A Fast QR Eigenvalue Method for a Class of Structured Matrices

D. A. Bini[1] P. Boito[2] Y. Eidelman[3] L. Gemignani[1]
I. Gohberg[3]

[1]University of Pisa (Italy)

[2]University of Toulouse 3 - Paul Sabatier (France)

[3]Tel-Aviv University (Israel)

Structured Linear Algebra Problems: Analysis, Algorithms and Applications, Cortona, 15-19 September 2008

Outline

Computing Eigenvalues

Problem

Given $A \in \mathbb{R}^{n \times n}$ or $\mathbb{C}^{n \times n}$, compute all its eigenvalues.

This task is usually accomplished by applying the QR method:

- Computational cost: $\mathcal{O}(n^3)$
- Storage: $\mathcal{O}(n^2)$.

The method is effective but not suitable for large matrices. But the structure of A can be exploited to achieve a computational cost of $\mathcal{O}(n^2)$ with linear memory space.

Computing Eigenvalues

Problem

Given $A \in \mathbb{R}^{n\times n}$ or $\mathbb{C}^{n\times n}$, compute all its eigenvalues.

This task is usually accomplished by applying the QR method:

- Computational cost: $\mathcal{O}(n^3)$
- Storage: $\mathcal{O}(n^2)$.

The method is effective but not suitable for large matrices.

But the structure of A can be exploited to achieve a computational cost of $\mathcal{O}(n^2)$ with linear memory space.

Computing Eigenvalues

Problem

Given $A \in \mathbb{R}^{n \times n}$ or $\mathbb{C}^{n \times n}$, compute all its eigenvalues.

This task is usually accomplished by applying the QR method:

- Computational cost: $\mathcal{O}(n^3)$
- Storage: $\mathcal{O}(n^2)$.

The method is effective but not suitable for large matrices. But the structure of A can be exploited to achieve a computational cost of $\mathcal{O}(n^2)$ with linear memory space.

The structure we are interested in...

We consider matrices $A \in \mathbb{C}^{n \times n}$ which are upper Hessenberg and have the form

$$A = U - pq^T$$

where

- $U \in \mathbb{C}^{n \times n}$ is unitary,
- $p, q \in \mathbb{C}^n$ (perturbation vectors).

Observe that U belongs to the class $\mathcal{U}_n$ of $n \times n$ unitary matrices which can be written as a rank one correction of an upper Hessenberg matrix.

The structure we are interested in...

We consider matrices $A \in \mathbb{C}^{n \times n}$ which are upper Hessenberg and have the form

$$A = U - pq^T$$

where

- $U \in \mathbb{C}^{n \times n}$ is unitary,
- $p, q \in \mathbb{C}^n$ (perturbation vectors).

Observe that U belongs to the class $\mathcal{U}_n$ of $n \times n$ unitary matrices which can be written as a rank one correction of an upper Hessenberg matrix.

Some History

- Restarted QR:
  - Calvetti, Kim, Reichel (2002)
- Explicit structured QR for companion, fellow, unitary-quasiseparable matrices:
  - Bini, Daddi, Gemignani (2004)
  - Bini, Eidelman, Gemignani, Gohberg (2007)
- Implicit structured QR for unitary (and more general) matrices: Gragg (1986), Leuven group.
- Implicit structured QR for companion / sequentially semiseparable matrices:
  - Chandrasekaran, Gu, Xia, Zhu (2007)

Some History

- Restarted QR:
  - Calvetti, Kim, Reichel (2002)
- Explicit structured QR for companion, fellow, unitary-quasiseparable matrices:
  - Bini, Daddi, Gemignani (2004)
  - Bini, Eidelman, Gemignani, Gohberg (2007)
- Implicit structured QR for unitary (and more general) matrices: Gragg (1986), Leuven group.
- Implicit structured QR for companion / sequentially semiseparable matrices:
  - Chandrasekaran, Gu, Xia, Zhu (2007)

Some History

- Restarted QR:
  - Calvetti, Kim, Reichel (2002)
- Explicit structured QR for companion, fellow, unitary-quasiseparable matrices:
  - Bini, Daddi, Gemignani (2004)
  - Bini, Eidelman, Gemignani, Gohberg (2007)
- Implicit structured QR for unitary (and more general) matrices: Gragg (1986), Leuven group.
- Implicit structured QR for companion / sequentially semiseparable matrices:
  - Chandrasekaran, Gu, Xia, Zhu (2007)

Some History

- Restarted QR:
  - Calvetti, Kim, Reichel (2002)
- Explicit structured QR for companion, fellow, unitary-quasiseparable matrices:
  - Bini, Daddi, Gemignani (2004)
  - Bini, Eidelman, Gemignani, Gohberg (2007)
- Implicit structured QR for unitary (and more general) matrices: Gragg (1986), Leuven group.
- Implicit structured QR for companion / sequentially semiseparable matrices:
  - Chandrasekaran, Gu, Xia, Zhu (2007)

Our Result

For theoretical background, see also the 2006 SIAM Annual Meeting (Boston), session on Structured Matrices and Fast Algorithms.

We present here a new algorithm which

- is based on the implicit QR method,
- computes the eigenvalues of Hessenberg matrices which are unitary + rank 1,
- achieves quadratic computational cost and linear memory,
- shows good stability properties.

Our Result

For theoretical background, see also the 2006 SIAM Annual Meeting (Boston), session on Structured Matrices and Fast Algorithms.
We present here a new algorithm which

- is based on the implicit QR method,
- computes the eigenvalues of Hessenberg matrices which are unitary + rank 1,
- achieves quadratic computational cost and linear memory,
- shows good stability properties.

Representation of U

$$U = A + pq^T$$

with U unitary, A upper Hessenberg. We will use two different representations for U:

- as product of "small" unitary matrices,
- as a quasiseparable matrix.

About product structure: compare Schur parametrization (Gragg) and generalized Givens representation (Leuven group).

Representation of U

$$U = A + pq^T$$

with U unitary, A upper Hessenberg. We will use two different representations for U:

- as product of "small" unitary matrices,
- as a quasiseparable matrix.

