FEM and sparse linear system solving

Lecture 11, Dec 1, 2017: Multigrid

http://people.inf.ethz.ch/arbenz/FEM16

Peter Arbenz
Computer Science Department, ETH Zürich
E-mail: arbenz@inf.ethz.ch

Survey on lecture

- The finite element method
- Direct solvers for sparse systems
- Iterative solvers for sparse systems
  - Stationary iterative methods, preconditioning
  - Preconditioned conjugate gradient method (PCG)
  - Krylov space methods for nonsymmetric systems
    GMRES, MINRES
  - Preconditioning
  - Multigrid (preconditioning)
  - Nonsymmetric Lanczos iteration based methods
    Bi-CG, QMR, CGS, BiCGstab

Outline of this lecture

1. Geometric multigrid preconditioning
   - Multigrid restricted to rectangular grid. Here: square grid.
   - Restricted to SPD matrices

Literature

- Y. Saad: *Iterative methods for sparse linear systems* (2nd ed.). SIAM, 2003.
- J. Demmel: *Applied Numerical Linear Algebra*. SIAM, 1997.
- H. Elman, D. Silvester, & A. Wathen. *Finite elements and fast iterative solvers.* Oxford University Press, 2005. Chapter 2.

Preconditioned conjugate gradients

- Given a system of equations

$$A\boldsymbol{x} = \boldsymbol{b}, \qquad A \in \mathbb{R}^{n\times n} \text{ is SPD.} \tag{1}$$

- n is related to mesh width h in FE or FD, $\kappa(A) = \mathcal{O}(1/h^2)$.
- For large systems, we need to precondition (1) to get reasonable iteration counts.
- Simple and popular preconditioners are Jacobi (diagonal), Gauss-Seidel (GS), or IC(0) preconditioners.
- These methods tend to be slow as problem size n increases.
- But, both Jacobi and GS preconditioners are good smoothers: they effectively damp the high-frequency modes of the errors.
- Coarse grid correction takes care about low-frequency modes.

1D Poisson problem

The FE/FD discretization of

$$-u''(x) = f(x), \qquad u(0) = u(1) = 0,$$

leads to a linear system with the system matrix as below. Using the trigonometric identity

$$\sin(j-1)\vartheta + \sin(j+1)\vartheta = 2\sin j\vartheta\cos\vartheta$$

gives

$$\begin{bmatrix} 2 & -1 & & & & & \\ -1 & 2 & -1 & & & & \\ & -1 & 2 & -1 & & & \\ & & -1 & 2 & -1 & & \\ & & & -1 & 2 & -1 & \\ & & & & -1 & 2 & -1 \\ & & & & & -1 & 2 \end{bmatrix} \begin{bmatrix} \sin\vartheta \\ \sin 2\vartheta \\ \sin 3\vartheta \\ \sin 4\vartheta \\ \sin 5\vartheta \\ \sin 6\vartheta \\ \sin 7\vartheta \end{bmatrix} = \underbrace{2(1-\cos\vartheta)}_{4\sin^2\frac{\vartheta}{2}} \begin{bmatrix} \sin\vartheta \\ \sin 2\vartheta \\ \sin 3\vartheta \\ \sin 4\vartheta \\ \sin 5\vartheta \\ \sin 6\vartheta \\ \sin 7\vartheta \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ \sin 8\vartheta \end{bmatrix}$$

1D Poisson problem (cont.)

If ϑ is such that $\sin(n+1)\vartheta = 0$ (here $n = 7$) then we have found an eigenvalue $\lambda = 2(1-\cos\vartheta)$ and a corresponding eigenvector. Clearly,

$$\vartheta_k = \frac{k\pi}{n+1} \Longrightarrow \sin\vartheta_k = 0 \Longrightarrow \lambda_k = 2(1-\cos\vartheta_k) = 4\sin^2\frac{\vartheta_k}{2}$$

