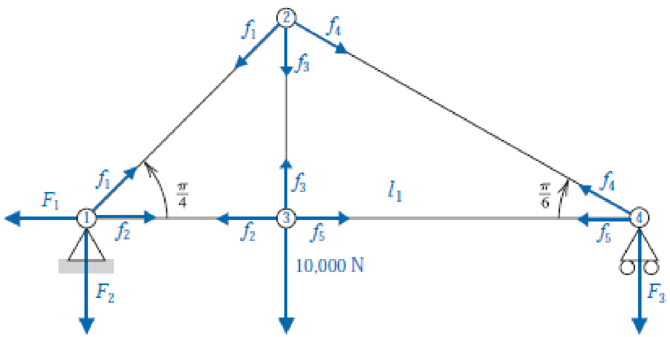
MAP 2220 – FUNDAMENTOS DE ANÁLISE NUMÉRICA

2º Semestre - 2017

Prof. Dr. Luis Carlos de Castro Santos

lsantos@ime.usp.br/lccs13@yahoo.com

7 Iterative Techniques in Matrix Algebra 431



$$\equiv \begin{bmatrix} -1 & 0 & 0 & \frac{\sqrt{2}}{2} & 1 & 0 & 0 & 0 \\ 0 & -1 & 0 & \frac{\sqrt{2}}{2} & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 0 & 0 & \frac{1}{2} & 0 \\ 0 & 0 & 0 & -\frac{\sqrt{2}}{2} & 0 & -1 & -\frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & -\frac{\sqrt{2}}{2} & 0 & 0 & \frac{\sqrt{3}}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{\sqrt{3}}{2} & -1 \end{bmatrix} \begin{bmatrix} F_1 \\ F_2 \\ F_3 \\ f_1 \\ f_2 \\ f_3 \\ f_4 \\ f_5 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 10{,}000 \\ 0 \\ 0 \end{bmatrix}.$$

Jacobi's Method

$$x_i^{(k)} = \frac{1}{a_{ii}}\left[\sum_{\substack{j=1\\ j\neq i}}^{n}\left(-a_{ij}x_j^{(k-1)}\right) + b_i\right], \qquad \text{for } i = 1, 2, \ldots, n.$$

$\mathbf{x}^{(k)} = T_j\mathbf{x}^{(k-1)} + \mathbf{c}_j.$ $\quad T_j = D^{-1}(L+U)$ and $\mathbf{c}_j = D^{-1}\mathbf{b}$

The Gauss-Seidel Method

$$x_i^{(k)} = \frac{1}{a_{ii}}\left[-\sum_{j=1}^{i-1}(a_{ij}x_j^{(k)}) - \sum_{j=i+1}^{n}(a_{ij}x_j^{(k-1)}) + b_i\right],$$

$\mathbf{x}^{(k)} = T_g\mathbf{x}^{(k-1)} + \mathbf{c}_g.$ $\quad T_g = (D-L)^{-1}U$ and $\mathbf{c}_g = (D-L)^{-1}\mathbf{b}$,

SOR, for Successive Over-Relaxation,

$$x_i^{(k)} = (1-\omega)x_i^{(k-1)} + \frac{\omega}{a_{ii}}\left[b_i - \sum_{j=1}^{i-1}a_{ij}x_j^{(k)} - \sum_{j=i+1}^{n}a_{ij}x_j^{(k-1)}\right]$$

$\mathbf{x}^{(k)} = T_\omega\mathbf{x}^{(k-1)} + \mathbf{c}_\omega.$ $\quad T_\omega = (D-\omega L)^{-1}[(1-\omega)D + \omega U]$ and $\mathbf{c}_\omega = \omega(D-\omega L)^{-1}\mathbf{b}$.

7.6 The Conjugate Gradient Method

Theorem 7.31 The vector $\mathbf{x}^*$ is a solution to the positive definite linear system $A\mathbf{x} = \mathbf{b}$ if and only if $\mathbf{x}^*$ produces the minimal value of

$$g(\mathbf{x}) = \langle \mathbf{x}, A\mathbf{x} \rangle - 2\langle \mathbf{x}, \mathbf{b} \rangle.$$ ■

Theorem 7.32 Let $\{\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(n)}\}$ be an A-orthogonal set of nonzero vectors associated with the positive definite matrix A, and let $\mathbf{x}^{(0)}$ be arbitrary. Define

$$t_k = \frac{\langle \mathbf{v}^{(k)}, \mathbf{b} - A\mathbf{x}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle} \quad \text{and} \quad \mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + t_k \mathbf{v}^{(k)},$$

for $k = 1, 2, \ldots, n$. Then, assuming exact arithmetic, $A\mathbf{x}^{(n)} = \mathbf{b}$. ■

Como produzir uma sequência de vetores A-ortogonais ?

