Discontinuous Galerkin Sparse Grid Method for Maxwell's Equations

Tianyang Wang[*]

August 1, 2018

Abstract

In this paper, we will use discontinuous Galerkin method to solve the Maxwell's equations, where we will use sparse grid method to help us reduce the degrees of freedom from $\mathcal{O}(h^{-d})$ to $\mathcal{O}(h^{-1}|log_2 h|^{d-1})$ for d-dimensional problems without compromising much accuracy. Here h refers to the mesh size. And we will test different numerical fluxes to see the order of accuracy, for example, central flux, alternating flux, and upwinding flux. Moreover, we will test the order of Δt for time advanced methods, including explicit 3^{rd} order TVD Runge Kutta method, implicit Backward Euler, implicit Trapezoidal Rule, implicit 4^{th} order Gauss-Legendre method and semi-implicit Bacward Euler. Then we can compare the results between different time advanced methods to get our conlusion.

1 Introduction

1.1 Maxwell's Equations

In this paper, we consider the Maxwell's equations, which describe how electric and magnetic fields are generated by charges, currents, and charges of the fields. And one important consequence of the equations is that they demonstrate how fluctuating electric and magnetic fields propagate at the speed of light.

Let the domain $\Omega \times (0, T)$, the following problem will be considered in this paper,

$$\frac{\partial \boldsymbol{B}}{\partial t} + \nabla \times \boldsymbol{E} = 0$$

$$\epsilon\mu \frac{\partial \boldsymbol{E}}{\partial t} - \nabla \times \boldsymbol{B} = -\mu \boldsymbol{J}$$

$$\nabla \cdot \boldsymbol{E} = \frac{\rho_c}{\epsilon_c}$$

$$\nabla \cdot \boldsymbol{B} = 0$$

where $\boldsymbol{E}$ and $\boldsymbol{B}$ denote the electric and magnetic field densities, ϵ is the electric permittivity, and μ is the magnetic permeability; we assume the material parameters as piecewise constant and independent of time. At time $t = 0$, the problem are prescribed by initial conditions

$$\boldsymbol{E}^0 = \boldsymbol{E}(\boldsymbol{x}, 0), \ \boldsymbol{B}^0 = \boldsymbol{B}(\boldsymbol{x}, 0) \text{ on } \Omega$$

In addition, we consider the problem with periodic boundary condition on $\partial\Omega \times (0, T)$.

1.2 Discontinuous Galerkin Method

Discontinuous Galerkin (DG) methods have been attracted progressively research interest for numerical simulation of electromagnetic wave propagation problems in the time-domain. DG method is a finite element method where the discontinuous basis functions are used in the approximation.

Reed and Hill first introduced the DG methods for linear neutron transport equations in 1973 [1]. As a class of FE methods, DG methods adopt finite element type meshes for spatial discretization and inherit the high geometrical adaptability of FE. Moreover, they allow the solutions to be discontinuous across the element interface

[*] Department of Mathematics, The Chinese University of Hong Kong

by using discontinuous basis functions over the elements and defining numerical fluxes in the element interfaces, which differs from traditional continuous nodal FE methods. The local basis functions over one element are completely independent of those in neighboring elements, which offers inherent parallelism and allows the use of non-conforming meshes. There has been increasing researches on EM simulations using DG methods.

The DG method have gained popularity in many applications over the past few decades due to their distinctive features. And the advantage of DG method include high order approximation of the unknown field, their flexibility in choosing the discretization meshes and the approxiamation spaces, and the adaption to the parallel computing. The compact nature of DG discretization scheme is in favour of a high computation to communication ratio especially when the interpolation order is increased [2, 3]. However, the DG method will have large computation cost owing to the large number of degrees of freedom of the approximation space, which are the main bottleneck for simulations in high dimensions.

1.3 Sparse Grid Method

The main challenge of DG method for high dimensional problem is commonly known as the curse of dimensionality, which refers to the fact that the computational cost and storage requirements scale as $\mathcal{O}(h^{-d})$ for a d-dimensional problem, where h is the uniform mesh size in each dimension [4]. This challenge typically can not be resolved through barely increasing computational resources, and it requires the the improvement of numerical techniques as well as efficient computational implementations.

The sparse grid techniques have been developed as a major tool to break the curse if dimensionality of grid-based approach [5]. The idea relies on a tensor product hierarchical basis representation, which is able to reduce the degrees of freedom from $\mathcal{O}(h^{-d})$ to $\mathcal{O}(h^{-1}|log_2 h|^{d-1})$ for d-dimensional problems without compromising much accuracy. The sparse grid method can achieve the accuracy of $\mathcal{O}(h^k|log_2 h|^{d-1})$ in the energy norm, where k is the degree of polynomials used [4].

2 Numerical Methods

2.1 DG Notations

Throughout the paper, standard notations will be used for Soblev spaces. Given a bounded domain $B \subset \mathbb{R}$, we denote by $H^m(B)$ the L^2-Sobolev space of order m ≥ 0 and by $||\cdot||_{m,B}$ and $|\cdot|_{m,B}$ the usual Sobolev norm and seminorm, respectively. For $m = 0$, we write $L^2(B)$ instead of $H^0(B)$.