About product structure: compare Schur parametrization (Gragg) and generalized Givens representation (Leuven group).

Representation of U

$$U = A + pq^T$$

with U unitary, A upper Hessenberg. We will use two different representations for U:

- as product of "small" unitary matrices,
- as a quasiseparable matrix.

About product structure: compare Schur parametrization (Gragg) and generalized Givens representation (Leuven group).

Product Structure

Choose 2×2 unitary matrices $\{V_i\}_{i=2,\dots,n-1}$ such that

$$V_i^* \cdot \begin{bmatrix} p_i \\ \beta_{i+1} \end{bmatrix} = \begin{bmatrix} \beta_i \\ 0 \end{bmatrix}, \quad i = n-1, \dots, 2$$

for some complex numbers $\{\beta_i\}_{i=2,\dots,n}$, with $\beta_n = p(n)$. For each i set

$$\tilde{V}_i = \begin{bmatrix} I_{i-1} & & \\ & V_i & \\ & & I_{n-i-1} \end{bmatrix}$$

and define $V = \tilde{V}_{n-1} \cdot \tilde{V}_{n-2} \cdot \ldots \cdot \tilde{V}_2$.

Product Structure

$$U =$$

$$\begin{bmatrix} \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ p(3)q(1) & \times & \times & \times & \times & \times \\ p(4)q(1) & p(4)q(2) & \times & \times & \times & \times \\ p(5)q(1) & p(5)q(2) & p(5)q(3) & \times & \times & \times \\ p(6)q(1) & p(6)q(2) & p(6)q(3) & p(6)q(4) & \times & \times \end{bmatrix}$$

Product Structure

$$\tilde{V}_5^* \cdot U =$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ p(3)q(1) & \times & \times & \times & \times & \times \\ p(4)q(1) & p(4)q(2) & \times & \times & \times & \times \\ \beta(5)q(1) & \beta(5)q(2) & \beta(5)q(3) & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \end{bmatrix}$$

Product Structure

$$\tilde{V}_4^* \cdot \tilde{V}_5^* \cdot U =$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ p(3)q(1) & \times & \times & \times & \times & \times \\ \beta(4)q(1) & \beta(4)q(2) & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \end{bmatrix}$$

Product Structure

$$\tilde{V}_3^* \cdot \tilde{V}_4^* \cdot \tilde{V}_5^* \cdot U =$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \beta(3)q(1) & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \end{bmatrix}$$

Product Structure

Now choose 3×3 unitary matrices $\{F_i\}_{i=1,\dots,n-2}$ and for each i set

$$\tilde{F}_i = \begin{bmatrix} I_{i-1} & 0 & 0 \\ 0 & F_i & 0 \\ 0 & 0 & I_{n-i-2} \end{bmatrix}.$$

Define $F = \tilde{F}_1 \cdot \tilde{F}_2 \cdot \dots \cdot \tilde{F}_{n-2}$, so that we have...

Product Structure

$$V^* \cdot U =$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \end{bmatrix}$$

Product Structure

$$\tilde{F}_1^* \cdot V^* \cdot U =$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \end{bmatrix}$$

Product Structure

$$\tilde{F}_2^* \cdot \tilde{F}_1^* \cdot V^* \cdot U =$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \end{bmatrix}$$

Product Structure

$$\tilde{F}_3^* \cdot \tilde{F}_2^* \cdot \tilde{F}_1^* \cdot V^* \cdot U =$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \end{bmatrix}$$

Product Structure

$$\tilde{F}_4^* \cdot \tilde{F}_3^* \cdot \tilde{F}_2^* \cdot \tilde{F}_1^* \cdot V^* \cdot U =$$

$$\begin{bmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{bmatrix}$$

Product Structure

...and the upper triangular factor must be the identity. We obtain a factorization

$$U = V \cdot F$$

where the unitary matrices V and F have the form

$$V = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & \times & \times & 0 & 0 & 0 \\ 0 & \times & \times & \times & 0 & 0 \\ 0 & \times & \times & \times & \times & 0 \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \end{bmatrix}, \quad F = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \end{bmatrix}$$

(V is lower Hessenberg and $\texttt{tril}(F, -3)$=0).

Quasiseparable Structure

First introduced in [Eidelman, Gohberg (1999)].

Definition

$M \in \mathbb{C}^{n \times n}$ *is* *(n_L, n_U)-quasiseparable* *if*

- $n_L = \max_{1 \le k \le n-1} \mathit{rank}\, M(k+1:n, 1:k)$,
- $n_U = \max_{1 \le k \le n-1} \mathit{rank}\, M(1:k, k+1:n)$.

Quasiseparable matrices can be represented using $\mathcal{O}((n_L + n_U)n)$ parameters.

Quasiseparable Representation of U

Theorem

The upper triangular part of U has the structure

$$U(i,j) = g_i \cdot b_i \cdot b_{i+1} \cdot \cdots \cdot b_{j-1} \cdot h_j, \quad i \leq j$$

where

- $g_1, \ldots, g_n$ *are row vectors of length* 2,
- $h_1, \ldots, h_n$ *are column vectors of length* 2,
- $b_1, \ldots, b_{n-1}$ *are* 2×2 *matrices.*

Recall that the `tril`$(U, -2)$ is defined by

$$U(i,j) = p(i) \cdot q(j), \quad i > j+1$$

Quasiseparable Representation of U

Theorem

The upper triangular part of U has the structure

$$U(i,j) = g_i \cdot b_i \cdot b_{i+1} \cdot \dots \cdot b_{j-1} \cdot h_j, \quad i \le j$$

where

- $g_1, \dots, g_n$ *are row vectors of length* 2,
- $h_1, \dots, h_n$ *are column vectors of length* 2,
- $b_1, \dots, b_{n-1}$ *are* 2×2 *matrices.*

