Exploiting BiStructured Computational Problems

Ralph Byers

University of Kansas

Joint work with

Daniel Kressner

University of Zagreb

Outline

This is a talk about solving computational problems with methods **not preserving the special structure** of the underlying problem!

Outline

This is a talk about solving computational problems with methods not preserving the special structure of the underlying problem!

1. Introduction
2. Simple (embarrassingly simple?) Examples
3. Cleaning up BiStructured problems
4. Less simple examples
5. Conclusions

Roughly speaking . . .

Roughly speaking . . .

All computational problems have special structure!

Roughly speaking . . .

All computational problems have special structure!

Few have readily available, easily used numerical methods.

Computational Issues

Specialized methods typically have . . .

Computational Issues

Specialized methods typically have . . .

- mathematical interest and elegance,

Computational Issues

Specialized methods typically have . . .

- mathematical interest and elegance,
- reduced computational costs,

Computational Issues

Specialized methods typically have . . .

- mathematical interest and elegance,
- reduced computational costs,
- improved numerical stability,

Computational Issues

Specialized methods typically have . . .

- mathematical interest and elegance,
- reduced computational costs,
- improved numerical stability,
- and sometimes, the ability to solve otherwise intractable problems.

Practical Issues

But, specialized methods are rarely used (even by us!), because . . .

Practical Issues

But, specialized methods are rarely used (even by us!), because ...

- The structure of the problem may be unrecognized or ill-understood.

Practical Issues

But, specialized methods are rarely used (even by us!), because . . .

- The structure of the problem may be unrecognized or ill-understood.
- A satisfactory structured algorithm may be unknown or may not exist.

Practical Issues

But, specialized methods are rarely used (even by us!), because . . .

- The structure of the problem may be unrecognized or ill-understood.
- A satisfactory structured algorithm may be unknown or may not exist.
- A satisfactory implementation may be unavailable, difficult or expensive. (. . . or eig is too easy to use.)

Questions

By not using structure preserving numerical methods, one gives up elegance and reduced computational cost.

- What about numerical stability?
- Is there an easy, inexpensive way to recover the accuracy of a structured numerical method from the computed results of an unstructured method?

$A \in \mathbf{R}^{n \times n}$

$$(A + E)x = \lambda x, \quad (\lambda, x) \in \mathbf{C} \times \mathbf{C}^n, \quad \textit{but}\ldots$$

$A \in \mathbf{R}^{n \times n}$

$$(A + E)x = \lambda x, \quad (\lambda, x) \in \mathbf{C} \times \mathbf{C}^n, \quad \textit{but}\ldots$$

- $E \in \mathbf{R}^{n \times n}$, for real arithmetic algorithms,

$A \in \mathbf{R}^{n \times n}$

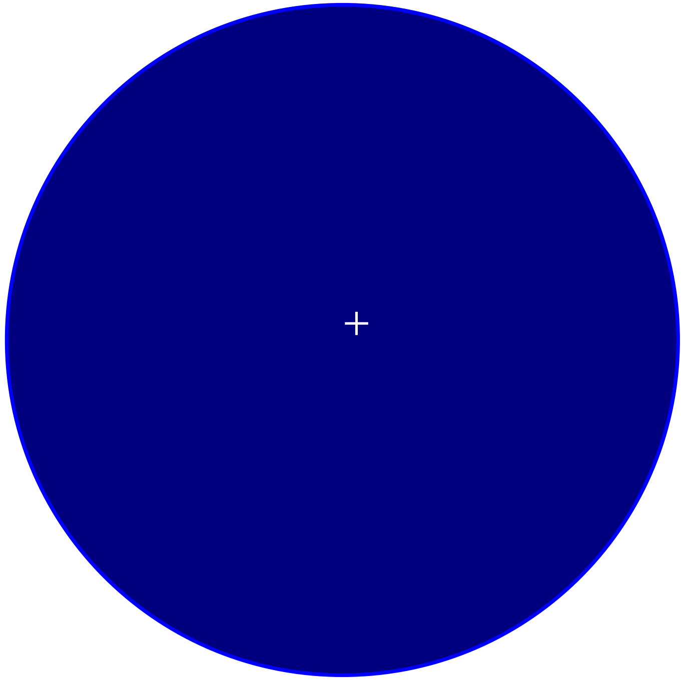
$$(A + E)x = \lambda x, \quad (\lambda, x) \in \mathbf{C} \times \mathbf{C}^n, \quad \textit{but} \ldots$$

- $E \in \mathbf{R}^{n \times n}$, for real arithmetic algorithms,
- $E \in \mathbf{C}^{n \times n}$, for complex arithmetic algorithms.

$A \in \mathbf{R}^{n\times n}$

$$(A+E)x = \lambda x, \quad (\lambda, x) \in \mathbf{C} \times \mathbf{C}^n, \quad \textit{but}\ldots$$

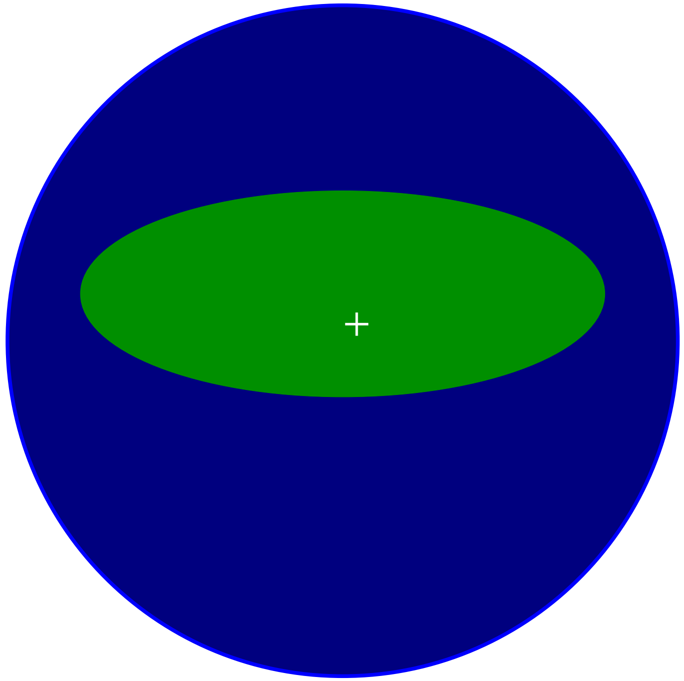
- $E \in \mathbf{R}^{n\times n}$, for real arithmetic algorithms,
- $E \in \mathbf{C}^{n\times n}$, for complex arithmetic algorithms.
- Are real arithmetic algorithms significantly more accurate?
- Can real-arithmetic-accuracy be recovered from a complex arithmetic computation?



$\underline{A \in \mathbf{R}^{n \times n}}$

$$\{\lambda(A+E) \mid E = E \in \mathbf{C}^{n\times n},\ \|E\| \le \varepsilon\}$$

ε-pseudospectral component containing an isolated, complex eigenvalue $\lambda = +$.



