THE LINE-BASED DISCONTINUOUS GALERKIN METHOD FOR EQUATIONS OF FLUID DYNAMICS*

MARISSA RAMIREZ ZWEIGER†

Abstract. In this study, we examine the line-based discontinuous Galerkin (LDG) method for the Euler equations. This method is an extension of the standard discontinuous Galerkin (DG) method for the same equations. Like the standard DG method, LDG is advantageous for convection-dominated problems with complex geometries, is highly parallelizable, and can be easily made higher order for increased accuracy. The line-based component takes advantage of the low number of connectivities in the problem by treating each dimension independently, increasing computational speed. In this paper we will present a LDG method for pure convection and convection-diffusion, finally building up to the Euler equations. Ultimately, we plan to introduce shocks using artificial viscosity, and test the efficiency and accuracy of our method. This work is applicable to any instances of a shock in a fluid, including but not limited to those problems occurring in the fields of aerospace engineering, nuclear engineering, and computational fluid dynamics.

Key words. Discontinuous Galerkin, Shock Capturing, Euler Equations, Convection-Diffusion

1. Introduction. Mathematics is the language in which we express the physical phenomena of our world. Since as early as the ancient Greeks, when Aristotle noted the relationship between mass and velocity, we have been explaining physics in terms of math. As our knowledge of the physical world has increased, so has the complexity of these mathematical expressions. In modern physics and engineering many phenomena such as heat, sound, electrodynamics, fluid flow, and quantum mechanics can be described by partial differential equations (PDEs). PDEs are notoriously difficult to solve. Many have no analytical solution, meaning they cannot be solved by hand. Instead engineers and scientists use sufficiently accurate approximations generated by computer algorithm, referred to as numerical solutions. The field of numerical analysis (to which the work contained in this paper belongs) is dedicated to the development of these algorithms.

One area of physics that is heavy in partial differential equations is fluid dynamics. Any liquid or gas is considered a fluid, and fluid dynamics is concerned with the movement, or flow, of these fluids. The air around the wings of an airplane, the water flowing through your sink, and the air heating up in your oven are all governed by the equations of fluid dynamics. Arguably the most famous of these equations are the Navier-Stokes equations. They describe the motion of a viscous fluid, and are used to model the flow of water in a pipe, the currents of the ocean, and the airflow around an airplane wing, among other things. They have proved to be quite perplexing to mathematicians, so much so that proving the existence and smoothness of a solution to the 3D Navier-Stokes equations is considered to be one of the seven most important open problems in mathematics. This study will explore the much simpler, though hardly trivial, case of Navier-Stokes in two dimensions.

Many numerical methods have been used to solve Navier-Stokes, however they can all be improved upon. Numerical methods are generally judged on two main qualities: speed and accuracy. As engineers are dealing with increasingly large amounts of data (the number of unknowns in a system of equations can sometimes reach into the biillions!), speed is essential. The method explored in this paper (the line-based

*This work was supported McNair Scholars Program, UC Berkeley

†Department of Mathematics, University of California, Berkeley (mzweig@berkeley.edu). Questions, comments, or corrections to this document may be directed to that email address.

discontinuous Galerkin method) has been shown to be faster than its predecessors for certain types of problems including Navier-Stokes.

I will start out by giving a mathematical background of the equations I am working with and discontinuous Galerkin methods. I then will go into a more detailed explanation of the LDG method. My work was comprised of building solvers for simpler equations, in preparation for creating a LDG solver for the Navier-Stokes equations. I will discuss this process and will provide preliminary numerical results. This work is a replication of work that has previously been done by P.O. Persson [4], but serves as a build up to new research into shock capturing. This will be discussed further in the *Future Work* section.

2. Background. This paper examines various equations of fluid dynamics using the line-based discontinuous Galerkin method (LDG). Discontinuous Galerkin (DG) methods were first introduced by Reed and Hill [1] in 1973 to solve the hyperbolic neutron transport equation. Since then they have been used successfully with hyperbolic, elliptic [2], parabolic, and mixed form problems for a variety of applications. The DG method is a finite element formulation, meaning the computational domain is divided into discrete "elements" or cells in which each calculation is performed. Furthermore, these elements are "discontinuous," meaning they do not necessarily match up at the boundaries. The discontinuous nature of these elements eliminates the need for storage in a global matrix, thus reducing demand on memory. The element structure also provides geometric flexibility and the ability to increase accuracy of local approximations by using higher order polynomials.

