Introduction to Multigrid Methods

Jaap van der Vegt

Numerical Analysis and Computational Science Group
Department of Applied Mathematics
Universiteit Twente

Enschede, The Netherlands

Part 2.

USTC Summer School, August 29 – September 2, 2022

Basic Multigrid I

- The two main ingredients of a multigrid method are:
  - Error smoothing
  - Coarse grid correction

Error Smoothing Procedure

Pointwise smoothers, Jacobi and Gauss-Seidel methods

- Jacobi-type iteration

$$z_h^{m+1}(x_i, y_j) = \frac{1}{4}\big[h^2 f_h(x_i, y_j) + u_h^m(x_i - h, y_j) + u_h^m(x_i + h, y_j) + u_h^m(x_i, y_j - h) + u_h^m(x_i, y_j + h)\big]$$

$$u_h^{m+1} = z_h^{m+1} \qquad (x_i, y_j) \in \Omega_h.$$

Jacobi Iteration

- We can write the Jacobi iteration as

$$u_h^{m+1} = S_h u_h^m + \frac{h^2}{4} f_j$$

with Jacobi iteration operator $S_h = I_h - \frac{h^2}{4} L_h$ and

$$\begin{aligned} L_h u_h &= \frac{1}{h^2} \big[4u_h(x_i, y_j) - u_h(x_i - h, y_j) - u_h(x_i + h, y_j) - u_h(x_i, y_j - h) \\ &\qquad - u_h(x_i, y_j + h) \\ &= \frac{1}{h^2} \begin{bmatrix} & -1 & \\ -1 & 4 & -1 \\ & -1 & \end{bmatrix}_h u_h(x_i, y_j) \qquad \text{(stencil notation)} \end{aligned}$$

Damped Jacobi Iteration

- Introducing a relaxation parameter ω gives the damped Jacobi iteration method

$$u_h^{m+1} = u_h^m + \omega(z_h^{m+1} - u_h^m)$$

with z_h^{m+1} the result of the undamped Jacobi iteration.

- Using the undamped Jacobi iteration operator $S_h = I_h - \frac{h^2}{4} L_h$ gives

$$\begin{aligned} u_h^{m+1} &= u_h^m + \omega(z_h^{m+1} - u_h^m) \\ &= u_h^m + \omega(S_h u_h^m + \frac{h^2}{4} f_h - u_h^m) \\ &= (I_h + \omega I_h - \frac{\omega h^2}{4} L_h - \omega I_h) u_h^m + \frac{\omega h^2}{4} f_h \\ &= S_h(\omega) u_h^m + \frac{\omega h^2}{4} f_h \end{aligned}$$

with $S_h(\omega) = I_h - \frac{\omega h^2}{4} L_h$ the damped Jacobi iteration operator.

Analysis of Jacobi Iteration Method for the Laplace Equation

- Consider the Laplace eigenvalue problem

$$-\Delta u = \lambda u \qquad \text{in } (0,1)^2 = \Omega$$
$$u = 0 \qquad \text{at } \partial\Omega$$

with eigenfunctions

$$u(x,y) = A \sin k\pi x \sin l\pi y$$

and eigenvalues

$$\lambda = (k^2 + l^2)\pi^2$$

Analysis of Jacobi Iteration Method for the Laplace Equation

- Consider the discrete eigenvalue problem with periodic boundary conditions

$$L_h u_h = \lambda_h u_h$$

$$L_h u_h = \frac{1}{h^2}(4u_{i,j} - u_{i-1,j} - u_{i+1,j} - u_{i,j-1} - u_{i,j+1})$$

with discrete eigenfuctions

$$u_h(x_i, y_j) = e^{\iota k\pi ih + \iota l\pi jh} =: \phi_h^{k,l}(x, y)$$

with $\iota = \sqrt{-1}$ and discrete eigenvalues

$$\lambda_h = \frac{1}{h^2}(4 - 2\cos k\pi h - 2\cos l\pi h)$$

Analysis of Jacobi Iteration Method for Laplace Equation

- Expand λ_h in a Taylor series

$$\begin{aligned}\lambda_h &= \frac{1}{h^2}(4 - 2(1 - \frac{k^2\pi^2h^2}{2} + \frac{k^4\pi^4h^4}{24} - \cdots) \\ &\quad - 2(1 - \frac{l^2\pi^2h^2}{2} + \frac{l^4\pi^4h^4}{24} - \cdots)) \\ &= k^2\pi^2 - \frac{k^4\pi^4h^2}{12} + \cdots + l^2\pi^2 - \frac{l^2\pi^4h^2}{12} + \cdots)\end{aligned}$$

