Some Properties of Krylov Projection Methods for Large Linear Systems

Jocelyne Erhel
INRIA, Campus de Beaulieu, 35042 Rennes Cedex, France

2011

Note

This paper is the preprint version of the article published in Computational Technology Reviews [20].

Abstract

In this paper, we consider the problem of solving a large sparse linear system; after a brief introduction on existing direct, semi-iterative and iterative methods, the paper is focused on Krylov projection methods. It presents a unified framework of these methods and derives some properties from the projection condition. Then it expands these properties for the most well-known methods. Some numerical experiments illustrate convergence behavior. Finally, some practical issues such as numerical errors, preconditioning, and parallel processing are discussed.

Keywords: linear solver, Krylov subspace, Galerkin condition, preconditioning, parallel computing

1 Introduction

Solving a linear system is at the heart of many scientific and engineering applications. Generally, this operations is the most time and memory consuming part of the simulation. In this paper, we thus consider the following problem: solve the linear system

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

where $A \in \mathbb{R}^{n \times n}$ is a nonsingular matrix and $b \in \mathbb{R}^n$ the right-hand side. Let x^* be the solution.

In computational science and engineering, the system size n is quite large, ranging from 10^3 to 10^9 and more, thus requiring petascale and soon exascale computing resources. The matrix of the system is sparse, in other words, many coefficients are zero; several adapted sparse storage schemes exist and algorithms must deal with these specific ways of storing only nonzero elements. Algorithms also depend on the type of the matrix. Indeed, the matrix can be symmetric positive definite (SPD), if it arises for example from an elliptic or parabolic problem (diffusion, flow in porous media, etc). It can also be symmetric but indefinite if it arises for example from a saddle point problem. Otherwise, it is nonsymmetric, for example when the underlying problem is hyperbolic (advection, etc).

Several methods and solvers exist for these linear systems. They can be divided into three classes: direct, iterative or semi-iterative. Direct methods are highly robust and efficient but require a large memory space [16]. Moreover, the complexity increases rapidly with the size of the system. Software like SuiteSparse for sequential computers (used in Matlab) [15, 14], SuperLU [17, 34] and Mumps [1, 31] for sequential or parallel computers, are widely used in computational science and engineering. Iterative methods of multigrid type are often efficient and scalable [58]; software like HYPRE [26], implementing geometric and algebraic multigrid, is also widely used; multigrid methods are competitive compared to direct methods, especially for elliptic problems [22]. Semi-iterative methods such as subdomain methods are hybrid direct/iterative methods which can be good tradeoffs [54].

This paper focuses on some properties of Krylov iterative methods. Iterative methods of Krylov type require less memory than direct methods, but the number of iterations increases rapidly with the size of the system. The convergence rate and the accuracy of the results depend on the condition number which can blow up at large scale. Therefore, it is essential to combine these methods with a preconditioner; the idea is to solve with $M^{-1}A$ or with AM^{-1} instead of A, where M is close to A but where solving a linear system with M is fast; also, on parallel computers, it must be scalable. In Krylov iterative methods, the matrix is not transformed but the kernel operation is the matrix-vector product $y = Ax$; thus it is possible to use matrix-free versions without storing the matrix. However, preconditioning will sometimes require the matrix. Krylov methods are described in many books, such as [46],[36]; they are unified as a minimization problem in [7]; Krylov methods are thoroughly described, with many references, in the survey [50]; in [48], parallel features are discussed, whereas a survey of preconditioners is given in [6]. Software like HYPRE, pARMS [35] and PETSc [3] implement Krylov methods and various preconditioners.

Here, we follow the lines developed in [12]. Polynomial methods are defined in section 2, Krylov projection methods and their properties are given in section 3. Then, we study specific methods for the three different types of matrices: the case of SPD matrices is analyzed in section 4, followed by the case of symmetric indefinite matrices in section 5. The general case of nonsymmetric matrices is studied in section 6. Finally, some practical issues, preconditioning and parallelism are discussed in section 7.

Throughout the paper, we note x_k the approximation at iteration k, we define the residual $r_k = b - Ax_k$ and the error $e_k = x^* - x_k$ so that $r_k = Ae_k$.

2 Polynomial methods

The use of polynomial methods is motivated by the following result:

Proposition 2.1 *There exists a polynomial p of degree at most $(n-1)$ such that $A^{-1} = p(A)$.*

Proof. By the Cayley-Hamilton theorem, the characteristic polynomial q of A, defined by $q(x) = det(A - xI)$, is of degree n and verifies $q(A) = 0,\ q(0) = det(A)$, where $det(A)$ is the determinant of A. Let $q(X) = det(A) + Xq_1(X)$ then $Aq_1(A) = -det(A)I$. Since A is nonsingular, $det(A) \neq 0$ and $-\frac{1}{det(A)}Aq_1(A) = I$. Let p defined by $p(X) = -\frac{1}{det(A)}q_1(X)$, then $A^{-1} = p(A)$. ◇

Let x_0 an initial approximation and $r_0 = b - Ax_0$. Then the solution x^* satisfies $x^* = x_0 + A^{-1}r_0 = x_0 + p(A)r_0$. The objective of polynomial methods is to approximate this polynomial $p(X)$. Two different types of methods can be defined :

- in the first approach, the polynomials are explicitely defined. For example, the method CHEBY can be used for SPD matrices [4].
- in the second approach, the polynomials are defined implicitely, through their application to the matrix. Thus the coefficients are not computed. Most of polynomial methods follow this line.

Throughout the paper, the space of polynomials of degree at most k is denoted by $\mathcal{P}_k$ and the set of residual polynomials is defined by

$$\mathcal{P}_k^0 = \{q \in \mathcal{P}_k \text{ with } q(0) = 1\}.$$

Definition 2.1 *The Krylov subspace associated with A and r_0 is defined by*

$$\mathcal{K}_k(A, r_0) = eng\{r_0, Ar_0, \ldots, A^{k-1}r_0\} = \{s_{k-1}(A)r_0,\ \textit{where } s_{k-1} \in \mathcal{P}_{k-1}\}.$$

Definition 2.2 *A polynomial iterative method satisfies the subspace condition $x_k = x_0 + s_{k-1}(A)r_0,$ with $s_{k-1} \in \mathcal{P}_{k-1}$. Equivalently, the condition can be written*

$$x_k \in x_0 + \mathcal{K}_k(A, r_0),\ \textit{or } x_{k+1} \in x_k + \mathcal{K}_{k+1}(A, r_0). \tag{2}$$

Proposition 2.2 *In a polynomial method, the residual r_k and the error e_k satisfy*

$$r_k = q_k(A)r_0,\ \ e_k = q_k(A)e_0, \tag{3}$$

where $q_k(X) = 1 - Xs_{k-1}(X) \in \mathcal{P}_k^0$.

Proof. $r_k = b - A(x_0 + s_{k-1}(A)r_0) = r_0 - As_{k-1}(A)r_0 = q_k(A)r_0$ where $q_k(X) = 1 - Xs_{k-1}(X)$.
$e_k = A^{-1}r_k = A^{-1}q_k(A)r_0 = q_k(A)A^{-1}r_0 = q_k(A)e_0$. ◇

Since the problem is in finite dimension, polynomial methods compute the exact solution in a finite number of iterations. Thus they are indeed direct methods but they are used as iterative methods, in the sense that the approximate solution is accurate enough before achieving the exact solution.

Proposition 2.3 *The series of Krylov subspaces $\mathcal{K}_k(A, r_0)$ is increasing, thus is stationary from number $p \leq n$; for $k \leq p$, the dimension of $\mathcal{K}_k(A, r_0)$ is equal to k and the system $\{r_0, Ar_0, \ldots, A^{k-1}r_0\}$ is a basis of $\mathcal{K}_k(A, r_0)$. The subspace $\mathcal{K}_p(A, r_0)$ is an invariant subspace of A; the solution of $Ay = r_0$ is in this subspace.*

Proof. The first part is evident since a Krylov subspace is of dimension at most n. Let p such that the series becomes stationary. Then $\mathcal{K}_{p+1}(A, r_0) = \mathcal{K}_p(A, r_0)$ and

$$A\mathcal{K}_p(A, r_0) \subset \mathcal{K}_{p+1}(A, r_0) = \mathcal{K}_p(A, r_0),$$

which proves the invariance of the subspace. Thus the restriction of A to this subpsace is a bijection and the vector $r_0 \in \mathcal{K}_p(A, r_0)$ has an antecedent in this subspace. ◇

In practice, iterations are stopped when the residual (ideally, the error) becomes small enough. Convergence rate can be related to spectral decomposition.

Theorem 2.1 *If A is diagonalisable, let $A = U\Sigma U^{-1}$, where the columns of U are a basis of eigenvectors and where $\Sigma = diag(\lambda_1, \ldots, \lambda_n)$; let $\kappa(U) = \|U\|_2\|U^{-1}\|_2$ the condition number of U.*

Krylov iterations satisfy

$$\|r_k\|_2 \leq \|r_0\|_2 \, \kappa(U) \max_{i=1,\ldots,n} |q_k(\lambda_i)|, \tag{4}$$

where q_k is defined by (3).

