ESFR Pull Request - Status April 20, 2022

April 20, 2022

This document addresses all comments for the pull request, and gives my suggestion on how to proceed with the review.

1 Comments/Suggestions that were not implemented

Here I will outline some comments that I chose not to implement and I will provide the rebuttal to each.

1. Solution Initialization in another file for advection explicit periodic
    - I did not do this because to properly initialize for the curvilinear mesh, the function must be projected. Thus, to do this in the physics/initial_condition part of the code, we will have to have physics get either a dg or operators object. Note, that I found that `dealii` interpolate does not perform correctly on curvilinear meshes, which motivated me to hard code it in the first place. I think that adding this generality to the initial condition class should be its own stand alone PR.
2. The cleaning up of all parameters
    - This will be raised as an issue and to be done in another PR.
3. Move each if block for the evaluation of the ESFR c parameter to their own function
    - I left it as a series of if statements because each block is only a few lines.
4. Use flux_average in the added central and entropy conserving numerical fluxes. Also, to move them into the convective flux class.
    - An issue to be raised, and addressed in another PR is to change the convective numerical flux class to just one general convective flux, and all other fluxes can be recovered by changing the convective eigenvalue from physics, as discussed by Doug and I. This will recover all types currently implemented, and since the eigenvalues change by the pde this is the logical step. Thus, I didn't do those 2 changes because they will be changed in the next issue.
5. Use current_time from ODESolverBase instead of storing the time in DGBase

- As I stated on the PR, "the issue is that for DG to use current_time from ODESolverBase, then DG would need to store an ODESolver object. I agree that the spatial discretization shouldn't have time, but it most definitely shouldn't have an ODESolver object since the ODESolver is separate from the spatial discretization. Thus, in order to have the source term be time-dependent, which is needed for the unsteady problems, either DG needs to store the current_time or it needs the ODESolver object. I went with the former. If you have a suggestion on how to use ODESolverBases's current_time without DG needing info about the object I will gladly change it."

6. Rename the example manufactured solution.

    - I decided not to because I will open an issue for unsteady, time-dependent manufactured solution and use that as the base for the PR.

7. Derived class in Burgers' for the source term

    - No because via the issue/PR above, the unsteady manufactured solution source term will take care of it and we will then have to delete the added derived class.

8. Move compute conservation to Testbase.

    - No because an issue will be raised that conservation and energy computations should be done within the ODESolverBase on the fly.

Review of issues being raised by the PR for future PRs

1. Generalize Initial Condition to handle projections.
2. Clean up all parameters.
3. Implement a general convective numerical flux, where all current versions are derived through appropriate calculations of the eigenvalue.
4. Implement unsteady, time-dependent manufactured solution function.
5. Compute conservation and energy within ODESolverBase.

2 Overview of Changes Made

1. I made the indentation consistent throughout the files.
2. Template/restructure Operators.

    - The big one for this was, to make it array of arrays, I had to template the class. Since nstate isn't compile-time constant for DGBase, but is for DGBaseState, Operators now has OperatorsBase and OperatorsBaseState, with respective operators and functions depending on the state. I also added the

sum-factorization functions and test. Sum-factorization was not yet implemented throughout the code, so to review it, it's just the 2 base functions and the test in unit_tests/operator_tests.

3. The entropy conserving flux flag is done at the parameter level by requiring that the physics is Burgers. In the future this will be taken care of by issue # 3.

4. DG was cleaned up, especially with the mass matrix. Implicit strong fails by default. Auxiliary calls the DG assemble residual with a flag and assembles the appropriate residuals when the flag is true.

5. Curvilinear meshes have their own files with references in the mesh folder.

6. Explicit ode solver now has c Butcher Tableau values and sets current time appropriately.

7. Relevant names and citations added.

3 Moving Forward

Since this round took care of remarks on the unsteady tests, and then the allocation of DG with the loop. I would suggest reviewing the changes, then proceeding with suggestion 4 in the previous document. That is, review the DG strong form along with operators. In operators, I'd suggest going through unit_tests/operators_tests and seeing how they are computed/the properties they have.

Other than that, the PR is basically all covered. In the meantime, I will keep working towards getting the transonic testcase to work and verifying all tests.

As always, please let me know of any changes/remarks to be made, and I am always available to discuss!:)