Discontinuous Galerkin Finite Element Methods for Parabolic and Elliptic Partial Differential Equations

Harvard School of Engineering and Applied Sciences | AC 274 Final Project

John Aoussou
Sydney Sroka

May 15, 2014

Acknowledgements

We would like to thank Professor Knezevic and TF Philip Mocz for their guidance and support. We learned a lot, and really enjoyed the course.

Abstract

We implement a discontinuous Galerkin (DG) finite element method to solve parabolic and elliptic partial differential equations. Solutions to the diffusion equation, advection-diffusion equation, and Poisson equation are computed on 1D and 2D domains. The techniques specific to a DG implementation are discussed and the results of selected test cases are verified analytically.

0. Introduction

Finite element methods are used to represent physical phenomena relevant to a broad spectrum of industrial applications and fields of research. As the scale and intricacy of the desired model increases, the implementation techniques fall under close scrutiny so as to optimize the recurring trade-off between computational cost and accuracy. A DG method holds several advantages over a continuous Galerkin (CG) method, including ease of parallelization, capacity to represent shocks, and, for some problems, DG exhibits better conservation properties. However, the DG method often incurs a larger computational cost from an increased number of degrees of freedom and the scheme is subject to weak instability and inconsistency from which other methods, such as finite differences, are free. When resolving a time dependent problem, the system of equations to solve is underdetermined, an issue originating from the increased number of degrees of freedom and from which a CG scheme does not suffer. Controlling the sources of instability is the most apparent challenge we encountered during the implementation of a DG method for parabolic and elliptic partial differential equations.
The diffusion, Poisson, and advection-diffusion equations are partial differential equations (PDE's) which are used to represent phenomena in many fields, but the physical applications we consider concentrate on conserved quantities relevant in computational fluid dynamics.

I. The Diffusion Equation

We start with the time-dependent diffusion equation, also called the heat equation.

$$\frac{\partial u}{\partial t} = \varepsilon \nabla^2 u \tag{1}$$

Where u represents a density of a conserved quantity, and is a function of space (x, y, z) and time (t), ε is a coefficient of diffusivity, and the Laplacian is a differential operator that can be generically written $\nabla^2 = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2} + \frac{\partial^2}{\partial z^2}$. For a 1D problem, the heat equation reduces to

$$\frac{\partial u}{\partial t} = \varepsilon \frac{\partial^2 u}{\partial x^2} \tag{2}$$

Equation 2 is the strong form of the PDE, referring to the increased "strength" of the requirements on u to be twice differentiable in space. To put equation 2 into weak form, in preparation for finite element discretization, it seems appropriate to follow the approach outlined in literature for DG methods with hyperbolic PDEs; specifically, multiply the equation by any test function, *v,* and integrate over the domain.

$$\int_{\Omega} \frac{\partial u}{\partial t} v dx = \varepsilon \int_{\Omega} \nabla^2 u \, v dx \tag{3a}$$

Then integrate the right-hand-side of equation 3a by parts to arrive at the following formulation.

$$\int_{\Omega} \frac{\partial u}{\partial t} v dx = \varepsilon \left(\int_{\partial\Omega} (\nabla u \cdot \vec{n}) v \, ds - \int_{\Omega} \nabla u \cdot \nabla v \, dx \right) \tag{3b}$$

However, upon implementation, this formulation is subject to instabilities derived from the flux term (a term that is absent in CG, since CG requires the solution to be continuous across elements). More information on this manifestation of instability can be found in Hesthaven and Warburton (H&W) [1].

To overcome this intrinsic obstacle, the diffusion equation can be rewritten as a system of first order PDEs [1, pp.244]. We consider first a 1D case.

$$\frac{\partial u}{\partial t} = \varepsilon \frac{\partial^2 u}{\partial x^2} = \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\right) \tag{4a}$$

$$\frac{\partial u}{\partial t} = \varepsilon \frac{\partial q}{\partial x} \tag{4b}$$

$$q = \frac{\partial u}{\partial x} \tag{4c}$$

These equations are solved serially to evolve the solution in time. A weak formulation of the system of first order PDEs is constructed for a DG implementation.

$$\int_{T_k} \frac{du_h}{dt}\phi_i^k dx = \varepsilon \int_{T_k} \frac{dq_h}{dx}\phi_i^k dx = \varepsilon\left[q_h^* \phi_i^k\right]_{x_l^k}^{x_r^k} - \varepsilon \int_{T_k} q_h \frac{d\phi_i^k}{dx} dx \tag{5a}$$

Doing a direct sum over all the elements:

$$M\frac{dU}{dt} = \varepsilon\left(C_{avg} - S\right)Q \tag{5b}$$

$$\int_{T_k} q_h \phi_i^k \, dx = \int_{T_k} \frac{du_h}{dx}\phi_i^k dx = \left[u_h^* \phi_i^k\right]_{x_l^k}^{x_r^k} - \int_{T_k} u_h \frac{d\phi_i^k}{dx} dx \tag{6a}$$

Doing a direct sum over all the elements:

$$MQ = \left(C_{avg} - S\right)U \tag{6b}$$

Where $u_h \subset V_h$ and the test functions ϕ_i^k represents the interpolant corresponding to the i^{th} node on element k. Familiar matrices include a mass matrix, denoted $M = \int_{T_k} \phi_j^k \, \phi_i^k dx$, an advection matrix, denoted $S = \int_{T_k} \frac{d\phi_j^k}{dx}\phi_i^k dx$ and an average coupling matrix, C_{avg} which comes from evaluating the central flux over a boundary.

The boundary flux terms q^* and u^* are taken to be central fluxes $\{\{q_h\}\}$ and $\{\{u_h\}\}$ (reference H&W pp. 20).
In 1D, the central flux on an element boundary can be represented as follows, where '+' denotes element interior information, and '-' denotes exterior information.

$$\{\{q_h\}\} \equiv \frac{q_h^- + q_h^+}{2}$$

for q_l^k equal to the value of q on the left boundary of element k and q_r^{k-1} equal to the value of q on the right boundary of element $k-1$.

A necessary condition for stability of the diffusion equation is the relevant form of the Courant–Friedrichs–Lewy condition (CFL condition). The condition helps ensure that the time step is sufficiently small so that the solution does not diffuse across more than one cell between successive time level calculations. This condition exists in different form for different equations. For the diffusion equation, the conditions can be written as $\varepsilon\frac{\Delta t}{\Delta x^2} < \frac{1}{2}$ in 1D and $\varepsilon\Delta t\left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}\right) < \frac{1}{4}$ in 2D ([1] pp. 250).

We demonstrate our implementation of the diffusion equation for two initial conditions, a square pulse and a Gaussian. Neumann boundary conditions are enforced at the domain boundaries to represent walls at either end of the domain. Imposing no penetration conditions is achieved by setting homogeneous conditions on the intermediate variable, q, which represents the flux of u. Since $q = u_x$, requiring $q(a) = q(b) = 0$ imposes no penetration.

We initialize a square pulse, using the Matlab *Heaviside* function. The initial condition in Figure 1, is $u(x,0) = 1$ for $|x| < 0.5$ and $u(x,0) = 0$ for $|x| > 0.5$ and the diffusion evolves for $t \in (0,1.5]$ and $x \in [-1,1]$. We use 75 elements, linear basis functions, and a time step = 0.0001.

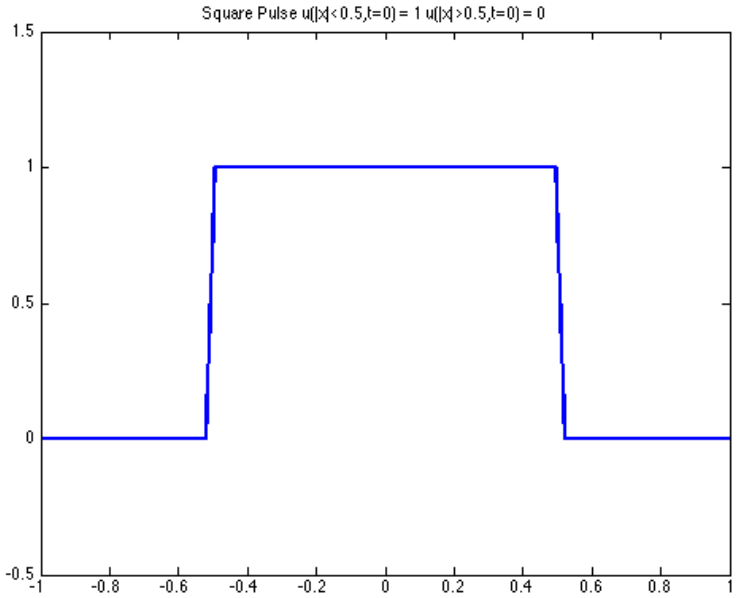

Figure 1: Initial condition for pure diffusion of a square pulse

We compare different diffusivities, $\varepsilon \in [0.01;\ 0.1;\ 1]$, over the same time interval as seen in Figure 2. The solution is expected to converge to a uniform distribution along the domain such that mass is conserved.

