Conjugate Direction Methods for Systems of Linear Algebraic Equations

Neculai Andrei,
Research Institute for Informatics,
Center for Advanced Modeling and Optimization,
8-10 Averescu Avenue, Bucharest 1, Romania
E-mail: nandrei@ici.ro

Abstract. The purpose of this work is to introduce the main aspects of conjugate direction methods for solving systems of linear algebraic equations. We emphasize the algorithms and their analysis concerning the convergence. Some numerical examples are presented.

1. Steepest descent method for quadratic function

Consider the linear algebraic system of equations

$$Ax = b, \tag{1}$$

where A is a symmetric and positive definite matrix, as well as the functional

$$\Phi(x) = \frac{1}{2}x^T Ax - x^T b, \tag{2}$$

which is strictly convex. Then, there is a unique solution x^* of the problem

$$\min \Phi(x). \tag{3}$$

Since $\nabla\Phi(x) = Ax - b,$ it follows that the minimization problem (3) is equivalent to the linear algebraic system of equations (1).

For solving (3) consider the following iterative method. Let x_0 be an approximate solution of (3) and assume that the residual $r_k = b - Ax_k \neq 0.$ Choose a search direction $d_k \neq 0$ and determine

$$\min_{t\in R} \Phi(x_k + td_k). \tag{4}$$

With these we have

$$\Psi(t) = \Phi(x_k + td_k) = \frac{1}{2}t^2 d_k^T Ad_k - td_k^T r_k + \Phi(x_k). \tag{5}$$

Having in view that A is a positive definite matrix, then $d_k^T Ad_k > 0.$ Therefore, $\Psi(t)$ has an unique minimum obtained from the necessary condition

$$\Psi'(t) = td_k^T Ad_k - d_k^T r_k = 0,$$

i.e.

$$t_k = \frac{d_k^T r_k}{d_k^T Ad_k}. \tag{6}$$

With this, the new approximation to the minimum point x^* of (3) is

$$x_{k+1} = x_k + t_k d_k, \tag{7}$$

where t_k is given by (6). Of course, if $r_{k+1} = b - Ax_{k+1} \neq 0,$ we can repeat the above process of one dimensional line search using a new search direction d_{k+1}.

Now, let see the efficiency of the above method, by computing the reduction of the functional Φ when we make a step from x_k to x_{k+1}. From (5) and (6) we obtain

$$\begin{aligned}\Phi(x_k) - \Phi(x_{k+1}) &= \Phi(x_k) - \Phi(x_k + t_k d_k) = \Phi(x_k) - \Psi(t_k) \\ &= -\frac{1}{2} t_k^2 d_k^T A d_k + t_k d_k^T r_k \\ &= \frac{\left(d_k^T r_k\right)^2}{d_k^T A d_k} > 0. \end{aligned} \tag{8}$$

Therefore the functional Φ reduce its value along the above line step if and only if the residual r_k is not orthogonal on the search direction d_k.

To complete the method and to have an algorithm to minimize (2) we have to consider a way to select the search direction d_k at step k. As we know, given the approximation x_k, the most rapid decreasing of the functional Φ is in the negative direction of its gradient in x_k. This is exactly the method of steepest descent [Cauchy, 1847]. Therefore, the local optimum choice of the search direction is

$$d_k = -\nabla\Phi(x_k) = b - Ax_k = r_k. \tag{9}$$

Therefore, for quadratic functions the iterative formula is

$$x_{k+1} = x_k + \frac{r_k^T r_k}{r_k^T A r_k} r_k. \tag{10}$$

With this we have the following steepest descent algorithm

```
Steepest descent algorithm
1. Select $x_0$ and set $r_0 = b - Ax_0$.
2. for $k = 0,1,\ldots$ until convergence do:
3.     $v_k = Ar_k$,
4.     $t_k = (r_k^T r_k)/(r_k^T v_k)$,
5.     $x_{k+1} = x_k + t_k r_k$,
6.     $r_{k+1} = r_k - t_k v_k$.
7. end for ♦
```

Some remarks are in order:

1) The cost of the steepest descent algorithm consists of one matrix-vector product in step 3 and two scalar products in step 4.

2) In step 6 the residual r_{k+1} is computed in a recursive manner using the residual from the step k.

3) A criterion for stopping the algorithm is to test if $\|r_k\| \leq \varepsilon$, where ε is a tolerance specified, usually $\varepsilon = 10^{-6}$.

Observe that, for solution x^* of (1), for every $x \in R^n$, we can write:

$$(x - x^*)^T A(x - x^*) = 2(\Phi(x) - \Phi(x^*)).$$

Therefore, $\tilde{x}$ is a minimum point of Φ if and only if it minimizes the norm

$$\left\| x^* - \tilde{x} \right\|_A = \sqrt{(x^* - \tilde{x})^T A (x^* - \tilde{x})} \tag{11}$$

of the error of $\tilde{x}$.

Theorem 1. *Let* A *be a* $n \times n-$*symmetric and positive definite matrix with eigenvalues* $\lambda_1 \le \lambda_2 \le \ldots \le \lambda_n$. *For every initial point* x_0 *the sequence* x_k *generated by the steepest descent algorithm converges to the solution* x^* *of (1) and*

$$\left\| x_k - x^* \right\|_A \le \left(\frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} \right)^k \left\| x_0 - x^* \right\|_A . \tag{12}$$

Proof. Consider an arbitrary point $x \in R^n$ and the corresponding residual $r = b - Ax$. The step taken by the steepest descent algorithm is

$$\begin{aligned} y - x + \frac{\|r\|_2^2}{r^T A r}\left(b - Ax\right) &= x + \frac{\|r\|_2^2}{r^T A r} b - \frac{\|r\|_2^2}{r^T A r} Ax \\ &= \left(I - t(x)A\right)x + t(x)b, \end{aligned}$$

where $t(x) = \|r\|_2^2 / (r^T A r)$. Therefore the error is

$$\begin{aligned} y - x^* &= \left(I - t(x)A\right)x + t(x)b - x^* \\ &= \left(I - t(x)A\right)x + t(x)Ax^* - x^* \\ &= \left(I - t(x)A\right)\left(x - x^*\right). \end{aligned}$$

Now, let us consider $z(\sigma) = \left(I - \sigma A\right)x + \sigma b$, where $\sigma \in R$. For every $\sigma \in R$ we have

$$\left\| y - x^* \right\|_A \le \left\| z(\sigma) - x^* \right\|_A .$$

We prove that for $\sigma = \tilde{\sigma} \equiv 2/(\lambda_n + \lambda_1)$ the inequality

$$\left\| z(\tilde{\sigma}) - x^* \right\|_A \le \frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} \left\| x - x^* \right\|_A$$

is satisfied. Indeed, let us consider the errors $e = x - x^*$ and $\tilde{e} = z(\tilde{\sigma}) - x^*$, then

$$\begin{aligned} \tilde{e} = z(\tilde{\sigma}) - x^* &= \left(I - \tilde{\sigma}A\right)x + \tilde{\sigma}b - x^* \\ &= \left(I - \tilde{\sigma}A\right)x + \tilde{\sigma}Ax^* - x^* = \left(I - \tilde{\sigma}A\right)e. \end{aligned}$$

Observe that the matrix $I - \tilde{\sigma}A$ is symmetric with eigenvalues $1 - \tilde{\sigma}\lambda_j$. Therefore its spectral norm is given by

$$\left\| I - \tilde{\sigma}A \right\|_2 = \max\left\{ \left| 1 - \frac{2\lambda_1}{\lambda_n + \lambda_1} \right|, \left| 1 - \frac{2\lambda_n}{\lambda_n + \lambda_1} \right| \right\} = \frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1}.$$