The corresponding eigenvectors (of $T_n\boldsymbol{x} = \lambda\boldsymbol{x}$) are

$$\boldsymbol{q}_k = (\sin\vartheta_k, \sin 2\vartheta_k, \ldots, \sin n\vartheta_k)^T.$$

The smallest/largest eigenvalues are

$$\lambda_1 = 4\sin^2\frac{\pi}{2(n+1)} = \mathcal{O}(h^2), \qquad \lambda_n = 4\sin^2\frac{n\pi}{2(n+1)} = 4-\mathcal{O}(h^2) \approx 4.$$

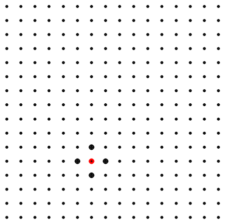
2D Poisson problem

The FE/FD discretization of

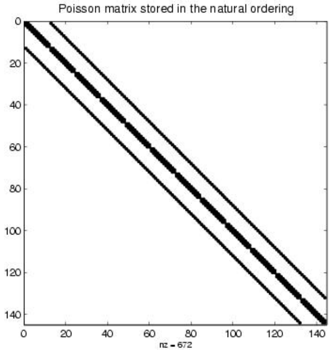
$$-\Delta u(x) = f(x) \text{ in } \Omega = (0,1)^2, \qquad u = 0 \text{ on } \partial\Omega$$

on a grid with n-by-n (interior) gridpoints leads to a matrix of the structure given on the next slide.

16×16 grid
(including boundary points)
$n = 14$
grid width $h = 1/15 = 1/(n+1)$
$\Omega = (0,1)^2 = (0,15h)^2$



2D Poisson problem (cont.)



$$A_{n\times n} = T_n \otimes I_n + I_n \otimes T_n$$

Kronecker product

The discretization in every (interior) grid point is given by

$$4u_{\mathrm{center}} - u_{\mathrm{west}} - u_{\mathrm{south}} - u_{\mathrm{east}} - u_{\mathrm{north}} = h^2 \cdot f_{\mathrm{center}}$$

2D Poisson problem (cont.)

The eigenvalues and eigenvectors are given by

$$\lambda_{k,\ell} = \lambda_k^{(1D)} + \lambda_\ell^{(1D)} = 4\left(\sin^2\frac{\vartheta_k}{2} + \sin^2\frac{\vartheta_\ell}{2}\right), \qquad 1 \le k,\ell \le n.$$

The corresponding eigenvectors are obtained by a tensor product of the 1D-eigenvectors,

$$\boldsymbol{x}_{k,\ell} = \mathrm{Vec}\left(\boldsymbol{x}_k^{(1D)}\left(\boldsymbol{x}_\ell^{(1D)}\right)^T\right), \qquad 1 \le k,\ell \le n.$$

Remark: Vec makes a vector from a matrix by stacking column on top of each other.
In MATLAB this is obtained by the colon operator: `a = A(:);`

Damped Jacobi iteration

Damped Jacobi iteration is given by

$$\boldsymbol{x}_{k+1} = \boldsymbol{x}_k + \omega D^{-1}\boldsymbol{r}_k = \boldsymbol{x}_k + \omega D^{-1}(\boldsymbol{b} - A\boldsymbol{x}_k)$$

where $D = \mathrm{diag}(A)$. So far we considered $\omega = 1$.
The eigenvalues μ_k of the iteration matrix $I - \omega D^{-1}A$ for the 1D Poisson matrix are ($D = 2I$)

$$\mu_k = 1 - \frac{\omega}{2}\lambda_k \ = 1 - 2\omega \sin^2 \frac{\vartheta_k}{2}, \quad 1 \le k \le n,$$

One sees that we must have $0 < \omega \le 1$ to have convergence at all.
If we want a maximal reduction of the high-order modes $\boldsymbol{q}_k$, $k = \frac{n}{2}, \ldots, n$, then we choose $\mu_{\frac{n}{2}} = -\mu_n$ whence $\omega = 2/3$.

Damped Jacobi iteration (cont.)

