Technische Universität Berlin

Institut für Mathematik

STCSSP: A FORTRAN 77 routine to compute a structured staircase form for a (skew-)symmetric/(skew-)symmetric matrix pencil.

T. Brüll and V. Mehrmann

Preprint 31-2007

Preprint-Reihe des Instituts für Mathematik
Technische Universität Berlin

Report 31-2007 **March 2008**

STCSSP: A FORTRAN 77 routine to compute a structured **st**air**c**ase form for a (skew-) **s**ymmetric / (skew-) **s**ymmetric matrix **p**encil. *

Tobias Brüll † Volker Mehrmann †

March 19, 2008

1 Purpose

Let $(N, H) \in \mathbb{R}^{n,n} \times \mathbb{R}^{n,n}$ be a matrix pencil that satisfies one of the conditions

$$N = N^T \text{ and } H = H^T, \tag{1a}$$

$$N = N^T \text{ and } H = -H^T, \tag{1b}$$

$$N = -N^T \text{ and } H = H^T, \tag{2a}$$

$$N = -N^T \text{ and } H = -H^T. \tag{2b}$$

Then we call (N, H) *(skew-) symmetric / (skew-) symmetric*. Likewise, one could say, that a pencil (N, H) is (skew-) symmetric / (skew-) symmetric if and only if

$$N = op_N(N) \text{ and } H = op_H(H), \tag{3}$$

where $op_N(N) = N^T$ or $op_N(N) = -N^T$ and $op_H(H) = H^T$ or $op_H(H) = -H^T$. Pencils that satisfy (2a) are called *even*, see [1]. STCSSP computes a structured staircase form for a real (skew-) symmetric / (skew-) symmetric matrix pencil. The staircase form is achieved by an orthogonal transformation of the form

$$(U^T NU, U^T HU) = (N_{new}, H_{new}), \tag{4}$$

with $U \in \mathbb{R}^{n,n}$ orthogonal. Obviously, we have $(N_{new}, H_{new}) = (op_N(N_{new}), op_H(H_{new}))$ and thus the pencil in staircase form (N_{new}, H_{new}) is (skew-) symmetric / (skew-) symmetric again.

For a real symmetric matrix A we know that all eigenvalues are real. Thus, one can count the positive, negative, and zero eigenvalues. We call the triple (π, ν, ξ) the *inertia index* of A, if A has π positive eigenvalues, ν negative eigenvalues, and ξ zero eigenvalues.

*The work has been supported by DFG research grant no. Me790/16-1.

†Institut für Mathematik, MA 4-5, Technische Universität Berlin, Straße des 17. Juni 136, D-10623 Berlin, Fed. Rep. Germany. E-mail: {bruell,mehrmann}@math.tu-berlin.de.

2 The theory

The algorithm implemented is based on the following theorem, which is a slight generalization of [1, Theorem 3.1] from even matrix pencils to (skew-) symmetric / (skew-) symmetric matrix pencils.

Theorem 2.1. ***(Skew-) symmetric / (skew-) symmetric staircase form.*** *With the operators defined in (3) consider the (skew-) symmetric / (skew-) symmetric matrix pencil $(N, H) = (op_N(N), op_H(H))$, where $N, H \in \mathbb{R}^{n,n}$. There exists a real orthogonal matrix $U \in \mathbb{R}^{n,n}$, such that*

$$U^T N U =$$

$$\begin{array}{c} n_1 \\ \vdots \\ \vdots \\ n_m \\ \hline l \\ \hline q_m \\ \vdots \\ q_2 \\ q_1 \end{array}
\left[\begin{array}{cccc|c|cccc}
N_{11} & \cdots & \cdots & N_{1,m} & N_{1,m+1} & N_{1,m+2} & \cdots & N_{1,2m} & 0 \\
\vdots & \ddots & & \vdots & \vdots & \vdots & \iddots & \iddots & \\
\vdots & & \ddots & \vdots & \vdots & N_{m-1,m+2} & \iddots & & \\
op_N(N_{1,m}) & \cdots & \cdots & N_{m,m} & N_{m,m+1} & 0 & & & \\
\hline
op_N(N_{1,m+1}) & \cdots & \cdots & op_N(N_{m,m+1}) & N_{m+1,m+1} & & & & \\
\hline
op_N(N_{1,m+2}) & \cdots & op_N(N_{m-1,m+2}) & 0 & & & & & \\
\vdots & \iddots & \iddots & & & & & & \\
op_N(N_{1,2m}) & \iddots & & & & & & & \\
0 & & & & & & & &
\end{array}\right] \tag{5}$$

$$U^T H U =$$

$$\begin{array}{c} n_1 \\ \vdots \\ \vdots \\ n_m \\ \hline l \\ \hline q_m \\ \vdots \\ \vdots \\ q_1 \end{array}
\left[\begin{array}{cccc|c|cccc}
H_{11} & \cdots & \cdots & H_{1,m} & H_{1,m+1} & H_{1,m+2} & \cdots & \cdots & H_{1,2m+1} \\
\vdots & \ddots & & \vdots & \vdots & \vdots & & \iddots & \\
\vdots & & \ddots & \vdots & \vdots & \vdots & \iddots & & \\
op_H(H_{1,m}) & \cdots & \cdots & H_{m,m} & H_{m,m+1} & H_{m,m+2} & & & \\
\hline
op_H(H_{1,m+1}) & \cdots & \cdots & op_H(H_{m,m+1}) & H_{m+1,m+1} & & & & \\
\hline
op_H(H_{1,m+2}) & \cdots & \cdots & op_H(H_{m,m+2}) & & & & & \\
\vdots & & \iddots & & & & & & \\
\vdots & \iddots & & & & & & & \\
op_H(H_{1,2m+1}) & & & & & & & &
\end{array}\right],$$

