Numerical Methods in Physics Week 2

Nikolai Plambech Nielsen

1 Diffusion

For this assignment, diffusion from an initial delta-function peak is evaluated numerically. The differential equation is the diffusion equation, which in one dimension is

$$\frac{\partial C}{\partial t} = D\frac{\partial^2 C}{\partial t^2} \tag{1.1}$$

where D is the diffusion constant. This is a partial differential equation which necessitates the use of other methods. First the Forward in Time Centered in Space (FTCS) method is used. The integration scheme is

$$C_j^{n+1} = C_j^n + \frac{D\Delta t}{h^2}(C_{j-1}^n + C_{j+1}^n - 2C_j^n), \tag{1.2}$$

where h is the distance between each point in space, equivalent to Δt in time.

To numerically evaluate the delta-function, the following numerical equivalent is used

$$\delta(x - x_0) \rightarrow \begin{cases} 1/h & x = x_0 \\ 0 & x \neq 0, \end{cases} \tag{1.3}$$

as this is a sharp peak around x_0 and "integrating" this over all space gives 1:

$$\sum \delta(x - x_0)h = \sum_{x_0}^{x_0} \frac{h}{h} = 1 \tag{1.4}$$

The analytical solution to this problem is

$$C(x,t) = \frac{1}{\sigma(t)\sqrt{2\pi}} \exp\left(-\frac{(x-x_0)^2}{2\sigma(t)}\right), \quad \sigma(t) = \sqrt{2Dt}. \tag{1.5}$$

which is a delta function in the limit of $t \rightarrow 0$, but spreads out as time passes. To compare the numerical and analytical solutions, the following values are used: $D = 0.1, L = 2\pi, N = 128$, with $h = L/(N-1)$ as this is the spacing MATLAB uses with the `linspace` function, and $\Delta t = h^2/8D$. The positions after $t = 1$ are plotted, along with the difference between the two solutions
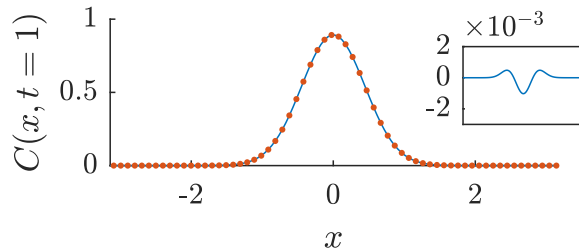

Figure 1: The numerical (blue line) and analytical (orange points) solutions to the diffusion equation for a delta function initial condition. An inset of the difference between the numerical and analytical solutions is also shown.

The analytical and numerical solution to the differential equation are in agreement, with a max difference of $4.82 \cdot 10^{-4}$. However, at the edges of the simulation, where both solutions are approximately zero, the fractional error does exceed 0.3 at places.

Next the relation between error, grid size and time step size is explored. However, comparing errors between simulations with different grid sizes becomes difficult, as this does not allow for easy plotting. Therefore the absolute value of the error on the final time step is summed up and divided by the number of points to normalize the error. First the simulation is run with 128 points, for 1 unit of time, with 100 linearly spaced values of Δt between $\Delta t_0 = h^2/2D$ and $h^2/100D$. Next the simulation is run with between 20 and 1000 points in increments of 20, with $\Delta t = h^2/4D$, where $h = L/(N-1)$, to satisfy the stability condition. The results are shown in the figures below:
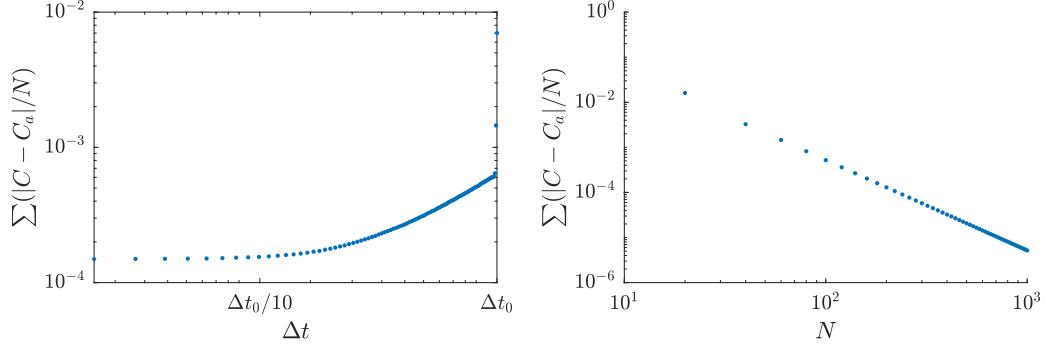

Figure 2: Plots of the normalized cumulative error as a function of the time step size Δt (left) and grid size N (right). On the right plot, the grid size is $N = 128$, and on the left plot, the time step is $\Delta t = h^2/4D$.