With these we get

$$\begin{aligned} \left\| y - x^* \right\|_A \le \left\| \tilde{e} \right\|_A &= \left\| A^{1/2}\tilde{e} \right\|_2 = \left\| A^{1/2}(I - \tilde{\sigma}A)e \right\|_2 \\ &= \left\| (I - \tilde{\sigma}A)A^{1/2}e \right\|_2 \le \frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} \left\| A^{1/2}e \right\|_2 = \frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} \left\| e \right\|_A , \end{aligned}$$

and (12) is obtained by induction. ♦

Remark 1. If λ_1 and λ_n are the eigenvalues of A as defined in theorem 1 and $\kappa(A)=\lambda_n/\lambda_1$ is the condition number of A with respect to the Euclidian norm, then the steepest descent algorithm is characterized by the following reduction factor:

$$\left\|x_k - x^*\right\|_A \le \eta \left\|x_{k-1} - x^*\right\|_A, \tag{13}$$

where

$$\eta = \frac{\lambda_n - \lambda_1}{\lambda_n + \lambda_1} = \frac{\kappa(A)-1}{\kappa(A)+1}. \tag{14}$$

Hence, the smaller the condition number of A, the faster the steepest descent algorithm. In particular, when $\kappa(A)=1,$ then $\eta = 0$ and the gradient algorithm find the minimum point in one step.

Example 1. Consider

$$\Phi(x) = \frac{1}{2} x^T \begin{bmatrix} 1 & 0 \\ 0 & \gamma \end{bmatrix} x = \frac{1}{2}\left(x_1^2 + \gamma x_2^2\right),$$

where $\gamma > 0.$ The minimum point is $x^* = 0$ and $\Phi(x^*) = 0.$ The eigenvalues of A are 1 and $\gamma.$ We apply the steepest descent algorithm starting at the point $x_0 = (\lambda, 1).$ In this case we can derive the following analytic solution for iterates:

$$x_{1(k)} = \gamma\left(\frac{\gamma-1}{\gamma+1}\right)^k, \qquad x_{2(k)} = \left(-\frac{\gamma-1}{\gamma+1}\right)^k,$$

and the corresponding function values is

$$\Phi(x_k) = \frac{\gamma(\gamma+1)}{2}\left(\frac{\gamma-1}{\gamma+1}\right)^{2k} = \left(\frac{\gamma-1}{\gamma+1}\right)^{2k} \Phi(x_0).$$

For this example we see that the convergence is linear, i.e. at each iteration the error is reduced as a geometric series, by the factor $(\gamma-1)^2/(\gamma+1)^2$. For $\gamma = 1$ the exact solution is obtained in one iteration. For γ not far from one the convergence is rapid. The convergence is very slow for $\gamma \gg 1$ or $\gamma \ll 1.$ For different values of γ, Table 1 contains the number of iterations (# iter) necessary to get the solution for which $\left\|r_k\right\|_2 \le 10^{-9}.$

Table 1. Steepest descent algorithm for example 1. Number of iterations.

γ	#iter	γ	# iter
10	117	10^{-1}	94
10^2	1284	10^{-2}	824
10^3	13989	10^{-3}	7082
10^4	151401	10^{-4}	59298

This bad performance of the steepest descent algorithm for ill conditioned problems is explained by the fact that the level curves of Φ are very deformed ellipses. The method is zig-zagging through the flat and step-sided valley $\left\{x \in R^2 : \Phi(x) \le \Phi(x_0)\right\}$. Although, the minimum is situated in the x_1 direction, along the iterations the gradient is commuting from x_1 direction to x_2 direction, this slowing the convergence of the algorithm. For ill-conditioned problems, this is the typical behavior of the steepest descent algorithm.

2. The conjugate direction method

As we have already seen to avoid the zig-zagging of the steepest descent algorithm we must consider some other methods for search direction computation. We ask for an ideal line search which would have the following property.

> *„If the approximations $x_1, x_2, \ldots, x_k$ are determined by exact line searches along the directions $d_1, d_2, \ldots, d_k$, then the minimum point x_{k+1} of the functional Φ along the line $x_k + td_k$, $t \in R$, should minimize Φ on the affine subspace $x_0 + span\{d_1, \ldots, d_k\}$ which is generated by all the previous directions $d_1, d_2, \ldots, d_k$."*

Therefore, if we are able to generate linear independent directions satisfying this property, then the corresponding method is not only convergent, as we have seen above, but it is finite, because after at most n steps it finds the minimum point of Φ on R^n, since $x_0 + span\{d_1, \ldots, d_n\} = R^n$. We say that the method has the *finite termination property.*

Definition 1. *Let $A \in R^{n \times n}$ be a symmetric and positive definite matrix. The set of vectors $d_1, d_2, \ldots, d_k \in R^n$ is called $A-$conjugate if $d_j \neq 0$ for all $j = 1, \ldots, k$ and $d_j^T A d_i \neq 0$ for all $i, j = 1, \ldots, k$ and $i \neq j$.*

Theorem 2. *Let $x_0 \in R^n$ be the initial approximation of x^* and $x_1, x_2, \ldots$ iterates determined by a line search procedure for minimizing (2), where the directions $d_1, d_2, \ldots$ are mutually $A-$conjugate. Then x_{k+1} minimizes the functional Φ over the affine space $x_0 + span\{d_1, \ldots, d_k\}$.*

Proof. Consider a point $x \in x_0 + span\{d_1, \ldots, d_k\}$, then x has the following representation $x = \tilde{x} + td_k$, where $t \in R$ and

$$\tilde{x} = x_0 + \sum_{j=1}^{k} \xi_j d_j.$$

With this we have

$$\Phi(x) = \Phi(\tilde{x}) + t\sum_{j=1}^{k} \xi_j d_j^T A d_k - td_k^T r_0 + \frac{1}{2} t^2 d_k^T A d_k.$$

If $d_j^T A d_k = 0$ for all $j = 1, \cdot\cdot, k$, then

$$\Phi(x) = \Phi(\tilde{x}) + \frac{1}{2} t^2 d_k^T A d_k - td_k^T r_0$$

and the problem of minimizing Φ over $x_0 + span\{d_1, \ldots, d_k\}$ is *decoupled* into two minimizing sub-problems. The first one consists of minimizing the functional Φ over $x_0 + span\{d_1, \ldots, d_k\}$ and the second one is minimizing the functional

$$\Psi(t) = \frac{1}{2} t^2 d_k^T A d_k - td_k^T r_0,$$

which is independent by $\tilde{x}$ but depends only by the current search direction d_k.
The minimum t_k of Ψ is

$$t_k = \frac{d_k^T r_0}{d_k^T A d_k}.$$

But, we have

$$\begin{aligned} d_k^T r_k &= d_k^T \left(b - A\left(x_0 + \sum_{j=1}^{k-1} t_j d_j \right) \right) \\ &= d_k^T \left(b - Ax_0 - \sum_{j=1}^{k-1} t_j A d_j \right) \\ &= d_k^T r_0 - \sum_{j=1}^{k-1} t_j d_k^T A d_j = d_k^T r_0. \end{aligned}$$

Therefore

$$t_k = \frac{d_k^T r_k}{d_k^T A d_k}. \ ♦$$

If the directions d_j are mutually conjugate, i.e. $d_i^T A d_j = 0$, for $i \neq j$, then the minimization of Φ over the affine subspace $x_0 + span\{d_1, \ldots, d_k\}$ can be replaced by the consecutive k line searches along the directions $d_1, d_2, \ldots, d_k$. Hence, for solving the linear system $Ax = b$, the following conjugate direction algorithm can be presented.

