On Jacobi Methods for the Positive Definite Generalized Eigenvalue Problem

Vjeran Hari

Department of Mathematics, Faculty of Science, University of Zagreb
hari@math.hr

Department of Mathematics and Statistics, University of Missouri at Kansas City, USA

OUTLINE

OUTLINE

- GEP and PGEP

OUTLINE

- GEP and PGEP
- Derivation of the algorithms

This work has been fully supported by Croatian Science Foundation under the project IP-09-2014-3670.

OUTLINE

- GEP and PGEP
- Derivation of the algorithms
- Convergence, global and asymptotic

This work has been fully supported by Croatian Science Foundation under the project IP-09-2014-3670.

OUTLINE

- GEP and PGEP
- Derivation of the algorithms
- Convergence, global and asymptotic
- Stability and relative accuracy

This work has been fully supported by Croatian Science Foundation under the project IP-09-2014-3670.

OUTLINE

- GEP and PGEP
- Derivation of the algorithms
- Convergence, global and asymptotic
- Stability and relative accuracy
- Block algorithms

This work has been fully supported by Croatian Science Foundation under the project IP-09-2014-3670.

OUTLINE

- GEP and PGEP
- Derivation of the algorithms
- Convergence, global and asymptotic
- Stability and relative accuracy
- Block algorithms
- Global convergence of block algorithms

This work has been fully supported by Croatian Science Foundation under the project IP-09-2014-3670.

OUTLINE

- GEP and PGEP
- Derivation of the algorithms
- Convergence, global and asymptotic
- Stability and relative accuracy
- Block algorithms
- Global convergence of block algorithms

- We are considering element-wise, two-sided Jacobi-type methods for PGEP which can be used as kernel algorithms for the block methods.

This work has been fully supported by Croatian Science Foundation under the project IP-09-2014-3670.

GEP and PGEP

Let $A = A^T, \quad B = B^T.$

GEP and PGEP

Let $A = A^T, \quad B = B^T$.
We consider the Generalized Eigenvalue Problem (GEP)

$$Ax = \lambda Bx, \quad x \neq 0.$$

If $B \succ O$, GEP is usually called Positive definite GEP or shorter PGEP.

GEP and PGEP

Let $A = A^T, \quad B = B^T$.
We consider the Generalized Eigenvalue Problem (GEP)

$$Ax = \lambda Bx, \quad x \neq 0.$$

If $B \succ O$, GEP is usually called Positive definite GEP or shorter PGEP.

For such a pair there is a nonsingular matrix F such that

$$F^T AF = \Lambda_A\,, \qquad F^T BF = \Lambda_B,$$

$\Lambda_A = \mathrm{diag}(\alpha_1, \ldots, \alpha_n), \quad \Lambda_B = \mathrm{diag}(\beta_1, \ldots, \beta_n) \succ O.$

GEP and PGEP

Let $A = A^T, \quad B = B^T$.
We consider the Generalized Eigenvalue Problem (GEP)

$$Ax = \lambda Bx, \quad x \neq 0.$$

If $B \succ O$, GEP is usually called Positive definite GEP or shorter PGEP.

For such a pair there is a nonsingular matrix F such that

$$F^T AF = \Lambda_A\,, \qquad F^T BF = \Lambda_B,$$

$\Lambda_A = \mathrm{diag}(\alpha_1, \ldots, \alpha_n), \quad \Lambda_B = \mathrm{diag}(\beta_1, \ldots, \beta_n) \succ O.$

The eigenpairs of (A, B) are: $(\alpha_i/\beta_i, Fe_i)$, $1 \leq i \leq n$; $\ I_n = [e_1, \ldots, e_n]$.

Little Proof

$$
\begin{aligned}
F^T A F = \Lambda_A \quad &\Rightarrow \quad AF = F^{-T}\Lambda_A, \\
F^T B F = \Lambda_B \quad &\Rightarrow \quad BF = F^{-T}\Lambda_B.
\end{aligned}
$$

Little Proof

$$
\begin{aligned}
F^T A F = \Lambda_A \quad &\Rightarrow \quad AF = F^{-T}\Lambda_A, \\
F^T B F = \Lambda_B \quad &\Rightarrow \quad BF = F^{-T}\Lambda_B.
\end{aligned}
$$

$$F^{-T}\Lambda_A = F^{-T}\Lambda_B(\Lambda_A\Lambda_B^{-1}) = BF(\Lambda_A\Lambda_B^{-1}),$$

Little Proof

$$F^T A F = \Lambda_A \quad \Rightarrow \quad AF = F^{-T}\Lambda_A,$$
$$F^T B F = \Lambda_B \quad \Rightarrow \quad BF = F^{-T}\Lambda_B.$$

$$F^{-T}\Lambda_A = F^{-T}\Lambda_B(\Lambda_A\Lambda_B^{-1}) = BF(\Lambda_A\Lambda_B^{-1}),$$

$$AF = F^{-T}\Lambda_A = BF(\Lambda_A\Lambda_B^{-1}) = BF\,\mathrm{diag}(\alpha_1/\beta_1,\ldots,\alpha_n/\beta_n),$$

Little Proof

$$
\begin{aligned}
F^T A F = \Lambda_A \quad &\Rightarrow \quad AF = F^{-T}\Lambda_A, \\
F^T B F = \Lambda_B \quad &\Rightarrow \quad BF = F^{-T}\Lambda_B.
\end{aligned}
$$

$$F^{-T}\Lambda_A = F^{-T}\Lambda_B(\Lambda_A\Lambda_B^{-1}) = BF(\Lambda_A\Lambda_B^{-1}),$$

$$AF = F^{-T}\Lambda_A = BF(\Lambda_A\Lambda_B^{-1}) = BF\,\mathrm{diag}(\alpha_1/\beta_1, \ldots, \alpha_n/\beta_n),$$

$$A\,Fe_i = BF\,\mathrm{diag}(\alpha_1/\beta_1, \ldots, \alpha_n/\beta_n)e_i = (\alpha_i/\beta_i)\,B\,Fe_i, \quad 1 \le i \le n.$$

$\square$

How to Solve PGEP?

One can try with the transformation $(A, B) \mapsto (L^{-1}AL^{-T}, I)$, $B = LL^T$ and reduce PGEP to the standard EP for one symmetric matrix.
If B has very high condition, then L will have high condition

How to Solve PGEP?

One can try with the transformation $(A, B) \mapsto (L^{-1}AL^{-T}, I)$, $B = LL^T$ and reduce PGEP to the standard EP for one symmetric matrix.
If B has very high condition, then L will have high condition

$$(\quad \text{recall: } \kappa_2(L) = \sqrt{\kappa_2(B)} \quad),$$

How to Solve PGEP?

One can try with the transformation $(A, B) \mapsto (L^{-1}AL^{-T}, I)$, $B = LL^T$ and reduce PGEP to the standard EP for one symmetric matrix.
If B has very high condition, then L will have high condition

$$(\text{ recall: } \kappa_2(L) = \sqrt{\kappa_2(B)} \ ),$$

then the computed matrix $L^{-1}AL^{-T}$ will have very high condition, so inaccuracy in L plus high condition of L will imply that the eigenvalues of $L^{-1}AL^{-T}$ are corrupt.

How to Solve PGEP?

One can try with the transformation $(A, B) \mapsto (L^{-1}AL^{-T}, I)$, $B = LL^T$ and reduce PGEP to the standard EP for one symmetric matrix.
If B has very high condition, then L will have high condition

$$\left(\text{ recall: } \kappa_2(L) = \sqrt{\kappa_2(B)} \ \right),$$

then the computed matrix $L^{-1}AL^{-T}$ will have very high condition, so inaccuracy in L plus high condition of L will imply that the eigenvalues of $L^{-1}AL^{-T}$ are corrupt.

Then one can try to maximize the minimum eigenvalue of B by rotating the pair

$$(A, B) \mapsto (A_\varphi, B_\varphi) = (A\cos\varphi + B\sin\varphi, -A\sin\varphi + B\cos\varphi),$$

or derive a method which works with the initial pair (A, B).

How to Solve PGEP?

One can try with the transformation $(A, B) \mapsto (L^{-1}AL^{-T}, I)$, $B = LL^T$ and reduce PGEP to the standard EP for one symmetric matrix.
If B has very high condition, then L will have high condition

$$(\text{ recall: } \kappa_2(L) = \sqrt{\kappa_2(B)} \ ),$$

then the computed matrix $L^{-1}AL^{-T}$ will have very high condition, so inaccuracy in L plus high condition of L will imply that the eigenvalues of $L^{-1}AL^{-T}$ are corrupt.

Then one can try to maximize the minimum eigenvalue of B by rotating the pair

$$(A, B) \mapsto (A_\varphi, B_\varphi) = (A\cos\varphi + B\sin\varphi, -A\sin\varphi + B\cos\varphi),$$

or derive a method which works with the initial pair (A, B).

We follow the second path.

Jacobi methods for PGEP

We have two diagonalization methods for PGEP

Jacobi methods for PGEP

We have two diagonalization methods for PGEP

- Falk-Langemeyer method (shorter: FL method)
  (Elektronische Datenverarbeitung, 1960)

Jacobi methods for PGEP

We have two diagonalization methods for PGEP

- Falk-Langemeyer method (shorter: FL method)
  (Elektronische Datenverarbeitung, 1960)
- Hari-Zimmermann variant of the FL method (shorter: HZ method)
  (Hari Ph.D. 1984)

Jacobi methods for PGEP

We have two diagonalization methods for PGEP

- Falk-Langemeyer method (shorter: FL method)
  (Elektronische Datenverarbeitung, 1960)
- Hari-Zimmermann variant of the FL method (shorter: HZ method)
  (Hari Ph.D. 1984)

The two methods are connected: the FL method can be viewed as the HZ method with “fast scaled” transformations. So, the FL method seems to be somewhat faster and the HZ method seems to be more robust.

Jacobi methods for PGEP

However, numerical tests on large matrices, on parallel machines, have confirmed the advantage of the HZ approach. In the paper

V. Novaković, S. Singer, S. Singer (Parallel Comput., 2015)

However, numerical tests on large matrices, on parallel machines, have confirmed the advantage of the HZ approach. In the paper

V. Novaković, S. Singer, S. Singer (Parallel Comput., 2015)

it has been shown/written:

Jacobi methods for PGEP

However, numerical tests on large matrices, on parallel machines, have confirmed the advantage of the HZ approach. In the paper

V. Novaković, S. Singer, S. Singer (Parallel Comput., 2015)

it has been shown/written:

> *When implemented as one-sided block algorithm for the GSVD, it is almost perfectly parallelizable, so parallel shared memory versions of the algorithm are highly scalable, and their speedup almost solely depends on the number of cores used.*

Jacobi methods for PGEP

However, numerical tests on large matrices, on parallel machines, have confirmed the advantage of the HZ approach. In the paper

V. Novaković, S. Singer, S. Singer (Parallel Comput., 2015)

it has been shown/written:

> *When implemented as one-sided block algorithm for the GSVD, it is almost perfectly parallelizable, so parallel shared memory versions of the algorithm are highly scalable, and their speedup almost solely depends on the number of cores used.*

(In this paper the method was first time referred to as the HZ method!)

Derivation of the HZ Method

Preliminary transformation: $\quad A^{(0)} = D_0 A D_0, \;\; B^{(0)} = D_0 B D_0$

Derivation of the HZ Method

Preliminary transformation: $A^{(0)} = D_0 A D_0, \ B^{(0)} = D_0 B D_0$

$D_0 = [\mathrm{diag}(B)]^{-\frac{1}{2}}$, so that $b_{11}^{(0)} = b_{22}^{(0)} = \cdots = b_{nn}^{(0)} = 1.$

Derivation of the HZ Method

Preliminary transformation: $A^{(0)} = D_0 A D_0, \; B^{(0)} = D_0 B D_0$

$D_0 = [\mathrm{diag}(B)]^{-\frac{1}{2}}$, so that $b_{11}^{(0)} = b_{22}^{(0)} = \cdots = b_{nn}^{(0)} = 1.$

This property of $B^{(0)}$ is maintained during the iteration process:

$$A^{(k+1)} = Z_k^T A^{(k)} Z_k, \qquad B^{(k+1)} = Z_k^T B^{(k)} Z_k, \quad k \geq 0.$$

Derivation of the HZ Method

Preliminary transformation: $A^{(0)} = D_0 A D_0, \; B^{(0)} = D_0 B D_0$

$D_0 = [\mathrm{diag}(B)]^{-\frac{1}{2}}$, so that $b_{11}^{(0)} = b_{22}^{(0)} = \cdots = b_{nn}^{(0)} = 1.$

This property of $B^{(0)}$ is maintained during the iteration process:

$$A^{(k+1)} = Z_k^T A^{(k)} Z_k, \qquad B^{(k+1)} = Z_k^T B^{(k)} Z_k, \quad k \geq 0.$$

Each Z_k is a nonsingular elementary plane matrix

Preliminary transformation: $A^{(0)} = D_0 A D_0, \ B^{(0)} = D_0 B D_0$

$D_0 = [\mathrm{diag}(B)]^{-\frac{1}{2}}$, so that $b_{11}^{(0)} = b_{22}^{(0)} = \cdots = b_{nn}^{(0)} = 1.$

This property of $B^{(0)}$ is maintained during the iteration process:

$$A^{(k+1)} = Z_k^T A^{(k)} Z_k, \qquad B^{(k+1)} = Z_k^T B^{(k)} Z_k, \quad k \geq 0.$$

Each Z_k is a nonsingular elementary plane matrix

$$Z_k = \begin{array}{cc} \begin{bmatrix} I & & & & \\ & c_k & & -s_k & \\ & & I & & \\ & \tilde{s}_k & & \tilde{c}_k & \\ & & & & I \end{bmatrix} & \begin{array}{c} \\ i(k) \\ \\ j(k) \\ \\ \end{array} \end{array}, \qquad i(k) < j(k) \text{ are pivot indices at step } k,$$

$$c_k^2 + s_k^2 = \tilde{c}_k^2 + \tilde{s}_k^2 = 1 / \sqrt{1 - b_{i(k)j(k)}^2} \quad \text{(Gose 1979).}$$

Derivation of the HZ Method

Preliminary transformation: $A^{(0)} = D_0 A D_0, \; B^{(0)} = D_0 B D_0$

$D_0 = [\mathrm{diag}(B)]^{-\frac{1}{2}}$, so that $b_{11}^{(0)} = b_{22}^{(0)} = \cdots = b_{nn}^{(0)} = 1.$

This property of $B^{(0)}$ is maintained during the iteration process:

$$A^{(k+1)} = Z_k^T A^{(k)} Z_k, \qquad B^{(k+1)} = Z_k^T B^{(k)} Z_k, \quad k \geq 0.$$

Each Z_k is a nonsingular elementary plane matrix

$$Z_k = \begin{array}{cc} \begin{bmatrix} I & & & & \\ & c_k & & -s_k & \\ & & I & & \\ & \tilde{s}_k & & \tilde{c}_k & \\ & & & & I \end{bmatrix} & \begin{array}{c} \\ i(k) \\ \\ j(k) \\ \\ \end{array} \end{array}, \qquad i(k) < j(k) \text{ are pivot indices at step } k,$$

$$c_k^2 + s_k^2 = \tilde{c}_k^2 + \tilde{s}_k^2 = 1/\sqrt{1 - b_{i(k)j(k)}^2} \quad \text{(Gose 1979).}$$

The selection of pivot pairs $(i(k), j(k))$ defines pivot strategy.