$$\forall t, \int_a^b u(x,t)\ dx = \int_a^b u(x,t=0)\ dx$$

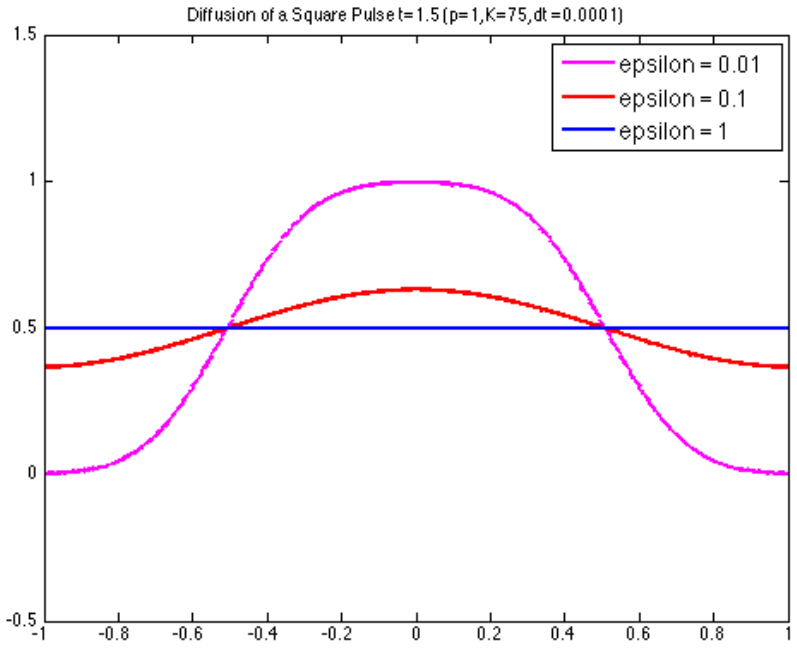

Figure 2: The diffusion equation solved for different diffusivities over the same time interval. The largest coefficient of diffusivity results in the most uniform profile, as expected. This test case confirms the implementation is conservative since the mass at t=0 is equal to the mass at the final time.

The solutions converge to the expected profile, conserving mass from the initial $(0.5 - (-0.5)) \times 1 = 1$ sq unit of mass, distributed over $x \in [-1,1]$ suggests a steady state profile of

$$u(x) = \frac{1}{\left(1 - (-1)\right)} = 0.5$$

The second test case has the Gaussian initial condition seen in Figure 3, and the diffusion transpires similarly. The integrated area under the initial curve is $u(x,0) = \int_{-1}^{1} \mathrm{e}^{-100*\mathrm{x}^2}\, dx = 0.1772$ which implies that the steady state solution for no penetration boundary conditions is $u(x) = 0.0886$, and we find that our solution again converges to a reasonable result (see callout from Figure 4) which gives us confidence in our application of the no penetration boundary conditions.

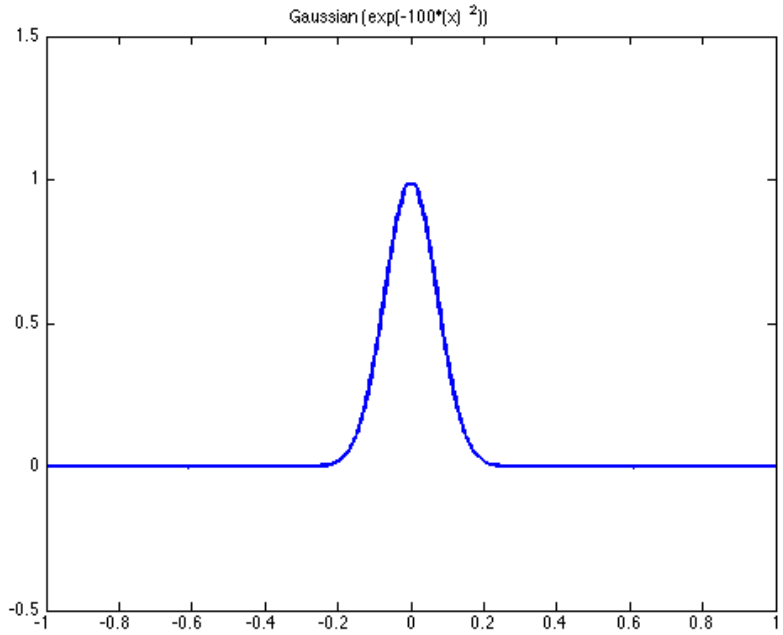

Figure 3: Gaussian initial condition for diffusion

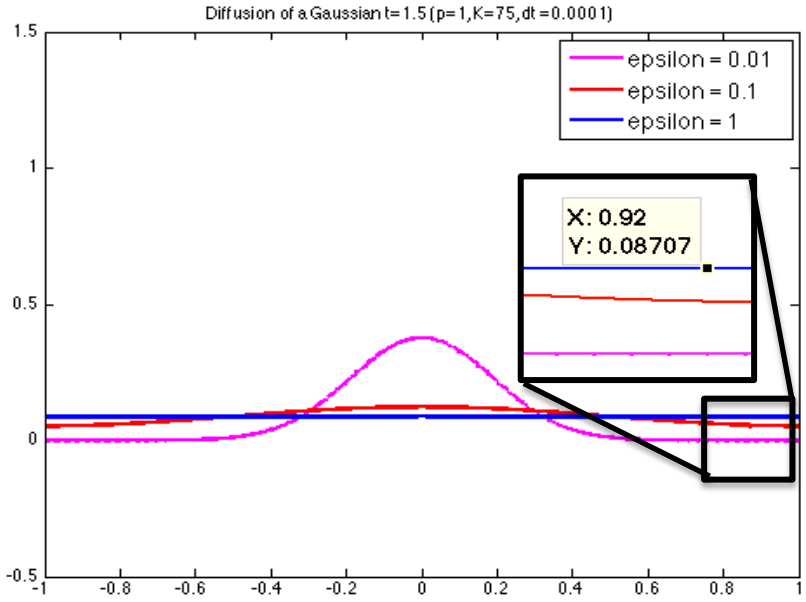

Figure 4: Diffusion of a Gaussian. Since the integral of the Gaussian over [-1,1] is computed analytically to be 0.0886, the amplitude of the solution, referenced in the callout, with the largest coefficient of diffusivity (in blue above) agrees well with the analytical result.

The contribution from numerical diffusion is investigated by comparing the solutions with interpolants of different polynomial orders. The results in Figure 5 suggest that numerical diffusion does not significantly distort the solution for this test case.

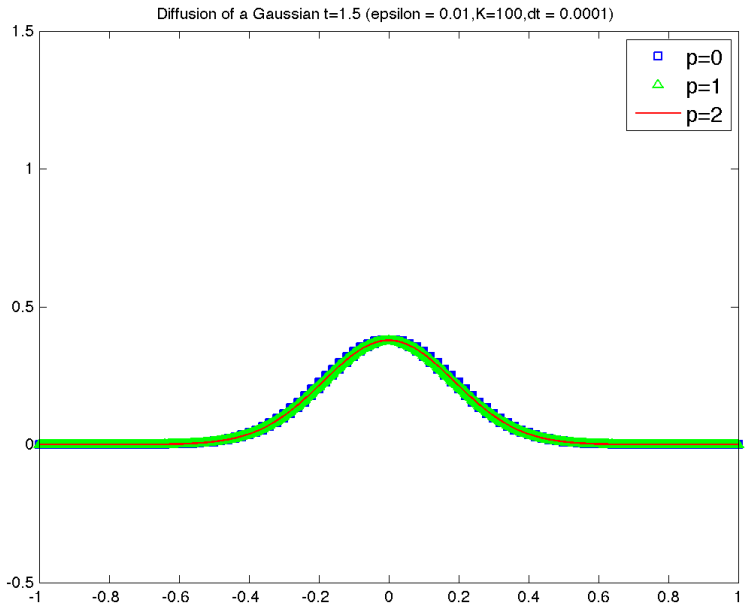

Figure 5: The diffusion equation is solved for consistent initial condition and domain parameters, varying the order of the polynomial basis function to examine contributions from numerical diffusion that may be influencing the solution

II. Advection-Diffusion Equation

We proceed to incorporate the advection into the simulation following the governing equation (equation 7).

$$\frac{\partial u}{\partial t} = \varepsilon \nabla^2 u - \nabla \cdot (\vec{w} u) \tag{7}$$

The weak form is derived with a system of first order PDE's as it was for the diffusion equation.
Equations 4(a-c) become equations 8(a-c)

$$\frac{\partial u}{\partial t} = \varepsilon \frac{\partial^2 u}{\partial x^2} - w \frac{\partial u}{\partial x} = \varepsilon \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\right) - w \frac{\partial u}{\partial x} \tag{8a}$$

$$\frac{\partial u}{\partial t} = \varepsilon \frac{\partial q}{\partial x} - w \frac{\partial u}{\partial x} \tag{8b}$$

$$q = \frac{\partial u}{\partial x} \tag{8c}$$

The system of first order equations is put into weak form in preparation for finite element discretization.

$$\int_{T_k} \frac{du_h}{dt} \phi_i^k dx = \varepsilon \int_{T_k} \frac{dq_h}{dx} \phi_i^k dx - w \int_{T_k} \frac{du_h}{dx} \phi_i^k dx$$

$$= \varepsilon \left(\left[q_h^* \phi_i^k \right]_{x_l^k}^{x_r^k} - \int_{T_k} q_h \frac{d\phi_i^k}{dx} dx \right) - w \left(\left[u_h^* \phi_i^k \right]_{x_l^k}^{x_r^k} - \int_{T_k} u_h \frac{d\phi_i^k}{dx} dx \right) \tag{9a}$$

Equations 6(a,b) remain the same.

