Discontinuous Galerkin Sparse Grid method for Maxwell's equations

Student: Tianyang Wang

Mentor: Dr. Lin Mu, Dr. David L.Green, Dr. Ed D'Azevedo, Dr. Kwai Wong

Motivation

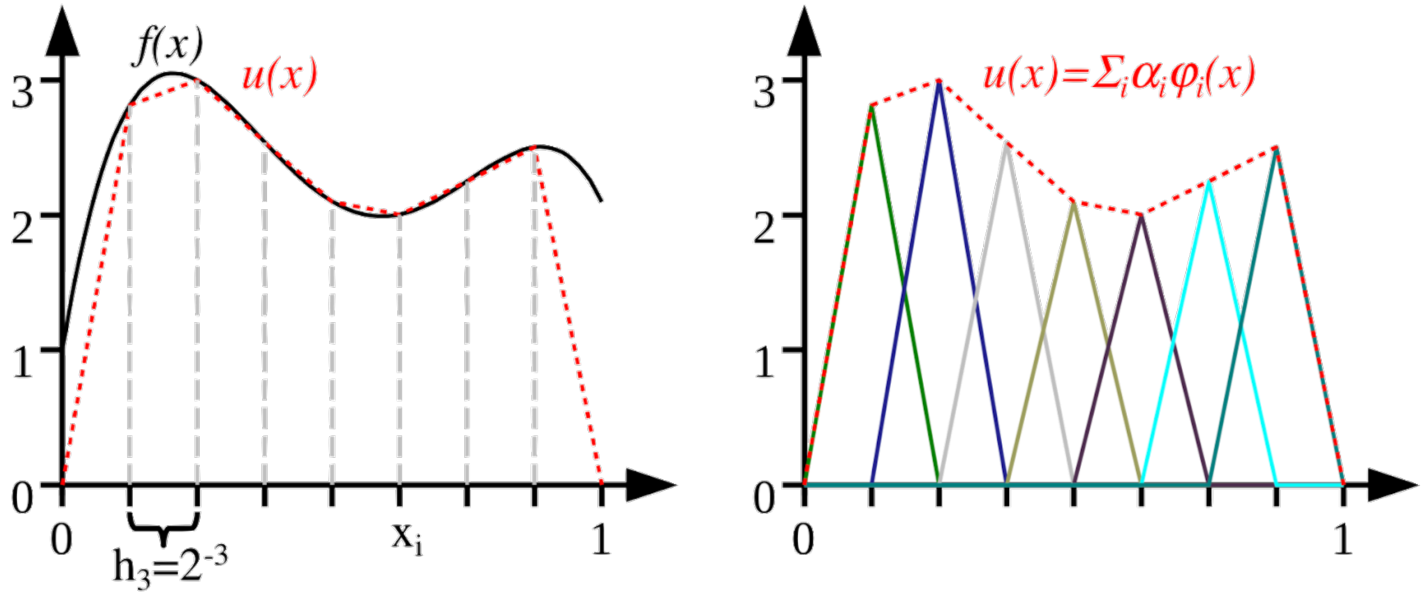
- We consider the Maxwell's equations, which is a set of high dimensional partial differential equations describing how electric and magnetic fields are generated by charges, currents, and charges of the fields. And one important consequence of the equations is that they demonstrate how fluctuating electric and magnetic fields propagate at the speed of light.

- We choose to use discontinuous Galerkin method (DG), whose advantages include high order approximation and adaption to parallel computing. But the shortage is that DG method will have large degree of freedom for high dimensional problem, which may have too much computational cost.

- So we use the sparse grid method which based on the tensor product of hierarchical basis functions, in order to reduce the computational cost, especially in high dimensions.

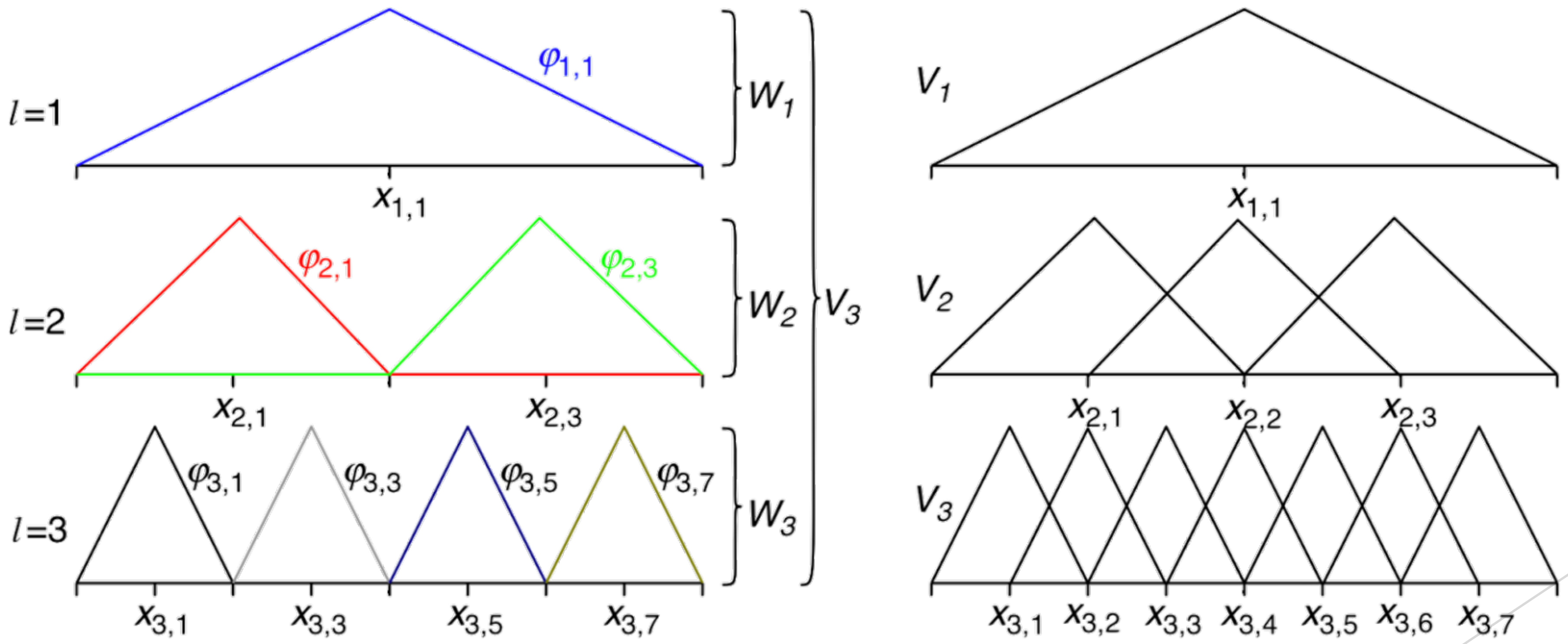
Discontinuous Galerkin Method

The DG method will use basis functions that are discontinuous in the boundary of each grid. And we use the set of basis functions to approximate the target function that we want.

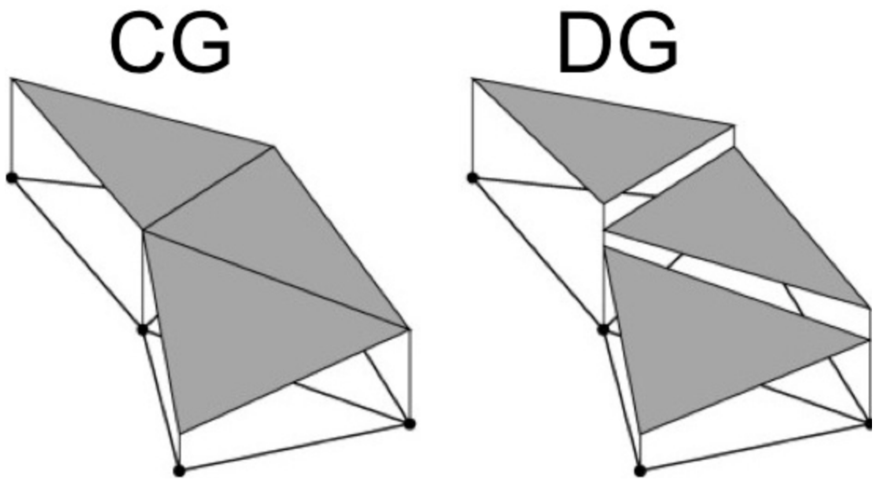


Discontinuous Galerkin Method

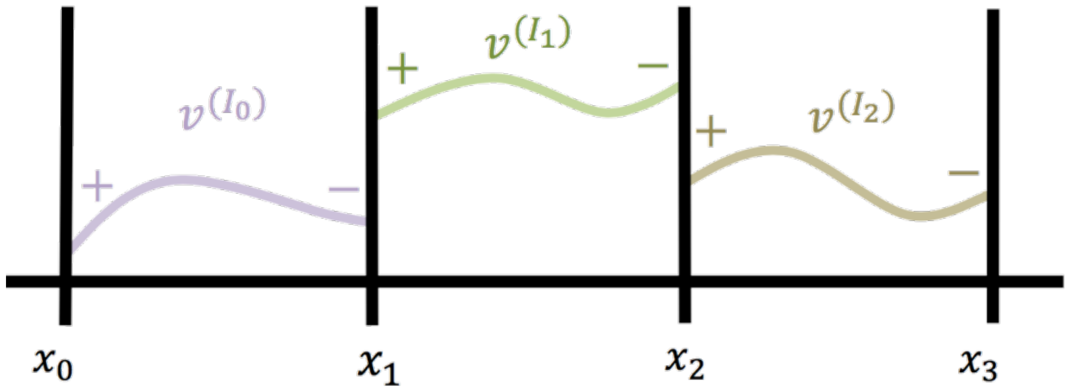
Here we use Hierarchical basis (bottom left) rather than nodal basis (bottom right)