Conjugate direction algorithm

```
1. Select $x_0$ and set $r_0 = b - Ax_0$.
2. for $k = 0,1,\ldots$ until convergence do:
3.     choose $d_k$ such that $d_k^T A d_j = 0$, for $j = 0,\ldots,k-1$ and $d_k^T r_k \neq 0$,
4.     $t_k = (d_k^T r_k)/(d_k^T A d_k)$,
5.     $x_{k+1} = x_k + t_k d_k$,
6.     $r_{k+1} = r_k - Ax_{k+1}$.
7. end for ♦
```

3. The conjugate gradient method

As we know, two vectors $x, y \in R^n$ are $A-$conjugate if and only if they are orthogonal with respect to the scalar product $\langle x, y \rangle_A = y^T A x$. Therefore, if $d_1, d_2, \ldots, d_k$ are $A-$conjugate, then they are linearly independent. Given a set of linearly independent vectors $v_1, \ldots, v_n \in R^n$, we can construct a set of n $A-$conjugate vectors using the Gram-Schmidt orthogonalization algorithm of $v_1, \ldots, v_n$ with respect to the inner product $\langle .,. \rangle_A$.

To implement the conjugate direction algorithm we have to specify a procedure for the directions d_k selection in such a manner that $d_k A d_j = 0$, for all $j = 0, \ldots, k-1$ and $d_k^T r_k \neq 0$. The first condition is satisfied by the $A-$orthogonal complement of any vector

$v \notin span\{d_1,\dots,d_k\}$ with respect to $span\{d_1,\dots,d_k\}$. To fulfill the second requirement we select $v = r_k$, i.e.

$$d_k = r_k - \sum_{j=1}^{k-1} \frac{d_j^T A r_k}{d_j^T A d_j} d_j. \tag{15}$$

Therefore,

$$d_k^T r_k = r_k^T r_k - \sum_{j=1}^{k-1} \frac{d_j^T A r_k}{d_j^T A d_j} d_j^T r_k = r_k^T r_k > 0,$$

which means that d_k is an direction satisfying the above second requirement. The following proposition shows that the sum in (15) reduces to the last term.

Proposition 1. *Let $x_0 \in R^n$ be given and assume that $x_1,\dots,x_k$ are generated by the conjugate direction algorithm, where directions are given by (15). Then the vectors r_j and d_j satisfy:*

$$r_k^T r_j = 0, \quad j = 0,\dots,k-1, \tag{16}$$

$$d_j^T A r_k = 0, \quad j = 1,\dots,k-1. \tag{17}$$

Proof. From (15) we have

$$r_j = d_j + \sum_{i=1}^{j-1} \frac{d_i^T A r_j}{d_i^T A d_i} d_i.$$

Now

$$r_k^T r_j = r_k^T d_j + \sum_{i=1}^{j-1} \frac{d_i^T A r_j}{d_i^T A d_i} r_k^T d_i.$$

But $r_k^T d_i = 0$, for all $i = 1,\dots,j$, i.e. (16) holds. To prove (17), for $j = 1,\dots,k-1$, we can write:

$$d_{j+1} = r_{j+1} - \sum_{i=1}^{j} \frac{d_i^T A r_{j+1}}{d_i^T A d_i} d_i = b - A x_{j+1} - \sum_{i=1}^{j} \frac{d_i^T A r_{j+1}}{d_i^T A d_i} d_i$$

$$= b - A\left(x_j + t_j d_j\right) - \sum_{i=1}^{j} \frac{d_i^T A r_{j+1}}{d_i^T A d_i} d_i.$$

Since $t_j \neq 0$, we have

$$A d_j = \frac{1}{t_j}\left(r_j - d_{j+1} - \sum_{i=1}^{j} \frac{d_i^T A r_{j+1}}{d_i^T A d_i} d_i \right)$$

and for $j = 1,\dots,k-1$

$$r_k^T A d_j = \frac{1}{t_j}\left(r_k^T r_j - r_k^T d_{j+1} - \sum_{i=1}^{j} \frac{d_i^T A r_{j+1}}{d_i^T A d_i} r_k^T d_i \right) = 0,$$

i.e. (17) hold. ♦

Therefore, from (15), using proposition 1, it follows that

$$d_k = r_k - \frac{d_{k-1}^T A r_k}{d_{k-1}^T A d_{k-1}} d_{k-1} = r_k + \beta_k d_{k-1}. \tag{18}$$

Hence, in this process it is not necessary to store the search directions along the iterations to perform the $A-$orthogonalization, but only the most recent direction.

Remark 2. It is possible to improve the efficiency of the algorithm, observing that

$$r_{k+1} = b - Ax_{k+1} = b - A(x_k + t_k d_k) = r_k - t_k A d_k. \qquad (19)$$

This saves one matrix-vector multiplication Ax_{k+1} in every iteration, because the product Ad_k is already computed in step 4 of the conjugate direction algorithm. ♦

With this, for $k \geq 2,$ from (19) we have

$$Ad_{k-1} = \frac{1}{t_{k-1}}(r_{k-1} - r_k).$$

The orthogonality of residuals yields to

$$d_{k-1}^T A r_k = \frac{1}{t_{k-1}}(r_{k-1} - r_k)^T r_k = -\frac{1}{t_{k-1}} r_k^T r_k.$$

On the other hand

$$\begin{aligned} d_{k-1}^T A d_{k-1} &= d_{k-1}^T A(r_{k-1} + \beta_{k-1} d_{k-2}) = d_{k-1}^T A r_{k-1} \\ &= \frac{1}{t_{k-1}}(r_{k-1} - r_k)^T r_{k-1} = \frac{1}{t_{k-1}} r_{k-1}^T r_{k-1}. \end{aligned}$$

Therefore, from (18) we get the following expression for the direction:

$$d_k = r_k + \frac{r_k^T r_k}{r_{k-1}^T r_{k-1}} d_{k-1} \equiv r_k + \beta_k d_{k-1}. \qquad (20)$$

Additionally, it is possible to get a new algebraic expression for the step length t_k. Indeed

$$d_k^T r_k = (r_k + \beta_k d_{k-1})^T r_k = r_k^T r_k,$$

which was computed in (20).
Having in view algebraic manipulations, finally we arrive to the following algorithm of conjugate gradients, which basically is the method of Hestenes and Stiefel [1952].

Conjugate gradient algorithm

```
1. Select $x_0$ and set $r_0 = b - Ax_0$. Set $\alpha_0 = r_0^T r_0$ and $d_0 = r_0$.
2. for $k = 0,1,...$ until convergence do:
3.     $s_k = Ad_k$,
4.     $t_k = \alpha_k / (d_k^T s_k)$,
5.     $x_{k+1} = x_k + t_k d_k$,
6.     $r_{k+1} = r_k - t_k s_k$,
7.     $\alpha_{k+1} = r_{k+1}^T r_{k+1}$,
8.     $\beta_{k+1} = \alpha_{k+1} / \alpha_k$,
9.     $d_{k+1} = r_{k+1} + \beta_{k+1} d_k$.
7. end for ♦
```

We already know that the solution of (1) satisfies

$$\left\| x - x^* \right\|_A^2 = 2\left(\Phi(x) - \Phi(x^*) \right) \qquad (21)$$

for every $x \in R^n$. Hence, the estimate x_k minimizes the error $e = x - x^*$ with respect to the norm $\|.\|_A$ over the affine space $x_0 + span\{d_0, \dots, d_k\}$. The representation of $span\{d_0, \dots, d_k\}$ is given by the following proposition which is important to establish the error bound for the conjugate gradient algorithm.

