Parallel assembly of mass matrix

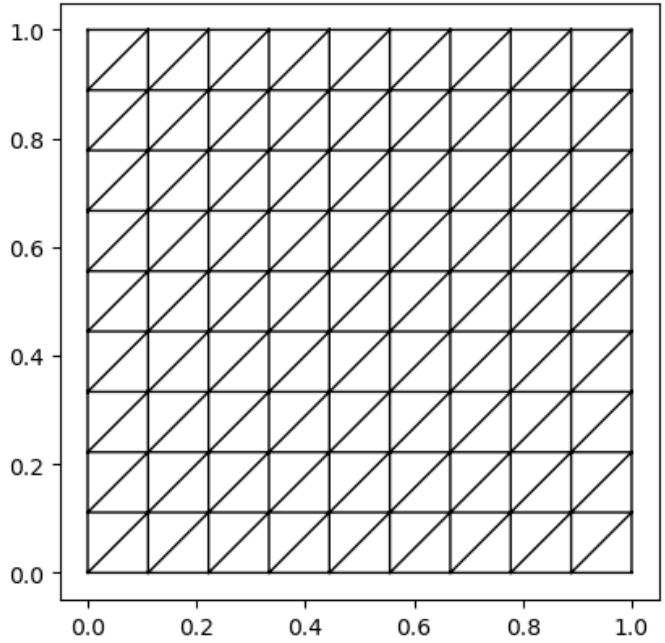
PHYS-743 Parallel programming

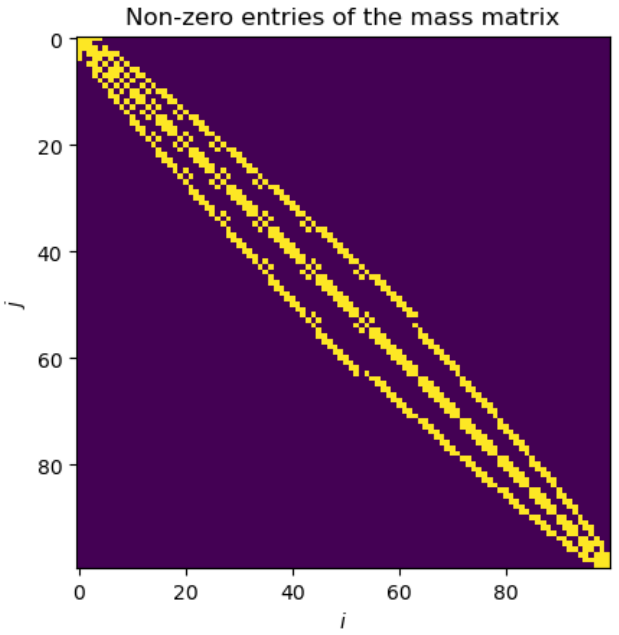
Théophile Boinnard

Novembre 20th 2025

Goal

- Compute the entries of the mass matrix $M_{ij} = \int_\Omega \phi_i \phi_j$.
- $V = \{v \in H^1(\Omega), v|_K \in \mathbb{P}^1(K), \forall K \in \mathcal{T}_h\}$, where $\mathcal{T}_h$ is a conformal mesh of Ω.
- $V = \mathrm{span}(\phi_1, \dots, \phi_M)$, M number of mesh vertices.





Théophile Boinnard

Serial algorithm

- Loop over the elements K.
- For each K, we have a mapping $\left|\begin{array}{c}\{1,2,3\} \to \{a_1, a_2, \dots, a_M\} \\ i \mapsto \mathcal{N}_i\end{array}\right.$.

For $K \in \mathcal{T}_h$
  For $i = 1,2,3$
    For $j = 1,2,3$
      $A_{\mathcal{N}_i \mathcal{N}_j} \mathrel{+}= \int_K \phi_{i,K} \phi_{j,K}$

- Complexity $O(9N_{\text{elems}})$.