- Hence

$$\lim_{h\to 0} \lambda_h = \lambda$$

Analysis of Jacobi Iteration Method for Laplace Equation

- Consider the damped Jacobi iteration

$$u_h^{m+1} = S_h(\omega) u_h^m + f_h$$

with Jacobi operator

$$S_h(\omega) = I_h - \frac{\omega h^2}{4} L_h$$

- Since $\phi_h^{k,l} = e^{\iota k \pi i h + \iota l \pi j h}$ are eigenfunctions of L_h they are also eigenfunctions of $S_h(\omega)$

Analysis of Jacobi Iteration Method for Laplace Equation

- Consider

$$S_h(\omega)u_h = \lambda_h u_h$$

Use the ansatz $u_h = e^{\iota k\pi ih + \iota l\pi jh}$, then we obtain

$$\frac{\omega}{4}\left(4\left(\frac{1}{\omega} - 1\right) + e^{-\iota k\pi h} + e^{\iota k\pi h} + e^{\iota l\pi h} + e^{-\iota l\pi h}\right) = \lambda_h$$

Analysis of Jacobi Iteration Method for Laplace Equation

- The discrete eigenvalues of the Jacobi iteration methods are

$$\begin{aligned}
\lambda_h &= \frac{\omega}{4}\Big(4\big(\frac{1}{\omega}-1\big)+2\cos k\pi h+2\cos l\pi h\Big)\\
&= 1-\omega+\frac{\omega}{2}\cos k\pi h+\frac{\omega}{2}\cos l\pi h\\
&= 1-\frac{\omega}{2}(2-\cos k\pi h-\cos l\pi h)\\
&=: \chi_h^{k,l}
\end{aligned}$$

Analysis of Jacobi Iteration Method for Laplace Equation

- Spectral radius

$$\rho(S_h) = \max\{|\chi_h^{k,l}| \;\; (k,l = 1, \cdots, n-1)\}$$

- The spectral radius of the Jacobi iteration then is equal to

$$\text{for } 0 < \omega \leq 1: \quad \rho(S_h) = |\chi_h^{1,1}| = |1 - \omega + \omega \cos \pi h| = 1 - O(\omega h^2)$$

$$\text{else} \qquad\qquad \rho(S_h) \geq 1 \qquad \text{for } h \text{ small enough}$$

Smoothing Properties of Damped Jacobi Relaxation

- As an iterative solver, without multigrid acceleration, the best (but poor) asymptotic convergence rate of the damped Jacobi method is for $\omega = 1$.

  Hence relaxation does not improve convergence of the Jacobi method.

- The smoothest eigenfunction $\phi_h^{1,1}$ is responsible for the slow convergence of the Jacobi method.

- Highly oscillatory modes can be damped better with $0 < \omega < 1$.

Smoothing Properties of Damped Jacobi Relaxation

- Consider w_h and $\bar{w}_h$ before and after one relaxation step (e.g. Jacobi iteration). Then, the error is

$$v_h := u_h - w_h = \sum_{k,l}^{n-1} \alpha_{k,l} \phi_h^{k,l}$$

$$\bar{v}_h := u_h - \bar{w}_h = \sum_{k,l}^{n-1} \chi_h^{k,l} \alpha_{k,l} \phi_h^{k,l}$$

- Split the solution into high and low frequency modes.

Smoothing Properties of Damped Jacobi Relaxation

- Smoothing Factor

The smoothing factor $\mu(h;\omega)$ of $S_h(\omega)$ represents the worst factor by which high frequency error modes are reduced per relaxation step.

$$\mu(h;\omega) := \max\{|\chi_h^{k,l}(\omega)| \quad \frac{n}{2} \leq \max(k,l) \leq n-1\}$$

$$\mu^*(\omega) := \sum_{h\in\mathcal{H}} \mu(h;\omega)$$

with $\mathcal{H}$ the set of all admissible mesh sizes.