where for $i = 1, \ldots, m$ we have $N_{ii} = op_N(N_{ii})$, $H_{ii} = op_H(H_{ii})$. Further, we know that $q_1 \geq n_1 \geq q_2 \geq n_2 \geq \ldots \geq q_m \geq n_m$,

$$\begin{aligned}
N_{j,2m+1-j} &\in \mathbb{R}^{n_j,q_{j+1}}, \qquad 1 \leq j \leq m-1, \\
N_{m+1,m+1} &= \begin{bmatrix} \Delta & 0 \\ 0 & 0 \end{bmatrix}, \quad \Delta = op_N(\Delta) \in \mathbb{R}^{p,p}, \\
H_{j,2m+2-j} &= \begin{bmatrix} \Gamma_j & 0 \end{bmatrix} \in \mathbb{R}^{n_j,q_j}, \quad \Gamma_j \in \mathbb{R}^{n_j,n_j}, \qquad 1 \leq j \leq m, \\
H_{m+1,m+1} &= \begin{bmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{bmatrix}, \quad \Sigma_{11} \in \mathbb{R}^{p,p}, \quad \Sigma_{22} \in \mathbb{R}^{l-p,l-p}, \\
H_{m+1,m+1} &= op_H(H_{m+1,m+1}),
\end{aligned}$$

and the blocks Σ_{22} and Δ and Γ_j, $j = 1, \ldots, m$ are nonsingular.

Note, that what is called p in Theorem 2.1 is corresponding to $2p$ in [1, Theorem 3.1]. The form (5) is by far not unique. Even the quantities $q_1, n_1, q_2, n_2, \ldots$ are not unique, but they become unique when using the following algorithm to compute the form (5) which is a slight generalization of [1, Algorithm 1] from even matrix pencils to (skew-) symmetric / (skew-) symmetric matrix pencils and represents a constructive proof of Theorem 2.1.

Algorithm 2.2. *Staircase algorithm for (skew-) symmetric / (skew-) symmetric matrix pencils.*
With the operators defined in (3) consider the (skew-) symmetric / (skew-) symmetric matrix pencil $(N, H) = (op_N(N), op_H(H))$, where $N, H \in \mathbb{R}^{n,n}$. Then this algorithm computes an orthogonal matrix $U \in \mathbb{R}^{n,n}$ such that $U^T NU$, $U^T HU$ are in the form (5). In addition, for each of the matrices N and H, which is real symmetric, the algorithm produces a **unique** sequence of inertia indices, i.e., if only N or H is real symmetric one sequence of inertia indices is generated and if both N and H are real symmetric two sequences of inertia indices are generated.
`Set flag` $= 0$, $m = n_0 = q_0 = r_0 = 0$, $l = n$,

$$\mathcal{N} = \mathcal{N}_{22} = N, \quad \mathcal{H} = H, \quad U = I.$$

`DO WHILE flag` $= 0$

`Perform a rank revealing factorization of` $\mathcal{N}_{22} \in \mathbb{R}^{l-r_m,l-r_m}$,

$$\mathcal{N}_{22} = U_1 \begin{bmatrix} \Delta & 0 \\ 0 & 0 \end{bmatrix} U_1^T,$$

`with` $\Delta = op_N(\Delta) \in \mathbb{R}^{p,p}$ `nonsingular. If the matrix` N `is real symmetric, also store the inertia indices of` Δ `as` $(\pi^N_{m+1}, \nu^N_{m+1}, 0)$. `Set`

$$\begin{aligned} \mathcal{N}_1 &= \begin{bmatrix} U_1 & 0 \\ 0 & I_{r_m} \end{bmatrix}^T \mathcal{N} \begin{bmatrix} U_1 & 0 \\ 0 & I_{r_m} \end{bmatrix} = \begin{bmatrix} \Delta & 0 \\ 0 & 0 \end{bmatrix}, \\ \mathcal{H}_1 &= \begin{bmatrix} U_1 & 0 \\ 0 & I_{r_m} \end{bmatrix}^T \mathcal{H} \begin{bmatrix} U_1 & 0 \\ 0 & I_{r_m} \end{bmatrix} = \begin{bmatrix} \hat{\mathcal{H}}_{11} & \hat{\mathcal{H}}_{12} \\ op_H(\hat{\mathcal{H}}_{12}) & \hat{\mathcal{H}}_{22} \end{bmatrix}, \end{aligned}$$

`partitioned analogously, with` $\hat{\mathcal{H}}_{11} = op_H(\hat{\mathcal{H}}_{11})$, `and` $\hat{\mathcal{H}}_{22} = op_H(\hat{\mathcal{H}}_{22})$.
`(Here` $\hat{\mathcal{H}}_{22} \in \mathbb{R}^{l-p,l-p}$`).`

`IF` $p = l$ `THEN`

`Set flag` $= 1$ `and`

$$\mathcal{U} = \begin{bmatrix} I_{n_1+\ldots+n_m} & 0 & 0 \\ 0 & U_1 & 0 \\ 0 & 0 & I_{q_1+\ldots+q_m} \end{bmatrix}. \tag{6}$$

`ELSE`

`Set` $m = m + 1$.