Proof. We have $q_k(A) = Uq_k(\Sigma)U^{-1}$; let $r_0 = U\mu$, then $q_k(A)r_0 = Uq_k(\Sigma)\mu$.
Thus $\|q_k(A)r_0\|_2 \leq \|U\|_2\|q_k(\Sigma)\|_2\|\mu\|_2$.
But $\mu = U^{-1}r_0$ hence $\|\mu\|_2 \leq \|U^{-1}\|_2\|r_0\|_2$.
Also $\|q_k(\Sigma)\|_2 = \max_{\lambda_i} |q_k(\lambda_i)|$, giving inequality (4). ◇

2.1 Preconditioned polynomial methods

For large systems, iterations will be stopped before finding the exact solution. The convergence rate depends on properties of the matrix A. An efficient way to speed-up convergence is to transform the problem (1) into an equivalent problem by preconditioning the system with a given matrix.

Definition 2.3 *A preconditioner is defined by a nonsingular matrix $C \in \mathbb{R}^{n,n}$. A left preconditioned system is given by $CAx = Cb$, whereas a right preconditioned system is given by $AC(C^{-1}x) = b$.*

In a left preconditioned system, the matrix is thus CA, and the matrix-vector product $v = Au$ is replaced by two matrix-vector products $w = Au, v = Cw$. In general, the matrix C is the inverse of a given matrix, so that the operation $v = Cw$ amounts to solving a linear system.If $C = A^{-1}$, then $CA = I$ and the preconditioned system is trivial. The matrix C is chosen in order to approximate A^{-1}. Also, the matrix C is chosen so that solving $v = Cw$ is cheaper than solving $Ax = b$. If computations are done on parallel or distributed architectures, the preconditioner must also be parallel and scalable. To find an efficient preconditioner is a big issue; a recent survey can be found in [6]; parallel aspects are discussed in [48].

Proposition 2.4 *A left preconditioned polynomial method is characterized by $x_k \in x_0 + \mathcal{K}_k(CA, Cr_0)$, and a right preconditioned polynomial method is characterized by $x_k \in x_0 + C\mathcal{K}_k(AC, r_0)$.*

Proof. Let us consider the system $CAx = Cb$. Let x_0 the initial approximation, then the residual s_0 satisfies $s_0 = Cb - CAx_0 = C(b - Ax_0) = Cr_0$ and the polynomial p_k is applied to CA. The proof is similar for right preconditioning. ◇

3 Krylov projection methods

The objective of Krylov methods is to minimize the error e_k, for a given scalar product. This is not always possible, therefore a more general orthogonality condition is also defined.

Definition 3.1 *A Krylov projection method is a polynomial method defined by a matrix B and two conditions: the subspace condition (2) and the Petrov-Galerkin condition:*

$$(Be_k)^T u = 0, \ \forall u \in \mathcal{K}_k(A, r_0). \tag{5}$$

The choice of the matrix B defines the method. The Galerkin condition must be computable, since the error e_k is of course unknown.

3.1 Properties of Krylov projection methods

A first question is the existence and uniqueness of the iterate x_k. Then a second question is the convergence of the method.

Definition 3.2 *For x_0 given, a Krylov projection method fails at iteration k if the Galerkin condition (5) has no unique solution.*

The method stagnates if $x_{k+1} = x_k$, or equivalently $r_{k+1} = r_k$.

The method finishes if $x_k = x^*$, or equivalently $r_k = 0$.

Theorem 3.1 *Let V_k an orthonormal basis of the Krylov subspace $\mathcal{K}_k(A, r_0)$ and let $C_k = V_k^T B V_k$. The Krylov projection method does not fail at iteration k if and only if the matrix C_k is nonsingular. If the method does not fail, then*

$$e_k = P_k e_0 \text{ where } P_k = I - V_k C_k^{-1} V_k^T B$$

is the projection matrix onto $(B^T \mathcal{K}_k(A, r_0))^\perp$ parallel to $\mathcal{K}_k(A, r_0)$.

The iterate x_k is defined by $x_k = x_0 + V_k y$ where y is solution of the linear system

$$C_k y = V_k^T B e_0. \tag{6}$$

If the method did not fail until iteration k and if $\mathcal{K}_{k+1}(A, r_0) = \mathcal{K}_k(A, r_0)$, then the method has finished, thus $r_k = e_k = 0$.

As long as the method does not fail and does not finish, then $dim(\mathcal{K}_k(A, r_0)) = k$.

If the method does not fail, then it converges in at most n iterations.

Proof. The subspace condition is written $x_k = x_0 + V_k y$ and the Galerkin condition is written $V_k^T B e_k = 0$, or $V_k^T B e_0 - C_k y = 0$. This linear system has a unique solution if and only if C_k is nonsingular.

If C_k is nonsingular then

$$\begin{aligned} y &= C_k^{-1} V_k^T B e_0, \\ e_k &= e_0 - V_k y = (I - V_k C_k^{-1} V_k^T B) e_0 = P_k e_0. \end{aligned}$$

If $\mathcal{K}_{k+1}(A, r_0) = \mathcal{K}_k(A, r_0)$, then $A^{-1} r_0 \in \mathcal{K}_k(A, r_0)$, thus $x^* = x_0 + A^{-1} r_0$ is solution of the Galerkin condition. If the method did not fail, this solution is unique and $x_k = x^*$.

When the series of Krylov subspaces is stationary, the method finishes. ◇

Theorem 3.2 *If the matrix B is definite, the associated Krylov projection method does not fail for any x_0. If B is not definite, there exists x_0 such that the associated Krylov projection method fails.*

Proof. If B is definite then $\forall z \neq 0,\ V_k z \neq 0,\ (V_k z)^T B (V_k z) \neq 0$ and $z^T C_k z \neq 0$ thus $C_k z \neq 0$ and C_k is nonsingular.

If B is not definite, let $z \neq 0$ such that $z^T B z = 0$, $r_0 = z$ and $V_1 = \{r_0 / \|r_0\|\}$. Then $C_1 = 0$ and system (6) has no unique solution (it can have an infinity of solutions if $r_0^T B A r_0 = 0$). ◇

When the matrix B defines a scalar product, the Petrov-Galerkin condition can be written as a minimization problem and convergence can be characterized.

Proposition 3.1 *If B is SPD, then for x_k satisfying (2), the condition (5) is equivalent to minimize the error:*

$$\|e_k\|_B = \min_{y \in \mathcal{K}_k(A,r_0)} \|e_0 - y\|_B, \tag{7}$$

Also, convergence is monotoneous: $\|e_k\|_B \leq \|e_{k-1}\|_B$.

If, moreover, BA^{-1} is definite, then convergence is strictly monotoneous: there exists $\epsilon > 0$, such that for any k,

$$\|e_k\|_B \leq (1-\epsilon)\|e_{k-1}\|_B.$$

Proof. Let $x_k = x_0 + y$ with $y \in \mathcal{K}_k(A, r_0)$. The condition (5) means that $e_k = x^* - x_k = x^* - (x_0 + y) = e_0 - y$ must belong to the B-orthogonal of the subspace $\mathcal{K}_k(A, r_0)$. Since $e_k \in e_0 + \mathcal{K}_k(A, r_0)$, it follows that e_k is the B-orthogonal projection of e_0 onto $(\mathcal{K}_k(A, r_0))^{\perp_B}$. The vector y, solution of the least-squares problem (7) is thus the B-orthogonal projection of e_0 onto $\mathcal{K}_k(A, r_0)$.

The minimization condition can be written

$$\|e_k\|_B = \min_{x \in x_0 + \mathcal{K}_k(A,r_0)} \|x - x^*\|_B.$$

It is easy to apply this condition to x_{k-1} to get monotoneous convergence.

The proof of strictly monotoneous convergence is done for example in [33]. ◇

3.2 Computing the orthonormal basis V_k

For using theorem 3.1, each iteration of the method requires computing a basis V_k of the Krylov subspace $\mathcal{K}_k(A, r_0)$, as long as this Krylov subspace is of dimension k. When the matrix is symmetric, the Lanczos process does the job; when the matrix is nonsymmetric, both bi-Lanczos process and Arnoldi process compute this basis.

Proposition 3.2 *Let us assume that A is symmetric. Let $v_1 = r_0/\|r_0\|_2$. The Lanczos process builds the orthonormal V_k which satisfies*

$$AV_k = V_k T_k + \delta_k v_{k+1} u_k^T = V_{k+1} \overline{T}_k, \tag{8}$$

where $T_k \in \mathbb{R}^{k,k}$ is a tridiagonal symmetric matrix, $u_k^T = (0 \ldots 0\ 1) \in \mathbb{R}^k$ and

$$\overline{T}_k = \begin{pmatrix} T_k \\ \delta_k u_k^T \end{pmatrix}.$$

Thus $V_k^T A V_k = T_k$.

