




ARC4CFD: Learning how to leverage High-Performance Computing with Computational Fluid Dynamics

Jean-Pierre Hickey ^{1*}, Francesco Ambrogi ^{1*}, Sophie Hillcoat¹, Jeswin Joseph¹, and Nipin Lokanathan¹

¹ Department of Mechanical and Mechatronics Engineering, University of Waterloo, Canada * These authors contributed equally.

DOI: [10.21105/jose.00252](https://doi.org/10.21105/jose.00252)

Software

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

Submitted: 18 April 2024

Published: 05 May 2025

License

Authors of papers retain copyright and release the work under a Creative Commons Attribution 4.0 International License ([CC BY 4.0](https://creativecommons.org/licenses/by/4.0/)).

Summary

Computational Fluid Dynamics (CFD) is a field of computational physics that relies heavily on modern Advanced Research Computing (ARC) resources (Cant, 2002). The spatial and temporal resolutions required to solve modern CFD problems means that they can take advantage of the full benefits of large-scale distributed-memory parallelization that is available on high-performance computing (HPC) systems on ARC infrastructure. The CFD user base is broad, diverse and interdisciplinary. As CFD tools have progressed over the past decades, the improved robustness, predictive capabilities, and user-friendliness led them to be adopted by nontraditional HPC users such as new graduate students, experimentalists, theoreticians, and student design teams. Advanced Research Computing for Computational Fluid Dynamics, or ARC4CFD, is an open source, asynchronous online course (<https://arc4cfid.github.io>) that was developed to give users a basic understanding of fluid dynamics and CFD to bridge the knowledge gap toward an effective use of CFD on modern ARC resources.

Statement of need

Most science and engineering university programs offer CFD courses, which have become a standard part of the curriculum in mechanical, civil, chemical, and aerospace engineering. These courses generally focus on the fundamental understanding of the physics, modeling, and numerics as well as the practical use of CFD tools. At the end of a typical undergraduate CFD course, students have learned to use a CFD tool to solve small-scale problems, understand the modelling assumptions, quantify numerical errors, and visualize the results. The CFD problems used in these courses are designed to run on students' personal computers or local workstations. The jump from small-scale CFD on a local workstation to the effective use of the same CFD tool on a much larger problem size, on modern HPC systems, is nontrivial and requires additional specific training. Although generic HPC and parallel computing training is widely available, targeted training material for CFD users on HPC systems is not readily found. Thus, these learners must either: a) “translate” the knowledge from generic HPC training material to the field of CFD, or b) rely on mentorship and external help to effectively use these computational resources. As CFD tools are continually improving, they are increasingly being adopted to complement experimental campaigns, inform the design of experiments, or help solve theoretical fluid dynamics problems. The broad user base needs focused training materials for effectively using CFD on HPC systems, as they may not have the mentorship opportunities or access to expertise as graduate students. ARC4CFD was developed to provide this CFD-specific training. The course uses an asynchronous format for broad adoption, and integrates many hands-on problems to help the learner further their understanding and learn independently.

We value the ability to freely share and exchange with the community, and thus focused on an entirely open-source toolset, from meshing, solving, and visualizing the results. Although this course is intended for use on remote HPC systems, and more specifically on the Digital Research Alliance of Canada clusters, it is possible to follow the course on most other HPC systems that use a Linux-based environment with a SLURM workload manager.

ARC4CFD is intended to be an introductory course that combines concepts from CFD and computing with an *end-user focus*. The course is built with the assumption that students have: a) a undergraduate-level knowledge in fluid dynamics, b) introductory knowledge in Computational Fluid Dynamics, and c) familiarity with navigating terminal and bash commands on remote systems. The target audience includes:

- New graduate students in computational physics or engineering.
- Experimentalists and theoreticians complementing their work with numerical simulations on HPC.
- Undergraduate students on student design teams interested in leveraging CFD with HPC.

Given the prerequisite knowledge and target audience, the course needs to provide both a theoretical understanding of parallel computing and a systematic, hands-on approach to set up effective workflows for CFD on HPC. The lessons first offer a conceptual understanding and then apply that understanding to specific CFD problems; a summative example is woven into the course which follows the learner through the main CFD workflow.