In the 2D case, the eigenvalues $\mu_{k,\ell}$ of the iteration matrix $I - \omega D^{-1} A$ are ($D = 4I$)

$$\mu_{k,\ell} = 1 - \frac{\omega}{4}\lambda_{k,\ell} \;= 1 - \omega\left(\sin^2\frac{\vartheta_k}{2} + \sin^2\frac{\vartheta_\ell}{2}\right), \quad 1 \le k,\ell \le n.$$

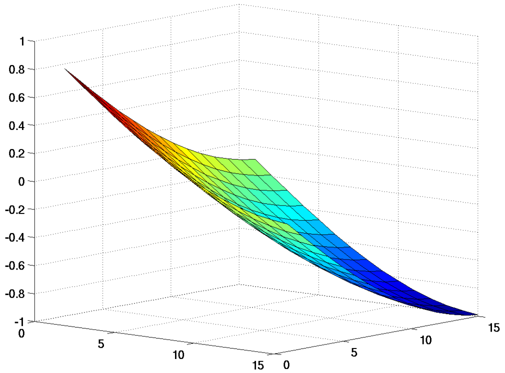
Again, $0 < \omega \le 1$ is required to have convergence.

Here, the high-order modes are those with eigenvalue $\lambda_{k,\ell}$ with $k \ge \frac{n}{2}$ or $\ell \ge \frac{n}{2}$.

Therefore, we require that $\mu_{\frac{n}{2},0} (= \mu_{0,\frac{n}{2}}) = |\mu_{n,n}| = -\mu_{n,n}$ or $1 - \frac{\omega}{2} = -(1 - 2\omega)$ whence $\omega = 4/5$.

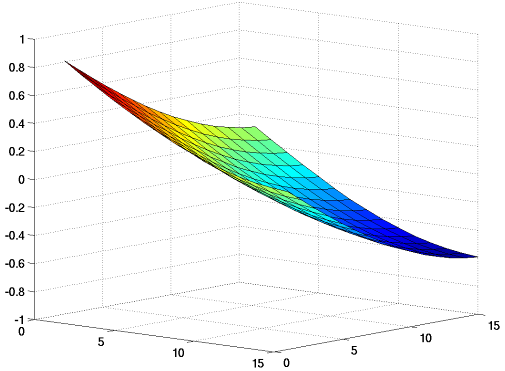
Note: In 3D we request that $\mu_{\frac{n}{2},0,0} = -\mu_{n,n,n}$. Thus $\omega = 6/7$.

2D case: plot of $\mu_{k,\ell}$ for $\omega = 1$



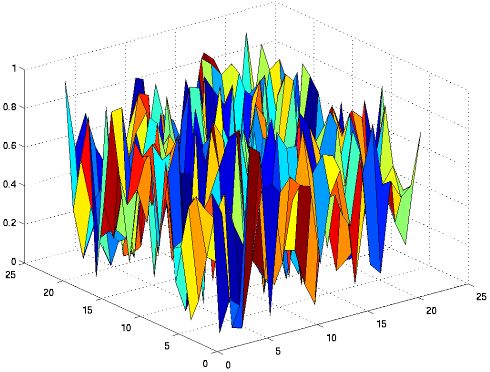
The eigenvalues are between ± 1. The eigenvalues closest to 1 (in modulus) correspond to very smooth ($k \approx \ell \approx 0$) and very "rough" ($k \approx \ell \approx n$) eigenfunctions. (Here, $n = 15$.)

2D case: plot of $\mu_{k,\ell}$ for $\omega = 4/5$



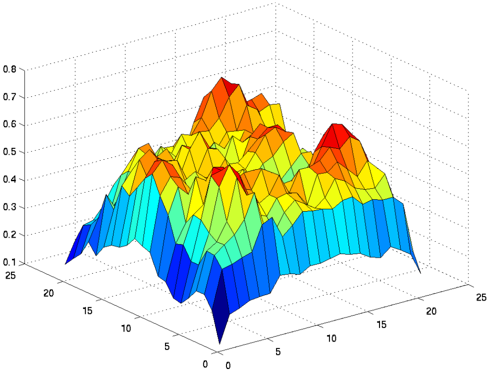
Eigenvalues closest to 1 (in modulus) correspond to very smooth ($k \approx \ell \approx 0$). The rough eigenvalues are around $1 - 2\omega = -3/5$. In fact, $|\mu_{k,\ell}| \leq 3/5$ for all $k \geq n/2$ or $\ell \geq n/2$

