1 SUMMARY

Given a **real unsymmetric** $n \times n$ **matrix** $\mathbf{A} = \{a_{ij}\}$, this routine uses subspace iteration to **calculate the** r **eigenvalues** λ_i**,** $i = 1, 2, ..., r$**, that are right-most, left-most, or are of largest modulus.** The right-most (respectively, left-most) eigenvalues are the eigenvalues with the most positive (respectively, negative) real part. A second entry will return the associated eigenvectors $\mathbf{y}_i$, $i = 1, 2, ..., r$, where $\mathbf{A}\mathbf{y}_i = \lambda_i \mathbf{y}_i$. The routine may also be used to calculate a group of eigensolutions elsewhere in the spectrum (see Section 2.7).

ATTRIBUTES — **Version:** 1.2.2. (22nd February 2023) **Types:** Real (single, double). **Calls:** `FA14`, `FD15`, `KB06`, `_COPY`, `_DOT`, `_GEMM`, `_GEMV`, `_GER`, `_NRM2`, `_SCAL`. **Language:** Fortran 77. **Original date:** April 2001. **Remark:** The `EB22` entries are threadsafe versions of `EB12`. **Origin:** I.S. Duff and J.A. Scott, Rutherford Appleton Laboratory.

2 HOW TO USE THE PACKAGE

2.1 Overall control

There are three entries:

(a) `EB22I/ID` must be called first to perform initialization.

(b) `EB22A/AD` calculates the selected eigenvalues.

(c) `EB22B/BD` uses the basis vectors computed by `EB22A/AD` to calculate the eigenvectors corresponding to the selected eigenvalues. Optionally the scaled eigenvector residuals

$$\frac{\|(\mathbf{A}\mathbf{y}_i - \lambda_i \mathbf{y}_i)\|_2}{\|(\mathbf{A}\mathbf{y}_i)\|_2}, \quad 1 \le i \le r, \tag{1}$$

are computed.

The algorithm used by `EB22A/AD` and `EB22B/BD` requires the multiplication of sets of vectors by the matrix **A**. The user does not have to pass the matrix **A** to `EB22A/AD` or `EB22B/BD` but, each time a matrix-matrix multiplication **AW** is required, control is returned to the user. A simple example to illustrate the calling sequence is given in Section 5.

2.2 Argument lists

Initialization

The `EB22I/ID` entry must be called prior to the first call to the `EB22A/AD` entry to initialize the control array and private workspace.

The single precision version

```
CALL EB22I(ICNTL,KEEP,RKEEP)
```

The double precision version

```
CALL  EB22ID(ICNTL,KEEP,RKEEP)
```

`ICNTL` is an `INTEGER` array of length 10, see Section 2.3.

`KEEP` is an `INTEGER` array of length 25 used by `EB22` as private workspace and should not be altered by the user. The element `KEEP(22)` holds the random number seed used by `FA14`, which is used by `EB22` to generate a random start vector. In exceptional circumstances the user may wish to reset it after the `EB22I/ID` call to to obtain a different random sequence.

RKEEP is a REAL (DOUBLE PRECISION in the D version) array of length 20 used by EB22 as private workspace and should not be altered by the user.

To calculate the selected eigenvalues

The single precision version

```
      CALL EB22A(IND,N,NUMEIG,NUMCOL,NLOW,NUP,EPS,X,U,W,LN,TW,
     +    LTW,IW,ER,EI,IPOS,IFLAG,ICNTL,INFO,RINFO,KEEP,RKEEP)
```

The double precision version

```
      CALL EB22AD(IND,N,NUMEIG,NUMCOL,NLOW,NUP,EPS,X,U,W,LN,TW,
     +     LTW,IW,ER,EI,IPOS,IFLAG,ICNTL,INFO,RINFO,KEEP,RKEEP)
```

IND is an INTEGER variable which must be set by the user. If $\texttt{IND} \geq 1$ the right-most eigenvalues are computed; if $\texttt{IND} \leq -1$ the left-most eigenvalues are computed; and if $\texttt{IND} = 0$ the eigenvalues of largest modulus are computed. If the user knows the eigenvalues of largest modulus are the right-most (respectively, left-most) eigenvalues, the user should set $\texttt{IND} \geq 1$ (respectively, $\texttt{IND} \leq -1$), rather then $\texttt{IND} = 0$. This argument is not altered by the routine.

N is an INTEGER variable which must be set by the user to n, the order of the matrix. This variable must be preserved by the user between calls to EB22A/AD and between a call to EB22A/AD and EB22B/BD. This argument is not altered by the routine. **Restriction**: $\texttt{N} \geq 1$.

NUMEIG is an INTEGER variable which must be set by the user to r, the number of eigenvalues required. This argument is not altered by the routine. **Restriction**: $1 \leq \texttt{NUMEIG} \leq \texttt{N}$.

