6.3 Multigrid Methods

The Jacobi and Gauss-Seidel iterations produce smooth errors. The error vector e has its high frequencies nearly removed in a few iterations. But low frequencies are reduced very slowly. Convergence requires $O(N^2)$ iterations—which can be unacceptable. The extremely effective **multigrid idea** is to change to a coarser grid, on which "smooth becomes rough" and low frequencies act like higher frequencies.

On that coarser grid a big piece of the error is removable. *We iterate only a few times before changing from fine to coarse and coarse to fine.* The remarkable result is that multigrid can solve many sparse and realistic systems to high accuracy in a **fixed number of iterations**, not growing with n.

Multigrid is especially successful for symmetric systems. The key new ingredients are the (rectangular!) matrices R and I that change grids:

1. A ***restriction matrix R*** transfers vectors from the fine grid to the coarse grid.

2. The return step to the fine grid is by an ***interpolation matrix*** $\boldsymbol{I = I_{2h}^h}$.

3. The original matrix A_h on the fine grid is approximated by $\boldsymbol{A_{2h} = RA_hI}$ on the coarse grid. You will see how this A_{2h} is smaller and easier and faster than A_h. I will start with interpolation (a 7 by 3 matrix I that takes 3 v's to 7 u's):

Interpolation $\boldsymbol{Iv = u}$
$\boldsymbol{u}$ on the fine ($\boldsymbol{h}$) grid from
$\boldsymbol{v}$ on the coarse ($\boldsymbol{2h}$) grid
values are the $\boldsymbol{u}$'s.

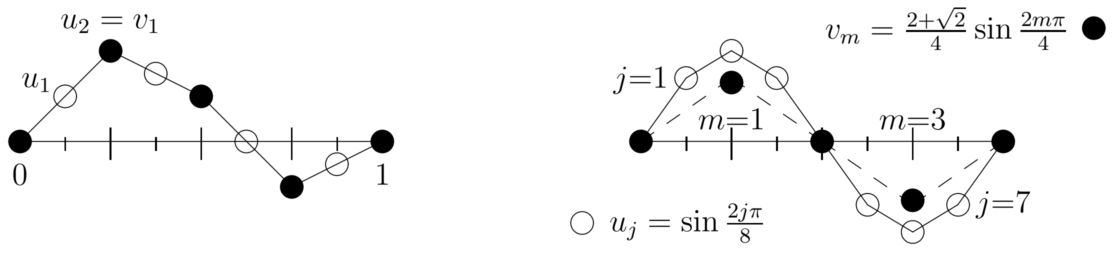
$$\frac{1}{2}\begin{bmatrix}1 & & \\ 2 & & \\ 1 & 1 & \\ & 2 & \\ & 1 & 1 \\ & & 2 \\ & & 1\end{bmatrix}\begin{bmatrix}v_1\\ v_2\\ v_3\end{bmatrix}=\begin{bmatrix}v_1/2\\ v_1\\ v_1/2+v_2/2\\ v_2\\ v_2/2+v_3/2\\ v_3\\ v_3/2\end{bmatrix}=\begin{bmatrix}u_1\\ u_2\\ u_3\\ u_4\\ u_5\\ u_6\\ u_7\end{bmatrix} \tag{1}$$

This example has $h = \frac{1}{8}$ on the interval $0 \le x \le 1$ with zero boundary conditions. The seven interior values are the u's. The grid with $2h = \frac{1}{4}$ has three interior v's.

Notice that u_2, u_4, u_6 from rows 2, 4, 6 are the same as v_1, v_2, v_3! Those coarse grid values v_j are just moved to the fine grid at the points $x = \frac{1}{4}, \frac{2}{4}, \frac{3}{4}$. The in-between values u_1, u_3, u_5, u_7 on the fine grid are coming from *linear interpolation* between $0, v_1, v_2, v_3, 0$:

Linear interpolation in rows 1, 3, 5, 7
$$u_{2j+1} = \frac{1}{2}(v_j + v_{j+1})\,. \tag{2}$$

The odd-numbered rows of the interpolation matrix have entries $\frac{1}{2}$ and $\frac{1}{2}$. We almost always use grid spacings $h, 2h, 4h, \ldots$ with the convenient ratio 2. Other matrices I are possible, but linear interpolation is easy and effective. Figure 6.10a shows the new values u_{2j+1} (open circles) between the transferred values $u_{2j} = v_j$ (solid circles).



$u_2 = v_1$
u_1
0
1
$v_m = \frac{2+\sqrt{2}}{4}\sin\frac{2m\pi}{4}$ ●
j=1
m=1
m=3
j=7
○ $u_j = \sin\frac{2j\pi}{8}$

(a) Linear interpolation by $u = I_{2h}^h v$ (b) Restriction by $R_h^{2h} u = \frac{1}{2}(I_{2h}^h)^{\mathrm{T}} u$

Figure 6.10: Interpolation to the h grid (7 u's). Restriction to the $2h$ grid (3 v's).

When the v's represent smooth errors on the coarse grid (because Jacobi or Gauss-Seidel has been applied on that grid), interpolation gives a good approximation to the errors on the fine grid. A practical code can use 8 or 10 grids.

The second matrix we need is a **restriction matrix R_h^{2h}**. It transfers u on a fine grid to v on a coarse grid. One possibility is the one-zero "injection matrix" that simply copies v from the values of u at the same points on the fine grid. This ignores the odd-numbered fine grid values u_{2j+1}. Another possibility (which we adopt) is the **full weighting operator R** that comes from transposing I_{2h}^h.

