INTRODUCTION TO MULTIGRID METHODS

LONG CHEN

We give a short introduction to multigrid methods for solving the linear algebraic equation arising from the discretization of Poisson equation in one dimension.

1. An Algebraic Equation of Two Point Boundary Value Problems

We consider the discretization of Poisson equation with homogenous Dirichlet boundary condition in one dimension:

$$-u'' = f, \quad x \in (0,1) \qquad u(0) = u(1) = 0. \tag{1}$$

For a positive integer N, chose a uniform grid, denoted by $\mathcal{T}_h$, of the interval $[0,1]$ as follows:

$$0 = x_0 < x_1 < \dots x_N < x_{N+1} = 1, \quad x_j = jh,\ j = 0 : N+1,$$

where $h = 1/(N+1)$ is the length of each subinterval. For finite difference methods, we consider vector $\boldsymbol{v} = (v_1, \dots, v_N)^\intercal$ where $v_i \approx v(x_i)$ for $i = 1 : N$. The boundary nodes ($i = 0, N+1$) are excluded due to the homogenous Dirichlet boundary condition. We can connect the function value at grid points to get a piecewise linear function $v = \sum_{i=1}^N v_i \phi_i$, where ϕ_i is the hat basis function at x_i for $i = 1 : N$.

The algebraic system of a scaling of finite element or finite difference discretization is

$$\boldsymbol{A}\boldsymbol{u} = \boldsymbol{b}, \tag{2}$$

where

$$\boldsymbol{A} = \operatorname{diag}(-1, 2, -1), \quad \boldsymbol{b} = (b_i),\ b_i = h^2 f(x_i).$$

Due to the special structure of the matrix $\boldsymbol{A}$, we can write out eigenvalues and eigenvectors of $\boldsymbol{A}$ explicitly. Recall that the eigenvalues and eigenfunctions for $-u'' = \lambda u$ in $(0,1)$ with $u(0) = u(1) = 0$ are

$$\lambda_k = k^2, \quad \xi^k = \sin(k\pi x), \quad k = 1, 2, 3, \dots$$

In continuous level, $\lambda_{\min} = 1$ and $\lambda_k \to \infty$. In the discrete level, the $N \times N$ matrix $\boldsymbol{A}$ has only finite N eigenvalues. Therefore infinite many eigen-pairs (for $k > N$) cannot be preserved in the discretization. With h^2 rescaling, $\lambda_{\min} \to 0$ and $\lambda_{\max} \leq 4$. To better describe the eigenvalues of $\boldsymbol{A}$, we introduce a uniform grid of the angle interval $(0, \pi)$

$$0 = \theta_0 < \theta_1 < \dots \theta_N < \theta_{N+1} = \pi, \quad \theta_k = k\pi h,\ k = 0 : N+1,$$

The discrete eigenvectors will the nodal interpolation I_h of the continuous one to the uniform grid $\{x_j\}$ and eigenvalues will be a function defined on the uniform grid $\{\theta_k\}$. Here for a continuous function u, $I_h u(x_i) = u(x_i)$, for $i = 0, \dots, N+1$, i.e. $I_h u$ is a discrete function matching the function values at grid points.

Date: Latest update on November 13, 2020.

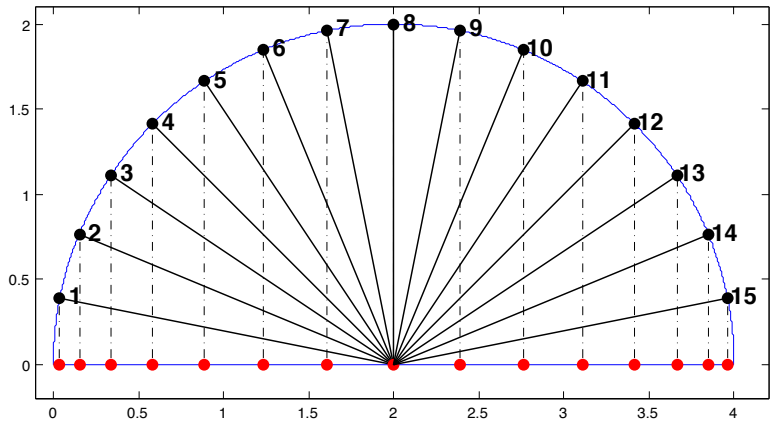


FIGURE 1. Eigenvalues of matrix $\boldsymbol{A}$ with $N = 15$.

