Report 4: Partial Differential Equations (PDEs)

Git: https://github.com/simonblaue/MCP-Ex4.git

Simon Blaue

December 16, 2022

Universität Göttingen
Faculty of Physics
Instructor: Prof. Dr. S. Schumann
Tutors: Dr. E. Bothmann, M. Knobbe

1 Laplace Equation

1.1 Iterator methods

First lets check which method converges the fastest:

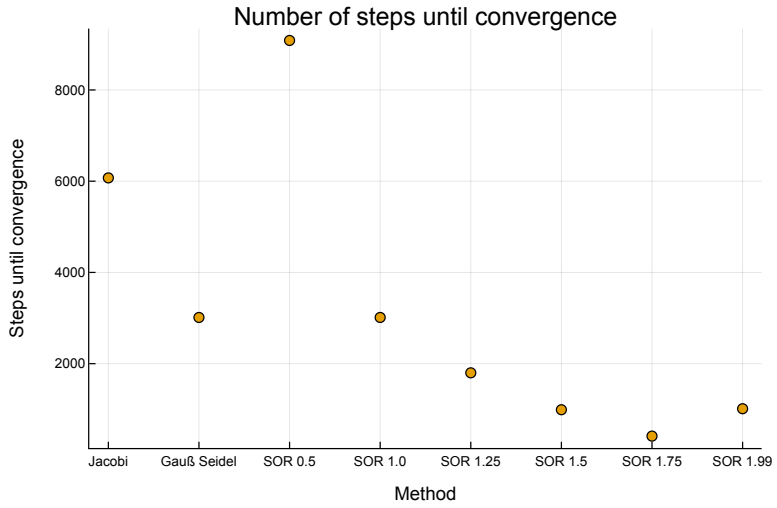


Figure 1: Number of steps until convergence concerning the Laplace error $\max \epsilon < 1 \times 10^{-3}$.

I observe that Gauß-Seidel and SOR with $\alpha = 1.0$ need the same number of time steps as expected. The fastest method is SOR with $\alpha = 1.75$, because with $\alpha = 1.99$ we get to close to unstable regimes. Obviously SOR with $\alpha = 0.5$ takes the longest, as the updating step is damped with the factor α. Now I will observe the evolution of the maximal error and the average error of the different methods.

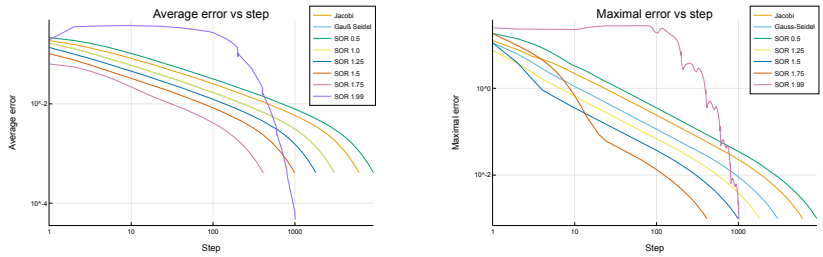


Figure 2: Maximal and average error for different iteration methods.

As discussed before, SOR with $\alpha = 1.0$ and Gauß-Seidel method are the same, hence I plotted only the latter. The error development for all methods besides SOR with$\alpha \geq 1.5$ seem qualitatively the same. However for higher α I observe indents in the curve, which seems to boost the algorithm. This is due to the algorithm taking bigger steps in the right direction once it found this direction. For $\alpha = 1.99$ the curve is very rigid, and I was not sure, if it is relay converged to the right solution, hence I plotted the result below. As it turns out the method also converges to the expected result.

