Chapter 13

Rayleigh quotient and trace minimization

13.1 Introduction

In this chapter we restrict ourselves to the symmetric/Hermitian eigenvalue problem

$$A\mathbf{x} = \lambda M\mathbf{x}, \qquad A = A^*, \quad M = M^* > 0. \tag{13.1}$$

We want to exploit the property of the Rayleigh quotient that

$$\lambda_1 = \min_{\mathbf{x}\neq\mathbf{0}} \rho(\mathbf{x}) \qquad \rho(\mathbf{x}) = \frac{\mathbf{x}^* A\mathbf{x}}{\mathbf{x}^* M\mathbf{x}}, \tag{13.2}$$

which was proved in Theorem 2.30. The basic idea of Rayleigh quotient minimization is to construct a sequence $\{\mathbf{x}_k\}_{k=1,2,\ldots}$ such that $\rho(\mathbf{x}_{k+1}) < \rho(\mathbf{x}_k)$ for all k. The hope is that the sequence $\{\rho(\mathbf{x}_k)\}$ converges to λ_1 and by consequence the vector sequence $\{\mathbf{x}_k\}$ towards the corresponding eigenvector.

The procedure is as follows: For any given $\mathbf{x}_k$ let us choose a **search direction** $\mathbf{p}_k$, so that

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \delta_k \mathbf{p}_k. \tag{13.3}$$

The parameter δ_k is determined such that the Rayleigh quotient of the new iterate $\mathbf{x}_{k+1}$ becomes minimal,

$$\rho(\mathbf{x}_{k+1}) = \min_{\delta} \rho(\mathbf{x}_k + \delta\mathbf{p}_k). \tag{13.4}$$

We can write the Rayleigh quotient of the linear combination $\mathbf{x}_k + \delta\mathbf{p}_k$ of two (linearly independant) vectors $\mathbf{x}_k$ and $\mathbf{p}_k$ as

(13.5)

$$\rho(\mathbf{x}_k + \delta\mathbf{p}_k) = \frac{\mathbf{x}_k^* A\mathbf{x}_k + 2\delta\mathbf{x}_k^* A\mathbf{p}_k + \delta^2\mathbf{p}_k^* A\mathbf{p}_k}{\mathbf{x}_k^* M\mathbf{x}_k + 2\delta\mathbf{x}_k^* M\mathbf{p}_k + \delta^2\mathbf{p}_k^* M\mathbf{p}_k} = \frac{\begin{pmatrix}1\\ \delta\end{pmatrix}^* \begin{bmatrix}\mathbf{x}_k^* A\mathbf{x}_k & \mathbf{x}_k^* A\mathbf{p}_k\\ \mathbf{p}_k^* A\mathbf{x}_k & \mathbf{p}_k^* A\mathbf{p}_k\end{bmatrix} \begin{pmatrix}1\\ \delta\end{pmatrix}}{\begin{pmatrix}1\\ \delta\end{pmatrix}^* \begin{bmatrix}\mathbf{x}_k^* M\mathbf{x}_k & \mathbf{x}_k^* M\mathbf{p}_k\\ \mathbf{p}_k^* M\mathbf{x}_k & \mathbf{p}_k^* M\mathbf{p}_k\end{bmatrix} \begin{pmatrix}1\\ \delta\end{pmatrix}}.$$

This is the Rayleigh quotient associated with the generalized 2×2 eigenvalue problem

$$\begin{bmatrix}\mathbf{x}_k^* A\mathbf{x}_k & \mathbf{x}_k^* A\mathbf{p}_k\\ \mathbf{p}_k^* A\mathbf{x}_k & \mathbf{p}_k^* A\mathbf{p}_k\end{bmatrix} \begin{pmatrix}\alpha\\ \beta\end{pmatrix} = \lambda \begin{bmatrix}\mathbf{x}_k^* M\mathbf{x}_k & \mathbf{x}_k^* M\mathbf{p}_k\\ \mathbf{p}_k^* M\mathbf{x}_k & \mathbf{p}_k^* M\mathbf{p}_k\end{bmatrix} \begin{pmatrix}\alpha\\ \beta\end{pmatrix}. \tag{13.6}$$

The smaller of the two eigenvalues of (13.6) is the searched value $\rho_{k+1} := \rho(\mathbf{x}_{k+1})$ in (13.4) that minimizes the Rayleigh quotient. The corresponding eigenvector is normalized such that its first component equals one[1]. The second component of this eigenvector is $\delta = \delta_k$. Inserting the solution $[1, \delta_k]^*$ into the second line of (13.6) we obtain

$$\mathbf{p}_k^*(A - \rho_{k+1}M)(\mathbf{x}_k + \delta_k\mathbf{p}_k) = \mathbf{p}_k^*\mathbf{r}_{k+1} = 0. \tag{13.7}$$

So, the 'next' residual $\mathbf{r}_{k+1}$ is orthogonal to the actual search direction $\mathbf{p}_k$.

There are various ways how to choose the search direction $\mathbf{p}_k$. A simple way is to cycle through the coordinate vectors, a method that is called coordinate relaxation [3]. It cannot compete with the methods we discuss next; but it has some potential for parallelization.

13.2 The method of steepest descent

Let us make a detour to solving systems of equations

$$A\mathbf{x} = \mathbf{b}, \tag{13.8}$$

where A is symmetric/Hermitian positive definite. Let us define the functional

$$\varphi(\mathbf{x}) \equiv \frac{1}{2}\mathbf{x}^*A\mathbf{x} - \mathbf{x}^*\mathbf{b} + \frac{1}{2}\mathbf{b}^*A^{-1}\mathbf{b} = \frac{1}{2}(A\mathbf{x} - \mathbf{b})^*A^{-1}(A\mathbf{x} - \mathbf{b}). \tag{13.9}$$

The functional φ is minimized at the solution $\mathbf{x}_*$ of (13.8). (With the above definition of φ the minimum is actually zero.) The negative gradient of φ is

$$-\nabla\varphi(\mathbf{x}) = \mathbf{b} - A\mathbf{x} =: \mathbf{r}(\mathbf{x}). \tag{13.10}$$

It is nonzero except at $\mathbf{x}_*$. In the method of steepest descent [2, 3] a sequence of vectors $\{\mathbf{x}_k\}_{k=1,2,\ldots}$ is constructed such that the relation

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \delta_k\mathbf{p}_k \tag{13.3}$$

holds among any two consecutive vectors. The search direction $\mathbf{p}_k$ is chosen to be the negative gradient $-\nabla\phi(\mathbf{x}_k) = \mathbf{r}_k = \mathbf{b} - A\mathbf{x}_k$. This is the direction in which φ decreases the most. Setting $\mathbf{x}_{k+1}$ as in (13.3) we get

$$0 = \left.\frac{\partial\varphi(\mathbf{x}_{k+1})}{\partial\delta}\right|_{\delta=\delta_k} = \mathbf{p}_k^*(A\mathbf{x}_k - \mathbf{b}) + \delta_k\mathbf{p}_k^*A\mathbf{p}_k = -\mathbf{p}_k^*\mathbf{r}_k + \delta_k\mathbf{p}_k^*A\mathbf{p}_k.$$

Thus,

$$\delta_k = \frac{\mathbf{p}_k^*\mathbf{r}_k}{\mathbf{p}_k^*A\mathbf{p}_k} \tag{13.11}$$

which, for steepest descent, becomes

$$\delta_k = \frac{\mathbf{r}_k^*\mathbf{r}_k}{\mathbf{r}_k^*A\mathbf{r}_k} \tag{13.12}$$

Remark 13.1. Notice that

$$\mathbf{r}_{k+1} = \mathbf{b} - A\mathbf{x}_{k+1} = \mathbf{b} - A(\mathbf{x}_k + \delta_k\mathbf{p}_k) = \mathbf{r}_k - \delta_kA\mathbf{p}_k. \tag{13.13}$$

[1] The first component of this eigenvector is nonzero if it has a component in the direction of the 'smallest eigenvector'.

