

## 1 Introduction

Iterative methods for solving linear systems  $Ax = b$  become necessary when  $A \in \mathbb{R}^{n \times n}$  is too large to factorize directly. The meaning of *too large* depends on the context, but  $n = 10000$  up to  $n = 10^8$  is typical. In optimization, the following symmetric (but possibly indefinite) examples arise in computing search directions:

- Newton’s method for unconstrained optimization:  $H\Delta x = -g$  (where  $g$  and  $H$  are the gradient and Hessian of the objective function).
- Newton’s method for optimization with linear constraints  $Jx = b$ :  
Solve  $Z^T H Z v = -Z^T g$  and set  $\Delta x = Zv$ , where  $Z$  spans the null space of the constraint matrix ( $JZ = 0$ ).
- KKT systems with linearized constraints: 
$$\begin{pmatrix} -H & J^T \\ J & \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix} = \begin{pmatrix} g - J^T y \\ -c - Jx \end{pmatrix}.$$

The *conjugate-gradient method* (CG) is the prototype solver for  $Ax = b$  when  $A$  is symmetric and positive definite (spd). Distinguishing features of CG follow:

- $A$  is regarded as an *operator*. For various vectors  $v$ , CG asks for the matrix-vector product  $y = Av$ . This is the only way that  $A$  is defined. The first  $v$  is a multiple of  $b$ .
- Only a few work  $n$ -vectors of storage are needed to generate each approximate solution  $x_k$  ( $k = 1, 2, \dots$ ).
- With exact arithmetic, CG would terminate in at most  $n$  iterations. In practice it may need *far fewer* iterations if  $A$  has clustered eigenvalues, or *far more* if we are not so lucky. Of course the first situation is preferable.
- Favorable eigenvalue distributions can be achieved by finding a *preconditioner*  $M$  such that  $M = CC^T \approx A$  (in some sense) and solving a transformed system  $\bar{A}\bar{x} = \bar{b}$ , where  $\bar{A}$  is the operator  $C^{-1}AC^{-T} \approx I$  and the remaining quantities are obtained by solving  $C\bar{b} = b$  and  $C^T x = \bar{x}$ .
- The matrix-vector product  $y = \bar{A}v$  means “Solve  $C^T w_1 = v$ , form  $w_2 = Aw_1$ , and solve  $Cy = w_2$ ”. Thus it must be possible to solve with  $C$  and  $C^T$  reasonably efficiently (as well as multiplying by  $A$ ). The simplest example is *diagonal preconditioning* with  $C = \text{diag}(\sqrt{A_{jj}})$ .
- PCG (preconditioned CG) is a rearrangement of CG that allows solves with  $M$  itself, rather than  $C$  and  $C^T$  separately. Diagonal preconditioning then means working with the preconditioner  $M = \text{diag}(A)$ .

## 2 Lanczos-based methods for symmetric systems

We review three methods for solving symmetric systems  $Ax = b$ . As described in [11], the methods CG, MINRES, and SYMMLQ are based on the Lanczos process [8] for tridiagonalizing  $A$ . A helpful framework for viewing such methods was suggested by Paige [10]:

An iterative *process* generates certain quantities from the data. At each iteration a *subproblem* is defined, suggesting how those quantities may be combined to give a new estimate of the required solution. Different subproblems define different *methods* for solving the original problem. Different ways of solving a subproblem lead to different *implementations* of the associated method.

Typically the subproblems may be solved efficiently and stably (though stability questions are sometimes overlooked). The numerically difficult aspects are usually introduced by the process.

## 2.1 Existence

The Lanczos process is an iterative form of the symmetric orthogonal tridiagonalization  $V^T A V = T$ , derivable from the existence of the slightly larger tridiagonalization

$$\begin{pmatrix} 1 & \\ & V^T \end{pmatrix} \begin{pmatrix} 0 & b^T \\ b & A \end{pmatrix} \begin{pmatrix} 1 & \\ & V \end{pmatrix} = \begin{pmatrix} 0 & \beta_1 e_1^T \\ \beta_1 e_1 & T \end{pmatrix}, \quad (1)$$

where  $V$  could be a product of  $n - 1$  Householder matrices. From (1) we know that there exists an orthogonal  $V$  such that

$$\begin{aligned} \begin{pmatrix} 0 & b^T \\ b & A \end{pmatrix} \begin{pmatrix} 1 & \\ & V \end{pmatrix} &= \begin{pmatrix} 1 & \\ & V \end{pmatrix} \begin{pmatrix} 0 & \beta_1 e_1^T \\ \beta_1 e_1 & T \end{pmatrix} \\ \Rightarrow \begin{pmatrix} 0 & b^T V \\ b & AV \end{pmatrix} &= \begin{pmatrix} 0 & \beta_1 e_1^T \\ V \beta_1 e_1 & VT \end{pmatrix} \\ \Rightarrow b = \beta_1 v_1 &\text{ and } AV = VT. \end{aligned}$$

If  $T = \text{tridiag}(\beta_k, \alpha_k, \beta_{k+1})$  with  $v_0 = v_{n+1} \equiv 0$ , the  $k$ th column of  $AV = VT$  gives  $Av_k = \beta_k v_{k-1} + \alpha_k v_k + \beta_{k+1} v_{k+1}$  for  $k = 1, 2, \dots, n$ .

