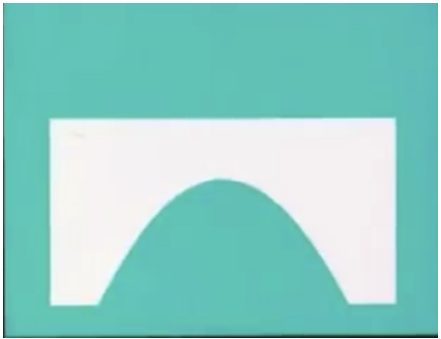
FEM Multigrid

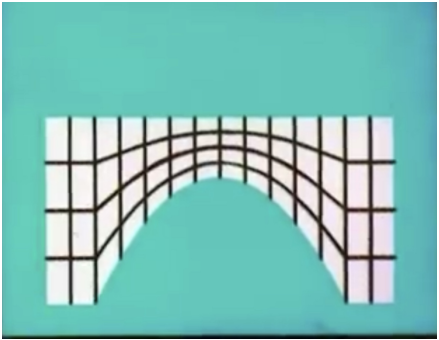
Trevor Crupi
Yonah Moise
Hannah Odom

Finite Element Method

Definition

The Finite Element Method (FEM) is a particular numerical method for solving partial differential equations in (typically two or three) space variables

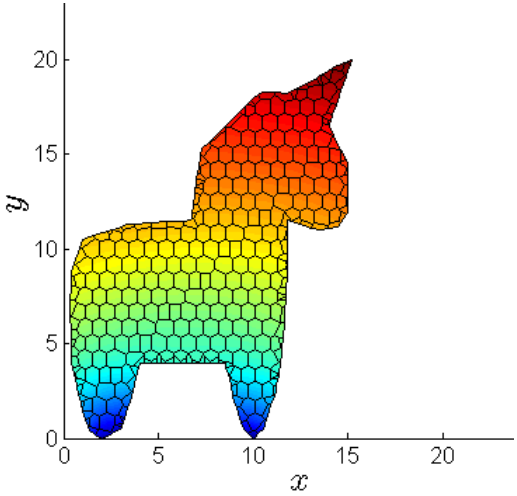




Meshes & Domains

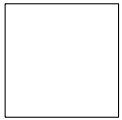
Definition

A *mesh* is a set of subdivisions of a chosen domain $\Omega \subset \mathbb{R}^n$ into discrete geometric cells



Meshes & Domains

Our two-dimension domain is the unit square



We divide our domain into different mesh levels. Mesh levels are dependent on the number of cells created.

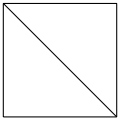


Figure: Mesh Level 1

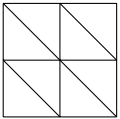


Figure: Mesh Level 2

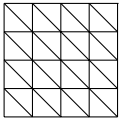


Figure: Mesh Level 3

Function Spaces

Definition

$L^2(\Omega) = \{f : \Omega \to \mathbb{R} : \int_\Omega |f|^2 dV < \infty\}$

L^2 is the space of all square integrable functions.

Definition

$H^1(\Omega) = \{f \in L^2(\Omega) : \frac{\partial f}{\partial x_i} \in L^2(\Omega), 1 \leq i \leq n\}$

H^1 is the subset of all functions in L^2 that have first derivatives in L^2. Think of H^1 as the space of functions in L^2 with nice first derivatives

Polynomial Spaces

Definition

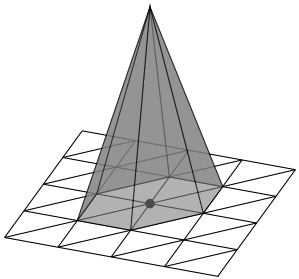
$V_h = \{v_h : v_h\big|_T = ax + by + c, \forall T \in \tau_h$, and v_h is globally continuous$\}$, where $a, b, c \in \mathbb{R}$ and τ_h is the triangulation of our domain Ω

V_h is the set of all functions that, when restricted to one triangle in our mesh, can be written as $ax + by + c$.