Therefore, from (13.11) we have

$$\mathbf{p}_k^* \mathbf{r}_{k+1} = \mathbf{p}_k^* \mathbf{r}_k - \delta_k \mathbf{p}_k^* A \mathbf{p}_k = 0, \tag{13.14}$$

which corresponds to (13.7) in the linear system case. □

For the eigenvalue problem we can proceed similarly by choosing $\mathbf{p}_k$ to be the negative gradient of the Rayleigh quotient ρ,

$$\mathbf{p}_k = -\mathbf{g}_k = -\nabla\rho(\mathbf{x}_k) = -\frac{2}{\mathbf{x}_k^* M \mathbf{x}_k}(A\mathbf{x}_k - \rho(\mathbf{x}_k) M \mathbf{x}_k).$$

Notice that $\mathbf{g}_k$ points in the *same* direction as the residual $\mathbf{r}_k$. (This is in contrast to the linear system case!) Since in eigenvalue problems we only care about directions we can equivalently set

$$\mathbf{p}_k = \mathbf{r}_k = A\mathbf{x}_k - \rho_k M \mathbf{x}_k, \qquad \rho_k = \rho(\mathbf{x}_k). \tag{13.15}$$

With this choice of search direction we immediately have from (13.7) that

$$\mathbf{r}_k^* \mathbf{r}_{k+1} = 0. \tag{13.16}$$

Not surprisingly, the method of steepest descent often converges slowly, as it does for linear systems. This happens if the spectrum is very much spread out, i.e., if the condition number of A relative to M is big.

13.3 The conjugate gradient algorithm

As with linear systems of equations a remedy against the slow convergence of steepest descent are conjugate search directions. So, let's again first look at linear systems [5]. There, we define the search directions as[2]

$$\mathbf{p}_k = -\mathbf{g}_k + \beta_k \mathbf{p}_{k-1}, \qquad k > 0. \tag{13.17}$$

where the coefficient β_k is determined such that $\mathbf{p}_k$ and $\mathbf{p}_{k-1}$ are **conjugate**, i.e.,

$$\mathbf{p}_k^* A \mathbf{p}_{k-1} = -\mathbf{g}_k^* A \mathbf{p}_{k-1} + \beta_k \mathbf{p}_{k-1}^* A \mathbf{p}_{k-1} = 0, \tag{13.18}$$

such that

$$\beta_k = \frac{\mathbf{g}_k^* A \mathbf{p}_{k-1}}{\mathbf{p}_{k-1}^* A \mathbf{p}_{k-1}}. \tag{13.19}$$

Premultiplying (13.17) by $\mathbf{g}_k^*$ gives

$$\mathbf{g}_k^* \mathbf{p}_k = -\mathbf{g}_k^* \mathbf{g}_k + \beta_k \mathbf{g}_k^* \mathbf{p}_{k-1} \overset{(13.14)}{=} -\mathbf{g}_k^* \mathbf{g}_k. \tag{13.20}$$

Furthermore, since $\mathbf{x}_k^* M \mathbf{x}_k = 1$ we have $\mathbf{r}_k = -\mathbf{g}_k$ and

$$\begin{aligned} 0 \overset{(13.14)}{=} \mathbf{g}_{k+1}^* \mathbf{p}_k &\overset{(13.17)}{=} -\mathbf{g}_{k+1}^* \mathbf{g}_k + \beta_k \mathbf{g}_{k+1}^* \mathbf{p}_{k-1} \\ &\overset{(13.13)}{=} -\mathbf{g}_{k+1}^* \mathbf{g}_k + \beta_k \mathbf{g}_k^* \mathbf{p}_{k-1} + \beta_k \delta_k \mathbf{p}_k^* A \mathbf{p}_{k-1}. \end{aligned}$$

[2] In linear systems the residual $\mathbf{r} = \mathbf{b} - A\mathbf{x}$ is defined as the negative gradient whereas in eigenvalue computations it is defined as $\mathbf{r} = A\mathbf{x} - \rho(\mathbf{x})M\mathbf{x}$, i.e., in the same direction as the gradient. To reduce the confusion we proceed using the gradient.

From (13.14) we have that $\mathbf{g}_k^* \mathbf{p}_{k-1} = 0$ and by construction of $\mathbf{p}_k$ and $\mathbf{p}_{k-1}$ being conjugate we have that $\mathbf{p}_k^* A \mathbf{p}_{k-1} = 0$. Thus,

$$\mathbf{g}_{k+1}^* \mathbf{g}_k = 0, \tag{13.21}$$

as with the method of steepest descent. Still in the case of linear systems, using these identities we find formulae equivalent to (13.19),

$$\beta_k = -\frac{\mathbf{g}_k^* A \mathbf{p}_{k-1}}{\mathbf{p}_{k-1}^* A \mathbf{p}_{k-1}} \overset{(13.13)}{=} -\frac{\mathbf{g}_k^* (\mathbf{g}_k - \mathbf{g}_{k-1})}{\mathbf{p}_{k-1}^* (\mathbf{g}_k - \mathbf{g}_{k-1})} \overset{(13.14)}{=} -\frac{\mathbf{g}_k^* (\mathbf{g}_k - \mathbf{g}_{k-1})}{-\mathbf{p}_{k-1}^* \mathbf{g}_{k-1}}$$

$$\overset{(13.20)}{=} \frac{\mathbf{g}_k^* (\mathbf{g}_k - \mathbf{g}_{k-1})}{\mathbf{g}_{k-1}^* \mathbf{g}_{k-1}} \tag{13.22}$$

$$\overset{(13.21)}{=} \frac{\mathbf{g}_k^* \mathbf{g}_k}{\mathbf{g}_{k-1}^* \mathbf{g}_{k-1}}. \tag{13.23}$$

The equivalent identities (13.19), (13.22), and (13.23) can be used to define β_k the most economic being (13.23).

We now look at how a conjugate gradient algorithm for the eigenvalue problem can be devised. The idea is straightforward. The algorithm differs from steepest descent by the choice of the search directions that are kept conjugate, i.e., consecutive search directions satisfy $\mathbf{p}_k^* A \mathbf{p}_{k-1} = 0$.

The crucial difference to linear systems stems from the fact, that the functional that is to be minimized, i.e., the Rayleigh quotient, is not quadratic anymore. (In particular, there is no finite termination property.) The gradient of $\rho(\mathbf{x})$ is

$$\mathbf{g} = \nabla \rho(\mathbf{x}_k) = \frac{2}{\mathbf{x}^* M \mathbf{x}} (A\mathbf{x} - \rho(\mathbf{x}) M \mathbf{x}).$$

So, in particular, the equation (13.14), does not hold:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \delta_k \mathbf{p}_k \quad \not\Longrightarrow \quad \mathbf{g}_{k+1} = \mathbf{g}_k + \delta_k A \mathbf{p}_k.$$

Therefore, in the context of nonlinear systems or eigenvalue problems the formulae in (13.19), (13.22), and (13.23) that define β_k are not equivalent anymore! Feng and Owen [4] extensively compared the three formulae and found that in the context of eigenvalue problems the last identity (13.23) leads to the fastest convergence. So, we opt for this equation and define the search directions according to

$$\begin{cases} \mathbf{p}_0 = -\mathbf{g}_0, & k = 0, \\ \mathbf{p}_k = -\mathbf{g}_k + \dfrac{\mathbf{g}_k^* M \mathbf{g}_k}{\mathbf{g}_{k-1}^* M \mathbf{g}_{k-1}} \mathbf{p}_{k-1}, & k > 0, \end{cases} \tag{13.24}$$

where we have given the formulae for the generalized eigenvalue problem $A\mathbf{x} = \lambda M \mathbf{x}$. The complete procedure is given in Algorithm 13.1

Convergence

The construction of Algorithm 13.1 guarantees that $\rho(\mathbf{x}_{k+1}) < \rho(\mathbf{x}_k)$ unless $\mathbf{r}_k = \mathbf{0}$, in which case $\mathbf{x}_k$ is the searched eigenvector. In general, i.e., if the initial vector $\mathbf{x}_0$ has a nonvanishing component in the direction of the 'smallest' eigenvector $\mathbf{u}_1$, convergence is toward the smallest eigenvalue λ_1. This assumption must also hold for vector iteration or the Lanczos algorithm.

