Multigrid Methods
(Book: Iterative methods for sparse linear systems - Yousef Saad)

Sreevatsa Anantharamu
Motivation from 1D problem - Why multigrid methods?

- Motivation: Divide and conquer the error. How?
- 1D problem: \(-\frac{d^2 u(x)}{dx^2} = f(x) \) \( \text{in} \ x \in (0, 1); \ u(0) = 0; \ u(1) = 0 \)

- Mesh: \( \{x_i\}_{i=0}^{n+1} ; x_i := h * i \ ; \ h = \frac{1}{n+1} \)
- Discretize using Finite difference: \( Au = f \ ; \ A \in \mathbb{R}^{n \times n}; u, f \in \mathbb{R}^n \)

\[
\begin{bmatrix}
2 & -1 & 0 & \ldots & \ldots & \ldots \\
-1 & 2 & -1 & \ldots & \ldots & \ldots \\
0 & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & \ldots & 2 & -1 \\
\ldots & \ldots & \ldots & \ldots & 0 & -1 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
\vdots \\
u_{n-1} \\
u_n \\
\end{bmatrix}
= \begin{bmatrix}
h^2 f(x_1) \\
h^2 f(x_2) \\
\vdots \\
h^2 f(x_{n-1}) \\
h^2 f(x_n) \\
\end{bmatrix}
\]

- Eigenvalues/vectors of A: \( Aw_k = \lambda_k w_k \ ; \ \lambda_k = 4 \sin^2 \left(\frac{\theta_k}{2}\right) \)

\( w_k = \begin{bmatrix}
\sin(\theta_k) & \sin(2\theta_k) & \ldots & \sin(n\theta_k)
\end{bmatrix} ; \ \theta_k := \frac{k\pi}{n+1} ; \ k = 1, \ldots, n \)

\( \lambda_k > 0 \ ; \ \lambda_1 < \lambda_2 < \cdots < \lambda_n \)

- Call \( w_k \) for \( k \leq n/2 \) as low frequency modes and \( w_k \) for \( k > n/2 \) as high frequency modes.
Motivation from 1D problem - Why multigrid methods?

- Motivation: Divide and conquer the error. How?
- 1D problem: \(- \frac{d^2 u(x)}{dx^2} = f(x) \) in \( x \in (0, 1); u(0) = 0; u(1) = 0 \)

\[
\begin{bmatrix}
2 & -1 & 0 & \cdots & \cdots & \cdots \\
-1 & 2 & -1 & \cdots & \cdots & \cdots \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots \\
\cdots & \cdots & \cdots & \cdots & -1 & 2 & -1 \\
\cdots & \cdots & \cdots & \cdots & 0 & -1 & 2 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
\vdots \\
u_{n-1} \\
u_n \\
\end{bmatrix}
= \begin{bmatrix}
h^2 f(x_1) \\
h^2 f(x_2) \\
\vdots \\
h^2 f(x_{n-1}) \\
h^2 f(x_n) \\
\end{bmatrix}
\]

- Mesh: \( \{x_i\}_{i=0}^{n+1} ; x_i := h \times i ; h = \frac{1}{n+1} \)
- Discretize using Finite difference: \( Au = f ; A \in \mathbb{R}^{n \times n} ; u, f \in \mathbb{R}^n \)

- Eigenvalues/vectors of \( A \): \( Aw_k = \lambda_k w_k \); \( \lambda_k = 4 \sin^2 \left( \frac{\theta_k}{2} \right) \)
\[
w_k = \begin{bmatrix}
\sin(\theta_k) & \sin(2\theta_k) & \cdots & \sin(n\theta_k)
\end{bmatrix}; \theta_k := \frac{k\pi}{n+1}; k = 1, \ldots, n
\]

- \( \lambda_k > 0 \); \( \lambda_1 < \lambda_2 < \cdots < \lambda_n \)
- Call \( w_k \) for \( k \leq n/2 \) as low frequency modes and \( w_k \) for \( k > n/2 \) as high frequency modes.
Motivation from 1D problem - Why multigrid methods?

- Motivation: Divide and conquer the error. How?
- 1D problem: \[-\frac{d^2 u(x)}{dx^2} = f(x) \quad \text{in} \quad x \in (0, 1); u(0) = 0; u(1) = 0\]

- Mesh: \(\{x_i\}_{i=0}^{n+1} ; x_i := h \times i ; h = \frac{1}{n+1}\)

- Discretize using Finite difference: \(Au = f ; A \in \mathbb{R}^{n \times n}; u, f \in \mathbb{R}^n\)

\[
\begin{bmatrix}
2 & -1 & 0 & \ldots & \ldots & \ldots \\
-1 & 2 & -1 & \ldots & \ldots & \ldots \\
0 & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & -1 & 2 & -1 \\
\ldots & \ldots & \ldots & 0 & -1 & 2 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
\vdots \\
u_{n-1} \\
u_n
\end{bmatrix}
= \begin{bmatrix}
h^2f(x_1) \\
h^2f(x_2) \\
\vdots \\
h^2f(x_{n-1}) \\
h^2f(x_n)
\end{bmatrix}
\]

- Eigenvalues/vectors of A: \(Aw_k = \lambda_k w_k ; \lambda_k = 4 \sin^2(\frac{\theta_k}{2})\)

\(w_k = [\sin(\theta_k) \sin(2\theta_k) \ldots \sin(n\theta_k)] ; \theta_k := \frac{k\pi}{n+1} ; k = 1, \ldots, n\)

- \(\lambda_k > 0 \quad ; \lambda_1 < \lambda_2 < \cdots < \lambda_n\)
- Call \(w_k\) for \(k \leq n/2\) as low frequency modes and \(w_k\) for \(k > n/2\) as high frequency modes.
Motivation from 1D problem - Why multigrid methods?

- Motivation: Divide and conquer the error. How?
- 1D problem: $-\frac{d^2 u(x)}{dx^2} = f(x)$ in $x \in (0, 1)$; $u(0) = 0$; $u(1) = 0$

- Mesh: $\{x_i\}_{i=0}^{n+1}$; $x_i := h \ast i$; $h = \frac{1}{n+1}$

- Discretize using Finite difference: $Au = f$; $A \in \mathbb{R}^{n \times n}$; $u, f \in \mathbb{R}^n$

$$
\begin{bmatrix}
2 & -1 & 0 & \ldots & \ldots & \ldots \\
-1 & 2 & -1 & \ldots & \ldots & \ldots \\
0 & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & -1 & 2 & -1 \\
\ldots & \ldots & \ldots & 0 & -1 & 2
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
\vdots \\
u_{n-1} \\
u_n
\end{bmatrix}
= 
\begin{bmatrix}
h^2 f(x_1) \\
h^2 f(x_2) \\
\vdots \\
h^2 f(x_{n-1}) \\
h^2 f(x_n)
\end{bmatrix}
$$

- Eigenvalues/vectors of $A$: $Aw_k = \lambda_k w_k$; $\lambda_k = 4\sin^2\left(\frac{\theta_k}{2}\right)$
$$
w_k = \begin{bmatrix}
\sin(\theta_k) & \sin(2\theta_k) & \ldots & \sin(n\theta_k)
\end{bmatrix}; \theta_k := \frac{k\pi}{n+1}; k = 1, \ldots, n
$$

- $\lambda_k > 0$; $\lambda_1 < \lambda_2 < \cdots < \lambda_n$

- Call $w_k$ for $k \leq n/2$ as low frequency modes and $w_k$ for $k > n/2$ as high frequency modes.
Motivation from 1D problem - Why multigrid methods?

