SIAM J. Matrix Anal. Appl.
Vol. 37, No. 3, pp. 1071–1087

NUMERICAL EQUIVALENCES AMONG KRYLOV SUBSPACE ALGORITHMS FOR SKEW-SYMMETRIC MATRICES*

C. GREIF†, C. C. PAIGE‡, D. TITLEY-PELOQUIN§, AND J. M. VARAH†

Abstract. We briefly review Krylov subspace methods based on the Galerkin and minimum residual conditions for solving $Ax = b$ with real A and b, followed by two implementations: the conjugate gradient (CG) based methods CGNE and CGNR. We then show the numerical equivalence of Lanczos tridiagonalization and Golub–Kahan bidiagonalization for any real skew-symmetric matrix A. We give short derivations of two algorithms for solving $Ax = b$ with skew-symmetric A and use the above equivalence to show that these are numerically equivalent to the Golub–Kahan bidiagonalization variants of CGNE and CGNR. These last two numerical equivalences add to the theoretical equivalences in the work by Eisenstat [*Equivalence of Krylov Subspace Methods for Skew-Symmetric Linear Systems*, Department of Computer Science, Yale University, preprint, arXiv:1512.00311, 2015] that unified and extended earlier work. We next present a method based on the Lanczos tridiagonalization process for minimizing $\|A^T(b - Ax_k)\|_2$ when $A^T = -A$ and show that for skew-symmetric systems it is numerically equivalent to LSMR developed by Fong and Saunders [*SIAM J. Sci. Comput.*, 33 (2011), pp. 2950–2971]. Finally, we illustrate the typical convergence behaviors of these algorithms with a numerical example and use these and an analysis to give new insights into algorithm choices for *general* large sparse matrix solution of equations problems.

Key words. Krylov subspace methods, skew-symmetric systems, Lanczos tridiagonalization, Golub–Kahan bidiagonalization, CGNE, Craig's method, CGNR, LSQR, LSMR

AMS subject classifications. 65F10, 65F20, 65F25, 65F30, 65F50, 15A23, 15A57

DOI. 10.1137/15M1030078

1. Introduction.

For skew-symmetric matrices $A = -A^T \in \mathbb{R}^{n\times n}$ we will examine iterative orthogonal transformations to tridiagonal or bidiagonal forms and the use of Krylov subspace methods based on these for solving systems of equations $Ax = b$ and least squares problems $\min_x \|b - Ax\|_2$. For the kth approximation x_k to such x we will consider three optimality criteria: minimizing the error $\|x - x_k\|_2$, the residual $\|b - Ax_k\|_2$, and the normal equations residual $\|A^T(b - Ax_k)\|_2$.

Skew-symmetric linear systems present distinct challenges compared to symmetric systems. For example, if A is a skew-symmetric matrix, then $x^TAx = 0$ for any vector x, which eliminates the possibility of considering a method based directly on minimizing the A-norm of the error $\sqrt{(x - x_k)^TA(x - x_k)}$, such as the conjugate gradient (CG) method. In addition, it is not easy to develop preconditioners that preserve skew-symmetry. It is enough for the dimensions of a skew-symmetric matrix to be odd for it to be singular, so since we will be interested in solution of equations, for simplicity here we will assume that all our skew-symmetric matrices are nonsingular.

On the other hand, skew-symmetric matrices have unique mathematical properties. Their diagonal entries are zero, and their nonzero entries are completely deter-

*Received by the editors July 10, 2015; accepted for publication (in revised form) by J. Liesen April 5, 2016; published electronically August 18, 2016.

http://www.siam.org/journals/simax/37-3/M103007.html

†Department of Computer Science, The University of British Columbia, Vancouver, BC, V6T 1Z4, Canada (greif@cs.ubc.ca, jmvarah@gmail.com). The first author's research was supported by NSERC of Canada grant 261539.

‡School of Computer Science, McGill University, Montreal, Quebec, H3A 2A7, Canada (paige@cs.mcgill.ca). This author's research was supported by NSERC of Canada grant OGP0009236.

§Bioresource Engineering Department, McGill University, Macdonald Campus, Ste-Anne-de-Bellevue, Quebec, H9X 3V9, Canada (david.titley-peloquin@mcgill.ca).

mined by their strictly lower or upper triangular part. The symmetry of their (pure imaginary) eigenvalues with respect to the origin is intriguing. These properties call out for exploitation by specialized solution methods. We discuss some history of these in section 3 after an introduction to the relevant Krylov subspace methods.

This paper was initially motivated by the realization that the development in [9] could be simplified. Greif and Varah [9] derived Krylov subspaces via the Lanczos tridiagonalization [15] for skew-symmetric A and applied the Galerkin and minimum residual conditions to provide two iterative solution methods for skew-symmetric systems. In a recent manuscript, Eisenstat [4] provided algorithm-independent proofs showing that for skew-symmetric systems, with exact arithmetic there is no advantage in using the Galerkin method over the classic Craig's method or in using the minimum residual method over the CG method applied to the normal equations. We elaborate on these observations in section 3 and below.

Following these insights the present paper shows the *numerical* equivalence of carefully implemented pairs of the above, and other solution of equations methods, by showing the numerical equivalence of Lanczos tridiagonalization and Golub–Kahan bidiagonalization (GKB) [7] for skew-symmetric matrices.

In section 2 we briefly review Krylov subspace methods based on the Galerkin and minimum residual conditions for solving general $Ax = b$ problems, followed by two implementations: the CG based methods CGNE and CGNR. These are described, for example, in [8, Chap. 7]. In section 3 we provide some remarks on related work and relevant references. Recently we have realized that the structure of the tridiagonal matrix arising in the Lanczos tridiagonalization process for skew-symmetric matrices, described in section 4, can be used to develop $Ax = b$ solution methods based on this Lanczos process that do not require the usual decomposition of the tridiagonal matrix. We show in sections 5 and 6 that for skew-symmetric A two Lanczos process steps are equivalent to one step of the GKB process. Sections 7 and 8 use this structure and equivalence to show that if the Galerkin and minimum residual approaches to solving $Ax = b$ with such A are based on the Lanczos process, the resulting algorithms are equivalent to the well-known CGNE and CGNR methods, respectively. The method LSMR in [5] uses GKB to minimize $\|A^T(b - Ax_k)\|_2$ at each step for general A. In section 9 we develop the Lanczos process to carry out this minimization and show that it is equivalent to LSMR for skew-symmetric A. We briefly discuss preconditioning in section 10 and examine the behavior of these algorithms in section 11. Section 12 corrects an error in the printed version of the classic Hestenes and Stiefel paper [11] and provides new insights into the choice of algorithms for large sparse systems of equations with general unsymmetric matrices.

Notation. We will use U_k, V_k, and W_k to indicate matrices of k vectors, e.g., $W_k = [w_1, \ldots, w_k]$, and produce theoretically orthonormal versions of these where appropriate. We use "$\triangleq$" to mean "is defined to be" and use ":=" to mean "is computed to be," where for simplicity any normalization of the form "$w\beta := b$" will be shorthand for "$\beta := \|b\|_2$, stop if $\beta = 0$, otherwise $w := b/\beta$."

2. Krylov subspace methods for linear systems of equations. We consider general A to set up the background. First, Krylov subspace methods for solving

$$Ax = b, \qquad A \in \mathbb{R}^{n\times n} \quad \text{nonsingular}, \tag{2.1}$$

compute a sequence $\{x_k\}$ of approximate solutions where for simplicity we take $x_0 = 0$:

$$x_k \in \mathcal{K}_k(A, b) \triangleq \text{span}\{b, Ab, \ldots, A^{k-1}b\} = \text{Range}(W_k), \quad \text{say},$$

where W_k is a rank-k matrix, typically with orthonormal columns. A vector x_k^G satisfies the Galerkin condition ($\cdot^G$ for "Galerkin") if

$$x_k^G = W_k y_k^G, \qquad W_k^T(b - AW_k y_k^G) = 0, \qquad \text{Range}(W_k) = \mathcal{K}_k(A, b). \tag{2.2}$$

If A is symmetric positive definite (SPD), then a unique x_k^G always exists and minimizes the A-norm of the error, as is seen by differentiating with respect to y_k

$$\|x - x_k\|_A^2 \triangleq (x - x_k)^T A(x - x_k) = (x - W_k y_k)^T (b - AW_k y_k). \tag{2.3}$$

In our case, considering $(x - x_k)^T A(x - x_k)$ is inappropriate since with $A^T = -A$ this will be zero for all x_k. Worse still, y_k^G might not always exist in (2.2); see section 3.

Instead of the Galerkin condition, x_k could be the vector that minimizes the Euclidean norm of the residual $r_k \triangleq b - Ax_k$, where with $\cdot^M$ for "Minimum residual,"

$$x_k^M = W_k y_k^M, \qquad y_k^M = \arg\min_{y_k} \|b - AW_k y_k\|_2^2, \qquad \text{Range}(W_k) = \mathcal{K}_k(A, b). \tag{2.4}$$

This x_k^M always exists and is unique at least until the last step. It minimizes the A^TA norm of the error associated with x_k^M and satisfies $W_k^T A^T(b - Ax_k^M) = 0$.

