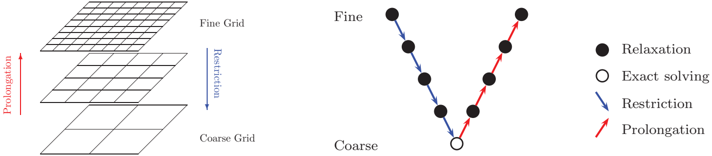
Section 03. Basic Ideas of Multigrid Methods

Multilevel Iterative Methods

Examples of multilevel algorithms

- Quick Sort, FFT, FMM, GMG, AMG, H-Matrix, H^2-Matrix, ...

Multigrid V-cycle



Key ingredients for multilevel iterative methods

- Construct multilevel hierarchy in an efficient way (setup)
- Find effective (yet cheap) smoothers for each level
- Find good coarse-grid correction (CGC) algorithms

Need complementary smoothing and CGC steps to get better convergence.

Finite Difference Methods

In one-dimensional case, we can assume $\Omega = (0,1)$ and it is divided into $N+1$ equally spaced pieces. So we get a uniform mesh with meshsize $h = \frac{1}{N+1}$; see the following figure for illustration.

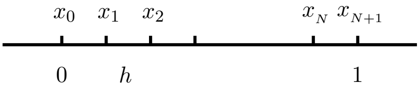


Figure: Uniform mesh in 1D.

For the right-hand side, we can use an approximation: $\vec{f} := \big(f_i\big)_{i=1}^{N} = \big(f(x_i)\big)_{i=1}^{N}$. For the left-hand side, using the Taylor's expansion, we can easily obtain that

$$
\begin{aligned}
u''(x_i) &= \frac{1}{h^2}\Big[u(x_{i-1}) - 2u(x_i) + u(x_{i+1})\Big] + O(h^2) \\
&\approx \frac{1}{h^2}\Big[u_{i-1} - 2u_i + u_{i+1}\Big],
\end{aligned}
$$

where $u_i \approx u(x_i)$ is an approximate solution (finite difference solution).

Nested Grids

Solve the 1D Poisson's equation:

$$A\vec{u} = \vec{f} \qquad \text{with} \;\; A = \frac{1}{h^2}\,\mathrm{tridiag}(-1, 2, -1), \;\; f_i = f(x_i).$$

Suppose there are a hierarchy of $L+1$ grids with $h_l = (\frac{1}{2})^{l+1}$ ($l = 0, 1, \ldots, L$). It is clear that

$$h_0 > h_1 > h_2 > \cdots > h_L =: h$$

and $N = 2^{L+1} - 1$. We call level L the finest level and level 0 the coarsest level.

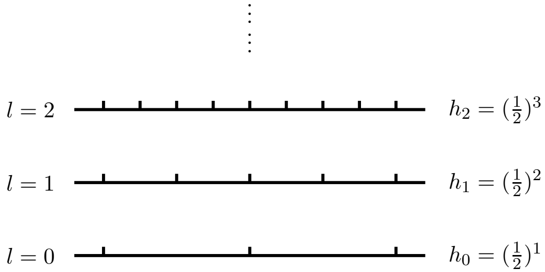


Figure: Hierarchical grids for 1D multigrid method.

Transfer Operators

In the 1D case, the transfer operators can be easily given:

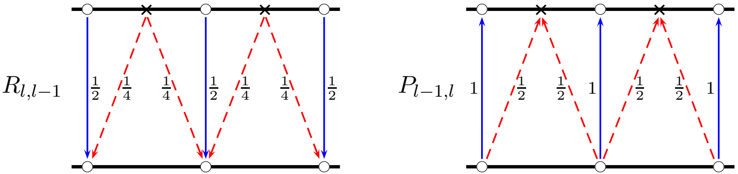


Figure: Transfer operators (Left: restriction operator; right: prolongation operator).

$$R_{l,l-1} := \frac{1}{4}\begin{pmatrix} \ddots & & & & & & \\ & 1 & 2 & 1 & & & \\ & & & 1 & 2 & 1 & \\ & & & & & & \ddots \end{pmatrix}, \quad P_{l-1,l} := \frac{1}{2}\begin{pmatrix} \ddots & & & \\ & 1 & & \\ & 2 & & \\ & 1 & 1 & \\ & & 2 & \\ & & 1 & \\ & & & \ddots \end{pmatrix}.$$

It is straight-forward to check that: $A_{l-1} = R_{l,l-1} A_l P_{l-1,l}$. So $R_{l,l-1} = ?P^T_{l-1,l}$.

Multigrid Algorithm

Error correction for linear problems: Suppose that $\vec{u}^{(m)}$ is an approximate solution. Then we have

$$A\left(\vec{u}-\vec{u}^{(m)}\right)=\vec{r}^{(m)}:=\vec{f}-A\vec{u}^{(m)}$$

and the error equation can be written

$$A\vec{e}^{(m)}=\vec{r}^{(m)}.$$

We then update the iterative solution by $\vec{u}^{(m+1)}=\vec{u}^{(m)}+\vec{e}^{(m)}$ to obtain a new approximation of $\vec{u}$.