- Motivation: Divide and conquer the error. How?
- 1D problem: \(-\frac{d^2u(x)}{dx^2} = f(x)\) \(\text{in} \ x \in (0, 1); u(0) = 0; u(1) = 0\)

- Mesh: \(\{x_i\}_{i=0}^{n+1}; x_i := h \times i; h = \frac{1}{n+1}\)
- Discretize using Finite difference: \(Au = f; A \in \mathbb{R}^{n \times n}; u, f \in \mathbb{R}^n\)

\[
\begin{bmatrix}
2 & -1 & 0 & \cdots & \cdots & \cdots \\
-1 & 2 & -1 & \cdots & \cdots & \cdots \\
0 & \cdots & \cdots & \cdots & \cdots & \cdots \\
\cdots & \cdots & \cdots & -1 & 2 & -1 \\
\cdots & \cdots & \cdots & 0 & -1 & 2 \\
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
\vdots \\
u_{n-1} \\
u_n \\
\end{bmatrix}
=
\begin{bmatrix}
h^2f(x_1) \\
h^2f(x_2) \\
\vdots \\
h^2f(x_{n-1}) \\
h^2f(x_n) \\
\end{bmatrix}
\]

- Eigenvalues/vectors of A: \(Aw_k = \lambda_k w_k; \lambda_k = 4\sin^2\left(\frac{\theta_k}{2}\right)\)

\(w_k = [\sin(\theta_k) \ \sin(2\theta_k) \ \cdots \ \sin(n\theta_k)]; \theta_k := \frac{k\pi}{n+1}; k = 1, \ldots, n\)

\(\lambda_k > 0 ; \lambda_1 < \lambda_2 < \cdots < \lambda_n\)

\(\text{Call } w_k \text{ for } k \leq n/2 \text{ as low frequency modes and } w_k \text{ for } k > n/2 \text{ as high frequency modes.}\)
Motivation from 1D problem - Why multigrid methods?

- Motivation: Divide and conquer the error. How?
- 1D problem: \(- \frac{d^2 u(x)}{dx^2} = f(x) \) in \( x \in (0, 1); u(0) = 0; u(1) = 0 \)

Mesh: \( \{x_i\}_{i=0}^{n+1}; x_i := h \times i; h = \frac{1}{n+1} \)
- Discretize using Finite difference: \( Au = f; A \in \mathbb{R}^{n \times n}; u, f \in \mathbb{R}^n \)

\[
\begin{bmatrix}
2 & -1 & 0 & \ldots & \ldots & \ldots \\
-1 & 2 & -1 & \ldots & \ldots & \ldots \\
0 & \ldots & \ldots & \ldots & \ldots & \ldots \\
\ldots & \ldots & \ldots & -1 & 2 & -1 \\
\ldots & \ldots & \ldots & 0 & -1 & 2
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
\vdots \\
u_{n-1} \\
u_n
\end{bmatrix}
= 
\begin{bmatrix}
h^2 f(x_1) \\
h^2 f(x_2) \\
\vdots \\
h^2 f(x_{n-1}) \\
h^2 f(x_n)
\end{bmatrix}
\]
- Eigenvalues/vectors of A: \( Aw_k = \lambda_k w_k; \lambda_k = 4\sin^2 \left( \frac{\theta_k}{2} \right) \)

\[
w_k = \begin{bmatrix}
\sin(\theta_k) & \sin(2\theta_k) & \ldots & \sin(n\theta_k)
\end{bmatrix}; \theta_k := \frac{k \pi}{n+1}; k = 1, \ldots, n
\]

\( \lambda_k > 0 \); \( \lambda_1 < \lambda_2 < \cdots < \lambda_n \)
- Call \( w_k \) for \( k \leq n/2 \) as low frequency modes and \( w_k \) for \( k > n/2 \) as high frequency modes.
Motivation from 1D problem - Why multigrid methods?

- Motivation: Divide and conquer the error. How?
- 1D problem: 
  \[- \frac{d^2 u(x)}{dx^2} = f(x) \text{ in } x \in (0, 1); u(0) = 0; u(1) = 0\]

- Mesh: 
  \[\{x_i\}_{i=0}^{n+1}; x_i := h \times i; h = \frac{1}{n+1}\]

- Discretize using Finite difference: 
  \[Au = f; A \in \mathbb{R}^{n \times n}; u, f \in \mathbb{R}^n\]

\[
\begin{bmatrix}
  2 & -1 & 0 & \ldots & \ldots & \ldots \\
  -1 & 2 & -1 & \ldots & \ldots & \ldots \\
  0 & \ldots & \ldots & \ldots & \ldots & \ldots \\
  \ldots & \ldots & \ldots & -1 & 2 & -1 \\
  \ldots & \ldots & \ldots & 0 & -1 & 2
\end{bmatrix}
\begin{bmatrix}
  u_1 \\
  u_2 \\
  \vdots \\
  u_{n-1} \\
  u_n
\end{bmatrix}
= \begin{bmatrix}
  h^2 f(x_1) \\
  h^2 f(x_2) \\
  \vdots \\
  h^2 f(x_{n-1}) \\
  h^2 f(x_n)
\end{bmatrix}
\]

- Eigenvalues/vectors of A: 
  \[Aw_k = \lambda_k w_k; \lambda_k = 4\sin^2\left(\frac{\theta_k}{2}\right)\]
  \[w_k = \begin{bmatrix}
  \sin(\theta_k) & \sin(2\theta_k) & \ldots & \sin(n\theta_k)
\end{bmatrix}; \theta_k := \frac{k\pi}{n+1}; k = 1, \ldots, n\]

  ▶ \(\lambda_k > 0\); \(\lambda_1 < \lambda_2 < \cdots < \lambda_n\)

  ▶ Call \(w_k\) for \(k \leq n/2\) as low frequency modes and \(w_k\) for \(k > n/2\) as high frequency modes.
Motivation from 1D problem - Why multigrid methods?

- Motivation: Divide and conquer the error. How?
- 1D problem: \[- \frac{d^2 u(x)}{dx^2} = f(x) \quad \text{in} \quad x \in (0, 1); \quad u(0) = 0; \quad u(1) = 0\]

- Mesh: \(\{x_i\}_{i=0}^{n+1} \quad ; \quad x_i := h \times i \quad ; \quad h = \frac{1}{n+1}\)

- Discretize using Finite difference: \(Au = f \quad ; \quad A \in \mathbb{R}^{n \times n}; \quad u, f \in \mathbb{R}^n\)

\[
\begin{bmatrix}
2 & -1 & 0 & \ldots & \ldots & \ldots & \ldots \\
-1 & 2 & -1 & \ldots & \ldots & \ldots & \ldots \\
0 & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\
\ldots & \ldots & \ldots & -1 & 2 & -1 \\
\ldots & \ldots & \ldots & 0 & -1 & 2
\end{bmatrix}
\begin{bmatrix}
u_1 \\
u_2 \\
\vdots \\
u_{n-1} \\
u_n
\end{bmatrix} =
\begin{bmatrix}
h^2f(x_1) \\
h^2f(x_2) \\
\vdots \\
h^2f(x_{n-1}) \\
h^2f(x_n)
\end{bmatrix}
\]

- Eigenvalues/vectors of A: \(Aw_k = \lambda_k w_k \quad ; \quad \lambda_k = 4\sin^2\left(\frac{\theta_k}{2}\right)\)

\(w_k = [\sin(\theta_k) \quad \sin(2\theta_k) \quad \ldots \quad \sin(n\theta_k)] \quad ; \quad \theta_k := \frac{k\pi}{n+1}; \quad k = 1, \ldots, n\)

- \(\lambda_k > 0 \quad ; \quad \lambda_1 < \lambda_2 < \cdots < \lambda_n\)
- Call \(w_k\) for \(k \leq n/2\) as low frequency modes and \(w_k\) for \(k > n/2\) as high frequency modes.
Motivation form 1D problem - Why multigrid methods?

