Locking and Purging for the Hamiltonian Lanczos Process

Martin Stoll

Professur Mathematik in Industrie und Technik
Fakultät für Mathematik
Technische Universität Chemnitz

13. September 2005

Fakultät für Mathematik
TECHNISCHE UNIVERSITÄT CHEMNITZ

Basic definitions

Definition

A matrix S is symplectic $\iff S^T J S = J$ with $J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}$.

Definition

$$\text{A matrix } H \text{ is Hamiltonian} \iff (JH)^T = JH$$

$$\iff H = \begin{bmatrix} A & G \\ Q & -A^T \end{bmatrix}, \quad \begin{matrix} G = G^T \\ Q = Q^T \end{matrix}$$

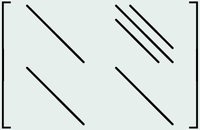
Properties

- If H is Hamiltonian and S is symplectic then $S^{-1}HS$ is Hamiltonian.
- If $\lambda \in \mathbb{R}$ is an eigenvalue of H then $-\lambda$ is also an eigenvalue of H.
- If $\lambda \in \mathbb{C}$ is an eigenvalue of H then $-\lambda, \bar{\lambda}, -\bar{\lambda}$ are also an eigenvalue of H.

Definition

A Hamiltonian J-Hessenberg matrix H is of the following structure

$H =$



SR decomposition

Definition

For $A \in \mathbb{R}^{2n,2n}$ is $A = SR$, the SR decomposition with a symplectic matrix S and a J-triangular matrix R.

- The SR decomposition is unique $\Rightarrow$ an equivalent to the implicit Q theorem holds.
- If H is a Hamiltonian J-Hessenberg matrix and p_k is a polynomial, then $S^{-1}HS$ is also a Hamiltonian J-Hessenberg matrix with S from the decomposition of $p_k(H) = SR$.

Implicit shifts

1. Single shift $(H - \mu I)$
2. Double shift $(H - \mu I)(H + \mu I)$
3. Quadrupel shift $(H - \mu I)(H + \bar{\mu} I)(H + \mu I)(H - \bar{\mu} I)$
    - Compute first column v
    - Compute transformation W such that $Wv = \alpha e_1$
    - Compute bulge chasing algorithm for $W^{-1}HW$

SR algorithm

SR algorithm

Perform implicit shifts until the permuted form H_p of the Hamiltonian J-Hessenberg matrix H is deflated and the blocks are of size 2×2 or 4×4.

$$H = \left[\begin{array}{ccc|ccc} \delta_1 & & & \beta_1 & \zeta_2 & \\ & \ddots & & \zeta_2 & \ddots & \\ & & \delta_n & & \zeta_n & \beta_n \\ \hline \nu_1 & & & -\delta_1 & & \\ & \ddots & & & \ddots & \\ & & \nu_n & & & -\delta_n \end{array}\right]$$

Eigenvalue and eigenvector computation

Eigenvalue computation

Only blocks of size 2×2 and 4×4 have to be considered. Explicit formulas for computing λ could be given.

Eigenvector computation

If $Re(\lambda) < 0$ easy eigenvector computation out of the SR algorithm by [Mehrmann/Bunse-Gerstner,1986].
If $Re(\lambda) > 0$ two ways for the eigenvector computation were developed

- Use inverse iteration with consideration of the sparse Hamiltonian structure.
- A Schur-like form could be used making advantage of the sparse Hamiltonian structure.

The symplectic Lanczos algorithm

- Use Lanczos recursion $H_p S_p^{2n,2k} = S_p^{2n,2k} \tilde{H}_p^{2k,2k} + \zeta_{k+1} v_{k+1} e_{2k}^T$
- Lanczos vectors $S_p^{2n,2k} = [v_1, w_1, v_2, w_2, \ldots, v_k, w_k]$ could be obtained
- Parameters of $\tilde{H}_p$ are computed

Symplectic Lanczos algorithm [Benner/Faßbender,1997]

1. $\delta_m = v_m^T H_p v_m$
2. $\tilde{w}_m = H_p v_m - \delta_m v_m$
3. $\nu_m = v_m^T J_p H_p v_m$
4. $w_m = \frac{1}{\nu_m} \tilde{w}_m$
5. $\beta_m = -w_m^T J_p H_p w_m$
6. $\tilde{v}_{m+1} = H_p w_m - \zeta_m v_{m-1} - \beta_m v_m + \delta_m w_m$
7. $\zeta_{m+1} = \|\tilde{v}_{m+1}\|_2$
8. $v_{m+1} = \frac{1}{\zeta_{m+1}} \tilde{v}_{m+1}$

The symplectic Lanczos algorithm

- Use Lanczos recursion $H_p S_p^{2n,2k} = S_p^{2n,2k} \tilde{H}_p^{2k,2k} + \zeta_{k+1} v_{k+1} e_{2k}^T$
- Lanczos vectors $S_p^{2n,2k} = [v_1, w_1, v_2, w_2, \ldots, v_k, w_k]$ could be obtained
- Parameters of $\tilde{H}_p$ are computed

Symplectic Lanczos algorithm [Benner/Faßbender,1997]

1. $\delta_m = v_m^T H_p v_m$
2. $\tilde{w}_m = H_p v_m - \delta_m v_m$
3. $\nu_m = v_m^T J_p H_p v_m$
4. $w_m = \frac{1}{\nu_m} \tilde{w}_m$
5. $\beta_m = -w_m^T J_p H_p w_m$
6. $\tilde{v}_{m+1} = H_p w_m - \zeta_m v_{m-1} - \beta_m v_m + \delta_m w_m$
7. $\zeta_{m+1} = \|\tilde{v}_{m+1}\|_2$
8. $v_{m+1} = \frac{1}{\zeta_{m+1}} \tilde{v}_{m+1}$

