Summary Report from the Discontinuous Galerkin Group
[HPC]^3 Workshop 2012, KAUST, Saudi Arabia

Scott Moe [1] and James A. Rossmanith [2]

1 Goal of the project

The DoGPack[3] software package is a C++ code with the capability to solve 1D and 2D time-dependent hyperbolic conservation laws using the discontinuous Galerkin (DG) method. The PyClaw[4] software package, which is an off-shoot of the CLAWPACK[5] package, is a Python-based code with the capability to solve 1D, 2D, and 3D time-dependent hyperbolic conservation laws using a variety of finite volume methods. The main purpose of this project is to integrate the DG solvers from DoGPack into the PyClaw software package. The two main reasons for doing this are as follows.

1. It would extend the functionality of the PyClaw code by allowing the user to choose a DG method in space.
2. It would allow DoGPack to take advantage of some of the features of the PyClaw code, including access to parallelized PETSc[6] arrays, as well as possible adaptive mesh refinement features.

The objective of giving PyClaw access to DoGPack objects and functions was not completed during the [HPC]^3 workshop. However, this project is still ongoing. In the subsequent sections we briefly describe some details of what was accomplished during the [HPC]^3 workshop and what future work still remains.

2 Results from the [HPC]^3 workshop

What was accomplished during the [HPC]^3 workshop can be summarized as follows.

1. The DoGPack routines for writing to output files was appended to include output that is directly readable by the VizClaw plotting routines.
2. A Python script was written that executes the DoGPack code through a systems call, and then runs the VizClaw plotting routines (either in interactive or html plot form) on the resulting data.

In order to accomplish this for several examples, we wrote a few short Python scripts. The main script was called `PyDOG.py`:

```
import os
import pyclaw
import numpy as np
def runDOG(iplot=False,htmlplot=False,outputdir='output'):
        # call dogpack and output to specified (or not) directory
        os.system('dog.exe -o %s' % outputdir)
if iplot:
        pyclaw.plot.interactive_plot('_output','asciiy')
if htmlplot:
        pyclaw.plot.html_plot('_output','asciiy')
```

For example, we used this script for the so-called *blast wave problem* for the 1D compressible Euler equations. In order to run this we wrote the short python script called `euler.py`:

[1] Department of Applied Mathematics, University of Washington, Box 352420, Seattle, WA 98195-2420, USA
[2] Department of Mathematics, University of Wisconsin, 480 Lincoln Drive, Madison, WI 53706-1388, USA
[3] http://www.dogpack-code.org
[4] http://numerics.kaust.edu.sa/pyclaw/
[5] http://www.clawpack.org
[6] http://www.mcs.anl.gov/petsc/

```
from PyDOG import runDOG
import pyclaw
def euler(iplot=False,htmlplot=False,outputdir='output'):
if iplot:
        runDOG(True,False,outputdir)
elif htmlplot:
        runDOG(False,True,outputdir)
else:
        runDOG(False,False,outputdir)

if __name__=="__main__":
        import sys
        from pyclaw.util import run_app_from_main
        output = run_app_from_main(euler)
```

In order to execute the script we typed at the UNIX command line:

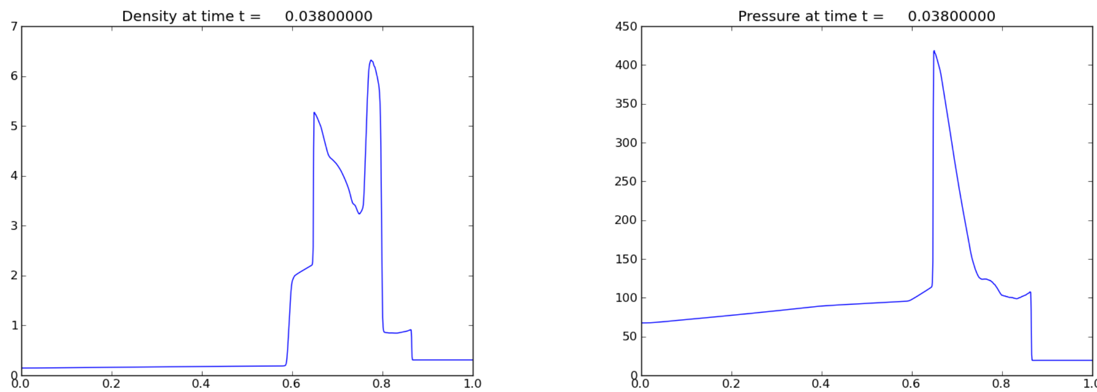
```
$ python euler.py htmlplot=1
```

This produced several plots, including those shown in Figure 1. We also considered several other examples including 2D advection (see Figure 2) and 1+1 Vlasov-Poisson (see Figure 3). We also show a 2D advection on an unstructured grid in Figure 4, although at this point we are only able to use MATLAB to plot unstructured grids and not VizClaw.