NUMCOL is an INTEGER variable which must be set by the user to the number of trial vectors on which subspace iteration will be performed. Information concerning a suitable value for NUMCOL is given in Section 2.6. This variable must be preserved by the user between calls to EB22A/AD and between a call to EB22A/AD and EB22B/BD. This argument is not altered by the routine. **Restriction**: $\min(\texttt{NUMEIG+1}, \texttt{N}) \leq \texttt{NUMCOL} \leq \texttt{N}$.

NLOW and NUP are INTEGER variables which need not be set by the user. On each exit with a value of IPOS greater than 0, the user must multiply columns NLOW, NLOW+1, ..., NUP of the array W by the matrix **A** and place the results in the corresponding columns of the array U. These arguments must not be altered by the user between calls to EB22A/AD.

EPS is a REAL (DOUBLE PRECISION in the D version) array of length 2. EPS(1) is a convergence parameter which must be set by the user but EPS(2) need not be set by the user. When the user calls EB22A/AD for the first time, the code gives EPS(2) the same value as EPS(1), unless EPS(1) is supplied outside the range $(u, 1.0)$ (where u is the machine precision), in which case EPS(2) is given the default value $\sqrt{u}$. EPS(2) is used to determine if the computed eigenvalues are sufficiently accurate (see Section 4 for more details). If the initial value of EPS(2) demands an accuracy greater than can be achieved or convergence becomes very slow, the EPS(2) is increased so that on exit with $\texttt{IPOS} = 0$, EPS(2) is set to the actual convergence parameter which has been used by EB22A/AD. EPS(2) must not be altered by the user between calls to EB22A/AD. EPS(1) is not altered by the routine.

X is a REAL (DOUBLE PRECISION in the D version) two-dimensional array with dimensions (LN, NUMCOL). If the user wishes to supply an initial estimate of the NUMEIG basis vectors which span the invariant subspace corresponding to the sought-after eigenvalues of **A**, they should be input (in any order) in the first NUMEIG columns of X (and IPOS set to –1). On exit with $\texttt{IPOS} = 0$, the first NUMEIG columns of X hold in order orthonormalized basis vectors which span the subspace corresponding to the NUMEIG sought-after eigenvalues of **A**, while the remaining columns hold approximations to other basis vectors. This array must not be altered by the user between calls to EB22A/AD and between a call to EB22A/AD and EB22B/BD.

U is a REAL (DOUBLE PRECISION in the D version) two-dimensional array with dimensions (LN, NUMCOL), which is used as workspace by EB22A/AD. On each exit with a value of IPOS greater than 0, columns NLOW to NUP of

this array must be set by the user to hold the results of multiplying the corresponding columns of W by **A**. The user must then recall EB22A/AD without altering the remaining columns of U.

W is a REAL (DOUBLE PRECISION in the D version) two-dimensional array with dimensions (LN, NUMCOL), which is used as workspace by EB22A/AD. On each exit with a value of IPOS greater than 0, the user must multiply columns NLOW to NUP of W by **A**, place the results in the corresponding columns of U, and recall EB22A/AD. W must not be altered by the user between calls to EB22A/AD.

LN is an INTEGER variable which must be set by the user to the first dimension of the arrays X, U, and W. This argument is not altered by the routine. **Restriction:** LN ≥ N.

TW is a REAL (DOUBLE PRECISION in the D version) array of length LTW which is used as workspace by EB22A/AD. This array must not be altered by the user between calls to EB22A/AD, and the first NUMCOL*NUMCOL entries must not be altered between a call to EB22A/AD and EB22B/BD.

LTW is an INTEGER variable which must be set by the user to the length of array TW. This argument is not altered by the routine. **Restrictions:** If IND = 0, LTW ≥ NUMCOL*(2 + NUMCOL + 3) + 5; otherwise, LTW ≥ NUMCOL*(2 + NUMCOL + 10) + 4.

IW is an INTEGER array of length NUMCOL which is used as workspace by EB22A/AD. This array must not be altered by the user between calls to EB22A/AD.

ER is a REAL (DOUBLE PRECISION in the D version) array of length NUMCOL. On each exit with a positive value of IPOS, the first NUMEIG entries of ER hold in order approximations to the real parts of the sought-after eigenvalues of **A**, while the rest of the array contains estimates of the real parts of other eigenvalues. This array must not be altered by the user between calls to EB22A/AD and between a call to EB22A/AD and EB22B/BD.

EI is a REAL (DOUBLE PRECISION in the D version) array of length NUMCOL. On each exit with a positive value of IPOS, the first NUMEIG entries of EI hold in order approximations to the imaginary parts of the sought-after eigenvalues of **A**, while the rest of the array contains estimates of the imaginary parts of other eigenvalues. Complex conjugate pairs of eigenvalues appear consecutively with the eigenvalue with positive imaginary part appearing first. This array must not be altered by the user between calls to EB22A/AD and between a call to EB22A/AD and EB22B/BD.

