Solution and Analysis of Space-Time Discontinuous Galerkin method applied to Coupled Advection-Diffusion Hyperbolic System of Equations on Periodic Domain

Christian Howard

May 9, 2013

Abstract

The solution and analysis of a system of advection-diffusion equations was investigated utilizing a Space-Time Discontinuous Galerkin formulation. A brief description of the formulation and implementation of such a solution method is explained. Verification that the solution of the hyperbolic system of equations is capable of approximating the parabolic solution was accomplished. An experimental convergence study was done to understand the effect of polynomial order within elements and mesh refinement on the convergence rate of the solution fields. The possible impacts of the results and other topics to look into based on the results are discussed.

Contents

1 Problem Statement **3**

2 Formulation **4**

2.1 Basis for Space-Time Discontinuous Elements 4

2.2 Obtaining the Weak Statement 4

3 Conservation, Implementation & Analysis **8**

3.1 Conservation Properties 8

3.1.1 Case 1 8

3.1.2 Case 2 8

3.2 Implementation 9

3.2.1 Code Language 9

3.2.2 Data Structures 9

3.2.3 Conditioning Fix 10

3.2.4 Enforcing Periodic Boundary Condition 10

3.3 Verification 11

3.3.1 Verification Discussion 15

3.4 Convergence Study 17

3.4.1 General Convergence Observations 23

3.4.2 Experimental Convergence Rate Discussion 24

4 Conclusion **25**

1 Problem Statement

Consider the following linear two-field problem acting on a one dimensional spacial dimension and in time with a the prescribed uniform fluid velocity field $\mathbf{a} = \mathrm{a}\mathbf{e}_x$:a > 0, uniform and isotropic diffusivity and relaxation tensors, $\boldsymbol{\kappa} = \kappa\mathbf{I}$:$\kappa > 0$ and $\boldsymbol{\tau} = \tau\mathbf{I} : \tau > 0$, and periodic boundary conditions. Find a pollutant concentration field u and the diffusive flux field $\mathbf{q} = q\mathbf{e}_x \ni$

$$\begin{aligned}
\dot{u} + (au + q)_{,x} &= 0 \; in \; \Omega := (0, 2\pi) \times (0, T) && (1)\\
q + \tau\dot{q} + (\tau a q + \kappa u)_{,x} &= 0 \; in \; \Omega && (2)\\
u|_{t=0} - u_0 &= 0 \; in \; (0, 2\pi) && (3)\\
q|_{t=0} - q_0 &= 0 \; in \; (0, 2\pi) && (4)\\
u(2\pi, \cdot) &= u(0, \cdot) && (5)\\
q(2\pi, \cdot) &= q(0, \cdot) && (6)
\end{aligned}$$

With the following as the initial conditions:

$$\begin{aligned}
u_0 &= sin(x)\\
q_0 &= -\kappa cos(x)
\end{aligned}$$

Extra information useful to the problem:

1. The diffusivity transport speed, c, is defined as the following

$$c = \sqrt{\frac{\kappa}{\tau}}$$

2. Define a dimensionless variable $H = \frac{|a|}{c}$
   (a) $H < 1$: Subcritical Flow
   (b) $H = 1$: Critical Flow
   (c) $H > 1$: Supercritical Flow

2 Formulation

2.1 Basis for Space-Time Discontinuous Elements

The Space-Time Discontinuous Galerkin formulation is quite different compared to typical Galerkin procedures due to the basis used for the elements. Each element has basis functions that aren't dependent on any values from outside elements. This means that continuity across elements isn't directly enforced, which can lead to discontinuities at element boundaries. Due to this, a special basis must be constructed to solve this problem. Given that, let a disjoint partition of Ω be constructed, denoted by $\mathcal{P}_h(\Omega)$, where the members of this partition are rectangular elements indexed as $\Omega_{jl} := (x_{j-1}, x_j) \times (t_{l-1}, t_l)$. The discrete discontinuous Galerkin space is then defined as the following:

$$\mathcal{V}_h := \left\{ v \in L^1(\Omega) : v|_{\Omega_{jl}} \in P^p(\Omega_{jl}) \forall \Omega_{jl} \in \mathcal{P}_h(\Omega) \right\}$$

Where $P^p(\Omega_{jl})$ is the space of space-time polynomials on Ω_{jl} that is complete to order p.

2.2 Obtaining the Weak Statement

Note| Since there are two equations that are coupled, there will need to be two independent sets of weighting functions used for the Galerkin weighted residual formulation. The following residual formulation is taken from [1].

Find $(u, q) \in \mathcal{V}_h \times \mathcal{V}_h \ni \Omega_{jl} \in \mathcal{P}_h(\Omega)$

$$\int_{\Omega_{jl}} \{ v[\dot{u} + (au + q)_{,x}] + w[q + \tau \dot{q} + (\tau a q + \kappa u)_{,x}] \} \, dx dt$$

$$+ \int_{\partial \Omega_{jl}} \{ [(va + w\kappa)(u^* - u) + (v + wa\tau)(q^* - q)]\mathbf{e}_x + [v(u^* - u) + \tau w(q^* - q)]\mathbf{e}_t \} \cdot \mathbf{n} ds$$

$$= 0 \; \forall (v, w) \in \mathcal{V}_h \times \mathcal{V}_h \tag{7}$$

After some work, this statement can be simplified to the following form:

$$\int_{\Omega_{jl}} [-\dot{v}u - v_{,x}(au+q) + wq - \dot{w}\tau q - w_{,x}(a\tau q + \kappa u)]dxdt$$

$$+\int_{t_{l-1}}^{t_l} [(va + w\kappa)u^* + (v + wa\tau)q^*]_{x_{j-1}}^{x_j} dt + \int_{x_{j-1}}^{x_j} [vu^* + \tau wq^*]_{t_{l-1}}^{t_l} dx$$

$$= 0 \; \forall (v,w) \in \mathcal{V}_h \times \mathcal{V}_h \tag{8}$$

Where the Space-Time element used for this simplified form has the following appearance:

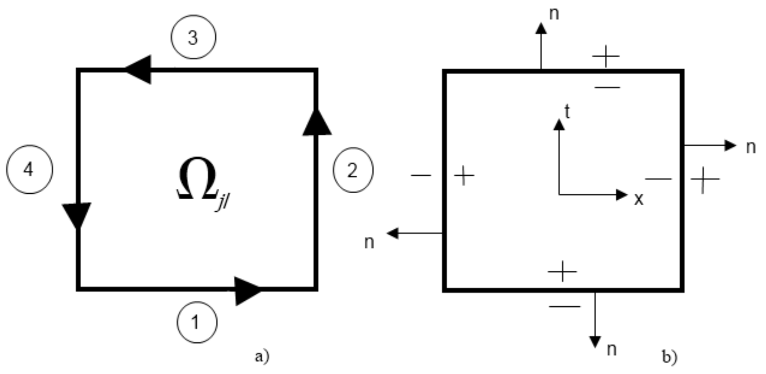

Figure W1: Space-Time Element Details
a) The element with the line integral orientation and definition of the separate line integral terms (1),(2),(3) and (4)
b) The element normals, as well as the positive and negative solution definitions for use in the flux quantities u^* and q^*

