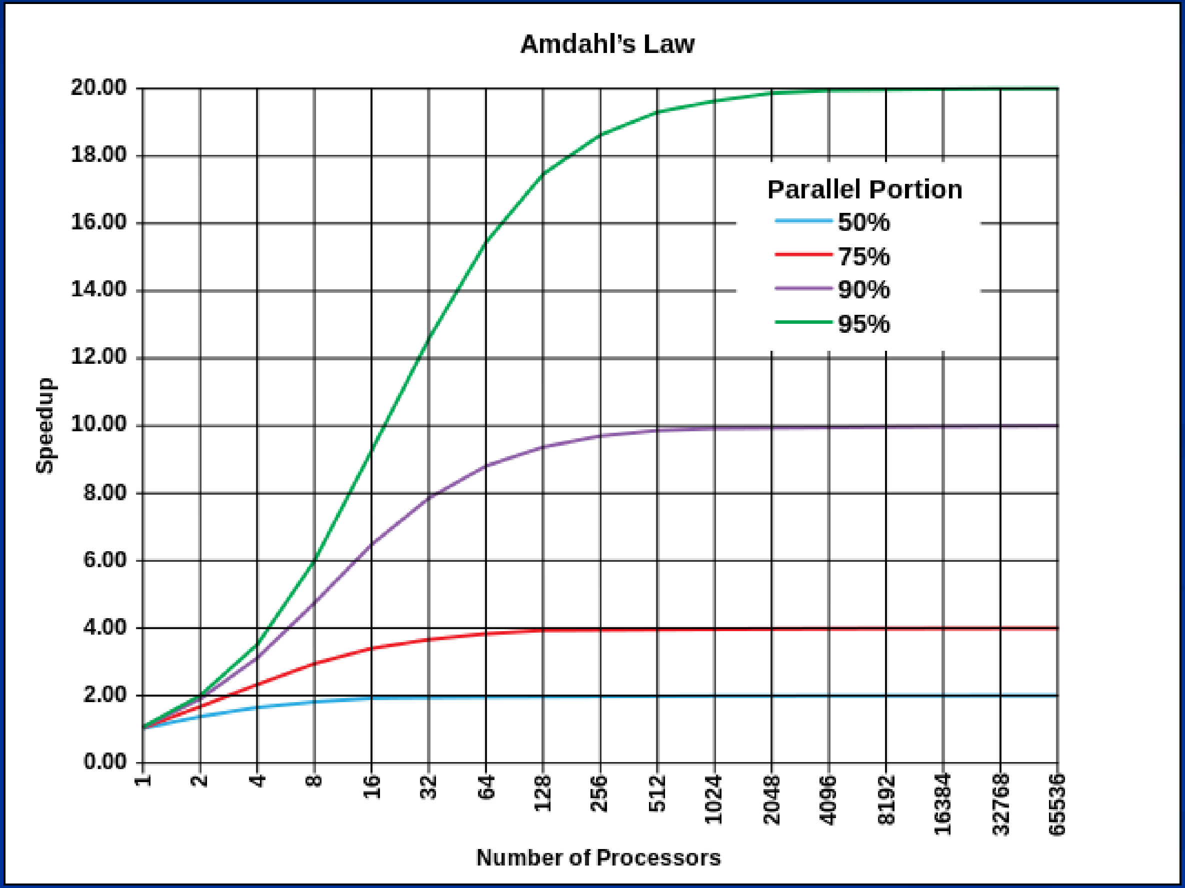
Brownian Bridge



Start and stop points are fixed by the algorithm.

Good at speeding up the convergence of stochastic processes.

Parallel Computing



Large problems should be divided into smaller problems.

