8

Finite Element Method and Multigrid method in Convolution

8.1 A one dimensional problem

8.1.1 Two-point boundary problems and finite element discretization

Define the functional space

$$V = \{v : [0,1] \to R,\ v \text{ is continuous and } v(0) = v(1) = 0\}.$$

Given any $f : [0,1] \to R$, consider

$$J(v) = \frac{1}{2}\int_0^1 |v'|^2 dx - \int_0^1 fv dx.$$

Find $u \in V$ such that

$$u = \arg\min_{v\in V} J(v) \tag{8.1}$$

Theorem 18. *Problem* (8.1) *is equivalent to: Find $u \in V$ such that*

$$\begin{cases} -u'' = f, & 0 < x < 1, \\ u(0) = u(1) = 0. \end{cases} \tag{8.2}$$

Proof. For any $v \in V, t \in R$, let $g(t) = J(u + tv)$. Since $u = \arg\min_{v\in V} J(v)$ means $g(t) \geq g(0)$. Hence, for any $v \in V$, 0 is the global minimum of the function $g(t)$. Therefore $g'(0) = 0$ implies

$$\int_0^1 u'v'dx = \int_0^1 fvdx \quad \forall v \in V.$$

By integration by parts, which is equivalent to

$$\int_0^1 (-u'' - f)vdx = 0 \quad \forall v \in V.$$

It can be proved that the above identity holds if and only if $-u'' = f$ for all $x \in (0,1)$. Namely u satisfies (8.10). □

Let V_h be finite element space and $\{\varphi_1, \varphi_2, \cdots \varphi_n\}$ be a nodal basis of the V_h (see Section 5.2). Let $\{\psi_1, \psi_2, \cdots, \psi_n\}$ be a dual basis of $\{\varphi_1, \varphi_2, \cdots \varphi_n\}$, namely $(\varphi_i, \psi_j) = \delta_{ij}$.

$$J(v_h) = \frac{1}{2}\int_0^1 |v_h'|^2 dx - \int_0^1 f v_h dx. \tag{8.3}$$

Let

$$v_h = \sum_{i=1}^n v_i \varphi_i,$$

then

$$J(v_h) = I(v) = \frac{1}{2} v^T A * v - b^T v$$

and

$$\nabla I(v) = A * v - b.$$

At the same time, let $u_h = \displaystyle\sum_{i=1}^n \mu_i \varphi_i$,

$$u_h = \arg\min_{v_h \in V_h} J(v_h) \Leftrightarrow \mu = \arg\min_{v \in R^n} I(v) \tag{8.4}$$

And u_h solves the problem: Find $u_h \in V_h$

$$\frac{d}{dt} J(u_h + t v_h)|_{t=0} = a(u_h, v_h) - \langle f, v_h \rangle = 0 \quad \forall v_h \in V_h. \tag{8.5}$$

where

$$a(u_h, v_h) = \int_0^1 u_h' v_h' dx.$$

which is equivalent to solving $\underset{\sim}{A}\mu = b$, where $\underset{\sim}{A} = (a_{ij})_{ij}^n$ and $a_{ij} = a(\varphi_j, \varphi_i)$ and $b_i = \int_0^1 f \varphi_i dx$. Namely

$$\frac{1}{h}\begin{pmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{pmatrix}\begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \\ \mu_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ \\ b_n \end{pmatrix}. \tag{8.6}$$

Which can be rewritten as

$$\frac{-\mu_{i-1} + 2\mu_i - \mu_{i+1}}{h} = b_i, \quad 1 \le i \le n, \quad \mu_0 = \mu_{n+1} = 0. \tag{8.7}$$

Using the convolution notation, (8.7) can be written as

$$A * \mu = b, \tag{8.8}$$

where $A = \frac{1}{h}[-1, 2, -1]$.