Discontinuous Galerkin Method



Here the choice of the value on the boundary will be important, which we call the numerical flux.



x_j^+: start of the “left” interval I_j

x_j^-: start of the “right” interval I_{j-1}

Discontinuous Galerkin Method

Maxwell Equation:

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

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

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

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

Numerical Flux:

$$central\ flux: \quad \hat{\boldsymbol{E}}_h = \frac{1}{2}({\boldsymbol{E}_h}^+ + {\boldsymbol{E}_h}^-),\ \hat{\boldsymbol{B}}_h = \frac{1}{2}({\boldsymbol{B}_h}^+ + {\boldsymbol{B}_h}^-)$$

$$alternating\ flux: \quad \hat{\boldsymbol{E}}_h = {\boldsymbol{E}_h}^+,\ \hat{\boldsymbol{B}}_h = {\boldsymbol{B}_h}^-\ \ or\ \ \hat{\boldsymbol{E}}_h = {\boldsymbol{E}_h}^-,\ \hat{\boldsymbol{B}}_h = {\boldsymbol{B}_h}^+$$

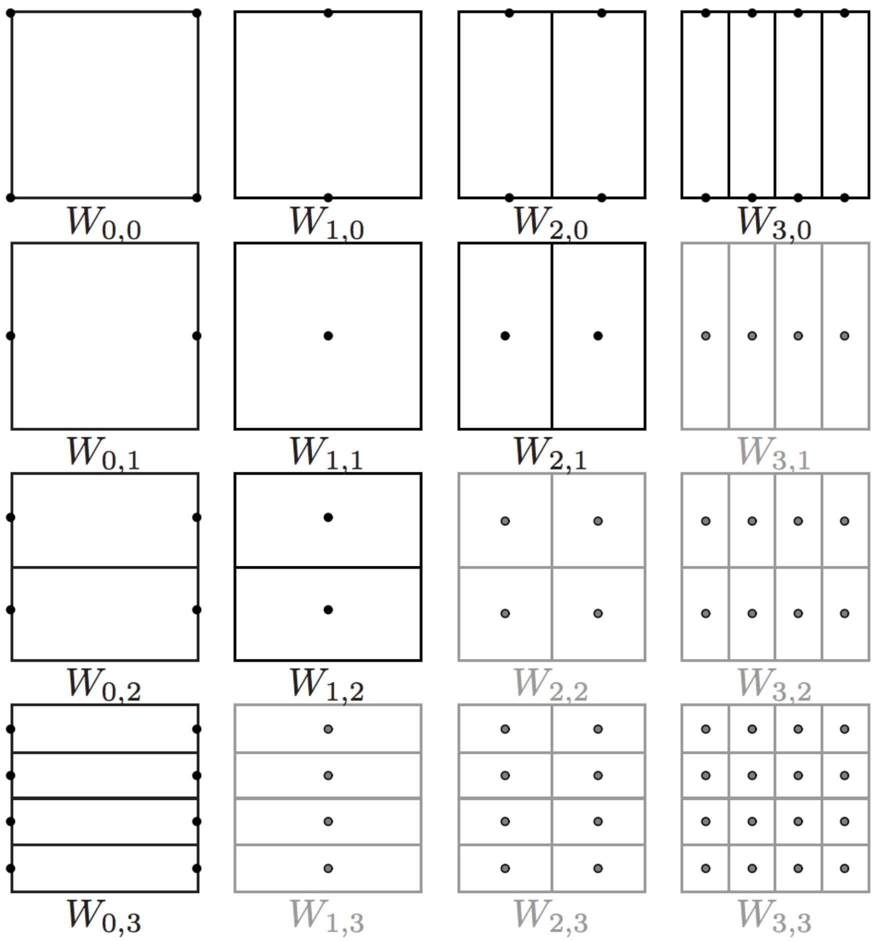
$$upwind\ flux: \quad \hat{\boldsymbol{E}}_h = \{\boldsymbol{E}_h\} + \frac{1}{2}[\boldsymbol{B}_h]_\tau,\ \hat{\boldsymbol{B}}_h = \{\boldsymbol{B}_h\} - \frac{1}{2}[\boldsymbol{E}_h]_\tau$$

Sparse Grid Method

Briefly we can understand in this way, full grid method requires the level in each dimension should be less than or equal to N, while sparse grid method requires the sum of levels in each dimension should be less than or equal to N. Here we denote k as the degree, N as the maximum level and d as the dimension.

$$\mathbf{V}_N^k = \bigoplus_{\substack{|\mathbf{l}|_\infty \leq N \\ \mathbf{l} \in \mathbb{N}_0^d}} \mathbf{W}_\mathbf{l}^k.$$

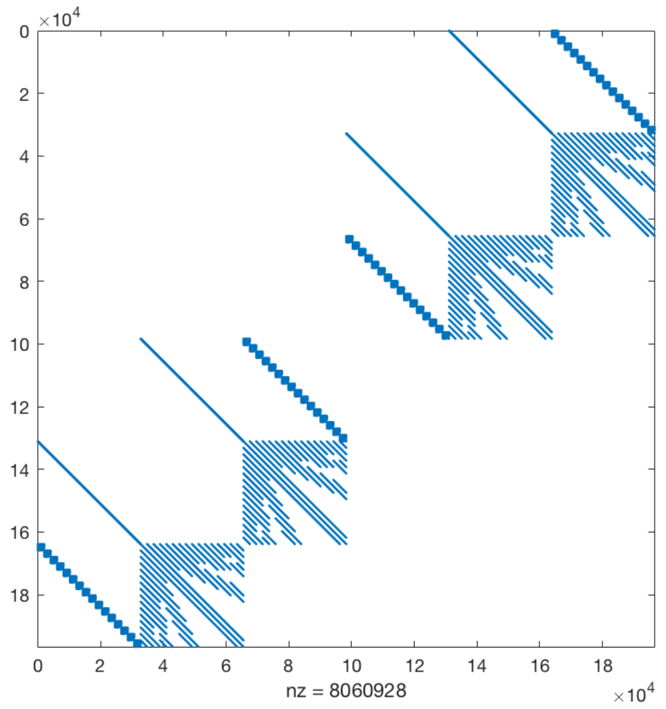
$$\hat{\mathbf{V}}_N^k := \bigoplus_{\substack{|\mathbf{l}|_1 \leq N \\ \mathbf{l} \in \mathbb{N}_0^d}} \mathbf{W}_\mathbf{l}^k.$$

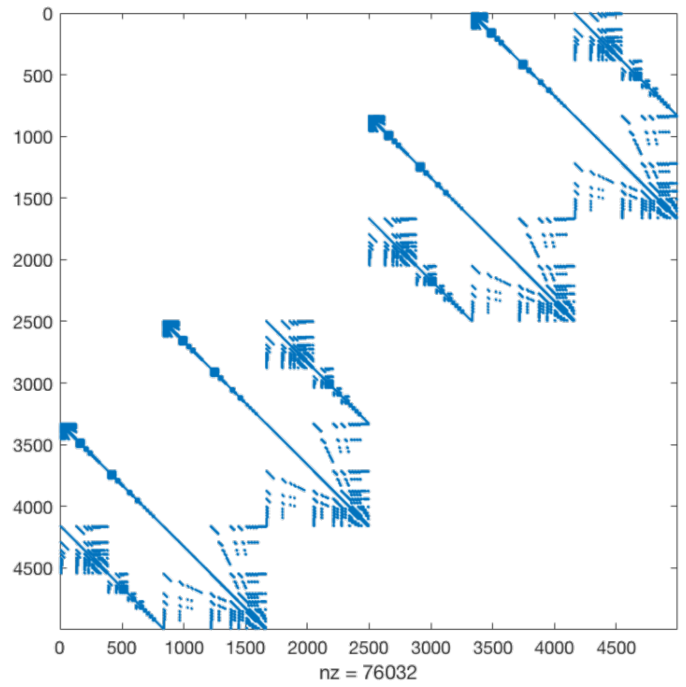


Able to reduce the degree of freedom from $O(h^{-d})$ to $O(h^{-1}|\log_2 h|^{d-1})$