`Perform a rank revealing decomposition of` $\hat{\mathcal{H}}_{22}$,

$$\hat{\mathcal{H}}_{22} = U_2 \begin{bmatrix} \Sigma & 0 \\ 0 & 0 \end{bmatrix} U_2^T,$$

`where` $\Sigma = op_H(\Sigma) \in \mathbb{R}^{\mu,\mu}$ `is nonsingular. If the matrix` H `is real symmetric, also store the inertia indices of` Σ `as` $(\pi_m, \nu_m, 0)$. `Set` $r_m = \mu = \pi_m + \nu_m$.

Set

$$\mathcal{N}_2 = \begin{bmatrix} I_p & 0 \\ 0 & U_2 \end{bmatrix}^T \mathcal{N}_1 \begin{bmatrix} I_p & 0 \\ 0 & U_2 \end{bmatrix} = \begin{bmatrix} \Delta & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix},$$

$$\mathcal{H}_2 = \begin{bmatrix} I_p & 0 \\ 0 & U_2 \end{bmatrix}^T \mathcal{H}_1 \begin{bmatrix} I_p & 0 \\ 0 & U_2 \end{bmatrix} = \begin{bmatrix} \tilde{\mathcal{H}}_{11} & \tilde{\mathcal{H}}_{12} & \tilde{\mathcal{H}}_{13} \\ op_H(\tilde{\mathcal{H}}_{12}) & \Sigma & 0 \\ op_H(\tilde{\mathcal{H}}_{13}) & 0 & 0 \end{bmatrix},$$

partitioned analogously.

IF $\mu = l - p$ THEN

Set flag $= 1$ and

$$\hat{\mathcal{U}} = \begin{bmatrix} U_1 & 0 \\ 0 & I_{r_{m-1}} \end{bmatrix} \begin{bmatrix} I_p & 0 \\ 0 & U_2 \end{bmatrix},$$

$$\mathcal{U} = \begin{bmatrix} I_{n_1+\ldots+n_{m-1}} & 0 & 0 \\ 0 & \hat{\mathcal{U}} & 0 \\ 0 & 0 & I_{q_1+\ldots+q_{m-1}} \end{bmatrix}$$

ELSE

Perform a rank revealing factorization or SVD

$$\tilde{\mathcal{H}}_{13} = U_3 \begin{bmatrix} \Gamma_m & 0 \\ 0 & 0 \end{bmatrix} V_3^T,$$

where $\Gamma_m \in \mathbb{R}^{\tau,\tau}$ is nonsingular.

Set $n_m = \tau$, $q_m = l - p - \mu$ and

$$\mathcal{N}_3 = \begin{bmatrix} U_3 & 0 & 0 \\ 0 & I_\mu & 0 \\ 0 & 0 & V_3 \end{bmatrix}^T \mathcal{N}_2 \begin{bmatrix} U_3 & 0 & 0 \\ 0 & I_\mu & 0 \\ 0 & 0 & V_3 \end{bmatrix}$$

$$= \left[\begin{array}{c|cc|cc} \mathcal{N}_{11} & \mathcal{N}_{12} & 0 & 0 & 0 \\ \hline op_N(\mathcal{N}_{12}) & \mathcal{N}_{22} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right],$$

$$\mathcal{H}_3 = \begin{bmatrix} U_3 & 0 & 0 \\ 0 & I_\mu & 0 \\ 0 & 0 & V_3 \end{bmatrix}^T \mathcal{H}_2 \begin{bmatrix} U_3 & 0 & 0 \\ 0 & I_\mu & 0 \\ 0 & 0 & V_3 \end{bmatrix}$$

$$= \left[\begin{array}{c|cc|cc} \mathcal{H}_{11} & \mathcal{H}_{12} & \mathcal{H}_{13} & \Gamma_m & 0 \\ \hline op_H(\mathcal{H}_{12}) & \mathcal{H}_{22} & \mathcal{H}_{23} & 0 & 0 \\ op_H(\mathcal{H}_{13}) & op_H(\mathcal{H}_{23}) & \Sigma & 0 & 0 \\ \hline op_H(\Gamma_m) & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{array}\right],$$

$$\hat{\mathcal{U}} = \begin{bmatrix} U_1 & 0 \\ 0 & I_{r_{m-1}} \end{bmatrix} \begin{bmatrix} I_p & 0 \\ 0 & U_2 \end{bmatrix} \begin{bmatrix} U_3 & 0 & 0 \\ 0 & I_\mu & 0 \\ 0 & 0 & V_3 \end{bmatrix},$$

$$\mathcal{U} = \begin{bmatrix} I_{n_1+\ldots+n_{m-1}} & 0 & 0 \\ 0 & \hat{\mathcal{U}} & 0 \\ 0 & 0 & I_{q_1+\ldots+q_{m-1}} \end{bmatrix}.$$

Set

$$\mathcal{N} = \begin{matrix} p-\tau \\ \mu \end{matrix} \overset{\begin{matrix} p-\tau & \mu \end{matrix}}{\begin{bmatrix} \mathcal{N}_{22} & 0 \\ 0 & 0 \end{bmatrix}}, \quad \mathcal{H} = \begin{matrix} p-\tau \\ \mu \end{matrix} \overset{\begin{matrix} p-\tau & \mu \end{matrix}}{\begin{bmatrix} \mathcal{H}_{22} & \mathcal{H}_{23} \\ op_H(\mathcal{H}_{23}) & \Sigma \end{bmatrix}} \in \mathbb{R}^{l,l},$$

and $l = p - \tau + \mu$.