Then we have the following recursively-defined algorithm:

Algorithm (One iteration of multigrid method $\vec{u}_l = MG(l, \vec{f}_l, \vec{u}_l)$)

1. Pre-smoothing: $\vec{u}_l \leftarrow \vec{u}_l + \frac{1}{2}D_l^{-1}\left(\vec{f}_l - A_l\vec{u}_l\right)$.
2. Restriction: $\vec{r}_{l-1} \leftarrow R_{l,l-1}\left(\vec{f}_l - A_l\vec{u}_l\right)$.
3. Coarse-grid correction: If $l=1$, $\vec{e}_{l-1} \leftarrow A_{l-1}^{-1}\vec{r}_{l-1}$; $\vec{e}_{l-1} \leftarrow MG(l-1, \vec{r}_{l-1}, \vec{0}_{l-1})$, otherwise.
4. Prolongation: $\vec{u}_l \leftarrow \vec{u}_l + P_{l-1,l}\vec{e}_{l-1}$.
5. Post-smoothing: $\vec{u}_l \leftarrow \vec{u}_l + \frac{1}{2}D_l^{-1}\left(\vec{f}_l - A_l\vec{u}_l\right)$.

A Simple Numerical Experiment

In the following table, we give the numerical results of the above algorithm for the 1D Poisson's equation (using three G-S iterations as smoother). From the table, we find that, unlike the classical Jacobi and G-S methods, this multigrid method converges uniformly with respect to the meshsize h. This is, of course, a very desirable feature of the multilevel iterative methods, which will be investigated in this course.

#Levels	#DOF	#Iter	Contract factor
5	31	4	0.0257
6	63	4	0.0259
7	127	4	0.0260
8	255	4	0.0260
9	511	4	0.0261
10	1023	4	0.0262

Table: Convergence behavior of 1D geometric multigrid method.

Textbook multigrid efficiency: "TME means solving a discrete PDE problem in a computational work which is only a small (less than 10) multiple of the operation count in the discretized system of equations itself."

Computational Cost

Assumptions:

- Denote the work needed by $\mathcal{B}_l$ is W_l.
- Assume the each smoothing sweep costs $O(N_l)$ operations and $N_l \sim h_l^{-d} \sim \gamma^{-ld}$. Then it requires $2m\,O(N_l)$ operations for the pre- and post-smoothing (m-steps) on level l.
- The prolongation and restriction also requires $O(N_l)$ operations.

Work of multilevel cycles:

$$\begin{aligned} W_{k+1} &= O\big(N_{k+1}\big) + W_k \\ &= \cdots\cdots \\ &= O\Big(\sum_{j=0}^{k+1} N_j\Big) = O\big(N_{k+1}\big)\sum_{j=0}^{k+1} \gamma^{jd}. \end{aligned}$$

Let $N = N_L$ be the number of unknowns on the finest grid. The V-cycle costs $O(N)$ operations in each cycle. Apparently, this analysis also yields computational complexity of the W-cycle, if we choose an appropriate μ_1 such that $\mu_1\gamma^d < 1$.

One question remains: How many iterations (cycles) needed to reach certain accuracy?

General Multilevel Iterative Methods

Generally speaking, multilevel iterations: Setup Phase (fixed) $\Longrightarrow$ Solve Phase (variable)

Setup phase

- Constructing transfer operators, coarse problems, etc
- Using geometric information or algebraic information or both
- Needed only once in each solution procedure
- Sometimes even shared by multiple solution procedures

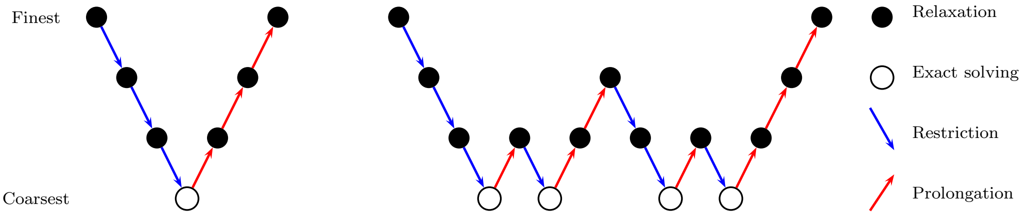
Solve phase

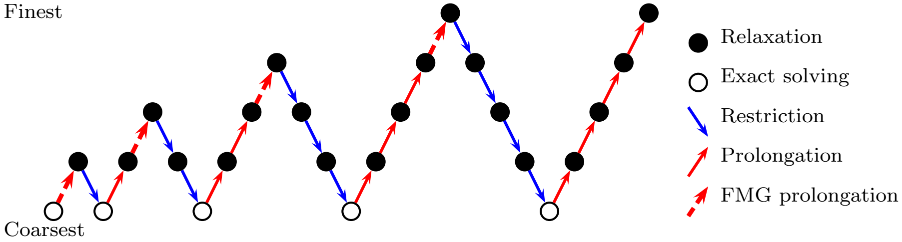
- Applying relaxation (simple iterative methods) on different levels and putting components together
- Needed many times as iteration or precondition step, but hopeful not too many
- Main concern: How to approximate coarse solution accurately without costing too much

Two-level method

- Simplest case: Two-level method (solve the coarse level exactly or approximately)
- Easier to implement and analyze, provide insight for design multilevel methods

Examples of Multilevel Cycles





V, VV, VW, W, AMLI, K, N, H, ... , Nested Iterations

A General Workflow