The flux quantities u^* and q^* are defined as the following:

$$(u^*, q^*) = \begin{cases} (u^-, q^-) & on\ faces\ (1)\ and\ (3) \\ (\frac{u^-+u^+}{2} + \frac{1}{c}\frac{q^- - q^+}{2}, \frac{q^-+q^+}{2} + c\frac{u^- - u^+}{2}) & on\ faces\ (2)\ and\ (4) \end{cases}$$

Plugging in these definitions of the fluxes into equation (8) gives the following resulting terms of equations:

$$\begin{aligned}
a_0(u_k^e, v_k) &= \frac{a+c}{2}\int_{t_{l-1}}^{t_l}(v_k u_k^e|_{x_j} dt + \frac{a-c}{2}\int_{t_l}^{t_{l-1}}(v_k u_k^e|_{x_{j-1}} dt \\
&\quad -\int_{x_j}^{x_{j-1}}(v_k u_k^e|_{t_l} dx - \int_\Omega \dot{v}_k u_k^e d\Omega - a\int_\Omega v_{k,x} u_k^e d\Omega \\
a_{-1}(u_k^{e-1}, v_k) &= \frac{a+c}{2}\int_{t_l}^{t_{l-1}}(v_k u_k^{e-1}|_{x_{j-1}} dt \\
a_1(u_k^{e+1}, v_k) &= \frac{a-c}{2}\int_{t_{l-1}}^{t_l}(v_k u_k^{e+1}|_{x_j} dt \\
b_0(q_k^e, v_k) &= \frac{1+\frac{a}{c}}{2}\int_{t_{l-1}}^{t_l}(v_k q_k^e|_{x_j} dt + \frac{1-\frac{a}{c}}{2}\int_{t_l}^{t_{l-1}}(v_k q_k^e|_{x_{j-1}} dt - \int_\Omega v_{k,x} q_k^e d\Omega \\
b_{-1}(q_k^{e-1}, v_k) &= \frac{1+\frac{a}{c}}{2}\int_{t_l}^{t_{l-1}}(v_k q_k^{e-1}|_{x_{j-1}} dt \\
b_1(q_k^{e+1}, v_k) &= \frac{1-\frac{a}{c}}{2}\int_{t_{l-1}}^{t_l}(v_k q_k^{e+1}|_{x_j} dt \\
c_0(u_k^e, w_k) &= \frac{\kappa+ac\tau}{2}\int_{t_{l-1}}^{t_l}(w_k u_k^e|_{x_j} dt + \frac{\kappa-ac\tau}{2}\int_{t_l}^{t_{l-1}}(w_k u_k^e|_{x_{j-1}} dt - \kappa\int_\Omega w_{k,x} q_k^e d\Omega \\
c_{-1}(u_k^{e-1}, w_k) &= \frac{\kappa+ac\tau}{2}\int_{t_l}^{t_{l-1}}(w_k u_k^{e-1}|_{x_{j-1}} dt \\
c_1(u_k^{e+1}, w_k) &= \frac{\kappa-ac\tau}{2}\int_{t_{l-1}}^{t_l}(w_k u_k^{e+1}|_{x_j} dt \\
d_0(q_k^e, w_k) &= \frac{a\tau+\frac{\kappa}{c}}{2}\int_{t_{l-1}}^{t_l}(w_k q_k^e|_{x_j} dt + \frac{a\tau-\frac{\kappa}{c}}{2}\int_{t_l}^{t_{l-1}}(w_k q_k^e|_{x_{j-1}} dt \\
&\quad -\tau\int_{x_j}^{x_{j-1}}(v_k q_k^e|_{t_l} dx + \int_\Omega w_k q_k^e d\Omega - \tau\int_\Omega \dot{w}_k q_k^e d\Omega - a\tau\int_\Omega w_{k,x} q_k^e d\Omega \\
d_{-1}(q_k^{e-1}, w_k) &= \frac{a\tau+\frac{\kappa}{c}}{2}\int_{t_l}^{t_{l-1}}(w_k u_k^{e-1}|_{x_{j-1}} dt
\end{aligned}$$

$$\begin{aligned} d_1(q_k^{e+1}, w_k) &= \frac{a\tau - \frac{\kappa}{c}}{2} \int_{t_{l-1}}^{t_l} (w_k q_k^{e+1}|_{x_j} dt \\ F(v_k) &= \int_{x_{j-1}}^{x_j} (v_k u_{k-1}^e|_{t_{l-1}} dx \\ G(w_k) &= \tau \int_{x_{j-1}}^{x_j} (w_k q_{k-1}^e|_{t_{l-1}} dx \end{aligned}$$

Where the following statements are true:

1. k references the current slab being solved
2. e references the current element being looked at in the slab
3. $e-1$ and $e+1$ are the left and right neighboring elements, respectively
4. $k-1$ references the element under the current element, which is in the most recently solved slab

And the local equations are defined as:

$$\sum_{i=-1}^{1} a_i(u_k^{e+i}, v_k) + \sum_{i=-1}^{1} b_i(q_k^{e+i}, v_k) = F(v_k) \tag{9}$$

$$\sum_{i=-1}^{1} c_i(u_k^{e+i}, w_k) + \sum_{i=-1}^{1} d_i(q_k^{e+i}, w_k) = G(w_k) \tag{10}$$

3 Conservation, Implementation & Analysis

3.1 Conservation Properties

To show the conservation properties of the Space-Time Discontinuous Galerkin method utilized for this problem, two cases are considered:

1. $w = 0$ & $v_{jl}|_{\Omega_{rs}} = \delta_{jr}\delta_{ls}$, thus $v_{jl} = 1$ only on element Ω_{jl} and 0 elsewhere
2. $v = 0$ & $w_{jl}|_{\Omega_{rs}} = \delta_{jr}\delta_{ls}$, thus $w_{jl} = 1$ only on element Ω_{jl} and 0 elsewhere

3.1.1 Case 1

Plugging in the weight functions defined for this case into equation (8) leads to the following expression:

$$\int_{t_{l-1}}^{t_l} [au^* + q^*]_{x_{j-1}}^{x_j} dt + \int_{x_{j-1}}^{x_j} [u^*]_{t_{l-1}}^{t_l} dx = 0$$

The equation above shows that, essentially, the flux for the concentration field within a given element is conserved since the flux integrals sum to zero. This is expected since there is no source/sink effecting the concentration field directly, so the flux into element Ω_{jl} should equal the flux out of it. Based on these details, it is seen the SDG ensures conservation for the concentration field on an element by element basis.

