Numerical Programming Projects (WS 22/23)

- All the projects have to be presented with the corresponding code (Python, C++, ..). An efficient/elegant implementation will be taken into account.
  The presentation (slides or blackboard) must contain a general introduction to the problem, a discussion of the implementation, and results in numerical and graphical form.

- After a discussion about the project, questions related to other connected topics discussed during the course will follow.

1. **Root Finding with numerical derivatives**
   Newton's method for root finding is based on the knowledge of the function's first derivative. If an analitical expression for f'(x) is not pre-calculated, a numerical approximation can be used. Write a code implementing the Newton's method coupled with a numerical determination of the derivative of your choice. Assess the precision of the root finding as a function of the parameters of the numerical derivative (e.g. the step h).

2. **Modification of the fixed-step root finding**
   Consider the fixed-step root finding algorithm and modify it with a simple adaptive-step solution. The idea consists in stepping back while halving the step if the solution is crossed. Check if this method can improve the root finding precision and speed. Remember to implement all the checks for stopping the algorithm. Compare the algorithm with the fixed step one.

3. **Pivoting**
   Wirte a program which implements the pivoting procedure for the solution of lienar systems. Compare the pivoting algorithm to the non-pivoting one on suitable test-systems where pivot elements are small/zero. Investigate how the algorithm scales as a funciton of the matrix dimension (time-complexity).

4. **Epidemiological Model 1**
   Implement the simple SIRV epidemiological model (4 coupled lin-

ear differential equations). Compare the solutions with the Euler's method with the Runge-Kutta method. Calculate the (time-dependent) difference betweeen the two solutions.

$$\begin{cases} \frac{dS}{dt} &= -\beta I \frac{S}{N} - \xi \frac{S}{N} + \alpha V \\ \frac{dI}{dt} &= \beta I \frac{S}{N} - \gamma I \\ \frac{dR}{dt} &= \gamma I \\ \frac{dV}{dt} &= \xi \frac{S}{N} - \alpha V \end{cases} \tag{1}$$

5. **Epidemiological Model 2**
   Implement the simple SIRD epidemiological model (4 coupled linear differential equations). Compare the solutions with the Euler's method with the Runge-Kutta method. Calculate the (time-dependent) difference betweeen the two solutions.

$$\begin{cases} \frac{dS}{dt} &= -\beta I \frac{S}{N} \\ \frac{dI}{dt} &= \beta I \frac{S}{N} - \gamma I - \mu I \\ \frac{dR}{dt} &= \gamma I \\ \frac{dD}{dt} &= \mu I \end{cases} \tag{2}$$

6. **Lorenz System** Implement the Lorenz chaotic system (3 coupled differential equations). Compare the solutions with the Euler's method with the Runge-Kutta method. Calculate the (time-dependent) difference betweeen the two solutions.

$$\begin{cases} \frac{dx}{dt} &= \sigma(y - x) \\ \frac{dy}{dt} &= x(\rho - z) - y \\ \frac{dz}{dt} &= xy - \beta z \end{cases} \tag{3}$$

   with $\sigma = 10$, $\beta = 8/3$, $\rho = 28$. Consider also the case $\rho < 0$ and compare it with the previous (chaotic) case.

7. **Multidimensional Newton's algorithm** Implement the multidimensional Newton's algorithm for finding a root. Couple it with the Gauss Elimination algorithm for solving the Joacobian equation.

8. **Compare Integration Methods: Trapezoid vs Simpson** Implement (as functions) the trapezoid and Simpson methods and compare

their numerical precision using them on functions with a known integral and growing difficulty. Compare the scaling of the time complexity as function of h.

9. **Compare Integration Methods: Simpson vs Romberg** Implement (as functions) the Simpson and Romberg methods and compare their numerical precision using them on functions with a known integral and growing difficulty. Compare the scaling of the time complexity as function of h.

10. **Compare Integration Methods: Monte Carlo vs Importance Sampling** Implement (as functions) the MC and MC+IS methods and compare their numerical precision using them on functions with a known integral (e.g. the Gaussian or a polynomial). Compare the scaling of the time complexity as function of h.

11. **Dimensionality in Integration problems** Compare the precision and speed of the MC method and a simple integration routine in 3 dimensions.