Polynomial Spaces

$V_h = \{\varphi_i\}_{i=1}^{N}$, where N is the number of nodes

$$\varphi_i(v_j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases}$$



Weak Derivatives

Weak derivatives are a generalization of the classic derivative concept. They are similar; however, there are certain inputs that will yield a result from the weak derivative but not the strong derivative.
For example:

Our starting equation requires that function u be double differentiable by x while the final equation does not have this restriction.

$$
\begin{aligned}
-u_{xx} + u &= f \\
v(-u_{xx} + u) &= vf \\
\int v(-u_{xx} + u) &= \int vf \\
-vu_x|_{-\infty}^{\infty} + \int v_x u_x + \int vu &= \int vf \\
\int v_x u_x + \int vu &= \int vf \\
\int \nabla u \nabla v + uv &= \int vf
\end{aligned}
$$

Reaction-Diffusion Equation

Our goal is to find some $u \in H^1(\Omega)$ that satisfies our the *The Reaction-Diffusion Equation*:

$$-\Delta u + u = f$$

using *Neumann Boundary Conditions*:

$$\nabla u \cdot \vec{n} = 0 \text{ on } \partial\Omega$$

FEM & Polynomial Spaces

Our weak form of the previous equation:

$$\int_\Omega \nabla u \cdot \nabla v + uv = \int_\Omega fv \quad \forall v \in H^1(\Omega)$$

However $H^1(\Omega)$, being infinite-dimensional, is difficult to work with. Therefore, we approximate by letting $u_h, v_h \in V_h$, where V_h is finite, and seek u_h and v_h such that

$$\int_\Omega \nabla u_h \cdot \nabla v_h + u_h v_h = \int_\Omega f v_h \quad \forall v_h \in V_h.$$

FEM & Polynomial Spaces

We define u_h and v_h in terms of the basis function, φ:

$$u_h = \sum_{j=1}^{n} c_j \cdot \varphi_j \quad , \quad v_h = \varphi$$

And use these in our weak formulation:

$$\int_\Omega \sum_{j=1}^{n} (c_j \cdot \nabla\varphi_j \cdot \nabla\varphi_i + c_j \cdot \varphi_j \cdot \varphi_i) = \int_\Omega f\varphi$$

FEM & Polynomial Spaces

We transform the equation into matrix-vector form $\mathbf{A}\vec{c} = \vec{b}$ where:

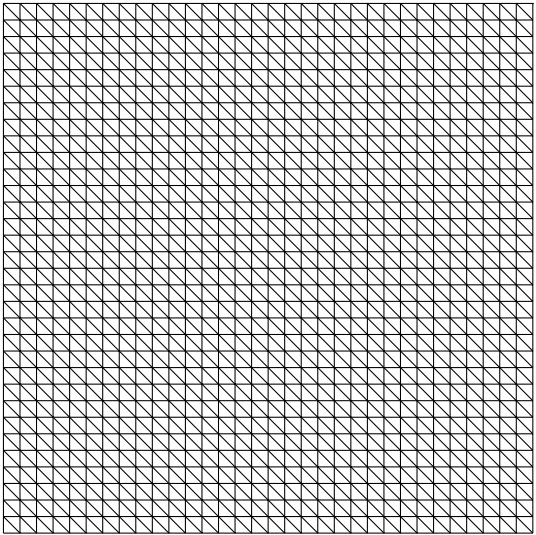
$$A_{i,j} = \int_\Omega \nabla\varphi_i \cdot \nabla\varphi_j + \varphi_i\varphi_j \quad , \quad b_i = \int_\Omega f\varphi_i$$

$$\begin{bmatrix} \int_\Omega(\nabla\varphi_1 \cdot \nabla\varphi_1 + \varphi_1 \cdot \varphi_1) & \dots & \int_\Omega(\nabla\varphi_n \cdot \nabla\varphi_1 + \varphi_n \cdot \varphi_1) \\ \int_\Omega(\nabla\varphi_1 \cdot \nabla\varphi_2 + \varphi_1 \cdot \varphi_2) & \dots & \int_\Omega(\nabla\varphi_n \cdot \nabla\varphi_2 + \varphi_n \cdot \varphi_2) \\ . & \dots & . \\ . & \dots & . \\ . & \dots & . \\ \int_\Omega(\nabla\varphi_1 \cdot \nabla\varphi_n + \varphi_1 \cdot \varphi_n) & \dots & \int_\Omega(\nabla\varphi_n \cdot \nabla\varphi_n + \varphi_n \cdot \varphi_n) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ . \\ . \\ . \\ c_n \end{bmatrix} = \begin{bmatrix} \int_\Omega f\varphi_1 \\ \int_\Omega f\varphi_2 \\ . \\ . \\ . \\ \int_\Omega f\varphi_n \end{bmatrix}$$

