

NAG Library Routine Document

F08SPF (ZHEGVX)

Note: before using this routine, please read the Users' Note for your implementation to check the interpretation of *bold italicised* terms and other implementation-dependent details.

1 Purpose

F08SPF (ZHEGVX) computes selected eigenvalues and, optionally, eigenvectors of a complex generalized Hermitian-definite eigenproblem, of the form

$$Az = \lambda Bz, \quad ABz = \lambda z \quad \text{or} \quad BAz = \lambda z,$$

where A and B are Hermitian and B is also positive definite. Eigenvalues and eigenvectors can be selected by specifying either a range of values or a range of indices for the desired eigenvalues.

2 Specification

```

SUBROUTINE F08SPF (ITYPE, JOBZ, RANGE, UPLO, N, A, LDA, B, LDB, VL, VU, IL,      &
                  IU, ABSTOL, M, W, Z, LDZ, WORK, LWORK, RWORK, IWORK,      &
                  JFAIL, INFO)
INTEGER           ITYPE, N, LDA, LDB, IL, IU, M, LDZ, LWORK,              &
                  IWORK(5*N), JFAIL(*), INFO
REAL (KIND=nag_wp) VL, VU, ABSTOL, W(N), RWORK(7*N)
COMPLEX (KIND=nag_wp) A(LDA,*), B(LDB,*), Z(LDZ,*), WORK(max(1,LWORK))
CHARACTER(1)     JOBZ, RANGE, UPLO

```

The routine may be called by its LAPACK name *zhegvx*.

3 Description

F08SPF (ZHEGVX) first performs a Cholesky factorization of the matrix B as $B = U^H U$, when $UPLO = 'U'$ or $B = LL^H$, when $UPLO = 'L'$. The generalized problem is then reduced to a standard symmetric eigenvalue problem

$$Cx = \lambda x,$$

which is solved for the desired eigenvalues and eigenvectors; the eigenvectors are then backtransformed to give the eigenvectors of the original problem.

For the problem $Az = \lambda Bz$, the eigenvectors are normalized so that the matrix of eigenvectors, Z , satisfies

$$Z^H A Z = \Lambda \quad \text{and} \quad Z^H B Z = I,$$

where Λ is the diagonal matrix whose diagonal elements are the eigenvalues. For the problem $ABz = \lambda z$ we correspondingly have

$$Z^{-1} A Z^{-H} = \Lambda \quad \text{and} \quad Z^H B Z = I,$$

and for $BAz = \lambda z$ we have

$$Z^H A Z = \Lambda \quad \text{and} \quad Z^H B^{-1} Z = I.$$

4 References

Anderson E, Bai Z, Bischof C, Blackford S, Demmel J, Dongarra J J, Du Croz J J, Greenbaum A, Hammarling S, McKenney A and Sorensen D (1999) *LAPACK Users' Guide* (3rd Edition) SIAM, Philadelphia <http://www.netlib.org/lapack/lug>

Demmel J W and Kahan W (1990) Accurate singular values of bidiagonal matrices *SIAM J. Sci. Statist. Comput.* **11** 873–912

Golub G H and Van Loan C F (1996) *Matrix Computations* (3rd Edition) Johns Hopkins University Press, Baltimore

5 Parameters

- 1: ITYPE – INTEGER *Input*
On entry: specifies the problem type to be solved.
 ITYPE = 1
 $Az = \lambda Bz.$
 ITYPE = 2
 $ABz = \lambda z.$
 ITYPE = 3
 $BAz = \lambda z.$
Constraint: ITYPE = 1, 2 or 3.
- 2: JOBZ – CHARACTER(1) *Input*
On entry: indicates whether eigenvectors are computed.
 JOBZ = 'N'
 Only eigenvalues are computed.
 JOBZ = 'V'
 Eigenvalues and eigenvectors are computed.
Constraint: JOBZ = 'N' or 'V'.
- 3: RANGE – CHARACTER(1) *Input*
On entry: if RANGE = 'A', all eigenvalues will be found.
 If RANGE = 'V', all eigenvalues in the half-open interval (VL, VU] will be found.
 If RANGE = 'I', the ILth to IUth eigenvalues will be found.
Constraint: RANGE = 'A', 'V' or 'I'.
- 4: UPLO – CHARACTER(1) *Input*
On entry: if UPLO = 'U', the upper triangles of A and B are stored.
 If UPLO = 'L', the lower triangles of A and B are stored.
Constraint: UPLO = 'U' or 'L'.
- 5: N – INTEGER *Input*
On entry: n , the order of the matrices A and B .
Constraint: $N \geq 0$.
- 6: A(LDA,*) – COMPLEX (KIND=nag_wp) array *Input/Output*
Note: the second dimension of the array A must be at least $\max(1, N)$.
On entry: the n by n Hermitian matrix A .

