




OpenCMP: An Open-Source Computational Multiphysics Package

Elizabeth J. Monte¹, Alexandru Andrei Vasile¹, James Lowman¹, and Nasser Mohieddin Abukhdeir^{*1, 2}

¹ Department of Chemical Engineering, University of Waterloo, Ontario, Canada ² Department of Physics and Astronomy, University of Waterloo, Ontario, Canada

DOI: [10.21105/joss.03742](https://doi.org/10.21105/joss.03742)

Software

- [Review](#) 
- [Repository](#) 
- [Archive](#) 

Editor: [Lucy Whalley](#) 

Reviewers:

- [@bonh](#)
- [@WilkAndy](#)

Submitted: 16 August 2021

Published: 21 September 2021

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](#)).

Summary

OpenCMP is a computational multiphysics software package based on the finite element method ([Ferziger & Perić, 2002](#)). It is primarily intended for physicochemical processes in which fluid convection plays a significant role. OpenCMP uses the NGSolve finite element library ([Schöberl, 2020](#)) for spatial discretization and provides a configuration file-based interface for pre-implemented models and time discretization schemes. It also integrates with Netgen ([Schöberl, 2020](#)) and Gmsh ([Geuzaine & Remacle, 2009](#)) for geometry construction and meshing. Additionally, it provides users with built-in functionality for post-processing, error analysis, and data export for visualisation using Netgen ([Schöberl, 2020](#)) or ParaView ([Ahrens et al., 2005](#)).

OpenCMP development follows the principles of ease of use, performance, and extensibility. The configuration file-based user interface is intended to be concise, readable, and intuitive. Furthermore, the code base is structured such that experienced users with appropriate background can add their own models with minimal modifications to existing code. The finite element method enables the use of high-order polynomial interpolants for increased simulation accuracy, however, continuous finite element methods suffer from stability and accuracy (conservation) for fluid convection-dominated problems. OpenCMP addresses this by providing discontinuous Galerkin method ([Cockburn et al., 2000](#)) solvers, which are locally conservative and improve simulation stability for convection-dominated problems. Finally, OpenCMP implements the diffuse interface method ([Monte et al., 2021](#)), an immersed boundary method ([Mittal & Iaccarino, 2005](#)), which allows complex simulation domains to be meshed by non-conforming structured meshes for improved simulation stability and reduced computational complexity (under certain conditions ([Monte et al., 2021](#))).

Statement of Need

Simulation-based analysis continues to revolutionise the engineering design process. Simulations offer a fast, inexpensive, and safe alternative to physical prototyping for preliminary design screening and optimisation. Simulations can also offer more detailed insight into the physical phenomena of interest than is feasible from experimentation. Computational multiphysics is an area of particular importance since coupled physical and chemical processes are ubiquitous in science and engineering.

However, there are several barriers to more wide spread use of simulations. One such barrier is a lack of suitable software accessible to the general scientific and engineering community. Many

^{*}Corresponding author

of the most widely-used computational multiphysics software packages, such as COMSOL Multiphysics ([COMSOL Multiphysics](#)®, n.d.) or ANSYS Fluent ([Ansys](#)® *Fluent*, n.d.), are closed-source and cost-prohibitive. Furthermore, the predominance of the finite volume method for fluid dynamics simulations inherently limits simulation accuracy for a given mesh compared to the finite element method ([Ferziger & Perić, 2002](#)). Open-source packages that uses the finite element method, such as the computational multiphysics software MOOSE ([Permann et al., 2020](#)) or the finite element libraries NGSolve ([Schöberl, 2020](#)) and FEniCS ([Alnaes et al., 2015](#)), have very steep learning curves and are inaccessible to the general scientific and engineering communities.

A second barrier is the complex geometries inherent to many real-world problems. Conformal meshing of these complex geometries is time and labour intensive and frequently requires user-interaction, making conformal mesh-based simulations infeasible for high-throughput design screening of many industrially-relevant processes ([Yu et al., 2015](#)). A possible solution is the use of immersed boundary methods ([Mittal & Iaccarino, 2005](#)) to allow the use of non-conforming structured meshes - simple and fast to generate - for any geometry. Use of structured meshes can also potentially improve simulation stability compared to unstructured meshes ([Yu et al., 2015](#)). The diffuse interface method, a form of immersed boundary method, has been shown by Monte *et al* ([Monte et al., 2021](#)) to significantly speed up inherently low accuracy simulations - such as those used for preliminary design screening and optimisation - compared to conformal mesh-based simulations. Providing an easy-to-use publicly available implementation of said method would enable broader use by the research community and further development.

The goal of OpenCMP is to fill this evident need for an open-source computational multiphysics package which is user friendly, based on the finite element method, and which includes the diffuse interface method. OpenCMP is built on top of the NGSolve finite element library ([Schöberl, 2020](#)) to take advantage of its extensive finite element spaces and high performance solvers and preconditioners. OpenCMP provides pre-implemented models and a configuration file-based user interface in order to be accessible to the general simulation community, not just finite element experts. The user interface is designed to be intuitive, readable, and requires no programming experience - solely knowledge of the command line interface. Users must choose the model that suites their application, but need no experience with the actual numerical implementation of said model. Finally, OpenCMP provides the only publicly available implementation of the diffuse interface method.