Definition 2 *Let $A \in R^{n\times n}$ and $b \in R^n$ be given. Then the linear space*

$$K_k(b, A) = span\{b, Ab, \dots, A^{k-1}b\}$$

is called the $k-$ the Krylov space of A corresponding to b.

Proposition 2. *Assume that the search directions $d_0, \dots, d_k$ are defined by the conjugate gradient algorithm. Then*

$$span\{d_0, \dots, d_k\} = span\{r_0, Ar_0, \dots, A^k r_0\}.$$

Proof. For $k = 0$ the statement is trivial because $d_0 = r_0$. Suppose that it already has been shown for some $k < n$. Then

$$\begin{aligned} d_{k+1} &= r_{k+1} + \beta_{k+1} d_k \\ &= r_k - t_k A d_k + \beta_{k+1} d_k \\ &= d_k - \beta_k d_{k-1} - t_k A d_k + \beta_{k+1} d_k. \end{aligned} \tag{22}$$

But, $d_k, d_{k-1} \in K_k(r_0, A)$. Hence

$$span\{d_0, \dots d_{k+1}\} \subset K_{k+1}(r_0, A).$$

The inverse inclusion follows from (22) and $t_k \neq 0$. ♦

To prove the convergence of the conjugate gradient algorithm we must consider an expression for the error. As before denote by $e_0 = x^* - x_0$, the error of x_0. Then from $r_0 = b - Ax_0 = A(x^* - x_0) = Ae_0$ it follows that $A^j r_0 = A^{j+1} e_0$, i.e. the elements of the Krylov space $z \in K_k(r_0, A)$ can be written as

$$z = \sum_{j=1}^{k} \alpha_j A^j e_0, \quad \alpha_j \in R.$$

Therefore, the error $e_k = x^* - x_k$ corresponding to the k-th iteration x_k satisfies

$$\begin{aligned} \|e_k\|_A^2 &= \min_{x \in x_0 + K_k(r_0, A)} \|x^* - x\|_A^2 \\ &= \min_{z \in K_k(r_0, A)} \|x^* - x_0 + z\|_A^2 \\ &= \min_{\alpha_1, \dots, \alpha_k \in R} \|e_0 + \alpha_1 A e_0 + \dots + \alpha_k A^k e_0\|_A^2. \end{aligned} \tag{23}$$

Now, let us consider P_k as the set of polynomials p of maximum degree k such that $p(0) = 1$. Then (23) can be expressed as

$$\|e_k\|_A^2 = \min_{p \in P_k} \|p(A) e_0\|_A^2, \tag{24}$$

showing that the error of the k-th iterate can be expressed as the product of a polynomial $p(A)$ in matrix A and the initial error e_0.

In order to get a more convenient form of the error let us consider the spectral decomposition of A as

$$A = \sum_{j=1}^{n} \lambda_j z_j z_j^T, \tag{25}$$

where z_j, $j = 1,\dots,n$, is an orthonormal set of eigenvectors corresponding to the eigenvalues $0 < \lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$. Observe that

$$A^2 = \sum_{i,j=1}^{n} \lambda_i \lambda_j z_i z_i^T z_j z_j^T = \sum_{j=1}^{n} \lambda_j^2 z_j z_j^T,$$

since $z_i^T z_j = \delta_{ij}$. By induction we can prove that

$$A^i = \sum_{j=1}^{n} \lambda_j^i z_j z_j^T.$$

The error e_0 can be expressed as

$$e_0 = \sum_{j=1}^{n} \gamma_j z_j. \tag{26}$$

Having in view the above developments, it follows that the error of the k-th iterate can be written as

$$\left\| e_k \right\|_A^2 = \min \left\| \sum_{j=1}^{n} \gamma_j p(\lambda_j) z_j \right\|_A^2 . \tag{27}$$

But

$$\left\| \sum_{j=1}^{n} \gamma_j p(\lambda_j) z_j \right\|_A^2 = \left(\sum_{i=1}^{n} \gamma_i p(\lambda_i) z_i \right) A \left(\sum_{j=1}^{n} \gamma_j p(\lambda_j) z_j \right)$$

$$= \sum_{j=1}^{n} \gamma_j^2 \lambda_j \left(p(\lambda_j) \right)^2 . \tag{28}$$

Therefore, we get a very important representation of the error of x_k as:

$$\left\| e_k \right\|_A^2 = \min_{p \in P_k} \sum_{j=1}^{n} \gamma_j^2 \lambda_j \left(p(\lambda_j) \right)^2 . \tag{29}$$

Using (29) we can determine a rough upper bound of the convergence properties of the conjugate gradient algorithm. Indeed, from

$$\left\| e_0 \right\|_A^2 = \sum_{j=1}^{n} \lambda_j \gamma_j^2$$

we obtain an estimation of the values $\left| p(\lambda_j) \right|$ by the maximum of $|p|$ on the spectrum of A:

$$\left\| e_k \right\|_A \le \min_{p \in P_k} \max_{\lambda \in \sigma(A)} \left| p(\lambda) \right| \left\| e_0 \right\|_A . \tag{30}$$

The constant

$$\min_{p \in P_k} \max_{\lambda \in \sigma(A)} \left| p(\lambda) \right|$$

is not easy to evaluate. However, the following bound is useful

$$\max_{\lambda \in \sigma(A)} \left| p(\lambda) \right| \le \max_{\lambda_1 \le \lambda \le \lambda_n} \left| p(\lambda) \right|$$

to get the following estimation

$$\min_{p \in P_k} \max_{\lambda \in \sigma(A)} |p(\lambda)| \le \max_{\lambda_1 \le \lambda \le \lambda_n} |\overline{p}_k(\lambda)|,$$

where $\overline{p}_k(x)$ is a k-degree polynomial for which $\overline{p}_k(0)=1$ and it is easy to evaluate $\max\limits_{\lambda_1 \le \lambda \le \lambda_n} |\overline{p}_k(\lambda)|$.

The Chebyshev polynomials of the first kind are the right polynomial for this estimate. This polynomial has the following definition in the interval $[-1,1]$:

$$T_k(x) = \cos\left(k \arccos(x)\right). \tag{31}$$

It is easy to see that $T_k(x)$ is a polynomial by the use of the following identities:

$$\cos(\alpha+\beta) = \cos\alpha\cos\beta - \sin\alpha\sin\beta,$$
$$\cos(\alpha+\beta)+\cos(\alpha-\beta) = 2\cos\alpha\cos\beta.$$

Let us denote $\theta = \arccos(x)$. Then we have

$$\begin{aligned} T_0(x) &= \cos(0\theta) = 1, \\ T_1(x) &= \cos(1\theta) = x, \\ T_2(x) &= \cos(2\theta) = \cos^2\theta - \sin^2\theta = 2\cos^2\theta - 1 = 2x^2 - 1, \\ T_{k+1}(x) + T_{k-1}(x) &= \cos((k+1)\theta) + \cos((k-1)\theta) \\ &= 2\cos(k\theta)\cos\theta = 2xT_k(x). \end{aligned}$$

Generally we have the following recurrence:

$$T_0(x) = 1,\ \ T_1(x) = x,$$
$$T_{k+1}(x) = 2xT_k(x) - T_{k-1}(x).$$

Solving the above recurrence equation we obtain the explicit form of the Chebyshev polynomials as:

$$T_k(x) = \frac{1}{2}\left[\left(x+\sqrt{x^2-1}\right)^k + \left(x-\sqrt{x^2-1}\right)^k\right],$$

which is valid in the interval $(-\infty,\infty)$, figures 1-4.