Theorem 7.33 The residual vectors $\mathbf{r}^{(k)}$, where $k = 1, 2, \ldots, n$, for a conjugate direction method, satisfy the equations

$$\langle \mathbf{r}^{(k)}, \mathbf{v}^{(j)} \rangle = 0, \quad \text{for each} \quad j = 1, 2, \ldots, k. \qquad \blacksquare$$

The conjugate gradient method of Hestenes and Stiefel chooses the search directions $\{\mathbf{v}^{(k)}\}$ during the iterative process so that the residual vectors $\{\mathbf{r}^{(k)}\}$ are mutually orthogonal. To construct the direction vectors $\{\mathbf{v}^{(1)}, \mathbf{v}^{(2)}, \ldots\}$ and the approximations $\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots\}$, we start with an initial approximation $\mathbf{x}^{(0)}$ and use the steepest descent direction $\mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)}$ as the first search direction $\mathbf{v}^{(1)}$.

Assume that the conjugate directions $\mathbf{v}^{(1)}, \ldots, \mathbf{v}^{(k-1)}$ and the approximations $\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(k-1)}$ have been computed with

$$\mathbf{x}^{(k-1)} = \mathbf{x}^{(k-2)} + t_{k-1}\mathbf{v}^{(k-1)},$$

where

$$\langle \mathbf{v}^{(i)}, A\mathbf{v}^{(j)} \rangle = 0 \quad \text{and} \quad \langle \mathbf{r}^{(i)}, \mathbf{r}^{(j)} \rangle = 0, \quad \text{for} \quad i \neq j.$$

If $\mathbf{x}^{(k-1)}$ is the solution to $A\mathbf{x} = \mathbf{b}$, we are done. Otherwise, $\mathbf{r}^{(k-1)} = \mathbf{b} - A\mathbf{x}^{(k-1)} \neq \mathbf{0}$ and Theorem 7.33 implies that $\langle \mathbf{r}^{(k-1)}, \mathbf{v}^{(i)} \rangle = 0$, for each $i = 1, 2, \ldots, k-1$.

We use $\mathbf{r}^{(k-1)}$ to generate $\mathbf{v}^{(k)}$ by setting

$$\mathbf{v}^{(k)} = \mathbf{r}^{(k-1)} + s_{k-1}\mathbf{v}^{(k-1)}.$$

We want to choose s_{k-1} so that

$$\langle \mathbf{v}^{(k-1)}, A\mathbf{v}^{(k)} \rangle = 0.$$

Since

$$A\mathbf{v}^{(k)} = A\mathbf{r}^{(k-1)} + s_{k-1}A\mathbf{v}^{(k-1)}$$

and

$$\langle \mathbf{v}^{(k-1)}, A\mathbf{v}^{(k)} \rangle = \langle \mathbf{v}^{(k-1)}, A\mathbf{r}^{(k-1)} \rangle + s_{k-1}\langle \mathbf{v}^{(k-1)}, A\mathbf{v}^{(k-1)} \rangle,$$

we will have $\langle \mathbf{v}^{(k-1)}, A\mathbf{v}^{(k)} \rangle = 0$ when

$$s_{k-1} = -\frac{\langle \mathbf{v}^{(k-1)}, A\mathbf{r}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k-1)}, A\mathbf{v}^{(k-1)} \rangle}.$$

It can also be shown that with this choice of s_{k-1} we have $\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(i)} \rangle = 0$, for each $i = 1, 2, \ldots, k-2$ (see [Lu], p. 245). Thus $\{\mathbf{v}^{(1)}, \ldots \mathbf{v}^{(k)}\}$ is an A-orthogonal set.

Having chosen $\mathbf{v}^{(k)}$, we compute

$$t_k = \frac{\langle \mathbf{v}^{(k)}, \mathbf{r}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle} = \frac{\langle \mathbf{r}^{(k-1)} + s_{k-1}\mathbf{v}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle}$$

$$= \frac{\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle} + s_{k-1}\frac{\langle \mathbf{v}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle}.$$

By Theorem 7.33, $\langle \mathbf{v}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle = 0$, so

$$t_k = \frac{\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle}. \tag{7.30}$$

Thus

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + t_k\mathbf{v}^{(k)}.$$

To compute $\mathbf{r}^{(k)}$, we multiply by A and subtract $\mathbf{b}$ to obtain

$$A\mathbf{x}^{(k)} - \mathbf{b} = A\mathbf{x}^{(k-1)} - \mathbf{b} + t_kA\mathbf{v}^{(k)}$$

or

$$\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - t_kA\mathbf{v}^{(k)}.$$