Fine grid h to coarse grid 2h by a restriction matrix $R_h^{2h} = \frac{1}{2}(I_{2h}^h)^{\mathrm{T}}$

Full weighting $Ru = v$
Fine grid u to coarse grid v

$$\frac{1}{4}\begin{bmatrix} 1 & 2 & 1 & & & & \\ & & 1 & 2 & 1 & & \\ & & & & 1 & 2 & 1 \end{bmatrix}\begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \\ u_7 \end{bmatrix} = \begin{bmatrix} v_1 \\ v_2 \\ v_3 \end{bmatrix}. \qquad (3)$$

The effect of this restriction matrix is shown in Figure 6.10b. We intentionally chose the special case in which $u_j = \sin(2j\pi/8)$ on the fine grid (*open circles*). Then v on the coarse grid (*dark circles*) is also a pure sine vector. *But the frequency is doubled* (a full cycle in 4 steps). So a smooth oscillation on the fine grid becomes "half as smooth" on the coarse grid, which is the effect we wanted.

Interpolation and Restriction in Two Dimensions

Coarse grid to fine grid in two dimensions from bilinear interpolation: Start with values $v_{i,j}$ on a square or rectangular coarse grid. Interpolate to fill in $u_{i,j}$ by a sweep (interpolation) in one direction followed by a sweep in the other direction. We could allow two spacings h_x and h_y, but one meshwidth h is easier to visualize. A horizontal sweep along row i of the coarse grid (which is row $2i$ of the fine grid) will fill in values of u at odd-numbered columns $2j+1$ of the fine grid:

$$\textbf{Horizontal sweep} \qquad u_{2i,2j} = v_{i,j} \quad \text{and} \quad u_{2i,2j+1} = \frac{1}{2}(v_{i,j} + v_{i,j+1}) \quad \text{as in 1D.} \qquad (4)$$

Now sweep vertically, up each column of the fine grid. Interpolation will keep those values (4) on even-numbered rows $2i$. It will average those values to find $\boldsymbol{u} = \boldsymbol{I2\mathrm{D}\,v}$ on the fine-grid odd-numbered rows $2i+1$:

$$\begin{array}{llcl} \textbf{Vertical sweep} & u_{2i+1,2j} & = & (v_{i,j} + v_{i+1,j})/2 \\ \textbf{Averages of (4)} & u_{2i+1,2j+1} & = & (v_{i,j} + v_{i+1,j} + v_{i,j+1} + v_{i+1,j+1})/4\,. \end{array} \qquad (5)$$

The entries in the tall thin coarse-to-fine interpolation matrix $I2\mathrm{D}$ are 1, $\frac{1}{2}$, and $\frac{1}{4}$.

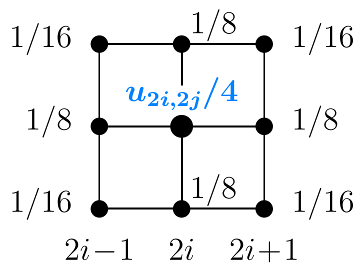
The full weighting fine-to-coarse restriction operator $R2\mathrm{D}$ is the *transpose* $I2\mathrm{D}^{\mathrm{T}}$, multiplied by $\frac{1}{4}$. That factor is needed (like $\frac{1}{2}$ in one dimension) so that a constant vector of 1's will be restricted to a constant vector of 1's. (The entries along each row of the wide matrix R add to 1.) This restriction matrix has entries $\frac{1}{4}$, $\frac{1}{8}$, and $\frac{1}{16}$ and *each coarse-grid value v is a weighted average of nine fine-grid values u*:

Restriction matrix $\boldsymbol{R = \frac{1}{4} I^{\mathrm{T}}}$

Row $\boldsymbol{i, j}$ of $\boldsymbol{R}$ produces $\boldsymbol{v_{i,j}}$

$\boldsymbol{v_{i,j}}$ uses $\boldsymbol{u_{2i,2j}}$ and 8 neighbors

The nine weights add to 1



You can see how a sweep along each row with weights $\frac{1}{4}$, $\frac{1}{2}$, $\frac{1}{4}$, followed by a sweep down each column, gives the nine coefficients in that "restriction molecule." Its matrix $R2\mathrm{D}$ is an example of a *tensor product* or *Kronecker product* `kron`(R, R). A 3 by 7 matrix R in one dimension becomes a 9 by 49 restriction matrix $R2\mathrm{D}$ in two dimensions.

Now we can transfer vectors between grids. We are ready for the ***geometric multigrid*** method, when the geometry is based on spacings h and $2h$ and $4h$. The idea extends to triangular elements (each triangle splits naturally into four similar triangles). The geometry can be more complicated than our model on a square.

When the geometry becomes too difficult, or we are just given a matrix, we turn (in the final paragraph) to ***algebraic multigrid***. This will imitate the multi-scale idea, but it works directly with $Au = b$ and not with any underlying geometric grid.

A Two-Grid V-Cycle (a v-cycle)

Our first multigrid method only involves two grids. The iterations on each grid can use Jacobi's $I - D^{-1}A$ (possibly weighted by $\omega = 2/3$ as in the previous section) or Gauss-Seidel. For the larger problem on the fine grid, iteration converges slowly to the low frequency smooth part of the solution u. *The multigrid method transfers the current residual* $r_h = b - Au_h$ *to the coarse grid.* We iterate a few times on that $2h$ grid, to approximate the coarse-grid error by E_{2h}. Then interpolate back to E_h on the fine grid, make the correction to $u_h + E_h$, and begin again.

This fine-coarse-fine loop is a **two-grid V-cycle**. We call it a **v-cycle** (small v). Here are the steps (remember, the error solves $A_h(u - u_h) = b_h - A_h u_h = r_h$):

1. **Iterate** on $A_h u = b_h$ to reach u_h (say 3 Jacobi or Gauss-Seidel steps).
2. **Restrict** the residual $r_h = b_h - A_h u_h$ to the coarse grid by $\boxed{r_{2h} = R_h^{2h} r_h}$.
3. **Solve** $\boxed{A_{2h}E_{2h} = r_{2h}}$ (or come close to E_{2h} by 3 iterations from $E = 0$).
4. **Interpolate** E_{2h} back to $\boxed{E_h = I_{2h}^h E_{2h}}$. Add E_h to u_h.
5. **Iterate** 3 more times on $A_h u = b_h$ starting from the improved $u_h + E_h$.

