




OpenCCM: An Open-Source Continuous Compartmental Modelling Package

Alexandru Andrei Vasile ¹, Matthew Peres Tino ¹, Yuvraj Aseri ¹, and Nasser Mohieddin Abukhdeir ^{1,2,3}✉

¹ Department of Chemical Engineering, University of Waterloo, Ontario, Canada ² Department of Physics and Astronomy, University of Waterloo, Ontario, Canada ³ Waterloo Institute for Nanotechnology, University of Waterloo, Ontario, Canada ✉ Corresponding author

DOI: [10.21105/joss.06963](https://doi.org/10.21105/joss.06963)

Software

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

Editor: Philip Cardiff 

Reviewers:

- [@luohancfd](#)
- [@speth](#)

Submitted: 02 April 2024

Published: 25 October 2024

License

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

Summary

OpenCCM is a compartmental modelling ([Jourdan et al., 2019](#)) software package based on recently developed fully automated flow alignment compartmentalization methods ([Vasile et al., 2024](#)). It is primarily intended for large-scale flow-based processes with weak coupling between composition changes, e.g. through (bio)chemical reactions, and convective mass transport in the system. Compartmental modelling is an important approach used to develop reduced-order models ([Chinesta et al., 2017](#)) ([Benner et al., 2020](#)) using a priori knowledge of process hydrodynamics ([Jourdan et al., 2019](#)). Compartmental modelling methods, such as those implemented in OpenCCM, enable simulations of these processes with far less computational complexity while still capturing the key aspects of process dynamics.

OpenCCM integrates with two multiphysics simulation software packages, OpenCMP ([Monte et al., 2022](#)) and OpenFOAM ([Greenshields, 2024](#)), allowing for ease of transferring simulation data for compartmentalization. Additionally, it provides users with built-in functionality for computing residence times and exporting for use in other simulation or visualization software, including ParaView ([Ayachit, 2015](#)). Post-processing methods are included for mapping simulation results from compartment domains to the original simulation domain, which are useful for visualization purposes and for further simulations in using other software (e.g., multi-scale modelling).

Statement of Need

Simulation-based design and analysis continue to be widely applied in the research and development of physicochemical processes. Processes with large differences in characteristic time and length scales result in the infeasibility of direct multiphysics simulations due to computational limitations. This imposes significant computational costs, which severely reduce the utility of these simulations for entire classes of processes. Compartmental modeling is well-suited for such applications because it enables the generation of reduced-order models which are less computationally demanding, frequently by orders of magnitude, compared to direct continuum mechanical simulations. This is enabled by taking advantage, when present, of weak couplings between the short and long-time-scale dynamic phenomena.

However, several barriers prevent the more widespread use of compartmental models. The largest of these is the lack of software for automating the generation of compartmental models. Closed-source software packages, specifically AMBER ([Quintessa, n.d.](#)), exist for manually creating and solving well-mixed compartment networks. However, the cost of these packages is prohibitive for much of the research community and it lacks automated compartmentalization. Open-source software, Cantera ([Goodwin et al., 2023](#)), also exists to solve compartment

networks. However, it does not incorporate flow information, when available, either from direct observation or continuum mechanical simulations. Furthermore, neither of these software allow for the usage and direct transfer of flow information from continuum mechanical simulations, such as computational fluid dynamics (CFD) simulations, which are typically feasible over short (hydrodynamic) time scales.

The overall aim of OpenCCM is to fill the need for an open-source compartmental modeling package that is user-friendly, compatible with a variety of simulation package back-ends (e.g., OpenFOAM and OpenCMP), and which fits into the user's existing simulation and post-processing software toolchain, i.e., ParaView.

Features

FEATURE	DESCRIPTION
Model support	Accepts OpenCMP (Monte et al., 2022) and OpenFOAM (Greenshields, 2024) results
Compartmentalization	Single-phase flow-based compartment identification
Compartmental Modelling	Plug Flow Reactors (PFRs)-in-series-based model Previous state-of-the-art Continuous Stirred-tank reactor (CSTR)-based models
CM Simulations	Linear, non-linear, and reversible arbitrary reactions 1st Order upwinding finite-difference-based Adaptive time-stepping
Post-Processing	Residence time distribution
Output	Intermediary mesh format Labeled compartments in Paraview format Concentrations from CM simulations in both Paraview and simulation package format
Performance	Multi-threading Caching of intermediary results to speed up subsequent runs

User Interface

The OpenCCM Python package can be used via text-based configuration files centred around the command line interface (CLI), where each simulation run/project is self-contained in a project directory. In addition to the OpenCCM configuration files, the required contents include flow information from one of two open-source simulation packages: OpenCMP or OpenFOAM. For OpenCMP three files are required:

- 1) The OpenCMP config file,
- 2) The mesh on which the simulation was run, and
- 3) The .sol solution file containing the velocity profile to create the compartmental model.