This gives

$$\langle \mathbf{r}^{(k)}, \mathbf{r}^{(k)} \rangle = \langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k)} \rangle - t_k \langle A\mathbf{v}^{(k)}, \mathbf{r}^{(k)} \rangle = -t_k \langle \mathbf{r}^{(k)}, A\mathbf{v}^{(k)} \rangle.$$

Further, from Eq. (7.30),

$$\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle = t_k \langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle,$$

so

$$s_k = -\frac{\langle \mathbf{v}^{(k)}, A\mathbf{r}^{(k)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle} = -\frac{\langle \mathbf{r}^{(k)}, A\mathbf{v}^{(k)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle} = \frac{(1/t_k)\langle \mathbf{r}^{(k)}, \mathbf{r}^{(k)} \rangle}{(1/t_k)\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle} = \frac{\langle \mathbf{r}^{(k)}, \mathbf{r}^{(k)} \rangle}{\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle}.$$

In summary, we have

$$\mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)}; \quad \mathbf{v}^{(1)} = \mathbf{r}^{(0)};$$

and, for $k = 1, 2, \ldots, n$,

$$t_k = \frac{\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle}, \quad \mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + t_k \mathbf{v}^{(k)}, \quad \mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - t_k A\mathbf{v}^{(k)}, \quad s_k = \frac{\langle \mathbf{r}^{(k)}, \mathbf{r}^{(k)} \rangle}{\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle},$$

and

$$\mathbf{v}^{(k+1)} = \mathbf{r}^{(k)} + s_k \mathbf{v}^{(k)}. \tag{7.31}$$

Example 2 The linear system $A\mathbf{x} = \mathbf{b}$ given by

$$\begin{aligned} 4x_1 + 3x_2 &= 24, \\ 3x_1 + 4x_2 - x_3 &= 30, \\ - x_2 + 4x_3 &= -24 \end{aligned}$$

has solution $(3, 4, -5)^t$. Use the conjugate gradient method with $\mathbf{x}^{(0)} = (0, 0, 0)^t$

$$\mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)}; \quad \mathbf{v}^{(1)} = \mathbf{r}^{(0)};$$

and, for $k = 1, 2, \ldots, n$,

$$t_k = \frac{\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle}, \quad \mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + t_k \mathbf{v}^{(k)}, \quad \mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - t_k A\mathbf{v}^{(k)}, \quad s_k = \frac{\langle \mathbf{r}^{(k)}, \mathbf{r}^{(k)} \rangle}{\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle},$$

and

$$\mathbf{v}^{(k+1)} = \mathbf{r}^{(k)} + s_k \mathbf{v}^{(k)}. \tag{7.31}$$

MAP2220

Preconditioning

Rather than presenting an algorithm for the conjugate gradient method using these formulas, we extend the method to include *preconditioning*. If the matrix A is ill-conditioned, the conjugate gradient method is highly susceptible to rounding errors. So, although the exact answer should be obtained in n steps, this is not usually the case. As a direct method the conjugate gradient method is not as good as Gaussian elimination with pivoting. The main use of the conjugate gradient method is as an iterative method applied to a better-conditioned system. In this case an acceptable approximate solution is often obtained in about $\sqrt{n}$ steps.

When preconditioning is used, the conjugate gradient method is not applied directly to the matrix A but to another positive definite matrix that a smaller condition number. We need to do this in such a way that once the solution to this new system is found it will be easy to obtain the solution to the original system. The expectation is that this will reduce the rounding error when the method is applied. To maintain the positive definiteness of the resulting matrix, we need to multiply on each side by a nonsingular matrix. We will denote this matrix by C^{-1}, and consider

$$\tilde{A} = C^{-1}A(C^{-1})^t,$$

with the hope that $\tilde{A}$ has a lower condition number than A. To simplify the notation, we use the matrix notation $C^{-t} \equiv \left(C^{-1}\right)^t$. Later in the section we will see a reasonable way to select C, but first we will consider the conjugate applied to $\tilde{A}$.

Consider the linear system

$$\tilde{A}\tilde{\mathbf{x}} = \tilde{\mathbf{b}},$$

where $\tilde{\mathbf{x}} = C^t\mathbf{x}$ and $\tilde{\mathbf{b}} = C^{-1}\mathbf{b}$. Then

$$\tilde{A}\tilde{\mathbf{x}} = (C^{-1}AC^{-t})(C^t\mathbf{x}) = C^{-1}A\mathbf{x}.$$

Thus, we could solve $\tilde{A}\tilde{\mathbf{x}} = \tilde{\mathbf{b}}$ for $\tilde{\mathbf{x}}$ and then obtain $\mathbf{x}$ by multiplying by C^{-t}. However, instead of rewriting equations (7.31) using $\tilde{\mathbf{r}}^{(k)}$, $\tilde{\mathbf{v}}^{(k)}$, $\tilde{t}_k$, $\tilde{\mathbf{x}}^{(k)}$, and $\tilde{s}_k$, we incorporate the preconditioning implicitly.

