1. **Random Number Generators**
The traditional random number generator (RNG) in computer science generates a sequence of the form

$$X_{n+1} = (aX_n + c) \bmod k. \tag{1}$$

(a) Verify the point made in the lecture that such pseudo-random numbers are highly correlated.

(b) Try $k = 256$ and $k = 1024$ and choose good parameters for $a$ and $c$. Can you explain the behavior?

(c) The performance of a bad RNG can be vastly improved by coupling two bad RNGs. We want to study this possibility. Run two differently seeded versions (also try to use different $a$, $c$, and $k$) of the above bad RNG. A random number from RNG 1 is only taken if RNG 2 produces a number that is a multiple of either 2, 3, 5, 7, or 13. Study the correlation of this new RNG.

2. **Numerical integration**

$$\int_0^\infty e^{-x}\,\mathrm{d}x \tag{2}$$

$$\int_0^\pi \sin(x)\,\mathrm{d}x \tag{3}$$

$$\int_0^1 \frac{\ln(\cos(x))}{x}\,\mathrm{d}x \tag{4}$$

(a) In the last term, we discussed numerical integration of a known function. This exercise is meant to warm up your skills on this topic of Computational Physics I. Use the *Trapezoidal* rule and *Simpson's* rule to compute the integrals given above. Compare the relative accuracy of the result to the number of function evaluations needed. The numerical value of Eq. (4) is $-0.27568727380043716\cdots$. Visualize the result in a log-log plot and determine the rate of convergence for each method. What rate would you expect? You can find a code skeleton in the code section of the homepage, which includes a plotting script.

(b) Implement *Monte Carlo* integration.

- Implement two different versions of Monte Carlo integration (by-rejection, by-mean) and solve the above integrals. Use a linear transformation from the interval of your random numbers $[0, 1]$ to the integration boundaries $[a, b]$.
- Try to use estimates for the extremal values of the integrand for the by-rejection integration that are two orders of magnitude too large. Can you explain the behavior of the algorithm performance?
- Now, try to use importance sampling for those integrals where it is sensible and potentially improves the integration. Find a suitable probability density function close/similar to the integrand to draw from; do not use the integrand itself.
- Which of all your algorithms performs best? For which problems should you use Monte Carlo integration?