The implicit QZ algorithm for the palindromic eigenvalue problem

David S. Watkins

watkins@math.wsu.edu

Department of Mathematics

Washington State University

Context

Context

- LQG problem in discrete time (optimal control)

Context

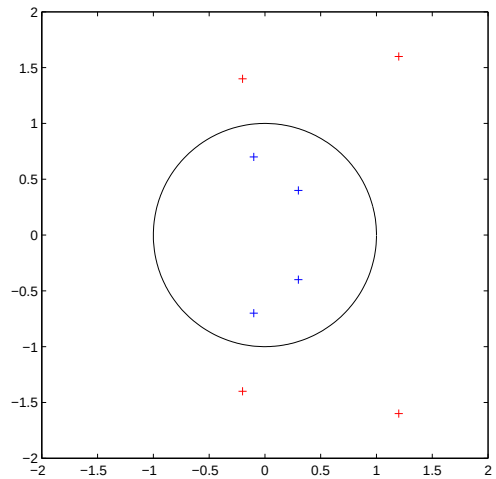
- LQG problem in discrete time (optimal control)
- $\Rightarrow$ symplectic eigenvalue problem

Context

- LQG problem in discrete time (optimal control)
- $\Rightarrow$ symplectic eigenvalue problem
- eigenvalue symmetry: λ, λ^{-1}

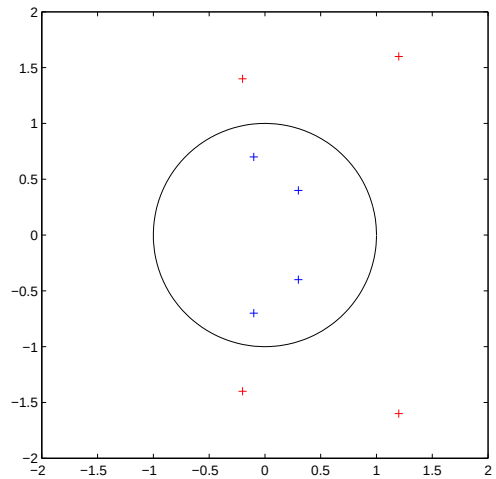
Context

- LQG problem in discrete time (optimal control)
- $\Rightarrow$ symplectic eigenvalue problem
- eigenvalue symmetry: λ, λ^{-1}



Context

- LQG problem in discrete time (optimal control)
- $\Rightarrow$ symplectic eigenvalue problem
- eigenvalue symmetry: λ, λ^{-1}



- want stable invariant subspace

Palindromic Eigenvalue Problem

Palindromic Eigenvalue Problem

- Mackey / Mackey / Mehl / Mehrmann

Palindromic Eigenvalue Problem

- Mackey / Mackey / Mehl / Mehrmann
- $(C - \lambda C^T)x = 0$

Palindromic Eigenvalue Problem

- Mackey / Mackey / Mehl / Mehrmann
- $(C - \lambda C^T)x = 0$
- generalized eigenvalue problem $A - \lambda B$

Palindromic Eigenvalue Problem

- Mackey / Mackey / Mehl / Mehrmann
- $(C - \lambda C^T)x = 0$
- generalized eigenvalue problem $A - \lambda B$
- $B = A^T$

Palindromic Eigenvalue Problem

- Mackey / Mackey / Mehl / Mehrmann
- $(C - \lambda C^T)x = 0$
- generalized eigenvalue problem $A - \lambda B$
- $B = A^T$
- $(C - \lambda C^T)x = 0 \quad \Leftrightarrow \quad x^T(C^T - \lambda C) = 0$

Palindromic Eigenvalue Problem

- Mackey / Mackey / Mehl / Mehrmann
- $(C - \lambda C^T)x = 0$
- generalized eigenvalue problem $A - \lambda B$
- $B = A^T$
- $(C - \lambda C^T)x = 0 \quad \Leftrightarrow \quad x^T(C^T - \lambda C) = 0$
- eigenvalue symmetry: λ, λ^{-1}

Palindromic Eigenvalue Problem

- Mackey / Mackey / Mehl / Mehrmann
- $(C - \lambda C^T)x = 0$
- generalized eigenvalue problem $A - \lambda B$
- $B = A^T$
- $(C - \lambda C^T)x = 0 \quad \Leftrightarrow \quad x^T(C^T - \lambda C) = 0$
- eigenvalue symmetry: λ, λ^{-1}
- Formulate control problems as palindromic eigenvalue problems.

QR-like algorithms

QR-like algorithms for palindromic problems

QR-like algorithms for palindromic problems

- C. Schröder (Berlin)

QR-like algorithms for palindromic problems

- C. Schröder (Berlin)
- explicit QR-like algorithm

QR-like algorithms for palindromic problems

- C. Schröder (Berlin)
- explicit QR-like algorithm (It works!)

QR-like algorithms for palindromic problems

- C. Schröder (Berlin)
- explicit QR-like algorithm (It works!)
- implicit QR-like algorithm (bulge chase)

QR-like algorithms for palindromic problems

- C. Schröder (Berlin)
- explicit QR-like algorithm (It works!)
- implicit QR-like algorithm (bulge chase) (It works!)

QR-like algorithms for palindromic problems

- C. Schröder (Berlin)
- explicit QR-like algorithm (It works!)
- implicit QR-like algorithm (bulge chase) (It works!)
- Are they equivalent?

The Bulge Chase

The Bulge Chase

- compact form needed

The Bulge Chase

- compact form needed

$$C = \left[\begin{array}{cccc|cccc} & & & & & & 0 & * \\ & & & & & 0 & * & * \\ & & & & 0 & * & * & * \\ & & & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

$$C^T = \left[\begin{array}{cccc|cccc} & & & * & * & * & * & * \\ & & & * & & * & * & * \\ & & & * & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & & 0 & * & * & * & * & * \\ & 0 & * & * & * & * & * & * \\ 0 & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

$$C^T = \left[\begin{array}{cccc|cccc} & & & * & * & * & * & * \\ & & & & & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & & 0 & * & * & * & * & * \\ & 0 & * & * & * & * & * & * \\ 0 & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

- not always attainable!

The Bulge Chase

The Bulge Chase

- single-shift case

The Bulge Chase

- single-shift case (for simplicity)

The Bulge Chase

- single-shift case (for simplicity)
- Pick a shift μ.