For OpenFOAM, two sub-directories are required:

- 1) The constant/ directory, which contains the mesh information in ASCII format,

- 2) A directory containing the simulation results, which will be used to create the compartmental model saved in ASCII format.

The path to the solution directory is specified in the OpenCCM configuration file and the constant/ directory is assumed to be in the same parent folder. OpenCCM will create several output directories: log/, which contains detailed debugging information (if enabled); cache/, which contains intermediary files; and output_ccm/, which contains both simulation results of the compartmental model (in various user-specified formats) and ParaView files for visualization.

A sample config file, CONFIG, which outlines the available parameters, is included in the main directory. An excerpt of it, showing the compartmental modeling parameters, is shown below. Square brackets indicate parameters with default values along with those values indicated inside the brackets.

```
[COMPARTMENT MODELLING]
# Whether to use PFRs-in-series or a CSTR to model each compartment.
model                = PFR
# Volumetric flow through a single facet
# below which the flow is considered 0.
flow_threshold_facet = [1e-15]
# Volumetric flow threshold through a surface
# below which the flow is considered 0.
flow_threshold       = [1e-15]
# Maximum allowable difference (in % of compartment volume)
# between connections to merge them into one location.
dist_threshold       = [5 / 100]
# Absolute tolerances for checking that mass is conserved
# after the flow optimization is performed.
atol_opt             = [1e-2]
```

Reaction Configuration File

The chemical reaction parser in OpenCCM reads and parses the reaction configuration files and can handle general reaction equations of the form,

$$aA + bB + [\dots] \rightarrow cC + dD + [\dots]$$

with associated numeric rate constants. It intentionally does not support the standard \rightleftharpoons symbol for reversible chemical reactions, so that each independent reaction has an explicitly defined rate constant. Therefore, a reversible reaction must be written as two independent forward reactions, each with its own rate constant. Each species *label* can contain letters and numbers, but cannot contain brackets or special characters, e.g. "(", ")", "+", "-", "^", etc. Kinetic rate constants must be expressed as positive real numbers in standard or scientific notation. Additionally, each reaction/rate pair must have a unique *identifier* (i.e., R1, R2). For example, the reversible reaction,



with $k_f = 5e-2$ and $k_r = 2$ (chosen to be dimensionless for example). A configuration file for this reversible reaction may then be:

```
[REACTIONS]
R1: 2NaCl + CaCO3 -> Na2CO3 + CaCl2
R2: Na2CO3 + CaCl2 -> 2NaCl + CaCO3

[RATES]
R1: 5e-2
```

R2: 2

where **R1** and **R2** are the reaction *identifiers* for the forward and reverse reactions, respectively. An example reaction config file, `CONFIG_REACTIONS`, is provided in the main package directory.

Examples of Usage

Several examples are provided in the OpenCCM documentation, which demonstrates the using of both OpenCMP and OpenFOAM simulation-based flow information for compartmentalization. One example, located in `examples/OpenCMP/pipe_with_recirc_2d/`, uses the geometry from (Vasile et al., 2024) and shows how to execute the needed CFD simulation for flow information (both using OpenCMP and OpenFOAM), create/visualize the compartmental model results, and compare the predicted RTD to the reference result directly from CFD simulation.

For this illustrative example, the steady-state hydrodynamic flow-profile is obtained by running the OpenCMP simulation through the `run_OpenCMP.py` script in the folder. The resulting flow profile was opened in ParaView and a line integral convolution visualization of the velocity field is shown below, coloured by velocity magnitude.

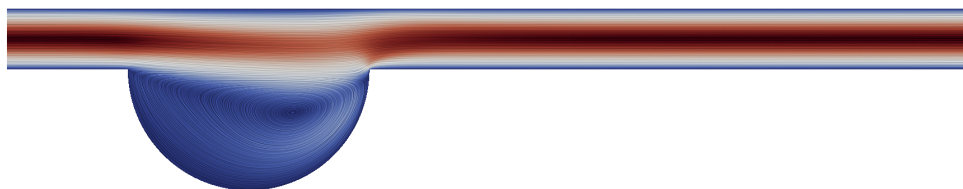


Figure 1: Visualization of hydrodynamics from CFD simulation with line integral convolutions indicating local flow direction and color corresponding to velocity magnitude.