8.1.2 Spectrum properties of $A*$

We recall that λ is an eigenvalue of A and and $\xi \in R^N \setminus \{0\}$ is a corresponding eigenvector if

$$A * \xi = \lambda\xi.$$

Because of the special structure of A, all the N eigenvalues, λ_k, and the corresponding eigenvectors, $\xi^k = (\xi^k_j)$, of A can be obtained, for $1 \le k \le N$, as follows:

$$\lambda_k = \frac{4}{h}\sin^2\frac{k\pi}{2(N+1)}, \quad \xi^k_j = \sin\frac{kj\pi}{N+1}(1 \le j \le N).$$

Indeed, the relation $A\xi^k = \lambda_k\xi^k$ can be verified by following elementary trigonometric identities:

$$-\sin\frac{k(j-1)\pi}{N+1}+2\sin\frac{kj\pi}{N+1}-\sin\frac{k(j+1)\pi}{N+1} = 4\sin^2\frac{k\pi}{2(N+1)}\sin\frac{kj\pi}{N+1},$$

where $1 \le j \le N$. Actually, it is not very difficult to derive these formula directly (see appendix A).

To understand the behavior of these eigenvectors, let us take $N = 6$ and plot the linear interpolations of all these 6 eigenvectors. We immediately observe that each vector ξ^k corresponds to a given frequency, and larger k corresponds to higher frequency. As λ_k is increasing with respect to k, we can then say that a larger eigenvalue of A corresponds to a higher frequency eigenvector. From a numerical point of view, we say a relatively low frequency vector is relatively smoother whereas a high frequency vector is nonsmooth.

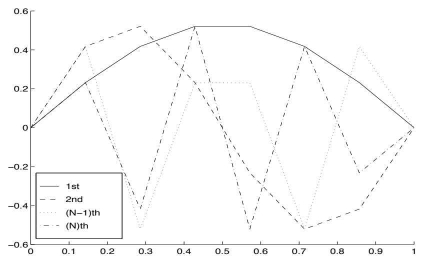

Fig. 8.1. The eigenvectors

We note that the set of eigenvectors $\xi^k = (\xi^k_j)$ forms an orthogonal basis of R^N. (This fact can be checked directly, or it also follows from the fact that the matrix A is symmetric and has N distinctive eigenvalues.) Therefore, any $\xi \in R^N$ can be expanded in terms of these eigenvectors:

$$\xi = \sum_{k=1}^{N} \alpha_k \xi^k.$$

This type of expansion is often called discrete Fourier expansion. The smoothness of the vector ξ has a lot to do with the relative size of the coefficients α_k. To see this numerically, let us again take $N = 4$ and consider the following two vectors

$$\xi = \sum_{k=1}^{4} 2^{1-k} \xi^k, \quad \sigma = \sum_{k=1}^{4} 2^{k-4} \xi^k.$$

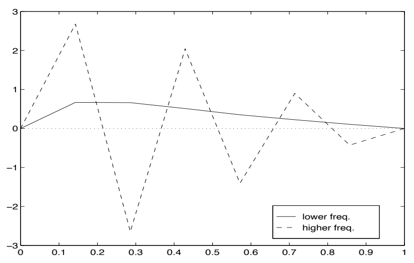

Fig. 8.2. Plots of ξ and σ. ξ-solid line; σ dashed line

The first vector ξ has larger coefficients in front of lower frequencies whereas the second vector σ has larger coefficients in front of higher frequencies. From Figure 8.2, it is easy to see how the smoothness of a vector depends on the relative size of its Fourier coefficients. We conclude that, in general, a vector with relatively small Fourier coefficients in front of the higher frequencies is relatively smooth and conversely, a vector with relatively large Fourier coefficients in front of the higher frequencies is relatively rough or nonsmooth.

8.1.3 Gradient descent method

Noting that $\nabla I(v) = A * v - b$ and applying the gradient descent method to solve problem (8.4), we obtain

$$\mu^{(l)} = \mu^{(l-1)} - \eta(A * \mu^{(l-1)} - b), \quad l = 1, \cdots, \nu.$$

After ν iterations of gradient descent method, we denote the solution as u_h^{ν}.