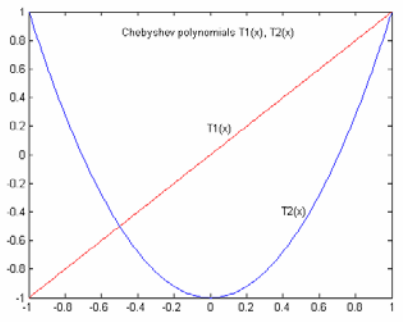

Fig.1. Chebyshev polynomials $T_1(x), T_2(x)$.

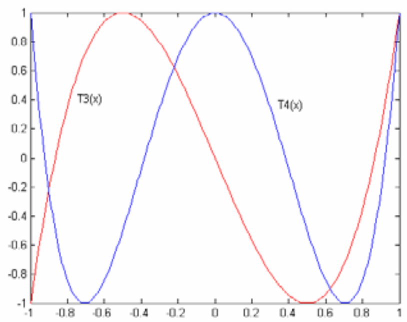

Fig. 2. Chebyshev polynomials $T_3(x), T_4(x)$.

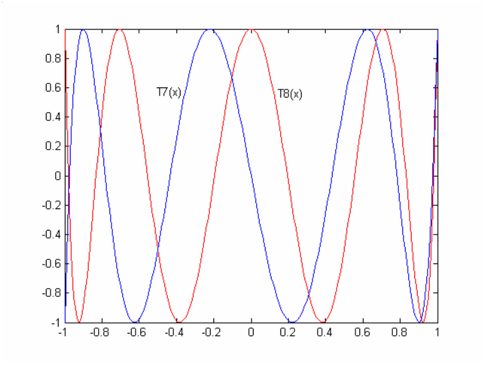

Fig.3. Chebyshev polynomials $T_7(x), T_8(x)$.

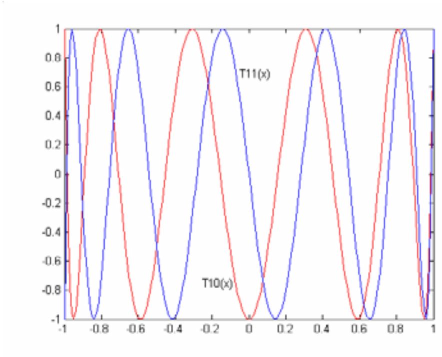

Fig.4. Chebyshev polynomials $T_{10}(x), T_{11}(x)$.

The translated and scaled Chebyshev polynomial is defined as

$$T_k(x;a,b) = T_k\left(\frac{a+b-2x}{b-a}\right),$$

where we have $\left|T_k(x;a,b)\right| < 1$ for all $x \in [a,b]$.

The Chebyshev polynomials are very useful for convergence study of iterative methods. Their most interesting property in this respect is that they have the smallest maximum norm. As we have already said, we are interesting in solving the minimization problem:

$$\underset{p\in P_k}{min}\ \underset{x\in[a,b]}{max}\left|p(x)\right|.$$

A solution to this problem is given by the shifted and scaled Chebyshev polynomial

$$\underset{p\in P_k}{min}\ \underset{x\in[a,b]}{max}\left|p(x)\right| = \max\left|\frac{T_k\left(\frac{2x-(a+b)}{b-a}\right)}{T_k\left(\frac{a+b}{b-a}\right)}\right| = \frac{1}{\left|T_k\left(\frac{a+b}{b-a}\right)\right|}.$$

If $a=-b$, then $T_k\left(\frac{a+b}{b-a}\right) = 1$, but in more general cases we need an upper bound for $\left|T_k\left(\frac{a+b}{b-a}\right)\right|^{-1}$. To obtain this bound we use the following proposition

Proposition 3. *If* $0<a<b$, *then*

$$\underset{p\in P_k}{min}\ \underset{x\in[a,b]}{max}\left|p(x)\right| \le 2\left(\frac{\sqrt{\frac{b}{a}}-1}{\sqrt{\frac{b}{a}}+1}\right)^k.$$

Proof. We have $x=\frac{b+a}{b-a}$ so $x^2-1=\frac{2\sqrt{ab}}{b-a}$, $x+\sqrt{x^2-1}=\frac{(\sqrt{b}-\sqrt{a})^2}{b-a}=\frac{\sqrt{b}+\sqrt{a}}{\sqrt{b}-\sqrt{a}}$ and $x-\sqrt{x^2-1}=\frac{\sqrt{b}-\sqrt{a}}{\sqrt{b}+\sqrt{a}}$. Therefore

$$T_k\left(\frac{b+a}{b-a}\right) \geq \frac{1}{2}\left(\frac{\sqrt{b}+\sqrt{a}}{\sqrt{b}-\sqrt{a}}\right)^k . \ ♦$$

With this the following theorem of convergence rate of conjugate gradient algorithm can be proved.

Theorem 3. *Let $A \in R^{n\times n}$ be symmetric and positive definite and denote by λ_1 and λ_n the smallest and the largest eigenvalue of A, respectively. Let x_0 be any initial vector and x_k be the k-th approximate to the solution x^* of $Ax = b$ obtained by means of the conjugate gradient algorithm. Then the following error estimate*

$$\left\|x_k - x^*\right\|_A \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^k \left\|x_0 - x^*\right\|_A \tag{32}$$

holds, where $\kappa = \lambda_n / \lambda_1$ is the condition number of A.

Proof. From the estimate $\|e_k\|_A \leq \max\limits_{\lambda_1\leq\lambda\leq\lambda_n} |p_k(\lambda)| \|e_0\|_A$ choosing

$$p_k(x) = \frac{T_k(x;\lambda_1,\lambda_n)}{T_k(0;\lambda_1,\lambda_n)}$$

and having in view that $|T_k(x;\lambda_1,\lambda_n)| < 1$ for $x \in [\lambda_1,\lambda_n]$, we have

$$\|e_k\|_A \leq \frac{\|e_0\|_A}{T_k(0;\lambda_1,\lambda_n)} = T_k\left(\frac{\lambda_n+\lambda_1}{\lambda_n-\lambda_1}\right)^{-1} \|e_0\|_A .$$

Now, using the proposition 3 we get the conclusion of the theorem. ♦

Therefore, $\|e_k\|_A$ is bounded above by a sequence that converges to zero. Moreover, the decrease is monotone and this explains why the method of conjugate gradient can be regarded as an iterative one. Observe that the closer κ is to one, the faster the convergence of the method. As we can see, the convergence of the algorithm depends on the distribution of *all* eigenvalues of the matrix A.

Theorem 3 provides very coarse upper bounds of the speed of convergence of the conjugate gradient algorithm. Besides, the finite termination property is not reflected by the conclusion (32) of the theorem. Since the matrix A has at most n distinct eigenvalues, then for every polynomial $q \in P_k$, for which $q(\lambda_j) = 0$, $j = 1,\ldots,n$, from (29) it follows:

$$\|e_k\|_A^2 = \min_{p\in P_k} \sum_{j=1}^{n} \gamma_j^2 \lambda_j \left(p(\lambda_j)\right)^2 \leq \sum_{j=1}^{n} \gamma_j^2 \lambda_j \left(q(\lambda_j)\right)^2 = 0. \tag{33}$$

Theorem 4. *Let $A \in R^{n\times n}$ be symmetric and positive definite and x_k be the approximate to the solution x^* of (1) given by the conjugate gradient algorithm. If A has $m \leq n$ distinct eigenvalues, then in exact arithmetic, the conjugate gradient algorithm achieves $x_k = x^*$ after at most m iterations.*

Proof. If A has the distinct eigenvalues $\lambda_1 < \cdots < \lambda_m$, then selecting

$$q(\lambda) = \prod_{j=1}^{m} \left(\lambda - \lambda_j\right)$$

in (33) we get $\|e_m\|_A^2 = 0.$ ♦

The following result was proven by Axelsson (1976).