Recall that the $\texttt{tril}(U, -2)$ is defined by

$$U(i,j) = p(i) \cdot q(j), \quad i > j+1$$

Quasiseparable Representation of U

Therefore U can be represented as

$$U = \begin{bmatrix} g_1 \cdot h_1 & g_1 \cdot b_1 \cdot h_2 & g_1 \cdot b_1 \cdot b_2 \cdot h_3 & \dots \\ \sigma_1 & g_2 \cdot h_2 & g_2 \cdot b_2 \cdot h_3 & \dots \\ p(3)q(1) & \sigma_2 & g_3 \cdot h_3 & \dots \\ p(4)q(1) & p(4)q(2) & \sigma_3 & \\ \vdots & & & \end{bmatrix}$$

and it is completely determined by the sets of generators $\{g_i\}$, $\{h_i\}$, $\{b_i\}$, $\{\sigma_i\}$ and the perturbation vectors p and q. to algorithm

Recovery of Quasiseparable Structure

The quasiseparable representation can easily be recovered from the product representation:

- $h_k = F_k(1:2,1)$ for $k = 1, \dots, n-2$
- $h_{n-1} = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$, $h_n = \begin{pmatrix} 0 \\ 1 \end{pmatrix}$
- $b_k = F_k(1:2, 2:3)$ for $k = 1, \dots, n-2, \qquad b_{n-1} = I_2$
- $\gamma_1 = \begin{pmatrix} 0 & 1 \end{pmatrix}, \quad g_1 = \begin{pmatrix} 1 & 0 \end{pmatrix}$
- $\begin{pmatrix} \sigma_k & g_{k+1} \\ q(k) & \gamma_{k+1} \end{pmatrix} = V_{k+1} \begin{pmatrix} \gamma_k & 0 \\ 0 & 1 \end{pmatrix} F_k, \quad k = 1, \dots, n-2$
- $\sigma_{n-1} = \gamma_{n-1} h_{n-1}, \quad g_n = \gamma_{n-1} b_{n-1}$

An Interesting Example: Companion Matrices

Let $P(z) = z^n + \sum_{j=2}^{n+1} c_j z^j$ a real or complex monic polynomial. Then the associated companion matrix

$$A_P = \begin{bmatrix} -c_2 & -c_3 & \dots & -c_n & -c_{n+1} \\ 1 & 0 & \dots & 0 & 0 \\ 0 & 1 & \dots & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \dots & 1 & 0 \end{bmatrix}$$

displays an upper Hessenberg and (unitary + rank one) structure.

Classic Explicit QR

Let $A \in \mathbb{R}^{n \times n}$ or $\mathbb{C}^{n \times n}$. Then the iteration defined by

$$\begin{aligned} A^{(0)} &= A \\ A^{(i)} &= Q^{(i)} \cdot R^{(i)} \\ A^{(i+1)} &:= R^{(i)} \cdot Q^{(i)}, \qquad i = 0, 1, 2, \ldots \end{aligned}$$

converges to an upper triangular matrix B which has the same eigenvalues as A.

Classic Implicit QR

Let $A \in \mathbb{R}^{n\times n}$ or $\mathbb{C}^{n\times n}$. Then the iteration defined by

$$\begin{aligned} A^{(0)} &= A \\ A^{(i+1)} &:= (Q^{(i)})^* \cdot A^{(i)} \cdot Q^{(i)}, \end{aligned}$$

where $A^{(i)} = Q^{(i)} \cdot R^{(i)}$ for $i = 0, 1, 2, \ldots$
converges to an upper triangular matrix B which has the same eigenvalues as A.

Shift Strategies

To accelerate convergence, QR iterations are generally applied to $\mathcal{P}(A^{(i)})$, where $\mathcal{P}(X)$ is a carefully chosen polynomial, rather than to $A^{(i)}$ itself. Usual choices for $\mathcal{P}(X)$ include:

- (single shift) $\mathcal{P}(X) = X - \alpha I$, with $\alpha = X(n,n)$
- (double shift) $\mathcal{P}(X) = (X - \alpha_1 I)(X - \alpha_2 I)$, with α_1 and α_2 eigenvalues of

$$\begin{pmatrix} X(n-1,n-1) & X(n-1,n) \\ X(n,n-1) & X(n,n) \end{pmatrix}$$

Shift Strategies

To accelerate convergence, QR iterations are generally applied to $\mathcal{P}(A^{(i)})$, where $\mathcal{P}(X)$ is a carefully chosen polynomial, rather than to $A^{(i)}$ itself. Usual choices for $\mathcal{P}(X)$ include:

- (single shift) $\mathcal{P}(X) = X - \alpha I$, with $\alpha = X(n, n)$
- (double shift) $\mathcal{P}(X) = (X - \alpha_1 I)(X - \alpha_2 I)$, with α_1 and α_2 eigenvalues of

$$\begin{pmatrix} X(n-1, n-1) & X(n-1, n) \\ X(n, n-1) & X(n, n) \end{pmatrix}$$

Shift Strategies

To accelerate convergence, QR iterations are generally applied to $\mathcal{P}(A^{(i)})$, where $\mathcal{P}(X)$ is a carefully chosen polynomial, rather than to $A^{(i)}$ itself. Usual choices for $\mathcal{P}(X)$ include:

- (single shift) $\mathcal{P}(X) = X - \alpha I$, with $\alpha = X(n, n)$
- (double shift) $\mathcal{P}(X) = (X - \alpha_1 I)(X - \alpha_2 I)$, with α_1 and α_2 eigenvalues of

$$\begin{pmatrix} X(n-1, n-1) & X(n-1, n) \\ X(n, n-1) & X(n, n) \end{pmatrix}$$

Bulge Chasing

Practical implementation of implicit QR with single shift on an upper Hessenberg matrix A:

- choose Q_1 unitary 2×2 such that $Q_1^* \cdot \begin{bmatrix} A(1,1) - \alpha \\ A(2,1) \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}$,
- perform bulge chasing:

$$A = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$

Bulge Chasing

Practical implementation of implicit QR with single shift on an upper Hessenberg matrix A:

- choose Q_1 unitary 2×2 such that
  $Q_1^* \cdot \begin{bmatrix} A(1,1) - \alpha \\ A(2,1) \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}$,
- perform bulge chasing:

$$\tilde{Q}_1^* \cdot A = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$

Bulge Chasing

Practical implementation of implicit QR with single shift on an upper Hessenberg matrix A:

- choose Q_1 unitary 2×2 such that $Q_1^* \cdot \begin{bmatrix} A(1,1)-\alpha \\ A(2,1) \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}$,
- perform bulge chasing:

$$\tilde{Q}_1^* \cdot A \cdot \tilde{Q}_1 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$

Bulge Chasing

Practical implementation of implicit QR with single shift on an upper Hessenberg matrix A:

- choose Q_1 unitary 2×2 such that $Q_1^* \cdot \begin{bmatrix} A(1,1) - \alpha \\ A(2,1) \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}$,
- perform bulge chasing:

$$\tilde{Q}_2^* \cdot \tilde{Q}_1^* \cdot A \cdot \tilde{Q}_1 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$

Bulge Chasing

Practical implementation of implicit QR with single shift on an upper Hessenberg matrix A:

- choose Q_1 unitary 2×2 such that $Q_1^* \cdot \begin{bmatrix} A(1,1) - \alpha \\ A(2,1) \end{bmatrix} = \begin{bmatrix} \times \\ 0 \end{bmatrix}$,
- perform bulge chasing:

$$\tilde{Q}_2^* \cdot \tilde{Q}_1^* \cdot A \cdot \tilde{Q}_1 \cdot \tilde{Q}_2 = \begin{bmatrix} \times & \times & \times & \times & \times & \times \\ \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \end{bmatrix}$$

Application to the Quasiseparable Structure

What happens in the structured case?

- The QR iteration preserves the upper Hessenberg and unitary + rank one structure.
- We can carry out each QR iteration working only on the quasiseparable and product representations.

We apply the shifted implicit QR strategy to $A = U - pq^T$. At each iteration:

- the bulge chasing process is applied to A in order to compute the Q_i's (using the quasiseparable structure), q.s.rep.
- the product structure of the next iterate $A^{(1)} = Q^* \cdot A \cdot Q$ is computed,
- the quasiseparable structure of $A^{(1)}$ is recovered.

Application to the Quasiseparable Structure

What happens in the structured case?

- The *QR* iteration preserves the upper Hessenberg and unitary + rank one structure.
- We can carry out each QR iteration working only on the quasiseparable and product representations.

We apply the shifted implicit QR strategy to $A = U - pq^T$. At each iteration:

- the bulge chasing process is applied to A in order to compute the Q_i's (using the quasiseparable structure), q.s.rep.
- the product structure of the next iterate $A^{(1)} = Q^* \cdot A \cdot Q$ is computed,
- the quasiseparable structure of $A^{(1)}$ is recovered.

Application to the Quasiseparable Structure

What happens in the structured case?

- The QR iteration preserves the upper Hessenberg and unitary + rank one structure.
- We can carry out each QR iteration working only on the quasiseparable and product representations.

We apply the shifted implicit QR strategy to $A = U - pq^T$. At each iteration:

- the bulge chasing process is applied to A in order to compute the Q_i's (using the quasiseparable structure), q.s.rep.
- the product structure of the next iterate $A^{(1)} = Q^* \cdot A \cdot Q$ is computed,
- the quasiseparable structure of $A^{(1)}$ is recovered.

Application to the Quasiseparable Structure

What happens in the structured case?

- The QR iteration preserves the upper Hessenberg and unitary + rank one structure.
- We can carry out each QR iteration working only on the quasiseparable and product representations.

We apply the shifted implicit QR strategy to $A = U - pq^T$. At each iteration:

- the bulge chasing process is applied to A in order to compute the Q_i's (using the quasiseparable structure), q.s.rep.
- the product structure of the next iterate $A^{(1)} = Q^* \cdot A \cdot Q$ is computed,
- the quasiseparable structure of $A^{(1)}$ is recovered.

Application to the Quasiseparable Structure

What happens in the structured case?

- The QR iteration preserves the upper Hessenberg and unitary + rank one structure.
- We can carry out each QR iteration working only on the quasiseparable and product representations.

We apply the shifted implicit QR strategy to $A = U - pq^T$. At each iteration:

- the bulge chasing process is applied to A in order to compute the Q_i's (using the quasiseparable structure), q.s.rep.
- the product structure of the next iterate $A^{(1)} = Q^* \cdot A \cdot Q$ is computed,
- the quasiseparable structure of $A^{(1)}$ is recovered.

Application to the Quasiseparable Structure

What happens in the structured case?

- The QR iteration preserves the upper Hessenberg and unitary + rank one structure.
- We can carry out each QR iteration working only on the quasiseparable and product representations.

We apply the shifted implicit QR strategy to $A = U - pq^T$. At each iteration:

- the bulge chasing process is applied to A in order to compute the Q_i's (using the quasiseparable structure), q.s.rep.
- the product structure of the next iterate $A^{(1)} = Q^* \cdot A \cdot Q$ is computed,
- the quasiseparable structure of $A^{(1)}$ is recovered.

Application to the Quasiseparable Structure

What happens in the structured case?

- The QR iteration preserves the upper Hessenberg and unitary + rank one structure.
- We can carry out each QR iteration working only on the quasiseparable and product representations.

We apply the shifted implicit QR strategy to $A = U - pq^T$. At each iteration:

- the bulge chasing process is applied to A in order to compute the Q_i's (using the quasiseparable structure), q.s.rep.
- the product structure of the next iterate $A^{(1)} = Q^* \cdot A \cdot Q$ is computed,
- the quasiseparable structure of $A^{(1)}$ is recovered.

Numerical Experiments

We implemented this algorithm (fastQR) in Fortran 95 for the case of companion matrices. Numerical experiments have been carried out on several families of test polynomials, in order to check:

- growth of running time,
- accuracy of output (comparison with output given by LAPACK),
- (backward) stability,
- comparison of performance with other fast eigenvalue solvers.