END IF

END IF

Form $H = \mathcal{U}^T H \mathcal{U}$, $N = \mathcal{U}^T N \mathcal{U}$, and $U = U\mathcal{U}$.

END WHILE

Algorithm 2.2 is implemented in the function STCSSP.

Remark 1. *When we compute the form (5) for an even pencil with the help of Algorithm 2.2 we can determine the structured Kronecker canonical form of Thompson [3], for the original even pencil (N, H) with the help of the values computed by the algorithm, see [1, Theorem 3.3].*

Let us end this section with two graphs that give an overview of the subroutines that come with STCSSP.

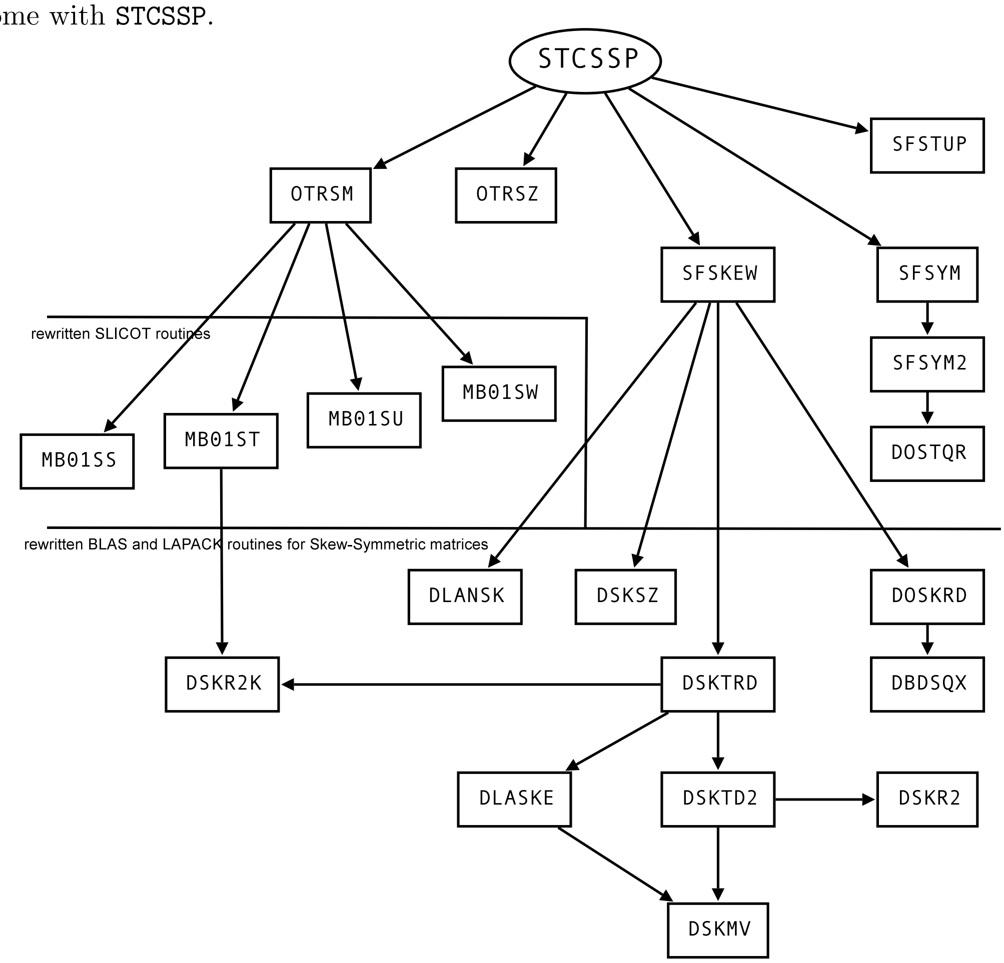

Fig.1: STCSSP and its subroutines. An arrow pointing from A to B means that routine A may call subroutine B at some point.

DBDSQX	slight adaption of the LAPACK routine DBDSQR, which computes a singular value decomposition
DLANSK	skew-symmetric modification of the LAPACK routine DLANSY
DLASKE	skew-symmetric modification of the LAPACK routine DLATRD
DOSKRD	compute the eigenvectors and eigenvalues of a real skew-symmetric tridiagonal matrix with the help of DBDSQX
DOSTQR	slight adaption of the LAPACK routine DSTEQR
DSKMV	skew-symmetric modification of the BLAS routine DSYMV
DSKR2	skew-symmetric modification of the BLAS routine DSYR2
DSKR2K	skew-symmetric modification of the BLAS routine DSYR2K
DSKSZ	split off the odd dimension from a skew-symmetric matrix
DSKTD2	skew-symmetric modification of the LAPACK routine DSYTD2
DSKTRD	skew-symmetric modification of the LAPACK routine DSYTRD
MB01SS	adaption of MB01SW (see below) for skew-symmetric matrices
MB01ST	adaption of MB01SU (see below) for skew-symmetric matrices
MB01SU	a slight adaption of the SLICOT routine MB01RU (uses BLAS3)
MB01SW	a copy of the SLICOT routine MB01RW (uses BLAS2)
OTRSM	apply (a part of) a block-diagonal **O**rthogonal equivalence **TR**ansformation to a (skew-)**S**ymmetric **M**atrix
OTRSZ	apply (a part of) a block-diagonal **O**rthogonal equivalence **TR**ansformation to a (skew-)**S**ymmetric matrix, which contains some **Z**ero blocks
SFSKEW	returns the real **S**chur **F**orm of a real **SKEW**-symmetric matrix
SFSTUP	once **SF**SKEW or **SF**SYM has finished its computations, this routine **S**e**T**s **UP** a matrix therefrom
SFSYM2	computes an ordered real **S**chur **F**orm of a real **SYM**metric matrix
SFSYM	returns an ordered real **S**chur **F**orm of a real **SYM**metric matrix

Tab.1: The functionality implemented in the subroutines.