Theorem 5. *Assume that* $a \le \lambda_1 \le \lambda_2 \le \cdots \le \lambda_{n-m} \le b \le \lambda_{n-m+1} \le \cdots \le \lambda_n$ *and let* $k \ge m$, *then*

$$\|x_k - x^*\|_A \le 2\left(\frac{\sqrt{\frac{b}{a}}-1}{\sqrt{\frac{b}{a}}+1}\right)^{(k-m)} \|x_0 - x^*\|_A .$$

Proof. In the estimate $\|e_k\|_A \le \max\limits_{\lambda_1 \le \lambda \le \lambda_n} |p_k(\lambda)| \|e_0\|_A$ select

$$p_k(\lambda) = \prod_{j=1}^{m}\left(1 - \frac{\lambda}{\lambda_{n-j+1}}\right)\frac{T_{k-m}\left(\frac{a+b-2\lambda}{b-a}\right)}{T_{k-m}\left(\frac{a+b}{b-a}\right)}.$$

Obviously, $p_k(\lambda_j) = 0$ for all λ_j, with $j = n, n-1, \ldots, n-m+1$. But, when $\lambda \in [a,b]$,

$$\left|1 - \frac{\lambda}{\lambda_{n-j+1}}\right| < 1, \quad j = 1, \ldots, m.$$

Hence,

$$\max_{a \le \lambda \le b} |p_k(\lambda)| \le \max_{a \le \lambda \le b}\left|\frac{T_{k-m}\left(\frac{a+b-2\lambda}{b-a}\right)}{T_{k-m}\left(\frac{a+b}{b-a}\right)}\right|,$$

which prove the theorem. ♦

This result is very useful when a few of the largest eigenvalues are well separated from the rest of eigenvalues. In this case, m is small and $k-m$ is not too much different from k.

The matrix A seldom has a small number $m << n$ of distinct eigenvalues, but more plausibly its spectrum is distributed in a number of m disjoint intervals I_j, $j = 1, \ldots, m$, of small length. In this case, after m iterations, the conjugate gradient algorithm will produce a small residual. Indeed, selecting a polynomial q of degree m with $q(0) = 1$ having the eigenvalues distributed in these intervals I_j, then the upper bound of the reduction of the error in m iterations, i.e. $\max\left\{|q(\lambda_j)|\right\}$, ($\lambda_j$ being the eigenvalues of A), will be small. The following example illustrates this situation.

Example 2. In order to see the effect of clustered eigenvalues on the speed of convergence of the conjugate gradient algorithm, let us consider the linear system $Ax = b$, where A is a diagonal matrix and b is chosen in such a way that the solution of the system is always $x = [1,1,...,1]$. The criterion for stopping the iteration is $\|b - Ax_k\| \le 10^{-6}$. Consider $n = 1000$ and four distributions of the eigenvalues: $A = diag(1,2,...,n)$ with $\kappa(A) = 1000$; a diagonal matrix with elements uniformly distributed in $[0,1)$ with $\kappa(A) = 994.637$; a diagonal matrix with elements distributed in 10 intervals with $\kappa(A) = 19.0178$, and a diagonal matrix with elements distributed in 5 intervals with $\kappa(A) = 9.00975$.

Figure 5 illustrate the speed of convergence of the conjugate gradient algorithm for these four cases in a semi-logarithmic scale.

Figure 6 illustrates the situation in which the matrix A has 999 eigenvalues uniformly distributed in $[0,1)$ and the last one equal with 1000. In this case condition number of A is $\kappa(A) = 996182.625$.

This is a typical behavior of the conjugate gradient algorithm. Observe that when the eigenvalues are well clustered into a small number of intervals, then the reduction of the error is accelerated, only a few iterations are needed to get a solution with the imposed accuracy. If the largest and the smallest eigenvalues of A are few in number (or clustered closely together), then the conjugate gradient algorithm will converge much more quickly than the analysis based just on A's condition number would indicate.

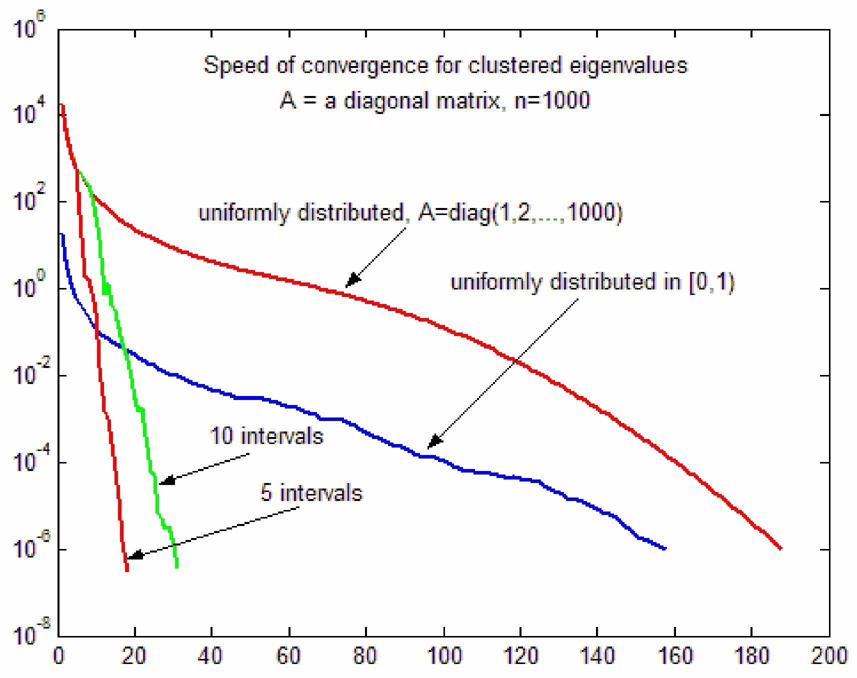

Fig. 5. Speed of convergence for clustered eigenvalues.
Evolution of $\|b - Ax_k\|_2$, subject to the number of iterations.

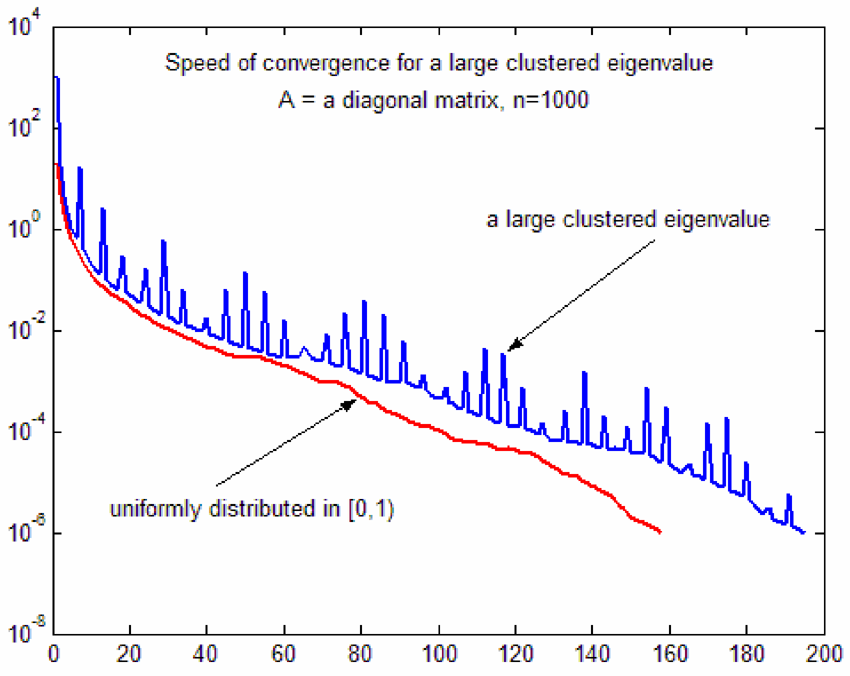

Fig. 6. Speed of convergence for a large clustered eigenvalue.
Evolution of $\|b - Ax_k\|_2$, subject to the number of iterations.