Figure: $w_k \; k=1$

Figure: $w_k \; k=2$

Figure: $w_k \; k=3$

Figure: $w_k \; k=5$

Figure: $w_k \; k=6$

Figure: $w_k \; k=7$
Motivation from 1D problem - Why multigrid methods?

- All relaxation methods to solve $Au = f$ can be written as $u_{j+1} = Gu_j + b$.
- For simplicity, choose $G$ of Richardson iteration $G_r := I - \alpha A$ and $b_r := \alpha f$. We have $u_{j+1} = G_r u_j + b_r$
- Can be shown that Richardson iteration converges for $0 < \alpha < 2/\rho(A)$ (How?).
- For our 1D problem $\rho(A) < 4$ (How?). Choose $\alpha = 1/4$.
- Eigenvalues of $G_r$, $\eta_k = \cos^2(\theta_k/2)$ (How?). Recall $\theta_k := \frac{k\pi}{n+1}$. Eigenvectors of $G_r$ same as $A$ i.e. $w_k$ (How?).

- Error in iteration $j$ is $e_j := u - u_j$ and $e_j = G^j e_o$.
- Expanding initial error $e_o$ in the eigenbasis of $G$ (same as $A$), $e_o = \sum_{k=1}^n c_k w_k$. Then, we have $e_j = \sum_{k=1}^n \eta_k^j c_k w_k$. 
Motivation from 1D problem - Why multigrid methods?

- All relaxation methods to solve \( Au = f \) can be written as \( u_{j+1} = Gu_j + b \).
- For simplicity, choose \( G \) of Richardson iteration \( G_r := I - \alpha A \) and \( b_r := \alpha f \). We have \( u_{j+1} = G_r u_j + b_r \).
- Can be shown that Richardson iteration converges for \( 0 < \alpha < 2/\rho(A) \) (How?).
- For our 1D problem \( \rho(A) < 4 \) (How?). Choose \( \alpha = 1/4 \).
- Eigenvalues of \( G_r, \eta_k = \cos^2(\theta_k/2) \) (How?). Recall \( \theta_k := \frac{k\pi}{n+1} \).
  Eigenvectors of \( G_r \) same as \( A \) i.e. \( w_k \) (How?).

- Error in iteration \( j \) is \( e_j := u - u_j \) and \( e_j = G^j e_o \).
- Expanding initial error \( e_o \) in the eigenbasis of \( G \) (same as \( A \)), \( e_o = \sum_{k=1}^n c_k w_k \). Then, we have \( e_j = \sum_{k=1}^n \eta_k^j c_k w_k \).
Motivation from 1D problem - Why multigrid methods?

- All relaxation methods to solve \( Au = f \) can be written as \( u_{j+1} = Gu_j + b \).
- For simplicity, choose \( G \) of Richardson iteration \( G_r := I - \alpha A \) and \( b_r := \alpha f \). We have \( u_{j+1} = G_r u_j + b_r \).
- Can be shown that Richardson iteration converges for \( 0 < \alpha < 2/\rho(A) \) (How?).
- For our 1D problem \( \rho(A) < 4 \) (How?). Choose \( \alpha = 1/4 \).
- Eigenvalues of \( G_r, \eta_k = \cos^2(\theta_k/2) \) (How?). Recall \( \theta_k := \frac{k\pi}{n+1} \). Eigenvectors of \( G_r \) same as \( A \) i.e. \( w_k \) (How?).

- Error in iteration \( j \) is \( e_j := u - u_j \) and \( e_j = G^j e_o \).
- Expanding initial error \( e_o \) in the eigenbasis of \( G \) (same as \( A \)), \( e_o = \sum_{k=1}^{n} c_k w_k \). Then, we have \( e_j = \sum_{k=1}^{n} \eta_k^j c_k w_k \).
Motivation from 1D problem - Why multigrid methods?

- All relaxation methods to solve \( Au = f \) can be written as
  \[ u_{j+1} = Gu_j + b. \]
- For simplicity, choose \( G \) of Richardson iteration \( G_r := I - \alpha A \) and \( b_r := \alpha f \). We have \( u_{j+1} = G_r u_j + b_r \)
- Can be shown that Richardson iteration converges for \( 0 < \alpha < 2/\rho(A) \) (How?).
- For our 1D problem \( \rho(A) < 4 \) (How?). Choose \( \alpha = 1/4 \).
- Eigenvalues of \( G_r, \eta_k = \cos^2(\theta_k/2) \) (How?). Recall \( \theta_k := \frac{k\pi}{n+1} \).
  Eigenvectors of \( G_r \) same as \( A \) i.e. \( w_k \) (How?).

- Error in iteration \( j \) is \( e_j := u - u_j \) and \( e_j = G^j e_o \).
- Expanding initial error \( e_o \) in the eigenbasis of \( G \) (same as \( A \)),
  \( e_o = \sum_{k=1}^n c_k w_k \). Then, we have \( e_j = \sum_{k=1}^n \eta_k^j c_k w_k \).
Motivation from 1D problem - Why multigrid methods?

- All relaxation methods to solve $Au = f$ can be written as $u_{j+1} = Gu_j + b$.
- For simplicity, choose $G$ of Richardson iteration $G_r := I - \alpha A$ and $b_r := \alpha f$. We have $u_{j+1} = G_r u_j + b_r$
- Can be shown that Richardson iteration converges for $0 < \alpha < 2/\rho(A)$ (How?).
- For our 1D problem $\rho(A) < 4$ (How?). Choose $\alpha = 1/4$.
- Eigenvalues of $G_r, \eta_k = \cos^2(\theta_k/2)$ (How?). Recall $\theta_k := \frac{k\pi}{n+1}$.
  Eigenvectors of $G_r$ same as $A$ i.e. $w_k$ (How?).

- Error in iteration $j$ is $e_j := u - u_j$ and $e_j = G^j e_o$.
- Expanding initial error $e_o$ in the eigenbasis of $G$ (same as $A$), $e_o = \sum_{k=1}^{n} c_k w_k$. Then, we have $e_j = \sum_{k=1}^{n} \eta_k^j c_k w_k$. 
Motivation from 1D problem - Why multigrid methods?

- All relaxation methods to solve $Au = f$ can be written as $u_{j+1} = Gu_j + b$.
- For simplicity, choose $G$ of Richardson iteration $G_r := I - \alpha A$ and $b_r := \alpha f$. We have $u_{j+1} = G_r u_j + b_r$.
- Can be shown that Richardson iteration converges for $0 < \alpha < 2/\rho(A)$ (How?).
- For our 1D problem $\rho(A) < 4$ (How?). Choose $\alpha = 1/4$.
- Eigenvalues of $G_r$, $\eta_k = \cos^2(\theta_k/2)$ (How?). Recall $\theta_k := \frac{k\pi}{n+1}$. Eigenvectors of $G_r$ same as $A$ i.e. $w_k$ (How?).

