HySoP: Hybrid Simulation with Particles

Jean-Matthieu Etancelin[1*¶], **Jean-Baptiste Keck**[3*], **Franck Pérignon**[3*], **Chloé Mimeau**[2*], **Nicolas Grima**[1*], **Christophe Picard**[3*], and **Georges-Henri Cottet**[3*]

1 Université de Pau et des Pays de l'Adour, E2S UPPA, CNRS, LMAP, Pau, France. **2** Laboratoire M2N, EA7340, CNAM, 2 rue Conté, Paris 75003, France. **3** Laboratoire Jean Kuntzmann, Grenoble INP, Université Grenobles Alpes and CNRS, Grenoble, France. ¶ Corresponding author * These authors contributed equally.

DOI: 10.xxxxxx/draft

Software

- Review
- Repository
- Archive

Editor: Philip Cardiff

Reviewers:

- @kburns
- @lluisjofre

Submitted: 14 November 2024
Published: unpublished

Summary

The present numerical tool `HySoP` is a Python package designed for high performance numerical simulations of fluid-related problems based on Remeshed Particle Methods (RPM) semi-Lagrangian methods. This tool has been successfully used for external flows, including cases involving bluff bodies, as well as for flows coupled with transport equations and flows through porous media.

In the Computational Fluid Dynamics community, the most widely used methods – such as finite difference, finite volume, finite element and spectral methods – operate within a purely Eulerian framework and have been extensively studied both in terms of consistency and stability, as well as numerical dissipation characterization.

Meanwhile, particle approaches have experienced significant development in the context of incompressible flows. These methods differ from Eulerian approaches due to their intuitive and natural description of the fluid flow, as well as their low numerical dissipation and by their ability to circumvent the nonlinearities related to the advection phenomenon. Many efforts have been devoted to addressing the main intrinsic difficulties of purely Lagrangian particle methods. These efforts have led, in particular, to the design of semi-Lagrangian approaches – also referred to as Remeshed Particle Method (RPM) – where the particles discretizing the flow are regularly remeshed onto a cartesian grid. This approach combines the strengths of both Eulerian and Lagrangian methods (Mimeau & Mortazavi, 2021).

Statement of need

The `HySoP` library (Hybrid Simulation with Particles) is designed for distributed hybrid architectures, supporting CPU and GPU compute devices through `MPI+OpenCL`. Its high level functionalities and the user interface are mainly written in `Python`. In general, `HySoP` aims to provide an architecture-agnostic, performance-portable and easily reusable numerical code. It should be noted that `HySoP` is a scientific research software under continuous development. Among the most advanced open-source softwares related to RPM and similar to `HySoP` are OpenFPM (Incardona et al., 2019) and Murphy (Gillis & Rees, 2022). Both are parallel and accelerated libraries. OpenFPM is an open-source C++ framework designed for parallel particles-only and hybrid particle-mesh codes. The interpolation schemes it provides are built to support various types of meshes, whereas `HySoP` relies solely on one-dimensional interpolations along the directions of a Cartesian grid. Murhpy is a multiresolution adaptive grid framework for numerical simulations on 3D block-structured collocated grids designed for distributed computational architectures. This multiresolution finds its benefits dealing with unbounded domains while HySoP is restricted to a homogeneous cartesian grid that does not require interpolations nor handling computational load imbalance.

Features of the software

Description of the software design

HySoP has been designed on the basis of an decoupling between the mathematical specifications of the problem to solve and the numerical methods and algorithms. The goal is to allow the user to specify only the higher level specifications, using a formulation that remains close to the mathematical formalism:

- problem parameters;
- computational domain;
- variables defined on domain;
- operators;
- overall discretisation for the cartesian grid.

Specifying lower-level details like numerical methods, target architectures, and parallelism layout is optional. The corresponding code is interchangeable and upgradable without requiring any changes to the user's code.