And receive the accuracy of $O(h^k|\log_2 h|^{d-1})$

Sparse Grid Method





Full Grid

DOF : 64 k
Memory : 1406 MB
FLOPS : 351 M

Sparse Grid

DOF : 4 k (16x)
Memory : 62 MB (22x)
FLOPS : 15 M (23x)

Time Advanced Method

Explicit time advanced method:

3rd Order TVD Runge Kutta: dt=dx^(1/3)*CFL

Implicit time advanced method:

Backward Euler: dt=dx*CFL

Trapezoidal Rule: dt=dx^(1/2)*CFL

4th Order Gauss-Legendre: dt=dx^(1/4)*CFL

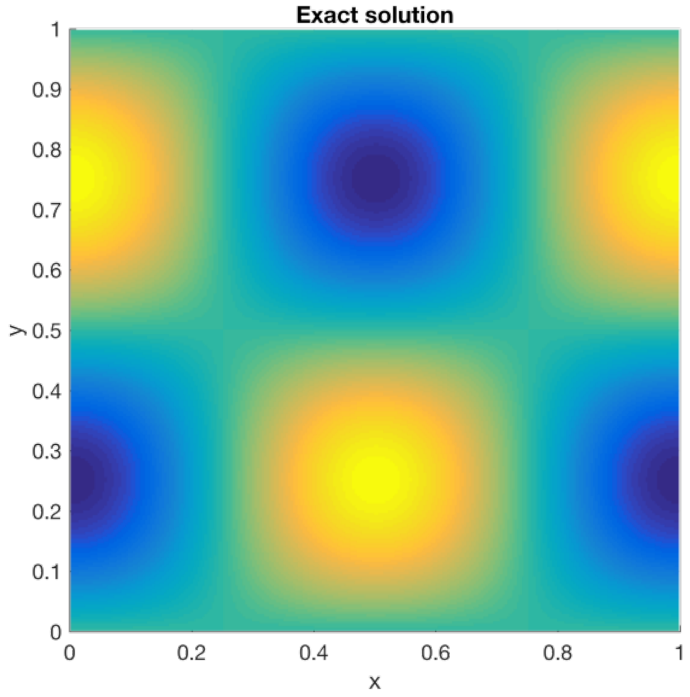
3rd Order TVD Runge Kutta

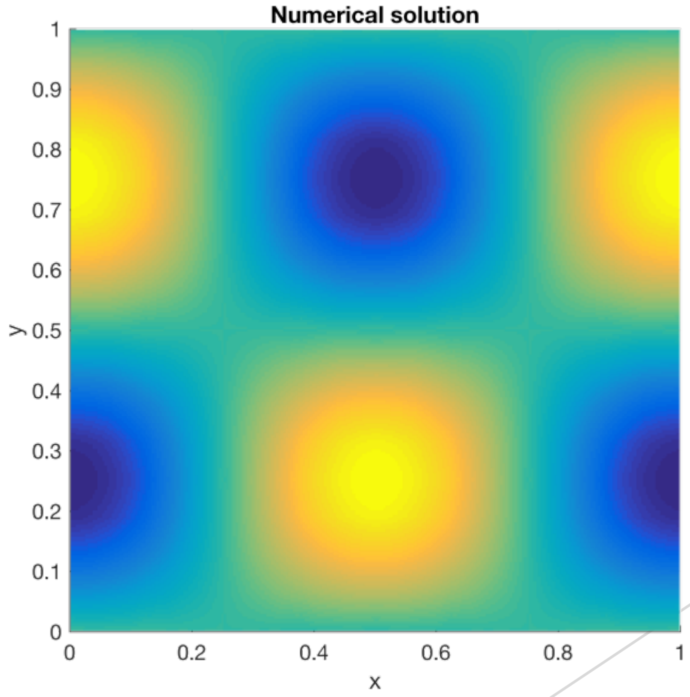
we want to solve $\frac{d}{dt}G_h = R(G_h)$

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

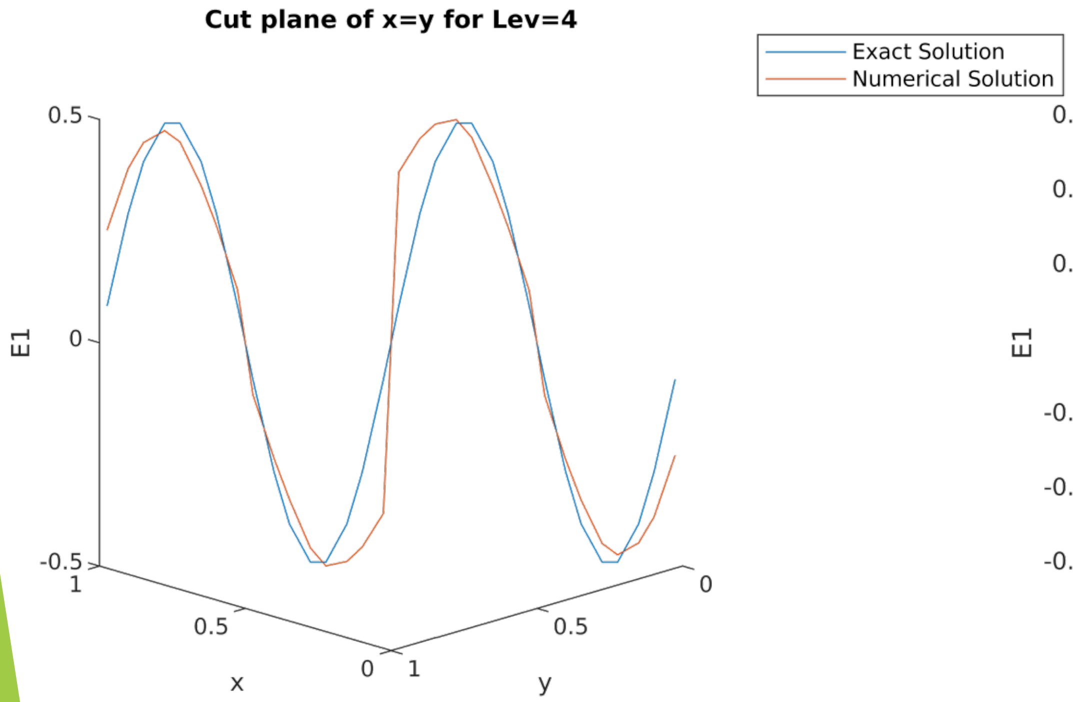
$$G_h^{(2)} = \frac{3}{4}G_h^n + \frac{1}{4}G_h^{(1)} + \frac{1}{4}\Delta t R(G_h^{(1)})$$

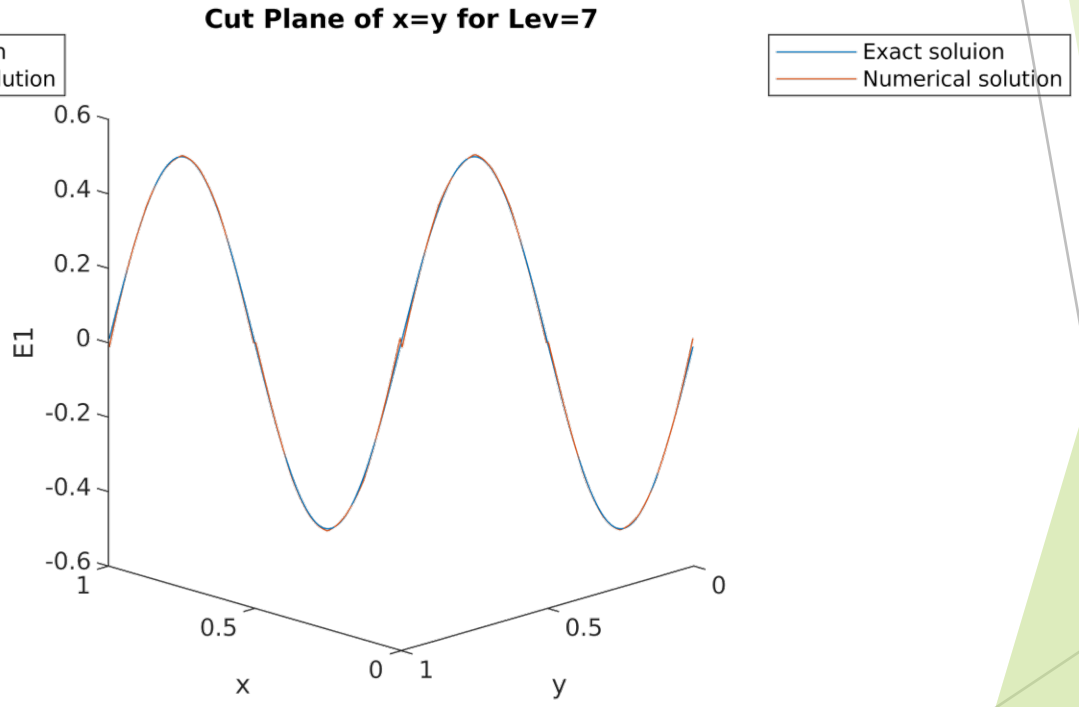
$$G_h^{n+1} = \frac{1}{3}G_h^n + \frac{2}{3}G_h^{(2)} + \frac{2}{3}\Delta t R(G_h^{(2)})$$