IPOS is an INTEGER variable used to control the subroutine (see Section 2.1). Prior to the first call to EB22A/AD, IPOS must be set to 0 if the user does not wish to provide an initial estimate of the NUMEIG basis vectors which span the invariant subspace corresponding to the required eigenvalues and to –1 if the user does wish to provide an initial estimate. On exit, a value of IPOS greater than 0 indicates further iterations are to be performed; the user must multiply columns NLOW, NLOW+1, . . . , NUP of the array W by **A**, place the results in the corresponding columns of U, and recall EB22A/AD, without altering the value of IPOS. A value of IPOS equal to 0 on exit indicates that the calculation is complete.

IFLAG is an INTEGER variable which is only used when restarting the computation, see Section 2.5. In other cases it need not be set by the user and on exit from EB22A/AD it will have the same value as the error and warning indicator INFO(1), see Section 2.3.

ICNTL is an INTEGER array of length 10 whose first three elements can optionally be set by the user, see Section 2.3

INFO is an INTEGER array of length 20 used to return integer information to the user, see Section 2.3.

RINFO is an REAL array of length 10 used to return floating point information to the user, see Section 2.3.

KEEP is an INTEGER array of length 25 used by EB22 as private workspace, see description of EB22I/ID.

RKEEP is a REAL (DOUBLE PRECISION in the D version) array of length 20 used by EB22 as private workspace, see description of EB22I/ID.

To compute the eigenvectors using the basis vectors from EB22A/AD.

The single precision version

```
CALL EB22B(N,NUMCOL,ER,EI,Y,X,U,W,LN,TW,RES,IPOS,INFO,
           KEEP,RKEEP)
```

The double precision version

```
CALL EB22BD(N,NUMCOL,ER,EI,Y,X,U,W,LN,TW,RES,IPOS,INFO,
            KEEP,RKEEP)
```

N is an INTEGER variable which must be unchanged since the call to EB22A/AD that returned IPOS = 0. This argument is not altered by the routine.

NUMCOL is an INTEGER variable which must be unchanged since the call to EB22A/AD that returned IPOS = 0. This argument is not altered by the routine.

ER is a REAL (DOUBLE PRECISION in the D version) array of length NUMCOL which must be unchanged since the call to EB22A/AD that returned IPOS = 0. This argument is not altered by the routine.

EI is a REAL (DOUBLE PRECISION in the D version) array of length NUMCOL which must be unchanged since the call to EB22A/AD that returned IPOS = 0. This argument is not altered by the routine.

Y is a REAL (DOUBLE PRECISION in the D version) two-dimensional array with dimensions (LN, NUMCOL) which need not be set by the user. On exit, if the I-th computed eigenvalue is real (EI(I) = 0), the I-th column of Y contains the corresponding eigenvector; if the I-th computed eigenvalue is complex with positive imaginary part (EI(I) > 0), the I-th and (I+1)-th columns of Y contain, respectively, the real and imaginary parts of the corresponding eigenvector. This array must not be altered by the user between calls to EB22B/BD.

X is a REAL (DOUBLE PRECISION in the D version) two-dimensional array with dimensions (LN, NUMCOL). This array must be unchanged since the call to EB22A/AD that returned IPOS = 0. This argument is not altered by the routine.

U is a REAL (DOUBLE PRECISION in the D version) two-dimensional array with dimensions (LN, NUMCOL), which is used as workspace by EB22B/BD. On exit with IPOS equal to 1, if the user wishes to calculate the scaled eigenvector residuals

$$\frac{\|(\mathbf{A}\mathbf{y}_i - \lambda_i \mathbf{y}_i)\|_2}{\|(\mathbf{A}\mathbf{y}_i)\|_2},$$

this array must be set by the user to hold the results of multiplying the columns of W by **A**. The user must then recall EB22B/BD.

W is a REAL (DOUBLE PRECISION in the D version) two-dimensional array with dimensions (LN, NUMCOL), which is used as workspace by EB22B/BD. If the user wants to compute the scaled eigenvector residuals then, on exit with IPOS equal to 1, the user must multiply columns 1 to NUMCOL of W by **A**, place the result in U, and recall EB22B/BD. W must not be altered by the user between calls to EB22B/BD.

LN is an INTEGER variable which must be set by the user to the first dimension of the arrays Y, X, U, and W. LN must be unchanged since the call to EB22A/AD that returned IPOS = 0. This argument is not altered by the routine.

TW is a REAL (DOUBLE PRECISION in the D version) array of length at least NUMCOL*NUMCOL which is used as workspace by EB22B/BD. The first NUMCOL*NUMCOL entries must be unchanged since the call to EB22A/AD that returned IPOS = 0.