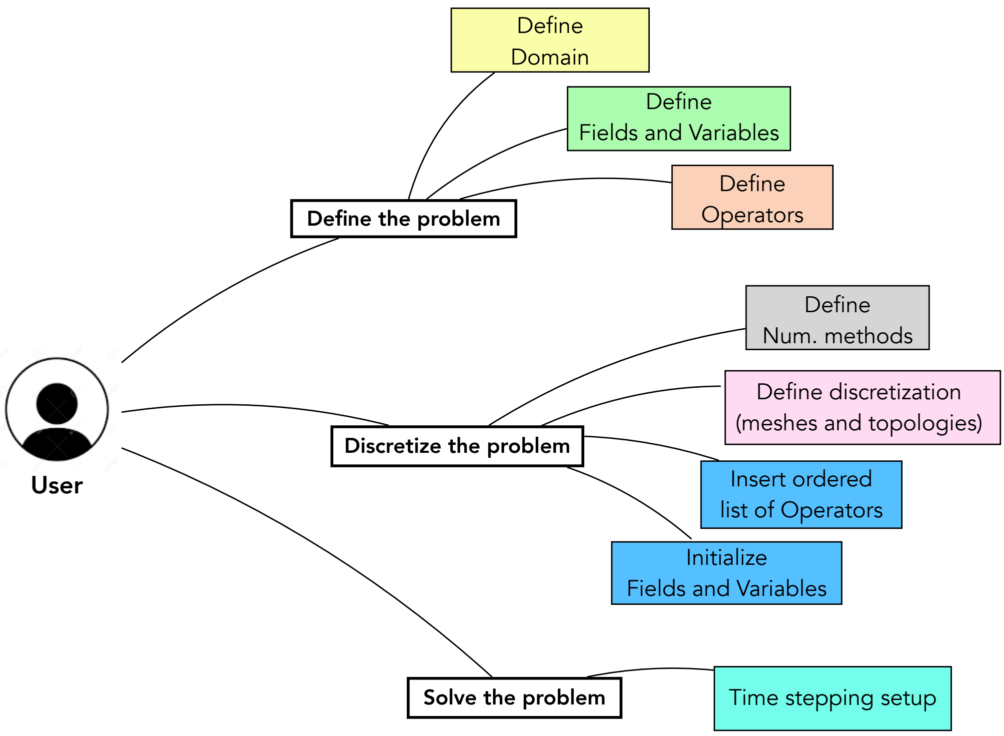

Figure 1: Use case diagram

From the user's perspective, the main use case can be decomposed into three main steps (cf Figure 1):

1. Problem description: expressed as a mathematical PDE formalism using domain, variables and operators;
2. Problem initialization: after describing numerical methods and their parameters, the user specifies the main grid resolution, the mesh decomposition for parallel execution, and the

The Journal of Open Source Software

compute backend. Then, the user defines the operator ordering and data initialization procedures. At this stage, all memory allocations are completed, generated code is optimized and compiled, and communication layout is established.

3. Problem solving: For time-dependant problems, the user provides additional parameters (e.g. time steps) and computations proceed by sequentially applying the operators.

Programming languages and external dependencies

The main drawback of pure Python programs is performance, which is mitigated by using arrays and tools from the `numpy` module. Additional performance improvements are achieved through optimized external libraries like `fftw` for fast Fourier transforms. `HySoP` is also using the `f2py` python module to use a Fortran implementation of RPM ((Lagaert et al., 2014)). Further speedup is obtained via just-in-time compiling using `numba` or `OpenCL` towards multi-core architectures. `Numba` translates Python code into compiled code at runtime, while `OpenCL` requires an explicit code. In `HySoP`, this code is generated from formal representation of the numerical methods. Additionally, micro-benchmarks are performed during initialization to optimize the parameters of the generated OpenCL code.

The interaction between the computing backends and the base `Python` layer is summarized in Figure 2. Our software architecture allows easy integration of new backends for which ensuring complete inter-operability remains the main challenge.