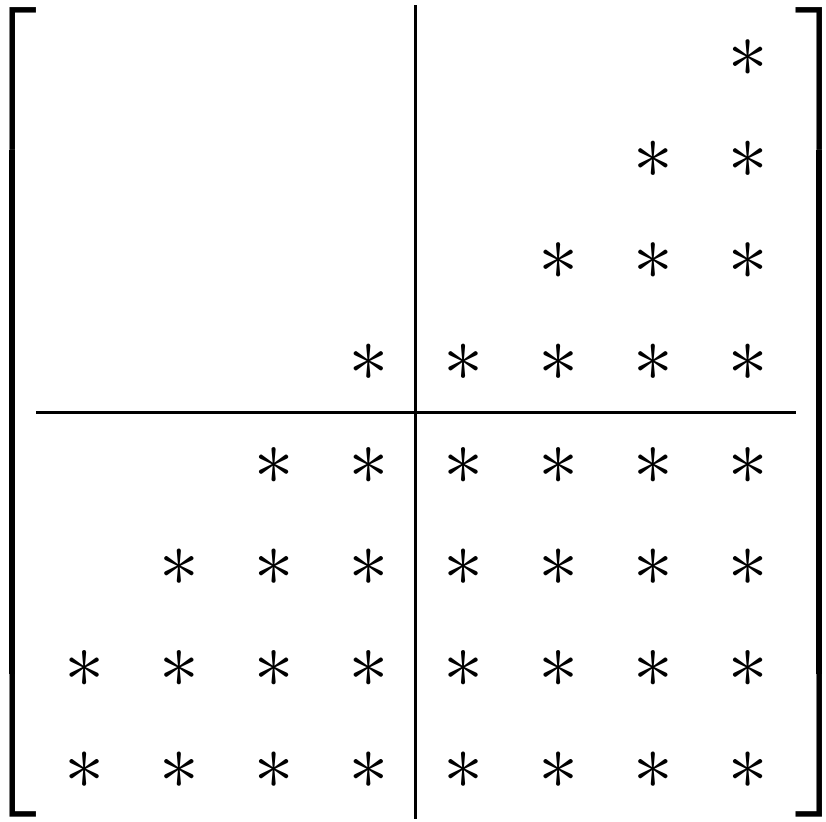
The Bulge Chase

- single-shift case (for simplicity)
- Pick a shift μ.
- $x = (C - \mu C^T)e_1 = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \alpha \\ \beta \end{bmatrix}$

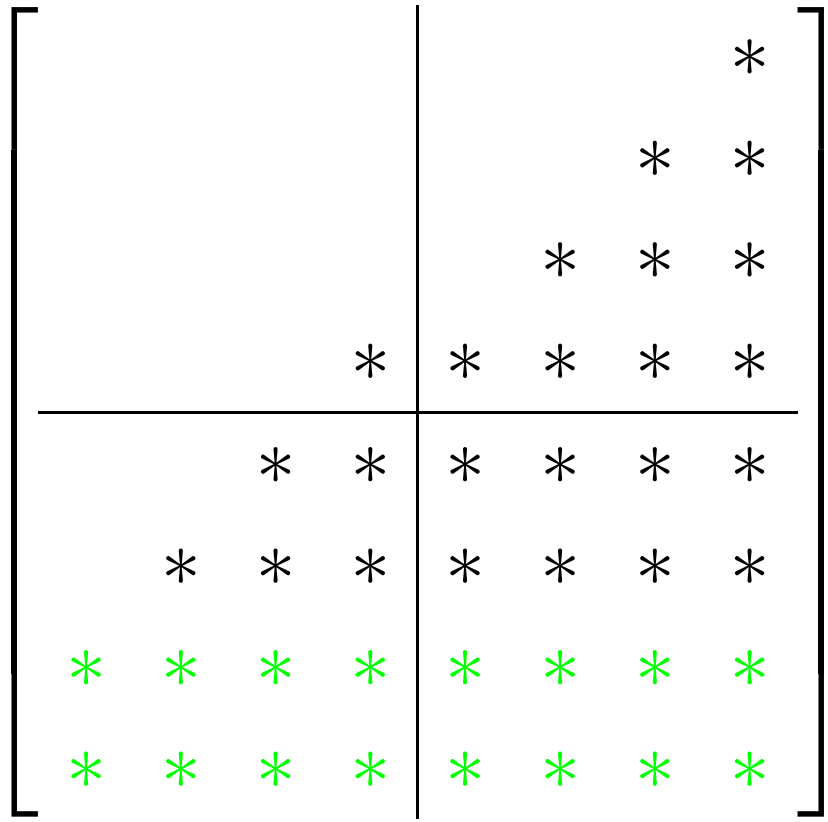
The Bulge Chase

- single-shift case (for simplicity)
- Pick a shift μ.
- $x = (C - \mu C^T)e_1 = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ \alpha \\ \beta \end{bmatrix}$
- Build a Givens rotation (for example) that annihilates α.