RES is a REAL (DOUBLE PRECISION in the D version) array of length NUMCOL. On exit with IPOS equal to 0, either RES(I) contains the scaled residual for the I-th computed eigenvector or if, for some i, $1 \le i \le$ NUMCOL, $\|(\mathbf{A}\mathbf{y}_i)\|_2 \le \sqrt{u}$, where u is the machine precision, then RES(I), I = i, is set equal to zero.

IPOS is an INTEGER variable used to control the subroutine (see Section 2.1). Prior to the first call to EB22B/BD,

IPOS must be set to 0. On exit from the first call to EB22B/BD, a value of IPOS equal to 1 is returned, which indicates that the eigenvectors have been computed. If the user wishes to calculate the scaled eigenvector residuals, EB22B/BD must be called again without altering IPOS. On exit from the second call to EB22B/BD, a value of IPOS equal to 0 indicates the scaled eigenvector residuals have been successfully computed.

INFO is an INTEGER array of length 20 used to return integer information to the user, see Section 2.3.

KEEP is an INTEGER array of length 25 used by EB22 as private workspace and must not be altered by the user.

RKEEP is a REAL (DOUBLE PRECISION in the D version) array of length 20 used by EB22 as private workspace and must not be altered by the user.

2.3 The control and information arrays

The ICNTL array argument which must be of length 10 can be used to pass optional control values to the routine.

ICNTL(1) specifies the unit number to be used to output error messages, see INFO below. It has a default value of 6 which can be reset by the user to another unit or set to a negative value if error messages are to be suppressed.

ICNTL(2) specifies the unit number to be used to output warning messages, see INFO below. It has a default value of 6 which can be reset by the user to another unit or set to a negative value if warning messages are to be suppressed.

ICNTL(3) can be set to the maximum number of matrix-vector multiplications the user wishes to allow. A matrix-matrix multiplication **AW** where **W** is an N$\times$NUMCOL matrix is taken to be NUMCOL matrix-vector multiplications. This parameter ensures finite termination of the code. The default value is 10^4.

ICNTL(4) to ICNTL(10) should not be altered and at present are not used by EB22.

The INFO array argument which must be of length 20 is used to return integer information to the user.

INFO(1) is used as an error indicator. It is set to zero when there are no errors or warnings; a negative value indicates an error has occurred during the last call to EB22A/AD; and a value greater than 0 indicates a warning has been issued during one (or more) of the calls to EB22A/AD. Details of error and warning messages are given in Section 2.4.

INFO(2) is set to supplementary information to the error condition, see Section 2.4.

INFO(3) On each exit from EB22A/AD, this will be set to the number of eigenvalues found so far. This may equal NUMEIG+1 because complex eigenvalues are found as a complex conjugate pair.

INFO(4) On each exit from EB22A/AD, this is set to the maximum degree of the iteration polynomial used by the algorithm to date.

INFO(5) On each exit from EB22A/AD, this is set to the current degree of the iteration polynomial used by the algorithm.

INFO(6) On each exit from EB22A/AD, this is set to the total number of matrix-vector products used by the routine so far.

INFO(7) On exit from EB22B/BD with IPOS = 0, this is set to the number of computed eigenvectors $\mathbf{y}_i$, $1 \le i \le$ NUMCOL, for which $\|(\mathbf{A}\mathbf{y}_i)\|_2 \le \sqrt{u}$, where u is the machine precision. A nonzero value for INFO(7) would occur if, for example, one of the computed eigenvalues of **A** was equal to zero.

INFO(8) to INFO(20) are not used at present by EB22.

The RINFO array argument which must be of length 10 is used to return floating point information to the user.

RINFO(1), RINFO(2) and RINFO(3) are returned set to the values of D, A2, and C2 respectively which are the values used when IND$\neq$0. They are the current ellipse parameters (the ellipse has centre D, foci D+C, D–C, and crosses the real axis at D+A, D–A, where C$=\sqrt{\text{C2}}$ and A$=\sqrt{\text{A2}}$).

RINFO(4) to INFO(10) are not used at present by EB22.

2.4 Error diagnostics

If EB22A/AD returns with a negative value of INFO(1) an error has occurred; if EB22A/AD returns with a positive value of INFO(1) a warning has been issued. There are no error or warning returns from EB22B/BD. Error messages are output on unit ICNTL(1) and warnings on unit ICNTL(2). In the event of an error, further information is output in INFO(2). Possible non-zero values of INFO(1) are given below.

–1 Value of N out of range. N < 1. Immediate return with input parameters unchanged with INFO(2) set to N.

–2 Value of NUMEIG out of range. Either NUMEIG > N or NUMEIG < 1. Immediate return with input parameters unchanged with INFO(2) set to NUMEIG.

–3 Value of NUMCOL out of range. Either NUMCOL > N or NUMCOL < min(NUMEIG+1, N). Immediate return with input parameters unchanged with INFO(2) set to NUMCOL.