The underlying velocity field data is then processed using OpenCCM to generate a network of compartments by executing the `run_compartment.py` script. The figure below shows each element of the original mesh coloured according to the compartment to which it belongs.

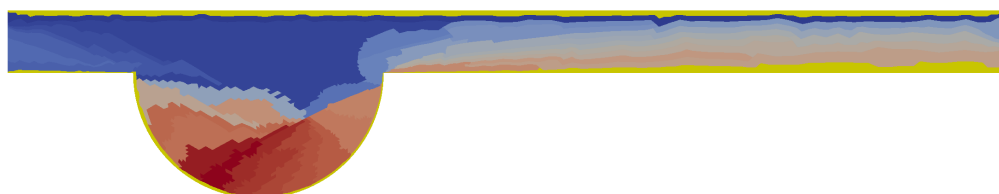


Figure 2: Visualization of flow-informed compartmentalization with coloring corresponding to compartment number.

That network of compartments is further processed, as each compartment is represented by a series of plug-flow reactors (PFRs). The resulting network (graph) of PFRs is shown in the figure below. Nodes are the centres of the PFRs and edges are connections (flows) between PFRs.

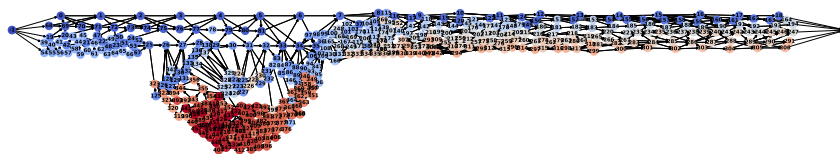


Figure 3: Undirectly graph of the compartment network resulting from both (i) flow-information compartmentalization and (ii) the use of spatially-varying compartment approximations (PFRs).

RTD Curves

The residence time distribution (RTD) curve from both the CFD and compartmental model (CM) simulations is computed using the script in the supplementary material of (Vasile et al., 2024).

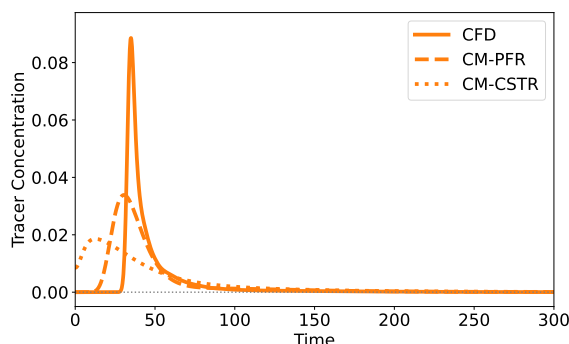


Figure 4: Residence time distribution curves for CFD and CM simulations.

Reactions

Finally, to demonstrate how to simulate chemical reaction, an example is included using the reversible reaction system from above:



with $k_f = 5e - 2$ and $k_r = 2$ as the forward and backward rate constants (chosen to be dimensionless for example). All species have an initial dimensionless concentration of 0 and have inlet boundary dimensionless concentration values of $[\text{NaCl}] = [\text{CaCO}_3] = 1$ and $[\text{Na}_2\text{CO}_3] = [\text{CaCl}_2] = 0$. The equations and conditions have already been specified, and the simulation can be run by using the `run_compartment_w_rxn.py` script. Note that when this script is run multiple times, computational times following the first will be much shorter in that the compartmental model is pre-computed (stored within the project directory).

To analyze the results, the equilibrium values for this reversible system are calculated as follows:

$$k_f[\text{NaCl}]^2[\text{CaCO}_3] = k_r[\text{Na}_2\text{CO}_3][\text{CaCl}_2]$$