Errors

- Absolute forward error: distance between the eigenvalues found by the structured methods and the eigenvalues computed by LAPACK (sort vectors and compute max distance);
- Relative backward error:

$$\text{b.e.} = \frac{\|Q_a^* A_0 Q_a - (V_f F_f - p_f q_f^T)\|_\infty}{\|A_0\|_\infty}$$

where Q_a is the accumulated unitary similarity transformation and V_f, F_f, p_f, q_f are generators for the final iterate A_f.

Estimate for the forward error:

$$K \cdot \epsilon \cdot \|A\|_2 \cdot \texttt{max(condeig(A))}.$$

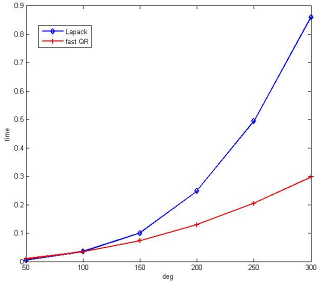
Single Shift, Random Coefficients

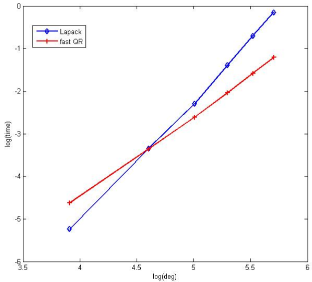
Example 1: Monic polynomials with complex pseudorandom coefficients; both the real and imaginary part belong to $[-1, 1]$:

- compare running time to LAPACK (routine ZGEEV),
- check growth of running time,
- check forward and backward errors.

Single Shift, Random Coefficients

Comparison with LAPACK

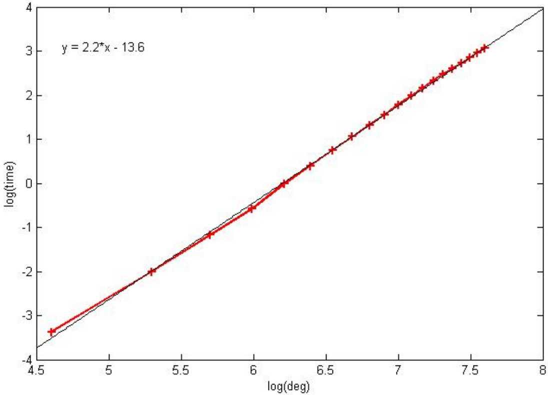




Single Shift, Random Coefficients

Time growth, log-log plot

Polynomials of degrees from 50 to 2000



Single Shift, Random Coefficients II

Example 2: $P(z) = z^n + \sum_{j=0}^{n-1} a_j z^j$, with $a_j = u_j \cdot 10^{v_j}$, where $|u_j| \in [-1, 1]$ and $v_j \in [-5, 5]$.

deg	b.e.	f.e.	δ
50	4.91×10^{-15}	6.24×10^{-10}	2.21×10^{-8}
100	6.63×10^{-15}	6.95×10^{-10}	1.61×10^{-8}
150	7.02×10^{-15}	3.19×10^{-10}	1.12×10^{-7}
500	7.43×10^{-15}	8.30×10^{-10}	8.28×10^{-7}
1000	1.44×10^{-14}	1.47×10^{-9}	1.59×10^{-6}

$\delta = \epsilon \cdot \|A\|_2 \cdot \texttt{max(condeig(A))}$

Single Shift, Random Coefficients II

Example 2: $P(z) = z^n + \sum_{j=0}^{n-1} a_j z^j$, with $a_j = u_j \cdot 10^{v_j}$, where $|u_j| \in [-1, 1]$ and $v_j \in [-5, 5]$.

deg	b.e.	f.e.	δ
50	4.91×10^{-15}	6.24×10^{-10}	2.21×10^{-8}
100	6.63×10^{-15}	6.95×10^{-10}	1.61×10^{-8}
150	7.02×10^{-15}	3.19×10^{-10}	1.12×10^{-7}
500	7.43×10^{-15}	8.30×10^{-10}	8.28×10^{-7}
1000	1.44×10^{-14}	1.47×10^{-9}	1.59×10^{-6}

$\delta = \epsilon \cdot \|A\|_2 \cdot \texttt{max(condeig(A))}$

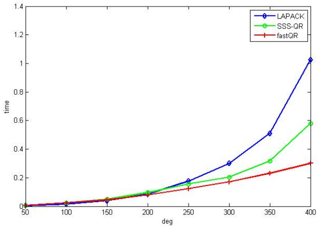
Double Shift, Random Coefficients

Example 3: Monic polynomials with real pseudorandom coefficients in $[-1, 1]$:

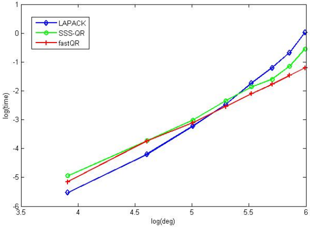
- compare running time to LAPACK (routine DGEEV) and SSS-QR (Chandrasekaran et al.),
- check growth of running time,
- check forward errors.