Now we solve for constants within the c-vector

FEM & Polynomial Spaces

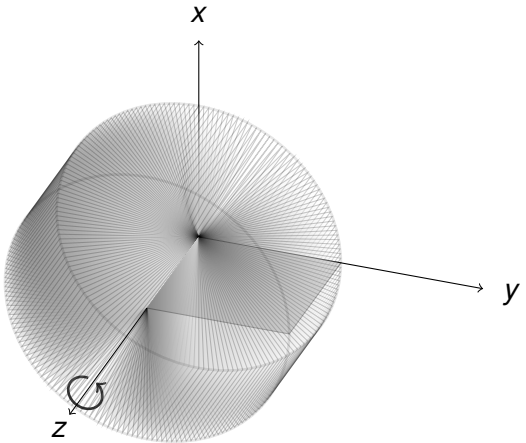
How do we solve this matrix system more efficiently?

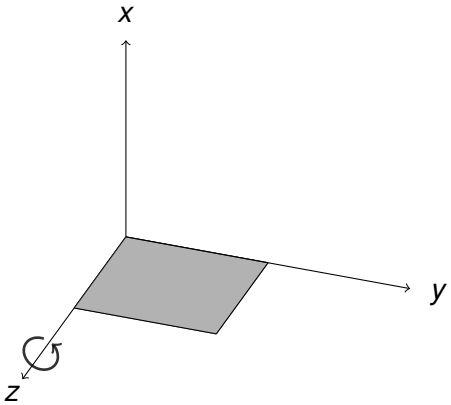


Axisymmetric Domains

Axisymmetric Domains

An axisymmetric domain $\breve{\Omega}$ is a domain symmetric about an axis. Due to this symmetry, we can reduce our problem to the 2D domain Ω (the *rz* plane).





Fourier Decomposition

Let $\breve{\Omega}$ be an axisymmetric domain. For a vector-valued function $u \in L^2(\breve{\Omega})$. the Fourier decomposition is $u = u^s + u^a$, given by:

$$u^s = \begin{bmatrix} u_r^0 \\ 0 \\ u_z^0 \end{bmatrix} + \sum_{k=1}^{\infty} \begin{bmatrix} u_r^k cos(k\theta) \\ u_\theta^k sin(k\theta) \\ u_z^k cos(k\theta) \end{bmatrix}$$

$$u^a = \begin{bmatrix} 0 \\ u_\theta^0 \\ 0 \end{bmatrix} + \sum_{k=1}^{\infty} \begin{bmatrix} u_r^{-k} sin(k\theta) \\ u_\theta^{-k} cos(k\theta) \\ u_z^{-k} sin(k\theta) \end{bmatrix}$$

A specific value of k is referred to as the k^{th} *order Fourier mode*.

Curl Operators

The curl operator in cylindrical coordinates can be written:

$$curl\ \vec{u} = \begin{bmatrix} \frac{1}{r}\frac{\partial u_z}{\partial \theta} - \frac{\partial u_\theta}{\partial z} \\ \frac{\partial u_r}{\partial z} - \frac{\partial u_z}{\partial r} \\ \frac{1}{r}\left(\frac{\partial u_\theta}{\partial r} - \frac{\partial u_r}{\partial \theta}\right) \end{bmatrix} \tag{1}$$

If we apply it to the Fourier decomposition $u^s + u^a$, we get:

$$curl^k_{rz}\vec{u} = \begin{bmatrix} -\frac{k}{r}u_z - \partial_z u_\theta \\ \partial_z u_r - \partial_r u_z \\ \partial_r u_\theta + \frac{1}{r}u_\theta + \frac{k}{r}u_r \end{bmatrix} \tag{2}$$

