HPC - Week 4 Assignment

Peter Archibald Whalley

February 24, 2021

Workshop 5 - Exercise 2

We consider the following 2D wave equation:

$$u_{tt} = u_{xx} + u_{yy} \text{ for } x, y \in (-1, 1), t \in (0, T)$$

subject to the initial conditions

$$u(x, y, t = 0) = e^{-40((x-0.4)^2+y^2)}$$
$$u_t(x, y, t = 0) = 0$$

and homogeneous Dirichlet boundary conditions

$$u(x = \pm 1, y, t) = u(x, y = \pm 1, t) = 0$$

Using central differencing for space and leap-frog for time we can discretise the equation in the form:

$$U_{ij}^{n+1} = 2U_{ij}^{n} - U_{ij}^{n-1} + \frac{\Delta t^2}{\Delta x^2}\left(U_{i+1,j}^{n} + U_{i-1,j}^{n} + U_{i,j+1}^{n} + U_{i,j-1}^{n} - 4U_{ij}^{n}\right) \tag{1}$$

Our results for $t = 0$, $t = 0.3333$, $t = 0.6667$ and $t = 1$ are presented in the graphs below for the serial code and the cases of horizontal and vertical domain splitting for MPI computation.

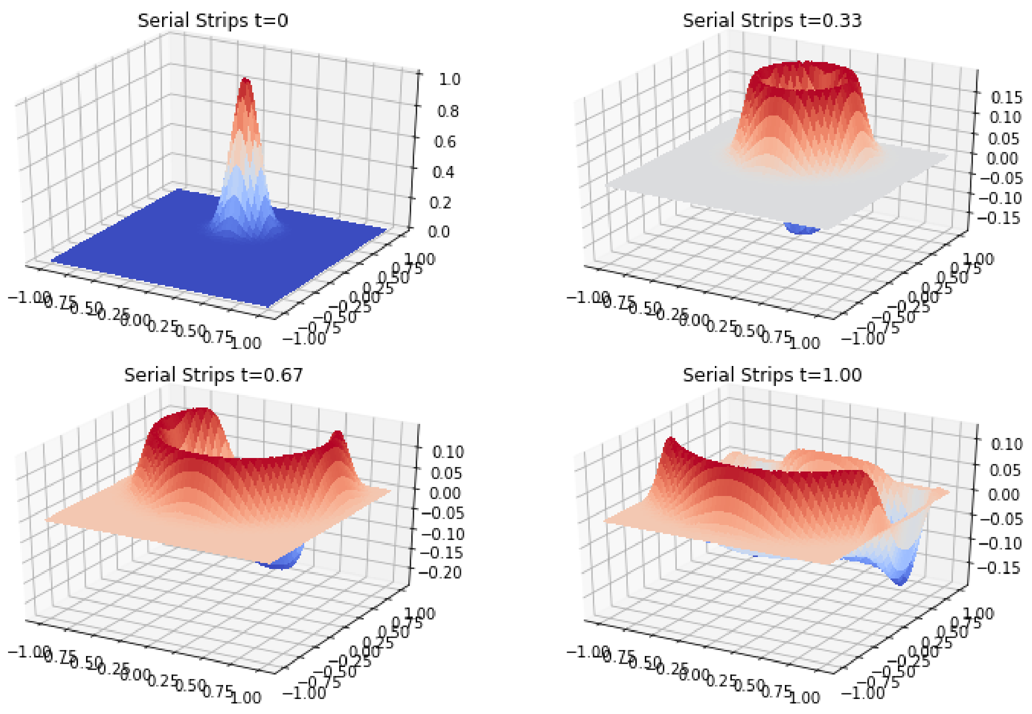


Figure 1: Serial code solution of wave equation

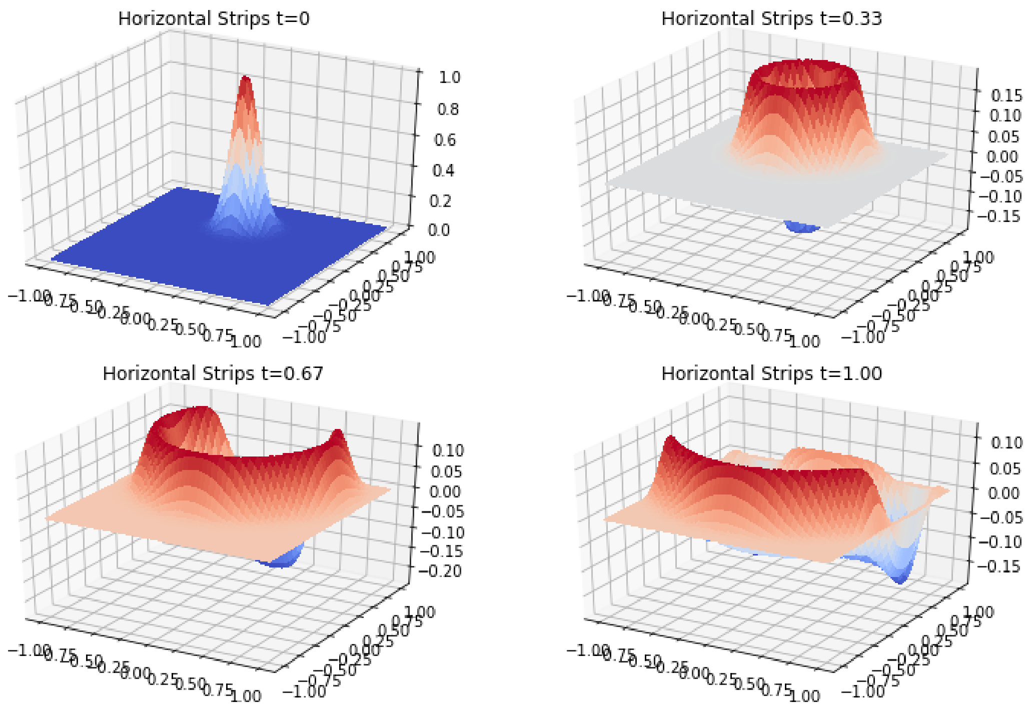


Figure 2: Horizontal Splitting Solution of wave equation

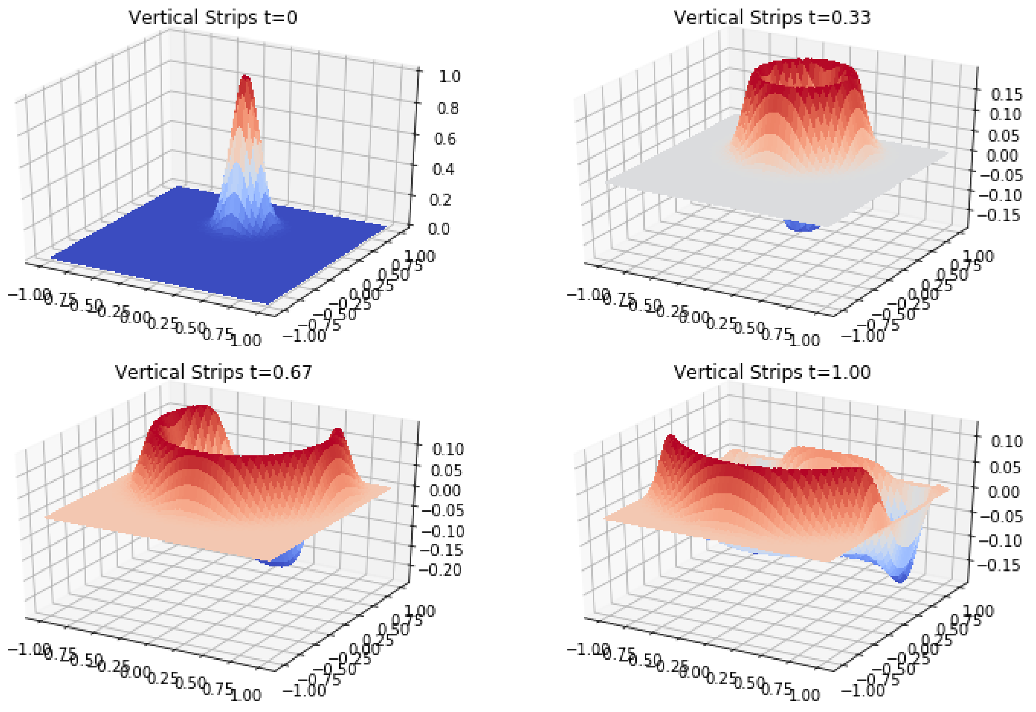


Figure 3: Vertical Splitting solution of wave equation

As one can see above the solutions are the same for all three methods we shall now explain the methods for Horizontal Splitting, Vertical splitting and square splitting followed by a performance analysis of all the methods.

