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.
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

\[ V_h = \{ v_h : v_h|_T = ax + by + c, \quad \forall T \in \tau_h, \text{ and } v_h \text{ is globally continuous} \}, \]

where \( a, b, c \in \mathbb{R} \) and \( \tau_h \) is the triangulation of our domain \( \Omega \).

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

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

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

\[ \int_{\Omega} \nabla u_h \cdot \nabla v_h + u_h v_h = \int_{\Omega} f v_h \quad \forall v_h \in V_h. \]
We define $u_h$ and $v_h$ in terms of the basis function, $\phi$:

$$
U_h = \sum_{j=1}^{n} c_j \cdot \phi_j, \quad V_h = \phi
$$

And use these in our weak formulation:

$$
\int_{\Omega} \sum_{j=1}^{n} (c_j \cdot \nabla \phi_j \cdot \nabla \phi_i + c_j \cdot \phi_j \cdot \phi_i) = \int_{\Omega} f \phi
$$
FEM & Polynomial Spaces

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

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

\[
\begin{bmatrix}
\int_{\Omega} (\nabla \phi_1 \cdot \nabla \phi_1 + \phi_1 \cdot \phi_1) & \cdots & \int_{\Omega} (\nabla \phi_n \cdot \nabla \phi_1 + \phi_n \cdot \phi_1) \\
\int_{\Omega} (\nabla \phi_1 \cdot \nabla \phi_2 + \phi_1 \cdot \phi_2) & \cdots & \int_{\Omega} (\nabla \phi_n \cdot \nabla \phi_2 + \phi_n \cdot \phi_2) \\
\cdot & \cdots & \cdot \\
\cdot & \cdots & \cdot \\
\cdot & \cdots & \cdot \\
\int_{\Omega} (\nabla \phi_1 \cdot \nabla \phi_n + \phi_1 \cdot \phi_n) & \cdots & \int_{\Omega} (\nabla \phi_n \cdot \nabla \phi_n + \phi_n \cdot \phi_n)
\end{bmatrix}
\begin{bmatrix}
c_1 \\
c_2 \\
\vdots \\
c_n
\end{bmatrix}
= 
\begin{bmatrix}
\int_{\Omega} f \phi_1 \\
\int_{\Omega} f \phi_2 \\
\cdot \\
\cdot \\
\cdot \\
\int_{\Omega} f \phi_n
\end{bmatrix}
\]

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 $\tilde{\Omega}$ is a domain symmetric about an axis. Due to this symmetry, we can reduce our problem to the 2D domain $\Omega$ (the $rz$ plane).
Fourier Decomposition

Let $\Omega$ be an axisymmetric domain. For a vector-valued function $u \in L^2(\Omega)$. 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:

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

(1)

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

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

(2)
Weighted Spaces

Due to the Jacobian from the change of coordinates $dx dy dz \mapsto r dr d\theta dz$, we obtain weighted function spaces:

$$L^2_r(\Omega) = \{ \bar{u} : \int_{\Omega} \bar{u}^2 r dr dz < \infty \}$$  \hspace{1cm} (3)

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

$$H_r(\text{curl}_{rz}^k; \Omega) = \{ \bar{u} \in L^2_r(\Omega) : \text{curl}_{rz}^k \bar{u} \in L^2_r(\Omega) \}$$  \hspace{1cm} (4)
### The Space $W_h$

Recall $V_h$:

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

Where $e$ is an edge and $\mathbf{t}$ is the unit tangent vector.