3.1.2 Case 2

$$\int_{\Omega_{jl}} q \, d\Omega + \int_{t_{l-1}}^{t_l} [\kappa u^* + a\tau q^*]_{x_{j-1}}^{x_j} dt + \int_{x_{j-1}}^{x_j} [\tau q^*]_{t_{l-1}}^{t_l} dx = 0$$

The equation above shows that the net flux across the boundaries will be dependent on the volume integral of the source term. This then shows conservation because if the source was to be zero, the flux terms will sum to zero, meaning the flux in is equal to the flux out. While the source term is nonzero, this then shows that the flux in and out will be based on purely on the source within the element. This implies conservation of the SDG method on element Ω_{jl}.

3.2 Implementation

3.2.1 Code Language

The code developed to solve this problem was written in C++ for a compromise between efficiency, readability and scalability.

3.2.2 Data Structures

The primary data structures utilized for the solution of the problem at hand are the following:

1. Space-Time Element Class
    (a) Holds the solution coefficients for the concentration and flux fields on this element
    (b) Holds the Jacobian values used for the volume and line integrals
    (c) Held functions for mapping between global coordinates and natural coordinates
    (d) Holds functions for obtaining geometric properties of the element, such as width and height and coordinates for the corners
    (e) Holds functions to evaluate single basis functions and their derivatives, as well as evaluate the concentration or flux field within the element and their derivatives
2. Grid Class
    (a) Holds a dynamic 2D array of Space-Time elements
    (b) Holds general line and volume integral functions
    (c) Holds functions that represent the integral terms for the local equations specified earlier
    (d) Holds functions for evaluating an element's u^* and q^* on any given edge
    (e) Holds functions for obtaining the error indicators
    (f) Holds functions for generating the global stiffness matrix and global load vector

   (g) Holds functions to write the solution field across all the elements into separate text files for each field

3. Simulation Class

   (a) Holds a grid object

   (b) Holds a function for running a SDG simulation for a given N_x, N_t, set of initial conditions for the concentration and flux fields, span of the global space and time domain and the desired polynomial order within elements

3.2.3 Conditioning Fix

It was stated that the basis functions for this problem would be of the following form:

$$\{\phi_{00}, \phi_{10}, \phi_{01}, \phi_{20}, \phi_{11}, \phi_{02}, ...\} = \{c_{00}, c_{10}\zeta, c_{01}\eta, c_{20}\zeta^2, c_{11}\zeta\eta, c_{02}\eta^2, ...\}$$

Where $c_{\alpha\beta}$ are found such that the condition number of the stiffness matrix is sufficiently low. To obtain such a value for the $c_{\alpha\beta}$, it was given that one should choose $c_{\alpha\beta}$ such that $||\phi_{\alpha\beta}||_{L^2(\Omega_{jl})} = 1$. The following general formula was obtained to obtain these coefficients:

$$c_{\alpha\beta} = \sqrt{\frac{(2\alpha+1)(2\beta+1)}{\Delta x \Delta t}}$$

3.2.4 Enforcing Periodic Boundary Condition

To enforce the periodic boundary condition, one must look back to equations (9) and (10), where their expanded forms are given as:

$$\begin{aligned} a_{-1}(u_k^{e-1}, v_k) + a_0(u_k^e, v_k) + a_1(u_k^{e+1}, v_k) + b_{-1}(q_k^{e-1}, v_k) + b_0(q_k^e, v_k) + b_1(q_k^{e+1}, v_k) &= F(v_k) \\ c_{-1}(u_k^{e-1}, w_k) + c_0(u_k^e, w_k) + c_1(u_k^{e+1}, w_k) + d_{-1}(q_k^{e-1}, w_k) + d_0(q_k^e, w_k) + d_1(q_k^{e+1}, w_k) &= G(w_k) \end{aligned}$$

Based on these equations, and the fact e spans from $e = 0, 1, 2, ..., N_x - 1$, the following actions were taken to enforce the periodic boundary conditions:

1. If $e = 0$, then $e - 1 = N_x - 1$

2. If $e = N_x - 1$, then $e + 1 = 0$

3.3 Verification

After the design and implementation of the code to solve this problem, verification of the code and SDG solution method had to be done. To accomplish this task, an attempt to make the hyperbolic solution converge to the parabolic limit occurred. The parameters used for the simulations were:

1. A polynomial order of $p = 1$
2. $\kappa = 1$
3. $\tau_c = 1/(4\pi^2)$
4. $a = 2\pi$
5. $H = 1 \rightarrow c = 2\pi$
6. $T = 5$
7. Simulations ran with $\tau = C\tau_c$ for $C = \frac{1}{2}, \frac{1}{4}, \frac{1}{16}, \frac{1}{256}$
8. $N_t = 10T/(\pi^2\tau)$
9. For 4 cases, simulations ran with $N_x = 10$
10. For 1 added case when $C = \frac{1}{256}$, a simulation ran with $N_x = 100$
11. Initial conditions
    - (a) $u_0(x) = sin(x)$
    - (b) $q_0(x) = -\kappa cos(x)$

The note to make before moving forward is that the parabolic solution to these equations, which is used for comparison in the figures obtained, is given below:

$$u(x,t) = e^{-\kappa t}u_0(x - at) = e^{-t}sin(x)$$

After running the various simulations and extracting their data, the following figures were producted at times t = 1, 2, 3, 4 and 5 seconds:

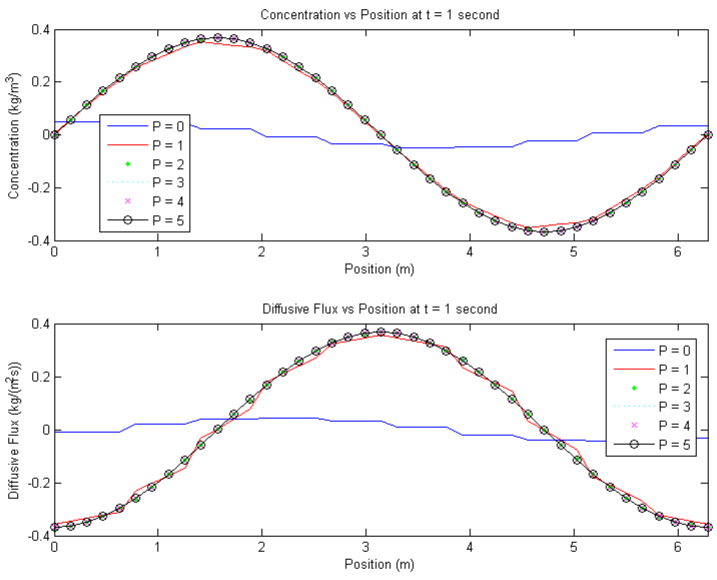

Figure FV0: Polynomial Interpolation differences at t = 1 second

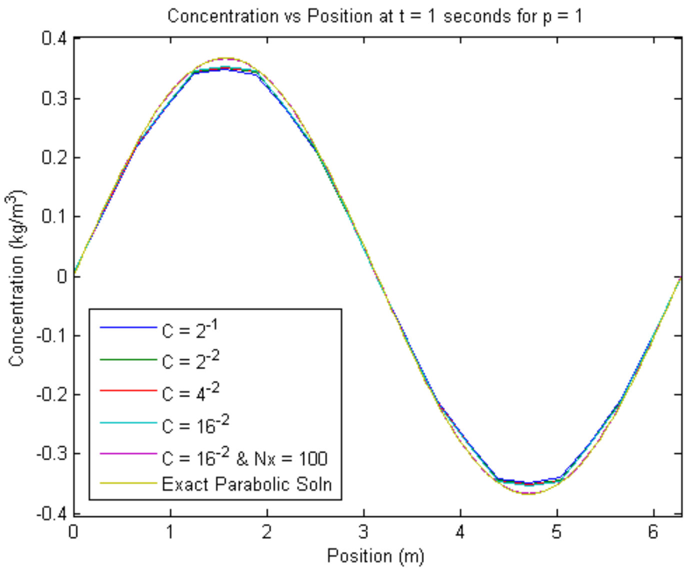

Figure FV1: Concentration vs Position at t = 1 second

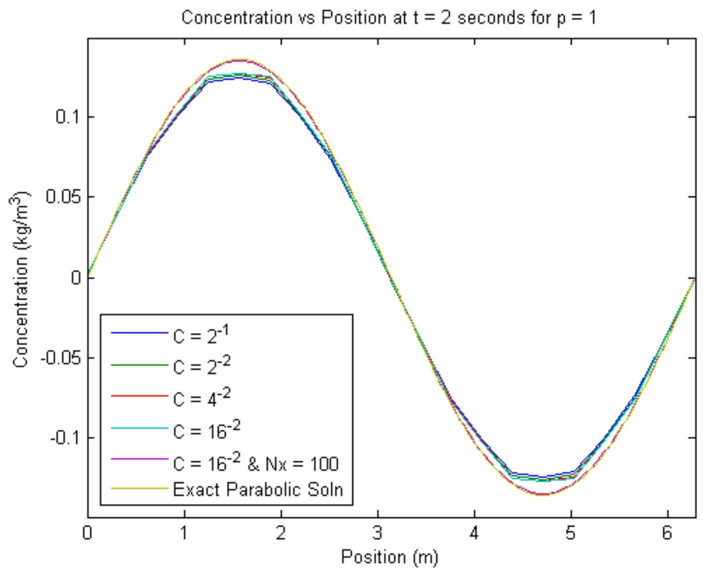

Figure FV2: Concentration vs Position at t = 2 seconds

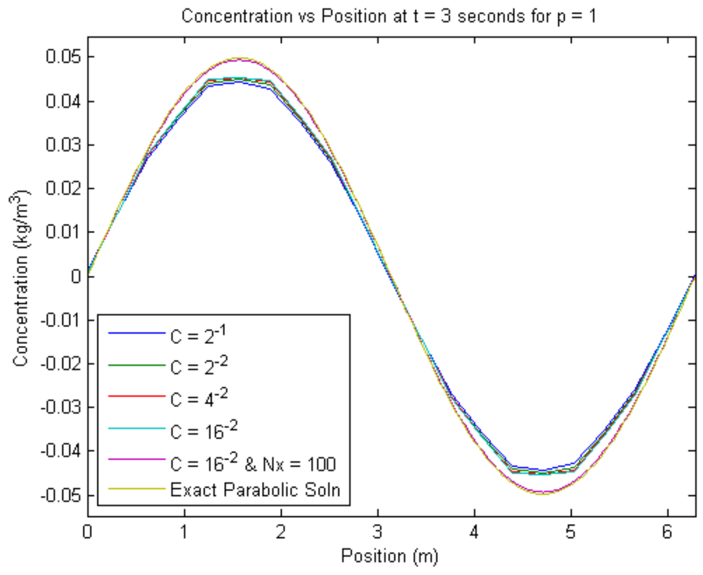

Figure FV3: Concentration vs Position at t = 3 seconds

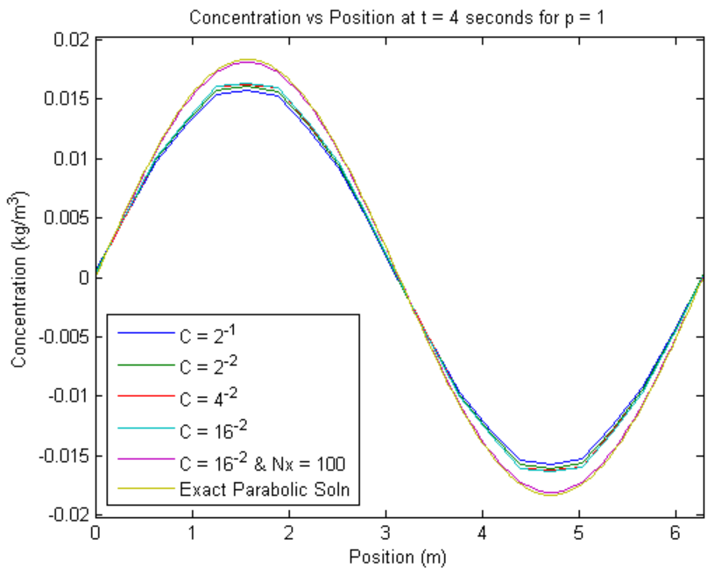

Figure FV4: Concentration vs Position at t = 4 seconds

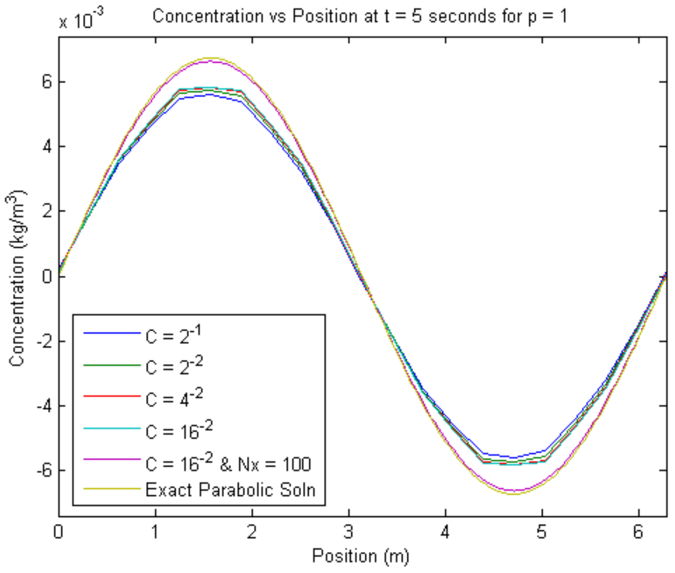

Figure FV5: Concentration vs Position at t = 5 seconds

3.3.1 Verification Discussion

After taking some moments to view the figures, a few obvious characteristics become known. First, it is very easy to see that as time moved on, the approximate solutions began to drop in amplitude more so than the parabolic solution. This is interesting to see because it seemed that, at least initially, most of the hyperbolic solutions matched up pretty well with the parabolic solution. However, time moved forward and the divergence of the approximate solutions from the exact parabolic solution became more and more evident.

One other thing to look at with respect to this thought is the differences in the solutions with the varying values of C. What one might notice is that solutions with larger values of C, which are also solutions furthest from the parabolic limit, dropped more in their amplitude than the solutions that were closer to a the parabolic problem. This difference appears to show that the more hyperbolic the equations, the faster the solution dampened out. This also shows that as τ was decreased, which made the system of equations more parabolic, the approximate SDG finite element solutions approached the parabolic limit. Now, even if they approached it, one can still see in **Figure FV5**, for example, that the hyperbolic solution for a very small τ and refined mesh still isn't perfectly aligned with the parabolic solution. This implies that even for very small τ, if the equations are any bit hyperbolic, they will diverge from the exact parabolic solution to some extent later in time.

Some other interesting comparisons to make are comparing the solutions of when $C = 16^{-2} = \frac{1}{256}$ for $N_x = 10$ and $N_x = 100$. These solutions had the same value for N_t, so the Space-Time mesh was very refined in both cases in the time dimension. However, the spacial mesh refinement obviously plays a large role in the accuracy of the method, as the figures show. First thing to note is that the $N_x = 10$ case leads to having some fairly choppy solutions, with very obvious discontinuous slopes at element interfaces. Although the solution for $N_x = 10$ approximated fairly well at early times, based off of **Figure FV1**, one can see that as time moved on, the solution fell away from the parabolic solution much more so than the $N_x = 100$ solution. To make some comments on the $N_x = 100$ solution, one can see that the high refinement actually generates a solution that appears quite smooth and thus fits to the sinusoidal solution curve quite well. Some obvious differences, again, are that this highly refined solution in space and time diverged much less from the parabolic solution than that solution utilizing $N_x = 10$. The results give strong evidence that refinement in space and time must be done to minimize error in the solution field and, although refinement in one dimension is better than nothing, it surely isn't enough.