The flux term from the advection expression requires an artificial diffusion term. We implement a jump coupling matrix with a global Lax-Friedrichs flux condition. More on this stability condition can be found in H&W ([1], pp 33).

The boundary flux terms q^* and u^* are taken to be $\{\{q_h\}\} + [\![u_h]\!]$ and $\{\{u_h\}\}$,respectively, where the double bracket denotes a jump operator defined with the same convention for '+' and '-' symbols ([1] pp. 20).

$$[\![u_h]\!] = n^- u^- + n^+ u^+$$

A DG scheme is implemented with the following system of equations.

$$Q = M^{-1}(C_{avg} - S)U \tag{9b}$$

$$M\frac{dU}{dt} = (C_{avg} - S)Q + \left[(S - C_{avg})w - \frac{C_{GLF}}{2}C_j\right]U \tag{9c}$$

Where C_j is the jump coupling matrix originating from the jump operator. (We drop the subscript 'h' and understand $u = u_h$from here on).

We solve a test case with a Gaussian (exp(-100 x^2)) initial condition on $-1 \leq x \leq 1$ for $t \in (0,1]$, an advection velocity of 0.5, and periodic boundary conditions. The solution is plotted alongside the initial condition in Figure 6.

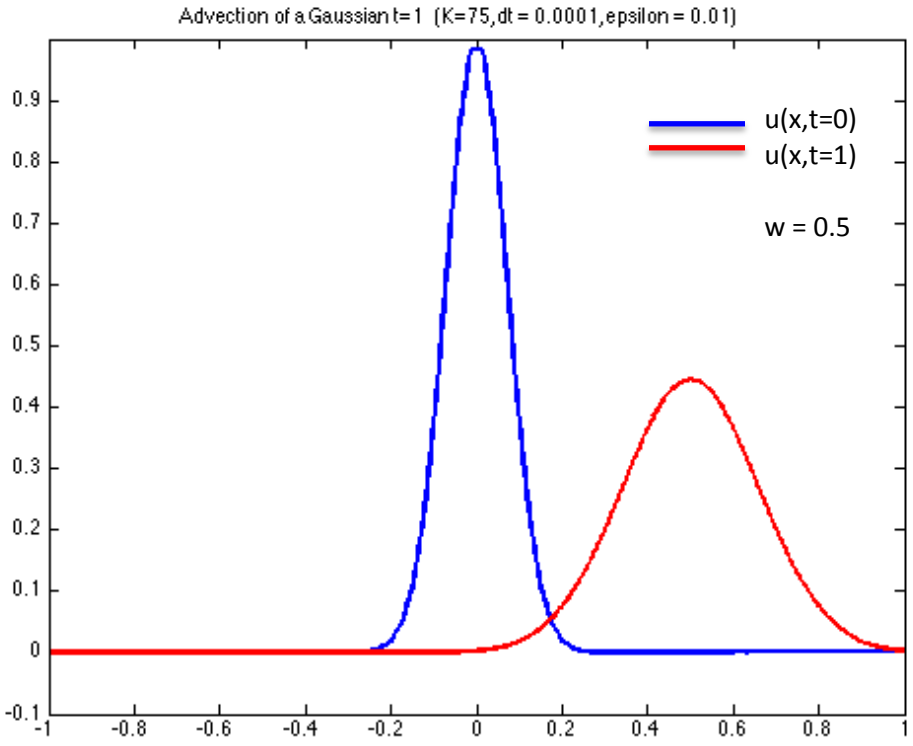

Figure 6: DG solution to the Advection-Diffusion equation (red) plotted against the initial condition (blue) for a advection velocity = 0.5

Advection introduces an opportunity to observe numerical diffusion due to basis order differences. We compare the solutions from constant, linear, and quadratic basis functions for the same test case in Figure 7. As expected, the zeroth order basis functions exhibit more numerical diffusion than the higher order basis functions.

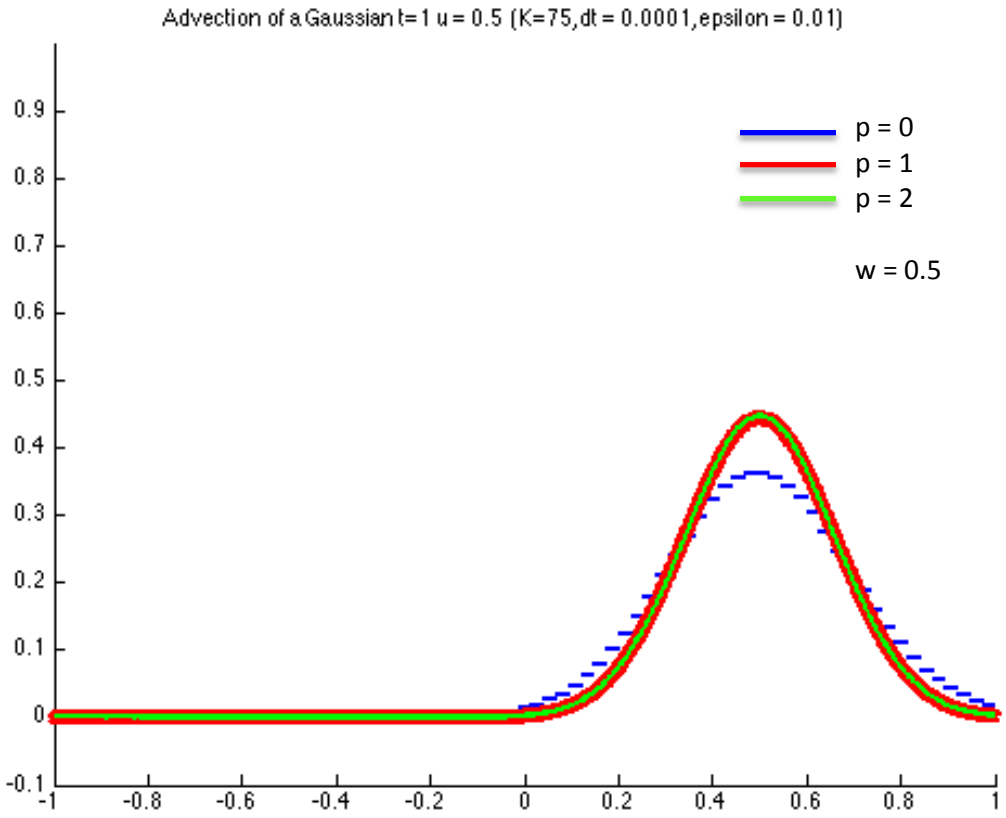

Figure 7: The advection-diffusion equation using a different polynomial order for the basis functions for successive solutions over the same domain and time interval.

III. Steady State with a Forcing Function

We demonstrate a DG implementation for the steady state advection-diffusion equation with a forcing term- also referred to as the Poisson equation- and discuss important considerations that are particular to a DG scheme. We begin, as we did for the unsteady case, by formulating a system of first order equations. Equations 4(a-c) become

$$f = \varepsilon \frac{\partial^2 u}{\partial x^2} = \frac{\partial}{\partial x}\left(\frac{\partial u}{\partial x}\right) \tag{10a}$$

$$f = \varepsilon \frac{\partial q}{\partial x} \tag{10b}$$

$$q = \frac{\partial u}{\partial x} \tag{10c}$$

$$\int_{T_k} f\,\phi_i^k dx = \varepsilon \int_{T_k} \frac{dq_h}{dx}\phi_i^k dx = \varepsilon\left(\left[q_h^* \phi_i^k\right]_{x_l^k}^{x_r^k} - \int_{T_k} q_h \frac{d\phi_i^k}{dx} dx\right) \tag{11a}$$

$$MF = \varepsilon\left(C_{avg} - S\right)Q - \tau C_j U \tag{11b}$$

The f represents a forcing function. We need to impose a penalty term on the flux: $q^* = \{\{q\}\} - \tau[|u|]$. This yields:

$$\int_{T_k} q_h \phi_i^k\, dx = \int_{T_k} \frac{du_h}{dx}\phi_i^k dx = \left[u_h^* \phi_i^k\right]_{x_l^k}^{x_r^k} - \int_{T_k} u_h \frac{d\phi_i^k}{dx} dx \tag{12a}$$

With $u^* = \{\{u\}\}$

$$MQ = \big(C_{avg} - S\big)U \tag{12b}$$

On Weakly Imposed Dirichlet Boundary Conditions: We have partitioned the responsibility of boundary fluxes so that the coupling matrices, C_{avg} and C_j only contain interior boundary edge terms, and a boundary matrix C_{BC} is utilized to impose boundary conditions on exterior boundaries. However, the system of first order equations poses a challenge to this construction In order to impose Dirichlet boundary condition weakly for the steady state problem, the average coupling matrix in equation (12b) needs to include coupling terms that correspond to the exterior boundary.

$$MF = \Big(\big(C_{avg} + \boldsymbol{C_{BC}}\big) - S\Big)M^{-1}\big(C_{avg} - S\big)U - \tau C_j U \tag{13}$$

To intuitively understand the motivation behind this construction, let us consider the equation obtained for a hyperbolic PDE, specifically, an advection problem. $u_t + f(u)_x = 0$.

$$M\frac{dU}{dt} = \big(C_{avg} - S\big)F + C_{BC}F_{BC} + \frac{C}{2}C_j U \tag{14}$$

The vector denoted F is a function of U and represents the discretized flux function (f). For an advection problem, F can be written as $F = WU$ where W is a velocity matrix, so that (14) can be rewritten as

$$M\frac{dU}{dt} = \big(C_{avg} - S\big)WU + C_{BC}W_{BC}U_{BC} + \frac{C}{2}C_j U \tag{15}$$

Where U_{BC} is equal to the vector U at the current time step which is modified to respect the boundary conditions. Since the flux function is the same on the boundary, $W_{BC} = W$.

$$M\frac{dU}{dt} = \big(C_{avg} - S\big)WU + C_{BC}WU_{BC} + \frac{C}{2}C_j U \tag{16}$$

In the steady state case, $U_{BC} = U$ and (13) can be rewritten as

$$MF = \big(C_{avg} - S + C_{BC}\big)WU - \tau C_j U \tag{17}$$

This is our formulation for homogeneous Dirichlet boundary conditions. To impose inhomogeneous boundary conditions, we must take into account the boundary terms that cancel when the boundary conditions are homogeneous.

