One-dimensional turbulence (ODT): computationally efficient modeling and simulation of turbulent flows

Victoria B. Stephens, David O. Lignell*

Chemical Engineering Department, Brigham Young University, Provo, UT 84602, USA

Abstract

Write this last. About 100 words.

Keywords: turbulence, reacting flows, one-dimensional turbulence

Code Metadata

Nr.	Code metadata description	Please fill in this column
C1	Current code version	1.0
C2	Permanent link to code/repository used for this code version	*github.com/BYUignite/ODT*
C3	Code Ocean compute capsule	N/A
C4	Legal Code License	MIT
C5	Code versioning system used	Git
C6	Software code languages, tools, and services used	C++, Python 3.x, Yaml,
C7	Compilation requirements, operating environments & dependencies	CMake 3.12+, Cantera, Git, Doxygen (optional)
C8	If available Link to developer documentation/manual	N/A
C9	Support email for questions	davidlignellbyu.edu

Table 1: Code metadata (mandatory)

1. Motivation and significance

Turbulent flows characterize the vast majority of fluid flows in practical engineering applications, and simulations of turbulent flows provide re-

*Corresponding author.

Email address: `davidlignell@byu.edu` (David O. Lignell)

searchers with valuable insights into complex systems, particularly reacting turbulent flows such as combustion processes. Turbulence is a complex phenomenon that affects the full range of a flow's length and time scales. As a result, resolving the entire flow field by numerically solving the Navier-Stokes equations of fluid flow, as is done in direct numerical simulations (DNS), requires substantial computational resources. DNS is a powerful research tool, but its high computational cost makes it intractable for simulating most practical engineering flows. In order to achieve numerical solutions to practical flow problems, researchers can use alternative frameworks that model turbulence rather than resolving it directly.

Large-eddy simulations (LES) address the problem of wide-ranging length and time scales by combining direct resolution of grid-scale quantities, as in DNS, with subgrid modeling of smaller turbulence structures. The more complex the flow, the more modeling is required; for example, a jet flame simulation might require subgrid modeling for the combustion chemistry, radiative heat transfer, or soot chemistry in addition to turbulence structures, all of which form a tightly coupled system in which each model interacts heavily with the others. While subgrid modeling makes LES more computationally affordable than DNS, it can introduce empiricism into simulations, which can lead to inaccurate results. Additionally, unresolved quantities are often parameterized in state space with empirical relationships or assumed distributions that lack universal applicability. LES is a valuable simulation tool, but its approach to turbulence modeling can introduce unwanted empiricism and make errors difficult to isolate and quantify.

The one-dimensional turbulence model (ODT) functionally reverses the LES approach, modeling large-scale turbulent advection and directly resolving small-scale flow structures, simulating the full range of length and time scales in a single dimension. Because large-scale structures are much easier to study and model than small-scale structures, ODT mitigates or sidesteps many of the subgrid modeling issues that complicate LES. Previous studies show that ODT can attain accuracy comparable to DNS at a fraction of the computational cost [1, 2], making it an attractive tool for simulating turbulent flows. Because the model is one-dimensional, it is limited to homogeneous or boundary layer flows such as jets, wakes, and mixing layers; such flows, however, are extremely common in both nature and turbulence research. ODT's computational efficiency and resolution of a full range of scales make it a valuable tool that complements experimental studies and other simulation tools like DNS and LES.

2. Software description

2.1. Model description

The ODT model is described in detail in the literature [3, 4, 5, 6, 7]; only a brief explanation will be given here. In ODT, turbulent advection is modeled with stochastic processes called eddy events, which punctuate the solution of unsteady, one-dimensional transport equations for mass, momentum, and enthalpy. The ODT code uses a Lagrangian finite-volume formulation for diffusive advancement that includes adaptive mesh refinement [7]. In this approach, mass remains constant inside each grid cell while cell volumes increase or decrease according to dilation. Because the ODT model is one-dimensional, it is limited to homogeneous or boundary-layer flows, such as jets, wakes, and mixing layers; these types of flows, however, are common in nature and central to turbulence research.