The implicitly restarted symplectic Lanczos algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$
2. $(\tilde{H}_p^{2m} - \mu_i I)(\tilde{H}_p^{2m} + \mu_i I)(\tilde{H}_p^{2m} - \mu_{i+1} I)(\tilde{H}_p^{2m} + \mu_{i+1} I) = S_p^{(i)} R_p^{(i)}$
3. $H_p \check{S}_p^{2n,2m} = \check{S}_p^{2n,2m} \check{H}_p^{2m,2m} + \check{\zeta}_{m+1} \check{v}_{m+1} e_{2m}^T S_p$, with $\check{S}_p^{2n,2m} = S_p^{2n,2m} S_p$ and $\check{H}_p^{2m,2m} = S_p^{-1} \tilde{H}_p^{2m,2m} S_p$
4. Residual term

$$\zeta_{m+1} v_{m+1} (s_{2m,2m-4q} e_{2m-4q}^T + s_{2m,2m-3q} e_{2m-3q}^T + \ldots + s_{2m,2m} e_{2m}^T)$$

5. $H_p \check{S}_p^{2n,2k} = \check{S}_p^{2n,2k} \check{H}^{2k,2k} + r_k e_{2k}^T$
6. Residual $r_k = \check{\zeta}_{k+1} \check{v}_{k+1} + \zeta_{m+1} s_{2m,2k} v_{m+1}$

The implicitly restarted symplectic Lanczos algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$
2. $(\tilde{H}_p^{2m} - \mu_i I)(\tilde{H}_p^{2m} + \mu_i I)(\tilde{H}_p^{2m} - \mu_{i+1} I)(\tilde{H}_p^{2m} + \mu_{i+1} I) = S_p^{(i)} R_p^{(i)}$
3. $H_p \check{S}_p^{2n,2m} = \check{S}_p^{2n,2m} \check{H}_p^{2m,2m} + \check{\zeta}_{m+1} \check{v}_{m+1} e_{2m}^T S_p$, with $\check{S}_p^{2n,2m} = S_p^{2n,2m} S_p$ and $\check{H}_p^{2m,2m} = S_p^{-1} \tilde{H}_p^{2m,2m} S_p$
4. Residual term

$$\zeta_{m+1} v_{m+1} (s_{2m,2m-4q} e_{2m-4q}^T + s_{2m,2m-3q} e_{2m-3q}^T + \ldots + s_{2m,2m} e_{2m}^T)$$

5. $H_p \check{S}_p^{2n,2k} = \check{S}_p^{2n,2k} \check{H}^{2k,2k} + r_k e_{2k}^T$
6. Residual $r_k = \check{\zeta}_{k+1} \check{v}_{k+1} + \zeta_{m+1} s_{2m,2k} v_{m+1}$

The implicitly restarted symplectic Lanczos algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$
2. $(\tilde{H}_p^{2m} - \mu_i I)(\tilde{H}_p^{2m} + \mu_i I)(\tilde{H}_p^{2m} - \mu_{i+1} I)(\tilde{H}_p^{2m} + \mu_{i+1} I) = S_p^{(i)} R_p^{(i)}$
3. $H_p \check{S}_p^{2n,2m} = \check{S}_p^{2n,2m} \check{H}_p^{2m,2m} + \check{\zeta}_{m+1} \check{v}_{m+1} e_{2m}^T S_p$, with $\check{S}_p^{2n,2m} = S_p^{2n,2m} S_p$ and $\check{H}_p^{2m,2m} = S_p^{-1} \tilde{H}_p^{2m,2m} S_p$
4. Residual term

$$\zeta_{m+1} v_{m+1} (s_{2m,2m-4q} e_{2m-4q}^T + s_{2m,2m-3q} e_{2m-3q}^T + \ldots + s_{2m,2m} e_{2m}^T)$$

5. $H_p \check{S}_p^{2n,2k} = \check{S}_p^{2n,2k} \check{H}^{2k,2k} + r_k e_{2k}^T$
6. Residual $r_k = \check{\zeta}_{k+1} \check{v}_{k+1} + \zeta_{m+1} s_{2m,2k} v_{m+1}$

The implicitly restarted symplectic Lanczos algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$
2. $(\tilde{H}_p^{2m} - \mu_i I)(\tilde{H}_p^{2m} + \mu_i I)(\tilde{H}_p^{2m} - \mu_{i+1} I)(\tilde{H}_p^{2m} + \mu_{i+1} I) = S_p^{(i)} R_p^{(i)}$
3. $H_p \check{S}_p^{2n,2m} = \check{S}_p^{2n,2m} \check{H}_p^{2m,2m} + \check{\zeta}_{m+1} \check{v}_{m+1} e_{2m}^T S_p$, with $\check{S}_p^{2n,2m} = S_p^{2n,2m} S_p$ and $\check{H}_p^{2m,2m} = S_p^{-1} \tilde{H}_p^{2m,2m} S_p$
4. Residual term

$$\zeta_{m+1} v_{m+1} (s_{2m,2m-4q} e_{2m-4q}^T + s_{2m,2m-3q} e_{2m-3q}^T + \ldots + s_{2m,2m} e_{2m}^T)$$

5. $H_p \check{S}_p^{2n,2k} = \check{S}_p^{2n,2k} \check{H}^{2k,2k} + r_k e_{2k}^T$
6. Residual $r_k = \check{\zeta}_{k+1} \check{v}_{k+1} + \zeta_{m+1} s_{2m,2k} v_{m+1}$