Consider the finite element discretization of Poisson equation in 1D: One very simple iterative method for (8.6) is the following gradient descent method

$$\mu^{(l)} = \mu^{(l-1)} + \eta(b - A * \mu^{(l-1)}),$$

or, for $j = 1 : N$,

$$\mu_j^{(l)} = \mu_j^{(l-1)} + \eta\Big(\beta_j - \frac{-\mu_{j-1}^{(l-1)} + 2\mu_j^{(l-1)} - \mu_{j+1}^{(l-1)}}{h}\Big),$$

where $\eta > 0$ is a positive parameter named learning rate.

It is not so difficult to properly choose η so that the above iterative scheme converges, namely for any initial guess μ^0, the sequence $(\mu^(l-1))$ generated by the above iteration converges to the exact solution μ of (8.6):

Note that

$$\mu = \mu + \eta(b - A * \mu),$$

we get

$$\mu - \mu^{(l)} = (I - \eta A*)(\mu - \mu^{(l-1)}),$$

or

$$\mu - \mu^{(l)} = (I - \eta A*)^l(\mu - \mu^0), l = 1, 2, 3, \cdots$$

As we known,

$$(I - \eta A*)^l \longrightarrow o$$

if and only if $\rho(I - \eta A*) < 1$. Here $\rho(B)$ is the spectral radius of matrix B. However, $\rho(I - \eta A*) < 1$ if and only if

$$0 < \text{all the eigenvalue of } A* < 2\eta^{-1}.$$

Thus, a necessary and sufficient condition for the convergence is the following

$$0 < \eta < \frac{2}{\rho(A*)}.$$

It is easy to see that (for example, $4/h$ is an upper bound of its row sums)

$$\frac{4}{h} > \rho(A*)$$

Therefore it is reasonable to make the following choice:

$$\eta = \frac{h}{4}$$

and the resulting algorithm is

$$\mu^{(l)} = \mu^{(l-1)} + \frac{h}{4}(b - A * \mu^{(l-1)}). \tag{8.9}$$

In the rest of this section, unless otherwise noted, we shall choose η as above for simplicity.

On Figure 8.3 the convergence history plot of the above gradient descent iterative method for typical application is shown. As we see, this iterative scheme converges very slowly.

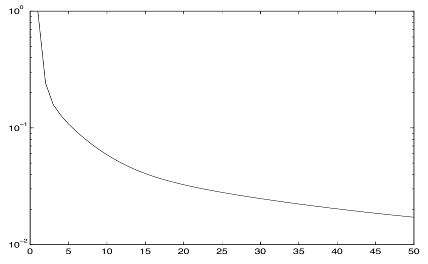

Fig. 8.3. A picture on the GD method convergence history

Our main goal is to find a way to speed up such kind of rather slowly convergent iterative scheme. To do that, we need to study its convergent property in more microscopic level. First of all, let us now take a careful look at the convergence history picture and make the following observation:

> *Observation 1.* The scheme converges rather fast in the very beginning but then slows down after a few steps. Overall, the method converges very slowly.

To further understand this phenomenon, let us plot the detailed pictures of the error functions in the first few iterations. After a careful look at these pictures, we have the following observation:

> *Observation 2.* The scheme not only converges fast in the first few steps, but also smooth out the error function very quickly.

In other words, the error function becomes a much smoother function after a few such simple iterations. This property of the itera-

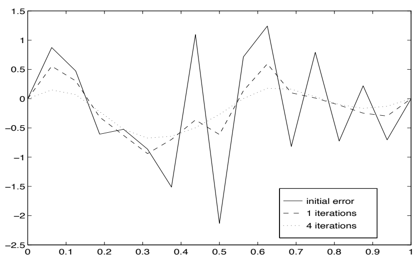

Fig. 8.4. The smoothing effect of the Richardson method

tive scheme is naturally called *a smoothing property* and an iterative scheme having this smoothing property is called a *smoother*.