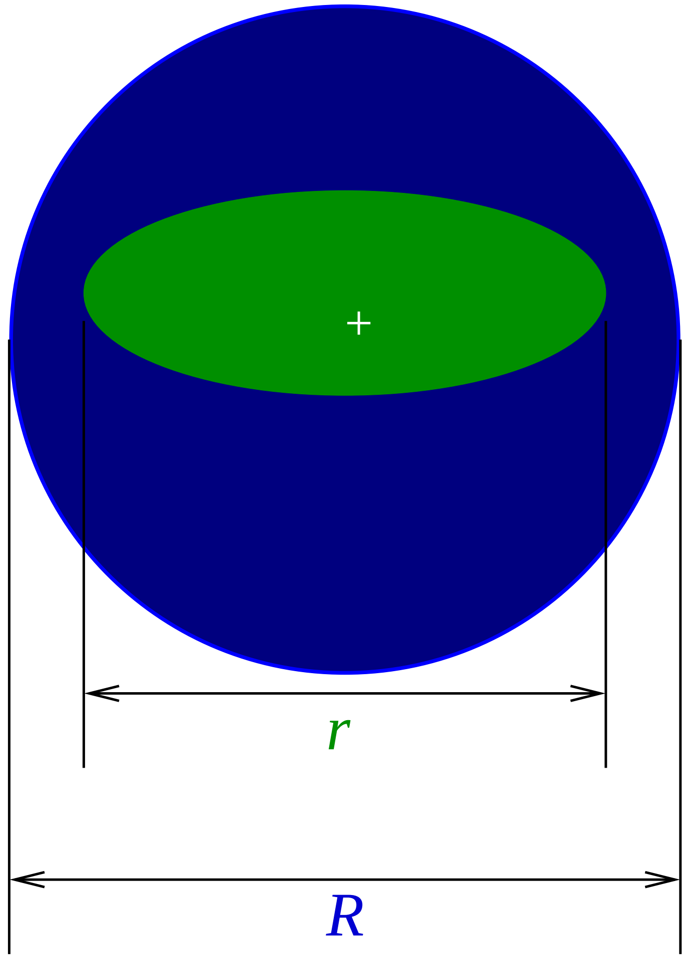
$A \in \mathbf{R}^{n\times n}$

$\{\lambda(A+E) \mid E = E \in \mathbf{C}^{n\times n},\ \|E\| \leq \varepsilon\}$ and

$\{\lambda(A+E) \mid E = E \in \mathbf{R}^{n\times n},\ \|E\| \leq \varepsilon\}$

ε-real-pseudospectral component containing an isolated complex eigenvalue $\lambda = +$.

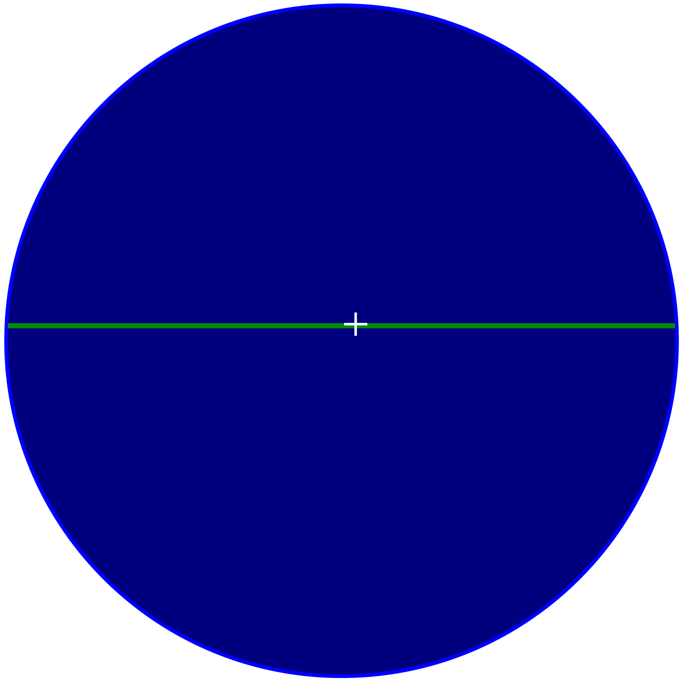
$$A \in \mathbf{R}^{n \times n}$$



$$r \leq R \leq \sqrt{2} r$$

B, Kressner (2004). See also Karow(2003)

$A \in \mathbf{R}^{n\times n}$



$\{\lambda(A+E) \mid E = E \in \mathbf{C}^{n\times n},\ \|E\| \le \varepsilon\}$ and
$\{\lambda(A+E) \mid E = E \in \mathbf{R}^{n\times n},\ \|E\| \le \varepsilon\}$
ε-real-pseudospectral component containing an isolated, real eigenvalue $\lambda = +$.

$A \in \mathbf{R}^{n \times n}$

Are real arithmetic algorithms applied to real matrices significantly more accurate than complex arithmetic algorithms?

$A \in \mathbf{R}^{n\times n}$

Are real arithmetic algorithms applied to real matrices significantly more accurate than complex arithmetic algorithms?

- No, worst case eigenvalue perturbations are within a factor of $\sqrt{2}$.

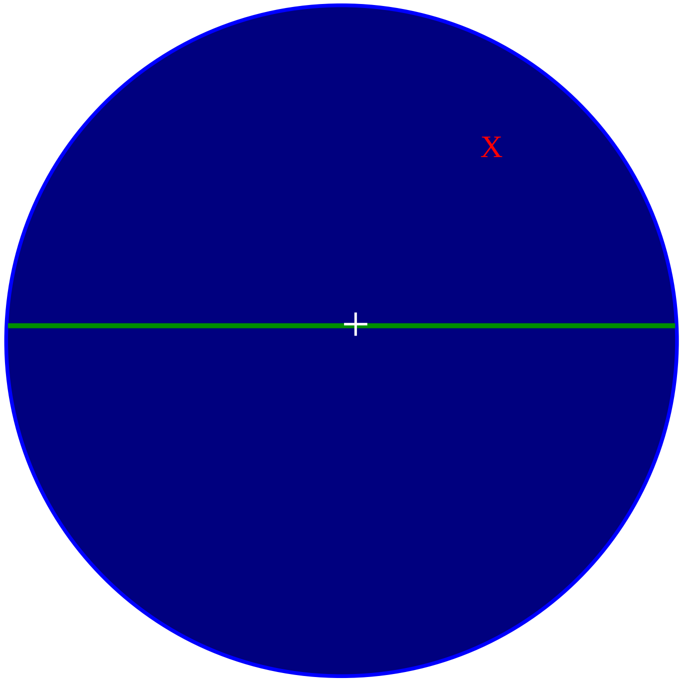
$A \in \mathbf{R}^{n \times n}$

Are real arithmetic algorithms applied to real matrices significantly more accurate than complex arithmetic algorithms?

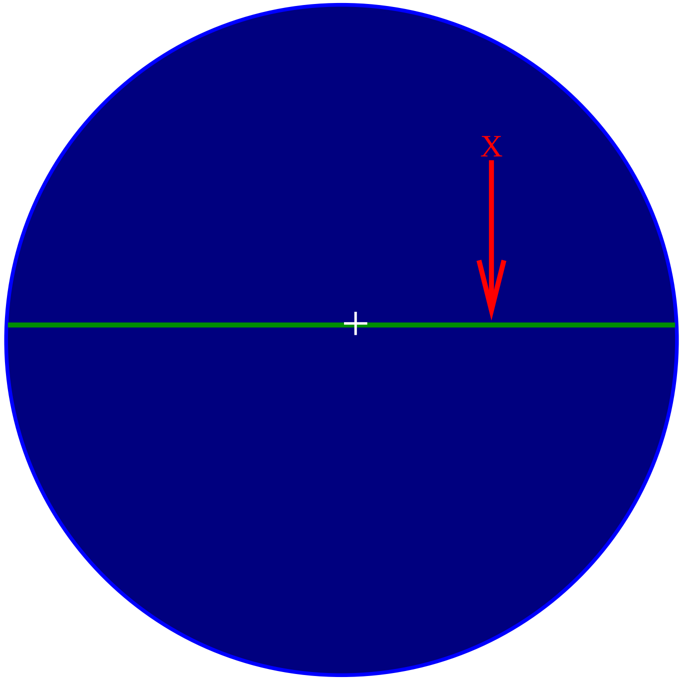
- No, worst case eigenvalue perturbations are within a factor of $\sqrt{2}$.

- Yes, perturbed eigenvalues must lie in a sometimes significantly smaller region.

$\underline{A \in \mathbf{R}^{n\times n}}$



$\{\lambda(A+E) \mid E = E \in \mathbf{C}^{n\times n},\ \|E\| \le \varepsilon\}$ and
$\{\lambda(A+E) \mid E = E \in \mathbf{R}^{n\times n},\ \|E\| \le \varepsilon\}$
ε-real-pseudospectral component containing an isolated, real eigenvalue $\lambda = +$.



