Chapter 5
Lyapunov and Sylvester Matrix Equations

Abstract ADI iterative solution of Lyapunov and Sylvester matrix equations may be enhanced by availability of a stable algorithm for similarity reduction of a full nonsymmetric real matrix to low bandwidth Hessenberg form. An efficient and seemingly stable method described here has been applied successfully to an assortment of test problems. Significant reduction in computation for low rank right-hand sides is possible with ADI but not with alternatives. Initial analysis by Penzl was improved upon by Li and White. A Lanczos algorithm is exposed here for approximating a full matrix by a sum of low rank matrices. This is especially useful in a parallel environment where the low rank component solutions may be computed in parallel.

5.1 Similarity Reduction of Nonsymmetric Real Matrices to Banded Hessenberg Form

In Chap. 3 Sect. 3.8 it was observed that the search for an efficient and robust similarity reduction of a real matrix to banded Hessenberg form was motivated by application to ADI iterative solution of Lyapunov and Sylvester matrix equations. One promising candidate will now be described. The limited numerical studies thus far performed have been encouraging.

Any real $n \times n$ symmetric matrix may be reduced to a similar symmetric tridiagonal matrix in $2n^3/3$ flops by successive Householder (HH) transformations [Golub and vanLoan, 1983]. All eigenvalues may then be computed with another $2n^3/3$ flops. Any unsymmetric real matrix may be reduced similarly to upper Hessenberg form with $5n^3/3$ flops [Golub and vanLoan, 1983 (p. 223)]. All eigenvalues may now be computed with the implicit QR algorithm in around $8n^3$ flops [Golub and vanLoan, 1983 (p. 235)]. Half as many flops are required when gaussian rather than HH transformations are used. However, greater stability of HH has led to its use. The MATLAB program EIG computes eigenvalues in this manner.

A banded matrix may be stored in sparse form. Eigenvalues of a sparse matrix may be computed with the MATLAB EIGS program. This program uses an Arnoldi iteration and all eigenvalues may be found with $O(bn^2)$ flops, where b is the upper bandwidth. This possibility stimulated search for stable similarity reduction of unsymmetric matrices to banded form. Early studies [Dax and Kaniel, 1981] indicated that eigenvalues of a tridiagonal matrix obtained by gaussian reduction of an unsymmetric matrix with unbounded multipliers were fairly accurate. Although this eliminated the $8n^3$ flops of the QR algorithm, loss of stability, accuracy, and robustness resulted in few applications. More recent research was directed toward gaussian reduction to banded upper Hessenberg form with limited gaussian multipliers [Geist, Lu and Wachspress, 1989; Howell and Geist, 1995]. The BHESS program [Howell, Geist and Diaa, 2005] culminating these studies was a promising alternative to QR. However interaction of large gaussian multipliers limited accuracy and stability. Relationship between bandwidth, multiplier bound, and accuracy was not amenable to analysis. QR reduction to upper Hessenberg form followed by the implicit QR algorithm with a double Wilkinson shift as in the MATLAB EIG program remained the method of choice. In 1995 I suggested a possible improvement of BHESS which seems not to have been programmed until my QGBAND in 2011. When applied to symmetric matrices this algorithm is identical to the standard HH reduction. It reduces a nonsymmetric real matrix to a banded upper Hessenberg matrix. The reduction progresses from row $k = 1$ to $n - 2$. In all numerical studies reported here an input matrix A was first normalized to $S_0 = \text{snorm} * A$ where $\text{snorm} = 1/\sqrt{\|A\|_1 \|A\|_\infty}$. This results in a bound of unity on the spectral radius of S_0. The 1-norm of S_0 was close to unity for all random matrices reported here. Let matrix S_0 reduced through $k - 1$ be S_{k-1} and let $S \equiv S_{n-2}$. Let $k(1)$ be the first row in S_{k-1} with a nonzero element beyond col kV:

```
D X 0................................0
X D X 0 ...........................0
0 X D X X 0.....................0
0 0 X D X X 0..................0
0 0 0 X D X X 0...............0
0 0 0 0 X D X X...............X        k(1)
0 0 0 0 0 X D X................X        k
0 0 0 0 0 0 X D X.............X
0 0 0 0 0 0 X X D X.........X
.........................................
0 0 0 0 0 0 X X.................D
```