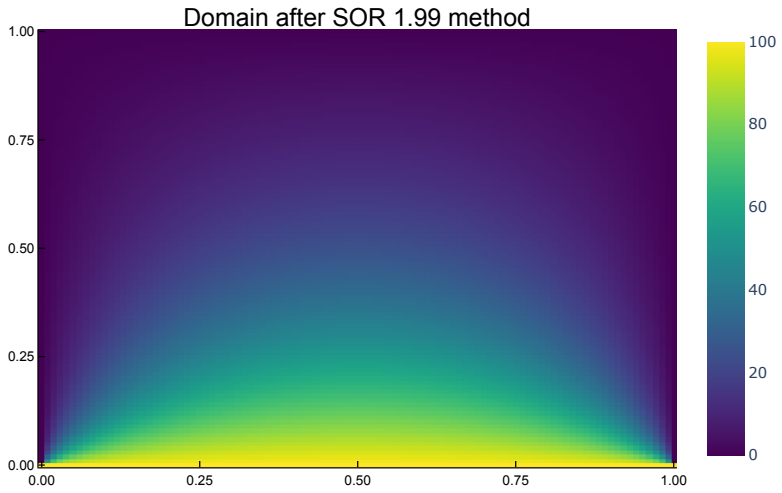
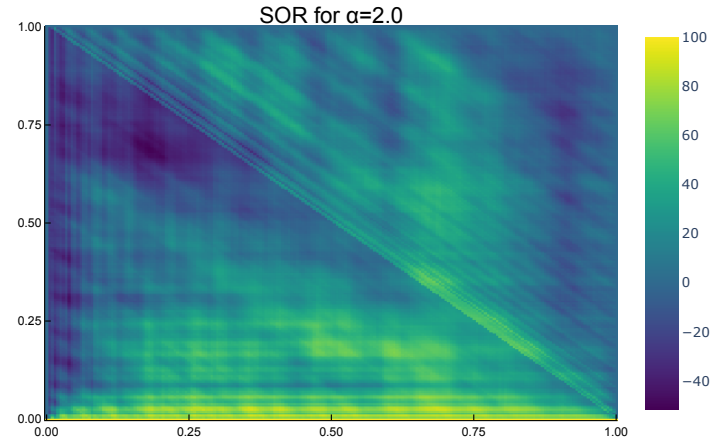
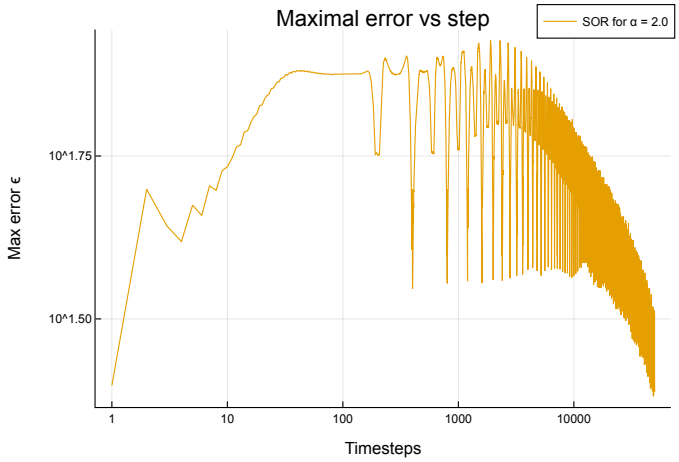


Figure 3: Domain after iterating with the SOR 1.99 method to verify right convergence.

1.2 SOR

The natural question to ask is what happens for an even further boosted SOR method? I found that for $\alpha = 2.0$ the algorithm does not converge in 50000 steps. It seems that it would not have converged to the right domain, as the result shows stripes and the maximal error starts to fluctuate a lot after 100 time steps. The system can not recover from that.





1.3 Infinite sum solution

Now I will cross verify the iterative solution with an aproximated analytical solution. First have a look at the analytical "infinite" sum solution for different numbers of summands.