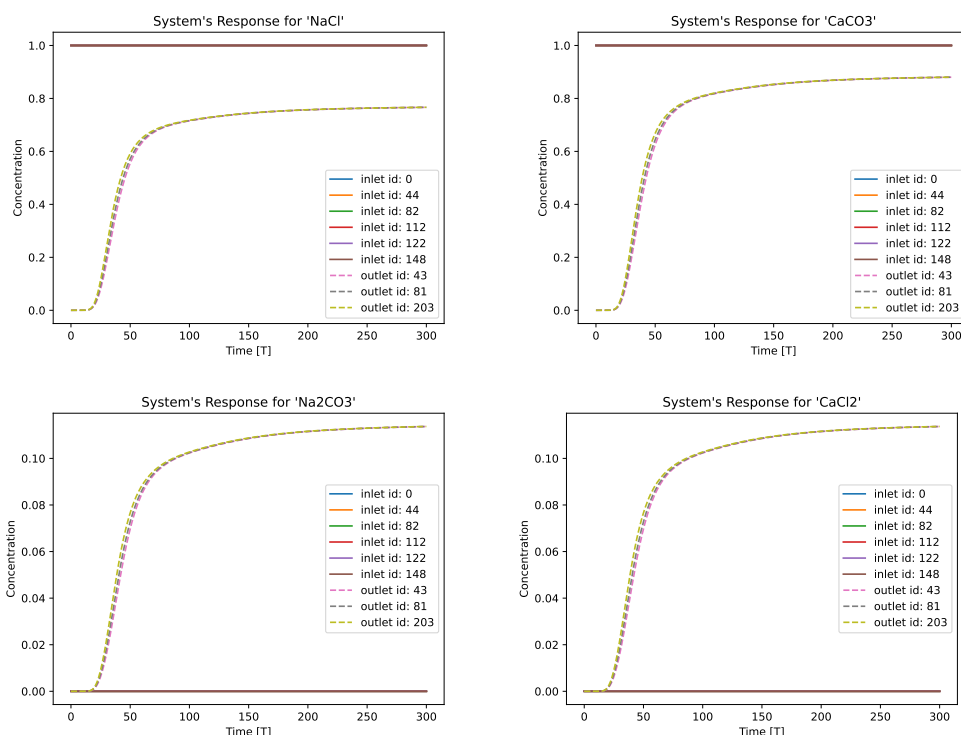
$$\frac{k_f}{k_r} = \frac{[\text{Na}_2\text{CO}_3][\text{CaCl}_2]}{[\text{NaCl}]^2[\text{CaCO}_3]}$$

$$\frac{5 \times 10^{-2}}{2} = \frac{(x)(x)}{(1 - 2x)^2(1 - x)}$$

$$x \approx 0.1147$$

where x is the change in CaCO_3 , in dimensionless units.

The expected equilibrium concentrations for the four species are: $[\text{NaCl}] = 0.7706$, $[\text{CaCO}_3] = 0.8853$, $[\text{Na}_2\text{CO}_3] = 0.1147$, and $[\text{CaCl}_2] = 0.1147$. Based on the figures below and from direct inspection of the CM simulation results, correct steady-state values are obtained at the reactor outlet.



To output the ParaView visualizations, change `output_VTK` to `True` in the `CONFIG` file and re-run the simulation.

Acknowledgements

This research was supported by the Natural Sciences and Engineering Research Council of Canada (NSERC) and the Digital Research Alliance of Canada.

References

- Ayachit, U. (2015). *The paraview guide: A parallel visualization application*. Kitware, Inc.
- Benner, P., Schilders, W., Grivet-Talocia, S., Quarteroni, A., Rozza, G., & Miguel Silveira, L. (Eds.). (2020). *Model Order Reduction: Volume 3 Applications*. De Gruyter. <https://doi.org/10.1515/9783110499001>
- Chinesta, F., Huerta, A., Rozza, G., & Willcox, K. (2017). Model reduction methods. In *Encyclopedia of computational mechanics second edition* (pp. 1–36). John Wiley & Sons, Ltd. <https://doi.org/10.1002/9781119176817.ecm2110>
- Goodwin, D. G., Moffat, H. K., Schoegl, I., Speth, R. L., & Weber, B. W. (2023). *Cantera: An object-oriented software toolkit for chemical kinetics, thermodynamics, and transport processes*. <https://www.cantera.org>. <https://doi.org/10.5281/zenodo.8137090>

- Greenshields, C. (2024). *OpenFOAM v12 user guide*. The OpenFOAM Foundation. <https://doc.cfd.direct/openfoam/user-guide-v12>
- Jourdan, N., Neveux, T., Potier, O., Kanniche, M., Wicks, J., Nopens, I., Rehman, U., & Le Moullec, Y. (2019). Compartmental modelling in chemical engineering: A critical review. *Chemical Engineering Science*, 210, 115196. <https://doi.org/10.1016/j.ces.2019.115196>
- Monte, E. J., Vasile, A. A., Lowman, J., & Abukhdeir, N. M. (2022). OpenCMP: An Open-Source Computational Multiphysics Package. *Journal of Open Source Software*, 7(73), 3742. <https://doi.org/10.21105/joss.03742>
- Quintessa. (n.d.). *Compartment modelling software*. Quintessa; Online. Retrieved January 3, 2024, from <https://www.quintessa.org/software/AMBER>
- Vasile, A. A., Aucoin, M. G., Budman, H., & Abukhdeir, N. M. (2024). A flow alignment-informed method for compartmental modelling. *Computers & Chemical Engineering*, 185, 108650. <https://doi.org/10.1016/j.compchemeng.2024.108650>