Numerical Analysis - Part II

Anders C. Hansen

Lecture 24

Multigrid methods

Multigrid methods

The speed of convergence of some iterative methods (Jacobi with relaxation, Gauss–Seidel, etc.) can be increased drastically when the linear system originates in the discretization of PDEs, using *multigrid methods*.
Here we look at the system $A\boldsymbol{u} = \boldsymbol{b}$ originating from the 3-point formula for the Poisson equation on an m-grid $\Omega_h = \{ih : 1 \leq i \leq m\}$, $h = 1/(m+1)$, being solved by the weighted Jacobi iteration.
Recall that the matrix A in this case is given by

$$A = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & & \ddots & & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} \in \mathbb{R}^{m \times m}.$$

Multigrid methods – Eigenvectors and eigenvalues

The diagonal part of A is $D = 2I$. Thus the weighted Jacobi iterations takes the form:

$$\boldsymbol{u}^{(\nu+1)} = H_\omega \boldsymbol{u}^{(\nu)} + (\omega/2)\boldsymbol{b}$$

where $\nu = 0, 1, \ldots$ is the iteration count, $H = I - D^{-1}A = I - \frac{1}{2}A$, and $H_\omega = \omega H + (1-\omega)I = I - \frac{\omega}{2}A$. The error decay is expressed in terms of the iteration matrix H_ω:

$$\boldsymbol{e}^{(\nu)} = H_\omega^\nu \boldsymbol{e}^{(0)}.$$

We know from the results of Lecture 2 that the eigenvectors and the eigenvalues of H_ω are

$$\boldsymbol{w}^k = \left[\sin i\frac{k\pi}{m+1}\right]_{i=1,\ldots,m}, \qquad \lambda_k(\omega) = 1 - 2\omega \sin^2 \frac{k\pi}{2(m+1)} \qquad (k = 1, \ldots, m).$$

Multigrid methods – Eigenvectors and eigenvalues

Consider the choice $\omega = 1/2$; then the eigenvalues of H_ω are $\lambda_k = 1 - \sin^2 \frac{k\pi}{2(m+1)} = \cos^2 \frac{k\pi}{2(m+1)}$. With this choice, the eigenvalues are all positive and decreasing with k, see Figure below.

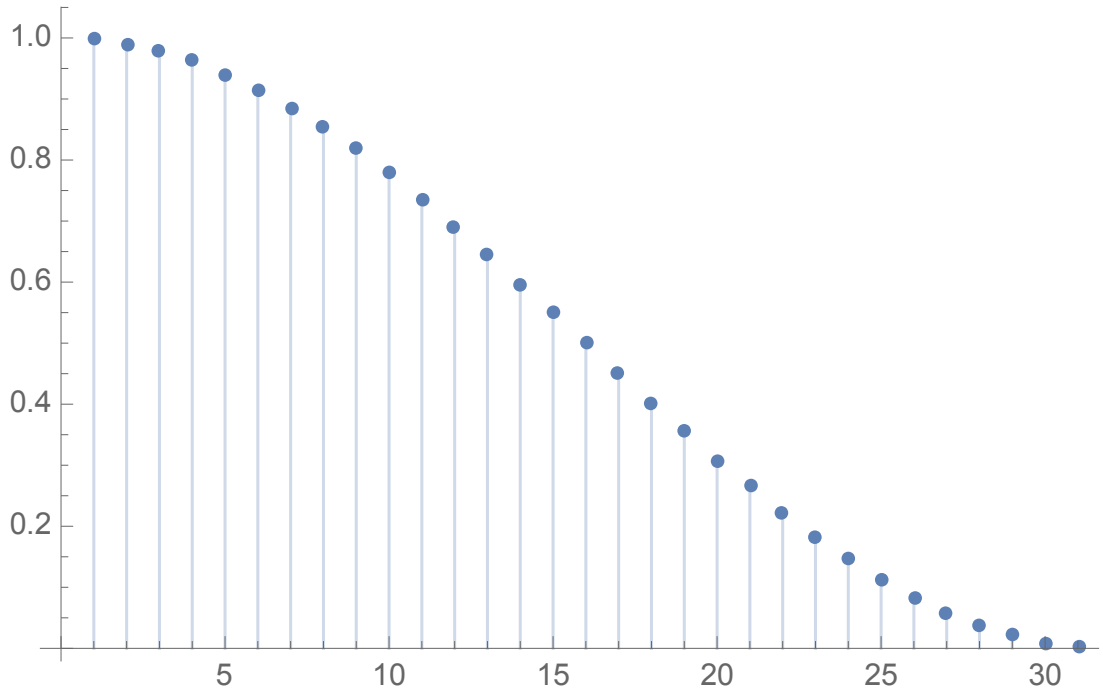


Figure: Eigenvalues of H_ω for $\omega = 1/2$ ($m = 31$).