Smoothing Properties of Damped Jacobi Relaxation

- For the damped Jacobi iteration method the smoothing factor is

$$\mu(h;\omega) := \max\{|1 - \frac{\omega}{2}(2 - \cos k\pi h - \cos l\pi h)| \quad : \quad \frac{n}{2} \leq \max(k,l) \leq n-1\}$$

$$\mu^*(\omega) := \max\{|1 - \frac{\omega}{2}|, |1 - 2\omega|\}$$

- The choice $\omega = \frac{4}{5}$ is optimal since

$$\min\{\mu^*(\omega) \ : \ 0 < \omega \leq 1\} = \mu^*(\frac{4}{5}) = \frac{3}{5}$$

- Thus for $\omega = \frac{4}{5}$ the damped Jacobi method reduces all high frequency components by at least a factor $\frac{3}{5}$, independent of the mesh size h.

- This type of analysis is called a smoothing analysis.

Gauss-Seidel Relaxation

- For the Poisson equation $-\Delta u = f$ the lexicographic Gauss-Seidel method with relaxation parameter ω is

$$z_{i,j}^{m+1} = \frac{1}{4}(h^2 f_{i,j} + u_{i-1,j}^{m+1} + u_{i+1,j}^{m} + u_{i,j-1}^{m+1} + u_{i,j+1}^{m})$$

$$u_{i,j}^{m+1} = u_{i,j}^{m} + \omega(z_{i,j}^{m+1} - u_{i,j}^{m})$$

- A similar scheme is possible for red-black ordering.

Gauss-Seidel Relaxation

- The convergence rate can be substantially improved by overrelaxation:

$$\omega^* = \frac{2}{1+\sqrt{1-\rho(JAC)^2}} \qquad \text{with } \rho(JAC) = |\cos \pi h|$$

$$= \frac{2}{1+\sin \pi h}$$

Then

$$\rho(\omega^* - GS) = \omega^* - 1 = \frac{1-\sin \pi h}{1+\sin \pi h} = 1 - O(h)$$

instead of

$$\rho(GS) = 1 - O(h^2) \qquad \text{for } \omega = 1.$$

- The convergence acceleration results also apply to Gauss-Seidel methods with other orderings.

Gauss-Seidel Relaxation

- The smoothing analysis of the Gauss-Seidel method is more complicated than for the Jacobi methods.

  The main results are:

$$\mu(\text{GS-Lexicographic Ordering}) = 0.50 \qquad \text{for } \omega = 1$$

$$\mu(\text{GS-Red-Black Ordering}) = 0.25 \qquad \text{for } \omega = 1$$

Parallel Properties of Smoothers

- $\omega - JAC$ is fully parallel. The relation can be applied simultaneously to all points

  Degree of parallelism is: $\text{par-deg}(\omega - JAC) = \sharp\Omega_h$.

- $GS - LEX$ has dependencies. Grid points lying on a diagonal of Ω^h can be treated in parallel.

$$\text{par-deg}(GS - LEX) = (\sharp\Omega_h)^{\frac{1}{2}}$$

- For $GS - RB$ each step contains two half steps, viz. red and black points, which can be treated simultaneously.

$$\text{par-deg}(GS - RB) = \frac{1}{2}\sharp\Omega_h$$

Parallel Properties of Smoothers

- Overview performance of various smoothers

Relaxation	Smoothing Factor	Smoothing	Parallel Degree
$\omega - JAC, \omega = 1$	1	No	N Full
$\omega - JAC, \omega = 0.5$	0.75	Unsatisfactory	N Full
$\omega - JAC, \omega = 0.8$	0.6	Acceptable	N Full
$GS - LEX, \omega = 1$	0.5	Good	$\leq \sqrt{N}$
$GS - RB, \omega = 1$	0.25	Very Good	$\frac{1}{2}N$

Two-Grid Multigrid Cycle

- Consider a discrete elliptic boundary value problem (e.g. Laplace equation)

$$L_h u_h = f_h \qquad \text{in } \Omega_h$$

- Assume L_h^{-1} exists.

Iteration by Approximate Solution of Defect Equation

- For any approximation u_h^m of u_h we denote the error as

$$v_h^m := u_h - u_h^m$$

- Define the defect d_h^m as

$$d_h^m := f_h - L_h u_h^m$$

- The defect equation is then

$$\begin{aligned} L_h v_h^m &= L_h u_h - L_h u_h^m \\ &= f_h - L_h u_h^m \\ &= d_h^m \end{aligned}$$

Iteration by Approximate Solution of Defect Equation

- The defect algorithm (not useful for computation) is

$$u_h^m \longrightarrow d_h^m = f_h - L_h u_h^m \longrightarrow L_h v_h^m = d_h^m \longrightarrow u_h = u_h^m + v_h^m$$

- This algorithm is not useful for computation since it's computational cost is identical to the original problem $L_h u_h = f_h$.