In this case, (12a) modifies to

$$\int_{T_k} q_h \phi_i^k \, dx = \int_{T_k} \frac{du_h}{dx}\phi_i^k dx = \big[u_h^* \phi_i^k\big]_{x_l^k}^{x_r^k} + \big[u_{h_{BC}}^* \phi_i^k\big]_{x_l^k}^{x_r^k} - \int_{T_k} u_h \frac{d\phi_i^k}{dx} dx \tag{18a}$$

$$MQ = \big(C_{avg} - S\big)U + C_{BC}U_{in} \tag{18b}$$

with U_{in} being a vector containing the left boundary value as its first element and minus the right boundary condition as its last element.

Equation 17 thus becomes:

$$MF + \big(C_{avg} - S\big)M^{-1}C_{BC}U_{in} = \big(C_{avg} - S\big)M^{-1}\big(C_{avg} - S\big)U - \tau C_j U \tag{19}$$

Or more concisely: $MF + F_b = \mathcal{A}U \quad \Longrightarrow U = \mathcal{A}^{-1}(MF + F_b)$

We use our implementation of DG to solve the Poisson equation for f(x) =- sin(x), since the diffusion equation can be solved analytically for this choice of f(x)- the solution is u(x) = sin(x)- and can verify our solution with code from H&W.

We seek to verify our implementation with the parameters from H&W, and the corresponding code (see PoissonCDriver1D.m from [2]). The code solves the Poisson equation with f(x) = -sin(x) with a domain $0 \leq x \leq 2\pi$. The results in Figure 8 agree very well, the largest difference between corresponding coefficients (recall the node locations are the same) is less than 5%.

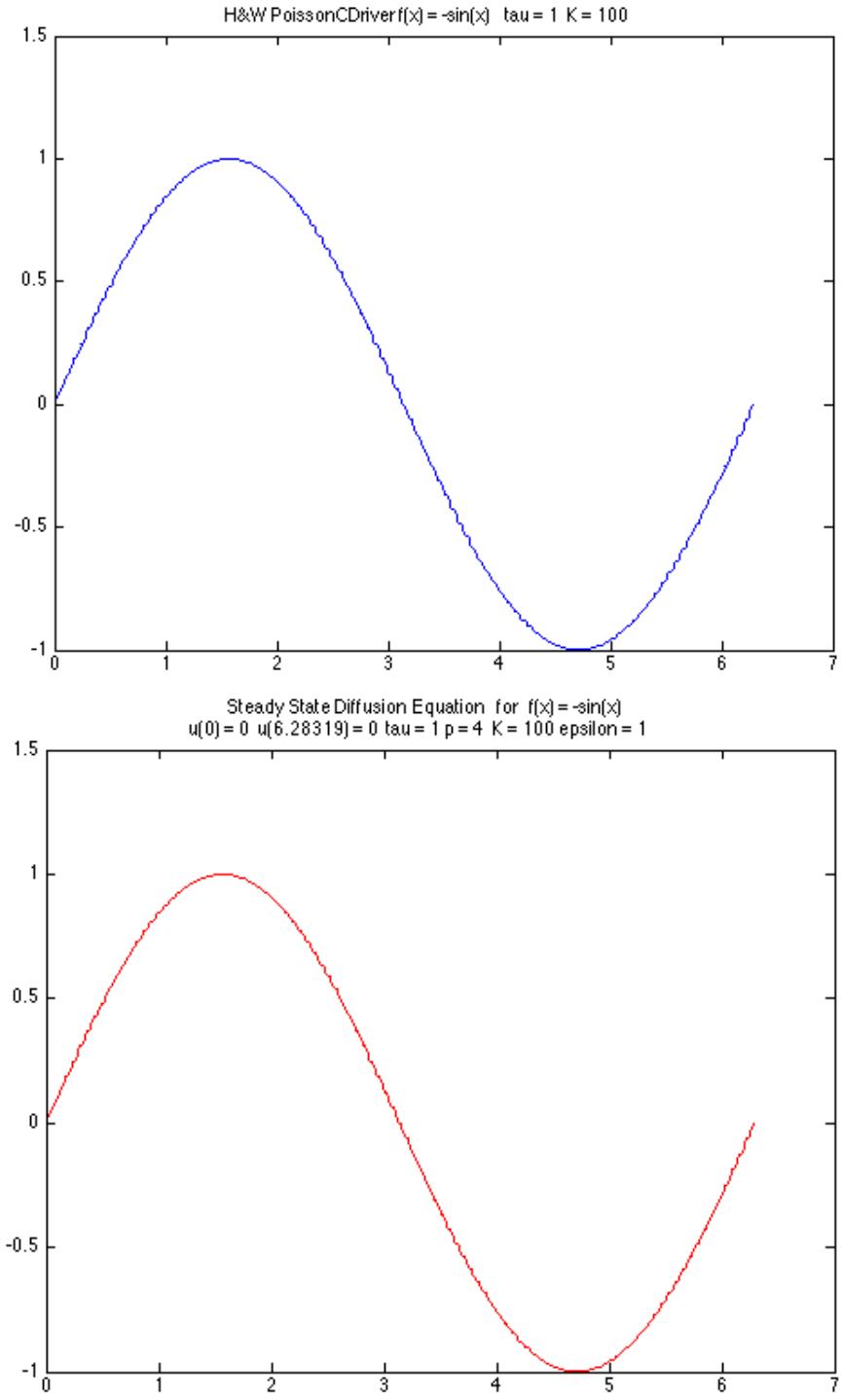

Figure 8: The Poisson Equation solved with DG from the H&W reference code (top) and our solution (bottom) from our DG implementation for the same equation, homogeneous Dirichlet boundary conditions, and problem parameters including tau = 1.

We demonstrate the influence of the penalty term in Figure 9 by executing each simulation again, but with $\tau = 0$.

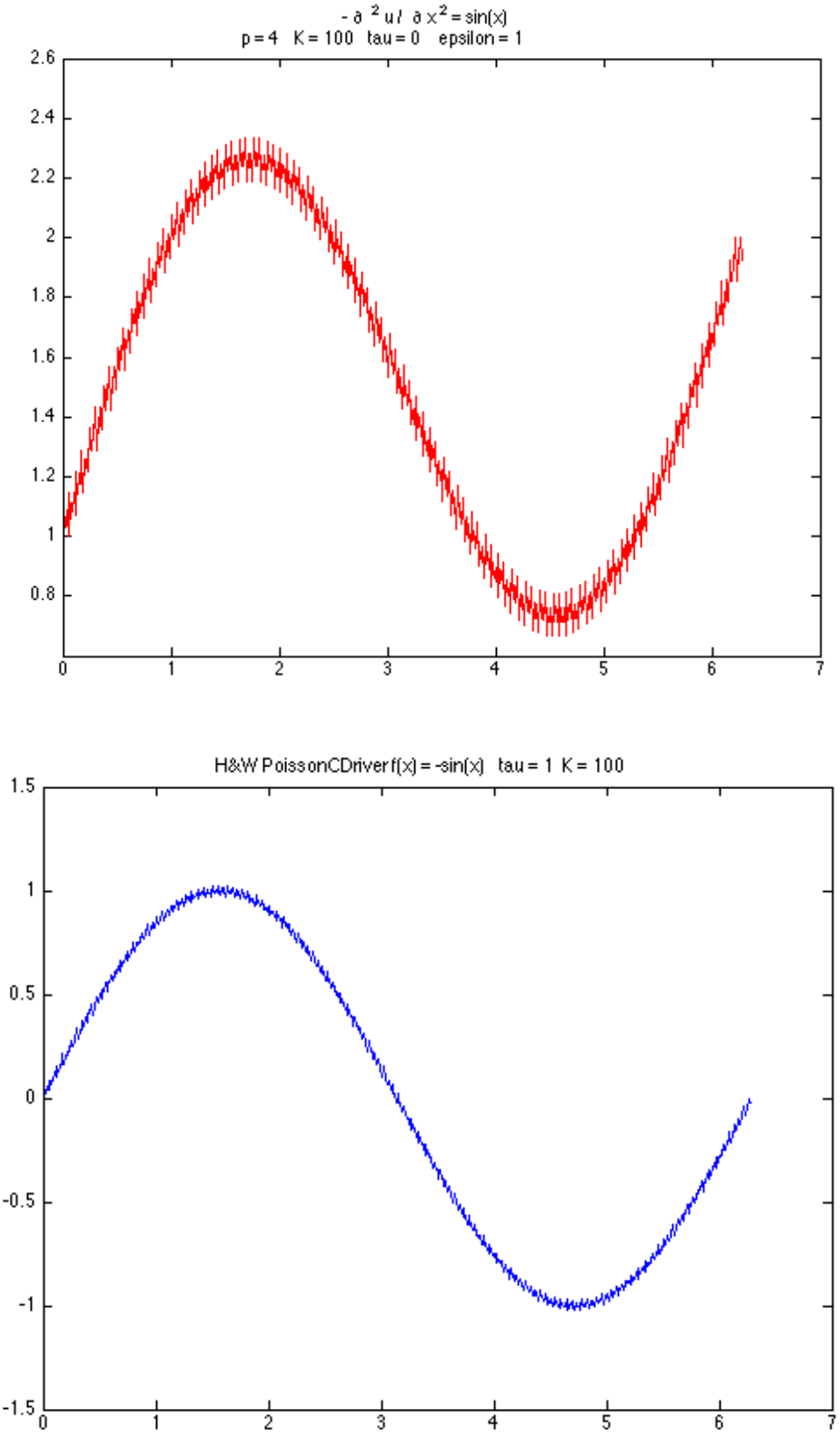

Figure 9: The Poisson Equation solved with DG from the H&W reference code (top) and our solution (bottom) from our DG implementation for the same equation but without the penalty term to demonstrate the observable instability the penalty term removes.