- Error in iteration $j$ is $e_j := u - u_j$ and $e_j = G^j e_o$.
- Expanding initial error $e_o$ in the eigenbasis of $G$ (same as $A$), $e_o = \sum_{k=1}^n c_k w_k$. Then, we have $e_j = \sum_{k=1}^n \eta_k^j c_k w_k$. 
Motivation from 1D problem - Why multigrid methods?

- All relaxation methods to solve $Au = f$ can be written as $u_{j+1} = Gu_j + b$.
- For simplicity, choose $G$ of Richardson iteration $G_r := I - \alpha A$ and $b_r := \alpha f$. We have $u_{j+1} = G_r u_j + b_r$
- Can be shown that Richardson iteration converges for $0 < \alpha < 2/\rho(A)$ (How?).
- For our 1D problem $\rho(A) < 4$ (How?). Choose $\alpha = 1/4$.
- Eigenvalues of $G_r, \eta_k = \cos^2(\theta_k/2)$ (How?). Recall $\theta_k := \frac{k\pi}{n+1}$. Eigenvectors of $G_r$ same as $A$ i.e. $w_k$ (How?).

- Error in iteration $j$ is $e_j := u - u_j$ and $e_j = G^j e_o$.
- Expanding initial error $e_o$ in the eigenbasis of $G$ (same as $A$), $e_o = \sum_{k=1}^{n} c_k w_k$. Then, we have $e_j = \sum_{k=1}^{n} \eta_k^j c_k w_k$. 
Motivation from 1D problem - Why multigrid methods?

\[ e_j = \sum_{k=1}^{n} \eta_k^j c_k w_k. \eta_k \text{ is also called reduction coefficient (Why?).} \]

If \( \eta_k \ll 1 \) then, error along mode \( k \) is reduced rapidly. \( (\eta_k \text{ is eigenvalue of } G') \)

Error along:
- Low frequency modes \((k \leq n/2)\) Eg: \( k = 1, \eta_1 = \cos^2(\theta_1/2) = 1 - O(h^2) \). Depends on \( h \)!! and increases with decrease in \( h \)!!
- High frequency modes: \((k > n/2) \) \( \eta_k \geq 1/2 \). Independent of \( h \)!

Observation: Error along high frequency modes are reduced more rapidly than those along low frequency modes. Also, the reduction of error along high frequency modes is independent of \( h \)!!.
Motivation from 1D problem - Why multigrid methods?

\[ e_j = \sum_{k=1}^{n} \eta_k^j c_k w_k. \eta_k \text{ is also called reduction coefficient (Why?).} \]

If \( \eta_k \ll 1 \) then, error along mode \( k \) is reduced rapidly. (\( \eta_k \) is eigenvalue of \( G' \))

Error along:
- **Low frequency modes** \((k \leq n/2)\) Eg: \( k = 1, \eta_1 = \cos^2(\theta_1/2) = 1 - O(h^2)\). Depends on \( h!! \) and increases with decrease in \( h!! \)
- **High frequency modes**: \((k > n/2)\) \( \eta_k \geq 1/2 \). Independent of \( h!! \)

Observation: Error along **high frequency modes** are reduced more rapidly than those along **low frequency modes**. Also, the reduction of error along **high frequency modes** is independent of \( h!! \).

Figure 13.2: Reduction coefficients for Richardson’s method applied to the 1-D model problem
Motivation from 1D problem - Why multigrid methods?

\[ e_j = \sum_{k=1}^{n} \eta_k^j c_k w_k. \]
\( \eta_k \) is also called reduction coefficient (Why?).

If \( \eta_k \ll 1 \) then, error along mode \( k \) is reduced rapidly. 
\( (\eta_k \) is eigenvalue of \( G' \))

Error along:
- **Low frequency modes** \((k \leq n/2)\) Eg: \( k = 1, \eta_1 = \cos^2(\theta_1/2) \)
  \( = 1 - O(h^2) \). Depends on \( h \)!! and increases with decrease in \( h \)!!.
- **High frequency modes**: \((k > n/2)\) \( \eta_k \geq 1/2 \). Independent of \( h \)!!.

Observation: Error along high frequency modes are reduced more rapidly than those along low frequency modes. Also, the reduction of error along high frequency modes is independent of \( h \)!!.

Figure 13.2: Reduction coefficients for Richardson’s method applied to the 1-D model problem
Motivation from 1D problem - Why multigrid methods?

\[ e_j = \sum_{k=1}^{n} \eta_k^j c_k w_k. \]

\( \eta_k \) is also called reduction coefficient (Why?).

If \( \eta_k \ll 1 \) then, error along mode \( k \) is reduced rapidly. (\( \eta_k \) is eigenvalue of \( G' \)).

Error along:

- **Low frequency modes** \( (k \leq n/2) \) Eg : \( k = 1, \eta_1 = \cos^2(\theta_1/2) = 1 - O(h^2) \). Depends on \( h \)!! and increases with decrease in \( h \)!!
- **High frequency modes** : \( (k > n/2) \) \( \eta_k \geq 1/2 \). Independent of \( h \)!!

Observation: Error along **high frequency modes** are reduced more rapidly than those along **low frequency modes**. Also, the reduction of error along **high frequency modes** is independent of \( h \)!!.
Motivation from 1D problem - Why multigrid methods?

Idea!!

- Some *low frequency modes* become *high frequency modes* in a coarse mesh. The error along *low frequency modes* can be reduced better on a coarse grid, than the current grid.
- Subscript and superscript $h$ and $2h$ indicates quantities on the current mesh and coarse mesh respectively.
- Current problem $A_h u^h = f^h$. Coarse grid problem $A_{2h} u^{2h} = f^{2h}$

For the fine grid, mode $k = 2$ is $< 7/2$. Hence, it is in the low frequency regime in fine grid i.e. $\eta_2 > 1/2$. For the coarse grid, mode $k = 2$ is $> 3/2$. Hence, the same mode is in the high frequency regime in coarse grid. i.e. $\eta_2 < 1/2$. 
Motivation from 1D problem - Why multigrid methods?

Idea!!
- Some *low frequency modes* become *high frequency modes* in a coarse mesh. The error along *low frequency modes* can be reduced better on a coarse grid, than the current grid.
- Subscript and superscript $h$ and $2h$ indicates quantities on the current mesh and coarse mesh respectively.
- Current problem $A_h u^h = f^h$. Coarse grid problem $A_{2h} u^{2h} = f^{2h}$

For the fine grid, mode $k = 2$ is $< 7/2$. Hence, it is in the low frequency regime in fine grid i.e. $\eta_2 > 1/2$. For the coarse grid, mode $k = 2$ is $> 3/2$. Hence, the same mode is in the high frequency regime in coarse grid. i.e. $\eta_2 < 1/2$. 
Motivation from 1D problem - Why multigrid methods?

Idea!!
- Some low frequency modes become high frequency modes in a coarse mesh. The error along low frequency modes can be reduced better on a coarse grid, than the current grid.
- Subscript and superscript $h$ and $2h$ indicates quantities on the current mesh and coarse mesh respectively.
- Current problem $A_h u^h = f^h$. Coarse grid problem $A_{2h} u^{2h} = f^{2h}$

For the fine grid, mode $k = 2$ is $< 7/2$. Hence, it is in the low frequency regime in fine grid i.e. $\eta_2 > 1/2$. For the coarse grid, mode $k = 2$ is $> 3/2$. Hence, the same mode is in the high frequency regime in coarse grid. i.e. $\eta_2 < 1/2$. 
Motivation from 1D problem - Why multigrid methods?