Algorithm 13.1 The Rayleigh quotient algorithm

Let $\mathbf{x}_0$ be a unit vector, $\|\mathbf{x}_0\|_M = 1$.
$\mathbf{v}_0 := A\mathbf{x}_0, \quad \mathbf{u}_0 := M\mathbf{x}_0,$
$\rho_0 := \dfrac{\mathbf{v}_0^*\mathbf{x}_0}{\mathbf{u}_0^*\mathbf{x}_0},$
$\mathbf{g}_0 := 2(\mathbf{v}_0 - \rho_0\mathbf{u}_0)$
while $\|\mathbf{g}_k\| > tol$ **do**
  if $k = 1$ **then**
    $\mathbf{p}_k := -\mathbf{g}_{k-1}$;
  else
    $\mathbf{p}_k := -\mathbf{g}_{k-1} + \dfrac{\mathbf{g}_{k-1}^* M \mathbf{g}_{k-1}}{\mathbf{g}_{k-2}^* M \mathbf{g}_{k-2}} \mathbf{p}_{k-1}$;
  end if
  Determine the smallest Ritz value ρ_k and corresponding Ritz vector $\mathbf{x}_k$ of (A, M) in $\mathcal{R}([\mathbf{x}_{k-1}, \mathbf{p}_k])$
  $\mathbf{v}_k := A\mathbf{x}_k, \quad \mathbf{u}_k := M\mathbf{x}_k$
  $\rho_k := \mathbf{x}_k^*\mathbf{v}_k / \mathbf{x}_k^*\mathbf{u}_k$
  $\mathbf{g}_k := 2(\mathbf{v}_k - \rho_k\mathbf{u}_k)$
end while

Let

$$\mathbf{x}_k = \cos\vartheta_k \mathbf{u}_1 + \sin\vartheta_k \mathbf{z}_k =: \cos\vartheta_k \mathbf{u}_1 + \mathbf{w}_k, \tag{13.25}$$

where $\|\mathbf{x}_k\|_M = \|\mathbf{u}_1\|_M = \|\mathbf{z}_k\|_M = 1$ and $\mathbf{u}_1^* M \mathbf{z}_k = 0$. Then we have

$$\begin{aligned}\rho(\mathbf{x}_k) &= \cos^2\vartheta_k \lambda_1 + 2\cos\vartheta_k \sin\vartheta_k \mathbf{u}_1^* A \mathbf{z}_k + \sin^2\vartheta_k \mathbf{z}_k^* A \mathbf{z}_k \\ &= \lambda_1(1 - \sin^2\vartheta_k) + \sin^2\vartheta_k \rho(\mathbf{z}_k),\end{aligned}$$

or,

$$\rho(\mathbf{x}_k) - \lambda_1 = \sin^2\vartheta_k \left(\rho(\mathbf{z}_k) - \lambda_1\right) \le (\lambda_n - \lambda_1)\sin^2\vartheta_k. \tag{13.26}$$

As seen earlier, in symmetric eigenvalue problems, the eigenvalues are much more accurate than the eigenvectors.

Let us now suppose that the eigenvalues have already converged, i.e.,

$$\rho(\mathbf{x}_k) = \rho_k \cong \lambda_1,$$

while the eigenvectors are not yet as accurate as desired. Then we can write

$$\mathbf{r}_k = (A - \rho_k M)\mathbf{x}_k \cong (A - \lambda_1 M)\mathbf{x}_k = \sum_{j=1}^{n} (\lambda_j - \lambda_1) M \mathbf{u}_j \mathbf{u}_j^* M \mathbf{x}_k, \tag{13.27}$$

which entails $\mathbf{u}_1^* \mathbf{r}_k = 0$ since the first summand on the right of (13.27) vanishes. From (13.25) we have $\mathbf{w}_k = \sin\vartheta_k \mathbf{z}_k \perp_M \mathbf{u}_1$. Thus,

$$\begin{cases} (A - \lambda_1 M)\mathbf{w}_k = (A - \lambda_1 M)\mathbf{x}_k = \mathbf{r}_k \perp \mathbf{u}_1 \\ \mathbf{w}_k^* M \mathbf{u}_1 = 0 \end{cases} \tag{13.28}$$

If λ_1 is a simple eigenvalue of the pencil $(A; B)$ then $A - \lambda_1 M$ is a bijective mapping of $\mathcal{R}(\mathbf{u}_1)^{\perp_M}$ onto $\mathcal{R}(\mathbf{u}_1)^{\perp}$. If $\mathbf{r} \in \mathcal{R}(\mathbf{u}_1)^{\perp}$ then the equation

$$(A - \lambda_1 M)\mathbf{w} = \mathbf{r} \tag{13.29}$$

has a *unique* solution in $\mathcal{R}(\mathbf{u}_1)^{\perp_M}$.

So, close to convergence, Rayleigh quotient minimization does nothing else but solving equation (13.29) by the conjugate gradient algorithm. The convergence of the Rayleigh quotient minimization is determined by the condition number of $A-\lambda_1 M$ (as a mapping of $\mathcal{R}(\mathbf{u}_1)^{\perp_M}$ onto $\mathcal{R}(\mathbf{u}_1)^{\perp}$), according to the theory of conjugate gradients for linear system of equations. This condition number is

$$\kappa_0 = \mathcal{K}(A-\lambda_1 M)\Big|_{\mathcal{R}(\mathbf{u}_1)^{\perp_M}} = \frac{\lambda_n - \lambda_1}{\lambda_2 - \lambda_1}, \tag{13.30}$$

and the rate of convergence is given by

$$\frac{\sqrt{\kappa_0}-1}{\sqrt{\kappa_0}+1}. \tag{13.31}$$

A high condition number implies slow convergence. We see from (13.31) that the condition number is high if the distance of λ_1 and λ_2 is much smaller than the spread of the spectrum of $(A;B)$. This happens more often than not, in particular with FE discretizations of PDE's.

Preconditioning

In order to reduce the condition number of the eigenvalue problem we change

$$A\mathbf{x} = \lambda M\mathbf{x}$$

into

$$\tilde{A}\tilde{\mathbf{x}} = \tilde{\lambda}\tilde{M}\tilde{\mathbf{x}}, \tag{13.32}$$

such that

$$\kappa(\tilde{A}-\tilde{\lambda}_1\tilde{M}) \ll \kappa(A-\lambda_1 M). \tag{13.33}$$

To further investigate this idea, let C be a nonsingular matrix, and let $\mathbf{y} = C\mathbf{x}$. Then,

$$\rho(\mathbf{x}) = \frac{\mathbf{x}^* A\mathbf{x}}{\mathbf{x}^* M\mathbf{x}} = \frac{\mathbf{y}^* C^{-*} A C^{-1}\mathbf{y}}{\mathbf{y}^* C^{-*} M C^{-1}\mathbf{y}} = \frac{\mathbf{y}^*\tilde{A}\mathbf{y}}{\tilde{\mathbf{y}}^*\tilde{M}\mathbf{y}} = \tilde{\rho}(\mathbf{y}) \tag{13.34}$$

Thus,

$$\tilde{A}-\lambda_1\tilde{M} = C^{-*}(A-\lambda_1 M)C^{-1},$$

or, after a similarity transformation,

$$C^{-1}(\tilde{A}-\lambda_1\tilde{M})C = (C^*C)^{-1}(A-\lambda_1 M).$$

How should we choose C to satisfy (13.33)? Let us tentatively set $C^*C = A$. Then we have

$$(C^*C)^{-1}(A-\lambda_1 M)\mathbf{u}_j = A^{-1}(A-\lambda_1 M)\mathbf{u}_j = (I-\lambda_1 A^{-1}M)\mathbf{u}_j = \left(1-\frac{\lambda_1}{\lambda_j}\right)\mathbf{u}_j.$$

Note that

$$0 \le 1-\frac{\lambda_1}{\lambda_j} < 1.$$

Dividing the largest eigenvalue of $A^{-1}(A-\lambda_1 M)$ by the smallest *positive* gives the condition number

$$\kappa_1 := \kappa\left(A^{-1}(A-\lambda_1 M)\big|_{\mathcal{R}(\mathbf{u}_1)^{\perp_M}}\right) = \frac{1-\frac{\lambda_1}{\lambda_n}}{1-\frac{\lambda_1}{\lambda_2}} = \frac{\lambda_2}{\lambda_n}\frac{\lambda_n-\lambda_1}{\lambda_2-\lambda_1} = \frac{\lambda_2}{\lambda_n}\kappa_0. \tag{13.35}$$

If $\lambda_2 \ll \lambda_n$ then the condition number is much reduced. Further, κ_1 is bounded independently of n,

$$\kappa_1 = \frac{1-\lambda_1/\lambda_n}{1-\lambda_1/\lambda_2} < \frac{1}{1-\lambda_1/\lambda_2}. \tag{13.36}$$

So, with this particular preconditioner, κ_1 does not dependent on the choice of the mesh-width h in the FEM application.

