Kogbetliantz–like Method for the Hyperbolic SVD

Sanja Singer, Vedran Novaković

Faculty of Mechanical Engineering and Naval Architecture
University of Zagreb, Croatia

International Conference on Scientific Computing, SC2011
Santa Margherita di Pula, Sardinia, Italy
14th October 2011

Outline of the talk

Main topics:

- motivation for the construction of the hyperbolic SVD,
- the basics of the hyperbolic SVD,
- 2×2 matrices and their hyperbolic SVD,
- remaining problems and possible solutions,
- numerical examples.

Introduction

Modern eigenvalue algorithms need to be:

- **accurate** in the relative sense ("accurate"):

$$|\tilde{\lambda}_i - \lambda_i| \leq f(n)\varepsilon|\lambda_i|,$$

  where f is a slowly growing function of the matrix dimension n, for all eigenvalues λ_i, $\lambda_i \neq 0$.
- **fast** – comparable in speed with the "inaccurate" algorithms (algorithms accurate in absolute sense)

$$|\tilde{\lambda}_i - \lambda_i| \leq f(n)\varepsilon|\lambda_{\mathrm{max}}|.$$

Motivation — accurate eigenvalue computation

Common knowledge

- For general nonsymmetric matrices – we know almost nothing about accurate eigenvalue computation.
- For symmetric (Hermitian) positive definite matrices – eigenvalue computation is equivalent to the SVD of the full column rank (e.g. Cholesky) factor G (or SVD of G^*) of A. If

$$A = GG^* \quad \text{and} \quad G = U \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} V^* \quad \Longrightarrow \quad \lambda_i(A) = \sigma_i^2(G).$$

  This is the easiest case, with several accurate algorithms
  - the one-sided Jacobi algorithm,
  - the Kogbetliantz algorithm,
  - differential qd algorithm...

Motivation — accurate eigenvalue computation (cnt.)

Common knowledge

- For symmetric (Hermitian) indefinite matrices – eigenvalue computation is equivalent to hyperbolic SVD (HSVD) of the Hermitian indefinite factor G of $A = GJG^*$, where $J = \mathrm{diag}(\pm 1)$ is a signature matrix.
  If $G \in \mathbb{C}^{m\times n}$, $m \geq n$ is of full column rank then
  $$G = U \begin{bmatrix} \Sigma \\ 0 \end{bmatrix} V^*,$$
  where $U \in \mathbb{C}^{m\times m}$ is unitary, Σ diagonal with nonnegative elements, and $V \in \mathbb{C}^{n\times n}$ is J-unitary, i.e., $V^*JV = J$.
  If $A = GJG^*$ is given by G and J then HSVD of G implies
  $$G = U\Sigma V^*, \quad V^*JV = J \quad \Longrightarrow \quad \lambda_i(A) = \sigma_i^2(G)J_{ii}.$$

The one–sided hyperbolic Jacobi algorithm

Accurate algorithm for the HSVD

1. Optional first step: if A is given, A is factored by the Hermitian indefinite factorization (Bunch, Parlett ('71)) to obtain full column rank factor G:

   $$A = GJG^*.$$

   Spectrum of A = spectrum of the matrix pair (G^*G, J).
2. The matrix pair (G^*G, J) is simultaneously diagonalized (Veselić ('93)) by
   - ordinary trigonometric rotations (signs in J equal), or
   - hyperbolic rotations (signs in J different).

   This diagonalization is performed implicitly—as the one-sided algorithm.

The one–sided hyperbolic Jacobi algorithm (cnt.)

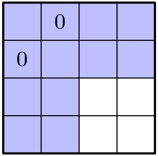
The sines/cosines of the angles are computed from the pair (G^*G, J), but applied from the right-hand side on G.

For example, if

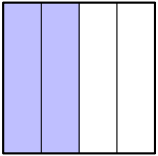
$$J = \mathrm{diag}(1, -1, 1, -1)$$

and the strategy is row-cyclic

alg. on G^*G:



alg. on G:



Diagonalization of a pivot block in G^*G is equivalent to orthogonalization of the two columns in G.