## 2.2 The Lanczos process (orthogonal tridiagonalization)

$\text{Tridiag}(A, b) \rightarrow (T_k, V_k)$  denotes the following process. Given a symmetric matrix  $A$  and a starting vector  $b$ , the Lanczos process generates vectors  $v_k$  and scalars  $\alpha_k, \beta_k$  ( $k = 1, 2, \dots$ ) according to these steps (with  $v_0 \equiv 0$ ):

1. Set  $\beta_1 v_1 = b$ . (This means  $\beta_1 \leftarrow \|b\|_2$  and then  $v_1 \leftarrow b/\beta_1$ , but exit if  $\beta_1 = 0$ .)
2. For  $k = 1, 2, \dots, \ell$  set
 

$w = Av_k$	or	$w = Av_k - \beta_k v_{k-1}$
$\alpha_k = v_k^T w$		$\alpha_k = v_k^T w$
$\beta_{k+1} v_{k+1} = w - \alpha_k v_k - \beta_k v_{k-1}$		$\beta_{k+1} v_{k+1} = w - \alpha_k v_k$

After  $k$  steps with  $\beta_1, \dots, \beta_k > 0$ , the situation may be summarized as

$$AV_k = V_k T_k + \beta_{k+1} v_{k+1} e_k^T = V_{k+1} H_k, \quad (2)$$

where  $e_k$  is the  $k$ th unit vector,  $V_k = (v_1 \ v_2 \ \dots \ v_k)$ ,  $T_k$  is tridiagonal, and  $H_k$  is also tridiagonal with one extra row:

$$T_k = \begin{pmatrix} \alpha_1 & \beta_2 & & \\ \beta_2 & \alpha_2 & \beta_3 & \\ & \ddots & \ddots & \ddots \\ & & \beta_k & \alpha_k \end{pmatrix}, \quad H_k = \begin{pmatrix} \alpha_1 & \beta_2 & & \\ \beta_2 & \alpha_2 & \beta_3 & \\ & \ddots & \ddots & \ddots \\ & & \beta_k & \alpha_k \\ & & & & \beta_{k+1} \end{pmatrix} = \begin{pmatrix} T_k \\ \beta_{k+1} e_k^T \end{pmatrix}.$$

In exact arithmetic, the columns of  $V_k$  are orthonormal and the process stops with  $k = \ell$  and  $\beta_{\ell+1} = 0$  for some  $\ell \leq n$ , and then  $AV_\ell = V_\ell T_\ell$ . For derivation purposes we assume that this happens, though in practice it is unlikely unless  $v_{k+1}$  is reorthogonalized with respect to  $V_k$  at each iteration. In any case, (2) holds to machine precision and the computed vectors satisfy  $\|V_k\|_1 \approx 1$  (even if  $k \gg n$ ).

### 2.3 Properties of the Lanczos process

From the way the Lanczos vectors are generated, it is clear that  $v_k$  lies in the *Krylov subspace*  $\mathcal{K}_k(A, b) \equiv \text{span}\{b, Ab, A^2b, \dots, A^{k-1}b\}$ . The following properties can be proved:

1. If  $A$  is changed to  $A - \sigma I$  for some scalar shift  $\sigma$ ,  $T_k$  becomes  $T_k - \sigma I$  and  $V_k$  is unaltered, showing that singular systems are commonplace. Shifted problems appear in inverse iteration or Rayleigh quotient iteration.
2. If  $A$  is positive definite, so is  $T_k$  for all  $k$ .
3. If  $A$  is indefinite, some  $T_k$  might be singular, but then by the Sturm sequence property (see [7]),  $T_k$  has exactly one zero eigenvalue and the strict interlacing property implies that  $T_{k\pm 1}$  are nonsingular. Hence  $T_k$  cannot be singular twice in a row (whether  $A$  is singular or not).
4.  $H_k$  has full column rank  $k$  for all  $k < \ell$ .
5.  $T_\ell$  is nonsingular if and only if  $b \in \text{range}(A)$ .

### 2.4 CG, MINRES, and SYMMLQ

Table 1 lists three ways to choose “optimal” points within each Krylov subspace  $\mathcal{K}_k$  (i.e., points of the form  $x_k = V_k y_k$  for some vector  $y_k$ ). The three choices lead to three *methods* for solving  $Ax = b$ , namely CG, MINRES, and SYMMLQ. Note that the CG method is not meaningful if the quadratic form is unbounded below. This means that  $A$  must be positive-definite for CG.

