Worksheet 9: Solving Differential Equations 1

June 4, 2014

General Remarks

- The deadline for the worksheets is **Wednesday, 18 June 2014, 10:00** for the tutorial on Friday and **Friday, 20 June 2014, 10:00** for the tutorials on Tuesday and Wednesday.
- On this worksheet, you can achieve a maximum of 10 points.
- To hand in your solutions, send an email to
  - Johannes (`zeman@icp.uni-stuttgart.de`; Tuesday, 9:45–11:15)
  - Tobias (`richter@icp.uni-stuttgart.de`; Wednesday, 15:45–17:15)
  - Shervin (`shervin@icp.uni-stuttgart.de`; Friday, 11:30–13:00)

Task 9.1: Solving the One-dimensional Poisson Equation (6 points)

In this task, you are to numerically approximate the solution of the one-dimensional Poisson equation

$$\frac{\mathrm{d}^2}{\mathrm{d}x^2}\phi(x) = \rho(x). \tag{1}$$

- **9.1.1** (1 point) Using finite differences, discretize the one-dimensional Poisson equation as shown for the Bessel equation in the script in section 6.1.3. Write down the equation that corresponds to equation (6.17) in the script, that allows to read off the matrix A from the system of linear equations $A\vec{\phi} = \vec{\rho}$.
- **9.1.2** (2 points) Using the discretization from the previous task, implement a Python function `solve_poisson1d_exact(rho,h)` that approximates the solution of the Poisson equation, where `rho` is a one-dimensional NumPy-Array that contains N values of the charge distribution ρ and `h` is the step size between the values of `rho`. Let $\phi = 0$ on the boundaries.

  ### Hints

  - To solve the system of linear equations, use the Python function `scipy.linalg.solve`.
  - Consequently, the solution of the system of linear equations is exact, while the solution of the differential equation itself is not.
- **9.1.3** (1 point) Using the Python function `solve_poisson1d_exact(rho,h)` from the previous task, approximate the solution of the Poisson equation for the charge distribution $\rho(x) = \sin(\frac{2\pi}{L}x)$ on the interval $[0, L]$, where $L = 10$ at $N = 10, 50$ points on the interval. Plot the numerical solutions for both values of N and the analytical solution.
- **9.1.4** (2 points) Approximate the same solution as in the previous subtask for $N = 10$ points using the successive overrelaxation method as implemented in the Python function `sor_step(A,b,omega,x)` from chapter 8.1.3 of the script. Plot the solution after 5, 10 and 20 steps of the method at $\omega = 1$.

Task 9.2: Two-dimensional Poisson Equation (4 points)

In this task the two-dimensional Poisson equation

$$\frac{\partial^2\phi(x,y)}{\partial x^2} + \frac{\partial^2\phi(x,y)}{\partial y^2} = \rho(x,y)$$

should be solved numerically for a two-dimensional charge distribution ρ.

The notebook `ws9.ipynb` and the Python script `ws9.py` generate and plot a two-dimensional sample charge distribution `rho`.

- **9.2.1** (2 points) Discretize the two-dimensional Poisson equation as explained in chapter 8 of the lecture script and write down the defining equations. Implement a Python function `solve_poisson2d(rho,h,sor_steps,omega)` that uses the SOR method to approximate the solution to the equation. The function should return the potential ϕ, where `rho` is a charge distribution, `h` is the discretization step and `sor_steps` is the number of steps of the successive overrelaxation scheme and `omega` is the SOR parameter ω.

  Hint The Python command `rho_new=rho.reshape(N*N)` transforms any $N \times N$-matrix into a one-dimensional array with a length of N^2.

- **9.2.2** (2 points) Use the function to solve the equation for the sample charge distribution `rho`. Plot the potential after 1, 10 and 50 SOR steps at $\omega = 1.8$.