- Approximate L_h by a simpler operator $\widehat{L}_h$, such that $\widehat{L}^{-1}$ exists.

Iteration by Approximate Solution of Defect Equation

- New defect algorithm

$$\widehat{L}_h \widehat{v}_h^m = d_h^m$$

$$u_h^{m+1} = u_h^m + \widehat{v}_h^m$$

- Note, for the defect equation we do not need an exact solution of the error, only a reasonable approximation $\widehat{v}_h^m$

- The algorithm can be summarized as

$$u_h^m \longrightarrow d_h^m = f_h - L_h u_h^m \longrightarrow \widehat{L}_h \widehat{v}_h^m = d_h^m \longrightarrow u_h^{m+1} = u_h^m + \widehat{v}_h^m$$

Iteration Operators Defect Correction Process

- The iteration operator of the defect correction process is

$$u_h^{m+1} = M_h u_h^m + s_h,$$

with iteration operator

$$M_h = I_h - C_h L_h : \mathcal{G}_h \longrightarrow \mathcal{G}_h, \quad \text{with } C_h := \widehat{L}_h^{-1}.$$

Iteration Operators Defect Correction Process

- Proof of defect correction iteration operator M_h

$$
\begin{aligned}
u_h^{m+1} &= u_h^m + \widehat{v}_h^m \\
&= u_h^m + \widehat{L}_h^{-1} d_h^m \\
&= u_h^m + \widehat{L}_h^{-1}(f_h - L_h u_h^m) \\
&= (I_h - \widehat{L}_h^{-1} L_h) u_h^m + \widehat{L}_h^{-1} f_h \\
&= M_h u_h^m + s_h
\end{aligned}
$$

Iteration Operators Defect Correction Process

- For the error $v_h^m = u_h - u_h^m$ we obtain the iteration process

$$\begin{aligned} v_h^{m+1} &= u_h - u_h^{m+1} \\ &= u_h - M_h u_h^m - s_h \\ &= u_h - (I_h - \widehat{L}^{-1} L_h) u_h^m - \widehat{L}^{-1} f_h \\ &= u_h - (I_h - \widehat{L}^{-1} L_h) u_h^m - \widehat{L}^{-1} L_h u_h \\ &= (I_h - \widehat{L}^{-1} L_h) u_h - (I_h - \widehat{L}^{-1} L_h) u_h^m \\ &= (I_h - \widehat{L}^{-1} L_h) v_h^m \\ &= M_h v_h^m \end{aligned}$$

Iteration Operators Defect Correction Process

- The defect d_h^m is computed in the following steps

$$d_h^m \longrightarrow \hat{L}_h \widehat{v}_h^m = d_h^m \longrightarrow u_h^{m+1} = u_h^m + \widehat{v}_h^m \longrightarrow d_h^{m+1}$$

- The iteration process for d_h^m is

$$\begin{aligned} d_h^{m+1} &= f_h - L_h u_h^{m+1} \\ &= f_h - L_h(u_h^m + \widehat{v}_h^m) \\ &= f_h - L_h(u_h^m + \widehat{L}_h^{-1} d_h^m) \\ &= d_h^m - L_h \widehat{L}^{-1} d_h^m \\ &= (I_h - L_h \widehat{L}_h^{-1}) d_h^m \\ &= L_h(I_h - \widehat{L}_h^{-1} L_h) L_h^{-1} d_h^m = L_h M_h L_h^{-1} d_h^m \\ &= (I_h - L_h C_h) d_h^m \end{aligned}$$

- The defect d_h^m is important since it can be computed easily, whereas the error v_h^m cannot.

Iteration Process for Solution u_h^m

- In order to obtain the iteration operator for the solution we use the relation

$$
\begin{aligned}
&(I + M + M^2 + \cdots + M^{m-1})(I - M) \\
&= I - M + M - M^2 + M^2 - M^3 + \cdots - M^{m-2} + M^{m-2} + M^{m-1} - M^m \\
&= I - M^m
\end{aligned}
$$