From (2) we see that the *residual vector* associated with a point  $x_k \in \mathcal{K}_k$  is

$$r_k \equiv b - Ax_k \quad (3)$$

$$\begin{aligned} &= \beta_1 v_1 - AV_k y_k \\ &= V_{k+1}(\beta_1 e_1 - H_k y_k) \\ &= V_{k+1} t_{k+1}, \end{aligned} \quad (4)$$

$$\text{where } t_{k+1} \equiv \beta_1 e_1 - H_k y_k. \quad (5)$$

This suggests that  $\|r_k\|$  will be small if  $y_k$  makes  $t_{k+1}$  small by some measure. Indeed, we find that Table 1’s subproblems for  $x_k$  lead to the corresponding subproblems for  $y_k$  shown in Table 2 (which also shows the factorizations needed to solve the subproblems). The CG subproblem makes  $t_{k+1} = 0$  everywhere except its last element, while the MINRES subproblem is more balanced in minimizing  $\|t_{k+1}\|$ . The SYMMLQ subproblem makes  $t_{k+1} = 0$  everywhere except its last *two* elements, while keeping  $\|y_k\|$  as small as possible.

Table 1: Minimization properties for three methods for solving  $Ax = b$ . They seek points  $x_k = V_k y_k$  that give small residual vectors  $r_k \equiv b - Ax_k$ .

Method	Definition of optimal $x_k$ in Krylov subspace
CG	$\min \frac{1}{2} x_k^T A x_k - b^T x_k \quad \text{s.t. } x_k \in \mathcal{K}_k$ $\equiv \min \ r_k\ _{A^{-1}}^2 \quad \text{s.t. } x_k \in \mathcal{K}_k$
MINRES	$\min \ r_k\ ^2 \quad \text{s.t. } x_k \in \mathcal{K}_k$
SYMMLQ	$\min \ x_k\ ^2 \quad \text{s.t. } x_k \in \mathcal{K}_k, r_k \perp \mathcal{K}_{k-1}$ $\equiv \min \ x - x_k\ ^2 \quad \text{s.t. } x_k \in A\mathcal{K}_{k-1}$

Table 2: Subproblems defining  $y_k$  and  $x_k = V_k y_k$  for four methods.

Method	Subproblem	Factorization	Estimate of $x$
CG	$T_k y_k = \beta_1 e_1$	$T_k = L_k D_k L_k^T$	$x_k^C = V_k y_k$
MINRES	$\min \ H_k y_k - \beta_1 e_1\ $	$Q_k H_k = \begin{pmatrix} R_k \\ 0 \end{pmatrix}$	$x_k^M = V_k y_k$
SYMMLQ	$\min \ y_k\ $ s.t. $H_{k-1}^T y_k = \beta_1 e_1$	$H_{k-1}^T Q_{k-1}^T = (L_{k-1} \ 0)$	$x_k^L = V_k y_k$
MINRES-QLP	$\min \ y_k\ $ s.t. $\ H_k y_k - \beta_1 e_1\  = \min$	$Q_k H_k = \begin{pmatrix} R_k \\ 0 \end{pmatrix}$ $R_k P_k = \bar{L}_k$	$x_k^Q = V_k y_k$

Table 3: Definition of  $W_k$ ,  $\bar{W}_k$ ,  $z_k$ ,  $\bar{z}_k$  such that  $x_k = V_k y_k = W_k z_k$  or  $\bar{W}_k \bar{z}_k$ .

	$W_k$	$z_k$	Estimate of $x$
CG	$V_k L_k^{-T}$	$L_k D_k z_k = \beta_1 e_1$	$x_k^C = W_k z_k$
MINRES	$V_k R_k^{-1}$	$\begin{pmatrix} z_k \\ \bar{\zeta}_{k+1} \end{pmatrix} = Q_k \begin{pmatrix} \beta_1 e_1 \\ 0 \end{pmatrix}$	$x_k^M = W_k z_k$
SYMMLQ	$\bar{W}_k = V_k Q_{k-1}^T$ $= [W_{k-1} \ \bar{w}_k]$	$L_{k-1} z_{k-1} = \beta_1 e_1$ $\bar{\zeta}_k = 0$	$x_k^L = \bar{W}_k \bar{z}_k$ $= W_{k-1} z_{k-1}$
MINRES-QLP	$\bar{W}_k = V_k P_k$	$\bar{L}_k \bar{z}_k = z_k$	$x_k^Q = \bar{W}_k \bar{z}_k$

If  $A$  is positive definite, each  $T_k$  is theoretically positive definite and CG can obtain Cholesky factors  $T_k = L_k D_k L_k^T$ . MINRES uses the QR factorization of  $H_k$ , and is applicable to any symmetric  $A$  (including singular systems if suitable stopping criteria are implemented). SYMMLQ uses the same QR factorization, disguised as the LQ factorization of  $H_k^T$ , and is again applicable to any symmetric  $A$ , except that  $Ax = b$  must be consistent. MINRES-QLP works with a two-sided orthogonal factorization of  $H_k$  for greater reliability on ill-conditioned or singular systems (see Choi [1], Choi et al. [2]). The QLP name comes from Stewart [18].