Proposition 3.3 *Now, let us assume that A is nonsymmetric. Let $v_1 = r_0/\|r_0\|_2$ and $w_1 = \mu \tilde{r}_0$ such that $v_1^T w_1 = 1$. The nonsymmetric Lanczos process or bi-Lanczos*

process builds two basis $V_k = (v_1, \ldots, v_k)$ *and* $W_k = (w_1, \ldots, w_k)$ *of the Krylov subspaces* $\mathcal{K}_k(A, r_0)$ *and* $\mathcal{K}_k(A^T, \tilde{r}_0)$ *which satisfy*

$$\begin{aligned} AV_k &= V_k T_k + \delta_k v_{k+1} u_k^T, \\ A^T W_k &= W_k T_k^T + \gamma_k w_{k+1} u_k^T, \\ V_k^T W_k &= I, \end{aligned} \tag{9}$$

where $T_k \in \mathbb{R}^{k \times k}$ *is a tridiagonal matrix and* $u_k^T = (0 \ldots 0\ 1) \in \mathbb{R}^k$.

Proposition 3.4 *Let us still assume that* A *is nonsymmetric. Arnoldi process builds the orthonormal* V_k *which satisfies*

$$AV_k = V_k H_k + h_{k+1,k} v_{k+1} u_k^T = V_{k+1} \overline{H}_k \tag{10}$$

where $H_k \in \mathbb{R}^{k \times k}$ *is a Hessenberg matrix,* $\overline{H}_k \in \mathbb{R}^{k+1 \times k}$ *is defined by*

$$\overline{H}_k = \begin{pmatrix} H_k \\ h_{k+1,k} u_k^T \end{pmatrix}$$

and $u_k^T = (0 \ldots 0\ 1) \in \mathbb{R}^k$. *Thus* $V_k^T A V_k = H_k$.

3.3 Descent directions

Theorem 3.1 gives a first way of computing the iterates, provided that it is possible to evaluate Be_0, since e_0 is unknown. Each iteration of the method can solve the projected system (6), which depends on the choice of B. Another possibility is to use descent directions.

Theorem 3.3 *Let us assume that* $r_0^T B r_0 \neq 0$ *and let* $p_0 = r_0$. *Let us also assume that the method does not fail until iteration* $k+1$, *and that* x_k *is not the solution (*$r_k \neq 0$*).*

The method stagnates if and only if $r_k^T B e_k = 0$.

If the method does not stagnate, then the method is equivalent to

$$x_{k+1} = x_k + \alpha_k p_k,$$

where $p_k \in \mathcal{K}_{k+1}(A, r_0)$ *is given by*

$$p_k = r_k + \sum_{j=0}^{k-1} \beta_j p_j,$$

and satisfies $p_k^T B p_k \neq 0$ *with* $p_j^T B p_k = 0, j \leq k-1$; *the scalar* $\alpha_k \in \mathbb{R}^*$ *is defined by* $\alpha_k = \frac{r_k^T B e_k}{p_k^T B p_k}$.

Proof. If $x_{k+1} = x_k$, then $e_{k+1} = e_k$ and $r_k^T B e_k = 0$ by applying (5) at iteration $k+1$ with $u = r_k$. Converserly, if $r_k^T B e_k = 0$, then e_k satisfies (5) at iteration $k+1$ and, by unicity of the solution, $x_{k+1} = x_k$.

If the method does not stagnate, then the subspace condition can be written $x_{k+1} = x_k + \alpha_k p_k$, with $\alpha_k \neq 0$ and $p_k \in \mathcal{K}_{k+1}(A, r_0)$.

The proof of orthogonality is by recurrence. By assumption, the system $\{p_0, \ldots, p_{k-1}\}$ is a basis of $\mathcal{K}_k(A, r_0)$; thus the Galerkin condition (5) at iteration $k+1$ is equivalent to $p_j^T B e_{k+1} = 0,\ j \leq k$, thus to $p_j^T B e_k - \alpha_k p_j^T B p_k = 0$. By assumption, for $j \leq k-1, p_j^T B e_k = 0$, yielding $p_j^T B p_k = 0$, since $\alpha_k \neq 0$. Now, for $j = k$, since there is a unique solution, $p_k^T B p_k \neq 0$ and $\alpha_k = \frac{p_k^T B e_k}{p_k^T B p_k}$.

Since $\{p_0, \ldots, p_{k-1}, r_k\}$ is a basis of $\mathcal{K}_{k+1}(A, r_0)$, we can write $p_k = \beta_k r_k + \sum_{j=0}^{k-1} \beta_j p_j$; thus $p_k^T B e_k = \beta_k r_k^T B e_k$ and $\beta_k \neq 0$; we can choose $\beta_k = 1$, yielding $p_k^T B e_k = r_k^T B e_k$ and the result for α_k. ◇

For some particular methods, it is possible to compute the descent directions with a short recurrence. In [24, 25], these methods are fully characterized. Here, we give a sufficient condition.

Proposition 3.5 *If B and BA are symmetric, then it is possible to compute the descent directions with a short recurrence. If the method does not fail, does not finish, does not stagnate, then p_{k+1} is given by*

$$p_{k+1} = r_{k+1} + \beta_k p_k,$$

with $\beta_k = -\frac{r_{k+1}^T B p_k}{p_k^T B p_k}$

Proof. We first remark that, since B is symmetric, $p_i^T B p_j = 0, i \neq j$. The proof is by recurrence. For $k = 0$, $p_0 = r_0$. Now, assume that $p_0, \ldots, p_k$ are built. The next vector p_{k+1} can be written $p_{k+1} = r_{k+1} + \sum_{j=0}^{k} \beta_j p_j$.

Since B is symmetric, $(Bp_{k+1})^T p_i = p_{k+1}^T (Bp_i), i \leq k$. Hence $r_{k+1}^T B p_i + \beta_i p_i^T B p_i = 0$; for $i \leq k-1$, we have $r_{k+1}^T B p_i = (Ae_{k+1})^T B p_i = (Be_{k+1})^T A p_i$, by symmetry of B and BA, thus $r_{k+1}^T B p_i = 0$ by the Galerkin condition. Finally, $\beta_i = 0, i \leq k-1$. Taking then $i = k$ yields the result wanted. ◇

4 Case of SPD matrices

When the matrix A is symmetric positive definite, the unique Krylov method is the Conjugate Gradient method, noted CG. Practically, it is preconditioned and is noted PCG. There are many ways to describe PCG. Here, we use the framework of Krylov projection methods.

4.1 Conjugate Gradient method

Definition 4.1 *The Conjugate Gradient method is the Krylov projection method defined by $B = A$, when A is SPD.*

The general properties of Krylov methods can be derived for CG (or PCG).

Proposition 4.1 *The method CG does not fail, the subspace condition is equivalent to*

$$x_{k+1} = x_k + \alpha_k p_k, \tag{11}$$

where $\alpha_k \in \mathbb{R}$, $p_k \in \mathcal{K}_{k+1}(A, r_0)$ is a descent direction. The Galerkin condition is equivalent to

$$\|e_k\|_A = \|r_k\|_{A^{-1}} = \min_{y \in \mathcal{K}_k} \|e_0 - y\|_A$$

and to

$$r_k \perp \mathcal{K}_k(A, r_0). \tag{12}$$

As long as the residuals are nonzero, the system $\{r_0, r_1, \ldots, r_k\}$ is an orthogonal basis of the subspace $\mathcal{K}_{k+1}(A, r_0)$ and the system $\{p_0, p_1, \ldots, p_k\}$ is a $A-$orthogonal basis of this subspace. The descent directions can be defined by the short recurrence

$$p_0 = r_0,\ p_{k+1} = r_{k+1} + \beta_k p_k. \tag{13}$$

The coefficients β_k and α_k exist and are unique as long as $r_k \neq 0$; they are defined by $\alpha_k = \frac{r_k^T r_k}{p_k^T A p_k}$ and $\beta_k = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}$. Convergence is strictly monotoneous.

Proof. It is simply an application of previous theorems with $B = A$. The expression of β_k is however different and must be proved.

Descent directions are conjugate and in particular, $p_{k+1}^T A p_k = 0$. Then

$$r_{k+1}^T A p_k = \frac{1}{\alpha_k} r_{k+1}^T (r_k - r_{k+1}) = -\frac{1}{\alpha_k} r_{k+1}^T r_{k+1}$$

and

$$\beta_k = \frac{r_{k+1}^T r_{k+1}}{\alpha_k p_k^T A p_k} = \frac{r_{k+1}^T r_{k+1}}{r_k^T r_k}.$$

$\diamond$

CG algorithm is described in Table 4.1.

4.2 Link with Lanczos method

The previous description of CG uses the descent directions. Another way is to use the projected linear system. We first prove that CG builds the Lanczos orthonormal basis.