Steps **2-3-4** give the restriction-coarse solution-interpolation sequence that is the heart of multigrid. Recall the three matrices we are working with:

$$\begin{aligned} \boldsymbol{A} &= \boldsymbol{A_h} &&= \textit{\textbf{original matrix}} \\ \boldsymbol{R} &= \boldsymbol{R_h^{2h}} &&= \textit{\textbf{restriction matrix}} \\ \boldsymbol{I} &= \boldsymbol{I_{2h}^h} &&= \textit{\textbf{interpolation matrix.}} \end{aligned}$$

Step **3** involves a fourth matrix A_{2h}, to be defined now. A_{2h} is square and it is smaller than the original A_h. In words, we want to "project" the larger matrix A_h onto the coarse grid. There is a natural choice! The variationally correct A_{2h} comes directly and beautifully from R and A and I:

$$\textit{\textbf{The coarse grid matrix is}}\ \boldsymbol{A_{2h} = R_h^{2h} A_h I_{2h}^h = RAI.} \tag{6}$$

When the fine grid has $N = 7$ interior meshpoints, the matrix A_h is 7 by 7. Then the coarse grid matrix RAI is (3 by 7)(7 by 7)(7 by 3) = **3 by 3**.

Example In one dimension, $A = A_h$ might be the second difference matrix K/h^2. Our first example came from $h = \frac{1}{8}$. Now choose $h = \frac{1}{6}$, so that multigrid goes from five meshpoints inside $0 < x < 1$ to two meshpoints (I is 5 by 2 and R is 2 by 5): The neat multiplication (we will use it again later) is $RA_h = RK_5/h^2$:

$$RA = \frac{1}{4}\begin{bmatrix} 1 & 2 & 1 & & \\ & & 1 & 2 & 1 \end{bmatrix} \frac{1}{h^2} \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & -1 & 2 & -1 & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix} = \frac{1}{(2h)^2}\begin{bmatrix} 0 & \mathbf{2} & 0 & \mathbf{-1} & 0 \\ 0 & \mathbf{-1} & 0 & \mathbf{2} & 0 \end{bmatrix}. \tag{7}$$

A natural choice for A_{2h} on the coarse grid is $K_2/(2h)^2$ and multigrid makes this choice:

$$\textbf{Coarse grid matrix} \qquad A_{2h} = RAI = \frac{1}{(2h)^2}\begin{bmatrix} 2 & -1 \\ -1 & 2 \end{bmatrix}. \tag{8}$$

The reader will appreciate that the $I^{\mathrm{T}}AI$ rule preserves symmetry and positive definiteness, when A has those properties. The rule arises naturally in Galerkin methods [–,page –], including the finite element method. Notice how the restriction operator R with the factor $\frac{1}{4}$ automatically adjusts $1/h^2$ to $1/(2h)^2$.

Steps **1** and **5** are necessary, but they are really outside the essential multigrid idea. The **smoother** is step **1**, the **post-smoother** is step **5**. Those are normal iterations for which weighted Jacobi or Gauss-Seidel is satisfactory.

The Errors e_h and E_h

Suppose we solve the coarse grid equation exactly at step **3**. Is the multigrid error correction E_h then equal to the true fine-grid error $e_h = u - u_h$? No, that is too much to expect! We have only solved the smaller problem on the coarse grid, not the full problem. But the connection between E_h and e_h is simple and crucial for understanding multigrid. We now track down the steps from E to e.

Four matrices multiply e. At step **2**, the residual $r = b - Au_h = A(u - u_h) = Ae$ multiplies by A. The restriction is multiplication by R. The solution step **3** multiplies by $A_{2h}^{-1} = (RAI)^{-1}$. The interpolation step **4** multiplies by I to find the correction E. *Altogether,* E *is* $\boldsymbol{IA_{21}^{-1}RA_h e}$:

$$E = I(RAI)^{-1}RAe \quad \text{and we call this} \quad \boldsymbol{E = Se}\,. \tag{9}$$

When I is 5 by 2 and R is 2 by 5, that matrix S on the right side is 5 by 5. It can't be the identity matrix, since RAI and its inverse are only 2 by 2 (rank two). But $S = I(RAI)^{-1}RA$ has the remarkable property $S^2 = S$. This says that S *is the identity matrix on its 2-dimensional column space.* (Of course S is the zero matrix on its 3-dimensional nullspace.) $S^2 = S$ is easy to check:

$$S^2 = (I(RAI)^{-1}RA)(I(RAI)^{-1}RA) = S \quad \text{because } (RAI)^{-1}RAI \text{ disappears.} \tag{10}$$

So the multigrid correction $E = Se$ is not the whole error e, *it is a projection of* e. The new error is $e - E = e - Se = (I - S)e$. ***This matrix $I - S$ is the two-grid operator.*** $I - S$ plays the same fundamental role in describing the multigrid steps **2–4** that the usual $M = I - P^{-1}A$ plays for each iteration in steps **1** and **5**:

$$\textbf{v-cycle matrix} = \boldsymbol{I - S} \qquad \textbf{iteration matrix} = \boldsymbol{I - P^{-1}A}\,.$$

Example (continued). The 5 by 5 matrix $A_h = K_5/h^2$ and the rectangular I and R led in (8) to $A_{2h} = K_2/(2h)^2$. To find $S = IA_{2h}^{-1}RA_h$, we multiply (7) by IA_{2h}^{-1}:

$$A_{2h}^{-1}RA_h = \begin{bmatrix} 0 & \mathbf{1} & 0 & \mathbf{0} & 0 \\ 0 & \mathbf{0} & 0 & \mathbf{1} & 0 \end{bmatrix}$$

