Multivariate Calculus

(Multiple Integrals and Vector Calculus)

Andre Lukas, HT 2026

Computational Problems

Computational methods are increasingly important in physics, science more broadly and society at large. The following problems provide an opportunity to practice some of these methods in the context of this lecture course. They a voluntary but strongly recommended. Solutions can be realised in either Mathematica or Python although other programming languages are possible as well.

Question 1. (Gradient descent)

Write a simple piece of code which finds minima of a given function by performing a gradient descent. Try the code out on a few functions of two variables and illustrate the outcomes by creating contour plots.

Question 2. (Neural networks)

Familiarise yourself with the MNIST data set of hand-written digits. Design a suitable neural network and train it on the MNIST data set, in order to obtain a classifier for hand-written digits. Demonstrate that the trained neural network does indeed work by applying it to test data and by computing the confusion matrix.

Question 3. (Lagrangians and Hamiltonians)

Assume you are given a Lagrangian $L = L(q, \dot{q})$, depending on generalised coordinates $q = (q, \ldots, q_n)$, which describes a system in Classical Mechanics. Write a code which takes L and q as an input and which computes the Euler-Lagrange equations, the Hamiltonian and the Hamilton's equations. Apply this code to the motion of a point particle in spherical coordinates, assuming that the potential only depends on the radius.

Question 4. (Integrating vector fields over curves and surfaces)

Consider vector fields A in n dimensions and curves C parametrised by $X : [a, b] \to \mathbb{R}^n$. Write a piece of code which takes A, X (and the interval $[a, b]$) as an input and which evaluates the curve integral $\int_X A \cdot dx$. Do the same for surface integrals $\int_S A \cdot dS$, with the surface $S \subset \mathbb{R}^3$ specified in terms of a parametrisation $X : U \to \mathbb{R}^3$. For simplicity, assume that U is a Cartesian product of two intervals. Test your code on a few examples.