In particular $\rho(H_\omega) = \lambda_1 = \cos^2 \frac{\pi}{2(m+1)} \approx 1 - \frac{\pi^2}{4m^2} < 1$, guaranteeing convergence, although a very slow one when m is large!

However, expanding the error with respect to the (orthogonal) eigenvectors we obtain

$$\boldsymbol{e}^{(\nu)} = \sum_{k=1}^{m} a_k^{(\nu)} \boldsymbol{w}^k, \qquad \boldsymbol{e}^{(\nu)} = H_\omega^\nu \boldsymbol{e}^{(0)} \quad \Rightarrow \quad |a_k^{(\nu)}| = |\lambda_k|^\nu |a_k^{(0)}|\,,$$

i.e. the components of $\boldsymbol{e}^{(\nu)}$ (with respect to the basis of eigenvectors) decay at a different rate for different frequencies $k = 1, \ldots, m$. More precisely, the high frequencies, where k is close to m, will decay faster than the low frequencies, where k is closer to 1.

Multigrid methods – Eigenvectors and eigenvalues

Let us say that $k \in (0, m+1) = (0, \frac{1}{h})$ is *high frequency* (HF) with respect to the grid Ω_h if $kh \geq 1/2$ (i.e., $\frac{m+1}{2} \leq k \leq m$). Then the decay rate for the high frequency components of the error $\boldsymbol{e}$ is at least:

$$\mu_* = |\lambda_{(m+1)/2}| = 1 - \sin^2(\pi/4) = 1/2.$$

Therefore, for the coefficients at the HF components of $\boldsymbol{e}^{(\nu)}$ we obtain

$$|a_k^{(\nu)}| \leq |\mu_*|^\nu |a_k^{(0)}| = \left(\frac{1}{2}\right)^\nu |a_k^{(0)}| \ll |a_k^{(0)}|\,,$$

i.e. the Jacobi method converges fast for high frequencies.

Multigrid methods – Eigenvectors and eigenvalues

The main observation of the multigrid is to note that the low frequencies $k \in (\frac{1}{4h}, \frac{1}{2h})$ with respect to the grid Ω_h become high frequencies for the *coarser grid* Ω_{2h} with step $2h$; indeed for such k we have $k(2h) \geq 1/2$.

Changing the grid

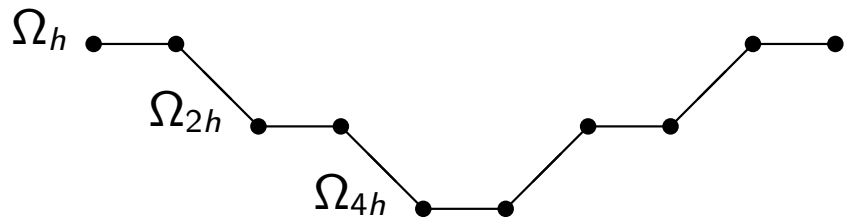
The idea of the multigrid method then is that, although the global error may decrease slowly by iteration, its components with high frequencies relative to Ω_h are suppressed very quickly, and that dealing with the remaining components (with low frequencies relative to Ω_h) we can move to the coarse grid Ω_{2h}, where these components (in part) would be of high frequencies, and thus they can be suppressed in a similar way. Therefore, we cover the domain $[0, 1]$ by a range of nested grids, of increasing coarseness, say,

$$\Omega_h \subset \Omega_{2h} \subset \Omega_{4h} \subset \cdots \subset \Omega_{2^j h}\,.$$

At every Ω_{h_i}, the iterations (Jacobi, or Gauss-Seidel) remove the high frequencies relative to this grid, and we move to Ω_{2h_i}. On the coarsest grid, where the number of variables is small, we can afford to solve the equations with a direct method, by Cholesky, say.