Eddy events occur as a Poisson process in accordance with their eddy rates, where a given eddy event of size l and location x_0 has an eddy timescale t and an associated eddy rate $1/t$. Three user-defined ODT parameters control the eddy event process: the eddy rate parameter C scales the rate of occurrence of the eddies; the viscous penalty parameter Z suppresses small eddies; and the large eddy suppression parameter β constrains eddies such that they do not reach over the elapsed simulation time.

Eddy events modify domain variables using triplet maps, as illustrated for a cylindrical domain in Figure 1. For a region of eddy size l, the domain is copied to create three map images; the three images are then placed back to back with the middle image inverted to maintain continuity, and the composite is reapplied to the domain. This process applies to all transported variables on the domain. Applied properly, the triplet map increases scalar gradients and decreases length scales consistent with the application of turbulent eddies in real flows, conserves all quantities and their statistical moments, and maintains continuity in property profiles. Subsequent eddies in the same region will result in a cascade of scales, and eddy rates depend on eddy size and the local kinetic energy such that they follow turbulent cascade scaling laws.

2.2. Software Architecture

The ODT code is a relatively self-contained C++ package. The system of nonlinear ODEs is solved using CVODE [8] and user input files are processed with YAML [9], both of which are installed locally during the ODT build process. For reacting flow cases, chemical kinetics and transport are handled by Cantera [10], which must be previously installed by the user.

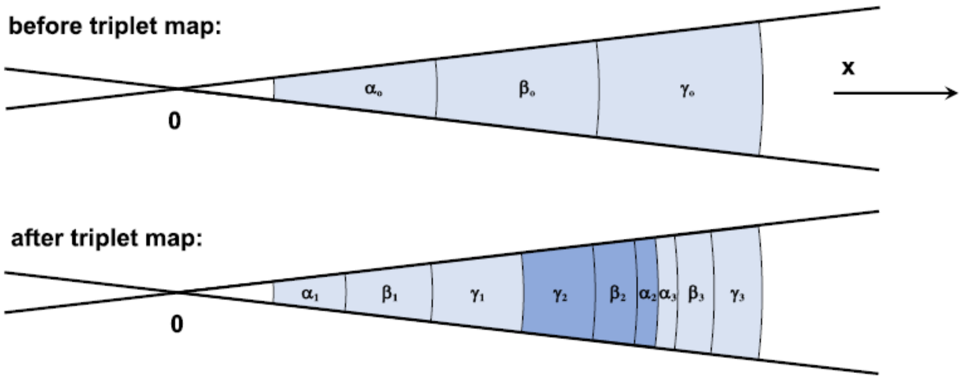

Figure 1: Schematic diagram of a cylindrical triplet map, adapted from [6]. Before the triplet map, the domain contains three grid cells of equal volume, while after the triplet map has been applied, the domain contains nine cells. The nine final cells are labeled according to the cells from which they originated and shaded to indicate that three map images were combined to create the final composite.

3. Example Cases

4. Impact

Questions to answer in this section (from SoftwareX template)

1. How can new research questions be pursued with this software?
    - possibility of parametric studies (much harder with DNS/LES/RANS)
    - study of late-flame soot and radiation interactions, soot emissions as smoke
    - comparative radiation model studies?
2. How does the software improve pursuit of existing research questions?
    - late-flame behavior becomes easier to study
    - validation of LES subgrid models
    - soot stuff, especially late in the flame (because soot moves slowly compared to gas species and therefore short simulation times like in DNS aren't enough to study it effectively)
3. How does the software change the daily practice of its users?
    - cases take hours or days rather than weeks using supercomputer resources
    - test cases can be run on local computers (unlike something like DNS) and as background tasks without disrupting other tasks

- ODT as a tool complements other approaches, can cover blind spots and be used in validation

4. How widespread is the software? Who uses it? (Within and outside of intended research area and/or group.)
    - BYU group
    - JCH at Sandia
    - Chalmers group in Sweden (Marco Fistler, etc.)
    - German university group (Heiko Schmidt, Juan Media, Marten Klein, etc.)
    - TO DO: find other groups who have used or currently use ODT
5. How is the software used in commercial settings (if any)? Has it led to creation of spin-off companies?
    - No commercial use (I think).

