Technische Universität Braunschweig

Prof. Dr. Dirk Langemann & Jannis Marquardt, M.Sc.
Institute for Partial Differential Equations

Assignment 3

- Programming exercises -

Upload your solution until Mo, 08 May 2023, 03:00 pm.

Programming exercise 3.1 *Trapezoid rule* **(5 points)**

(a) Write a function file (a ".m"-file) `trapez.m`, which calculates the approximate value $Q(f)$ of the integral $\int_a^b f(x)\,\mathrm{d}x$ using the trapezoidal rule for an given arbitrary function f and boundaries of the interval $[a, b]$.

(b) Now, write a script in Matlab that calculates the integral of the function $f(x) = \exp(x)$ on the interval $[0, 1]$ using the Matlab function file you have just created. Compute the same integral using the following two commands in Matlab: `q=integral(f,a,b)` and `q=trapz(x,y)`. Compare the three values with the analytical value. What do you observe? Make a comment in the script file.

Programming exercise 3.2 *Monte Carlo integration* **(5 points)**

Consider a given function $f : [a, b] \to \mathbb{R}$ for which we are interested in the value of $\int_a^b f(x)\,\mathrm{d}x$. Monte Carlo integration works similar to the Midpoint rule. Let m be the midpoint of the interval $[a, b]$, i.e. $m = \frac{b-a}{2}$. Instead of approximating $\int_a^b f(x)\,\mathrm{d}x \approx (b-a) \cdot f(m)$ as for the Midpoint rule, we now compute the integral by $\int_a^b f(x)\,\mathrm{d}x \approx (b-a) \cdot f(x)$, where $x \in [a, b]$ is chosen randomly.

As you may imagine, this procedure may not be very accurate for more complicated functions f. Therefore, we do not approximate the integral for a single random number x, but $N \in \mathbb{N}$ randomly chosen numbers $x_1, \ldots, x_N$. The idea which leads to this version of Monte Carlo integration is that the mean value of these N integral approximations is a more accurate result than the single approximations. Hence we compute

$$\int_a^b f(x)\,\mathrm{d}x \approx \frac{1}{N} \sum_{i=1}^{N} \underbrace{(b-a) \cdot f(x_i)}_{\text{single integral approximation}} = \frac{(b-a)}{N} \sum_{i=1}^{N} f(x_i).$$

(a) Create a Matlab function file `monte_carlo.m` which computes the value of the Interval with the Monte Carlo method. In this file, the function `I = monte_carlo(f, a, b, N)` is supposed to get the function `f`, the interval boundaries `a` and `b` and the number of random numbers `N` as arguments.

(b) Let us consider $f(x) = \sin(x)$ on the interval $[1, 3]$ and $N = 10$. Use the function from (a) to compute an approximation of $\int_1^3 f(x)\,\mathrm{d}x$. Execute the code multiple times. Do you always get the same value for the integral? Comment in the code on why the observed behaviour occurs.

(c) Compute the real value of this integral and compare it to the approximate value of the integral which you get with Monte Carlo for $N = 10, 20, 30, \ldots, 1000$. Depict your findings graphically in a plot (with N on the x-axis and the error on the y-axis). Compare the difference between plots created with `plot()` and `loglog()`.