The simulationDriver for cantera is being upgraded to the YAML way because it will allow surface reactions to be modified on the fly.

To convert to the YAML way, we have two things to convert:

1. The input files (currently in cti format for starting top of cti file and also at intermediate steps).
2. The functions from my simulationDriver.

The functions are actually easier to convert. However, we need to convert the input files to the YAML way first because the old cti way does not create the right reaction object type (in cantera python objects) for on-the-fly modification.

\*\*\*

First, let’s consider how the CTI file gets made so that we can switch it to the YAML way. Actually, we can just, for now, use the CTI to YAML converter at the very end of full cti file creation because we already know that will convert to a suitable YAML file.

The current flow basically occurs inside a single function, as seen inside example\_runfile.py, it is the function call: canteraSimulate.create\_cti\_and\_SimulatePFRorTPRwithCantera

That function takes a CTI top info file (like ceO2\_cti\_top\_info.cti ) which has the species names and phases, and it also takes an array of *reactions and parameters*.

It passes the information to another function, create\_cti\_and\_cantera\_phases, which creates the cti\_file using canteraKineticsParametersParser.create\_full\_cti and then goes on to create the phases using cantera directly.

Those phases are then passed back to canteraSimulate.create\_cti\_and\_SimulatePFRorTPRwithCantera which then also calls the simulation function.

So to interrupt that flow and introduce YAML while allowing backwards compatibility, there are a few obvious ways to do it.

1. Create a function create\_yaml\_and\_cantera\_phases, make that function then call a helper function called canteraKineticsParametersParser.create\_full\_yaml
   1. Make that function then simply convert the cti file.
   2. Note: The YAML way still has a file that can be divided into top and full. So it may be worth converting just the top into yaml with the cti converter and to then add the yaml way.
   3. It looks like I can probably take the cti reactions string and convert it directly the yaml way with the yaml converter rather than having to make the yaml. This also has a backwards compatibility advantage in that the cti to yaml converter will automatically convert “surface\_reaction” into “reaction”
      1. The following syntax seems to work (but see next point):
         1. import cantera
         2. cantera.cti2yaml.convert(text='surface\_reaction("Acetaldehyde + CeCation(S) => Acetaldehyde1-Ce(S)", stick(1, 0, 2000))', output\_name="Testing.yaml")
      2. However, it seems to be only appropriate, or at least most appropriate, for converting an entire cti file string to yaml. So evidently I will need to change “create\_full\_cti” to “create\_full\_yaml” essentially one line at a time, but it does not seem too hard.

There is going to need to be a full development directory for this upgrade, and it will need the following steps.

1. Execute the existing run.
2. Use the cantera cti to yaml converter to get the \*\*full\*\* file into yaml. Use the full yaml to do the simulation (instead of the cti).
   1. Will do this by making this function: “create\_full\_yaml” as hardcoded to point to the converted yaml file.
3. Then go back and fix create\_full\_yaml.

\*\*\*

1. – could not run the existing cti file with cantera 2.6.0. It seems it’s no longer compatible. I am getting an error that looks like this if I try to run example-runfile.py:

File "C:\Users\fvs\Documents\GitHub\PEUQSE\PEUQSE\simulationDriver\canteraSimulate.py", line 247, in simulatePFRorTPRwithCantera

gas\_rates.append(surf.get\_net\_production\_rates('gas'))

File "cantera/kinetics.pyx", line 978, in cantera.\_cantera.InterfaceKinetics.get\_net\_production\_rates

File "cantera/kinetics.pyx", line 946, in cantera.\_cantera.InterfaceKinetics.\_phase\_slice

File "cantera/kinetics.pyx", line 943, in cantera.\_cantera.InterfaceKinetics.phase\_index

KeyError: 'gas'

I was able to install an earlier version of cantera using: conda install --channel cantera cantera==2.5

Now I was able to run “example-runfile.py”