3rd Order TVD Runge Kutta

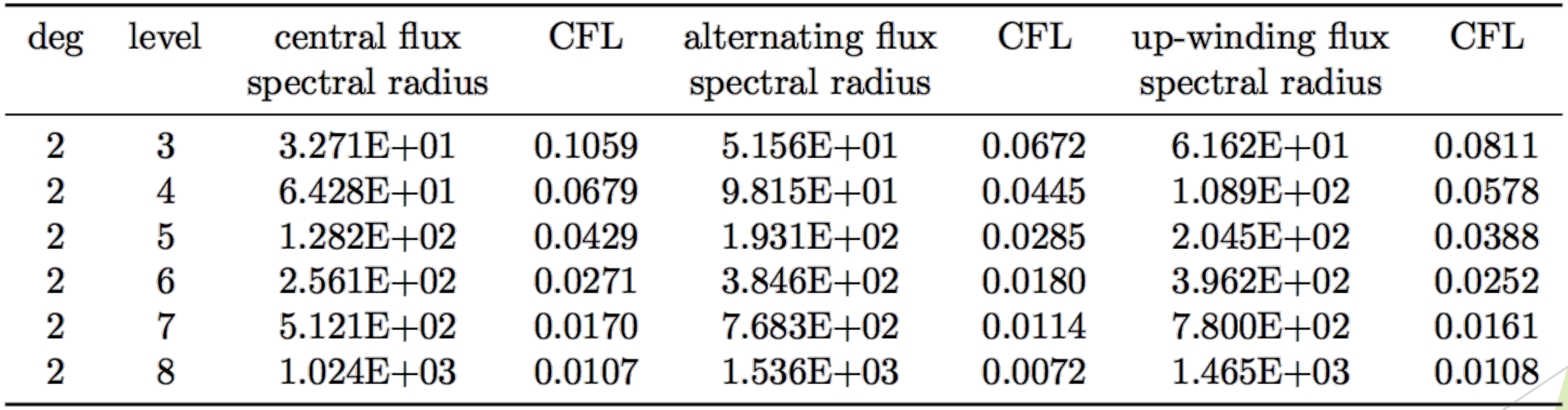




3rd Order TVD Runge Kutta

This method have bounded condition for the CFL number to avoid the blow up of the error.

dt=dx^(1/3)*CFL

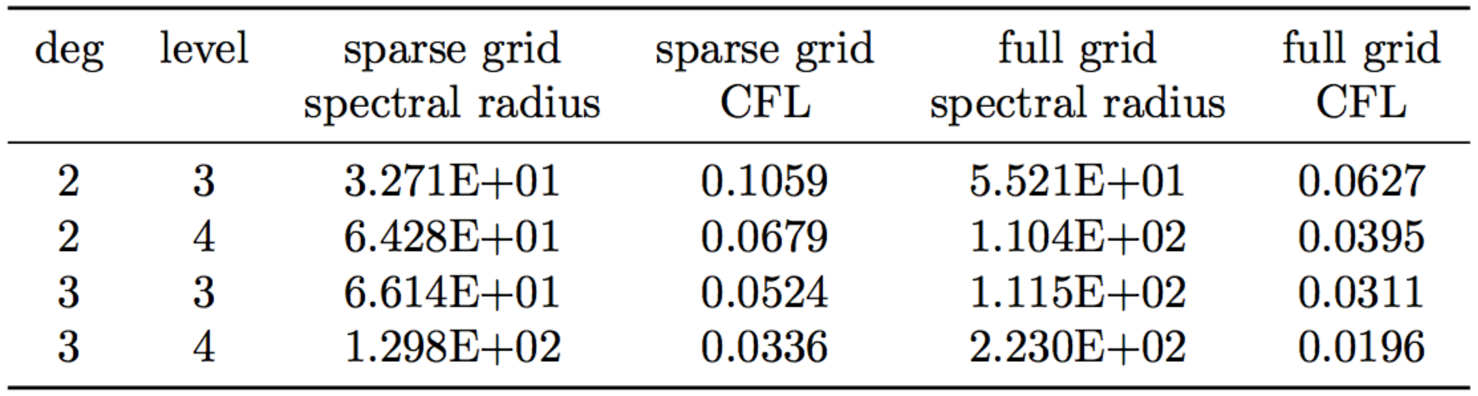


deg	level	central flux spectral radius	CFL	alternating flux spectral radius	CFL	up-winding flux spectral radius	CFL
2	3	3.271E+01	0.1059	5.156E+01	0.0672	6.162E+01	0.0811
2	4	6.428E+01	0.0679	9.815E+01	0.0445	1.089E+02	0.0578
2	5	1.282E+02	0.0429	1.931E+02	0.0285	2.045E+02	0.0388
2	6	2.561E+02	0.0271	3.846E+02	0.0180	3.962E+02	0.0252
2	7	5.121E+02	0.0170	7.683E+02	0.0114	7.800E+02	0.0161
2	8	1.024E+03	0.0107	1.536E+03	0.0072	1.465E+03	0.0108

3rd Order TVD Runge Kutta

If we use 3rd order Runge Kutta Method, there will be a maximum boundary condition of CFL number, otherwise the error will blow up. And the following table will show the bounded condition of CFL number between sparse grid and full grid.

dt=dx^(1/3)*CFL



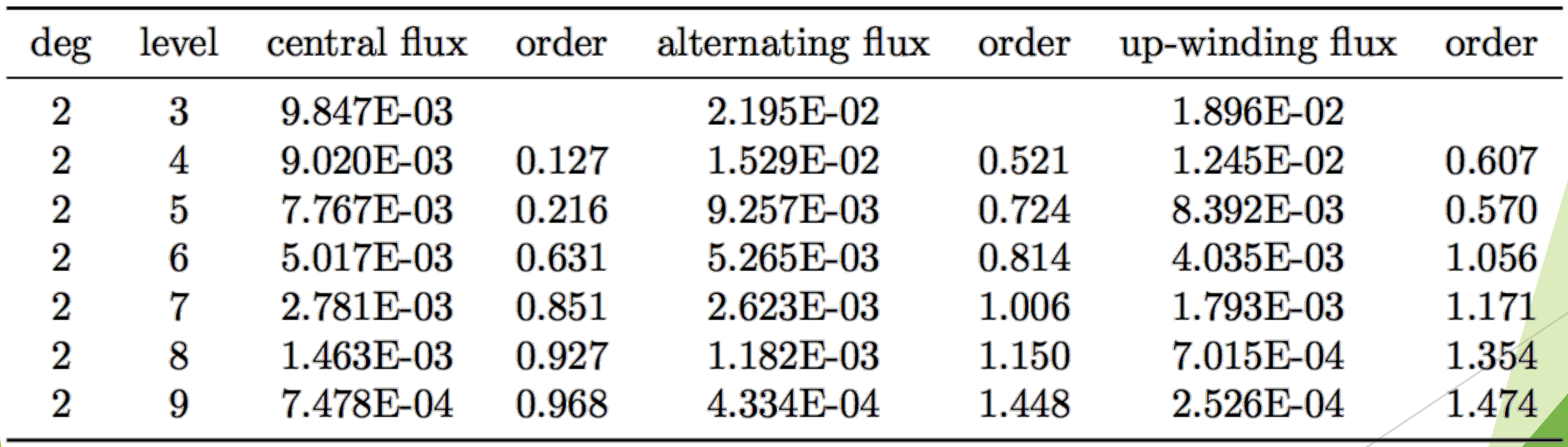
deg	level	sparse grid spectral radius	sparse grid CFL	full grid spectral radius	full grid CFL
2	3	3.271E+01	0.1059	5.521E+01	0.0627
2	4	6.428E+01	0.0679	1.104E+02	0.0395
3	3	6.614E+01	0.0524	1.115E+02	0.0311
3	4	1.298E+02	0.0336	2.230E+02	0.0196

3rd Order TVD Runge Kutta

Then we are able to choose the proper CFL and dt that are small enough to make the solution converge.