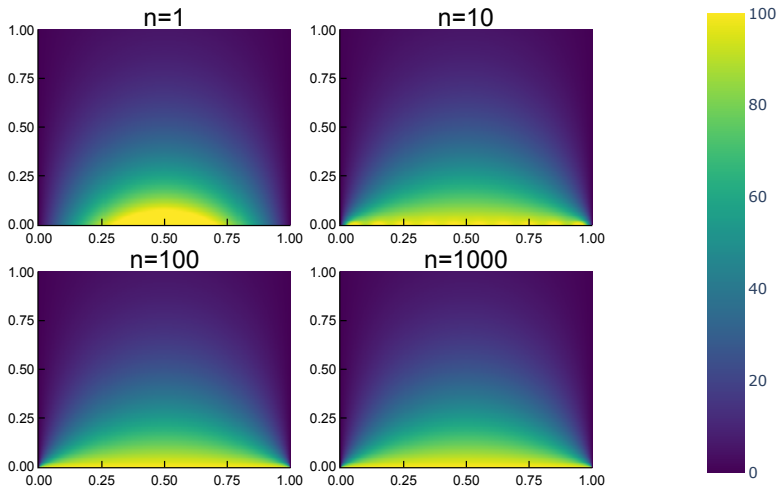
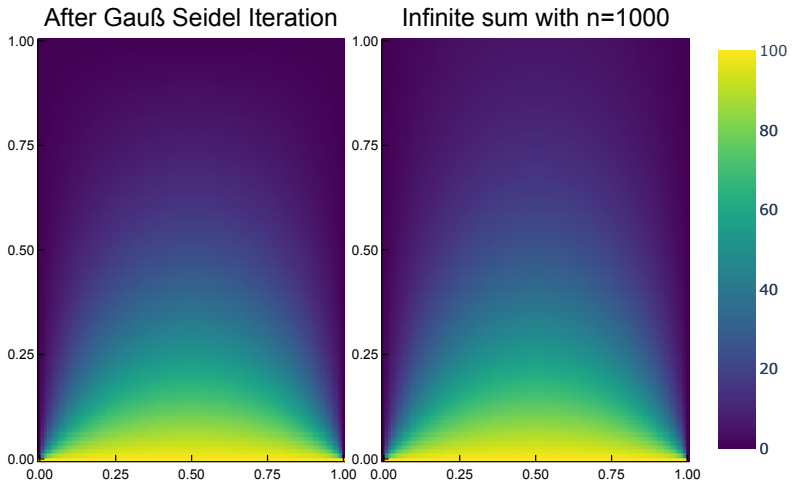
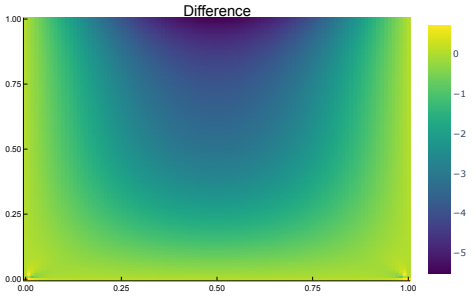


Figure 4: Results for the "infinite" sum solution to the Laplace equation for different numbers of terms n.

As I increase the number of terms more and more sine functions are added left and right to the first one, and towards the top an exponential decay fills in. Comparing the "infinite" sum solution for 1000 terms with the Gauß-Seidel method reveals that the exponential decay is to weak, or the Gauß-Seidel method did not propagate long enough. Subtracting both results displays that at the edges the solutions are identical (yellow) in the top they do not fit together as the iterative solution did not propergate that far.



(a) Comparison of the infinite sum solution with 1000 terms and the Gauß Seidel iterator solution.



(b) Difference between the infinite sum solution with 1000 terms and the Gauß Seidel iterator solution.

2 Diffusion

In this task I implemented four different method for solving the diffusion equation. To varyfiy they all converged to the same solution I plotted a time evolution of all of them. The time domain is in the y-axis, the rod domain in the x-axis, displayed in Figure 6. As expected, th rod cools down, but holds the qualitative temperature profile.

2.1 Integration methods

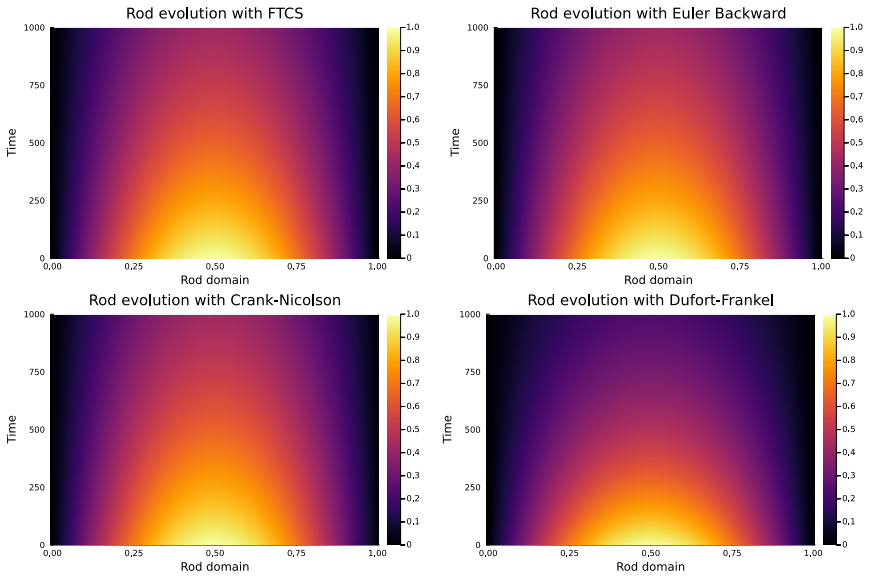


Figure 6: Temporal evolution of the rod's temperature along the y-axis. The evolution seems to be the same for all four integration methods.