The previous discussion suggests to choose C in such way that $C^*C \cong A$. C could, for instance, be obtained form an Incomplete Cholesky decomposition. We make this choice in the numerical example below.

Notice that the transformation $\mathbf{x} \longrightarrow \mathbf{y} = C\mathbf{x}$ need not be made explicitly. In particular, the matrices $\tilde{A}$ and $\tilde{M}$ must not be formed. As with the preconditioned conjugate gradient algorithm for linear systems there is an additional step in the algorithm where the *preconditioned residual* is computed, see Fig. 13.1 on page 248.

13.4 Locally optimal PCG (LOPCG)

The parameter δ_k in the RQMIN und (P)CG algorithms is determined such that

$$\rho(\mathbf{x}_{k+1}) = \rho(\mathbf{x}_k + \delta_k \mathbf{p}_k), \quad \mathbf{p}_k = -\mathbf{g}_k + \alpha_k \mathbf{p}_{k-1} \tag{13.37}$$

is minimized. α_k is chosen to make consecutive search directions conjugate. Knyazev [6] proposed to optimize both parameters, α_k and δ_k, at once.

$$\rho(\mathbf{x}_{k+1}) = \min_{\delta,\gamma} \rho(\mathbf{x}_k - \delta \mathbf{g}_k + \gamma \mathbf{p}_{k-1}) \tag{13.38}$$

This results in potentially smaller values for the Rayleigh quotient, as

$$\min_{\delta,\gamma} \rho\big(\mathbf{x}_k - \delta \mathbf{g}_k + \gamma \mathbf{p}_{k-1}\big) \le \min_{\delta} \big(\mathbf{x}_k - \delta(\mathbf{g}_k - \alpha_k \mathbf{p}_k)\big).$$

Hence, Knyazev coined the notion "locally optimal".

$\rho(\mathbf{x}_{k+1})$ in (13.38) is the minimal eigenvalue of the 3×3 eigenvalue problem

$$\begin{bmatrix} \mathbf{x}_k^* \\ -\mathbf{g}_k^* \\ \mathbf{p}_{k-1}^* \end{bmatrix} A[\mathbf{x}_k, -\mathbf{g}_k, \mathbf{p}_{k-1}] \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} = \lambda \begin{bmatrix} \mathbf{x}_k^* \\ -\mathbf{g}_k^* \\ \mathbf{p}_{k-1}^* \end{bmatrix} M[\mathbf{x}_k, -\mathbf{g}_k, \mathbf{p}_{k-1}] \begin{pmatrix} \alpha \\ \beta \\ \gamma \end{pmatrix} \tag{13.39}$$

We normalize the eigenvector corresponding to the smallest eigenvalue such that its first component becomes 1,

$$[1, \delta_k, \gamma_k] := [1, \beta/\alpha, \gamma/\alpha].$$

These values of δ_k and γ_k are the parameters that minimize the right hand side in (13.38). Then we can write

$$\mathbf{x}_{k+1} = \mathbf{x}_k - \delta_k \mathbf{g}_k + \gamma_k \mathbf{p}_{k-1} = \mathbf{x}_k + \delta_k \underbrace{(-\mathbf{g}_k + (\gamma_k/\delta_k)\mathbf{p}_{k-1})}_{=:\mathbf{p}_k} = \mathbf{x}_k + \delta_k \mathbf{p}_k. \tag{13.40}$$

We can consider $\mathbf{x}_{k+1}$ as having been obtained by a Rayleigh quotient minimization from $\mathbf{x}_k$ along $\mathbf{p}_k = -\mathbf{g}_k + (\gamma_k/\delta_k)\mathbf{p}_{k-1}$. Notice that this direction is needed in the next iteration step. (Otherwise it is not of a particular interest.)

```
function [x,rho,log] = rqmin1(A,M,x,tol,C)
%RQMIN1    [x,rho] = rqmin1(A,M,x0,tol,C)
%       cg-Rayleigh quotient minimization for the computation
%       of the smallest eigenvalue of  A*x = lambda*M*x,
%       A and M are symmetric, M spd.  x0 initial vector
%       C'*C preconditioner
%       tol: convergence criterium:
%            ||2*(C'*C)\(A*x - lam*M*x)|| < tol

% PA 16.6.2000

u = M*x;
q = sqrt(x'*u);
x = x/q; u = u/q;
v = A*x;
rho = x'*v;

k = 0; g = x; gnorm = 1;  log=[]; % Initializations

while gnorm > tol,
  k = k + 1;
  galt = g;
  if exist('C'),
    g = 2*(C\(C'\(v - rho*u)));   %  preconditioned gradient
  else
    g = 2*(v - rho*u);      %  gradient
  end
  if k == 1,
    p = -g;
  else
    p = -g + (g'*M*g)/(galt'*M*galt)*p;
  end

  [qq,ll] = eig([x p]'*[v A*p],[x p]'*[u M*p]);
  [rho,ii] = min(diag(ll));
  delta = qq(2,ii)/qq(1,ii);

  x = x + delta*p;
  u = M*x;
  q = sqrt(x'*u);
  x = x/q; u = u/q;
  v = A*x;
  gnorm = norm(g);
  if nargout>2, log = [log; [k,rho,gnorm]]; end
end
```

Figure 13.1: MATLAB code RQMIN: Rayleigh quotient minimization

```
function [x,rho,log] = lopcg(A,M,x,tol,C)
%RQMIN1 [x,rho] = lopcg(A,M,x0,tol,C)
%       Locally Optimal Proconditioned CG algorithm for
%       computing the smallest eigenvalue of  A*x = lambda*M*x,f
%       where A and M are symmetrisch, M spd.
%       x0 initial vektor
%       C'*C preconditioner
%       tol: stopping criterion:
%            (C'*C)\(A*x - lam*M*x) < tol

% PA 2002-07-3

n = size(M,1);
u = M*x;
q = sqrt(x'*u);
x = x/q; u = u/q;
v = A*x;
rho = x'*v;

k = 0;  gnorm = 1;  log=[]; % initializations

while gnorm > tol,
  k = k + 1;
  g = v - rho*u;        %  gradient
  gnorm = norm(g);
  if exist('C'),
    g = (C\(C'\g));   %  preconditioned gradient
  end
  if k == 1, p = zeros(n,0); end

  aa = [x -g p]'*[v A*[-g p]]; aa = (aa+aa')/2;
  mm = [x -g p]'*[u M*[-g p]]; mm = (mm+mm')/2;
  [qq,ll] = eig(aa,mm);
  [rho,ii] = min(diag(ll));
  delta = qq(:,ii);

  p = [-g p]*delta(2:end);
  x = delta(1)*x + p;
  u = M*x;
  q = sqrt(x'*u);
  x = x/q; u = u/q;
  v = A*x;
  if nargout>2, log = [log; [k,rho,gnorm]]; end
end
```

Figure 13.2: MATLAB code LOPCG: Locally Optimal Preconditioned Conjugate Gradient algorithm

13.5 The block Rayleigh quotient minimization algorithm (BRQMIN)

The above procedures converge *very* slowly if the eigenvalues are clustered. Hence, these methods should be applied only in **blocked** form.