Now, in terms of computational expense, a few comments can be made based on the results and resources used to construct these simulations. First, based on the formula for N_t given earlier, it is simple to see that as C got small, the mesh got very refined and thus had to utilize a lot of memory and time to work through the solution. One could also, after some recent comments, say that refinement of the spacial dimension is required to obtain a sufficiently accurate result. The combination of these two requirements, so then the hyperbolic model can approximate the parabolic one, can lead to enormous simulations and possibly lead to a lack in memory resources to finish the job. Now, conclusions on whether this is better than solving the parabolic system directly is something one would have to look into. At the present time, knowledge about the difference in the computation required to solve the parabolic system is unknown. With that, it seems this might be a smart question to look into down the road.

After the discussion above and the figures obtained, one might see that the hyperbolic solutions are capable of approximating the parabolic solution quite well at earlier times, while losing that parabolic character as time moves on. What this shows is that the hyperbolic nature of the solution doesn't come into play as much early on in the solution field. However, as time proceeds, the hyperbolic nature of the solution field becomes more and more evident. This realization leads to a possibly odd idea. If in fact the hyperbolic problem was found to be cheaper than the parabolic problem, one might think it could be intelligent to utilize a hyperbolic approximation of the parabolic equations for small time frames and then switch over after that to solving the parabolic equation directly. The logic, if the hyperbolic problem was cheaper to solve, seems to be a possibly intelligent move. The thought process is that after the earlier times, the parabolic nature is to smooth and dampen the solution field anyways. This means that after the earlier time frame, it would be easier to solve the parabolic problem directly, especially if one utilized an adaptive time integrator to move through time. With this, the overall time and resources used for the problem might be minimized.

3.4 Convergence Study

A convergence study was accomplished on the SDG solution method with the problem at hand. Mesh refinement was done at various polynomial orders within elements to see the difference in convergence rates for the variety of polynomial orders. The orders used for this study were polynomial orders p = 0, 1, 2 & 3. A table of the error indicator values and found convergence rates accompanied by a pair of hp-convergence plots are shown below and figures of the various solutions at time $T = 2$. Note that although the figures just show the N_x is being doubled, N_t is for each approximation as well, with a starting value of $N_t = 320$.

Polynomial Order	N_x	N_t	ζ^u	ζ^q
0	10	320	2.19641	55.6964
0	20	640	3.19704	80.7534
0	40	1280	4.21895	108.344
0	80	2560	5.0508	130.583
1	10	320	1.23482	10.7452
1	20	640	0.721079	8.40573
1	40	1280	0.403697	5.55747
1	80	2560	0.215252	3.21516
2	10	320	0.1427	1.74891
2	20	640	0.0374825	0.498069
2	40	1280	0.00977865	0.140258
2	80	2560	0.00251122	0.037682
3	10	320	0.00819868	0.0987838
3	20	640	0.00108047	0.0143293
3	40	1280	0.000139279	0.00196368
3	80	2560	0.0000177078	0.000258194

Table TC1: Error Indicator Values

Polynomial Order	Concentration Convergence Order	Flux Convergence Order
0	-.4005	-.4098
1	0.8401	0.5802
2	1.9428	1.8455
3	2.9516	2.8599