$A \in \mathbf{R}^{n \times n}$

Real arithmetic accuracy can be easily recovered for isolated, real eigenvalues: set the imaginary part of λ to zero.

$$A = A_R + iA_I \in \mathbf{C}^{n\times n}$$

A complex matrix is a structured real matrix.

$$A \in \mathbf{C}^{n\times n} \hookrightarrow \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}.$$

These are related to quaternion matrices

$$A \in \mathbf{Q}^{n\times n} \hookrightarrow \begin{bmatrix} A_1 & -\bar{A}_2 \\ A_2 & \bar{A}_1 \end{bmatrix}. \qquad \begin{matrix} A_1 \in \mathbf{C}^{n\times n} \\ A_2 \in \mathbf{C}^{n\times n} \end{matrix}$$

quantum chemistry, for example.

$A = A_R + iA_I \in \mathbf{C}^{n \times n}$

Are structure preserving algorithms for

$$\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}$$

significantly more accurate than others?

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

$$A \in \mathbf{C}^{n\times n} \hookrightarrow \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}.$$

$$x \in \mathbf{C}^{n} \hookrightarrow \left(\begin{bmatrix} x \\ -ix \end{bmatrix}, \begin{bmatrix} \bar{x} \\ i\bar{x} \end{bmatrix} \right)$$

$$\lambda \in \mathbf{C} \hookrightarrow (\lambda, \bar{\lambda})$$

$$\lambda \in \mathbf{R} \hookrightarrow (\lambda, \lambda) \qquad \text{multiple}$$

$$(\lambda, \bar{\lambda}) \in \mathbf{C} \times \mathbf{C} \hookrightarrow (\lambda, \lambda),\ (\bar{\lambda}, \bar{\lambda}),\ \text{multiple}$$

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

The good news about

$$A \in \mathbf{C}^{n\times n} \hookrightarrow \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}$$

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

The good news about

$$A \in \mathbf{C}^{n\times n} \hookrightarrow \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}$$

is that the condition of the eigenvalues is unchanged by the embedding despite increased dimension and multiplicity. This is a consequence of Gerschgorin and special structure.

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

The bad news about

$$A \in \mathbf{C}^{n\times n} \hookrightarrow \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} \cdots$$

$A = A_R + iA_I \in \mathbf{C}^{n \times n}$

The bad news about

$$A \in \mathbf{C}^{n \times n} \hookrightarrow \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} \cdots$$

- Half the eigenvalue-vector pairs of the big matrix are extraneous. May need accurate eigenvectors to distinguish which are which.

$A = A_R + iA_I \in \mathbf{C}^{n \times n}$

The bad news about

$$A \in \mathbf{C}^{n\times n} \hookrightarrow \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} \cdots$$

- Half the eigenvalue-vector pairs of the big matrix are extraneous. May need accurate eigenvectors to distinguish which are which.
- Well conditioned eigenvectors of A may embed into ill-conditioned eigenvectors.

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

MATLAB:

```
AA = [real(A) -imag(A); imag(A) real(A)];
[V, D] = eig(AA);
diag(D)
ans =
    0 + 25i
    0 - 25i
    0 + 25i
    0 - 25i
```

$A = A_R + iA_I \in \mathbf{C}^{n \times n}$

MATLAB:

```
V(:,[1 3])
ans =
  0.7065                    0.0168 - 0.0112i
  0.0292 - 0.0022i         -0.7068
 -0.0022 - 0.2259i         -0.0031 + 0.6739i
  0.0006 - 0.6701i         -0.0107 - 0.2140i
```

Not in $\begin{bmatrix} x \\ ix \end{bmatrix}$ or $\begin{bmatrix} x \\ -ix \end{bmatrix}$ form.

Which eigenvalues are extraneous?

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

Perturbing with $\|E\| < \varepsilon$, $\|F\| < \varepsilon$,

$$\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} + \begin{bmatrix} E_R & -E_I \\ E_I & E_R \end{bmatrix} + \begin{bmatrix} F_R & F_I \\ F_I & -F_R \end{bmatrix},$$

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

Perturbing with $\|E\| < \varepsilon$, $\|F\| < \varepsilon$,

$$\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} + \begin{bmatrix} E_R & -E_I \\ E_I & E_R \end{bmatrix} + \begin{bmatrix} F_R & F_I \\ F_I & -F_R \end{bmatrix},$$

perturbs a desired simple eigenvector to

$$\begin{bmatrix} x \\ -ix \end{bmatrix} + \begin{bmatrix} \hat{X} \\ -i\hat{X} \end{bmatrix} p + \begin{bmatrix} \bar{X} \\ i\bar{X} \end{bmatrix} q$$

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

$$\begin{bmatrix} x \\ -ix \end{bmatrix} + \begin{bmatrix} \hat{X} \\ -i\hat{X} \end{bmatrix} p + \begin{bmatrix} \bar{X} \\ i\bar{X} \end{bmatrix} q + O(\varepsilon^2)$$

$A = A_R + iA_I \in \mathbf{C}^{n \times n}$

$$\begin{bmatrix} x \\ -ix \end{bmatrix} + \begin{bmatrix} \hat{X} \\ -i\hat{X} \end{bmatrix} p + \begin{bmatrix} \bar{X} \\ i\bar{X} \end{bmatrix} q + O(\varepsilon^2)$$

where $p = T^{-1}(\mathcal{P}(E))$, where $\mathcal{P}$ is a projection and T is linear and nonsingular if the corresponding eigenvalue is a simple eigenvalue of A.

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

$$\begin{bmatrix} x \\ -ix \end{bmatrix} + \begin{bmatrix} \hat{X} \\ -i\hat{X} \end{bmatrix} p + \begin{bmatrix} \bar{X} \\ i\bar{X} \end{bmatrix} q + O(\varepsilon^2)$$

. . . and $q = S^{-1}(\mathcal{Q}(F))$, where $\mathcal{Q}$ is a projection and S is linear and nonsingular if the corresponding eigenvalue is a simple eigenvalue of $\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}$.

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

- Possibly, $\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}$ has multiple or nearly multiple eigenvalues when $A = A_R + iA_I$ does not, e.g., when A has a real eigenvalue or a complex conjugate pair.

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

- Possibly, $\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}$ has multiple or nearly multiple eigenvalues when $A = A_R + iA_I$ does not, e.g., when A has a real eigenvalue or a complex conjugate pair.
- Consequently, possibly, $\|q\|_2 \gg \|p\|_2$, i.e., an unstructured algorithm may be numerically unstable , but. . .

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

A simple *a posteriori* correction,

$$\begin{bmatrix} y \\ z \end{bmatrix} \in \mathbf{C}^n \times \mathbf{C}^n \longrightarrow \begin{bmatrix} y + iz \\ -i(y + iz) \end{bmatrix}$$

reduces F and q to second order perturbations.

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

MATLAB:

```
W=[V(1:2,:)+i*V(3:4,:)  ; -i*V(1:2,:)+V(3:4,:)];
W(:,[1 2])
ans =
   0.9324 - 0.0022i   0.4806 - 0.0022i
   0.6993 - 0.0016i  -0.6409 + 0.0029i
  -0.0022 - 0.9324i  -0.0022 - 0.4806i
  -0.0016 - 0.6993i   0.0029 + 0.6409i
eigval = diag(W'*AA*W)./diag(W'*W)
ans =
    0.0000 +25.0000i
    0.0000 -25.0000i
```

$A = A_R + iA_I \in \mathbf{C}^{n \times n}$

Are structure preserving algorithms significantly more accurate than others?

$A = A_R + iA_I \in \mathbf{C}^{n\times n}$

Are structure preserving algorithms significantly more accurate than others?

- Yes, the eigenvector conditioning under structured perturbations may be significantly smaller.

$A = A_R + iA_I \in \mathbf{C}^{n \times n}$

Are structure preserving algorithms significantly more accurate than others?