The implicitly restarted symplectic Lanczos algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$
2. $(\tilde{H}_p^{2m} - \mu_i I)(\tilde{H}_p^{2m} + \mu_i I)(\tilde{H}_p^{2m} - \mu_{i+1} I)(\tilde{H}_p^{2m} + \mu_{i+1} I) = S_p^{(i)} R_p^{(i)}$
3. $H_p \check{S}_p^{2n,2m} = \check{S}_p^{2n,2m} \check{H}_p^{2m,2m} + \check{\zeta}_{m+1} \check{v}_{m+1} e_{2m}^T S_p$, with $\check{S}_p^{2n,2m} = S_p^{2n,2m} S_p$ and $\check{H}_p^{2m,2m} = S_p^{-1} \tilde{H}_p^{2m,2m} S_p$
4. Residual term

$$\zeta_{m+1} v_{m+1} (s_{2m,2m-4q} e_{2m-4q}^T + s_{2m,2m-3q} e_{2m-3q}^T + \ldots + s_{2m,2m} e_{2m}^T)$$

5. $H_p \check{S}_p^{2n,2k} = \check{S}_p^{2n,2k} \check{H}^{2k,2k} + r_k e_{2k}^T$
6. Residual $r_k = \check{\zeta}_{k+1} \check{v}_{k+1} + \zeta_{m+1} s_{2m,2k} v_{m+1}$

The implicitly restarted symplectic Lanczos algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$
2. $(\tilde{H}_p^{2m} - \mu_i I)(\tilde{H}_p^{2m} + \mu_i I)(\tilde{H}_p^{2m} - \mu_{i+1} I)(\tilde{H}_p^{2m} + \mu_{i+1} I) = S_p^{(i)} R_p^{(i)}$
3. $H_p \check{S}_p^{2n,2m} = \check{S}_p^{2n,2m} \check{H}_p^{2m,2m} + \check{\zeta}_{m+1} \check{v}_{m+1} e_{2m}^T S_p$, with $\check{S}_p^{2n,2m} = S_p^{2n,2m} S_p$ and $\check{H}_p^{2m,2m} = S_p^{-1} \tilde{H}_p^{2m,2m} S_p$
4. Residual term

$$\zeta_{m+1} v_{m+1} (s_{2m,2m-4q} e_{2m-4q}^T + s_{2m,2m-3q} e_{2m-3q}^T + \ldots + s_{2m,2m} e_{2m}^T)$$

5. $H_p \check{S}_p^{2n,2k} = \check{S}_p^{2n,2k} \check{H}^{2k,2k} + r_k e_{2k}^T$
6. Residual $r_k = \check{\zeta}_{k+1} \check{v}_{k+1} + \zeta_{m+1} s_{2m,2k} v_{m+1}$

Stopping criteria for IRSL

1. Assume $\tilde{H}^{2k,2k}$ could be diagonalized by usage of Y.
2. $HSY = SYY^{-1}\tilde{H}Y + \tilde{r}_k e_{2k}^T Y$ or $HX = X\Lambda + \tilde{r}_k e_{2k}^T Y$
3. Consider columns $Hx_i = -\lambda_i x_i + y_{2k,i}\tilde{r}_k$ and $Hx_{k+i} = \lambda_i x_{k+i} + y_{2k,k+i}\tilde{r}_k$
4. From the last equations we get eigentriples $(\lambda_i, x_{k+i}, (Jx_i)^T)$ and $(-\lambda_i, x_i, (Jx_{k+i})^T)$
5. From [Kahan/Parlett/Jiang,1982] backward error $\|E_{\lambda_i}\|$,where $(H - E_{\lambda_i})x = \lambda_i x$, could be computed

$$\|E_{\lambda_i}\| = \max\left\{\frac{|y_{2k,k+i}|\|\tilde{r}_k\|}{\|x_{k+i}\|}, \frac{|y_{2k,i}|\|\tilde{r}_k^T J\|}{\|Jx_i\|}\right\}.$$

Stopping criteria for IRSL

1. Assume $\tilde{H}^{2k,2k}$ could be diagonalized by usage of Y.
2. $HSY = SYY^{-1}\tilde{H}Y + \tilde{r}_k e_{2k}^T Y$ or $HX = X\Lambda + \tilde{r}_k e_{2k}^T Y$
3. Consider columns $Hx_i = -\lambda_i x_i + y_{2k,i}\tilde{r}_k$ and $Hx_{k+i} = \lambda_i x_{k+i} + y_{2k,k+i}\tilde{r}_k$
4. From the last equations we get eigentriples $(\lambda_i, x_{k+i}, (Jx_i)^T)$ and $(-\lambda_i, x_i, (Jx_{k+i})^T)$
5. From [Kahan/Parlett/Jiang,1982] backward error $\|E_{\lambda_i}\|$,where $(H - E_{\lambda_i})x = \lambda_i x$, could be computed

$$\|E_{\lambda_i}\| = \max\left\{\frac{|y_{2k,k+i}|\|\tilde{r}_k\|}{\|x_{k+i}\|}, \frac{|y_{2k,i}|\|\tilde{r}_k^T J\|}{\|Jx_i\|}\right\}.$$