The CG method [11] is a Krylov subspace method based on the Galerkin condition (2.2) for solving $M\tilde{x} = c$ for SPD M. Thus from (2.2)

$$\tilde{x}_j = \widetilde{W}_j z_j, \qquad \widetilde{W}_j^T(c - M\tilde{x}_j) = 0, \qquad \text{Range}(\widetilde{W}_j) = \mathcal{K}_j(M, c). \tag{2.5}$$

Instead of (2.1), to handle general possibly nonsquare A, CG can be applied to different forms of $M\tilde{x} = c$ with SPD M; see, e.g., [8, Chap. 7].

First, Craig's method [3] is useful for consistent systems $Ax = b$, e.g., when A is $m \times n$ of rank m. Craig's method is also called CGNE, and uses CG to solve $AA^Ty = b$, and then evaluates $x = A^Ty$, so that using $\cdot^E$ to denote CGNE, (2.5) leads to

$$y_j^E = U_j z_j^E, \quad U_j^T(b - AA^T U_j z_j^E) = 0, \quad \text{Range}(U_j) = \mathcal{K}_j(AA^T, b), \quad x_j^E = A^T y_j^E. \tag{2.6}$$

Because of the Galerkin condition, with full row rank A this minimizes the AA^T-norm of $y - y_j^E$; see (2.3). But $(y - y_j^E)^T AA^T(y - y_j^E) = \|x - x_j^E\|_2^2$, so it also minimizes the Euclidean norm of $x - x_j^E$. This suggests that CGNE means "CG minimizing the Norm of the Error." Unfortunately, CGNE is sometimes interpreted as "CG Normal Equations."

Instead of Craig's method, another application of CG to solving $Ax = b$ or minimizing $\|Ax - b\|_2$ was given by Hestenes and Stiefel in [11, eq. (10.2)]. It was called CGNR in [8, Chap. 7] and CGLS in [20, sect. 7.1] and uses CG to solve the normal equations $A^TAx = A^Tb$. With full column rank A, as is the case here, the approximate solution x_j^R ($\cdot^R$ from CGNR) at step j is the unique vector that satisfies (see (2.5))

$$x_j^R = V_j z_j^R, \quad 0 = V_j^T A^T(b - AV_j z_j^R), \quad \text{Range}(V_j) = \mathcal{K}_j(A^TA, A^Tb). \tag{2.7}$$

Whether A has full column rank or not, z_j^R minimizes $\|b - AV_j z_j\|_2^2$ since (2.7) gives the normal equations for this minimization, so x_j^R minimizes $\|b - Ax_j\|_2^2$ over all $x_j \in \text{Range}(V_j)$, leading to least squares solutions to $Ax \approx b$. Thus CGNR probably means "CG minimizing the Norm of the Residual"; see, e.g., [8, pp. 105–106]. An implementation based GKB [7] is LSQR in [20].

3. Related work. An understanding of the special properties of, and theoretical relationships among, algorithms for skew-symmetric matrix problems $Ax = b$ developed over the years. As noted in [4], early on Huang, Wathen, and Li [12] gave an algorithm computing only the even minimum residual iterates x_{2j}^M (see (2.4)). Idema and Vuik [13] also applied the minimum residual condition (2.4) and observed that the algorithm in [12] is equivalent to CGNR (see (2.7)), i.e., that $x_{2j}^M = x_j^R$. Jiang [14] again applied the minimum residual condition, while Gu and Qian [10] imposed the Galerkin condition (see (2.2)). Greif and Varah [9] gave algorithms based on the Lanczos process for both the minimum residual and the Galerkin conditions, showing that in theory the odd iterates x_{2j+1}^G in (2.2) do not exist (they would require solutions of incompatible singular systems for y_{2j+1}^G), while the even iterates x_{2j}^G are equivalent to CGNE iterates x_j^E (see (2.6)). They also showed that in theory, in (2.4), $x_{2j+1}^M = x_{2j}^M$. The Lanczos process has been applied to other structured matrices; for example, in [2] the Lanczos process is tailored to take advantage of symplectic structure.

Eisenstat unified and extended the results of [9] by using his elegant Lemma 1 in [4] to give algorithm independent proofs that, among other things, if $A^T = -A$, then in theory $x_{2j}^G = x_j^E$ in (2.2) and (2.6), and $x_{2j+1}^M = x_{2j}^M = x_j^R$ in (2.4) and (2.7), and in each case there is no advantage in using the first approach over the second.

These results assumed mathematically exact relationships. In the following we will show that many of the equivalences can also occur in finite precision computation.

4. Lanczos tridiagonalization of skew-symmetric matrices. Given an integer $k>0$ and a unit norm vector $w_1 \in \mathbb{R}^n$, the k-step Arnoldi iteration [1] for general $A \in \mathbb{R}^{n\times n}$ generates a sequence of orthonormal vectors $\{w_j\}$ such that

$$AW_k = W_{k+1}H_{k+1,k} = W_kH_k + w_{k+1}\gamma_{k+1}e_k^T, \quad W_k^TW_k = I_k, \quad W_k^TAW_k = H_k, \tag{4.1}$$

where $W_k \triangleq [w_1, \ldots, w_k]$, $H_{k+1,k}$ is a $(k+1)\times k$ upper Hessenberg matrix, and $\text{Range}(W_k) = \mathcal{K}_k(A, w_1)$. These relations may be used in (2.2). If $A = -A^T \in \mathbb{R}^{n\times n}$, then $H_k^T = W_k^TA^TW_k = -H_k$, giving

$$H_{k+1,k} = \begin{bmatrix} 0 & -\gamma_2 & & & \\ \gamma_2 & 0 & -\gamma_3 & & \\ & \cdot & \cdot & \cdot & \\ & & \cdot & 0 & -\gamma_k \\ & & & \gamma_k & 0 \\ & & & & \gamma_{k+1} \end{bmatrix} \equiv \begin{bmatrix} H_k \\ \gamma_{k+1}e_k^T \end{bmatrix}. \tag{4.2}$$

This leads to Algorithm 1, a short recurrence Lanczos process computing W_k and H_k.

Algorithm 1 The Lanczos process [15] for skew-symmetric $A \in \mathbb{R}^{n\times n}$

Given $w_1 \in \mathbb{R}^n$ with $\|w_1\|_2 = 1$ compute $w_2\gamma_2 := Aw_1$
for $k = 2$ step 1 until convergence **do**
  $w_{k+1}\gamma_{k+1} := Aw_k + w_{k-1}\gamma_k$
end for

Note that in Algorithm 1 each γ_j results simply from the normalization of its w_j.

There are at most n orthonormal vectors, so in theory this process must stop in $k_0 \le n$ steps. And since H_{k_0} is skew-symmetric in $AW_{k_0} = W_{k_0}H_{k_0}$,

nonsingular $A = -A^T \Rightarrow$ Algorithm 1 stops in an even number of $k_0 \triangleq 2j_0$ steps. (4.3)

Because there is no diagonal element in (4.2), Lanczos tridiagonalization for skew-symmetric matrices is likely to be marginally more well behaved numerically than the symmetric Lanczos process, but cancellation will still eventually lead to loss of orthogonality among the w_k vectors, and the process is unlikely to stop in n steps. It is the special form of the tridiagonal that leads to the following equivalences.

5. Golub–Kahan bidiagonalization. Often the best way to understand, compare, and implement algorithms such as CGNE and CGNR is via Golub–Kahan bidiagonalization (GKB) [7] in the form of "Bidiag 1" in [20, sect. 3], which we now state as Algorithm 2. Again note that every scalar results from a normalization.

Algorithm 2 Golub–Kahan bidiagonalization [7] for general $A \in \mathbb{R}^{m\times n}$

Given $u_1 \in \mathbb{R}^m$ with $\|u_1\|_2 = 1$ compute $v_1\alpha_1 := A^T u_1$
for $j = 1$ step 1 until convergence **do**
  $u_{j+1}\beta_{j+1} := Av_j - u_j\alpha_j$
  $v_{j+1}\alpha_{j+1} := A^T u_{j+1} - v_j\beta_{j+1}$
end for

This gives lower bidiagonal $B_{j,j-1}$ (denoted by B_{j-1} in [20]), where after $j-1$ steps with $U_j \triangleq [u_1, \dots, u_j]$ and $V_j \triangleq [v_1, \dots, v_j]$ (see [20, sect. 3]),

$$(5.1)\quad AV_{j-1} = U_j B_{j,j-1}, \quad A^T U_j = V_j B_j^T, \quad B_{j,j-1} \triangleq \begin{bmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & \\ & \beta_3 & \ddots & \\ & & \ddots & \alpha_{j-1} \\ & & & \beta_j \end{bmatrix} \equiv \begin{bmatrix} B_{j-1} \\ \beta_j e_{j-1}^T \end{bmatrix},$$

$$(5.2)\quad U_j^T U_j = V_j^T V_j = I_j, \quad \text{Range}(U_j) = \mathcal{K}_j(AA^T, u_1), \quad \text{Range}(V_j) = \mathcal{K}_j(A^T A, A^T u_1),$$

where U_j and V_j can be used for CGNE in (2.6) and CGNR in (2.7), respectively.