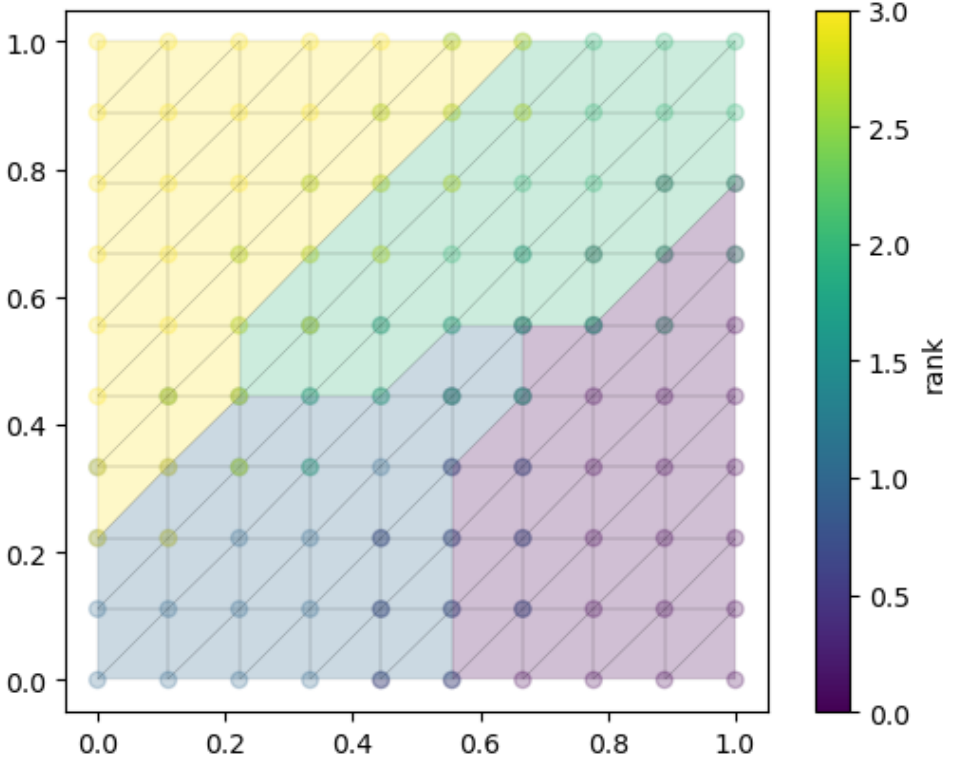
Théophile Boinnard

Parallel Algorithm

- Partition of the mesh.
- Compute mass matrix contribution on sub-mesh, using serial algorithm.
- Communicate the entries belonging to other processes.
- Build a part of the mass matrix on each process.