–4 Value of LN out of range. LN < N. Immediate return with input parameters unchanged with INFO(2) set to LN.

–5 Value of LTW out of range. If IND = 0, LTW < NUMCOL*(N + NUMCOL + 3) + 5; otherwise LTW < NUMCOL*(N + NUMCOL + 10) + 4. Immediate return with input parameters unchanged with INFO(2) set to LTW.

–6 Algorithm required more matrix-vector multiplications than specified by ICNTL(3). The user may increase either ICNTL(3) or NUMCOL and restart the computation; full details are given in Section 2.5. INFO(2) is returned set to the value of ICNTL(3).

–7 This value of ICNTL(1) can only be returned if IND ≠ 0. The algorithm was unable to find all NUMEIG eigenvalues requested by the user with the user-supplied number (NUMCOL) of trial vectors because at some stage there were no points on which to construct the Chebychev ellipse (see Section 4). This could happen if, for example, the matrix **A** has more than one eigenvalue (or more than one complex conjugate pair of eigenvalues) with real part equal to the real part of one or more of the requested eigenvalues. The code will return with the number of successfully computed eigenvalues in INFO(3) (see Section 2.3) and the converged basis vectors in the first INFO(3) columns of the array X. To compute all NUMEIG requested eigenvalues the user is advised to increase NUMCOL and restart the computation (see Section 2.5). INFO(2) is returned set to NUMEIG.

–8 There is an infinitesimal chance of a catastrophic failure causing termination of the algorithm (see Section 4). This failure is indicated by the message

```
UNEXPECTED ERROR IN EB22A/AD
```

on unit ICNTL(1) and EB22A/AD returns with a value of INFO(1) equal to –8 and INFO(2) equal to zero. In this case the user should contact the Numerical Analysis Group at the Rutherford Appleton Laboratory for assistance.

+1 The user-supplied value of EPS(1) lies outside the interval $(u, 1.0)$. The default convergence parameter $\sqrt{u}$ will be used.

+2 The accuracy requested by the user (or the default accuracy if the user-supplied value of EPS(1) was out of range) was not achieved. The code increases EPS(2) and, on exit with IPOS = 0, EPS(2) will be set to the actual convergence parameter used by EB22A/AD. Greater accuracy can often be achieved by increasing the value of NUMCOL and restarting the computation (see Section 2.5).

+3 both the conditions in warnings +1 and +2 apply.

2.5 Restarting the computation

In certain circumstances, the user may wish to restart the computation after an error or warning has been issued. If INFO(1) = −6 on exit from EB22A/AD, the algorithm has failed to converge in the number of matrix-vector multiplications specified by ICNTL(3). To restart the computation from the point at which INFO(1) = −6 was returned, the user should increase ICNTL(3), set IFLAG = 0, multiply columns NLOW to NUP of W by **A**, place the

results in the corresponding columns of U, and recall EB22A/AD without making any other changes to the input parameters. However, it may be more efficient to start the computation again with an increased value for NUMCOL and IPOS set to –1.

If INFO(1) = –7 on exit from EB22A/AD, the user may restart the computation taking advantage of the basis vectors already found by increasing NUMCOL, setting IPOS = –1, EPS(1) = EPS(2), and recalling EB22A/AD.

The user may also wish to restart the computation after EB22A/AD has returned with INFO(1) = 2 and IPOS = 0. In this case the algorithm has successfully converged but not with the requested convergence tolerance. The required accuracy can often be achieved by increasing NUMCOL (and optionally increasing NUMEIG), setting IPOS = –1, and recalling EB22A/AD.

For some problems, if the user requires r eigenvalues, it can be advantageous to choose NUMEIG to be greater than r. On each exit with positive IPOS, INFO(3) is set to the number of eigenvalues found so far (see Section 2.3). The user is therefore able to monitor when r eigenvalues have been found. Once EB22A/AD returns with INFO(3) $\geq r$ and a positive value of IPOS, the user should set NUMEIG = r, IPOS = –1, EPS(1) = EPS(2), and recall EB22A/AD.

2.6 Choosing a value for NUMCOL

NUMCOL is an INTEGER variable which must be set by the user to the number of vectors on which subspace iteration will be performed. The amount of storage required by the code and the number of matrix-vector multiplications on each iteration depends upon NUMCOL, which implies NUMCOL should not be chosen unnecessarily large. But if NUMCOL is too small, the number of iterations required for convergence may be high. Since we need to obtain more eigenvalues than the NUMEIG sought-after ones, NUMCOL must exceed NUMEIG. If IND = 0 it must be at least min(NUMEIG+1, N); otherwise, it must normally be at least min(NUMEIG+2, N), but if the NUMEIG+1 right-most (or left-most) eigenvalues are known to be real, it may equal NUMEIG+1. However, it is advisable to choose NUMCOL to be larger than the minimum required value. The user should try choosing NUMCOL to be about 2*NUMEIG, although for a given value of NUMEIG, the best value for NUMCOL is problem-dependent.