The dependence on Δt seems approximately linear from $\Delta t = \Delta t_0$ to $\Delta t = 8\Delta t_0/10$, before becoming approximately constant. The first part is as expected, however the second is not. I do not know, if I have run into any limit to the accuracy of the simulation, or if my definition of error has anything to do with the result.

The dependence on N is fully polynomial between in the chosen range, with a degree of roughly -2. This is in agreement with the expectation that the error in goes as $O(h^2) = O(N^{-2})$.

1.1 von Neumann stability

The von Neumann stability condition is a method of determining the stability of a numerically simulated problem. To derive this, a trial function is inserted into the integration scheme for the problem, and the ratio of amplitudes between timesteps is isolated. The trial function is that of a complex standing wave:

$$C_{\text{trial}}(x,t) = A(t)e^{ikx} \rightarrow C_j^n = A^n e^{ikjh} \tag{1.6}$$

where n is the index in time, k is the wave number, j is the index in space and h is the grid spacing. The ratio $\xi = A^{n+1}/A^n$ is then isolated in the integration scheme, and if $|\xi| \leq 1$, then the solution is stable. For this problem one gets

$$A^{n+1}e^{ikjh} = A^n e^{ikjh} + \frac{D\Delta t}{h^2}(A^n e^{ik(j-1)h} + A^n e^{ik(j+1)h} - 2A^n e^{ikjh}) \tag{1.7}$$

And upon dividing by $A^n e^{ikjh}$ this becomes

$$\xi = \frac{A^{n+1}}{A^n} = 1 + \frac{D\Delta t}{h^2}(e^{-ikh} + e^{ikh} - 2) = 1 + \frac{D\Delta t}{h^2}(2\cos kh - 2) = 1 - \frac{4D\Delta t}{h^2}\sin^2 kh/2. \tag{1.8}$$

where Eulers formula and the trigonometric identity $\sin^2(x) = (1 - \cos(2x))/2$ are used. Now, $\sin^2 kh/2$ oscillates between 0 and 1, and the constants $D, \Delta t, h$ are all real and positive. This means that $\xi \leq 1$ for all values of the constants. But it might still be less than -1 for certain values of the constants. The worst case scenario is when $\sin^2 kh/2 = 1$, where the condition becomes

$$-1 \leq 1 - \frac{4D\Delta t}{h^2} \Rightarrow 2 \geq \frac{4D\Delta t}{h^2} \Rightarrow \Delta t \leq \frac{h^2}{2D}. \tag{1.9}$$

which is the Von Neumann stability condition for this numerical integration scheme. To test this, the simulation is run with $\Delta t = (1 \pm 10^{-2})\frac{h^2}{2D}$, or a 1% deviation from the von Neumann limit in each direction:
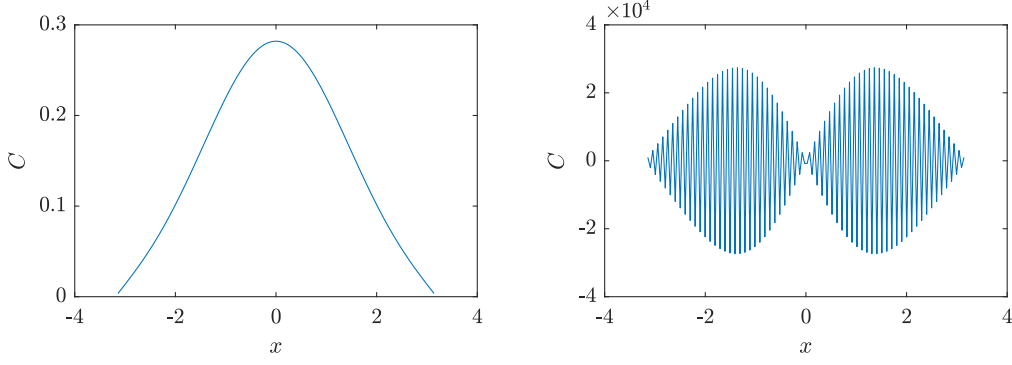
Figure 3: Plots of the numerical solutions to the simple diffusion problem, with $\Delta t \approx h^2/2D$. Both are plotted at $t = 10$. On the left $\Delta t = 0.99 \cdot (h^2/2D)$ and on the right $\Delta t = 1.01 \cdot (h^2/2D)$.