Column k is now reduced to zero below row $k + 1$ with a HH step:

```
D X 0................................0
X D X 0 ...........................0
0 X D X X 0.....................0
0 0 X D X X 0..................0
0 0 0 X D X X 0...............0
0 0 0 0 X D X...................X          k(1)
0 0 0 0 0 X D X................X          k
0 0 0 0 0 0 X D X.............X
0 0 0 0 0 0 0 X D X..........X
.........................................
0 0 0 0 0 0 0 X..................D
```

For the standard HH reduction to upper Hessenberg form this yields S_k and column $k+1$ is then reduced. Further row treatment is generally required for banded reduction. All rows from $k(1)$ on may now have nonzero entries beyond column k. A HH reduction to reduce row $k(1)$ to zero beyond column $k+1$ would reintroduce nonzero elements below row $k+1$ in column k. The HH row reduction is chosen instead to reduce row $k(1)$ beyond column $k+2$:

```
D X 0................................0
X D X 0 ...........................0
0 X D X X 0.....................0
0 0 X D X X 0..................0
0 0 0 X D X X 0 0...,,,,.....0
0 0 0 0 X D X X X 0........0          k(1)
0 0 0 0 0 X D X................X          k
0 0 0 0 0 0 X D X.............X
0 0 0 0 0 0 0 X D X..........X
.........................................
0 0 0 0 0 0 0 X..................D
```

This leaves column k unchanged. The only nonzero entries in row $k(1)$ beyond column k are $S_{k(1),k+1}$ and $S_{k(1),k+2}$. A bound M is specified on the magnitude of gaussian multipliers. If the magnitude $|S_{k(1),k+2}/S_{k(1),k+1}|$ is greater than M, there is no further row reduction before proceeding to reduction of column $k+1$. The bandwidth is increased by one. On the other hand, if the ratio is less than M element $S_{k(1),k+2}$ is reduced to zero with a gaussian step:

```
D X 0................................0
X D X 0 ...........................0
0 X D X X 0.....................0
0 0 X D X X 0..................0
0 0 0 X D X X 0...............0
0 0 0 0 X D X X 0............0          k(1)
0 0 0 0 0 X D X................X        k
0 0 0 0 0 0 X D X.............X
0 0 0 0 0 0 0 X D X..........X
.........................................
0 0 0 0 0 0 0 X.........x.......D
```

After this step, $k(1)$ is increased to $k(1)+1$ and the algorithm progresses to $k+1$. (In this example, the fact that $k(1)$ was $k-1$ meant that the bandwidth was increased before step k from one to two nonzero elements beyond the diagonal. Each time the width is increased $k-k(1)$ increases by one. Thus, the two rather than one elements to the right of the diagonal at $k(1)$ after reduction at k were not an increase at k.) The QG reduction requires $8n^3/3$ flops. However, eigenvalues of the banded matrix can be computed in $O(bn^2)$ flops instead of the $8n^3$ flops of QR.

In earlier attempts the entire reduction was done with gaussian transformations. Later attempts used HH reduction of columns followed by gaussian reduction of rows but the preliminary HH row reduction was not performed. There is significantly less interaction of gaussian multipliers with this preliminary HH step to reduce row $k(1)$ to zero beyond element $(k(1), k+2)$. A measure of the stability of the reduction from S_0 to S is the ratio $||S||_1/||S(0)||_1$.