- Hence

$$I + M + M^2 + \cdots + M^{m-1} = (I - M^m)(I - M)^{-1}$$

Iteration Process for Solution u_h^m

- Start with $u_h^0 = 0$ and use $u_h^{m+1} = M_h u_h^m + \widehat{L}_h^{-1} f_h$, then

$$
\begin{aligned}
u_h^1 &= \widehat{L}_h^{-1} f_h \\
u_h^2 &= M_h u_h^1 + \widehat{L}_h^{-1} f_h = (M_h + I_h)\widehat{L}_h^{-1} f_h \\
u_h^3 &= M_h u_h^2 + \widehat{L}_h^{-1} f_h = (M_h^2 + M_h + I_h)\widehat{L}_h^{-1} f_h \\
&\vdots \\
u_h^m &= (I_h + M_h + \cdots + M_h^{m-1})\widehat{L}_h^{-1} f_h \\
&= (I_h - M_h^m)(I_h - M_h)^{-1}\widehat{L}_h^{-1} f_h \\
&= (I_h - M_h^m)(I_h - I_h + \widehat{L}_h^{-1} L_h)^{-1}\widehat{L}_h^{-1} f_h \\
&= (I_h - M_h^m) L_h^{-1} f_h
\end{aligned}
$$

Convergence Factors

- Asymptotic convergence factor

$$\rho(M_h) = \rho(I_h - C_h L_h) = \rho(I_h - L_h C_h)$$

- Error reduction factor

$$\| I_h - C_h L_h \|$$

- Defect reduction factor

$$\| I_h - L_h C_h \|$$

Coarse Grid Correction

- One way to approximately solve the defect equation is to use a coarser grid Ω_H.
- Replace L_h with L_H in the defect equation and assume that L_H^{-1} exists

$$L_H \widehat{v}_H^m = d_H^m$$

with

$$L_H : \mathcal{G}(\Omega_H) \longrightarrow \mathcal{G}(\Omega_H), \qquad \dim\mathcal{G}(\Omega_H) < \dim\mathcal{G}(\Omega_h)$$

Transfer Operators

- The coarse grid correction requires transfer operators that connect the coarse and fine grid data

$$I_h^H : \mathcal{G}(\Omega_h) \longrightarrow \mathcal{G}(\Omega_H) \qquad \text{restriction operator}$$

$$I_H^h : \mathcal{G}(\Omega_H) \longrightarrow \mathcal{G}(\Omega_h) \qquad \text{prolongation operator}$$

- The defect and error correction on the coarse and fine mesh then are related as

$$d_H^m = I_h^H d_h^m$$

$$\widehat{v}_h^m = I_H^h \widehat{v}_H^m$$

- The simplest restriction operator is an injection

$$d_H(P) = I_h^H d_h(P) := d_h(P) \qquad \text{for } P \in \Omega_H \subset \Omega_h$$

Coarse Grid Correction

- A coarse correction consists of the following steps

Compute defect	$d_h^m = f_h - L_h u_h^m$
Restrict defect	$d_H^m = I_h^H d_h^m$
Solve on Ω_H	$L_H \widehat{v}_H^m = d_H^m$
Interpolate correction	$\widehat{v}_h^m = I_H^h \widehat{v}_M^m$
Compute new approximation	$u_h^{m+1} = u_h^m + \widehat{v}_h^m$

Two Grid Operator

- Consider

$$
\begin{aligned}
u_h^{m+1} &= u_h^m + \widehat{v}_h^m \\
&= u_h^m + I_H^h \widehat{v}_H^m \\
&= u_h^m + I_H^h L_H^{-1} d_H^m \\
&= u_h^m + I_H^h L_H^{-1} I_h^H d_h^m \\
&= u_h^m + I_H^h L_H^{-1} I_h^H (f_h - L_h u_h^m) \\
&= (I_h - I_H^h L_H^{-1} I_h^H L_h) u_h^m + I_H^h L_H^{-1} I_h^H f_h
\end{aligned}
$$

- The two grid iteration operator is

$$
M_h = I_h - I_H^h L_H^{-1} I_h^H L_h = I_h - C_h L_h \qquad \text{with } C_h = I_H^h L_H^{-1} I_h^H
$$

Two Grid Operator

- The multigrid algorithm with only the coarse grid correction is not convergent

$$\rho(I_h - I_H^h L_H^{-1} I_h^H L_h) \geq 1$$

- The lack of convergence of the two-grid algorithm follows directly from the fact that I_h^H maps $\mathcal{G}(\Omega)$ into the lower dimensional space $\mathcal{G}(\Omega_H)$.

  The matrix $C_h = I_H^h L_H^{-1} I_h^H$ is therefore not invertible. This implies that

$$C_h L_h v_h = 0 \quad \text{for certain } v_h \neq 0$$

  Hence these modes will not be corrected by the coarse grid correction.