To describe step k, we assume:

$$A = A^{(k)}, \quad A' = A^{(k+1)}, \quad Z_k = Z,$$

$$\hat{Z} = \begin{bmatrix} c & -s \\ \tilde{s} & \tilde{c} \end{bmatrix} \qquad \text{the pivot submatrix of } Z.$$

Derivation of the HZ Method

To describe step k, we assume:

$$A = A^{(k)}, \quad A' = A^{(k+1)}, \quad Z_k = Z,$$

$$\hat{Z} = \begin{bmatrix} c & -s \\ \tilde{s} & \tilde{c} \end{bmatrix} \qquad \text{the pivot submatrix of } Z.$$

We have

$$A' = Z^T A Z, \quad B' = Z^T B Z \qquad \left(\hat{A}' = \hat{Z}^T \hat{A} \hat{Z}, \quad \hat{B}' = \hat{Z}^T \hat{B} \hat{Z}\right).$$

Derivation of the HZ Method

To describe step k, we assume:

$$A = A^{(k)}, \quad A' = A^{(k+1)}, \quad Z_k = Z,$$

$$\hat{Z} = \begin{bmatrix} c & -s \\ \tilde{s} & \tilde{c} \end{bmatrix} \qquad \text{the pivot submatrix of } Z.$$

We have

$$A' = Z^T A Z, \quad B' = Z^T B Z \qquad \left(\hat{A}' = \hat{Z}^T \hat{A} \hat{Z}, \quad \hat{B}' = \hat{Z}^T \hat{B} \hat{Z} \right).$$

Z is chosen/constructed to annihilate the pivot elements a_{ij} and b_{ij}.

To describe step k, we assume:

$$A = A^{(k)}, \quad A' = A^{(k+1)}, \quad Z_k = Z,$$

$$\hat{Z} = \begin{bmatrix} c & -s \\ \tilde{s} & \tilde{c} \end{bmatrix} \qquad \text{the pivot submatrix of } Z.$$

We have

$$A' = Z^T A Z, \quad B' = Z^T B Z \qquad \left(\hat{A}' = \hat{Z}^T \hat{A} \hat{Z}, \quad \hat{B}' = \hat{Z}^T \hat{B} \hat{Z} \right).$$

Z is chosen/constructed to annihilate the pivot elements a_{ij} and b_{ij}.

$\hat{Z}$ is sought in the form of a product of two Jacobi rotations and one diagonal matrix. We have two possibilities:

$\hat{Z}$ is sought in the form:

$$\text{(a)} \quad \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{1+b_{ij}}} & 0 \\ 0 & \frac{1}{\sqrt{1-b_{ij}}} \end{bmatrix} \begin{bmatrix} \cos(\theta - \frac{\pi}{4}) & -\sin(\theta - \frac{\pi}{4}) \\ \sin(\theta - \frac{\pi}{4}) & \cos(\theta - \frac{\pi}{4}) \end{bmatrix}$$

$$\text{(b)} \quad \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{1-b_{ij}}} & 0 \\ 0 & \frac{1}{\sqrt{1+b_{ij}}} \end{bmatrix} \begin{bmatrix} \cos(\theta + \frac{\pi}{4}) & -\sin(\theta + \frac{\pi}{4}) \\ \sin(\theta + \frac{\pi}{4}) & \cos(\theta + \frac{\pi}{4}) \end{bmatrix}$$

$$\downarrow \qquad\qquad \downarrow \qquad\qquad \downarrow$$

$$\hat{B} \to \text{diag} \qquad \hat{B} \to I_2 \qquad \hat{A} \to \text{diag}$$

$\hat{Z}$ is sought in the form:

$$\text{(a)} \quad \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{1+b_{ij}}} & 0 \\ 0 & \frac{1}{\sqrt{1-b_{ij}}} \end{bmatrix} \begin{bmatrix} \cos(\theta - \frac{\pi}{4}) & -\sin(\theta - \frac{\pi}{4}) \\ \sin(\theta - \frac{\pi}{4}) & \cos(\theta - \frac{\pi}{4}) \end{bmatrix}$$

$$\text{(b)} \quad \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix} \begin{bmatrix} \frac{1}{\sqrt{1-b_{ij}}} & 0 \\ 0 & \frac{1}{\sqrt{1+b_{ij}}} \end{bmatrix} \begin{bmatrix} \cos(\theta + \frac{\pi}{4}) & -\sin(\theta + \frac{\pi}{4}) \\ \sin(\theta + \frac{\pi}{4}) & \cos(\theta + \frac{\pi}{4}) \end{bmatrix}$$

$$\downarrow \qquad\qquad\qquad \downarrow \qquad\qquad\qquad \downarrow$$

$$\hat{B} \to \text{diag} \qquad\qquad \hat{B} \to I_2 \qquad\qquad \hat{A} \to \text{diag}$$

The both possibilities yield the same algorithm.

Essential Part of the Algorithm

$$\xi = \frac{b_{ij}}{\sqrt{1+b_{ij}}+\sqrt{1-b_{ij}}}, \quad \rho = \xi + \sqrt{1-b_{ij}}, \quad \xi^2+\rho^2=1,$$

Essential Part of the Algorithm

$$\xi = \frac{b_{ij}}{\sqrt{1+b_{ij}} + \sqrt{1-b_{ij}}}, \quad \rho = \xi + \sqrt{1-b_{ij}}, \quad \xi^2 + \rho^2 = 1,$$

$$\tan(2\theta) = \frac{2a_{ij} - (a_{ii} + a_{jj})\, b_{ij}}{\sqrt{1-(b_{ij})^2}\,(a_{ii} - a_{jj})}, \qquad -\frac{\pi}{4} \le \theta \le \frac{\pi}{4},$$

Essential Part of the Algorithm

$$\xi = \frac{b_{ij}}{\sqrt{1+b_{ij}} + \sqrt{1-b_{ij}}}, \quad \rho = \xi + \sqrt{1-b_{ij}}, \quad \xi^2 + \rho^2 = 1,$$

$$\tan(2\theta) = \frac{2a_{ij} - (a_{ii} + a_{jj})\, b_{ij}}{\sqrt{1-(b_{ij})^2}\,(a_{ii} - a_{jj})}, \qquad -\frac{\pi}{4} \leq \theta \leq \frac{\pi}{4},$$

$$\begin{aligned}
\cos\phi &= \rho\cos\theta - \xi\sin\theta \\
\sin\phi &= \rho\sin\theta + \xi\cos\theta \\
\cos\psi &= \rho\cos\theta + \xi\sin\theta \\
\sin\psi &= \rho\sin\theta - \xi\cos\theta
\end{aligned}$$

Essential Part of the Algorithm

$$\xi = \frac{b_{ij}}{\sqrt{1+b_{ij}} + \sqrt{1-b_{ij}}}, \quad \rho = \xi + \sqrt{1-b_{ij}}, \quad \xi^2 + \rho^2 = 1,$$

$$\tan(2\theta) = \frac{2a_{ij} - (a_{ii} + a_{jj})\, b_{ij}}{\sqrt{1-(b_{ij})^2}\,(a_{ii} - a_{jj})}, \qquad -\frac{\pi}{4} \le \theta \le \frac{\pi}{4},$$

$$\begin{aligned}
\cos\phi &= \rho\cos\theta - \xi\sin\theta \\
\sin\phi &= \rho\sin\theta + \xi\cos\theta \\
\cos\psi &= \rho\cos\theta + \xi\sin\theta \\
\sin\psi &= \rho\sin\theta - \xi\cos\theta
\end{aligned}$$

$$\boxed{\hat{Z} = \frac{1}{\sqrt{1-b_{ij}^2}} \begin{bmatrix} \cos\phi & -\sin\phi \\ \cos\psi & \sin\psi \end{bmatrix}.}$$

Essential Part of the Algorithm

$$\xi = \frac{b_{ij}}{\sqrt{1+b_{ij}} + \sqrt{1-b_{ij}}}, \quad \rho = \xi + \sqrt{1-b_{ij}}, \quad \xi^2 + \rho^2 = 1,$$

$$\tan(2\theta) = \frac{2a_{ij} - (a_{ii} + a_{jj})\, b_{ij}}{\sqrt{1-(b_{ij})^2}\,(a_{ii} - a_{jj})}, \qquad -\frac{\pi}{4} \leq \theta \leq \frac{\pi}{4},$$

$$\begin{aligned}
\cos\phi &= \rho\cos\theta - \xi\sin\theta \\
\sin\phi &= \rho\sin\theta + \xi\cos\theta \\
\cos\psi &= \rho\cos\theta + \xi\sin\theta \\
\sin\psi &= \rho\sin\theta - \xi\cos\theta
\end{aligned}$$

$$\boxed{\hat{Z} = \frac{1}{\sqrt{1-b_{ij}^2}} \begin{bmatrix} \cos\phi & -\sin\phi \\ \cos\psi & \sin\psi \end{bmatrix}.}$$

$$\begin{aligned}
a'_{ii} &= a_{ii} + \frac{1}{1-b_{ij}^2}\left[(b_{ij}^2 - \sin^2\phi)\, a_{ii} + 2\cos\phi\sin\psi\, a_{ij} + \sin^2\psi\, a_{jj}\right] \\
a'_{jj} &= a_{jj} - \frac{1}{1-b_{ij}^2}\left[(\sin^2\psi - b_{ij}^2)\, a_{jj} + 2\cos\psi\sin\phi\, a_{ij} + \sin^2\phi\, a_{ii}\right]
\end{aligned}$$

There are more formulas!

$$\rho = \frac{1}{2}(\sqrt{1+b_{ij}} + \sqrt{1-b_{ij}}), \quad 2\rho\xi = b_{ij}.$$

It is easy to show the following relations: $\quad |\xi| \leq \sqrt{2}/2$, $\sqrt{2}/2 \leq \rho \leq 1$.

$$\begin{aligned}
\cos\phi\sin\psi &= \cos\theta\sin\theta - \rho\xi = 0.5\,(\sin 2\theta - b_{ij}), \\
\cos\psi\sin\phi &= \cos\theta\sin\theta + \rho\xi = 0.5\,(\sin 2\theta + b_{ij}), \\
\cos\phi\cos\psi &= \rho^2\cos^2\theta - \xi^2\sin^2\theta, \\
\sin\phi\sin\psi &= \rho^2\sin^2\theta - \xi^2\cos^2\theta.
\end{aligned}$$

$$\begin{aligned}
\min\{\cos\phi\,,\,\cos\psi\} &\geq \rho\cos\theta - \frac{|b_{ij}|}{2\rho}|\sin\theta| \geq (\rho - \frac{|b_{ij}|}{2\rho})\cos\theta > 0, \\
\max\{\cos\phi\,,\,\cos\psi\} &= \rho\cos\theta + |\xi\sin\theta| \geq \cos(\theta) \geq \frac{\sqrt{2}}{2}.
\end{aligned}$$

There are more formulas!

Let $\sin\gamma = b_{ij}$, $\cos\gamma = \sqrt{1-b_{ij}^2}$. Then

$$\frac{1}{\cos\gamma}\begin{bmatrix} a_{ii} & a_{ij} \\ a_{ij} & a_{jj} \end{bmatrix}\begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\psi & \cos\psi \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\phi & \cos\phi \end{bmatrix}\begin{bmatrix} a'_{ii} & \\ & a'_{jj} \end{bmatrix},$$

$$\frac{1}{\cos\gamma}\begin{bmatrix} 1 & b_{ij} \\ b_{ij} & 1 \end{bmatrix}\begin{bmatrix} \cos\phi & -\sin\phi \\ \sin\psi & \cos\psi \end{bmatrix} = \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\phi & \cos\phi \end{bmatrix},$$

$$\begin{aligned} \cos\gamma &= \frac{\cos\phi}{\cos\psi} + b_{ij}\tan\psi = \frac{\cos\psi}{\cos\phi} - b_{ij}\tan\phi, \\ 2\cos(\phi+\psi)a_{ij} &= a_{ii}\sin(2\phi) - a_{jj}\sin(2\psi). \end{aligned}$$

There are more formulas!

$$
\begin{aligned}
a'_{ii} &= \frac{1}{\cos\gamma}\left(a_{ii}\frac{\cos\phi}{\cos\psi} + a_{ij}\tan\psi\right) = \frac{a_{ii} + a_{ij}\frac{\sin\psi}{\cos\phi}}{1 + b_{ij}\frac{\sin\psi}{\cos\phi}}, \\
a'_{jj} &= \frac{1}{\cos\gamma}\left(a_{jj}\frac{\cos\psi}{\cos\phi} - a_{ij}\tan\phi\right) = \frac{a_{jj} - a_{ij}\frac{\sin\phi}{\cos\psi}}{1 - b_{ij}\frac{\sin\phi}{\cos\psi}}.
\end{aligned}
$$

We also have

$$
\phi + \psi = 2\theta\,, \qquad \text{hence} \qquad \begin{aligned} \phi &= \theta + \gamma/2, \\ \psi &= \theta - \gamma/2. \end{aligned}
$$

All these relations are used in the global convergence proof and in the proof of high relative accracy of the method.

Digression: Complex Matrices

If $A = A^*$ and $B = B^*$ are complex, with $B \succ O$ and $\mathrm{diag}(B) = I_n$, then one step of the HZ method uses the transformation

Digression: Complex Matrices

If $A = A^*$ and $B = B^*$ are complex, with $B \succ O$ and $\text{diag}(B) = I_n$, then one step of the HZ method uses the transformation

$$A' = Z^* A Z, \quad B' = Z^* B Z,$$

Z is chosen to annihilate the pivot elements a_{ij} and b_{ij}.

Digression: Complex Matrices

If $A = A^*$ and $B = B^*$ are complex, with $B \succ O$ and $\mathrm{diag}(B) = I_n$, then one step of the HZ method uses the transformation

$$A' = Z^* A Z, \quad B' = Z^* B Z,$$

Z is chosen to annihilate the pivot elements a_{ij} and b_{ij}.

It is proved that that pivot submatrix of Z has form

$$\hat{Z} = \begin{bmatrix} c & \bar{s} \\ -\tilde{s} & \tilde{c} \end{bmatrix}.$$

Digression: Complex Matrices

If $A = A^*$ and $B = B^*$ are complex, with $B \succ O$ and $\mathrm{diag}(B) = I_n$, then one step of the HZ method uses the transformation

$$A' = Z^* A Z, \quad B' = Z^* B Z,$$

Z is chosen to annihilate the pivot elements a_{ij} and b_{ij}.

It is proved that that pivot submatrix of Z has form

$$\hat{Z} = \begin{bmatrix} c & \bar{s} \\ -\tilde{s} & \tilde{c} \end{bmatrix}.$$

We obtain $\quad \hat{A}' = \hat{Z}^* \hat{A} \hat{Z}, \quad \hat{B}' = \hat{Z}^* \hat{B} \hat{Z}.$

Digression: Complex Matrices

If $A = A^*$ and $B = B^*$ are complex, with $B \succ O$ and $\mathrm{diag}(B) = I_n$, then one step of the HZ method uses the transformation

$$A' = Z^* A Z, \quad B' = Z^* B Z,$$

Z is chosen to annihilate the pivot elements a_{ij} and b_{ij}.