Stopping criteria for IRSL

1. Assume $\tilde{H}^{2k,2k}$ could be diagonalized by usage of Y.
2. $HSY = SYY^{-1}\tilde{H}Y + \tilde{r}_k e_{2k}^T Y$ or $HX = X\Lambda + \tilde{r}_k e_{2k}^T Y$
3. Consider columns $Hx_i = -\lambda_i x_i + y_{2k,i}\tilde{r}_k$ and $Hx_{k+i} = \lambda_i x_{k+i} + y_{2k,k+i}\tilde{r}_k$
4. From the last equations we get eigentriples $(\lambda_i, x_{k+i}, (Jx_i)^T)$ and $(-\lambda_i, x_i, (Jx_{k+i})^T)$
5. From [Kahan/Parlett/Jiang,1982] backward error $\|E_{\lambda_i}\|$,where $(H - E_{\lambda_i})x = \lambda_i x$, could be computed

$$\|E_{\lambda_i}\| = \max\left\{\frac{|y_{2k,k+i}|\|\tilde{r}_k\|}{\|x_{k+i}\|}, \frac{|y_{2k,i}|\|\tilde{r}_k^T J\|}{\|Jx_i\|}\right\}.$$

Stopping criteria for IRSL

1. Assume $\tilde{H}^{2k,2k}$ could be diagonalized by usage of Y.
2. $HSY = SYY^{-1}\tilde{H}Y + \tilde{r}_k e_{2k}^T Y$ or $HX = X\Lambda + \tilde{r}_k e_{2k}^T Y$
3. Consider columns $Hx_i = -\lambda_i x_i + y_{2k,i}\tilde{r}_k$ and $Hx_{k+i} = \lambda_i x_{k+i} + y_{2k,k+i}\tilde{r}_k$
4. From the last equations we get eigentriples $(\lambda_i, x_{k+i}, (Jx_i)^T)$ and $(-\lambda_i, x_i, (Jx_{k+i})^T)$
5. From [Kahan/Parlett/Jiang,1982] backward error $\|E_{\lambda_i}\|$,where $(H - E_{\lambda_i})x = \lambda_i x$, could be computed

$$\|E_{\lambda_i}\| = \max\left\{\frac{|y_{2k,k+i}|\|\tilde{r}_k\|}{\|x_{k+i}\|}, \frac{|y_{2k,i}|\|\tilde{r}_k^T J\|}{\|Jx_i\|}\right\}.$$

Stopping criteria for IRSL

1. Assume $\tilde{H}^{2k,2k}$ could be diagonalized by usage of Y.
2. $HSY = SYY^{-1}\tilde{H}Y + \tilde{r}_k e_{2k}^T Y$ or $HX = X\Lambda + \tilde{r}_k e_{2k}^T Y$
3. Consider columns $Hx_i = -\lambda_i x_i + y_{2k,i}\tilde{r}_k$ and $Hx_{k+i} = \lambda_i x_{k+i} + y_{2k,k+i}\tilde{r}_k$
4. From the last equations we get eigentriples $(\lambda_i, x_{k+i}, (Jx_i)^T)$ and $(-\lambda_i, x_i, (Jx_{k+i})^T)$
5. From [Kahan/Parlett/Jiang,1982] backward error $\|E_{\lambda_i}\|$,where $(H - E_{\lambda_i})x = \lambda_i x$, could be computed

$$\|E_{\lambda_i}\| = \max\left\{\frac{|y_{2k,k+i}|\|\tilde{r}_k\|}{\|x_{k+i}\|}, \frac{|y_{2k,i}|\|\tilde{r}_k^T J\|}{\|Jx_i\|}\right\}.$$

Krylov-Schur like decomposition

Krylov decomposition

Take $S_p^{2k} = (s_1, \ldots, s_{2k})$ with linear independent vectors s_j. A decomposition of the following structure $H_p S_p^{2k} = S_p^{2k} \tilde{H}_p^{2k} + s_{2k+1} \tilde{h}_{2k+1}^T$ is called *Krylov decomposition of order* $2k$.

Krylov-Schur like decomposition

Take $S_p^{2k} = (s_1, \ldots, s_{2k})$ with J-orthogonal vectors $s_j \Rightarrow$ $(S_p^{2k})^T J_p S_p^{2k} = J_p$ holds. A decomposition $H_p S_p^{2k} = S_p^{2k} \tilde{H}_p^{2k} + s_{2k+1} \tilde{h}_{2k+1}^T$ is called *Krylov-Schur like decomposition of order* $2k$, where $\tilde{H}_p^{2k}$ is a permuted Hamiltonian J-Hessenberg matrix in Schur form.

Krylov-Schur like decomposition

Krylov decomposition

Take $S_p^{2k} = (s_1, \ldots, s_{2k})$ with linear independent vectors s_j. A decomposition of the following structure $H_p S_p^{2k} = S_p^{2k} \tilde{H}_p^{2k} + s_{2k+1} \tilde{h}_{2k+1}^T$ is called *Krylov decomposition of order* $2k$.

Krylov-Schur like decomposition

Take $S_p^{2k} = (s_1, \ldots, s_{2k})$ with J-orthogonal vectors $s_j \Rightarrow$ $(S_p^{2k})^T J_p S_p^{2k} = J_p$ holds. A decomposition $H_p S_p^{2k} = S_p^{2k} \tilde{H}_p^{2k} + s_{2k+1} \tilde{h}_{2k+1}^T$ is called *Krylov-Schur like decomposition of order* $2k$, where $\tilde{H}_p^{2k}$ is a permuted Hamiltonian J-Hessenberg matrix in Schur form.

