****

These answers indicate the solutions to the 2014 exam questions. Obviously you should plot graphs where I have simply described the key features. It is important when plotting graphs to label the axes etc. and to mention key features like approximate gradients that are important when interpreting the graph.

1. (a) By Ito's Lemma,

$$\mathrm{d}(lnS) = (\mu - (1/2)\sigma^2)\mathrm{d}t + \sigma \mathrm{d}W_t$$

Hence

$$\mathrm{d}(\frac{lnS - (\mu - (1/2)\sigma^2)t}{\sigma}) = \mathrm{d}W_t$$

So by the definintion of Brownian motion, we can simulate stock prices at time point i as follows. Generate normally distributed (mean 0, s.d. 1) ϵ_i. Define $s_0 = \log(S_0)$. Then define

$$s_i = s_{i-1} + (\mu - (1/2)\sigma^2)i\delta t + \sigma\sqrt{\delta t}\epsilon_i$$

Hence define $S_i = \exp(s_i)$.

(b) The process above with $\mu = r$. Run lots of simulations in the Q measure, compute the payoff in each case. The price is sample mean discounted payoff.

(c) The sample standard deviation of the discounted payoff is an unbiased estimator of the standard deviation. So by central limit theorem, error is approx sample standard deviation divided by root number of paths.

(d) If the barrier is high, we expect the call price, if the barrier is low it should be zero.

(e) Decompose the code into two pieces, test the monte carlo pricing by pricing a call option, test computing the payoff of a knockout option separately. For high barriers, one would expect to see the call option price. (Either will do as an answer)

See Next Page

2. (a) See the lecture notes for the definitions. VaR is easy to understand and easy to back test but suffers from hiding risk in the tail. Expected shortfall deals with the tail, but at the expense of straightforward back testing.

(b) Parameteric VaR approximates returns on risk factors as being normally distributed. The returns of more complex products are approximated as depending linearly on these risk factors. One can then use analytic formulae to compute VaR. This method is fast, but highly inaccurate for nonlinear products and tied to a particular risk model.

Monte Carlo VaR is calculated by performing repeated P measure simulations using the stochastic model of your choice and then calculating the desired percentiles. It is very flexible but computationally expensive. The choice of stochastic model is subjective.

Historic VaR uses historical log-return data to provide scenarios. For each historical scenario, the risk factors today are scaled to change in a manner proportionate to the historical changes. This is done by multiplying historic log-returns over a 1-day time period by $\sqrt{d}$ to obtain d-day returns. One can then use these log-returns to compute the change in asset prices and the portfolio is then repriced accordingly. One then reads off the given percentile. Its key advantage is the lack of subjectivity of the model, but it suffers from a lack of historical data and the assumption that the past will reflect the future.

(c) Note that the code below could be simplified a little by removing the logs.

```
function var = computeVar( historicalPrices, quantity, price )
n = length(historicalPrices);
logReturns = log(historicalPrices(2:n)./historicalPrices(1:n-1));
profits = quantity*price*(exp(logReturns)-1);
var = prctile( -profits, 99);
end
```

See Next Page

3. (a) The strategy is to ensure that, at all times, your portfolio (including) your liabilities has a Black Scholes delta of zero.

(b) At time 0 we receive P and purchase delta stocks. Bank balance is $b_0 = P - \Delta_0 S_0$. At time point i we receive interest and purchase $\Delta_i - \Delta_{i-1}$ stocks. So the new bank balance is $b_i = e^{r\delta t} b_{i-1} - (\Delta_i - \Delta_{i-1}) S_i$. At time point N we dissolve our stock holding and make good on our liability, so $b_N = e^{r\delta t} b_{N-1} + \Delta_{N-1} S_N - \max\{S_N - K, 0\}$.

(c)
```
S = generatePricePaths();
delta = blackScholesDelta(sValues);
b(0) = P - delta(0)*S(0)
for (i=1:N-1)
    b(i)=exp(r*dt)*b(i-1) - (delta(i)-delta(i-1))*S(i);
end
b(N) = exp(r*dt)*b(N-1) + delta(N-1)*S(N) - max(S(N)-K,0)
```

(d) The histogram should be centered on zero and look approximately normal.

(e) The graph should be of root mean squared error or some similar measure against N. It should have gradient $-\frac{1}{2}$. This means that delta hedging converges rather slowly.

(f) If you add in transaction costs, after a brief period of improvement as N, the graph begins to move upwards.