Double Shift, Random Coefficients

Comparison with LAPACK and SSS-QR



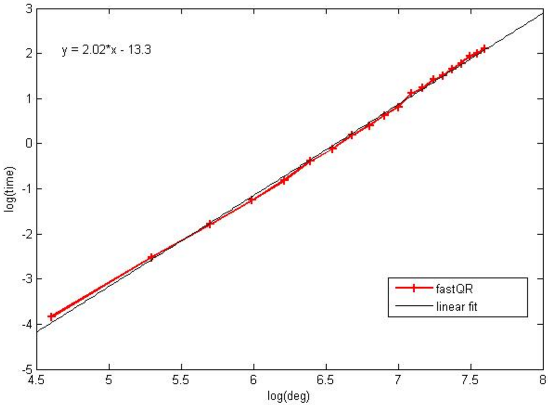
absolute forward errors $\sim 10^{-14}$



Double Shift, Random Coefficients

Time growth, log-log plot

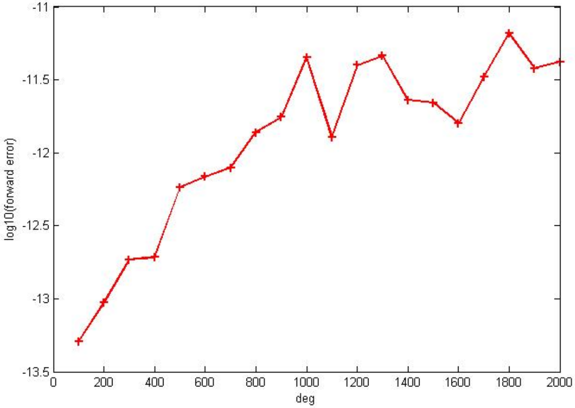
Polynomials of degrees from 50 to 2000



Double Shift, Random Coefficients

Forward errors, log plot

Polynomials of degrees from 50 to 2000



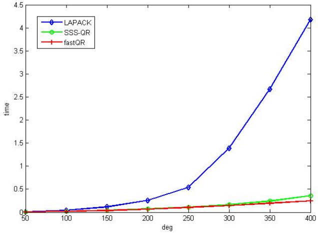
Double Shift, Roots of 1

Example 4: $P(x) = x^n - 1$

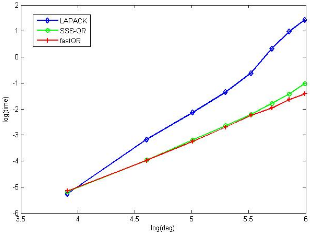
- compare running time to LAPACK (routine DGEEV) and SSS-QR (Chandrasekaran et al.),
- check forward errors.

Double Shift, Roots of 1

Comparison with LAPACK and SSS-QR



absolute forward errors $\sim 10^{-15}$



Summary and Future Work

- We have developed and implemented a fast version of the implicit QR method for Hessenberg matrices which are rank-one perturbations of unitary matrices.
- Numerical tests show that the algorithm has good stability properties and confirm theoretical estimates on computational cost ($\mathcal{O}(n^2)$) and required memory space ($\mathcal{O}(n)$).
- Open issue: give a theoretical proof of stability (the use of a representation via unitary matrices may prove helpful).

Summary and Future Work

- We have developed and implemented a fast version of the implicit QR method for Hessenberg matrices which are rank-one perturbations of unitary matrices.
- Numerical tests show that the algorithm has good stability properties and confirm theoretical estimates on computational cost ($\mathcal{O}(n^2)$) and required memory space ($\mathcal{O}(n)$).
- Open issue: give a theoretical proof of stability (the use of a representation via unitary matrices may prove helpful).

Summary and Future Work

- We have developed and implemented a fast version of the implicit QR method for Hessenberg matrices which are rank-one perturbations of unitary matrices.
- Numerical tests show that the algorithm has good stability properties and confirm theoretical estimates on computational cost ($\mathcal{O}(n^2)$) and required memory space ($\mathcal{O}(n)$).
- Open issue: give a theoretical proof of stability (the use of a representation via unitary matrices may prove helpful).

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^* \cdot \cdots \cdot \tilde{Q}_1^* \cdot \tilde{V}_{n-1} \cdot \cdots \cdot \tilde{V}_2 \cdot \tilde{F}_1 \cdot \ldots \tilde{F}_{n-2} \cdot \tilde{Q}_1 \cdot \cdots \cdot \tilde{Q}_{n-1}$$

Observe that

- $\tilde{Q}_k^*$ commutes with $\tilde{V}_{n-1}, \ldots, \tilde{V}_{k+2}$ for $1 \leq k \leq n-3$
- $\tilde{Q}_k$ commutes with $\tilde{F}_{n-2}, \ldots, \tilde{F}_{k+3}$ for $1 \leq k \leq n-4$

so we have

$$A^{(1)} = \tilde{Q}_{n-1}^* \cdot \cdots \cdot \tilde{V}_4 \cdot \tilde{Q}_2^* \cdot \tilde{V}_3 \cdot \tilde{Q}_1^* \cdot \tilde{V}_2 \cdot \tilde{F}_1 \cdot \tilde{F}_2 \cdot \tilde{Q}_1 \cdot \tilde{F}_3 \cdot \tilde{Q}_2 \cdot \tilde{F}_4 \cdot \cdots \cdot \tilde{Q}_{n-1}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^* \cdot \cdots \cdot \tilde{Q}_1^* \cdot \tilde{V}_{n-1} \cdot \cdots \cdot \tilde{V}_2 \cdot \tilde{F}_1 \cdot \ldots \tilde{F}_{n-2} \cdot \tilde{Q}_1 \cdot \cdots \cdot \tilde{Q}_{n-1}$$

Observe that

- $\tilde{Q}_k^*$ commutes with $\tilde{V}_{n-1}, \ldots, \tilde{V}_{k+2}$ for $1 \leq k \leq n-3$
- $\tilde{Q}_k$ commutes with $\tilde{F}_{n-2}, \ldots, \tilde{F}_{k+3}$ for $1 \leq k \leq n-4$

so we have

$$A^{(1)} = \tilde{Q}_{n-1}^* \cdot \cdots \cdot \tilde{V}_4 \cdot \tilde{Q}_2^* \cdot \tilde{V}_3 \cdot \tilde{Q}_1^* \cdot \tilde{V}_2 \cdot \tilde{F}_1 \cdot \tilde{F}_2 \cdot \tilde{Q}_1 \cdot \tilde{F}_3 \cdot \tilde{Q}_2 \cdot \tilde{F}_4 \cdot \cdots \cdot \tilde{Q}_{n-1}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^* \cdot \dots \cdot \tilde{Q}_1^* \cdot \tilde{V}_{n-1} \cdot \dots \cdot \tilde{V}_2 \cdot \tilde{F}_1 \cdot \ldots \tilde{F}_{n-2} \cdot \tilde{Q}_1 \cdot \dots \cdot \tilde{Q}_{n-1}$$