3 Future work

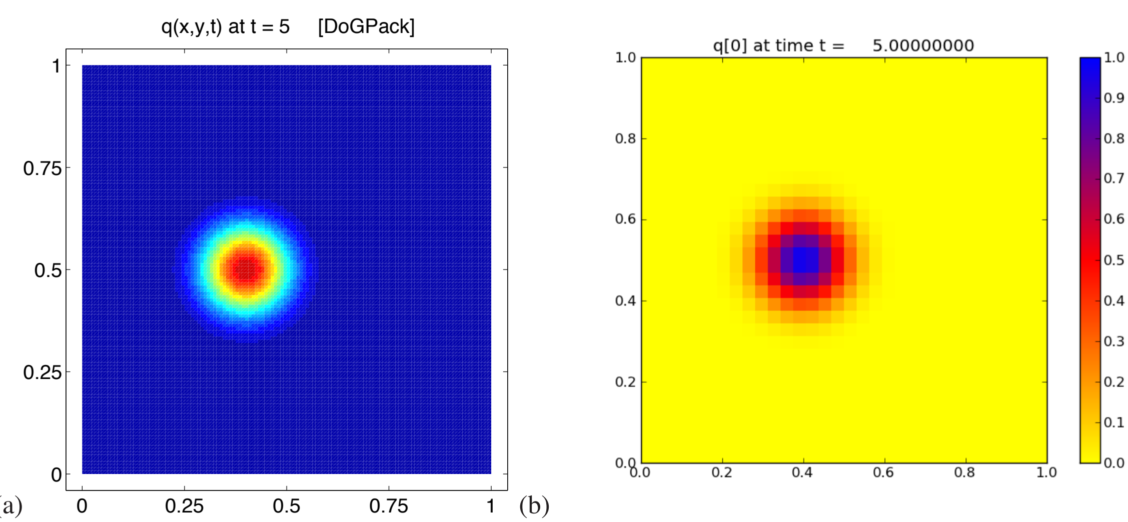
This project only represents the beginning of a more serious attempt to connect the DoGPack and PyClaw methodologies. In the future we would like to accomplish all of the following.

- Modifications to PyClaw:
  - Allow for the "state" variable to handle multiple degrees of freedom.
  - Create Python objects that can link to the DoGPack objects and functions.
  - Allow PyClaw to have easy access to the functions responsible for creating the right-hand sides of the PDEs so that existing PyClaw time-stepping methods can easily be called on top of the DoGPack-constructed right-hand sides.
  - Allow DoGPack to access PETSc and possible AMR capabilities of the PyClaw framework.
- Modifications to VizClaw:
  - Allow for VizClaw to plot multiple points per element.
  - Allow for VizClaw to directly read DoGPack output files rather than modified DoGPack output files.
  - Allow VizClaw to read and plot unstructured grid data. Again with the capability to handle multiple points per element.



(a) (b)

Fig. 1: Solutions to the blast wave problem for the 1D Euler equations from gas dynamics. Shown are the (a) density and (b) pressure at the final time in the simulation. The plots were done with VizClaw, showing only the average state in each element.



(a) (b)

Fig. 2: Solutions to the 2D advection equation with a purely azimuthal velocity field after five full rotations. Panel (a) shows the MATLAB plot of the DoGPack solution, while (b) shows the VizClaw plot of the same solution. Currently we are only able to plot one value per element using VizClaw, which explains why Panel (b) seems to have much less resolution than Panel (a).