See Next Page

4. (a) Let N be number of steps. Define

$$h = (b - a)/N;$$

Define

$$s_n = a + (n - \frac{1}{2})h$$

Then the rectangle rule estimate for the integral of f is

$$\sum_{n=1^N} hf(s_n)$$

(b)
```
function r=integral(f, a, b, N)
h = (b-a)/N;
s = a+ ((1:N)-1/2);
r = 0;
for i=1:N
    r = r + h*f(s(i));
end
end

function example() {
    function ret=expXSquared(x)
        ret = exp( -x*x);
    end
    integral( @expXSquared, a, b, 1000);
}
```

(c) One choice would be the substitution $y = N(x)$ where N is the pdf of the normal distribution.

(d) The trapezium rule has slope -2, Simpson's rule has slope -4, Monte Carlo has slope -1/2. For high numbers of steps, the lines should start going up with an approx gradient of 1/2 due t rounding errors.

(e) The pricing kernel is the pdf in the risk neutral measure. So integrating the payoff times the pricing kernel gives the risk neutral price.

See Next Page

5. (a)

	Rectangle rule	Monte Carlo	Finite Difference
European Call Option	Yes	Yes	Yes
Knockout Call Option		Yes	Yes
Asian Call Option		Yes	
American Call Option			Yes

(b) The quickest way to do this is using the Feynman–Kac theorem. The S.D.E. associated by Feynman–Kac to the given p.d.e. is:

$$\mathrm{d}S = S\sigma\,\mathrm{d}W_t$$

so writing $z = \log S$ we have by Ito's lemma:

$$\mathrm{d}z = -\frac{1}{2}\sigma^2\,\mathrm{d}t + \sigma\,\mathrm{d}W_t$$

hence writing $x = \log S + \frac{1}{2}\sigma^2 t$ we have that x satisfies:

$$\mathrm{d}x = \sigma\,\mathrm{d}W_t$$

So by the Feynman–Kac theorem the associated p.d.e. for the expectation of a function of x at time T is:

$$C_t = -\frac{1}{2}\sigma^2 C_{xx}$$

Alternatively you might guess the change of variables and use brute force as follows:

Define $x = \log S + \frac{1}{2}\sigma^2 t$ and $\tau = t$. It's important to introduce the variable τ otherwise you will find the partial differentiation formulae misleading. By the chain rule we have:

$$\begin{aligned}\frac{\partial C}{\partial S} &= \frac{\partial C}{\partial x}\frac{\partial x}{\partial S} + \frac{\partial C}{\partial \tau}\frac{\partial \tau}{\partial S}\\ &= \frac{1}{S}\frac{\partial C}{\partial x}\end{aligned}$$

Similarly.

$$\frac{\partial C}{\partial t} = \frac{1}{2}\sigma^2\frac{\partial C}{\partial x} + \frac{\partial C}{\partial \tau}$$

Differentiating the first of these formulae and applying the chain rule again gives:

$$\frac{\partial^2 C}{\partial S^2} = -\frac{1}{S^2}\frac{\partial C}{\partial x} + \frac{1}{S}\frac{\partial}{\partial S}\left(\frac{\partial C}{\partial x}\right) = -\frac{1}{S^2}\frac{\partial C}{\partial x} + \frac{1}{S^2}\frac{\partial^2 C}{\partial x^2}$$

See Next Page

Putting these together we have

$$\frac{\partial C}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial C}{\partial S^2} = 0$$

is equivalent to:

$$\frac{\partial C}{\partial \tau} = -\frac{1}{2}\sigma^2 \frac{\partial C}{\partial x^2}$$

one can now write this as

$$C_\tau = -\frac{1}{2}\sigma^2 C_{xx}.$$

The reason why you need to introduce τ is that $\frac{\partial}{\partial t}$ is a slightly unclear notation. In the context of the Black–Scholes equation it means "the derivative with respect to time holding S fixed". So even though we've defined $t = \tau$ it's better to write $\frac{\partial}{\partial \tau}$ whenever we mean "the derivative with respect to time holding x fixed".

(c)

$$\frac{u_{t+1,x} - u_{t,x}}{\delta t} = -\frac{1}{2}\sigma^2 \frac{u_{t+1,x+1} - 2u_{t+1,x} + u_{t+1,x-1}}{\delta x^2}$$

$$u_{t,x} = \beta u_{t+1,x+1} + (1 - 2\beta)u_{t+1,x} + \beta u_{t+1,x-1}$$

Where $\beta = \frac{1}{2}\sigma^2\alpha$. Starting from the boundary condition of the final payoff, step backwards in time using this recurrence relation. Use additional boundary conditions as appropriate on the boundary of the solution region - for example for a knock out option one might take the boundary condition that the option has value zero on the boundary.

(d) We interpret β as a risk neutral probability of the the variable x moving up or down by δx. One can consider it as being calibrated by requiring that the resulting variance corresponds to σ.

(e) A numerically stable algorithm is one whose values remains stable when small random rounding errors are added. The stability condition is that $(1 - 2\beta) > 0$, which we can interpret as saying that it must represent a probability.

Final Page