Proposition 1.1. *If $\boldsymbol{A} = \operatorname{diag}(b, a, b)$ is an $N \times N$ tri-diagonal matrix, then the eigenvalue of $\boldsymbol{A}$ is: for $k = 1, \cdots, N$*

$$\lambda_k = a + 2b\cos\theta_k$$

and its corresponding eigenvector is:

$$\boldsymbol{\xi}^k = \sqrt{2}\, I_h \sin(k\pi x).$$

Proof. It can be easily verified by the trigonometric identity

$$\sin((j-1)\theta) + \sin((j+1)\theta) = 2\cos(\theta)\sin(i\theta).$$

□

It is interesting to note that eigenvectors are independent of a and b. Applying to the special case that $a = 2$ and $b = -1$, we get

$$\lambda_k(\boldsymbol{A}) = 2(1 - \cos\theta_k) = 4\sin^2\frac{\theta_k}{2}. \tag{3}$$

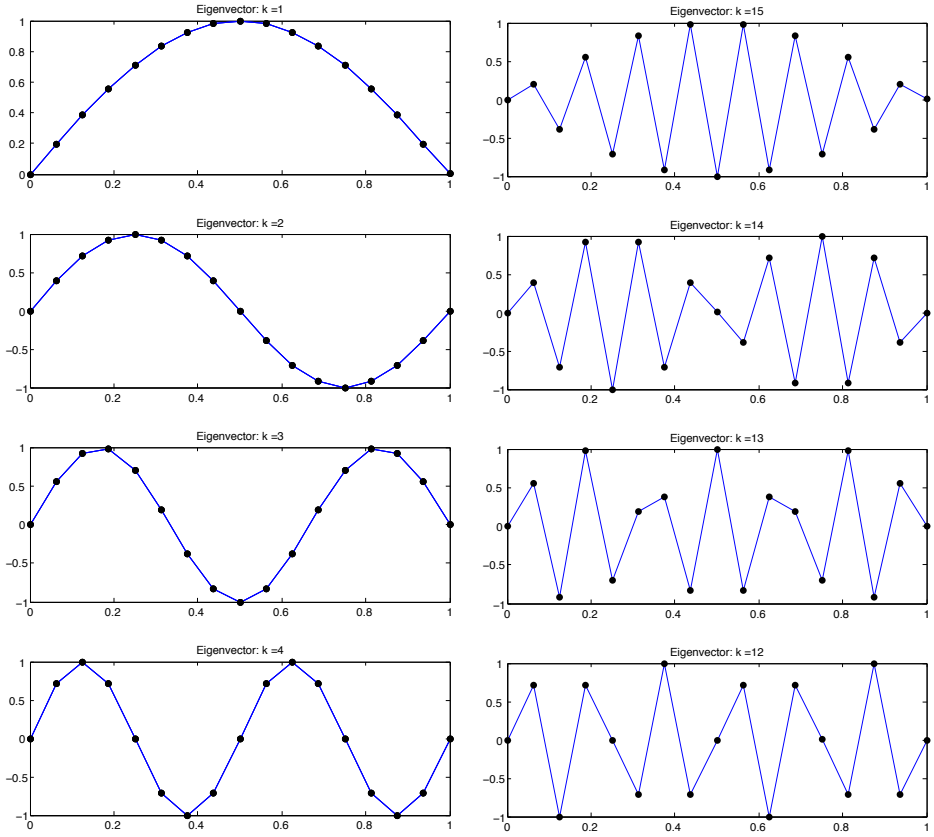
Notice that $\lambda_1 = \mathcal{O}(h^2)$ and $\lambda_N = \mathcal{O}(1)$, see Fig 1, and therefore $\kappa(\boldsymbol{A}) = \mathcal{O}(h^{-2})$, i.e., the matrix $\boldsymbol{A}$ is ill conditioned. For the finite difference method, the corresponding matrix is $\boldsymbol{A}/h^2$ and for finite element method $\boldsymbol{A}/h$. The scaling will introduce a scaling of all eigenvalues but not change the condition number.

Exercise 1.2. Apply Proposition 1.1 to the mass matrix $M = (m_{ij})$ with $m_{ij} = \int_0^1 \phi_i\phi_j \,\mathrm{d}x$ and conclude M is well conditioned. □

Exercise 1.3. Figure out the eigenvalues and eigenvectors for the Neumann problem $-u'' = \lambda u$ in $(0,1)$ with $u'(0) = u'(1) = 0$. □

The integer k in the function $\sin(k\pi x)$ is called *frequency*. For a uniform grid of $[0,1]$ with length $h = 1/(N+1)$, the range of k is $1, \ldots, N$. Recall that in the continuous level, there are infinite many frequency of all eigenfunctions. The frequency higher than N cannot be seen in the grid $\mathcal{T}_h$. This phenomenon is called the aliasing.