The line-based discontinuous Galerkin (LDG) method [4] is a modification of standard discontinuous Galerkin (DG) descretization scheme. It has been shown to be successful for first and second order systems of partial differential equations with fully unstructured meshes with quadrilateral or hexahedral elements.

3. Implementation. Line-based discontinuous Galerkin methods were developed with the aim of maximizing the sparsity of the Jacobian matrices used in standard DG methods [4]. A sparse Jacobian leads to higher performance, especially for implicit solvers, and the desirable properties of DG are preserved. One-dimensional DG methods are then applied in each coordinate direction in an element. The scheme only connects each node to a line of nodes in each direction (hence "line-based") greatly reducing the number of connectivities.

Although LDG can be used with unstructured meshes, for the sake of simplicity, we used a structured Cartesian grid in our model problems. It has been shown [4] that the findings can also be applied to an unstructured mesh.

3.1. Standard DG. Consider the 1D conservation law

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} = 0 \tag{3.1}$$

with domain Ω and solution $\mathbf{u}$, flux $f(u)$. This is a first order partial differential equation. To solve, the domain in divided into discrete elements. Within each element, $p+1$ node points are introduced.

The Galerkin formulation given by finding $u_h \in V_h$ such that

$$\int_0^1 \frac{\partial u_h}{\partial t} v dx + \int_0^1 \frac{\partial f(u_h)}{\partial x} v dx = 0, \forall v \in V_h$$

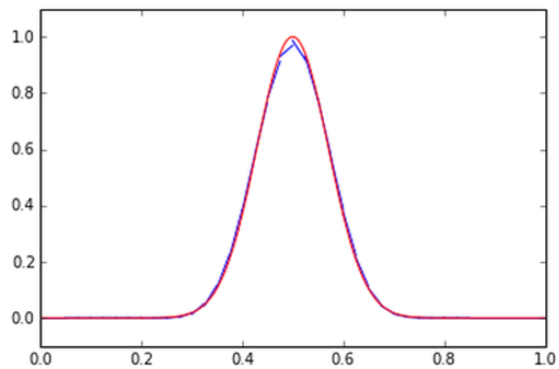

Fig. 3.1: 1D Conservation, n=40, p=1

We set the basis functions, $v = \phi_i^k$ and integrate by parts, giving the formulation

$$\int_{x_{k-1}}^{x_k} \frac{\partial u_h}{\partial t}\phi_i^k dx + [f(u_h(x))\phi_i^k(x)]_{xk-1}^{x_k} - \int_{x_{k-1}}^{x_k} f(u_h(x))\frac{d\phi_i^k}{dx}dx = 0$$

Because the elements are discontinuous at the boundaries, we use a numerical flux function $F(u_R, u_L)$ giving

$$\int_{x_{k-1}}^{x_k} \frac{\partial u_h}{\partial t}\phi_i^k dx + F(u_0^{k+1}, u_p^k)\phi_i^k(x_{k+1}) - F(u_0^k, u_p^{k-1})\phi_i^k(x_k) - \int_{x_{k-1}}^{x_k} f(u_h(x))\frac{d\phi_i^k}{dx}dx = 0$$

To test the method I considered the 1D conservation law with $f(u) = u$ and initial conditions $u(x, 0) = e^{-100(x-0.5)^2}$ on the domain [0, 1]. With these parameters, the formulation becomes

$$\begin{aligned}\int_{x_{k-1}}^{x_k} \frac{\partial}{\partial t}\Big(\sum_{j=0}^{p} u_j^k\phi_j^k(x)\Big)\phi_i^k dx - \int_{x_{k-1}}^{x_k} \Big(\sum_{j=0}^{p} u_j^k\phi_j^k(x)\Big)\frac{d\phi_i^k}{dx}dx \\ + u_p^k\phi_i^k(x_k) - u_p^{k-1}\phi_i^k(x_{k-1}) = 0\end{aligned}$$

We finally rearrange to obtain a linear system of equations

$$M^k\dot{\boldsymbol{u}}^k - C^k\boldsymbol{u}^k + \begin{pmatrix} -u_p^{k-1} \\ 0 \\ . \\ . \\ . \\ 0 \\ u_p^k \end{pmatrix} = 0$$

Where k is the index of the element and M and C are elementary matrices defined as

$$M_{ij}^k = \int_{x_{k-1}}^{x_k} \phi_i^k\phi_j^k dx$$

and

$$C_{ij}^k = \int_{x_{k-1}}^{x_k} \frac{d\phi_i^k}{dx} \phi_j^k dx$$

Elementary matrices are generated, inserted into the main equation, and we then solve for u. A Runge-Kutta procedure is used for the time step. The results are shown in Fig 3.1.