Now consider the situation in which the matrix A is diagonal, with $n = 1000$, and with different sets of distinct eigenvalues. The convergence properties are summarized in the following table:

# of distinct eigenvalues	2	10	20	50	100	500	1000
# of iterations to converge	3	11	21	43	62	142	188

We see that the number of iterations to converge grows with the number of distinct eigenvalues. In absence of roundoff, we know that conjugate gradient would take exactly the same number of iterations as the number of distinct eigenvalues (see theorem 4). However, in floating point arithmetic the behavior of conjugate gradient algorithm can differ significantly from its behavior in exact arithmetic [Demmel, 1997].

Example 3. Let us consider the system $Ax = b$, where:

$$A = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix}.$$

Figure 7 illustrates the evolution of $\|b - Ax_k\|_2$, subject to the number of iterations to get the solution with 10^{-6} accuracy.

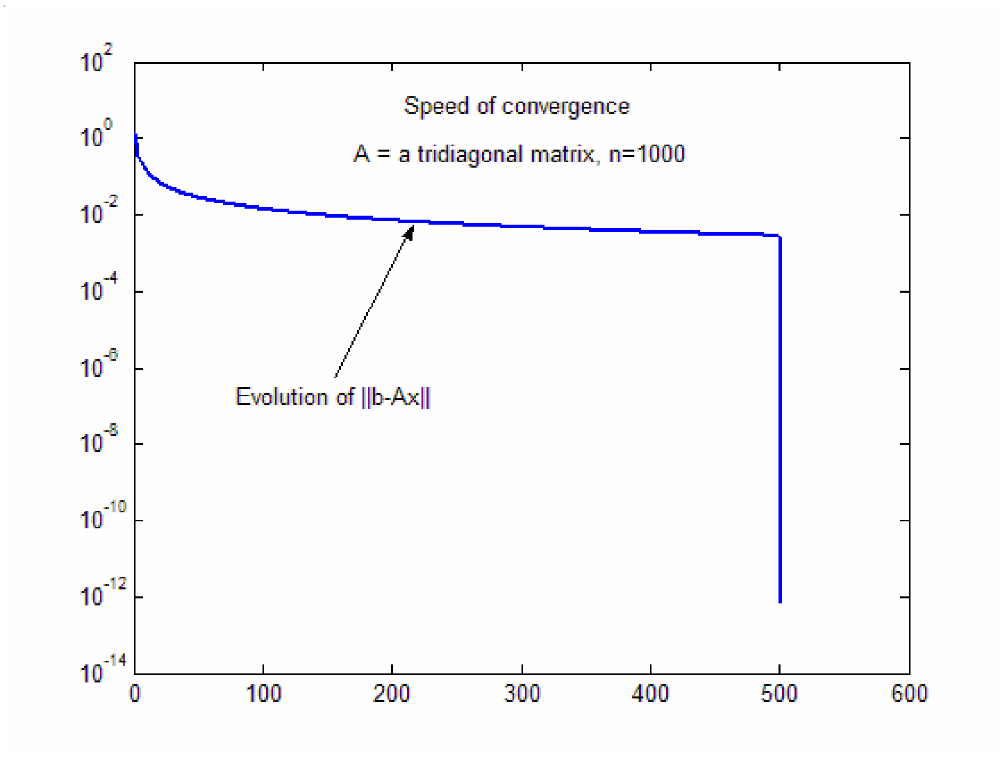

Fig. 7. Speed of convergence.
Evolution of $\|b - Ax_k\|_2$, subject to the number of iterations.

The eigenvalues of A are $\lambda_i = 2\left(1-\cos\frac{\pi i}{n+1}\right)$. Figure 8 is a plot of the eigenvalues of A for $n = 21$. Observe that the largest eigenvalue is $\lambda_n = 2\left(1-\cos\frac{\pi n}{n+1}\right) \approx 4$. The smallest eigenvalue is $\lambda_1 = 2\left(1-\cos\frac{\pi}{n+1}\right)$. For small i we have

$$\lambda_i = 2\left(1-\cos\frac{\pi i}{n+1}\right) \approx 2\left(1-\left(1-\frac{\pi^2 i^2}{2(n+1)^2}\right)\right) = \left(\frac{\pi i}{n+1}\right)^2.$$

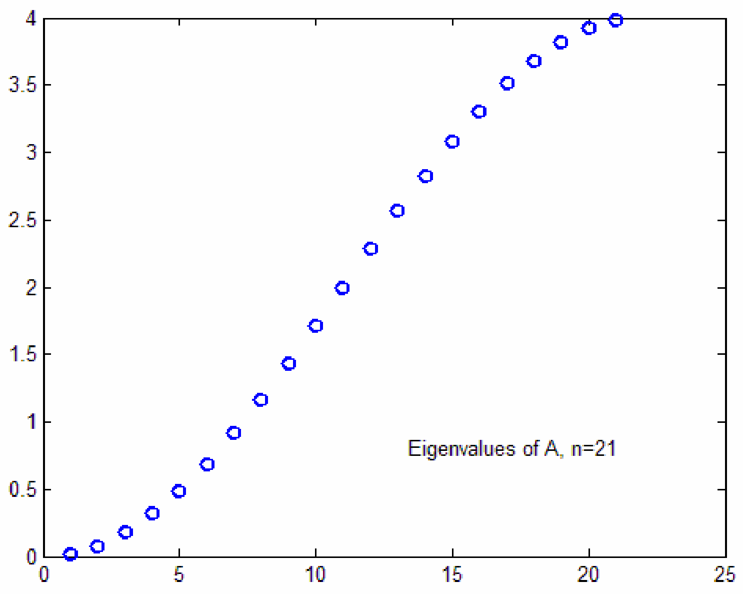

Fig. 8. The eigenvalues of A.

Therefore, A is positive definite with condition number

$$\frac{\lambda_n}{\lambda_1} \approx \frac{4(n+1)^2}{\pi^2}, \text{ for large } n.$$

In this case the behavior includes long „plateaus” in the convergence, with $\|b - Ax_k\|_2$ decreasing little for many iterations. An explanation of this behavior is given in Greenbaum and Strakos [1992] where it is shown that the conjugate gradient algorithm applied to the system $Ax = b$ in floating point arithmetic behaves exactly like conjugate gradient applied to $\tilde{A}\tilde{x} = \tilde{b}$ in exact arithmetic, where $\tilde{A}$ is close to A in the following sense: $\tilde{A}$ has a much larger dimension than A, but the eigenvalues of $\tilde{A}$ all lie in narrow clusters around the eigenvalues of A.