Since the condition number of the matrix,$\mathcal{A}$, that is inverted to solve for U is very large, slight differences in computational techniques may result in slightly different magnitudes of instability. However, the penalty term controls the instability in both simulations such that the solutions from the two simulations converge to the same result for the same coefficient τ, which gives us confidence in our implementation.

Inhomogeneous boundary conditions

The final test case for the 1D Poisson Equation is f(x) = -sin(x) with inhomogeneous Dirichlet boundary conditions u(0) = 1 and $u(2\pi) = 2$. The problem parameters are kept consistent, and the result in Figure 10, is both free from severe oscillations and respects the Dirichlet conditions.

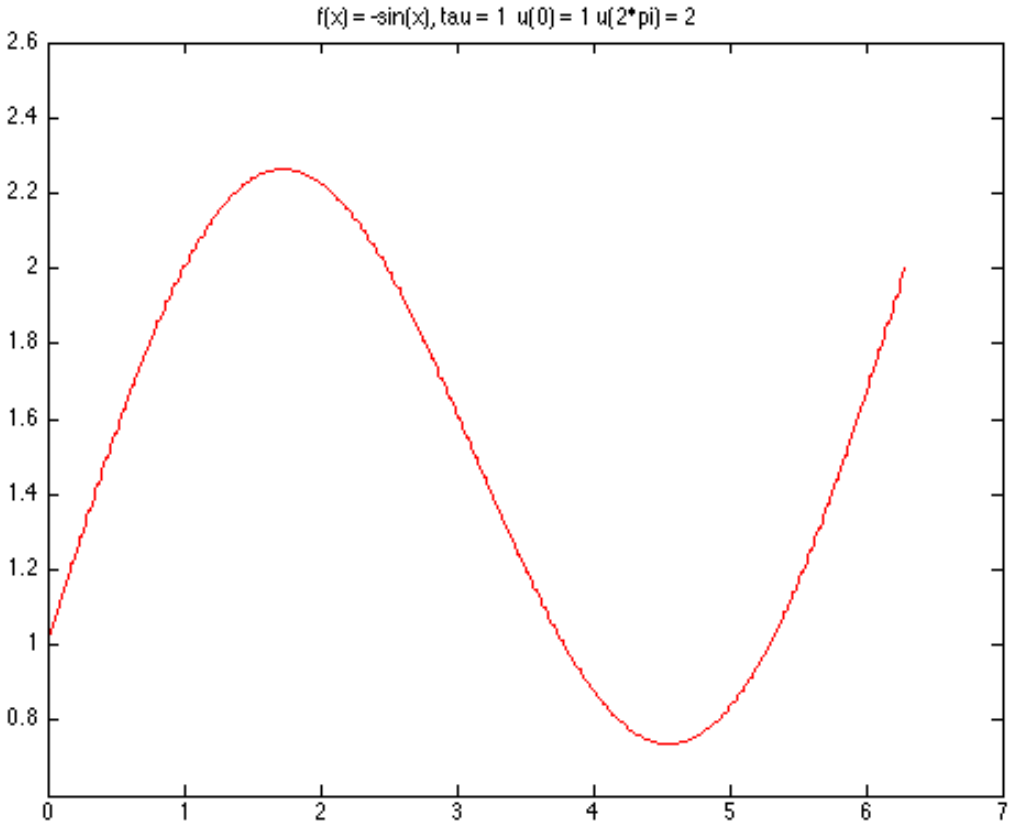

Figure 10: DG solution to the Poisson equation with inhomogeneous Dirichlet Boundary conditions

IV. The Diffusion Equation in 2D

We extend the 1D formulation to solve the diffusion equation on a 2D domain. The equations are similar to those of the 1D case, however while u remains a scalar quantity, q is now a vector quantity and will be denoted as $\vec{q} = (q_1, q_2)$. Recall the diffusion equation (1). Now the vector quantity appears in our system of first order PDE's seen in equations 19 and 20.

$$u_t = \varepsilon \nabla^2 u \tag{1}$$

$$u_t = \varepsilon(\nabla \cdot \vec{q}) \tag{19}$$

$$\vec{q} = \nabla u \tag{20}$$

Our weak formulation from equations 5 and 6 are modified to for a vector $\vec{q}$ (the flux term is again represented with a central difference as in 1D).

$$\int_{T_k} u_t \phi_i \, dx = \varepsilon \int_{T_k} (\nabla \cdot \vec{q}) \phi_i \, dx = \varepsilon \left(\int_{\partial T_k} \phi_i \overrightarrow{q^*} \cdot \vec{n} \, ds - \int_{T_k} \vec{\nabla} \phi_i \cdot \vec{q} \, dx \right) \tag{21a}$$

$$M \frac{dU}{dt} = \varepsilon \left(C_{avg_1} Q_1 + C_{avg_2} Q_2 - S_1 Q_1 - S_2 Q_2 \right) \tag{21b}$$

$$\begin{cases} \int_{T_k} q_1 \phi_i \, dx = \int_{\partial T_k} u \, \phi_i n_1 ds - \int_{T_k} u \frac{\partial \phi_i}{\partial x_1} dx \\ \int_{T_k} q_2 \phi_i \, dx = \int_{\partial T_k} u \, \phi_i n_2 ds - \int_{T_k} u \frac{\partial \phi_i}{\partial x_2} dx \end{cases} \tag{22a}$$

$$\begin{cases} MQ_1 = C_{avg_1} U - S_1 U \\ MQ_2 = C_{avg_2} U - S_2 U \end{cases} \tag{22b}$$

To implement the formulations in 21 and 22 we again reference the matrices with which we are familiar.

$$MQ_1 = C_{avg_1} U - S_1 U$$
$$MQ_2 = C_{avg_2} U - S_2 U$$
$$M \frac{dU}{dt} = \varepsilon \left(C_{avg_1} Q_1 + C_{avg_2} Q_2 - S_1 Q_1 - S_2 Q_2 \right)$$

The test case we examine is on a domain $\Omega = [-0.5, 0.5]^2$ with a Gaussian initial condition of $u(x, y, 0) = 100 * \exp(-100(x^2 + y^2))$. The no penetration boundary conditions for this 2D test case are imposed weakly through the flux, analogously to the 1D, case by requiring the intermediate variable, *q*, to be zero at the exterior boundaries.
We use the **`distmesh`** package to generate the mesh, depicted in Figure 11, which consists of triangles with straight edges. We have chosen quadratic interpolants (p=2) and have included several realizations in Figure 12. The simulation reaches a steady state for $\varepsilon = 0.01$around t=1 agreeing well with the expected profile.