3 Specification

```
      SUBROUTINE STCSSP( SYMN, SYMH, UPLON, UPLOH, COMPZ, NDIM, N, LDN,
     $                   H, LDH, U, LDU, M, PINVEC, NUNVEC, PIVEC,
     $                   NUVEC, NVEC, QVEC, P, L, TOL, DWORK, LDWORK,
     $                   INFO )
C     .. Scalar Arguments ..
      CHARACTER*1       SYMN, SYMH, UPLON, UPLOH, COMPZ
      INTEGER           NDIM, LDN, LDH, LDU, INFO, LDWORK, M, P, L
      DOUBLE PRECISION  TOL
C     .. Array Arguments ..
      DOUBLE PRECISION  N( LDN, * ), H( LDH, * ), U( LDU, * ),
     $                  DWORK( * )
      INTEGER           PINVEC( * ), NUNVEC( * ), PIVEC( * ),
     $                  NUVEC( * ), NVEC( * ), QVEC( * )
```

4 Argument List

4.1 Mode Parameters

`SYMN, SYMH` - *CHARACTER*1*

These parameters define which of the cases (1a), (1b), (2a), or (2b) is considered. Each of the two parameters may thereby be set to either `'S'`, which means that the corresponding matrix is assumed to be symmetric, or to `'N'`, which means that the corresponding matrix is not assumed to be symmetric but it is assumed to be skew-symmetric. `SYMN` refers to the matrix N and `SYMH` refers to the matrix H, as in Algorithm 2.2. Other values are not allowed and will result in an erroneous termination of the algorithm. For example, to compute the structured staircase form of an even pencil, set `SYMN = 'N'` and `SYMH = 'S'`.
With the operators defined in (3) one could also say that `SYMN = 'N'` (or `SYMH = 'N'`, resp.) means $op_N(N) = -N^T$ (or $op_H(H) = -H^T$, resp.) and `SYMN = 'S'` (or `SYMH = 'S'`, resp.) means $op_N(N) = N^T$ (or $op_H(H) = H^T$, resp.).
Since the diagonal of a real skew-symmetric matrices is always zero, it does not have to be stored. This is why the diagonal elements in the array `N` (or `H`, resp.) are not considered in the computation, once `SYMN = 'N'` (or `SYMH = 'N'`, resp.).

`UPLON, UPLOH` - *CHARACTER*1*

These parameters tell the algorithm how the matrices N and H (as in Algorithm 2.2) are given on input and also how the staircase form is to be stored, once the algorithm has finished successfully. Each of the two parameters may thereby be set to either `'U'`, which means that the upper triangular part of the array `N` (or `H`, resp.) is assumed to hold the upper triangular part of the matrix N (or H, resp.), or to `'L'`, which means that the lower triangular part of the array `N` (or `H`, resp.) is assumed to hold the lower triangular part of the matrix N (or H, resp.). `UPLON` refers to the matrix N and `UPLOH` refers to the matrix H, as in Algorithm 2.2. Other values are not allowed and will result in an erroneous termination of the algorithm. The unused part of the arrays may be used as workspace, but the values in this unused part are not considered. Note, that if one of the parameters `SYMN` or `SYMH` is set to `'N'` then only the strict upper or lower, respectively, part of the corresponding array is used for computation (although the rest may be used as workspace). These parameters also control how the staircase form is returned in the case of an successful exit. The same regions in the arrays `N` and `H` that matter on entry contain the matrices that are transformed to staircase form, on exit.

`COMPZ` - *CHARACTER*1*

This parameter controls, whether the orthogonal transformation U as in Algorithm 2.2 shall also be computed or not. It is faster not to compute the transformation. Setting this parameter to `'N'` does not compute the orthogonal transformation. In this case the array `U` is not referenced, and can be supplied as a dummy array (i.e. set `LDU = 1` and declare this array to be `U(1,1)` in the calling program). Setting this parameter to `'V'` causes the algorithm to compute the orthogonal transformation into the array `U`.

4.2 Input/Output Parameters

`NDIM` - *(input) INTEGER*

The order of the matrices N and H, and thus the dimension of the problem. `NDIM` has to be greater or equal to 0, otherwise the algorithm will return in failure.

`N` - *(input/output) DOUBLE PRECISION, array (LDN, NDIM)*

On entry, the leading `NDIM`-by-`NDIM` part of the array has to contain the matrix N (as in Algorithm 2.2), according to the parameters `SYMN` and `UPLON`.
On successful exit, the leading `NDIM`-by-`NDIM` part of this array contains a part of the staircase form of the matrix N, according to Algorithm 2.2. The parameters `SYMN` and `UPLON` also determine which part of this array contains the actual data. If `UPLON = 'U'`, then only the upper triangular part contains the upper triangular part of the staircase form and if `UPLON = 'L'` only the lower triangular part contains the lower triangular part of the staircase form. Also, if `SYMN = 'N'`, then the diagonal in the array `N` has no meaning, on exit.