Idea!!
- Some **low frequency modes** become **high frequency modes** in a coarse mesh. The error along **low frequency modes** can be reduced better on a coarse grid, than the current grid.
- Subscript and superscript $h$ and $2h$ indicates quantities on the current mesh and coarse mesh respectively.
- Current problem $A_h u^h = f^h$. Coarse grid problem $A_{2h} u^{2h} = f^{2h}$

For the fine grid, mode $k = 2$ is $< 7/2$. Hence, it is in the low frequency regime in fine grid i.e. $\eta_2 > 1/2$. For the coarse grid, mode $k = 2$ is $> 3/2$. Hence, the same mode is in the high frequency regime in coarse grid. i.e. $\eta_2 < 1/2$. 
Motivation from 1D problem - Why multigrid methods?

Idea!!
- Some \textit{low frequency modes} become \textit{high frequency modes} in a coarse mesh. The error along \textit{low frequency modes} can be reduced better on a coarse grid, than the current grid.
- Subscript and superscript $h$ and $2h$ indicates quantities on the current mesh and coarse mesh respectively.
- Current problem $A_h u^h = f^h$. Coarse grid problem $A_{2h} u^{2h} = f^{2h}$

For the fine grid, mode $k = 2$ is $< 7/2$. Hence, it is in the low frequency regime in fine grid i.e. $\eta_2 > 1/2$. For the coarse grid, mode $k = 2$ is $> 3/2$. Hence, the same mode is in the high frequency regime in coarse grid. i.e. $\eta_2 < 1/2$. 
Ingredients of multigrid method

- **Concept of Smoothing**: A fixed number of relaxation iterations to reduce the components of error in the current mesh is termed smoothing. These iterations majorly remove *high frequency* \((k > n/2)\) error components.

- Smoothed \(u^h_0\) is denoted as \(u^h_\nu := smooth^\nu(A^h, u^h_0, f^h)\)

1: \(u^h_\nu = u^h_0\)
2: **for** iter = 1 to \(\nu\) **do**
3: \(u^h_\nu = Gu^h_\nu + b\)
4: **end for**

**Algorithm**: Pseudocode for \(u^h_\nu := smooth^\nu(A^h, u^h_0, f^h)\)

- Let’s create a sample multigrid method with our ideas to see the required ingredients with just two levels \(h\) and \(2h\).
Ingredients of multigrid method

- **Concept of Smoothing**: A fixed number of relaxation iterations to reduce the components of error in the current mesh is termed smoothing. These iterations majorly remove *high frequency* ($k > n/2$) error components.

- Smoothed $u^h_o$ is denoted as $u^h_v := \text{smooth}^v(A,h, u^h_o, f^h)$

  1. $u^h_v = u^h_o$
  2. **for** iter = 1 to $\nu$ **do**
  3. $u^h_v = G u^h_v + b$
  4. **end for**

**Algorithm**: Pseudocode for $u^h_v := \text{smooth}^v(A,h, u^h_o, f^h)$

- Let’s create a sample multigrid method with our ideas to see the required ingredients with just two levels $h$ and $2h$. 
Ingredients of multigrid method

- **Concept of Smoothing**: A fixed number of relaxation iterations to reduce the components of error in the current mesh is termed smoothing. These iterations majorly remove high frequency \((k > n/2)\) error components.

- Smoothed \(u^h_o\) is denoted as \(u^h_\nu := smooth^\nu(A_h, u^h_o, f^h)\)

  1. \(u^h_\nu = u^h_o\)
  2. **for** iter = 1 to \(\nu\) **do**
  3. \(u^h_\nu = Gu^h_\nu + b\)
  4. **end for**

**Algorithm**: Pseudocode for \(u^h_\nu := smooth^\nu(A_h, u^h_o, f^h)\)

- Let’s create a sample multigrid method with our ideas to see the required ingredients with just two levels \(h\) and \(2h\).
Ingredients of multigrid method

What are the ingredients of multigrid methods?

1. $u^{2h} = u^{2h}_0$
2. **for** iter = 1 to max_iter **do**
3. $u^{2h} := smooth^{ν1}(A_{2h}, u^{2h}, f^{2h})$
4. Create $u^h$ from $u^{2h}$
5. $u^h := smooth^{ν2}(A_h, u^h, f^h)$
6. Create $u^{2h}$ from $u^h$
7. **end for**

**Algorithm:** Pseudocode for a sample multigrid method with 2-levels

Let sub/superscript $h$ and $H$ denote fine and coarse grid quantities.

- Smoother - Common to use Gauss seidel (GS), red black GS.
- Prolongator - $P^h_H : u^H \mapsto u^h$
- Restrictor - $R^H_h : u^h \mapsto u^H$
- Creating coarse grid problems $A_H$ from fine grid problems $A_h$
Ingredients of multigrid method

What are the ingredients of multigrid methods?

1. \( u^{2h} = u^o \)
2. \textbf{for} iter = 1 to max_iter \textbf{do}
3. \( u^{2h} := \text{smooth}^{\nu_1}(A_{2h}, u^{2h}, f^{2h}) \)
4. Create \( u^h \) from \( u^{2h} \)
5. \( u^h := \text{smooth}^{\nu_2}(A_h, u^h, f^h) \)
6. Create \( u^{2h} \) from \( u^h \)
7. \textbf{end for}

\textbf{Algorithm:} Pseudocode for a sample multigrid method with 2-levels

Let sub/superscript \( h \) and \( H \) denote fine and coarse grid quantities.

- **Smother** - Common to use Gauss seidel (GS), red black GS.
- **Prolongator** - \( P_h^H : u^H \mapsto u^h \)
- **Restrictor** - \( R_h^H : u^h \mapsto u^H \)
- Creating coarse grid problems \( A_H \) from fine grid problems \( A_h \)
Ingredients of multigrid method

What are the ingredients of multigrid methods?

1. $u^{2h} = u^{2h}_o$
2. for $iter = 1$ to $max_iter$ do
3. $u^{2h} := \text{smooth}^{\nu_1}(A_{2h}, u^{2h}, f^{2h})$
4. Create $u^h$ from $u^{2h}$
5. $u^h := \text{smooth}^{\nu_2}(A_h, u^h, f^h)$
6. Create $u^{2h}$ from $u^h$
7. end for

Algorithm: Pseudocode for a sample multigrid method with 2-levels

Let sub/superscript $h$ and $H$ denote fine and coarse grid quantities.

- **Smoother** - Common to use Gauss seidel (GS), red black GS.
- **Prolongator** - $P^h_H : u^H \mapsto u^h$
- **Restrictor** - $R^H_h : u^h \mapsto u^H$
- Creating coarse grid problems $A_H$ from fine grid problems $A_h$
Ingredients of multigrid method

What are the ingredients of multigrid methods?

1: $u^{2h} = u^{2h}_o$

2: \textbf{for} iter = 1 to max_iter \textbf{do}

3: $u^{2h} := \text{smooth}^{\nu_1}(A_{2h}, u^{2h}, f^{2h})$

4: \text{Create } u^h \text{ from } u^{2h}$

5: $u^h := \text{smooth}^{\nu_2}(A_h, u^h, f^h)$

6: \text{Create } u^{2h} \text{ from } u^h$

7: \textbf{end for}

\textbf{Algorithm:} Pseudocode for a sample multigrid method with 2-levels

Let sub/superscript $h$ and $H$ denote fine and coarse grid quantities.