- Yes, the eigenvector conditioning under structured perturbations may be significantly smaller.
- No, eigenvalue conditioning is unchanged. There is a simple *a posteriori* fix for eigenvectors.

Introduction to BiStructure

In the previous examples there are two structures. The data has a structure and the desired computed results also have a structure.

Introduction to BiStructure

In the previous examples there are two structures. The data has a structure and the desired computed results also have a structure.

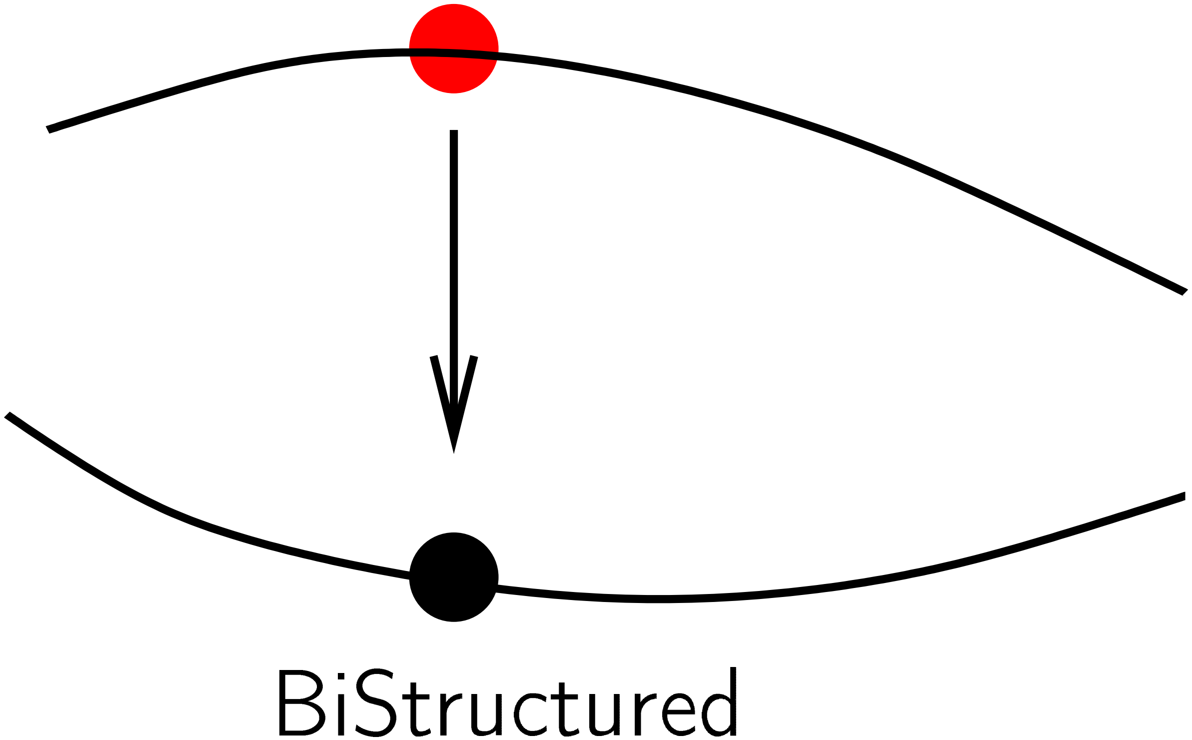
- Compute a real eigenvalue of a real matrix: $A \in \mathbf{R}^{n \times n} \subset \mathbf{C}^{n \times n} \longrightarrow \lambda \in \mathbf{R}$.

Introduction to BiStructure

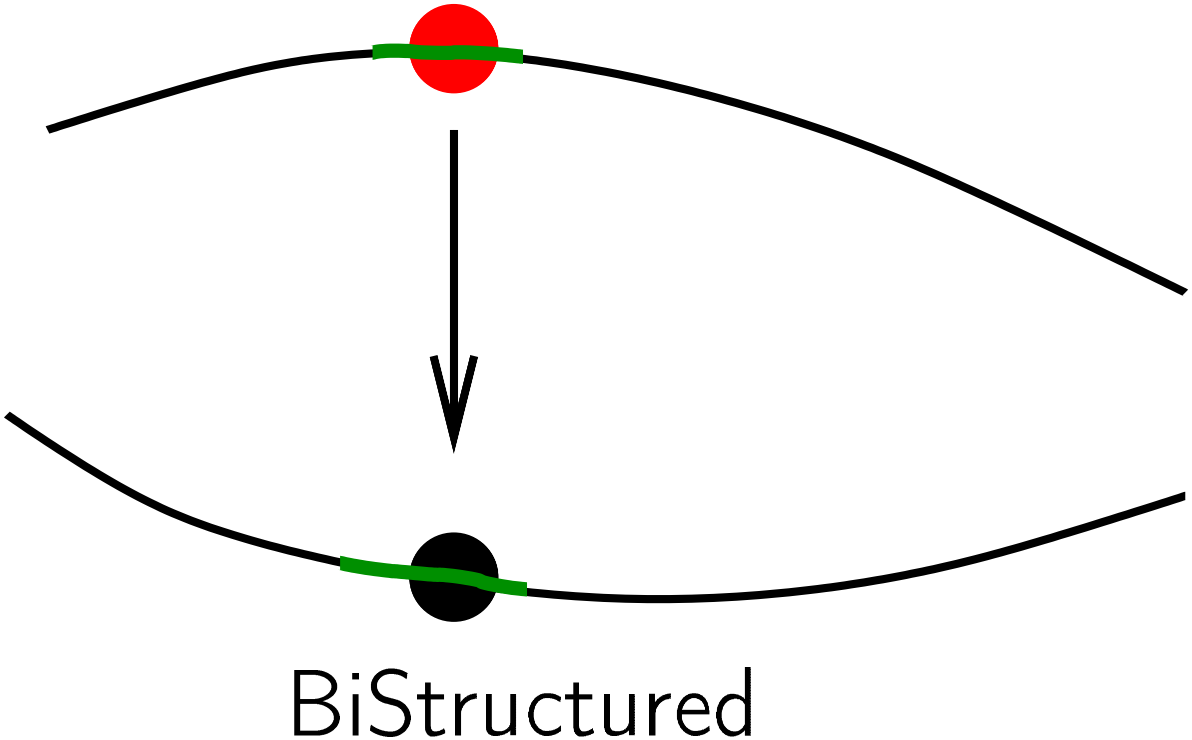
In the previous examples there are two structures. The data has a structure and the desired computed results also have a structure.

- Compute a real eigenvalue of a real matrix: $A \in \mathbf{R}^{n \times n} \subset \mathbf{C}^{n \times n} \rightarrow \lambda \in \mathbf{R}$.
- Compute eigenvalues and vectors of the structured matrix $\begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix}$:
$\left\{ \begin{bmatrix} A_R & -A_I \\ A_I & A_R \end{bmatrix} \right\} \rightarrow \lambda \in \mathbf{C}$ with $\begin{bmatrix} x \\ -ix \end{bmatrix} \in \mathbf{C}^{2n}$.