Table TC2: Experimental Convergence Rates

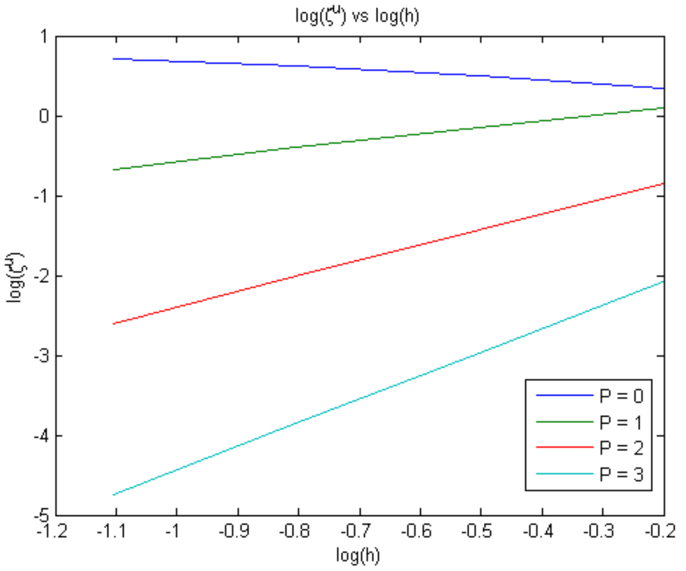

Figure FC1: $\log(\zeta^u)$ vs log(h), where h $= 2\pi/N_x$

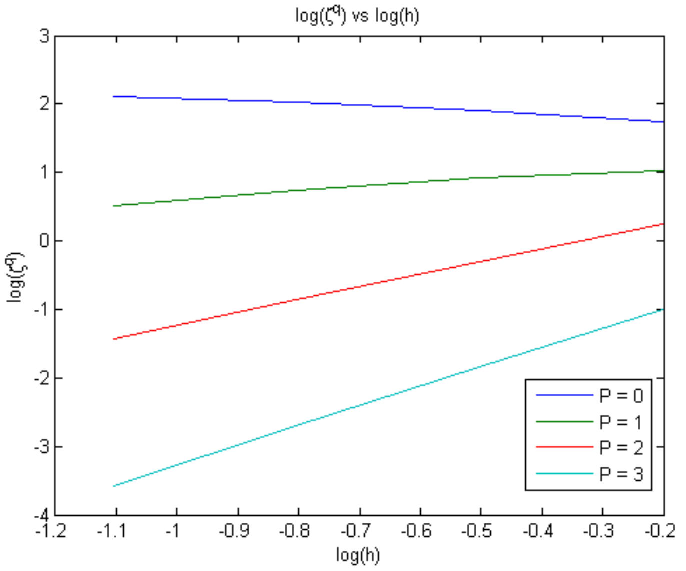

Figure FC2: $\log(\zeta^q)$ vs log(h)