Longsine and McCormick [8] suggested several variants for blocking Algorithm 13.1. See [1] for a recent numerical investigation of this algorithm.

13.6 The locally-optimal block preconditioned conjugate gradient method (LOBPCG)

In BRQMIN the Rayleigh quotient is minimized in the $2q$-dimensional subspace generated by the eigenvector approximations X_k and the search directions $P_k = -H_k + P_{k-1}B_k$, where the H_k are the preconditioned residuals corresponding to X_k and B_k is chosen such that the *block* of search directions is conjugate. Instead, Knyazev [6] suggests that the space for the minimization be augmented by the q-dimensional subspace $\mathcal{R}(H_k)$. The resulting algorithm is deemed 'locally-optimal' because $\rho(\mathbf{x})$ is minimized with respect to all available vectors.

Algorithm 13.2 The locally-optimal block preconditioned conjugate gradient method (LOBPCG) for solving $A\mathbf{x} = \lambda M\mathbf{x}$ with preconditioner N of [1]

Choose random matrix $X_0 \in \mathbb{R}^{n\times q}$ with $X_0^T M X_0 = I_q$. Set $Q := [\,]$.
Compute $(X_0^T A X_0)S_0 = S_0\Theta_0$ /* (Spectral decomposition) */
where $S_0^T S_0 = I_q$, $\Theta_0 = \mathrm{diag}(\vartheta_1, \ldots, \vartheta_q)$, $\vartheta_1 \le \ldots \le \vartheta_q$.
$X_0 := X_0 S_0$; $R_0 := AX_0 - MX_0\Theta_0$; $P_0 := [\,]$; $k := 0$.
while $\mathrm{rank}(Q) < p$ **do**
  Solve the preconditioned linear system $NH_k = R_k$
  $H_k := H_k - Q(Q^T M H_k)$.
  $\widetilde{A} := [X_k, H_k, P_k]^T A [X_k, H_k, P_k]$.
  $\widetilde{M} := [X_k, H_k, P_k]^T M [X_k, H_k, P_k]$.
  Compute $\widetilde{A}\widetilde{S}_k = \widetilde{M}\widetilde{S}_k\widetilde{\Theta}_k$ /* (Spectral decomposition) */
  where $\widetilde{S}_k^T\widetilde{M}\widetilde{S}_k = I_{3q}$, $\widetilde{\Theta}_k = \mathrm{diag}(\vartheta_1, \ldots, \vartheta_{3q})$, $\vartheta_1 \le \ldots \le \vartheta_{3q}$.
  $S_k := \widetilde{S}_k[\mathbf{e}_1, \ldots, \mathbf{e}_q]$, $\Theta := \mathrm{diag}(\vartheta_1, \ldots, \vartheta_q)$.
  $P_{k+1} := [H_k, P_k]\, S_{k,2}$; $X_{k+1} := X_k S_{k,1} + P_{k+1}$.
  $R_{k+1} := AX_{k+1} - MX_{k+1}\Theta_k$.
  $k := k + 1$.
  for $i = 1, \ldots, q$ **do**
    /* (Convergence test) */
    if $\|R_k\mathbf{e}_i\| < \mathrm{tol}$ **then**
      $Q := [Q, X_k\mathbf{e}_i]$; $X_k\mathbf{e}_i := \mathbf{t}$, with $\mathbf{t}$ a random vector.
      M-orthonormalize the columns of X_k.
    end if
  end for
end while

If $\mathbf{d}_j = [\mathbf{d}_{1j}^T, \mathbf{d}_{2j}^T, \mathbf{d}_{3j}^T]^T$, $\mathbf{d}_{ij} \in \mathbb{R}^q$, is the eigenvector corresponding to the j-th eigenvalue of (13.1) restricted to $\mathcal{R}([X_k, H_k, P_{k-1}])$, then the j-th column of X_{k+1} is the corresponding

Ritz vector

$$X_{k+1}\mathbf{e}_j := [X_k, H_k, P_{k-1}]\ \mathbf{d}_j = X_k\mathbf{d}_{1j} + P_k\mathbf{e}_j, \tag{13.41}$$

with

$$P_k\mathbf{e}_j := H_k\mathbf{d}_{2j} + P_{k-1}\mathbf{d}_{3j}.$$

Notice that P_0 is an empty matrix such that the eigenvalue problem in step (8) of the locally-optimal block preconditioned conjugate gradient method (LOBPCG), displayed in Algorithm 13.2, has order $2q$ only for $k = 0$.

The algorithm as proposed by Knyazev [6] was designed to compute just a few eigenpairs and so a memory efficient implementation was not presented. For instance, in addition to X_k, R_k, H_k, P_k, the matrices MX_k, MH_k, MP_k and AX_k, AH_k, AP_k are also stored. The resulting storage needed is prohibitive if more than a handful of eigenpairs are needed.

A more memory efficient implementation results when we iterate with blocks of width q in the space orthogonal to the already computed eigenvectors. The computed eigenvectors are stored in Q and neither MQ nor AQ are stored. Hence only storage for $(p + 10q)n + \mathcal{O}(q^2)$ numbers is needed.

Here, the columns of $[X_k, H_k, P_k]$ may become (almost) linearly dependent leading to ill-conditioned matrices $\widetilde{A}$ and $\widetilde{M}$ in step (9) of the LOBPCG algorithm. If this is the case we simply restart the iteration with random X_k orthogonal to the computed eigenvector approximations. More sophisticated restarting procedures that retain X_k but modify H_k and/or P_k were much less stable in the sense that the search space basis again became linearly dependent within a few iterations. Restarting with random X_k is a rare occurrence and in our experience, has little effect on the overall performance of the algorithm.

13.7 A numerical example

We again look at the determination the acoustic eigenfrequencies and modes in the interior of a car, see section 1.6.3. The computations are done with the finest grid depicted in Fig. 1.9. We compute the smallest eigenvalue of the problem with RQMIN and LOPCG, with preconditioning and without. The preconditioner we chose was the incomplete Cholesky factorization without fill-in, usually denoted IC(0). This factorization is implemented in the MATLAB routine `ichol`.

```
>> [p,e,t]=initmesh('auto');
>> [p,e,t]=refinemesh('auto',p,e,t);
>> [p,e,t]=refinemesh('auto',p,e,t);
>> p=jigglemesh(p,e,t);
>> [A,M]=assema(p,t,1,1,0);
>> whos
  Name      Size                    Bytes  Class

  A      1095x1095                  91540  double array (sparse)
  M      1095x1095                  91780  double array (sparse)
  e         7x188                   10528  double array
  p         2x1095                  17520  double array
  t         4x2000                  64000  double array

Grand total is 26052 elements using 275368 bytes

>> n=size(A,1);
```

```
>> R=ichol(A)';   % Incomplete Cholesky factorization
>> x0=rand(n,1)-.5;
>> tol=1e-6;
>> [x,rho,log0] = rqmin1(A,M,x0,tol);
>> [x,rho,log1] = rqmin1(A,M,x0,tol,R);
>> [x,rho,log2] = lopcg(A,M,x0,tol);
>> [x,rho,log3] = lopcg(A,M,x0,tol,R);
>> whos log*
  Name       Size                    Bytes  Class

  log0     346x3                      8304  double array
  log1     114x3                      2736  double array
  log2     879x3                     21096  double array
  log3     111x3                      2664  double array

Grand total is 4350 elements using 34800 bytes

>> L = sort(eig(full(A),full(M)));
>> format short e, [L(1) L(2) L(n)], format

ans =

  -7.5901e-13   1.2690e-02   2.6223e+02

>> k0= L(n)/L(2);
>> (sqrt(k0) - 1)/(sqrt(k0) + 1)

ans =

    0.9862

>> l0=log0(end-6:end-1,2).\log0(end-5:end,2);
>> l1=log1(end-6:end-1,2).\log1(end-5:end,2);
>> l2=log2(end-6:end-1,2).\log2(end-5:end,2);
>> l3=log3(end-6:end-1,2).\log3(end-5:end,2);
>> [l0 l1 l2 l3]

ans =

    0.9292    0.8271    0.9833    0.8046
    0.9302    0.7515    0.9833    0.7140
    0.9314    0.7902    0.9837    0.7146
    0.9323    0.7960    0.9845    0.7867
    0.9320    0.8155    0.9845    0.8101
    0.9301    0.7955    0.9852    0.8508

>> semilogy(log0(:,1),log0(:,3)/log0(1,3),log1(:,1),log1(:,3)/log1(1,3),...
   log2(:,1),log2(:,3)/log2(1,3),log3(:,1),log3(:,3)/log3(1,3),'LineWidth',2)
>> legend('rqmin','rqmin + prec','lopcg','lopcg + prec')
```

The convergence histories in Figure 13.3 for RQMIN and LOPCG show that preconditioning helps very much in reducing the iteration count.