Let $\mathcal{T}_h = \{K\}$ be partitions of Ω_x, with K being Cartesian elements and h being the mesh size, then $\mathcal{T}_h = \{K : \forall K \in \mathcal{T}_h\}$ defines a partition of Ω. Let $\mathcal{E}$ be the set of the edges of $\mathcal{T}_h$. Next we define the discontinuous finite element space [6]:

$$V_h^k = \{v \in L^2(\Omega) : v|_k \in P^k(K),\ \forall K \in \mathcal{T}_h\}$$

where $P^k(B)$ denotes the set of polynomials of total degree at most k on B, and k is a nonnegative integer.

And for piecewise functions respect to $\mathcal{T}_h$, we introduce the jumps and averages as follows. For any edge $e = \{K^+ \cap K^-\} \in \mathcal{E}$, with $\boldsymbol{n}^{\pm}$ as the outward unit normal to $\partial K^{\pm}$ and $\boldsymbol{U}^{\pm} = \boldsymbol{U}|_{K^{\pm}}$, the jumps across e are defined as

$$[\boldsymbol{U}]_{\boldsymbol{n}} = \boldsymbol{U}^+ \cdot \boldsymbol{n}^+ + \boldsymbol{U}^- \cdot \boldsymbol{n}^-$$

$$[\boldsymbol{U}]_{\tau} = \boldsymbol{U}^+ \times \boldsymbol{n}^+ + \boldsymbol{U}^- \times \boldsymbol{n}^-$$

and the averages are

$$\{\boldsymbol{U}\} = \frac{1}{2}(\boldsymbol{U}^+ + \boldsymbol{U}^-)$$

2.2 SG Notations

Without loss of generality, we consider the interval $\Omega = [0, 1]$, then the grids are defined as the nested grids, where the n-th level grid Ω_n consists of 2^n uniform cells $I_j^n = (2^{-n}j, 2^{-n}(j+1)), j = 0, \ldots, 2^n - 1$, we can define [4]:

$$V_k^n := \{v : v \in P^k(I_j^n), \forall j = 0, \ldots, 2^n - 1\}$$

to be the usual piecewise polynomials of degree at most k on the n-th level grid Ω_n. Then we have

$$V_k^0 \subset V_k^1 \subset V_k^2 \subset \cdots$$

We can now define the multiwavelet subspace W_k^n, $n = 1,2,\ldots$ as the orthogonal complement of V_k^{n-1} in V_k^n with respect to the L^2 inner product on Ω,

$$V_k^{n-1} \bigoplus W_k^n = V_k^n, W_k^n \perp V_k^{n-1}$$

Here we let the base space $W_k^0 := V_k^0$, which consists of all standard piecewise polynomials of up to degree n. The dimension of W_k^n is $2^{n-1}(k+1)$ when $n \geq 1$, and $k+1$ when $n = 0$. In summary, we have found a hierarchical representation of the standard piecewise polynomial space V_k^n on Ω_n as $V_k^n = \bigoplus_{0 \leq j \leq n} W_k^j$.

For implementation purpose, we need to introduce basis functions for space W_k^n. The multiwavelet bases for different hierarchical levels are introduced as [7]:

$$v_{i,n}^j(x) = 2^{(n-1)/2} h_i(2^{n-1}x - j), \quad i = 1, \ldots, k+1, \ j = 0, \ldots, 2^{n-1} - 1$$

where functions $h_1, \ldots, h_k$ are piecewise polynomial, vanish outside the interval $[0,1]$, and are orthogonal to low-order polynomials.

Now, for a multi-index $l = [l_1, l_2, \ldots, l_d] \in \mathbb{N}_0^d$, where d is the dimension and $\mathbb{N}_0^d$ denotes the set of nonnegative integers, the l^1 and l^∞norms are defined as

$$|\boldsymbol{\alpha}|_1 := \sum_{m=1}^{d} \alpha_m, |\boldsymbol{\alpha}|_\infty := \max_{1 \leq m \leq d} \alpha_m$$

We define the tensor-product mesh grid $\Omega_l = \Omega_{l_1} \bigotimes \Omega_{l_2} \bigotimes \ldots \bigotimes \Omega_{l_d}$ and the crossing mesh size $h_l = (h_{l_1}, h_{l_2}, \ldots, h_{l_d})$, where $h_{l_m} = 2^{-l_m}$, then we define an elementary cell $I_j^l = \{\boldsymbol{x} : x_m \in (h_m j_m, h_m(j_m + 1)), m = 1, 2, \ldots, d\}$, and then

$$\boldsymbol{V}_k^l := \{\boldsymbol{v} : \boldsymbol{v}(\boldsymbol{x}) \in [P^k(I_j^l)], \quad 0 \leq j \leq 2^l - 1\}$$

where $P^k(I_j^l)$ contains all the polynomials of degree up to k in each dimension on cell I_j^l. And if we use equal mesh refinement of size $h_N = 2^{-N}$ in each direction, we denote the space grid and space to be Ω_N and $\boldsymbol{V}_k^N$, respectively.