2.7 Use of EB22A/AD to obtain other parts of the spectrum

Subroutine EB22A/AD can be used to find other parts of the spectrum than that corresponding to the eigenvalues of largest modulus or the right-most (or left-most) eigenvalues of **A**. If, for example, we wish to compute a group of interior eigenvalues, say those closest to p, then **A** is replaced by $(\mathbf{A}-p\,\mathbf{I})^{-1}$ and at each iteration the matrix-matrix multiplication $\mathbf{U}=(\mathbf{A}-p\,\mathbf{I})^{-1}\mathbf{W}$ is required. This is equivalent to solving the system

$$(\mathbf{A}-p\,\mathbf{I})\,\mathbf{U}=\mathbf{W}. \qquad (2)$$

If **A** is large the solution of the system (2) may itself be quite time-consuming but note that, if a direct method of solution is used, the decomposition of $\mathbf{A}-p\,\mathbf{I}$ into triangular factors needs only to be done once for the entire calculation, the only operations in the inner loop being the relatively cheap forward and backward substitutions.

2.8 Underflows

The nature of the calculations performed in this subroutine means that underflows are likely to occur. It is quite safe to set numbers that underflow to zero, and action by the user may be required to ensure that this is done efficiently by the computing system in use.

3 GENERAL INFORMATION

Use of common: none.

Other routines called directly: internal to the package: EB22E/ED, EB22F/FD, EB22G/GD, EB22H/HD, EB22J/JD, EB22K/KD, EB22L/LD, EB22M/MD, EB22N/ND, EB22O/OD, EB22P/PD, EB22Q/QD, EB22R/RD, EB22S/SD, EB22T/TD, EB22U/UD, EB22V/VD, EB22W/WD, EB22X/XD, EB22Y/YD, and EB22Z/ZD. Outside the package: FA14A/AD, FD15A/AD, KB06A/AD, SCOPY/DCOPY, SDOT/DDOT, SGEMM/DGEMM, SGEMV/DGEMV,

SGER/DGER, SNRM2/DNRM2 and SSCAL/DSCAL.

Input/output: error messages are printed on unit ICNTL(1) and warnings on unit ICNTL(2); see Section 2.4.

Restrictions:

N ≥ 1,
1 ≤ NUMEIG ≤ N,
min(NUMEIG+1, N) ≤ NUMCOL ≤ N.
If IND = 0 LTW ≥ NUMCOL*(2 + NUMCOL + 3) + 5;
otherwise LTW ≥ NUMCOL*(2 + NUMCOL + 10) + 4.

4 METHOD

EB22A/AD

EB22A/AD uses a subspace iteration algorithm, combined with Chebychev acceleration if the right-most (or left-most) eigenvalues are required, or if the eigenvalues of largest modulus are known to be the right-most (or left-most) eigenvalues.

The code first checks the input data. If a fatal error is found, IFLAG is given a negative value and control is returned to the user. If no errors are encountered, the convergence parameter EPS(2) is set. If the user has not supplied an initial estimate for the NUMEIG basis vectors spanning the invariant subspace corresponding to the sought-after eigenvalues, a random vector $\mathbf{x}_1$ is generated using FA14A, IPOS is given a positive value, and control is passed to the user for the matrix-vector multiplication $\mathbf{A}\mathbf{x}_1$. EB22A/AD must then be recalled. $\mathbf{A}\mathbf{x}_1$ is orthonormalised with respect to $\mathbf{x}_1$ to give $\mathbf{x}_2$. The process is repeated to yield an orthonormal set of vectors $\mathbf{x}_1$, $\mathbf{x}_2$,..., $\mathbf{x}_{\texttt{NUMCOL}}$. If the user has provided initial estimates of the NUMEIG basis vectors corresponding to the sought-after eigenvalues, NUMCOL-NUMEIG random vectors are generated and the resulting set of NUMCOL vectors are orthonormalised. The algorithm used for orthonormalising a set of vectors is the modified Gram-Schmidt algorithm (see Golub and van Loan (1989)). If the code fails to generate a set of NUMCOL orthonormal vectors, IFLAG = 8 is returned to the user and the computation is terminated.