Results of two numerical studies illustrate characteristics of the QG reduction Table 5.1. Eigenvalues of S differed from those of S_0 by $O(10^{-11})$ in all cases. The first matrix was a random matrix with $n = 100$ and the second was a random matrix of order 300. Extra is the number of nonzero entries beyond the tridiagonal:

Table 5.1 Effect of multiplier bounds

n	M	Extra	$\|S\|$
100	10,000	0	178
100	1,000	0	178
100	100	171	45
100	10	2,551	50
300	10,000	239	1,557
300	1,000	317	87
300	100	4,870	80
300	10	36,550	53

For $n = 100$ a bound of $M = 1,000$ sufficed to reduce to tridiagonal form. For $n = 300$ the bound of $M = 1,000$ led to extra nonzero elements beyond the tridiagonal but a relatively small increase in the norm. Although $M = 10,000$

led to fewer elements beyond the tridiagonal the norm increased significantly. The eigenvalues remained accurate. The default bound of $M = 1,000$ was chosen on the basis of this and other numerical studies.

Studies were performed on a PC and matrices of large order required significant memory and computing time. A random matrix of order 1,000 was reduced in about ten minutes. The default value of $M = 1000$ resulted in 15 band increases with a total of 8,941 nonzero elements beyond the tridiagonal. The 1-norm of S was 113.5. The absolute values of the eigenvalues ranged from 2×10^{-4} to 0.037 and the reduced matrix values differed from the true values by less than 5×10^{-10}. These preliminary studies suggest that QG is a robust algorithm for reducing a full matrix to low-bandwidth upper Hessenberg form with accurate retention of eigenvalues. More extensive numerical studies are needed.

5.2 Application of QG to the Lyapunov Matrix Equation

My recognition in 1982 that the Lyapunov matrix equation is a model ADI problem initiated extensive development of methods for solving Lyapunov and Sylvester equations. [Benner, Li and Truhar, 2009] provided an excellent overview of this effort. They introduced a projection method as a competitive alternative to other methods for a wide class of problems of practical concern. They also observed that the existence of an algorithm for efficient reduction of a full matrix to banded Hessenberg form as a prelude to ADI iterative solution could be quite efficient. The QG algorithm seems to offer that possibility. Direct solution by ADI requires solution of two linear systems of order n for each iteration. This requires around $4n^3/3$ flops so that J iterations require around $4Jn^3/3$ flops. The break-even point with B–S is when $4J/3 = 20$ or $J = 15$. Efficient ADI iteration requires knowledge of the spectrum of A which may itself be computation intensive. Once the ADI linear system for a two-step iteration has been factored in $2n^3/3$ flops, the back-substitution stage on the n columns of the right-hand side can be performed in parallel. This leads to $2Jn^3/3$ flops or a break-even number of $J = 30$. The number of ADI iterations required to achieve prescribed accuracy varies as the log of the condition of matrix A.

The HH similarity G that reduces the Lyapunov matrix A to upper Hessenberg form S requires $5n^3/3$ flops. The QG transformation from A to the banded S with two HH transformations at each step requires $8n^3/3$ flops. One may either store G or the HH and gauss transformations in n^2 words of memory. Transforming the right-hand side C to H requires another $10n^3/3$ flops for a total of $6n^3$ flops to reduce Eqs. 1.1 and 1.2 of Chap. 3 to Eq. 2 of Chap. 3. This compares with $25n^3/6$ flops for the B–S transformation. Recovery of X from Y requires another $20n^3/9$ flops when one takes advantage of symmetry. A total of $74n^3/9$ flops are needed. B–S recovery requires $10n^3/9$ flops for a total of $55n^3/9$ flops. The B–S solution of the transformed equation is also $O(n^3)$ and is a major part of the computation, leading to a total of around $20n^3$ flops.

The only eigenvalues needed for determining effective ADI iteration parameters are those with small real part and one of largest magnitude (the spectral radius of S). After QG reduction to banded form the sparse matrix MATLAB EIGS program may be used to compute these crucial values efficiently. When S is of order n one may compute $O(n^{1/2})$ small eigenvalues and $O(n^{1/4})$ values of largest magnitude. This is an $O(n^{3/2})$ computation. Solution of Eq. 2 of Chap. 3 for Y is $O(bn^2)$. Thus, even though the transformation to banded form with QG requires around 50% more flops than transformation to Hessenberg form, the overall $O(n^3)$ flop count for ADI–QG is around 41% of the B–S value.