If UPLO = 'U', the upper triangular part of A must be stored and the elements of the array below the diagonal are not referenced.

If UPLO = 'L', the lower triangular part of A must be stored and the elements of the array above the diagonal are not referenced.

On exit: the lower triangle (if UPLO = 'L') or the upper triangle (if UPLO = 'U') of A , including the diagonal, is overwritten.

7: LDA – INTEGER *Input*

On entry: the first dimension of the array A as declared in the (sub)program from which F08SPF (ZHEGVX) is called.

Constraint: $LDA \geq \max(1, N)$.

8: B(LDB,*) – COMPLEX (KIND=nag_wp) array *Input/Output*

Note: the second dimension of the array B must be at least $\max(1, N)$.

On entry: the n by n Hermitian matrix B .

If UPLO = 'U', the upper triangular part of B must be stored and the elements of the array below the diagonal are not referenced.

If UPLO = 'L', the lower triangular part of B must be stored and the elements of the array above the diagonal are not referenced.

On exit: the triangular factor U or L from the Cholesky factorization $B = U^H U$ or $B = LL^H$.

9: LDB – INTEGER *Input*

On entry: the first dimension of the array B as declared in the (sub)program from which F08SPF (ZHEGVX) is called.

Constraint: $LDB \geq \max(1, N)$.

10: VL – REAL (KIND=nag_wp) *Input*

11: VU – REAL (KIND=nag_wp) *Input*

On entry: if RANGE = 'V', the lower and upper bounds of the interval to be searched for eigenvalues.

If RANGE = 'A' or 'T', VL and VU are not referenced.

Constraint: if RANGE = 'V', $VL < VU$.

12: IL – INTEGER *Input*

13: IU – INTEGER *Input*

On entry: if RANGE = 'T', the indices (in ascending order) of the smallest and largest eigenvalues to be returned.

If RANGE = 'A' or 'V', IL and IU are not referenced.

Constraints:

if RANGE = 'T' and $N = 0$, $IL = 1$ and $IU = 0$;

if RANGE = 'T' and $N > 0$, $1 \leq IL \leq IU \leq N$.

14: ABSTOL – REAL (KIND=nag_wp) *Input*

On entry: the absolute error tolerance for the eigenvalues. An approximate eigenvalue is accepted as converged when it is determined to lie in an interval $[a, b]$ of width less than or equal to

$$ABSTOL + \epsilon \max(|a|, |b|),$$

where ϵ is the *machine precision*. If ABSTOL is less than or equal to zero, then $\epsilon \|T\|_1$ will be used in its place, where T is the tridiagonal matrix obtained by reducing C to tridiagonal form.

Eigenvalues will be computed most accurately when `ABSTOL` is set to twice the underflow threshold $2 \times X02AMF()$, not zero. If this routine returns with `INFO = 1` to `N`, indicating that some eigenvectors did not converge, try setting `ABSTOL` to $2 \times X02AMF()$. See Demmel and Kahan (1990).

15: `M` – INTEGER *Output*

On exit: the total number of eigenvalues found. $0 \leq M \leq N$.

If `RANGE = 'A'`, $M = N$.

If `RANGE = 'T'`, $M = IU - IL + 1$.

16: `W(N)` – REAL (`KIND=nag_wp`) array *Output*

On exit: the first M elements contain the selected eigenvalues in ascending order.

17: `Z(LDZ,*)` – COMPLEX (`KIND=nag_wp`) array *Output*

Note: the second dimension of the array `Z` must be at least $\max(1, M)$ if `JOBZ = 'V'`, and at least 1 otherwise.

On exit: if `JOBZ = 'V'`, then

if `INFO = 0`, the first M columns of `Z` contain the orthonormal eigenvectors of the matrix `A` corresponding to the selected eigenvalues, with the i th column of `Z` holding the eigenvector associated with `W(i)`. The eigenvectors are normalized as follows:

if `ITYPE = 1` or `2`, $Z^H B Z = I$;

if `ITYPE = 3`, $Z^H B^{-1} Z = I$;

if an eigenvector fails to converge (`INFO = 1` to `N`), then that column of `Z` contains the latest approximation to the eigenvector, and the index of the eigenvector is returned in `JFAIL`.

If `JOBZ = 'N'`, `Z` is not referenced.

Note: you must ensure that at least $\max(1, M)$ columns are supplied in the array `Z`; if `RANGE = 'V'`, the exact value of M is not known in advance and an upper bound of at least N must be used.

18: `LDZ` – INTEGER *Input*

On entry: the first dimension of the array `Z` as declared in the (sub)program from which `F08SPF` (`ZHEGVX`) is called.

Constraints:

if `JOBZ = 'V'`, $LDZ \geq \max(1, N)$;

otherwise $LDZ \geq 1$.