2.2 Error comparison

To investigate the stability of the different method, I plotted the error against the analytical solution for different time step widths. I expect unconditionally stability for the implicit methods Euler Backwards and Crank-Nicolson and the Dufort-Frankel method. The FCTS method is not unconditionally stable and should result in a huge error at some large enough Δt, depending on Δx ($\Delta x = 0.01$). However also the other methods haev a time step dependent truncation error. I displayed their evolution individually in Figure 8. For comparing the other erros I will cut off the instability of the FCTS scheme at an error of $\epsilon = 0.007$. I observe that the other schemes seem stable in the tested regime.

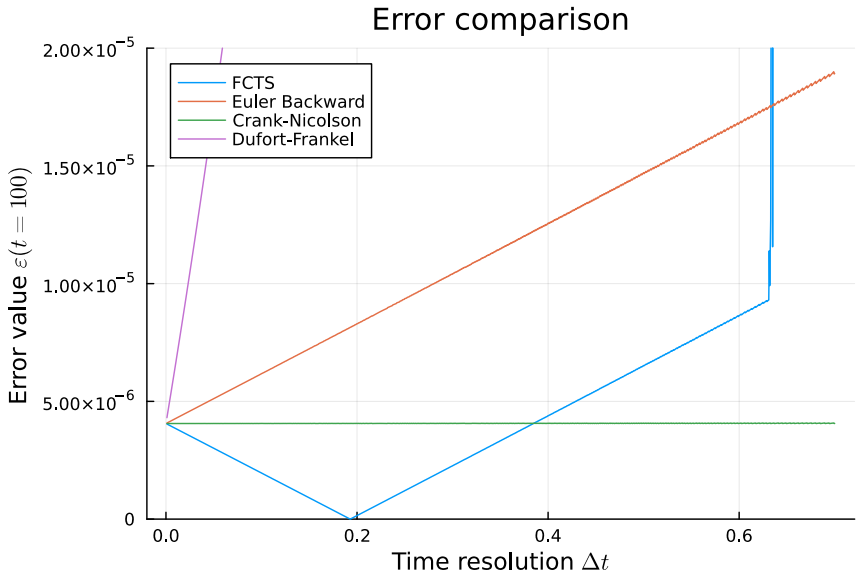


Figure 7: Error comparison for the different methods

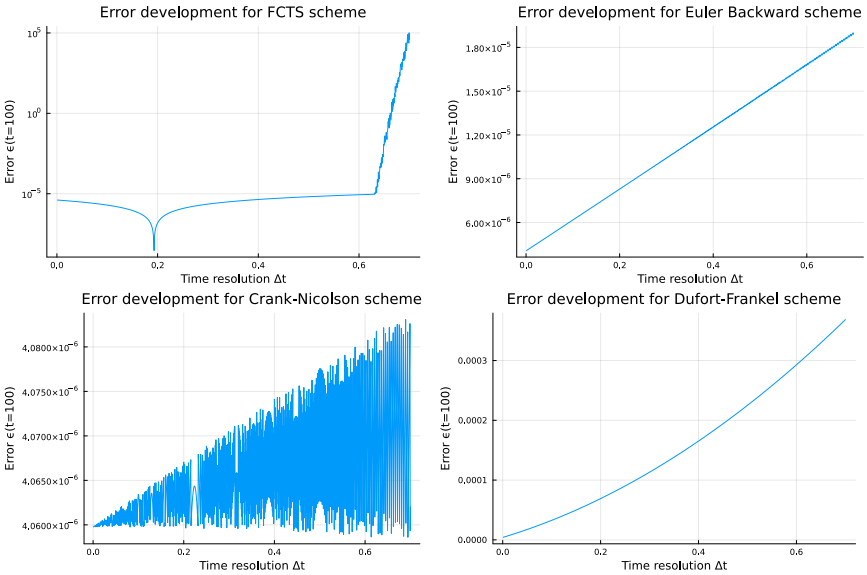


Figure 8

3 Solitons

3.1 Numerical method

We write $u(x,t) \to u_j^n(j\Delta x, n\Delta t) = U_j^n$. For the time derivative we have

$$\frac{\partial u}{\partial t} = \frac{u_{j-1}^n - u_{j+1}^n}{2\Delta t} + \mathcal{O}(\Delta t^2). \tag{1}$$