Illustration of smoothing with symmetric Gauss–Seidel

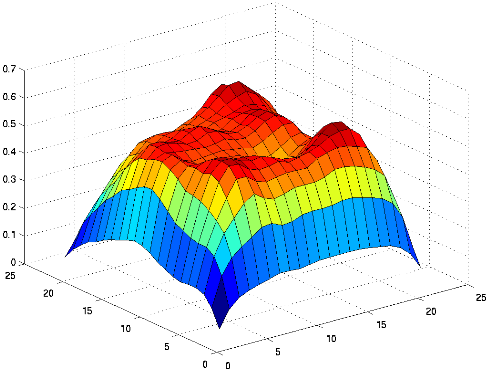


- 2D Poisson equation, 21×21 mesh. Random initial condition.

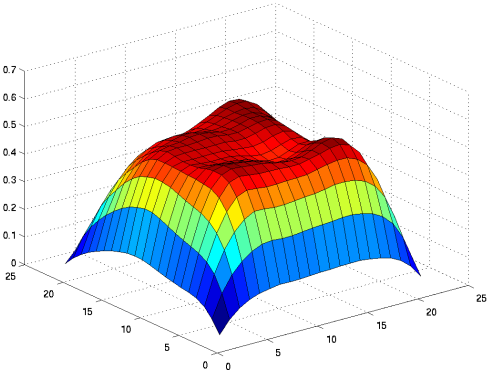
Sym. Gauss–Seidel: error after 1 step



Sym. Gauss–Seidel: error after 2 steps



Sym. Gauss–Seidel: error after 3 steps



- Notice slow overall convergence!
- Can represent smoother error on *coarser* grid $\Longrightarrow$ multigrid.

Two-grid idea

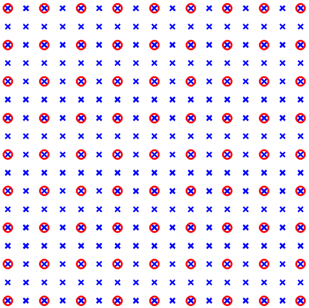
- From convergence analysis for stationary iterative solvers we know that error and residual are reduced similarly,

$$\mathbf{e}_{k+1} = (I - M^{-1}A)\mathbf{e}_k, \qquad \mathbf{r}_{k+1} = (I - AM^{-1})\mathbf{r}_k.$$

- A well-designed smoother reduces high-frequency components of errors/residuals.
- We try to reduce the smooth low-frequency error components by means of a coarse grid.

Two-grids

We stick with our square $n \times n$ grid. We assume n to be odd and set $N + 1 = (n + 1)/2$.



Here $n = 15$ and $N = 7$.
(We do not count the grid points on the boundary.
We denote the fine grid by Ω_h and the coarse grid by Ω_H.

Note that here we also display boundary points.

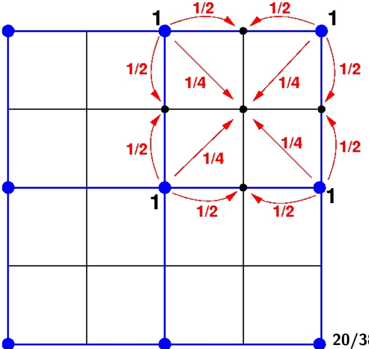
Prolongation

- The prolongation takes a vector from Ω_H and defines an analogous vector on Ω_h,

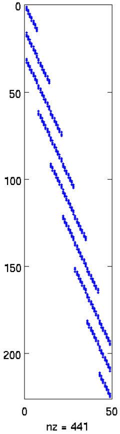
$$I_H^h : \Omega_H \longrightarrow \Omega_h.$$

- The simplest way to define a prolongation operator is by linear interpolation

  The values at those fine grid points that are also coarse grid points are taken over from the coarse grid points.