Weighted Spaces

Due to the Jacobian from the change of coordinates $dxdydz \mapsto rdrd\theta dz$, we obtain *weighted* function spaces:

$$L^2_r(\Omega) = \{\vec{u} : \int_\Omega \vec{u}^2 r dr dz < \infty\} \tag{3}$$

Note that $L^2(\Omega) \subset L^2_r(\Omega)$. We can also define the Sobolev space based on the differential operator $curl^k_{rz}$:

$$H_r(curl^k_{rz}; \Omega) = \{\vec{u} \in L^2_r(\Omega) : curl^k_{rz}\vec{u} \in L^2_r(\Omega)\} \tag{4}$$

The Space W_h

Recall V_h:

$$V_h = \{v_h : v_h\big|_T = ax + by + c, \forall T \in \tau_h, \text{ and } v_h \text{ is globally continuous}\} \tag{5}$$

Where e is an edge and t is the unit tangent vector.

$$W_h = \{w_h : w_h|_T = \begin{bmatrix} B - Ay \\ C + Ax \end{bmatrix}, \int_e w_h \cdot t \, dS \text{ is continuous } \} \tag{6}$$

Our basis for this space is:

$$\int_{e_j} \psi_i \cdot t \, dS = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \tag{7}$$

The Space $Z_{h,k}$

We introduce a new finite element space $Z_{h,k}$:

$$Z_{h,k} = \left\{ z_h : z_h\big|_T = \begin{bmatrix} -\frac{1}{k}\beta_1 + \beta_4 x - \frac{1}{k}\beta_3 y - \beta_6 xy \\ \beta_1 + \beta_2 x + \beta_3 y \\ \beta_5 x + \beta_6 x^2 \end{bmatrix}, \beta_i \in \mathbb{R}, \forall 1 \leq i \leq 6, \forall T \in \tau_h \right\}$$

The space $Z_{h,k}$ has a basis $\eta_{i=1}^{n+N}$, $\eta_i = (\eta_i^r, \eta_i^\theta, \eta_i^z)^T$ satisfying the following constraints:

$$\eta_i^\theta(v_j) = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \qquad \int_{e_j} \begin{bmatrix} \frac{k\eta_i^r + \eta_i^\theta}{x} \\ \frac{k\eta_i^z}{x} \end{bmatrix} \cdot \vec{t} ds = \begin{cases} 1 & \text{if } i = n + j \\ 0 & \text{if } i \neq n + j \end{cases}$$

η Basis Functions

The basis functions can be written:

$$\eta = \begin{bmatrix} -\frac{1}{k}\beta_1 + \beta_4 r - \frac{1}{k}\beta_3 z - \beta_6 rz \\ \beta_1 + \beta_2 r + \beta_3 z \\ \beta_5 r + \beta_6 r^2 \end{bmatrix}$$

We can use the constraints to solve for $\beta_4, \beta_5, \beta_6$ and set $\beta_1, \beta_2, \beta_3 = 0$ we obtain:

$$\varphi_i = \begin{bmatrix} -\frac{c_i}{k} - \frac{a_i}{k} r - \frac{b_i}{k} z \\ a_i r + b_i z + c_i \\ 0 \end{bmatrix} \qquad \psi_i = \begin{bmatrix} \frac{B_i}{k} r - \frac{A_i}{k} rz \\ 0 \\ \frac{C_i}{k} r + \frac{A_i}{k} r^2 \end{bmatrix}$$

We will let $X_i = \{\varphi_i, \psi_j\}_{i=1,j=1+N}^{i=N,j=n+N}$.