Observe that

- $\tilde{Q}_k^*$ commutes with $\tilde{V}_{n-1}, \ldots, \tilde{V}_{k+2}$ for $1 \le k \le n-3$
- $\tilde{Q}_k$ commutes with $\tilde{F}_{n-2}, \ldots, \tilde{F}_{k+3}$ for $1 \le k \le n-4$

so we have

$$A^{(1)} = \tilde{Q}_{n-1}^* \cdots \cdot \tilde{V}_4 \cdot \tilde{Q}_2^* \cdot \tilde{V}_3 \cdot \tilde{Q}_1^* \cdot \tilde{V}_2 \cdot \tilde{F}_1 \cdot \tilde{F}_2 \cdot \tilde{Q}_1 \cdot \tilde{F}_3 \cdot \tilde{Q}_2 \cdot \tilde{F}_4 \cdots \cdot \tilde{Q}_{n-1}$$

Computation of New Product Structure (Single shift)

$$A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{Q}^*_{k+1} \cdot \tilde{V}_{k+2} \cdot \tilde{V}^{(\tau)}_{k+1} \cdot \tilde{V}^{(1)}_k \cdot \ldots \cdot \tilde{V}^{(1)}_2 \cdot$$
$$\cdot \tilde{F}^{(1)}_1 \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{F}^{(\tau)}_k \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_k \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{Q}^*_{k+1} \cdot \tilde{V}_{k+2} \cdot \tilde{V}^{(\tau)}_{k+1} \cdot \tilde{H}^* \cdot \tilde{V}^{(1)}_k \cdot \ldots \cdot \tilde{V}^{(1)}_2 \cdot$$
$$\cdot \tilde{F}^{(1)}_1 \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{H} \cdot \tilde{F}^{(\tau)}_k \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_k \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{V}^{(\tau)}_{k+2} \cdot \tilde{V}^{(1)}_{k+1} \cdot \tilde{V}^{(1)}_k \cdot \ldots \cdot \tilde{V}^{(1)}_2 \cdot$$
$$\cdot \tilde{F}^{(1)}_1 \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{F}^{(1)}_k \cdot \tilde{F}^{(\tau)}_{k+1} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

Computation of New Product Structure (Single shift)

$$A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{Q}^*_{k+1} \cdot \tilde{V}_{k+2} \cdot \tilde{V}^{(\tau)}_{k+1} \cdot \tilde{V}^{(1)}_k \cdot \ldots \cdot \tilde{V}^{(1)}_2 \cdot$$
$$\cdot \tilde{F}^{(1)}_1 \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{F}^{(\tau)}_k \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_k \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{Q}^*_{k+1} \cdot \tilde{V}_{k+2} \cdot \tilde{V}^{(\tau)}_{k+1} \cdot \tilde{H}^* \cdot \tilde{V}^{(1)}_k \cdot \ldots \cdot \tilde{V}^{(1)}_2 \cdot$$
$$\cdot \tilde{F}^{(1)}_1 \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{H} \cdot \tilde{F}^{(\tau)}_k \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_k \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{V}^{(\tau)}_{k+2} \cdot \tilde{V}^{(1)}_{k+1} \cdot \tilde{V}^{(1)}_k \cdot \ldots \cdot \tilde{V}^{(1)}_2 \cdot$$
$$\cdot \tilde{F}^{(1)}_1 \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{F}^{(1)}_k \cdot \tilde{F}^{(\tau)}_{k+1} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

Computation of New Product Structure (Single shift)

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{Q}_{k+1}^{*} \cdot \tilde{V}_{k+2} \cdot \tilde{V}_{k+1}^{(\tau)} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot \\ \cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{Q}_{k+1}^{*} \cdot \tilde{V}_{k+2} \cdot \tilde{V}_{k+1}^{(\tau)} \cdot \tilde{H}^{*} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot \\ \cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{H} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{V}_{k+2}^{(\tau)} \cdot \tilde{V}_{k+1}^{(1)} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot \\ \cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{F}_{k}^{(1)} \cdot \tilde{F}_{k+1}^{(\tau)} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

Computation of New Product Structure (Single Shift)

$$\begin{aligned}A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{Q}^*_{k+1} \cdot \tilde{V}_{k+2} \cdot \tilde{V}^{(\tau)}_{k+1} \cdot \tilde{V}^{(1)}_k \cdot \ldots \cdot \tilde{V}^{(1)}_2 \cdot \\ \cdot \tilde{F}^{(1)}_1 \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{F}^{(\tau)}_k \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_k \cdot \ldots \cdot \tilde{Q}_{n-1}\end{aligned}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{Q}^*_{k+1} \cdot \tilde{V}_{k+2} \cdot \tilde{V}^{(\tau)}_{k+1} \cdot \tilde{V}^{(1)}_{k} \cdot \ldots \cdot \tilde{V}^{(1)}_{2} \cdot$$
$$\cdot \tilde{F}^{(1)}_{1} \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{F}^{(\tau)}_{k} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\left[\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & & \\ 0 & & V_{k+2} \end{array}\right] \cdot \left[\begin{array}{cc|c} & V^{(\tau)}_{k+1} & 0 \\ & & 0 \\ \hline 0 & 0 & 1 \end{array}\right]$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{Q}^*_{k+1} \cdot \tilde{V}_{k+2} \cdot \tilde{V}^{(\tau)}_{k+1} \cdot \tilde{V}^{(1)}_k \cdot \ldots \cdot \tilde{V}^{(1)}_2 \cdot$$
$$\cdot \tilde{F}^{(1)}_1 \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{F}^{(\tau)}_k \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_k \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\begin{bmatrix} \times & \times & 0 \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{Q}_{k+1}^{*} \cdot \tilde{V}_{k+2} \cdot \tilde{V}_{k+1}^{(\tau)} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot$$
$$\cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\left[\begin{array}{cc|c} \multicolumn{2}{c|}{\multirow{2}{*}{Q_{k+1}^{*}}} & 0 \\ & & 0 \\ \hline 0 & 0 & 1 \end{array}\right] \cdot \qquad \begin{bmatrix} \times & \times & 0 \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{Q}_{k+1}^{*} \cdot \tilde{V}_{k+2} \cdot \tilde{V}_{k+1}^{(\tau)} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot$$
$$\cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{Q}^*_{k+1} \cdot \tilde{V}_{k+2} \cdot \tilde{V}^{(\tau)}_{k+1} \cdot \tilde{V}^{(1)}_{k} \cdot \ldots \cdot \tilde{V}^{(1)}_{2} \cdot$$
$$\cdot \tilde{F}^{(1)}_{1} \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{F}^{(\tau)}_{k} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\begin{bmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix} \qquad \cdot \left[\begin{array}{cc|c} H^* & & 0 \\ & & 0 \\ \hline 0 & 0 & 1 \end{array}\right]$$