`LDN` - *(input) INTEGER*

The leading dimension of the array `N`. The parameter `LDN` has to be greater or equal to `MAX(1,NDIM)`, otherwise the algorithm will return in failure.

`H` - *(input/output) DOUBLE PRECISION, array (LDH, NDIM)*

On entry, the leading `NDIM`-by-`NDIM` part of the array has to contain the matrix H (as in Algorithm 2.2), according to the parameters `SYMH` and `UPLOH`.
On successful exit, the leading `NDIM`-by-`NDIM` part of this array contains a part of the staircase form of the matrix H, according to Algorithm 2.2. The parameters `SYMH` and `UPLOH` also determine which part of this array contains the actual data. If

UPLOH = 'U', then only the upper triangular part contains the upper triangular part of the staircase form and if UPLOH = 'L' only the lower triangular part contains the lower triangular part of the staircase form. Also, if SYMH = 'N', then the diagonal in the array H has no meaning, on exit.

LDH - *(input) INTEGER*

The leading dimension of the array H. The parameter LDH has to be greater or equal to MAX(1,NDIM), otherwise the algorithm will return in failure.

U - *(output) DOUBLE PRECISION, array (LDU, NDIM)*

If COMPZ = 'V' and the algorithm terminated successfully, then the leading NDIM-by-NDIM part of this array contains the orthogonal transformation matrix U as computed by Algorithm 2.2, which has been used to reduce the original matrices N and H to structured staircase form (5). If COMPZ = 'N', then U is not referenced and can be supplied as a dummy array (i.e. set LDU = 1 and declare this array to be U(1,1) in the calling program).

LDU - *(input) INTEGER*

The leading dimension of the array U. LDU has to be greater or equal to 1 in any case and also to be greater or equal to NDIM, if COMPZ = 'V'.

M - *(output) INTEGER*

The number of reduction steps that where necessary to reveal the Kronecker structure. M is always greater or equal to 0 and is corresponds to m in Algorithm 2.2.

PINVEC, NUNVEC - *(output) INTEGER, array (NDIM+1)*

On exit, with INFO = 0 and SYMN = 'S', the first M+1 entries of the arrays contain the inertia indices corresponding to the N matrix, analogously to the π_i^N's and ν_i^N's in Algorithm 2.2. PINVEC(M+1) and NUNVEC(M+1) may not be used. In this case they are both zero. This happens when the algorithm does not exit because a submatrix of N with full rank was discovered (i.e. the algorithm did not exit from (6)). On exit, with INFO = 0 and SYMN = 'N', the first M+1 entries of the array contain only zeros.

PIVEC, NUVEC - *(output) INTEGER, array (NDIM)*

On exit, with INFO = 0 and SYMH = 'S', the first M entries of the arrays contain the inertia indices corresponding to the H matrix, analogously to the π_i's and ν_i's in Algorithm 2.2. On exit, with INFO = 0 and SYMH = 'N', the first M entries of the array contain only zeros.

NVEC, QVEC - *(output) INTEGER, array (NDIM)*

On exit, with INFO = 0, the first M entries of the arrays contain the n_i's and q_i's as in Theorem 2.1 and Algorithm 2.2. Thus, these arrays describe the block structure of the structured staircase form.

P - *(output) INTEGER*

On exit, with `INFO = 0`, this integer contains the number of the finite eigenvalues of the pencil and thus the number of finite eigenvalues of the regular, index 1 part of the pencil. We have $0 \leq P \leq L$, on successful exit. This parameter corresponds to the value p in Algorithm 2.2 and Theorem 2.1.

`L` - *(output) INTEGER*

On exit, with `INFO = 0`, this integer contains the size of the regular, index 1 part of the pencil. We have $P \leq L \leq NDIM$ on successful exit. This parameter corresponds to the value l in Algorithm 2.2 and Theorem 2.1.

4.3 Tolerance

`TOL` - *(input) DOUBLE PRECISION*

Absolute value, below which an eigenvalue or singular value shall be considered zero. If `TOL <= 0.0` is given on entry, the tolerance is automatically chosen to be `NDIM`·*eps*, where *eps* is the machine precision, as returned by the `LAPACK` routine `DLAMCH`.
In a later version of `STCSSP` we will implement the rank decision algorithm described in [2, Section 5] which involves an additional input parameter `GAP`. This rank decision algorithm also uses different tolerances for rank decisions in N and H depending on the Frobenius norms of N and H.

4.4 Workspace

`DWORK` - *DOUBLE PRECISION, array (LDWORK)*

On exit, if `INFO = 0`, `DWORK(1)` returns the optimal value of `LDWORK`.

`LDWORK` - *INTEGER*

The length of the array `DWORK` that may be used by the algorithm as workspace. `LDWORK` has to be greater or equal to `NDIM*NDIM + 3*NDIM + 3`, otherwise the algorithm will issue an error message (i.e., the algorithm will return with `INFO = -22`).

4.5 Error Indicator

`INFO` - *INTEGER*

`INFO = 0`: Successful exit.
`INFO < 0`: If `INFO = -i`, the i-th argument had an illegal value.
`INFO = 1`: Calculating the (skew-)symmetric (which one was computed depends on the parameter `SYMN`) Schur-form of a part of the matrix N failed.
`INFO = 2`: Calculating the (skew-)symmetric (which one was computed depends on the parameter `SYMH`) Schur-form of a part of the matrix H failed.
`INFO = 3`: The `LAPACK` routine `DGESVD` returned with an info value greater than zero, i.e., `DGESVD` was not able to compute a SVD.