As seen, for $\Delta t < \Delta t_0$ the simulation runs as expected, but for $\Delta t > \Delta t_0$ (even by just 1%) the result blows up.

2   Diffusion with constant source and decay

In this problem, two new terms are added to the differential equation: a decay term and a constant source term:

$$\frac{\partial C}{\partial t} = S\delta(x - x_0) + D\frac{\partial^2 C}{\partial x^2} - \frac{C}{t} \tag{2.1}$$

where the analytical solution at steady state is

$$\frac{C(x)}{C(x_0)} = e^{-\frac{|x-x_0|}{\sqrt{\tau D}}}. \tag{2.2}$$

For this exercise both FTCS and a new method, called the Spectral Method, will be used. The Spectral Method uses discrete Fourier Transformation to convert a PDE to an ODE, allowing the use of simple Euler integration, and then just inverse Fourier transformation at the desired timestep. Now, due to the way MATLAB handles its discrete Fourier Transforms, only even values of N can be used. Because of this, the source point $x_0 = x_{N/2}$ is chosen, which has the value $x_0 = -h/2$.

2.1   von Neumann stability

The new integration scheme is

$$C_j^{n+1} = C_j^n + \Delta t\left(S\delta(x - x_0) + \frac{D}{h^2}(C_{j-1}^n + C_{j+1}^n - 2C_j^n) - \frac{C_j^n}{\tau}\right), \tag{2.3}$$

where the delta function term is, in essence, an initial condition. For this reason, it is discarded in deriving the stability condition. This then becomes:

$$A^{n+1}e^{ikjh} = A^n e^{ikjh} + \Delta t A^n\left(\frac{D}{h^2}(e^{ik(j-1)h} + e^{ik(j+1)h} - 2e^{ikjh}) - A^n\frac{e^{ikjh}}{\tau}\right). \tag{2.4}$$

and isolating ξ gives

$$\xi = \frac{A^{n+1}}{A^n} = 1 - \frac{4D\Delta t}{h^2}\sin^2(hk/2) - \frac{\Delta t}{\tau} \tag{2.5}$$

where again Eulers formula and the trigonometric identity has been used. This is again always smaller than 1, but the worst case scenario of $\sin^2(hk/2) = 1$ gives

$$2 \geq \Delta t\left(\frac{4D}{h^2} + \frac{1}{\tau}\right) \Rightarrow \Delta t \leq \frac{2\tau h^2}{4D\tau + h^2}. \tag{2.6}$$

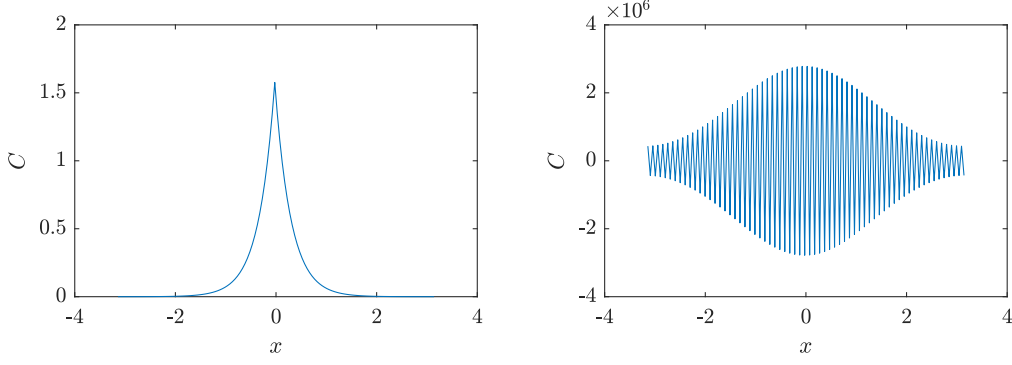and with the same conditions as before ($\Delta t = (1 \pm 10^-2)\Delta t_0$, end at $t = 10$), the results are

3

Figure 4: Plots of the numerical solutions to the simple diffusion problem, with $\Delta t \approx \Delta t_0$. Both are plotted at $t = 10$. On the left $\Delta t = 0.99\Delta t_0$ and on the right $\Delta t = 1.01\Delta t_0$.

And just as before, the results are as expected.