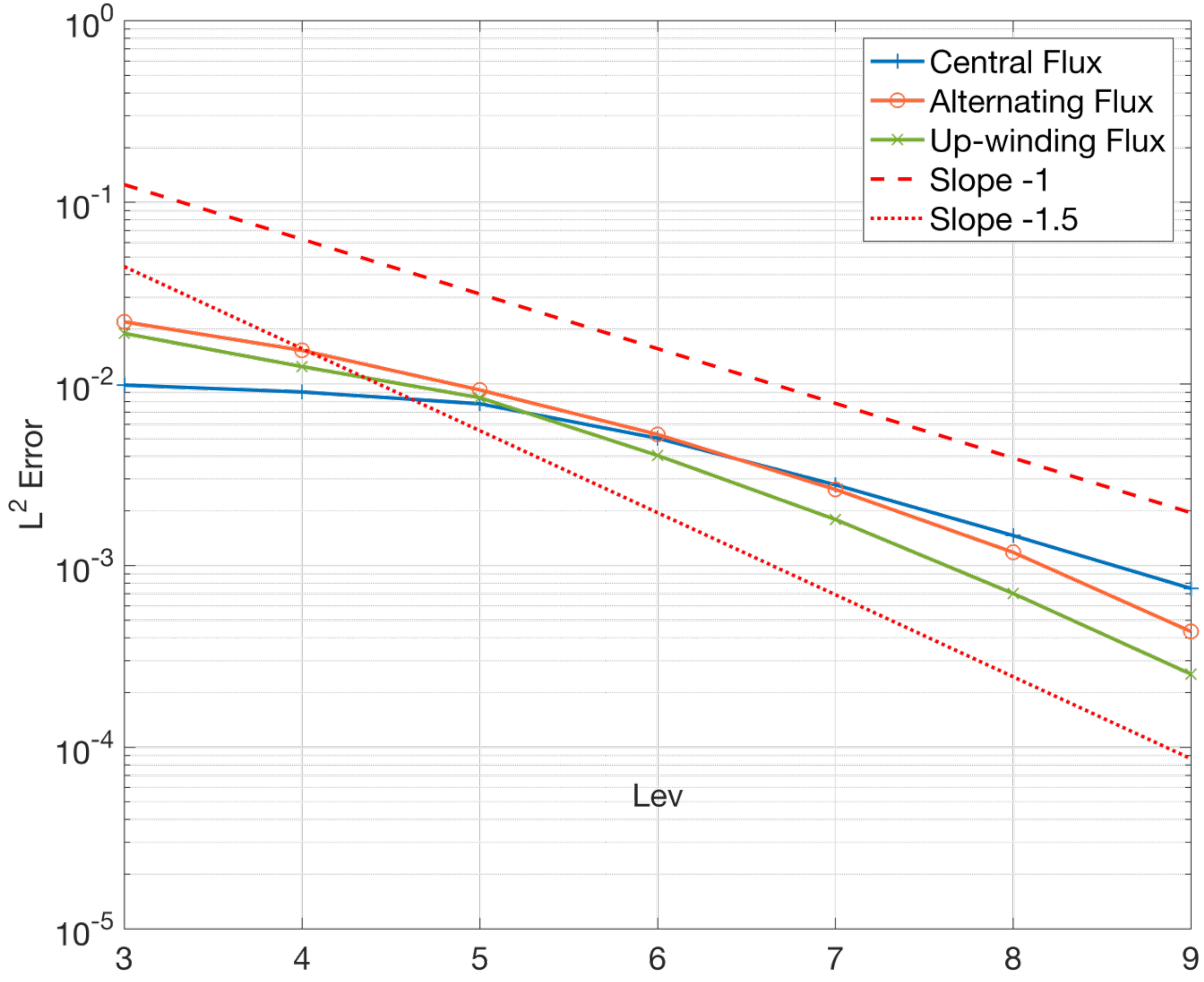
First we fix dt=1/10000 and MaxT=100

Table 1: The L2 error and accuracy of order with dt=1/1000 and MaxT=100



deg	level	central flux	order	alternating flux	order	up-winding flux	order
2	3	9.847E-03		2.195E-02		1.896E-02	
2	4	9.020E-03	0.127	1.529E-02	0.521	1.245E-02	0.607
2	5	7.767E-03	0.216	9.257E-03	0.724	8.392E-03	0.570
2	6	5.017E-03	0.631	5.265E-03	0.814	4.035E-03	1.056
2	7	2.781E-03	0.851	2.623E-03	1.006	1.793E-03	1.171
2	8	1.463E-03	0.927	1.182E-03	1.150	7.015E-04	1.354
2	9	7.478E-04	0.968	4.334E-04	1.448	2.526E-04	1.474

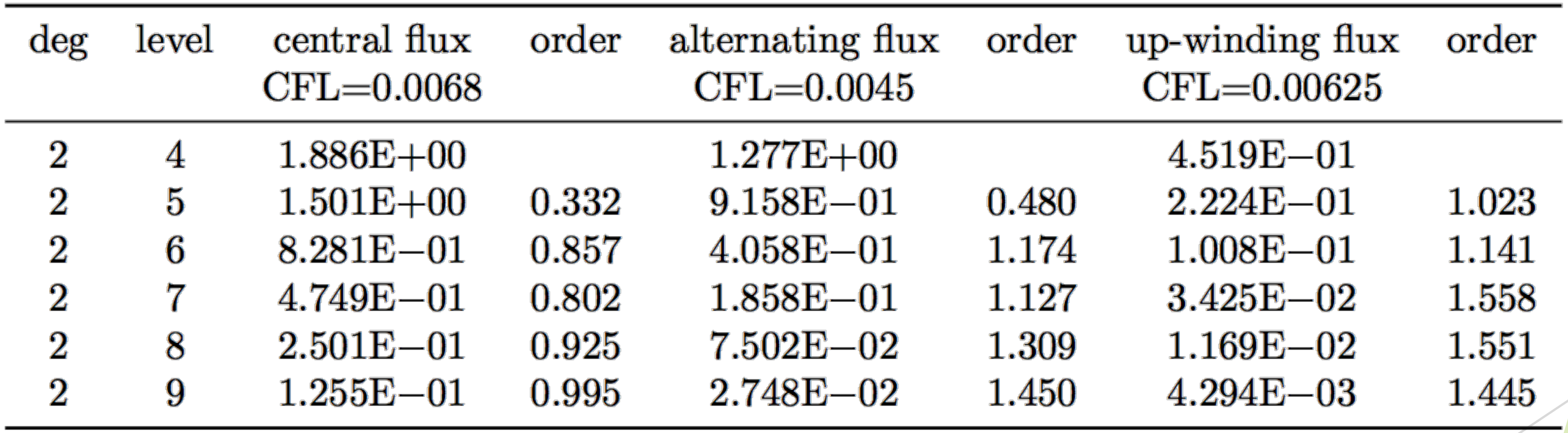
3rd Order TVD Runge Kutta



3rd Order TVD Runge Kutta

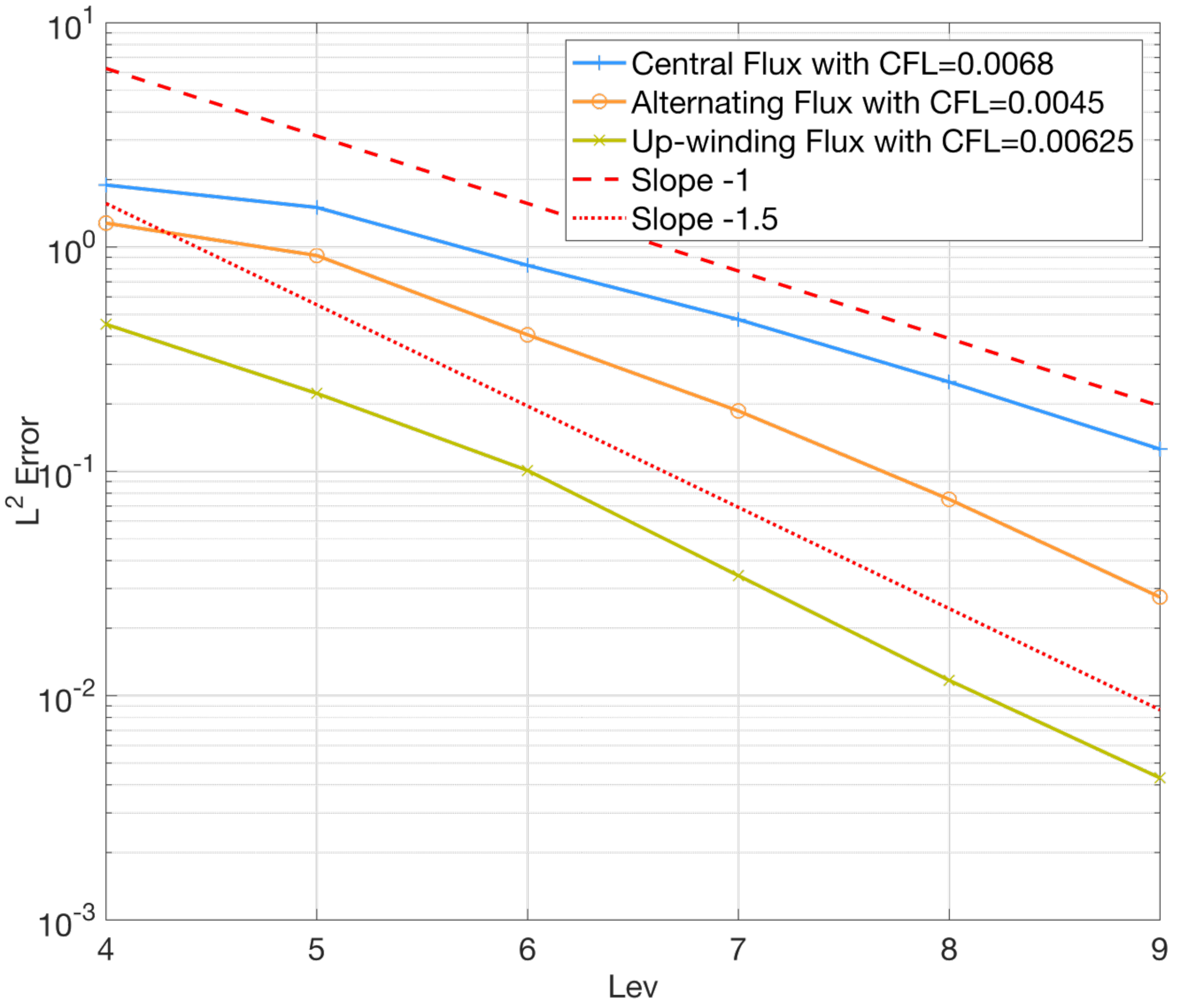
Next we fix Time period and set different CFL number for different numerical flux