Lanczos factorisation → Krylov-Schur like decomposition

1. Start with **Lanczos factorisation**

$$H_p S_p^{2n,2k} = S_p^{2n,2k} \tilde{H}_p^{2k,2k} + r_k e_{2k}^T.$$

2. Take S_p form SR algorithm to get $H_p S_p^{2n,2k} S_p = S_p^{2n,2k} S_p (S_p^{-1} \tilde{H}_p^{2k,2k} S_p) + r_k e_{2k}^T S_p$
3. Result is **Krylov-Schur like decompostion**

$$H_p \breve{S}_p^{2n,2k} = \breve{S}_p^{2n,2k} \breve{H}_p^{2k,2k} + r_k s_p^T$$

where $\breve{S}_p = S_p^{2n,2k} S_p, \breve{H}_p^{2k,2k} = (S_p^{-1} \tilde{H}_p^{2k,2k} S_p)$ and $s_p^T = e_{2k}^T S_p$.

Lanczos factorisation → Krylov-Schur like decomposition

1. Start with **Lanczos factorisation**

$$H_p S_p^{2n,2k} = S_p^{2n,2k} \tilde{H}_p^{2k,2k} + r_k e_{2k}^T.$$

2. Take S_p form SR algorithm to get
$H_p S_p^{2n,2k} S_p = S_p^{2n,2k} S_p (S_p^{-1} \tilde{H}_p^{2k,2k} S_p) + r_k e_{2k}^T S_p$

3. Result is **Krylov-Schur like decompostion**

$$H_p \breve{S}_p^{2n,2k} = \breve{S}_p^{2n,2k} \breve{H}_p^{2k,2k} + r_k s_p^T$$

where $\breve{S}_p = S_p^{2n,2k} S_p, \breve{H}_p^{2k,2k} = (S_p^{-1} \tilde{H}_p^{2k,2k} S_p)$ and $s_p^T = e_{2k}^T S_p$.

Lanczos factorisation → Krylov-Schur like decomposition

1. Start with **Lanczos factorisation**

$$H_p S_p^{2n,2k} = S_p^{2n,2k} \tilde{H}_p^{2k,2k} + r_k e_{2k}^T.$$

2. Take S_p form SR algorithm to get
$H_p S_p^{2n,2k} S_p = S_p^{2n,2k} S_p (S_p^{-1} \tilde{H}_p^{2k,2k} S_p) + r_k e_{2k}^T S_p$
3. Result is **Krylov-Schur like decompostion**

$$H_p \breve{S}_p^{2n,2k} = \breve{S}_p^{2n,2k} \breve{H}_p^{2k,2k} + r_k s_p^T$$

where $\breve{S}_p = S_p^{2n,2k} S_p, \breve{H}_p^{2k,2k} = (S_p^{-1} \tilde{H}_p^{2k,2k} S_p)$ and $s_p^T = e_{2k}^T S_p$.

Krylov-Schur like decomposition → Lanczos factorisation

1. Start with **Krylov-Schur like decomposition** $H_p S_p^{2k} = S_p^{2k} \tilde{H}_p^{2k} + s_{2k+1} \tilde{h}_{2k+1}^T$
2. Compute matrix Q_p (product of Givens matrices) such that $\tilde{h}_{2k+1}^T Q_p = \alpha e_{2k}^T$ multiplication with Q_p results in $H_p S_p^{2k} Q_p = \check{S}_p^{2k} \check{H}_p^{2k} + \alpha s_{2k+1} e_{2k}^T$.
3. Transform $\check{H}_p^{2k}$ to Hamiltonian J-Hessenberg form. Use rowwise variation of JHESS algorithm by [Mehrmann/Bunse-Gerstner]. Structure of $\alpha s_{2k+1} e_{2k}^T$ is preserved and the result is a **Lanczos factorisation** $H_p \check{S}_p^{2k} = \check{S}_p^{2k} \check{H}_p^{2k} + \check{s}_{2k+1} e_{2k}^T$.

Krylov-Schur like decomposition → Lanczos factorisation

1. Start with **Krylov-Schur like decomposition** $H_p S_p^{2k} = S_p^{2k} \tilde{H}_p^{2k} + s_{2k+1} \tilde{h}_{2k+1}^T$
2. Compute matrix Q_p (product of Givens matrices) such that $\tilde{h}_{2k+1}^T Q_p = \alpha e_{2k}^T$ multiplication with Q_p results in $H_p S_p^{2k} Q_p = \check{S}_p^{2k} \check{H}_p^{2k} + \alpha s_{2k+1} e_{2k}^T$.
3. Transform $\check{H}_p^{2k}$ to Hamiltonian J-Hessenberg form. Use rowwise variation of JHESS algorithm by [Mehrmann/Bunse-Gerstner]. Structure of $\alpha s_{2k+1} e_{2k}^T$ is preserved and the result is a **Lanczos factorisation** $H_p \check{S}_p^{2k} = \check{S}_p^{2k} \check{H}_p^{2k} + \check{s}_{2k+1} e_{2k}^T$.

Krylov-Schur like decomposition → Lanczos factorisation

1. Start with **Krylov-Schur like decomposition** $H_p S_p^{2k} = S_p^{2k} \tilde{H}_p^{2k} + s_{2k+1} \tilde{h}_{2k+1}^T$
2. Compute matrix Q_p (product of Givens matrices) such that $\tilde{h}_{2k+1}^T Q_p = \alpha e_{2k}^T$ multiplication with Q_p results in $H_p S_p^{2k} Q_p = \check{S}_p^{2k} \check{H}_p^{2k} + \alpha s_{2k+1} e_{2k}^T$.
3. Transform $\check{H}_p^{2k}$ to Hamiltonian J-Hessenberg form. Use rowwise variation of JHESS algorithm by [Mehrmann/Bunse-Gerstner]. Structure of $\alpha s_{2k+1} e_{2k}^T$ is preserved and the result is a **Lanczos factorisation** $H_p \check{S}_p^{2k} = \check{S}_p^{2k} \check{H}_p^{2k} + \check{s}_{2k+1} e_{2k}^T$.