Based on the tensor product construction, we have

$$\boldsymbol{W}_k^l = W_{k,x_1}^{l_1} \times W_{k,x_2}^{l_2} \times \cdots \times W_{k,x_d}^{l_d}$$

Also based on the one-dimensional hierarchical decomposition, we have

$$\boldsymbol{V}_k^l = V_{k,x_1}^{l_1} \times V_{k,x_2}^{l_2} \times \cdots \times V_{k,x_d}^{l_d} = \bigoplus_{j_1 \leq l_1, \ldots, j_d \leq l_d} \boldsymbol{W}_k^j$$

$$\boldsymbol{V}_k^N = V_{k,x_1}^N \times V_{k,x_2}^N \times \cdots \times V_{k,x_d}^N = \bigoplus_{|l|_\infty \leq N} \boldsymbol{W}_k^j$$

The basis functions for $\boldsymbol{W}_k^l$ can be defined by a tensor product

$$v_{i,l}^j := \prod_{m=1}^{d} v_{i_m, l_m}^{j_m}(x_m), \quad j_m = 1, \ldots, \max(0, 2^{l_m - 1}), \ i_m = 1, \ldots, k+1$$

And the sparse finite element approximation space $\hat{\boldsymbol{V}}_k^N$ is defined on Ω_N by

$$\hat{\boldsymbol{V}}_k^N = \bigoplus_{|l|_1 \leq N} \boldsymbol{W}_k^j$$

The sparse finite element space is a subset of traditional piecewise polynomial space, and its number of degrees of freedom scales as $\mathcal{O}((k+1)^d 2^N N^{d-1})$, which is significantly less than that of traditional space with $\mathcal{O}((2^N(k+1))^d)$. This is the key for computational savings and reduction in high dimensions [4].

2.3 Numerical Scheme

The semi-discrete DG method for Maxwell system are defined by following: given $k \geq 0$, look for $\boldsymbol{E}_h, \boldsymbol{B}_h \in V_h^k$, such that for any $\boldsymbol{U}, \boldsymbol{V} \in V_h^k$

$$\int_K \partial_t \boldsymbol{B}_h \cdot \boldsymbol{V} \ d\boldsymbol{x} = -\int_K \boldsymbol{E}_h \cdot \nabla \times \boldsymbol{V} \ d\boldsymbol{x} - \int_{\partial K} \widehat{\boldsymbol{n} \times \boldsymbol{E}_h} \cdot \boldsymbol{V} \ ds$$

$$\int_K \epsilon\mu \partial_t \boldsymbol{E}_h \cdot \boldsymbol{U} \ d\boldsymbol{x} = \int_K \boldsymbol{B}_h \cdot \nabla \times \boldsymbol{U} \ d\boldsymbol{x} + \int_{\partial K} \widehat{\boldsymbol{n} \times \boldsymbol{B}_h} \cdot \boldsymbol{U} \ ds - \int_K \mu \boldsymbol{J} \cdot \boldsymbol{U} \ d\boldsymbol{x}$$

where $\boldsymbol{n}$ is the outward unit normal of ∂K, and all hat functions denote the numerical fluxes.

$$\widehat{\boldsymbol{n} \times \boldsymbol{E}_h} := \boldsymbol{n} \times \hat{\boldsymbol{E}}_h, \quad \widehat{\boldsymbol{n} \times \boldsymbol{B}_h} := \boldsymbol{n} \times \hat{\boldsymbol{B}}_h$$

Thus the numerical method becomes the following: look for $\boldsymbol{E}_h, \boldsymbol{B}_h \in V_h^k$, such that

$$b_h(\boldsymbol{E}_h, \boldsymbol{B}_h; \boldsymbol{U}, \boldsymbol{V}) = l_h(\boldsymbol{J}; \boldsymbol{U})$$

where

$$\begin{aligned} b_h(\boldsymbol{E}_h, \boldsymbol{B}_h; \boldsymbol{U}, \boldsymbol{V}) = \sum_{K \in \mathcal{T}_h} (& \int_K \epsilon\mu \partial_t \boldsymbol{E}_h \cdot \boldsymbol{U} d\boldsymbol{x} - \int_K \boldsymbol{B}_h \cdot \nabla \times \boldsymbol{U} \ d\boldsymbol{x} \\ & + \int_K \partial_t \boldsymbol{B}_h \cdot \boldsymbol{V} \ d\boldsymbol{x} + \int_K \boldsymbol{E}_h \cdot \nabla \times \boldsymbol{V} \ d\boldsymbol{x} \\ & - \int_{\partial K} \widehat{\boldsymbol{n} \times \boldsymbol{B}_h} \cdot \boldsymbol{U} \ ds + \int_{\partial K} \widehat{\boldsymbol{n} \times \boldsymbol{E}_h} \cdot \boldsymbol{V} \ ds) \\ l_h(\boldsymbol{J}, \boldsymbol{U}) = - \sum_{K \in \mathcal{T}_h} & \int_K \mu \boldsymbol{J} \cdot \boldsymbol{U} d\boldsymbol{x} \end{aligned}$$

Here may use the central flux, alternating flux, and upwind flux.