It is proved that that pivot submatrix of Z has form

$$\hat{Z} = \begin{bmatrix} c & \bar{s} \\ -\tilde{s} & \tilde{c} \end{bmatrix}.$$

We obtain $\hat{A}' = \hat{Z}^* \hat{A} \hat{Z}, \quad \hat{B}' = \hat{Z}^* \hat{B} \hat{Z}$. $\hat{Z}$ is sought as product of two complex Jacobi rotations and two diagonal matrices.

$\hat{Z}$ is sought in the form:

$$
\begin{array}{ccc}
 & \hat{B} \to \mathrm{diag} & \hat{B} \to I_2 \\
 & \uparrow & \uparrow
\end{array}
$$

$$
\hat{Z} = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} e^{\imath \arg(b_{ij})} \\ \frac{\sqrt{2}}{2} e^{-\imath \arg(b_{ij})} & \frac{\sqrt{2}}{2} \end{bmatrix} \cdot \begin{bmatrix} \frac{1}{\sqrt{1+|b_{ij}|}} & 0 \\ 0 & \frac{1}{\sqrt{1-|b_{ij}|}} \end{bmatrix}
$$

$$
\cdot \begin{bmatrix} \cos(\theta + \frac{\pi}{4}) & e^{\imath\alpha} \sin(\theta + \frac{\pi}{4}) \\ -e^{-\imath\alpha} \sin(\theta + \frac{\pi}{4}) & \cos(\theta + \frac{\pi}{4}) \end{bmatrix} \cdot \begin{bmatrix} e^{\imath\omega_i} & 0 \\ 0 & e^{\imath\omega_j} \end{bmatrix}
$$

$$
\begin{array}{cc}
\downarrow & \downarrow \\
\hat{A} \to \mathrm{diag} & \mathrm{diag}(\hat{Z}) \succ O
\end{array}
$$

Essential Part of the Algorithm

Let

$$b = |b_{ij}|, \quad t = \sqrt{1 - b^2}, \quad e = a_{jj} - a_{ii}, \quad \epsilon = \begin{cases} 1, & e \geq 0 \\ -1, & e < 0 \end{cases},$$

Essential Part of the Algorithm

Let

$$b = |b_{ij}|, \quad t = \sqrt{1-b^2}, \quad e = a_{jj} - a_{ii}, \quad \epsilon = \begin{cases} 1, & e \geq 0 \\ -1, & e < 0 \end{cases},$$

$$\begin{aligned}
u + \imath\, v &= e^{-\imath \arg(b_{ij})}\, a_{ij}, \qquad \tan\gamma = 2\tfrac{v}{|e|}, \quad -\tfrac{\pi}{2} < \gamma \leq \tfrac{\pi}{2} \\
\tan 2\theta &= \epsilon \frac{2u-(a_{ii}+a_{jj})b}{t\sqrt{e^2+4v^2}}, \quad -\tfrac{\pi}{4} < \theta \leq \tfrac{\pi}{4} \\
2\cos^2\phi &= 1 + b\sin 2\theta + t\cos 2\theta\cos\gamma, \qquad 0 \leq \phi \leq \tfrac{\pi}{2} \\
2\cos^2\psi &= 1 - b\sin 2\theta + t\cos 2\theta\cos\gamma, \qquad 0 \leq \psi \leq \tfrac{\pi}{2} \\
e^{\imath\alpha}\sin\phi &= \frac{e^{\imath \arg(b_{ij})}}{2\cos\psi}\left[\sin 2\theta - b - \imath t\cos 2\theta\sin\gamma\right] \\
e^{-\imath\beta}\sin\psi &= \frac{e^{-\imath \arg(b_{ij})}}{2\cos\phi}\left[\sin 2\theta + b + \imath t\cos 2\theta\sin\gamma\right].
\end{aligned}$$

Essential Part of the Algorithm

Let

$$b = |b_{ij}|, \quad t = \sqrt{1-b^2}, \quad e = a_{jj} - a_{ii}, \quad \epsilon = \begin{cases} 1, & e \geq 0 \\ -1, & e < 0 \end{cases},$$

$$\begin{aligned}
u + \imath\, v &= e^{-\imath \arg(b_{ij})}\, a_{ij}, \qquad \tan\gamma = 2\tfrac{v}{|e|}, \quad -\tfrac{\pi}{2} < \gamma \leq \tfrac{\pi}{2} \\
\tan 2\theta &= \epsilon \tfrac{2u-(a_{ii}+a_{jj})b}{t\sqrt{e^2+4v^2}}, \quad -\tfrac{\pi}{4} < \theta \leq \tfrac{\pi}{4} \\
2\cos^2\phi &= 1 + b\sin 2\theta + t\cos 2\theta\cos\gamma, \qquad 0 \leq \phi \leq \tfrac{\pi}{2} \\
2\cos^2\psi &= 1 - b\sin 2\theta + t\cos 2\theta\cos\gamma, \qquad 0 \leq \psi \leq \tfrac{\pi}{2} \\
e^{\imath\alpha}\sin\phi &= \tfrac{e^{\imath \arg(b_{ij})}}{2\cos\psi}\left[\sin 2\theta - b - \imath t\cos 2\theta\sin\gamma\right] \\
e^{-\imath\beta}\sin\psi &= \tfrac{e^{-\imath \arg(b_{ij})}}{2\cos\phi}\left[\sin 2\theta + b + \imath t\cos 2\theta\sin\gamma\right].
\end{aligned}$$

Then

$$\hat{Z} = \frac{1}{\sqrt{1-b^2}} \begin{bmatrix} \cos\phi & e^{\imath\alpha}\sin\phi \\ -e^{-\imath\beta}\sin\psi & \cos\psi \end{bmatrix}$$

New Algorithms: Based on LL^T and RR^T Factorizations

Consider the Cholesky foctorization of $\hat{B}$:

$$\begin{bmatrix} 1 & b_{ij} \\ b_{ij} & 1 \end{bmatrix} = \hat{B} = \hat{L}\hat{L}^T = \begin{bmatrix} 1 & 0 \\ a & c \end{bmatrix} \begin{bmatrix} 1 & a \\ 0 & c \end{bmatrix} = \begin{bmatrix} 1 & a \\ a & a^2 + c^2 \end{bmatrix}.$$

New Algorithms: Based on LL^T and RR^T Factorizations

Consider the Cholesky foctorization of $\hat{B}$:

$$\begin{bmatrix} 1 & b_{ij} \\ b_{ij} & 1 \end{bmatrix} = \hat{B} = \hat{L}\hat{L}^T = \begin{bmatrix} 1 & 0 \\ a & c \end{bmatrix} \begin{bmatrix} 1 & a \\ 0 & c \end{bmatrix} = \begin{bmatrix} 1 & a \\ a & a^2 + c^2 \end{bmatrix}.$$

Assuming $c > 0$, one obtains $a = b_{ij}$, $c = \sqrt{1 - b_{ij}^2}$, hence

New Algorithms: Based on LL^T and RR^T Factorizations

Consider the Cholesky foctorization of $\hat{B}$:

$$\begin{bmatrix} 1 & b_{ij} \\ b_{ij} & 1 \end{bmatrix} = \hat{B} = \hat{L}\hat{L}^T = \begin{bmatrix} 1 & 0 \\ a & c \end{bmatrix} \begin{bmatrix} 1 & a \\ 0 & c \end{bmatrix} = \begin{bmatrix} 1 & a \\ a & a^2 + c^2 \end{bmatrix}.$$

Assuming $c > 0$, one obtains $a = b_{ij}$, $c = \sqrt{1 - b_{ij}^2}$, hence

$$\hat{L} = \begin{bmatrix} 1 & 0 \\ b_{ij} & \sqrt{1 - b_{ij}^2} \end{bmatrix}, \quad \hat{L}^{-1} = \begin{bmatrix} 1 & 0 \\ -\frac{b_{ij}}{\sqrt{1-b_{ij}^2}} & \frac{1}{\sqrt{1-b_{ij}^2}} \end{bmatrix}.$$

New Algorithms: Based on LL^T and RR^T Factorizations

Consider the Cholesky foctorization of $\hat{B}$:

$$\begin{bmatrix} 1 & b_{ij} \\ b_{ij} & 1 \end{bmatrix} = \hat{B} = \hat{L}\hat{L}^T = \begin{bmatrix} 1 & 0 \\ a & c \end{bmatrix} \begin{bmatrix} 1 & a \\ 0 & c \end{bmatrix} = \begin{bmatrix} 1 & a \\ a & a^2 + c^2 \end{bmatrix}.$$

Assuming $c > 0$, one obtains $a = b_{ij}$, $c = \sqrt{1 - b_{ij}^2}$, hence

$$\hat{L} = \begin{bmatrix} 1 & 0 \\ b_{ij} & \sqrt{1 - b_{ij}^2} \end{bmatrix}, \quad \hat{L}^{-1} = \begin{bmatrix} 1 & 0 \\ -\frac{b_{ij}}{\sqrt{1-b_{ij}^2}} & \frac{1}{\sqrt{1-b_{ij}^2}} \end{bmatrix}.$$

If we write $\hat{F}_1 = \hat{L}^{-T}$, then $\hat{F}_1^T \hat{B} \hat{F}_1 = I_2$ and

The Algorithm Based on LL^T Factorization

$$
\begin{aligned}
\hat{F}_1^T \hat{A} \hat{F}_1 &= \begin{bmatrix} 1 & 0 \\ f_{ij} & f_{jj} \end{bmatrix} \begin{bmatrix} a_{ii} & a_{ij} \\ a_{ij} & a_{jj} \end{bmatrix} \begin{bmatrix} 1 & f_{ij} \\ 0 & f_{jj} \end{bmatrix} \\
&= \begin{bmatrix} a_{ii} & f_{ij} a_{ii} + f_{jj} a_{ij} \\ f_{ij} a_{ii} + f_{jj} a_{ij} & f_{ij}^2 a_{ii} + 2 f_{ij} f_{jj} a_{ij} + f_{jj}^2 a_{jj} \end{bmatrix} \\
&= \begin{bmatrix} a_{ii} & \frac{a_{ij} - b_{ij} a_{ii}}{\sqrt{1-b_{ij}^2}} \\ \frac{a_{ij} - b_{ij} a_{ii}}{\sqrt{1-b_{ij}^2}} & a_{jj} - \frac{2a_{ij} - (a_{ii} + a_{jj}) b_{ij}}{1-b_{ij}^2} b_{ij} \end{bmatrix},
\end{aligned} \tag{1}
$$

where we have used $f_{ij} = -b_{ij}/\sqrt{1-b_{ij}^2}$, $f_{jj} = 1/\sqrt{1-b_{ij}^2}$.

The Algorithm Based on LL^T Factorization

$$
\begin{aligned}
\hat{F}_1^T \hat{A} \hat{F}_1 &= \begin{bmatrix} 1 & 0 \\ f_{ij} & f_{jj} \end{bmatrix} \begin{bmatrix} a_{ii} & a_{ij} \\ a_{ij} & a_{jj} \end{bmatrix} \begin{bmatrix} 1 & f_{ij} \\ 0 & f_{jj} \end{bmatrix} \\
&= \begin{bmatrix} a_{ii} & f_{ij}a_{ii} + f_{jj}a_{ij} \\ f_{ij}a_{ii} + f_{jj}a_{ij} & f_{ij}^2 a_{ii} + 2f_{ij}f_{jj}a_{ij} + f_{jj}^2 a_{jj} \end{bmatrix} \\
&= \begin{bmatrix} a_{ii} & \frac{a_{ij}-b_{ij}a_{ii}}{\sqrt{1-b_{ij}^2}} \\ \frac{a_{ij}-b_{ij}a_{ii}}{\sqrt{1-b_{ij}^2}} & a_{jj} - \frac{2a_{ij}-(a_{ii}+a_{jj})b_{ij}}{1-b_{ij}^2} b_{ij} \end{bmatrix},
\end{aligned} \tag{1}
$$

where we have used $f_{ij} = -b_{ij}/\sqrt{1-b_{ij}^2}$, $f_{jj} = 1/\sqrt{1-b_{ij}^2}$.

The final $\hat{F}$ has the form $\hat{F} = \hat{F}_1 \hat{R}$, where $\hat{R}$ is the Jacobi transformation which diagonalizes $\hat{F}_1^T \hat{A} \hat{F}_1$. Its angle ϑ is determined by the formula

The Algorithm Based on LL^T Factorization

$$\tan(2\vartheta) = \frac{2(a_{ij} - b_{ij}a_{ii})\sqrt{1 - b_{ij}^2}}{a_{ii} - a_{jj} + 2(a_{ij} - b_{ij}a_{ii})b_{ij}}, \quad -\frac{\pi}{4} \le \vartheta \le \frac{\pi}{4}.$$

The Algorithm Based on LL^T Factorization

$$\tan(2\vartheta) = \frac{2(a_{ij} - b_{ij}a_{ii})\sqrt{1-b_{ij}^2}}{a_{ii} - a_{jj} + 2(a_{ij} - b_{ij}a_{ii})b_{ij}}, \quad -\frac{\pi}{4} \le \vartheta \le \frac{\pi}{4}.$$

The transformation formulas for the diagonal elements of A read

$$a'_{ii} = a_{ii} + \tan\vartheta \cdot \frac{a_{ij} - a_{ii}b_{ij}}{\sqrt{1-b_{ij}^2}} \tag{2}$$

$$a'_{jj} = a_{jj} - \frac{2a_{ij}b_{ij} - b_{ij}^2(a_{ii} + a_{jj})}{1 - b_{ij}^2} - \tan\vartheta \cdot \frac{a_{ij} - a_{ii}b_{ij}}{\sqrt{1-b_{ij}^2}} \tag{3}$$

The Algorithm Based on LL^T Factorization

$$\tan(2\vartheta) = \frac{2(a_{ij} - b_{ij}a_{ii})\sqrt{1-b_{ij}^2}}{a_{ii} - a_{jj} + 2(a_{ij} - b_{ij}a_{ii})b_{ij}}, \quad -\frac{\pi}{4} \le \vartheta \le \frac{\pi}{4}.$$

The transformation formulas for the diagonal elements of A read

$$a'_{ii} = a_{ii} + \tan\vartheta \cdot \frac{a_{ij} - a_{ii}b_{ij}}{\sqrt{1-b_{ij}^2}} \tag{2}$$

$$a'_{jj} = a_{jj} - \frac{2a_{ij}b_{ij} - b_{ij}^2(a_{ii}+a_{jj})}{1-b_{ij}^2} - \tan\vartheta \cdot \frac{a_{ij} - a_{ii}b_{ij}}{\sqrt{1-b_{ij}^2}} \tag{3}$$

If $a_{ii} = a_{jj}$, $a_{ij} = a_{ii}b_{ij}$ then ϑ is determined from expression $0/0$, so we choose $\vartheta = 0$. In this case a'_{ii} and a'_{jj} reduce to a_{ii} and a_{jj}, respectively.

