Exercises in Parallel Programming in Fortran and C/C++

Dr. Sara Collins SoSe 2022

Project

Please consider the following when completing the project.

- You may write your program in either Fortran, C or C++.
- What you submit should enable all parts of the project to be produced and illustrated without requiring modification to your code. Instructions should be provided on how to run the code and visualise the output.
- The program should be able to run on the CIP-pool machines (where the program will be tested when it is graded) and any output should be able to be visualised on these machines.
- The project must be *entirely your own work* (for example, using another person's code, even with some modification is not acceptable).
- Some marks will be given for good programming practice and well organised code.
- If the program consists of more than one file a makefile should be provided and a script should be procided which runs the code.

Solving Poisson's equation

Implement the following 2-dimensional problem as a parallel program using MPI:

- First implement a serial version of the program (without parallelisation) that has the correct functionality which will serve as a reference for the parallel program.
- The 2-dimensional grid should be parallelised in both dimensions. The size of the grid should be input to the program and the distribution of the grid to the processes should be performed automatically.
- Modify your parallel program to include OpenMP directives (= hybrid parallelisation: MPI+OpenMP). Hybrid programs can be compiled and run using (for the example of a Fortran program where the number of threads is not fixed in the code)

```
$ mpif90 -fopenmp poisson.f90
$ export OMP_NUM_THREADS=4
$ mpirun -np 10 ./a.out
```

- The output of the program should be a single binary file containing the values of the scalar function $\phi(x, y)$. Use MPI-IO to achieve this. The binary file can be read, for example, in gnuplot using

  ```
  splot 'poisson.bin' binary array=(50,50)
  ```

  for a file where $\phi(x, y)$ is discretized on a 50×50 grid.

Given the following partial differential equation (Poisson's equation)

$$\Delta\phi(\vec{r}) = f(\vec{r}) \qquad \vec{r} \in \Omega, \quad \Omega \subset \mathbb{R}^2$$

where $\vec{r} = (x, y)$,

$$\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2},$$

with the source function $f(x, y)$ which is zero everywhere ($0 \leq x, y < L$) apart from:

$$f(x,y) = \begin{cases} 10 & 15 < rL < 20 \\ -10 & 6 < rL < 10 \\ 10 & rL < 2 \end{cases} \tag{1}$$

where $r = \sqrt{(x/L - 0.5)^2 + (y/L - 0.5)^2}$. Use the Jacobi algorithm to find the function ϕ which satisfies Poisson's equation with the constraint of Dirichlet boundary conditions

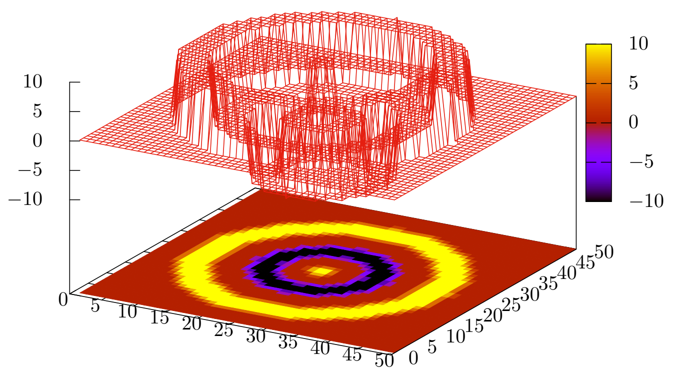
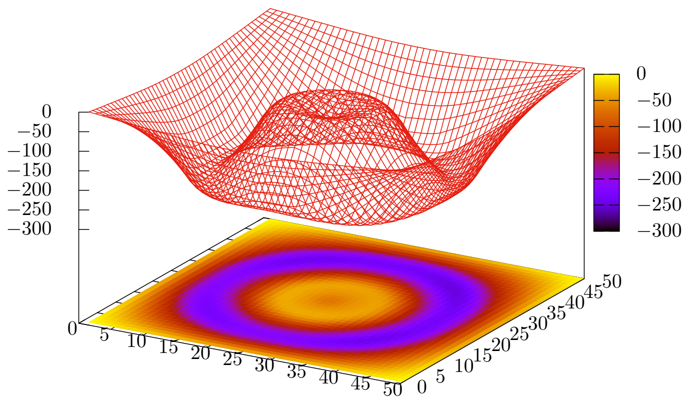
$$\phi(\vec{r}) = 0 \qquad \vec{r} \in \partial\Omega.$$

For the Jacobi algorithm, the following stopping criterium should be used:

$$\left(\sum_{x,y} \left[\phi^{k+1}(x,y) - \phi^k(x,y)\right]^2 \right)^{1/2} < \epsilon$$

where $\phi^k(x, y)$ is the field after k iterations of the algorithm and ϵ should be an input to the program.

Below on the right is a plot of ϕ for a 50×50 grid, where ϕ is generated with a residual $\epsilon = 10^{-6}$. On the left, the source function is shown.


10
5
0
−5
−10
0
5
10
15
20
25
30
35
40
45
50
0
5
10
15
20
25
30
35
40
45
50
10
5
0
−5
−10

0
−50
−100
−150
−200
−250
−300
0
5
10
15
20
25
30
35
40
45
50
0
5
10
15
20
25
30
35
40
45
50
0
−50
−100
−150
−200
−250
−300