Since

$$\tilde{\mathbf{x}}^{(k)} = C^t\mathbf{x}^{(k)},$$

we have

$$\tilde{\mathbf{r}}^{(k)} = \tilde{\mathbf{b}} - \tilde{A}\tilde{\mathbf{x}}^{(k)} = C^{-1}\mathbf{b} - (C^{-1}AC^{-t})C^t\mathbf{x}^{(k)} = C^{-1}(\mathbf{b} - A\mathbf{x}^{(k)}) = C^{-1}\mathbf{r}^{(k)}.$$

Let $\tilde{\mathbf{v}}^{(k)} = C^t\mathbf{v}^{(k)}$ and $\mathbf{w}^{(k)} = C^{-1}\mathbf{r}^{(k)}$. Then

$$\tilde{s}_k = \frac{\langle \tilde{\mathbf{r}}^{(k)}, \tilde{\mathbf{r}}^{(k)} \rangle}{\langle \tilde{\mathbf{r}}^{(k-1)}, \tilde{\mathbf{r}}^{(k-1)} \rangle} = \frac{\langle C^{-1}\mathbf{r}^{(k)}, C^{-1}\mathbf{r}^{(k)} \rangle}{\langle C^{-1}\mathbf{r}^{(k-1)}, C^{-1}\mathbf{r}^{(k-1)} \rangle},$$

so

$$\tilde{s}_k = \frac{\langle \mathbf{w}^{(k)}, \mathbf{w}^{(k)} \rangle}{\langle \mathbf{w}^{(k-1)}, \mathbf{w}^{(k-1)} \rangle}. \tag{7.32}$$

Thus

$$\tilde{t}_k = \frac{\langle \tilde{\mathbf{r}}^{(k-1)}, \tilde{\mathbf{r}}^{(k-1)} \rangle}{\langle \tilde{\mathbf{v}}^{(k)}, \tilde{A}\tilde{\mathbf{v}}^{(k)} \rangle} = \frac{\langle C^{-1}\mathbf{r}^{(k-1)}, C^{-1}\mathbf{r}^{(k-1)} \rangle}{\langle C^t\mathbf{v}^{(k)}, C^{-1}AC^{-t}C^t\mathbf{v}^{(k)} \rangle} = \frac{\langle \mathbf{w}^{(k-1)}, \mathbf{w}^{(k-1)} \rangle}{\langle C^t\mathbf{v}^{(k)}, C^{-1}A\mathbf{v}^{(k)} \rangle}$$

and, since

$$\begin{aligned}\langle C^t\mathbf{v}^{(k)}, C^{-1}A\mathbf{v}^{(k)} \rangle &= [C^t\mathbf{v}^{(k)}]^t C^{-1}A\mathbf{v}^{(k)} \\ &= [\mathbf{v}^{(k)}]^t CC^{-1}A\mathbf{v}^{(k)} = [\mathbf{v}^{(k)}]^t A\mathbf{v}^{(k)} = \langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle,\end{aligned}$$

we have

$$\tilde{t}_k = \frac{\langle \mathbf{w}^{(k-1)}, \mathbf{w}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle}. \tag{7.33}$$

Further,

$$\tilde{\mathbf{x}}^{(k)} = \tilde{\mathbf{x}}^{(k-1)} + \tilde{t}_k\tilde{\mathbf{v}}^{(k)}, \quad \text{so} \quad C^t\mathbf{x}^{(k)} = C^t\mathbf{x}^{(k-1)} + \tilde{t}_k C^t\mathbf{v}^{(k)}$$

and

$$\mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + \tilde{t}_k\mathbf{v}^{(k)}. \tag{7.34}$$

Continuing,

$$\tilde{\mathbf{r}}^{(k)} = \tilde{\mathbf{r}}^{(k-1)} - \tilde{t}_k\tilde{A}\tilde{\mathbf{v}}^{(k)},$$

so

$$C^{-1}\mathbf{r}^{(k)} = C^{-1}\mathbf{r}^{(k-1)} - \tilde{t}_k C^{-1}AC^{-t}\tilde{v}^{(k)}, \qquad \mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \tilde{t}_k AC^{-t}C^t\mathbf{v}^{(k)},$$

and

$$\mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - \tilde{t}_k A\mathbf{v}^{(k)}. \tag{7.35}$$