6. Equivalence of Lanczos tridiagonalization and Golub–Kahan bidiagonalization for skew-symmetric matrices. When $A^T = -A$, we now show that one step of Algorithm 2 is equivalent to two steps of Algorithm 1 if $u_1 = w_1$.

First make the following substitutions in Algorithm 2:

$$(6.1)\qquad A^T \to -A, \quad u_j \to \tilde{w}_{2j-1}, \quad v_j \to \tilde{w}_{2j}, \quad \alpha_j \to -\tilde{\gamma}_{2j}, \quad \beta_j \to \tilde{\gamma}_{2j-1},$$
$$\text{giving} \quad [u_1, v_1, u_2, v_2, \dots, u_j, v_j] \to \widetilde{W}_{2j} = [\tilde{w}_1, \tilde{w}_2, \tilde{w}_3, \dots, \tilde{w}_{2j-1}, \tilde{w}_{2j}].$$

Note that $\alpha_j \to -\tilde{\gamma}_{2j}$ is a substitution, not an equality. Both α_j and $\tilde{\gamma}_{2j}$ are positive. The body of the loop in Algorithm 2 then becomes $\tilde{w}_{2j+1}\tilde{\gamma}_{2j+1} := A\tilde{w}_{2j} + \tilde{w}_{2j-1}\tilde{\gamma}_{2j}$ and $\tilde{w}_{2j+2}\tilde{\gamma}_{2j+2} := A\tilde{w}_{2j+1} + \tilde{w}_{2j}\tilde{\gamma}_{2j+1}$, or replacing $2j$ by k, $\tilde{w}_{k+1}\tilde{\gamma}_{k+1} := A\tilde{w}_k + \tilde{w}_{k-1}\tilde{\gamma}_k$ and $\tilde{w}_{k+2}\tilde{\gamma}_{k+2} := A\tilde{w}_{k+1} + \tilde{w}_k\tilde{\gamma}_{k+1}$, i.e., two successive steps, so Algorithm 2 becomes the following algorithm.

Algorithm 3 If $A^T = -A$, Golub–Kahan bidiagonalization $\equiv$ Lanczos process

Given $\tilde{w}_1 \in \mathbb{R}^n$ with $\|\tilde{w}_1\|_2 = 1$ compute $\tilde{w}_2\tilde{\gamma}_2 := A\tilde{w}_1$ (assume nonsingular A)
for $k = 2$ step 1 until convergence **do**
  $\tilde{w}_{k+1}\tilde{\gamma}_{k+1} := A\tilde{w}_k + \tilde{w}_{k-1}\tilde{\gamma}_k$ (ideally this stops with $\tilde{w}_{k_0+1}\tilde{\gamma}_{k_0+1} = 0$; see (4.3))
end for

Thus by expanding one step of GKB into two successive steps, the resulting Algorithm 3 has the same form as Algorithm 1, and if $\tilde{w}_1 = u_1 = w_1$, then Algorithm 3

develops *exactly* as Algorithm 1. Therefore each GKB step in Algorithm 2 with $A^T = -A$ (so that $m = n$) can be made *identical* to two Lanczos process steps in Algorithm 1, and the computational cost and accuracy of this special version of Algorithm 2 must then be identical to that of Algorithm 1.

Algorithm 1 terminates in $k_0 = 2j_0$ steps for nonsingular A (see (4.3)), so with $u_1 = w_1$ it follows from (6.1) and (5.2) that for U_j and V_j in (6.1), for $j = 1\!:\!j_0$,

$$\begin{aligned} (6.2)\quad & \mathrm{Range}(U_j) \equiv \mathrm{Range}([w_1, w_3, \ldots, w_{2j-1}]) = \mathcal{K}_j(AA^T, w_1) = \mathcal{K}_j(A^2, w_1), \\ & \mathrm{Range}(V_j) \equiv \mathrm{Range}([w_2, w_4, \ldots, w_{2j}]) = \mathcal{K}_j(A^TA, A^Tw_1) = \mathcal{K}_j(A^2, Aw_1), \\ & U_j^TV_j = 0, \quad \mathrm{Range}([U_j, V_j]) = \mathcal{K}_{2j}(A, u_1) = \mathcal{K}_{2j}(A, w_1), \quad AV_{j_0} = U_{j_0}B_{j_0}, \end{aligned}$$

where $U_j^TV_j = 0$ follows from $W_{2j}^TW_{2j} = I_{2j}$. Thus the odd indexed vectors from the Lanczos process are the u_j vectors from GKB, while the even indexed vectors from the Lanczos process are the v_j vectors from GKB, so interestingly, the two sets of GKB vectors are orthogonal. Here (6.2) parallels the more theoretical derivation by Eisenstat [4, Lem. 1] of the following: If $A^T = -A$ and $Ax = b$, then (with $b = w_1\beta_1 = u_1\beta_1$)

$$\begin{aligned} (6.3)\quad & \text{for } j = 1\!:\!j_0, \quad \mathcal{K}_{2j}(A, b) = \mathcal{K}_j(A^2, b) \cup \mathcal{K}_j(A^2, Ab), \quad \mathcal{K}_j(A^2, b) \perp \mathcal{K}_j(A^2, Ab); \\ & 0 = V_{j_0}^Tu_1\beta_1 = V_{j_0}^Tb = V_{j_0}^TAx = -V_{j_0}^TA^Tx = -B_{j_0}^TU_{j_0}^Tx, \quad \text{so} \quad x \perp \mathcal{K}_{j_0}(A^2, b). \end{aligned}$$

Another approach is to note that if $B_{j_0}y = e_1\beta_1$, then $AV_{j_0}y = U_{j_0}e_1\beta_1 = b$, so

$$x = V_{j_0}y \in \mathcal{K}_{j_0}(A^2, Ab) \perp \mathrm{Range}(U_{j_0}) = \mathcal{K}_{j_0}(A^2, b).$$

Now $k = 2j$ steps of Lanczos tridiagonalization are equivalent to j steps of GKB, and $\mathrm{Range}(W_k) = \mathrm{Range}([U_j, V_j])$. But in theory $k = 2j \le n$, so $j \le \frac{n}{2}$ for nonsingular A, and GKB must in theory (i.e., in the absence of roundoff error) stop in at most $\frac{n}{2}$ steps. Another way of looking at this is that the nonzero eigenvalues of $A = -A^T$ come in imaginary pairs, $\pm\sigma_i\sqrt{-1}$, and so each nonzero singular value σ_i of A occurs twice, so that GKB must in theory stop in at most $\frac{n}{2}$ steps for nonsingular A.

The Lanczos process–GKB equivalence suggests that with skew-symmetric matrices, for any computation involving the Lanczos process, there is a *numerically* equivalent computation involving GKB. To support this, and show the required reductions, in each of the next three sections we illustrate a solution of equations method for $Ax = b$ based on the Lanczos process, and its GKB equivalent, so we take $w_1\beta_1 = u_1\beta_1 = b$ in Algorithms 1 and 2 to relate (4.1)–(6.3).

7. Lanczos–Galerkin and CGNE methods for skew-symmetric matrices. Let $A = -A^T \in \mathbb{R}^{n\times n}$ and $b \in \mathbb{R}^n$, with A nonsingular so that n is even. Assume (4.1)–(4.2) hold with $\beta_1 := \|b\|_2$, $w_1 := b/\beta_1$. At step k of the Lanczos process the Galerkin condition for solving (2.1) is with $y_k^G \triangleq (\eta_1, \ldots, \eta_k)^T$ (see (2.2)),

$$(7.1)\quad x_k^G = W_ky_k^G, \quad W_k^Tb = e_1\beta_1 = W_k^TAW_ky_k^G = H_ky_k^G, \quad \mathrm{Range}(W_k) = \mathcal{K}_k(A, b).$$

This was analyzed and implemented in [9]. Here we show how the same process can be more briefly presented by using the zero structure in (4.2) and taking $k = 2j$. The $k = 4$ case will make the development easily understandable. From (7.1) we need to solve $H_4y_4^G = e_1\beta_1$; see (7.2). Note that the 4th row of (7.2) gives $\eta_3 = 0$, so the 2nd row gives $\eta_1 = 0$, and these two elements and their columns can then be deleted to

give the middle matrix equation in (7.2). With (6.1), deleting the 2nd and 4th rows gives the final matrix equation where the final matrix is B_2 in (5.1):

$$(7.2)\quad \begin{bmatrix} 0 & -\gamma_2 & & \\ \gamma_2 & 0 & -\gamma_3 & \\ & \gamma_3 & 0 & -\gamma_4 \\ & & \gamma_4 & 0 \end{bmatrix}\begin{bmatrix}\eta_1\\ \eta_2\\ \eta_3\\ \eta_4\end{bmatrix}=\begin{bmatrix}-\gamma_2 & \\ 0 & \\ \gamma_3 & -\gamma_4 \\ & 0\end{bmatrix}\begin{bmatrix}\eta_2\\ \eta_4\end{bmatrix}=\begin{bmatrix}\beta_1\\ 0\\ 0\\ 0\end{bmatrix}\rightarrow\begin{bmatrix}\alpha_1 & \\ \beta_2 & \alpha_2\end{bmatrix}\begin{bmatrix}\eta_2\\ \eta_4\end{bmatrix}=\begin{bmatrix}\beta_1\\ 0\end{bmatrix}.$$