In a parallel environment ADI iteration has another advantage. Although the transformation can be performed in parallel for both reduction algorithms, columns in each of the two ADI iteration steps can be computed in parallel while the B–S solution of the transformed equations is less amenable to parallel computation.

5.3 Overview of Low-Rank Right-Hand Sides

The ADI method can take advantage of low-rank right-hand side C in the Lyapunov equation. This of not possible with the alternative methods thus far used. Low-rank right-hand sides recur in application. Let R be an $n \times m$ matrix with $m << n$. Then $C = RR'$ in Eq. 1 is of rank m. Penzl [Penzl, 1999] observed that low-rank Lyapunov equations could be solved more efficiently with ADI iteration. Similar savings could not be realized with the B–S algorithm. Subsequently, Penzl's algorithm was improved in [Li and White, 2002]. The Li–White (LW) algorithm requires solution of one linear system of order n with an $n \times m$ right-hand side for each ADI iteration. Li and White did not transform from Eqs. 1.1 and 1.2. They considered sparse A and approximated solution of the ADI linear systems by an iteration with programs like GMRES. In a parallel environment the number of processors to solve for all columns in parallel is decreased from n to m. The B–S algorithm cannot take full advantage of low-rank right-hand sides. The LW approach without preliminary transformation to banded form has been adopted by some practitioners. The Sylvester matrix equation was discussed in Eqs. 9.1 and 9.2 of Chap. 3. Low-rank equations may be treated in similar fashion to low-rank Lyapunov equations [Wachspress, 2008].

5.4 The Penzl Algorithm

Penzl's algorithm will now be described. The low-rank Lyapunov equation $AX + XA^\top = CC^\top$ may be reduced to

$$SY + YS^\top = H, \tag{1}$$

where $S = GAG^{-1}$, $Y = GXG^{\top}$, and $H = FF^{\top}$ with $F = GC$ of rank $r << n$. Matrix G^{-1}, used to recover X from Y, is accumulated during the QG reduction of A to S. The nonfactored ADI iteration equations with $Y_o = 0$ and the number of iterations J determined from the spectrum and a prescribed bound on the solution error are

$$[S + w_j I]Y_{j-1/2} = H + Y_{j-1}[w_j I - S]^{\top}, \tag{2.1}$$

$$[S + w_j I]Y_j = H + Y_{j-1/2}^{\top}[w_j I - S]^{\top}, \tag{2.2}$$

for $j = 1, 2, \ldots, J$. For each value of j, matrix $S + w_j I$ is factored, and the $2n$ linear systems for the columns of $Y_{j-1/2}$ and then Y_j are solved. The number of iterations J is often $O(\log n)$. The reduction to banded form and recovery of X from Y are the $O(n^3)$ steps. Although Y_j is symmetric, $Y_{j-1/2}$ is in general not symmetric. If the iteration matrix for the j-th step is defined as

$$R_j = [w_j I + S]^{-1}[S - w_j I], \tag{3}$$

then

$$Y_j = 2w_j[w_j I + S]^{-1}H[w_j I + S]^{-\top} + R_j Y_{j-1} R_j^{\top}. \tag{4}$$

Equation 4 provides the basis for improved efficiency when $r << n$. Let the ADI approximation after iteration j with parameter w_j be Y_j. Suppose

$$Y_j = \sum_{i=1}^{j} Z_i(j) Z_i(j)^{\top} \tag{5}$$

Then, with $Z_i(0) = 0$,

$$Z_j(j) = \sqrt{2w_j}[w_j I + S]^{-1}F \tag{6.1}$$

$$Z_i(j) = R_j Z_i(j-1) \qquad i = 1, 2, \ldots, j-1. \tag{6.2}$$

Now the ADI iteration of Eqs. 6.1 and 6.2 replaces Eqs. 2.1 and 2.2. We note that Y_j is of rank jr. Note that $Z_i(j-1)$ is a matrix of order nrj, so the algorithm requires solving $rJ(rJ + 1)/2$ linear systems of order n.