Two Grid Operator

- Example: for the simple injection operator $d_H(P) = I_h^H d_h(P)$ for $P \in \Omega_H \subset \Omega_h$ any error function $v_h \in \mathcal{G}(\Omega_h)$ with

$$L_h v_h(P) = \begin{cases} 0 & \text{for } P \in \Omega_H \\ \text{arbitrary} & \text{for } P \notin \Omega_H \end{cases}$$

  is annihilated by I_h^H and therefore by C_h.

Structure of Two Grid Operator

- For an efficient multigrid algorithm we need both smoothing and coarse grid correction.
- Two grid cycle

$$u_h^{m+1} = TGCYC(u_h^m, L_h, f_h, \nu_1, \nu_2)$$

1. Pre-smoothing
   - Compute $\bar{u}_h^m$ by applying $\nu_1 (\geq 0)$ smoothing steps to u_h^m.

$$\bar{u}_h^m = SMOOTH^{\nu_1}(u_h^m, L_h, f_h)$$

Structure of Two Grid Operator

2. Coarse grid correction
   - Compute defect $\bar{d}_h^m = f_h - L_h \bar{u}_h^m$
   - Restrict defect $\bar{d}_H^m = I_h^H \bar{d}_h^m$
   - Solve on Ω_H $L_H \hat{v}_H^m = \bar{d}_H^m$
   - Interpolate correction $\hat{v}_h^m = I_H^h \hat{v}_H^m$
   - Compute correction $\tilde{u}_h^m = \bar{u}_h^m + \hat{v}_h^m$
3. Post-smoothing
   - Compute u_h^{m+1} by applying $\nu_2 (\geq 0)$ smoothing steps

$$u_h^m = SMOOTH^{\nu_2}(\tilde{u}_h^m, L_h, f_h)$$

Structure of Two-Grid Cycle

$$
\begin{array}{ccccc}
u_h^m & \xrightarrow[SMOOTH^{\nu_1}]{} & \bar{u}_h^m & \longrightarrow & \bar{d}_h^m = f_h - L_h \bar{u}_h^m \\
 & & & & \Big\downarrow I_h^H \bar{d}_h^m \\
 & & & & \bar{d}_H^m \\
 & & & & \Big\downarrow \\
 & & & & L_H \hat{v}_H^m = \bar{d}_H^m \\
 & & & & \Big\uparrow I_H^h \hat{v}_H^m \\
\hat{v}_h^m & \longrightarrow & \bar{u}_h^m + \hat{v}_h^m & \xrightarrow[SMOOTH^{\nu_2}]{} & u_h^{m+1}
\end{array}
$$

- Multigrid iteration operator

$$M_h^H = S_h^{\nu_2} K_h^H S_h^{\nu_1} \qquad \text{with} \quad K_h^H = I_h - I_H^h L_H^{-1} I_h^H L_h.$$

Multigrid Efficiency

- Multigrid efficiency is (strongly) affected by
  - Type of smoother
  - Number of pre- and post-relaxation steps ν_1 and ν_2
  - Coarse grid Ω_H
  - Restriction operator I_h^H
  - Coarse grid operator L_H
  - Prolongation operator I_H^h
- There are no simple rules to select these components. In general it requires a detailed mathematical analysis using e.g. Fourier multilevel analysis to obtain optimal multigrid algorithms.

Coarse Grids

- Coarse grids use either
  - Standard coarsening $H = 2h$, which implies $\sharp\Omega_H \cong \frac{1}{2^d}\sharp\Omega_h$.
  - Semi-coarsening, which is useful for anisotropic operators.

Coarse Grids

- Coarse grid operators for the Laplacian:
  - Stencil standard coarse grid operator

$$L_H = \frac{1}{H^2} \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}_H$$

  - Galerkin coarse grid correction

$$L_H = I_h^H L_h I_H^h,$$

    which preserves important properties of L_h such as symmetry and positive definiteness and is useful for e.g. problems with discontinuous coefficients.