Note that all elements of  $y_k$  may change in  $y_{k+1}$ . Also, we don't wish to store all of  $V_{k+1}$  in order to form  $V_{k+1} y_{k+1}$ . Thus, each method computes certain quantities  $W_k$  and  $z_k$  that allow the solution estimates to be *updated*. Table 3 shows all the possibilities we can think of. (Tables 2–3 are from [15], except for the more recent MINRES-QLP entries.)

Also note that  $V_k(\beta_1 e_1) = b$  *exactly* for all  $k$  because  $v_1$  is a multiple of  $b$ . Thus, the relations  $r_k = V_{k+1} t_{k+1}$  (4) and  $t_{k+1} = \beta_1 e_1 - H_k y_k$  (5) hold accurately for any  $y_k$  even though the columns of  $V_{k+1}$  lose orthogonality. Since

$$\|r_k\| \leq \|V_{k+1}\| \|\beta_1 e_1 - H_k y_k\|$$

with  $\|V_{k+1}\| = O(1)$  and  $\|\beta_1 e_1 - H_k y_k\|$  tending to decrease for the given choices of  $y_k$ , we can expect  $\|r_k\|$  to become small eventually.

**The CG iteration** CG treats  $T_k y_k = \beta_1 e_1$  as  $L_k D_k (L_k^T y_k) = \beta_1 e_1$ , defines  $z_k \equiv L_k^T y_k$ , and solves the lower-triangular systems

$$\begin{aligned} L_k D_k z_k &= \beta_1 e_1, & z_k &= \begin{pmatrix} z_{k-1} \\ \zeta_k \end{pmatrix} \\ L_k W_k^T &= V_k^T, & W_k^T &= \begin{pmatrix} W_{k-1}^T \\ w_k^T \end{pmatrix} \end{aligned}$$

by computing *only the last elements of the solutions* at each iteration  $k$ , taking advantage of the fact that the preceding parts of  $z_k$  and  $W_k$  have already been computed. For example, since  $L_k$  is *lower bidiagonal with unit diagonals*, we can form  $w_k = v_k - \lambda_k w_{k-1}$  efficiently, where  $\lambda_k$  is the  $(k, k-1)$  element of  $L_k$ . Thus,

$$\begin{aligned} x_k &= V_k y_k = W_k L_k^T y_k \\ &= W_k z_k \\ &= W_{k-1} z_{k-1} + w_k \zeta_k \\ &= x_{k-1} + \zeta_k w_k \end{aligned}$$

can also be formed cheaply. The vectors in  $V_{k-2}$  and  $W_{k-1}$  are no longer needed.

Although we don't want all of  $y_k$ , we see from  $L_k^T y_k = z_k$  that the last element of  $y_k$  is  $\eta_k = \zeta_k$ . Also from (4)–(5) and the fact that CG makes  $t_{k+1}$  zero except for its last element  $\tau_{k+1} \equiv e_{k+1}^T t_{k+1}$ , we see that the CG residual vector satisfies  $r_k^C = \tau_{k+1} v_{k+1}$  and hence  $\|r_k^C\| = |\tau_{k+1}| = |-\beta_{k+1} \eta_k| = |\beta_{k+1} \zeta_k|$ . Thus when CG is implemented this way, we have an accurate estimate of  $\|r_k^C\|$  at essentially no cost.

**The MINRES iteration** By definition, the MINRES point  $x_k^M$  solves the problem

$$\min_{y_k} \|r_k\| \text{ such that } x_k = V_k y_k,$$

so that  $\|r_k^M\|$  decreases monotonically as long as the columns of  $V_k$  remain independent. (Remember they are theoretically orthonormal.) Many users prefer MINRES for this reason. To allow for inconsistent systems, the stopping rule must check both  $\|r_k\|$  and  $\|Ar_k\|$ . The iteration is similar to CG in solving  $R_k^T W_k^T = V_k^T$  for each  $w_k$  in turn and updating  $x_k = x_{k-1} + \zeta_k w_k$ . There is more work and storage because  $R_k^T$  is lower *tridiagonal*. A numerical concern is that the columns of  $W_k = V_k R_k^{-1}$  could be large if some of the  $R_k$  are ill-conditioned.

If  $A$  is positive definite, we now know that  $\|x_k^M\|$  is monotonically increasing (Fong [3, 5]), so there should not be significant cancellation error in forming  $x_k^M = x_{k-1}^M + \zeta_k w_k$ . But there does seem to be a risk of cancellation when  $A$  is indefinite. This risk is avoided by MINRES-QLP (see below).