Features

The table below summarises the current capabilities of OpenCMP. Future work on OpenCMP will focus on adding models - for multi-phase flow, turbulence, and heat transfer - and enabling running simulations in parallel over multiple nodes with MPI ([Forum, 2015](#)).

Feature	Description
Meshing	Accepts Netgen (Schöberl, 2020) or Gmsh (Geuzaine & Remacle, 2009) meshes
Numerical Methods	Standard continuous Galerkin finite element method Discontinuous Galerkin finite element method Diffuse interface method
Models	Poisson equation Stokes equations Single-component incompressible Navier-Stokes equations Multi-component reacting incompressible Navier-Stokes equations
Time Schemes	First-, second-, and third- order discretizations Adaptive time-stepping

Feature	Description
Solvers	Direct or iterative solvers Direct, Jacobi, or multigrid preconditioners Oseen (Cockburn et al., 2003) or IMEX (Ascher et al., 1995) linearization of nonlinear models
Post-Processing	Error norms calculated based on reference solutions Mesh and polynomial refinement convergence tests General simulation parameters (surface traction, divergence of velocity...) Exports results to Netgen (Schöberl, 2020) or ParaView (Ahrens et al., 2005) format
Performance	Multi-threading

Acknowledgements

The authors would like to thank Prof. Sander Rhebergen for useful discussions regarding the discontinuous Galerkin method and Prof. Joachim Schöberl for useful discussions regarding IMEX time integration, preconditioning, and usage of the NGSolve finite element library. This research was supported by the Natural Sciences and Engineering Research Council (NSERC) of Canada and Compute Canada.

References

- Ahrens, J., Geveci, B., & Law, C. (2005). *ParaView: An end-user tool for large data visualization* (Technical Report LA-UR-03-1560). Los Alamos National Laboratory.
- Alnaes, M., Blechta, J., Hake, J., Johansson, A., Kehlet, B., Logg, A., Richardson, C., Ring, J., Rognes, M. E., & Wells, G. N. (2015). The FEniCS Project Version 1.5. *Archive of Numerical Software*, 3. <https://doi.org/10.11588/ans.2015.100.20553>
- Anslys[®] Fluent. (n.d.). ANSYS Inc. Online. <https://www.ansys.com/products/fluids/ansys-fluent>
- Ascher, U. M., Ruuth, S. J., & Wetton, B. T. R. (1995). Implicit-explicit methods for time-dependent partial differential equations. *SIAM Journal on Numerical Analysis*, 32(3), 797–823. <https://doi.org/10.1137/0732037>
- Cockburn, B., Kanschat, G., & Schötzau, D. (2003). The local discontinuous Galerkin method for the Oseen equations. *Mathematics of Computation*, 73(246), 569–593.
- Cockburn, B., Karniadakis, G. E., & Shu, C.-W. (2000). *Discontinuous Galerkin methods: Theory, computation and applications* (1st ed., pp. 3–50). Springer-Verlag. ISBN: 3-540-66787-3
- COMSOL Multiphysics[®]. (n.d.). COMSOL AB; Online. <https://www.comsol.com/>
- Ferziger, J. H., & Perić, M. (2002). *Computational methods for fluid dynamics* (3rd ed.). Springer-Verlag. ISBN: 3-540-42074-6
- Forum, M. P. I. (2015). *MPI: A message-passing interface standard version 3.1*. High-Performance Computing Center.
- Geuzaine, C., & Remacle, J.-F. (2009). Gmsh: A three-dimensional finite element mesh generator with built-in pre- and post-processing facilities. *International Journal for Numerical Methods in Engineering*, 79(11), 1309–1331. <https://doi.org/10.1002/nme.2579>

- 106 Mittal, R., & Iaccarino, G. (2005). Immersed boundary methods. *Annual Review of Fluid*
107 *Mechanics*, 37, 239–261. <https://doi.org/10.1146/annurev.fluid.37.061903.175743>
- 108 Monte, E. J., Lowman, J., & Abukhdeir, N. M. (2021). *A diffuse interface method for*
109 *simulation-based screening of heat transfer processes with complex geometries*.
- 110 Permann, C. J., Gaston, D. R., Andrš, D., Carlsen, R. W., Kong, F., Lindsay, A. D., Miller,
111 J. M., Peterson, J. W., Slaughter, A. E., Stogner, R. H., & Martineau, R. C. (2020).
112 MOOSE: Enabling massively parallel multiphysics simulation. *SoftwareX*, 11, 100430.
113 <https://doi.org/10.1016/j.softx.2020.100430>
- 114 Schöberl, J. (2020). *Netgen/NGSolve*. Online. <https://ngsolve.org/>
- 115 Yu, W., Zhang, K., & Li, X. (2015, July). Recent algorithms on automatic hexahedral mesh
116 generation. *The 10th International Conference on Computer Science & Education*.

DRAFT