Now multiply by I to find S

$$S = \begin{bmatrix} 0 & 1/2 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 1/2 & 0 & 1/2 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1/2 & 0 \end{bmatrix}. \qquad (11)$$

The eigenvalues of this S are $1, 1, 0, 0, 0$. If you square S, you recover $S^2 = S$. With its three columns of zeros, the nullspace of S contains all fine-grid vectors of the form $(e_1, 0, e_3, 0, e_5)$. Those are vectors that don't appear on the coarse grid. If the error e had this form, then $E = Se$ would be zero (no improvement from multigrid). *But we don't expect a large component of those high frequency vectors in e, because of the smoothing.*

The column space of S contains column 2 $= (\frac{1}{2}, 1, \frac{1}{2}, 0, 0)$ and column 4 $= (0, 0, \frac{1}{2}, 1, \frac{1}{2})$. These are "mixed-frequency vectors." We do expect them to appear in e, because the smoothing step didn't remove them. But these are vectors for which $E = Se = e$ and they are the errors that multigrid catches! After step **4** they are gone.

Let me say this in another way. Because $S^2 = S$, the only eigenvectors are $\lambda = 0$ and $\lambda = 1$. (If $Su = \lambda u$ we always have $S^2u = \lambda^2 u$. Then $S^2 = S$ gives $\lambda^2 = \lambda$.) Our example has $\lambda = 1, 1, 0, 0, 0$. The eigenvalues of $I - S$ are $0, 0, 1, 1, 1$. The eigenvectors e reveal what multigrid is doing:

$\boldsymbol{E = Se = 0}$ In this case multigrid gives no improvement. The correction E added to u_h in step **4** is zero. IN the example, this happens for errors $e = (e_1, 0, e_3, 0, e_5)$ that are zero on the coarse grid where step **3** is working.

$\boldsymbol{E = Se = e}$ In this case multigrid is perfect. The correction E_h added to u_h in step **4** is the whole error e_h. In the example, two eigenvectors of S for $\lambda = 1$ are $e = (1, 2, 2, 2, 1)$ and $e = (1, 2, 0, -2, -1)$. Those have large low-frequency components. They oscillate up and down only once and twice. They are in the column space of I. They are no perfect sines, but an important part of the low-frequency error is caught and removed. The number of independent vectors with $Se = e$ is the number of coarse gridpoints (here 2). That measures the A_{2h} problem that step **3** deals with. It is the rank of S and also I. The other $5 - 2$ gridpoints account for the nullspace of S, where $E = Se = 0$ means no improvement from multigrid.

Note The "high-frequency" vectors $(u_1, 0, u_3, 0, u_5)$ with $Su = 0$ are *not exactly* combinations of the last three discrete sines y_3, y_4, y_5. The frequencies are mixed by S, as equations (18–19) will clearly show. The exact statements are *column space of S = column space of I* and *nullspace of S = nullspace of RA*. The mixing of frequencies does not affect our main point: **Iteration handles the high frequencies and multigrid handles the low frequencies.**

You can see that a perfect smoother followed by perfect multigrid (exact solution at step **3**) would leave no error. In reality, this will not happen. Fortunately, a careful (not so simple) analysis will show that a multigrid cycle with good smoothing can reduce the error by a constant factor ρ that is **independent of h**:

$$\|\textbf{error after step 5}\| \;\leq\; \rho\,\|\textbf{error before step 1}\| \quad \text{with} \quad \rho < 1\,. \tag{12}$$

A typical value is $\rho = \frac{1}{10}$. Compare with $\rho = .99$ for Jacobi alone. This is the Holy Grail of numerical analysis, to achieve a convergence factor ρ (a spectral radius of the overall iteration matrix) that *does not move up to* 1 *as* $h \to 0$. We can achieve a given relative accuracy in a fixed number of cycles. Since each step of each cycle requires only $O(n)$ operations on sparse problems of size n, **multigrid is an $O(n)$ algorithm**. This does not change in higher dimensions.

There is a further point about the number of steps and the accuracy. The user may want the solution error e to be as small as the discretization error (when the original differential equation was replaced by $Au = b$). In our examples with second differences, this demands that we continue until $e = O(h^2) = O(N^{-2})$. In that case we need more than a fixed number of v-cycles. To reach $\rho^k = O(N^{-2})$ requires $k = O(\log N)$ cycles. Multigrid has an answer for this too.

Instead of repeating v-cycles, or nesting them into V-cycles or W-cycles, it is better to use **full multigrid**: FMG cycles are described below. Then the operation count comes back to $O(n)$ even for this higher required accuracy $e = O(h^2)$.

V-Cycles and W-Cycles and Full Multigrid

Clearly multigrid need not stop at two grids. If it did stop, it would miss the remarkable power of the idea. The lowest frequency is still low on the $2h$ grid, and that part of the error won't decay quickly until we move to $4h$ or $8h$ (or a very coarse $512h$).

The two-grid v-cycle extends in a natural way to more grids. It can go down to coarser grids $(2h, 4h, 8h)$ and back up to $(4h, 2h, h)$. This nested sequence of v-cycles is a **V-cycle** (capital V). *Don't forget that coarse grid sweeps are much faster than fine grid sweeps.* Analysis shows that time is well spent on the coarse grids. So the **W-cycle** that stays coarse longer (Figure 6.11b) is generally superior to a V-cycle.