- **Smoother** - Common to use Gauss seidel (GS), red black GS.
- **Prolongator** - $P^h_H : u^H \mapsto u^h$
- **Restrictor** - $R^H_h : u^h \mapsto u^H$
- Creating coarse grid problems $A_H$ from fine grid problems $A_h$
Ingredients of multigrid method

What are the ingredients of multigrid methods?

1. $u^{2h} = u_0^{2h}$
2. **for** iter = 1 to max_iter **do**
   3. $u^{2h} := \text{smooth}^{\nu_1}(A_{2h}, u^{2h}, f^{2h})$
   4. Create $u^h$ from $u^{2h}$
   5. $u^h := \text{smooth}^{\nu_2}(A_h, u^h, f^h)$
   6. Create $u^{2h}$ from $u^h$
3. **end for**

**Algorithm:** Pseudocode for a sample multigrid method with 2-levels 
Let sub/superscript $h$ and $H$ denote fine and coarse grid quantities.

- **Smotherer** - Common to use Gauss seidel (GS), red black GS.
- **Prolongator** - $P^h_H : u^H \mapsto u^h$
- **Restrictor** - $R^H_h : u^h \mapsto u^H$
- Creating coarse grid problems $A_H$ from fine grid problems $A_h$
Prolongator $P_H^h : u^H \mapsto u^h$

$$v^h = P_H^h v^H$$

$P_H^h$ takes vector from coarse mesh to the corresponding vector in fine mesh.

Eg : $P_H^h$ in 1D. Take H=2h

$$v_{2j}^h = v_j^{2h}$$
$$v_{2j+1}^h = \frac{1}{2}(v_j^{2h} + v_{j+1}^{2h})$$

$$\begin{bmatrix}
v_1^h \\ v_2^h \\ \vdots \\ v_{n-1}^h \\ v_n^h
\end{bmatrix} = 1/2\begin{bmatrix}
1 & 0 & \ldots & 0 & 0 \\
2 & 0 & \ldots & 0 & 0 \\
1 & 1 & \ldots & 0 & 0 \\
0 & 2 & \ldots & 0 & 0 \\
0 & 0 & \ldots & 0 & 1
\end{bmatrix}\begin{bmatrix}
v_1^{2h} \\ v_2^{2h} \\ \vdots \\ v_{(n-1)/2}^{2h} \\ v_{(n-1)/2+1}^{2h}
\end{bmatrix}$$
Prolongator $P_H^h : u^H \mapsto u^h$

$v^h = P_H^h v^H$

$P_H^h$ takes vector from coarse mesh to the corresponding vector in fine mesh.

Eg : $P_H^h$ in 1D. Take $H=2h$

\[ v_{2j}^h = v_j^{2h} \]
\[ v_{2j+1}^h = \frac{1}{2}(v_j^{2h} + v_{j+1}^{2h}) \]

\[
\begin{bmatrix}
    v_1^h \\
    v_2^h \\
    \vdots \\
    v_{n-1}^h \\
    v_n^h \\
\end{bmatrix} = 1/2 \\
\begin{bmatrix}
    1 & 0 & \ldots & 0 & 0 \\
    2 & 0 & \ldots & 0 & 0 \\
    1 & 1 & \ldots & 0 & 0 \\
    0 & 2 & \ldots & 0 & 0 \\
    \vdots \\
    0 & 0 & \ldots & 0 & 1 \\
\end{bmatrix} \\
\begin{bmatrix}
    v_1^{2h} \\
    v_2^{2h} \\
    \vdots \\
    v_{(n-1)/2}^{2h} \\
    v_{(n-1)/2}^{2h} \\
\end{bmatrix}
\]
Prolongator $P^h_H : u^H \mapsto u^h$

$v^h = P^h_H v^H$

$P^h_H$ takes vector from coarse mesh to the corresponding vector in fine mesh.

Eg : $P^h_H$ in 1D. Take $H=2h$

\[ v^h_{2j} = v^h_j \]
\[ v^h_{2j+1} = \frac{1}{2}(v^h_j + v^h_{j+1}) \]

\[
\begin{bmatrix}
  v^h_1 \\
  v^h_2 \\
  \vdots \\
  v^h_{n-1} \\
  v^h_n
\end{bmatrix}
= \frac{1}{2}
\begin{bmatrix}
  1 & 0 & \ldots & 0 & 0 \\
  2 & 0 & \ldots & 0 & 0 \\
  1 & 1 & \ldots & 0 & 0 \\
  0 & 2 & \ldots & 0 & 0 \\
  0 & 0 & \ldots & 0 & 1
\end{bmatrix}
\begin{bmatrix}
  v^h_1 \\
  v^h_2 \\
  \vdots \\
  v^h_{(n-1)/2-1} \\
  v^h_{(n-1)/2}
\end{bmatrix}
\]
Restrictor $R^H_h : u^h \mapsto u^H$

$$v^H = R^H_h v^h$$

$R^H_h$ takes vector from fine mesh to corresponding vector in coarse mesh.

Eg: $R^H_h$ in 1D. Take $H=2h$

$$v_{2j}^{2h} = \frac{1}{4}(v_{2j-1}^h + 2v_{2j}^h + v_{2j+1}^h)$$

$$\begin{bmatrix} v_1^{2h} \\ v_2^{2h} \\ \vdots \\ v_{(n-1)/2}^{2h} \\ v_{(n-1)/2}^{2h} \end{bmatrix} = \frac{1}{4} \begin{bmatrix} 1 & 2 & 1 & \ldots & 0 & 0 & 0 \\ 0 & 0 & \ddots & \ddots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \ddots & \ddots & \ddots & \ddots \\ 0 & 0 & 0 & \ldots & 1 & 2 & 1 \end{bmatrix} \begin{bmatrix} v_1^h \\ v_2^h \\ \vdots \\ v_{n-1}^h \\ v_n^h \end{bmatrix}$$

It can be shown that $R^H_h = 1/2P^H_h^T$. 
Restrictor $R^H_h : u^h \mapsto u^H$

$v^H = R^H_h v^h$

$R^H_h$ takes vector from fine mesh to corresponding vector in coarse mesh.

Eg: $R^H_h$ in 1D. Take $H=2h$

$$v^{2h}_j = \frac{1}{4}(v^{h}_{2j-1} + 2v^{h}_{2j} + v^{h}_{2j-1})$$

$$\begin{bmatrix}
    v^{2h}_1 \\
    v^{2h}_2 \\
    \vdots \\
    v^{2h}_{(n-1)/2-1} \\
    v^{2h}_{(n-1)/2}
\end{bmatrix} \cdot \begin{bmatrix}
    1 & 2 & 1 & \ldots & 0 & 0 & 0 \\
    0 & 0 & \ddots & \ddots & 0 & 0 \\
    0 & 0 & 0 & \ldots & 1 & 2 & 1
\end{bmatrix} \cdot \begin{bmatrix}
    v^{h}_1 \\
    v^{h}_2 \\
    \vdots \\
    v^{h}_{n-1} \\
    v^{h}_n
\end{bmatrix}$$

It can be shown that $R^H_h = 1/2P^h_H^T$. 
Restrictor $R_h^H : u^h \mapsto u^H$

\[ \nu^H = R_h^H \nu^h \]

$R_h^H$ takes vector from fine mesh to corresponding vector in coarse mesh.