The subspace iteration algorithm used by EB22A/AD then proceeds as follows:

1. *Start:* Let $\mathbf{X}$ be the matrix with columns $(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_{\texttt{NUMCOL}})$. Set $l = 1$ and $p_l(\lambda)$=1.
2. *Iteration:* Compute $\mathbf{X} \Leftarrow p_l(\mathbf{A})\mathbf{X}$, with control returned to the user for each premultiplication by $\mathbf{A}$.
3. *Projection:* Orthonormalise $\mathbf{X}$ using the modified Gram-Schmidt algorithm. Compute $\mathbf{B} = \mathbf{X}^{\mathrm{T}}\mathbf{A}\mathbf{X}$, with control returned to the user for the matrix-matrix multiplication $\mathbf{A}\mathbf{X}$. Reduce $\mathbf{B}$ to upper Hessenberg form $\mathbf{H}$ using a modified version of the EISPACK routines ORTHES and ORTRAN. Reduce $\mathbf{H}$ to the real Schur form $\mathbf{T} = \mathbf{Z}^{\mathrm{T}}\mathbf{B}\mathbf{Z}$ using a modified version of the algorithm HQR3 given by Stewart (1976). Each diagonal block $\mathbf{T}_{ii}$ is either of order 1 or is a 2×2 matrix having complex conjugate eigenvalues, with the eigenvalues ordered along the diagonal blocks. Set $\mathbf{X} \Leftarrow \mathbf{X}\mathbf{Z}$.
4. *Convergence test:* If $\mathbf{X}$ satisfies the convergence criterion

   $$\|(\mathbf{A}\mathbf{X} - \mathbf{X}\mathbf{T})_j\|_2 \leq \texttt{EPS(2)} * \|(\mathbf{A}\mathbf{X})_j\|_2, \quad 1 \leq j \leq \texttt{NUMEIG},$$

   then **stop**, else determine the degree l of the polynomial $p_l(\lambda)$ for the next iteration. If the eigenvalues of largest modulus are the right-most (or left-most) eigenvalues, or if the right-most (or left-most) eigenvalues are required, find the ellipse parameters d, c for the Chebychev ellipse containing the unwanted eigenvalues, update the real reference point γ which approximates $\lambda_{\texttt{NUMEIG}}$, and set $p_l(\lambda) = \dfrac{T_l[(\lambda - d)/c]}{T_l[(\gamma - d)/c]}$, where T_l is the Chebychev polynomial of degree l of the first kind. Otherwise, set $p_l(\lambda) = \lambda^l$.
   Go to 2.

Construction of the Chebychev ellipse uses techniques described by Saad (1984), and is implemented in EB22A/AD using modified versions of routines by Manteuffel (1975).

EB22A/AD uses an implicit deflation technique which consists of working only with the basis vectors which have not yet converged, thus locking those which have converged. This reduces the number of matrix-vector multiplications required by the routine when more than one eigenvalue (or more than one complex conjugate pair of eigenvalues) is required. Once the first estimate of the sought-after eigenvalues has been computed, the parameters NLOW and NUP indicate those basis vectors which have not converged.

EB22B/BD

Once EB22A/AD has terminated successfully, EB22B/BD may be called to compute the eigenvectors corresponding to the converged eigenvalues. EB22B/BD computes the eigenvectors $\mathbf{V}$ of the real Schur form $\mathbf{T}=\mathbf{X}^{\mathrm{T}}\mathbf{AX}$ using back substitution and then takes $\mathbf{y}_i=(\mathbf{AV})_i$ to be the approximate eigenvector of $\mathbf{A}$ corresponding to λ_i. If the user wishes to know the scaled eigenvector residuals

$$\frac{\|(\mathbf{A}\mathbf{y}_i-\lambda_i\mathbf{y}_i)\|_2}{\|(\mathbf{A}\mathbf{y}_i)\|_2},$$

the user must compute $\mathbf{AY}$, where $\mathbf{Y}$ has columns $\mathbf{y}_1,\ldots,\mathbf{y}_{\mathtt{NUMCOL}}$ and recall EB22B/BD.

Full details of the algorithm used by EB22A/AD and EB22B/BD are given in Duff and Scott (1991).

References

Duff, I. S. and Scott, J. A. (1993). Computing selected eigenvalues of sparse unsymmetric matrices using subspace iteration. *ACM Trans. Math. Softw.* **19,** 137-159.

Golub, G. H. and van Loan, C. F. (1989). Matrix Computations. Second edition. Johns Hopkins University Press.

Manteuffel, T. A. (1975). An iterative method for solving nonsymmetric linear systems with dynamic estimation of parameters. Ph.D. dissertation, Tech. Report UIUCDCS-75-758. Univ. of Illinois.

Saad, Y. (1984). Chebychev acceleration techniques for solving nonsymmetric eigenvalue problems. *Math. Comp.* **42**, 567-588.

