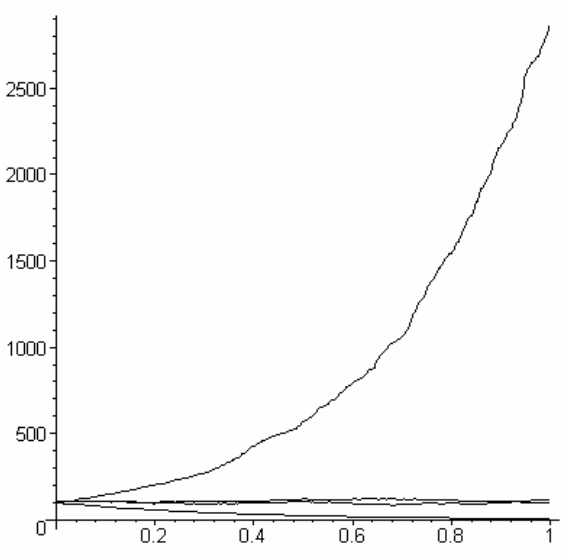
Pricing Asian Options



Introduction

This worksheet demonstrates the use of Maple for computing the price of an Asian option, a derivative security that has gained popularity in financial markets in the last fifteen years.

The payoff of an Asian option is based on the difference between an asset's average price over a given time period, and a fixed price called the strike price. Asian options are popular because they tend to have lower volatility than options whose payoffs are based purely on a single price point. It is also harder for big traders to manipulate an average price over an extended period than a single price, so Asian options offers further protection against risk.

One disadvantage of Asian options is that their prices are very hard to compute using standard techniques. Unlike European options, which can be priced using the classic Black-Scholes formula, there is no analytical formula for pricing an Asian option when

the underlying asset is assumed to have a lognormal distribution, which is par for the course in financial modeling.

In this application, we use two different approaches for computing these prices numerically. (1) Solving a partial differential equation, and (2) Monte Carlo simulation. We will see that the two numerical solutions that Maple derives are the same, providing strong validation for both the techniques and Maple's numerics.

In doing so, the following Maple techniques are highlighted:

- Solving partial differential equations numerically
- Calling external C++ routines from within Maple

Problem statement

Arithmetic average Asian options are securities whose payoff depends on the average of the underlying stock price over a certain period of time. To be more precise, the value of the continuous arithmetic Asian call option at time $t \le T$ is given by

$$C_{t,T}(K) = \mathbf{e}^{(-r(T-t))} E\left[\left(\frac{\int_0^T \mathrm{S}(u)\,du}{T} - K\right)^+\right]$$

where $\mathrm{S}(t)$ is the stock price at time t, T is the expiration date, and K is the strike price.

Since no general closed form solution for the price of the arithmetic average Asian option is known, a variety of numerical methods have been developed. These include formulation as a PDE, Monte Carlo simulation, and numeric inversion of the Laplace transform.

In this worksheet we demonstrate how these methods can be easily and quickly implemented in Maple.

Solution 1: PDE formulation

It was shown by Vecer ("A new PDE approach for pricing arithmetic average Asian options", Journal of Computational Finance, vol. 4, No. 4, 105-113) that the value of the Asian option at time 0 is given by

$$S_0\, u\left(0, \frac{S_0 - K}{S_0}\right)$$

where the function $\mathrm{u}(t, z)$ satisfies the following PDE

$$u_t + r\left(1 - \frac{t}{T} - z\right) u_z + \frac{\left(1 - \frac{t}{T} - z\right)^2 \sigma^2\, u_{z,z}}{2} = 0$$

with the boundary condition

$$\mathrm{u}(T, z) = z^+$$

We first set up the equation:

```
> q := t -> 1-t/T:
> pde := diff(u(t, z), t)+r*(q(t)-z)*diff(u(t, z),
  z)+1/2*(q(t)-z)^2*sigma^2*diff(u(t, z), z, z) = 0;
```

$$pde := \left(\frac{\partial}{\partial t}\mathrm{u}(t, z)\right) + r\left(1 - \frac{t}{T} - z\right)\left(\frac{\partial}{\partial z}\mathrm{u}(t, z)\right) + \frac{1}{2}\left(1 - \frac{t}{T} - z\right)^2 \sigma^2 \left(\frac{\partial^2}{\partial z^2}\mathrm{u}(t, z)\right) = 0$$

