pyGEMS - 1D:
A 1D Python Port of the General Mesh and Equations Solver

Christopher R. Wentland, Ashish S. Nair, Cheng Huang, Karthik Duraisamy

July 29, 2020

Contents

1 Introduction

This code is intended to be a sort of low-fidelity Python port of the General Mesh and Equations Solver (GEMS) [1], originally written by Guoping Xia at Purdue University, and since expanded upon over the years (decades!) by researchers from Purdue University, the University of Michigan, Ann Arbor, and the Air Force Research Laboratory. Although this port is "low-fidelity" in the sense that the code lacks much of the functionality of the original GEMS solver (e.g. robustness constrols, a wide variety of flow physics and reaction models, complex numerical solver algorithms), the hope is that this might serve as a useful code for reduced-order model (ROM) practitioners to develop and test new methods on simple 1D reacting flow problems, ideally with a low barrier to implementation and application. Additional functionality may be added over time to equip this code with tools found in GEMS.

As a disclaimer, none of the developers on this project are by any means proficient software developers, as we're all computational physics and data-driven modeling researchers, and we're continually learning new tricks and tools for making code more flexible, portable, and robust. There are probably plenty of bugs and there is always a need for better error catching! If you have a suggestion for improving this code, either to fix something or make it more flexible for a certain type of ROM formulation, please create a new issue at the GitHub repo or email Chris Wentland at `chriswen[at]umich[dot]edu`. Similarly, if you'd like to contribute something to the code, please let Chris know!

2 Physics

2.1 Governing Equations

2.2 Equations of State

2.3 Reaction Model

3 Numerics

3.1 Spatial Discretization

3.2 Temporal Discretization

3.3 Boundary Conditions

4 Projection-based Reduced-Order Modeling

4.1 Galerkin Projection

4.2 Least-Squares Petrov-Galerkin Projection

4.3 Linear Subspace Methods

4.3.1 Proper Orthogonal Decomposition

4.4 Nonlinear Manifold Methods via Convolutional Autoencoders

4.4.1 An Encoder Projection Approximation

5 Inputs

User inputs are provided by text files, and are ingested by the code by interpretting the text as regular expressions. Each line is searched for the character "="; all text to the left of that character is treated as the variable name, and all text to the right is treated as the variable value (after stripping leading and trailing white space). This takes the general form `varName = varValue`, though as much white space may be added before and after `varName` and `varValue` to suit whatever the user finds aesthetically pleasing. All variable name inputs are, as of the writing of this section, case-sensitive. Variable values follow general Python formatting: strings should be enclosed by double quotes, integers and floats should be contain only numerals and at most one period, boolean values should be either `False` or `True` (case-sensitive), and lists should be enclosed in brackets (e.g. `[value1, value2, value3]`. Similarly, lists of lists should be enclosed in nested brackets (e.g. `[[value11, value12], [value21, value22]]`.

There are three user input file which is strictly required for running FOM cases. The first is `solverParams.inp`, and it must be named this way. The other two, the mesh parameters file and the chemistry parameters file, may be arbitrarily named and specified in `solverParams.inp`. If not initializing the fluid state from a NumPy binary or restart file, an initial conditions parameter file must additionally be specified in `solverParams.inp`. One additional file is strictly required for running ROM cases, and must be named `romParams.inp`.

All the possible entries for these input files are detailed in the sections below. Whether they are required or optional, as well as any default settings, is noted where applicable. If a variable's requirement or default value is listed "Dep," this indicates that this is dependent on other settings, and is explained further in the "Details" column.

5.1 Solver and I/O Parameters

Table 1:

Variable Name	Type	Req?	Default	Details
gasFile	`str`	Yes	N/A	Path to chemistry parameters file.
meshFile	`str`	Yes	N/A	Path to mesh parameters file.
initFile	`str`	No	N/A	Path to initial condition NumPy binary.
icParamsFile	`str`	Dep	N/A	Path to initial conditions parameters file. If initFile is not provided and initFromRestart == False, this must be set.

Table 2:

Variable Name	Type	Req?	Default	Details
dt	`float`	Yes	N/A	Physical time step.
numSteps	`int`	Yes	N/A	Number of physical time steps to compute.
timeScheme	`str`	Yes	N/A	Time integration scheme. Currently accepts "rk".
timeOrder	`int`	Yes	N/A	Time integration scheme order of accuracy. "rk" accepts 2, 3, or 4.
numSubIters	`int`	No	Dep	Number of time step sub-iterations. For implicit schemes, the default value is 20. For explicit schemes this is determined by the scheme and order of accuracy.
resTol	`float`	No	1e-10	Lower threshold for ℓ^2-norm of fully-discrete residual for terminating implicit solve.

Table 3:

Variable Name	Type	Req?	Default	Details
spaceScheme	`str`	No	"roe"	Numerical flux scheme. Currently accepts "roe".
spaceOrder	`int`	No	1	Spatial scheme order of accuracy. Currently accepts 1.
viscScheme	`int`	No	0	Viscosity model. Currently accepts 0 for inviscid and 1 for Sutherland model.

Table 4:

Variable Name	Type	Req?	Default	Details
boundType_inlet	`str`	Yes	N/A	Inlet boundary type. Currently accepts "characteristic".
press_inlet	`float`	Dep	N/A	Fixed inlet pressure, if required by inlet boundary condition.
vel_inlet	`float`	Dep	N/A	Fixed inlet velocity, if required by inlet boundary condition.
temp_inlet	`float`	Dep	N/A	Fixed inlet temperature, if required by inlet boundary condition.
massFrac_inlet	`list`, `float`	Dep	N/A	Fixed inlet mass fraction, if required by inlet boundary condition. List should have as many entries as species in chemistry file.