Table 2: The L2 error and accuracy of order with time period=1



deg	level	central flux	order	alternating flux	order	up-winding flux	order
		CFL=0.0068		CFL=0.0045		CFL=0.00625	
2	4	1.886E+00		1.277E+00		4.519E−01	
2	5	1.501E+00	0.332	9.158E−01	0.480	2.224E−01	1.023
2	6	8.281E−01	0.857	4.058E−01	1.174	1.008E−01	1.141
2	7	4.749E−01	0.802	1.858E−01	1.127	3.425E−02	1.558
2	8	2.501E−01	0.925	7.502E−02	1.309	1.169E−02	1.551
2	9	1.255E−01	0.995	2.748E−02	1.450	4.294E−03	1.445

3rd Order TVD Runge Kutta



3rd Order TVD Runge Kutta

Then we can have some conclusion of the CFL bounded condition and convergence rate for dx for different numerical flux

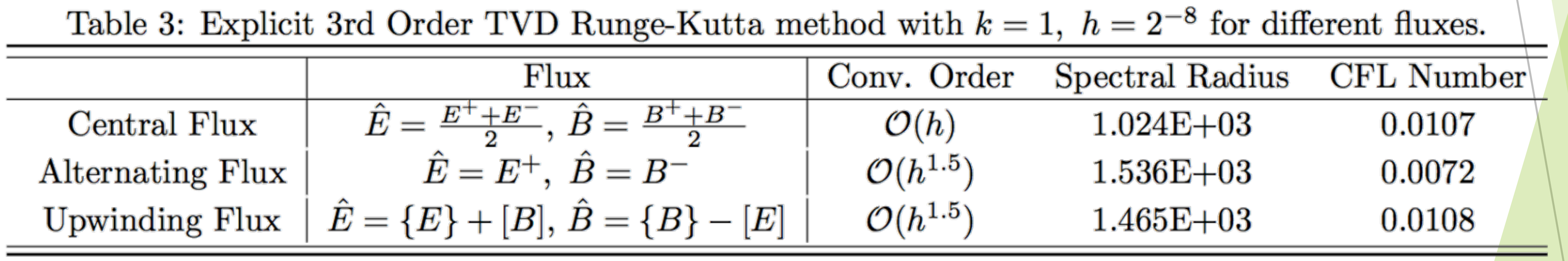
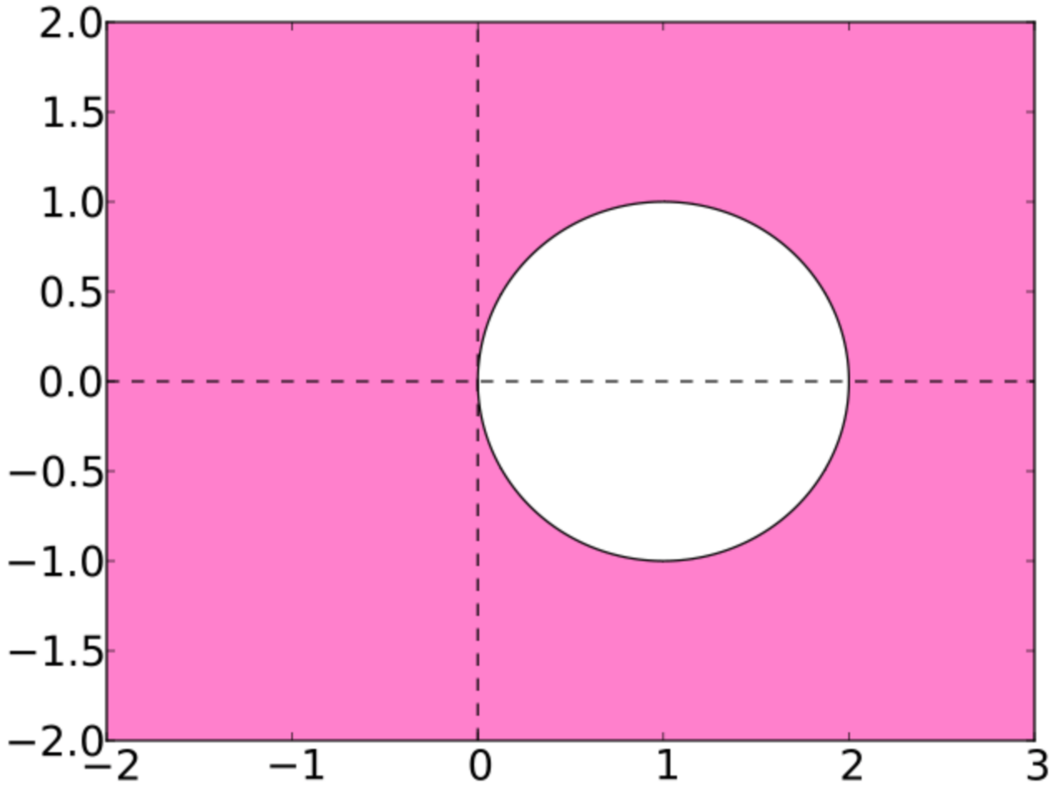


Table 3: Explicit 3rd Order TVD Runge-Kutta method with $k = 1,\ h = 2^{-8}$ for different fluxes.

	Flux	Conv. Order	Spectral Radius	CFL Number
Central Flux	$\hat{E} = \frac{E^+ + E^-}{2},\ \hat{B} = \frac{B^+ + B^-}{2}$	$\mathcal{O}(h)$	1.024E+03	0.0107
Alternating Flux	$\hat{E} = E^+,\ \hat{B} = B^-$	$\mathcal{O}(h^{1.5})$	1.536E+03	0.0072
Upwinding Flux	$\hat{E} = \{E\} + [B],\ \hat{B} = \{B\} - [E]$	$\mathcal{O}(h^{1.5})$	1.465E+03	0.0108

Backward Euler

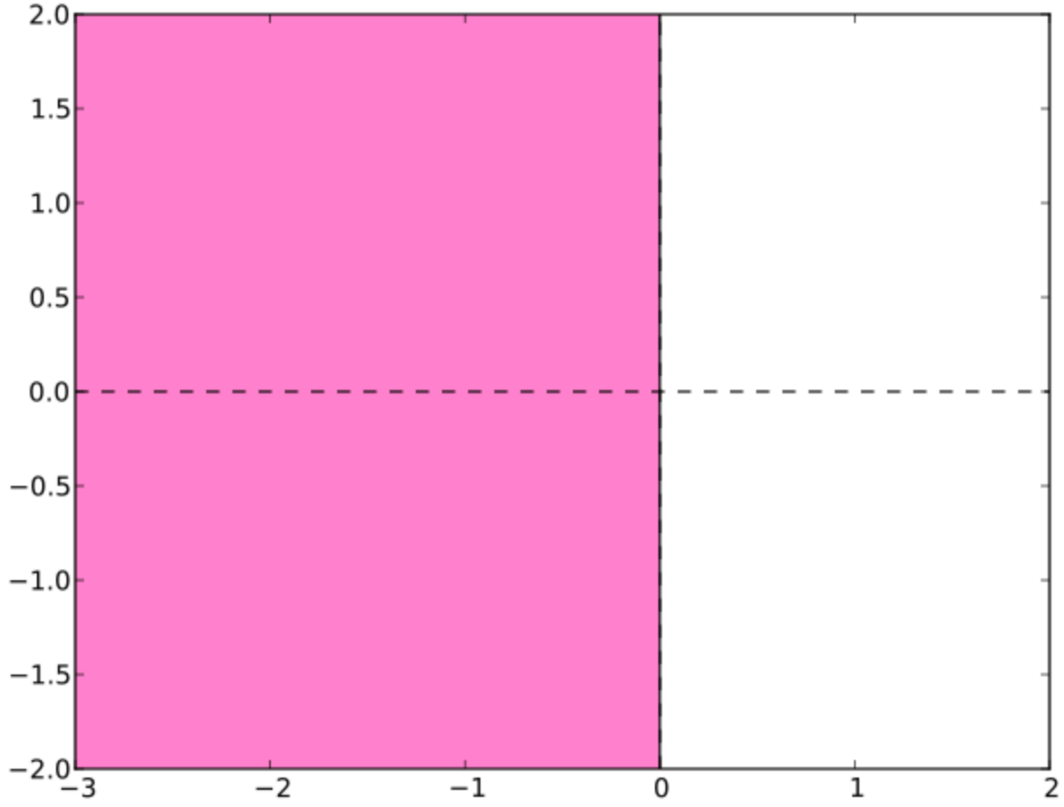
$$G_h^{n+1} = G_h^n + \Delta t R(G_h^{n+1})$$