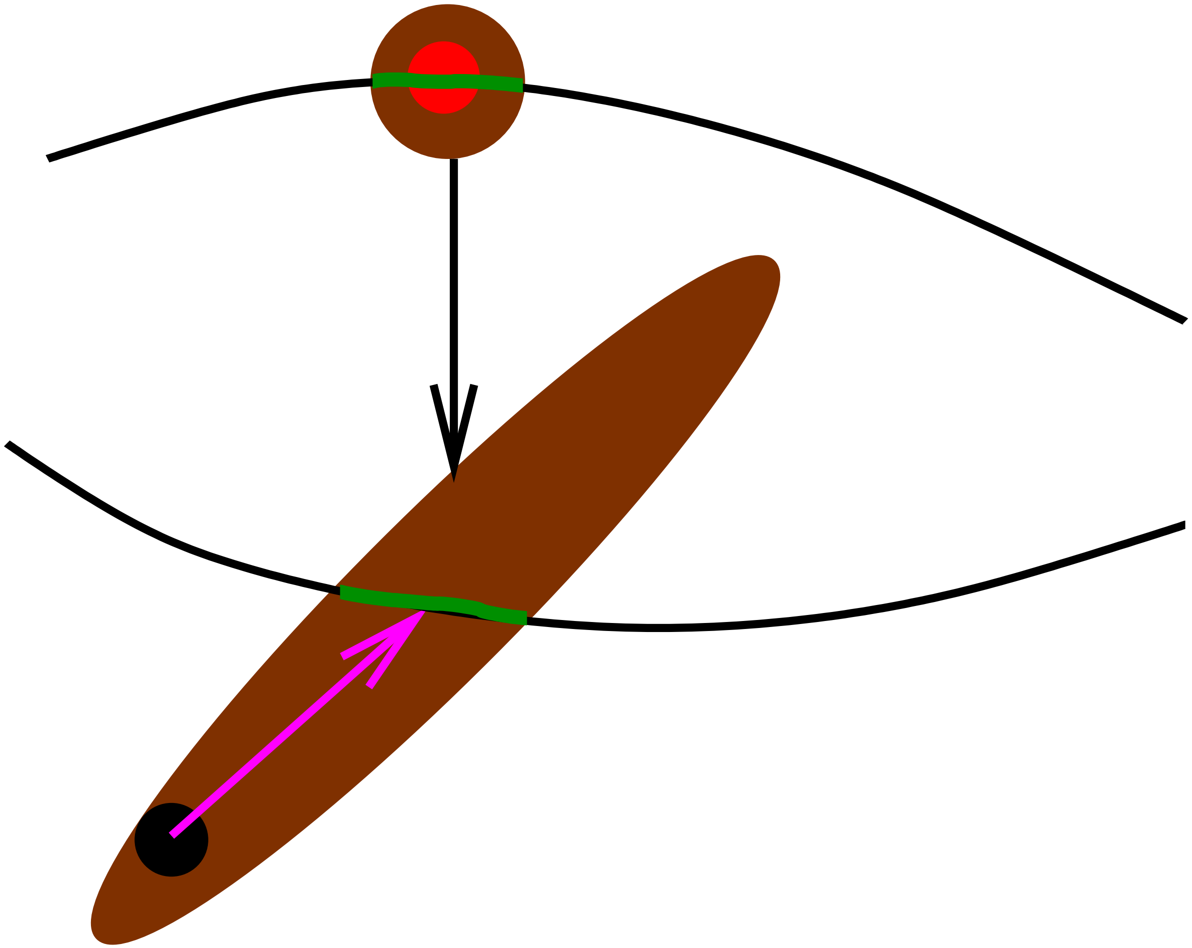
BiStructure Introduction



BiStructure Introduction

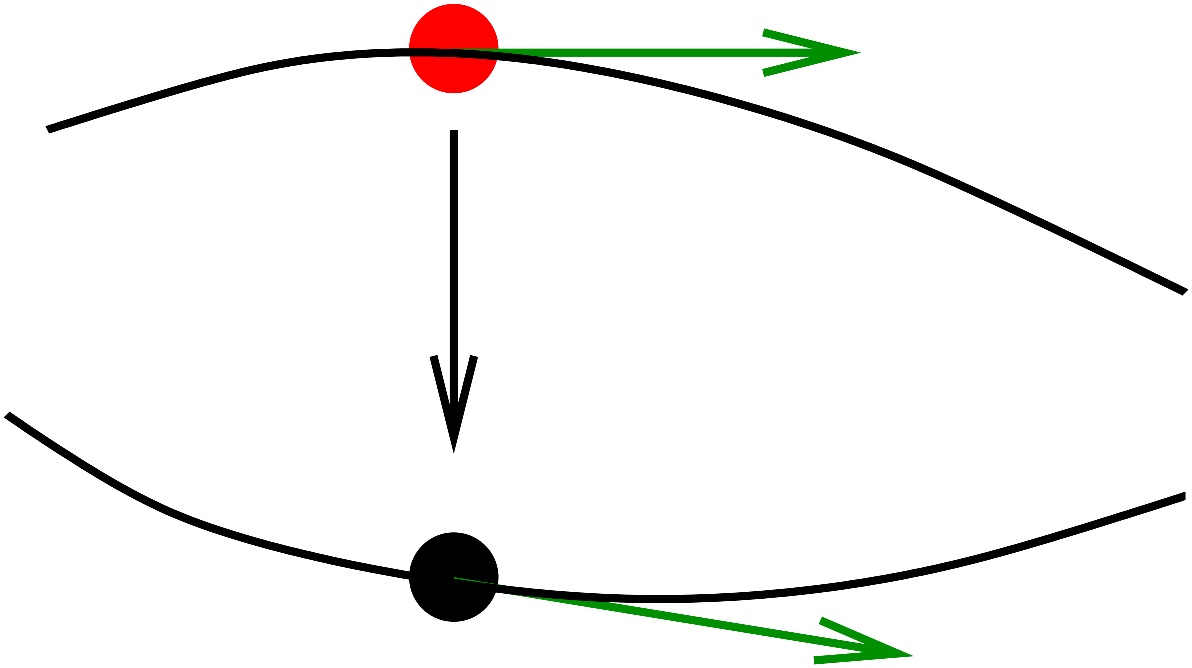


BiStructure Introduction