The Algorithm Based on LL^T Factorization

This leads to a simpler matrix

$$\begin{aligned}
\hat{Z} &= \frac{1}{\sqrt{1-b_{ij}^2}} \begin{bmatrix} \sqrt{1-b_{ij}^2} & -b_{ij} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} c_\vartheta & -s_\vartheta \\ s_\vartheta & c_\vartheta \end{bmatrix} \\
&= \frac{1}{\sqrt{1-b_{ij}^2}} \begin{bmatrix} c_{\tilde{\vartheta}} & -s_{\tilde{\vartheta}} \\ s_\vartheta & c_\vartheta \end{bmatrix}, \qquad \begin{aligned} c_{\tilde{\vartheta}} &= c_\vartheta \sqrt{1-b_{ij}^2} - s_\vartheta b_{ij}, \\ s_{\tilde{\vartheta}} &= c_\vartheta b_{ij} + s_\vartheta \sqrt{1-b_{ij}^2}. \end{aligned}
\end{aligned}$$

It is easy to check that $c_{\tilde{\vartheta}}^2 + s_{\tilde{\vartheta}}^2 = 1$.

The Algorithm Based on RR^T Factorizations

Consider the RR^T factorization of $\hat{B}$:

$$\begin{bmatrix} 1 & b_{ij} \\ b_{ij} & 1 \end{bmatrix} = \hat{B} = \hat{R}\hat{R}^T = \begin{bmatrix} c & a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} c & 0 \\ a & 1 \end{bmatrix} = \begin{bmatrix} a^2 + c^2 & a \\ a & 1 \end{bmatrix}.$$

Assuming positive c, one obtains $a = b_{ij}$, $c = \sqrt{1 - b_{ij}^2}$, hence

$$\hat{R} = \begin{bmatrix} \sqrt{1 - b_{ij}^2} & b_{ij} \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \hat{R}^{-1} = \begin{bmatrix} \frac{1}{\sqrt{1-b_{ij}^2}} & -\frac{b_{ij}}{\sqrt{1-b_{ij}^2}} \\ 0 & 1 \end{bmatrix}.$$

The Algorithm Based on RR^T Factorizations

Consider the RR^T factorization of $\hat{B}$:

$$\begin{bmatrix} 1 & b_{ij} \\ b_{ij} & 1 \end{bmatrix} = \hat{B} = \hat{R}\hat{R}^T = \begin{bmatrix} c & a \\ 0 & 1 \end{bmatrix} \begin{bmatrix} c & 0 \\ a & 1 \end{bmatrix} = \begin{bmatrix} a^2 + c^2 & a \\ a & 1 \end{bmatrix}.$$

Assuming positive c, one obtains $a = b_{ij}$, $c = \sqrt{1 - b_{ij}^2}$, hence

$$\hat{R} = \begin{bmatrix} \sqrt{1 - b_{ij}^2} & b_{ij} \\ 0 & 1 \end{bmatrix} \quad \text{and} \quad \hat{R}^{-1} = \begin{bmatrix} \frac{1}{\sqrt{1-b_{ij}^2}} & -\frac{b_{ij}}{\sqrt{1-b_{ij}^2}} \\ 0 & 1 \end{bmatrix}.$$

If we write $\hat{F}_2 = \hat{R}^{-T}$, then $\hat{F}_2^T \hat{B} \hat{F}_2 = I_2$ and

The Algorithm Based on RR^T Factorization

$$
\begin{aligned}
\hat{F}_2^T \hat{A} \hat{F}_2 &= \begin{bmatrix} f_{ii} & f_{ji} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a_{ii} & a_{ij} \\ a_{ij} & a_{jj} \end{bmatrix} \begin{bmatrix} f_{ii} & 0 \\ f_{ji} & 1 \end{bmatrix} \\
&= \begin{bmatrix} f_{ii}^2 a_{ii} + 2 f_{ii} f_{ji} a_{ij} + f_{ji}^2 a_{jj} & f_{ii} a_{ij} + f_{ji} a_{jj} \\ f_{ii} a_{ij} + f_{ji} a_{jj} & a_{jj} \end{bmatrix} \\
&= \begin{bmatrix} a_{ii} - \frac{2a_{ij} - (a_{ii} + a_{jj}) b_{ij}}{1 - b_{ij}^2} b_{ij} & \frac{a_{ij} - b_{ij} a_{jj}}{\sqrt{1 - b_{ij}^2}} \\ \frac{a_{ij} - b_{ij} a_{jj}}{\sqrt{1 - b_{ij}^2}} & a_{jj} \end{bmatrix},
\end{aligned} \tag{4}
$$

where we have used $\quad f_{ii} = 1/\sqrt{1 - b_{ij}^2}, \;\; f_{ji} = -b_{ij}/\sqrt{1 - b_{ij}^2}.$

The Algorithm Based on RR^T Factorization

$$
\begin{aligned}
\hat{F}_2^T \hat{A} \hat{F}_2 &= \begin{bmatrix} f_{ii} & f_{ji} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} a_{ii} & a_{ij} \\ a_{ij} & a_{jj} \end{bmatrix} \begin{bmatrix} f_{ii} & 0 \\ f_{ji} & 1 \end{bmatrix} \\
&= \begin{bmatrix} f_{ii}^2 a_{ii} + 2 f_{ii} f_{ji} a_{ij} + f_{ji}^2 a_{jj} & f_{ii} a_{ij} + f_{ji} a_{jj} \\ f_{ii} a_{ij} + f_{ji} a_{jj} & a_{jj} \end{bmatrix} \\
&= \begin{bmatrix} a_{ii} - \frac{2a_{ij} - (a_{ii} + a_{jj}) b_{ij}}{1 - b_{ij}^2} b_{ij} & \frac{a_{ij} - b_{ij} a_{jj}}{\sqrt{1 - b_{ij}^2}} \\ \frac{a_{ij} - b_{ij} a_{jj}}{\sqrt{1 - b_{ij}^2}} & a_{jj} \end{bmatrix},
\end{aligned} \tag{4}
$$

where we have used $\quad f_{ii} = 1/\sqrt{1 - b_{ij}^2}, \;\; f_{ji} = -b_{ij}/\sqrt{1 - b_{ij}^2}$.

The final $\hat{F}$ has the form $\hat{F} = \hat{F}_2 \hat{J}$, where $\hat{J}$ is the Jacobi transformation which diagonalizes $\hat{F}_2^T \hat{A} \hat{F}_2$. Its angle ϑ is determined by the formula

The Algorithm Based on RR^T Factorization

$$\tan(2\vartheta) = \frac{2(a_{ij} - b_{ij}a_{jj})\sqrt{1 - b_{ij}^2}}{a_{ii} - a_{jj} - 2(a_{ij} - b_{ij}a_{jj})b_{ij}}, \quad -\frac{\pi}{4} \le \vartheta \le \frac{\pi}{4}.$$

The Algorithm Based on RR^T Factorization

$$\tan(2\vartheta) = \frac{2(a_{ij} - b_{ij}a_{jj})\sqrt{1-b_{ij}^2}}{a_{ii} - a_{jj} - 2(a_{ij} - b_{ij}a_{jj})b_{ij}}, \quad -\frac{\pi}{4} \le \vartheta \le \frac{\pi}{4}.$$

The transformation formulas for the diagonal elements of A read

$$\begin{aligned} a'_{ii} &= a_{ii} - \frac{2a_{ij} - (a_{ii} + a_{jj})b_{ij}}{1 - b_{ij}^2} b_{ij} + \tan\vartheta \cdot \frac{a_{ij} - a_{jj}b_{ij}}{\sqrt{1 - b_{ij}^2}} \\ a'_{jj} &= a_{jj} - \tan\vartheta \cdot \frac{a_{ij} - a_{jj}b_{ij}}{\sqrt{1 - b_{ij}^2}} \end{aligned}$$

$$\tan(2\vartheta) = \frac{2(a_{ij} - b_{ij}a_{jj})\sqrt{1-b_{ij}^2}}{a_{ii} - a_{jj} - 2(a_{ij} - b_{ij}a_{jj})b_{ij}}, \quad -\frac{\pi}{4} \le \vartheta \le \frac{\pi}{4}.$$

The transformation formulas for the diagonal elements of A read

$$\begin{aligned} a'_{ii} &= a_{ii} - \frac{2a_{ij} - (a_{ii} + a_{jj})b_{ij}}{1 - b_{ij}^2} b_{ij} + \tan\vartheta \cdot \frac{a_{ij} - a_{jj}b_{ij}}{\sqrt{1-b_{ij}^2}} \\ a'_{jj} &= a_{jj} - \tan\vartheta \cdot \frac{a_{ij} - a_{jj}b_{ij}}{\sqrt{1-b_{ij}^2}} \end{aligned}$$

If $a_{ii} = a_{jj}$, $a_{ij} = a_{jj}b_{ij}$ then ϑ is determined from expression $0/0$, so we choose $\vartheta = 0$. In this case a'_{ii} and a'_{jj} reduce to a_{ii} and a_{jj}, respectively.

The Algorithm Based on RR^T Factorization

This leads to a simpler matrix

$$\begin{aligned}
\hat{Z} &= \frac{1}{\sqrt{1-b_{ij}^2}} \begin{bmatrix} 1 & 0 \\ -b_{ij} & \sqrt{1-b_{ij}^2} \end{bmatrix} \begin{bmatrix} c_\vartheta & -s_\vartheta \\ s_\vartheta & c_\vartheta \end{bmatrix} \\
&= \frac{1}{\sqrt{1-b_{ij}^2}} \begin{bmatrix} c_\vartheta & -s_\vartheta \\ s_{\tilde{\vartheta}} & c_{\tilde{\vartheta}} \end{bmatrix}, \qquad \begin{aligned} c_{\tilde{\vartheta}} &= c_\vartheta \sqrt{1-b_{ij}^2} + s_\vartheta b_{ij}, \\ s_{\tilde{\vartheta}} &= s_\vartheta \sqrt{1-b_{ij}^2} - c_\vartheta b_{ij}. \end{aligned}
\end{aligned}$$

It is easy to check that $c_{\tilde{\vartheta}}^2 + s_{\tilde{\vartheta}}^2 = 1$.

The algorithms based on LL^T and RR^T factorizations can be generalized to work with complex matrices

Definition of a Hybrid and a General Method

Definition

Let $\mathcal{H}$ denote a collection of Jacobi methods for the positive definite generalized eigenvalue problem $Ax = \lambda Bx$ which satisfy the following two rules:

1. at step k the pivot submatrix $\hat{A}^{(k)}$ is diagonalized and $\hat{B}^{(k)}$ is transformed to I_2,
2. at least one of the two diagonal elements of the pivot submatrix $\hat{F}_k$ is not smaller than $\sqrt{2}/2$.

An element of $\mathcal{H}$ is called a *general PGEP Jacobi method*. A *hybrid Jacobi method* is any method from $\mathcal{H}$ that uses at each step either the HZ, $LL^T J$ or $RR^T J$ algorithm.

Definition of a Hybrid and a General Method

Definition

Let $\mathcal{H}$ denote a collection of Jacobi methods for the positive definite generalized eigenvalue problem $Ax = \lambda Bx$ which satisfy the following two rules:

1. at step k the pivot submatrix $\hat{A}^{(k)}$ is diagonalized and $\hat{B}^{(k)}$ is transformed to I_2,
2. at least one of the two diagonal elements of the pivot submatrix $\hat{F}_k$ is not smaller than $\sqrt{2}/2$.

An element of $\mathcal{H}$ is called a *general PGEP Jacobi method*. A *hybrid Jacobi method* is any method from $\mathcal{H}$ that uses at each step either the HZ, LL^TJ or RR^TJ algorithm.

In this definition the pivot strategy is not specified, hence any can be used. If a Jacobi method uses only the HZ (LL^TJ, RR^TJ) algorithm, it will be called the HZ (LL^TJ, RR^TJ) method.

Some Remarks

- It is easy to show that HZ, LL^TJ and RR^TJ methods belong to the class $\mathcal{H}$.

Some Remarks

- It is easy to show that HZ, LL^TJ and RR^TJ methods belong to the class $\mathcal{H}$.
- Algorithms based on LL^T and RR^T factorizations have got their names LL^TJ and RR^TJ algorithm, because LL^T and RR^T factorizations are followed by one step of the Jacobi method for the symmetric matrix.

Some Remarks

- It is easy to show that HZ, $LL^T J$ and $RR^T J$ methods belong to the class $\mathcal{H}$.
- Algorithms based on LL^T and RR^T factorizations have got their names $LL^T J$ and $RR^T J$ algorithm, because LL^T and RR^T factorizations are followed by one step of the Jacobi method for the symmetric matrix.
- The general (PGEP) Jacobi method can use at each step any conceivable algorithm which satisfies the above two rules. For example, it can use the FL method combined with normalization of the elements of B.

Some Remarks

- All real algorithms have the form

$$\hat{Z} = \frac{1}{\sqrt{1 - b_{ij}^2}} \begin{bmatrix} \cos\phi & -\sin\phi \\ \cos\psi & \sin\psi \end{bmatrix}.$$

  This follows from a result of Gose (ZAMM 59, 1979), who found the general form of a matrix $\hat{Z}$ which diagonalizes a positive definite symmetric matrix $\hat{B}$ of order 2 via the congruence transformation $\hat{B} \mapsto \hat{Z}^T \hat{B} \hat{Z}$.

Some Remarks

- All real algorithms have the form

$$\hat{Z} = \frac{1}{\sqrt{1 - b_{ij}^2}} \begin{bmatrix} \cos\phi & -\sin\phi \\ \cos\psi & \sin\psi \end{bmatrix}.$$

  This follows from a result of Gose (ZAMM 59, 1979), who found the general form of a matrix $\hat{Z}$ which diagonalizes a positive definite symmetric matrix $\hat{B}$ of order 2 via the congruence transformation $\hat{B} \mapsto \hat{Z}^T \hat{B} \hat{Z}$.

  If we assume $b_{11} = \cdots = b_{nn}$ and the same condition for $\hat{Z}^T \hat{B} \hat{Z}$, then this form of $\hat{Z}$ is just the Gose's theorem.

Some Remarks

- All real algorithms have the form

$$\hat{Z} = \frac{1}{\sqrt{1 - b_{ij}^2}} \begin{bmatrix} \cos\phi & -\sin\phi \\ \cos\psi & \sin\psi \end{bmatrix}.$$

This follows from a result of Gose (ZAMM 59, 1979), who found the general form of a matrix $\hat{Z}$ which diagonalizes a positive definite symmetric matrix $\hat{B}$ of order 2 via the congruence transformation $\hat{B} \mapsto \hat{Z}^T \hat{B} \hat{Z}$.

If we assume $b_{11} = \cdots = b_{nn}$ and the same condition for $\hat{Z}^T \hat{B} \hat{Z}$, then this form of $\hat{Z}$ is just the Gose's theorem.

Later Hari generalized that result to complex matrices.

Global Convergence (Real and Complex Algorithm)

We have used the following measure in the convergence analysis:

$$S^2(A) = \|A - \mathrm{diag}(A)\|_F^2, \quad S(A, B) = \left[S^2(A) + S^2(B)\right]^{1/2}.$$

The HZ method converges globally if

$$A^{(k)} \to \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n), \quad B^{(k)} \to I_n \qquad \text{as} \quad k \to \infty,$$

holds for any initial pair of symmetric matrices (A, B) with $B \succ O$.

Global Convergence (Real and Complex Algorithm)

We have used the following measure in the convergence analysis:

$$S^2(A) = \|A - \mathrm{diag}(A)\|_F^2, \quad S(A,B) = \left[S^2(A) + S^2(B)\right]^{1/2}.$$

The HZ method converges globally if

$$A^{(k)} \to \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n), \quad B^{(k)} \to I_n \qquad \text{as} \quad k \to \infty,$$

holds for any initial pair of symmetric matrices (A, B) with $B \succ O$.