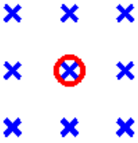
The corresponding matrix is $n \times N$.



A typical column of I_H^h has this structure

$$\frac{1}{4}\begin{pmatrix} \vdots \\ 1 \\ 2 \\ 1 \\ \vdots \\ 2 \\ 4 \\ 2 \\ \vdots \\ 1 \\ 2 \\ 1 \\ \vdots \end{pmatrix}$$

Compare with



$$\frac{1}{4}\begin{pmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{pmatrix}$$

$$= \frac{1}{4}\begin{pmatrix} 1 \\ 2 \\ 1 \end{pmatrix} (1,\ 2,\ 1)$$

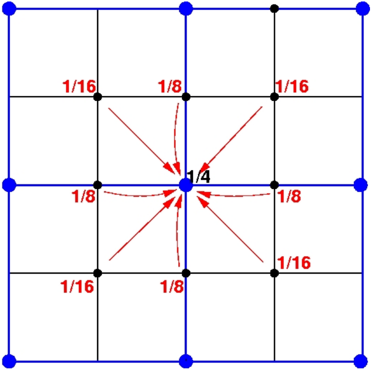
Restriction

- The restriction operation is the reverse of the prolongation. It takes a vector from the fine grid Ω_h and defines a vector on the coarse grid Ω_h.
- The *injection operator* is the simplest variant,

$$v_{i,j}^{2h} = v_{2i,2j}^{h}.$$

- Another common restriction operator, called full weighting (FW), defines $v_{i,j}^{2h}$ to be a weighted average of all neighboring points.

Restriction (cont.)



With these definitions the restriction becomes $$I_h^H = \frac{1}{4}\left(I_H^h\right)^T.$$

Coarse grid problem

- At the highest level, i.e., on the finest grid, a mesh size h is used and the problem to solve has the form

$$A_h \boldsymbol{x}_h = \boldsymbol{f}_h.$$

- One of the requirements of MG techniques is that a system similar to the one above should be solved on the coarser levels.
- One may discretize the same, e.g. PDE, on the coarser grid.
- An alternative is to directly define the coarse linear system by a *Galerkin projection*, where the coarse problem is defined by

$$A_H = I_h^H A_h I_H^h, \qquad \boldsymbol{f}_H = I_h^H \boldsymbol{f}_h.$$

Two-grid cycle

$\boldsymbol{x}^h = \text{2-grid cycle}(A_h, \boldsymbol{x}_0^h, \boldsymbol{f}^h)$

1. Presmooth:	$\boldsymbol{x}^h := \texttt{smooth}^{\nu_1}(A_h, \boldsymbol{x}_0^h, \boldsymbol{f}^h)$
2. Get residual:	$\boldsymbol{r}^h = \boldsymbol{f}^h - A_h\boldsymbol{x}^h$
3. Coarsen:	$\boldsymbol{r}^H = I_h^H \boldsymbol{r}^h$
4. Solve:	$A_H \boldsymbol{d}^H = \boldsymbol{r}^H$
5. Correct:	$\boldsymbol{x}^h = \boldsymbol{x}^h + I_H^h \boldsymbol{d}^H$
6. Postsmooth:	$\boldsymbol{x}^h := \texttt{smooth}^{\nu_2}(A_h, \boldsymbol{x}^h, \boldsymbol{f}^h)$
7. Return $\boldsymbol{x}^h$	

This two-grid cycle can be written in the form

$$\boldsymbol{x}_{\text{new}}^h = M_h \boldsymbol{x}_0^h + \boldsymbol{g}_{M_h}.$$

What is the iteration matrix M_h of the two-grid cycle?
(We do not care about $\boldsymbol{g}_{M_h}$.)