To see how the QR factorization  $Q_k (H_k \ \beta_1 e_1) = \begin{pmatrix} R_k & z_k \\ 0 & \bar{\zeta}_{k+1} \end{pmatrix}$  proceeds, consider the effect of the first plane rotation when  $k = 4$ :

$$\begin{pmatrix} c_1 & s_1 & & & \\ s_1 & -c_1 & & & \\ & & 1 & & \\ & & & 1 & \\ & & & & 1 \end{pmatrix} \begin{pmatrix} \alpha_1 & \beta_2 & & & \beta_1 \\ \beta_2 & \alpha_2 & \beta_3 & & 0 \\ & \beta_3 & \alpha_3 & \beta_4 & 0 \\ & & \beta_4 & \alpha_4 & 0 \\ & & & \beta_5 & 0 \end{pmatrix} = \begin{pmatrix} \rho_1 & \sigma_2 & \tau_3 & & \zeta_1 \\ & \bar{\rho}_2 & \bar{\sigma}_3 & & \bar{\zeta}_2 \\ & & \beta_3 & \alpha_3 & \beta_4 & 0 \\ & & & \beta_4 & \alpha_4 & 0 \\ & & & & \beta_5 & 0 \end{pmatrix},$$

where

$$\begin{aligned} \rho_1 &= \sqrt{\alpha_1^2 + \beta_2^2} & \sigma_2 &= c_1 \beta_2 + s_1 \alpha_2 & \tau_3 &= s_1 \beta_3 & \zeta_1 &= c_1 \beta_1 \\ c_1 &= \alpha_1 / \rho_1, \quad s_1 = \beta_2 / \rho_1 & \bar{\rho}_2 &= s_1 \beta_2 - c_1 \alpha_2 & \bar{\sigma}_3 &= -c_1 \beta_3 & \bar{\zeta}_2 &= s_1 \beta_1 \end{aligned}$$

and barred items will become unbarred after the second rotation:

$$\begin{pmatrix} 1 & & & & \\ & c_2 & s_2 & & \\ & s_2 & -c_2 & & \\ & & & 1 & \\ & & & & 1 \end{pmatrix} \begin{pmatrix} \rho_1 & \sigma_2 & \tau_3 & & \zeta_1 \\ & \bar{\rho}_2 & \bar{\sigma}_3 & & \bar{\zeta}_2 \\ & \beta_3 & \alpha_3 & \beta_4 & 0 \\ & & \beta_4 & \alpha_4 & 0 \\ & & & \beta_5 & 0 \end{pmatrix} = \begin{pmatrix} \rho_1 & \sigma_2 & \tau_3 & & \zeta_1 \\ & \rho_2 & \sigma_3 & \tau_4 & \zeta_2 \\ & & \bar{\rho}_3 & \bar{\sigma}_4 & \bar{\zeta}_3 \\ & & \beta_4 & \alpha_4 & 0 \\ & & & \beta_5 & 0 \end{pmatrix}.$$

Similarly,

$$\begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & c_3 & s_3 & \\ & & s_3 & -c_3 & \\ & & & & 1 \end{pmatrix} \begin{pmatrix} \rho_1 & \sigma_2 & \tau_3 & & \zeta_1 \\ & \rho_2 & \sigma_3 & \tau_4 & \zeta_2 \\ & & \bar{\rho}_3 & \bar{\sigma}_4 & \bar{\zeta}_3 \\ & & \beta_4 & \alpha_4 & 0 \\ & & & \beta_5 & 0 \end{pmatrix} = \begin{pmatrix} \rho_1 & \sigma_2 & \tau_3 & & \zeta_1 \\ & \rho_2 & \sigma_3 & \tau_4 & \zeta_2 \\ & & \rho_3 & \sigma_4 & \zeta_3 \\ & & & \bar{\rho}_4 & \bar{\zeta}_4 \\ & & & \beta_5 & 0 \end{pmatrix},$$

and finally

$$\begin{pmatrix} 1 & & & & \\ & 1 & & & \\ & & 1 & & \\ & & & c_4 & s_4 \\ & & & s_4 & -c_4 \end{pmatrix} \begin{pmatrix} \rho_1 & \sigma_2 & \tau_3 & & \zeta_1 \\ & \rho_2 & \sigma_3 & \tau_4 & \zeta_2 \\ & & \rho_3 & \sigma_4 & \zeta_3 \\ & & & \bar{\rho}_4 & \bar{\zeta}_4 \\ & & & \beta_5 & 0 \end{pmatrix} = \begin{pmatrix} \rho_1 & \sigma_2 & \tau_3 & & \zeta_1 \\ & \rho_2 & \sigma_3 & \tau_4 & \zeta_2 \\ & & \rho_3 & \sigma_4 & \zeta_3 \\ & & & \rho_4 & \zeta_4 \\ & & & & \bar{\zeta}_5 \end{pmatrix}.$$

In particular, we see that  $\bar{\zeta}_{k+1} = s_k \bar{\zeta}_k = s_k s_{k-1} \dots s_1 \beta_1$  is monotonically decreasing. This is the residual norm for the least-squares problem  $\min \|t_{k+1}\| \equiv \|H_k y_k - \beta_1 e_1\|$ . If we assume  $V_{k+1}$  is orthonormal, (3)–(5) show that  $\|r_k^M\| = \|t_{k+1}\| = \bar{\zeta}_{k+1}$ , which is cheaply available.