3.2. The Convection Diffusion Equation. Next, I considered the Convenction-Diffusion Equation

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} - \mu \frac{\partial^2 u}{\partial x^2} = 0 \tag{3.2}$$

This is a second order partial differential equation, so in order to use our method, we must first split it into a system of first order equations.

$$\begin{aligned} \frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} - \mu \frac{\partial \sigma}{\partial x} &= 0 \\ \frac{\partial u}{\partial x} &= \sigma \end{aligned}$$

A similar Galerkin formulation as above is found. This time, for each element k, we solve for sigma, and then plug into main equation to solve for u.

Below (Fig. 3.2) are the results of the test problem on the domain [0, 1] where $f(u) = u$ and $\mu = .001$. Again, numerical fluxes were used at the discontinuities.

$$F(u_R, u_L) = u_L, \hat{\sigma}(\sigma_R, \sigma_L) = \sigma_L, \hat{u}(u_R, u_L) = u_R$$

Upwinding was used for the convection and LDG upwinding/downwinding was used for the diffusion.

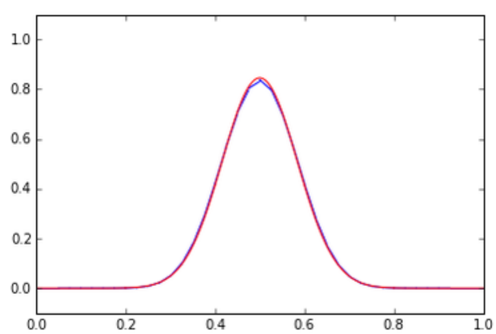

Fig. 3.2: 1D Convection-Diffusion, n=40, p=1

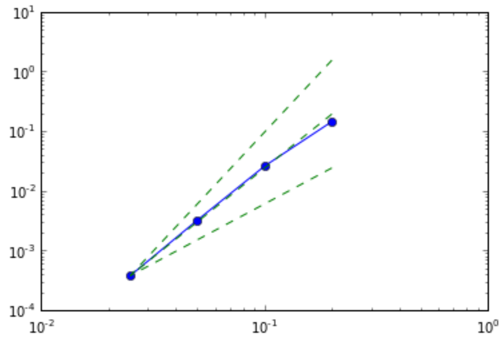

Fig. 3.3: Convergence plot for n=5, 10, 20, 40

3.3. Extending to Two Spatial Dimensions. The next step was to extend the solver for the 1D conservation and convection-diffusion equations to two spatial dimensions. The following are the 2D conservation equation (3.3) and the 2D convection-diffusion equation (3.4).

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} + \frac{\partial g(u)}{\partial y} = 0 \tag{3.3}$$

$$\frac{\partial u}{\partial t} + \frac{\partial f(u)}{\partial x} + \frac{\partial g(u)}{\partial y} - \mu(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) = 0 \tag{3.4}$$

Now, instead of descritizing a one-dimensional domain, we will use a two dimensional square domain with a Cartesian mesh (Fig. 3.4). Although we are using a Cartesian mesh here, LDG has been shown to work well with a variety of element shapes and unstructured meshes [4].

The domain is divided into n^2 elements, and then each element is given $(p+1)^2$ node points (Fig. 3.5).

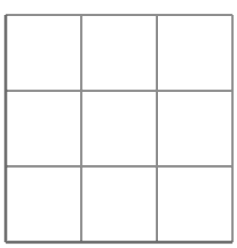

Fig. 3.4: A square domain with Cartesian mesh

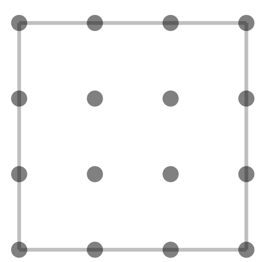

Fig. 3.5: A mapping of an element with p = 3

A standard Discontinuous Galerkin method would then choose a polynomial $u(x)$ interpolating the grid function $u_{ij} = u(x_{ij})$ and defining a numerical scheme for the spatial dimension. Instead, LDG considers each spatial dimension separately. In this case, there are two spatial dimensions that are approximated numerically using the one-dimensional Discontinuous Galerkin formulation discussed above along the horizontal lines and then the vertical lines created by the nodes in the element.

