VU Scientific Computing

Gruppe:

|  |  |
| --- | --- |
| Name | Matrikelnummer |
| David Gstir | 12112059 |
| Thomas Schwarz | 12035825 |

In this file there can be found all the plots and answers to questions for all the exercises. All the used parameters for the simulations and plots can be seen in the individual python files.

*1.5*

Ein Bild, das Diagramm, Screenshot, Design enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Diagramm, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

The temperature always gets closer to the equilibrium. The speed of temperature change also decreases while getting closer to the equilibrium. If we violate the CFL condition, the plot looks like this:

Ein Bild, das Screenshot, Rechteck, Design enthält.

Automatisch generierte Beschreibung

This happens because CFL imposes a limit on the size of the time step relative to the spatial discretization in order to maintain stability. When we violate this condition, it can lead to numerical instability.

*1.6*

The decay rate depends on the initial temperature profiles because it is different for perhaps constant, linear, and also for oscillations in the initial temperature distribution. For example, a constant initial temperature profile does typically not have a decay rate because there are no spatial temperature differences. On the other side on linear initial temperature profiles, the decay rate depends on the slope of the temperature gradient.

*1.7*

Implemented the same problem as in 1.6 but this time we used homogeneous Neumann boundary conditions on the right end of the rod. As we can see the right end of the rod gets colder and colder unlike in the previous exercise where the end of the rod temperature was constant. This makes sense, because we defined the boundary condition like that.

Ein Bild, das Diagramm, Reihe, Screenshot, Design enthält.

Automatisch generierte Beschreibung

Ein Bild, das Text, Screenshot, Diagramm, Reihe enthält.

Automatisch generierte Beschreibung

*1.9*

Ein Bild, das Diagramm, Screenshot, Reihe, Design enthält.

Automatisch generierte BeschreibungEin Bild, das Diagramm, Reihe, Text, Screenshot enthält.

Automatisch generierte Beschreibung

For the implicit Euler scheme, we used a much larger time step with the size of 0.01 in comparison to the time step 0.000025 with the explicit Euler scheme. Although the time step is so much larger with the implicit Euler scheme, we still get stable numerical solutions with it.

*1.10*

Ein Bild, das Diagramm, Reihe, Screenshot, Design enthält.

Automatisch generierte BeschreibungEin Bild, das Text, Diagramm, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

*1.11*

Ein Bild, das Diagramm, Reihe, Screenshot, Design enthält.

Automatisch generierte BeschreibungEin Bild, das Text, Diagramm, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

If we compare these two implementations with the explicit Euler scheme, we can see the following runtimes:

*1.12/1.13*

We plotted the results at three different times. First time directly at the beginning, second time after half the time steps and the third time at the end.

At the beginning:

Ein Bild, das Diagramm, Design, Würfel enthält.

Automatisch generierte BeschreibungHalf of the time steps:  
Ein Bild, das Diagramm, Design, Würfel enthält.

Automatisch generierte Beschreibung

At the end:

Ein Bild, das Diagramm, Design, Würfel enthält.

Automatisch generierte Beschreibung

We can see in the upper plots, that the equilibrium is nearly a plane surface through the given boundary points.

*1.14*

We added here another plot, a contour 2D plot after half the time steps, to the 3D plots at the beginning, after half the time steps and at the end.

At the beginning: Ein Bild, das Screenshot, Diagramm, Design enthält.

Automatisch generierte Beschreibung

Half of the time steps: Ein Bild, das Screenshot, Diagramm, Design enthält.

Automatisch generierte Beschreibung

At the end:

Ein Bild, das Screenshot, Diagramm, Design enthält.

Automatisch generierte Beschreibung

Contour plot after half of the time steps:

Ein Bild, das Text, Screenshot, Display enthält.

Automatisch generierte Beschreibung

Performance comparison of the ADI scheme and the explicit Euler scheme for the two-dimensional heat equation:

*2.1*

First example mesh:

Ein Bild, das Reihe, Symmetrie, Rechteck, Muster enthält.

Automatisch generierte Beschreibung

Second example mesh:

Ein Bild, das Reihe, Origami, Symmetrie, Dreieck enthält.