Integral Form of Our Problem

Given a domain Ω and $F \in L^2_r(\Omega)$, we wish to find a $u \in H_r(curl^k_{rz}; \Omega)$ such that:

$$\int_\Omega (curl^k_{rz} u \cdot curl^k_{rz} v + u \cdot v) r dr dz = \int_\Omega (F \cdot v) r dr dz \tag{8}$$

For all $v \in H_r(curl^k_{rz}; \Omega)$. We use boundary conditions:

$$\begin{cases} (curl^k_{rz} \vec{u})_{rz} \cdot \vec{t} = 0 & \text{on } \Gamma_1 \\ (curl^k_{rz} \vec{u})_\theta = 0 & \text{on } \Gamma_1 \end{cases} \tag{9}$$

Breaking Down the Problem

Weak Formulation:

$$\int_{\Omega} (curl_{rz}^{k} u \cdot curl_{rz}^{k} v + u \cdot v) r dr dz = \int_{\Omega} (F \cdot v) r dr dz \tag{10}$$

Breaking Down the Problem

Weak Formulation:

$$\int_{\Omega} (curl_{rz}^{k} u \cdot curl_{rz}^{k} v + u \cdot v) r dr dz = \int_{\Omega} (F \cdot v) r dr dz \tag{11}$$

Approximation:

$$\int_{\Omega} (curl_{rz}^{k} u_h \cdot curl_{rz}^{k} v_h + u_h \cdot v_h) r dr dz = \int_{\Omega} (F \cdot v_h) r dr dz \tag{12}$$

Breaking Down the Problem

Fill in:

$$u_h = \sum_{j=1}^{n+N} c_j \cdot X_j$$

Breaking Down the Problem

Fill in:

$$u_h = \sum_{j=1}^{n+N} c_j \cdot X_j$$

And we get:

$$\int_\Omega (\sum_{j=1}^{n+N} c_j \cdot curl_{rz}^k X_j \cdot curl_{rz}^k X + \sum_{j=1}^{n+N} c_j \cdot X_j \cdot X) r dr dz = \int_\Omega (F \cdot X_j) r dr dz \qquad (13)$$

Breaking Down the Problem

We can transform this into a matrix system by setting:

$$A_{ij} = \int_{\Omega} (curl^k_{rz} X_j \cdot curl^k_{rz} X + X_j \cdot X) r dr dz$$

$$b_i = \int_{\Omega} (F \cdot X_i) r dr dz$$

This gives us the matrix system:

$$Ac = b \tag{14}$$

FEM on Maxwell's Equations Results

Mesh Level	Weighted L_2 Error	Convergence Rate
1	0.008	0.25
2	0.007	1.70
3	0.002	1.90
4	0.0006	1.99
5	0.0001	2.00

$$u(r,z) = \begin{bmatrix} z - \frac{1}{k}(\frac{1}{3}r^3 - \frac{1}{2}r^2) \\ -kz + \frac{1}{3}r^3 - \frac{1}{2}r^2 \\ r \end{bmatrix}, \; F(r,z) = \begin{bmatrix} k(r-1) + u_r(r,z) \\ -2r + 1 + u_\theta(r,z) \\ u_z(r,z) \end{bmatrix}, \tag{15}$$

Solving For The C-Vector

$$Ac = b \tag{16}$$

$$Ac = b \tag{16}$$

This was easy to solve – using the Matlab backslash-operator:

$$c = A \backslash b$$

$$Ac = b \tag{16}$$

This was easy to solve – using the Matlab backslash-operator:

$$c = A \backslash b$$

However, Gaussian elimination is slow and inefficient when dealing with large matrices.

$$Ac = b \tag{16}$$

This was easy to solve – using the Matlab backslash-operator:

$$c = A \backslash b$$

However, Gaussian elimination is slow and inefficient when dealing with large matrices.

Now: we will use iterative methods to solve this system.

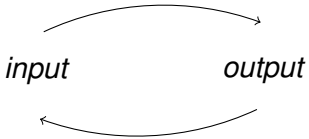
Iterative Methods

Iterative Methods

Implement some algorithm over-and-over again, refining the input each time.

Iterative Methods

Implement some algorithm over-and-over again, refining the input each time.

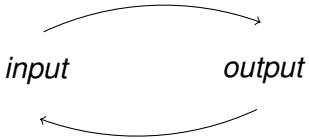


Idea of the Structure

while Stopping-criterion is not fulfilled **do**

Iterative Methods

Implement some algorithm over-and-over again, refining the input each time.

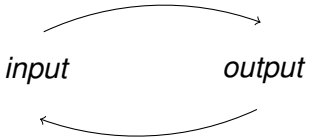


Idea of the Structure

while Stopping-criterion is not fulfilled **do**
    Process the input
    Return to top
end while

Iterative Methods

Implement some algorithm over-and-over again, refining the input each time.