The following figures show solutions generated using LDG at $T = 1.0$ for conservation and convection-diffusion with $f(u) = u$ and initial conditions $u(x, y, 0) = e^{-100((x-0.5)^2+(y-0.5)^2)}$ with the unit square as a domain.

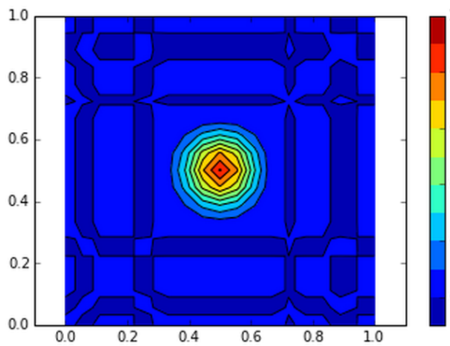

Fig. 3.6: 2D Conservation

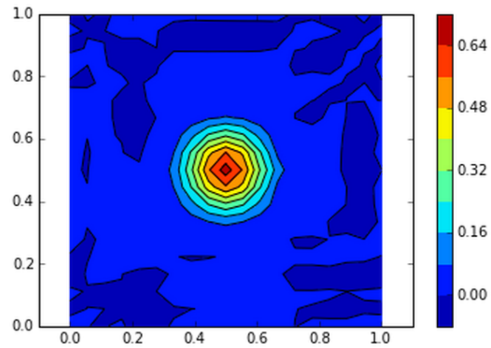

Fig. 3.7: 2D Convection-Diffusion

4. Future Work. A more general convection-diffusion equation can be written as

$$\frac{\partial u}{\partial t} + \overrightarrow{\beta}(\frac{\partial f(u)}{\partial x} + \frac{\partial g(u)}{\partial y}) - \mu(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}) = 0 \tag{4.1}$$

Where $\overrightarrow{\beta}$ is the average velocity that the quantity is moving. In the above models, we assumed $\overrightarrow{\beta} = (1, 1)$. My next step is to extend the functionality of the program to handle variable $\overrightarrow{\beta}$. This will involve changing the structure of the elementary matrix C and providing for upwinding and downwinding in the convection. Also, previous integration for the elmentary matrices was done using built-in numpy functions for polynomials. As I make the program more general, I will need to introduce Gaussian quadrature.

This process of increasing functionality at each step will ultimately result in the construction of a LDG solver for the compressible Euler and Navier Stokes equations:

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial x_i}(\rho u_i) = 0 \tag{4.2}$$

$$\frac{\partial}{\partial t}(\rho u_i) + \frac{\partial}{\partial x_i}(\rho u_i u_j + p) = +\frac{\partial \tau_{ij}}{\partial x_j} \qquad \text{for } i = 1, 2, 3 \tag{4.3}$$

$$\frac{\partial}{\partial t}(\rho E) + \frac{\partial}{\partial x_i}(u_j(\rho E + p)) = -\frac{\partial q_j}{\partial x_j} + \frac{\partial}{\partial x_j}(u_j \tau_{ij}) \tag{4.4}$$

Finally, after building a LDG solver for these equations, we will begin to experiment with shocks. We hypothesize that LDG will be more computationally efficient for handling shocks than previously used DG methods.

REFERENCES

[1] Reed, Wm. H., and T.R. Hill. “Triangular Mesh Methods For The Neutron Transport Equation.” AEC Tech Inf Cent (CONF - 730414-P1 and P2) (1973): 10-31.

[2] Arnold, Douglas N., Franco Brezzi, Bernardo Cockburn, and L. Donatella Marini. “Unified Analysis of Discontinuous Galerkin Methods for Elliptic Problems.” *SIAM J. Numer. Anal. SIAM Journal on Numerical Analysis* 39.5 (2002): 1749-779. Web.

[3] Persson, Per-Olof. ”Shock capturing for high-order discontinuous Galerkin simulation of transient flow problems.” *21st AIAA Computational Fluid Dynamics Conference*. 2013.

[4] Persson, Per-Olof. ”A Sparse and High-Order Accurate Line-Based Discontinuous Galerkin Method for Unstructured Meshes.” *Journal of Computational Physics* 233 (2013): 414-429.