Exchanging eigenblocks

We have now a matrix $\check{H}_p^{2k}$ in block diagonal form. If we exchange the eigenblocks by using symplectic transformations we will be able to implement Purging and Locking. The permuted symplectic matrix $T_p(n)$ allows to exchange eigenblocks in $\check{H}_p^{2k}$.

$$T_p(n) = \begin{bmatrix} & & & & 1 \\ & & & -1 & \\ & & \cdot^{\cdot^{\cdot}} & & \\ & 1 & & & \\ -1 & & & & \end{bmatrix} \in \mathbb{R}^{2n,2n}.$$

Stopping Criteria for Krylov-Schur like algorithm

1. Assume $\breve{H}^{2k,2k}$ could be diagonalized by usage of Y.
2. $H\breve{S}Y = \breve{S}YY^{-1}\breve{H}Y + \breve{r}_k q^T Y$ or $HX = X\Lambda + \breve{r}_k q^T Y$
3. Consider columns $Hx_i = -\lambda_i x_i + \breve{r}_k q^T y_i$ and $Hx_{k+i} = \lambda_i x_{k+i} + \breve{r}_k q^T y_{k+i}$
4. From the last equations we get eigentriples $(\lambda_i, x_{k+i}, (Jx_i)^T)$ and $(-\lambda_i, x_i, (Jx_{k+i})^T)$
5. From [Kahan/Parlett/Jiang,1982] backward error $\|E_{\lambda_i}\|$,where $(H - E_{\lambda_i})x = \lambda_i x$, could be computed

$$\|E_{\lambda_i}\| = \max\left\{\frac{\|\breve{r}_k q^T y_{k+i}\|}{\|x_{k+i}\|}, \frac{\|\breve{r}_k^T q y_i^T J\|}{\|Jx_i\|}\right\}$$

Stopping Criteria for Krylov-Schur like algorithm

1. Assume $\breve{H}^{2k,2k}$ could be diagonalized by usage of Y.
2. $H\breve{S}Y = \breve{S}YY^{-1}\breve{H}Y + \breve{r}_k q^T Y$ or $HX = X\Lambda + \breve{r}_k q^T Y$
3. Consider columns $Hx_i = -\lambda_i x_i + \breve{r}_k q^T y_i$ and $Hx_{k+i} = \lambda_i x_{k+i} + \breve{r}_k q^T y_{k+i}$
4. From the last equations we get eigentriples $(\lambda_i, x_{k+i}, (Jx_i)^T)$ and $(-\lambda_i, x_i, (Jx_{k+i})^T)$
5. From [Kahan/Parlett/Jiang,1982] backward error $\|E_{\lambda_i}\|$,where $(H - E_{\lambda_i})x = \lambda_i x$, could be computed

$$\|E_{\lambda_i}\| = \max\left\{\frac{\|\breve{r}_k q^T y_{k+i}\|}{\|x_{k+i}\|}, \frac{\|\breve{r}_k^T q y_i^T J\|}{\|Jx_i\|}\right\}$$

Stopping Criteria for Krylov-Schur like algorithm

1. Assume $\breve{H}^{2k,2k}$ could be diagonalized by usage of Y.
2. $H\breve{S}Y = \breve{S}YY^{-1}\breve{H}Y + \breve{r}_k q^T Y$ or $HX = X\Lambda + \breve{r}_k q^T Y$
3. Consider columns $Hx_i = -\lambda_i x_i + \breve{r}_k q^T y_i$ and $Hx_{k+i} = \lambda_i x_{k+i} + \breve{r}_k q^T y_{k+i}$
4. From the last equations we get eigentriples $(\lambda_i, x_{k+i}, (Jx_i)^T)$ and $(-\lambda_i, x_i, (Jx_{k+i})^T)$
5. From [Kahan/Parlett/Jiang,1982] backward error $\|E_{\lambda_i}\|$,where $(H - E_{\lambda_i})x = \lambda_i x$, could be computed

$$\|E_{\lambda_i}\| = \max\left\{ \frac{\|\breve{r}_k q^T y_{k+i}\|}{\|x_{k+i}\|}, \frac{\|\breve{r}_k^T q y_i^T J\|}{\|Jx_i\|} \right\}$$

Stopping Criteria for Krylov-Schur like algorithm

1. Assume $\breve{H}^{2k,2k}$ could be diagonalized by usage of Y.
2. $H\breve{S}Y = \breve{S}YY^{-1}\breve{H}Y + \breve{r}_k q^T Y$ or $HX = X\Lambda + \breve{r}_k q^T Y$
3. Consider columns $Hx_i = -\lambda_i x_i + \breve{r}_k q^T y_i$ and $Hx_{k+i} = \lambda_i x_{k+i} + \breve{r}_k q^T y_{k+i}$
4. From the last equations we get eigentriples $(\lambda_i, x_{k+i}, (Jx_i)^T)$ and $(-\lambda_i, x_i, (Jx_{k+i})^T)$
5. From [Kahan/Parlett/Jiang,1982] backward error $\|E_{\lambda_i}\|$,where $(H - E_{\lambda_i})x = \lambda_i x$, could be computed

$$\|E_{\lambda_i}\| = \max\left\{\frac{\|\breve{r}_k q^T y_{k+i}\|}{\|x_{k+i}\|}, \frac{\|\breve{r}_k^T q y_i^T J\|}{\|Jx_i\|}\right\}$$