Table 5:

Variable Name	Type	Req?	Default	Details
boundType_outlet	`str`	Yes	N/A	Outlet boundary type. Currently accepts "subsonic".
press_outlet	`float`	Dep	N/A	Fixed outlet pressure, if required by outlet boundary condition.
vel_outlet	`float`	Dep	N/A	Fixed outlet velocity, if required by outlet boundary condition .
temp_outlet	`float`	Dep	N/A	Fixed outlet temperature, if required by outlet boundary condition.
massFrac_outlet	`list, float`	Dep	N/A	Fixed outlet mass fraction, if required by outlet boundary condition. List should have as many entries as species in chemistry file.
pertType_outlet	`str`	No	N/A	Perturbation method for boundary condition. "subsonic" accepts "pressure".
pertPerc_outlet	`float`	Dep	N/A	percentage perturbation about fixed outlet quantity, in decimal format (e.g. enter 0.1 for a 10% perturbation)
pertFreq_outlet	`list, float`	Dep	N/A	List of superimposed frequencies for outlet perturbation (Hertz).

Table 6:

Variable Name	Type	Req?	Default	Details
velAdd	`float`	No	0.0	Bulk velocity to add to initial flow field (m/s).

Table 7:

Variable Name	Type	Req?	Default	Details
saveRestarts	`bool`	No	`False`	Whether to save restart files.
restartInterval	`int`	No	100	Physical time step interval for saving restart files.
numRestarts	`int`	No	20	Maximum number of restart files to retain.
initFromRestart	`bool`	No	`False`	Whether to initialize solution from the most recent restart file.

Table 8:

Variable Name	Type	Req?	Default	Details
outInterval	`int`	No	1	Physical time step interval for saving field data.
primOut	`bool`	No	`True`	Whether to write primitive variable field data to disk.
consOut	`bool`	No	`False`	Whether to write conservative variable field data to disk.
RHSOut	`bool`	No	`False`	Whether to write RHS function field data to disk.

Table 9:

Variable Name	Type	Req?	Default	Details
visType	`str`	No	"field"	Visualization method, either probe plot over time ("probe") or spatial field snapshots ("field").
visVar	`str`	No	"pressure"	Field variable to plot. Accepts "pressure", "velocity", "temperature", "species", "density", "momentum", or "energy"
visSave	`bool`	No	`False`	Whether to save visualization plot(s) to disk.
visInterval	`int`	No	1	Number of physical time steps between visualization plotting. Warning: small visualization intervals make the code extremely slow.
probeLoc	`float`	Dep	N/A	Spatial location for probe. Must be set if visType == True.

Table 10:

Variable Name	Type	Req?	Default	Details
calcROM	`bool`	No	`False`	Whether to solve a ROM.

5.2 Grid Generation

Table 11:

Variable Name	Type	Req?	Default	Details
xL	`float`	Yes	N/A	Coordinate of left-most cell face.
xR	`float`	Yes	N/A	Coordinate of right-most cell face.
numCells	`float`	Yes	N/A	Number of finite-volume cells.

5.3 Chemistry Parameters

Table 12:

Variable Name	Type	Req?	Default	Details
numSpecies	`int`	Yes	N/A	Total number of species included in this case.
molWeights	`list`, `float`	Yes	N/A	Molecular weights of each species (g/mol).
enthRef	`list`, `float`	Yes	N/A	Reference enthalpy of each species (J/K).
tempRef	`float`	Yes	N/A	Reference temperature (K).
Cp	`list`, `float`	Yes	N/A	Heat capacity at constant pressure of each species (J/K).
Pr	`list`, `float`	Yes	N/A	Prandtl number for each species.
Sc	`list`, `float`	Yes	N/A	Schmidt number for each species.
muRef	`list`, `float`	Yes	N/A	Reference dynamic viscosity for each species, for the Sutherland model.

Table 13:

Variable Name	Type	Req?	Default	Details
nu	`list`, `float`	Yes	N/A	?????
nuArr	`list`, `float`	Yes	N/A	Global reaction Arrhenius molar concentration exponents.
actEnergy	`float`	Yes	N/A	Global reaction Arrhenius activation energy, divided by the universal gas constant.
preExpFact	`float`	Yes	N/A	Global reaction Arrhenius pre-exponential factor.

5.4 Initial Condition Parameters

Table 14:

Variable Name	Type	Req?	Default	Details
xSplit	`float`	Yes	N/A	Spatial location at which to split domain into left and right "chambers"
pressLeft	`float`	Yes	N/A	Pressure in left chamber.
velLeft	`float`	Yes	N/A	Velocity in left chamber.
tempLeft	`float`	Yes	N/A	Temperature in left chamber.
massFracLeft	`list`, `float`	Yes	N/A	Species mass fractions in left chamber.
pressRight	`float`	Yes	N/A	Pressure in right chamber.
velRight	`float`	Yes	N/A	Velocity in right chamber.
tempRight	`float`	Yes	N/A	Temperature in right chamber.
massFracRight	`list`, `float`	Yes	N/A	Species mass fractions in right chamber.

5.5 ROM Parameters

5.6 Restart Files

6 Utilities

6.1 POD Basis Generation

6.2 Projection Error Calculations

References

[1] Li, D., Xia, G., Sankaran, V., and Merkle, C. L., "Computational Framework for Complex Fluid Physics Applications," *Computational Fluid Dynamics 2004*, edited by C. Groth and D. W. Zingg, Springer Berlin Heidelberg, 2006, pp. 619–624.