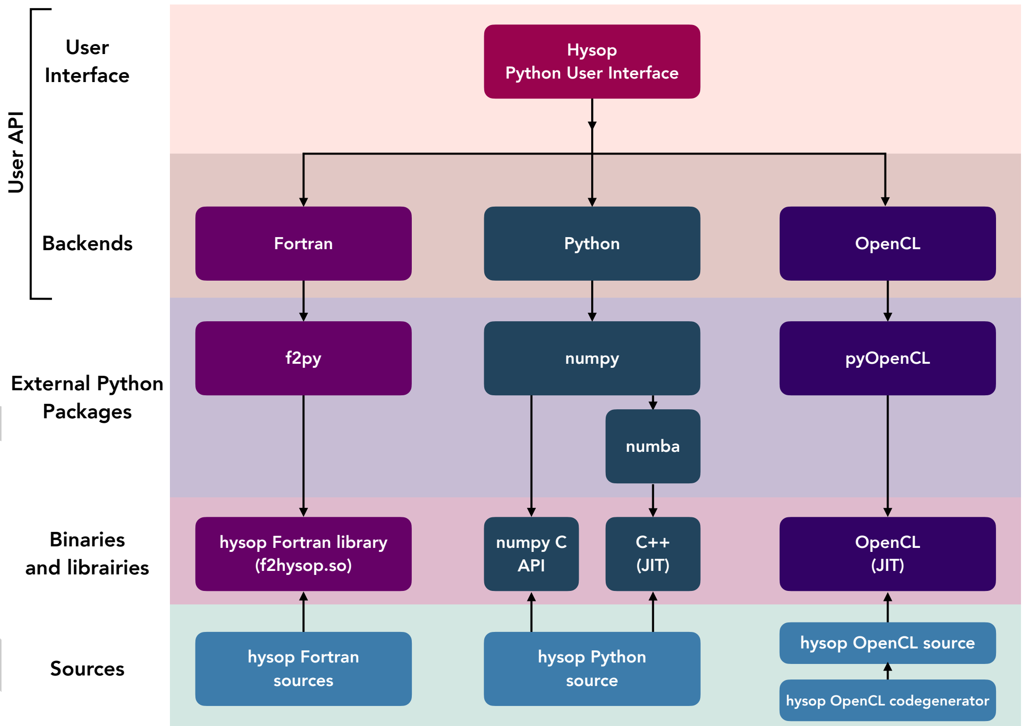

Figure 2: Computing backends in `HySoP`

Parallelism

The software supports distributed memory parallelism through domain decomposition relying on `MPI` parallelism through the `mpi4py` interface. As a result, `HySoP` may be considered as a Python-based solver that utilizes hybrid `MPI+OpenCL` programming for heterogeneous computing platforms.

`HySoP` also exploits parallelism through an operator splitting approach, enabling coarse-grained task parallelism by executing simultaneously distinct and weakly coupled operators, as defined in the overall mathematical formulation.

Evaluating HySoP's computational performance is challenging; however, references such as (Cottet et al., 2014), (Keck, 2019), and (Keck et al., 2021) describe performance insights for multi-GPU and heterogeneous platform simulations.

Applications

The following list illustrates the successful use of the HySoP library across a wide range of domains of applications, each involving diverse governing equations. This illustrates its versatility and flexibility:

Applications	Involved equations	Reference
- Bluff body flows	Navier-Stokes	(Mimeau et al., 2016, 2021)
- Large-Eddy Simulations (sub-grid scale modeling)	Filtered Navier-Stokes	(Crouy-Chanel et al., 2024)
- Transport of passive scalar at high Schmidt number	Navier-Stokes and a passive scalar advection-diffusion	(Cottet et al., 2014)
- Sedimentation in high Schmidt number flows	Navier-Stokes coupled with scalars advection-diffusion	(Keck et al., 2021)
- Passive flow control using porous media	Brinkman-Navier-Stokes	(Mimeau et al., 2017)
- Porous media dissolution at pore-scale	Darcy-Brinkman-Stokes coupled with reactive transport	(Etancelin et al., 2020)
- Porous media precipitation and crystallization at pore-scale	Darcy-Brinkman-Stokes coupled with reactive transport	(Perez et al., 2025)

For Navier-Stokes flows HySoP relies on the operator splitting to make use of most efficient numerical methods for convection terms (RPM) and diffusion terms (Fourier solvers). Scalar transport coupled with fluid flow exploits the spatial scales separations induced by large Schmidt numbers using bi-level discretization on nested Cartesian grids. Porous media are handled by the Brinkman penalization method that enables flows around permeable bodies or internal flows through porous media whose possibly complex structure is difficult to capture using traditional meshes. Finally, in the context of weakly coupled sub-problems, HySoP take advantages of task parallelism using concurrently several architectures (e.g. flow solver on CPU and reactive transport on GPU).