Finally,

$$\tilde{\mathbf{v}}^{(k+1)} = \tilde{\mathbf{r}}^{(k)} + \tilde{s}_k \tilde{\mathbf{v}}^{(k)} \quad \text{and} \quad C^t \mathbf{v}^{(k+1)} = C^{-1}\mathbf{r}^{(k)} + \tilde{s}_k C^t \mathbf{v}^{(k)},$$

so

$$\mathbf{v}^{(k+1)} = C^{-t}C^{-1}\mathbf{r}^{(k)} + \tilde{s}_k \mathbf{v}^{(k)} = C^{-t}\mathbf{w}^{(k)} + \tilde{s}_k \mathbf{v}^{(k)}. \tag{7.36}$$

The preconditioned conjugate gradient method is based on using equations (7.32)–(7.36) in the order (7.33), (7.34), (7.35), (7.32), and (7.36). Algorithm 7.5 implements this procedure.

ALGORITHM 7.5

Preconditioned Conjugate Gradient Method

To solve $A\mathbf{x} = \mathbf{b}$ given the preconditioning matrix C^{-1} and the initial approximation $\mathbf{x}^{(0)}$:

INPUT the number of equations and unknowns n; the entries a_{ij}, $1 \le i, j \le n$ of the matrix A; the entries b_j, $1 \le j \le n$ of the vector $\mathbf{b}$; the entries γ_{ij}, $1 \le i, j \le n$ of the preconditioning matrix C^{-1}, the entries x_i, $1 \le i \le n$ of the initial approximation $\mathbf{x} = \mathbf{x}^{(0)}$, the maximum number of iterations N; tolerance TOL.

OUTPUT the approximate solution $x_1, \ldots x_n$ and the residual $r_1, \ldots r_n$ or a message that the number of iterations was exceeded.

Step 1 Set $\mathbf{r} = \mathbf{b} - A\mathbf{x}$; (*Compute* $\mathbf{r}^{(0)}$.)
$\mathbf{w} = C^{-1}\mathbf{r}$; (*Note:* $\mathbf{w} = \mathbf{w}^{(0)}$)
$\mathbf{v} = C^{-t}\mathbf{w}$; (*Note:* $\mathbf{v} = \mathbf{v}^{(1)}$)
$\alpha = \sum_{j=1}^{n} w_j^2$.

Step 2 Set $k = 1$.

Step 3 While $(k \leq N)$ do Steps 4–7.

Step 4 If $\|\mathbf{v}\| < TOL$, then
OUTPUT ('Solution vector'; $x_1, \ldots, x_n$);
OUTPUT ('with residual'; $r_1, \ldots, r_n$);
(*The procedure was successful.*)
STOP

Step 5 Set $\mathbf{u} = A\mathbf{v}$; (*Note:* $\mathbf{u} = A\mathbf{v}^{(k)}$)

$$t = \frac{\alpha}{\sum_{j=1}^{n} v_j u_j}; \text{ (\textit{Note:} } t = t_k)$$

$\mathbf{x} = \mathbf{x} + t\mathbf{v}$; (*Note:* $\mathbf{x} = \mathbf{x}^{(k)}$)
$\mathbf{r} = \mathbf{r} - t\mathbf{u}$; (*Note:* $\mathbf{r} = \mathbf{r}^{(k)}$)
$\mathbf{w} = C^{-1}\mathbf{r}$; (*Note:* $\mathbf{w} = \mathbf{w}^{(k)}$)
$\beta = \sum_{j=1}^{n} w_j^2$. (*Note:* $\beta = \langle \mathbf{w}^{(k)}, \mathbf{w}^{(k)} \rangle$)

Step 6 If $|\beta| < TOL$ then
if $\|\mathbf{r}\| < TOL$ then
OUTPUT('Solution vector'; $x_1, \ldots, x_n$);
OUTPUT('with residual'; $r_1, \ldots, r_n$);
(*The procedure was successful.*)
STOP

Step 7 Set $s = \beta/\alpha$; $(s = s_k)$
$\mathbf{v} = C^{-t}\mathbf{w} + s\mathbf{v}$; (*Note:* $\mathbf{v} = \mathbf{v}^{(k+1)}$)
$\alpha = \beta$; (*Update* α.)
$k = k + 1$.

Step 8 If $(k > n)$ then
OUTPUT ('The maximum number of iterations was exceeded.');
(*The procedure was unsuccessful.*)
STOP.