5 EXAMPLE OF USE

The following program illustrates the use of EB22A/AD and EB22B/BD. We wish to calculate the right-most eigenvalue and corresponding eigenvector of the following matrix

$$\mathbf{A}=\begin{pmatrix}1&0&0&0&2\\3&2&0&0&0\\0&0&0&1&4\\0&0&3&1&0\\0&0&0&0&5\end{pmatrix}.$$

```
C  Code to illustrate the use of EB22AD and EB22BD
C
C     .. Parameters ..
      INTEGER NMAX,NZMAX,LTWMAX
      PARAMETER (NMAX=5,NZMAX=9,LTWMAX=80)
C     ..
C     .. Local Scalars ..
      INTEGER I,IFLAG,IND,IPOS,LN,LTW,N,NLOW,NUMCOL,NUMEIG,NUP,NZ
C     ..
C     .. Local Arrays ..
      DOUBLE PRECISION A(NZMAX),EI(NMAX),EPS(2),ER(NMAX),RES(NMAX),
     +                 TW(LTWMAX),U(NMAX,NMAX),W(NMAX,NMAX),
     +                 X(NMAX,NMAX),Y(NMAX,NMAX)
```

```
      INTEGER IRN(NZMAX),IW(NMAX),JCN(NZMAX)
      INTEGER KEEP(25),INFO(20),ICNTL(10)
      DOUBLE PRECISION RKEEP(20),RINFO(10)
C     ..
C     .. External Subroutines ..
      EXTERNAL EB22AD,EB22BD,MXMUL
C     ..
C     .. Data statements ..
      DATA N/5/,NUMEIG/1/,NUMCOL/4/,NZ/9/,EPS(1)/1.0D-05/
      DATA IRN/1,2,2,4,3,4,1,3,5/
      DATA JCN/1,1,2,3,4,4,5,5,5/
      DATA A/1.0D0,3.0D0,2.0D0,3.0D0,1.0D0,1.0D0,2.0D0,4.0D0,5.0D0/
C     ..
C     Prepare to call EB22A/AD
C
      CALL EB22ID(ICNTL,KEEP,RKEEP)
      IND = 1
      LN = N
      LTW = NUMCOL* (10+2*NUMCOL) + 4
      IPOS = 0
C
   20 CALL EB22AD(IND,N,NUMEIG,NUMCOL,NLOW,NUP,EPS,X,U,W,LN,TW,LTW,
     +            IW,ER,EI,IPOS,IFLAG,ICNTL,INFO,RINFO,KEEP,RKEEP)
      IF (IFLAG.LT.0) THEN
         WRITE (*,*) ' EB22A/AD failed'
         GO TO 40
      ELSE IF (IPOS.EQ.0) THEN
         WRITE (*,*) '  Algorithm terminated successfully.'
         WRITE (*,*) '  Computed eigenvalue with largest real part:'
         WRITE (*,FMT=9000) ER(1),EI(1)
         GO TO 30
      ELSE
         CALL MXMUL(N,NUMCOL,NLOW,NUP,NZ,IRN,JCN,A,U,W,LN)
         GO TO 20
      END IF
C
C     Compute corresponding eigenvector and residual
C
   30 CALL EB22BD(N,NUMCOL,ER,EI,Y,X,U,W,LN,TW,RES,IPOS,INFO,
     +            KEEP,RKEEP)
      CALL MXMUL(N,NUMCOL,1,NUMCOL,NZ,IRN,JCN,A,U,W,LN)
      CALL EB22BD(N,NUMCOL,ER,EI,Y,X,U,W,LN,TW,RES,IPOS,INFO,
     +            KEEP,RKEEP)
      IF (IPOS.EQ.0) THEN
         WRITE (*,FMT=9020) (Y(I,1),I=1,N)
      END IF
   40 STOP
 9000 FORMAT (/,'  Real part ',G11.3,'  Imaginary part ',G11.3)
 9020 FORMAT (/,'  Computed eigenvector is:',/,5G11.3)
      END
C
      SUBROUTINE MXMUL(N,NUMCOL,NLOW,NUP,NZ,IRN,JCN,A,U,W,LN)
C
C     .. Scalar Arguments ..
      INTEGER LN,N,NLOW,NUMCOL,NUP,NZ
C     ..
C     .. Array Arguments ..
      DOUBLE PRECISION A(NZ),U(LN,NUMCOL),W(LN,NUMCOL)
      INTEGER IRN(NZ),JCN(NZ)
C     ..
C     .. Local Scalars ..
```

```
      INTEGER I,ICOL,J,L
C     ..
      DO 30 ICOL = NLOW,NUP
         DO 10 I = 1,N
            U(I,ICOL) = 0.0D0
   10    CONTINUE
         DO 20 L = 1,NZ
            I = IRN(L)
            J = JCN(L)
            U(I,ICOL) = U(I,ICOL) + A(L)*W(J,ICOL)
   20    CONTINUE
   30 CONTINUE
      RETURN
      END
```

This gives the following output:

```
 Algorithm terminated successfully.
 Computed eigenvalue with largest real part:

Real part    5.00      Imaginary part    0.00

Computed eigenvector is:
0.294      0.294      0.554      0.416      0.589
```