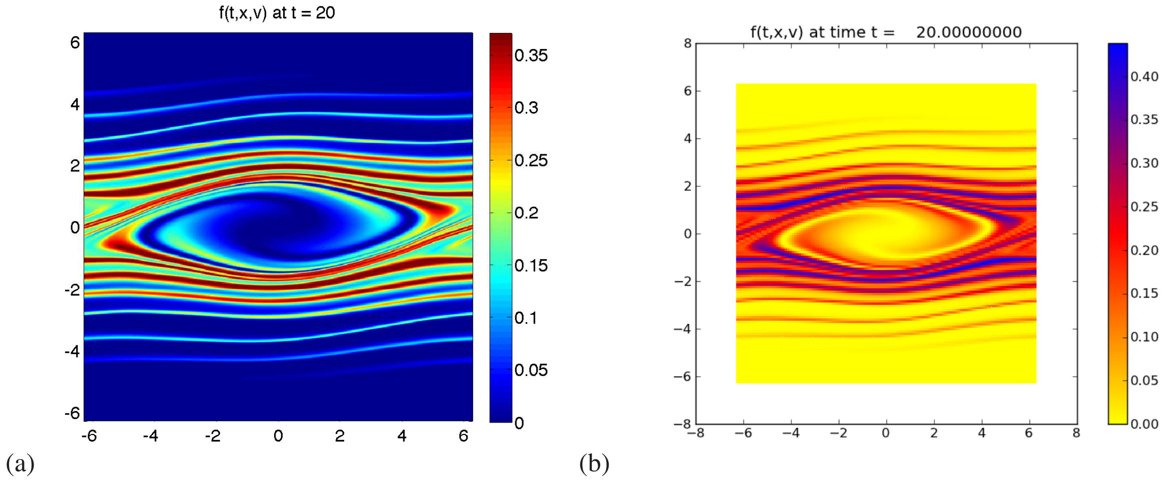


Fig. 3: Solutions to the 1+1 Vlasov-Poisson equation on the two-stream instability problem. Panel (a) shows the MATLAB plot of the DoGPack solution, while (b) shows the VizClaw plot of the same solution. Currently we are only able to plot one value per element using VizClaw, which explains why Panel (b) seems to have much less resolution than Panel (a).

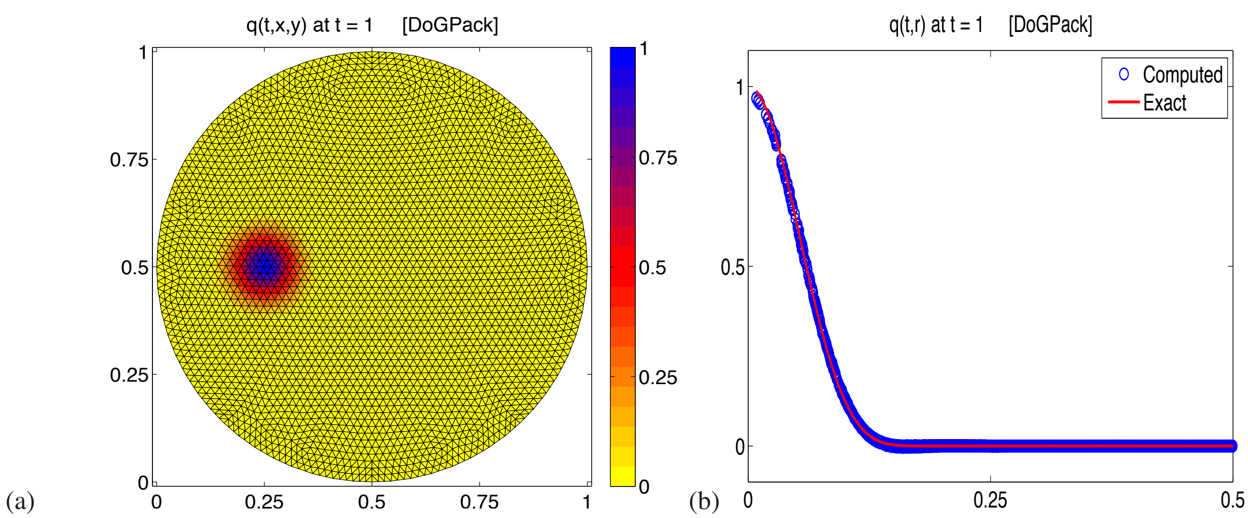


Fig. 4: Solutions to the 2D advection equation on an unstructured grid with a purely azimuthal velocity field after five full rotations. Panel (a) shows the MATLAB false-color plot of the density after one revolution, while (b) shows the MATLAB scatter plot of the density. Currently we are unable to plot unstructured grid data using VizClaw.