Actually, it is sufficient to show that $S(A,B) \to 0$ as $k \to \infty$.

Global Convergence (Real and Complex Algorithm)

We have used the following measure in the convergence analysis:

$$S^2(A) = \|A - \mathrm{diag}(A)\|_F^2, \quad S(A,B) = \left[S^2(A) + S^2(B)\right]^{1/2}.$$

The HZ method converges globally if

$$A^{(k)} \to \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n), \quad B^{(k)} \to I_n \qquad \text{as} \quad k \to \infty,$$

holds for any initial pair of symmetric matrices (A, B) with $B \succ O$.

Actually, it is sufficient to show that $S(A, B) \to 0$ as $k \to \infty$.

We have first proved the global convergence for the serial pivot strategies.

Global Convergence (Real and Complex Algorithm)

We have used the following measure in the convergence analysis:

$$S^2(A) = \|A - \mathrm{diag}(A)\|_F^2, \quad S(A,B) = \left[S^2(A) + S^2(B)\right]^{1/2}.$$

The HZ method converges globally if

$$A^{(k)} \to \Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n), \quad B^{(k)} \to I_n \qquad \text{as} \quad k \to \infty,$$

holds for any initial pair of symmetric matrices (A, B) with $B \succ O$.

Actually, it is sufficient to show that $S(A,B) \to 0$ as $k \to \infty$.

We have first proved the global convergence for the serial pivot strategies.

Then we have proved the global convergence for a new much larger class of generalized serial strategies which includes the class of weak wavefront strategies.

Asymptotic Convergence (Real and Complex Algorithm)

Let (A, B) have simple eigenvalues:

$$\lambda_1 > \lambda_2 > \cdots > \lambda_n, \qquad \mu = \max\{|\lambda_1|\,,\, |\lambda_n|\},$$

$$3\delta_i = \min_{\substack{1\le i\le n\\ j\neq i}} |\ \lambda_i - \lambda_j\ |, \quad 1 \le i \le n; \qquad \delta = \min_{1\le i\le n} \delta_i\,.$$

Asymptotic Convergence (Real and Complex Algorithm)

Let (A, B) have simple eigenvalues:

$$\lambda_1 > \lambda_2 > \cdots > \lambda_n, \qquad \mu = \max\{|\lambda_1|\,,\,|\lambda_n|\},$$

$$3\delta_i = \min_{\substack{1\leq i\leq n \\ j\neq i}} |\,\lambda_i - \lambda_j\,|, \quad 1 \leq i \leq n; \qquad \delta = \min_{1\leq i\leq n} \delta_i\,.$$

Theorem

If $S(B^{(0)}) < \dfrac{1}{n(n-1)}$ *and* $S(A^{(0)}, B^{(0)}) < \dfrac{\delta}{2\sqrt{1+\mu^2}}$,
then for the general cyclic and for the serial strategies it holds, respectively:

$$S(A^{(N)}, B^{(N)}) \;\leq\; \sqrt{N(1+\mu^2)}\,\frac{S^2(A^{(0)}, B^{(0)})}{\delta}, \quad N = n(n-1)/2$$

$$S(A^{(N)}, B^{(N)}) \;\leq\; \sqrt{1+\mu^2}\,\frac{S^2(A^{(0)}, B^{(0)})}{\delta}.$$

In the case of multiple eigenvalues, the method is not quadratically convergent, but can be modified to be such.

Multiple Eigenvalues

The situation complicates because the positive definite pair (A, B) with multiple eigenvalues, and with nearly diagonal matrices, has special structure.

Multiple Eigenvalues

The situation complicates because the positive definite pair (A, B) with multiple eigenvalues, and with nearly diagonal matrices, has special structure.

Let $\quad A = A^* \text{ with } a_{11} \geq a_{22} \geq \cdots \geq a_{nn},$

$\qquad B = B^* \text{ with } B \succ O, \ \text{diag}(B) = I_n.$

Multiple Eigenvalues

The situation complicates because the positive definite pair (A, B) with multiple eigenvalues, and with nearly diagonal matrices, has special structure.

Let $\quad A = A^*$ with $a_{11} \geq a_{22} \geq \cdots \geq a_{nn}$,

$\quad B = B^*$ with $B \succ O$, $\mathrm{diag}(B) = I_n$.

Let

$$\lambda_1 = \cdots = \lambda_{s_1} > \lambda_{s_1+1} = \cdots = \lambda_{s_2} > \cdots > \lambda_{s_{p-1}+1} = \cdots = \lambda_{s_p},$$

where $s_p = n$.

Multiple Eigenvalues

The situation complicates because the positive definite pair (A, B) with multiple eigenvalues, and with nearly diagonal matrices, has special structure.

Let $\quad A = A^* \text{ with } a_{11} \geq a_{22} \geq \cdots \geq a_{nn},$

$\qquad B = B^* \text{ with } B \succ O, \text{ diag}(B) = I_n.$

Let

$$\lambda_1 = \cdots = \lambda_{s_1} > \lambda_{s_1+1} = \cdots = \lambda_{s_2} > \cdots > \lambda_{s_{p-1}+1} = \cdots = \lambda_{s_p},$$

where $s_p = n$. Then

$$n_i = s_i - s_{i-1}, \qquad 1 \leq i \leq p \quad (s_0 = 0),$$

n_i is the multiplicity of λ_{s_i}. Again, let $\mu = \max\{|\lambda_{s_1}|, |\lambda_{s_p}|\}$.

Multiple Eigenvalues

The minimum distance between two distinct eigenvalues plays special role in the analysis. Let δ_r be the absolute gap (separation) of λ_{s_r} from other eigenvalues,

$$3\delta_r = \min_{\substack{1\leq t\leq p \\ t\neq r}} \mid \lambda_{s_r} - \lambda_{s_t} \mid, \quad 1 \leq r \leq p.$$

Multiple Eigenvalues

The minimum distance between two distinct eigenvalues plays special role in the analysis. Let δ_r be the absolute gap (separation) of λ_{s_r} from other eigenvalues,

$$3\delta_r = \min_{\substack{1\leq t\leq p \\ t\neq r}} \mid \lambda_{s_r} - \lambda_{s_t} \mid, \quad 1 \leq r \leq p.$$

Then $\quad \delta = \min_{1\leq r\leq p} \delta_r \quad$ is the minimum absolute gap.

Multiple Eigenvalues

Next we consider the following matrix block-partition

$$A = \begin{bmatrix} A_{11} & \cdots & A_{1p} \\ \vdots & \ddots & \vdots \\ A_{p1} & \cdots & A_{pp} \end{bmatrix}, \quad B = \begin{bmatrix} B_{11} & \cdots & B_{1p} \\ \vdots & \ddots & \vdots \\ B_{p1} & \cdots & B_{pp} \end{bmatrix},$$

A_{rt}, B_{rt} are $n_r \times n_t$ blocks, i.e. $A_{11}, \ldots, A_{pp}$ have orders $n_1, \ldots, n_p$, resp..
For a square matrix $X = (X_{rt})$ partitioned according to $n_1, \ldots, n_p$, let

$$\tau(X) = \|X - \mathrm{diag}(X_{11}, \ldots, X_{pp})\|_F.$$

For our positive definite pair (A, B), let

$$\tau(A, B) = \left[\tau^2(A) + \tau^2(B)\right]^{1/2}$$

Multiple Eigenvalues

Theorem (Hari 91)

Let $D_r + E_r = A - \lambda_{s_r} B$, $diag(E_r) = 0$, $1 \leq r \leq p$. *If*

$$\|E_r\|_2 < \delta_r, \quad 1 \leq r \leq p,$$

then

$$\|A_{rr} - \lambda_{s_r} B_{rr}\|_F \leq \frac{1}{\delta_r} \sum_{\substack{t=1 \\ t \neq r}}^{p} \|A_{rt} - \lambda_{s_r} B_{rt}\|_F^2, \quad 1 \leq r \leq p$$

and

$$\sum_{s=1}^{n} \left| \frac{a_{ss}}{b_{ss}} - \lambda_s \right|^2 \leq \sum_{r=1}^{p} \|A_{rr} - \lambda_{s_r} B_{rr}\|_F^2 \leq \left[\frac{(1+\mu^2)\tau^2(A,B)}{\delta} \right]^2.$$

Multiple Eigenvalues

Let us return to the HZ method.

Multiple Eigenvalues

Let us return to the HZ method. Let (A, B) be obtained at step k.

Multiple Eigenvalues

Let us return to the HZ method. Let (A, B) be obtained at step k. Suppose that k is large enough, so that the last theorem holds for (A, B).

Multiple Eigenvalues

Let us return to the HZ method. Let (A, B) be obtained at step k. Suppose that k is large enough, so that the last theorem holds for (A, B). Let $\tau = \tau(A, B)$, $\epsilon = S(A, B)$.

Multiple Eigenvalues

Let us return to the HZ method. Let (A, B) be obtained at step k. Suppose that k is large enough, so that the last theorem holds for (A, B). Let $\tau = \tau(A, B)$, $\epsilon = S(A, B)$. Note that $\tau \leq \epsilon$.

Multiple Eigenvalues

Let us return to the HZ method. Let (A, B) be obtained at step k.
Suppose that k is large enough, so that the last theorem holds for (A, B).
Let $\tau = \tau(A, B)$, $\epsilon = S(A, B)$. Note that $\tau \leq \epsilon$.
Then the theorem implies

$$A_{rr} = \lambda_{s_r} B_{rr} + F_{rr}, \qquad \|F_r\|_F = \mathcal{O}(\tau^2), \quad 1 \leq r \leq p.$$

Multiple Eigenvalues

Let us return to the HZ method. Let (A, B) be obtained at step k.
Suppose that k is large enough, so that the last theorem holds for (A, B).
Let $\tau = \tau(A, B)$, $\epsilon = S(A, B)$. Note that $\tau \leq \epsilon$.
Then the theorem implies

$$A_{rr} = \lambda_{s_r} B_{rr} + F_{rr}, \qquad \|F_r\|_F = \mathcal{O}(\tau^2), \quad 1 \leq r \leq p.$$

If the pivot element a_{ij} (b_{ij}) lies within the diagonal block A_{rr} (B_{rr}), then we shall have:

Multiple Eigenvalues

Let us return to the HZ method. Let (A, B) be obtained at step k.
Suppose that k is large enough, so that the last theorem holds for (A, B).
Let $\tau = \tau(A, B)$, $\epsilon = S(A, B)$. Note that $\tau \leq \epsilon$.
Then the theorem implies

$$A_{rr} = \lambda_{s_r} B_{rr} + F_{rr}, \qquad \|F_r\|_F = \mathcal{O}(\tau^2), \quad 1 \leq r \leq p.$$

If the pivot element a_{ij} (b_{ij}) lies within the diagonal block A_{rr} (B_{rr}), then we shall have:

- Huge cancelations in the numerator and denominator when computing

$$\tan(2\theta) = \frac{2a_{ij} - (a_{ii} + a_{jj})\, b_{ij}}{\sqrt{1 - (b_{ij})^2}\,(a_{ii} - a_{jj})} = \frac{\mathcal{O}(\tau^2)}{\mathcal{O}(\tau^2)}$$

Multiple Eigenvalues

Let us return to the HZ method. Let (A, B) be obtained at step k.
Suppose that k is large enough, so that the last theorem holds for (A, B).
Let $\tau = \tau(A, B)$, $\epsilon = S(A, B)$. Note that $\tau \leq \epsilon$.
Then the theorem implies

$$A_{rr} = \lambda_{s_r} B_{rr} + F_{rr}, \qquad \|F_r\|_F = \mathcal{O}(\tau^2), \quad 1 \leq r \leq p.$$

If the pivot element a_{ij} (b_{ij}) lies within the diagonal block A_{rr} (B_{rr}), then we shall have:

- Huge cancelations in the numerator and denominator when computing

$$\tan(2\theta) = \frac{2a_{ij} - (a_{ii} + a_{jj})\, b_{ij}}{\sqrt{1 - (b_{ij})^2}\,(a_{ii} - a_{jj})} = \frac{\mathcal{O}(\tau^2)}{\mathcal{O}(\tau^2)}$$

- Possibly large θ when ϵ and τ are tiny.

Multiple Eigenvalues

Let us return to the HZ method. Let (A, B) be obtained at step k. Suppose that k is large enough, so that the last theorem holds for (A, B). Let $\tau = \tau(A, B)$, $\epsilon = S(A, B)$. Note that $\tau \leq \epsilon$.
Then the theorem implies

$$A_{rr} = \lambda_{s_r} B_{rr} + F_{rr}, \qquad \|F_r\|_F = \mathcal{O}(\tau^2), \quad 1 \leq r \leq p.$$

If the pivot element a_{ij} (b_{ij}) lies within the diagonal block A_{rr} (B_{rr}), then we shall have:

- Huge cancelations in the numerator and denominator when computing

$$\tan(2\theta) = \frac{2a_{ij} - (a_{ii} + a_{jj})\, b_{ij}}{\sqrt{1 - (b_{ij})^2}\,(a_{ii} - a_{jj})} = \frac{\mathcal{O}(\tau^2)}{\mathcal{O}(\tau^2)}$$

- Possibly large θ when ϵ and τ are tiny.

This impacts asymptotic convergence and accuracy of the algorithm.

Multiple Eigenvalues

$$N = \frac{n(n-1)}{2}, \qquad M = N - \sum_{r=1}^{p} \frac{n_r(n_r-1)}{2}, \qquad n_{max} = \max_{1 \le r \le p} n_r$$

Let ϵ_N and τ_N denote ϵ and τ for the pair obtained after applying one sweep of the column-cyclic HZ method. If (A, B) satisfies $n \ge 3$, $p \ge 2$,

$$S(B) < \frac{1}{n(n-1)}, \quad \sqrt{1+\mu^2}\epsilon < \min\left\{\frac{1}{2}, \sqrt{\frac{\delta}{\mu+1}}\right\}\delta,$$

then

- $\tau_N \le \frac{3}{2}\sqrt{2.31^M \cdot n_{max}(1+\mu^2)}\,\frac{\epsilon}{\delta}\tau$
- $\tau_N \le \frac{3}{2}\sqrt{n_{max}(1+\mu^2)}\,\frac{\epsilon^2}{\delta}$
- if $n_{max} = 2$ then $\epsilon_N \le \frac{18}{17}\sqrt{1+\mu^2}\frac{\epsilon^2}{\delta}$.

Stability and High Relative Accuracy

- We are interested in how accurate are the methods: HZ, $LL^T J$ and $RR^T J$.

Stability and High Relative Accuracy

- We are interested in how accurate are the methods: HZ, $LL^T J$ and $RR^T J$.
- For that we need a detailed error analysis. J. Matejaš and V. Hari have made one although the paper is not yet completed.

Stability and High Relative Accuracy

- We are interested in how accurate are the methods: HZ, $LL^T J$ and $RR^T J$.
- For that we need a detailed error analysis. J. Matejaš and V. Hari have made one although the paper is not yet completed.
- For such an analysis we would need another seminar like this one, so here we shall present just the results of numerical tests on the accuracy of those methods.