19: `WORK(max(1, LWORK))` – COMPLEX (`KIND=nag_wp`) array *Workspace*

On exit: if `INFO = 0`, the real part of `WORK(1)` contains the minimum value of `LWORK` required for optimal performance.

20: `LWORK` – INTEGER *Input*

On entry: the dimension of the array `WORK` as declared in the (sub)program from which `F08SPF` (`ZHEGVX`) is called.

If `LWORK = -1`, a workspace query is assumed; the routine only calculates the optimal size of the `WORK` array, returns this value as the first entry of the `WORK` array, and no error message related to `LWORK` is issued.

Suggested value: for optimal performance, $LWORK \geq (nb + 1) \times N$, where nb is the optimal **block size** for F08FSF (ZHETRD).

Constraint: $LWORK \geq \max(1, 2 \times N)$.

21: RWORK($7 \times N$) – REAL (KIND=nag_wp) array Workspace

22: IWORK($5 \times N$) – INTEGER array Workspace

23: JFAIL(*) – INTEGER array Output

Note: the dimension of the array JFAIL must be at least $\max(1, N)$.

On exit: if JOBZ = 'V', then

if INFO = 0, the first M elements of JFAIL are zero;

if INFO = 1 to N, JFAIL contains the indices of the eigenvectors that failed to converge.

If JOBZ = 'N', JFAIL is not referenced.

24: INFO – INTEGER Output

On exit: INFO = 0 unless the routine detects an error (see Section 6).

6 Error Indicators and Warnings

Errors or warnings detected by the routine:

INFO < 0

If INFO = $-i$, argument i had an illegal value. An explanatory message is output, and execution of the program is terminated.

INFO = 1 to N

If INFO = i , F08FPF (ZHEEVX) failed to converge; i eigenvectors failed to converge. Their indices are stored in array JFAIL.

INFO > N

F07FRF (ZPOTRF) returned an error code; i.e., if INFO = $N + i$, for $1 \leq i \leq N$, then the leading minor of order i of B is not positive definite. The factorization of B could not be completed and no eigenvalues or eigenvectors were computed.

7 Accuracy

If B is ill-conditioned with respect to inversion, then the error bounds for the computed eigenvalues and vectors may be large, although when the diagonal elements of B differ widely in magnitude the eigenvalues and eigenvectors may be less sensitive than the condition of B would suggest. See Section 4.10 of Anderson *et al.* (1999) for details of the error bounds.

8 Further Comments

The total number of floating point operations is proportional to n^3 .

The real analogue of this routine is F08SBF (DSYGVX).

9 Example

This example finds the eigenvalues in the half-open interval $(-3, 3]$, and corresponding eigenvectors, of the generalized Hermitian eigenproblem $Az = \lambda Bz$, where

$$A = \begin{pmatrix} -7.36 & 0.77 - 0.43i & -0.64 - 0.92i & 3.01 - 6.97i \\ 0.77 + 0.43i & 3.49 & 2.19 + 4.45i & 1.90 + 3.73i \\ -0.64 + 0.92i & 2.19 - 4.45i & 0.12 & 2.88 - 3.17i \\ 3.01 + 6.97i & 1.90 - 3.73i & 2.88 + 3.17i & -2.54 \end{pmatrix}$$

and

$$B = \begin{pmatrix} 3.23 & 1.51 - 1.92i & 1.90 + 0.84i & 0.42 + 2.50i \\ 1.51 + 1.92i & 3.58 & -0.23 + 1.11i & -1.18 + 1.37i \\ 1.90 - 0.84i & -0.23 - 1.11i & 4.09 & 2.33 - 0.14i \\ 0.42 - 2.50i & -1.18 - 1.37i & 2.33 + 0.14i & 4.29 \end{pmatrix}.$$

The example program for F08SQF (ZHEGVD) illustrates solving a generalized Hermitian eigenproblem of the form $ABz = \lambda z$.