In general, the last row of $H_{2j}y_{2j}^G = e_1\beta_1$ gives $\gamma_{2j}\eta_{2j-1} = 0$, and the remaining even rows of H_{2j} show that $\eta_{2i-1} = 0$, $i = 1, 2, \ldots, j$. Deleting the columns corresponding to zero elements and the resulting zero rows means H_{2j} becomes B_j in (5.1). Writing $t_j^G \triangleq (\eta_2, \eta_4, \ldots, \eta_{2j})^T$ leads to the following Lanczos–Galerkin approximations:

$$(7.3)\qquad \text{Solve}\ \ B_j t_j^G = e_1\beta_1\ \ \text{and form}\ \ x_{2j}^G = W_{2j}y_{2j}^G = V_j t_j^G = x_{2j-2}^G + v_j\eta_{2j}.$$

Thus only the w_{2i} contribute to $x_{2j}^G = W_{2j}y_{2j}^G$, meaning that $x_{2i+1}^G = x_{2i}^G$ and the residuals $r_{2i+1}^G = r_{2i}^G$ for $i = 1 : j$. But the vectors $u_i = w_{2i-1}$ contribute to the residuals since from (4.1) and $H_{2j}y_{2j}^G = e_1\beta_1$,

$$(7.4)\quad r_{2j}^G \triangleq b - Ax_{2j}^G = w_1\beta_1 - AW_{2j}y_{2j}^G = w_1\beta_1 - W_{2j+1}H_{2j+1,2j}y_{2j}^G = -w_{2j+1}\beta_{j+1}\eta_{2j}.$$

Now to obtain the CGNE approximation x_j^E we take $u_1 = b/\beta_1$, $\beta_1 = \|b\|_2$ (so that $u_1 = w_1$ in (7.1)) in GKB Algorithm 2, together with the CGNE condition in (2.6). Then (5.1) and (6.1) give the GKB-CGNE approximations,

$$(7.5)\quad \begin{aligned} &0 = U_j^T(b - AA^TU_jz_j^E) = e_1\beta_1 - B_jV_j^TV_jB_j^Tz_j^E = e_1\beta_1 - B_jt_j^E, \quad t_j^E \triangleq B_j^Tz_j^E,\\ &x_j^E = A^TU_jz_j^E = V_jB_j^Tz_j^E = V_jt_j^E. \end{aligned}$$

With $t_j^E \triangleq (\tau_1, \tau_2, \ldots, \tau_j)^T$, $B_jt_j^E = e_1\beta_1$ in (7.5) shows that

$$(7.6)\qquad \tau_1 = \frac{\beta_1}{\alpha_1};\quad \tau_i = -\tau_{i-1}\frac{\beta_i}{\alpha_i},\ \ i = 2 : j;\quad x_j^E = \sum_{i=1}^{j} v_i\tau_i = x_{j-1}^E + v_j\tau_j.$$

We see that $t_j^E = t_j^G$ in (7.3) so that $x_j^E = x_{2j}^G$ and $r_j^E \triangleq b - Ax_j^E = r_{2j}^G$. Thus for skew-symmetric A, not only are the Lanczos–Galerkin and CGNE approximations mathematically the same (see [9] and section 3), but they are computationally the same if the above obvious implementations are used.

Greif and Varah [9] showed that each double step of the Lanczos–Galerkin algorithm is equivalent to a single step of CGNE for skew-symmetric A by scaling the Lanczos–Galerkin vectors to produce the standard variant of CGNE. In Algorithm 4 we give the CGNE version that comes from (7.6) with GKB in Algorithm 2.

The mathematical equivalence of Algorithm 4 with the standard CG variant was shown in [9]; see, e.g., [9, Alg. 2, p. 592]. Algorithm 4 requires one less inner product but two more vector scalings per step. If needed, $r_j^E = r_{2j}^G = -u_{j+1}\beta_{j+1}\tau_j$ follows from (7.4), (6.1), and $\eta_{2j} = \tau_j$.

We repeat the observation from section 6 that theoretically Algorithm 1 converges in at most n steps, so that Algorithm 2 and CGNE in Algorithm 4 must converge in at most $n/2$ steps, but of course the number of matrix-vector products is the same. The above observation also follows since CGNE is mathematically equivalent to applying CG to $AA^Ty = b$ and forming $x = A^Ty$. But $AA^T = -A^2$ has at most $n/2$ distinct eigenvalues, giving the ideal convergence of CGNE in at most $n/2$ steps.

Algorithm 4 Golub–Kahan bidiagonalization [7] version of CGNE for $Ax = b$ with $A^T = -A$

Given b compute $u_1\beta_1 := b$, $v_1\alpha_1 := -Au_1$, $\tau_1 := \beta_1/\alpha_1$, $x_1 := v_1\tau_1$
for $j = 1$ step 1 until convergence **do**
  $u_{j+1}\beta_{j+1} := Av_j - u_j\alpha_j$
  $v_{j+1}\alpha_{j+1} := -Au_{j+1} - v_j\beta_{j+1}$
  $\tau_{j+1} := -\tau_j\beta_{j+1}/\alpha_{j+1}$
  $x_{j+1} := x_j + v_{j+1}\tau_{j+1}$
end for

8. Lanczos minimum residual and CGNR methods for skew-symmetric matrices. To find x_k^M in (2.4) using the Lanczos process we take $w_1\beta_1 := b$ in (4.1) to give in the kth step

$$\begin{aligned} &\|b - AW_k y_k\|_2 = \|w_1\beta_1 - W_{k+1}H_{k+1,k}y_k\|_2 = \|e_1\beta_1 - H_{k+1,k}y_k\|_2, \\ &x_k^M = W_k y_k^M, \qquad y_k^M = \arg\min_{y_k} \|e_1\beta_1 - H_{k+1,k}y_k\|_2. \end{aligned} \tag{8.1}$$

This was carefully analyzed in [9]. But similarly to section 7, a solution method is more easily developed by using the zero structure in (4.2) and taking even $k = 2j$. Define $y_k \triangleq (\eta_1, \ldots, \eta_k)^T$. The expression $e_1\beta_1 - H_{k+1,k}y_k$ in (8.1) separates into two unconnected sets of rows, the odd rows involving the η_{2i}, and the even rows involving the η_{2i-1}, from which we will see that the $\eta_{2i-1} = 0$ for the minimum in (8.1). The $k = 4$ case makes the development easily understandable,

$$H_{5,4}y_4 - e_1\beta_1 = \begin{bmatrix} 0 & -\gamma_2 & & \\ \gamma_2 & 0 & -\gamma_3 & \\ & \gamma_3 & 0 & -\gamma_4 \\ & & \gamma_4 & 0 \\ & & & \gamma_5 \end{bmatrix} \begin{bmatrix} \eta_1 \\ \eta_2 \\ \eta_3 \\ \eta_4 \end{bmatrix} - \begin{bmatrix} \beta_1 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} -\gamma_2\eta_2 - \beta_1 \\ \gamma_2\eta_1 - \gamma_3\eta_3 \\ \gamma_3\eta_2 - \gamma_4\eta_4 \\ \gamma_4\eta_3 \\ \gamma_5\eta_4 \end{bmatrix}.$$

To minimize the Euclidean norm of this expression we can set the odd rows to zero without affecting the even ones. So with (6.1) and (5.1) we minimize

$$\left\| \begin{bmatrix} -\gamma_2 & \\ \gamma_3 & -\gamma_4 \\ & \gamma_5 \end{bmatrix} \begin{bmatrix} \eta_2 \\ \eta_4 \end{bmatrix} - \begin{bmatrix} \beta_1 \\ 0 \\ 0 \end{bmatrix} \right\|_2 = \left\| \begin{bmatrix} \alpha_1 & \\ \beta_2 & \alpha_2 \\ & \beta_3 \end{bmatrix} \begin{bmatrix} \eta_2 \\ \eta_4 \end{bmatrix} - \begin{bmatrix} \beta_1 \\ 0 \\ 0 \end{bmatrix} \right\|_2 = \left\| B_{3,2} \begin{bmatrix} \eta_2 \\ \eta_4 \end{bmatrix} - e_1\beta_1 \right\|_2.$$

In general, because of the structure of $H_{k+1,k}$ the odd numbered elements of $H_{k+1,k}y_k$ involve only even-indexed η_i, while the even numbered elements of $H_{k+1,k}y_k$ involve only odd-indexed η_i. So, as above, we can set the odd-indexed η_i to zero and, with $z_j \triangleq (\eta_2, \eta_4, \ldots, \eta_{2j})^T \equiv (\zeta_1, \zeta_2, \ldots, \zeta_j)^T$ and (6.1), solve

$$z_j^M = \arg\min_{z_j} \|B_{j+1,j}z_j - e_1\beta_1\|_2, \quad x_{2j}^M = \sum_{i=1}^{j} w_{2i}\eta_{2i}^M = \sum_{i=1}^{j} v_i\zeta_i^M = V_j z_j^M. \tag{8.2}$$