Let us first look at smoothing, that we write as

$$\boldsymbol{x}_\nu^h = \mathtt{smooth}^\nu(A_h, \boldsymbol{x}_0^h, \boldsymbol{f}^h).$$

One step of the ν (stationary) iterations has the form

$$\begin{aligned}\boldsymbol{x}_{j+1}^h &= \boldsymbol{x}_j^h + B_h(\boldsymbol{f}^h - A_h\boldsymbol{x}_j^h) = \boldsymbol{x}_j^h - B_hA_h\boldsymbol{x}_j^h + B_h\boldsymbol{f}^h \\ &= \underbrace{(I - B_hA_h)}_{S_h}\boldsymbol{x}_j^h + \underbrace{B_h\boldsymbol{f}^h}_{\boldsymbol{g}^h}, \qquad B_h = (I - S_h)A_h^{-1}.\end{aligned}$$

The effect of ν smoothing steps on the *error* is

$$\boldsymbol{d}_{j+1}^h = S_h^\nu\ \boldsymbol{d}_0^h$$

Trick: We get the iteration matrix S_h if we set $\boldsymbol{f}^h = \mathbf{0}$.

We apply the same trick to the two-grid cycle to get

$$M_h = S_h^{\nu_2}[I - I_H^h A_H^{-1} I_h^H A_h] S_h^{\nu_1} \equiv S_h^{\nu_2} T_h^H S_h^{\nu_1}$$

The matrix in brackets,

$$T_h^H = I - I_H^h A_H^{-1} I_h^H A_h,$$

is called coarse grid correction.

Remark: Evidently, $T_h^H I_H^h = O$

T_h^H is A_h-orthogonal projector on $\mathcal{R}(I_H^h)^\perp$

For an analysis of a multigrid method we have to investigate
(1) how the smooth error components are suppressed by T_h^H, and
(2) how the 'rough' error components are smoothed by S_h.

Two-grid example: 1D Poisson equation

```
n=33; N=(n-1)/2;

A=(p_1d(n));  I=eye(n);  D=diag(diag(A));
AC=(p_1d(N))/4;

R=abs(A(:,2:2:n-1));
%J=[2:2:n-1]';
% R=R(:,J);  P=R;
P=R/2;  R=R'/4;       % Prolongation, restriction

omega=2/3;
GS=(I-omega*(D\A));   % Iteration matrix for smoother
GCG=I-P*(AC\(R*A));   % Iteration matrix for
                      %     coarse grid correction
G2G=GS*GCG*GS;        % Iteration matrix for complete
                      %     2-grid preconditioning
```

Recursion $\longrightarrow$ Multigrid: Algorithm V-cycle

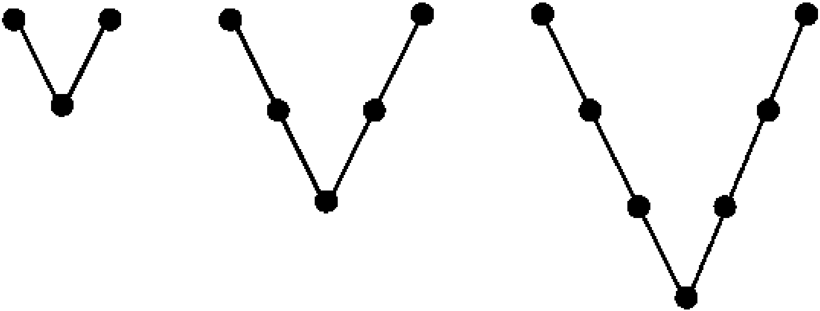
$\boldsymbol{x}^h = \text{V-Cycle}(A_h, \boldsymbol{x}_0^h, \boldsymbol{f}^h)$