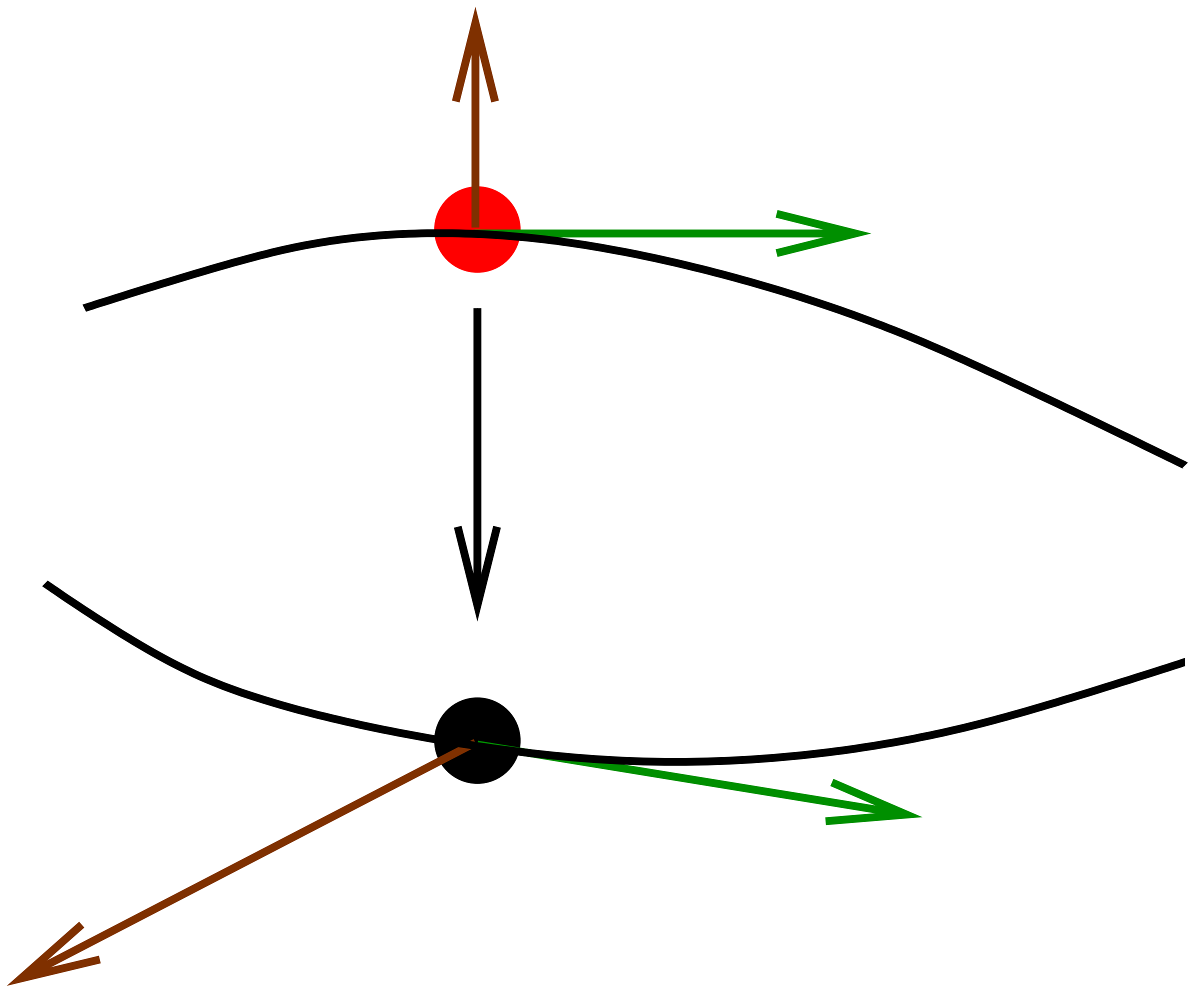
(Skew)-Projection Correction

BiStructure Introduction



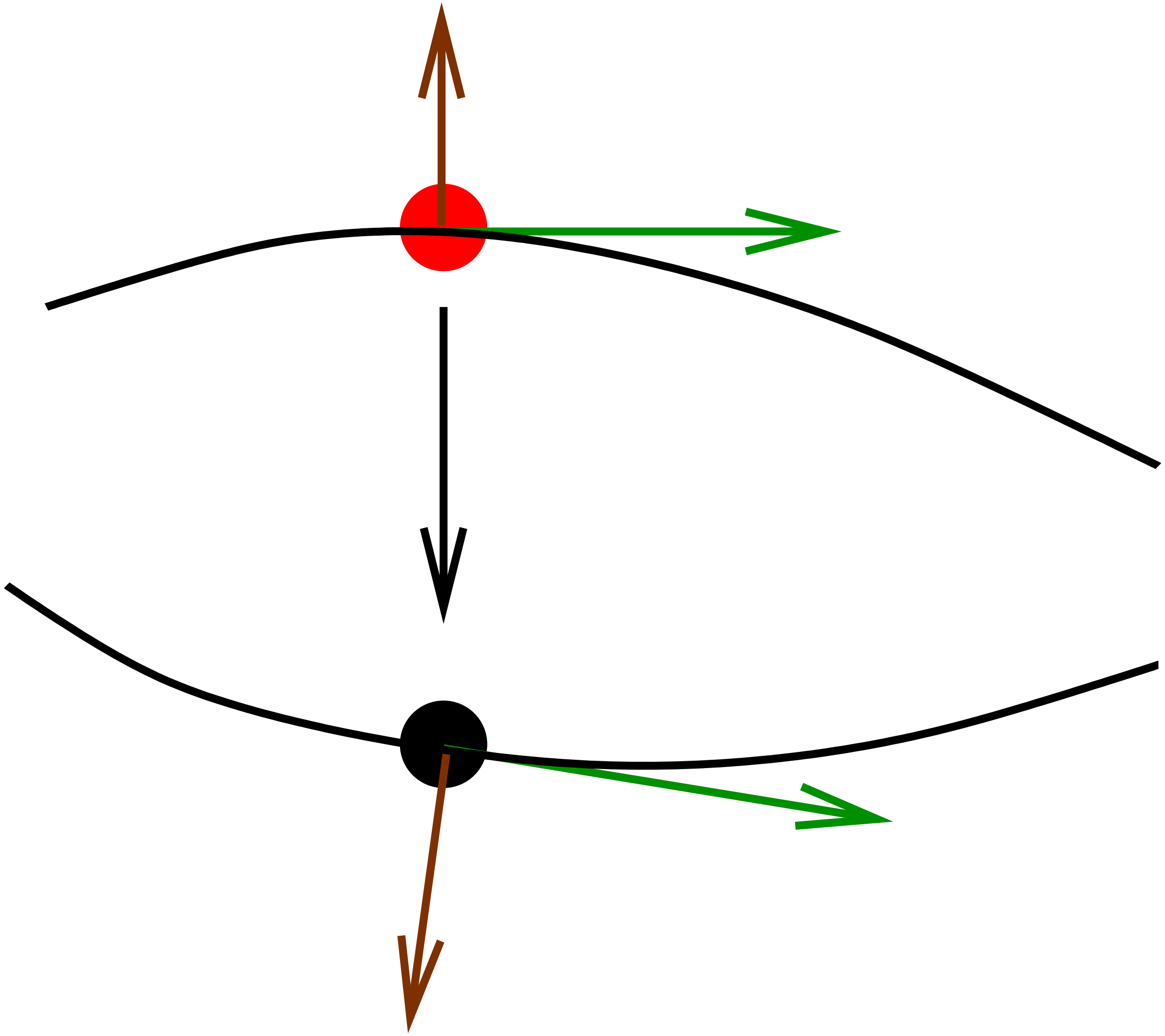
Tangent space $\longrightarrow$ tangent space