Table 1: CG algorithm

ALGORITHM 1: CG

* Initialization ;
choose x_0 ;
$r_0 = b - Ax_0$;
$p_0 = r_0$;
$\rho_0 = \|r_0\|^2$;
* Iterations ;
for $k = 0, 1, \ldots$ **until** convergence **do**
$q_k = Ap_k$;
$\alpha_k = \frac{rho_k}{p_k^T q_k}$;
$x_{k+1} = x_k + \alpha_k p_k$;
$r_{k+1} = r_k - \alpha_k q_k$;
$\rho_{k+1} = \|r_{k+1}\|_2^2$;
$\beta_k = \frac{\rho_{k+1}}{\rho_k}$;
$p_{k+1} = r_{k+1} + \beta_k p_k$;
end do

Proposition 4.2 *Let $v_{j+1} = \frac{r_j}{\|r_j\|}$, $j = 0, 1, \ldots$ and $V_k = (v_1, \ldots, v_k)$. The system V_k is an orthonormal basis of the Krylov subspace $\mathcal{K}_k(A, r_0)$ which verifies the Lanczos relation*

$$AV_k = V_k T_k + \delta_k v_{k+1} u_k^T \tag{14}$$

where $T_k \in \mathbb{R}^{k \times k}$ is the tridiagonal matrix given by

$$T_k = \begin{pmatrix} \gamma_1 & \delta_1 & & \\ \delta_1 & \gamma_2 & \delta_2 & \\ & . & . & . \\ & & \delta_{k-1} & \gamma_k \end{pmatrix},$$

and $u_k^T = (0 \ldots 0\ 1) \in \mathbb{R}^k$; the scalars are defined by

$$\delta_k = -\frac{\sqrt{\beta_{k-1}}}{\alpha_{k-1}}, \quad \gamma_{k+1} = \frac{1}{\alpha_k} + \frac{\beta_{k-1}}{\alpha_{k-1}}.$$

The matrix T_k is SPD.

Proof. Vectors v_k are orthonormal and span the Krylov subspace. Recurrence relations yield

$$\begin{aligned} &r_k = p_k - \beta_{k-1} p_{k-1} \text{ and } Ap_k = \tfrac{1}{\alpha_k}(r_k - r_{k+1}), \\ &Ar_k = \tfrac{1}{\alpha_k}(r_k - r_{k+1}) - \tfrac{\beta_{k-1}}{\alpha_{k-1}}(r_{k-1} - r_k), \\ &Ar_k = -\tfrac{\beta_{k-1}}{\alpha_{k-1}} r_{k-1} + (\tfrac{1}{\alpha_k} + \tfrac{\beta_{k-1}}{\alpha_{k-1}}) r_k - \tfrac{1}{\alpha_k} r_{k+1}, \end{aligned}$$

thus we get relation (14).

Since $y^T T_k y = (V_k y)^T A (V_k y)$, the matrix T_k is SPD. $\diamond$

Thus, the CG method solves a tridiagonal system at each iteration.

Corollary 4.1 *Each iteration of the GC method is equivalent to compute* $x_k = x_0 + V_k y$ *where* $y \in \mathbb{R}^k$ *is solution of*

$$T_k y = \|r_0\|_2 u_1, \tag{15}$$

where $u_1^T = (1\ 0 \ldots 0) \in \mathbb{R}^k$.

Proof. Using theorem 3.1, CG is equivalent to $x_k = x_0 + V_k y$ with $(V_k^T A V_k) y = B e_0$; using proposition 4.2, $V_k^T A V_k = T_k$; furthermore, $B e_0 = \|r_0\|_2 u_1$. $\diamond$

4.3 Convergence of CG

We know that convergence is strictly monotoneous. Here, we give explicit error bounds, in two results; we improve the first result by using polynomial properties. It should be noted that the A norm of the error is decreasing, thus the A^{-1} norm of the residual is decreasing (but not necessarily the euclidian norm).

Let $\sigma(A)$ the set of eigenvalues of A, with $0 < \lambda_1 \leq \ldots \leq \lambda_n$. Let

$$\kappa(A) = \|A\|_2 \|A^{-1}\|_2 = \lambda_n / \lambda_1$$

the spectral condition number of A.

Theorem 4.1 *The CG method has a strictly monotoneous convergence; more precisely*

$$\|e_{k+1}\|_A \leq (1 - \frac{1}{\kappa(A)})^{1/2} \|e_k\|_A. \tag{16}$$

Proof.

$r_{k+1} = r_k - \alpha_k A p_k$,
$\|e_{k+1}\|_A^2 = e_{k+1}^T A e_{k+1} = r_{k+1}^T A^{-1} r_{k+1} = r_k^T A^{-1} r_k - 2\alpha_k r_k^T p_k + \alpha_k^2 p_k^T A p_k$;
but $p_k^T A p_k = \frac{r_k^T r_k}{\alpha_k}$ and $r_k^T p_k = r_k^T r_k$, hence $\|e_{k+1}\|_A^2 = \|e_k\|_A^2 - \alpha_k \|r_k\|_2^2$.

We now give bounds for $\|r_k\|_2^2$ and α_k. We have

$\|e_k\|_A^2 = r_k^T A^{-1} r_k \leq \|A^{-1}\|_2 \|r_k\|_2^2$.
Also $p_k^T A p_k = (r_k + \beta_{k-1} p_{k-1})^T A p_k = r_k^T A p_k = r_k^T A r_k + \beta_{k-1} r_k^T A p_{k-1}$;
but $r_k^T A p_{k-1} = (p_k - \beta_{k-1} p_{k-1})^T A p_{k-1} = -\beta_{k-1} p_{k-1}^T A p_{k-1} \leq 0$,
thus $p_k^T A p_k \leq r_k^T A r_k \leq \|A\|_2 \|r_k\|_2^2$ and $\alpha_k = \frac{\|r_k\|_2^2}{p_k^T A p_k} \geq \frac{1}{\|A\|_2}$.

We deduce that

$$\alpha_k \|r_k\|_2^2 \geq \frac{1}{\|A\|_2 \|A^{-1}\|_2} \|e_k\|_A^2 \text{ and } \|e_{k+1}\|_A^2 \leq (1 - \frac{1}{\kappa(A)}) \|e_k\|_A^2.$$

$\diamond$

The error bound depends on the condition number; if it is large, then the coefficient is close to 1 and convergence may be slow. Now, we improve the error bound. We first use the minimum property and the polynomial behavior.

Proposition 4.3 *CG iterations satisfy*

$$\|e_k\|_A = \min_{q \in \mathcal{P}_k^0} \|q(A)e_0\|_A.$$

Proof. The set $\{\|q(A)e_0\|_A,\ q \in \mathcal{P}_k^0\}$ is nothing else than the set $\{\|b - Ay\|_{A^{-1}},\ y \in x_0 + \mathcal{K}_k(A, r_0)\}$. The minimization property is then equivalent when using Krylov subpsaces. ◇

This property can be translated in a "min-max" formulation, by using the eigenvalue decomposition.

Corollary 4.2 *CG iterations satisfy*

$$\|e_k\|_A \leq \|e_0\|_A \min_{q \in \mathcal{P}_k^0} \max_{z \in \sigma(A)} |q(z)|, \tag{17}$$

Proof. The proof follows the lines of theorem 2.1. Let $A = U\Sigma U^{-1}$ the spectral decomposition of A, where Σ is the diagonal matrix $diag(\lambda_1, \ldots, \lambda_n)$, and U is the orthogonal matrix of eigenvectors. Then $\kappa(U) = 1$ and $\|q(A)e_0\|_A \leq \max_i |q(\lambda_i)| \|e_0\|_A$. By using proposition 4.3, we get (17). ◇

The "min-max" property yields an error bound by using the approximation theory with polynomials.

Corollary 4.3 *CG iterations satisfy*

$$\|e_k\|_A \leq 2\|e_0\|_A (\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1})^k. \tag{18}$$

Proof. The result is based on the following property, see for example [46]:

$$\min_{q \in \mathcal{P}_k^0} \max_{z \in [\lambda_1, \lambda_n]} |q(z)| = \frac{1}{|C_k(\frac{\lambda_n + \lambda_1}{\lambda_n - \lambda_1})|},$$

where C_k is the Chebyshev polynomial of first kind of degree k. ◇

Remark 4.1 *The bound (18) is better than the bound (16). Indeed,*

$$\frac{\sqrt{\kappa(A)} - 1}{\sqrt{\kappa(A)} + 1} \leq \frac{\sqrt{\kappa(A) - 1}}{\sqrt{\kappa(A)}} = (1 - 1/\kappa(A))^{1/2}$$

since $\sqrt{\kappa(A)} - 1 \leq \sqrt{\kappa(A) - 1}$.

Figure 1: Convergence of CG related to condition number.