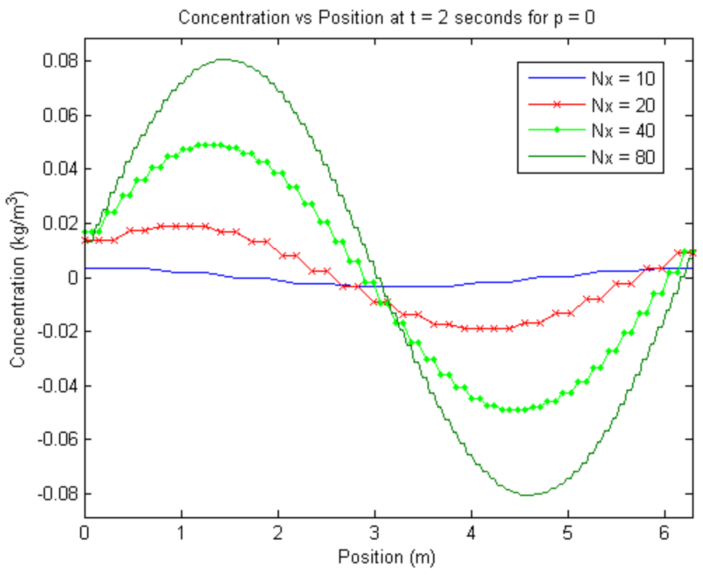

Figure FC3: Concentration vs Position for $N_x = 10, 20, 40, 80$ and $p = 0$

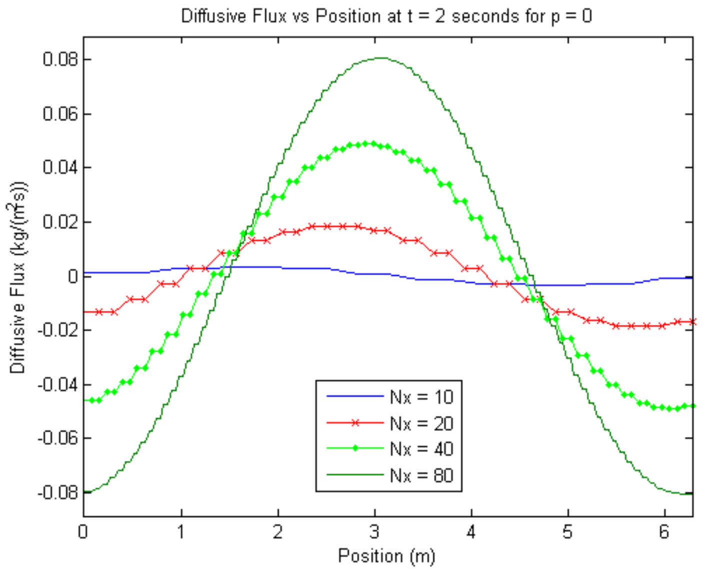

Figure FC4: Diffusive Flux vs Position for $N_x = 10, 20, 40, 80$ and $p = 0$

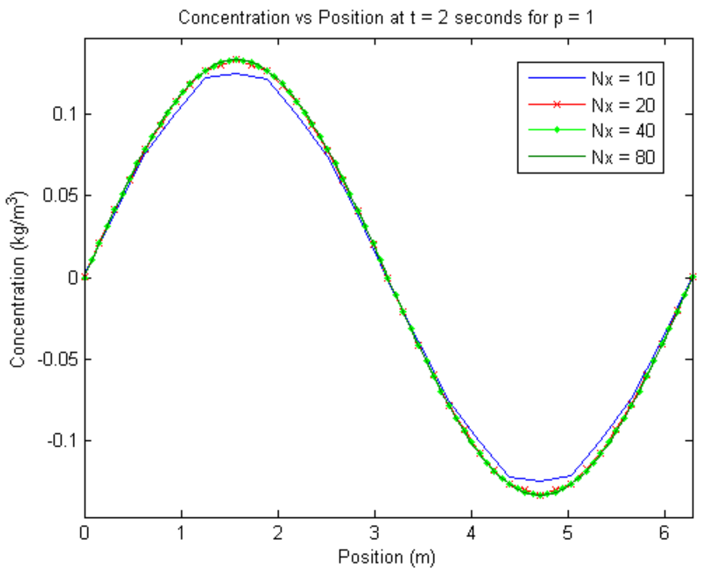

Figure FC5: Concentration vs Position for $N_x = 10, 20, 40, 80$ and $p = 1$

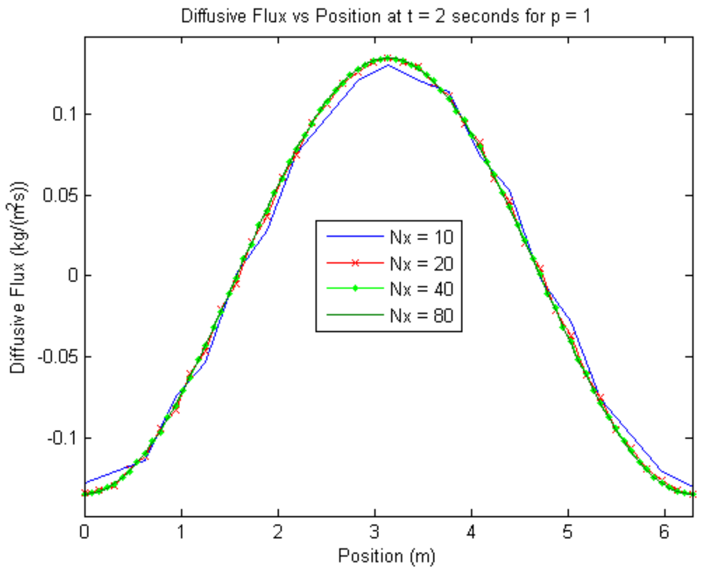

Figure FC6: Diffusive Flux vs Position for $N_x = 10, 20, 40, 80$ and $p = 1$

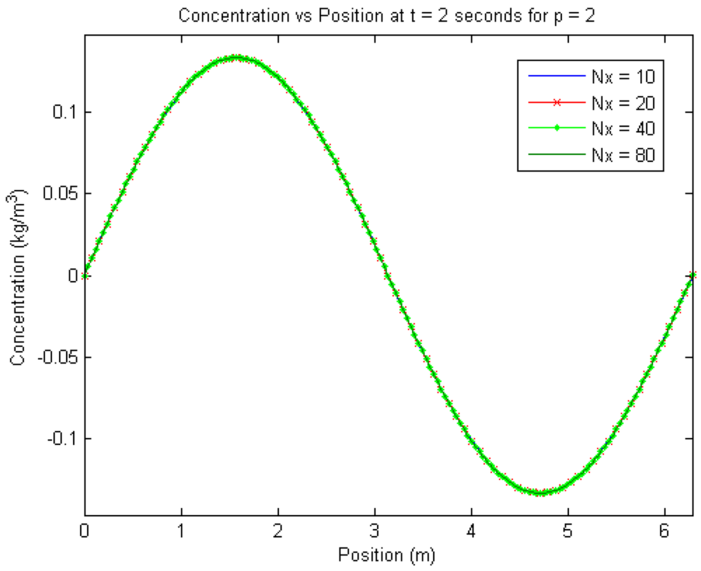

Figure FC7: Concentration vs Position for $N_x = 10, 20, 40, 80$ and $p = 2$

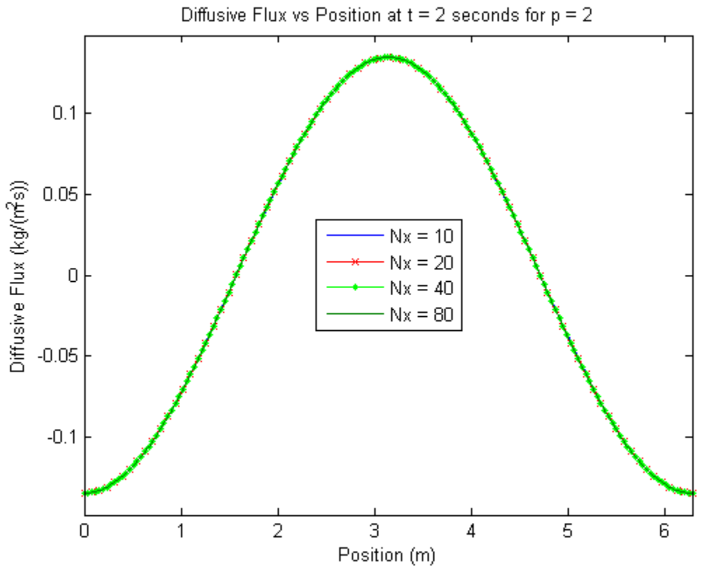

Figure FC8: Diffusive Flux vs Position for $N_x = 10, 20, 40, 80$ and $p = 2$

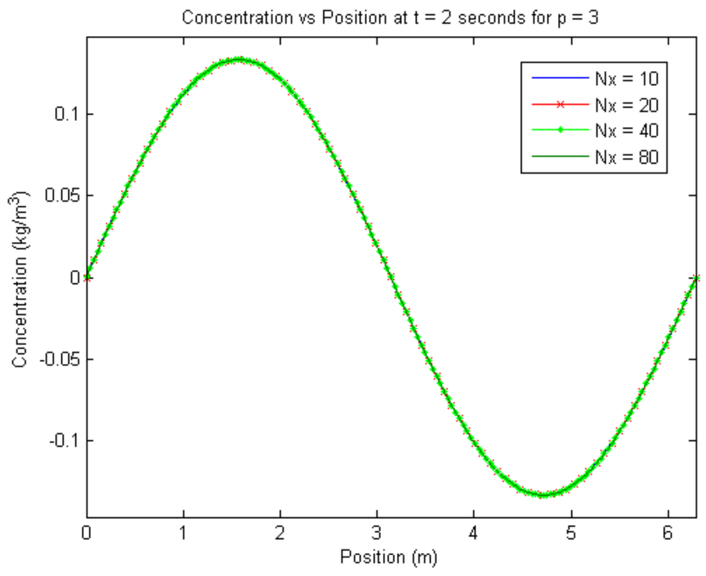

Figure FC9: Concentration vs Position for $N_x = 10, 20, 40, 80$ and $p = 3$

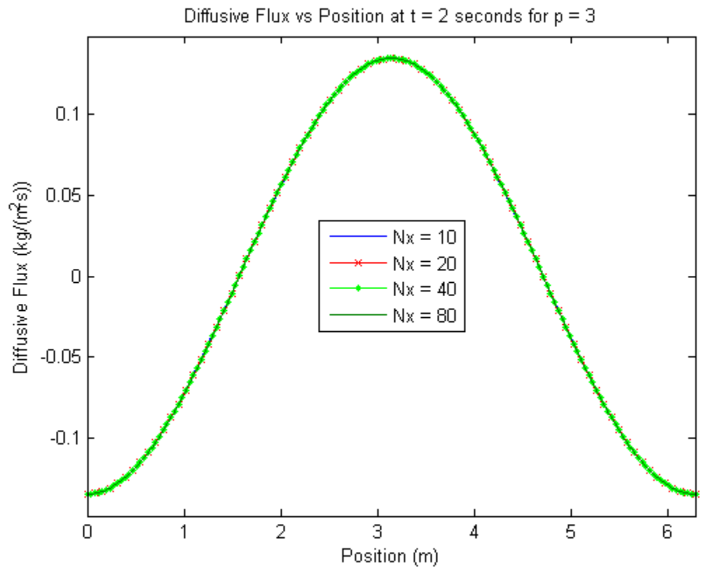

Figure FC10: Diffusive Flux vs Position for $N_x = 10, 20, 40, 80$ and $p = 3$

3.4.1 General Convergence Observations

The obtained figures and data found in the table gives some surprising, yet quite interesting results.

Looking first at the polynomial order of 0, the results came with initial surprise. Using constant elements appear to not only give bad results, according to the error indicators, but make the solution worse as the mesh is refined. This was surprising due to observation of plots showing the solution curves getting better as the mesh was refined. The first thing one might point out when viewing these results is to say that, when observing equation (8) from earlier, most of the terms in this equation go to zero due to the derivatives involved. Due to so many of these terms dropping out, the equations end up not acting like the original system, and so things appear to be wrong according to the error indicators. This result brings up curiosity as to whether the error indicators utilized were the best for such a polynomial order.

One last comment to make that is interesting is a result from analyzing the terms for the error indicator expression. What one might notice is that even if the volume terms in R^u_{jl} and R^q_{jl} go to zero, meaning the original equations are satisfied, if there is a big difference between the Riemenn values at a flux interface and the values of the element's solution at that same flux interface, the error indicator will return a larger value. This realization is made because for solution schemes utilizing p = 0 as the complete polynomial basis order, the difference between the Riemann value and the element's value of the solution will be larger compared to higher order elements. This leads to the conclusion that the error indicators may show divergent behavior, even if the equations themselves are satisfied.

For polynomial order of 1, the results did begin to converge, according to the error indicators. Though this order was converging, the convergence rate was pretty slow. Such results appear to stem from adding more degrees of freedom such that not all the terms would be zero when a derivative was taken, but enough were that good resolution couldn't be obtained.

For polynomial orders 2 and 3, it is great to see the solution converging faster than the p = 0 and p = 1 results, as well as starting off with what appears to be much smaller error.

3.4.2 Experimental Convergence Rate Discussion

The results in the **Table TC2** were quite interesting. The first concern was looking at the convergence rates for a polynomial order of 0. The negative convergence rates in fact convey the message that a p = 0 scheme doesn't converge to the correct solution. Based off of observations in the verification stage of this project, one could quickly see the p = 0 approximate solutions dampened out much quicker than the higher order versions. This quicker dampening seems to be an indicator that the p = 0 scheme doesn't converge and may be a visual explanation to these results.

