Master project: Investigate efficient time-integration methods for storm-surge models on CPU and GPU using Julia

§1 Project background

Coastal flooding by storm surges is one of the main risks for coastal cities worldwide. Because of the population growth, sea level rise and land subsidence, these risks are expected to increase in the coming years. Deltares has developed the Global Storm Surge Forecasting and Information System (GLOSSIS), which provides real-time water level and storm-surge forecasts with global coverage.

The basis of the GLOSSIS is a Global Tide and Surge Model (GTSM), also developed by Deltares. This model uses Delft3D Flexible Mesh (Delft3D FM), which is an open source numerical simulation software and facilitates the applications in hydrodynamics, wave, real time control and etc. In the D-Flow FM module of this software, the shallow water equations (SWE) are numerically solved [1].

Early flood warning for coastal cities by the GLOSSIS is important for extreme flooding situations. One essential contribution for the early warning is to speed up the solving procedure of the SWE in Delft3D FM, while at the same time keep high accuracy. To achieve this, new numerical methods on modern hardware such as graphics processing units (GPUs) become current interests.

Over the past decades, many numerical methods were developed to numerically solve partial differential equations, including SWE. Traditionally, the focus was to minimize the number of floating point operations while at the same time attaining a specified accuracy. However, recent CPUs and GPUs are often more limited by memory bandwith and communication between cores than by the computations. On GPUs we even see that simple explicit time-integration methods outperform implicit solvers, mostly due to the reduced memory access/communication. More investigation is required on the different time-integration methods on both CPU and GPU, to find an efficient method for SWE.

Julia, as a modern programming language, has drawn a lot of attentions in both industries and academia, due to its high performance in scientific computing. Julia has been developed dramatically in recent years, resulting in different packages. Therefore, for this project, Julia will be used for the exploration of the numerical methods on GPUs.

§2 Project description

In this study, we would solve a 2D shallow water model, applied in storm-surges, with finite difference discretization on a structured grid. We aim to:

1. Compare existing time-integration schemes,

2. Based on the comparison, we'll look for new methods that are simple and efficient on modern hardware. One possible approach is the use of pseudo-time-stepping in combination with a multi-grid like approach (see `https://pde-on-gpu.vaw.ethz.ch/lecture3/#pseudo-transient_method`).

More details are here:

- The performance will be measured both on CPU and GPU.
- We will use Julia, with two Julia packages: DifferentialEquations.jl and ParallelStencil.jl.
- We hope to compare the performance on different GPUs, including DelftBlue (TU Delft), Snellius (SURF) and Deltares new GPUs.

§3 Useful documents for this project

Below is a list of useful documents, including online courses for this project. It might be enriched during the project.

- MIT course on introduction to Julia, including the installation [2].
- Book about SciML, including writing fast program in Julia [3].
- ETH course on solving PDEs on GPU with Julia, including the pseudo time stepping method [4].
- Julia package ParallelStencil.jl to write high-level code for parallel high-performance stencil computation on CPUs and GPUs. [5].
- Julia package DifferentialEquations.jl for time integration methods [6].

§4 People for this project

People who will carry out this project are:

- Student: Mieke Daemen.
- Responsible supervisor: Prof. dr. ir. Martin Verlaan, TU Delft (Group of Mathematical physics) and Deltares.
- Daily supervisor (internal): Dr. Jing Zhao, Deltares and TU Delft (Group of Numerical Analysis)

§5 Initial time plan of this project

Our initial time plan for this project is as follows (it might change depending on the situation later):

- On 29 Feb. 2024, start.
- 29 Feb. 2024 - 20 June 2024, literature study.
- 21 June 2024 - 09 Jan. 2025, implementation and writing thesis.

References

[1] D-Flow Flexible Mesh, Technical Reference Manual (2023), Deltares, Delft, Netherlands. See `https://content.oss.deltares.nl/delft3dfm2d3d/D-Flow_FM_Technical_Reference_Manual.pdf`

[2] Tutorials and information on the Julia language for MIT numerical-computation courses. See `https://github.com/mitmath/julia-mit`.

[3] Parallel Computing and Scientific Machine Learning (SciML): Methods and Applications (2023). C. Rackauckas. See `https://book.sciml.ai/`.

[4] ETH's course: Solving PDEs in parallel on GPUs with Julia (2023). L.Räss, M. Werder, S. Omlin, I. Utkin. See `https://pde-on-gpu.vaw.ethz.ch/#solving_pdes_in_parallel_on_gpus_with_julia`.

[5] Julia package ParallelStencil.jl (2023). See `https://github.com/omlins/ParallelStencil.jl`.

[6] Julia package DifferentialEquations.jl (2023). See `https://github.com/SciML/DifferentialEquations.jl`.