The reduced equation form when complex parameters are used is maintained by rewriting Eq. 2 as

$$[S + w_j I]Y_{j-1/2} = H + Y_{j-1}[w_j I - S^{\top}], \tag{7.1}$$

$$Y_j[S^{\top} + w_j' I] = H + [w_j' I - S]Y_{j-1/2}, \tag{7.2}$$

where w' is the complex conjugate of w. Then for this iteration, Eqs. 3 and 6 become

$$Q_j = [w_j I + S]^{-1}[S - w_j' I], \tag{8.1}$$

$$Z_j(j) = \sqrt{w_j + w_j'}[w_j I + S]^{-1}F, \tag{8.2}$$

$$Z_i(j) = Q_j Z_i(j-1) \qquad i = 1, 2, \ldots, j-1. \tag{8.3}$$

We observe that when $w_{j+1} = w'_j$, $Q_j Q_{j+1} = R_j R_{j+1}$. Thus, the ADI iteration matrix is recovered when the roles of w and w' are interchanged for the iteration with the conjugate parameter. Since this introduces complex Z_j and complex arithmetic requires more flops than real arithmetic, the complex iteration parameters are saved for last. Repeated use of a single real parameter with more iterations may sometimes be more efficient than the "optimal" set of complex parameters. In some cases, it is best to apply complex parameters only to reduce error associated with eigenvalues close to the imaginary axis and to embed the remaining spectrum in a disk for which repeated use of a real parameter is optimal.

A measure of solution accuracy is found by plugging it into the equation. For the full system one computes AY and the residual error $||H - AY - YA^\top||_1$. For the low-rank system, $W_i = SY_i$ is first computed for each i and then $U_i = W_i Y_i^*$ is summed to yield SY.

5.5 The Li-White Algorithm

The Li-White recursive algorithm will now be described. Reduction to banded upper Hessenberg form was common to all Lyapunov solvers studied. By Eq. 8.2,

$$Z_J(J) = \sqrt{w_J + w'_J}[w_J I + S]^{-1} F, \tag{9.1}$$

$$Z_{J-1}(J-1) = \sqrt{w_{J-1} + w'_{J-1}}[w_{J-1} I + S]^{-1} F. \tag{9.2}$$

By Eqs. 9.1 and 9.2,

$$\begin{aligned} Z_{J-1}(J) &= [w_J I + S]^{-1}[S - w'_J I] Z_{J-1}(J-1) \\ &= \sqrt{\frac{w_{J-1} + w'_{J-1}}{w_J + w'_J}}[w_{J-1} I + S]^{-1}[S - w'_J I] Z_J(J). \end{aligned} \tag{10}$$

In general, proceeding back from $i = J$ to 1, the r columns of each $Z_i(J)$ are computed in succession:

$$Z_J(J) = \sqrt{w_J + w'_J}[w_J I + S]^{-1} F, \tag{11.1}$$

$$Z_{i-1}(J) = \sqrt{\frac{w_{i-1} + w'_{i-1}}{w_i + w'_i}}[I - (w'_i + w_{i-1})(S + w_{i-1} I)^{-1}] Z_i(J), \tag{11.2}$$

$$i = J, J-1, \ldots, 2.$$

The result is independent of parameter ordering. Complex arithmetic is reduced by ordering all complex parameters ahead of real parameters since the algorithm starts with w_J and proceeds backward. Now each iteration requires solution of only r linear systems for a total of rJ systems rather than the $rJ(rJ + 1)/2$ required of the Penzl algorithm.