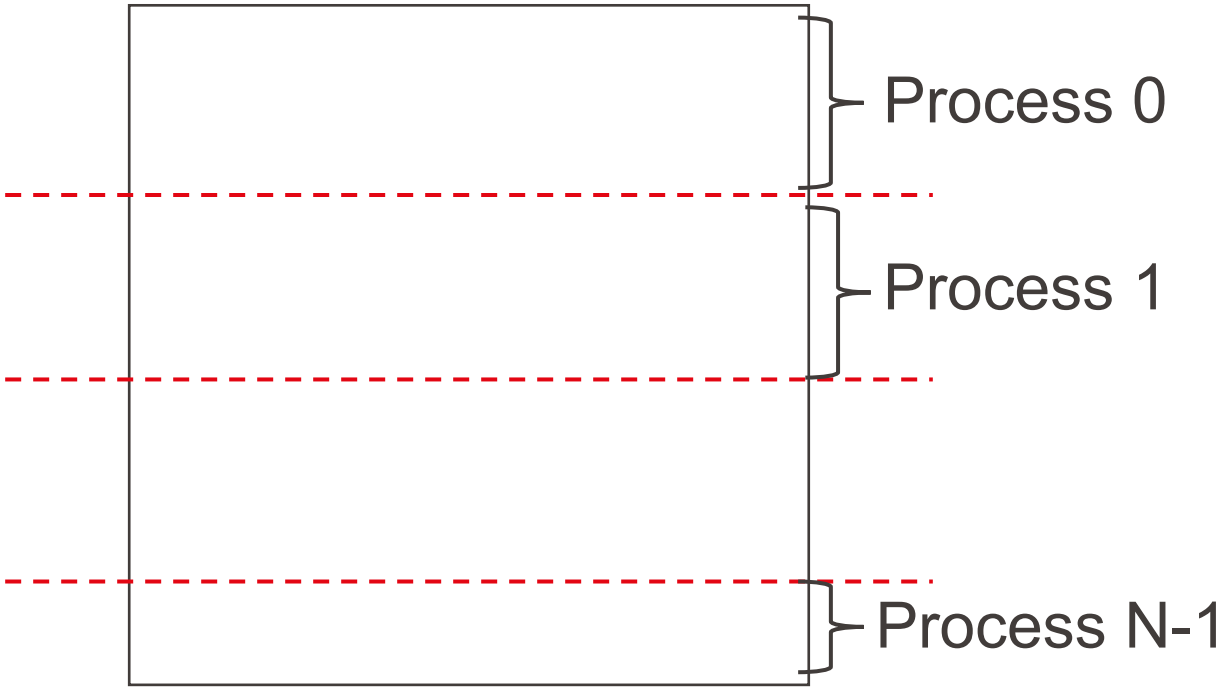
Mesh partioning

- This is done using dolfinx [1], which calls PT-Scotch [2], ParMETIS [3] or KaHIP [4].
- Process $n \in [0: N-1]$ owns its elements $\mathcal{E}_n$, its vertices $\mathcal{N}_n$ and need ghosts vertices $\widetilde{\mathcal{N}}_n$ which are vertices on other sub-meshes.



Matrix partitioning

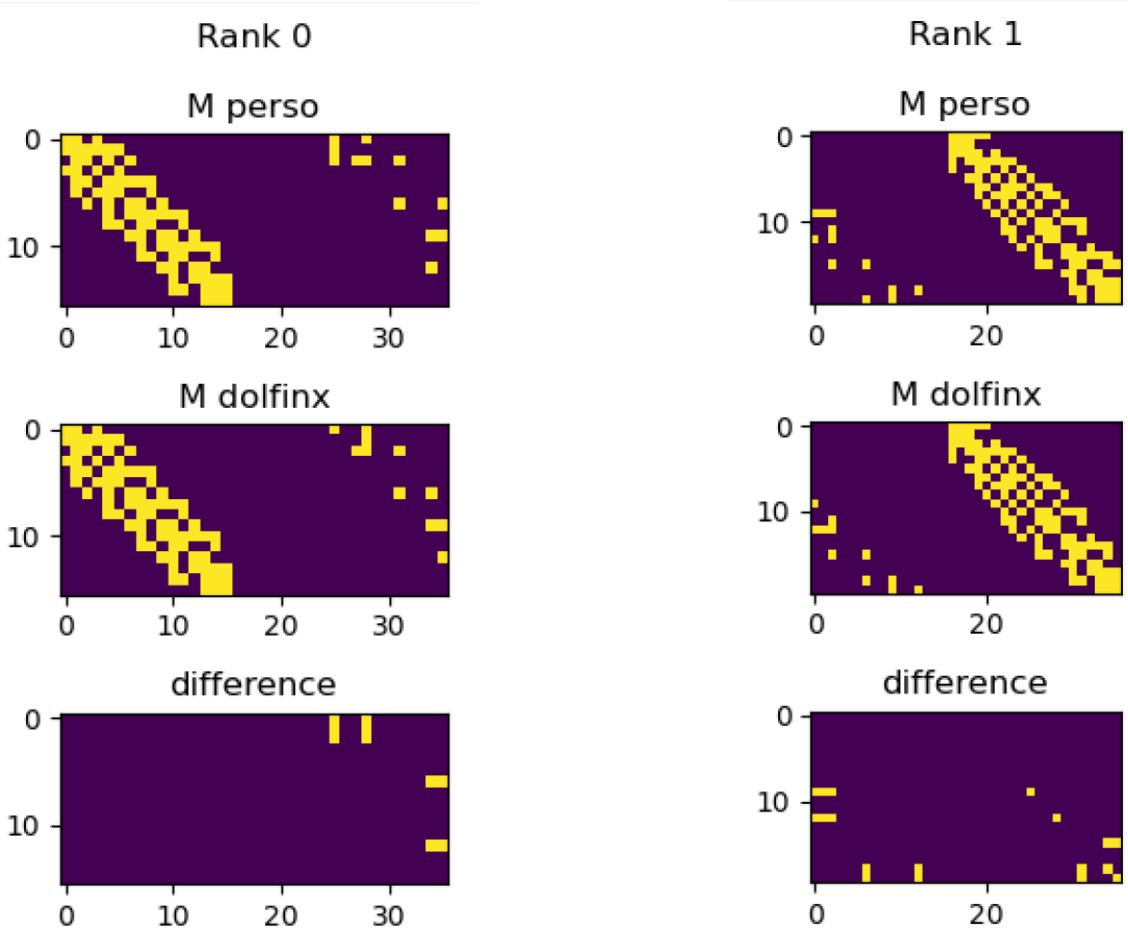
- Each process owns only a part of the mass matrix M_{ij}^{n}, where $i \in R_n$ and $j \in [1:M]$.