$$\begin{aligned} central \ flux: \quad & \hat{\boldsymbol{E}}_h = \frac{1}{2}({\boldsymbol{E}_h}^+ + {\boldsymbol{E}_h}^-), \ \hat{\boldsymbol{B}}_h = \frac{1}{2}({\boldsymbol{B}_h}^+ + {\boldsymbol{B}_h}^-) \\ alternating \ flux: \quad & \hat{\boldsymbol{E}}_h = {\boldsymbol{E}_h}^+, \ \hat{\boldsymbol{B}}_h = {\boldsymbol{B}_h}^- \ or \ \hat{\boldsymbol{E}}_h = {\boldsymbol{E}_h}^-, \ \hat{\boldsymbol{B}}_h = {\boldsymbol{B}_h}^+ \\ up-winding \ flux: \quad & \hat{\boldsymbol{E}}_h = \{\boldsymbol{E}_h\} + \frac{1}{2}[\boldsymbol{B}_h]_\tau, \ \hat{\boldsymbol{B}}_h = \{\boldsymbol{B}_h\} - \frac{1}{2}[\boldsymbol{E}_h]_\tau \end{aligned}$$

The order of accuracy for central flux, alternating flux and up-winding flux are $\mathcal{O}(h^k), \mathcal{O}(h^{k+1/2}), \mathcal{O}(h^{k+1/2})$ respectively. Here h is the mesh size and k is the degree.

2.4 Energy Conservation

We are able to prove that central flux and alternating flux guarantee the conservation of energy, and the energy will decrease for up-winding flux.

Let $\boldsymbol{U} = \boldsymbol{E}_h$, $\boldsymbol{V} = \boldsymbol{B}_h$,

$$\begin{aligned} -\sum_K \int_K \mu \boldsymbol{J} \cdot \boldsymbol{E}_h \ d\boldsymbol{x} = \frac{1}{2} \frac{d}{dt} \sum_K (& \int_K (\epsilon\mu |\boldsymbol{E}_h|^2 + |\boldsymbol{B}_h|^2) \ d\boldsymbol{x} \\ & - \int_{\partial K} ([\boldsymbol{E}_h \times \boldsymbol{B}_h] + \hat{\boldsymbol{B}}_h \times [\boldsymbol{E}_h]_\tau - \hat{\boldsymbol{E}}_h \times [\boldsymbol{B}_h]_\tau) \ ds) \end{aligned}$$

And since for central flux and alternating flux,

$$[\boldsymbol{E}_h \times \boldsymbol{B}_h]_{\boldsymbol{n}} + \hat{\boldsymbol{B}}_h \times [\boldsymbol{E}_h]_\tau - \hat{\boldsymbol{E}}_h \times [\boldsymbol{B}_h]_\tau = 0$$

Then we have energy conservation

$$-\sum_K \int_K \mu \boldsymbol{J} \cdot \boldsymbol{E}_h \ d\boldsymbol{x} = \frac{1}{2} \frac{d}{dt} \sum_K (\int_K (\epsilon\mu |\boldsymbol{E}_h|^2 + |\boldsymbol{B}_h|^2) \ d\boldsymbol{x}$$

And for upwind flux, the energy is decreasing,

$$\begin{aligned} -\sum_K \int_K \mu \boldsymbol{J} \cdot \boldsymbol{E}_h \ d\boldsymbol{x} = \frac{1}{2} \frac{d}{dt} \sum_K (& \int_K (\epsilon\mu |\boldsymbol{E}_h|^2 + |\boldsymbol{B}_h|^2) \ d\boldsymbol{x} \\ & + \frac{1}{2} \int_{\partial K} |[\boldsymbol{E}_h]_\tau|^2 + |[\boldsymbol{B}_h]_\tau|^2 \ ds) \end{aligned}$$

2.5 Time Advanced Method

We use explicit, implicit and semi-implicit time advanced method to solve the method of lines ODE resulting from the semi-discrete DG scheme, $\frac{d}{dt}G_h = R(G_h)$. We use total variation diminishing (TVD) high-order Runge-Kutta methods for explicit method [8], and for implicit method, we use Backward Euler, Trapezoidal Rule, and Gauss-Legendre method [9]. For semi-implicit scheme, we use semi-implicit Backward Euler Method. Here G_h^n, B_h^n, E_h^n represent a numerical approximation of the solution at discrete time t_n.

TVD Runge Kutta Method The commonly used third-order TVD Runge Kutta Method is given by

$$G_h^{(1)} = G_h^n + \Delta t R(G_h^n)$$
$$G_h^{(2)} = \frac{3}{4}G_h^n + \frac{1}{4}G_h^{(1)} + \frac{1}{4}\Delta t R(G_h^{(1))}$$
$$G_h^{n+1} = \frac{1}{3}G_h^n + \frac{2}{3}G_h^{(2)} + \frac{2}{3}\Delta t R(G_h^{(2)})$$