For Backward Euler Method, the pink region in the complex plane is the stable region, and according to the eigenvalue with largest magnitude we have computed, we can see that it is always stable for us

Trapezoidal Rule

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



Similar to the previous method, the pink region in the complex plane is the stable region for trapezoidal rule, and according to the eigenvalue with largest magnitude we have computed, we can see that it is always stable for us

Trapezoidal Rule

Then we choose Trapezoidal rule and to see the L^2 error with different time period

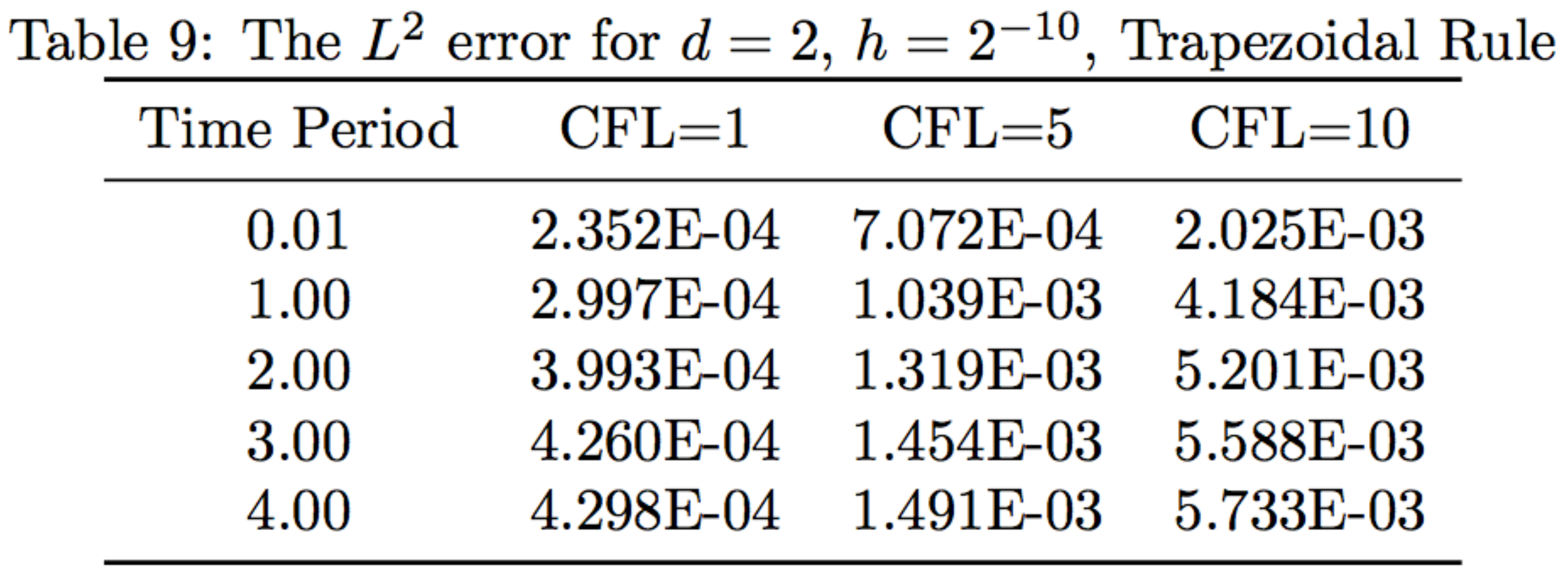
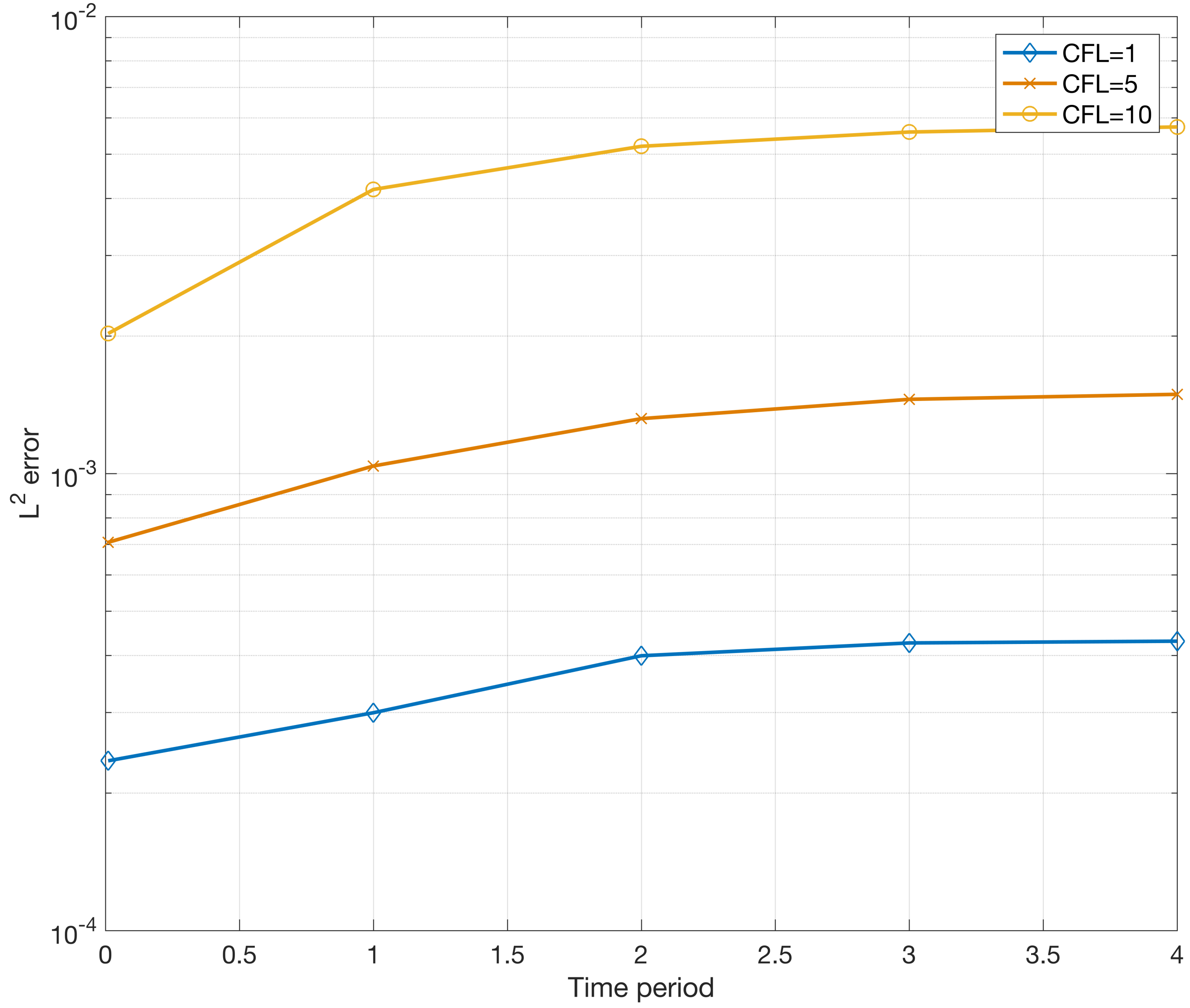


