PhD Progress Slides 03.21.2025

J. Skyler Sampayan

Last Meeting

- Personal Matters
  - ChatGPT for CSU (full version)
- Research Progress
  - isatab serial test cases
  - DQE question

Agenda

- Personal Matters
  - N/A
- Research Progress
  - Reactor ODE + NN code review
  - DQE Question

Personal Matters

N/A

- N/A

Research Progress

Reviewed ReactorODE

- Reviewed the code structure.
- Reviewed a little bit of the physics.
- Now that I have a better understanding of isatab, was able to understand what you did.
- Physics Equations used (as you have written) is to solve energy balance for an ideal gas under constant pressure conditions.

$$\frac{dT}{dt} = -\frac{\sum \bar{h} \frac{(dw)}{(dt)}}{\rho c_p}$$

Energy Conservation

$$\frac{dY}{dt} = \frac{(dw)}{dt} + MW \frac{1}{\rho}$$

Species Conservation

reactor.cpp code structure

- **Reactor ODE class**:
  - **Constructor**: Define object parameters for gas.
  - **eval**():
    1. Get current gas properties.
    2. Solve energy and species conservation equation.
  - getState(): Return temperature and mass fractions.
- **fromxhat**: un-normalize and return “ptcl”.
- **toxhat**: normalize and return ”x”.
- **myfnn**: neural net that returns approximation of fnn(x).
- **mymix**: Get current properties, then mix enthalphies and mass fractions, then update gas properties and states.

```
class ReactorODEs : public FuncEval {
public:
```

```
void eval(double t, do
```

```
*/
void getState(double*
```

```
void fromxhat(double x[
```

```
void toxhat(double ptcl[],
```

```
void myfnn(int &nx, double
```

reactor.cpp code structure (cont'd)

- myfgh: function for isatab needs to work.
  1. Call from xhat.
  2. Create ReactorODE object called "ode"
  3. New Integrator object
     1. Intialize
     2. Set tolerances
     3. Integrate (solve RHS) using "eval" function in ReactorODE class
  4. Normalize
  5. Then (separately) perform inference on "x" using neural net.
  6. Then (assuming) error is calculated by: f = f – x – fnn
  7. The "if (need(1) = 1" is for isatab stuff.

```
fromxhat(x,ptcl,nx,rusr);

T[0] = ptcl[0];
for (int ii = 1; ii < nx; ii++){ Y[ii-1] = ptcl[ii];}



gas->setState_TPY(T[0], p, Y);

/* ———————— CREATE ODE RHS EVALUATOR ———————— */
ReactorODEs odes = ReactorODEs(sol);

double tnow = 0.0;

//double dt = 1e-4;

shared_ptr<Integrator> integrator(newIntegrator("CVODE"));

integrator->initialize(tnow, odes);

integrator->setTolerances(aTol, rTol);

integrator->integrate(dt);

solution = integrator->solution();

toxhat(solution,f,nx,rusr);

myfnn(nx, x, fnn);

for (int ii=0; ii<nx; ii++){f[ii] = f[ii] - x[ii] - fnn[ii];}
```

driver_samples.f90 code structure (main)

- Initialize c++ function interface and define functions.
- Allocate memory then initialize parameters for ODE function integration and isatab itself.
  - This includes particles and neural net weights and biases.
- Begin time step
  - Normalize particles
  - Call isatab that passes myfgh: returns the error (f)??
  - Call fnn to perform inference.
  - “x = x + fnn + f” where f is the error from isatab, fnn is the inferred solution, and x is previous solution.
  - Un-normalize and returns ”ptcl”
  - Passing solution “ptcl” through mymix

DQE Question

- Material derivative.
- Conservative vs Non-Conservative form.
- Derive energy equation.

Questions?

After Meeting Notes

1. Research cantera and utilize sensitivies analysis for getting better jacobian.
    1. Look at jacobian based on initial composition
    2. Integrator=cantera class (integrator), you need the ”eval” function.
2. Look ”CVODES” for integrating ODE with sensitivies.
3. Jacobian fnn, back propagation.