BiStructure Introduction



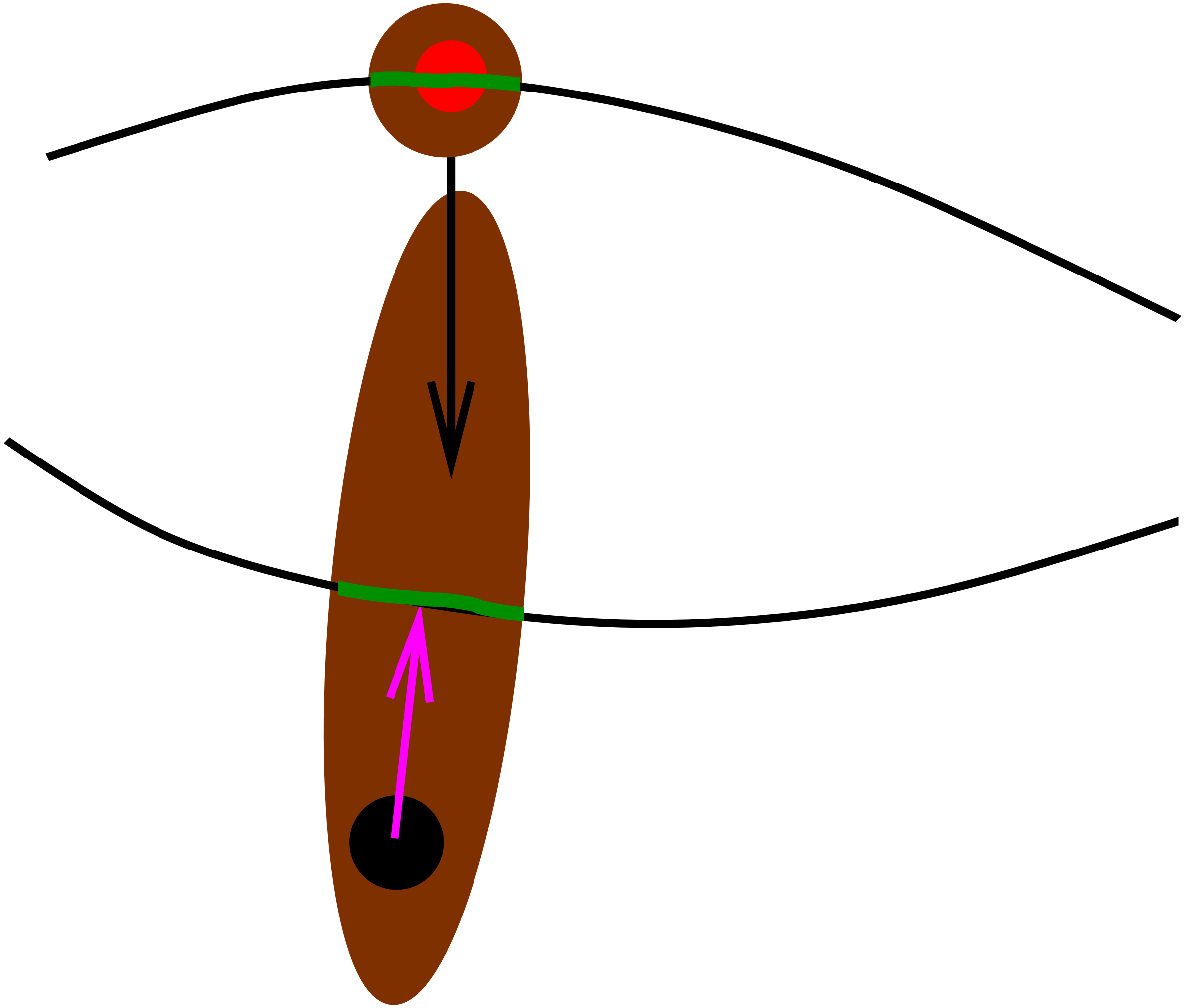
Not Orthogonaly Bistructured

BiStructure Introduction



Orthogonally Bistructured

BiStructure Introduction



Orthogonaly Bistructured → Simple correction

Block Hamiltonian-Schur Form

- $J = \begin{bmatrix} 0 & I \\ -I & 0 \end{bmatrix}$
- $H \in \mathbf{R}^{2n\times 2n}$ is Hamiltonian if $(JH)^T = JH$, i.e.,

$$H = \begin{bmatrix} A & G \\ F & -A^T \end{bmatrix}, \quad F = F^T, \quad G = G^T.$$

- $H \in \mathbf{R}^{2n\times 2n}$ is skew-Hamiltonian if $(JH)^T = -JH$, i.e.,

$$H = \begin{bmatrix} A & G \\ F & A^T \end{bmatrix}, \quad F = -F^T, \quad G = -G^T.$$

- $S \in \mathbf{R}^{2n\times 2n}$ is Symplectic if $S^T JS = J$.
- $V \in \mathbf{R}^{2n\times n}$ is Lagranian if $V^T JV = 0$.

Block Hamiltonian-Schur Form

Given a Hamiltonian matrix H, calculate S and T such that

$$HS = ST = \begin{bmatrix} S_1 & -S_2 \\ S_2 & S_1 \end{bmatrix} \begin{bmatrix} T_1 & T_2 \\ 0 & -T_1^T \end{bmatrix}$$

where

- T is Hamiltonian
- S is symplectic and orthogonal
- $\sigma(T_1) \subset \mathbf{C}_-$, i.e., eigenvalues of T_1 have non-positive real part.

Block Hamiltonian-Schur Form

- Progress on a structure preserving algorithm for Hamiltonian-Schur Form by Ammar, Benner, Bunse-Gerstner, B, Chu, Faßbender, He, Golub, Laub, Mehrmann, Paige, Van Loan, Xu, . . .
- Note particularly work by Mehrmann and Chu 2005 or 2006.

Laub Trick

Laub trick: (Laub circa 1985)

1. Calculate unstructured Schur form

$$HQ = QR = \begin{bmatrix} Q_1 & Q_3 \\ Q_2 & Q_4 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}.$$

   where $\sigma(R_{11}) \subset \mathbf{C}_-$.

2. $S = \begin{bmatrix} Q_1 & -Q_2 \\ Q_2 & Q_1 \end{bmatrix}$

In exact arithmetic $\begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}$ is Lagrangian and S is the symplectic-orthogonal factor of Hamiltonian-Schur Form.

Laub Trick

Problem:

Laub Trick

Problem: Not numericaly stable.

Laub Trick

Problem: Not numericaly stable.

As the stable invariant subspace span $\begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}$ becomes ill-conditioned, rounding errors destroy Symplectic structure—just when the extra numerical stability of a structure preserving algorithm is most important.

Laub Trick

Problem: Not numericaly stable.

As the stable invariant subspace span $\begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}$ becomes ill-conditioned, rounding errors destroy Symplectic structure—just when the extra numerical stability of a structure preserving algorithm is most important.

Gram-Schmidt-like Symplectic-reorthogonalization damages the invariant subspace.

A Bistructured Problem

Computational Problem:

$$F(H) = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}$$

(So, $S = \begin{bmatrix} Q_1 & -Q_2 \\ Q_2 & Q_1 \end{bmatrix}$ and $T = S^T H S$.)

A Bistructured Problem

Computational Problem:

$$F(H) = \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}$$

(So, $S = \begin{bmatrix} Q_1 & -Q_2 \\ Q_2 & Q_1 \end{bmatrix}$ and $T = S^T H S$.)

Domain

Structure: H Hamiltonian, i.e., $(JH)^T = JH$

Tangent space: $T_H = \{ M \mid (JM)^T = JM \}$

Normal space: $T_H^\perp = \{ M \mid (JM)^T = -JM \}$

A Bistructured Problem

Range

Structure: $$\left\{ Q \in \mathbf{R}^{2n\times n} \;\middle|\; \begin{matrix} Q^T J Q = 0 \\ Q^T Q = I \end{matrix} \right\}$$

Tangent space: $$T_Q = \left\{ \begin{bmatrix} Q_1 & -Q_2 \\ Q_2 & Q_1 \end{bmatrix} \begin{bmatrix} Z \\ W \end{bmatrix} \;\middle|\; \begin{matrix} Z^T = - \\ W^T = \end{matrix} \right.$$

Normal space: $$T_Q^{\perp} = \left\{ \begin{bmatrix} Q_1 & -Q_2 \\ Q_2 & Q_1 \end{bmatrix} \begin{bmatrix} W \\ Z \end{bmatrix} \;\middle|\; \begin{matrix} Z^T = - \\ W^T = \end{matrix} \right.$$

A Bistructured Problem

$$F(H) \;=\; \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix}$$

$$DF(H) \;:\; T_H \longrightarrow T_Q$$

$$DF(H) \;:\; T_H^{\perp} \longrightarrow T_Q^{\perp}$$

Recover structured algorithm accuracy (to first order) by projecting computed solution back onto T_Q

Corrected Laub Trick

1. Calculate unstructured Schur form

$$HQ = QR = \begin{bmatrix} Q_1 & Q_3 \\ Q_2 & Q_4 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ 0 & R_{22} \end{bmatrix}.$$

   where $\sigma(R_{11}) \subset \mathbf{C}_-$.

2. $Z = \frac{1}{2}\left(Q_1^T Q_2 - Q_2^T Q_1\right)$
3. $\begin{bmatrix} V_1 \\ V_2 \end{bmatrix} \rightarrow \begin{bmatrix} Q_1 \\ Q_2 \end{bmatrix} + \begin{bmatrix} -Q_2 \\ Q_1 \end{bmatrix} Z$
4. $S = \begin{bmatrix} V_1 & -V_2 \\ V_2 & V_1 \end{bmatrix}$

(No need to correct orthogonality.)

$A \in \mathcal{M}$, $F(A) \in \mathcal{F}$

Rounding errors in a backward stable numerical method are equivalent to evaluating $F(A+E)$ for some small magnitude perturbation E.

$A \in \mathcal{M}, \; F(A) \in \mathcal{F}$

Rounding errors in a backward stable numerical method are equivalent to evaluating $F(A+E)$ for some small magnitude perturbation E.

- For a structured algorithm, $A + E \in \mathcal{M}$.
- Otherwise, possibly, $A + E \notin \mathcal{M}$.

$A \in \mathcal{M}$, $F(A) \in \mathcal{F}$

Rounding errors in a backward stable numerical method are equivalent to evaluating $F(A+E)$ for some small magnitude perturbation E.

- For a structured algorithm, $A + E \in \mathcal{M}$.
- Otherwise, possibly, $A + E \notin \mathcal{M}$.