Again only the w_{2i} contribute to $x_{2j}^M = W_{2j}y_{2j}^M$, meaning that $x_{2j+1}^M = x_{2j}^M$ and the residuals $r_{2j+1}^M = r_{2j}^M$. With $k = 2j$ the residuals are then (see (4.1))

$$\begin{aligned} &r_{2j}^M \triangleq b - Ax_{2j}^M = w_1\beta_1 - AW_{2j}y_{2j}^M = W_{2j+1}(e_1\beta_1 - H_{2j+1,2j}y_{2j}^M), \\ &\|r_{2j}^M\|_2 = \|e_1\beta_1 - H_{2j+1,2j}y_{2j}^M\|_2 = \|B_{j+1,j}z_j^M - e_1\beta_1\|_2^2. \end{aligned} \tag{8.3}$$

Similarly to section 7, we can obtain the CGNR solution by taking $u_1 = w_1 = b/\beta_1,\ \beta_1 = \|b\|_2$ in GKB Algorithm 2 together with the CGNR condition in (2.7). This gives with (5.1) and (6.1) the CGNR solution $x_j^R = V_j z_j^R$ at step j via GKB, where it will be seen that $z_j^R \triangleq (\zeta_1^R, \zeta_2^R, \ldots, \zeta_j^R)^T = z_j^M$ in (8.2),

$$
\begin{aligned}
&0 = V_j^T A^T (b - AV_j z_j^R) = B_{j+1,j}^T U_{j+1}^T (b - U_{j+1} B_{j+1,j} z_j^R) = B_{j+1,j}^T (e_1\beta_1 - B_{j+1,j} z_j^R),\\
&\qquad z_j^R = \arg\min_{z_j} \|e_1\beta_1 - B_{j+1,j} z_j\|_2 = z_j^M, \qquad x_j^R = V_j z_j^R = V_j z_j^M = x_{2j}^M,\\
&\qquad r_j^R \triangleq b - Ax_j^R = u_1\beta_1 - AV_j z_j^R = u_1\beta_1 - U_{j+1} B_{j+1,j} z_j^R = r_{2j}^M,
\end{aligned}
\tag{8.4}
$$

and again we see that for $A^T = -A$ the Lanczos minimum residual approach is mathematically equivalent to the CGNE approach; see section 3. But if the above algorithms are used, we now see that they are numerically equivalent.

The method indicated by (8.4) is implemented as Algorithm LSQR in [20, sect. 4], [21], so there is no reason to repeat it here. The standard CGNE and CGNR algorithms are given in [8, p. 105]. The only thing to remember is to use $-A$ when A^T is required and to note that ideally the methods would stop in no more than $n/2$ steps.

9. Minimizing the normal equation residual for skew-symmetric matrices. If A is $m \times n$ with rank n, then the least squares solution to $Ax \approx b$ satisfies $A^T r = 0$, $r \triangleq b - Ax$, and it can make sense, even for general $Ax = b$, to choose x_k from a Krylov subspace to minimize $\|A^T r_k\|_2 = \|A^T(b - Ax_k)\|_2$; see Fong and Saunders [5], who developed the algorithm LSMR (Least Squares via MINRES on the normal equations) based on GKB to solve such problems. As far as we know, for skew-symmetric problems no one has published an algorithm based on Lanczos tridiagonalization to solve such problems, so we will quickly indicate such an approach and show how, as expected, it can be reduced to LSMR applied to skew-symmetric A.

For $A \in \mathbb{R}^{n\times n}$, using W_k in (4.1) would give ($\cdot^N$ for "Normal equations residual")

$$
w_1\beta_1 := b, \quad y_k^N := \arg\min_{y_k} \|A^T(b - AW_k y_k)\|_2, \quad x_k^N := W_k y_k^N. \tag{9.1}
$$

When $A^T = -A$, the Lanczos process (4.1) and (4.2) leads to the minimization of

$$
\begin{aligned}
\|A^T(b - AW_k y_k)\|_2 &= \| - A(w_1\beta_1 - W_{k+1} H_{k+1,k} y_k)\|_2\\
&= \| - w_2\gamma_2\beta_1 + W_{k+2} H_{k+2,k+1} H_{k+1,k} y_k\|_2\\
&= \|e_2\gamma_2\beta_1 - H_{k+2,k+1} H_{k+1,k} y_k\|_2.
\end{aligned}
\tag{9.2}
$$

But because of the structure of $H_{k+1,k}$, if $y_k \triangleq (\eta_1, \ldots, \eta_k)^T$, the odd numbered elements of $H_{k+1,k} y_k$ involve only even-indexed η_i, while the the even numbered elements of $H_{k+1,k} y_k$ involve only odd-indexed η_i. For the same reason the odd numbered elements of $H_{k+2,k+1} H_{k+1,k} y_k$ involve only odd-indexed η_i, while the even numbered elements of $H_{k+2,k+1} H_{k+1,k} y_k$ involve only even-indexed η_i. This breaks the minimization of (9.2) into two separate minimizations, one over the odd-indexed η_i and the other over the even. But the only nonzero element arising from $A^T b$ is $e_2\gamma_2\beta_1$, which is related to even-indexed η_i, so the odd-indexed η_i must be set to zero.

Setting the $\eta_{2i-1} = 0$ in (9.2) leads to the even-indexed elements of $H_{k+1,k} y_k$ being zero. Removing these and the related columns, and then zero rows, of $H_{k+2,k+1}$ leads with (6.1) to, for example, for $k = 4$ or $k = 5$,

$$\left\| \begin{bmatrix} \gamma_2\beta_1 \\ 0 \\ 0 \end{bmatrix} - \begin{bmatrix} \gamma_2 & -\gamma_3 & \\ & \gamma_4 & -\gamma_5 \\ & & \gamma_6 \end{bmatrix} \begin{bmatrix} -\gamma_2 & \\ \gamma_3 & -\gamma_4 \\ & \gamma_5 \end{bmatrix} \begin{bmatrix} \eta_2 \\ \eta_4 \end{bmatrix} \right\|_2 = \left\| -e_1\alpha_1\beta_1 + B_{j+1}^T B_{j+1,j} \begin{bmatrix} \eta_2 \\ \eta_4 \end{bmatrix} \right\|_2.$$

In general, with $k = 2j$, $z_j \triangleq (\zeta_1, \ldots, \zeta_j)^T \equiv (\eta_2, \ldots, \eta_{2j})^T$, and (6.1), (9.1) becomes

$$z_j^N := \arg\min_{z_j} \|e_1\alpha_1\beta_1 - B_{j+1}^T B_{j+1,j} z_j\|_2, \quad x_{2j}^N := \sum_{i=1}^{j} w_{2i}\eta_{2i}^N = \sum_{i=1}^{j} v_i\zeta_i^N = V_j z_j^N. \tag{9.3}$$

The GKB approach to minimizing $\|A^T r_j\|_2 = \|A^T(b - Ax_j)\|_2$ is, with $\cdot^L$ denoting LSMR (see (5.1) and (5.2)),

$$u_1\beta_1 := b, \quad z_j^L := \arg\min_{z_j} \|A^T(b - AV_j z_j)\|_2, \quad x_j^L := V_j z_j^L, \tag{9.4}$$

$$\begin{aligned} \|A^T(b - AV_j z_j)\|_2 &= \|A^T(u_1\beta_1 - U_{j+1}B_{j+1,j} z_j)\|_2 \\ &= \|v_1\alpha_1\beta_1 - V_{j+1}B_{j+1}^T B_{j+1,j} z_j\|_2 \\ &= \|e_1\alpha_1\beta_1 - B_{j+1}^T B_{j+1,j} z_j\|_2 = \|B_{j+1}^T(e_1\beta_1 - B_{j+1,j} z_j)\|_2 \end{aligned} \tag{9.5}$$

(cf. (8.4) for LSQR), so from (9.3) and (6.2), $z_j^L = z_j^N$, $x_j^L = x_{2j}^N \in \text{Range}(V_j) = \mathcal{K}_j(A^TA, A^Tb)$, and using the identical computation to solve these identical problems for z_j^L and z_j^N will lead to identical numerical approximations.

For this section, to minimize $\|A^T r_j\|_2$ at each step we need simply apply the GKB Algorithm LSMR in [5], with the same proviso as was given at the end of section 8. Fong and Saunders [5] give an ingenious way of minimizing (9.5) and updating x_j^L in (9.4) efficiently in a numerically reliable way. The algorithm for the updates, which includes careful implementations of Givens rotations for the QR factorizations, is laid out in steps 4, 5, and 6 of [5, sect. 2.8].

10. Preconditioning. Using a preconditioner can significantly accelerate convergence for $Ax = b$, so we briefly point out a few observations from this study.

A challenge in the design of preconditioners for skew-symmetric systems is that in general, skew-symmetric preconditioners do not easily accommodate the desired property of maintaining skew-symmetry of the preconditioned matrix. In other words, given a skew-symmetric matrix A, a skew-symmetric matrix M will typically yield a preconditioned matrix $M^{-1}A$ that is not skew-symmetric or "skew-symmetrizable" (and possibly not diagonalizable). On the other hand, an SPD preconditioner can maintain skew-symmetry, in the sense that for $M = FF^T$ the matrix $F^{-1}AF^{-T}$ is skew-symmetric if A is. See [9] for an illustration of an incomplete factorization approach for skew-symmetric matrices that computes a positive definite preconditioner.