Communication

- Mass matrix computation returns local indices $I_{\text{local}}, J_{\text{local}}$ and values V on the sub-mesh.
- Some are ghosts : they are sent to other process using MPI.COMM_WOLRD.allgather.
- The current process verify which of the node it recieved belongs to it.
- The new values are added to the mass matrix on process

Issue with indexing... (comparison with dolfinx)



Théophile Boinnard

Solution to run on Helvetios

- I did not manage to use fenicsx on Helvetios
- The solution was to partition the mesh on my computer, and save the partition on separate h5 files using h5py.
- The code on the cluster reads the h5 files to recover the partition of the mesh.

Amdahl's law

- N_V number of vertices, N_c number of cells, $N_V/n \sim N_{\text{loc}} + N_g$ number of vertices on current process.
- Reading mesh :

```
with h5py.File(filename, 'r') as f:
    nodes = f['nodes'][:]                                  O(N_V/n)
    cells = f['cells'][:]                                  O(N_c/n)
    local_to_global = f['local_to_global'][:]              O(N_V/n)
    global_to_local = f['global_to_local'][:]              O(N_V)!
    size_local = f['size_local'][()]
```

- Computing mass matrix : $O(N_c/n)$
- Find ghosts, etc… : at most $O(N_c/n)$
- All gather : $O\left(N_g \times n\right)$!
- Find accepted vertices : $O(\max(N_{\text{loc}}, N_g \times n))$
- Conclusion $p \sim 0.95$ (experimentally…)

Théophile Boinnard

Strong scaling

Théophile Boinnard

n	1	2	4	8	16	32	64
N triangles	~2.1M	~2.1M	~2.1M	~2.1M	~2.1M	~2.1M	~2.1M
T	2.095 s	1.087 s	0.597 s	0.366 s	0.270 s	0.169 s	0.118 s
S	1	1.93	3.51	5.72	7.56	12.37	17.75
E	1	0.96	0.88	0.72	0.48	0.39	0.28

n	1	2	4	8	16	32	64
Load mesh	2.57%	1.60%	4.55%	5.29%	9.93%	18.67%	31.29%
Compute matrix	54.40%	54.90%	52.13%	44.80%	33.90%	25.66%	20.68%
Communicate	43.03%	43.50%	43.32%	49.91%	56.16%	55.67%	48.03%

Weak scaling

n	1	2	4	8	16	32	64
N triangles	~2.1M	~4.2M	~8.4M	~16.8M	~33.6 M	~67.1M	~134.2M
T	2.095 s	2.264 s	2.298 s	2.504 s	2.570 s	4.057 s	6.731 s
S	1	1.85	3.65	6.70	13.04	16.52	19.92
E	1	0.93	0.91	0.84	0.82	0.52	0.31

n	1	2	4	8	16	32	64
Load mesh	2.57%	2.39%	1.92%	3.73%	7.16%	8.05%	34.63%
Compute matrix	54.40%	52.32%	50.78%	51.94%	46.20%	37.68%	25.84%
Communicate	43.03%	45.29%	47.30%	44.32%	46.63%	54.26%	39.54%

Conclusion

- Implemented mass matrix computation in parallel with MPI.
- Helped to understand better how fenicsx works.