Course structure and learning outcomes

The course is expected to take about 16 hours to complete and is divided into three sections. Each section consists of several lessons with individually-defined learning outcomes. The sections are structured in a way that the learner can first develop a foundational understanding of HPC ([Section 1](#)), translate those concepts to specific challenges in CFD and establish a systematic workflow ([Section 2](#)), and then effectively manage the resulting research data ([Section 3](#)). The course guides the learner through all the necessary steps towards effective use of CFD on HPC, and is supplemented with many practical examples and interactive quizzes. Section 2, the core section of the course, includes most of the CFD-specific knowledge for usage in HPC; for that reason, the lessons in this section directly map onto a CFD workflow on HPC system. Each step in the CFD workflow is an individual lesson, as follows:

- 2.1 Definition of the CFD workflow.
- 2.2 Plan the large-scale CFD simulation.
- 2.3 Estimate the HPC requirements.
- 2.4 Preprocess the CFD simulation.
- 2.5 Optimize the CFD for HPC.
- 2.6 Running the CFD on HPC.
- 2.7 A posteriori analysis.

Section 2 offers a comprehensive overview of a standard workflow for a CFD problem on HPC systems. It also includes a summative example, the simulation of a backward-facing step, following the main steps of the CFD workflow. This summative example highlights the methodology to estimate the computational cost, setup the simulation, mesh, run, and postprocess. Additional examples include boundary layers (both laminar and turbulent), highlighting the near-wall resolution requirements, and a cylinder in crossflow to illustrate a sample mesh-generation process.

ARC4CFD has five learning outcomes. At the end of the course, the learner should be able to:

1. Define the main concepts of parallel and high-performance computing.
2. Conduct an a priori estimate of the computational cost of a CFD simulation.
3. Explain the impact of the modelling assumptions on HPC cost.
4. Optimize the simulation parameters of a CFD problem for HPC.
5. Develop a research data-management strategy for a CFD workflow.

The learning outcomes for each lesson are presented and reinforced with questions at the end of each lesson.

Philosophy of learning

ARC4CFD is an asynchronous online course that accompanies the learner in developing skills to effectively use Advanced Research Computing for CFD. The course adopts an *integrative learning* approach in which concepts from a range of scientific disciplines are brought together (e.g., parallel computing, programming, numerics, fluid dynamics, CFD) and put into practice through hands-on examples and quizzes. The course uses an open-source meshing, solution, and visualization toolset. This is a key feature of the course that we hope will encourage broad adoption of the material. The course is licensed under the Creative Commons BY-NC-SA license, and all the original files (e.g., svg files or Python scripts) are included in the repository.

The course aims to: 1) emphasize a systematic approach for the effective use of CFD on ARC systems, 2) provide a high-level theoretical understanding of the main concepts for using CFD on HPC, and 3) provide hands-on examples for the learners to put these concepts into practice. The multimodal course content includes written content, videos, interactive quizzes, and hands-on examples that cover various aspects of ARC.

Course features

The asynchronous course is built on a GitHub Pages website [ARC4CFD](#) using the static site generator [Astro](#) and the [Astro Starlight](#) theme. This Astro template enables interactive engagement through the use of self-correcting quizzes, copy-pasting of code snippets, and visually appealing course navigation. The main website is supplemented with a more traditional [Git repository](#) that contains all the examples used in the course. Many of the lessons feature a short introductory video to guide the learners through the context. The main open-source toolkits used include: Gmsh (meshing), SU2 (CFD solver), OpenFoam (CFD solver), Paraview (visualization), and Python (postprocessing and analysis). Gmsh is a widely used meshing tool that can be used to generate structured and unstructured meshes for both solid and fluid mechanics ([Geuzaine et al., 2009](#)). The two different open-source CFD frameworks are used for better reach to the various communities; both solvers rely on a classical finite volume formulation. OpenFoam ([Weller et al., 1998](#)) is a toolbox to solve problems in continuum mechanics and is one of the most used CFD solvers, due to its versatility in developing complex numerical solvers. SU2, on the other hand, has a stronger aerospace heritage with a growing user base across various disciplines ([Palacios et al., 2013](#)). Finally, visualization of the results relies on Paraview ([Ahrens et al., 2005](#)).