For the present study of Krylov subspace methods for skew-symmetric matrices, a solution method based on the Lanczos process, and one based on the corresponding GKB method, lead to numerically identical results, so a preconditioner that maintains skew-symmetry will indeed give identical results for both approaches.

On the other hand, a difference between the two approaches is that a preconditioner that does *not* maintain skew-symmetry will destroy the desirable properties of a method based on the Lanczos process in Algorithm 1. The short recurrence of the Lanczos process in Algorithm 1 produces orthogonality of w_{k+1} against $w_1, w_2, \ldots, w_{k-2}$ only because A is skew-symmetric, and this orthogonality is the basis for the excellent qualities of the process. Note that even $w_1^T w_2\gamma_2 = w_1^T A w_1$ cannot be expected to be zero unless A is skew-symmetric. Nor does Algorithm 1 give the same

results as Algorithm 2 when A is not skew-symmetric. For example, Algorithm 1 uses A alone, while Algorithm 2 uses A and A^T.

A preconditioner that does not maintain skew-symmetry *can* be applied effectively with a GKB based method but can no longer in theory be expected to converge in $n/2$ steps—it can double the effective dimension of the problem. It may double the number of distinct singular values and even generate an adverse effect in terms of singular value grouping. Nevertheless, practical preconditioners are expected to generate a strong clustering effect, with the iteration count going significantly below $n/2$, and a good preconditioner that does not maintain skew-symmetry might still be superior to a less effective one that maintains skew-symmetry.

11. A numerical example. Numerical experiments for the examples that we have explored support the equivalence of Lanczos tridiagonalization and GKB for skew-symmetric matrices. As expected, orthogonality is lost so that none of $U_k^T U_k = I_k$, $V_k^T V_k = I_k$, or $U_k^T V_k = 0$ need hold—the supposedly zero elements may be large, slowing convergence significantly.

We illustrate our findings on an artificially generated example, which could be loosely described as a simple finite difference discretization of a constant convective term on the unit cube in three dimensions, $(0,1) \times (0,1) \times (0,1)$. For a given positive integer n and a real scalar α, let us denote by $T_n(\alpha)$ the $n \times n$ skew-symmetric tridiagonal matrix

$$T_n(\alpha) = \begin{bmatrix} 0 & \alpha & & & \\ -\alpha & 0 & \alpha & & \\ & -\alpha & 0 & \cdot & \\ & & \cdot & \cdot & \alpha \\ & & & -\alpha & 0 \end{bmatrix} \in \mathbb{R}^{n \times n}.$$

The matrix A for which we solve the linear system $Ax = b$ is defined by

$$A = I_n \otimes I_n \otimes T_n(\beta) + I_n \otimes T_n(\gamma) \otimes I_n + T_n(\delta) \otimes I_n \otimes I_n,$$

where $\otimes$ denotes the Kronecker product and (β, γ, δ) is a triplet of preselected scalars. Evidently A is of dimensions $n^3 \times n^3$. The right-hand side is randomly generated.

We show results with $n = 16$, i.e., a matrix A of dimensions 4096×4096, and the triplet $(\beta, \gamma, \delta) = (0.4, 0.5, 0.6)$. We do not expect fast convergence in this example because preconditioning is not incorporated; our goal here is to observe and confirm the theoretical properties of the various algorithms that we have studied.

A very useful summary of the theoretical monotonic or otherwise behavior of the Euclidean norms of various vectors from Conjugate Gradients in [11] and MINRES in [19] for symmetric systems, and LSQR and LSMR for least squares problems, is given by Fong and Saunders in [6, Tabs. 5.1 and 5.2]. In particular, it is stated that for LSQR and LSMR applied to linear least squares problems, $\|x - x_k\|_2$, $\|r - r_k\|_2$, and $\|r_k\|_2$ are monotonically decreasing, while $\|A^T r_k\|_2$ is monotonically decreasing for LSMR but not LSQR. Here x and r are the optimal solution and residual ($r = 0$ in our case). Our computations with skew-symmetric A support this for LSQR (i.e., CGNR) and LSMR.

We also encountered nonmonotonic residual norms for CGNE, in line with the theory for CG in [6]. From now on we will refer to LSQR rather than CGNR because LSQR is the implementation of (2.7) that we use. To further distinguish the colored plots, we order the labels in the figures from worst plot (top) to best (bottom).

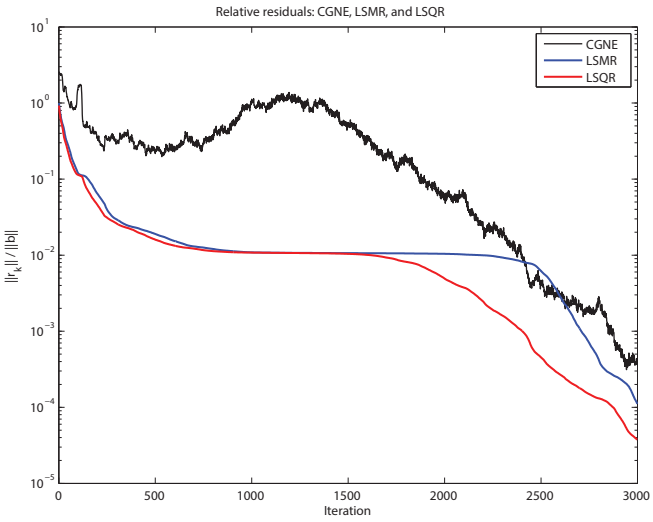

FIG. 11.1. *Recorded relative residuals,* $\|r_k\|/\|b\|$*, for CGNE, LSMR, and LSQR, applied to a skew-symmetric numerical example of dimensions* 4096×4096*.*

Figure 11.1 plots the relative residuals for CGNE, LSMR, and LSQR. CGNE features a nonmonotonic curve. LSQR minimizes the norm of the residual, and so its associated curve lies below the other two. Here the LSMR and LSQR curves are very close for a large number of the initial iterations, but in the later, probably more important iterations, LSQR shows a marked advantage.

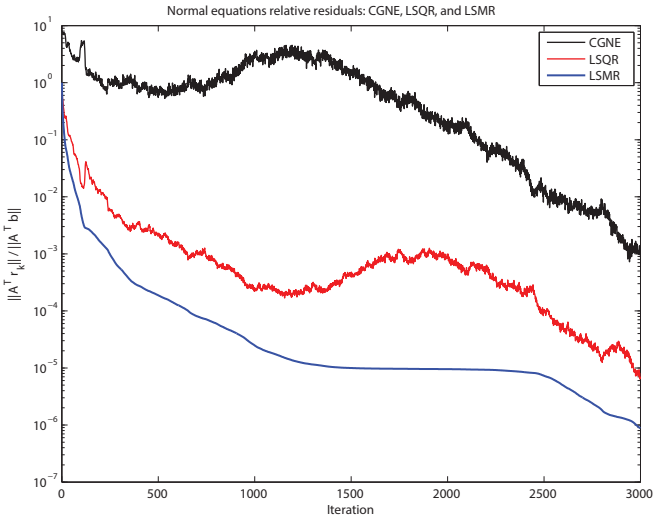

FIG. 11.2. *Recorded normal equations relative residuals,* $\|A^T r_k\|/\|A^T b\|$*, for CGNE, LSQR, and LSMR, applied to a skew-symmetric numerical example of dimensions* 4096×4096*.*

Figure 11.2 shows the convergence history of $\|A^T r_k\|/\|A^T b\|$ for CGNE, LSQR,

and LSMR. As expected, the curve for LSMR lies below the curve for LSQR since the former minimizes this particular norm. The convergence curves for CGNE and LSQR are nonmonotonic, as typically observed for other matrices (see the study in [5] for LSQR).

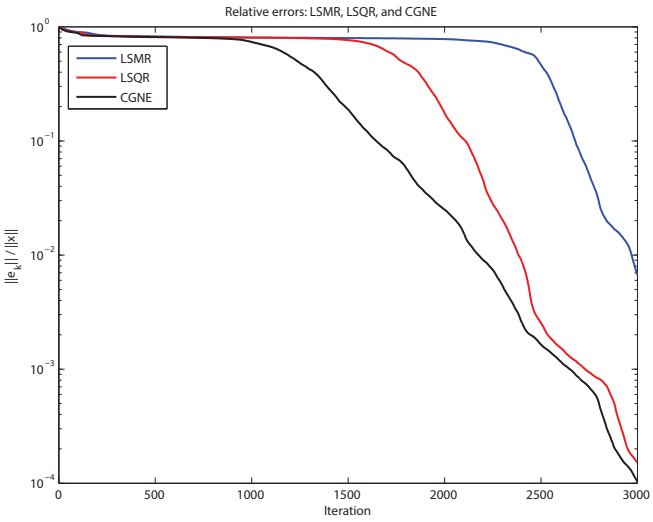

FIG. 11.3. *Recorded relative errors, $\|e_k\|/\|x\|$, for LSMR, LSQR, and CGNE, applied to a skew-symmetric example of dimensions* 4096×4096. *Here $e_k = x_k - x$, where x is the exact solution.*

Figure 11.3 compares the convergence behavior of the error norms for LSMR, LSQR, and CGNE. Note how CGNE minimizes the Euclidean norm of the error throughout the iterations; see the paragraph containing (2.6). In fact this theoretical error norm is always monotonic for CGNE, LSQR, and LSMR: the first follows from the minimization property of CGNE and the other two from [6, Tab. 5.2] (and from (12.3), which follows from [11]).