Computation of New Product Structure (Single Shift)

$$\begin{aligned} A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{Q}^*_{k+1} \cdot \tilde{V}_{k+2} \cdot \tilde{V}^{(\tau)}_{k+1} \cdot \tilde{V}^{(1)}_{k} \cdot \ldots \cdot \tilde{V}^{(1)}_{2} \cdot \\ \cdot \tilde{F}^{(1)}_{1} \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{F}^{(\tau)}_{k} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1} \end{aligned}$$

$$\begin{bmatrix} \times & \times & 0 \\ \times & \times & \times \\ \times & \times & \times \end{bmatrix}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{V}_{k+2}^{(\tau)} \cdot \tilde{V}_{k+1}^{(1)} \cdot \tilde{H}^{*} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot$$
$$\cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{H} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\left[\begin{array}{c|cc} 1 & 0 & 0 \\ \hline 0 & & \\ 0 & \multicolumn{2}{c}{V_{k+2}^{(\tau)}} \end{array}\right] \cdot \left[\begin{array}{cc|c} \multicolumn{2}{c|}{V_{k+1}^{(1)}} & 0 \\ & & 0 \\ \hline 0 & 0 & 1 \end{array}\right]$$

Computation of New Product Structure (Single Shift)

$$
\begin{aligned}
A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{V}_{k+2}^{(\tau)} \cdot \tilde{V}_{k+1}^{(1)} \cdot \tilde{H}^{*} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot \\
\cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{H} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}
\end{aligned}
$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{V}_{k+2}^{(\tau)} \cdot \tilde{V}_{k+1}^{(1)} \cdot \tilde{H}^{*} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot$$

$$\cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{H} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\left[\begin{array}{ccc|c} & & & 0 \\ & F_k^{(\tau)} & & 0 \\ & & & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \cdot \left[\begin{array}{c|ccc} 1 & 0 & 0 & 0 \\ \hline 0 & & & \\ 0 & & F_{k+1} & \\ 0 & & & \end{array}\right].$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{V}_{k+2}^{(\tau)} \cdot \tilde{V}_{k+1}^{(1)} \cdot \tilde{H}^{*} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot$$
$$\cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{H} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \end{bmatrix}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{V}_{k+2}^{(\tau)} \cdot \tilde{V}_{k+1}^{(1)} \cdot \tilde{H}^{*} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot$$
$$\cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{H} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \end{bmatrix} \cdot \left[\begin{array}{cc|cc} \multicolumn{2}{c|}{\multirow{2}{*}{Q_k}} & 0 & 0 \\ & & 0 & 0 \\ \hline 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{array}\right]$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{V}_{k+2}^{(\tau)} \cdot \tilde{V}_{k+1}^{(1)} \cdot \tilde{H}^{*} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot$$
$$\cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{H} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}^*_{n-1} \cdot \ldots \cdot \tilde{V}^{(\tau)}_{k+2} \cdot \tilde{V}^{(1)}_{k+1} \cdot \tilde{H}^* \cdot \tilde{V}^{(1)}_{k} \cdot \ldots \cdot \tilde{V}^{(1)}_{2} \cdot$$
$$\cdot \tilde{F}^{(1)}_{1} \cdot \ldots \cdot \tilde{F}^{(1)}_{k-1} \cdot \tilde{H} \cdot \tilde{F}^{(\tau)}_{k} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_k \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\cdot \left[\begin{array}{cc|cc} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ \hline 0 & 0 & H & \\ 0 & 0 & & \end{array}\right] \cdot \qquad \begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{bmatrix}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{V}_{k+2}^{(\tau)} \cdot \tilde{V}_{k+1}^{(1)} \cdot \tilde{H}^{*} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot$$
$$\cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{H} \cdot \tilde{F}_{k}^{(\tau)} \cdot \tilde{F}_{k+1} \cdot \tilde{Q}_{k} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\begin{bmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \end{bmatrix}$$

Computation of New Product Structure (Single Shift)

$$A^{(1)} = \tilde{Q}_{n-1}^{*} \cdot \ldots \cdot \tilde{V}_{k+2}^{(\tau)} \cdot \tilde{V}_{k+1}^{(1)} \cdot \tilde{H}^{*} \cdot \tilde{V}_{k}^{(1)} \cdot \ldots \cdot \tilde{V}_{2}^{(1)} \cdot$$
$$\cdot \tilde{F}_{1}^{(1)} \cdot \ldots \cdot \tilde{F}_{k-1}^{(1)} \cdot \tilde{H} \cdot \tilde{F}_{k}^{(1)} \cdot \tilde{F}_{k+1}^{(\tau)} \cdot \ldots \cdot \tilde{Q}_{n-1}$$

$$\left[\begin{array}{ccc|c} & & & 0 \\ & F_k^{(1)} & & 0 \\ & & & 0 \\ \hline 0 & 0 & 0 & 1 \end{array}\right] \cdot \left[\begin{array}{c|ccc} 1 & 0 & 0 & 0 \\ \hline 0 & & & \\ 0 & & F_{k+1}^{(\tau)} & \\ 0 & & & \end{array}\right].$$