For the first space derivative we have

$$\frac{\partial u}{\partial x} = \frac{u_{j-1}^n - u_{j+1}^n}{2\Delta x} + \mathcal{O}(\Delta x^2). \tag{2}$$

To get the third order space derivative I expand $u(x,t)$ around $(x \pm \Delta x, t)$ and $(x \pm 2\Delta x, t)$. This reads as follows

$$u(x \pm \Delta x, t) = u \pm u_x \Delta x + u_{xx}\frac{\Delta x^2}{2} \pm u_{xxx}\frac{\Delta x^3}{6} + u_{xxxx}\frac{\Delta x^4}{24} + \mathcal{O}(\Delta x^5)$$
$$u(x \pm 2\Delta x, t) = u \pm u_x 2\Delta x + u_{xx}2\Delta x^2 \pm u_{xxx}\frac{4\Delta x^3}{3} + u_{xxxx}\frac{2\Delta x^4}{3} + \mathcal{O}(\Delta x^5)$$

Rewrite $u(x \pm \Delta x, t) = u_{j\pm1}^n$ and $u(x \pm 2\Delta x, t) = u_{j\pm2}^n$ and calculate

$$\begin{aligned}
&u_{j+2}^n - 2u_{j+1}^n + 2u_{j-1}^n - u_{j-2}^n \\
&= u + u_x 2\Delta x + u_{xx}2\Delta x^2 + u_{xxx}\frac{4\Delta x^3}{3} + u_{xxxx}\frac{2\Delta x^4}{3} + \mathcal{O}(\Delta x^5) \\
&\quad - 2u - 2u_x\Delta x - 2u_{xx}\frac{\Delta x^2}{2} - 2u_{xxx}\frac{\Delta x^3}{6} - 2u_{xxxx}\frac{\Delta x^4}{24} - 2\mathcal{O}(\Delta x^5) \\
&\quad + 2u - 2u_x\Delta x + 2u_{xx}\frac{\Delta x^2}{2} - 2u_{xxx}\frac{\Delta x^3}{6} + 2u_{xxxx}\frac{\Delta x^4}{24} + 2\mathcal{O}(\Delta x^5) \\
&\quad - u + u_x 2\Delta x - u_{xx}2\Delta x^2 + u_{xxx}\frac{4\Delta x^3}{3} - u_{xxxx}\frac{2\Delta x^4}{3} - \mathcal{O}(\Delta x^5) \\
&= 2u_{xxx}\frac{4}{3}\Delta x^3 - 4u_{xxx}\frac{1}{6}\Delta x^3 \\
&= u_{xxx}\left(\frac{8-2}{3}\Delta x^3\right) \\
&\Leftrightarrow \frac{\partial^3 u}{\partial x^3} = \frac{u_{j+2}^n - 2u_{j+1}^n + 2u_{j-1}^n - u_{j-2}^n}{2\Delta x^3}
\end{aligned}$$

The first term $u(x,t)$ is approximated by

$$u(x,t) = \frac{u_{j-1}^n + u_j^n + u_{j+1}^n}{3}$$

Now we can put the four terms together to finally get

$$\frac{u_j^{n-1} - u_j^{n+1}}{2\Delta t} + \epsilon \frac{u_{j-1}^n + u_j^n + u_{j+1}^n}{3} \frac{u_{j-1}^n - u_{j+1}^n}{2\Delta x} + \mu \frac{u_{j+2}^n - 2u_{j+1}^n + 2u_{j-1}^n - u_{j-2}^n}{2\Delta x^3} = 0$$

$$\begin{aligned} u_j^{n+1} = u_j^{n-1} &- \frac{\epsilon}{3} \frac{\Delta t}{\Delta x} [u_{j-1}^n + u_j^n + u_{j+1}^n][u_{j-1}^n - u_{j+1}^n] \\ &- \mu \frac{\Delta t}{\Delta x^3} [u_{j+2}^n - 2u_{j+1}^n + 2u_{j-1}^n - u_{j-2}^n] \end{aligned} \tag{3}$$