Mandatory exercises

1. (a) Explain briefly the difference between a deterministic and a stochastic method.
   (b) Give an example of using a stochastic method to solve a deterministic problem.
   (c) Give an example of a stochastic process?

2. The integral

   \[ I = \int_{0}^{1} x^2 e^x \, dx \]

   can be computed analytically and the exact value is \( I = e - 2 \). We can use this integral as a test case for numerical integration methods, to see how well they are able to approximate \( I \). When the Trapeziodal Rule was applied with step size \( h = 0.25 \), the absolute error in the computed, approximate value of \( I \) was ca. 0.04. Using Simpson’s Rule with the same step size gave an absolute error of ca. 0.0006. The step size \( h = 0.25 \) corresponds to using \( n = 5 \) discretization points.

   You should now compute an approximation to \( I \) by using the Monte Carlo method that is described in Hellander’s compendium. Use \( n = 5 \) random values. Compare the approximation to the exact value of \( I \). What is the absolute error? Compare to the errors obtained for the Trapezoidal and Simpson’s Rule, respectively.

   If you repeat the experiments with more and more random numbers, explain how the error will then behave.

3. If we compare Monte Carlo with Simpsons rule for integration, for what dimension is it more preferable to use Monte Carlo methods? The error in Simpson is of order \( h^4 \).

4. As a model for how the price of a stock changes in time, a so called stochastic differential equation (SDE) is often used. The part of the model that describes the randomness in the value of the stock involves the same Brownian motion as you saw in the lab assignment. Assume that you are given a Matlab function

   \[
   \text{function } x = f(x0,T)\]

   If you repeat the experiments with more and more random numbers, explain how the error will then behave.
that simulates 10 stocks according to an SDE model. The input parameters are the initial value \( x_0 \) and final time \( T \). The vector \( x_0 \) contains the values of the stocks at initial time \( t = 0 \). The output parameter \( x \) is an array with 10 elements, containing the simulated values of the stocks at time \( T \). Write a Matlab script (with pen and paper) that uses a Monte Carlo method to approximate \( E_{x,T} \), the expected total value of the portfolio at time \( T \). The program should also compute the confidence interval for the computed value, according to formula (9) in Hellander’s compendium. You can use the built in function \texttt{std(x)} in Matlab to compute the standard deviation.

Note: you are not supposed to write the function \( x = f(x_0,T) \). You should use it to approximate \( E_{x,T} \).

Non-mandatory exercises

1. In Miniproject 3 we will need to generate random numbers from an exponential distribution

\[
f(\tau; a_0) = \begin{cases} a_0 e^{-a_0 \tau}, & \tau \geq 0 \\ 0, & \tau < 0 \end{cases}
\]

by using inverse transform sampling. Our goal is to derive a formula to sample exponentially distributed random numbers \( \tau \), given uniformly distributed random numbers \( u \).

(a) Compute the cumulative distribution function (CDF) \( F(t) = P(\tau \leq t) \) for the exponential distribution.

(b) Find the inverse of \( F \). That is, for a random number \( u \) uniformly distributed in \([0,1]\) find \( \tau \) such that \( F(\tau) = u \). Then \( \tau = F^{-1}(u) \).

2. In computations based on discretization, such as numerical integration and numerical solution of ODE, it is a common strategy to place the discretization points adaptively, with the objective to have more points in areas where the solution changes rapidly, and fewer points in regions where the solution has a more constant behaviour. The corresponding strategy in Monte Carlo integration is known as importance sampling. Instead of computing

\[
I_1 = \int_\Omega g(x)dx
\]

by sampling \( x \) from a uniform distribution, we choose a function \( p(x) \) and compute

\[
I_2 = \int_\Omega \frac{g(x)}{p(x)}p(x)dx
\]

(a) How would you use a Monte Carlo approach to compute \( I_2 \)?

(b) In principle, how would you choose \( p(x) \) in order for the variance in the approximation of \( I_2 \) to be smaller than that for \( I_1 \)?