Table 9: The L^2 error for $d = 2$, $h = 2^{-10}$, Trapezoidal Rule

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

Trapezoidal Rule



4th Order Gauss-Legendre Method

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

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

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

Semi-Implicit Backward Euler Method

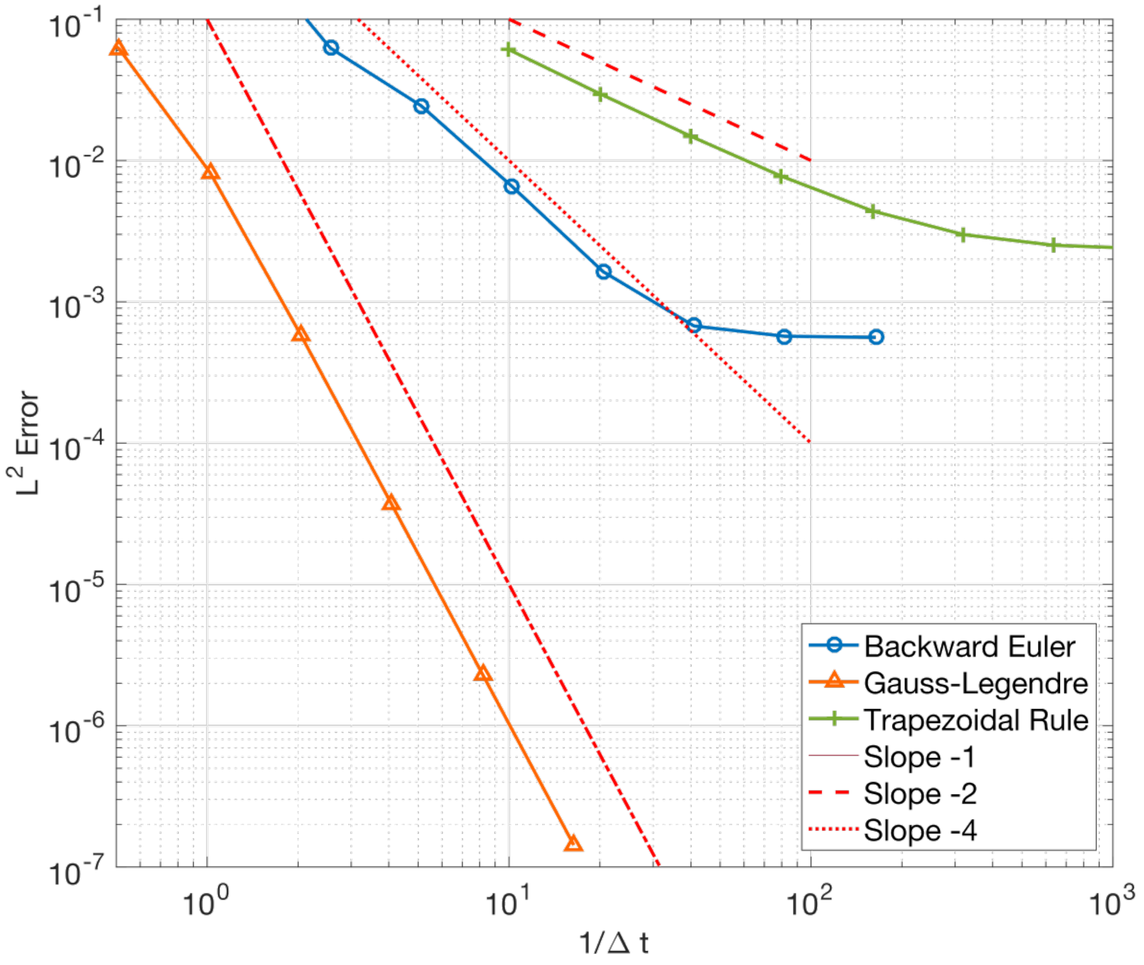
$$B_h^{n+1} = B_h^n + \Delta t R_1(E_h^n)$$

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

For Semi-Implicit Method, we use explicit time advanced method for B^n to get B^(n+1), and use B^(n+1) to get E^(n+1). The limitation of this method is that it also has a bounded condition for CFL number, but its upper bound is larger than that of explicit method.

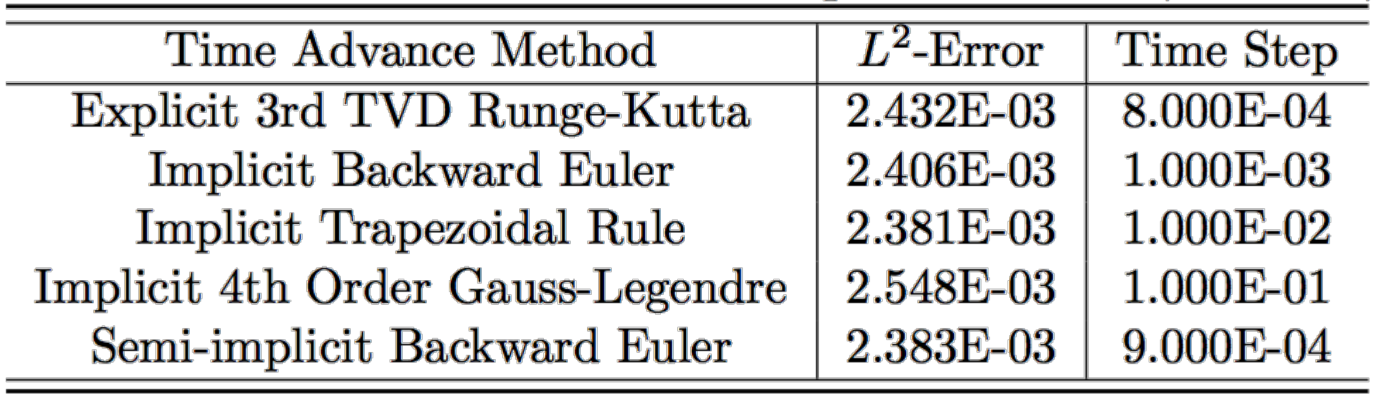
Implicit Time Advanced Method

Backward Euler: dt=dx*CFL (First Order)
Trapezoidal Rule: dt=dx^(1/2)*CFL (Second Order)
Gauss-Legendre Rule: dt=dx^(1/4)*CFL(Fourth Order)



Comparison and limitation

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

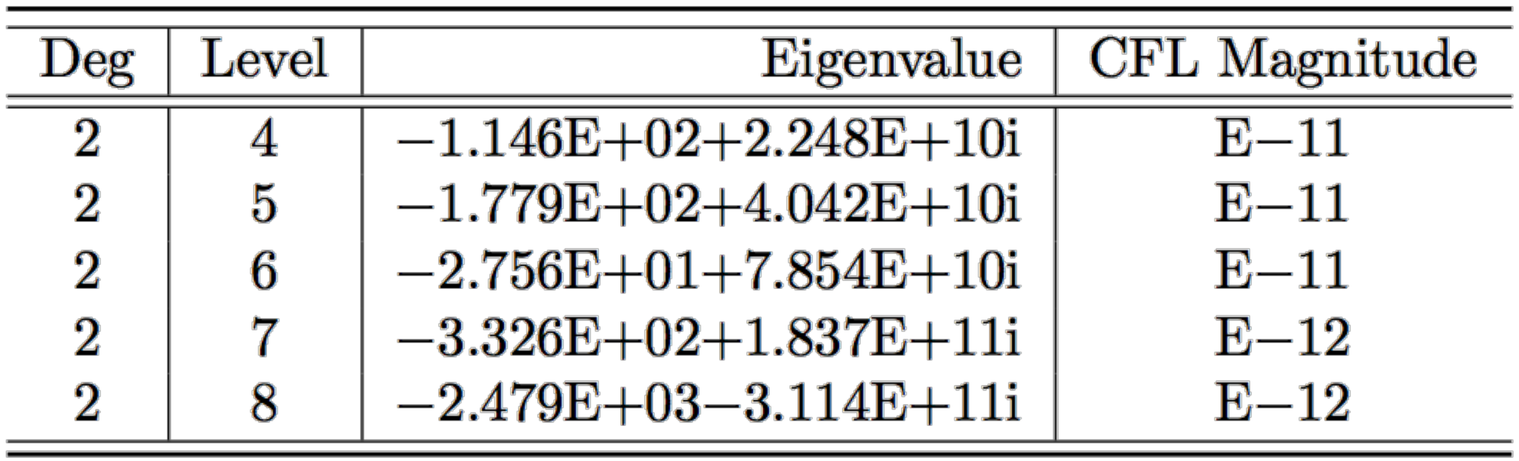


Time Advance Method	L^2-Error	Time Step
Explicit 3rd TVD Runge-Kutta	2.432E-03	8.000E-04
Implicit Backward Euler	2.406E-03	1.000E-03
Implicit Trapezoidal Rule	2.381E-03	1.000E-02
Implicit 4th Order Gauss-Legendre	2.548E-03	1.000E-01
Semi-implicit Backward Euler	2.383E-03	9.000E-04

We can see that the implicit scheme can have rather larger dt than explicit and semi-implicit scheme, but actually the CPU time is not so much less than that of explicit method. The calculation of inverse matrix is a big problem.

Future work

But for the real case that we need epsilon=8.8542x10^(-12) and mu=1.2566x10^(-6), which means that the CFL number should be very large and we need many times of iterations. In this case, if we can have a more efficient way to compute the inverse matrix, maybe using the pre-conditoner method, than we are able to reduce the calculation significantly.



Deg	Level	Eigenvalue	CFL Magnitude
2	4	$-1.146E+02+2.248E+10i$	$E-11$
2	5	$-1.779E+02+4.042E+10i$	$E-11$
2	6	$-2.756E+01+7.854E+10i$	$E-11$
2	7	$-3.326E+02+1.837E+11i$	$E-12$
2	8	$-2.479E+03-3.114E+11i$	$E-12$

Thanks!