For the horizontal and vertical strips method we have that the domain is equally split into strips width

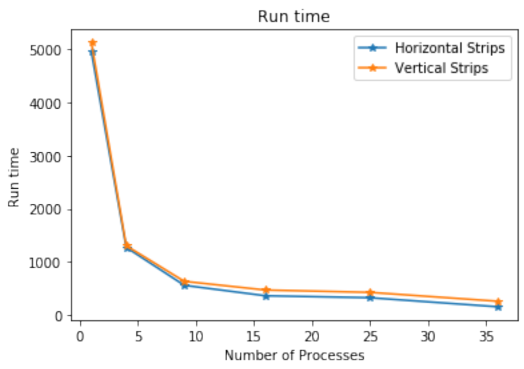
$$J = \frac{M-2}{size} + 2,$$

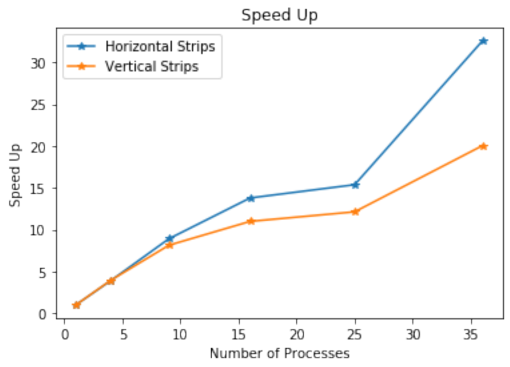
where $size$ is the number of processes and I perform halo swapping between adjacent strips after each iteration is performed on each local process. However for square decomposition I assumed that the number of processes was a square number and then I split the domain into squares of side length

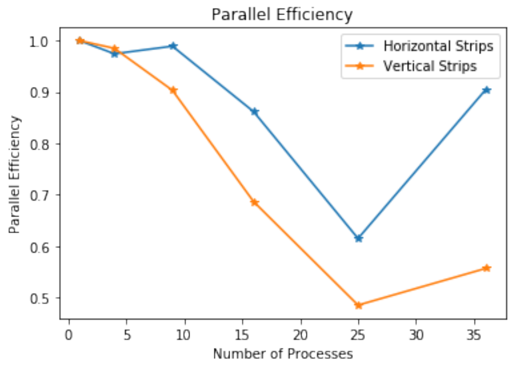
$$J = \frac{M-2}{\sqrt{size}} + 2$$

and then I perform halo swapping vertically and horizontally row and column wise. I also perform halo swapping adjacently for the corners of the squares. In both these cases we require $J > 3$.

I have now consider the case where $M = 3601$, then we run our code for the cases where $size = 1, 4, 9, 16, 25, 36$ to compare the run times for the different methods, where I shall claim that the serial runtime is roughly the same as the parallelised code, but run with one process. The results are as follows:







I have not included the square strip decomposition in the analysis plots because I have implemented the method and the code will be provided with the assignment, but there is a small bug somewhere. The run time plot shows that the horizontal strip decomposition is faster than the vertical strip decomposition. This is as expected due to the fact that $C{+}{+}$ is a programming language which works in row major order and so looping over rows and sending data which are not stored next to each other in memory will take longer than the horizontal case. I would expect that the square decomposition runtime would be inbetween horizontal strips and vertical strips because it has row and column communication.

I have also included two plots for metrics introduced in the lectures, which are the parallel efficiency and the speed up. However it seems that the parallel efficency is much higher for 36 processes this may be due to the fact that when cirrus is using all the nodes there is much better communication. Apart from this there is a clear trend that the parallel efficiency decreases with the number of processes and the speed up increases with the number of processes. Also when running comparing the time taken for the I/O process I saw that the fraction of the time taken to output the data which larger for the horizontal strip decomposition due to the fact that the parallel code is much faster for horizontal strip decomposition.

It is expected that the parallel efficency decreases with the number of processes, which means that per process we add the speed up decreases. This is expected because the problem size is being kept the same which is weak scaling. I would have liked to do an analysis of the strong scaling if I had further time.