The effect of the condition number is illustrated in Figure 1. Here, A is a diagonal matrix of size $n = 1000$ with eigenvalues ranging regularly from 1 up to 10^k, such that $\kappa(A) = 10^k$; clearly, the number of iterations increases with k. The figure plots the euclidian norm of the residual (which does not decrease at each iteration, recall that (the A^{-1} norm of the residual decreases).

In the Lanczos method, the Ritz values, which are the eigenvalues of the matrix T_k, converge towards the eigenvalues of A. In the CG method, when an eigenvalue has converged, iterations continue as if there had been a deflation in the spectrum of A and convergence becomes faster: this is called superlinear convergence. It is possible to observe several accelerations, each time a new Ritz value has converged. This superlinear convergence is studied for example in [55, 51, 5]. It is observed in Figure 1 and illustrated in Figure 2. Here, A is a diagonal matrix with k small eigenvalues regularly spaced from 0.01 and $n - k$ eigenvalues ranging from 1 to $n - k$. When k increases, the residual stagnates at the same level (with oscillations) during more and more iterations before reaching the same convergence rate as for $k = 0$.

This superlinear convergence is the basis of augmented and deflation methods used to accelerate convergence [23, 49, 40, 41]. Indeed, the idea is to use the Ritz values and to remove the smallest eigenvalues which slow down convergence. Augmented CG has been successfully applied to various engineering problems, see for example [11].

4.4 Preconditioned Conjugate Gradient

Let A a SPD matrix and C a nonsingular matrix. The preconditioned systems defined in 2.3 are not symmetric. Now, let C a SPD matrix. There are two ways of defining the

Figure 2: An example of superlinear convergence of CG.

PCG method with C. The first approach consists in using the Cholesky factorization (which exists) $C = LL^T$. Then the preconditioned system

$$L^T AL(L^{-1}x) = L^T b,$$

can be written

$$By = c,$$

with $B = L^T AL$ and $y = L^{-1}x,\ c = L^T b$. The matrix B is SPD thus it is possible to apply CG to the linear system $By = c$; this is called PCG.

By using a change of variable, it is easy to write PCG as in Table 4.4.

Now, since the matrix C is SPD, it defines the scalar product

$$< x, y >_C = x^T Cy. \tag{19}$$

Since the matrix AC is self-adjoint for this scalar product, it is possible to apply CG with the scalar product (19) to the preconditioned system $AC(C^{-1}x) = b$. It is easy to notice that this approach amounts to the same algorithm as previously.

More general assumptions about the preconditioning matrix C are studied in [2, 12].

Since each iteration requires more operations, the number of iterations must be reduced in order to reduce the total CPU time. Let N the cost of the matrix-vector product and M the cost of the product by C (which means in general solving a linear system). If the matrix A is sparse and stored with a compressed sparse row or column scheme, then $N = 2nz(A) + O(n)$, with $nz(A)$ the number of nonzero terms. Other operations of one iteration are vector operations, of type BLAS1. Thus the cost of one

Table 2: PCG algorithm preconditioned by C

ALGORITHM 2: PCG

* Initialization ;
choose x_0 ;
$r_0 = b - Ax_0$;
$z_0 = Cr_0$;
$p_0 = z_0$;
$\rho_0 = r_0^T z_0$;
* Iterations ;
for $k = 0, 1, \ldots$ **until** convergence **do**
  $q_k = Ap_k$;
  $\alpha_k = \frac{\rho_k}{p_k^T q_k}$;
  $x_{k+1} = x_k + \alpha_k p_k$;
  $r_{k+1} = r_k - \alpha_k q_k$;
  $z_{k+1} = Cr_{k+1}$;
  $\rho_{k+1} = r_{k+1}^T z_{k+1}$;
  $\beta_k = \frac{\rho_{k+1}}{\rho_k}$;
  $p_{k+1} = z_{k+1} + \beta_k p_k$;
end do

iteration in CG is $N + O(n)$ and it is $N + M + O(n)$ in PCG. In general, the term $O(n)$ can be neglected. If M is almost equal to N, then the number of iterations must be divided by two to get about the same amount of operations.

5 Symmetric indefinite case

When the matrix A is symmetric but indefinite, there are several methods. Two choices have been explored: $B = A^2$ and $B = A$. In both cases, Lanczos process is used to build the orthonormal basis and there is a short recurrence. In the first case, the method does not fail, the Galerkin condition is a minimization condition and the convergence is monotoneous, whereas in the second case, the method can fail. Several variants can be found in the literature, for example in [4]. Here we describe the MINRES and SYMMLQ methods [42], and the variant CR [27].

5.1 MINRES method

Here, $B = A^2$. Since $e_k^T A^2 y = r_k^T Ay$, the Galerkin condition can be written, according to 3.1,

$$\min \|r_k\|_2.$$

The subspace condition can be written $x_k = x_0 + V_k y$ thus

$$r_k = r_0 - AV_k y = \|r_0\|_2 v_1 - V_{k+1}\overline{T}_k y = V_{k+1}(\|r_0\|_2 u_1 - \overline{T}_k y)$$

and the Galerkin condition becomes

$$\min_{y\in\mathbb{R}^k} \|\|r_0\|_2 u_1 - \overline{T}_k y\| \tag{20}$$

The MINRES method solves (20) by factorizing $\overline{T}_k$ with Givens rotations. This factorization can be done with short recurrences, because T_k is symmetric. The method is generalized by GMRES for the nonsymmetric case, as described in 6.2.

5.2 Conjugate Residual method CR

Here too, $B = A^2$, so that the series of iterates is the same. However, the CR method uses descent directions. Galerkin condition can be written $e_{k+1}^T A^T A r_j = 0,\ j \leq k$ or

$$r_{k+1}^T A r_j = 0,\ j \leq k, \tag{21}$$

In other words, residuals are $A-$conjugate. The descent directions p_k are A^2 orthogonal thus the vectors Ap_k are orthogonal so that the Lanczos process is used to build the orthonormal basis $(Ap_0/\|Ap_0\|_2, \ldots, Ap_{k-1}/\|Ap_{k-1}\|_2)$ of the Krylov subspace $A\mathcal{K}_k(A, r_0)$. Short recurrences are used to satisfy orthogonality conditions $r_{k+1}^T A r_k = 0$ and $(Ap_{k+1})^T Ap_k = 0$.

5.3 SYMMLQ method

Here, $B = A$ as for CG method. The Lanczos process builds the matrix T_k and the basis V_k, so that the Galerkin condition is the same as for CG:

$$x_k = x_0 + V_k y,\ T_k y = \|r_0\|_2 u_1. \tag{22}$$

The SYMMLQ method solves the linear system (22) by using a LQ factorization of T_k, in order to deal with breakdowns. Indeed, since A is indefinite, the method can fail when the matrix T_k is singular: it is a serious breakdown. If it does not fail, the Lanczos method stops when the method has converged: it is a happy breakdown (theorem 3.1). Look-ahead methods are designed to avoid breakdowns [10].

6 Nonsymmetric case

The ideal case is, like in CG, to design a method with both a minimization property and a short recurrence, and a strict decreasing series of residuals. A first approach is to define a preconditioned SPD system, as in methods CGNR and CGNE. However,

convergence can be slow because the condition number is squared. Thus a second approach is to get a minimization property without short recurrences, as in GMRES method, which generalizes MINRES and uses Arnoldi process instead of Lanczos process. The main drawback is an increasing complexity, which is avoided by restarting, at the cost of a possible stagnation. And a third approach is to get short recurrences without a minimization property, as in BiCG method, which uses Bi-Lanczos process. The main drawback is the non monotoneous convergence. Some variants close to BiCG, such as CGS, BiCGStab and QMR are no longer Krylov projection methods. These methods are explained for example in [2, 46, 4, 12, 36, 32]. We describe them briefly in what follows.

6.1 Methods CGNR and CGNE

When A is nonsymmetric, the system (1) can be preconditioned to get a SPD system. This is the idea used with normal equations.

By left preconditioning with A^T, the system becomes

$$A^T A x = A^T b, \tag{23}$$

on which the CG method can be applied. The iterates are defined by

$$x_k \in x_0 + \mathcal{K}_k(A^T A, A^T r_0)$$

with

$$\|r_k\|_2 = \min_{y \in x_0 + \mathcal{K}_k(A^T A, A^T r_0)} \|b - Ay\|_2.$$

This method is called CGNR and minimizes the norm of the residual. A variant of CGNR, also called LSQR, is very often used to solve least-squares problems [43].

By right preconditioning with A^T, the system becomes

$$AA^T(A^{-T}x) = b, \tag{24}$$

and again the CG method can be applied. The iterates are defined by

$$x_k \in x_0 + A^T \mathcal{K}_k(AA^T, r_0)$$

with

$$\|e_k\|_2 = \min_{y \in x_0 + A^T \mathcal{K}_k(AA^T, r_0)} \|x^* - y\|_2.$$

This method is called CGNE and minimizes the norm of the error.