Acknowledgements

We acknowledge contributions from Jean-Baptiste Lagaert during the early stages of this project.

The development of HySoP has been partially supported by French ANR projects MPARME (ANR-17-CE23-0024) and HAMM (ANR-10-COSI-0009).

References

Cottet, G.-H., Etancelin, J.-M., Pérignon, F., & Picard, C. (2014). High order Semi-Lagrangian particle methods for transport equations: numerical analysis and implementation issues. *ESAIM: Mathematical Modelling and Numerical Analysis*, *48*(4), 1029–1060. https://doi.org/10.1051/m2an/2014009

Crouy-Chanel, M. de, Mimeau, C., Mortazavi, I., Mariotti, A., & Salvetti, M. V. (2024). Large-eddy simulations with remeshed vortex methods: An assessment and calibration of subgrid-scale models. *Computers & Fluids*, *277*, 106287. https://doi.org/10.1016/j.compfluid.2024.106287

Etancelin, J.-M., Moonen, P., & Poncet, P. (2020). Improvement of remeshed Lagrangian methods for the simulation of dissolution processes at pore-scale. *Advances in Water Resources*, *146*, 103780. https://doi.org/10.1016/j.advwatres.2020.103780

Gillis, T., & Rees, W. M. van. (2022). *MURPHY – a scalable multiresolution framework for scientific computing on 3D block-structured collocated grids*. https://doi.org/10.48550/arXiv.2112.07537

Incardona, P., Leo, A., Zaluzhnyi, Y., Ramaswamy, R., & Sbalzarini, I. F. (2019). OpenFPM: A scalable open framework for particle and particle-mesh codes on parallel computers. *Computer Physics Communications*, *241*, 155–177. https://doi.org/10.1016/j.cpc.2019.03.007

Keck, J.-B. (2019). *Numerical modelling and High Performance Computing for sediment flows* (Theses No. 2019GREAM067, Université Grenoble Alpes). https://tel.archives-ouvertes.fr/tel-02433509

Keck, J.-B., Cottet, G.-H., Meiburg, E., Mortazavi, I., & Picard, C. (2021). Double-diffusive sedimentation at high Schmidt numbers: Semi-Lagrangian simulations. *Physical Review Fluids*, *6*(2), L022301. https://doi.org/10.1103/PhysRevFluids.6.L022301

Lagaert, J.-B., Balarac, G., & Cottet, G.-H. (2014). Hybrid spectral-particle method for the turbulent transport of a passive scalar. *Journal of Computational Physics*, *260*, 127–142. https://doi.org/10.1016/j.jcp.2013.12.026

Mimeau, C., Cottet, G.-H., & Mortazavi, I. (2016). Direct numerical simulations of three-dimensional flows past obstacles with a vortex penalization method. *Computers and Fluids*, *136*, 331–347. https://doi.org/10.1016/j.compfluid.2016.06.020

Mimeau, C., Marié, S., & Mortazavi, I. (2021). A comparison of semi-Lagrangian Vortex method and Lattice Boltzmann method for incompressible flows. *Computers and Fluids*, *224*, 104946. https://doi.org/10.1016/j.compfluid.2021.104946

Mimeau, C., & Mortazavi, I. (2021). A Review of Vortex Methods and Their Applications: From Creation to Recent Advances. *Fluids*, *6*(2), 68. https://doi.org/10.3390/fluids6020068

Mimeau, C., Mortazavi, I., & Cottet, G.-H. (2017). Passive control of the flow around a hemisphere using porous media. *European Journal of Mechanics - B/Fluids*, *65*, 213–226. https://doi.org/10.1016/j.euromechflu.2017.03.002

Perez, S., Etancelin, J.-M., & Poncet, P. (2025). A semi-Lagrangian method for the direct numerical simulation of crystallization and precipitation at the pore scale. *Frontiers in Earth Science*, *13*, 1493305. https://doi.org/10.3389/feart.2025.1493305