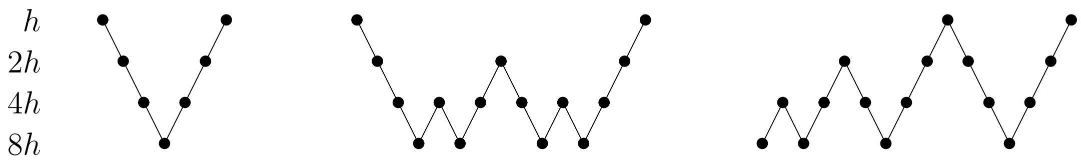


Figure 6.11: V-cycles and W-cycles and FMG use several grids several times.

The **full multigrid cycle** in Figure 6.11c is asymptotically better than V or W. *Full multigrid starts on the coarsest grid.* The solution on the $8h$ grid is interpolated to provide a good initial vector u_{4h} on the $4h$ grid. A v-cycle between $4h$ and $8h$ improves it. Then interpolation predicts the solution on the $2h$ grid, and a deeper

V-cycle makes it better (using $2h, 4h, 8h$). Interpolation of that improved solution onto the finest grid gives an excellent start to the last and deepest V-cycle.

The operation counts for a deep V-cycle and for full multigrid are certainly greater than for a two-grid v-cycle, but *only by a constant factor*. That is because the count is divided by a power of 2 every time we move to a coarser grid. For a differential equation in d space dimensions, we divide by 2^d. The cost of a V-cycle (as deep as we want) is less than a fixed multiple of the v-cycle cost:

$$\textbf{V-cycle cost} < \left(1+\frac{1}{2^d}+\left(\frac{1}{2^d}\right)^2+\cdots\right)\textbf{v-cycle cost} = \frac{2^d}{2^d-1}\,\textbf{v-cycle cost}\,. \quad (13)$$

Full multigrid is no more than a series of inverted V-cycles, beginning on a very coarse mesh. By the same reasoning that led to (13),

$$\textbf{Full multigrid cost} < \frac{2^d}{2^d-1}\,\textbf{V-cycle cost} < \left(\frac{2^d}{2^d-1}\right)^2\textbf{v-cycle cost}\,. \quad (14)$$

And the method works in practice. But good programming is required.

Multigrid Matrices

For a 3-grid V-cycle, what matrix S_3 corresponds to the 2-grid v-cycle projection $S = I(RAI)^{-1}RA$? No smoothing is involved here. S and S_3 only project to coarser problems. By itself, S_3 will not be a good solver without smoothers.

To construct A_{4h}, replace the matrix $A_{2h} = RAI$ on the middle grid by using a coarse restriction $R_c = R_{2h}^{4h}$ that transfers down to the $4h$ grid, and an interpolation $I_c = I_{4h}^{2h}$ that comes back to $2h$:

$$\textbf{Very coarse matrix} \qquad \boldsymbol{A_{4h} = R_c A_{2h} I_c}\,. \quad (15)$$

If $h = \frac{1}{16}$, that product is $(3 \times 7)(7 \times 7)(7 \times 3)$. The 3 by 3 problem uses A_{4h} on the $4h = \frac{1}{4}$ grid (with three interior unknowns). Then $S_3 = (S_3)^2$ is the matrix that goes down two grids, solves the very coarse problem by A_{4h}^{-1}, and comes back up:

$$\boldsymbol{E_3 = S_3 e} = \textbf{Error removed} \qquad S_3 = I_{2h}^{h} I_{4h}^{2h} A_{4h}^{-1} R_{2h}^{4h} R_h^{2h} A\,. \quad (16)$$

The error that remains after an unsmoothed V-cycle will be $(I - S_3)e$. On the $h = \frac{1}{16}$ grid, there are 15 frequencies in the error e. Only the lowest 3 are (approximately) removed by $S_3 e$. We have only solved a 3 by 3 problem with A_{4h}. It is the *smoothers*, on the fine h grid and the middle $2h$ grid, that reduce the high and middle frequency errors in the solution.

Note to myself: $S_{4h} = I_c A_{4h}^{-1} R_c A_{2h} = I_c A_{4h}^{-1} R_c RAI = 7 \times 7$ has I on right instead of left. Still $S_{4h}^2 = S_{4h}$.

Numerical Experiments

The real test of multigrid effectiveness is numerical! Its k-step approximation e_k should approach zero and the graphs will show how quickly this happens. An initial guess u_0 includes an error e_0. Whatever iteration we use, we are trying to drive u_k to u, and e_k to zero. The multigrid method jumps between two or more grids, so as to converge more quickly, and our graphs will always show the error on the fine grid.

We can work with the equation $Ae = 0$, whose solution is $e = 0$. Following [–], Figure 6.____a displays an initial error with low and high frequencies:

$$(e_0)_j = \sin 4j\pi h + \sin 20j\pi h \quad \text{with} \quad h = \frac{1}{32}\,.$$

We draw continuous functions $\sin 4\pi x$ and $\sin 20\pi x$ rather than 31 discrete values. Figure 6.____b shows the error vector e_3 after three fine-grid sweeps of weighted Jacobi (with $\omega = \frac{2}{3}$). As expected, the high frequency component has greatly decayed. The error $e_3 = (I - P^{-1}A)^3 e_0$ is much smoother than e_0. For our second difference matrix A_h (better known as K), the Jacobi preconditioner from Section 6.2 simply has $P^{-1} = \frac{1}{2}\omega I$. We can ignore the factors h^2 that divide K and P, because they cancel.

Figure 6.12: The initial error and the error e_3 after three weighted Jacobi relaxations.

Now multigrid begins. The current fine-grid residual is $r_h = -A_h e_3$. After restriction to the coarse grid it becomes r_{2h}. Three weighted Jacobi iterations on the coarse grid error equation $A_{2h}E_{2h} = r_{2h}$ start with the guess $E_{2h} = 0$. That produces...

ADD V-CYCLE AND FMG EXPERIMENTS

Figure 6.13: The fine-grid error after 3 sweeps on the coarse grid (a) and then the fine grid (b).