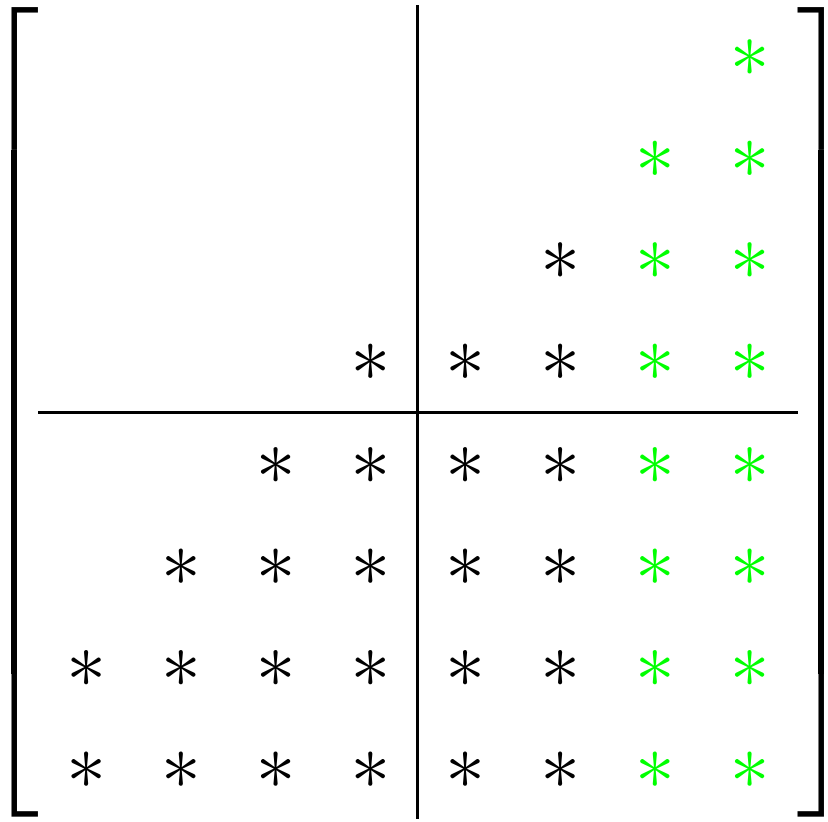
The Bulge Chase



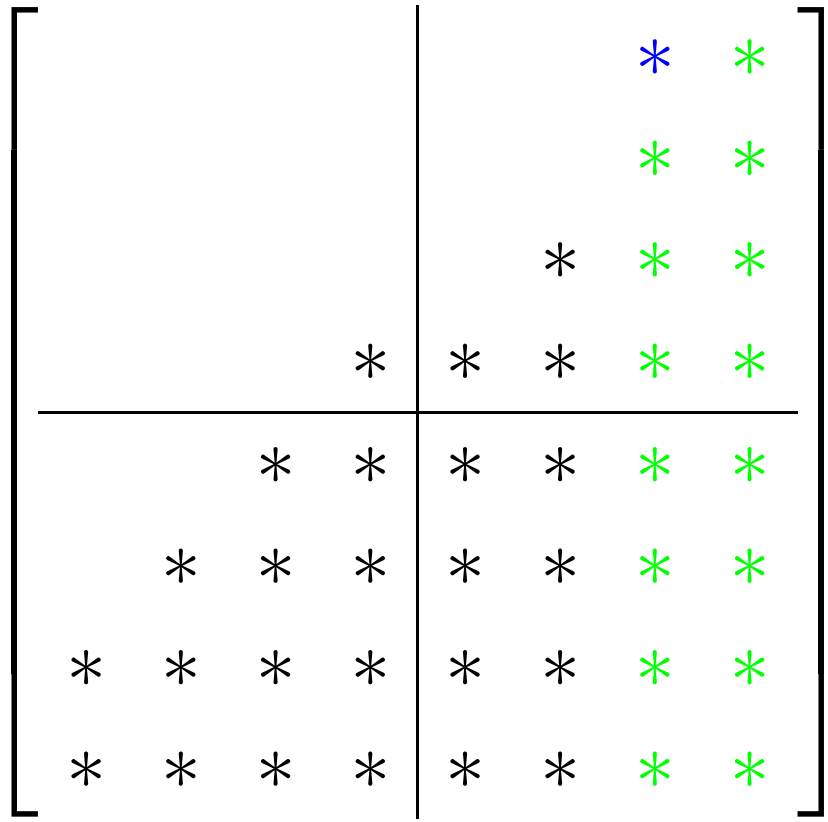
The Bulge Chase



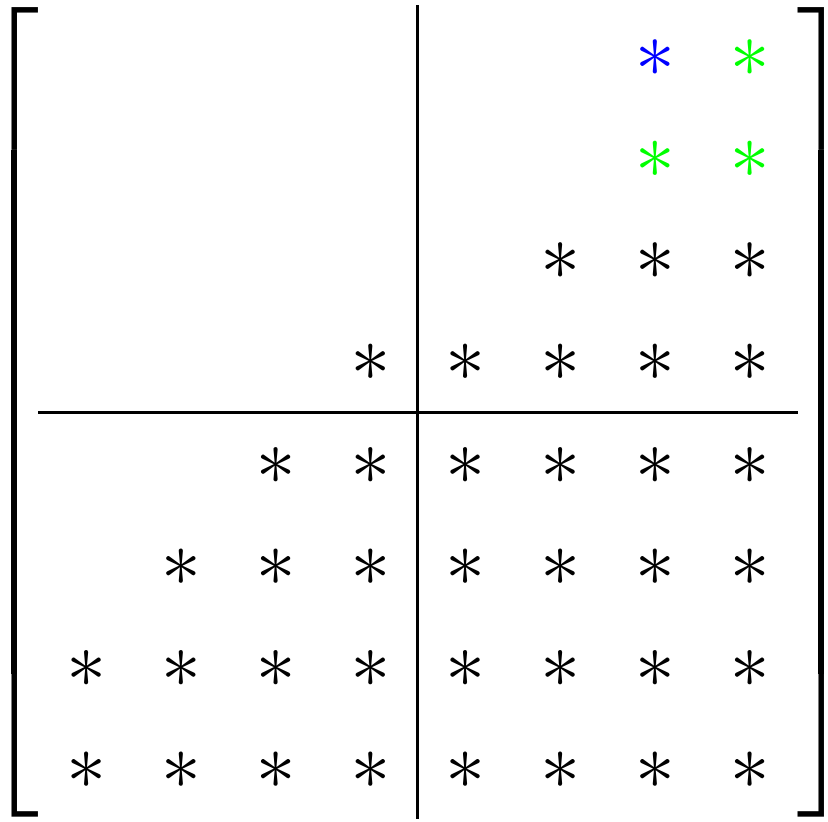
The Bulge Chase



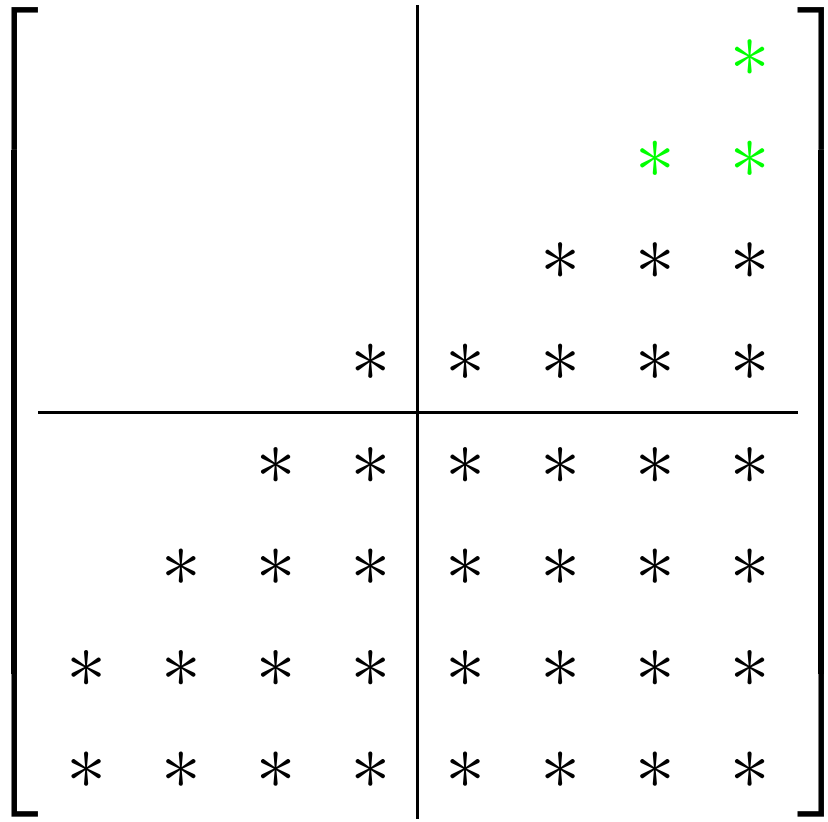
The Bulge Chase



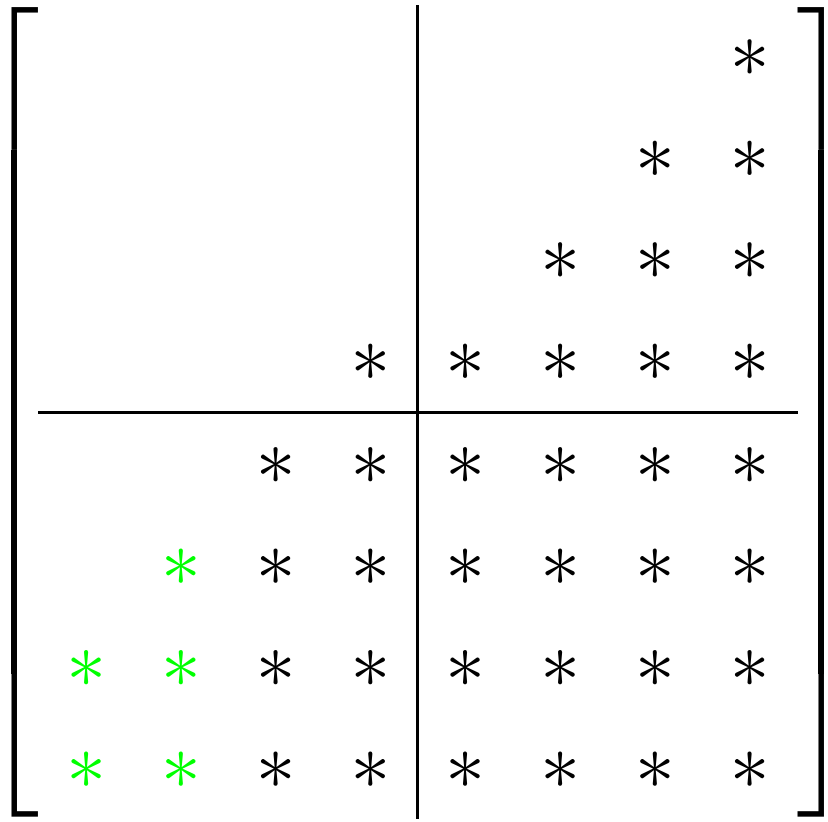
The Bulge Chase



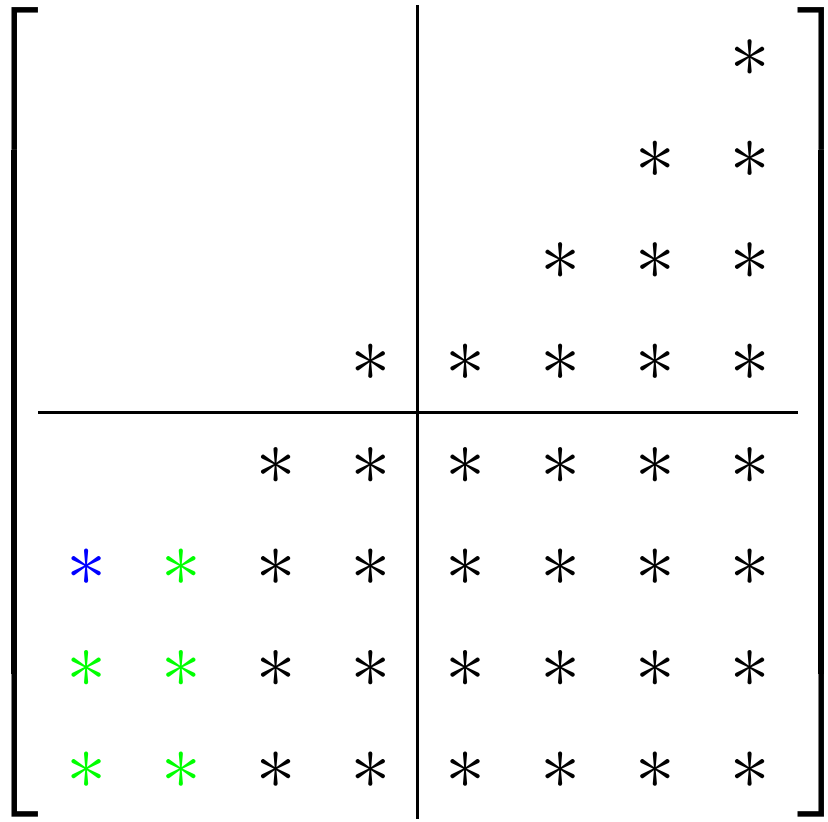
The Bulge Chase



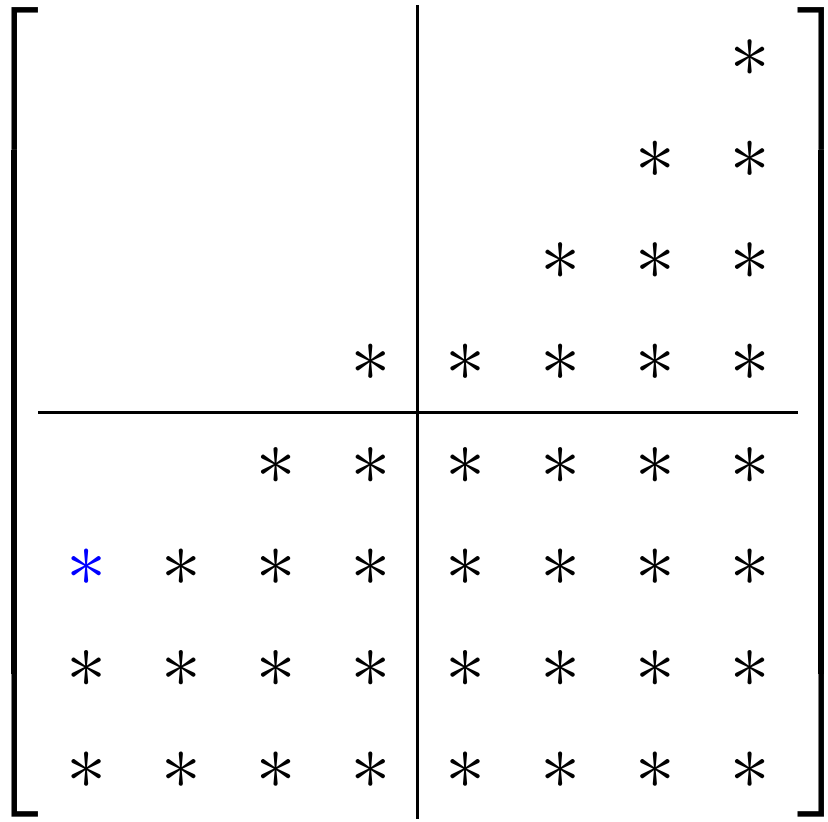
The Bulge Chase



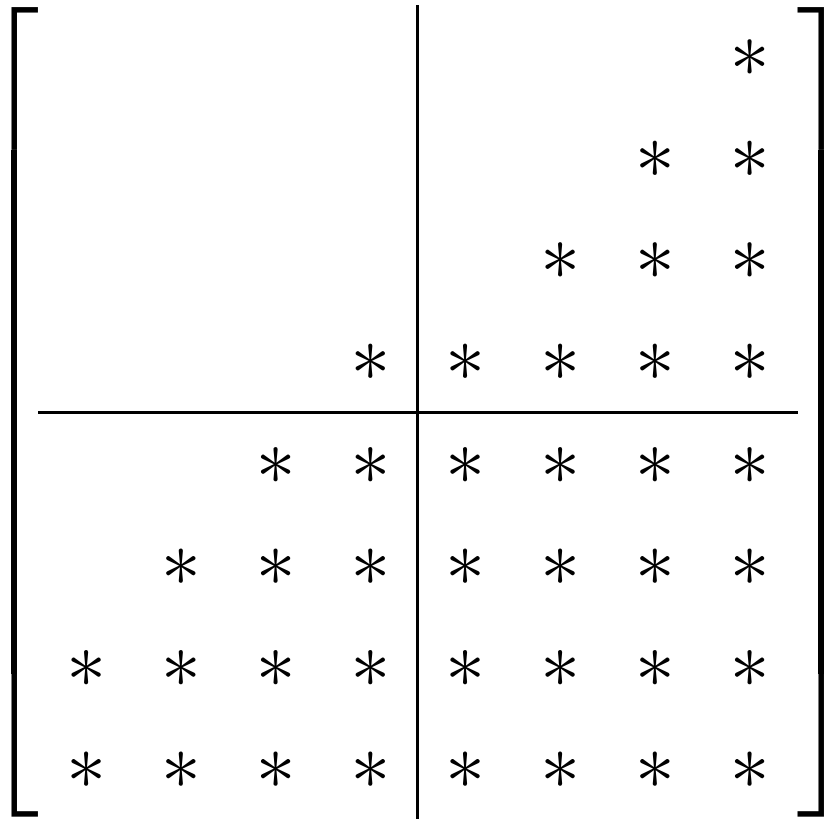
The Bulge Chase



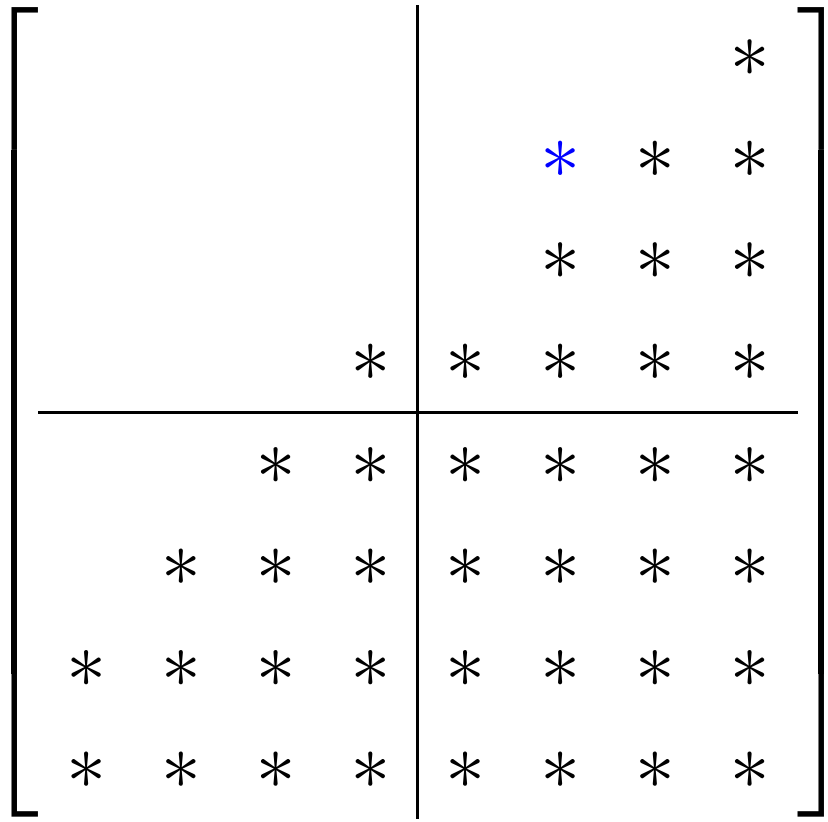
The Bulge Chase



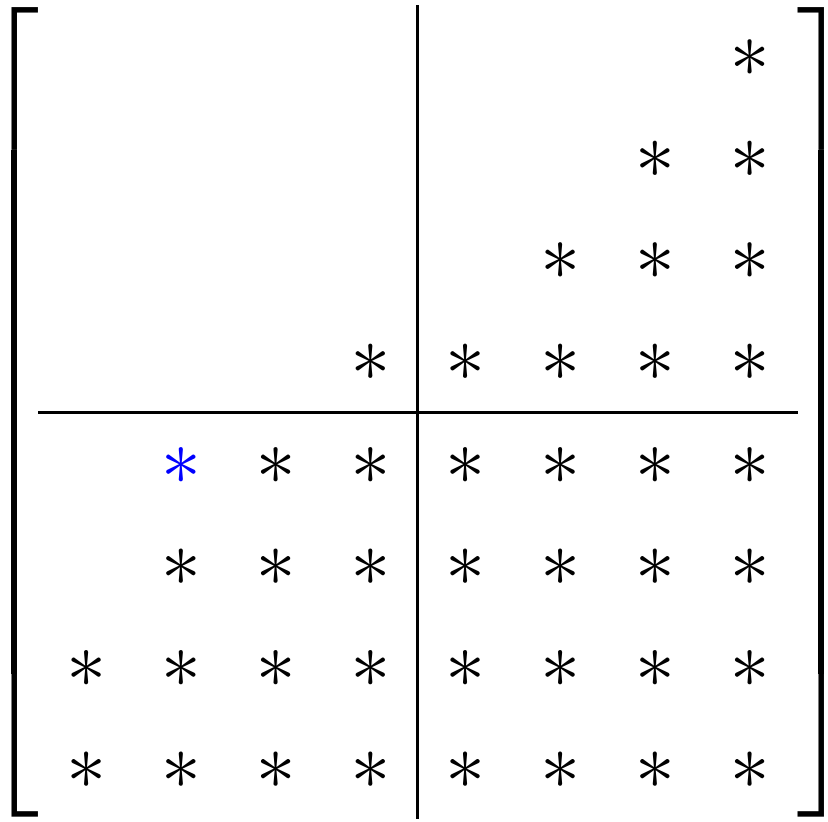
The Bulge Chase



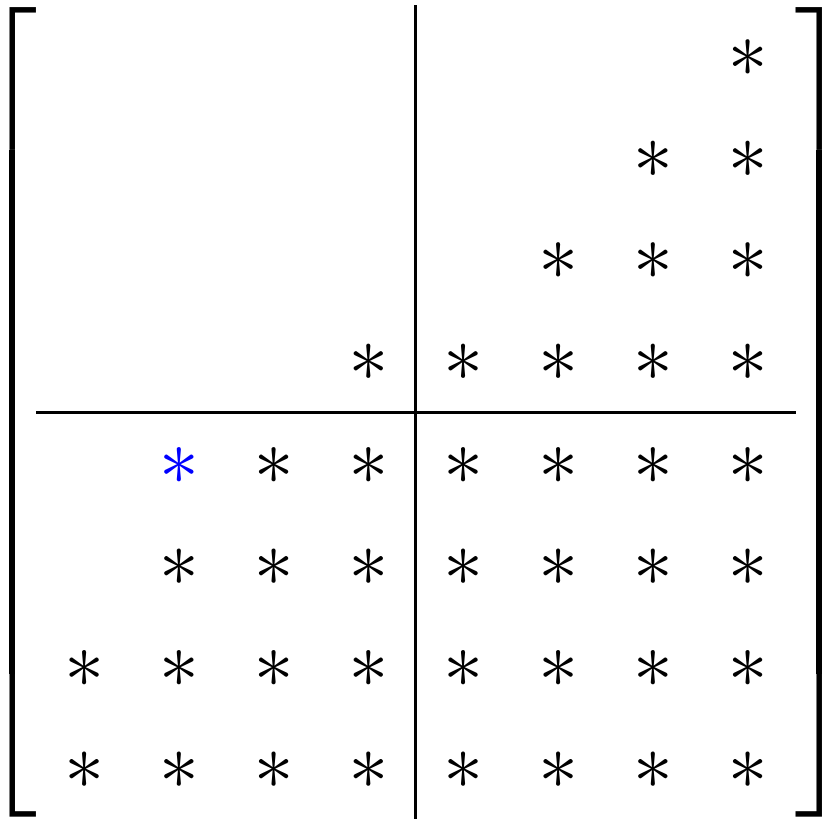
The Bulge Chase



The Bulge Chase



The Bulge Chase



But what happens when we get to the middle?

Bulge Chase in the Pencil

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & & * & * \\ & & & & & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

Bulge Chase in the Pencil

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & * & * & * \\ & & & & & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