Idea of the Structure

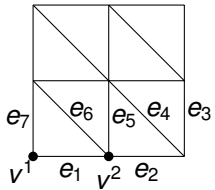
while Stopping-criterion is not fulfilled **do**
    Process the input
    Return to top
end while

We first implemented the Multiplicative Subspace Correction Method.

The Multiplicative Subspace Correction (MSC) Method

Subspace decomposition

$$Z_h = \sum_{l=1}^{N} Z_l$$



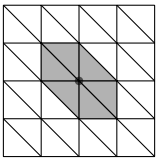


Figure: A Vertex Patch

For example:

$$Z_1 = \text{span}(\varphi_1, \psi_1, \psi_7)$$

The MSC-Method

$U_{i+1} = MSC(u_i, f)$

1) $u_i^{(0)} = u_i$

2) For j = 1, 2, . . ., N

$$u_i^{(j)} = u_i^{(j-1)} + A_j^{-1} Q_j (f - A u_i^{(j-1)})$$

3) $u_{i+1} = u_i^{(N)}$

Results of MSC-Method

Mesh Level	Number of Iterations	Convergence Rate
2	65	0.59
3	230	0.87
4	864	0.96
5	3274	0.99
6	12466	1

MSC $\rightarrow$ Multigrid

MSC-method

- is very slow
- requires many iterations
- eliminates high-frequency errors quickly.

Instead, we will apply the Multigrid method:

- much faster than MSC-method
- incorporates one iteration of MSC-Method to eliminate high-frequency errors.

The Multigrid Method

Multigrid V-Cycle

Uses a sequence of nested meshes to solve Ac=b

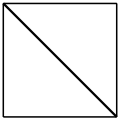


Figure: Mesh Level 1

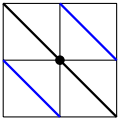


Figure: Mesh Level 2

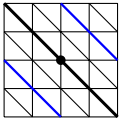


Figure: Mesh Level 3

Multigrid V-Cycle

Uses a sequence of nested meshes to solve Ac=b

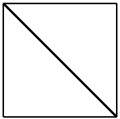


Figure: Mesh Level 1

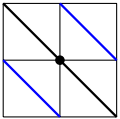


Figure: Mesh Level 2

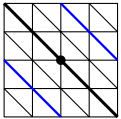


Figure: Mesh Level 3

$$Z^1_{h,k} \subset Z^2_{h,k} \subset Z^3_{h,k}$$

Multigrid V-Cycle

Set $MG_1(u, f) = A_1^{-1} f$.

Multigrid V-Cycle

Set $MG_1(u, f) = A_1^{-1} f$.

For $s > 1$, define $MG_s(u, f)$ recursively:

1) $V^{(1)}$: One iteration of $MSC(u, f)$
2) $V^{(2)} = V^{(1)} + MG_{s-1}(0, Q_{s-1}(f - A_s V^{(1)}))$
3) $V^{(3)}$: One iteration of $MSC^T(V^{(2)}, f)$.

Multigrid V-Cycle

Set $MG_1(u, f) = A_1^{-1} f$.

For $s > 1$, define $MG_s(u, f)$ recursively:

1) $V^{(1)}$: One iteration of $MSC(u, f)$
2) $V^{(2)} = V^{(1)} + MG_{s-1}(0, Q_{s-1}(f - A_s V^{(1)}))$
3) $V^{(3)}$: One iteration of $MSC^T(V^{(2)}, f)$.