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

Our basis for this space is:

$$\int_{e_{ij}} \varphi_i \cdot \mathbf{t} \, dS = \begin{cases} 1 & \text{if } i = j \\ 0 & \text{if } i \neq j \end{cases} \quad (7)$$
The Space $Z_{h,k}$

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

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

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

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

\[
\varphi_i = \begin{bmatrix}
-\frac{c_i}{k} - \frac{a_i}{k} r - \frac{b_i}{k} z \\
\frac{a_i}{k} r + \frac{b_i}{k} z + c_i \\
0
\end{bmatrix}
\]

\[
\psi_i = \begin{bmatrix}
\frac{B_i}{k} r - \frac{A_i}{k} r^2 \\
0 \\
\frac{C_i}{k} r + \frac{A_i}{k} r^2
\end{bmatrix}
\]

We will let \(X_i = \{\varphi_i, \psi_j\}_{i=1}^{N}, j=n+N\).
Integral Form of Our Problem

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

$$\int_\Omega (\text{curl}^k_rz u \cdot \text{curl}^k_rz v + u \cdot v) r dr dz = \int_\Omega (F \cdot v) r dr dz$$

(8)

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

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

(9)
Breaking Down the Problem

Weak Formulation:

\[
\int_{\Omega} (\text{curl}_{rz}^{k} u \cdot \text{curl}_{rz}^{k} v + u \cdot v) r dr dz = \int_{\Omega} (F \cdot v) r dr dz
\] (10)
Breaking Down the Problem

Weak Formulation:

\[
\int_{\Omega} (\text{curl}_{rz}^k u \cdot \text{curl}_{rz}^k v + u \cdot v) r dr dz = \int_{\Omega} (F \cdot v) r dr dz \tag{11}
\]

Approximation:

\[
\int_{\Omega} (\text{curl}_{rz}^k u_h \cdot \text{curl}_{rz}^k v_h + u_h \cdot v_h) r dr dz = \int_{\Omega} (F \cdot v_h) r dr dz \tag{12}
\]
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:

\[
\int_{\Omega} \left( \sum_{j=1}^{n+N} c_j \cdot \text{curl}_{rz}^k X_j \cdot \text{curl}_{rz}^k X + \sum_{j=1}^{n+N} c_j \cdot X_j \cdot X \right) r dr dz = \int_{\Omega} (F \cdot X_j) r dr dz \quad (13)
\]
Breaking Down the Problem

We can transform this into a matrix system by setting:

\[ A_{ij} = \int_{\Omega} (\text{curl}_{iz}^k X_j \cdot \text{curl}_{iz}^k X + X_j \cdot X) \, r \, dr \, dz \]

\[ b_i = \int_{\Omega} (F \cdot X_i) \, r \, dr \, dz \]

This gives us the matrix system:

\[ Ac = b \quad (14) \]
FEM on Maxwell’s Equations Results

<table>
<thead>
<tr>
<th>Mesh Level</th>
<th>Weighted $L_2$ Error</th>
<th>Convergence Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.008</td>
<td>0.25</td>
</tr>
<tr>
<td>2</td>
<td>0.007</td>
<td>1.70</td>
</tr>
<tr>
<td>3</td>
<td>0.002</td>
<td>1.90</td>
</tr>
<tr>
<td>4</td>
<td>0.0006</td>
<td>1.99</td>
</tr>
<tr>
<td>5</td>
<td>0.0001</td>
<td>2.00</td>
</tr>
</tbody>
</table>

\[
u(r, z) = \begin{bmatrix} z - \frac{1}{k} \left( \frac{1}{3} r^3 - \frac{1}{2} r^2 \right) \\ -kz + \frac{1}{3} r^3 - \frac{1}{2} r^2 \end{bmatrix}, \quad F(r, z) = \begin{bmatrix} k(r - 1) + u_r(r, z) \\ -2r + 1 + u_\theta(r, z) \end{bmatrix}, \quad (15)\]
Solving For The C-Vector
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.
This was easy to solve – using the Matlab backslash-operator:

$$c = A\backslash b$$
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.
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

\[
\text{while } \text{Stopping-criterion is not fulfilled} \text{ do}
\]
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.

We first implemented the Multiplicative Subspace Correction Method.