12. Choice of algorithms. There are important relationships between LSQR and LSMR when solving linear least squares problems $\min_x \|b - Ax\|_2$ where the matrix A has full column rank. Let us use the following notation:

- In solving $\min_x \|b - Ax\|_2$ with residuals $r_k = b - Ax_k$, denote LSQR vectors by $\cdot^R$ (see (2.7) and (8.4)) and LSMR by $\cdot^L$ (see (9.4)).
- For $A^TAx = A^Tb$ with residuals $\tilde{r}_k = A^T(b - A\tilde{x}_k)$, let CG and MINRES vectors be denoted by $\cdot^{CG}$ and $\cdot^{MR}$, respectively.

Since CGNR and LSQR are mathematically equivalent to CG applied to $A^TAx = A^Tb$ (see the last paragraph of section 2), we see that $\tilde{x}_k^{CG} = x_k^R$, so that

$$\tilde{r}_k^{CG} = A^T(b - A\tilde{x}_k^{CG}) = A^T(b - Ax_k^R) = A^Tr_k^R.$$

Now MINRES for $A^TAx = A^Tb$ and LSMR for $\min_x \|b - Ax\|_2$ both minimize $\|A^T(b - Ax_k)\|_2$ over the same subspace (see section 9), giving $\tilde{x}_k^{MR} = x_k^L$, so that

$$\tilde{r}_k^{MR} = A^T(b - A\tilde{x}_k^{MR}) = A^T(b - Ax_k^L) = A^Tr_k^L.$$

Equipped with the above definitions, we prove below that

(12.1)
$$\|r_k^L\|_2^2 = \|r_k^R\|_2^2 + \sum_{j=0}^{k-1} \frac{\|A^T r_k^L\|_2^4}{\|A^T r_j^L\|_2^4}(\|r_j^R\|_2^2 - \|r_{j+1}^R\|_2^2) \quad \text{(see Figure 11.1)},$$

(12.2)
$$\|A^T r_k^R\|_2 = \frac{\|A^T r_k^L\|_2}{\sqrt{1 - \|A^T r_k^L\|_2^2/\|A^T r_{k-1}^L\|_2^2}} \quad \text{(see Figure 11.2)},$$

(12.3)
$$\|x - x_k^R\|_2 \le \|x - x_{k-1}^R\|_2, \quad \|x - x_k^R\|_2 \le \|x - x_k^L\|_2 \le \|x - x_{k-1}^L\|_2 \quad \text{(see Figure 11.3)}.$$

Hestenes and Stiefel [11] applied CG to $\mathbf{A}h = k$ with $\mathbf{A}$ SPD; see [11, p. 411, 2nd para.], where the equivalence with our notation is $\mathbf{A} \equiv A^T A$, $h \equiv x$, $k \equiv A^T b$. They used x_k to denote their CG iterate (see the start of [11, sect. 6]) and $\bar{x}_k$ to denote their MINRES iterate (see the start of [11, sect. 7] and [11, Thm. 7:2]), so from our results above,

$$x_k \equiv \tilde{x}_k^{CG} = x_k^R, \quad \tilde{r}_k^{CG} = A^T r_k^R, \quad \bar{x}_k \equiv \tilde{x}_k^{MR} = x_k^L, \quad \tilde{r}_k^{MR} = A^T r_k^L. \tag{12.4}$$

They showed [11, Theorem 7:4, eq. (7:7)] that

$$f(\bar{x}_k) = f(x_k) + \|\bar{r}_k\|_2^4 \sum_{j=0}^{k-1} \frac{f(x_j) - f(x_{j+1})}{\|\bar{r}_j\|_2^4},$$

where from [11, eq. (4:5)], $f(x_j) \triangleq (h - x_j)^T \mathbf{A}(h - x_j)$. In our notation,

$$\begin{aligned} f(x_j) &= (x - x_j^R)^T (A^T A)(x - x_j^R) = \|Ax - Ax_j^R\|_2^2 = \|r_j^R - (b - Ax)\|_2^2 \\ &= \|r_j^R\|_2^2 + \|b - Ax\|_2^2 - 2(r_j^R)^T(b - Ax) = \|r_j^R\|_2^2 - \|b - Ax\|_2^2, \end{aligned}$$

where the last equality follows from

$$(r_j^R)^T(b - Ax) = (b - Ax_j^R)^T(b - Ax) = b^T(b - Ax) = (b - Ax)^T(b - Ax),$$

since for the least squares residual $b - Ax$, $A^T(b - Ax) = 0$. Similarly,

$$f(\bar{x}_j) = \|r_j^L\|_2^2 - \|b - Ax\|_2^2.$$

Then using (12.4), $\bar{r}_j = \tilde{r}_j^{\mathrm{MR}} = A^T r_j^L$, and (12.1) follows.

Next, (12.2) comes directly from the CG–MINRES relationship for symmetric systems $\|\tilde{r}_k^{\mathrm{CG}}\|_2 = \|\tilde{r}_k^{\mathrm{MR}}\|_2/\sqrt{1 - \|\tilde{r}_k^{\mathrm{MR}}\|_2^2/\|\tilde{r}_{k-1}^{\mathrm{MR}}\|_2^2}$, which follows from [19, eqs. (7.4) and (7.5)] using $s_k^2 + c_k^2 = 1$; see also [8, Exer. 5.1]. It is a special case of the well-known "peak-plateau" relationship between Galerkin and minimum residual methods, often attributed to Weiss; see [22, 23].

Finally, by using the equivalences in (12.4) we can obtain (12.3). The first inequality in (12.3), the monotonicity of the error norm using LSQR, follows directly from [11, Thm. 6:2], while the final two inequalities, the superiority of the error norm using LSQR and the monotonicity of the error norm using LSMR, follow directly from the corrected version of [11, Thm. 7:5], which contains typographical errors.

We will state a corrected version of the theorem and proof using their notation and equation references. The errors and corrections were pointed out to us by Liesen [16], who noted that the errors were not present in the typewritten report of Hestenes and Stiefel, which was the basis for the printed paper.

THEOREM 12.1 (see [11, Theorem 7:5, p. 419]). *The error vector $y_i = h - x_i$ is shorter than (i.e., has smaller 2-norm than) the error vector $\bar{y}_i = h - \bar{x}_i$. Moreover, $\bar{y}_i$ is shorter than $\bar{y}_{i-1}$.*

Proof. The first statement follows from (7:2) and theorem 6:2, since $\bar{x}_i$ is in S_i. By (7:2) the point $\bar{x}_i$ lies in the line segment joining x_i to $\bar{x}_{i-1}$. The distance from h to $\bar{x}_i$ exceeds the distance from h to x_i. It follows that as we move from $\bar{x}_i$ to $\bar{x}_{i-1}$ the distance from h is increased, as was to be proved. ∎

The simple proofs here did not require the matrices to be skew-symmetric. The results hold for general linear least squares problems $\min_x \|b - Ax\|_2$ when A has full column rank. Initial computations suggest that (12.1)–(12.3) might also hold using finite precision computations.

To understand the relative behavior of $\|r_k^R\|_2$ and $\|r_k^L\|_2$ in Figure 11.1, note that in (12.1), $\|A^T r_k^L\|_2 / \|A^T r_j^L\|_2 \leq 1$, so when $\|A^T r_k^L\|_2$ is decreasing quickly, or when the LSQR residual norm $\|r_k^R\|_2$ is decreasing slowly, we will have $\|r_k^L\|_2 \approx \|r_k^R\|_2$; see Figures 11.2 and 11.1. But when $\|A^T r_k^L\|_2$ stagnates and $\|r_k^R\|_2$ is decreasing quickly, then $\|r_k^L\|_2$ and $\|r_k^R\|_2$ will differ significantly; see the later iterations in Figure 11.1.

For other types of problems, Fong and Saunders [5] wrote that the LSMR residual norm "is never very far behind the corresponding value for LSQR." Their comments probably arose because the combination of two conditions is required to produce a clear advantage of LSQR over LSMR: $\|A^T r_k^L\|_2$ stagnating with $\|r_k^R\|_2$ decreasing quickly, while either of two conditions would be sufficient for the two curves to be close. However, we encountered the case in Figure 11.1 without seeking it. Figure 11.1 and (12.1) emphasize the continued useful role of LSQR in minimizing the residual norm.

For Figure 11.2, we see from (12.2) that when $\|A^T r_k^L\|_2$ stagnates we can have $\|A^T r_k^R\|_2$ significantly greater than $\|A^T r_j^L\|_2$. Thus while the stagnation of $\|A^T r_k^L\|_2$ can make the normal equations residual of LSQR significantly worse than that of LSMR, it can also contribute to the residual of LSQR being noticeably better than that of LSMR.