To illustrate the aliasing phenomenon, we consider a coarse grid with size $2h$. When restricted to the coarse grid $\mathcal{T}_{2h}$, some frequency in the fine grid $\mathcal{T}_h$ will be aliased with the



(a) Low frequency (b) High frequency

FIGURE 2. Some eigenvectors of A for $h = 1/16$. (a) Low frequency part which can be preserved on the coarse grid. (b) High frequency part which will be aliasing with the low frequency on the coarse grid.

other. Specifically, $I_{2h}(\boldsymbol{\xi}^k) = I_{2h}(-\boldsymbol{\xi}^{k'})$ for $k = 1 : N$ and $k' = N + 1 - k$, which can be verified as follows

$$I_{2h}\sin(k\pi x) = (\sin(k\pi j2h))_j = (\sin(j2\theta_k))_j$$
$$I_{2h}\sin(k'\pi x) = (\sin(k'\pi j2h))_j = (\sin(j2\theta_{k'}))_j = -(\sin(j2\theta_k))_j.$$

since $\theta_k + \theta_{k'} = \pi$ by the symmetry of the uniform grid of angles. One can also verify the aliasing by connecting coarse grid values in Fig. 1.

In general the grid $\mathcal{T}_h$ cannot see the frequency higher than $1/h$. So the coarser grid $\mathcal{T}_{2h}$ can only see the frequency less than $1/(2h)$. The frequency which can be only captured by the fine grid $\mathcal{T}_h$ but not $\mathcal{T}_{2h}$,

$$\frac{1}{2}(N+1) \le k \le N,$$

will be called *high frequency* (relative to $\mathcal{T}_{2h}$). We shall show the classic iteration methods will smooth out the high frequency part of the error very quickly while leave the low frequency part decay very slowly.

2. Smoothing Property of the Richardson Iteration

As an illustrative example, we apply the simplest iterative method, Richardson iteration, to solve (2). Recall that one iteration of Richardson method is

$$\boldsymbol{u}_{m+1} = \boldsymbol{u}_m + \alpha(\boldsymbol{b} - \boldsymbol{A}\boldsymbol{u}_m).$$

The error equation is

$$\boldsymbol{u} - \boldsymbol{u}_{m+1} = (I - \alpha\boldsymbol{A})(\boldsymbol{u} - \boldsymbol{u}_m)$$

and therefore

$$\|\boldsymbol{u} - \boldsymbol{u}_{m+1}\|_{\boldsymbol{A}} \leq \rho(I - \alpha\boldsymbol{A})\|\boldsymbol{u} - \boldsymbol{u}_m\|_A.$$

When $\alpha \in (0, 2/\lambda_{\max}(\boldsymbol{A}))$, the method is convergent and when $\alpha = 2/(\lambda_{\min}(\boldsymbol{A}) + \lambda_{\max}(\boldsymbol{A}))$ it achieves the optimal rate

$$\rho = \frac{\kappa(\boldsymbol{A}) - 1}{\kappa(\boldsymbol{A}) + 1} \leq 1 - Ch^2.$$

It means that the norm of the error, as summation of squares of all components, will decay with a slow rate $1 - Ch^2$.

We now apply refined analysis to get different decay rate for different frequency. First we change the coordinate. The set of all eigenvectors of $\boldsymbol{A}$ will form a basis of $\mathbb{R}^N$ which is orthogonal in both $(\cdot,\cdot)$ and $(\cdot,\cdot)_A$ inner products. Expand $\boldsymbol{e}_0 = \boldsymbol{u} - \boldsymbol{u}_0$ as

$$\boldsymbol{e}_0 = \sum_{k=1}^{N} c_k \boldsymbol{\xi}^k.$$

Then

$$\boldsymbol{e}_m = \boldsymbol{u} - \boldsymbol{u}_m = (I - \alpha\boldsymbol{A})^m \boldsymbol{e}_0 = \sum_{k=1}^{N} c_{m,k}\boldsymbol{\xi}^k,$$

where

$$c_{m,k} = (1 - \alpha\lambda_k)^m c_k.$$

Since eigenvectors are also A-orthogonal, we have

$$\|\boldsymbol{e}_m\|_A = \left(\sum_{k=1}^{N} c_{m,k}^2 \|\boldsymbol{\xi}^k\|_A^2\right)^{1/2}, \quad \|\boldsymbol{e}_0\|_A = \left(\sum_{k=1}^{N} c_k^2 \|\boldsymbol{\xi}^k\|_A^2\right)^{1/2}.$$