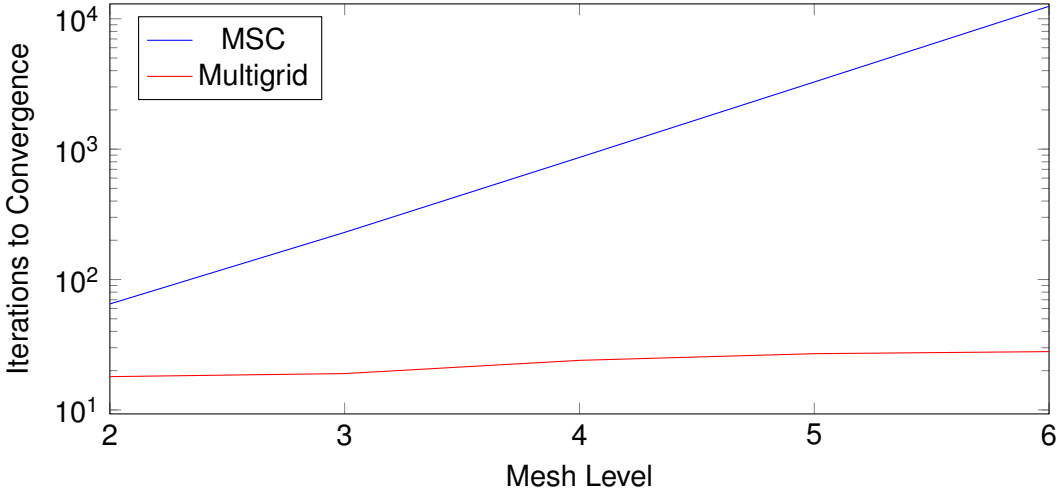
Also, for $s > 1$, we iterate over these steps multiple times – in order to refine u into a more-accurate approximation of the real solution.

Results of the Iterative Methods

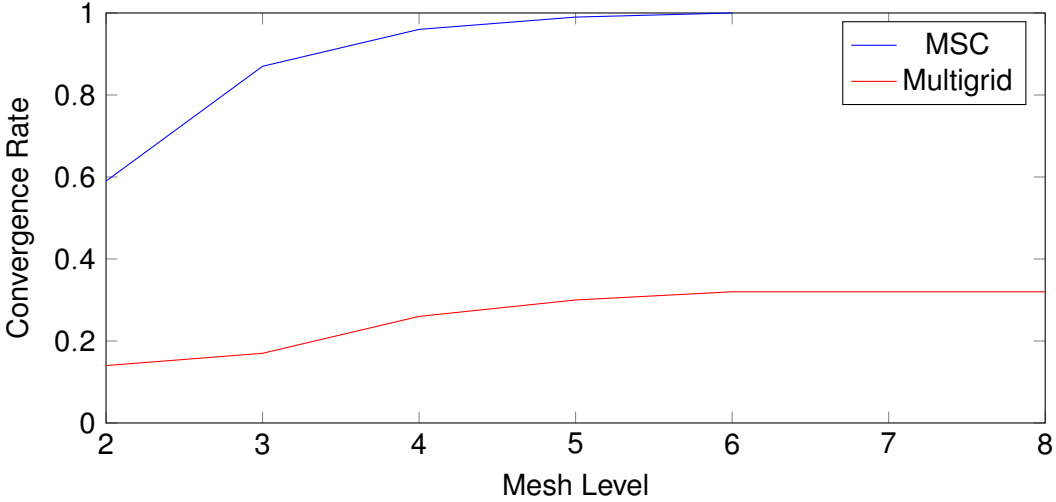
Multigrid Results for $k = 1$

Mesh Level	Number of Iterations L_2	Convergence Rate
2	18	0.14
3	19	0.17
4	24	0.26
5	27	0.30
6	28	0.32
7	27	0.32
8	27	0.32

Iterations to Convergence Results Comparison



Convergence Rate Results Comparison



Convergence Rate Comparison Between Various Fourier Modes

Mesh Level	$k = 1$	$k = 2$	$k = 3$	$k = 4$	$k = 5$
2	0.14	0.11	0.08	0.05	0.06
3	0.17	0.14	0.12	0.11	0.09
4	0.26	0.23	0.19	0.16	0.15
5	0.30	0.30	0.29	0.28	0.27
6	0.32	0.32	0.31	0.31	0.31
7	0.32	0.32	0.32	0.32	0.32
8	0.32	0.32	0.32	0.32	0.32

Concluding Remarks

Mathematical analysis of Multigrid on $H_r(curl^k_{rz})$ using our subspace decomposition has yet to be proven mathematically, however we have shown numerically that the convergence is uniform.

For future work we will be extending our analysis to include the time harmonic Maxwell's equation and exploring the applications.