Transfer Operators

- Transfer operators

1. Restriction

   - Pure injection operator
   - Full weighting operator

$$I_h^{2h} = \frac{1}{16} \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_h^{2h}$$

For $(x, y) \in \Omega_{2h}$ we have then

$$\begin{aligned} d_{2h}(x,y) &= I_h^{2h} d_h(x,y) \\ &= \frac{1}{16}\Big(4d_h(x,y) + 2d_h(x+h,y) + 2d_h(x-h,y) + 2d_h(x,y+h) + \\ &\quad + 2d_h(x,y-h) + d_h(x+h,y+h) + d_h(x+h,y-h) \\ &\quad + d_h(x-h,y+h) + d_h(x-h,y-h)\Big) \end{aligned}$$

Transfer Operators

- Half weighting operator

$$I_h^{2h} = \frac{1}{8} \begin{bmatrix} 0 & 1 & 0 \\ 1 & 4 & 1 \\ 0 & 1 & 0 \end{bmatrix}_h^{2h}$$

Transfer Operators

2. Interpolation operator

$$I_{2h}^{h}\widehat{v}_{2h}(x,y) = \begin{cases} \widehat{v}_{2h}(x,y), & (x,y) = (2nh, 2mh) \\ \frac{1}{2}(\widehat{v}_{2h}(x,y+h) + \widehat{v}_{2h}(x,y-h)), & (x,y) = (2nh, (2m+1)h) \\ \frac{1}{2}(\widehat{v}_{2h}(x+h,y) + \widehat{v}_{2h}(x-h,y)), & (x,y) = ((2n+1)h, 2mh) \\ \frac{1}{4}(\widehat{v}_{2h}(x+h,y+h) + \widehat{v}_{2h}(x+h,y-h) + \widehat{v}_{2h}(x-h,y+h) & \\ \quad +\widehat{v}_{2h}(x-h,y-h)), & (x,y) = ((2n+1)h, (2m+1)h) \end{cases}$$

with stencil

$$I_{2h}^{h} = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}_{2h}^{h}$$

Note, the full weighting restriction operator I_h^{2h} and the interpolation operator I_{2h}^h are, apart from a scaling factor, each others transpose.

Multigrid Cycle

- In a two grid method it is not necessary to solve the coarse grid defect equation exactly.
- It is more efficient to obtain a suitable approximation using a sequence of coarser meshes.

Recursive Definition Multigrid Algorithm

- Define a sequence of meshes Ω_{h_l} (finest mesh), $\Omega_{h_{l-1}}, \cdots, \Omega_{h_0}$ (coarsest mesh)
- Define on each mesh Ω_k

$$
\begin{aligned}
&L_k : \mathcal{G}(\Omega_k) \to \mathcal{G}(\Omega_k) && \text{(discretization)} \\
&S_k : \mathcal{G}(\Omega_k) \to \mathcal{G}(\Omega_k) && \text{(smoother)} \\
&I_k^{k-1} \mathcal{G}(\Omega_k) \to \mathcal{G}(\Omega_{k-1}) && \text{(restriction)} \\
&I_{k-1}^{k} \mathcal{G}(\Omega_{k-1}) \to \mathcal{G}(\Omega_k) && \text{(prolongation)} \\
&L_k u_k = f_k && \text{on } \Omega_k
\end{aligned}
$$

Recursive Definition Multigrid Algorithm

- Multigrid Cycle $u_k^{m+1} = MGCYC(k, \gamma, u_k^m, L_k, f_k, \nu_1, \nu_2)$

  1. Pre-smoothing $\bar{u}_k^m = SMOOTH^{\nu_1}(u_k^m, L_k, f_k)$
  2. Coarse grid correction
     - Compute defect $\bar{d}_k^m = f_k - L_k \bar{u}_k^m$
     - Restrict defect $\bar{d}_{k-1}^m = I_k^{k-1} \bar{d}_k^m$
     - Compute approximate solution $\hat{v}_{k-1}^m$ on Ω_{k-1} by solving

$$L_{k-1} \hat{v}_{k-1}^m = \bar{d}_{k-1}^m$$

if $k = 1$ use a direct solver (using an iterative solver generally decreases multigrid efficiency)

if $k > 1$ perform $\gamma (\geq 1)$ k-grid cycles using the zero solution as first approximation for $\hat{v}_{k-1}^m$

$$\hat{v}_{k-1}^m = MGCYC^{\gamma}(k-1, \gamma, 0, L_{k-1}, \bar{d}_{k-1}^m, \nu_1, \nu_2)$$

Recursive Definition Multigrid Algorithm

- Interpolate correction $\widehat{v}^m = I_{k-1}^k \widehat{v}_{k-1}^m$ to Ω_k
- Compute corrected approximation on Ω_k: $\tilde{u}_k^m = \bar{u}_k^k + \widehat{v}_k^m$