Both methods CGNR and CGNE have the properties of CG: short recurrence, minimization, strictly monotoneous convergence. However, in the error bound (18), the condition number is given by

$$\kappa(A^T A) = \kappa(AA^T) = \kappa(A)^2 = (\frac{\sigma_n}{\sigma_1})^2,$$

where $0 < \sigma_1 < \ldots < \sigma_n$ are the singular values of A. Thus convergence can be slow for some matrices.

6.2 GMRES method

In the GMRES method [47], the matrix B is chosen as $B = A^T A$, thus is SPD.

Proposition 6.1 *Method GMRES is defined by*

$$\begin{aligned} &x_k \in x_0 + \mathcal{K}_k(A, r_0), \\ &r_k \perp A\mathcal{K}_k(A, r_0), \end{aligned}$$

or equivalently

$$\|r_k\|_2 = \min_{x \in x_0 + \mathcal{K}_k(A, r_0)} \|b - Ax\|_2. \tag{25}$$

The GMRES algorithm can also be viewed as the linear system (6). Indeed, the GMRES method uses the Arnoldi process.

Theorem 6.1 *If the method has not finished, problem (25) is equivalent to*

$$\min_{y \in \mathbb{R}^k} (\|r_0\|_2 u_1 - \overline{H}_k y) \tag{26}$$

where V_{k+1} and $\overline{H}_k$ are the basis and matrix built by Arnoldi process applied to $v_1 = \frac{r_0}{\|r_0\|_2}$.

GMRES method does not fail, convergence is monotoneous.

Proof. System (6) is equivalent to problem (25). The matrix $C_k = V_k^T B V_k$ is equal to $(AV_k)^T (AV_k) = \overline{H}_k^T \overline{H}_k$ since V_{k+1} is orthonormal.

The right-hand side of system (6) is equal to $V_k^T B e_0 = \overline{H}_k^T V_{k+1}^T r_0 = \|r_0\|_2 \overline{H}_k^T u_1$ since v_1 is orthogonal to $v_i, i \geq 2$.

Thus, solving system (6) is equivalent to solving system

$$\overline{H}_k^T \overline{H}_k y = \|r_0\|_2 \overline{H}_k^T u_1,$$

which is the normal equation underlying the least-squares problem (26).

Using theorem 3.2, since B is definite, GMRES does not fail and convergence is monotoneous. ◇

Convergence can also be analyzed by using polynomials and the minimization property.

Theorem 6.2 *If A is diagonalisable, let $A = U\Sigma U^{-1}$, where the columns of U are a basis of eigenvectors and where $\Sigma = diag(\lambda_1, \ldots, \lambda_n)$; let $\kappa(U) = \|U\|_2 \|U^{-1}\|_2$ the condition number of U.*

GMRES iterations satisfy

$$\|r_k\|_2 \leq \|r_0\|_2 \, \kappa(U) \, \min_{q \in \mathcal{P}_k^0} \max_{z \in \sigma(A)} |q(z)|. \tag{27}$$

Proof. Minimization property is equivalent to

$$\|r_k\|_2 = \min_{q \in \mathcal{P}_k^0} \|q(A)r_0\|_2.$$

By applying theorem 2.1, we get inequality (27). ◇

However, since the matrix $BA^{-1} = A^T$ is not definite in general, convergence is not always strictly monotoneous and stagnation can occur.

Proposition 6.2 *If* $\|r_{k+1}\|_2 = \|r_k\|_2$ *then* $r_k^* A r_k = 0$.

Converserly, if $r_k^* A r_k = 0$*, then* $r_{k+1} = r_k$ *or* $r_k = r_{k-1}$.

Proof. If $\|r_{k+1}\|_2 = \|r_k\|_2$, then r_k is the unique solution of problem (25) for index $k+1$ thus $r_k = r_{k+1}$ and $r_k \perp A\mathcal{K}_{k+1}(A, r_0)$; but $r_k \in \mathcal{K}_{k+1}(A, r_0)$ thus $r_k \perp Ar_k$.

Converserly, if $r_k = 0$, then $r_{k+1} = r_k = 0$. One can assume that $r_k \neq 0$. Then the Krylov subspace $\mathcal{K}_{k+1}(A, r_0)$ is of dimension $k+1$.

We assume that $r_k^* A r_k = 0$.

Since $r_k \in \mathcal{K}_{k+1}(A, r_0)$, there are two possible cases.

If $r_k \notin \mathcal{K}_k(A, r_0)$ then $eng\{\mathcal{K}_k(A, r_0), r_k\} = \mathcal{K}_{k+1}(A, r_0)$. Since $r_k \perp A\mathcal{K}_k$ and $r_k \perp Ar_k$, it follows that $r_k \perp A\mathcal{K}_{k+1}$ and r_k is solution of problem (25) for index $k+1$ and by unicity, $r_{k+1} = r_k$.

Else, $r_k \in \mathcal{K}_k$ thus $x_k - x_0 \in \mathcal{K}_k, A(x_k - x_0) \in \mathcal{K}_k$ and, thanks to proposition (2.3), $x_k - x_0 \in \mathcal{K}_{k-1}$. Hence, r_k is solution of problem (25) at index $k-1$ and by unicity, $r_k = r_{k-1}$. ◇

Since $BA = A^T A^2$ is nonsymmetric, it is not possible to apply proposition 3.5 to define a short recurrence. Arnoldi process requires storing the vectors v_k; the number of operations, apart from the matrix-vector product, has a complexity in $O(nk^2)$. In order to reduce the storage and the CPU time, restarted GMRES is used.

The GMRES(m) method computes cycles of m iterations of GMRES, and restarts with the last approximation x_m. This confines memory and CPU requirements but the choice of m is not easy, because convergence can stagnate if m is too small. In a cycle of m iterations of GMRES(m), the number of operations is $2nm^2 + mN + O(nm)$, where N is the cost of the matrix-vector product; also, the $m+1$ vectors v_j must be stored.

The GMRES(m) algorithm is described in Table 6.2. The least-squares problem (26) is solved in general by using a QR factorization of the matrix $\overline{H}_k$ with Givens rotations. Then it is possible to compute $\|r_k\|_2$ without computing x_k.

Examples of convergence behavior are illustrated in Figure 3 and Figure 4. In the first example, the matrix $A = VDV^{-1}$ of order $n = 1000$ is diagonalized, with $\kappa(A) = k$ and $\kappa(V) = 19$. As for CG, the impact of $\kappa(A)$ can be observed. In the second example, $\kappa(A) = 1000$ and $\kappa(V)$ is variable. Here, the impact of $\kappa(V)$ can be

Table 3: GMRES(m) algorithm.

ALGORITHM 3: GMRES(m)

* Initialization ;
choose x_0 ;
$r_0 = b - Ax_0$;
* Iterations ;
until convergence **do**
  $v_1 = \frac{r_0}{\|r_0\|_2}$;
  * Arnoldi process ;
  for $j = 1, m$
    $w = Av_j$;
    for $i = 1, j$
      $h_{ij} = v_i^T w$;
      $w = w - h_{ij} v_i$;
    end for ;
    $h_{j+1,j} = \|w\|_2$;
    $v_{j+1} = w / h_{j+1,j}$;
    * least-squares problem
    $\overline{H}_j = Q_j R_j$;
    compute $\|r_j\|_2$;
    convergence test
  end for ;
  compute y_m solution of $\min_y \left(\|r_0\|_2 e_1 - \overline{H}_m y\right)$;
  $x_m = x_0 + V_m y_m$;
  $r_m = b - Ax_m$;
  convergence test
  $x_0 = x_m$; $r_0 = r_m$;
end do

Figure 3: Effect of eigenvalues on convergence of GMRES.

Figure 4: Effect of eigenvectors on convergence of GMRES.

observed, as predicted. In both cases, superlinear convergence is achieved after some iterations. Here, GMRES is not restarted.

As in the previous examples, superlinear convergence can be observed for some matrices; moreover, stagnation can occur when restarting; augmented and deflation methods have been designed in order to overcome this drawback and to accelerate convergence [21, 13, 38, 39, 19].

6.3 BICG methods

The BICG method solves the augmented system

$$\begin{pmatrix} A & 0 \\ 0 & A^T \end{pmatrix} \begin{pmatrix} x \\ \tilde{x} \end{pmatrix} = \begin{pmatrix} b \\ \tilde{b} \end{pmatrix}$$

by the Krylov projection method, where the matrix B is defined by

$$B = \begin{pmatrix} 0 & A^T \\ A & 0 \end{pmatrix}$$

Proposition 6.3 *The BICG method is defined by the initial choice of x_0 and $\tilde{x}_0$ and by the subspace and Galerkin conditions:*

$$\begin{aligned} &x_k \in x_0 + \mathcal{K}_k(A, r_0), \\ &\tilde{x}_k \in \tilde{x}_0 + \mathcal{K}_k(A^T, \tilde{r}_0), \\ &r_k \perp \mathcal{K}_k(A^T, \tilde{r}_0), \\ &\tilde{r}_k \perp \mathcal{K}_k(A, r_0). \end{aligned}$$

Proof. Let

$$\tilde{A} = \begin{pmatrix} A & 0 \\ 0 & A^T \end{pmatrix}.$$

The Krylov subspace is then

$$\begin{pmatrix} \mathcal{K}_k(A, r_0) \\ \mathcal{K}_k(A^T, \tilde{r}_0) \end{pmatrix},$$

which yields the subspace condition. Moreover,

$$B\tilde{A}^{-1} = \begin{pmatrix} 0 & I \\ I & 0 \end{pmatrix},$$

which yields the Galerkin condition. ◇

Since the matrix B is not definite, the method may fail, by theorem 3.2; since the matrices B and $B\tilde{A}$ are symmetric, the method can be defined by a short recurrence, by theorem 3.5. Also, the method is related to bi-Lanczos.