1. Presmooth:	$\boldsymbol{x}^h := \texttt{smooth}^{\nu_1}(A_h, \boldsymbol{x}_0^h, \boldsymbol{f}^h)$
2. Get residual:	$\boldsymbol{r}^h = \boldsymbol{f}^h - A_h \boldsymbol{x}^h$
3. Coarsen:	$\boldsymbol{r}^H = I_h^H \boldsymbol{r}^h$
4. If $(H == h_0)$	
5. Solve:	$A_H \boldsymbol{d}^H = \boldsymbol{r}^H$
6. Else	
7. Recursion:	$\boldsymbol{d}^H = \text{V-Cycle}(A_H, \boldsymbol{0}, \boldsymbol{r}^H)$
8. Endif	
9. Correct:	$\boldsymbol{x}^h = \boldsymbol{x}^h + I_H^h \boldsymbol{d}^H$
10. Postsmooth:	$\boldsymbol{x}^h := \texttt{smooth}^{\nu_2}(A_h, \boldsymbol{x}^h, \boldsymbol{f}^h)$
11. Return $\boldsymbol{x}^h$	

Multigrid idea

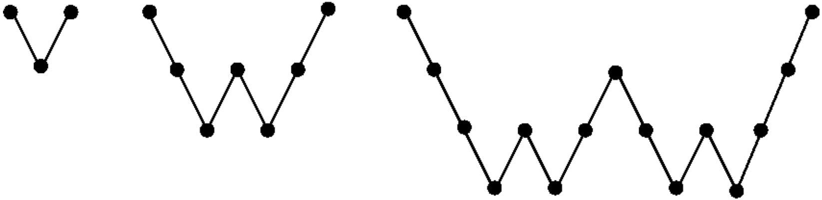
1. Smooth error on finest grid
   Highly oscillating error components (upper half of spectrum) are strongly damped.
2. Project smoothed error on next coarser grid.
   slowly oscillating error components (lower half of spectrum) are now treated.
3. In the recursive approach: these error components are split in two sets again:
   3.1 Highly oscillating error components (on this level)
   Smooth these.
   3.2 Slowly oscillating error components (on this level)
   Correct these on an again coarser grid.
4. Postsmooth after coarse grid correction.

V-cycles of varying depth (# of levels)



W-cycles of varying depth (# of levels)

In contrast to V-cycles, in W-cycles the coarse-grid correction is called twice in a row. Step 7 of the V-cycle algorithm is executed twice.



Some analysis

Let us assume that the following two properties are satisfied:

Smoothing property:

$$\|S_h \boldsymbol{e}^h\|_{A_h}^2 \leq \|\boldsymbol{e}^h\|_{A_h}^2 - \alpha \|A_h \boldsymbol{e}^h\|_{D^{-1}}^2 \quad \forall \boldsymbol{e}^h \in \Omega_h.$$

Approximation property:

$$\min_{\boldsymbol{e}_H \in \Omega_H} \|\boldsymbol{e}^h - I_H^h \boldsymbol{e}^H\|_D^2 \leq \beta \|\boldsymbol{e}^h\|_{A_h}^2$$

where both $\alpha > 0$ and $\beta > 0$ do not depend on the mesh size h.

The smoothing property means that high frequency errors are dampened much. ($\|A_h \boldsymbol{e}^h\| \approx 0$ for smooth errors)
The approximation property means that smooth errors are approximated well by the coarse grid space.

Some analysis (cont.)

Theorem : Let the smoothing and approximation properties be satisfied with $\alpha > 0$ and $\beta > 0$. Then $\alpha \leq \beta$, the two-level iteration converges, and the norm of the iteration matrix is bounded by

$$\|S_h T_h^H\|_{A_h} \leq \sqrt{1 - \frac{\alpha}{\beta}}.$$

Proof: See Saad: *Iterative methods for sparse linear systems* (2nd ed.), SIAM, 2003, p. 436.