In Figure 13.4 the convergence histories of LOBPCG for computing ten eigenvalues is shown. In 43 iteration steps all *ten* eigenvalues have converged to the desired accuracy ($\varepsilon = 10^{-5}$). Clearly, the iteration count has been decreased drastically. Note however, that each

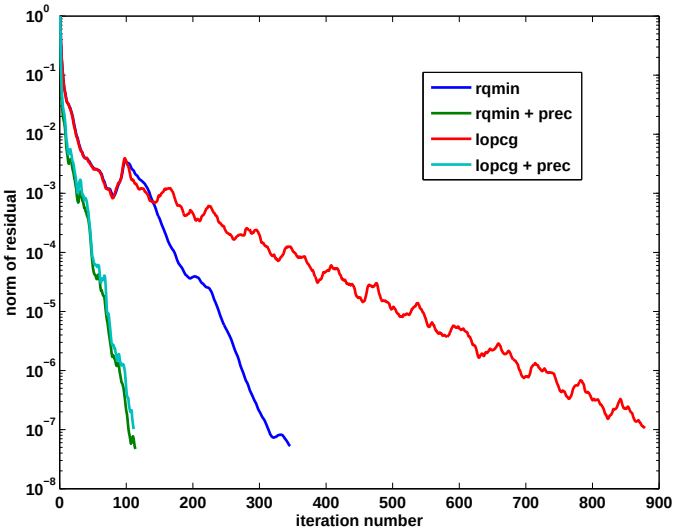

Figure 13.3: Convergence of variants of Rayleigh quotient minimization

iteration step requires solving ten systems of equation resulting in 430 system solves. (In fact, if converged eigenvectors are locked, only 283 systems had to be solved.) Nevertheless, when comparing with Fig. 13.3 one should remember that in the LOBPCG computation ten eigenpairs have been computed. If a single eigenpair is required then a blocksize of 10 is too big, but a smaller blocksize may reduce the execution time. If a small number of eigenvalues is desired then a blocksize equal or slightly bigger than theis number is certainly advantageous. Not that in step (5) of Algorithm 13.2 q linear systems of equations are solved concurrently. An efficient implementation accesses the preconditioner N only once. The MATLAB code does this naturally. A parallel implementation of LOBPCG can be found in the software package Block Locally Optimal Preconditioned Eigenvalue Xolvers (BLOPEX) [7].

13.8 Trace minimization

Remember that the **trace** of a matrix $A \in \mathbb{F}^{n\times n}$ is defined to be the sum of its diagonal elements. Similar matrices have equal trace. Hence, by the spectral theorem 2.14, we have

$$\operatorname{trace}(A) = \sum_{i=1}^{n} a_{ii} = \sum_{i=1}^{n} \lambda_i. \tag{13.42}$$

The following theorem [11] generalizes the trace theorem 2.33 for the generalized eigenvalue problem

$$A\mathbf{x} = \lambda M\mathbf{x}, \qquad A = A^*, \quad M = M^* > 0. \tag{13.1}$$

Theorem 13.1 ***(Trace theorem for the generalized eigenvalue problem)*** *Let A and M be as in (13.1). Then,*

$$\lambda_1 + \lambda_2 + \cdots + \lambda_p = \min_{X\in\mathbb{F}^{n\times p},\ X^*MX=I_p} \mathit{trace}(X^*AX) \tag{13.43}$$

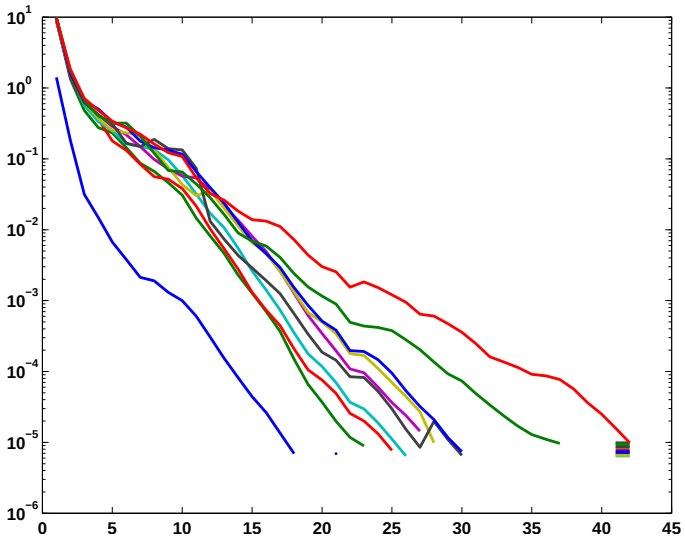

Figure 13.4: Convergence of 10 eigenvalues with LOBPCG preconditioned by IC(0)

where $\lambda_1, \ldots, \lambda_n$ are the eigenvalues of problem (13.1). Equality holds in (13.43) if and only if the columns of the matrix X that achieves the minimum span the eigenspace corresponding to the smallest p eigenvalues.

Sameh and coworkers [11, 10] suggested an algorithm to exploit this property of the trace, following the lines of Rayleigh quotient minimization. Let $X_k \in \mathbb{F}^{n\times p}$ with $X_k^* M X_k = I_p$ and

$$X_k^* A X_k = \Sigma_k = \operatorname{diag}(\sigma_1^{(k)}, \ldots, \sigma_p^{(k)}).$$

We want to construct the next iterate X_{k+1} by setting

$$X_{k+1} = (X_k - \Delta_k) S_k \tag{13.44}$$

such that

$$X_{k+1}^* M X_{k+1} = I_p, \tag{13.45}$$

$$X_{k+1}^* A X_{k+1} = \Sigma_{k+1} = \operatorname{diag}(\sigma_1^{(k+1)}, \ldots, \sigma_p^{(k+1)}), \tag{13.46}$$

$$\operatorname{trace}(X_{k+1}^* A X_{k+1}) < \operatorname{trace}(X_k^* A X_k). \tag{13.47}$$

S_k in (13.44) is needed to enforce the orthogonality of X_{k+1}. We choose the correction Δ_k to be orthogonal to X_k,

$$\Delta_k^* M X_k = 0. \tag{13.48}$$

Similarly as in Jacobi–Davidson [13] this choice of Δ_k is no loss of generality. We first assume that we have found the Δ_k that minimizes

$$\operatorname{trace}((X_k - \Delta_k)^* A (X_k - \Delta_k)) \tag{13.49}$$

under the constraint (13.48). Let $Z_{k+1} = X_k - \Delta_k$ be the solution of (13.49). Then, by construction,

$$\operatorname{trace}(Z_{k+1}^* A Z_{k+1}) \le \operatorname{trace}(X_k^* A X_k).$$

Furthermore,

$$Z_{k+1}^* M Z_{k+1} \overset{(13.48)}{=} X_k^* M X_k + \Delta_k^* M \Delta_k \geq X_k^* M X_k = I_p.$$