Stopping Criteria for Krylov-Schur like algorithm

1. Assume $\breve{H}^{2k,2k}$ could be diagonalized by usage of Y.
2. $H\breve{S}Y = \breve{S}YY^{-1}\breve{H}Y + \breve{r}_k q^T Y$ or $HX = X\Lambda + \breve{r}_k q^T Y$
3. Consider columns $Hx_i = -\lambda_i x_i + \breve{r}_k q^T y_i$ and $Hx_{k+i} = \lambda_i x_{k+i} + \breve{r}_k q^T y_{k+i}$
4. From the last equations we get eigentriples $(\lambda_i, x_{k+i}, (Jx_i)^T)$ and $(-\lambda_i, x_i, (Jx_{k+i})^T)$
5. From [Kahan/Parlett/Jiang,1982] backward error $\|E_{\lambda_i}\|$,where $(H - E_{\lambda_i})x = \lambda_i x$, could be computed

$$\|E_{\lambda_i}\| = \max\left\{\frac{\|\breve{r}_k q^T y_{k+i}\|}{\|x_{k+i}\|}, \frac{\|\breve{r}_k^T q y_i^T J\|}{\|Jx_i\|}\right\}$$

Locking

- For the **wanted** eigenvalue λ_i $\|E_{\lambda_i}\|$ is sufficiently small.
- The corresponding eigenblock in $\breve{H}_p^{2k,2k}$ has to be moved to the left upper corner of $\breve{H}_p^{2k,2k}$ using the T_p transformations.
- Accumulation of this transformations in $\breve{S}_p$ guarantees that the first columns of S_p are associated with λ_i.
- Shrink $\breve{H}_p$ such that the eigenblock for λ_i will not be involved in further computations.
- The J-reorthogonalisation has to be done against all columns of $\breve{S}_p$.

Locking

- For the **wanted** eigenvalue λ_i $\|E_{\lambda_i}\|$ is sufficiently small.
- The corresponding eigenblock in $\breve{H}_p^{2k,2k}$ has to be moved to the left upper corner of $\breve{H}_p^{2k,2k}$ using the T_p transformations.
- Accumulation of this transformations in $\breve{S}_p$ guarantees that the first columns of S_p are associated with λ_i.
- Shrink $\breve{H}_p$ such that the eigenblock for λ_i will not be involved in further computations.
- The J-reorthogonalisation has to be done against all columns of $\breve{S}_p$.

Locking

- For the **wanted** eigenvalue λ_i $\|E_{\lambda_i}\|$ is sufficiently small.
- The corresponding eigenblock in $\breve{H}_p^{2k,2k}$ has to be moved to the left upper corner of $\breve{H}_p^{2k,2k}$ using the T_p transformations.
- Accumulation of this transformations in $\breve{S}_p$ guarantees that the first columns of S_p are associated with λ_i.
- Shrink $\breve{H}_p$ such that the eigenblock for λ_i will not be involved in further computations.
- The J-reorthogonalisation has to be done against all columns of $\breve{S}_p$.

Locking

- For the **wanted** eigenvalue λ_i $\|E_{\lambda_i}\|$ is sufficiently small.
- The corresponding eigenblock in $\breve{H}_p^{2k,2k}$ has to be moved to the left upper corner of $\breve{H}_p^{2k,2k}$ using the T_p transformations.
- Accumulation of this transformations in $\breve{S}_p$ guarantees that the first columns of S_p are associated with λ_i.
- Shrink $\breve{H}_p$ such that the eigenblock for λ_i will not be involved in further computations.
- The J-reorthogonalisation has to be done against all columns of $\breve{S}_p$.

Purging

- For the **unwanted** eigenvalue λ_i $\|E_{\lambda_i}\|$ is sufficiently small.
- The corresponding eigenblock in $\breve{H}_p^{2k,2k}$ has to be moved to the left upper corner of $\breve{H}_p^{2k,2k}$ using the T_p transformations.
- Accumulation of this transformations in $\breve{S}_p$ guarantees that the first columns of S_p are associated with λ_i.
- Shrink $\breve{H}_p$ such that the eigenblock for λ_i will not be involved in further computations.
- The J-reorthogonalisation has to be done against all columns of $\breve{S}_p$.

Krylov-Schur like algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$.
2. Compute SR algorithm for $\tilde{H}_p$.
3. Transform Lanczos factorisation into Krylov-Schur like decomposition.
4. Compute backward error $\|E_{\lambda_i}\|$.
5. Move the eigenblocks to the proper places.
6. Create smaller Lanczos factorisation out of Krylov-Schur like decomposition.

Krylov-Schur like algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$.
2. Compute SR algorithm for $\tilde{H}_p$.
3. Transform Lanczos factorisation into Krylov-Schur like decomposition.
4. Compute backward error $\|E_{\lambda_i}\|$.
5. Move the eigenblocks to the proper places.
6. Create smaller Lanczos factorisation out of Krylov-Schur like decomposition.

Krylov-Schur like algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$.
2. Compute SR algorithm for $\tilde{H}_p$.
3. Transform Lanczos factorisation into Krylov-Schur like decomposition.
4. Compute backward error $\|E_{\lambda_i}\|$.
5. Move the eigenblocks to the proper places.
6. Create smaller Lanczos factorisation out of Krylov-Schur like decomposition.

Krylov-Schur like algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T.$
2. Compute SR algorithm for $\tilde{H}_p$.
3. Transform Lanczos factorisation into Krylov-Schur like decomposition.
4. Compute backward error $\|E_{\lambda_i}\|$.
5. Move the eigenblocks to the proper places.
6. Create smaller Lanczos factorisation out of Krylov-Schur like decomposition.