12. **Compare Iterative algorithms: Steepest Descent vs Conjugated Gradient** Compare the speed and precision of the steepest descent and conjugated gradient methods as function of the matrix dimension.

13. **Compare Root finding algorithms: Bisection vs Brent's method** Compare speed and precision of the bisection and the Brent method.

14. **Numerical differentiation: Backwards/Forwards Differences vs Central Differences** Verify experimentally writing a suitable program, that central differences are more accurate than backwards/forwards differences in estimating the derivatives of a function. Check this with functions of growing difficulty (faster variation).

15. **Choleski Decomposition** Solve a liner system with Gauss elimination and compare it with a solution based on the Choleski decomposition. Use an appropriate matrix and compare the time-complexity of the two algorithms as the dimension of the matrix increases.

16. **LU Decomposition** Solve a liner system with Gauss elimination and compare it with a solution based on the LU decomposition. Use

an appropriate matrix and compare the time-complexity of the two algorithms as the dimension of the matrix increases.

17. **Elliptic Equation 1** The Jacobi iteration is a relatively simple way to solve the Laplace equation. An alternative is the solution of the equivalent matrix equation: use one algorithm for solving linear systems for solving the Laplace equation and compare the solution to the Jacobi iteration method.

18. **Elliptic Equation 2** Solve with a Jacobi iteration the elliptic equation

$$\phi_{xx} + \phi_{yy} + A(y\phi_x + x\phi_y) = 0 \tag{4}$$

Choose equal steps in the two dimensions (h=k) and derive the corresponding numerical approximation using central differences for all the derivatives. Use $f(x,y) = x^2 - y^2$ as boundary condition on a grid of positive numbers ($x > 0, y > 0$). Show different solutions as the free parameter A varies.

19. **Parabolic Equation**
Solve the parabolic equation

$$\phi_t = \phi_{xx} + (x-2)\phi_x - 3\phi \tag{5}$$

with the initial boundary conditions
$\phi(x,0) = x^2 - 4x + 5$ for $0 \leq x \leq 4$ and
$\phi(0,t) = \phi(4,t) = 5e^{-t}$ with $t > 0$.
Show that the numerical result approximates the exact solution
$\phi(x,t) = (x^2 - 4x + 5)e^{-t}$.
Solve the problem with an explicit central difference numerical scheme.

20. **Fitting 1** Implement the linear fitting algorithm in the most efficient way you are able to find. Create random datasets and verify the functionality of the algorithm. Modify it adding errors to each point and take them into account in the fit.

21. **Fitting 2** Derive the formulas for the fit of a parabola $y = a + bx + c^2$ and write the corresponding code for the estimation of the three free parameters a, b, c. Try to optimize the code for limiting the amount of calculations. Fit with the parabola many randomly generated datasets and check the distribution of the obtained parameters.

22. **3-Dimensional Integration** Implement the 3D integration algorithm and compare the performance of the Trapeziod vs the Simpson integration. Show how the time-complexity scales as you go from dimension 1 to dimension 3 integration.

23. **Integration of Mechanical Systems** Implement an integration scheme of your choice for the differential equation describing the motion of a pendulum

$$\frac{d\theta^2}{dt^2} + \frac{g}{L}\sin\theta = 0 \quad . \tag{6}$$

Confront and discuss the solution of the latter equation with the known analytical solution obtained with the approximation $\sin\theta \approx \theta$. Solve numerically a second differential equation with the improved approximation $\sin\theta \approx \theta - (1/3!)\theta^3$ and compare it to the previous results.

24. **The Predator-Prey Lotka-Volterra Model** Implement two integration schemes and solve with them the "predator-prey" system of equations

$$\begin{cases} \frac{dx}{dt} &= ax - bxy \\ \frac{dy}{dt} &= cxy - dy \end{cases} \tag{7}$$

Discuss the differences between the solutions obtained with the two integration methods as the parameters vary.

25. **The Wave Equation** Implement an explicit difference scheme for solving the wave equation $\phi_{xx} - \phi_{tt} = 0$ with the initial/boundary conditions:
$a = 1$
$g_1(t) = e^{-t}$
$g_2(t) = 2 = e^{1-t}$
$f_1(x) = 2x + e^x$
$f_2(x) = -e^x$. The exact solution is $\phi = 2x + e^{x-y}$: compare it with your numerical procedure.