**The SYMMLQ iteration** In contrast to MINRES, SYMMLQ's point  $x_k^L$  solves

$$\min_{y_k} \|x_k\| \text{ such that } x_k = V_k y_k \text{ and } V_{k-1}^T r_k = 0,$$

so that  $\|x_k^L\|$  increases and the system must be consistent ( $\|r_k^L\| \rightarrow 0$ ). It also solves

$$\min_w \|x - x_k\| \text{ such that } x_k = AV_{k-1}w$$

[6, 9], so that the error norm  $\|x - x_k^L\|$  decreases monotonically.

The SYMMLQ solutions  $x_k^L = W_{k-1} z_{k-1} = x_{k-1}^L + \zeta_{k-1} w_{k-1}$  are accumulated as a sequence of theoretically *orthogonal* steps. Although the columns of  $V_k$  and  $W_{k-1}$  are not likely to be orthonormal in practice, we will always have  $\|w_{k-1}\| \approx 1$  with  $\|x_k^L\|$  increasing, so that forming  $x_k^L$  should involve very little cancellation error, even if  $A$  is indefinite.

By observation,  $\|r_k\|$  is often much larger for SYMMLQ than for the other methods. This is not cause for concern. It's a sign that SYMMLQ is stepping around points that would be troublesome for CG. Since the residual norms can be estimated cheaply, SYMMLQ has provision for *transferring* to the CG point upon termination if the residual is then smaller. Thus, if  $\|r_k^C\| < \|r_k^L\|$ , SYMMLQ takes a final step of the form  $x_k^C = x_k^L + \bar{\zeta}_k \bar{w}_k$ , where the last two items are already known.

Note that after  $k$  iterations, SYMMLQ has solved a single triangular system  $L_{k-1} z_{k-1} = \beta_1 e_1$ , and this is the only place where ill-conditioning in  $A$  becomes evident while  $k < \ell$ . (At the end with  $\beta_{\ell+1}$  theoretically zero, the condition of  $T_\ell$  is critical, but SYMMLQ reaches  $x_\ell^C$  in the safest way.) We therefore believe that SYMMLQ is the method of choice for indefinite consistent systems. MINRES-QLP should be comparable in reliability at the cost of slightly more work and storage per iteration, and it handles singular systems well.

**MINRES-QLP** The effects of rounding errors on the convergence of CG, MINRES, and SYMMLQ have been analyzed by Sleijpen et al. [16]. Some numerical examples confirm that MINRES may not achieve small  $\|r_k\|$  on consistent systems when  $A$  is very ill-conditioned.

MINRES-QLP is the only method that returns the minimum-length solution on singular inconsistent systems  $Ax \approx b$ . It is significantly more complex (see Choi [1]) but can be more reliable than MINRES when  $A$  is ill-conditioned. In [1] it is anticipated that the rounding errors in MINRES's solution of the  $n$  independent ill-conditioned triangular systems  $R_k^T W_k^T = V_k^T$  (i.e., in the  $n$  rows of  $W_k$ ) are more significant than in MINRES-QLP's solution of the *single* ill-conditioned system  $\bar{L}_k \bar{z}_k = z_k$ , as in SYMMLQ's  $L_{k-1} z_{k-1} = \beta_1 e_1$ .

## 2.5 Estimation of norms

At iteration  $k$  of the above solvers, estimates of  $\|r_k\|$ ,  $\|Ar_k\|$ ,  $\|x_k\|$ ,  $\|A\|$ , and  $\text{cond}(A)$  are needed in order to implement reliable stopping rules. The estimates have been studied most fully for MINRES and MINRES-QLP in Choi [1]. In particular,  $\|A\|_2 \approx \|T_k\|_2$  or  $\|H_k^T H_k\|_1^{1/2}$  are reasonable estimates that can be estimated cheaply as the iterations proceed. Different solvers estimate the other quantities in various ways.

## 2.6 Stopping rules

An approximate solution  $x_k$  may be regarded as acceptable if it is the *exact* solution for a slightly different problem (with  $A$  and  $b$  perturbed). This is the *backward error* point of view. For consistent systems  $Ax = b$  with uncertainty in  $A$  and  $b$ , we will see in the next chapter that we can stop if

$$\|r_k\| \leq \alpha \|A\| \|x_k\| + \beta \|b\| \quad (6)$$

for some user-specified tolerances that reflect the uncertainty in  $A$  and  $b$  (e.g.,  $\alpha = \beta = \text{tol} = 10^{-4}$ ,  $10^{-8}$ , or  $10^{-12}$  respectively for moderate, accurate, or very accurate solutions using 15-digit arithmetic). For inconsistent problems where  $\|r_k\| \not\rightarrow 0$ , a good stopping rule for MINRES and MINRES-QLP is

$$\|Ar_k\| \leq \alpha \|A\| \|r_k\|. \quad (7)$$

This is a special symmetric version of Stewart's result [17] for rectangular least-squares problems  $\min \|Ax - b\|$ , also used in the next chapter. The norms required in tests (6)–(7) can be estimated cheaply as the iterations proceed. Fortunately, the estimates of  $\|r_k\|$  and  $\|Ar_k\|$  are remarkably accurate until one of them approaches zero, and even then one of them tends to keep decreasing—hence causing termination as desired.

