Solver Implementations for Pressure Poisson Equations

# Problem Statement

One of the most computationally intensive part of fluidchen is solving the **pressure Poisson equation.** Until now we used SOR method to solve the equation. As a part of this project, we are exploring other solvers of 3 types: stationary methods: Jacobi, weighted Jacobi, Gauss Seidel; Kryolov Methods: Conjugate Gradient Method, Multigrid Methods: MultigridVCycle. The code is able to handle the Lid-Driven Cavity flow problem as of now for MultigridVcycle and boundaries with other solvers. We tried implementing boundary conditions for Multigrid methods on different levels but faced many hurdles on the way and time wasn’t sufficient to handle all the issues.

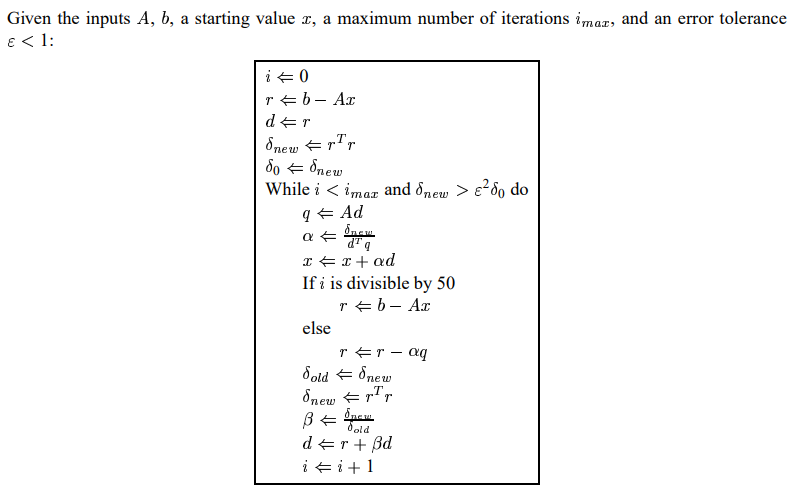
# Methods implemented and defined

## Jacobi Method:

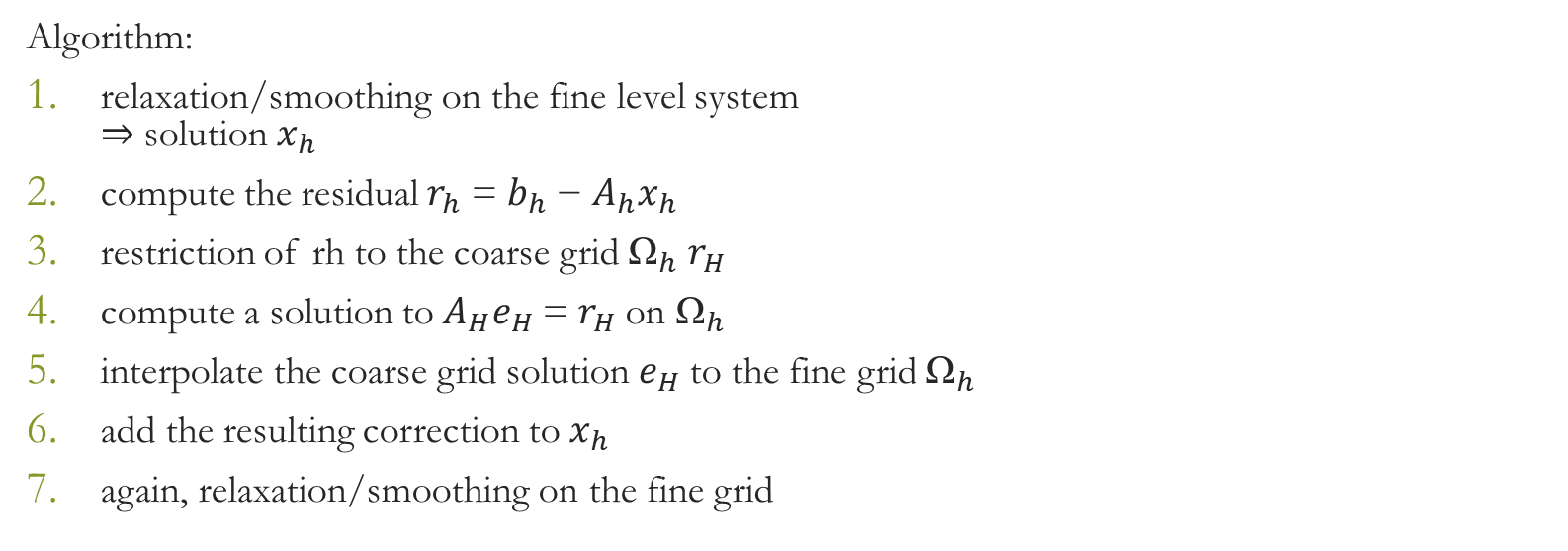
## Weighted Jacobi Method:

## Gauss Seidel Method:

## Conjugate Gradient Method:



## Multigrid V Cycle:



We chose H = 2\*h for our implementation

# Problems and Limitations:

1. Significant difference in results can be noted only on problems with complexities.
2. As of now, multigrid methods are only checked in serial running and we expect some weird behaviour when run in parallel due to the possible change in the size of the problem for each domain.
3. Issues with Boundary conditions are sorted until the compilation issues. There are some run-time bugs which give wrong solution to other problems apart from Lid-Driven Cavity.

# Nice implementation:

1. Implemented the solvers with good OOP paradigms. For example, a further implementation of a gradient method eg., steepest descent can just derive from the GradientMethod class or MultigridWCycle would just derive from MultiGrid methods

# Results and Analysis:

## Convergence Analysis

To observe convergence for the complex case, we checked the number of iterations it takes for Lid-Driven Cavity for the first time-step for different grid sizes for different solvers.

We noticed that even though MultiGrid comes with an extra cost, it is super-fast in convergence(mathematically) even with 3 levels.

## Evolution of Residuals with time

We then analysed the evolution of residuals for different methods(by letting it run for some number of iterations, varying for solver to optimise the run-time) with time to check the effect of the solvers in long simulations.

# Conclusions:

1. Against expectations, all solvers produce similar residuals with time but that may be because of the stability of the problem. Multi Grid methods are expected to produce better convergence than stationary methods for more complex problems e.g., channel flow with obstacle.
2. We observe that the residuals in Conjugate Gradient methods aren’t smooth at all as we don’t have a smoother influence on it.
3. Even though for this analysis, we ran lesser number of iterations for Multigrid methods when compared to the other stationary methods, MultiGrid still compete in terms of mathematical iterations even though they may take some more extra run-time in comparison to the other methods mentioned here.

## [Merge Request](https://gitlab.lrz.de/00000000014ADC3F/group-svk/-/merge_requests/8)