The coefficient of the k-th component decays with rates

$$|c_{m,k}| \leq (1 - \alpha\lambda_k)^m |c_k|.$$

We choose $\alpha = 1/4$ to simplify the rate as

$$\rho_k = |1 - \alpha\lambda_k| = \frac{1}{2}(1 + \cos\theta_k).$$

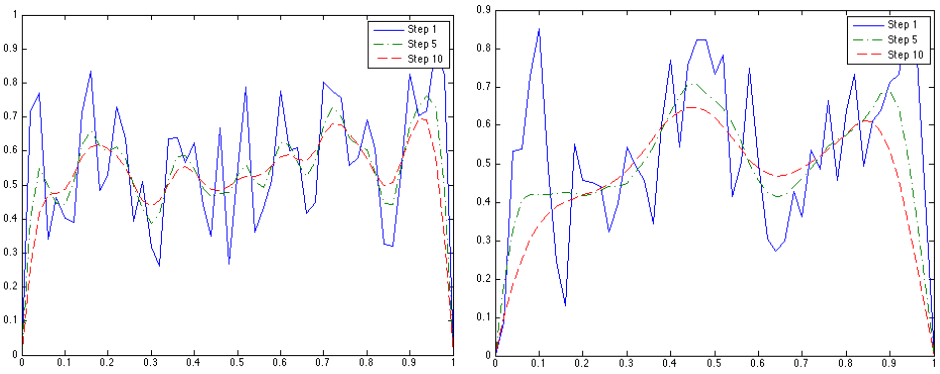
From the graph of ρ_k, see Fig 2 (a), it is easy to see that

$$\rho_1 \eqsim 1 - Ch^2,$$

but

$$\rho_N \leq Ch^2, \quad \text{and} \quad \rho_{(N+1)/2} = 1/2.$$

This means that *high frequency components get damped very quickly*, which is known smoothing property, while the low frequency converges very slowly.

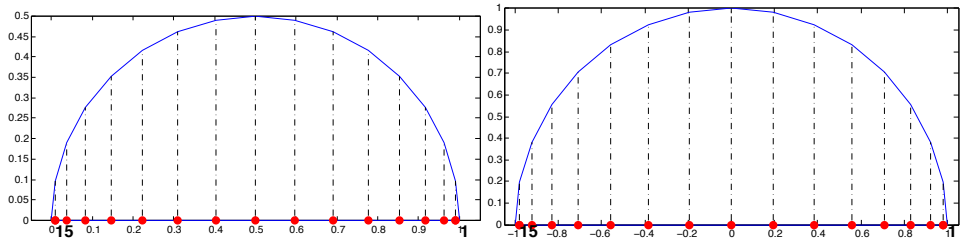


(a) Richardson iteration $\alpha = 1/4$ (b) Gauss-Seidel iteration

FIGURE 3. Smoothing effect of classic iterative methods.

Gauss-Seidel method, as a better iterative method, has the same affect. Indeed it is a better smoother than the Richardson method. Interesting enough, Jacobi method, corresponding to $\alpha = 1/2$, does not have a smoothing property, see Fig. 4 (b). We use the same example to compute the rate for $\alpha = 1/2$:

$$\rho_1 = \rho_N \approx 1 - Ch^2.$$



(a) Richardson iteration: $\alpha = 1/4$ (b) Richardson iteration: $\alpha = 1/2$

FIGURE 4. Contraction rate of the Richardson iteration for different parameters

Exercise 2.1. Although Jacobi method will not damp the high frequency, due to the shift, the rate of ρ_k is $\mathcal{O}(h)$ for k near $N/2$. Recall that for $\alpha = 1/4$, the rate is only $1/2$ for the same range of frequency. One can then apply two Richardson iterations consecutively with different parameters to acheive a better smoothing effect. In general, one can use *Chebyshev acceleration* technique

$$(I - \alpha_\ell A) \cdots (I - \alpha_1 A)$$

and chose optimal ℓ-parameters by the following optimization problem:

$$\min_{\alpha_i \in \mathbb{R}, i=1,\ldots,l} \left\{ \max_{\lambda \in [\lambda_{N/2}(A), \lambda_N(A)]} |(I - \alpha_\ell \lambda) \cdots (I - \alpha_1 \lambda)| \right\}. \tag{4}$$

Figure out an approximated solution for $\ell = 1, 2, 3$. □

3. Multigrid Methods

The multigrid methods is based on the following two observation

- High frequency will be damped by a smoother.
- Low frequency can be approximated well by a coarse grid.