The next example illustrates the effect of preconditioning on a poorly conditioned matrix. In this example, we use $D^{-1/2}$ to represent the diagonal matrix whose entries are the reciprocals of the square roots of the diagonal entries of the coefficient matrix A. This is used as the preconditioner. Because the matrix A is positive definite we expect the eigenvalues of $D^{-1/2}AD^{-1/2}$ to be close to 1, with the result that the condition number of this matrix will be small relative to the condition number of A.

Example 3 Use Maple to find the eigenvalues and condition number of the matrix

$$A = \begin{bmatrix} 0.2 & 0.1 & 1 & 1 & 0 \\ 0.1 & 4 & -1 & 1 & -1 \\ 1 & -1 & 60 & 0 & -2 \\ 1 & 1 & 0 & 8 & 4 \\ 0 & -1 & -2 & 4 & 700 \end{bmatrix}$$

and compare these with the eigenvalues and condition number of the preconditioned matrix $D^{-1/2}AD^{-t/2}$.

Solution We first need to load the *LinearAlgebra* package and then enter the matrix.

$with(LinearAlgebra)$:
$A := Matrix([[0.2, 0.1, 1, 1, 0], [0.1, 4, -1, 1, -1], [1, -1, 60, 0, -2],$
$[1, 1, 0, 8, 4], [0, -1, -2, 4, 700]])$

To determine the preconditioned matrix we first need the diagonal matrix, which being symmetric is also its transpose. its diagonal entries are specified by

$$a1 := \frac{1}{\sqrt{0.2}};\ a2 := \frac{1}{\sqrt{4.0}};\ a3 := \frac{1}{\sqrt{60.0}};\ a4 := \frac{1}{\sqrt{8.0}};\ a5 := \frac{1}{\sqrt{700.0}}$$

and the preconditioning matrix is

$CI := Matrix([[a1, 0, 0, 0, 0], [0, a2, 0, 0, 0], [0, 0, a3, 0, 0], [0, 0, 0, a4, 0], [0, 0, 0, 0, a5]])$

which Maple returns as

$$\begin{bmatrix} 2.23607 & 0 & 0 & 0 & 0 \\ 0 & .500000 & 0 & 0 & 0 \\ 0 & 0 & .129099 & 0 & 0 \\ 0 & 0 & 0 & .353553 & 0 \\ 0 & 0 & 0 & 0 & 0.0377965 \end{bmatrix}$$

The preconditioned matrix is

$AH := CI.A.Transpose(CI)$

$$\begin{bmatrix} 1.000002 & 0.1118035 & 0.2886744 & 0.7905693 & 0 \\ 0.1118035 & 1 & -0.0645495 & 0.1767765 & -0.0188983 \\ 0.2886744 & -0.0645495 & 0.9999931 & 0 & -0.00975898 \\ 0.7905693 & 0.1767765 & 0 & 0.9999964 & 0.05345219 \\ 0 & -0.0188983 & -0.00975898 & 0.05345219 & 1.000005 \end{bmatrix}$$

The eigenvalues of A and AH are found with

Maple gives these as

$$\textit{Eigenvalues of } A : 700.031, 60.0284, 0.0570747, 8.33845, 3.74533$$

$$\textit{Eigenvalues of } AH : 1.88052, 0.156370, 0.852686, 1.10159, 1.00884$$

The condition numbers of A and AH in the l_∞ norm are found with

$ConditionNumber(A); ConditionNumber(AH)$

which Maple gives as 13961.7 for A and 16.1155 for AH. It is certainly true in this case that AH is better conditioned that the original matrix A. ■

Illustration The linear system $A\mathbf{x} = \mathbf{b}$ with

$$A = \begin{bmatrix} 0.2 & 0.1 & 1 & 1 & 0 \\ 0.1 & 4 & -1 & 1 & -1 \\ 1 & -1 & 60 & 0 & -2 \\ 1 & 1 & 0 & 8 & 4 \\ 0 & -1 & -2 & 4 & 700 \end{bmatrix} \quad \text{and} \quad \mathbf{b} = \begin{bmatrix} 1 \\ 2 \\ 3 \\ 4 \\ 5 \end{bmatrix}$$

has the solution

$$\mathbf{x}^* = (7.859713071, 0.4229264082, -0.07359223906, -0.5406430164, 0.01062616286)^t.$$

Table 7.5 lists the results obtained by using the Jacobi, Gauss-Seidel, and SOR (with $\omega = 1.25$) iterative methods applied to the system with A with a tolerance of 0.01, as well as those when the Conjugate Gradient method is applied both in its unpreconditioned form and using the preconditioning matrix described in Example 3. The preconditioned conjugate gradient method not only gives the most accurate approximations, it also uses the smallest number of iterations. □