2.2   Comparing the analytical solution and FTCS method

The analytical and numerical solution is compared as before, with $\Delta t = \Delta t_0/2$ and $N = 128$. The results are plotted for $t = 1$:
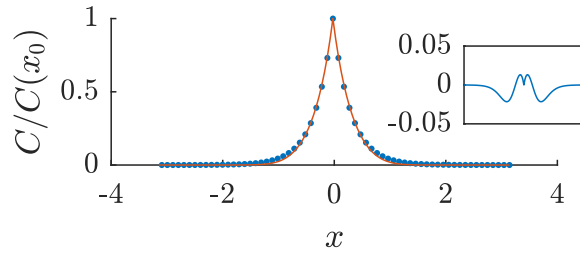

Figure 5: The numerical (blue line) and analytical (orange points) solutions to the diffusion equation with a constant source and decay. An inset of the difference between the numerical and analytical solutions is also shown.

For this problem the maximal fractional error is 2.16%, even at the tails, which is much better than the almost 50% for the simple diffusion problem.

2.3   The Spectral Method

2.3.1   An example

The spectral method utilizes the fact that Fourier transforming a differential equation converts it to an algebraic equation in the inverse space. Namely, the n'th derivative of an (at least) n times differentiable function (where it is assumed that the function and all derivatives vanish at infinity) becomes $(ik)^n$ times the function. So:

$$\frac{\partial u(x)}{\partial x} \to iku(k), \quad \frac{\partial^2 u(x)}{\partial x^2} = -k^2 u(k). \tag{2.7}$$

As an example, the sum of the first and second derivative of $\cos(2x)$ is calculated using this method. Analytically this is $-4\cos(2x) - 2\sin(2x)$. The function is plotted in the interval $x \in [-\pi, \pi]$, with $N = 128$. The numerical and analytical solutions are plotted below:
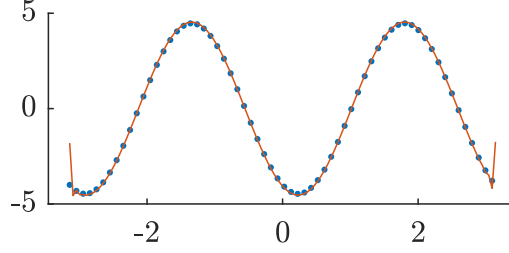
Figure 6: The numerically calculated sum of the first and second derivative of $\cos(2x)$ (orange) and the analytical (blue points). Solved using the spectral method for $N = 128$.

The numerical and analytical solutions are in agreement, except at the edges. This is due to the value of the function not going to zero at the edges, which means that the Fourier transformed function in reality is the desired function modulated by a square function, which changes the Fourier transform.

2.3.2 Solving the simple diffusion problem

To solve the problem using the spectral method, the whole equation has to be Fourier transformed into k-space. The right hand side just becomes $-Dk^2C(k,t)$, but Fourier transforming the initial condition, a delta-function, results in a complex wave e^{-ikx_0}, and for some reason (maybe because the "wave number" is $x_0 = -h/2$), this causes the spectral method to explode for any Δt (even $\Delta t = 5 \cdot 10^{-4} \cdot \Delta t_0$). To combat this, a different initial condition is used: a not too sharply peaked Gaussian function. This is an appropriate approximation, as the delta function can be defined as the limit of the normalized Gaussian for $\sigma \to 0$. Here $\sigma > h$ for there to be any difference between the Gaussian and a delta function. For the problem $\sigma = 1.5h$ is used.

This will of course give a larger error when comparing the numerical and analytical solutions, as they no longer solve the same problem. Comparing the two solutions is done as in the first assignment, with the two plotted together and their difference plotted in an inset, for the same $\Delta t, N$ and ending time ($t = 1$):
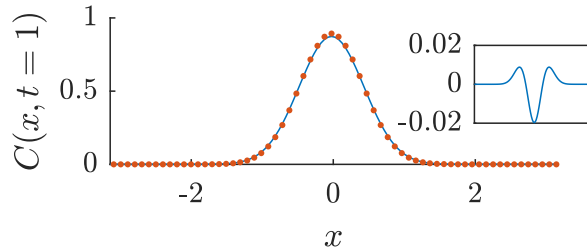

Figure 7: The numerical (blue line) and analytical (orange points) solutions to the diffusion equation for a delta function initial condition. An inset of the difference between the numerical and analytical solutions is also shown.

where the largest absolute difference is 0.0197, and the largest fractional difference is 436. This means that any gains in using the spectral method are severely limited due to the change in initial condition. For a more fair comparison, the FTCS method is also run with this new initial condition:
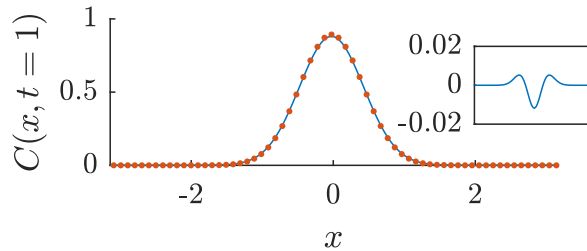

Figure 8: The simple diffusion problem in FTCS, but with a Gaussian-function initial condition

where the largest absolute difference here is 0.0118 and the largest fractional difference is 1.25. So again, FTCS wins for the simple diffusion problem.