The above two observations, especially the second one, concern the most important property of the simple gradient descent method that we can take advantage to get a much faster algorithm.

Example 7. Let $f(x) = \pi^2 \sin \pi x$. Consider

$$\begin{cases} -u'' = f, \ 0 < x < 1, \\ u(0) = u(1) = 0. \end{cases} \tag{8.10}$$

The true solution $u = \sin \pi x$. Given the partition with the grid points $x_i = \frac{i}{n+1}, i = 0, 1, \cdots, n+1$, then by finite element discretization, we obtain

$$A * \mu = b, A = \frac{1}{h}[-1, 2, -1]. \tag{8.11}$$

Use gradient descent method to solve (8.11) with random initial guess μ^0. Plot $\mu^\ell - \mu^0$ and $\|\mu^\ell - \mu^0\|$ for $\ell = 1, 2, 3$.

The gradient descent method can be written in terms of $S_0^\ell : \mathbb{R}^{N_\ell} \to \mathbb{R}^{N_\ell}$ satisfying

$$\mu^{(1)} = (S_0^\ell b) = \frac{h_\ell}{4} b, \tag{8.12}$$

for equation (8.7) with initial guess zero. If we apply this method twice, then

$$\mu^{(2)} = S_1^\ell(b) = S_0^\ell b + S_0^\ell(b - A_\ell * (S_0^\ell b)),$$

with element-wise form

$$\mu_i^{(2)} = \frac{h_\ell}{16}(b_{i-1} + 6b_i + b_{i+1}) \tag{8.13}$$

Then by the definition of convolution (6.17), we have

$$\mu^{(1)} = S_0^\ell * b \quad \mu^{(2)} = S_1^\ell * b. \tag{8.14}$$

with

$$S_0^\ell = \frac{h_\ell}{4}, \tag{8.15}$$

and

$$S_1^\ell = \frac{h_\ell}{16}[1, 6, 1]. \tag{8.16}$$

Hence we denote S_0^ℓ or S_1^ℓ as S^ℓ.

Now for any given $\mu^{(0)} = \tilde{\mu}^{(0)}$,

$$\begin{aligned} & j = 1, 2, \cdots, 2\nu \\ & \mu^{(j)} = \mu^{(j-1)} + S_0^\ell * (b - A * \mu^{(j-1)}) \end{aligned} \tag{8.17}$$

$$\Leftrightarrow$$

$$\begin{aligned} & j = 1, 2, \cdots, \nu \\ & \tilde{\mu}^{(j)} = \tilde{\mu}^{(j-1)} + S_1^\ell * (b - A * \tilde{\mu}^{(j-1)}) \end{aligned} \tag{8.18}$$

we obtain $\mu^{(j)} = \tilde{\mu}^{(j)}$ which means one step S_1^ℓ is equivalent to two steps of S_0^ℓ.

8.1.4 Convergence and smoothing properties of GD

Because of the extraordinary importance of this smoothing property, we shall now try to give some simple theoretical analysis. To do this, we make use of the eigenvalues and eigenvectors of the matrix A.

Fourier analysis for the gradient descent method

Our earlier numerical experiments indicate that the gradient descent method has a smoothing property. Based on our understanding of the relation between the smoothness and the size of Fourier coefficients, we can imagine that this smoothing property can be analyzed using the discrete Fourier expansion.