Stability and High Relative Accuracy

- We are interested in how accurate are the methods: HZ, $LL^T J$ and $RR^T J$.
- For that we need a detailed error analysis. J. Matejaš and V. Hari have made one although the paper is not yet completed.
- For such an analysis we would need another seminar like this one, so here we shall present just the results of numerical tests on the accuracy of those methods.
- Hence we first present the algorithms, then theoretical background for the tests and then the results.

Stability and High Relative Accuracy

- We are interested in how accurate are the methods: HZ, $LL^T J$ and $RR^T J$.
- For that we need a detailed error analysis. J. Matejaš and V. Hari have made one although the paper is not yet completed.
- For such an analysis we would need another seminar like this one, so here we shall present just the results of numerical tests on the accuracy of those methods.
- Hence we first present the algorithms, then theoretical background for the tests and then the results.
- One can hope for high relative accuracy of the methods only for well-behaved initial pairs (A, B).

Stability and High Relative Accuracy

- We are interested in how accurate are the methods: HZ, LL^TJ and RR^TJ.
- For that we need a detailed error analysis. J. Matejaš and V. Hari have made one although the paper is not yet completed.
- For such an analysis we would need another seminar like this one, so here we shall present just the results of numerical tests on the accuracy of those methods.
- Hence we first present the algorithms, then theoretical background for the tests and then the results.
- One can hope for high relative accuracy of the methods only for well-behaved initial pairs (A, B).
- An example of such pairs are the pairs of positive definite symmetric matrices which can be well symmetrically scaled. These are the pairs for which the conditions $\kappa_2(\Delta_A A \Delta_A)$ and $\kappa_2(\Delta_B B \Delta_B)$ are small for some diagonal matrices Δ_A and Δ_B.

Algorithm HZ

select the pivot pair (i, j)
if $a_{ij} \neq 0$ or $b_{ij} \neq 0$ then

$$\rho = 0.5\,(\sqrt{1+b_{ij}} + \sqrt{1-b_{ij}}); \quad \xi = b_{ij}/(2\rho);$$
$$\tau = \sqrt{(1+b_{ij})(1-b_{ij})}; \quad t2 = 2a_{ij} - (a_{ii}+a_{jj})b_{ij};$$

if $t2 = 0$ then $t = 0$;
else

$$ct2 = \tau\,(a_{ii} - a_{jj})/t2;$$
$$t = \mathtt{sign}(ct2)/(\mathtt{abs}(ct2) + (1 + \sqrt{1+ct2^2});$$

end

$$cs = 1/\sqrt{1+t^2}; \qquad sn = t/\sqrt{1+t^2};$$
$$c1 = (\rho \cdot cs - \xi \cdot sn)/\tau; \qquad s1 = (\rho \cdot sn + \xi \cdot cs)/\tau;$$
$$c2 = (\rho \cdot cs + \xi \cdot sn)/\tau; \qquad s2 = (\rho \cdot sn - \xi \cdot cs)/\tau;$$
$$\delta_i = (b_{ij}/\tau - s1)(b_{ij}/\tau + s1)a_{ii} + (2c1\,a_{ij} + s2\,a_{jj})\,s2;$$
$$\delta_j = (s2 - b_{ij}/\tau)(s2 + b_{ij}/\tau)\,a_{jj} + (2c2\,a_{ij} - s1\,a_{ii})\,s1;$$
$$a'_{ij} = (c1\,c2 - s1\,s2)a_{ij} + (c2\,s2\,a_{jj} - c1\,s1\;a_{ii}); \quad a'_{ji} = a'_{ij};$$
$$b'_{ij} = 0; \quad b'_{ji} = b'_{ij}; \qquad a'_{ii} = a_{ii} + \delta_i; \quad a'_{jj} = a_{jj} - \delta_j;$$

for $k = 1, \ldots, n,\ k \neq i, j$ do

$$a'_{ki} = c1 \cdot a_{ki} + s2 \cdot a_{kj}; \quad b'_{ki} = c1 \cdot b_{ki} + s2 \cdot b_{kj}; \quad a'_{ik} = a'_{ki}; \quad b'_{ik} = b'_{ki};$$
$$a'_{kj} = c2 \cdot a_{kj} - s1 \cdot a_{ki}; \quad b'_{kj} = c2 \cdot b_{kj} - s1 \cdot b_{ki}; \quad a'_{jk} = a'_{kj}; \quad b'_{jk} = b'_{kj};$$

endfor

endif

Algorithm $LL^T J$

```
select the pivot pair (i, j)
if a_ij ≠ 0 or b_ij ≠ 0 then
```

$\beta = b_{ij}\,,\ \tau = \mathtt{sqrt}((1+\beta)(1-\beta));\quad \alpha = a_{ij} - \beta a_{ii}\,;$

$\quad$ `if` $\ \alpha = 0\ $ `then` $\ t = 0;$

$\quad$ `else` $\ ct2 = (0.5\,(a_{ii} - a_{jj}) + \alpha\beta)/(\alpha\,\tau);$

$\qquad t = \mathtt{sign}(ct2)/(\mathtt{abs}(ct2) + \mathtt{sqrt}(1 + ct2^2));$

$\quad$ `endif`

$\quad cs = 1/\mathtt{sqrt}(1+t^2);\quad sn = t/\mathtt{sqrt}(1+t^2);$

$\quad c1 = cs - sn\,\beta/\tau\,;\quad s1 = sn + cs\,\beta/\tau\,;\quad c2 = cs/\tau\,;\quad s2 = sn/\tau\,;$

$\quad \delta_i = t\alpha/\tau\,;\quad \delta_j = (\,t\alpha + (\beta/\tau)\cdot(2a_{ij} - (a_{ii}+a_{jj})\beta)\,)/\tau\,;$

$\quad a'_{ij} = (c1\,c2 - s1\,s2)\,a_{ij} + (c2\,s2\,a_{jj} - c1\,s1\,a_{ii});\quad a'_{ji} = a'_{ij}\,;$

$\quad b'_{ij} = (c1\,c2 - s1\,s2)\,\beta + (c2\,s2 - c1\,s1);\quad b'_{ji} = b'_{ij}\,;$

$\quad a'_{ii} = a_{ii} + \delta_i\,;\quad a'_j = a_{jj} - \delta_j\,;$

$\quad$ `for` $\ k = 1,\ldots,n,\ k \neq i,j\ $ `do`

$\qquad a'_{ki} = c1\cdot a_{ki} + s2\cdot a_{kj};\quad b'_{ki} = c1\cdot b_{ki} + s2\cdot b_{kj};\quad a'_{ik} = a'_{ki};\quad b'_{ik} = b'_{ki}$

$\qquad a'_{kj} = c2\cdot a_{kj} - s1\cdot a_{ki};\quad b'_{kj} = c2\cdot b_{kj} - s1\cdot b_{ki};\quad a'_{jk} = a'_{kj};\quad b'_{jk} = b'_{kj}\,;$

$\quad$ `endfor`

`endif`

Algorithm RR^TJ

```
select the pivot pair (i, j)
if a_ij ≠ 0 or b_ij ≠ 0 then
    β = b_ij,  τ = sqrt((1 + β)(1 − β));   α = a_ij − β a_jj;
    if  α = 0   then  t = 0;
    else  ct2 = (0.5 (a_ii − a_jj) − αβ)/(α τ);
              t = sign(ct2)/(abs(ct2) + sqrt(1 + ct2^2));
    endif
    cs = 1/sqrt(1 + t^2);   sn = t/sqrt(1 + t^2);
    c1 = cs/τ;   s1 = sn/τ;   c2 = cs + sn β/τ;   s2 = sn − cs β/τ;
    δ_j = tα/τ;   δ_i = ( tα − (β/τ) · (2a_ij − (a_ii + a_jj)β) )/τ;
    a'_ij = (c1 c2 − s1 s2) a_ij + (c2 s2 a_jj − c1 s1 a_ii);   a'_ji = a'_ij;
    b'_ij = (c1 c2 − s1 s2) β + (c2 s2 − c1 s1);   b'_ji = b'_ij;
    a'_ii = a_ii + δ_i;   a'_j = a_jj − δ_j;
    for k = 1, ..., n, k ≠ i, j  do
      a'_ki = c1 · a_ki + s2 · a_kj;   b'_ki = c1 · b_ki + s2 · b_kj;   a'_ik = a'_ki;   b'_ik = b'_ki
      a'_kj = c2 · a_kj − s1 · a_ki;   b'_kj = c2 · b_kj − s1 · b_ki;   a'_jk = a'_kj;   b'_jk = b'_kj;
    endfor
endif
```

Theorem (Theorem 3.2, Drmač 1998)

Let $A = A^T \succ O$, $B = B^T \succ O$ *and* $\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_n$ *be the eigenvalues of the pair* (A, B).
Let $A_S = D_A^{-1/2} A D_A^{-1/2}$, $B_S = D_B^{-1/2} B D_B^{-1/2}$, $D_A = diag(A)$, $D_B = diag(B)$.
Let δA *and* δB *be symmetric perturbations such that*

$$\|(\delta A)_S\|_2 \|A_S^{-1}\|_2 < 1 \quad \text{and} \quad \|(\delta B)_S\|_2 \|B_S^{-1}\|_2 < 1,$$

where $(\delta A)_S = D_A^{-1/2} \delta A D_A^{-1/2}$, $(\delta B)_S = D_B^{-1/2} \delta B D_B^{-1/2}$.
If $\tilde{\lambda}_1 \geq \tilde{\lambda}_2 \geq \cdots \geq \tilde{\lambda}_n$ *are th eigenvalues of* $(A + \delta A, B + \delta B)$, *then*

$$\max_{1 \leq i \leq n} \frac{|\tilde{\lambda}_i - \lambda_i|}{\lambda_i} \leq \frac{\|(\delta A)_S\|_2 \|A_S^{-1}\|_2 + \|(\delta B)_S\|_2 \|B_S^{-1}\|_2}{1 - \|(\delta B)_S\|_2 \|B_S^{-1}\|_2} = \frac{\varepsilon_{A_S} \kappa_2(A_S) + \varepsilon_{B_S} \kappa_2(B_S)}{1 - \varepsilon_{B_S} \kappa_2(B_S)},$$

where $\varepsilon_{A_S} = \|(\delta A)_S\|_2 / \|A_S\|_2$, $\varepsilon_{B_S} = \|(\delta B)_S\|_2 / \|B_S\|_2$, *and* $\kappa_2(X)$ *is the spectral condition number of* X.

Theoretical Background

- For all considered methods the starting matrix $B^{(0)}$ is just B_S. Therefore

$$\max_{1\leq i\leq n} \frac{|\tilde{\lambda}_i - \lambda_i|}{\lambda_i} \leq \frac{\varepsilon_{A_S}\kappa_2(A_S) + \varepsilon_{B^{(0)}}\kappa_2(B_S)}{1 - \varepsilon_{B_S}\kappa_2(B^{(0)})},$$

Theoretical Background

- For all considered methods the starting matrix $B^{(0)}$ is just B_S. Therefore

$$\max_{1\le i\le n} \frac{|\tilde{\lambda}_i - \lambda_i|}{\lambda_i} \le \frac{\varepsilon_{A_S}\kappa_2(A_S) + \varepsilon_{B^{(0)}}\kappa_2(B_S)}{1 - \varepsilon_{B_S}\kappa_2(B^{(0)})},$$

- The initial normalization $B \mapsto B_S = B^{(0)}$, simplifies the algorithm. Moreover, it has a stabilizing effect on the iterative process, because it almost optimally reduces the condition of B and all $B^{(k)}$, $k \ge 1$ will have almost best possible conditions. Van der Sluis, A.: Condition numbers and equilibration of matrices. Numer. Math. 14 (1), 14–23 (1969)

Theoretical Background

- For all considered methods the starting matrix $B^{(0)}$ is just B_S. Therefore

$$\max_{1 \le i \le n} \frac{|\tilde{\lambda}_i - \lambda_i|}{\lambda_i} \le \frac{\varepsilon_{A_S} \kappa_2(A_S) + \varepsilon_{B^{(0)}} \kappa_2(B_S)}{1 - \varepsilon_{B_S} \kappa_2(B^{(0)})},$$

- The initial normalization $B \mapsto B_S = B^{(0)}$, simplifies the algorithm. Moreover, it has a stabilizing effect on the iterative process, because it almost optimally reduces the condition of B and all $B^{(k)}$, $k \ge 1$ will have almost best possible conditions. Van der Sluis, A.: Condition numbers and equilibration of matrices. Numer. Math. 14 (1), 14–23 (1969)
- For these well-behaved pairs we have to find out the methods which generate at every step only tiny relative errors $\varepsilon_{A_S^{(k)}}$, $\varepsilon_{B_S^{(k)}}$ and in the same time matrices with small or modest condition numbers $\kappa_2(A_S^{(k)})$ and $\kappa_2(B^{(k)})$.

Theoretical Background

- For all considered methods the starting matrix $B^{(0)}$ is just B_S. Therefore

$$\max_{1\le i\le n} \frac{|\tilde{\lambda}_i - \lambda_i|}{\lambda_i} \le \frac{\varepsilon_{A_S}\kappa_2(A_S) + \varepsilon_{B^{(0)}}\kappa_2(B_S)}{1 - \varepsilon_{B_S}\kappa_2(B^{(0)})},$$

- The initial normalization $B \mapsto B_S = B^{(0)}$, simplifies the algorithm. Moreover, it has a stabilizing effect on the iterative process, because it almost optimally reduces the condition of B and all $B^{(k)}$, $k \ge 1$ will have almost best possible conditions. Van der Sluis, A.: Condition numbers and equilibration of matrices. Numer. Math. 14 (1), 14–23 (1969)
- For these well-behaved pairs we have to find out the methods which generate at every step only tiny relative errors $\varepsilon_{A_S^{(k)}}$, $\varepsilon_{B_S^{(k)}}$ and in the same time matrices with small or modest condition numbers $\kappa_2(A_S^{(k)})$ and $\kappa_2(B^{(k)})$.

Nonetheless, this is a demanding task, so we shall go for a shortcut.

How to detect the high relative accuracy of a method?

How to detect the high relative accuracy of a method?

We can check numerically whether the inequality

$$\varrho_{(A,B)} = \max_{1\le i\le n} \frac{|\tilde{\lambda}_i - \lambda_i|}{\lambda_i} \Big/ \sqrt{\kappa_2^2(A_S^{(0)}) + \kappa_2^2(B^{(0)})} \le f(n)\mathbf{u}, \qquad (5)$$

holds for a larger sample Υ of the initial well-behaved pairs (A, B)!

How to detect the high relative accuracy of a method?

We can check numerically whether the inequality

$$\varrho_{(A,B)} = \max_{1\le i\le n} \frac{|\tilde{\lambda}_i - \lambda_i|}{\lambda_i} \Big/ \sqrt{\kappa_2^2(A_S^{(0)}) + \kappa_2^2(B^{(0)})} \le f(n)\mathbf{u}, \qquad (5)$$

holds for a larger sample Υ of the initial well-behaved pairs (A, B)! Here

- $\tilde{\lambda}_i$, $1 \le i \le n$ are the eigenvalues of the starting pair $(A^{(0)}, B^{(0)})$

How to detect the high relative accuracy of a method?