Automatisch generierte Beschreibung

*2.2*

We used different colors for the outer and the inner boundary as well as for the mesh itself.

Ein Bild, das Muster, Kunst, Kreis, Screenshot enthält.

Automatisch generierte Beschreibung

*2.3*

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

*2.4*

Visualization at the beginning:

Ein Bild, das Screenshot, Grafiken, Design enthält.

Automatisch generierte Beschreibung

Visualization after some iterations:

Ein Bild, das Screenshot, Mond enthält.

Automatisch generierte Beschreibung

Visualization at the end:

Ein Bild, das Screenshot, Mond enthält.

Automatisch generierte Beschreibung

When the python file for the simulation is executed, the implementation gets also checked by using the method of manufactured solutions. Therefore, the initial error and the final error is printed to the command line.

*2.5*

Used heat sink model:

Ein Bild, das Reihe, Rechteck, Diagramm, parallel enthält.

Automatisch generierte Beschreibung

*2.6/2.7*

Visualization at the beginning:

Ein Bild, das Screenshot, Reihe, Design enthält.

Automatisch generierte Beschreibung

Visualization after some iterations:

Ein Bild, das Screenshot enthält.

Automatisch generierte Beschreibung

Visualization at the end:

Ein Bild, das Screenshot, Design enthält.

Automatisch generierte Beschreibung

*3.2/3.3*

We implemented the explicit Euler centered difference scheme for the heat equation from exercise 1.5 in C++ and compared the performance of both implementations. The runtimes for the python and C++ implementation are:

|  |  |
| --- | --- |
| Python | C++ |
|  |  |

We directly implemented the 3.2 in the way descripted in exercise 3.3, so 3.2 and 3.3 has the same implementation with no changes because of the already implemented periodical write to a text file. To plot the written data, there are two python implementations, one for plotting the data 2D and one for plotting it in 3D, which must be executed separately after the execution of the C++ code finished.

2D plot:

Ein Bild, das Text, Diagramm, Screenshot, Reihe enthält.

Automatisch generierte Beschreibung

3D plot:

Ein Bild, das Diagramm, Reihe, Screenshot, Design enthält.

Automatisch generierte Beschreibung

*3.4*

In this exercise we implemented the memory bound copy benchmark and the compute bound quadrature benchmark. We executed the benchmarks on the slightly different systems.

System 1:

MacBook Pro with M2 Pro chip with 16 GB RAM and 10 cores.

First the runtimes without optimization flag:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Serial | 1 Thread | 2 Threads | 4 Threads | 8 Threads | 10 Threads |
| Memory bound | 2934 ms | 2913 ms | 1515 ms | 885 ms | 835 ms | 785 ms |
| Compute bound | 6494017 ms | 6461 ms | 3275 ms | 1734 ms | 1060 ms | 985 ms |

Runtimes with optimization flag -O3:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Serial | 1 Thread | 2 Threads | 4 Threads | 8 Threads | 10 Threads |
| Memory bound | 993 ms | 978 ms | 846 ms | 817 ms | 763 ms | 779 ms |
| Compute bound | 0 ms | 4850 ms | 2475 ms | 1305 ms | 996 ms | 760 ms |

System 2:

MacBook Pro with M2 Pro chip with 16 GB RAM and 10 cores.

First the runtimes without optimization flag:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Serial | 1 Thread | 2 Threads | 4 Threads | 8 Threads | 10 Threads |
| Memory bound | 2934 ms | 2913 ms | 1515 ms | 885 ms | 835 ms | 785 ms |
| Compute bound | 6494017 ms | 6461 ms | 3275 ms | 1734 ms | 1060 ms | 985 ms |

Runtimes with optimization flag -O3:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | Serial | 1 Thread | 2 Threads | 4 Threads | 8 Threads | 10 Threads |
| Memory bound | 993 ms | 978 ms | 846 ms | 817 ms | 763 ms | 779 ms |
| Compute bound | 0 ms | 4850 ms | 2475 ms | 1305 ms | 996 ms | 760 ms |

*3.5*