Eigenvector Analysis

The reader will recognize that one matrix (like $I - S$ for a v-cycle) can describe each multigrid method. The eigenvalues of a full multigrid matrix would be nice to know, but they are usually impractical to find. Numerical experiments build confidence. Computation also provides a *diagnostic* tool, to locate where convergence is stalled and a change is needed (often in the boundary conditions). If we want a *predictive* tool, the best is **modal analysis**.

The key idea is to watch the Fourier modes. In our example those are discrete sines, because of the boundary conditions $u(0) = u(1) = 0$. We will push this model problem all the way, to see what the multigrid matrix $I - S$ does to those sine vectors.

The final result in (18–19) shows why multigrid works and it also shows how *pairs of frequencies are mixed.* The eigenvectors are mixtures of two frequencies.

The frequencies that mix together are k and $N+1-k$. The discrete sines with those frequencies are y_k and Y_k:

$$y_k = \left(\sin\frac{k\pi}{N{+}1}, \sin\frac{2k\pi}{N{+}1}, \ldots\right) \text{ and } Y_k = \left(\sin\frac{(N{+}1{-}k)\pi}{N{+}1}, \sin\frac{2(N{+}1{-}k)\pi}{N{+}1}, \ldots\right).$$

Where y_k goes up and down k times, Y_k does that $N+1-k$ times.

There is something neat you have to see. **Y_k and y_k have the same components except for alternating signs!** Cancel $N{+}1$ with $N{+}1$ in those components of Y_k:

$$Y_k = \left(\sin\left[\pi - \frac{k\pi}{N{+}1}\right], \sin\left[2\pi - \frac{2k\pi}{N{+}1}\right], ..\right) = \left(+\sin\frac{k\pi}{N{+}1}, -\sin\frac{2k\pi}{N{+}1}, ..\right) \quad (17)$$

For our model problem with second differences, we can report the result of multiplying by $S = I(RAI)^{-1}RA$. In fact the true multigrid matrix is $I - S$, to give the error $e - E = (I-S)\,e$ that remains after steps **2**, **3**, **4**. Seeing $I - S$ is even better:

One v-cycle ($2k \le N{+}1$)

Smooth errors reduced

Frequencies mixed

$$(I-S)\,y_k = \frac{1}{2}\left(1 - \cos\frac{k\pi}{N{+}1}\right)(y_k + Y_k) \quad (18)$$

$$(I-S)Y_k = \frac{1}{2}\left(1 + \cos\frac{k\pi}{N{+}1}\right)(y_k + Y_k) \quad (19)$$

Such beautiful formulas can't be expected in more complicated problems, and we seize this chance to make four key points:

1. **Low frequencies like $k = 1, 2, 3$ are greatly reduced by multigrid.** The cosine in equation (18) will be near 1. The factor $(1 - \cos\frac{k\pi}{N+1})$ is very small, of order $(k/N)^2$. This shows how multigrid is powerful in nearly killing the low frequencies. These are exactly the frequencies on which Jacobi and Gauss-Seidel will stall in the smoothing iterations.

2. **The pair of sines y_k and Y_k is mixed together by multigrid.** We will see that the restriction R and interpolation I are responsible. Those are not like the square matrix A, which keeps frequencies separate (because the sines are its eigenvectors). Aliasing appears for rectangular matrices!

3. **The combinations $e = y_k + Y_k$ are eigenvectors of $I - S$ with $\lambda = 1$.** Just add equations (18) and (19), to get $(I-S)e = e$. Since Y_k has the same components as y_k with alternating signs, $y_k + Y_k$ has the correct form $(e_1, 0, e_3, 0, \ldots)$. Those are the vectors that we discovered earlier in the nullspace ($\lambda = 0$) of S. They are not touched by $I - S$ since $Se = 0$.

4. **The other eigenvectors of $I-S$ are just Sy_k.** We know that $(I-S)Sy_k = 0$ because $S = S^2$. In our example with $N = 5$, the vectors Sy_1 and Sy_2 are multiples of $(1, 2, 2, 2, 1)$ and $(1, 2, 0, -2, 1)$ that we found explicitly. Multigrid removes them from the error.

According to (18), these vectors Sy_k are combinations of y_k and Y_k. To find a good combination, multiply (18) and (19) by $(1 + \cos \frac{k\pi}{N+1})$ and $(1 - \cos \frac{k\pi}{N+1})$. The right-hand sides are now the same and we subtract:

$$\boldsymbol{S^* = e^*} \quad (I - S)e^* = (I - S) \left[\left(1 + \cos \frac{k\pi}{N+1}\right) y_k - \left(1 - \cos \frac{k\pi}{N+1}\right) Y_k \right] = 0 \quad (20)$$

These mixtures e^* in square brackets are completely killed ($\lambda = 0$) by multigrid. A smooth error vector e_h (after Jacobi iterations have reduced its high frequency components) has only small components along the eigenvectors $y_k + Y_k$. Multigrid doesn't touch those pieces ($\lambda = 1$). The larger components along the Z_k will die.

The Restriction R Produces Aliasing

To complete this analysis we have to see where and how a pair of frequencies is mixed. The aliasing comes from the restriction matrix $R = R_h$, when both vectors y_k^h and Y_k^h lead to multiples of the same output vector y_k^{2h}:

$$Ry_k^h = \left(\frac{1}{2} - \frac{1}{2} \cos \frac{k\pi}{N+1}\right) y_k^{2h} \quad \text{and} \quad RY_k^h = \left(-\frac{1}{2} - \frac{1}{2} \cos \frac{k\pi}{N+1}\right) y_k^{2h}. \qquad (21)$$

You see the aliasing by R. We cannot hope to decide the input y_k or Y_k from these outputs. This is normal for a short wide matrix. (The matrix R is 3 by 7 and 2 by 5 in our examples.) The coarse mesh output has only about half as many components as the fine mesh input.