Table 7.5

Method	Number of Iterations	$\mathbf{x}^{(k)}$	$\Vert\mathbf{x}^* - \mathbf{x}^{(k)}\Vert_\infty$
Jacobi	49	$(7.86277141, 0.42320802, -0.07348669, -0.53975964, 0.01062847)^t$	0.00305834
Gauss-Seidel	15	$(7.83525748, 0.42257868, -0.07319124, -0.53753055, 0.01060903)^t$	0.02445559
SOR ($\omega = 1.25$)	7	$(7.85152706, 0.42277371, -0.07348303, -0.53978369, 0.01062286)^t$	0.00818607
Conjugate Gradient	5	$(7.85341523, 0.42298677, -0.07347963, -0.53987920, 0.008628916)^t$	0.00629785
Conjugate Gradient (Preconditioned)	4	$(7.85968827, 0.42288329, -0.07359878, -0.54063200, 0.01064344)^t$	0.00009312

5. Perform only two steps of the conjugate gradient method with $C = C^{-1} = I$ on each of the following linear systems. Compare the results in parts (b) and (c) to the results obtained in parts (b) and (c) of Exercise 1 of Section 7.3 and Exercise 1 of Section 7.4.

a.
$$\begin{aligned} 3x_1 - x_2 + x_3 &= 1, \\ -x_1 + 6x_2 + 2x_3 &= 0, \\ x_1 + 2x_2 + 7x_3 &= 4. \end{aligned}$$

b.
$$\begin{aligned} 10x_1 - x_2 \phantom{{}- 2x_3} &= 9, \\ -x_1 + 10x_2 - 2x_3 &= 7, \\ - 2x_2 + 10x_3 &= 6. \end{aligned}$$

Set $\mathbf{r} = \mathbf{b} - A\mathbf{x}$; (*Compute* $\mathbf{r}^{(0)}$.)

$\mathbf{w} = C^{-1}\mathbf{r}$; (*Note:* $\mathbf{w} = \mathbf{w}^{(0)}$)

$\mathbf{v} = C^{-t}\mathbf{w}$; (*Note:* $\mathbf{v} = \mathbf{v}^{(1)}$)

$\alpha = \sum_{j=1}^{n} w_j^2$.

Set $\mathbf{u} = A\mathbf{v}$; (*Note:* $\mathbf{u} = A\mathbf{v}^{(k)}$)

$t = \dfrac{\alpha}{\sum_{j=1}^{n} v_j u_j}$; (*Note:* $t = t_k$)

$\mathbf{x} = \mathbf{x} + t\mathbf{v}$; (*Note:* $\mathbf{x} = \mathbf{x}^{(k)}$)

$\mathbf{r} = \mathbf{r} - t\mathbf{u}$; (*Note:* $\mathbf{r} = \mathbf{r}^{(k)}$)

$\mathbf{w} = C^{-1}\mathbf{r}$; (*Note:* $\mathbf{w} = \mathbf{w}^{(k)}$)

$\beta = \sum_{j=1}^{n} w_j^2$. (*Note:* $\beta = \langle \mathbf{w}^{(k)}, \mathbf{w}^{(k)} \rangle$)

Set $s = \beta/\alpha$; ($s = s_k$)

$\mathbf{v} = C^{-t}\mathbf{w} + s\mathbf{v}$; (*Note:* $\mathbf{v} = \mathbf{v}^{(k+1)}$)

$\alpha = \beta$; (*Update* α.)

$k = k + 1$.

$$\mathbf{r}^{(0)} = \mathbf{b} - A\mathbf{x}^{(0)}; \quad \mathbf{v}^{(1)} = \mathbf{r}^{(0)};$$

and, for $k = 1, 2, \ldots, n$,

$$t_k = \frac{\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle}{\langle \mathbf{v}^{(k)}, A\mathbf{v}^{(k)} \rangle}, \quad \mathbf{x}^{(k)} = \mathbf{x}^{(k-1)} + t_k \mathbf{v}^{(k)}, \quad \mathbf{r}^{(k)} = \mathbf{r}^{(k-1)} - t_k A\mathbf{v}^{(k)}, \quad s_k = \frac{\langle \mathbf{r}^{(k)}, \mathbf{r}^{(k)} \rangle}{\langle \mathbf{r}^{(k-1)}, \mathbf{r}^{(k-1)} \rangle},$$