Proposition 6.4 *The BICG method applies the Bi-Lanczos method to the initial vectors $v_1 = r_0/\|r_0\|_2$ and $w_1 = \mu\tilde{r}_0$ such that $v_1^T w_1 = 1$. The residuals and the descent directions satisfy the orthogonality conditions*

$$r_{k+1}^T \tilde{r}_k = 0, \; Ap_{k+1}^T \tilde{p}_k = p_{k+1}^T A^T \tilde{p}_k = 0.$$

The method solves the linear system

$$T_k y = \|r_0\|_2 u_1.$$

It fails if T_k is singular.

Table 4: BICG algorithm.

ALGORITHM 4: BICG

* Initialization ;
choose x_0 and $\tilde{b}$ and $\tilde{x}_0$;
$r_0 = b - Ax_0$; $\tilde{r}_0 = \tilde{b} - A\tilde{x}_0$;
$p_0 = r_0$; $\tilde{p}_0 = \tilde{r}_0$;
* Iterations ;
for $k = 0, 1, \ldots$ **until** convergence **do**
$\alpha_k = \frac{\tilde{r}_k^T r_k}{\tilde{p}_k^T A p_k}$;
$x_{k+1} = x_k + \alpha_k p_k$;
$r_{k+1} = r_k - \alpha_k A p_k$;
$\tilde{r}_{k+1} = \tilde{r}_k - \alpha_k A^T \tilde{p}_k$;
$\beta_{k+1} = \frac{\tilde{r}_{k+1}^T r_{k+1}}{\tilde{r}_k^T r_k}$;
$p_{k+1} = r_{k+1} + \beta_{k+1} p_k$;
$\tilde{p}_{k+1} = \tilde{r}_{k+1} + \beta_{k+1} \tilde{p}_k$;
end do

Proof.

The method builds the basis $V_k = (r_0, \ldots, r_{k-1})$ and $W_k = (\tilde{r}_0, \ldots, \tilde{r}_{k-1})$ such that $V_k^T W_k = I$. The method fails if

$$C_k = \begin{pmatrix} 0 & T_k^T \\ T_k & 0 \end{pmatrix},$$

is singular, thus if the matrix T_k is singular. ◇

The algorithm is given in Table 6.3.

When the method has finished, it is a happy breakdown. In order to avoid serious breakdowns, when the matrix is singular, a look-ahead version can be used [45, 29].

The algorithm involves not only a matrix-vector product with A, but also a matrix-vector product with A^T. It is possible to modify the polynomial underlying the method. This idea is used in CGS [52] and BICGSTAB methods [56]. They are no longer projection methods, but are still polynomial methods.

Since there is no minimization property, convergence is not monotoneous. The QMR method avoids irregular convergence. Again, this polynomial method is not a projection Krylov method. It is still based on Bi-Lanczos and replaces Galerkin condition by a quasi-minimization condition [30, 28]. More precisely,

$$x_k = x_0 + V_k y,$$
$$r_k = r_0 - AV_k y = V_{k+1}(\beta u_1 - \overline{T}_k y),$$

with y solution of

$$\min_y \|\Omega_{k+1}^{-1}(\gamma u_1 - \Omega_{k+1}\overline{T}_k y)\|_2 \tag{28}$$

where Ω_{k+1} is a diagonal matrix.

Again, a look-ahead version can be used to avoid breakdowns. A Transpose-Free variant (TFQMR) does not involve the product with A^T. Convergence is not monotoneous but is more regular than for BICG and a result of convergence is proved.

In Figures 5 and 6, the methods BiCGStab, QMR, Full-GMRES and GMRES(m) are compared. Matrices Sherman4 and Sherman5 come from the Harwell-Boeing collection [18]. Full GMRES converges faster than other methods (in terms of matrix-vector products, not necessarily CPU time). Restarting GMRES slows down convergence and can create stagnation. Eventually, GMRES(m) becomes slower than other methods when m becomes small. Convergence of QMR is more regular than convergence of BICGStab but not monotoneous, whereas the residuals of GMRES(m) decrease (or stagnate).

Figure 5: Convergence behavior of Krylov methods with matrix Sherman4.

7 Practical issues

7.1 Numerical problems

In practice, several numerical problems can arise when using Krylov methods. Krylov projection methods are based on orthogonality conditions, which are quite often verified by recurrence relations. Also, the residual r_k is computed by recurrence, not explicitely by $r_k = b - Ax_k$. Although these conditions are satisfied with exact

Figure 6: Convergence behavior of Krylov methods with matrix Sherman5.

arithmetic operations, is is no longer true when dealing with floating-point arithmetic and rounding errors. Three phenomena can occur, as discussed for example in [8, 9, 57, 53, 37, 44]:

- loss of orthogonality ; for example, descent directions are no longer orthogonal; this situation can slow down convergence. A partial or complete reorthogonalization can be done to remedy to this loss.

- there is a drift between the computed residual r_k and the exact residual $b - Ax_k$. Then the stopping criterium is no longer valid. The residual can be recomputed every so often to remedy to this drift.

- In Lanczos or bi-Lanczos methods, a breakdown can occur. Numerically, this problem can arise when the matrix is nearly singular, this is a near-breakdown. Thus look-ahead can be applied as soon as a numerical problem is detected.

7.2 Preconditioning

As already pointed out, preconditioning is essential in practice in order to accelerate convergence. Incomplete factorizations can be efficient in many cases but they suffer from two major drawbacks: the number of iterations still increases with the size of the system and parallelism is sometimes poor. Multilevel preconditioners are the most promising methods nowadays. Multigrid preconditioners can be very efficient for some classes of problems. Other multilevel approaches are in general based on domain decomposition methods, studied from a PDE point of view or from an algebraic point of view; several authors have designed methods based on Schur complement and

Schwarz methods. These methods can be combined with deflation or coarse grid correction. There is still a need of robust general preconditioners, which can guarantee convergence for most scientific applications, and this is an active area of research.

7.3 Parallelism

High performance computing is an essential key for dealing with huge linear systems; nowadays, sparse direct solvers are quite efficient and scalable. However, iterative methods are competitive, if they are combined with an efficient and scalable preconditioner. The two major operations are the sparse matrix-vector product and the preconditioning. Usually, the matrix is partitioned into rows or columns, in order to distribute the operations of the matrix-vector product. Domain decomposition approaches hold naturally parallel features. Moreover, hybrid methods, combining parallelism at various levels, are very promising. Parallel efficient preconditioning is the key for a widespread use of Krylov iterative methods.

Acknowledgements

The author would like to thank Bernard Philippe for helpful discussions about this work.

References

[1] P. R. Amestoy, I. S. Duff, and J.-Y. L'Excellent. Multifrontal parallel distributed symmetric ans unsymmetric solvers. *Computer Methods in Applied Mechanics and Engineering*, 138:501–520, 2000.

[2] S.F. Ashby, T.A. Manteuffel, and P.E. Saylor. A taxonomy for Conjugate Gradient Methods. *SIAM Journal on Numerical Analysis*, 26:1542–1568, 1990.

[3] Satish Balay, Kris Buschelman, William D. Gropp, Dinesh Kaushik, Matthew G. Knepley, Lois Curfman McInnes, Barry F. Smith, and Hong Zhang. PETSc Web page, 2009. http://www.mcs.anl.gov/petsc.

[4] R. Barret, M. Berry, T. Chan, J. Demmel, J. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. van der Vorst. *Templates for the solution of linear systems: building blocks for iterative methods - 2nd edition*. SIAM / netlib, Philadelphia, PA, 1994.

[5] B. Beckermann and A. B. J. Kuijlaars. Superlinear convergence of conjugate gradients. *SIAM Journal on Numerical Analysis*, 39(1):300–329, 2001.

[6] M. Benzi. Preconditioning techniques for large linear systems: a survey. *Journal of Computational Physics*, 182:418–477, 2002.

[7] C. Brezinski. Projection methods for linear systems. *Journal of Computational and Applied Mathematics*, 77:35–51, 1997.

[8] C. Brezinski, M. Redivo Zaglia, and H. Sadok. Avoiding breakdown and near-breakdown in lanczos type algorithms. *Numerical Algorithms*, 1:207–221, 1991.