Boundary conditions.

```
> ibc1 := u(T, z) = Heaviside(z)*z;
  ibc2 := u(t, -1) = 0;
  ibc3 := D[2](u)(t, 1) = 1;
```

$$ibc1 := \mathrm{u}(T, z) = \mathrm{Heaviside}(z)\, z$$

$$ibc2 := \mathrm{u}(t, -1) = 0$$

$$ibc3 := \mathrm{D}_2(u)(t, 1) = 1$$

Let's assume a fixed time period of 1 year, a starting price of $100, a strike price of $100, and $\sigma = .2, r = .15$

```
> T := 1; S0 := 100.; K := 100.; sigma := 0.2; r := 0.15;
```

$$T := 1$$

$$S0 := 100.$$

$$K := 100.$$

$$\sigma := 0.2$$

$$r := 0.15$$

Now we solve the PDE numerically using Maple's built-in solver.

```
> pds := pdsolve(pde, {ibc1, ibc2, ibc3}, numeric,
  spacestep = 0.001, timestep = 0.001):
  val := pds:-value(t = 0, output = listprocedure):
  v := eval(u(t, z), val):
```

Finally, we evaluate $S_0\, u\left(0, \frac{S_0 - K}{S_0}\right)$ at Maple's numerical solution of $u(z, t)$. This gives a price of $8.40088. This will provide a benchmark for the next solution approach.

```
> S0*v((S0-K)/S0);
```

$$8.4087879672540930000$$

Solution 2: Monte Carlo simulation

Monte Carlo is one of the most popular methods for pricing securities whose prices are given in terms of expected values. The approach consists of the following steps:

- simulate sample paths of the underlying assets
- evaluate the discounted cash flows of the derivative security on each sample path
- average the discounted cash flows over the sample paths.

To generate sample paths, we will use Marsaglia's random variate generator, implemented as an external C++ routine:

```
void MCInitialize()
{
    Ziggurat::Reset();
}
```

Maple's interface to the above C++ routine is defined in the Initialization section which is not shown in this file:

The following C++ function generates a sample stock path.

```
void MCSamplePath(double *S, int N, double T, double S0,
double r, double sigma)
{
    double h = 1./(N-1);
    double C = exp(r*h-sigma*sigma*h/2);

    S[0] = S0;
    for (int i = 0; i < N-1; i++) {
        S[i+1] =
S[i]*C*exp(sigma*sqrt(h)*Ziggurat::Next());
    }
}
```

We have defined a corresponding Maple routine called **MCSamplePath** that invokes this C++ routine:

```
> T := 1.: sigma := 0.2: r := 0.15: S0 := 100.:
```

Generate a sample path

```
> MCSamplePath(S, N, T, S0, r, sigma);
```

Plot the sample path.

```
> plots[pointplot]([seq([(i-1)*T/N, S[i]], i = 1..N)],
  connect = true);
```

```
> K := 100.;
```

$$K := 100.$$

The estimated price of the option using 10 sample paths

```
> ArithmeticAsian(T, S0, r, sigma, K, 10^4, 10,
  MCSamplePath);
```

$$7.686273473$$

And using 10,000 sample paths. We note that the estimated price is closer but not identical to the results obtained by the PDE approach.

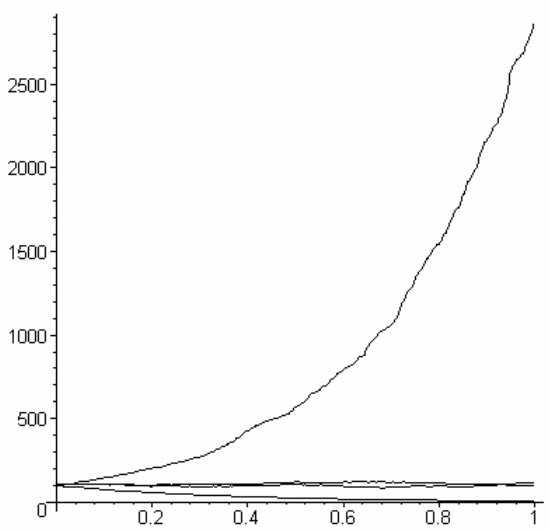
```
> ArithmeticAsian(T, S0, r, sigma, K, 10^4, 10^4,
  MCSamplePath);
```

$$8.267979498$$

One of the main weaknesses of the MC approach is that it tends to waste points due to clustering. This compromises performance when the sample size is small (or the dimension is large). We can often improve the performance by holding more regularly distributed (usually deterministic) points using a technique called Quasi Monte Carlo simulation, which we do using the C++ routine **QMCSamplePath**.

```
> P := Array(1..4):

  for j from 1 to 4 do
     QMCSamplePath(S2, N2, T, S0, r, sigma);
     P[j] := plots[pointplot]([seq([(i-1)*T/N2, S[i]], i =
     1..N2)], connect = true);
  end do:
  plots[display](convert(P, list));
```



Using the Quasi Monte Carlo simulation method, we arrive at virtually the same price as with the PDEs approach.

```
> ArithmeticAsian(T, S0, r, sigma, K, N, 10^5,
  QMCSamplePath);
```

8.412020983