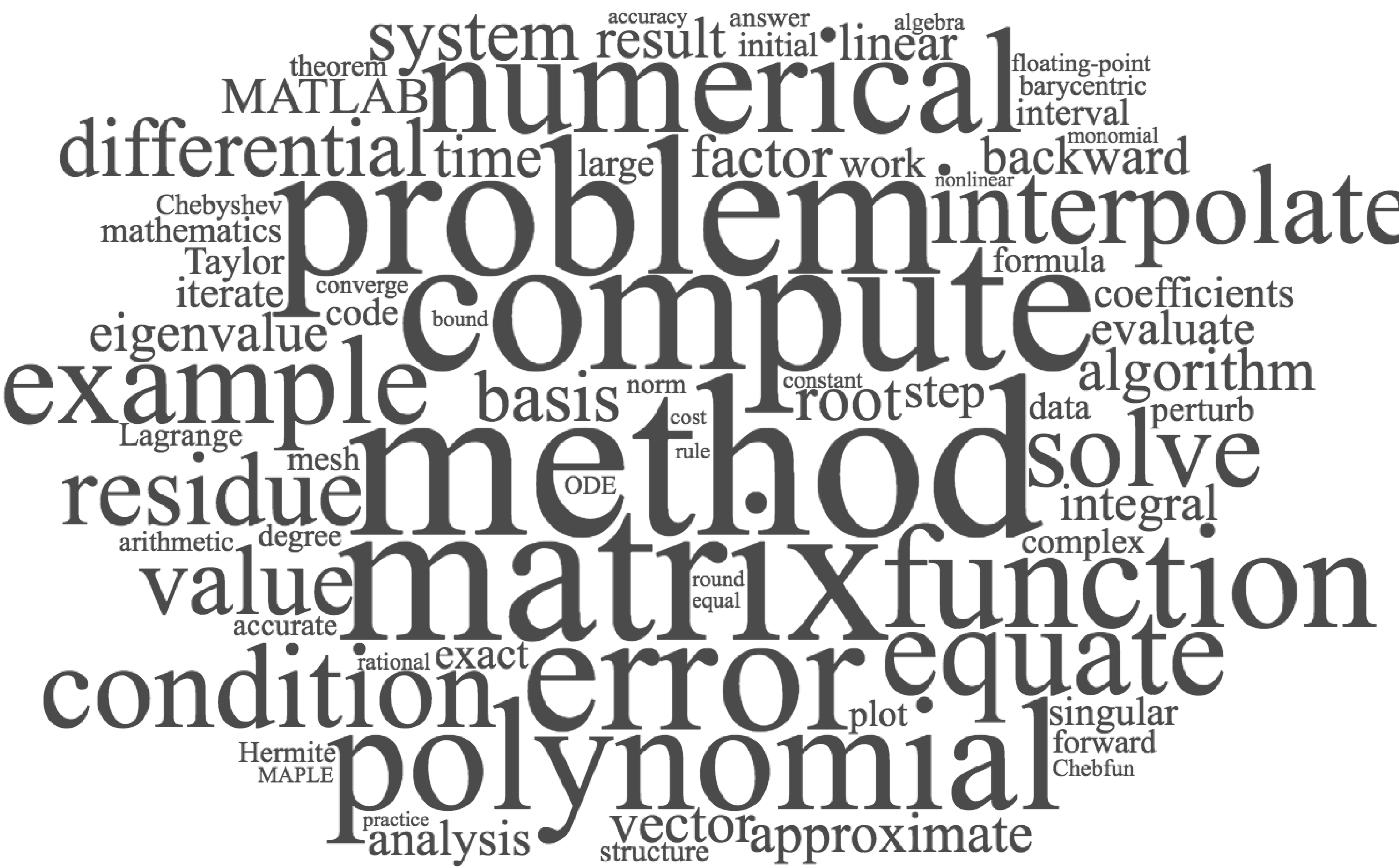
and

$$\mathbf{v}^{(k+1)} = \mathbf{r}^{(k)} + s_k \mathbf{v}^{(k)}. \tag{7.31}$$

MAP2220

7 Iterative Techniques in Matrix Algebra 431

MAP2220


accuracy
answer
algebra
system
result
initial
linear
theorem
floating-point
numerical
MATLAB
barycentric
interval
monomial
differential
time
large
factor
work
backward
nonlinear
Chebyshev
problem
interpolate
mathematics
Taylor
formula
converge
compute
iterate
coefficients
code
bound
eigenvalue
evaluate
algorithm
example
constant
basis
norm
root
step
data
perturb
cost
Lagrange
method
solve
mesh
rule
residue
ODE
integral
arithmetic
degree
complex
matrix
function
round
value
equal
accurate
exact
equate
rational
error
condition
plot
singular
Hermite
polynomial
forward
MAPLE
Chebfun
practice
vector
approximate
analysis
structure