Eg: $R_h^H$ in 1D. Take $H=2h$

\[ \nu_{2h}^j = \frac{1}{4}(\nu_{2j-1}^h + 2\nu_{2j}^h + \nu_{2j-1}^h) \]

\[
\begin{bmatrix}
\nu_{2h}^1 \\
\nu_{2h}^2 \\
\vdots \\
\nu_{2h}^{(n-1)/2-1} \\
\nu_{2h}^{(n-1)/2}
\end{bmatrix}
= \frac{1}{4}
\begin{bmatrix}
1 & 2 & 1 & \ldots & 0 & 0 & 0 \\
0 & 0 & \ddots & \ddots & 0 & 0 \\
0 & 0 & 0 & \ldots & 1 & 2 & 1 \\
\vdots \\
\nu_{n-1}^h \\
\nu_n^h
\end{bmatrix}
\begin{bmatrix}
\nu_1^h \\
\nu_2^h \\
\vdots \\
\nu_{n-1}^h \\
\nu_n^h
\end{bmatrix}
\]

It can be shown that $R_h^H = 1/2 P_h^H^T$. 
Coarse grid problems $A_H u^H = f^H$

Either of below 2 methods can be used to obtain coarse grid problems:
- Discretize the PDE again on the coarse mesh.
- Use Galerkin projection. Can be shown that we would then obtain $A_H = R^H_h A_h P^h_H$ and $f^H = R^H_h f^h$. 
Coarse grid problems $A_H u^H = f^H$

Either of below 2 methods can be used to obtain coarse grid problems:
- Discretize the PDE again on the coarse mesh.
- Use Galerkin projection. Can be shown that we would then obtain $A_H = R_h^H A_h P_H^h$ and $f^H = R_h^H f^h$. 

Either of below 2 methods can be used to obtain coarse grid problems:
- Discretize the PDE again on the coarse mesh.
- Use Galerkin projection. Can be shown that we would then obtain $A_H = R_h^H A_h P_H^h$ and $f^H = R_h^H f^h$. 

Coarse grid problems $A_H u^H = f^H$
A Multigrid method: 2-grid cycle

- Only 1 level of coarse mesh is used. Same as previously discussed sample multigrid method except that an exact solve is performed at the coarse level instead of smoothing.
- Will use the residual at the fine grid to obtain the correction to the fine grid solution from coarse grid. - Recall that $h$ denotes fine grid and $H$ denotes coarse grid.

1: Pre smooth : $u^h := \text{smooth}^{\nu_1}(A_h, u^h_o, f_h)$
2: Get residual : $r^h = f^h - A_h u^h$
3: Coarsen : $r^H = R^H_h r_h$
4: Solve : $A_H \delta^H = r^H$
5: Correct $u^h := u^h + P^h_H \delta^H$
6: Post smooth : $u^h := \text{smooth}^{\nu_2}(A_h, u^h, f_h)$

Algorithm: Pseudocode for one 2-grid cycle
A Multigrid method: 2-grid cycle

- Only 1 level of coarse mesh is used. Same as previously discussed sample multigrid method except that a exact solve is performed at the coarse level instead of smoothing.
- Will use the residual at the fine grid to obtain the correction to the fine grid solution from coarse grid. - Recall that $h$ denotes fine grid and $H$ denotes coarse grid.

1: Pre smooth : $u^h := smooth^{\nu_1}(A_h, u^h_0, f_h)$
2: Get residual : $r^h = f^h - A_h u^h$
3: Coarsen : $r^H = R^H_h r_h$
4: Solve : $A_H \delta^H = r^H$
5: Correct $u^h := u^h + P^h_H \delta^H$
6: Post smooth : $u^h := smooth^{\nu_2}(A_h, u^h, f_h)$

**Algorithm:** Pseudocode for one 2-grid cycle
General Multigrid cycle

- Multiple levels and different traversals between coarse to fine and vice versa used. At the coarsest level a direct solve is performed.
- Will use the residual at the fine grid to obtain the correction to the fine grid solution from coarse grid. - Recall that $h$ denotes fine grid and $H$ denotes coarse grid. $h_o$ denotes coarsest level.

0: $u^h = MG(A_H, u^h_o, f^h, \nu_1, \nu_2, \gamma)$
1: Pre smooth: $u^h := smooth^{\nu_1}(A_h, u^h_o, f_h)$
2: Get residual: $r^h = f^h - A_hu^h$
3: Coarsen: $r^H = R^H_h r_h$
4: If $(H == h_o)$
5: Solve: $A_H \delta^H = r^H$
6: Else
7: Recursion: $\delta^H = MG^\gamma(A_H, 0, r^H, \nu_1, \nu_2, \gamma)$
8: EndIf
9: Correct $u^h := u^h + P^h_H \delta^H$
10: Post smooth: $u^h := smooth^{\nu_2}(A_h, u^h, f_h)$
11: Return $u^h$

Algorithm: Pseudocode for a general multigrid cycle.

- $\gamma = 1$ gives a V-cycle.
- $\gamma = 2$ gives a W-cycle.
General Multigrid cycle

- Multiple levels and different traversals between coarse to fine and vice versa used. At the coarsest level a direct solve is performed.
- Will use the residual at the fine grid to obtain the correction to the fine grid solution from coarse grid.
- Recall that $h$ denotes fine grid and $H$ denotes coarse grid. $h_o$ denotes coarsest level

0: $u^h = MG(A_H, u^h_o, f^h, \nu_1, \nu_2, \gamma)$
1: Pre smooth: $u^h := smooth^{\nu_1}(A_h, u^h_o, f_h)$
2: Get residual: $r^h = f^h - A_h u^h$
3: Coarsen: $r^H = R^H_H r_h$
4: If ($H == h_o$)
5: Solve: $A_H \delta^H = r^H$
6: Else
7: Recursion: $\delta^H = MG^\gamma(A_H, 0, r^H, \nu_1, \nu_2, \gamma)$
8: EndIf
9: Correct $u^h := u^h + P^h_H \delta^H$
10: Post smooth: $u^h := smooth^{\nu_2}(A_h, u^h, f_h)$
11: Return $u^h$

Algorithm: Pseudocode for a general multigrid cycle

- $\gamma = 1$ gives a $V$-cycle.
- $\gamma = 2$ gives a $W$-cycle.
General Multigrid cycle

- Multiple levels and different traversals between coarse to fine and vice versa used. At the coarsest level a direct solve is performed.
- Will use the residual at the fine grid to obtain the correction to the fine grid solution from coarse grid. - Recall that \( h \) denotes fine grid and \( H \) denotes coarse grid. \( h_o \) denotes coarsest level

0: \( u^h = MG(A_H, u^h_o, f^h, \nu_1, \nu_2, \gamma) \)
1: Pre smooth : \( u^h := smooth^{\nu_1}(A_h, u^h_o, f_h) \)
2: Get residual : \( r^h = f^h - A_h u^h \)
3: Coarsen : \( r^H = R^H_h r_h \)
4: If \( (H == h_o) \)
5: Solve: \( A_H \delta^H = r^H \)
6: Else
7: Recursion: \( \delta^H = MG^\gamma(A_H, 0, r^H, \nu_1, \nu_2, \gamma) \)
8: EndIf
9: Correct \( u^h := u^h + P^h_H \delta^H \)
10: Post smooth : \( u^h := smooth^{\nu_2}(A_h, u^h, f_h) \)
11: Return \( u^h \)

Algorithm: Pseudocode for a general multigrid cycle.