References

[1] R. Byers, V. Mehrmann, and H. Xu. A structured staircase algorithm for skew-symmetric/symmetric pencils. *Electr. Trans. Num. Anal.*, 26:1–33, 2007.

[2] J. W. Demmel and B. Kågström. The generalized Schur decomposition of an arbitrary pencil A - λB: Robust software with error bounds and applications. Part II: Software and applications. *ACM Trans. Math. Software*, 19(2):175–201, 1993.

[3] R. C. Thompson. Pencils of complex and real symmetric and skew matrices. *Linear Algebra Appl.*, 147:323–371, 1991.

5 Example

We consider the following even matrix pencil which is taken from [1].

$$\alpha N - \beta H = \alpha \begin{bmatrix} 0 & 1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & -1 & 0 \end{bmatrix} - \beta \begin{bmatrix} 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 4 \end{bmatrix} \tag{7}$$

5.1 Program Text

```
      PROGRAM STCSSP_example
      implicit none
C
C     DEMO: Demonstration program for STCSSP
C
*
*     .. Parameters ..
      INTEGER            NIN, NOUT
      PARAMETER          ( NIN = 5, NOUT = 6 )
      INTEGER            NMAX
      PARAMETER          ( NMAX = 20 )
      INTEGER            LDN, LDH, LDU
      PARAMETER          ( LDN = NMAX, LDH = NMAX, LDU = NMAX )
      INTEGER            LDWORK
      PARAMETER          ( LDWORK = NMAX*NMAX + 3*NMAX + 3 )
*     .. Local Scalars ..
      CHARACTER*1        SYMN, SYMH, UPLON, UPLOH, COMPZ
      INTEGER            NDIM, M, P, L
      INTEGER            PINVEC(NMAX+1), NUNVEC(NMAX+1),
     1                   PIVEC(NMAX),  NUVEC(NMAX)
      INTEGER            NVEC(NMAX), QVEC(NMAX)
      DOUBLE PRECISION   N(NMAX,NMAX), H(NMAX,NMAX), U(NMAX,NMAX)
      DOUBLE PRECISION   TOL1
      DOUBLE PRECISION   DWORK(LDWORK)
      INTEGER            INFO
      INTEGER            I, J
*     .. External Functions ..
      LOGICAL            LSAME
      EXTERNAL           LSAME
*     .. External Subroutines ..
      EXTERNAL           STCSSP
*
      WRITE ( NOUT, FMT = 99999 )
*     Skip the heading in the data file and read the data.
      READ ( NIN, FMT = '()' )
      READ ( NIN, FMT = * ) SYMN, SYMH, UPLON, UPLOH, COMPZ
      READ ( NIN, FMT = * ) NDIM
      READ ( NIN, FMT = * ) TOL1
      IF ( NDIM.LT.0 .OR. NDIM.GT.NMAX ) THEN
         WRITE ( NOUT, FMT = 99991 ) N
      ELSE
         READ ( NIN, FMT = * ) ( ( N(I,J), J = 1,NDIM ), I = 1,NDIM )
         READ ( NIN, FMT = * ) ( ( H(I,J), J = 1,NDIM ), I = 1,NDIM )
C
```

```
C         Run the structured staircase algorithm
C
          CALL STCSSP( SYMN, SYMH, UPLON, UPLOH, COMPZ, NDIM, N, LDN,
     $                    H, LDH, U, LDU, M, PINVEC, NUNVEC, PIVEC,
     $                    NUVEC, NVEC, QVEC, P, L, TOL1, DWORK, LDWORK,
     $                    INFO )
C
C         Complete the matrices
C
          CALL SUPMAT( SYMN, UPLON, N, LDN, NDIM )
          CALL SUPMAT( SYMH, UPLOH, H, LDH, NDIM )
C
C         Show output to console
C
          IF ( INFO.NE.0 ) THEN
             WRITE ( NOUT, FMT = 99998 ) INFO
          ELSE
             WRITE ( NOUT, FMT = 99997 ) P, L, M
             WRITE ( NOUT, FMT = 99996 )
             DO 20 I = 1, NDIM
                WRITE ( NOUT, FMT = 99993 ) ( N(I,J), J = 1,NDIM )
   20        CONTINUE
             WRITE ( NOUT, FMT = 99995 )
             DO 30 I = 1, NDIM
                WRITE ( NOUT, FMT = 99993 ) ( H(I,J), J = 1,NDIM )
   30        CONTINUE
             IF( LSAME(COMPZ,'V') ) THEN
                WRITE ( NOUT, FMT = 99994 )
                DO 40 I = 1, NDIM
                   WRITE ( NOUT, FMT = 99993 ) ( U(I,J), J = 1,NDIM )
   40           CONTINUE
             END IF
             WRITE ( NOUT, FMT = 99990 )
             DO 50 I = 1,M+1
                WRITE ( NOUT, FMT = 99980 ) I, PINVEC(I), NUNVEC(I)
   50        CONTINUE
             WRITE ( NOUT, FMT = 99989 )
             DO 60 I = 1,M
                WRITE ( NOUT, FMT = 99980 ) I, PIVEC(I), NUVEC(I)
   60        CONTINUE
             WRITE ( NOUT, FMT = 99988 )
             DO 70 I = 1,M
                WRITE ( NOUT, FMT = 99980 ) I, NVEC(I), QVEC(I)
   70        CONTINUE
          END IF
      END IF
      STOP
C
99999 FORMAT ('␣STCSSP␣EXAMPLE␣PROGRAM',/1X,
     1        'Computing␣staircase␣form␣of␣the␣pencil␣alpha␣N␣-␣beta␣H')
99998 FORMAT ('␣STCSPP␣returned␣in␣failure␣with␣INFO␣=␣',I2)
99997 FORMAT (/'␣P␣␣␣=␣␣␣',I3,'␣;␣␣L␣␣␣=␣␣␣',I3,'␣;␣␣M␣␣␣=␣␣␣',I3)
99996 FORMAT (/'␣The␣staircase␣form␣of␣matrix␣N␣is␣')
99995 FORMAT (/'␣The␣staircase␣form␣of␣matrix␣H␣is␣')
99994 FORMAT (/'␣The␣orthogonal␣transformation␣U␣is␣')
99993 FORMAT (20(1X,F8.4))
99992 FORMAT (/'␣' )
```

```
99991 FORMAT (/'␣Dimension␣of␣the␣problem␣is␣out␣of␣range.',/'␣N␣=␣',I5)
99990 FORMAT (/'␣The␣characteristic␣values␣for␣N␣are',/'␣␣␣i',6X,
     1        'PI(i)',6X,'NU(i)')
99989 FORMAT (/'␣The␣characteristic␣values␣for␣H␣are',/'␣␣␣i',6X,
     1        'PI(i)',6X,'NU(i)')
99988 FORMAT (/'␣The␣dimension␣of␣the␣blocks␣are',/'␣␣␣i',6X,
     1        'N(i)',6X,'Q(i)')
99980 FORMAT (I4,3X,I7,3X,I7)
      END PROGRAM STCSSP_example
C
C     Subroutine to complete the returned matrix in all positions
C
      SUBROUTINE SUPMAT( SYM, UPLO, A, LDA, N )
      implicit none
C     .. Parameters ..
      DOUBLE PRECISION        ZERO, ONE
      PARAMETER               ( ZERO = 0.0D0, ONE = 1.0D0 )
C     .. Arguments ..
      CHARACTER*1             SYM, UPLO
      DOUBLE PRECISION        A(LDA,*), FACT
      INTEGER                 LDA, N, I, J
C     .. External Functions ..
      LOGICAL                 LSAME
      EXTERNAL                LSAME
C
C     Handle the strict upper/lower triangular part
C
      IF( LSAME( SYM, 'S' ) ) THEN
         FACT =  ONE
      ELSE
         FACT = -ONE
      END IF
      IF( LSAME( UPLO, 'U' ) ) THEN
         DO 200 I=1,N-1
            DO 100 J=I+1,N
               A(J,I) = FACT * A(I,J)
  100       CONTINUE
  200    CONTINUE
      ELSE
         DO 400 I=1,N-1
            DO 300 J=I+1,N
               A(I,J) = FACT * A(J,I)
  300       CONTINUE
  400    CONTINUE
      END IF
C
C     Handle the diagonal
C
      IF( LSAME( SYM, 'N' ) ) THEN
         DO 500 I=1, N
            A(I,I) = ZERO
  500    CONTINUE
      END IF
C
      END SUBROUTINE SUPMAT
```