Backward Euler The Backward Euler is a first order A-stable implicit Runge-Kutta method given by

$$G_h^{n+1} = G_h^n + \Delta t R(G_h^{n+1})$$

Trapezoidal Rule The Trapezoidal Rule is a second order A-stable implicit Runge-Kutta method given by

$$G_h^{n+1} = G_h^n + \frac{1}{2}\Delta t(R(G_h^n) + R(G_h^{n+1}))$$

Gauss-Legendre Method We use fourth-order Gauss-Legendre method which is also A-stable, given by

$$G_h^{(1)} = R(t_n + (\frac{1}{2} - \frac{1}{6}\sqrt{3})\Delta t,\ G_h^n + \frac{1}{4}\Delta t G_h^{(1)} + (\frac{1}{4} - \frac{1}{6}\sqrt{3})\Delta t G_h^{(2)})$$
$$G_h^{(2)} = R(t_n + (\frac{1}{2} + \frac{1}{6}\sqrt{3})\Delta t,\ G_h^n + (\frac{1}{4} + \frac{1}{6}\sqrt{3})\Delta t G_h^{(1)} + \frac{1}{4}\Delta t G_h^{(2)})$$
$$G_h^{n+1} = G_h^n + \frac{1}{2}\Delta t G_h^{(1)} + \frac{1}{2}\Delta t G_h^{(2)}$$

Semi-Implicit Backward Euler Method We divide the ODE into two parts, and the first order method is given by

$$B_h^{n+1} = B_h^n + \Delta t R_1(E_h^n)$$
$$E_h^{n+1} = E_h^n + \Delta t R_2(B_h^{n+1})$$

2.6 CFL condition

Then we can have the Courant-Friedrichs-Lewy condition for these time advanced method, which is a necessary condition for convergence while solving certain partial differential equations numerically.

$$\Delta t = CFL * \Delta x^{-r}$$

where r is the order of the different time advanced method we listed above and Δx refers to the mesh size h.

And we can have some derivation of the spectral radius of different method, then we are able to have the boundary condition for CFL number.

Suppose $G_h^{n+1} = MG_h^n + c$, and the target solution is G_h^*

$$G_h^* = MG_h^* + c$$
$$G_h^{n+1} = MG_h^n + c$$

Then we can see the error

$$e^{n+1} = Me^n$$
$$e^{n+1} = M^{n+1}e^0$$

We need the spectral radius of M less than 1, than the iteration scheme will converge. Hence we are able to get the boundary condition for CFL number to get the small enough Δt. And most important advantage of implicit time advanced method is that it can have much larger CFL number than that of explicit method, which means we are able to choose larger Δt, leading to a reduction of computation cost especially in high order and dimension cases.

3 Numerical Experiment

3.1 3D Case

In this subsection, we gather computational results for three-dimensional case.

Example 1 We solve the following three-dimensional problem using explicit time advanced method, we assume $\epsilon, \mu = 1$, and the domain $\Omega = [0,1]^3$.

$$\begin{cases} \dfrac{\partial \boldsymbol{B}}{\partial t} + \nabla \times \boldsymbol{E} = 0 \\ \dfrac{\partial \boldsymbol{E}}{\partial t} - \nabla \times \boldsymbol{B} = \boldsymbol{f} \end{cases}$$

where the source term $\boldsymbol{f}$, exact solution $\boldsymbol{B}, \boldsymbol{E}$ are given as

$$\boldsymbol{f} = \begin{bmatrix} -\frac{\pi}{2}\cos(2\pi x)\sin(2\pi y)\sin(2\pi z)\sin(\frac{\pi}{2}t) + 24\pi\cos(2\pi x)\sin(2\pi y)\sin(2\pi z)\sin(\frac{\pi}{2}t) \\ \\ x - x \\ \\ \frac{\pi}{2}\sin(2\pi x)\sin(2\pi y)\cos(2\pi z)\sin(\frac{\pi}{2}t) - 24\pi\sin(2\pi x)\sin(2\pi y)\cos(2\pi z)\sin(\frac{\pi}{2}t) \end{bmatrix}$$

$$\boldsymbol{B} = \begin{bmatrix} -4\sin(2\pi x)\cos(2\pi y)\cos(2\pi z)\sin(\frac{\pi}{2}t) \\ \\ 8\cos(2\pi x)\sin(2\pi y)\cos(2\pi z)\sin(\frac{\pi}{2}t) \\ \\ -4\cos(2\pi x)\cos(2\pi y)\sin(2\pi z)\sin(\frac{\pi}{2}t) \end{bmatrix}$$

$$\boldsymbol{E} = \begin{bmatrix} -\cos(2\pi x)\sin(2\pi y)\sin(2\pi z)\cos(\frac{\pi}{2}t) \\ \\ x - x \\ \\ \sin(2\pi x)\sin(2\pi y)\cos(2\pi z)\cos(\frac{\pi}{2}t) \end{bmatrix}$$

We first test the scheme with central flux, alternating flux and up-winding flux with $k = 1$, $\Delta t = 10^{-4}$, MaxT=100, here MaxT means the number of iteration. Then we can have the time period equals to 10^{-2}. And then we change the mesh size h by changing the level l, where $h = 2^{-l}$. From this test we can see that the order of accuracy for central flux, alternating flux and up-winding flux match $\mathcal{O}(h), \mathcal{O}(h^{1.5}), \mathcal{O}(h^{1.5})$ respectively. Following is the accuracy table of this test (Table 1) and the figure of L^2 error with respect to the Lev (Figure 1).