A crucial idea is that part of low-frequency errors on a fine mesh becomes the high-frequency errors on a coarser mesh. For the coarse grid problem, we can apply the smoothing and the separation of scales again. Recursively application of the smoothing to each level results in the classical formulation of multigrid.

We present the following recursive subroutine of a multigrid method below.

```
1  function e = MG(r,J,mu)
2  % r: residual;  J: level; mu: smoothing steps
3  if J == 1  % coarsest level: exact solve
4      e = A{J}\r;
5      return
6  end
7  e = 0;
8  % Presmoothing
9  for i = 1:mu
10     e = e + R(r-A{J}*e);
11 end
12 % Restriction
13 rc = Res(r-A{J}*e);
14 % Coarse grid correction
15 ec = MG(rc,J-1,mu);
16 if W-cycle
17     ec = ec + MG(rc-A{J}*ec,J-1,mu);  % W-cycle
18 end
19 % Prolongation
20 e = e + Pro(ec);
21 % Postsmoothing
22 for i = 1:mu
23     e = e + R'(r-A{J}*e);
24 end
```

The operator (or a subroutine) `R` in the pre-smoothing can be chosen as the Richardson or the Gauss-Seidel iteration and in the post-smoothing step we use its transpose `R'` to make the whole iteration symmetric. There are two subroutine `Res` and `Pro` to connect the fine level and coarse level which will be discussed in detail in the Programming of Multigrid Methods. The function `e = MG(r,*,*)` suggests that `mg` cycle is used to solve the residual equation $Ae = r$ and will be used as an iterator in the residual-correction form of the iterative method

$$u_{k+1} = u_k + MG(f - Au_k).$$

Due to the linearity of the iteration, we can also formulate a direct update form of multigrid method `u = MG(u,f,J,mu)`; see Programming of Multigrid Methods.

The above MG is named V-cycle. If we apply one more coarse grid corrections using MG (line 16 - 18), we will obtain W-cycle. The two typical MG cycles are illustrated in the following figure.

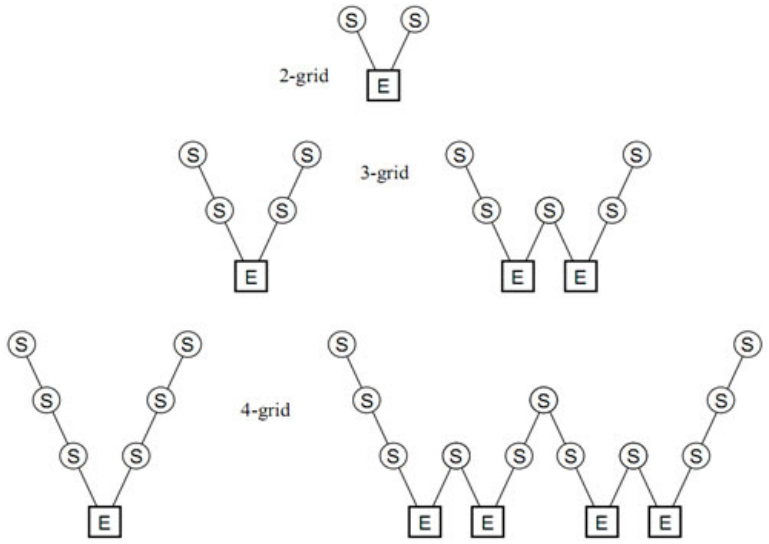


FIGURE 5. Multigrid Cycles. Left: V-cycle. Right: W-cycle

4. INTERPRETATION AS SUCCESSIVE SUBSPACE CORRECTION METHOD

We can interpret the multigrid method as a special case of the successive subspace correction method by choosing a multilevel decomposition of the space:

$$\mathbb{V}_1 \subset \mathbb{V}_2 ... \subset \mathbb{V}_J = \mathbb{V},$$

and a space decomposition

$$\mathbb{V} = \sum_{i=1}^{J} \mathbb{V}_i. \tag{5}$$

Denoted by $N_i = \dim \mathbb{V}_i$ and in practice $N_i = \gamma N_{i-1}$ for a factor $\gamma > 1$. For example, spaces based on a sequence of nested meshes in $\mathbb{R}^d$, the factor $\gamma \approx 2^d$.