Jacobi example

- In Jacobi smoothing we have $S(\omega) = I - \omega D^{-1}A$.
- We have convergence if $0 < \omega < 2/\rho(D^{-1}A)$.
- Now, (index h is omitted here)

$$
\begin{aligned}
\|S(\omega)\mathbf{e}\|_A^2 &= (A(I-\omega D^{-1}A)\mathbf{e}, (I-\omega D^{-1}A)\mathbf{e}) \\
&= (A\mathbf{e},\mathbf{e}) - 2\omega(AD^{-1}A\mathbf{e},\mathbf{e}) + \omega^2(AD^{-1}A\mathbf{e}, D^{-1}A\mathbf{e}) \\
&= (A\mathbf{e},\mathbf{e}) - 2\omega(D^{-1/2}A\mathbf{e}, D^{-1/2}A\mathbf{e}) \\
&\qquad + \omega^2((D^{-1/2}AD^{-1/2})D^{-1/2}A\mathbf{e}, D^{-1/2}A\mathbf{e}) \\
&= (A\mathbf{e},\mathbf{e}) - ([\omega(2I-\omega D^{-1/2}AD^{-1/2})]D^{-1/2}A\mathbf{e}, D^{-1/2}A\mathbf{e}) \\
&\le \|\mathbf{e}\|_A^2 - \lambda_{\min}[\omega(2I-\omega D^{-1/2}AD^{-1/2})]\|A\mathbf{e}\|_{D^{-1}}.
\end{aligned}
$$

- Let $\rho = \rho(D^{-1/2}AD^{-1/2}) = \rho(D^{-1}A)$. Then $2-\omega\rho > 0$.
- So, we can set $\alpha = \omega(2-\omega\rho)$.

Full weighting example

In 1D, the n-by-n Poisson matrix is the (-1,2,-1) - matrix on Slide 5 with eigenvectors

$$\boldsymbol{x}_k = (\sin\vartheta_k, \sin 2\vartheta_k, \ldots, \sin n\vartheta_k)^T, \quad \vartheta_k = \frac{k\pi}{n+1} \quad k = 1, \ldots, n.$$

To estimate β in the approximation property

$$\min_{\boldsymbol{e}_H \in \Omega_H} \|\boldsymbol{e}^h - I_H^h \boldsymbol{e}^H\|_D^2 \le \beta \|\boldsymbol{e}^h\|_{A_h}^2$$

We set $\boldsymbol{e}^h = \boldsymbol{x}_k$ and $\boldsymbol{e}^H$ is the vector that interpolates $\boldsymbol{e}^h$ at all even-numbered nodes. At the odd-numbered nodes the difference is

$$\begin{aligned} e_j^h - \frac{1}{2}(e_{j-1}^h + e_{j+1}^h) &= \sin j\vartheta_k - \frac{1}{2}(\sin(j-1)\vartheta_k + \sin(j+1)\vartheta_k) \\ &= (1 - \cos\vartheta_k)\sin j\vartheta_k = 2\sin^2\frac{\vartheta_k}{2}\sin j\vartheta_k \end{aligned}$$

It is an exercise to show that $\beta \le 1/2$.

Numerical example

We solve $-\Delta u = f$ with homogeneous boundary conditions on the square by the Finite Difference method on a $m \times m$ grid, $m = 31$, $m = 101$.

First, we solve with PCG where the preconditioner are Jacobi, block-Jacobi, symmetric Gauss-Seidel and IC(0).

	Jacobi	Block Jacobi	Sym. GS	ICCG(0)
$m = 31$	0.084 (76)	0.071 (57)	0.050 (33)	0.042 (28)
$m = 101$	0.99 (234)	0.86 (166)	0.47 (84)	0.50 (73)

Execution times (iteration count)

Numerical example (cont.)

Now we solve with PCG where the preconditioner is a two-grid solver. The smoother is either Jacobi or Gauss-Seidel. The smoothing parameter of Jacobi is $\omega = 1$ and $\omega = 4/5$. We also tried two ($\nu_1 = \nu_2 = 2$) steps of Jacobi(4/5) as the smoother.

	Jacobi(1)	Jacobi(4/5)	2×Jacobi(4/5)	GS
$m = 31$	0.036 (33)	0.0088 (7)	0.0077 (5)	0.0075 (5)
$m = 101$	0.91 (63)	0.10 (7)	0.083 (5)	0.083 (5)

Execution times (iteration count)