To first order, perturbing A to $A + E$ gives

$$F(A+E) = F(A) + \boxed{D_A(E)} + O(\|E\|^2).$$

$A \in \mathcal{M}, \; F(A) \in \mathcal{F}$

For BiStructured problems, in appropriate orthonormal bases of $\mathcal{M}$ and $\mathcal{F}$,

$$D_A \;=\; \begin{bmatrix} D_{\mathcal{M}} & D_{\mathcal{N}} \\ 0 & D_{\mathcal{M}^\perp} \end{bmatrix}$$

$$\sim \;\; \begin{bmatrix} T_A \rightarrow T_{F(A)} & T_A^\perp \rightarrow T_{F(A)} \\ T_A \rightarrow T_{F(A)}^\perp & T_A^\perp \rightarrow T_{F(A)}^\perp \end{bmatrix}$$

(orthogonally BiStructured $\Longleftrightarrow D_{\mathcal{N}} = 0$)

Absolute Condition Numbers

$$\kappa_{\text{str}} = \|D_{\mathcal{M}}\|_2$$

$$\kappa_{\text{abs}} = \|D_{A}\|_2 = \left\| \begin{bmatrix} D_{\mathcal{M}} & D_{\mathcal{N}} \\ 0 & D_{\mathcal{M}^\perp} \end{bmatrix} \right\|_2$$

Absolute Condition Numbers

$$\kappa_{\text{str}} = \|D_{\mathcal{M}}\|_2$$

$$\kappa_{\text{abs}} = \|D_A\|_2 = \left\| \begin{bmatrix} D_{\mathcal{M}} & D_{\mathcal{N}} \\ 0 & D_{\mathcal{M}^\perp} \end{bmatrix} \right\|_2$$

- If $\kappa_{\text{str}} \ll \kappa_{\text{abs}}$, then structured algorithms are probably more accurate.
- If D_A is well-scaled, then $\kappa_{\text{str}} \approx \kappa_{\text{abs}}$.

$A \in \mathcal{M}$, $F(A) \in \mathcal{F}$

$$F(A+E) = \begin{bmatrix} F_{\mathcal{F}} \\ F_{\mathcal{M}}^{\perp} \end{bmatrix} \sim \begin{bmatrix} \mathcal{F} \\ \mathcal{F}^{\perp} \end{bmatrix}$$

$$E = \begin{bmatrix} E_{\mathcal{M}} \\ E_{\mathcal{M}^{\perp}} \end{bmatrix} \sim \begin{bmatrix} T_A \\ T_{A^{\perp}} \end{bmatrix}$$

If $D_{\mathcal{M}^{\perp}} C = F_{\mathcal{M}^{\perp}} + O(||E||^2)$, then

$$\begin{bmatrix} F_{\mathcal{M}} - D_{\mathcal{N}} C \\ 0 \end{bmatrix} = F(A + E_{\mathcal{M}}) + O(||E||^2).$$

In orthogonaly BiStructured problems, $D_{\mathcal{N}} = 0$,

so correction is simple.

Linearizations

- Quadratic matrix polynomial:
  $Q(\lambda) = \lambda^2 A + \lambda B + C$.

Linearizations

- Quadratic matrix polynomial:
  $Q(\lambda) = \lambda^2 A + \lambda B + C$.
- $L_1(\lambda) = \lambda \begin{bmatrix} A & 0 \\ 0 & -C \end{bmatrix} + \begin{bmatrix} B & C \\ C & 0 \end{bmatrix}$

Linearizations

- Quadratic matrix polynomial:
  $Q(\lambda) = \lambda^2 A + \lambda B + C$.
- $L_1(\lambda) = \lambda \begin{bmatrix} A & 0 \\ 0 & -C \end{bmatrix} + \begin{bmatrix} B & C \\ C & 0 \end{bmatrix}$
- BiStructured with co-structure $(\lambda, \begin{bmatrix} \lambda x \\ x \end{bmatrix})$,
  $x \in \mathbf{C}^n$.

Linearizations

- Quadratic matrix polynomial:
  $Q(\lambda) = \lambda^2 A + \lambda B + C$.
- $L_1(\lambda) = \lambda \begin{bmatrix} A & 0 \\ 0 & -C \end{bmatrix} + \begin{bmatrix} B & C \\ C & 0 \end{bmatrix}$
- BiStructured with co-structure $(\lambda, \begin{bmatrix} \lambda x \\ x \end{bmatrix})$,
  $x \in \mathbf{C}^n$.
- Sometimes $\kappa_{\mathsf{str}} \ll \kappa_{\mathsf{abs}}$

Linearizations

- Quadratic matrix polynomial: $Q(\lambda) = \lambda^2 A + \lambda B + C$.
- $L_1(\lambda) = \lambda \begin{bmatrix} A & 0 \\ 0 & -C \end{bmatrix} + \begin{bmatrix} B & C \\ C & 0 \end{bmatrix}$
- BiStructured with co-structure $(\lambda, \begin{bmatrix} \lambda x \\ x \end{bmatrix})$, $x \in \mathbf{C}^n$.
- Sometimes $\kappa_{\mathsf{str}} \ll \kappa_{\mathsf{abs}}$
- Not orthogonaly BiStructured.

algebraic Riccati Equation

- $F(Q, A_L, A_R, G) = X$ where
  $Q + A_L X + X A_R - XGX = 0.$

algebraic Riccati Equation

- $F(Q, A_L, A_R, G) = X$ where
  $Q + A_L X + X A_R - X G X = 0.$

- $\mathcal{M} = \{G = G^T,\ F = F^T,\ A_L = A_R^T\}$,
  $\mathcal{F} = \{X \in \mathbf{R}^{n\times n} \mid X = X^T\}$

algebraic Riccati Equation

- $F(Q, A_L, A_R, G) = X$ where
  $Q + A_L X + X A_R - X G X = 0.$

- $\mathcal{M} = \{G = G^T,\ F = F^T,\ A_L = A_R^T\}$,
  $\mathcal{F} = \{X \in \mathbf{R}^{n \times n} \mid X = X^T\}$

- Orthogonally BiStructured

algebraic Riccati Equation

- $F(Q, A_L, A_R, G) = X$ where
  $Q + A_L X + X A_R - X G X = 0.$

- $\mathcal{M} = \{G = G^T,\ F = F^T,\ A_L = A_R^T\}$,
  $\mathcal{F} = \{X \in \mathbf{R}^{n \times n} \mid X = X^T\}$

- Orthogonally BiStructured

- $\kappa_{\text{str}} \approx \kappa_{\text{abs}}$ follows from B, Nash (1987).

algebraic Riccati Equation

- $F(Q, A_L, A_R, G) = X$ where
  $Q + A_L X + X A_R - X G X = 0.$
- $\mathcal{M} = \{G = G^T,\ F = F^T,\ A_L = A_R^T\}$,
  $\mathcal{F} = \{X \in \mathbf{R}^{n\times n} \mid X = X^T\}$
- Orthogonally BiStructured
- $\kappa_{\text{str}} \approx \kappa_{\text{abs}}$ follows from B, Nash (1987).
- *a posteriori* fix: $\hat{X} \leftarrow (\tilde{X} + \tilde{X}^T)/2$

Conclusions

Are structured algorithms significantly more accurate than unstructured algorithms?

Conclusions

Are structured algorithms significantly more accurate than unstructured algorithms?

- Yes, qualitative features such as symmetries and pairings are preserved.

Conclusions

Are structured algorithms significantly more accurate than unstructured algorithms?

- Yes, qualitative features such as symmetries and pairings are preserved.
- No, not always. It is not unusual that structured algorithm rounding errors may be as (nearly) great as unstructured algorithm errors, but…

Conclusions

Are structured algorithms significantly more accurate than unstructured algorithms?

- Yes, qualitative features such as symmetries and pairings are preserved.
- No, not always. It is not unusual that structured algorithm rounding errors may be as (nearly) great as unstructured algorithm errors, but. . .
- Orthogonally BiStructured problems admit an easy *a posteriori* correction to filter out unstructured errors (to first order).