Problem set 1

Workout 0.1.

(a) Explain briefly the difference between a deterministic and stochastic *model*. Exemplify.

(b) Explain briefly the difference between a deterministic and stochastic *method*. Exemplify.

Workout 0.2. Assume a Markov chain with three states A, B and C. The reactions (and parameters) is described by the model

$$\begin{aligned} &A \xrightarrow{\alpha} B \\ &A + B \xrightarrow{\beta} 2B \\ &C \xrightarrow{\gamma} A + B \end{aligned}$$

Set up the stoichiometry (state-change) matrix and the propensity functions for this particular model.

Workout 0.3.

(a) Calculate an approximation for the integral

$$I = \int_1^3 x^2 \sin(x) dx$$

using the Monte Carlo method. Use for example 10 random points (or 100 or more). You can use the random number generator in Python (`np.random.rand` or `np.random.uniform`), and you can of course use Python to calculate the function-values too. If you use `np.random.rand` do not forget to transport points from interval $[0, 1]$ to interval $[1, 3]$.

(b) The integral can be solved numerically with a deterministic method, for example using `scipy.integrate`. This is not part of this course and you do not need to do it here, but the solution will be 5.55342325 (rounded value). This value can work as the "exact" solution here. Calculate the absolute error in your calculation in (a).

(c) Assume that you repeat the calculation in (a) many times (M times). The integral approximation will vary as it is based on random numbers. What distribution will the different approximations of I tend to? Given that distribution, how can you estimate the accuracy in the computation?

(d) Let's say you get a certain error in the computation in (a), and you would like to reduce it with a factor $\frac{1}{4}$ (roughly). Explain what you must change in (a) and motivate why.

Workout 0.4. Brownian motion can be explained as the erratic movement of particles in a fluid, like water or oil. It is an important stochastic process in sciences. For example, it is used to model particles in biological membranes (that separates the cell from the external environment), diffusion of pollutants in the air, diffusion of calcium through the bones, movement of "holes" of electrical charge in semiconuctors etc.
Consider Brownian motion in 1D, denoted $X(t)$, where t is time. One step, h, in time can be expressed $X(t+h) = X(t) + \sqrt{h} \cdot Z$, where $Z \in \mathcal{N}(0,1)$ (meaning Z is a normal distributed random number with mean 0 and variance 1). In Python it can be generated with `numpy.sqrt(h)*numpy.random.randn()`

(a) What type of statistical process is this one example of? Motivate why?

(b) Assume a certain number of particles starting at a particular point x_0 in space at time $t = 0$. What is the effect of the formula when it is applied on the particles? (No calculation needed here, just explain in words).

Workout 0.5. The way the price of a stock changes over time can be modelled by a so called stochastic differential equation (you don't need to know anything about these kind of equations here, though). One part of the equation, that describes the randomness of the stock prices, is stochastic and this part is based on Brownian motion.
Assume that a Python function `stock` is available

```
x = stock(x0,T)
```

The function simulates 10 stocks. The input parameters are the initial values x_0 and T is the final time, where x_0 is an numpy array containing the 10 values of the stocks at time $t = 0$. The output parameter `x` is a vector (a numpy array) with 10 elements, containing the simulated values of the stocks at time T.

(a) Write a Python script or algorithm (pen and paper) that uses a Monte Carlo method to approximate the expected value of the full portfolio at time T. (Full portfolio means the sum of values of all 10 stocks).

(b) We would also like to know how accurate the solution is. Write a code (or algorithm) that computes the confidence interval for the simulated expected value (the mean). You can use the built-in function `numpy.std(x)` in Python to compute the standard deviation.

Note: You are not supposed to write the function `stock`, you are just supposed to use it.

Non-mandatory workouts:

Workout 0.6. Consider the function

$$f(x) = \frac{1}{W}\frac{1}{1+x}, \quad x \in [0, 20]$$

(a) Obtain the constant W such that f represents a probability distribution function (pdf) in interval $[0, 20]$.
(b) Compute the cdf F and use the inverse transform method to generate random variables from f. Write all details and the steps of the algorithm.

Workout 0.7. Consider a game where you roll a fair four-sided dice with sides labeled 1, 2, 3, and 4. You win \$1 if the dice shows 1 or 2, you win \$2 if the die shows 3, and you lose \$1 if the dice shows 4. How can the Monte Carlo simulation method be used to estimate the probability that your winnings will be negative after 10 rolls of the dice?

Workout 0.8. Suppose that we model the motion of a single large protein in surrounding of water molecules and that the system in confined to box $[-1, 1]^3$. The protein is pushed by water molecules and moves randomly at each time step (Brownian motion). Write a Python code and use Monte Carlo to estimate the *expected time* until the particle hits one of the sides of the box if it starts from the origin $[0, 0, 0]$. Also compute the confidence interval (error) of 95% probability for the simulated expected value.