Next, looking at the converge rate for the linear basis gave some unique results. First thing to notice is the convergence rate for the concentration solution and the convergence rate for the flux solution aren't at all close to being the same. In fact, one is around 45% different than the other. This was an interesting thing to note since the higher order elements and even the p = 0 elements gave relatively similar convergence rates for both fields. Anyways, this difference was unique and an explanation for such a result would be interesting to look into. One last comment is to notice that the convergence rate for the concentration field is fairly close to the order of the polynomial, which is p = 1, while the other is obviously a bit smaller.

Looking at polynomial orders 2 and 3, similar results are found for them both. First, the convergence rates for both fields are roughly the same. The next comment is that the convergence rates are also very close to the polynomial order used. So, in the case of p = 2, the convergence rates of 1.9428 for the concentration field and 1.8455 for the flux field are very close to p = 2. A similar thing is observed for p = 3.

Within these results, it has been interesting to see the correlations for the convergent elements and then speculate at the discrepancies for the divergent elements.

4 Conclusion

Over the course of this project, a variety things have occurred. Software to solve a system of hyperbolic advection-diffusion equations using the Space-Time Discontinuous Galerkin method was constructed. A verification of the conservation properties of the SDG formulation for this problem and a verification of the solution fields generated was accomplished as well. Lastly, an *hp* convergence study was done for the SDG formulation applied to this problem. After the accomplishment of these tasks, a few key things were discovered and enforced. First, it became apparent that the hyperbolic solution was capable of being a good parabolic solution approximation at very small times, but this property died out as time got larger. Second, to approximate the parabolic solution well, spacial and temporal refinement must be done together to minimize the error across the solution field. Third, it was found that using constant shape functions within the elements lead to divergent solution fields as the mesh was refined, for reasons that aren't totally certain. However, after increasing polynomial interpolants from that point and onward, the solution field was convergent and increasingly convergent as the polynomial order increased. For the generally convergent schemes, the experimental results appeared to push that the both the concentration and flux solution fields were roughly $O(h^p)$ convergent, where p is the polynomial order of the elements used, though still a little less convergent than this assertion. One more thing to add is that the flux field convergence rate was found to be slightly smaller than that of the concentration field.

This project has lead to some interesting data and results, as well as some ideas of where future work could head. One topic to consider is whether or not solving a parabolic problem directly is more work than solving the approximate hyperbolic problem. If these results were to show that the hyperbolic solution can be cheaper but with good accuracy, the next step might be to see if one might be able to utilize approximate hyperbolic solutions early on in a parabolic simulation, since this is the regime the hyperbolic solution is closest to the parabolic one, and then switch over to solving the parabolic problem directly after the hyperbolic solution begins to diverge. This idea could lead to saved resources and time, which can be quite meaningful in large-scale simulations. One last idea to consider would be trying to experiment with a different basis than the one utilized in the project and see if there is any particular basis that gives better results for the SDG formulation than the one utilized here.

References

[1] Haber, R.,*Term Project, TAM 574 - Spring 2013: Spacetime discontinuous Galerkin method for a hyperbolic advection-diffusion problem*,2013.