Table 1: The L^2 error and order of accuracy for fixed Δt and MaxT

k	l	central flux	order	alternating flux	order	up-winding flux	order
1	3	9.847E−03		2.195E−02		1.896E−02	
1	4	9.020E−03	0.127	1.529E−02	0.521	1.245E−02	0.607
1	5	7.767E−03	0.216	9.257E−03	0.724	8.392E−03	0.570
1	6	5.017E−03	0.631	5.265E−03	0.814	4.035E−03	1.056
1	7	2.781E−03	0.851	2.623E−03	1.006	1.793E−03	1.171
1	8	1.463E−03	0.927	1.182E−03	1.150	7.015E−04	1.354
1	9	7.478E−04	0.968	4.334E−04	1.448	2.526E−04	1.474

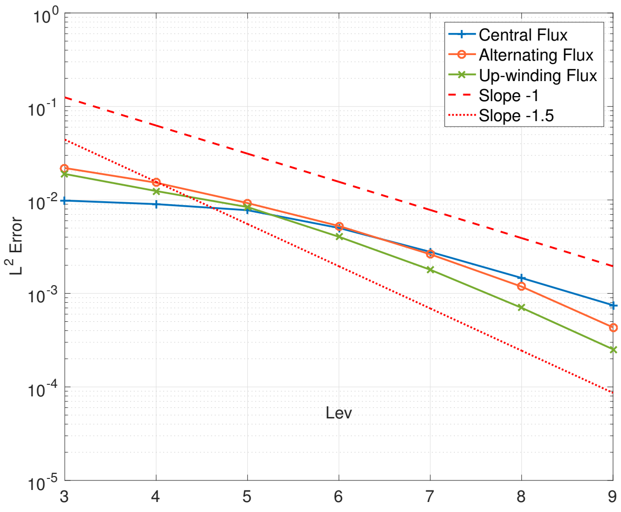

Figure 1: L^2 error for different numerical flux with fixed Δt and MaxT

Next we are going to see when we fix the CFL number for each scheme of numerical flux, what the order of accuracy will be. Here we choose CFL=0.0068 for central flux, CFL=0.0045 for alternating flux, and CFL=0.00625 for up-winding flux, in order to make the iteration scheme converge. And this time we choose time period=1. Here $\Delta t = CFL * 2^{(-l/3)}$, and l refers to the level. Following is the accuracy table of this test (Table 2) and the figure of L^2 error with respect to the level (Figure 2).

Table 2: The L^2 error and order of accuracy for fixed CFL

k	l	central flux CFL=0.0068	order	alternating flux CFL=0.0045	order	up-winding flux CFL=0.00625	order
1	4	1.886E+00		1.277E+00		4.519E−01	
1	5	1.501E+00	0.332	9.158E−01	0.480	2.224E−01	1.023
1	6	8.281E−01	0.857	4.058E−01	1.174	1.008E−01	1.141
1	7	4.749E−01	0.802	1.858E−01	1.127	3.425E−02	1.558
1	8	2.501E−01	0.925	7.502E−02	1.309	1.169E−02	1.551
1	9	1.255E−01	0.995	2.748E−02	1.450	4.294E−03	1.445

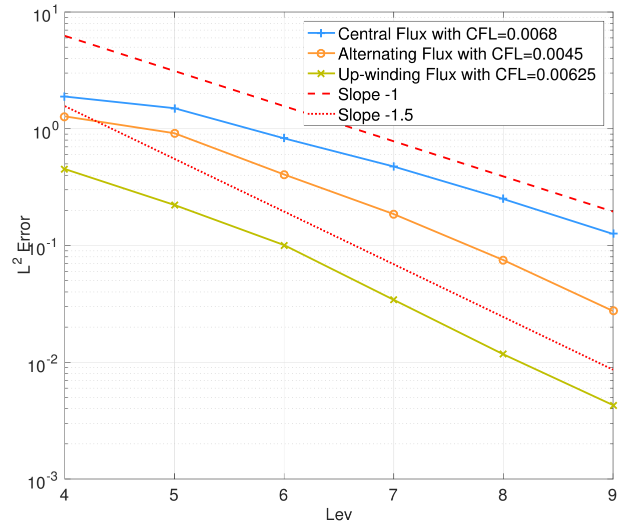

Figure 2: L^2 error for different numerical flux with fixed CFL number

Furthermore, through calculating the spectral radius we can see the boundary CFL condition for different numerical flux. According to this result (Table 3), the CFL number of alternating flux should be smaller than other two fluxes, which means Δt should be smaller for alternating flux. Moreover, we also test the central flux cases for sparse grid scheme and full grid scheme (Table 4), showing that the full grid cases should have smaller CFL number and Δt.