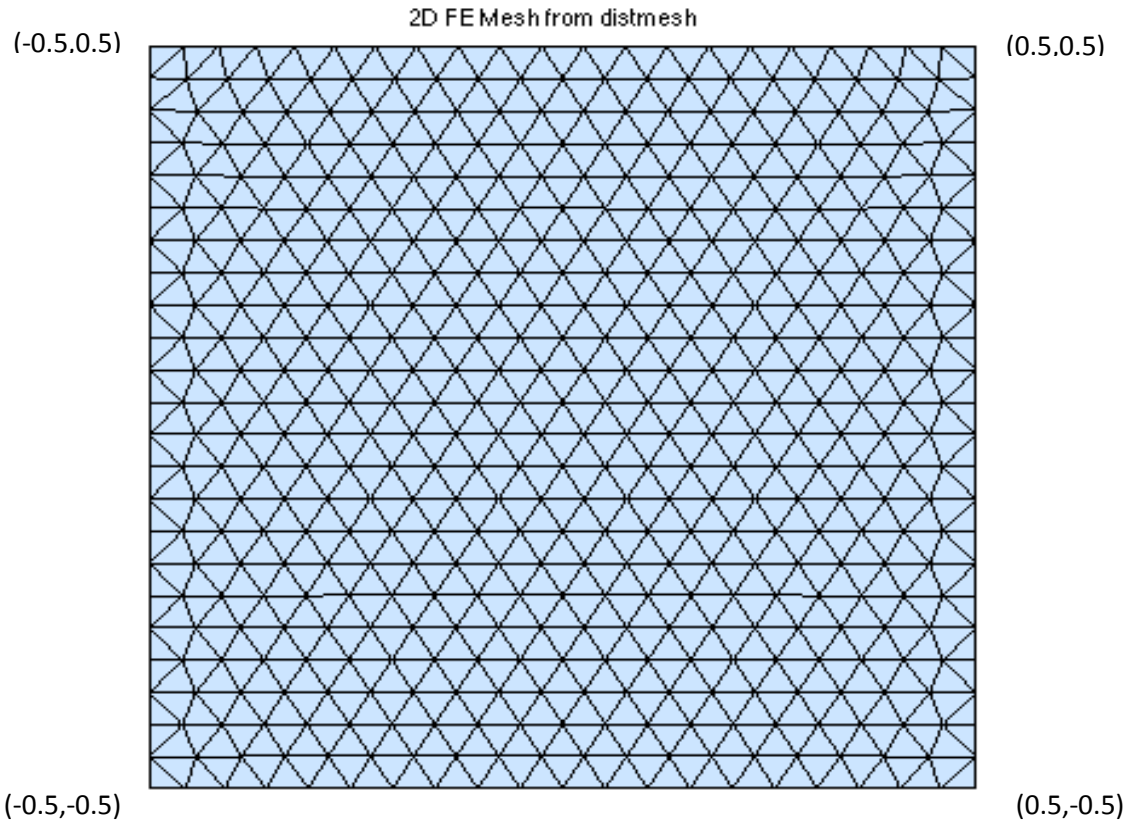

Figure 11: The Poisson Equation solved with DG from the H&W reference code (top) and our solution (bottom) from our DG