3. Post-smoothing

- Compute u_k^{m+1} using post-smoothing

$$u_k^{m+1} = SMOOTH^{\nu_2}(\tilde{u}_k^m, L_k, f_k)$$

Multigrid Iteration Operator

- The multigrid iteration operator is given by the recursion

$$M_0 = 0$$

$$M_k = S_k^{\nu_2}(I_k - I_{k-1}^k(I_{k-1} - (M_{k-1})^\gamma)(L_{k-1})^{-1} I_k^{k-1} L_k) S_k^{\nu_1}, \quad \text{for } k = 1, \cdots, l$$

- Note, compared to the two grid operator

$$(L_{k-1})^{-1} \text{is replaced with } (I_{k-1} - (M_{k-1})^\gamma)(L_{k-1})^{-1}$$

Computational Work

- The computational work W_l per multigrid cycle on Ω_l is given recursively

$$W_1 = W_1^0 + W_0, \qquad W_{k+1} = W_{k+1}^k + \gamma_k W_k \quad , k = 1, \cdots, l-1$$

  - W_{k+1}^k computational work of one (h_{k+1}, h_k) two grid cycle without the work needed to solve the defect equations on Ω_k.
  - W_0 work needed to compute the exact solution on the coarsest grid.

- If γ is independent of k we have

$$W_l = \sum_{k=1}^{l} \gamma^{l-k} W_k^{k-1} + \gamma^{l-1} W_0 \qquad l \geq 1$$

Computational Work

- Example: standard coarsening $N_k = \sharp\Omega_k$

  Assume $W_k^{k-1} \leq CN_k$, then

$$W_l = \begin{cases} \frac{4}{3} CN_l & \text{for } \gamma = 1 \\ 2CN_l & \text{for } \gamma = 2 \\ 4CN_l & \text{for } \gamma = 3 \\ O(N_l \log N_l) & \text{for } \gamma = 4 \end{cases}$$

- The amount of work for $\gamma = 1, 2$, or 3 is proportional to the number of grid points.

  Also, for h-elliptic problems the multigrid convergence is h-independent.

- Note, in general it does not pay off to have large values of ν_1 and ν_2. This does not improve multigrid performance. Common practice is that $\nu = \nu_1 + \nu_2 \leq 3$ or 4.

Full Multigrid

- A good initial approximation for the multigrid iterations can be obtained with nested iteration using coarser meshes.
- The combination of obtaining an initial solution using nested iteration and multigrid is called Full Multigrid.
- Full Multigrid is generally the most efficient multigrid algorithm.

Full Multigrid

- Full multigrid algorithm

  For $k = 0$ Solve $L_0 u_0 = f_0$, providing $u_0^{FMG} = u_0$.

  For $k = 1, \cdots, l$,

$$u_k^0 = \Pi_{k-1}^k u_{k-1}^{FMG}$$

$$u_k^{FMG} = MGCYC^r(k+1, \gamma, u_k^0, L_k, f_k, \nu_1, \nu_2) \qquad (r \text{ steps})$$

- The interpolation order in FMG should be larger than the order of accuracy of the discretization.

  For example: for a second order accurate discretization we can use a cubic interpolation method.

Properties of Full Multigrid

- An approximation u_h^{FMG} of u_h can be computed up to an error that is related to the discretization error

$$\|u_h - u_h^{FMG}\| \cong \|u - u_h\|$$

- It is not useful to compute a FMG error smaller than the discretization error.

- It is not sufficient to compute the initial solution on a coarse mesh and interpolate to a finer mesh.

  The interpolation process generates high frequency errors that need to be removed with smoothing steps.

Transfer Operators

- The order of interpolation is $k + 1$ if an interpolation is exact for polynomials of degree k.

  For instance, bilinear interpolation has order 2.

- The order of the restriction operator is equal to the order of its transposed.

  Bilinear interpolation is the transposed of Full Weighting restriction, which thus has order 2.

- Let m be the order of the operator L in $Lu = f$, m_i the order of the restriction operator and m^j the order of the interpolation operator.

  For good multigrid performance we need

$$m_i + m^j > m.$$

Transfer Operators

- Note, the transposed of pure injection does not even interpolate constant polynomials exactly since it does not give any information back to the fine grid points that are not part of the coarse grid. Its order is thus 0.

  The combination of injection and bilinear interpolation therefore does not satisfy the condition

  $$m_i + m^j > m$$

  for the second order finite difference discretization of the Laplace equation.