Note that stopping rule (6) is equivalent to stopping if  $\psi_k \equiv \|r_k\| / (\alpha \|A\| \|x_k\| + \beta \|b\|) \leq 1$ . On positive definite  $Ax = b$ ,  $\psi_k$  is *monotonically decreasing* for MINRES (because  $\|r_k^M\|$  and  $\|x_k^M\|$  are monotonic), and empirically becomes significantly smaller than  $\psi_k$  for CG. Hence, we believe that *MINRES is often preferable to CG* in the sense that it can stop sooner.

## 2.7 Cautions

The methods described above are reliable in practice, *even though the columns of  $V_k$  soon become far from orthonormal*. The work and storage per iteration are constant and minimal ( $O(n)$ ). The main question remaining is, how many iterations will be required? We hope for far fewer than  $n$  iterations, but it could be  $5n$  or  $10n$  or even more.

If reorthogonalization were used to maintain orthonormal  $V_k$ , the iterations would be bounded by  $n$  (and more precisely by the number of clusters in the eigenvalue of  $A$ ). However, all of the vectors  $v_k$  would need to be stored, and the work and storage would grow quadratically. We consider this not an option in general (although it is tolerated with GMRES [14]).

Many authors present equation (2) correctly, but then derive further results from two equations that don't hold unless full reorthogonalization is used. We emphasize that there

is *no need to assume* that  $V_k^T A V_k = T_k$  and/or  $V_k^T b = \beta_1 e_1$ , which quickly cease to be true without reorthogonalization of  $V_k$ . Luckily, equations (2) and (4) are sufficient as they stand.

Similarly, many authors allow an approximate solution  $x_0$  to be provided, and proceed to update the solution inside the solver when it is applied to the system  $Ad = r_0$ , where  $r_0 = b - Ax_0$  and  $x = x_0 + d$ . Compare the implementations

$$\begin{array}{ll} x \leftarrow x_0 & \text{for } k = 1 : K, \quad x \leftarrow x + \textit{correction}, \quad \text{end} \\ d \leftarrow 0 & \text{for } k = 1 : K, \quad d \leftarrow d + \textit{correction}, \quad \text{end} \quad x = x_0 + d. \end{array}$$

The first choice is *not recommended* when  $x_0$  is a good approximation, because the  $x$  *corrections* could be small relative to  $x_0$  and many significant digits could be lost. The second choice is safer, at the cost of storing  $x_0$  elsewhere. For this choice, we need to be conscious of solving  $Ad = r_0$  when choosing stopping tolerances. If anything is to be gained from  $x_0$ , we need *looser tolerances* than if we were solving  $Ax = b$  itself.

*Moral:* The MATLAB iterative solvers use  $\alpha = 0$  and  $\beta = \mathbf{tol}$ . If you have a good  $x_0$ , always input  $b = r_0$  and  $x_0 = 0$  and solve  $Ad = r_0$  with loose tolerance  $\mathbf{tol}0$ , then form  $x = x_0 + d$ . It is hard to guess the tolerance, but perhaps it could be  $\mathbf{tol}0 = \mathbf{tol} \times \|b\|/\|r_0\|$ .

*Simple solution:* Our own implementations of the iterative solvers enforce caution by not allowing  $x_0$  to be input. To help future users we could accept  $x_0$ , compute  $r_0 = b - Ax_0$ , and note that the residuals for  $Ad = r_0$  and  $Ax = b$  are the same:  $r_k = r_0 - Ad_k = b - A(x_0 + d_k)$ . Hence, although the solver is computing iterates  $d_k$  for solving  $Ad = r_0$ , we can estimate  $\|r_k\|$  cheaply and we can also compute  $\|x_k\| = \|x_0 + d_k\|$  for use in (6).