- \( \gamma = 1 \) gives a \( V \)-cycle.
- \( \gamma = 2 \) gives a \( W \)-cycle.
Notation and some specific multigrid cycles

- Smoothing
  - Evaluate residual and restrict it to coarse mesh
  - Prolongate coarse grid correction and use it to correct fine grid solution
- Exact solve for the coarse mesh update
  - Prolongate solution (Higher order than prolongation used for coarse grid correction)

**Figure: V-cycle - 1 level ($\gamma = 1$)**

**Figure: V-cycle - 2 level ($\gamma = 1$)**

**Figure: W-cycle - 2 level ($\gamma = 2$)**

**Figure: W-cycle - 3 level ($\gamma = 2$)**
Full multigrid cycle

1: $h := h_o$ (Coarsest level). Solve $A_h u^h = f^h$
2: $\textbf{for } l = 1 \textbf{ to } p \textbf{ do}$
3: $u^{h/2} = \hat{P}^{h/2}_h u^h$
4: $h := h/2$
5: $u^h = MG^\mu(A_h, u^h, f^h, \nu_1, \nu_2, \gamma)$
6: $\textbf{end for}$

Algorithm: Pseudocode for a general multigrid cycle

- $\hat{P}^{h/2}_h$ (operation indicated by dotted line) is a special interpolation matrix which is higher order that $P^{h/2}_h$
- Common to choose $\mu = 1$
Full multigrid cycle

1: \( h := h_0 \) (Coarsest level). Solve \( A_h u^h = f^h \)
2: \textbf{for} \( l = 1 \) to \( p \) \textbf{do}
3: \( u^{h/2} = \hat{P}_{h}^{h/2} u^h \)
4: \( h := h/2 \)
5: \( u^h = MG^\mu(A_h, u^h, f^h, \nu_1, \nu_2, \gamma) \)
6: \textbf{end for}

**Algorithm:** Pseudocode for a general multigrid cycle

- \( \hat{P}_{h}^{h/2} \) (operation indicated by dotted line) is a special interpolation matrix which is higher order than \( P_{h}^{h/2} \)
- Common to choose \( \mu = 1 \)
Full multigrid cycle

1: \( h := h_o \) (Coarsest level). Solve \( A_h u^h = f^h \)
2: for \( l = 1 \) to \( p \) do
3: \( u^{h/2} = \hat{P}^{h/2}_h u^h \)
4: \( h := h/2 \)
5: \( u^h = MG^\mu(A_h, u^h, f^h, \nu_1, \nu_2, \gamma) \)
6: end for

**Algorithm:** Pseudocode for a general multigrid cycle

- \( \hat{P}^{h/2}_h \) (operation indicated by dotted line) is a special interpolation matrix which is higher order than \( P^{h/2}_h \)
- Common to choose \( \mu = 1 \)
Full Multigrid method analysis results.

- Let $u$ be the exact solution of PDE, $u^h$ be the solution to discretized PDE problem, $\tilde{u}^h$ be the solution after 1 full multigrid cycle. Then, following results can be shown
  - $\|u - u^h\| \leq c h^\kappa$ (Approximation theory of Finite Difference methods)
  - $\|u^h - \tilde{u}^h\| \leq c_3 c_1 h^\kappa$

where $c_3, c_1, c$ are positive constants i.e. Error in $\tilde{u}^h$ is of the same order as the error in $u^h$!! No better approximate solution to discrete equation is needed.
- Operation count for $\gamma < \gamma_{\text{critical}}$ is $O(n)$
Full Multigrid method analysis results.

- Let $u$ be the exact solution of PDE, $u^h$ be the solution to discretized PDE problem, $\tilde{u}^h$ be the solution after 1 full multigrid cycle. Then, following results can be shown
- $\|u - u^h\| \leq ch^\kappa$ (Approximation theory of Finite Difference methods)
- $\|u^h - \tilde{u}^h\| \leq c_3 c_1 h^\kappa$
where $c_3, c_1, c$ are positive constants i.e. Error in $\tilde{u}^h$ is of the same order as the error in $u^h$!! No better approximate solution to discrete equation is needed.
- Operation count for $\gamma < \gamma_{critical}$ is $O(n)$
Full Multigrid method analysis results.

- Let $u$ be the exact solution of PDE, $u^h$ be the solution to discretized PDE problem, $\tilde{u}^h$ be the solution after 1 full multigrid cycle. Then, following results can be shown
  - $||u - u^h|| \leq ch^\kappa$ (Approximation theory of Finite Difference methods)
  - $||u^h - \tilde{u}^h|| \leq c_3 c_1 h^\kappa$
  where $c_3$, $c_1$, $c$ are positive constants i.e. Error in $\tilde{u}^h$ is of the same order as the error in $u^h$!! No better approximate solution to discrete equation is needed.
- Operation count for $\gamma < \gamma_{critical}$ is $O(n)$
Full Multigrid method analysis results.

- Let $u$ be the exact solution of PDE, $u^h$ be the solution to discretized PDE problem, $\tilde{u}^h$ be the solution after 1 full multigrid cycle. Then, following results can be shown:
  - $||u - u^h|| \leq ch^\kappa$ (Approximation theory of Finite Difference methods)
  - $||u^h - \tilde{u}^h|| \leq c_3 c_1 h^\kappa$
  where $c_3, c_1, c$ are positive constants i.e. Error in $\tilde{u}^h$ is of the same order as the error in $u^h$!! No better approximate solution to discrete equation is needed.
  - Operation count for $\gamma < \gamma_{critical}$ is $O(n)$
Full Multigrid method analysis results.

- Let \( u \) be the exact solution of PDE, \( u^h \) be the solution to
discretized PDE problem, \( \tilde{u}^h \) be the solution after 1 full multigrid
cycle. Then, following results can be shown
- \( \| u - u^h \| \leq c h^\kappa \) (Approximation theory of Finite Difference
methods)
- \( \| u^h - \tilde{u}^h \| \leq c_3 c_1 h^\kappa \)
where \( c_3, c_1, c \) are positive constants i.e. Error in \( \tilde{u}^h \) is of the same
order as the error in \( u^h \)! No better approximate solution to
discrete equation is needed.
- Operation count for \( \gamma < \gamma_{critical} \) is \( O(n) \)
Example

Problem -
\[- \Delta u = 13 \sin (2\pi x) \times \sin (3\pi y) \text{ for } (x, y) \in (0, 1) \times (0, 1) \text{ and Dirichlet BCs on all sides}

Sequence of meshes : \( nx = ny = \{ 9, 17, 33, 65, 129 \} \)

Quantity plotted -
Error \( \| u - \tilde{u}_h \|_2 \) for Full multigrid with different smoothing techniques are compared :
- Notation to denote smoothing technique is \( Smoothing\_type(\nu_1, \nu_2) \) where \( \nu_1 \) and \( \nu_2 \) are the number of pre-smoothing and post-smoothing iterations.
Figure 13.8: FMG error norms with various smoothers versus the discretization error as a function of the mesh size.
Other related techniques

This technique is also generalized to solve $Ax = b$ where the underlying mesh is not accessible - Algebraic multigrid method - Parallel implementation of Algebraic Multigrid techniques are present in packages like HYPRE, Trilinos. These techniques are essential for parallel scalability (100,000 cores) of incompressible unstructured Computational Fluid Dynamics codes, linear solid mechanics codes.
Thank you !!
Backup slides