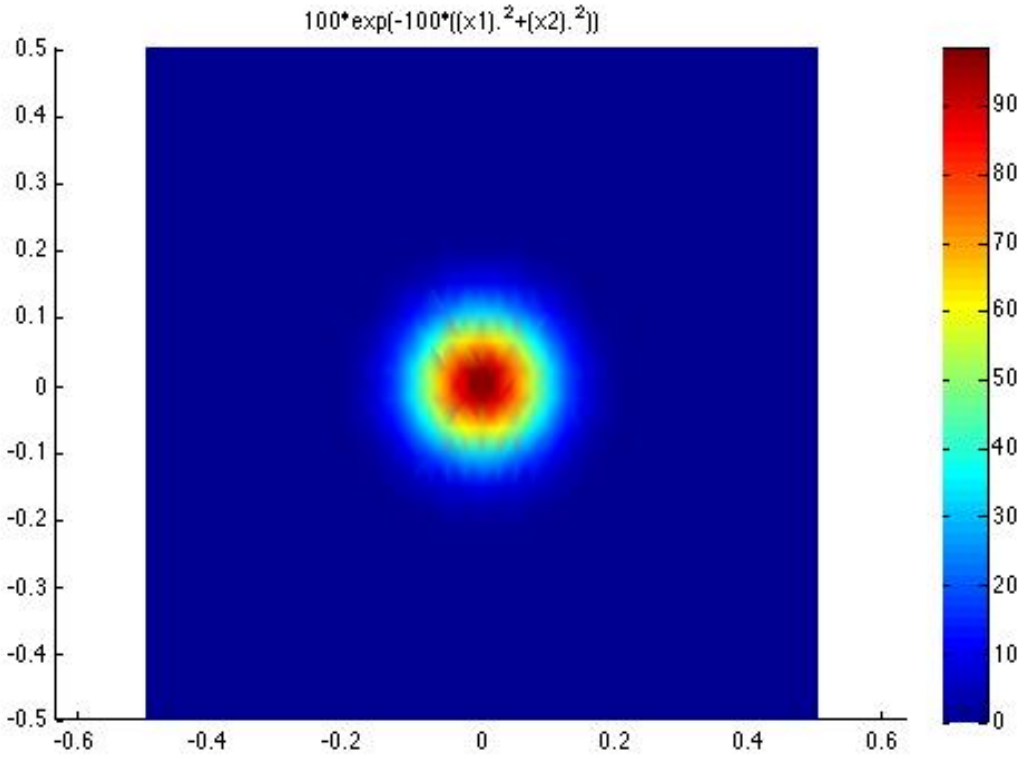

Figure 12 a

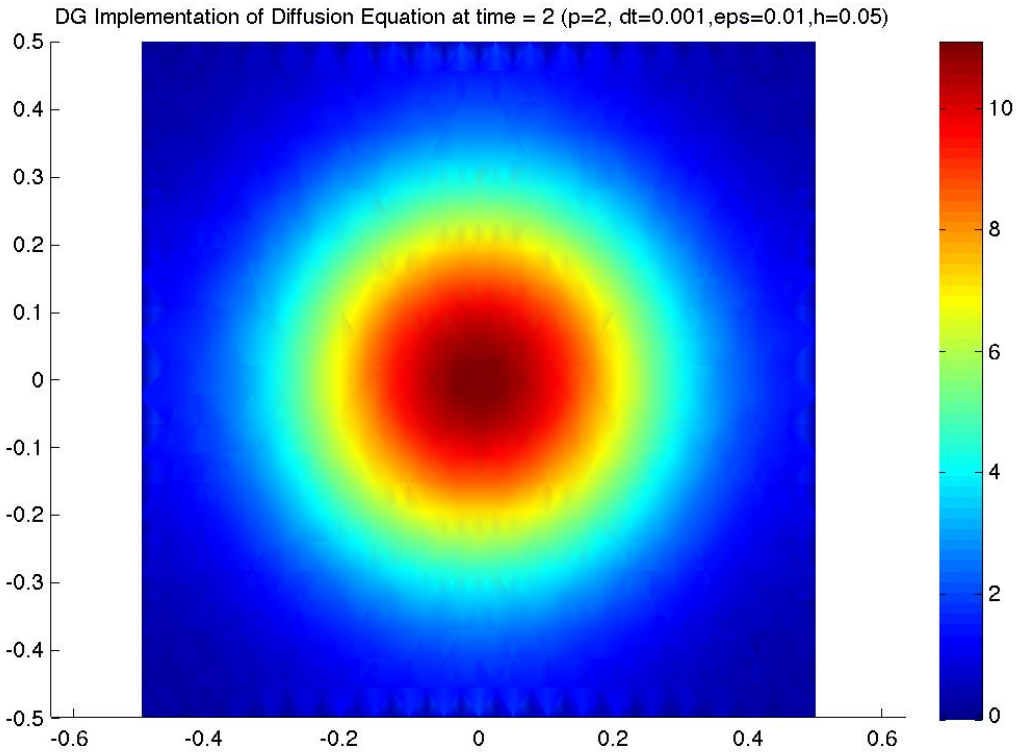

Figure 12 b

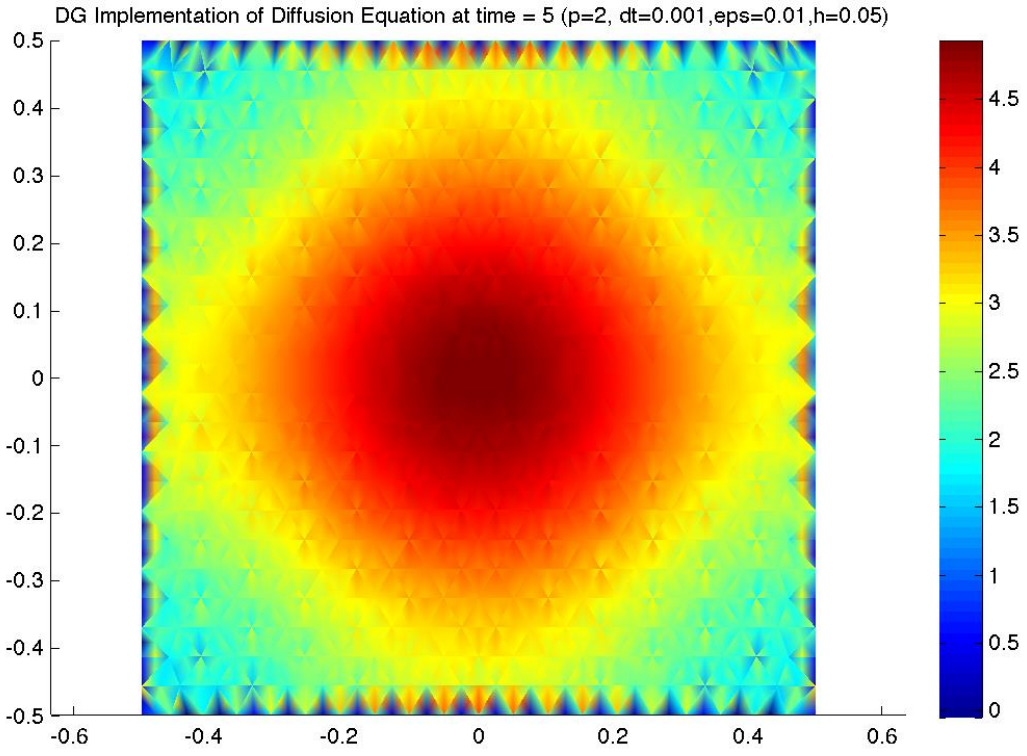

Figure 12 c

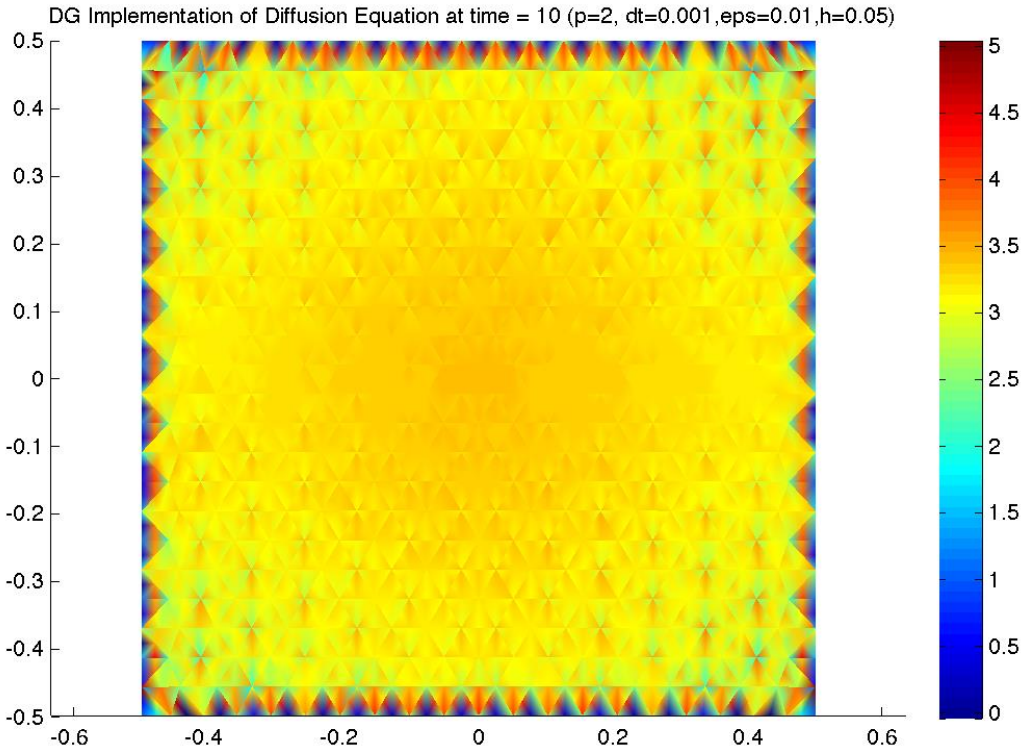

Figure 12d

Figure 12: Several realization for the Diffusion Equation solved with DG on a 2D domain, with Neumann no penetration boundary conditions on the exterior boundary edges.

V. The Advection-Diffusion Equation in 2D

The extension to 2D follows from the 1D case for the advection-diffusion equation. We incorporate our global Lax-Friedrichs flux as before, and write the advection terms in the x1 and x2 direction separately.

$$\begin{cases} MQ_1 = C_{avg_1}U - S_1U \\ MQ_2 = C_{avg_2}U - S_2U \end{cases} \tag{23a}$$

$$M\frac{dU}{dt} = \varepsilon\left(C_{avg_1}Q_1 + C_{avg_2}Q_2 - S_1Q_1 - S_2Q_2\right) - w_1\left(C_{avg_1}U - S_1U\right) - w_2\left(C_{avg_2}U - S_2U\right) - \frac{C_{GLF}}{2}C_jU \tag{23b}$$

The test case we examine for advection-diffusion in 2D is similar to the test case from the 2D diffusion equation with the same initial condition, see Figure 12a. We select an advection velocity of w = [0.2,0.2], a diffusivity of 0.01, and quadratic interpolants. The solution evolves until t = 0.5, see Figure 13, which should correspond to the peak location of approximately (0.1,0.1). Unfortunately, imposing boundary conditions weakly through the flux for the advection diffusion case in 2D is more complex in than for the purely diffusive case. Therefore we restricted our simulations to those which did not interact directly with the boundary and leave the task of introducing appropriate boundary conditions for future work.

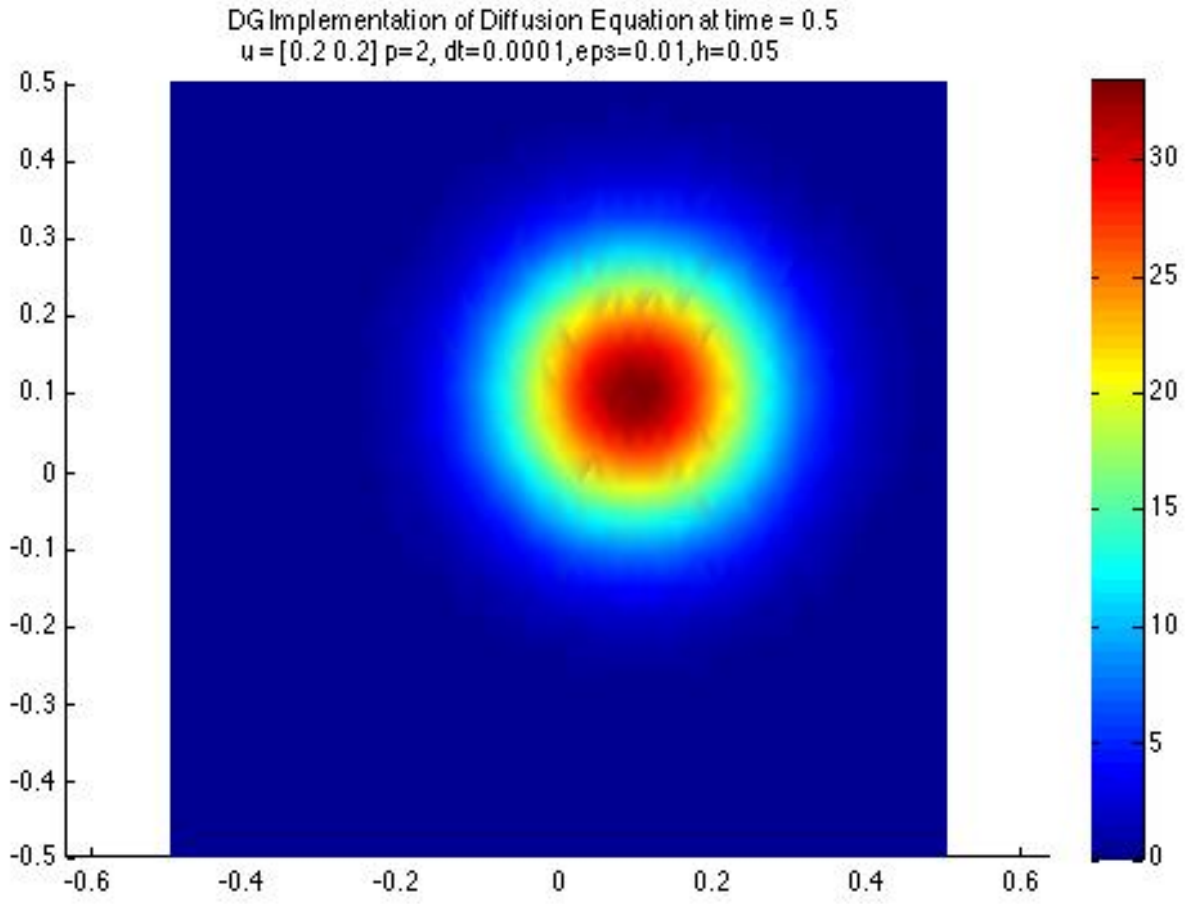

Figure 13: Solving the Advection-Diffusion Equation with a DG method in 2D using a Gaussian initial condition and advection velocity [0.2 0.2]

The solution agrees well with the expected solution, the peak diffused and advected approximately 0.1 units in each of the x_1 and x_2 directions.

VI. The Poisson Equation in 2D

The transition from an unsteady to a steady problem with a source term is similar to the 1D case. We extend our implementation from the 1D case as we did for advection-diffusion such that equations 12 and 13 become.

$$\begin{cases} MQ_1 = C_{avg_1}U - S_1U \\ MQ_2 = C_{avg_2}U - S_2U \end{cases} \quad (24a)$$

$$Mf = \varepsilon_1\left(C_{avg_1} + C_{BC_1}) - S_1\right)Q_1 + \ \varepsilon_2\left(C_{avg_2} + C_{BC_2}) - S_2\right)Q_2 \quad (24b)$$

$$Mf = \varepsilon_1\left(C_{avg_1} + C_{BC_1}) - S_1\right)M^{-1}\left(C_{avg_1} - S_1\right)U + \ \varepsilon_2\left(C_{avg_2} + C_{BC_2}) - S_2\right)M^{-1}\left(C_{avg_2} - S_2\right)U \quad (24c)$$

Figure 14 demonstrates excellent agreement between the 1D and 2D cases, where the 2D plot has analogous homogeneous Dirichlet boundary conditions along the x=0 and x = 0.5 boundaries. The upper figure shows the solution for the equation $\varepsilon_1\frac{\partial^2 u}{\partial x_1^2} + \varepsilon_2\frac{\partial^2 u}{\partial x_2^2} = f(x,y)$ where $\varepsilon_1 = 1$, $\varepsilon_2 = 0$ and $f = -\sin x$ restricting diffusion to the x_1 direction. The lower graph is the solution for the 1D problem $\varepsilon\frac{\partial^2 u}{\partial x^2} = f$ where $\varepsilon = 1$ and $f = -\sin x$. We expect the solution to the 1D problem is to represent a horizontal cross-section of the solution of the 2D problem, and observe that our expectation is met very well. This close agreement reinforces our confidence in both the 1D and 2D implementations.