The transpose of R does the opposite. Where R mixes two inputs y_k^h and Y_k^h into one output, the interpolation matrix I sends one input frequency on the coarse grid into a pair of frequencies on the fine grid:

$$2\, I_{2h}^h y_k^{2h} = \left(1 + \cos \frac{k\pi}{N+1}\right) y_k^h - \left(1 - \cos \frac{k\pi}{N+1}\right) Y_k^h. \qquad (22)$$

Interpolation of a smooth vector (low k) on a coarse grid will excite an oscillatory mode Y_k^h (high k) on the fine grid. But those oscillations have small amplitude, because the cosine of $k\pi/(N+1)$ is near 1.

The key formulas (18) and (19) that describe multigrid come from assembling (21) for R, (22) for I, and the known eigenvalues for A_h and A_{2h}. I think the calculation of S in (11) shows this best. Its zero columns put $y_k + Y_k$ in its nullspace. The nonzero columns in S come from the interpolation matrix I. So Se captures a part of the whole error e. *Multigrid solves a projection of the original problem.*

Example completed It would be useless to repeat steps **2**, **3**, **4** with no smoothing in between. Nothing would change! The untouched vectors $e = y_k + Y_k$ with $(I - S)e = e$ will still be untouched. It is the smoothing matrix $M = I - P^{-1}A$ that must reduce these errors.

If we apply Jacobi with weight $w = \frac{2}{3}$ at steps **1** and **5**, then $M = I - \frac{1}{3}A$. The overall matrix for all steps **1** to **5** will be $M(I - S)M$. The eigenvalues of that matrix will decide the success (or not) of multigrid. To my amazement, the 5 by 5 matrix $M(I - S)M$ has a *triple eigenvalue of* $\frac{1}{9}$!

The three eigenvalues $\lambda = 1$ of $I - S$ are reduced to $\lambda = \dfrac{1}{9}$ for $M(I - S)M$.

The largest eigenvalue of M is .91—you see the value of multigrid. I hope you will try eig$(M * M * (I - S) * M * M)$ with double smoothing (see Problems 9–12).

Fourier Modal Analysis

Pure modal analysis neglects the boundaries completely. It assumes an infinite grid! The vectors y_k in the example were sines because of the boundary conditions. When boundaries are gone, the y_k are replaced by infinitely long vectors y_ω coming from complex exponentials. Now there is a continuum of frequencies ω:

$$y_\omega = (\ldots, e^{-2i\omega}, e^{-i\omega}, 1, e^{i\omega}, e^{2i\omega}, \ldots) \quad \text{with} \quad -\pi \le \omega \le \pi \,. \tag{23}$$

We need infinite matrices K_∞ to multiply these infinite vectors. Second differences $-1, 2, -1$ appear on *all rows forever*. The key is that each y_ω is an eigenvector of K_∞, with eigenvalue $\lambda = 2 - 2\cos\omega$:

$$K_\infty \, y_\omega = (2 - 2\cos\omega)\, y_\omega \quad \text{because} \quad -e^{i\omega(n+1)} - e^{i\omega(n-1)} = -2\cos\omega \, e^{i\omega n} \,. \tag{24}$$

This tells us the action of $A_h = K_\infty/h^2$. It also tells us about A_{2h}, when the coarse mesh changes h^2 to $(2h)^2$ and ω to 2ω.

The restriction matrix R still introduces aliasing. The frequencies that mix together are now ω and $\omega + \pi$. Notice how increasing ω by π produces a factor $e^{i\pi n} = (-1)^n$ with alternating signs. This is exactly what we saw for y_k and Y_k. This time it is $y_\omega - y_{\omega+\pi}$ that has all zeros in its even-numbered components.

This pure Fourier analysis will go all the way to formulas for the infinite $(I - S)y_\omega$ and $(I - S)y_{\omega+\pi}$, just like equations (18) and (19). In that finite case, those equations explained why multigrid succeeds. They do the same in the infinite case.

Let me emphasize why this pure modal analysis (with no boundaries and constant coefficients) was mentioned. It allows Fourier to work freely. The differential equation $\operatorname{div}(c(x, y)\operatorname{grad} u) = f(x, y)$ on a general region would lead to giant complications in the eigenvectors for multigrid—impossible to find them. But if we fix $c =$ constant and ignore boundaries, those "interior eigenvectors" return to simple combinations of y_ω and $y_{\omega+\pi}$. That leaves difficulties associated with the boundary conditions, which Fourier doesn't easily resolve.

Algebraic Multigrid

We close this section with a few words about **algebraic multigrid**, when the problem comes as a system of equations $Au = b$. *There is no grid in the background.* We need to find replacements for the key ideas on which geometric multigrid was based: *smooth vectors, connected nodes, coarse meshes.* Replacing the first two is fairly easy, rescaling A_h to A_{2h} on a (nonexistent) coarse mesh is less clear.

1. **Smooth vectors.** In geometric multigrid these are vectors for which the norms of u and Au are comparable. That will be our indication of a smooth vector. High frequencies in u would be greatly amplified by A (just as the second derivative amplifies $\sin kt$ by k^2). For low frequencies, Multigrid works on smooth errors and Jacobi doesn't.

2. **Connected nodes.** On a grid, neighboring nodes are reflected by a nonzero entry in A. When there is no grid, we look directly at the matrix. Its significant nonzero entries A_{ij} (magnitude greater than αA_{ii}, say with $\alpha = \frac{1}{10}$) tell us when we should think of the "nodes" i and j as being connected.

3. **Coarse subset of nodes.** Each significant entry A_{ij} indicates that the value of u_j *strongly influences* the value of u_i. Probably the errors e_j and e_i are comparable, when the error is smooth. We don't need both i and j in the "coarse set C" of nodes. On a grid they would be neighbors, not both in C.

