First exam in **Computational Finance**

for participants of MSc. Quantitative Finance

Summer term 2017, Mathematisches Seminar, CAU Kiel

Lecturer: Prof. Dr. Vetter

Please note:

- Before you start, write your name **on all pages** of the solution sheet and **sign below**.
- This exam contains **5 exercises**.
- You have **150 minutes** of time.
- You may give your answers in **German or English**.
- When specifying **modifications of Scilab codes**, use the provided line numbers. E.g., write “insert after line 10: ...” or “replace line 20 by: ...”.
- Cosmetic changes of variable or function names, etc. are not necessary.

I hereby certify that I am capable to take the exam: ______________________

Exercise	1	2	3	4	5	Σ	grade
Points							

Exercise 1 (4 + 4 + 4 = 12 points)

Please answer the following questions in complete sentences.

(a) Suppose that we would like to compute option prices in the Black-Scholes model. Give a formal definition of the Cox-Ross-Rubinstein model with mesh $\Delta t = T/m$ and parameters q, u and d. How should the parameters be chosen in order to match the Black-Scholes model and what is the main idea behind the choice of $ud = 1$?

(b) We would like to price European options $X = f(S(T))$ in the Black-Scholes model. How can a pricing formula be derived using partial differential equations? Explain the main ideas behind this approach and sketch the necessary changes once we are interested in American options.

(c) Suppose that we want to price a European option. Give a pricing formula for $V(t)$ in terms of the bilateral Laplace transform and the conditional characteristic function and explain any additional notation. How can this approach be modified when we are interested in hedging?

Exercise 2 (5 + 3 + 2 = 10 points)

Let W be a standard Brownian motion and suppose that X is a stochastic process which evolves according to the stochastic differential equation

$$dX(t) = adt + bdW(t)$$

with $X(0) = 2$. Let further be $f : \mathbb{R} \to \mathbb{R}$ a twice continuously differentiable function.

(a) State the integral representation of $f(X(t))$. Explain your answer.

(b) Which conditions on a, b and f have to hold in order for $f(X(t))$ to be a martingale? Explain your answer.

(c) How do these conditions look like explicitly in the case of $f(x) = \exp(x)$?

Exercise 3 (3 + 5 + 2 = 10 points)

In this exercise we consider the importance sampling method for reducing the variance of a Monte-Carlo estimator. Specifically we will always work within a Black-Scholes model.

a) Describe the importance sampling method for variance reduction of Monte-Carlo estimation and explain in which situations it is used. Use a European call option with strike K and maturity T as an example.

b) Suppose we want to compute the price of a European call option with strike $K = 200$ and maturity $T = 2$ in the Black-Scholes model with parameters $r = 0.05, \sigma = 0.2$ and initial stock price $S(0) = 100$. For the computation of the price we want to use a Monte-Carlo approach where we use importance sampling for reducing the variance. Explicitly we want to replace the Brownian motion $W(T)$ at time T by a shifted Brownian motion $\widetilde{W}(T) = W(T) + \mu$. Write pseudo code for a function of the form

```
[V0, epsilon] = BS_EuCall_IS(S(0),r,sigma,T,K,N,mu)
```

that returns $V_{IS}(\mu)$ and the radius ε of a 95%-confidence interval for the true price

$$V(0) = e^{-rT}\mathbb{E}[(S(T) - K)^+]$$

based on the input parameters using the importance sampling method as described above.

Remark: All pre-implemented Scilab functions may be used in your pseudo code without further explanation.

c) Suppose we want to price an option which pays

$$g(S(1), S(2)) = \Big(\frac{S(1) + S(2)}{2} - 200\Big)^+$$

at time $T = 2$ instead of the European call option from part b). How could the importance sampling approach from part b) for the European call option be modified for pricing this Asian option?

Hint: The common density of a vector (X_1, X_2) of independent normally distributed random variables with means μ_1, μ_2 and variances equal to 1 is

$$f(x_1, x_2) = \frac{1}{2\pi} e^{-\frac{1}{2}((x_1-\mu_1)^2+(x_2-\mu_2)^2)}.$$

Please write your solution for this exercise on the extra sheets!