Introduce operators

- $I_i : \mathbb{V}_i \hookrightarrow \mathbb{V}$ the natural inclusion;
- $Q_i : \mathbb{V} \mapsto \mathbb{V}_i$ the projection in the inner product $(\cdot, \cdot)$;
- $A_i : \mathbb{V}_i \mapsto \mathbb{V}_i$ the restriction of A on the subspace $\mathbb{V}_i \times \mathbb{V}_i$;
- $R_i : \mathbb{V}_i \mapsto \mathbb{V}_i$ an approximation of A_i^{-1} which is often called a smoother.

By definition

$$(Q_i^\intercal v_i, v) = (v_i, Q_i v) = (v_i, v) = (I_i v_i, v) \qquad \forall\, v_i \in \mathbb{V}_i,\ v \in \mathbb{V},$$

therefore $Q_i^\intercal$ coincides with the natural inclusion I_i or equivalently $I_i^\intercal = Q_i$. Then the restriction of A on subspaces is $A_i = I_i^\intercal A I_i = Q_i A Q_i^\intercal$.

Given the current iteration u_k, one step of the successive (multiplicative) subspace correction (SSC) method is : let

$$v^1 = u_k, \quad v^{i+1} = v^i + I_i R_i I_i^\intercal (f - A v^i), i = 1, \ldots, J, \quad u_{k+1} = v^{J+1}. \tag{6}$$

That is: for $i = 1, 2, \ldots, J$, we solve $A_i e_i = r_i$ with $r_i = Q_i(f - Av^i)$ approximately using a smoother $e_i = R_i r_i$ and update $v^{i+1} = v^i + e_i$. The new approximation is $u_{k+1} = v^J$. The procedure is illustrated in the following figure.

$$r \to \mathbb{V}_1 \to e \to r \to \mathbb{V}_2 \to e \to r \to \mathbb{V}_3 \to e.$$

FIGURE 6. Successive subspace correction methods

When the subspaces are nested, we do not need to return to the finest space in each subspace. Suppose $r_i = I_i^\intercal(r - Ae_{\text{old}})$ in the subspace $\mathbb{V}_i$ is known, and the correction e_i is used to update $e_{\text{new}} = e_{\text{old}} + e_i$. We can compute r_{i-1} by the recrusion:

$$\begin{aligned} r_{i-1} &= Q_{i-1}(r - Ae_{\text{new}}) \\ &= Q_{i-1}Q_i(r - Ae_{\text{old}} - Ae_i) \\ &= Q_{i-1}(r_i - Q_i A Q_i^\intercal e_i) \\ &= Q_{i-1}(r_i - A_i e_i). \end{aligned}$$

Here in the second step, we make use of the nestedness property $\mathbb{V}_{i-1} \subset \mathbb{V}_i$ to write $Q_{i-1} = Q_{i-1}Q_i$. Similarly the correction step can be also done accumulatively. Let us rewrite the correction as

$$e = e_J + I_{J-1}e_{J-1} + \ldots + I_1 e_1.$$

The correction can be computed by the recrusion

$$e_i = e_i + I_{i-1}^i e_{i-1}, \quad i = 2 : J$$

Therefore only the prolongation and restriction operators between consecutive levels are needed. The cost at each level is reduced to $\mathcal{O}(N_i)$ and the total cost is $\mathcal{O}(N)$ as $\sum_i N_i = \mathcal{O}(N)$.

From this point of view, SSC on a nested space decomposition will result in a V-cycle multigrid method. This interpretation of multigrid methods is not only helpful for the convergence analysis but also results in the following non-recursive implementation of multigrid V-cycle.

```
function e = Vcycle(r,J)
ri = cell(J,1); ei = cell(J,1);
ri{J} = r;
for i = J:-1:2
    ei{i} = R{i}*ri{i};    % pre-smoothing: one step
    ri{i-1} = Res{i-1}*(ri{i}-Ai{i}*ei{i}); % update and restrict residual
end
ei{1} = Ai{1}\ri{1}; % exact solver in the coarsest level
for i = 2:J
    ei{i} = ei{i} + Pro{i}*ei{i-1};  % prolongation and correction
    ei{i} = ei{i} + R{i}'*(ri{i}-Ai{i}*ei{i}); % post-smoothing: one step
end
e = ei{J};
```

We refer to Subspace Correction Method and Auxiliary Space Method for detailed explanation and convergence analysis of subspace correction methods and Programming of Multigrid Methods for implementation.