From this it follows that Z_{k+1} has maximal rank and that all eigenvalues of $Z_{k+1}^* M Z_{k+1}$ are ≥ 1. Therefore, the spectral decomposition of $Z_{k+1}^* M Z_{k+1}$ can be written in the form

$$Z_{k+1}^* M Z_{k+1} = U D^2 U^*, \qquad U^* U = I_p, \quad D = \operatorname{diag}(\delta_1, \ldots, \delta_p), \quad \delta_i \geq 1.$$

This implies that the columns of $Z_{k+1} U D^{-1}$ are M-orthogonal. Let the spectral decomposition of $D^{-1} U^* Z_{k+1}^* A Z_{k+1} U D^{-1}$ be given by

$$D^{-1} U^* Z_{k+1}^* A Z_{k+1} U D^{-1} = V \Sigma_{k+1} V^*, \qquad V^* V = I_p.$$

Then,

$$V^* \underbrace{D^{-1} U^* Z_{k+1}^* M Z_{k+1} U D^{-1}}_{I_p} V = I_p, \tag{13.50}$$

$$V^* D^{-1} U^* Z_{k+1}^* A Z_{k+1} U D^{-1} V = \Sigma_{k+1}. \tag{13.51}$$

So, if we have found $Z_{k+1} = X_k - \Delta_k$ then S_k in (13.44) is given by

$$S_k = U D^{-1} V.$$

Thus, with $X_{k+1} = Z_{k+1} S_k$ we have

$$\begin{aligned}
\operatorname{trace}(X_{k+1}^* A X_{k_1}) = \operatorname{trace}(\Sigma_{k+1}) &= \operatorname{trace}(V^* D^{-1} U^* Z_{k+1}^* A Z_{k+1} U D^{-1} V) \\
&= \operatorname{trace}(D^{-1} \underbrace{U^* Z_{k+1}^* A Z_{k+1} U}_{W} D^{-1}) \\
&= \sum_{i=1}^{p} w_{ii}/\delta_i^2 \\
&\leq \sum_{i=1}^{p} w_{ii} \\
&= \operatorname{trace}(U^* Z_{k+1}^* A Z_{k+1} U) \\
&\leq \operatorname{trace}(X_k^* A X_k)
\end{aligned}$$

Equality can only hold if $\Delta_k = 0$.

To solve the minimization problem (13.49) we write

$$\begin{aligned}
\operatorname{trace}((X_k - \Delta_k)^* A (X_k - \Delta_k)) &= \sum_{i=1}^{p} \mathbf{e}_i^* (X_k - \Delta_k)^* A (X_k - \Delta_k) \mathbf{e}_i \\
&= \sum_{i=1}^{p} (\mathbf{x}_i - \mathbf{d}_i)^* A (\mathbf{x}_i - \mathbf{d}_i)
\end{aligned}$$

where $\mathbf{x}_i = X_k \mathbf{e}_i$ and $\mathbf{d}_i = \Delta_k \mathbf{e}_i$. Interestingly, these are p individual minimization problems, one for each $\mathbf{d}_i$!

$$\text{Minimize} \quad (\mathbf{x}_i - \mathbf{d}_i)^* A (\mathbf{x}_i - \mathbf{d}_i) \quad \text{subject to } X_k^* M \mathbf{d}_i = \mathbf{0}, \quad i = 1, \ldots, p. \tag{13.52}$$

To solve (13.52) we define the functional

$$f(\mathbf{d},\mathbf{l}) := (\mathbf{x}_i - \mathbf{d})^* A(\mathbf{x}_i - \mathbf{d}) + \mathbf{l}^* X_k^* M\mathbf{d}.$$

Here, the vector $\mathbf{l}$ contains the Lagrange multipliers. A necessary condition for $\mathbf{d}$ to be a solution of (13.52) is

$$\begin{aligned} \partial_{\mathbf{d}} f = \mathbf{0} \quad &\Longleftrightarrow \quad A(\mathbf{x}_i - \mathbf{d}) + MX_k\mathbf{l} = \mathbf{0}, \\ \partial_{\mathbf{l}} f = \mathbf{0} \quad &\Longleftrightarrow \quad X_k^* M\mathbf{d} = \mathbf{0}. \end{aligned}$$

In matrix form this is

$$\begin{bmatrix} A & MX_k \\ X_k^* M & O \end{bmatrix} \begin{pmatrix} \mathbf{d} \\ \mathbf{l} \end{pmatrix} = \begin{pmatrix} A\mathbf{x}_i \\ 0 \end{pmatrix}.$$

We can collect all p equations in one,

$$\begin{bmatrix} A & MX_k \\ X_k^* M & O \end{bmatrix} \begin{pmatrix} \Delta_k \\ L \end{pmatrix} = \begin{pmatrix} AX_k \\ O \end{pmatrix}. \tag{13.53}$$

Using the LU factorization

$$\begin{bmatrix} A & MX_k \\ X_k^* M & O \end{bmatrix} = \begin{bmatrix} I & 0 \\ X_k^* MA^{-1} & I \end{bmatrix} \begin{bmatrix} A & MX_k \\ O & -X_k^* MA^{-1} MX_k \end{bmatrix}$$

we obtain

$$\begin{bmatrix} A & MX_k \\ O & -X_k^* MA^{-1} MX_k \end{bmatrix} \begin{pmatrix} \Delta_k \\ L \end{pmatrix} = \begin{bmatrix} I & 0 \\ -X_k^* MA^{-1} & I \end{bmatrix} \begin{pmatrix} AX_k \\ O \end{pmatrix} = \begin{pmatrix} AX_k \\ -X_k^* MX_k \end{pmatrix}.$$

Since $X_k^* MX_k = I_p$, L in (13.53) becomes

$$L = (X_k^* MA^{-1} MX_k)^{-1}.$$

Multiplying the first equation in (13.53) by A^{-1} we get

$$\Delta_k + A^{-1} MX_k L = X_k,$$

such that

$$Z_{k+1} = X_k - \Delta_k = A^{-1} MX_k L = A^{-1} MX_k (X_k^* MA^{-1} MX_k)^{-1}.$$

Thus, one step of the above trace minimization algorithm amounts to one step of subspace iteration with shift $\sigma = 0$. This proves convergence of the algorithm to the smallest eigenvalues of (13.1). Remember the similar equation (12.13) for the Jacobi–Davidson iteration and Remark 12.2.

Let P be the orthogonal projection onto $\mathcal{R}(MX_k)^\perp$,

$$P = I - (MX_k)((MX_k)^*(MX_k))^{-1}(MX_k)^* = I - MX_k(X_k^* M^2 X_k)^{-1} X_k^* M. \tag{13.54}$$

Then the linear systems of equations (13.53) and

$$PAP\Delta_k = PAX_k, \qquad X_k^* M\Delta_k = 0, \tag{13.55}$$

are equivalent, i.e., they have the same solution Δ_k. In fact, let $\begin{pmatrix} \Delta_k \\ L \end{pmatrix}$ be the solution of (13.53). Then, from $X_k M\Delta_k = 0$ we get $P\Delta_k = \Delta_k$. Equation (13.55) is now obtained

Algorithm 13.3 Trace minimization algorithm to compute p eigenpairs of $A\mathbf{x} = \lambda M\mathbf{x}$.

1: Choose random matrix $V_1 \in \mathbb{R}^{n\times q}$ with $V_1^T M V_1 = I_q$, $q \geq p$.
2: **for** $k = 1, 2, \ldots$ until convergence **do**
3: Compute $W_k = AV_k$ and $H_k := V_k^* W_k$.
4: Compute spectral decomposition $H_k = U_k \Theta_k U_k^*$,
with $\Theta_k = \mathrm{diag}(\vartheta_1^{(k)}, \ldots, \vartheta_q^{(k)})$, $\quad \vartheta_1^{(k)} \leq \ldots \leq \vartheta_q^{(k)}$.
5: Compute Ritz vectors $X_k = V_k U_k$ and residuals $R_k = W_k U_k - M X_k \Theta_k$
6: For $i = 1, \ldots, q$ solve *approximatively*

$$P(A - \sigma_i^{(k)} M) P\mathbf{d}_i^{(k)} = P\mathbf{r}_i, \qquad \mathbf{d}_i^{(k)} \perp_M X_k$$

by some modified PCG solver.
7: Compute $V_{k+1} = [X_k - \Delta_k, R_k]$, $\Delta_k = \mathbf{d}_1^{(k)}, \ldots, \mathbf{d}_q^{(k)}]$, by a M-orthogonal modified Gram-Schmidt procedure.
8: **end for**

by multiplying the first equation in (13.53) by P. On the other hand, let Δ_k be the solution of (13.55). Since $P(AP\Delta_k - AX_k) = 0$ we must have $AP\Delta_k - AX_k = MX_kL$ for some L. As $X_k M\Delta_k = 0$ we get $P\Delta_k = \Delta_k$ and thus the first equation in (13.53).