5.6 Sylvester Equations

This approach also applies to Sylvester equations. Consider the banded system of order $n \times m$:

$$SY + YT = EF^\top \tag{12.1}$$

of the Sylvester equation

$$AX + XB = H, \tag{12.2}$$

where $H = CD^\top$, E is order $n \times r$, and F is order $m \times r$. The transformation matrices L_s and L_t are saved for computing $X = L_s Y L_t$. The ADI approximation to the solution after j iterations is

$$Y_j = \sum_{i=1}^{j} U_i(j) V_i(j). \tag{13}$$

Now matrices $U_i(j)$ and $V_i(j)$ must be computed for each j. There are in general two iteration parameters, u_j and v_j, for each iteration j. Matrix R_j of Eq. 3 is now

$$R_j = [u_j I + S]^{-1}[v_j I - S], \tag{14}$$

and similarly

$$Q_j = [v_j I + T]^{-1}[u_j I - T]. \tag{15}$$

Now Eq. 4 become

$$[S + u_j I]Y_{j-1/2} = EF^\top + Y_{j-1}[u_j I - T], \tag{16.1}$$

$$Y_j[T + v_j I] = EF^\top + [v_j I - S]Y_{j-1/2}. \tag{16.2}$$

The recursion formulas for the $U_i(j)$ are

$$U_i(j) = R_j U_i(j-1), \qquad i = 1, 2, \ldots, j-1, \tag{17.1}$$

$$U_j(j) = (u_j + v_j)[u_j I + S]^{-1} E, \tag{17.2}$$

and

$$V_i(j) = Q_j^* V_i(j-1), \qquad i = 1, 2, \ldots, j-1, \tag{18.1}$$

$$V_j(j) = [v_j^* I + T']^{-1} F. \tag{18.2}$$

The Li-White algorithm may be introduced to reduce iteration complexity.

5.7 Approximating a Full Matrix by a Sum of Low-Rank Matrices

The analysis applies when the right-hand side of the reduced Lyapunov equation (Eq. 1) is of the form $H = FPF^\top$ with P of order $r \times r$ or the r.h.s. of the reduced Sylvester equation (Eq. 12.1) is $EPF^\top$. The matrix P does not affect the ADI iteration equations. Significant reduction in computation may be realized if the r.h.s. can be approximated reasonably well in this form. A review of matrix factorization with discussion of low-rank approximation is given in [Hubert et al., 2000]. Penzl observed that "splitting up the right hand side matrix into a sum of low-rank matrices enables an efficient parallelization of [his] method."

This suggests a general procedure for solving all N-stable Lyapunov problems. Let the $n \times m$ matrix of orthonormal vectors of m Lanczos steps applied to matrix H be K and let the Lanczos coefficients determine the tridiagonal matrix T of order m. Then the matrix $F_1 = KTK^\top$ is a rank m approximation to H. We may, therefore, compute $H_1 = H - F_1$ and perform m Lanczos steps on H_1 with initial vector equal to what would have been the $m+1$ vector of the previous Lanczos steps on H. This may be continued to yield a set of F_j. The norms of successive H_j should decrease and the algorithm may be terminated when sufficient accuracy is achieved with the sum of the low-rank approximations. If H is of full rank with $n = 100$ and m is 5, for example, the algorithm should terminate when j is around 20. The matrix H_{21} will in general not be the zero matrix since the Lanczos vectors from each H_j are not orthogonal to the previous vectors. When the sum of the low-rank subspaces approaches H, the rank of subsequent subspaces may be smaller than m.

In a pilot MATLAB program, when the absolute value of the $(k, k+1)$ element of T (for $k < m$) was less than 0.001 times that of element (1,2), the rank of the subspace was chosen as k. The algorithm was terminated when the order of T was one. Having generated a set of low-rank matrices, one may solve the low-rank Lyapunov systems in parallel. In this application, the matrix A and its transformation to S is common to all low-rank problems. The reduction, spectrum evaluation, ADI iteration parameter determination, and back transformation need only be done once. This approach extends application of the Li–White algorithm to general right-hand sides. When H is not given in factored form but is of rank $r << n$ the Lanczos partitioning will expose the rank deficiency of H and may lead to more efficient solution.