Exercise 4 (2 + 2 + 2 + 4 = 10 points)

In this exercise we discuss the rate of convergence of the Monte Carlo estimator for a European average call. Let $Y_{\Delta t}$ be a simulation of a path of a geometric Brownian motion X with mesh size Δt using the Euler method. The Richardson extrapolation uses terms of the form $Z = 2f(Y_{\Delta t}) - f(Y_{2\Delta t})$ to achieve a faster rate of convergence compared to the use of the simpler terms $f(Y_\Delta)$.

(a) Assuming that the Euler method has weak order of convergence one for some sufficiently smooth function $f(x)$ explain why one can expect that the Richardson extrapolation then yields weak order of convergence two for the same function.

If the paths on the fine grid $Y_{\Delta t}$ and on the coarse grid $Y_{2\Delta t}$ are simulated with high correlation, the estimator has a smaller variance compared to the uncorrelated simulation.

(b) Calculate the variance of the estimator.

(c) How can one achieve a high correlation of those two simulations in the Euler scheme? Sketch a possible solution.

(d) Below is listed a code which calculates the price of a European option in a Black-Scholes model with the plain Euler method using M paths. How does the code have to be changed to calculate the price via a variant of the Richardson extrapolation which uses N steps on the coarse grid?

```
function V_0=BS_AvCall(S0, mu, sigma, T, r, N, M, K)

    function V_T=price_average(X, K)
        V_T = max(mean(X)-K,0)
    endfunction

    Delta_t=T/N;

    V=zeros(M);
    for j=1:M
        Delta_W=grand(N, 1, 'nor', 0, sqrt(Delta_t));
        X=S0*ones(N+1);
        for i=1:N
            X(i+1)=X(i)*(1+mu*Delta_t+sigma*Delta_W(i));
        end
        V(j)=price_average(X, K);
    end

    V_0=exp(-r*T)*mean(V);
endfunction

//Parameters
S0=100;
mu=0.05;
sigma=0.2;
T=1;
N=250;
r=0.03;
M=1000;
K=90;

V0=BS_AvCall(S0, mu, sigma, T, r, N, M, K)
```

Please write your solution for this exercise on the extra sheets!

Exercise 5 (3 + 4 + 3 = 10 points)

In this exercise we consider the explicit and implicit finite differences scheme for pricing European options.

a) The code below computes the price of a European put option in the Black-Scholes model using the explicit finite differences scheme. Suppose that we want to price a European call option instead. How has the code to be changed?

b) Suppose that in addition to pricing a European call we would like to use an implicit finite differences scheme. What additional changes have to be made in the code?

   Hint: You may assume that the standard solver in Scilab that solves equations of the form $Ax = b$ for x with $A \in \mathbb{R}^{d\times d}$, $x, b \in \mathbb{R}^d$ automatically recognizes if A is a tridiagonal matrix and works efficiently in that case. The Scilab-code for the standard solver for the linear equation $Ax = b$ is `x = A\b` .

c) Comparing the explicit and implicit finite differences scheme, what is the main advantage of using the explicit scheme over the implicit one and what is the main advantage of using the implicit scheme over the explicit one? What is the main advantage of the Crank-Nicholson scheme compared to the other two schemes?

```
function [V0, S] = BS_EuPut_FiDi_Explicit (r, sigma, a, b, m, nu_max, T, K)
    w = zeros(m + 1, 1);

    delta_t = 0.5*sigma^2*T / nu_max;
    delta_x = (b-a) / m;

    lambda = delta_t / delta_x^2;
    if (lambda >= 1/2) then
        error("Finite difference scheme unstable.");
    end

    t_tilde = (0:1:nu_max) * delta_t;
    x_tilde = a + (0:1:m) * delta_x;

    q = 2*r/sigma^2;
    alpha = 0.5 * (q-1);
    beta = 0.5 * (q+1);

    w = max( exp(alpha*x_tilde)-exp(beta*x_tilde), 0 );

    w(m+1) = 0;

    for nu=1:nu_max
        w(2:$-1) =  lambda*[w(1:$-2)] + (1-2*lambda)*w(2:$-1) + lambda*[w(3:$)]
        w(1) = exp(alpha*a + alpha^2*t_tilde(nu+1))-exp( beta*a + beta^2*t_tilde(nu+1) );
    end

    V0 =( K .* w .* exp(-alpha*x_tilde - 0.5* sigma^2*T*(alpha^2 + q)));

    S = K.*exp(x_tilde);
endfunction
```

Please write your solution for this exercise on the extra sheets!