Bulge Chase in the Pencil

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & & * & * \\ & & & & * & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & * & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

Bulge pencils on a collision course!

Bulge Chase in the Pencil

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & & * & * \\ & & & & * & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & * & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

Half a step further:

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & & * & * \\ & & & & & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

Swap the shifts

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & & * & * \\ & & & & & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

Swap the shifts

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & & * & * \\ & & & & & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

Reverse the process

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & & * & * \\ & & & & * & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & * & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

Reverse the process

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & * & * & * \\ & & & & & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

Reverse the process

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & & * & * \\ & & & & & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

Bulge Chase is complete

$$C - \lambda C^T = \left[\begin{array}{cccc|cccc} & & & & & & * & * \\ & & & & & * & * & * \\ & & & & * & * & * & * \\ & & & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

- This works very well.

- This works very well.
- Question:

- This works very well.
- Question: How do we explain it?

- This works very well.
- Question: How do we explain it?
- Answer:

- This works very well.
- Question: How do we explain it?
- Answer: I don’t have time …

- This works very well.
- Question: How do we explain it?
- Answer: I don’t have time …
- …but I’ll try.

- This works very well.
- Question: How do we explain it?
- Answer: I don’t have time …
- …but I’ll try.
- Compare with standard QZ.

QZ algorithm,

QZ algorithm, implicit version

QZ algorithm, implicit version

- $A - \lambda B$ (Hessenberg, triangular)

QZ algorithm, implicit version

- $A - \lambda B$ (Hessenberg, triangular)
- pick shifts $\mu_1, \ldots \mu_m$

QZ algorithm, implicit version

- $A - \lambda B$ (Hessenberg, triangular)
- pick shifts $\mu_1, \ldots \mu_m$
- $p(z) = (z - \mu_1) \cdots (z - \mu_m)$

QZ algorithm, implicit version

- $A - \lambda B$ (Hessenberg, triangular)
- pick shifts $\mu_1, \ldots \mu_m$
- $p(z) = (z - \mu_1) \cdots (z - \mu_m)$
- $x = p(AB^{-1})e_1$

QZ algorithm, implicit version

- $A - \lambda B$ (Hessenberg, triangular)
- pick shifts $\mu_1, \ldots \mu_m$
- $p(z) = (z - \mu_1) \cdots (z - \mu_m)$
- $x = p(AB^{-1})e_1$
- Make a bulge,

QZ algorithm, implicit version

- $A - \lambda B$ (Hessenberg, triangular)
- pick shifts $\mu_1, \ldots \mu_m$
- $p(z) = (z - \mu_1) \cdots (z - \mu_m)$
- $x = p(AB^{-1})e_1$
- Make a bulge, then chase it.