A set of test problems was considered with the matrix B in Eq. 12.1 chosen as $A + A'$. This B is symmetric but not necessarily SPD. In general, the unique solution X need not be SPD when B is not SPD. However, X is the identity matrix for this choice of B. It should be noted that the tridiagonal matrices T need not be SPD since only K appears as a rhs for the ADI iterations.

One test case was run with a random N-stable A of order 30 for which B (and hence the rhs H of the transformed equation) was not SPD. The initial value for m was chosen as 5. The first seven subspaces were of rank 5, but the eighth was of rank 1. The sum of the low-rank solutions agreed with the true solution to four significant places. Another problem was solved with A of order 100 and $m = 10$. The first 11 subspaces were of rank 10 and the 12th was of rank 1. The factored Y agreed with the nonfactored Y to four significant figures. Comparable accuracy was obtained with 14 subspaces of rank $m = 8$ and a 15th of rank 1. For this problem, an initial choice of $m = 12$ resulted in nine subspaces of rank 12, a tenth of rank 5, and a last subspace of rank 1. The factored result agreed with the nonfactored result to five significant places. To illustrate how the Lanczos algorithm exposes rank deficiency of a given full matrix, a random full matrix of order 100 and rank 20 was chosen as B. The value for m was chosen as 5. The algorithm terminated with four subspaces of rank 5, one of rank 4, and the last of rank 1.

An in-depth comparison of computation time for various approaches should be made. Many stages are easily parallelized. Reduction to banded upper Hessenberg form, the Lanczos algorithm, GMRES-type solution of the ADI iteration equations with sparse A, and simultaneous solution for all low-rank matrices generated by the Lanczos algorithm applied to a full-rank system are among these stages. It should be noted that the ADI iteration equations may be solved for all columns of X or Y simultaneously once the right-hand sides of the equations are computed. However, computation of these right-hand sides for a full rank H of order n requires a factor of n/m times computation with the $n \times m$ factor K in the Li–White algorithm. When the coefficient matrix A is sparse, solution without reduction may be best, provided one can determine good ADI iteration parameters. However, when A is not sparse, the GMRES approach may become less efficient even with good ADI parameters. Optimization and comparison of relative efficiency of the various methods is a fertile area for further research.

5.8 Summary

The ADI iteration originally proposed by Peaceman and Rachford in 1955 for numerical solution of difference equations for elliptic partial differential equations has been widely used for solving such problems. Analysis is less precise when the coefficient matrix is split into noncommuting components for the iteration. Almost thirty years after inception of ADI iteration, it was recognized that Lyapunov and Sylvester matrix equations have the commutation property. This led to generalization of the ADI iteration theory from real to complex spectra with a rather elegant application of the theory of modular transformations of elliptic functions. It also stimulated research into similarity reduction of a full nonsymmetric real matrix to sparse form and in particular to banded upper Hessenberg form. Despite the rapid convergence of ADI iteration for these problems, earlier methods of Bartels–Stewart and Smith remained competitive and were already incorporated in major software packages. After another twenty years had elapsed, it was observed by Penzl that low-rank equations could be treated by a low-rank ADI iteration more efficiently than by conventional methods which have not been shown to admit significant gains in efficiency for low-rank problems. The subsequent contribution by Li and White further reduced computational effort so that ADI iteration is clearly superior to the other methods for such problems. Relative merits of iterating with the full system and iterating after reduction to banded Hessenberg form are still under consideration and may well depend on specific applications.

The possibility of approximating a general right-hand side by a sum of low-rank matrices extends application of the Li–White algorithm to all N-stable Lyapunov systems. The Lanczos algorithm has worked well for generating a sum of low-rank approximations in the few test problems thus far considered. In general, ADI solution of Lyapunov equations by any of the methods discussed is well suited for parallel computation.