Remark 1

If we follow the recursive procedure outlined below, then we see that the algorithm starts at the finest grid, travels to the coarsest (where we apply a direct solver), and back to the finest:



For this reason, the algorithm above describes what is known as a V-cycle, hence the name of the routine.

Multigrid methods – The procedure

A typical multigrid method can be summarized by the following routine **MGV**, which gives an approximate solution to the linear system $Ax = b$, starting from the initial guess $\boldsymbol{u}^0$. We assume below that the size of the linear system is $m = 1/h - 1 = 2^\ell - 1$ for some integer ℓ.

MGV$(A, b, \boldsymbol{u}^0)$

1. If size of A is small enough, use a direct method to solve $Ax = b$ and exit. Else:
2. Presmoothing: Perform a small number (typically ≤ 5) of Jacobi or Gauss-Seidel iterations on $A\boldsymbol{u} = \boldsymbol{b}$ starting from $\boldsymbol{u}^0$.
3. Let $\boldsymbol{r} = \boldsymbol{b} - A\boldsymbol{u}$ be the residual, with $\boldsymbol{u}$ from the previous step.
4. Let $I_{2h}^h : \mathbb{R}^{\frac{m+1}{2}-1} \to \mathbb{R}^m$ be an *interpolation operator* that interpolates vectors on the coarse grid Ω_{2h} to vectors on the fine grid Ω_h; and let $R_h^{2h} : \mathbb{R}^m \to \mathbb{R}^{(m+1)/2-1}$ be a *restriction operator* that restricts vectors on the fine grid Ω_h to vectors on the coarse grid Ω_{2h}.
5. Let $\tilde{A} = R_h^{2h} A I_{2h}^h$ which is of size $\approx m/2 \times m/2$.
6. Let $\tilde{\boldsymbol{\delta}} = \textbf{MGV}(\tilde{A}, R_h^{2h}\boldsymbol{r}, 0)$ (approximate solution to the residual equation $A\boldsymbol{\delta} = \boldsymbol{r}$ on the coarse grid)
7. Let $\boldsymbol{u} = \boldsymbol{u} + I_{2h}^h \tilde{\boldsymbol{\delta}}$
8. Postsmoothing: apply a few Jacobi or Gauss-Seidel iterations starting from $\boldsymbol{u}$ on $Ax = \boldsymbol{b}$
9. Return $\boldsymbol{u}$

`https://en.wikipedia.org/wiki/Multigrid_method`

Interpolation

To make the algorithm above complete, one needs to define the interpolation and restriction operators. A common choice for the interpolation operator is linear interpolation, i.e.,

$$(I_{2h}^h \boldsymbol{v})_{2i} = \boldsymbol{v}_i \quad \text{and} \quad (I_{2h}^h \boldsymbol{v})_{2i+1} = (\boldsymbol{v}_i + \boldsymbol{v}_{i+1})/2$$

for $i \in [0, \frac{1}{2h})$. Assuming zero boundary conditions, this takes the following matrix form:

$$I_{2h}^h = 1/2 \begin{bmatrix} 1 & & & \\ 2 & & & \\ 1 & 1 & & \\ & 2 & & \\ & 1 & 1 & \\ & & \vdots & \\ & & \vdots & \\ & & 1 & 1 \\ & & & 2 \\ & & & 1 \end{bmatrix}.$$

Restriction

For the restriction operator R_h^{2h}, we take the following averaging operator

$$(R_h^{2h}\boldsymbol{v})_i = \frac{1}{4}(\boldsymbol{v}_{2i-1} + 2\boldsymbol{v}_{2i} + \boldsymbol{v}_{2i+1}) \qquad i \in (0, \tfrac{1}{2h})$$

which corresponds, up to scaling, to the transpose of the linear interpolation operator above; namely $R_h^{2h} = \frac{1}{2}(I_{2h}^h)^T$.
In practice, one may need to apply the routine **MGV** multiple times to improve the accuracy, each time starting from the previously obtained solution.
`https://www.damtp.cam.ac.uk/user/hf323/M21-II-NA/demos/multigrid/multigrid.html`