We can check numerically whether the inequality

$$\varrho_{(A,B)} = \max_{1\leq i\leq n} \frac{|\tilde{\lambda}_i - \lambda_i|}{\lambda_i} / \sqrt{\kappa_2^2(A_S^{(0)}) + \kappa_2^2(B^{(0)})} \leq f(n)\mathbf{u}, \qquad (5)$$

holds for a larger sample Υ of the initial well-behaved pairs (A, B)! Here

- $\tilde{\lambda}_i$, $1 \leq i \leq n$ are the eigenvalues of the starting pair $(A^{(0)}, B^{(0)})$
- $f(n)$ is a slowly growing function of n

How to detect the high relative accuracy of a method?

We can check numerically whether the inequality

$$\varrho_{(A,B)} = \max_{1 \le i \le n} \frac{|\tilde{\lambda}_i - \lambda_i|}{\lambda_i} / \sqrt{\kappa_2^2(A_S^{(0)}) + \kappa_2^2(B^{(0)})} \le f(n)\mathbf{u}, \qquad (5)$$

holds for a larger sample Υ of the initial well-behaved pairs (A, B)! Here

- $\tilde{\lambda}_i$, $1 \le i \le n$ are the eigenvalues of the starting pair $(A^{(0)}, B^{(0)})$
- $f(n)$ is a slowly growing function of n
- $\mathbf{u}$ is the round off unit

How to detect the high relative accuracy of a method?

We can check numerically whether the inequality

$$\varrho_{(A,B)} = \max_{1\leq i\leq n} \frac{|\tilde{\lambda}_i - \lambda_i|}{\lambda_i} / \sqrt{\kappa_2^2(A_S^{(0)}) + \kappa_2^2(B^{(0)})} \leq f(n)\mathbf{u}, \qquad (5)$$

holds for a larger sample Υ of the initial well-behaved pairs (A, B)! Here

- $\tilde{\lambda}_i$, $1 \leq i \leq n$ are the eigenvalues of the starting pair $(A^{(0)}, B^{(0)})$
- $f(n)$ is a slowly growing function of n
- $\mathbf{u}$ is the round off unit
- The relation (5) should hold irrespectively of how large is the condition $\kappa_2(A^{(0)})$. Therefore, we are interested in how $\varrho_{(A,B)}$ behaves with respect to $\chi_{(A,B)}$,

$$\chi_{(A,B)} := \kappa_2(A^{(0)}, B^{(0)}) = \sqrt{\kappa_2^2(A^{(0)}) + \kappa_2^2(B^{(0)})}.$$

- For the given sample of well behaved pairs Υ, and for each method, we shall make its graph of relative errors: $\mathcal{E}$,

$$\mathcal{E} = \{(\chi_{(A,B)}\ ,\ \varrho_{(A,B)})\ :\ (A,B) \in \Upsilon\}.$$

- For the given sample of well behaved pairs Υ, and for each method, we shall make its graph of relative errors: $\mathcal{E}$,

$$\mathcal{E} = \{(\chi_{(A,B)}\ ,\ \varrho_{(A,B)})\ :\ (A,B) \in \Upsilon\}.$$

- Then we shall depict that graph $\mathcal{E}$ by the M-function `scatter(x,y,3)`.

- For the given sample of well behaved pairs Υ, and for each method, we shall make its graph of relative errors: $\mathcal{E}$,

$$\mathcal{E} = \{(\chi_{(A,B)}\ ,\ \varrho_{(A,B)}) \,:\, (A,B) \in \Upsilon\}.$$

- Then we shall depict that graph $\mathcal{E}$ by the M-function `scatter(x,y,3)`.

- The method will be considered to be high relative accurate if the ordinates of the points on the graph are of order $\mathcal{O}(\mathbf{u})$ where $\mathbf{u} \approx 2.2 \cdot 10^{-16}$.

How to generate matrix pairs?

The starting pair $(A^{(0)}, B^{(0)})$ is generated by

- 4 the diagonal matrices : Δ_A, Δ_B, Σ, Δ and

How to generate matrix pairs?

The starting pair $(A^{(0)}, B^{(0)})$ is generated by

- 4 the diagonal matrices : Δ_A, Δ_B, Σ, Δ and
- 2 orthogonal matrices U, V of order n.

How to generate matrix pairs?

The starting pair $(A^{(0)}, B^{(0)})$ is generated by

- 4 the diagonal matrices : Δ_A, Δ_B, Σ, Δ and
- 2 orthogonal matrices U, V of order n.

It is done in two steps:

1: $F = U\Sigma V^T, \quad A = F^T \Delta_A F, \quad B = F^T \Delta_B F,$

How to generate matrix pairs?

The starting pair $(A^{(0)}, B^{(0)})$ is generated by

- 4 the diagonal matrices : Δ_A, Δ_B, Σ, Δ and
- 2 orthogonal matrices U, V of order n.

It is done in two steps:

1: $F = U\Sigma V^T, \quad A = F^T \Delta_A F, \quad B = F^T \Delta_B F,$

2: $B^{(0)} = B_S = D_B^{-1/2} B D_B^{-1/2}, \quad A^{(0)} = \Delta A_S \Delta,\ A_S = D_A^{-1/2} A D_A^{-1/2},$

How to generate matrix pairs?

The starting pair $(A^{(0)}, B^{(0)})$ is generated by

- 4 the diagonal matrices : Δ_A, Δ_B, Σ, Δ and
- 2 orthogonal matrices U, V of order n.

It is done in two steps:

1: $F = U\Sigma V^T, \quad A = F^T \Delta_A F, \quad B = F^T \Delta_B F,$

2: $B^{(0)} = B_S = D_B^{-1/2} B D_B^{-1/2}, \quad A^{(0)} = \Delta A_S \Delta,\ A_S = D_A^{-1/2} A D_A^{-1/2},$

where D_A and D_B are the diagonal parts of A and B.

How to generate matrix pairs?

The starting pair $(A^{(0)}, B^{(0)})$ is generated by

- 4 the diagonal matrices : Δ_A, Δ_B, Σ, Δ and
- 2 orthogonal matrices U, V of order n.

It is done in two steps:

1: $F = U\Sigma V^T, \quad A = F^T \Delta_A F, \quad B = F^T \Delta_B F,$

2: $B^{(0)} = B_S = D_B^{-1/2} B D_B^{-1/2}, \quad A^{(0)} = \Delta A_S \Delta,\ A_S = D_A^{-1/2} A D_A^{-1/2},$

where D_A and D_B are the diagonal parts of A and B. Then $\kappa_2(A_S^{(0)})$ and $\kappa_2(B^{(0)})$ can be controlled by the diagonal elements of Δ_A, Δ_B, Σ,

How to generate matrix pairs?

The starting pair $(A^{(0)}, B^{(0)})$ is generated by

- 4 the diagonal matrices : Δ_A, Δ_B, Σ, Δ and
- 2 orthogonal matrices U, V of order n.

It is done in two steps:

1: $F = U\Sigma V^T, \quad A = F^T \Delta_A F, \quad B = F^T \Delta_B F,$

2: $B^{(0)} = B_S = D_B^{-1/2} B D_B^{-1/2}, \quad A^{(0)} = \Delta A_S \Delta,\ A_S = D_A^{-1/2} A D_A^{-1/2},$

where D_A and D_B are the diagonal parts of A and B. Then $\kappa_2(A_S^{(0)})$ and $\kappa_2(B^{(0)})$ can be controlled by the diagonal elements of Δ_A, Δ_B, Σ, since

$$\kappa_2(A_S^{(0)}) \leq n\kappa_2^2(\Sigma)\kappa_2(\Delta_A) \quad \text{and} \quad \kappa_2(B^{(0)}) \leq n\kappa_2^2(\Sigma)\kappa_2(\Delta_B),$$

although most often $\kappa_2(A_S^{(0)})$ and $\kappa_2(B^{(0)})$ are much smaller than these bounds.

How to generate matrix pairs?

To simplify the construction we set $\Delta_B = I_n$.

If the method is high relative accurate, then $\varrho_{(A,B)}$ from the relation (5) should not depend on $\kappa_2(\Delta)$.

How to generate matrix pairs?

To simplify the construction we set $\Delta_B = I_n$.

If the method is high relative accurate, then $\varrho_{(A,B)}$ from the relation (5) should not depend on $\kappa_2(\Delta)$.

Note that

$$\kappa_2(A^{(0)}) \leq \kappa_2(A_S^{(0)})\kappa_2^2(\Delta).$$

How to generate matrix pairs?

To simplify the construction we set $\Delta_B = I_n$.

If the method is high relative accurate, then $\varrho_{(A,B)}$ from the relation (5) should not depend on $\kappa_2(\Delta)$.

Note that

$$\kappa_2(A^{(0)}) \leq \kappa_2(A_S^{(0)})\kappa_2^2(\Delta).$$

If we set $\Delta = I_n$ i $(A^{(0)}, B^{(0)}) = (D_B^{-1/2} A D_B^{-1/2}, B_S)$, then we know in advance the eigenvalues of $(A^{(0)}, B^{(0)})$ These are the quotients

$$(\Delta_A)_{jj}/(\Delta_B)_{jj}, \qquad 1 \leq j \leq n.$$

This way can be used when considering behavior of the methods on pairs with multiple eigenvalues.

More Details

- Diagonal matrices are constructed by help of the M-function `diag(d)`

More Details

- Diagonal matrices are constructed by help of the M-function `diag(d)`
- `d` is a vector, and vectors are constructed by the M-function `logspace(x1,x2,n)`. We use it for the diagonal matrices Σ and Δ_A.

More Details

- Diagonal matrices are constructed by help of the M-function `diag(d)`
- `d` is a vector, and vectors are constructed by the M-function `logspace(x1,x2,n)`. We use it for the diagonal matrices Σ and Δ_A.
- For the construction of Δ we use our m-function

  `scalvec(k1,k2,k3,n,k)`

More Details

- Diagonal matrices are constructed by help of the M-function `diag(d)`
- `d` is a vector, and vectors are constructed by the M-function `logspace(x1,x2,n)`. We use it for the diagonal matrices Σ and Δ_A.
- For the construction of Δ we use our m-function

  `scalvec(k1,k2,k3,n,k)`

  which generates vector of length n, $d = [10^{\mathrm{k1}}, \ldots, 10^{\mathrm{k2}}, \ldots, 10^{\mathrm{k3}}]$ where `k` determines the position of 10^{k2} within the components of d.

More Details

- Diagonal matrices are constructed by help of the M-function `diag(d)`
- `d` is a vector, and vectors are constructed by the M-function `logspace(x1,x2,n)`. We use it for the diagonal matrices Σ and Δ_A.
- For the construction of Δ we use our m-function

  `scalvec(k1,k2,k3,n,k)`

  which generates vector of length n, $d = [10^{\mathtt{k1}}, \ldots, 10^{\mathtt{k2}}, \ldots, 10^{\mathtt{k3}}]$ where `k` determines the position of $10^{\mathtt{k2}}$ within the components of d.
- To compute Δ, the function `scalvec` is used within triple loop controlled by the indices `k1`, `k2` and `k3`

More Details

- Diagonal matrices are constructed by help of the M-function `diag(d)`
- `d` is a vector, and vectors are constructed by the M-function `logspace(x1,x2,n)`. We use it for the diagonal matrices Σ and Δ_A.
- For the construction of Δ we use our m-function

  `scalvec(k1,k2,k3,n,k)`

  which generates vector of length n, $d = [10^{\mathtt{k1}}, \ldots, 10^{\mathtt{k2}}, \ldots, 10^{\mathtt{k3}}]$ where `k` determines the position of $10^{\mathtt{k2}}$ within the components of d.
- To compute Δ, the function `scalvec` is used within triple loop controlled by the indices `k1`, `k2` and `k3`
- Orthogonal matrices U and V are computed by the command

  `[Q,~]=qr(rand(n))`

More Details

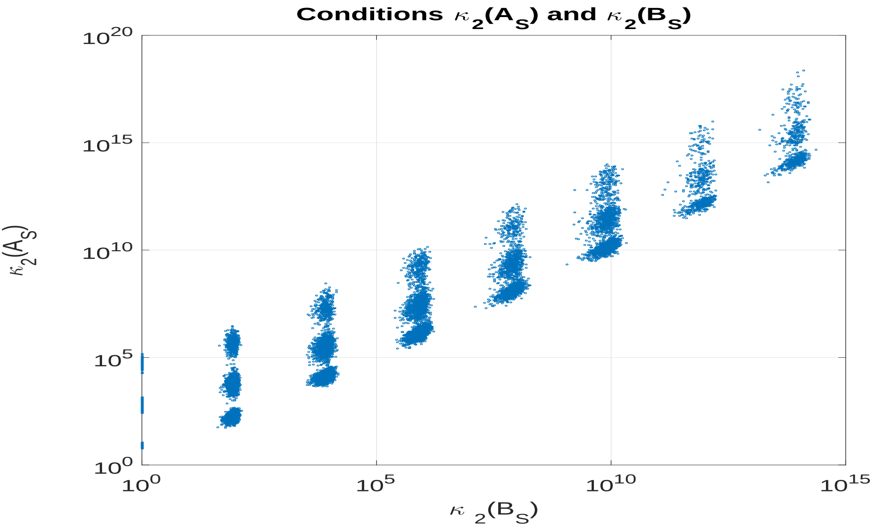
- Diagonal matrices are constructed by help of the M-function `diag(d)`
- `d` is a vector, and vectors are constructed by the M-function `logspace(x1,x2,n)`. We use it for the diagonal matrices Σ and Δ_A.
- For the construction of Δ we use our m-function

  `scalvec(k1,k2,k3,n,k)`

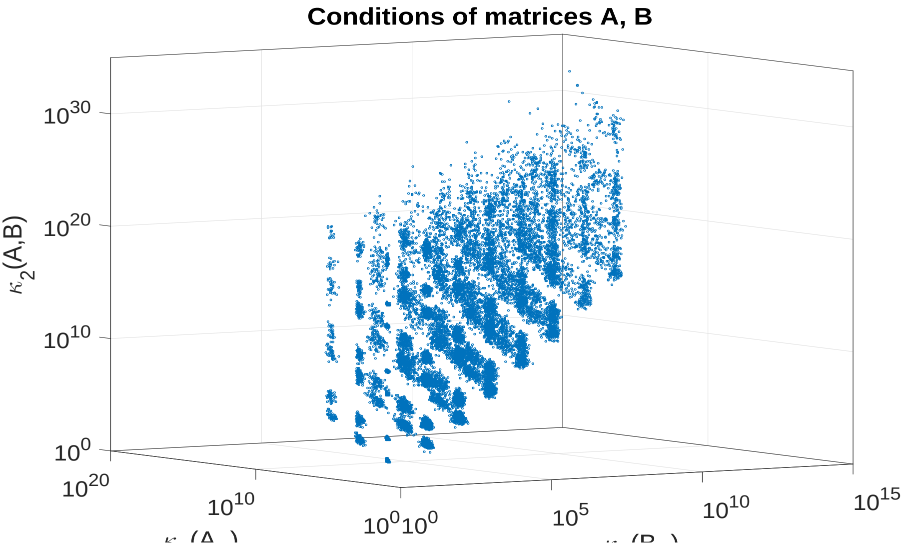
  which generates vector of length n, $d = [10^{\mathrm{k1}}, \ldots, 10^{\mathrm{k2}}, \ldots, 10^{\mathrm{k3}}]$ where `k` determines the position of 10^{k2} within the components of d.
- To compute Δ, the function `scalvec` is used within triple loop controlled by the indices `k1`, `k2` and `k3`
- Orthogonal matrices U and V are computed by the command

  `[Q,~]=qr(rand(n))`
- We have generated the sample Υ of 18900 pairs of matrices of order 10. As "exact eigenvalues" we have used the eigenvalues computed by the M-function `eig(A,B)` in variable precision arithmetic (VPA) using 80 decimal digits.