As PAP is positive semidefinite, equation (13.55) is easier to solve than equation (13.53) which is an indefinite system of equations. (13.55) can be solved by the (preconditioned) conjugate gradient method (PCG). The iteration has to be started by a vector $\mathbf{z}_0$ that satisfies the constrains $X_k^* M\mathbf{z}_0$. A straightforward choice is $\mathbf{z}_0 = \mathbf{0}$

reducion factor	10^{-4}		10^{-2}		0.5	
	#its	A mults	#its	A mults	#its	A mults
	59	6638	59	4263	77	4030

Table 13.1: The basic trace minimization algorithm (Algorithm 13.3). The inner systems are solved by the CG scheme which is terminated such that the 2-norm of the residual is reduced by a specified factor. The number of outer iterations (#its) and the number of multiplications with matrix A (A mults) are listed for different residual reduction factors.

In practice, we do not solve the p linear systems

$$P(A - \sigma_i^{(k)} M) P\mathbf{d}_i^{(k)} = P\mathbf{r}_i, \qquad \mathbf{d}_i^{(k)} \perp_M X_k \tag{13.56}$$

to high accuracy. In Table 13.1 the number of outer iterations (#its) are given and the number of multiplications of the matrix A with a vector for various relative stopping criteria for the inner iteration (reduction factor) [10].

Acceleration techniques

Sameh & Tong [10] investigate a number of ways to accelerate the convergence of the trace minimization algorithm 13.3.

1. *Simple shifts.* Choose a shift $\sigma_1 \leq \lambda_1$ until the first eigenpair is found. Then proceed with the shift $\sigma_2 \leq \lambda_2$ and *lock* the first eigenvector. In this way PCG can be used to solve the linear systems as before.

2. *Multiple dynamic shifts.* Each linear system (13.56) is solved with an individual shift. The shift is 'turned on' close to convergence. Since the linear systems are indefinite, PCG has to be adapted.

3. *Preconditioning.* The linear systems (13.56) can be preconditioned, e.g., by a matrix of the form $M = CC^*$ where $CC^* \approx A$ is an incomplete Cholesky factorization. One then solves

$$\tilde{P}(\tilde{A} - \sigma_i^{(k)}\tilde{M})\tilde{P}\tilde{\mathbf{d}}_i^{(k)} = \tilde{P}\tilde{\mathbf{r}}_i, \qquad \tilde{X}_k^*\tilde{M}\tilde{\mathbf{d}}_i^{(k)} = \mathbf{0} \tag{13.57}$$

with $\tilde{A} = C^{-1}AC^{-*}, \tilde{M} = C^{-1}MC^{-*}, \tilde{\mathbf{d}}_i^{(k)} = C^*\mathbf{d}_i^{(k)}, \tilde{X}_k = C^*X_k, \tilde{\mathbf{x}}_i^{(k)} = C^*\mathbf{x}_i^{(k)}$, and $\tilde{P} = I - \tilde{M}\tilde{X}_k(\tilde{X}_k^*\tilde{M}^2\tilde{X}_k)^{-1}\tilde{X}_k^*\tilde{M}$.

In Table 13.2 results are collected for some problems in the Harwell–Boeing collection [10]. These problems are diffcult because the gap ratios for the smallest eigenvalues are extremely small due to the huge span of the spectra. Without preconditioning, none of these problems can be solved with a reasonable cost. In the experiments, the incomplete Cholesky factorization (IC(0)) of A was used as the preconditioner for all the matrices of the form $A - \sigma B$.

Problem	Size	Max #	Block Jacobi–Davidson			Davidson-type tracemin		
		inner its	#its	A mults	time[sec]	#its	A mults	time[sec]
BCSST08	1074	40	34	3954	4.7	10	759	0.8
BCSST09	1083	40	15	1951	2.2	15	1947	2.2
BCSST11	1473	100	90	30990	40.5	54	20166	22.4
BCSST21	3600	100	40	10712	35.1	39	11220	36.2
BCSST26	1922	100	60	21915	32.2	39	14102	19.6

Table 13.2: Numerical results for problems from the Harwell–Boeing collection with four processors (reproduced from [10])

The Davidson-type trace minimization algorithm with multiple dynamic shifts works better than the block Jacobi–Davidson algorithm for three of the five problems. For the other two, the performance for both algorithms is similar.

Bibliography

[1] P. Arbenz, U. L. Hetmaniuk, R. B. Lehoucq, and R. Tuminaro, *A comparison of eigensolvers for large-scale 3D modal analysis using AMG-preconditioned iterative methods*, Internat. J. Numer. Methods Eng., 64 (2005), pp. 204–236.

[2] O. Axelsson and V. Barker, *Finite Element Solution of Boundary Value Problems*, Academic Press, Orlando FL, 1984.

[3] D. K. Faddeev and V. N. Faddeeva, *Computational Methods of Linear Algebra*, Freeman, San Francisco, 1963.

[4] Y. T. Feng and D. R. J. Owen, *Conjugate gradient methods for solving the smallest eigenpair of large symmetric eigenvalue problems*, Internat. J. Numer. Methods Eng., 39 (1996), pp. 2209–2229.

[5] M. R. Hestenes and E. Stiefel, *Methods of conjugent gradients for solving linear systems*, J. Res. Nat. Bur. Standards, 49 (1952), pp. 409–436.

[6] A. V. Knyazev, *Toward the optimal preconditioned eigensolver: Locally optimal block preconditioned conjugate gradient method*, SIAM J. Sci. Comput., 23 (2001), pp. 517–541.

[7] A. V. Knyazev, M. E. Argentati, I. Lashuk, and E. E. Ovtchinnikov, *Block locally optimal preconditioned eigenvalue Xolvers (BLOPEX) in hypre and PETSc*, SIAM J. Sci. Comput., 29 (2007), pp. 2224–2239.

[8] D. E. Longsine and S. F. McCormick, *Simultaneous Rayleigh–quotient minimization methods for $Ax = \lambda Bx$*, Linear Algebra Appl., 34 (1980), pp. 195–234.

[9] A. Ruhe, *Computation of eigenvalues and vectors*, in Sparse Matrix Techniques, V. A. Barker, ed., Lecture Notes in Mathematics 572, Berlin, 1977, Springer, pp. 130–184.

[10] A. Sameh and Z. Tong, *The trace minimization method for the symmetric generalized eigenvalue problem*, J. Comput. Appl. Math., 123 (2000), pp. 155–175.

[11] A. H. Sameh and J. A. Wisniewski, *A trace minimization algorithm for the generalized eigenvalue problem*, SIAM J. Numer. Anal., 19 (1982), pp. 1243–1259.

[12] H. R. Schwarz, *Rayleigh–Quotient–Minimierung mit Vorkonditionierung*, in Numerical Methods of Approximation Theory, Vol. 8, L. Collatz, G. Meinardus, and G. Nürnberger, eds., vol. 81 of International Series of Numerical Mathematics (ISNM), Basel, 1987, Birkhäuser, pp. 229–45.

[13] G. L. G. Sleijpen and H. A. van der Vorst, *Jacobi–Davidson method*, in Templates for the solution of Algebraic Eigenvalue Problems: A Practical Guide, Z. Bai, J. Demmel, J. Dongarra, A. Ruhe, and H. van der Vorst, eds., SIAM, Philadelphia, PA, 2000, pp. 238–246.