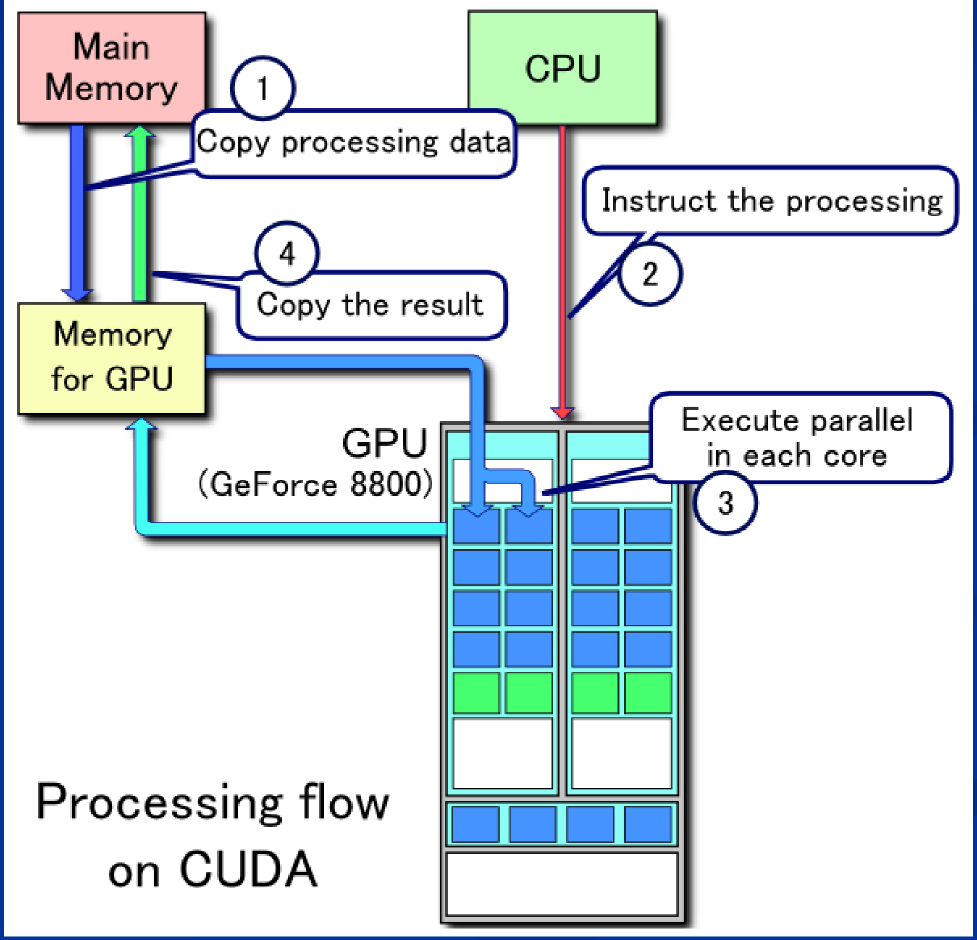
Using all of the resources to simultaneously compute the problem.

CUDA

(Compute Unified Device Architecture)

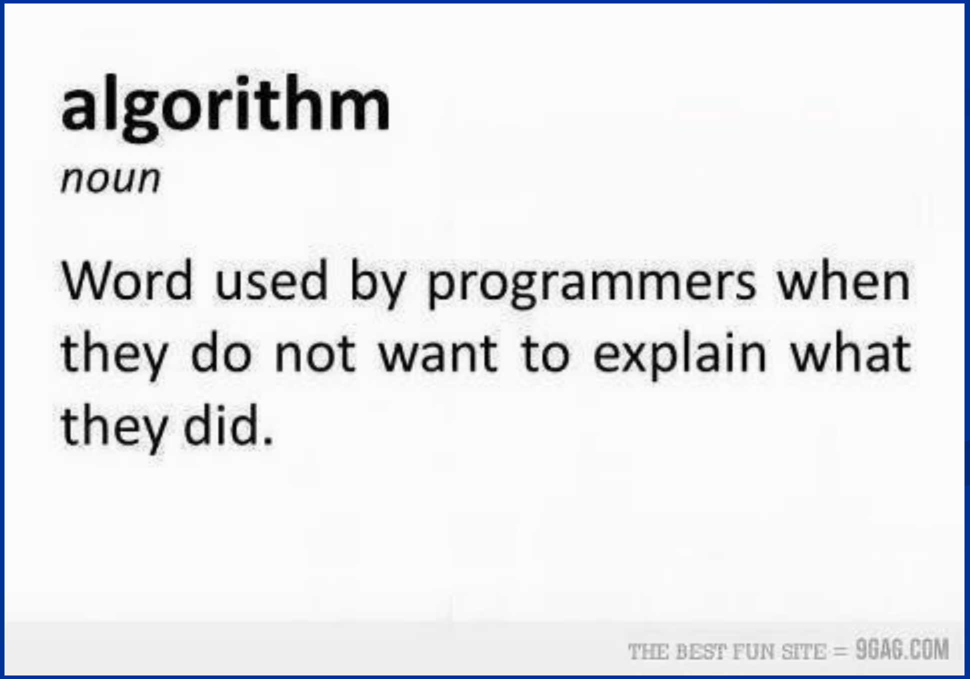
We use the GPU for even faster computing.

GPU is designed to do linear algebra.



The Algorithm

1. Take the Asian Option PDE, make it into a discrete stochastic problem.
2. Use MC simulations to get a distribution of prices.
3. Use CV methods to narrow distribution
4. Use Brownian Bridge to compartmentalize the code
5. Parallel port to speed convergence
6. Test each phase (speed, accuracy)



Conclusion

Conclusions

- MC alone requires about 100,000 iterations to give us useful results. If we add CV we found that we only need 16,000-18,000 iterations to reach the same accuracy.
- While the CV method does add 5% more to the computing time as compared to naïve MC. We make up for it by cutting down the number of iterations by half.
- The Brownian Bridge adds some complexity to the code. An additional 10% of computing time. But allows for 96% of the code to be parallel executed.
- Convergence time is negligible compared to that of naïve MC using parallel computing.

Conclusions

Trading and pricing of these derivative options can be very quick with a combination of these tools.

Future Work

- Increase stochastic operators, add jumps
- Reserve time on University Supercomputer, test against CPU clusters
- Test speeds using APU coupled with CPU
- Get a Kepler architecture GPU, anyone have a Titan card or a GTX 780 Ti they want to lend me?
- Add antithetic sampling and other random number manipulators for faster convergence.

Questions?