## 2.8 Augmented systems

The following symmetric system underlies several methods for more general problems:

$$\widehat{A}_{\gamma,\delta} \widehat{x} = \widehat{b} \quad \equiv \quad \begin{pmatrix} \gamma I & A \\ A^T & \delta I \end{pmatrix} \begin{pmatrix} s \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}, \quad (8)$$

where  $A$  is a square or rectangular matrix and  $\gamma, \delta$  are specified scalars. The Lanczos process  $\text{Tridiag}(\widehat{A}_{\gamma,\delta}, \widehat{b})$  has a special structure that is interesting to observe. After  $2k$  steps it generates

$$T_{2k} = \begin{pmatrix} \gamma & \alpha_1 & & & & & & \\ \alpha_1 & \delta & \beta_2 & & & & & \\ & \beta_2 & \gamma & \alpha_2 & & & & \\ & & & \cdot & \cdot & \cdot & & \\ & & & & \beta_k & \gamma & \alpha_k & \\ & & & & & \alpha_k & \delta & \end{pmatrix}, \quad V_{2k} = \begin{pmatrix} u_1 & & & & & & & \\ & u_2 & & & & & & \\ & & v_1 & & & & & \\ & & & v_2 & & & & \\ & & & & \dots & & & \\ & & & & & u_k & & \\ & & & & & & v_k & \end{pmatrix},$$

and the next step gives  $T_{2k+1}, V_{2k+1}$  in the obvious way. We find that the scalars  $\alpha_k, \beta_k$  and vectors  $u_k, v_k$  are independent of  $\gamma$  and  $\delta$ . We can therefore generate them with  $\gamma = \delta = 0$ . However, that case is more efficiently generated by the Golub-Kahan process  $\text{Bidiag}(A, b)$  described later.

For certain choices of  $\gamma$  and  $\delta$ , the subproblems associated with CG and MINRES can also be rearranged to improve efficiency, leading to algorithms CRAIG ( $\gamma = \delta = 0$ ) [12], LSQR and LSMR ( $\gamma = 1, \delta = 0$  or  $\gamma = -\delta \neq 0$ ) [12, 13, 4], and AMRES ( $\gamma = \delta$ ) [3, Ch5–6]. Again it is simpler to derive those algorithms directly from  $\text{Bidiag}(A, b)$ .



## References

- [1] S.-C. Choi. *Iterative Methods for Singular Linear Equations and Least-Squares Problems*. PhD thesis, ICME, Stanford University, Dec 2006.
- [2] S.-C. Choi, C. C. Paige, and M. A. Saunders. MINRES-QLP: A Krylov subspace method for indefinite or singular symmetric systems. *SIAM J. Sci. Comput.*, 33(4):1810–1836, 2011. <http://stanford.edu/group/SOL/software.html>.
- [3] D. C.-L. Fong. *Minimum-Residual Methods for Sparse Least-Squares Using Golub-Kahan Bidiagonalization*. PhD thesis, ICME, Stanford University, Dec 2011.
- [4] D. C.-L. Fong and M. A. Saunders. LSMR: An iterative algorithm for least-squares problems. *SIAM J. Sci. Comput.*, 33(5):2950–2971, 2011. <http://stanford.edu/group/SOL/software.html>.
- [5] D. C.-L. Fong and M. A. Saunders. CG versus MINRES: An empirical comparison. *SQU Journal for Science*, 17(1):44–62, 2012. <http://stanford.edu/group/SOL/reports/SOL-2011-2R.pdf>.
- [6] R. W. Freund. *Über einige CG-ähnliche Verfahren zur Lösung linearer Gleichungssysteme*. PhD thesis, Universität Würzburg, FRG, 1983.
- [7] G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. The Johns Hopkins University Press, Baltimore, third edition, 1996.
- [8] C. Lanczos. An iteration method for the solution of the eigenvalue problem of linear differential and integral operators. *J. Res. Natl. Bur. Stand.*, 45:255–282, 1950.
- [9] D. P. O’Leary, 1990. Private communication.
- [10] C. C. Paige. Krylov subspace processes, Krylov subspace methods and iteration polynomials. In J. D. Brown, M. T. Chu, D. C. Ellison, and R. J. Plemmons, editors, *Proceedings of the Cornelius Lanczos International Centenary Conference, Raleigh, NC, Dec. 1993*, pages 83–92. SIAM, Philadelphia, 1994.
- [11] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12:617–629, 1975.
- [12] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Software*, 8(1):43–71, 1982.
- [13] C. C. Paige and M. A. Saunders. Algorithm 583; LSQR: Sparse linear equations and least-squares problems. *ACM Trans. Math. Software*, 8(2):195–209, 1982.
- [14] Y. Saad and M. H. Schultz. GMRES: a generalized minimum residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. and Statist. Comput.*, 7:856–869, 1986.
- [15] M. A. Saunders. Solution of sparse rectangular systems using LSQR and CRAIG. *BIT Numer. Math.*, 35:588–604, 1995.
- [16] G. L. G. Sleijpen, H. A. Van der Vorst, and J. Modersitzki. Differences in the effects of rounding errors in Krylov solvers for symmetric indefinite linear systems. *SIAM J. Matrix Anal. Appl.*, 22(3):723–751, 2000.
- [17] G. W. Stewart. Research, development and LINPACK. In J. R. Rice, editor, *Mathematical Software III*, pages 1–14. Academic Press, New York, 1977.
- [18] G. W. Stewart. The QLP approximation to the singular value decomposition. *SIAM J. Sci. Comput.*, 20(4):1336–1348, 1999.