Table 3: The approximation of bounding CFL

k	l	central flux spectral radius	CFL	alternating flux spectral radius	CFL	up-winding flux spectral radius	CFL
1	3	3.271E+01	0.1059	5.156E+01	0.0672	6.162E+01	0.0811
1	4	6.428E+01	0.0679	9.815E+01	0.0445	1.089E+02	0.0578
1	5	1.282E+02	0.0429	1.931E+02	0.0285	2.045E+02	0.0388
1	6	2.561E+02	0.0271	3.846E+02	0.0180	3.962E+02	0.0252
1	7	5.121E+02	0.0170	7.683E+02	0.0114	7.800E+02	0.0161
1	8	1.024E+03	0.0107	1.536E+03	0.0072	1.465E+03	0.0108

Table 4: The spectral radius and CFL number for central flux

k	l	sparse grid spectral radius	sparse grid CFL	full grid spectral radius	full grid CFL
1	3	3.271E+01	0.1059	5.521E+01	0.0627
1	4	6.428E+01	0.0679	1.104E+02	0.0395
2	3	6.614E+01	0.0524	1.115E+02	0.0311
2	4	1.298E+02	0.0336	2.230E+02	0.0196

3.2 1D Case

In this subsection, we gather computational results for one-dimensional case.

Example 2 We solve the following one-dimensional problem using time advanced method listed above with central flux, we assum the domain is $\Omega = [0,1]$.

$$\begin{cases} \dfrac{\partial B_3}{\partial t} = \dfrac{\partial E_1}{\partial x_2} \\ \dfrac{\partial E_1}{\partial t} = \dfrac{\partial B_3}{\partial x_2} - j_1 \\ \dfrac{\partial E_2}{\partial t} = -j_2 \end{cases}$$

where the source term j_1, j_2, exact solution B_3, E_1, E_2 are given as

$$j_1 = \frac{15}{2}\pi\cos(2\pi x)\sin(\frac{\pi}{2}t), \quad j_2 = \frac{\pi}{2}\sin(2\pi x)\sin(\frac{\pi}{2}t)$$

$$B_3 = 4\sin(2\pi x)\sin(\frac{\pi}{2}t), \quad E_1 = -\cos(2\pi x)\cos(\frac{\pi}{2}t), \quad E_2 = \sin(2\pi x)\cos(\frac{\pi}{2}t)$$

For this case, we first test the order of Δt for Backward Euler, Trapezoidal Rule and Gauss-Legendre method, the following figure shows that the order of Δt for these three method is $\mathcal{O}(\Delta t), \mathcal{O}(\Delta t^2)$, and $\mathcal{O}(\Delta t^4)$ respectively (Figure 3). Here the y-axis is the L^2 error, and x-axis is $1/\Delta t$.

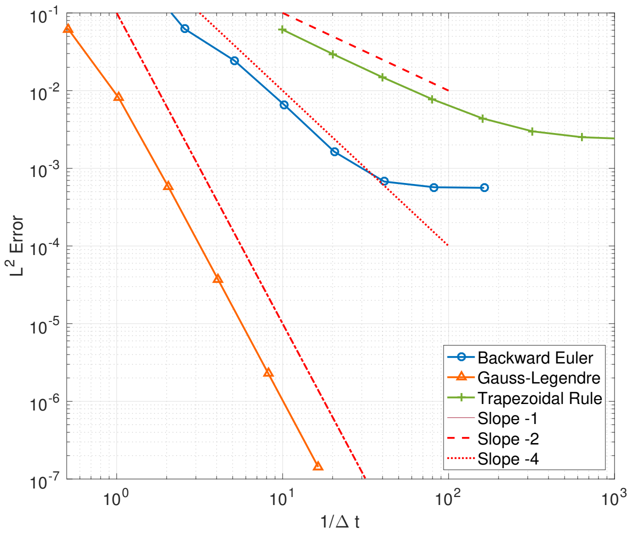

Figure 3: L^2 error for different different implicit method

Example 3 We solve the following one-dimensional problem using time advanced method listed above with central flux, we assum the domain is $\Omega = [0, 1]$.

$$\begin{cases} \dfrac{\partial B_3}{\partial t} = \dfrac{\partial E_1}{\partial x_2} \\ \dfrac{\partial E_1}{\partial t} = \dfrac{\partial B_3}{\partial x_2} - j_1 \\ \dfrac{\partial E_2}{\partial t} = -j_2 \end{cases}$$

where the source term j_1, j_2, exact solution B_3, E_1, E_2 are given as

$$j_1 = (2\pi + \frac{1}{2\pi})\cos(2\pi x)e^{-t}, \quad j_2 = \sin(2\pi x)e^{-t}$$

$$B_3 = \sin(2\pi x)e^{-t}, \quad E_1 = \frac{1}{2\pi}\cos(2\pi x)e^{-t}, \quad E_2 = \sin(2\pi x)e^{-t}$$

For this case we first use trapezoidal rule to test the L^2 error in different time period with different fixed CFL number, here $\Delta t = CFL * 2^{-l/2}$, l refers to the level size. We are able to see that with CFL number going larger, the L^2 error will also be larger. Following is the accuracy table of this test (Table 5) and the figure of L^2 error with respect to the level (Figure 4).

Table 5: The L^2 error for $k = 1$, $h = 2^{-10}$, Trapezoidal Rule

Time Period	CFL=1	CFL=5	CFL=10
0.01	2.352E−04	7.072E−04	2.025E−03
1.00	2.997E−04	1.039E−03	4.184E−03
2.00	3.993E−04	1.319E−03	5.201E−03
3.00	4.260E−04	1.454E−03	5.588E−03
4.00	4.298E−04	1.491E−03	5.733E−03

And we can also compare the explicit, implicit and semi-implicit time advanced method to see what the different Δt they need to make the iteration scheme converge to the target solution that we want with the error in acceptable level. In this test, we fix level $l = 9$, degree $k = 1$. The results are given in the following table (Table 6).