5. Conclusion

Write this part next to last

6. Conflict of Interest

We wish to confirm that there are no known conflicts of interest associated with this publication and there has been no significant financial support for this work that could have influenced its outcome.

Acknowledgements

This work is supported in part by the National Science Foundation under Grant No. CBET-1403403.

References

[1] D. O. Lignell, G. C. Fredline, A. D. Lewis, Comparison of one-dimensional turbulence and direct numerical simulations of soot formation and transport in a nonpremixed ethylene jet flame 35 (2) (2015) 1199–1206. doi:10.1016/j.proci.2014.05.046.

[2] A. W. Abboud, C. Schulz, T. Saad, S. T. Smith, D. D. Harris, D. O. Lignell, A numerical comparison of precipitating turbulent flows between large-eddy simulation and one-dimensional turbulence 61 (10) (2015) 3185–3197. doi:10.1002/aic.14870.

[3] A. R. Kerstein, One-dimensional turbulence: model formulation and application to homogeneous turbulence, shear flows, and buoyant stratified flows 392 (1999) 277–334. doi:10.1017/S0022112099005376.

[4] A. R. Kerstein, W. T. Ashurst, S. Wunsch, V. Nilsen, One-dimensional turbulence: vector formulation and application to free shear flows 447 (2001) 85–109. doi:10.1017/S0022112001005778.

[5] W. T. Ashurst, A. R. Kerstein, One-dimensional turbulence: Variable-density formulation and application to mixing layers 17 (2). doi:10.1063/1.1847413.

[6] D. O. Lignell, V. B. Lansinger, J. Medina, M. Klein, A. R. Kerstein, H. Schmidt, M. Fistler, M. Oevermann, One-dimensional turbulence modeling for cylindrical and spherical flows: model formulation and application 32 (4) (2018) 495–520. doi:10.1007/s00162-018-0465-1.

[7] D. O. Lignell, A. R. Kerstein, G. Sun, E. I. Monson, Mesh adaption for efficient multiscale implementation of one-dimensional turbulence 27 (3-4) (2013) 273–295. doi:10.1007/s00162-012-0267-9.

[8] A. C. Hindmarsh, R. Serban, D. R. Reynolds, CVODE, https://computing.llnl.gov/sites/default/files/public/cv_guide.pdf (2020).
URL `https://computing.llnl.gov/projects/sundials/cvode`

[9] J. Beder, yaml-cpp v0.6.3 (2008).
URL `https://github.com/jbeder/yaml-cpp/`

[10] D. G. Goodwin, R. L. Speth, H. K. Moffat, B. W. Weber, Cantera (2018). doi:10.5281/zenodo.1174508.
URL `https://cantera.org/`

Current executable software version

Ancillary data table required for sub version of the executable software: (x.1, x.2 etc.) kindly replace examples in right column with the correct information about your executables, and leave the left column as it is.

Nr.	**(Executable) software metadata description**	**Please fill in this column**
S1	Current software version	2.1
S2	Permanent link to executables of this version	For example: $https://github.com/combogenomics/DuctApe/releases/tag/DuctApe-0.16.4$
S3	Legal Software License	MIT
S4	Computing platforms/Operating Systems	Linux, OS X, Microsoft Windows
S5	Installation requirements & dependencies	CMake 3.12+, Cantera, Git, Doxygen (optional)
S6	If available, link to user manual - if formally published include a reference to the publication in the reference list	For example: $http://mozart.github.io/documentation/$
S7	Support email for questions	davidlignell@byu.edu

Table 2: Software metadata (optional)