umDetailedChemFoam: A low-Mach, detailed chemistry solver based on open source software OpenFOAM and Cantera

Developers

- Reaction model & documentation:
  Yihao Tang
  yhtang@umich.edu
- Low-Mach algorithm & turbulence model:
  Malik Hassanaly
  malik.hassanaly@gmail.com
- Supervision:
  Prof. Venkat Raman
  ramanvr@umich.edu

Advanced Propulsion Concept Laboratory,
Department of Aerospace Engineering, University of Michigan

Methodology

a) This solver applies a low-Mach algorithm similar to that implemented in the *pimpleFoam* solver in OpenFOAM-v2.1.x official release, while we made critical modifications to ensure mass conservation for combustion-related problem.

b) For turbulence modeling, the method is based the LES turbulence model *dynSmagorinsky* in OpenFOAM-2.1.x release, while we applied critical optimizations in the choice of numerical algorithms and schemes to ensure turbulent kinetic energy conservations.

- For details regarding a) and b), the users are strongly encouraged to read the documentations of our solver *umFlameletFoamV2.1*.

c) For reaction model, this solver uses detailed chemistry. All thermochemical properties and reaction kinetics are calculated with Cantera-2.4.0 release. For details please read the presentation slide.

Installation

- Compile Cantera-2.4.0
  This version is needed as it provides an interface of stiff ode solver that's compatible with the implementation of our code (see header files under cantera_src/include/ext).
  First, create source directory and download the package from Github. Note that, for some cases, we found a direct download through the webpage may cause compilation errors, and therefore we suggest to do this step via terminal command.
  ```
  git clone https://github.com/Cantera/cantera -b 2.4
  mv cantera your_cantera_src_dir
  ```
  Then, setup the compilation configuration file, please read the example configuration file we provided under umDetailedChemFOAM_tut_pkg/src/cantera.conf for reference
  ```
  cd your_cantera_src_dir
  vim cantera.conf
  prefix = 'your_cantera_build_dir_that_is_different_from_your_cantera_src_dir'
  ```
  Last, keep your finger crossed and compile

```
[sudo] scons build
[sudo] scons test
[sudo] scons install
```

- Compile *umDetailedChemFoam* solver
  First, following the instructions in the *umFlameletFoam* documentation (chapter 5), replace the functionality of *adjustPhi* in the original OpenFoam release, and compile the library *dynCompSansThermoSF.*
  Then, copy the source code of the solver to the following directory
  ```
  cp  -r umDetailedChemFOAM_tut_pkg/src/solver
  your_OF_build_dir/applications/solvers/combustion/umDetailedChemFoam
  ```
  Setup the compilation options and compile, make sure the the location of cantera libraries are pointing to your Cantera build directory.
  ```
  cd your_OF_build_dir/applications/solvers/combustion/umDetailedChemFoam
  vim make/options
  EXE_INC = \
  -I/your_cantera_build_dir/include \
  -I/your_cantera_build_dir/include/cantera/ext
  EXE_LIBS = \
  -L/your_cantera_build_dir/lib -lcantera
  wmake
  ```

How to use the solver

An example test case is provided here that solves a 2D planar laminar couterflow diffusion flame. The simulation setups, along with results compared with laminar 1-D flame solved using FlameMaster and Cantera can be found in the presentation slide.

- For a quick test run, start simulation run from a developed flow field at time instance 0.1802, by
  ```
  source your_cantera_build_dir/bin/setup_cantera
  source your_OF_build_dir/etc/bashrc
  cd umDetailedChemFOAM_tut_pkg/example_case
  umDetaileChemFoam
  ```
  It is also recommended to do a parallel test run with the compiled solver.
- To start the test case from scratch from time 0, here are a few useful tips:
  a) The folder example_case/0 contains all necessary fields that umDetailedChemFoam needs to read in to initiate the test case simulation. Notice that: for a turbulent case, a few more fields such as muSgs, k will be needed; Zmix is needed here just as a dummy field to construct the object of turbulence model that's required during the solver initialization; the mass fractions of few key species needs to be read-in directly from the initial time folder, whereas the rest of the species will be automatically created by the solver. For more details, please read
  umDetailedChemFOAM_tut_pkg/src/solver/ createDetailedChemistry.H
  b) The file example_case/constant/inputParameters contains some important simulation options. Please read the comments in the file to see the meaning of each option.
  c) First, develop a cold flow with unity Lewis number. For this, set a uniform flow fields with Y_O2, Y_N2 being the composition of air, set uniform flow field of T with cold inflow temperature (300K here), and use the following simulation options as:
  ```
  vim example_case/constant/inputParameters
  ```

```
initialize      true;
preffDiff       false;
```

d) Then, after cold flow mixing has been developed. Initiate reaction by patching the domain with a high temperature near the expected reaction location, here being the diffusion layer. The patch field is done via the OpenFoam utility *setField*:

```
vim system/setFieldsDict
setFields -time your_cold_flow_time
```

Start the simulation but with species and enthalpy transport equation disabled:

```
vim system/controlDict
startTime      your_cold_flow_time;
solveYEqn      false;
solvehEqn      false;
umDetailedChemFoam
```

This step is to allow the low-Mach algorithm to correct the velocity field after the drastic change of density introduced by the patch of temperature field, which is critical for convergence. After the velocity field has been corrected, as the magnitude of "max(mag(U))" and "max(p)-min(p)" drops to a magnitude that is comparable to the former cold flow simulation, re-enable solving the species transport equation and enthalpy equation. Notice that these options can be changed on-the-fly during a simulation.

```
vim system/controlDict
solveYEqn      true;
solvehEqn      true;
```

It is suggested to first enable species transport equation, and freeze the enthalpy equation for a few more time steps. Otherwise the simulation may end up in Cantera error during gas.setStates_HP.

e) Sometimes, if the case appears to be very unstable, the user may want to use a more dissipative divergence schemes, as:

```
vim system/fvSchemes
 div(((phi+phi_0)|4),U)          Gauss upwind;
div(((phi+phi_0)|4),U_0)         Gauss upwind;
div((phi|2),Yi)                  Gauss upwind;
div((phi|2),h)                   Gauss upwind;
div((phi|2),h_0)                 Gauss upwind;
```

f) With all these above, hopefully the simulation will go well. After all flow fields have reached a statistical steady state, stop and restart the simulation with preferential diffusion enabled. Don't forget to toggle off the option of flow field initialization, and change the divergence schemes above back to the original settings in the example case.