QZ algorithm, implicit version

- $A - \lambda B$ (Hessenberg, triangular)
- pick shifts $\mu_1, \ldots \mu_m$
- $p(z) = (z - \mu_1) \cdots (z - \mu_m)$
- $x = p(AB^{-1})e_1$
- Make a bulge, then chase it.
- Get $\hat{A} - \lambda \hat{B}$

QZ algorithm,

QZ algorithm, explicit version

QZ algorithm, explicit version

- $p(AB^{-1}) = QR$

QZ algorithm, explicit version

- $p(AB^{-1}) = QR \qquad p(B^{-1}A) = Z\tilde{R}$

QZ algorithm, explicit version

- $p(AB^{-1}) = QR \qquad p(B^{-1}A) = Z\tilde{R}$
- $\hat{A} = Q^*AZ, \quad \hat{B} = Q^*BZ$

QZ algorithm, explicit version

- $p(AB^{-1}) = QR \qquad p(B^{-1}A) = Z\tilde{R}$
- $\hat{A} = Q^*AZ, \quad \hat{B} = Q^*BZ$
- Explicit QZ step is complete.

QZ algorithm, explicit version

- $p(AB^{-1}) = QR \qquad p(B^{-1}A) = Z\tilde{R}$
- $\hat{A} = Q^*AZ, \quad \hat{B} = Q^*BZ$
- Explicit QZ step is complete.
- $\hat{A}\hat{B}^{-1} = Q^*(AB^{-1})Q$

QZ algorithm, explicit version

- $p(AB^{-1}) = QR \qquad p(B^{-1}A) = Z\tilde{R}$
- $\hat{A} = Q^*AZ, \quad \hat{B} = Q^*BZ$
- Explicit QZ step is complete.
- $\hat{A}\hat{B}^{-1} = Q^*(AB^{-1})Q$ (QR iteration on AB^{-1})

QZ algorithm, explicit version

- $p(AB^{-1}) = QR \qquad p(B^{-1}A) = Z\tilde{R}$
- $\hat{A} = Q^*AZ, \quad \hat{B} = Q^*BZ$
- Explicit QZ step is complete.
- $\hat{A}\hat{B}^{-1} = Q^*(AB^{-1})Q$ (QR iteration on AB^{-1})
- $\hat{B}^{-1}\hat{A} = Z^*(B^{-1}A)Z$

QZ algorithm, explicit version

- $p(AB^{-1}) = QR \qquad p(B^{-1}A) = Z\tilde{R}$
- $\hat{A} = Q^*AZ, \quad \hat{B} = Q^*BZ$
- Explicit QZ step is complete.
- $\hat{A}\hat{B}^{-1} = Q^*(AB^{-1})Q$ (QR iteration on AB^{-1})
- $\hat{B}^{-1}\hat{A} = Z^*(B^{-1}A)Z$ (QR iteration on $B^{-1}A$)

Explicit = Implicit?

Explicit = Implicit?

- AB^{-1} and $B^{-1}A$ are upper Hessenberg.

Explicit = Implicit?

- AB^{-1} and $B^{-1}A$ are upper Hessenberg.
- Utilize the Hessenberg form.

Explicit $=$ Implicit?

- AB^{-1} and $B^{-1}A$ are upper Hessenberg.
- Utilize the Hessenberg form.
- implicit-Q theorem, or ...

Explicit $=$ Implicit?

- AB^{-1} and $B^{-1}A$ are upper Hessenberg.
- Utilize the Hessenberg form.
- implicit-Q theorem, or ...
- work directly with the Krylov subspaces.

Explicit $=$ Implicit?

- AB^{-1} and $B^{-1}A$ are upper Hessenberg.
- Utilize the Hessenberg form.
- implicit-Q theorem, or ...
- work directly with the Krylov subspaces.
- time permitting ...

Back to the palindromic case:

Back to the palindromic case:

- Bulge chase gives $\hat{C} = G^{-1} C G^{-T}$

Back to the palindromic case:

- Bulge chase gives $\hat{C} = G^{-1} C G^{-T}$
- $\hat{C}\hat{C}^{-T} = G^{-1}(CC^{-T})G$

Back to the palindromic case:

- Bulge chase gives $\hat{C} = G^{-1} C G^{-T}$
- $\hat{C}\hat{C}^{-T} = G^{-1}(CC^{-T})G$ (similarity)

Back to the palindromic case:

- Bulge chase gives $\hat{C} = G^{-1} C G^{-T}$
- $\hat{C}\hat{C}^{-T} = G^{-1}(CC^{-T})G$ (similarity)
- $\hat{C}^{-T}\hat{C} = G^{T}(C^{-T}C)G^{-T}$

Back to the palindromic case:

- Bulge chase gives $\hat{C} = G^{-1} C G^{-T}$
- $\hat{C}\hat{C}^{-T} = G^{-1}(CC^{-T})G$ (similarity)
- $\hat{C}^{-T}\hat{C} = G^{T}(C^{-T}C)G^{-T}$ (similarity)

Back to the palindromic case:

- Bulge chase gives $\hat{C} = G^{-1} C G^{-T}$
- $\hat{C}\hat{C}^{-T} = G^{-1}(CC^{-T})G$ (similarity)
- $\hat{C}^{-T}\hat{C} = G^{T}(C^{-T}C)G^{-T}$ (similarity)
- Need $p(CC^{-T}) = G\tilde{R}$

Back to the palindromic case:

- Bulge chase gives $\hat{C} = G^{-1} C G^{-T}$
- $\hat{C}\hat{C}^{-T} = G^{-1}(CC^{-T})G$ (similarity)
- $\hat{C}^{-T}\hat{C} = G^{T}(C^{-T}C)G^{-T}$ (similarity)
- Need $p(CC^{-T}) = G\tilde{R}$ and $p(C^{-T}C) = G^{-T}R$

Back to the palindromic case:

- Bulge chase gives $\hat{C} = G^{-1} C G^{-T}$
- $\hat{C}\hat{C}^{-T} = G^{-1}(CC^{-T})G$ (similarity)
- $\hat{C}^{-T}\hat{C} = G^{T}(C^{-T}C)G^{-T}$ (similarity)
- Need $p(CC^{-T}) = G\tilde{R}$ and $p(C^{-T}C) = G^{-T}R$

or something like that.