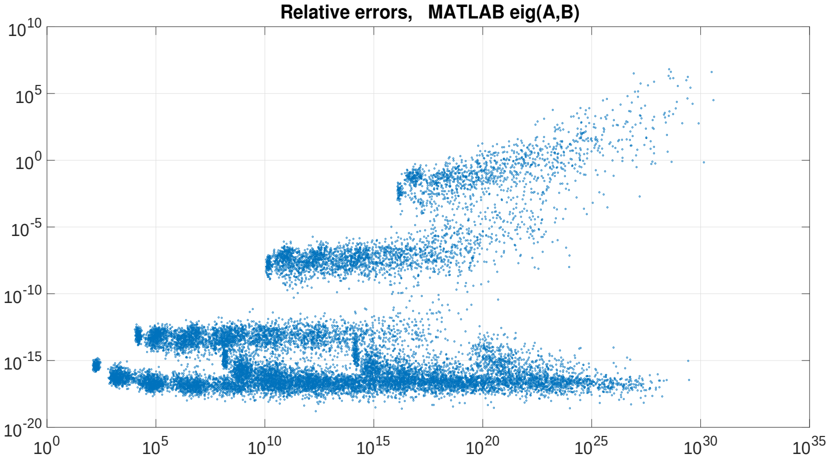
Matrix conditions



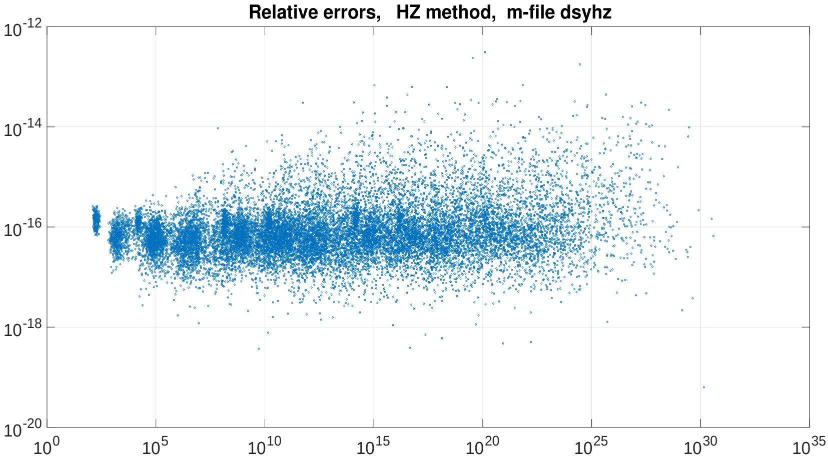
Matrix conditions



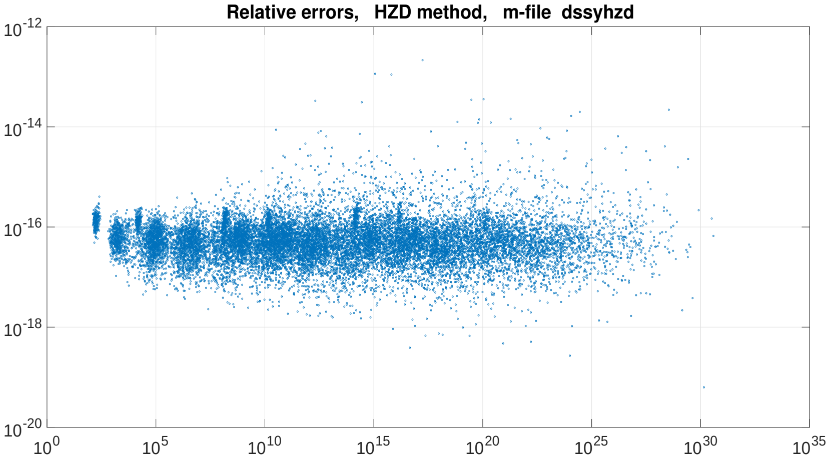
Relative errors: MATLAB eig function



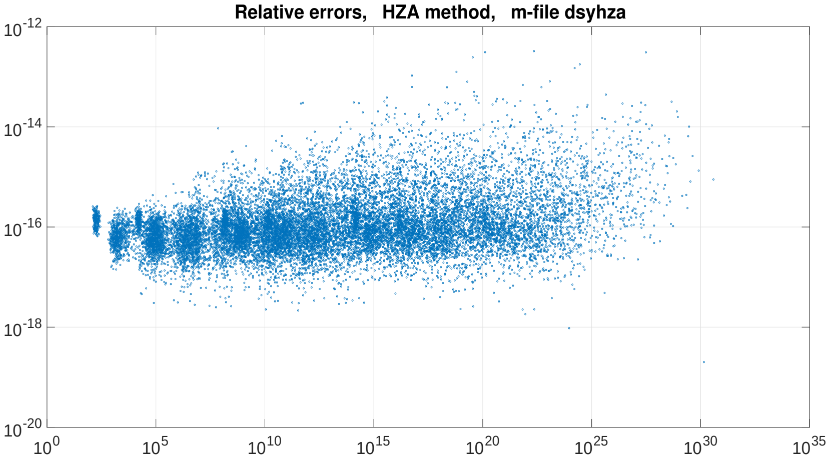
Relative errors: HZ method



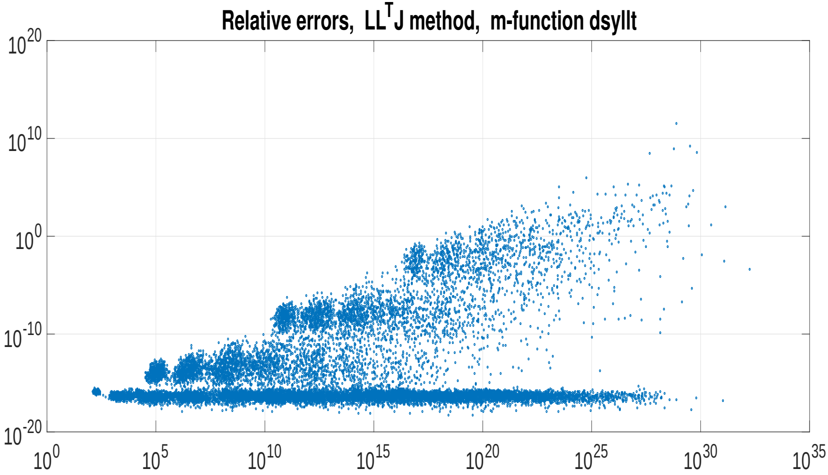
Relative errors: HZD method



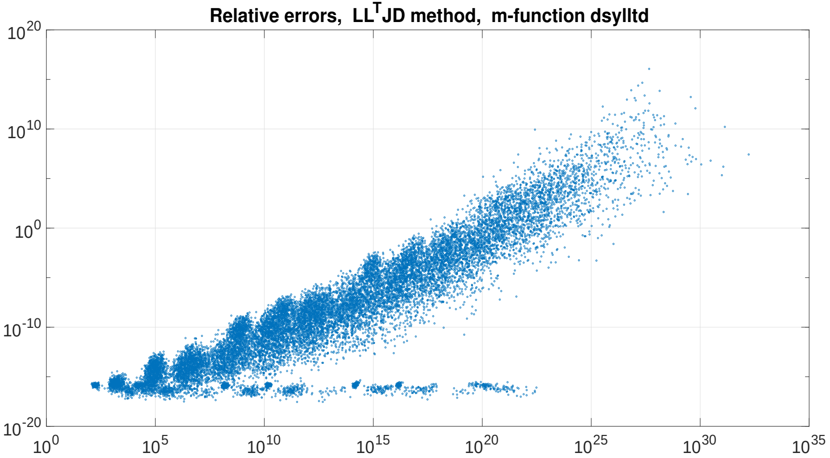
Relative errors: HZA method



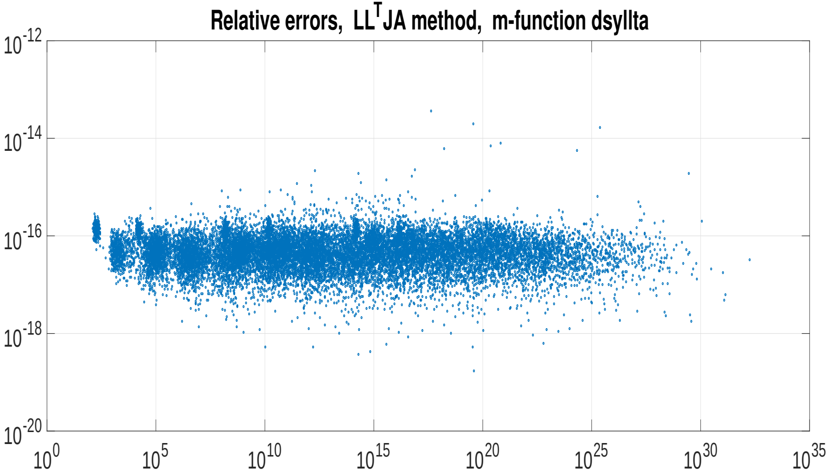
Relative errors: $LL^T J$ method



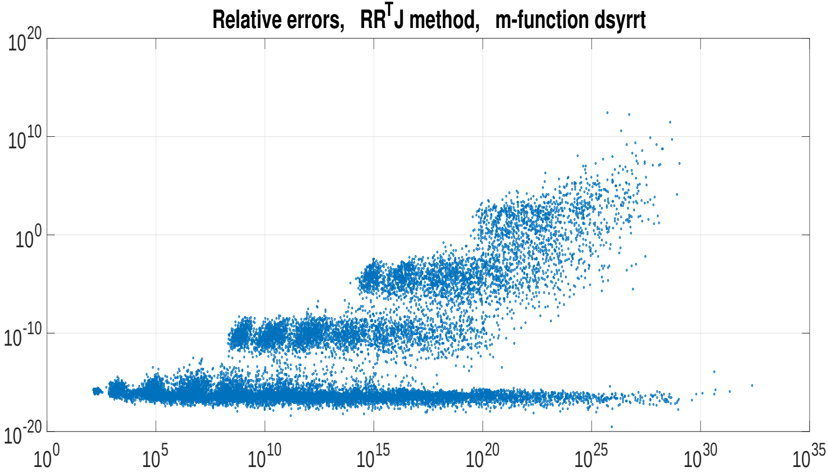
Relative errors: Descending LL^TJ method



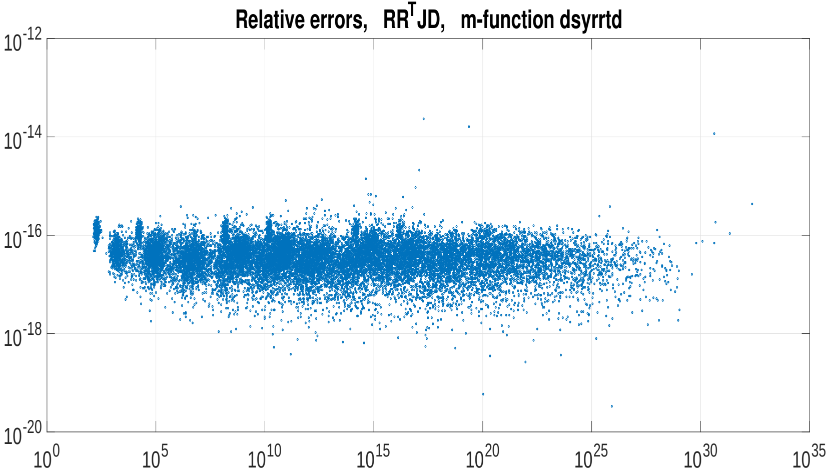
Relative errors: Ascending $LL^T J$ method



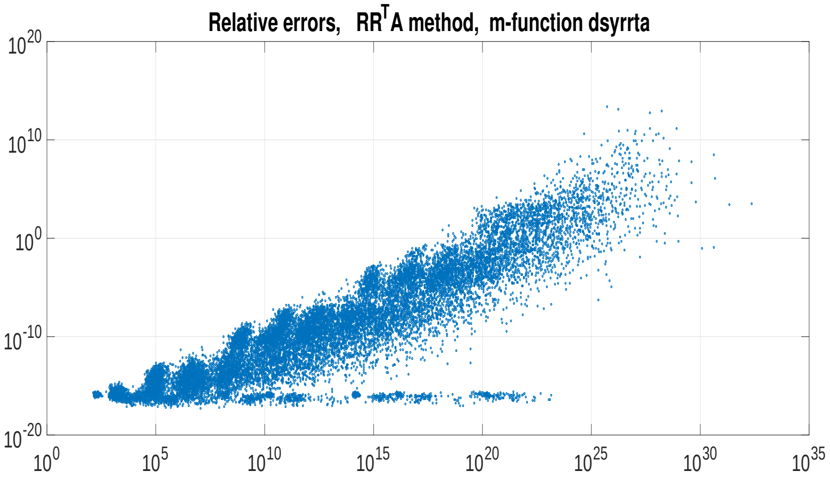
Relative errors: $RR^T J$ method



Relative errors: Descending RR^TJ method



Relative errors: Ascending RR^TJ method



How to define an accurate hybrid method?

We see that just one variant of $LL^T J$ method ($LL^T JA$) and just one variant of $RR^T J$ method ($RR^T JD$) is indicated as relatively accurate.

How to define an accurate hybrid method?

We see that just one variant of LL^TJ method (LL^TJA) and just one variant of RR^TJ method (RR^TJD) is indicated as relatively accurate.

This indicates how to a define highly accurate hybrid method, call it *Cholesky-Jacobi method* or shorter CJ method:

How to define an accurate hybrid method?

We see that just one variant of LL^TJ method (LL^TJA) and just one variant of RR^TJ method (RR^TJD) is indicated as relatively accurate.

This indicates how to a define highly accurate hybrid method, call it *Cholesky-Jacobi method* or shorter CJ method:

%%% Algorithm CJ

choose the pivot pair (i, j)

if $a_{ii} \geq a_{jj}$ **then** select LL^TJ algorithm
else select RR^TJ algorithm

endif

How to define an accurate hybrid method?

We see that just one variant of LL^TJ method (LL^TJA) and just one variant of RR^TJ method (RR^TJD) is indicated as relatively accurate.

This indicates how to a define highly accurate hybrid method, call it *Cholesky-Jacobi method* or shorter CJ method:

%%% Algorithm CJ

choose the pivot pair (i,j)
if $a_{ii} \geq a_{jj}$ **then** select LL^TJ algorithm
else select RR^TJ algorithm
endif

Its global convergence has been proved in an earlier theorem.

How to define an accurate hybrid method?

We see that just one variant of LL^TJ method (LL^TJA) and just one variant of RR^TJ method (RR^TJD) is indicated as relatively accurate.

This indicates how to a define highly accurate hybrid method, call it *Cholesky-Jacobi method* or shorter CJ method:

%%% Algorithm CJ

choose the pivot pair (i, j)
if $a_{ii} \geq a_{jj}$ **then** select LL^TJ algorithm
else select RR^TJ algorithm
endif

Its global convergence has been proved in an earlier theorem.

We complete our presentation with the graph associated with the CJ method.

Relative errors: *CJ* method