Parallel Computing

2022/2023 1st semester

Exercise Sheet #7 2022-11-23

Create a parallel processing code that implements the Gaussian Elimination Method

1. without pivoting;
2. with partial pivoting per column,
   (exchanging two lines so that the highest numerical value of the column appears in the diagonal as divisor $a_{mm}^{(m)}$, as on pages 9 and 10 in the presentation).

Start by creating a single-core program with the examples below to test the algorithm for both cases, and then parallelize the code.

For the parallelization, the suggestion is that for each step m of the elimination, the master distributes the lines over the processes. Try to minimize the amount of data that is exchanged through the messages after each step (do not send the whole matrix to each process, just the necessary parts).

Always verify the solution in the end by multiplying A with the resulting vector x and comparing it to b.

In order to create big matrices, use arrays of random numbers and save them to a file, so that the numbers don't always change when you debug your code.

To save resp. restore a numpy array A, you can use

```
np.save ( 'data10.npy', A )

A = np.load ( 'data10.npy' )
```

Run the parallel code with a system of at least 16 linear equations in 4 parallel processes. You can assume that the number of equations is dividible by the number of processes.

In the case that it turns out that not all equations are linearly independent, make sure that all processes terminate in a clean way and have the master process show an error message.

Simple examples for trying the code:

$$\begin{aligned} 2x_1 &+2x_2 &+x_3 &+4x_4 = &5 \\ x_1 &-3x_2 &+2x_3 &+3x_4 = &2 \\ -x_1 &+x_2 &-x_3 &-x_4 = &-1 \\ x_1 &-x_2 &+x_3 &+2x_4 = &2 \end{aligned}$$

The one below will fail without pivoting:

$$\begin{aligned} -\sqrt{2}x_1 &+2x_2 &= 1 \\ x_1 &-\sqrt{2}x_2 &+x_3 = 1 \\ &2x_2 &-\sqrt{2}x_3 = 1 \end{aligned}$$

Submit the codes of your programs (1 to 2) at `https://trixi.coimbra.lip.pt/cap` until the beginning of the next lectures.