What we actually get:

What we actually get:

- $r(CC^{-T}) = GL$

What we actually get:

- $r(CC^{-T}) = GL$
- where $r(z) = \prod_{k=1}^{m} \frac{z - \mu_k}{\mu_k z - 1}$

What we actually get:

- $r(CC^{-T}) = GL$
- where $r(z) = \prod_{k=1}^{m} \frac{z - \mu_k}{\mu_k z - 1}$
- L is *lower* triangular.

What we actually get:

- $r(CC^{-T}) = GL$
- where $r(z) = \prod_{k=1}^{m} \frac{z - \mu_k}{\mu_k z - 1}$
- L is *lower* triangular.
- $r(C^{-T}C) = G^{-T}R$

What we actually get:

- $r(CC^{-T}) = GL$
- where $r(z) = \prod_{k=1}^{m} \frac{z - \mu_k}{\mu_k z - 1}$
- L is *lower* triangular.
- $r(C^{-T}C) = G^{-T}R$
- $r(CC^{-T})^{-T} = r(C^{-T}C), \quad R = L^{-T}$

Our Approach:

Our Approach:

- want $r(CC^{-T}) = GL$

Our Approach:

- want $r(CC^{-T}) = GL$
- *Define* $L = G^{-1}r(CC^{-T})$

Our Approach:

- want $r(CC^{-T}) = GL$
- *Define* $L = G^{-1}r(CC^{-T})$
- Then show that L is lower triangular.

Our Approach:

- want $r(CC^{-T}) = GL$
- *Define* $L = G^{-1}r(CC^{-T})$
- Then show that L is lower triangular.
- This is not entirely straightforward.

Our Approach:

- want $r(CC^{-T}) = GL$
- *Define* $L = G^{-1}r(CC^{-T})$
- Then show that L is lower triangular.
- This is not entirely straightforward.
- Utilize the Hessenberg form.

Recall that

$$C = \left[\begin{array}{cccc|cccc} & & & * & & & 0 & * \\ & & & & & 0 & * & * \\ & & & & 0 & * & * & * \\ & & & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

This implies

This implies

$$CC^{-T} = \left[\begin{array}{cccc|cccc} * & * & * & * & * & & & \\ * & * & * & * & * & & & \\ * & * & * & * & * & & & \\ * & * & * & * & * & & & \\ \hline * & * & * & * & * & * & & \\ * & * & * & * & * & * & * & \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

This implies

$$CC^{-T} = \left[\begin{array}{cccc|cccc} * & * & * & * & * & & & \\ * & * & * & * & * & & & \\ * & * & * & * & * & & & \\ * & * & * & * & * & & & \\ \hline * & * & * & * & * & * & & \\ * & * & * & * & * & * & * & \\ * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \end{array}\right]$$

- partly lower Hessenberg ($n/2 - 1$ columns)

and

and

$$C^{-T}C = \left[\begin{array}{cccc|cccc} * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ & & * & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ & & * & * & * & * & * & * \\ & & * & * & * & * & * & * \\ & & * & * & * & * & * & * \end{array}\right]$$

and

$$C^{-T}C = \left[\begin{array}{cccc|cccc} * & * & * & * & * & * & * & * \\ * & * & * & * & * & * & * & * \\ & * & * & * & * & * & * & * \\ & & * & * & * & * & * & * \\ \hline & & * & * & * & * & * & * \\ & & * & * & * & * & * & * \\ & & * & * & * & * & * & * \\ & & * & * & * & * & * & * \end{array}\right]$$

- partly upper Hessenberg ($n/2 - 2$ columns)

- Use both.

- Use both.
- $L = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix}$

- Use both.
- $L = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix}$
- Using CC^{-T},

- Use both.
- $L = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix}$
- Using CC^{-T}, deduce $L_{12} = 0 \ldots$

- Use both.
- $L = \begin{bmatrix} L_{11} & L_{12} \\ L_{21} & L_{22} \end{bmatrix}$
- Using CC^{-T}, deduce $L_{12} = 0 \ldots$
- and L_{22} is lower triangular.

- $L = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}$

- $L = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}$
- just need L_{11} lower triangular

- $L = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}$
- just need L_{11} lower triangular
- $R = L^{-T} = \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}$

- $L = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}$
- just need L_{11} lower triangular
- $R = L^{-T} = \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}$
- Using $C^{-T}C$,

- $L = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}$
- just need L_{11} lower triangular
- $R = L^{-T} = \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}$
- Using $C^{-T}C$, deduce that R_{11} is upper triangular.

- $L = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}$
- just need L_{11} lower triangular
- $R = L^{-T} = \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}$
- Using $C^{-T}C$, deduce that R_{11} is upper triangular.
- Hence L_{11} is lower triangular.

- $L = \begin{bmatrix} L_{11} & \\ L_{21} & L_{22} \end{bmatrix}$
- just need L_{11} lower triangular
- $R = L^{-T} = \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}$
- Using $C^{-T}C$, deduce that R_{11} is upper triangular.
- Hence L_{11} is lower triangular.
- done!

Conclusion:

Conclusion:

Schröder's palindromic bulge-chasing algorithm

Conclusion:

Schröder's palindromic bulge-chasing algorithm effects a combination QL and QR iteration

Conclusion:

Schröder's palindromic bulge-chasing algorithm effects a combination QL and QR iteration driven by a rational function

$$r(z) = \prod_{k=1}^{m} \frac{z - \mu_k}{\mu_k z - 1}$$

Conclusion:

Schröder's palindromic bulge-chasing algorithm effects a combination QL and QR iteration driven by a rational function

$$r(z) = \prod_{k=1}^{m} \frac{z - \mu_k}{\mu_k z - 1}$$

with shifts $\mu_1, \ldots, \mu_m$ in the numerator and shifts $\mu_1^{-1}, \ldots, \mu_m^{-1}$ in the denominator.

Conclusion:

Schröder's palindromic bulge-chasing algorithm effects a combination QL and QR iteration driven by a rational function

$$r(z) = \prod_{k=1}^{m} \frac{z - \mu_k}{\mu_k z - 1}$$

with shifts $\mu_1, \ldots, \mu_m$ in the numerator and shifts $\mu_1^{-1}, \ldots, \mu_m^{-1}$ in the denominator.

That's all, folks!