.

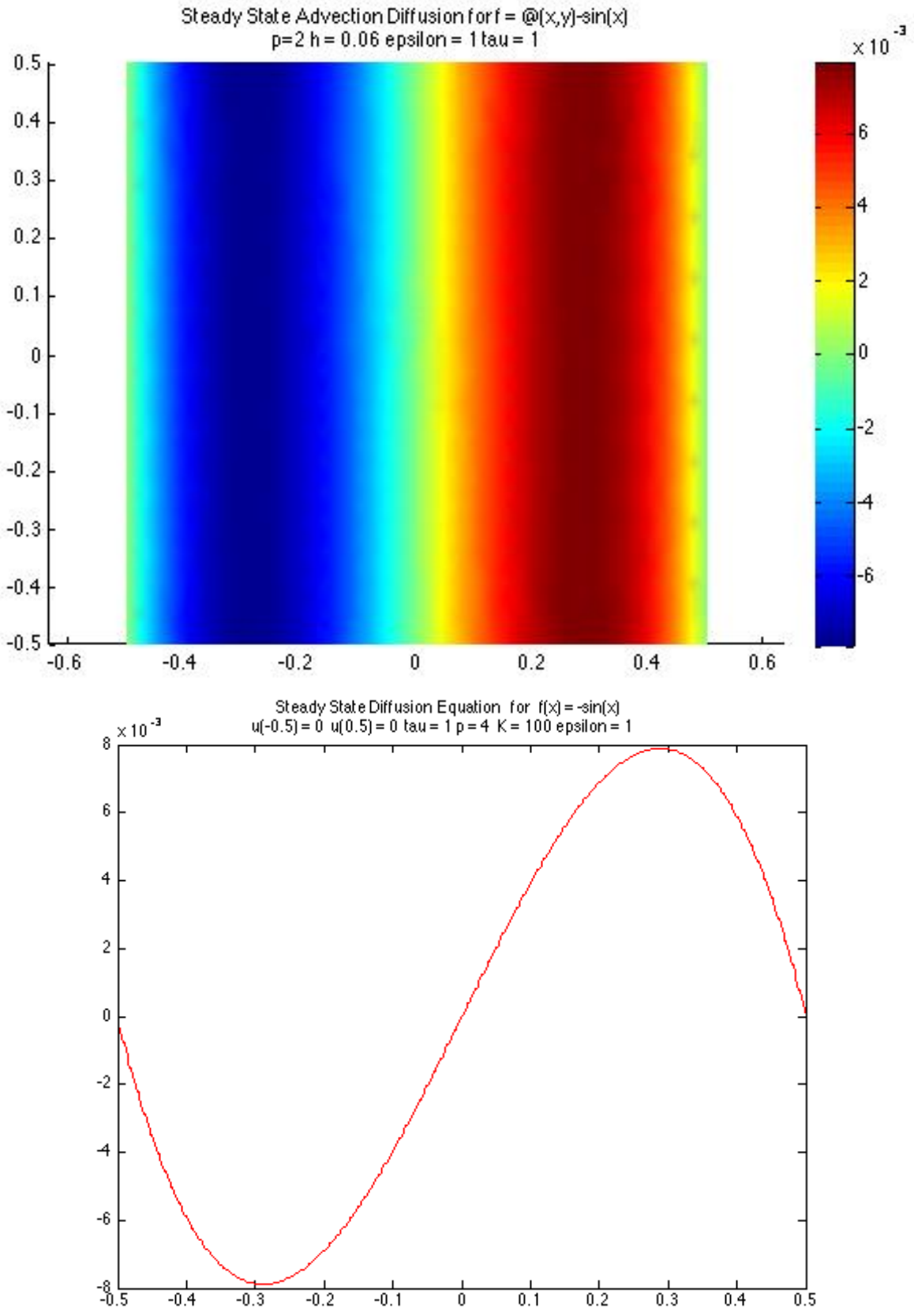

Figure 14: Solving the Poisson Equation on a 2D domain (top) (or steady state advection-diffusion equation with a non-zero forcing) for comparison with the 1D solution (bottom). The solutions agree very well, and can be verified analytically to be correct.

To incorporate advection into the implementation we modify equation 24(a-c)

$$\begin{cases} MQ_1 = C_{avg_1}U - S_1U \\ MQ_2 = C_{avg_2}U - S_2U \end{cases} \tag{25a}$$

$$Mf = \varepsilon_1\left(C_{avg_1} + C_{BC_1}) - S_1\right)Q_1 + \varepsilon_2\left(C_{avg_2} + C_{BC_2}) - S_2\right)Q_2 -$$

$$w_1\left(C_{avg_1} + C_{BC_1}) - S_1\right)U - w_2\left(C_{avg_2} + C_{BC_2}) - S_2\right)U - \hat{\tau}C_jU \tag{25b}$$

$$Mf = \varepsilon_1\left(C_{avg_1} + C_{BC_1}) - S_1\right)M^{-1}\left(C_{avg_1} - S_1\right)U + \varepsilon_2\left(C_{avg_2} + C_{BC_2}) - S_2\right)M^{-1}\left(C_{avg_2} - S_2\right)U -$$

$$w_1\left(C_{avg_1} + C_{BC_1}) - S_1\right)U - w_2\left(C_{avg_2} + C_{BC_2}) - S_2\right)U - \hat{\tau}C_jU \tag{25c}$$

Where $\hat{\tau} = \sqrt{\varepsilon_1^2 + \varepsilon_2^2}\tau$

Our final test problem has the same initial conditions as the advection-diffusion problem (Figure 12a), and an advection velocity of [5,5] (see Figure 15). The solution in Figure 15 seems reasonable, given that homogeneous Dirichlet boundary conditiona are applied the the exterior boundary in much the same way as in the 1D case.

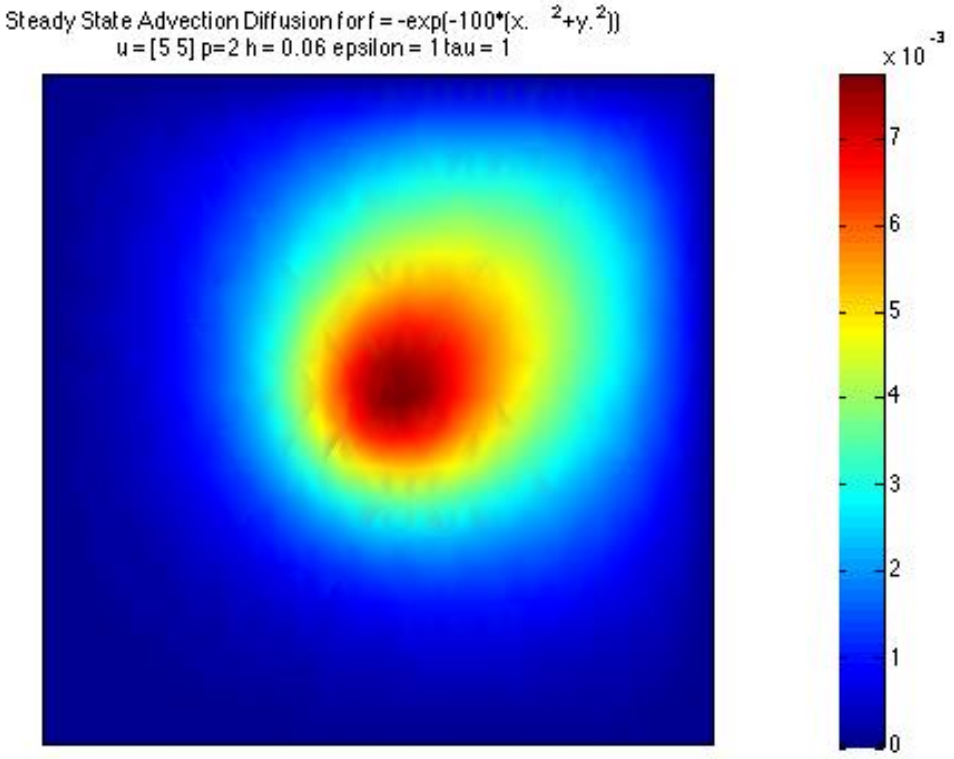

Figure 15: The steady State Advection Diffusion Equation with advection velocity w = 0.5,0.5

VII. Conclusions and Future Work

We have demonstrated the implementation of a DG finite element method to solve parabolic and elliptic partial differential equations. The specific techniques required for stability, as they related to the selected test problems are discussed, and the solutions for specific test cases could be computed analytically and/or verified with the H&W code.

We have described the implementation of Neumann, no penetration boundary conditions for the diffusion equation, and the implementation of homogeneous and inhomogeneous Dirichlet boundary conditions for the Poisson Equation.

During the project, several interesting topics were explored but, unfortunately, could not be fully illuminated. These topics, which may be interesting to pursue as part of future work with this code, include a more rigorous investigation of strongly versus weakly imposed boundary conditions for the Poisson equation, implementing inflow and outflow boundary conditions for the advection-diffusion equation in both the 1D and 2D domain, and ultimately incorporating viscous and pressure terms that represent to solve the Navier-Stokes equations with a DG finite element method.

References:

[1] J. S. Hesthaven and T. Warburton. Nodal Discontinuous Galerkin Methods: Algorithms, Analysis and Applications. Springer, 2008.

[2] H&W code can be found at www.nudg.org