References

[1] https://github.com/FEniCS/dolfinx

[2] C. Chevalier, F. Pellegrini, PT-Scotch: A tool for efficient parallel graph ordering, Parallel Computing, Volume 34, Issues 6–8, 2008, Pages 318-331, ISSN 0167-8191, https://doi.org/10.1016/j.parco.2007.12.001

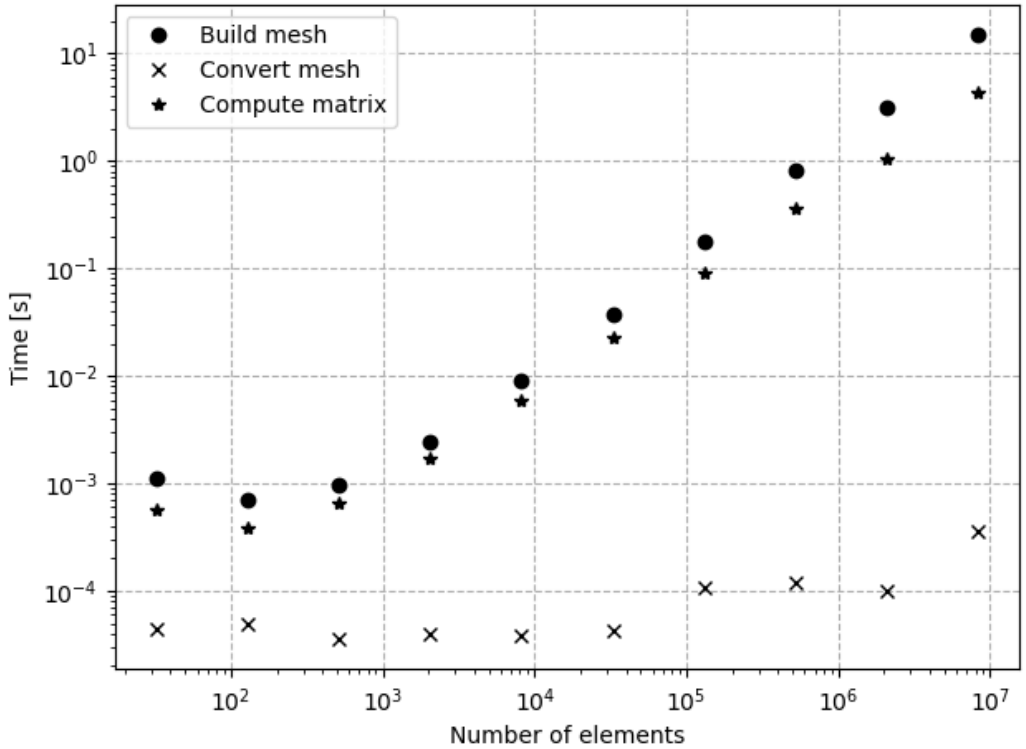
[3] George Karypis and Vipin Kumar. 1996. Parallel multilevel k-way partitioning scheme for irregular graphs. In Proceedings of the 1996 ACM/IEEE conference on Supercomputing (Supercomputing '96). IEEE Computer Society, USA, 35–es. https://doi.org/10.1145/369028.369103

[4] Sanders, P., Schulz, C. (2013). Think Locally, Act Globally: Highly Balanced Graph Partitioning. In: Bonifaci, V., Demetrescu, C., Marchetti-Spaccamela, A. (eds) Experimental Algorithms. SEA 2013. Lecture Notes in Computer Science, vol 7933. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-38527-8_16

Théophile Boinnard

Appendix – Profile in serial

- In serial, it takes $O(N_{\text{elems}})$ operations for dolfinx to create a mesh



Find p experimentally

- From Amdahl's law : $S = \frac{1}{1-p+p/n} \Rightarrow p = \frac{1-1/S}{1-1/n}$.
- For the strong scaling, we observe

$$p = [-, 96\%, 95\%, 94\%, 93\%, 95\%, 96\%]$$

Théophile Boinnard