2.3.3 Diffusion with constant source using the spectral method

For the problem with constant source and decay, the decay-term is unaffected by the transformation, and the constant source is changed from a delta function to the Gaussian function used in before.

The error, as a function of N is calculated as in the previous problem (normalized, cummulative absolute error in space), and to provide a proper comparison between FTCS and the spectral method, the calculations are also performed using FTCS:
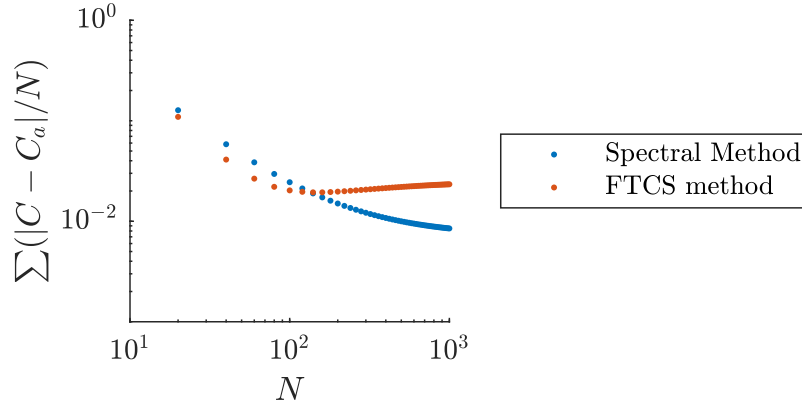

Figure 9: Plots of the normalized cumulative error as a function of grid resolution N for the spectral (blue) and FTCS (orange) methods.

As seen in the figure, FTCS beats the spectral method until around $N = 10^2$, at which FTCS actually starts to increase! This is most likely due to the change from a delta function to a Gaussian.

2.3.4 Performance of Euler and aRK4 methods for the spectral method

Lastly the performance of the adaptive, 4th order Runge-Kutta is compared to that of the Euler method. This is done by running the simulation for each of the methods 1000 times while using MATLABs profiler tool, to measure the time spent on each of these methods. The high number of repetitions is chosen to make sure that the accuracy of the time measurement is good enough for comparison.

A baseline $\Delta t = \Delta t_0/2$ is chosen, along with $N = 128$ and a stop time of $t = 1$. In this configuration the Euler method took 1.11 seconds, performing 164000 calls to the "spectral" function, which calculates the derivatives. This corresponds to 165 datapoints in time per run (the initial, plus 164 iterations). The adaptive Runge-Kutta took 30.50 seconds, on the other hand, with 120 iterations per run and 1668000 calls to the function "spectral". This is a 27-fold increase in runtime and a 10-fold increase in the number of calculations. The discrepancy between these two numbers is due to the extra calculations performed for the adaptive Runge-Kutta.

As for the actual accuracy. The normalized cumulative error for the Euler method is 0.6359, while it is 0.0201 for the adaptive Runge-Kutta method, which is a 31-fold increase in accuracy. This is also evident, if one looks at the value of Δt for the adaptive Runge-Kutta:
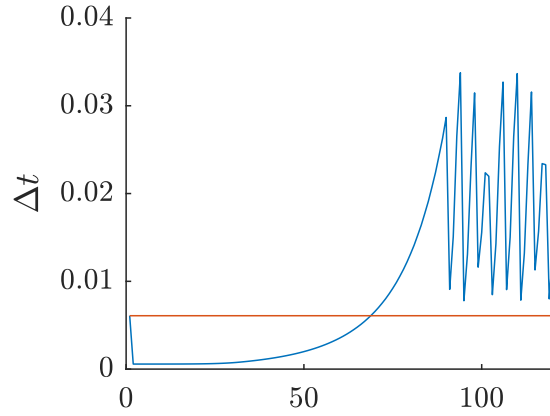
Figure 10: Δt as a function of index for the adaptive Runge-Kutta (blue) method. The baseline Δt is inlaid as the horizontal orange line.

As seen, for the first half of the simulation, the time step is significantly lower than the baseline. This is due to the high derivative of the function (in time). When the Euler method is run with this time step size, a rather large error accumulates, which the adaptive Runge-Kutta solves. Worth noting is that the point at which the steady state is reached is clearly evident from the graph, as this is where the step size oscillates.