5.2 Program Data

```
STCSSP Example Program Data
  N     S     U     L     V
  5
  1e-12
  -7      1      0      0      0
  -7     -7      0      0      0
  -7     -7     -7      0      0
  -7     -7     -7     -7      1
  -7     -7     -7     -7     -7
   0     -7     -7     -7     -7
   0      1     -7     -7     -7
   1      0      0     -7     -7
   0      0      0      1     -7
   0      0      0      0      4
```

5.3 Program Results

```
STCSSP EXAMPLE PROGRAM
Computing staircase form of the pencil alpha N - beta H

P   =     2 ;  L   =     3 ;  M   =     2

The staircase form of matrix N is
  0.0000   0.0000   0.0000   1.0000   0.0000
  0.0000   0.0000  -1.0000   0.0000   0.0000
  0.0000   1.0000   0.0000   0.0000   0.0000
 -1.0000   0.0000   0.0000   0.0000   0.0000
  0.0000   0.0000   0.0000   0.0000   0.0000

The staircase form of matrix H is
  0.0000   0.0000   0.0000   0.0000   1.0000
  0.0000   1.0000   0.0000   0.0000   0.0000
  0.0000   0.0000   4.0000   0.0000   0.0000
  0.0000   0.0000   0.0000   1.0000   0.0000
  1.0000   0.0000   0.0000   0.0000   0.0000

The orthogonal transformation U is
 -1.0000   0.0000   0.0000   0.0000   0.0000
  0.0000   0.0000   0.0000  -1.0000   0.0000
  0.0000   0.0000   0.0000   0.0000  -1.0000
  0.0000  -1.0000   0.0000   0.0000   0.0000
  0.0000   0.0000   1.0000   0.0000   0.0000

The characteristic values for N are
  i      PI(i)      NU(i)
  1        0          0
  2        0          0
  3        0          0

The characteristic values for H are
  i      PI(i)      NU(i)
  1        0          0
  2        1          0

The dimension of the blocks are
  i      N(i)      Q(i)
  1        1         1
  2        0         0
```