Krylov-Schur like algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$.
2. Compute SR algorithm for $\tilde{H}_p$.
3. Transform Lanczos factorisation into Krylov-Schur like decomposition.
4. Compute backward error $\|E_{\lambda_i}\|$.
5. Move the eigenblocks to the proper places.
6. Create smaller Lanczos factorisation out of Krylov-Schur like decomposition.

Krylov-Schur like algorithm

1. $H_p S_p^{2n,2m} = S_p^{2n,2m} \tilde{H}_p^{2m,2m} + \zeta_{m+1} v_{m+1} e_{2m}^T$.
2. Compute SR algorithm for $\tilde{H}_p$.
3. Transform Lanczos factorisation into Krylov-Schur like decomposition.
4. Compute backward error $\|E_{\lambda_i}\|$.
5. Move the eigenblocks to the proper places.
6. Create smaller Lanczos factorisation out of Krylov-Schur like decomposition.

Results of SR algorithm

Comparison of implemented SR algorithm and Matlab's eig routine for matrix H from CAREX (Example16) benchmark collection by [Benner/Laub/Mehrmann, 1995].

$$H = \left[\begin{array}{ccccc|ccccc}
-2 & 1 & & & 1 & 1 & & & & \\
1 & -2 & 1 & & & & 1 & & & \\
 & 1 & \ddots & \ddots & & & & \ddots & & \\
 & & \ddots & \ddots & 1 & & & & 1 & \\
1 & & & 1 & -2 & & & & & 1 \\
\hline
1 & & & & & -2 & 1 & & & 1 \\
 & 1 & & & & 1 & -2 & 1 & & \\
 & & \ddots & & & & 1 & \ddots & \ddots & \\
 & & & 1 & & & & \ddots & \ddots & 1 \\
 & & & & 1 & 1 & & & 1 & -2
\end{array}\right] \in \mathbb{R}^{2n,2n}$$

SR algorithm vs. Matlab's eig

eig	SR algorithm
-4.09007692547218	-4.09007692547218
-4.09007692547217	-4.09007692547217
-3.83317281079134	-3.83317281079134
-3.35764070604493	-3.35764070604493
-2.73657991827709	-2.73657991827709
-2.73657991827709	-2.73657991827702
-2.07265697232460	-2.07265697232467
-2.07265697232460	-2.07265697232460
-1.49292460661253	-1.49292460661264
-1.12803607263302	-1.12803607263302
-1.00907878741279	-1.00907878741279
-1.00000000000000	-1.00000000000000

- Example 16 of CAREX with size 24
- Maximal condition number for the SR algorithm is 51.75

Krylov-Schur for Quadratic Eigenvalue Problem

Consider the Quadratic Eigenvalue Problem (QEP)

$$(\lambda^2 M + \lambda G + K)u = 0$$

with matrices M, G and $K \in \mathbb{C}^{n,n}$. The special QEP where $M = M^T > 0$, $G = -G^T$ und $-K = -K^T > 0$ has Hamiltonian distribution of the eigenvalues and can be transformed in a Hamiltonian eigenvalue problem. See [Apel/Mehrmann/Watkins,2002] or [Mehrmann/Watkins,2001]. The result is the matrix H of the following form

$$H = \begin{bmatrix} I & -\frac{1}{2}G \\ 0 & I \end{bmatrix} \begin{bmatrix} 0 & -K \\ M^{-1} & 0 \end{bmatrix} \begin{bmatrix} I & -\frac{1}{2}G \\ 0 & I \end{bmatrix}.$$

Krylov-Schur like vs. SHIRA

Krylov-Schur like algorithm

eigenvalue	residual
-0.90592878885719	2.813e-16
-0.90634686034789	1.752e-16
-1.07560224930035	2.970e-16
-1.60332758476210	2.576e-15
-1.65786577098053	3.323e-15
-1.66121735256369	2.576e-15

SHIRA algorithm

eigenvalue	residual
-0.90592878886122	6.945e-17
-0.90634686034999	1.325e-16
-1.07560224930036	3.933e-17
-1.60332758477172	1.639e-15
-1.65786577098970	6.311e-16
-1.66121735256372	2.157e-16

- Both algorithms need 3 iterations to reach convergence bound of 10^{-8}
- SR algorithm has maximal condition number of 21178
- Size of matrices M, G and K is $n = 5139$

Krylov-Schur like vs. Matlab's eigs

Krylov-Schur like algorithm

Eigenwerte	Residuen
-0.90592878885661	3.244e-16
-0.90634686034754	2.242e-16
-1.07560224930008	1.591e-16
-1.60332758476411	7.370e-16
-1.65786577098821	2.488e-15
-1.66121735257659	1.169e-14
-1.75605775398702	2.057e-15

Matlab's eigs routine

Eigenwerte	Residuen
-0.90592878886324	1.864e-16
-0.90634686035071	1.648e-16
-1.07560224930012	1.823e-16
-1.60332758476362	5.733e-17
-1.65786577099183	5.241e-15
-1.66121735256949	3.922e-15
-1.75605775408379	9.367e-14

- Matlab's eigs needs 11 iteration steps and the Krylov-Schur like algorithm 8 (convergence bound of 10^{-10} for both)
- Maximal condition number during Krylov-Schur like is 2518
- Size of matrices M, G and K is again $n = 5139$

1. Fortran or C implementation of the SR algorithm and the Krylov-Schur Process
2. Time measurement when comparing these algorithms with SHIRA, eig and eigs
3. Integration of more computable examples such as Positive-Real-Balancing or special QEPs