[9] C. Brezinski, M. Redivo Zaglia, and H. Sadok. Addendum to: "avoiding breakdown and near-breakdown in lanczos type algorithms". *Numerical Algorithms*, 2:133–136, 1992.

[10] C. Brezinski, M. Redivo Zaglia, and H. Sadok. New look-ahead lanczos-type algorithms for linear systems. *Numerische Mathematik*, 83:53–85, 1999.

[11] M-O. Bristeau and J. Erhel. Augmented conjugate gradient. application in an iterative process for thesolution of scattering problems. *Numerical Algorithms*, 18:71–90, 1998.

[12] A.M. Bruaset. *A survey of preconditioned iterative methods*. Pitman Research Notes in Mathematics Series. Longman Scientific and Technical, 1995.

[13] K. Burrage and J. Erhel. On the performance of various adaptive preconditioned GMRES. *Numerical Linear Algebra with Applications*, 5:101–121, 1998.

[14] Y. Chen, T. A. Davis, W. W. Hager, and S. Rajamanickam. Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate. *ACM Transactions on Matematical Software*, 35(3):22:1–22:14, 2008.

[15] T. Davis. Algorithm 832: Umfpack, an unsymmetric-pattern multifrontal method. *ACM Transactions on Mathematical Software*, 30:196–199, 2004.

[16] T. A. Davis. *Direct Methods for Sparse Linear Systems*. SIAM Book Series on the Fundamentals of Algorithms. SIAM, Philadelphia, 2006.

[17] J. W. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li, and J. W. H. Liu. A supernodal approach to sparse partial pivoting. *SIAM J. Matrix Analysis and Applications*, 20:720–755, 1999.

[18] I. Duff, R. Grimes, and J. Lewis. Sparse matrix test problems. *ACM transactions on mathematical software*, pages 1–14, 1989.

[19] M. Eiermann, O. G. Ernst, and O. Schneider. Analysis of acceleration strategies for restarted minimal residual methods. *Journal of Computational and Applied Mathematics*, 123(1-2):261–292, 2000.

[20] J. Erhel. *Computational Technology Reviews*, volume 3, chapter Some Properties of Krylov Projection Methods for Large Linear Systems, pages 41–70. Saxe-Coburg Publications, 2011.

[21] J. Erhel, K. Burrage, and B. Pohl. Restarted GMRES preconditioned by deflation. *Journal of Computation and Applied Mathematics*, 69:303–318, 1996.

[22] J. Erhel, J.-R. de Dreuzy, A. Beaudoin, E. Bresciani, and D. Tromeur-Dervout. A parallel scientific software for heterogeneous hydrogeology. In Ismail H. Tuncer, Ulgen Gulcat, David R. Emerson, and Kenichi Matsuno, editors, *Parallel Computational Fluid Dynamics 2007*, volume 67 of *Lecture Notes in Computational Science and Engineering*, pages 39–48. Springer, 2009. invited plenary talk.

[23] J. Erhel and F. Guyomarc'h. An augmented conjugate gradient method for solving consecutive symmetric positive definite systems. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1279–1299, 2000.

[24] V. Faber and T. Manteuffel. Necessary and sufficient conditions for the existence of a conjugate gradient methods. *SIAM Journal on numerical analysis*, 21:352–362, 1984.

[25] V. Faber and T. Manteuffel. Orthogonal error methods. *SIAM journal on numerical analysis*, 24(1):170–187, 1987.

[26] R. D. Falgout, J. E. Jones, and U. Meier Yang. Pursuing scalability for Hypre's conceptual interfaces. *ACM Transactions on Mathematical Software*, 31(3):326–350, 2005.

[27] R. Fletcher. Conjugate gradient methods for indefinite systems. In G. Watson, editor, *Numerical Analysis*, volume 506 of *Lecture Notes in Mathematics*, pages 73–89. Springer Berlin / Heidelberg, 1976.

[28] R. Freund. A transpose-free quasi-minimal residual algorithm for non-hermitian linear systems. *SIAM journal on scientific computing*, 14:470–482, 1993.

[29] R. Freund, M. Gutknecht, and N. Nachtigal. An implementation of the look-ahead Lanczos algorithm for non-hermitian matrices. *SIAM journal on scientific computing*, 14:137–158, 1993.

[30] R. Freund and N. Nachtigal. QMR : a quasi-minimal residual method for non-Hermitian linear systems. *Numerische mathematik*, 60:315–339, 1991.

[31] A. Guermouche and J.-Y. L'Excellent. Constructing memory-minimizing schedules for multifrontal methods. *ACM Transactions Mathematical Software*, 32:17–32, 2006.

[32] M. Gutknecht. Lanczos-type solvers for nonsymmetric linear systems of equations. *Acta numerica*, 6:271–397, 1997.

[33] W. D. Joubert and T. A. Manteuffel. *Iterative Methods for Nonsymmetric Linear Systems*, chapter 10, pages 149–171. Academic Press, 1990.

[34] X. S. Li and J. W. Demmel. SuperLU-DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear systems. *ACM Transactions on Mathematical Software (TOMS)*, 29(2):110–140, 2003.

[35] Z. Li, Y. Saad, and M. Sosonkina. parms: A parallel version of the algebraic recursive multilevel solver. *Numerical Linear Algebra with Applications*, 10:485–509, 2003.

[36] G. Meurant. *Computer solution of large linear systems*. North Holland, Amsterdam, 1999.

[37] G. Meurant and Z. Strakos. The lanczos and conjugate gradient algorithms in finite precision arithmetic. *Acta Numerica*, 15:471–542, 2006.

[38] R.B. Morgan. A restarted GMRES method augmented with eigenvectors. *SIAM Journal on Matrix Analysis and Applications*, 16:1154–1171, 1995.

[39] R.B. Morgan. Gmres with deflated restarting. *SIAM Journal on Scientific Computing*, 24(1):20–37, 2002.

[40] R. Nabben and C. Vuik. A comparison of Deflation and Coarse Grid Correction applied to porous media flow. *SIAM J. Numer. Anal.*, 42:1631–1647, 2004.

[41] R. Nabben and C. Vuik. A comparison of deflation and the balancing preconditioner. *SIAM J. Sci. Comput.*, 27:1742–1759, 2006.

[42] C. Paige and M. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM journal on numerical analysis*, 12:617–629, 1975.

[43] C. Paige and M. Saunders. LSQR : an algorithm for sparse linear equations and sparse least squares. *ACM transactions on mathematical software*, 8:43–71, 1982.

[44] C.C. Paige, M. Rozloznik, and Z. Strakos. Modified gram-schmidt (mgs), least squares, and backward stability of mgs-gmres. *SIAM J. Matrix Anal. Appl.*, 28:264–284, 2006.

[45] B. Parlett, D. Taylor, and Z. Liu. A look-ahead Lanczos algorithm for unsymmetric matrices. *Mathematics of computation*, 44:105–124, 1985.

[46] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, 2003.

[47] Y Saad and H Schultz. GMRES: a generalized minimal residual algorithm for solving nonsymmetriclinear systems. *SIAM J. Sci. Statist. Comput.*, 7:856–869, 1986.

[48] Y. Saad and H. van der Vorst. iterative solution of linear systems in the 20th century. *Journal of Computational and Applied Mathematics*, 123:1–33, 2000.

[49] Y. Saad, M. Yeung, J. Erhel, and F. Guyomarc'h. A deflated version of the conjugate gradient algorithm. *SIAM Journal on Scientific Computing*, 21(5):1909–1926, 2000.

[50] V. Simoncini and D. Szyld. recent computational developments in krylov subpsace methods for linear systems. *Numerical linear algebra with applications*, 14:1–59, 2007.

[51] G. L. G. Sleijpen and H. A. Van der Vorst. A jacobi-davidson iteration method for linear eigenvalue problems. *SIAM J. Matrix Anal. Appl.*, 17:401–425, 1996.

[52] P. Sonnefeld. CGS, a fast Lanczos-type solver for nonsymmetric linear systems. *SIAM journal on scientific and statistical computing*, 10(36-52), 1989.

[53] Z. Strakos and P. Tichy. Error estimation in preconditioned conjugate gradients. *BIT Numerical Mathematics*, 45:789–817, 2005.

[54] A. Toseli and O. Widlund. *Domain decomposition methods-algorithms and theory*. springer series in computational mathematics, 2005.

[55] A. van der Sluis and H. van der Vorst. the rate of convergence of conjugate gradients. *Numerische Mathematik*, 48:543–560, 1986.

[56] H. van der Vorst. Bi-CGSTAB: a fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems. *SIAM journal on scientific and statistical computing*, 13:631–644, 1992.

[57] H. A. van der Vorst and Q. Ye. Residual replacement strategies for krylov subspace iterative methods for the convergence of true residuals. *SIAM J. Sci. Comput.*, 22:835–852, 2000.

[58] P. Wesseling. *An Introduction to Multigrid Methods*. Edwards, 2004.