A unique aspect of the training material is found in Section 3 on Research Data Management (RDM). As RDM approaches are continually being integrated into scientific disciplines, we opted to provide some CFD-specific perspective to the RDM discussion. The content will undoubtedly evolve with these concepts, but we hope that this training material will be of assistance to CFD users that are increasingly being asked to develop Data Management Plans (DMPs) as part of grant proposals.

Story of the project

This course was conceived primarily from the need to train new graduate students in the Multi-Physics Interaction Lab at the University of Waterloo to bridge the gap between their undergraduate CFD education and the effective use of these CFD tools on high-performance computing. The typical undergraduate CFD coursework focuses on numerical modelling and usage of CFD tools, whereas the majority of HPC training material tends to emphasize the computer science aspects of HPC. The effective use of modern Advanced Research Computing facilities requires integrating knowledge from both CFD and HPC. It was from the need to centralize the information on the usage of CFD in ARC that the idea for this course was born. Previously, this information was transmitted, piecemeal, to students as needed during their graduate training. This led to inhomogeneous training, which often left unresolved knowledge gaps for these otherwise highly qualified graduate students.

A timely opportunity arose during a sabbatical through a call for proposals from Compute Ontario for the development of training materials for underserved HPC users. This opportunity provided the means to build a team to centralize the necessary information and build this course. The team built the course by focusing on three user groups: 1) new graduate students that plan to use HPC resources to run CFD simulations, 2) theoreticians and experimentalists that want to use CFD on HPC to supplement their work, and 3) undergraduate design team members who want to develop skills in HPC. Inspired by the “12-steps to the Navier-Stokes” course (Barba & Forsyth, 2019), which is a staple of our graduate student training, we tried to recreate a systematic approach towards the usage of CFD on HPC systems. The initial idea of building everything within a Jupyter notebook eventually shifted to an interactive webpage with video content due to the challenges of directly using HPC systems through a Pythonic interface.

Looking ahead, this course will continue to evolve over the coming year and will serve as the basis for a synchronous course that will be offered as part of Compute Ontario’s summer school. As learners benefit from the course and we continually adjust the course based on user comments, we will be able to improve the content to meet the changing needs of new graduate students in CFD, experimentalists and theoreticians, and undergraduates that are interested to learn CFD on HPC.

Acknowledgments

This course was developed in the Multi-Physics Interaction Lab (www.mpilab.ca) with financial support from Compute Ontario.

References

- Ahrens, J., Geveci, B., Law, C., Hansen, C., & Johnson, C. (2005). Paraview: An end-user tool for large-data visualization. *The Visualization Handbook*, 717, 50038–50031. <https://doi.org/10.1016/b978-012387582-2/50038-1>
- Barba, L., & Forsyth, G. (2019). CFD Python: The 12 steps to Navier-Stokes equations. *Journal of Open Source Education*, 2(16), 21. <https://doi.org/10.21105/jose.00021>
- Cant, S. (2002). High-performance computing in computational fluid dynamics: Progress and challenges. *Philosophical Transactions of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences*, 360(1795), 1211–1225. <https://doi.org/10.1098/rsta.2002.0990>
- Geuzaine, C., Remacle, J.-F., & others. (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>

Palacios, F., Alonso, J., Duraisamy, K., Colonno, M., Hicken, J., Aranake, A., Campos, A., Copeland, S., Economon, T., Lonkar, A., Lukaczyk, T., & Taylor, T. (2013). Stanford University Unstructured (SU²): An open-source integrated computational environment for multi-physics simulation and design. In *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition*. American Institute of Aeronautics; Astronautics. <https://doi.org/10.2514/6.2013-287>

Weller, H. G., Tabor, G., Jasak, H., & Fureby, C. (1998). A tensorial approach to computational continuum mechanics using object-oriented techniques. *Computers in Physics*, 12(6), 620–631. <https://doi.org/10.1063/1.168744>