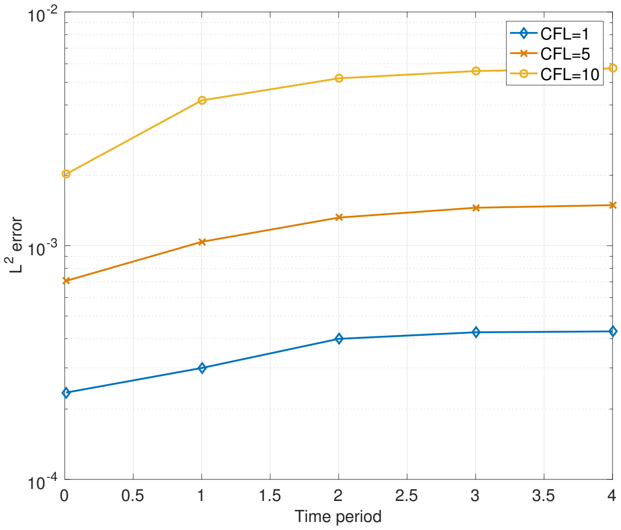

Figure 4: L^2 error for for different time period with fixed CFL number

Table 6: Time Advance Method for 1D Maxwell's Equation of $k = 1$, $h = 2^{-9}$, Central Flux.

Time Advance Method	L^2 error	Time Step
Explicit 3rd TVD Runge-Kutta	2.432E−03	8.000E−04
Implicit Backward Euler	2.406E−03	1.000E−03
Implicit Trapezoidal Rule	2.381E−03	1.000E−02
Implicit 4th Order Gauss-Legendre	2.548E−03	1.000E−01
Semi-implicit Backward Euler	2.383E−03	9.000E−04

4 Conclusion and Future Work

In this paper, we test different numerical fluxes using explicit 3^{rd} order TVD Runge Kutta method in 3D case. The testing results show that the order of accuracy for alternating flux and up-winding flux are half an order higher than that of central flux, and the CFL bounding condition for alternating flux is smaller than those of central flux and up-winding flux. This leads us to the conclusion that up-winding flux can have higher order of accuracy, and also have the fewest limitations of Δt among these three numerical fluxes.

Moreover, for the time advanced method, we test explicit 3^{rd} order TVD Runge Kutta method, implicit Backward Euler, implicit Trapezoidal Rule, implicit 4^{th} order Gauss-Legendre method and semi-implicit Bacward Euler. From the results shown above, it is easy for us to see that explicit and semi implicit method will have stricter limitations of Δt, but another problem for implicit time advanced method is that it will take much more time to compute the inverse of the matrix, which we need to make it more efficient in the future work. Pre-conditioner method is suggested to improve the calculation of the inverse problem.

Acknowledgement

This project is sponsored by Oak Ridge National Laboratory, Joint Institute for Computational Sciences, University of Tennessee, Knoxville and The Chinese University of Hong Kong.

Most sincere gratitude to my mentors: Dr.Lin Mu, Dr.David L.Green, Dr.Ed D'Azevedo, Dr.Kwai Wong.

References

[1] William H Reed and TR Hill. Triangular mesh methods for the neutron transport equation. Technical report, Los Alamos Scientific Lab., N. Mex.(USA), 1973.

[2] Bernardo Cockburn, George E Karniadakis, Chi-Wang Shu, and M Griebel. Discontinuous galerkin methods theory, computation and applications, lectures notes in computational science and engineering. *Inc. Marzo del*, 2000.

[3] Stephane Lanteri and Ronan Perrussel. *An implicit hybridized discontinuous Galerkin method for time-domain Maxwell's equations.* PhD thesis, INRIA, 2011.

[4] Zixuan Wang, Qi Tang, Wei Guo, and Yingda Cheng. Sparse grid discontinuous galerkin methods for high-dimensional elliptic equations. *Journal of Computational Physics*, 314:244–263, 2016.

[5] Hans-Joachim Bungartz and Michael Griebel. Sparse grids. *Acta numerica*, 13:147–269, 2004.

[6] Yingda Cheng, Irene M Gamba, Fengyan Li, and Philip J Morrison. Discontinuous galerkin methods for the vlasov–maxwell equations. *SIAM Journal on Numerical Analysis*, 52(2):1017–1049, 2014.

[7] Bradley K Alpert. A class of bases in lˆ2 for the sparse representation of integral operators. *SIAM journal on Mathematical Analysis*, 24(1):246–262, 1993.

[8] Chi-Wang Shu and Stanley Osher. Efficient implementation of essentially non-oscillatory shock-capturing schemes, ii. In *Upwind and High-Resolution Schemes*, pages 328–374. Springer, 1989.

[9] Roger Alexander. Diagonally implicit runge–kutta methods for stiff ode's. *SIAM Journal on Numerical Analysis*, 14(6):1006–1021, 1977.