Example 4. Let us consider the system $Ax = b$, in which:

$$A = \begin{bmatrix} B & -I & & & \\ -I & B & -I & & \\ & \ddots & \ddots & \ddots & \\ & & -I & B & -I \\ & & & -I & B \end{bmatrix}, \text{ where } B = \begin{bmatrix} 4 & -1 & & & \\ -1 & 4 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 4 & -1 \\ & & & -1 & 4 \end{bmatrix}$$

and $B \in R^{n_1 \times n_1}$, the matrix A having n_2 blocks on the main diagonal. Therefore, $A \in R^{n \times n}$, where $n = n_1 n_2$. The right-hand-side term b is selected in such a way that the solution of the system is $x=[1,1,...,1]^T$. Considering $n=5000$, the speed of convergence for different values of n_1 and n_2 is illustrated in Figure 9.

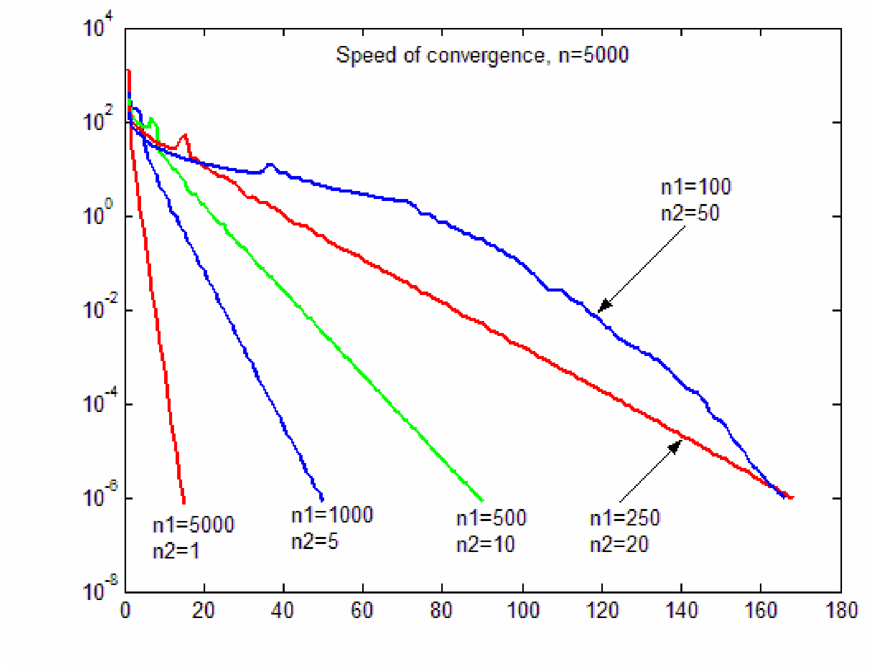

Fig. 9. Speed of convergence.
Evolution of $\|b-Ax_k\|_2$, subject to the number of iterations.

The eigenvalues of A are:

$$\lambda_{ij}=4\sin^2\left(\frac{i\pi}{2(n_1+1)}\right)+4\sin^2\left(\frac{j\pi}{2(n_2+1)}\right)$$

Figure 10 is a plot of the eigenvalues of A for $n_1=11$ and $n_2=21$. Observe that the largest eigenvalue is $\lambda_{\max}=8$ and the smallest eigenvalue is $\lambda_{\min}=8\sin^2(\pi/2)$. Therefore, A is positive definite with condition number

$$\frac{\lambda_{\max}}{\lambda_{\min}}\approx\frac{4}{\pi^2}.$$

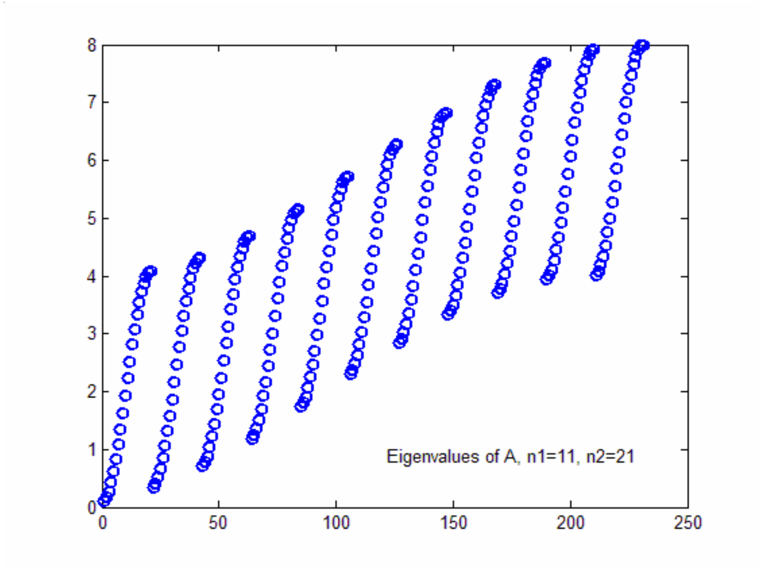

Fig. 10. The eigenvalues of A.

References

Axelsson, O., (1976) *A class of iterative methods for finite element equations*. Com. Meth. Appl. Mech. Eng., vol.9, pp.123-137. 1976

Axelsson, O., (1994) *Iterative Solution Methods*. Cambridge University Press. NY. 1994.

Faber, V. and Manteuffel, T., (1984) *Necessary and sufficient conditions for the existence of a conjugate gradient method*. SIAM J. Numer. Anal., 21, pp.352-362.

Faber V., and Manteuffel, T., (1987) *Orthogonal error methods*. SIAM J. Numer. Anal., 24, pp. 170-187.

Demmel, J.W., (1997) *Applied Numerical Linear Algebra*. SIAM, 1997.

Duff, I.S., Erisman, R.G. and Reid, J.K., (1986) *Direct Methods for Sparse Matrices*. Clarendon Press, Oxford. 1986

Golub, G.H. and Van Loan, C., (1981) *Matrix Computation*. Academic Press. 1981.

Greenbaum, A., (1997) *Iterative Methods for Solving Linear Equations*. SIAM. 1997.

Greenbaum, A. and Strakos, Z., (1992) *Predicting the behavior of finite precision Lanczos and conjugate gradient computations*. SIAM J. Matrix Anal. Appl., 13: 121-137, 1992.

Gutknecht, M.H., (1997) *Lanczos-type solvers for nonsymmetric linear systeme of equations*. Acta Numerica 6, pp.271-397.

Hackbusch, W., (1994) *Iterative solution of large linear systems of equations*. Springer Verlag, NY, 1994.

Hestenes, M.R., (1980) *Conjugate Direction Methods in Optimization*. Springer Verlag, 1980.

Hestenes, M.R. and Stiefel, E.L., (1952) *Methods of conjugate gradients for solving linear systems*. Journal of Research of the National Bureau of Standards, Section B, vol. 49, no.6, pp. 409-436, 1952.

Meurant, G., (1999) *Computer Solution of Large Linear Systems*. Elsevier, Amsterdam, 1999.

Paige, C.C. and Saunders, M.A., (1975) *Solution of sparse indefinite systems of linear equations*. SIAM J. Numer. Anal. 12, pp.617-629.

Reid, J.K., (1971) *On the method of conjugate gradients for the solution of large sparse systems of linear equations*. In J.K. Reid (Ed.) „Large Sparse Sets of Linear Equations", pages 231-254, Academic Press, 1971.

Reid, J.K., (1972) *The use of conjugate gradients for systems of linear equations possessing "Property A"*. SIAM J. Numer. Anal., vol. 9, no. 2, pp.325-332, 1972.

Saad, Y., (1996) *Iterative Methods for Sparse Linear Systems*. PWS Publishing Company. 1996.

August 22, 2005