From the above analysis and computations we now suggest some possible choices among these three GKB based algorithms, CGNE (Craig's method), LSQR (CGNR), and LSMR, for more general unsymmetric matrix problems.

For full row rank A, in theory, CGNE minimizes $\|\hat{x} - x_k^E\|_2$ where $\hat{x}$ is the minimum Euclidean norm solution of $Ax = b$. For the error norm, CGNE showed a clear advantage over LSQR in Figure 11.3, which showed a clear advantage over LSMR, so that CGNE appears to be a possible choice for minimizing the error norm, but there are difficulties. If A is ill-conditioned for solution of equations, then the computed solution can be very different from the exact solution, and it is probably better to seek a method giving a small residual. More significantly, there is usually no good measure of $\|\hat{x} - x_k^E\|_2$ available during the computation. Unfortunately, the nonmonotonic nature of $\|r_k^E\|_2$ and $\|A^T r_k^E\|_2$ (see, e.g., Figures 11.1 and 11.2) indicates that stopping criteria based on these are also less than ideal for CGNE. Nevertheless, if one criterion is to obtain as small as possible a computed error in a fixed number of steps, CGNE would be a good choice. CGNE is also an important method for solving full row rank underdetermined systems.

For solving $Ax = b$ with nonsingular A, LSQR cannot only give a significantly smaller residual norm than LSMR (see Figure 11.1) but also a significantly smaller error norm too (see Figure 11.3 and (12.3)), and it would appear to be the algorithm of choice in this situation.

For incompatible full column rank linear least squares problems, LSMR minimizes the useful criterion $\|A^T r_k\|_2$, while $\|A^T r_k^E\|_2$ and $\|A^T r_k^R\|_2$ can be nonmonotonic. The LSMR residual norm is also monotonic, and often quite close to that given by LSQR, so that LSMR is an ideal choice for least squares, as was pointed out in [5, 6].

To find practical codes for such general problems, follow the relevant links available from http://stanford.edu/group/SOL/software/lsqr/.

For skew-symmetric systems, we have dealt only with nonsingular A, where using LSQR with a criterion based on the residual norm would seem to be the correct approach, unless using another criterion and algorithm has an obvious advantage.

13. Summary and conclusions. Because of the numerical equivalence of the Lanczos process and GKB for skew-symmetric matrices, we have seen that for each of the above solution of equations methods using the Lanczos process there is an equivalent method based on GKB, and the latter is easier to develop because we do not have to figure out the nonzero elements and their relationships from among the zero elements. Briefly, for

$$x_k \in \mathcal{K}_k(A, b) \triangleq \text{span}\{b, Ab, \ldots, A^{k-1}b\} = \text{Range}(W_k), \quad \text{say, with} \quad r_k \triangleq b - Ax_k,$$

the following pairs of algorithms for $Ax = b$, $A^T = -A$ are *theoretically* equivalent:

- Lanczos–Galerkin ($W_k^T r_k = 0$) and CGNE (Craig's method),
- Lanczos Minimum Residual ($\min \|r_k\|_2$) and CGNR (CGLS, LSQR),
- Lanczos $\min \|A^T r_k\|_2$ and LSMR (minimum normal equations residual).

But if these are implemented carefully with the second of each pair being implemented via GKB in the form of "Bidiag 1," each pair is *numerically* identical. The first two pairs support the theoretical results in [4] and earlier works and extend them to computational equivalences.

We introduced a method based on the Lanczos process for minimizing the normal equations residual when $A^T = -A$, and section 9 showed the equivalence of this method to LSMR in [5] for such matrices. Adapting LSMR to skew-symmetric matrices is straightforward, and our numerical example showed that the smoothness of $\|r_j\|$ and $\|A^T r_j\|$ is as observed by Fong and Saunders in [5, 6] for other types of matrices. The residual norms for LSMR and LSQR decrease monotonically, while that for CGNE can be quite erratic. The residual norm for LSMR can be much larger than that for LSQR, but it usually is not. However, while $\|A^T r_j\|$ decreases monotonically for LSMR, it decreases in an erratic fashion for both CGNE and LSQR.

GKB provides a concise way of deriving Galerkin, minimum residual, and minimum normal equations residual Krylov subspace methods for skew-symmetric linear systems. The unique mathematical structure of the tridiagonal matrix that arises during the Lanczos process, and specifically the fact that the diagonal is zero and that the subdiagonal is just the negation of the superdiagonal, is the reason for being able to derive the equivalence with GKB. The numerical behaviors of the versions based on the Lanczos process and GKB should be identical, but the insight gained from the development of these algorithms should make it easier to carry out rounding error analyses with the approach initiated in [17, 18].

Acknowledgments. We are indebted to Stan Eisenstat for his superb insights on this topic and his very helpful comments on this paper. The authors would also

like to thank Jörg Liesen and the three referees for very valuable comments on an earlier version of the paper, which improved the paper greatly.

REFERENCES

[1] W. E. Arnoldi, *The principle of minimized iteration in the solution of the matrix eigenvalue problem*, Quart. Appl. Math., 9 (1951), pp. 17–29.
[2] P. Benner and H. Faßbender, *An implicitly restarted symplectic Lanczos method for the symplectic eigenvalue problem*, SIAM J. Matrix Anal. Appl., 22 (2000), pp. 682–713, doi: 10.1137/S0895479898343115.
[3] J. Craig, *The n-step iteration procedure*, J. Math. and Phys., 34 (1955), pp. 64–73.
[4] S. C. Eisenstat, *Equivalence of Krylov Subspace Methods for Skew-Symmetric Linear Systems*, Department of Computer Science, Yale University, preprint, arXiv:1512.00311, 2015.
[5] D. C.-L. Fong and M. Saunders, *LSMR: An iterative algorithm for sparse least-squares problems*, SIAM J. Sci. Comput., 33 (2011), pp. 2950–2971, doi:10.1137/10079687X.
[6] D. C.-L. Fong and M. A. Saunders, *CG versus MINRES: An empirical comparison*, SQU J. Sci., 17 (2012), pp. 44–62.
[7] G. Golub and W. Kahan, *Calculating the singular values and pseudo-inverse of a matrix*, SIAM J. Numer. Anal., 2 (1965), pp. 205–224, doi:10.1137/0702016.
[8] A. Greenbaum, *Iterative Methods for Solving Linear Systems*, Frontiers Appl. Math. 17, SIAM, Philadelphia, 1997, doi:10.1137/1.9781611970937.
[9] C. Greif and J. M. Varah, *Iterative solution of skew-symmetric linear systems*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 584–601, doi:10.1137/080732390.
[10] C. Gu and H. Qian, *Skew-symmetric methods for nonsymmetric linear systems with multiple right-hand sides*, J. Comput. Appl. Math., 223 (2009), pp. 567–577, doi:10.1016/j.cam.2008.01.001.
[11] M. R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Research Nat. Bur. Standards, 49 (1952), pp. 409–436.
[12] Y. Huang, A. J. Wathen, and L. Li, *An iterative method for skew-symmetric systems*, Information, 2 (1999), pp. 147–153.
[13] R. Idema and C. Vuik, *A Minimal Residual Method for Shifted Skew-Symmetric Systems*, Technical Report 07-09, Delft University of Technology, Delft, The Netherlands, 2007.
[14] E. Jiang, *Algorithm for solving shifted skew-symmetric linear system*, Front. Math. China, 2 (2007), pp. 227–242, doi:10.1007/s11464-007-0016-3.
[15] C. Lanczos, *An iteration method for the solution of the eigenvalue problem of linear differential and integral operators*, J. Research Nat. Bur. Standards, 45 (1950), pp. 255–282.
[16] J. Liesen, *private communication*, 2015.
[17] C. C. Paige, *A useful form of unitary matrix obtained from any sequence of unit* 2*-norm n-vectors*, SIAM J. Matrix Anal. Appl., 31 (2009), pp. 565–583, doi:10.1137/080725167.
[18] C. C. Paige, *An augmented stability result for the Lanczos Hermitian matrix tridiagonalization process*, SIAM. J. Matrix Anal. Appl., 31 (2010), pp. 2347–2359, doi:10.1137/090761343.
[19] C. C. Paige and M. A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal., 12 (1975), pp. 617–629, doi:10.1137/0712047.
[20] C. C. Paige and M. A. Saunders, *LSQR: An algorithm for sparse linear equations and sparse least squares*, ACM Trans. Math. Software, 8 (1982), pp. 43–71, doi:10.1145/355984.355989.
[21] C. C. Paige and M. A. Saunders, *ALGORITHM* 583*: LSQR: Sparse linear equations and sparse least squares problems*, ACM Trans. Math. Software, 8 (1982), pp. 195–209.
[22] H. F. Walker, *Residual smoothing and peak/plateau behavior in Krylov subspace methods*, Appl. Numer. Math., 19 (1995), pp. 279–286, doi:10.1016/0168-9274(95)00087-9.
[23] R. Weiss, *Convergence Behavior of Generalized Conjugate Gradient Methods*, Ph.D. thesis, University of Karlsruhe, Karlsruhe, Germany, 1990.