Let μ be the exact solution of (8.6) and $\mu^{(l)}$ the result of $l-th$ iteration from the gradient descent method (8.9). Then

$$\mu - \mu^{(l)} = (1 - \eta A*)(\mu - \mu^{(l-1)}) = \ldots = (1 - \eta A*)^l(\mu - \mu^{(0)}).$$

Consider the Fourier expansion of the initial error:

$$\mu - \mu^{(0)} = \sum_{k=1}^{N} \alpha_k \xi^k.$$

Then

$$\mu - \mu^{(l)} = \sum_{k=1}^{N} \alpha_k (I - \eta A*)^l \xi^k.$$

Note that $\eta = h/4$ and for any polynomial p

$$p(A*)\xi^k = p(\lambda_k)\xi^k,$$

we get

$$\mu - \mu^{(m)} = \sum_{k=1}^{N} \alpha_k (1 - \eta\lambda_k)^m \xi^k = \sum_{k=1}^{N} \alpha_k^{(m)} \xi^k$$

where

$$\alpha_k^{(m)} = \left(1 - \sin^2 \frac{k\pi}{2(N+1)}\right)^m \alpha_k.$$

Note that

$$1 - \sin^2 \frac{k\pi}{2(N+1)} = \cos^2 \frac{k\pi}{2(N+1)} = \sin^2(\frac{\pi}{2} - \frac{k\pi}{2(N+1)})$$

implies

$$\alpha_k^{(m)} = \alpha_k \sin^{2m} \frac{N+1-k}{N+1}\frac{\pi}{2} \le \alpha_k \left(\frac{N+1-k}{N+1}\frac{\pi}{2}\right)^{2m}$$

which, for k close to N, for example $k = N$, approaches to 0 very rapidly when $m \to \infty$. This means that high frequency components get damped very quickly. However, for k far away from N, for example $k = 1$, $\alpha_k^{(m)}$ approaches to 0 very slowly when $m \to \infty$.

This simple analysis clearly justifies the smoothing property that has been observed by numerical experiments.

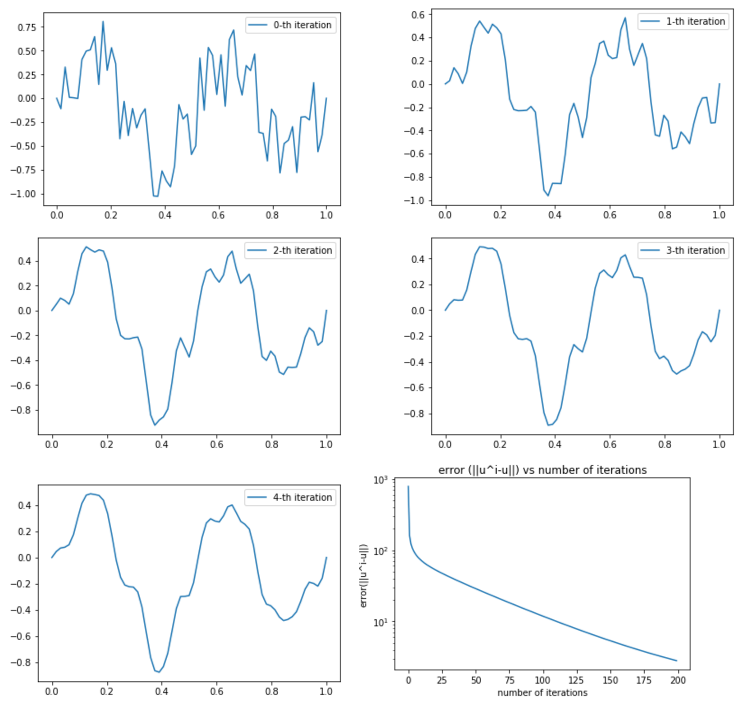

Fig. 8.5. u^0, u^1, u^2, u^3, u^4

An intuitive discussion

Both the gradient decsent and Gauss-Seidel methods are oftentimes called *local relaxation* methods. This name refers to the fact that what both of these algorithms do are just trying to correct the residual vector locally at one nodal point at a time (recall that $\mu_j \approx u(x_j)$). This local relaxation procedure is then effective to the error components that are local in nature. Incidently, the nonsmooth or high frequency component which oscillates across one or few grid points have a strong local feature. Therefore, it is not surprising the both gradient descent and Gauss-Seidel iteration can damp out these nonsmooth components more easily. These methods are very inefficient for relatively smoother components in the error since a smoother function is more globally related in nature.