9.1 Program Text

Program f08spfe

```
!      F08SPF Example Program Text
!
!      Mark 24 Release. NAG Copyright 2012.
!
!      .. Use Statements ..
!      Use nag_library, Only: nag_wp, x04daf, zhegvx
!      .. Implicit None Statement ..
!      Implicit None
!      .. Parameters ..
!      Real (Kind=nag_wp), Parameter      :: zero = 0.0E+0_nag_wp
!      Integer, Parameter                 :: nb = 64, nin = 5, nout = 6
!      .. Local Scalars ..
!      Real (Kind=nag_wp)                 :: abstol, vl, vu
!      Integer                             :: i, ifail, il, info, iu, lda, ldb,      &
!                                          ldz, lwork, m, n
!
!      .. Local Arrays ..
!      Complex (Kind=nag_wp), Allocatable :: a(:,,:), b(:,,:), work(:), z(:,,:)
!      Complex (Kind=nag_wp)              :: dummy(1)
!      Real (Kind=nag_wp), Allocatable    :: rwork(:), w(:)
!      Integer, Allocatable                :: iwork(:), jfail(:)
!
!      .. Intrinsic Procedures ..
!      Intrinsic                           :: max, nint, real
!
!      .. Executable Statements ..
!      Write (nout,*) 'F08SPF Example Program Results'
!      Write (nout,*)
!      Skip heading in data file
!      Read (nin,*)
!      Read (nin,*) n
!      lda = n
!      ldb = n
!      ldz = n
!      m = n
!      Allocate (a(lda,n),b(ldb,n),z(ldz,m),rwork(7*n),w(n),iwork(5*n), &
!               jfail(n))
!
!      Read the lower and upper bounds of the interval to be searched.
!      Read (nin,*) vl, vu
!
!      Use routine workspace query to get optimal workspace.
!      lwork = -1
!      The NAG name equivalent of zhegvx is f08spf
!      Call zhegvx(1,'Vectors','Values in range','Upper',n,a,lda,b,ldb,vl,vu, &
!               il,iu,abstol,m,w,z,ldz,dummy,lwork,rwork,iwork,jfail,info)
!
!      Make sure that there is enough workspace for blocksize nb.
!      lwork = max((nb+1)*n,nint(real(dummy(1))))
!      Allocate (work(lwork))
```

```

!      Read the upper triangular parts of the matrices A and B

      Read (nin,*)(a(i,i:n),i=1,n)
      Read (nin,*)(b(i,i:n),i=1,n)

!      Set the absolute error tolerance for eigenvalues.  With abstol
!      set to zero, the default value is used instead

      abstol = zero

!      Solve the generalized Hermitian eigenvalue problem
!      A*x = lambda*B*x (itype = 1)

!      The NAG name equivalent of zhegvx is f08spf
      Call zhegvx(1,'Vectors','Values in range','Upper',n,a,lda,b,ldb,vl,vu, &
        il,iu,abstol,m,w,z,ldz,work,lwork,rwork,iwork,jfail,info)

      If (info>=0 .And. info<=n) Then

!      Print solution

        Write (nout,99999) 'Number of eigenvalues found =', m
        Write (nout,*)
        Write (nout,*) 'Eigenvalues'
        Write (nout,99998) w(1:m)
        Flush (nout)

!      Normalize the eigenvectors
      Do i = 1, m
        z(1:n,i) = z(1:n,i)/z(1,i)
      End Do

!      ifail: behaviour on error exit
!      =0 for hard exit, =1 for quiet-soft, =-1 for noisy-soft
      ifail = 0
      Call x04daf('General',' ',n,m,z,ldz,'Selected eigenvectors',ifail)

      If (info>0) Then
        Write (nout,99999) 'INFO eigenvectors failed to converge, INFO =', &
          info
        Write (nout,*) 'Indices of eigenvectors that did not converge'
        Write (nout,99997) jfail(1:m)
      End If
      Else If (info>n .And. info<=2*n) Then
        i = info - n
        Write (nout,99996) 'The leading minor of order ', i, &
          ' of B is not positive definite'
      Else
        Write (nout,99999) 'Failure in ZHEGVX. INFO =', info
      End If

99999 Format (1X,A,I5)
99998 Format (3X,(8F8.4))
99997 Format (3X,(8I8))
99996 Format (1X,A,I4,A)
      End Program f08spfe

```

9.2 Program Data

F08SPF Example Program Data

```

4                                     :Value of N
-3.0          3.0                     :Values of VL and VU
(-7.36, 0.00) ( 0.77, -0.43) (-0.64, -0.92) ( 3.01, -6.97)
              ( 3.49, 0.00) ( 2.19, 4.45) ( 1.90, 3.73)
              ( 0.12, 0.00) ( 2.88, -3.17)
              (-2.54, 0.00) :End of matrix A

```

```
( 3.23, 0.00) ( 1.51, -1.92) ( 1.90, 0.84) ( 0.42, 2.50)
                ( 3.58, 0.00) (-0.23, 1.11) (-1.18, 1.37)
                                ( 4.09, 0.00) ( 2.33, -0.14)
                                                ( 4.29, 0.00) :End of matrix B
```

9.3 Program Results

F08SPF Example Program Results

Number of eigenvalues found = 2

Eigenvalues

-2.9936 0.5047

Selected eigenvectors

	1	2
1	1.0000	1.0000
	0.0000	-0.0000
2	0.1491	0.1882
	0.0777	-0.7410
3	-1.2303	-0.2080
	-0.4192	-0.4733
4	0.5811	0.4524
	1.0051	0.9265