   But if i is *not* in C, then every j that strongly influences i should either be in C or be strongly influenced by another J that *is* in C. This heuristic rule is discussed more fully in [–], with an algorithm for constructing C. (In general, too many coarse unknowns are better than too few.)

The excellent book [–] also constructs the coarse-to-fine interpolation matrix I. This starts with the errors E_j for j in C, and leaves them unchanged. If i is not in C, the interpolated value E_i at step **2** of multigrid will be a *weighted combination* of the E_j that do have j in C. In our one-dimensional model problem, that weighted combination was the average of the two neighbors of i. That model problem certainly had a grid!

The interpolating combination will give greatest weight to the e_j for which j in C strongly influences i. But there may be smaller entries A_{ij} that cannot be completely ignored. The final decision on the weights for each interpolated value is more subtle than a simple average. Algebraic multigrid is more expensive than geometric multigrid, but it applies to a much wider range of sparse matrices A (and the software can control AMG without us). We still expect a "smoothing + multigrid" combination that is close to $O(n)$ steps for an accurate solution of $Au = b$.

Let me mention an important contrast between solid mechanics and fluid mechanics. For solids, the original finite element grid is already relatively coarse. We are typically looking for "engineering accuracy" and we don't have fine scale motions to resolve. Multigrid is not such a common choice for structural problems (elimination is simpler). Fluids do have fine scales so that multigrid becomes a natural idea, not only numerically but physically. Of course fluids don't generally present symmetric matrices, because of the convective terms, and finite element methods may require "upwind adjustments." The analysis of multigrid convergence becomes harder for fluids, just when a multiscale approach becomes attractive.

Problem Set 6.3

1 What is the 3 by 7 matrix $R_{\textbf{injection}}$ that copies v_1, v_2, v_3 from u_2, u_4, u_6 and ignores u_1, u_3, u_5, u_7 ?

2 Write down a 9 by 4 bilinear interpolation matrix I that uses the four grid values at the corners of a square (side $2h$) to produce nine values at the (h) gridpoints. What constant should multiply the transpose of I to give a one-element restriction matrix R?

3 In Problem 2, the four small squares (side h) subdivide into 16 smaller squares (side $h/2$). How many rows and columns in the interpolation matrix $I_{h/2}$?

4 If A is the 5-point discrete Laplace matrix (with $-1, -1, 4, -1, -1$ on a typical row) what is a typical row of $A_{2h} = RAI$ using bilinear interpolation as in Problems 2–3 ?

5 Suppose A comes from the 9-point stencil (8/3 surrounded by eight entries of $-1/3$). What is now the stencil for $A_{2h} = RAI$?

6 Verify $Ry_k^h = \frac{1}{2}(1+\cos\frac{k\pi}{N+1})y_k^{2h}$ in equation (23) for the linear restriction matrix R applied to discrete sines y_k^h with $k \le \frac{1}{2}(N+1)$.

7 Show that $RY_k^h = \frac{1}{2}(-1-\cos\frac{k\pi}{N+1})y_k^{2h}$ in equation (23) for the "complementary" discrete sines $Y_k^h = y_{N+1-k}$. Now $N+1-k > \frac{1}{2}(N+1)$.

8 Verify equation (24) for linear interpolation applied to the discrete sines y_k^h with $k \le \frac{1}{2}(N+1)$.

9 With $h = \frac{1}{8}$, use the 7 by 3 and 3 by 7 matrices I and R in equations (1–2) to find the 3 by 3 matrix $A_{2h} = RAI$ with $A = K_7/h^2$.

10 Continue Problem 9 to find the 7 by 7 matrix $S = I(RAI)^{-1}RA$. Verify that $S^2 = S$, and that S has the same nullspace as RA and the same column space as I. What are the seven eigenvalues of S ?

11 Continue Problem 10 to find (by MATLAB) the multigrid matrix $I - S$ and the presmoothed/postsmoothed matrix MSM, where M is the Jacobi smoother $I - \omega D^{-1}A = I - \frac{1}{3}K_7$ with $\omega = \frac{2}{3}$. Find the eigenvalues of MSM !

12 Continue Problem 11 to find the eigenvalues of M^2SM^2 with two Jacobi smoothers ($\omega = \frac{2}{3}$) before and after the grid changes.

13 With unweighted Jacobi ($\omega = 1$ and $M = I - \frac{1}{2}K_7$) find MSM and its eigenvalues. Is the weighting useful?

14 Starting from $h = \frac{1}{16}$ and the second difference matrix $A = K_{15}/h^2$, compute the projected 7 by 7 matrix $A_{2h} = RAI$ and the doubly projected 3 by 3 matrix $A_{4h} = R_c A_{2h} I_c$. Use linear interpolation matrices I and I_c, and restriction matrices $R = \frac{1}{2}I^{\mathrm{T}}$ and $R_c = \frac{1}{2}I_c^{\mathrm{T}}$.

15 Use A_{4h} from Problem 14 to compute the projection matrix S_3 in (16). Verify that this 15 by 15 matrix has rank 3, and that $(S_3)^2 = S_3$.

16 The weighted Jacobi smoothers are $M_h = I_{15} - \frac{1}{3}K_{15}$ and $M_{2h} = I_7 - \frac{1}{3}K_7$. With MATLAB, compute the smoothed error reduction matrix V_3 and its eigenvalues:

$$V_3 = M_h I M_{2h}(I_7 - S_{2h})M_{2h}RAM_h\,.$$

$S_{2h} = I_c A_{4h}^{-1} R_c A_{2h}$ is the projection matrix for the v-cycle within the V-cycle.

17 Compute the $(15 \times 7)(7 \times 3)$ linear interpolation matrix $I_{2h}^{h} I_{4h}^{2h}$. What is the restriction matrix?