**Idea of the Structure**

```
while Stopping-criterion is not fulfilled do
    Process the input
    Return to top
end while
```
The Multiplicative Subspace Correction (MSC) Method
Subspace decomposition

\[ Z_h = \sum_{l=1}^{N} Z_l \]

For example:
\[ Z_1 = \text{span}(\phi_1, \psi_1, \psi_7) \]

Figure: A Vertex Patch
The MSC-Method

\[ U_{i+1} = MSC(u_i, f) \]

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

2) For \( j = 1, 2, \ldots, N \)

\[ u_i^{(j)} = u_i^{(j-1)} + A_j^{-1} Q_j (f - A u_i^{(j-1)}) \]

3) \( u_{i+1} = u_i^{(N)} \)
## Results of MSC-Method

<table>
<thead>
<tr>
<th>Mesh Level</th>
<th>Number of Iterations</th>
<th>Convergence Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>65</td>
<td>0.59</td>
</tr>
<tr>
<td>3</td>
<td>230</td>
<td>0.87</td>
</tr>
<tr>
<td>4</td>
<td>864</td>
<td>0.96</td>
</tr>
<tr>
<td>5</td>
<td>3274</td>
<td>0.99</td>
</tr>
<tr>
<td>6</td>
<td>12466</td>
<td>1</td>
</tr>
</tbody>
</table>
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 $Ac=b$
Multigrid V-Cycle

Uses a sequence of nested meshes to solve $Ac=b$

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

Figure: Mesh Level 1
Figure: Mesh Level 2
Figure: Mesh Level 3
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 ($V(2)$, $f$).

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.
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 \( k = 1 \)

<table>
<thead>
<tr>
<th>Mesh Level</th>
<th>Number of Iterations</th>
<th>( L_2 ) Convergence Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>18</td>
<td>0.14</td>
</tr>
<tr>
<td>3</td>
<td>19</td>
<td>0.17</td>
</tr>
<tr>
<td>4</td>
<td>24</td>
<td>0.26</td>
</tr>
<tr>
<td>5</td>
<td>27</td>
<td>0.30</td>
</tr>
<tr>
<td>6</td>
<td>28</td>
<td>0.32</td>
</tr>
<tr>
<td>7</td>
<td>27</td>
<td>0.32</td>
</tr>
<tr>
<td>8</td>
<td>27</td>
<td>0.32</td>
</tr>
</tbody>
</table>
Iterations to Convergence Results Comparison

![Graph showing iterations to convergence for MSC and Multigrid methods across different mesh levels. The graph plots iterations on a logarithmic scale against mesh level. The MSC method shows a steep increase, while the Multigrid method remains relatively flat.](image-url)
Convergence Rate Results Comparison

![Comparison of Convergence Rates for MSC and Multigrid]

Crupi & Moise & Odom

FEM Multigrid

31st July 2020
## Convergence Rate Comparison Between Various Fourier Modes

<table>
<thead>
<tr>
<th>Mesh Level</th>
<th>$k = 1$</th>
<th>$k = 2$</th>
<th>$k = 3$</th>
<th>$k = 4$</th>
<th>$k = 5$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>0.14</td>
<td>0.11</td>
<td>0.08</td>
<td>0.05</td>
<td>0.06</td>
</tr>
<tr>
<td>3</td>
<td>0.17</td>
<td>0.14</td>
<td>0.12</td>
<td>0.11</td>
<td>0.09</td>
</tr>
<tr>
<td>4</td>
<td>0.26</td>
<td>0.23</td>
<td>0.19</td>
<td>0.16</td>
<td>0.15</td>
</tr>
<tr>
<td>5</td>
<td>0.30</td>
<td>0.30</td>
<td>0.29</td>
<td>0.28</td>
<td>0.27</td>
</tr>
<tr>
<td>6</td>
<td>0.32</td>
<td>0.32</td>
<td>0.31</td>
<td>0.31</td>
<td>0.31</td>
</tr>
<tr>
<td>7</td>
<td>0.32</td>
<td>0.32</td>
<td>0.32</td>
<td>0.32</td>
<td>0.32</td>
</tr>
<tr>
<td>8</td>
<td>0.32</td>
<td>0.32</td>
<td>0.32</td>
<td>0.32</td>
<td>0.32</td>
</tr>
</tbody>
</table>
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.