The one–sided hyperbolic Jacobi algorithm (cnt.)

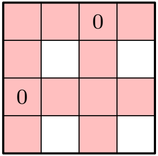
The sines/cosines of the angles are computed from the pair (G^*G, J), but applied from the right-hand side on G.

For example, if

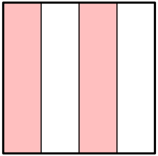
$$J = \mathrm{diag}(1, -1, 1, -1)$$

and the strategy is row-cyclic

alg. on G^*G:



alg. on G:



Diagonalization of a pivot block in G^*G is equivalent to orthogonalization of the two columns in G.

The one–sided hyperbolic Jacobi algorithm (cnt.)

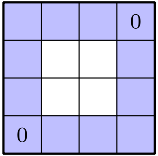
The sines/cosines of the angles are computed from the pair (G^*G, J), but applied from the right-hand side on G.

For example, if

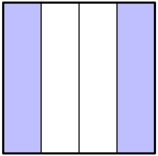
$$J = \mathrm{diag}(1, -1, 1, -1)$$

and the strategy is row-cyclic

alg. on G^*G:



alg. on G:



Diagonalization of a pivot block in G^*G is equivalent to orthogonalization of the two columns in G.

The one–sided hyperbolic Jacobi algorithm (cnt.)

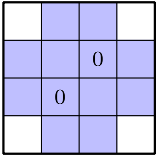
The sines/cosines of the angles are computed from the pair (G^*G, J), but applied from the right-hand side on G.

For example, if

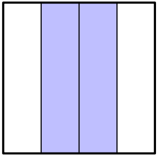
$$J = \mathrm{diag}(1, -1, 1, -1)$$

and the strategy is row-cyclic

alg. on G^*G:



alg. on G:



Diagonalization of a pivot block in G^*G is equivalent to orthogonalization of the two columns in G.

The one–sided hyperbolic Jacobi algorithm (cnt.)

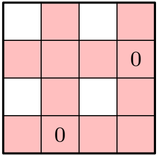
The sines/cosines of the angles are computed from the pair (G^*G, J), but applied from the right-hand side on G.

For example, if

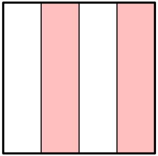
$$J = \mathrm{diag}(1, -1, 1, -1)$$

and the strategy is row-cyclic

alg. on G^*G:



alg. on G:



Diagonalization of a pivot block in G^*G is equivalent to orthogonalization of the two columns in G.

The one–sided hyperbolic Jacobi algorithm (cnt.)

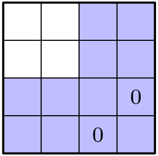
The sines/cosines of the angles are computed from the pair (G^*G, J), but applied from the right-hand side on G.

For example, if

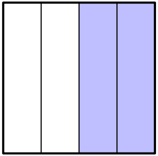
$$J = \mathrm{diag}(1, -1, 1, -1)$$

and the strategy is row-cyclic

alg. on G^*G:



alg. on G:



Diagonalization of a pivot block in G^*G is equivalent to orthogonalization of the two columns in G.

The Kogbetliantz algorithm

Accurate algorithm for the SVD

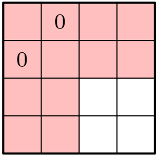
1. If it is used for the eigenvalue computation, matrix A should be factored by the Cholesky factorization as $A = GG^*$.
2. Matrix G is diagonalized (directly, i.e., two-sided) by ordinary trigonometric rotations from both left and right, but with different angles, φ and ψ.
3. If the matrix G is symmetric, the Kogbetliantz algorithm is just the ordinary two-sided Jacobi eigenvalue algorithm, with $\varphi = \psi$.
4. The initial matrix G is usually preprocessed, to be "more diagonal", by one or two QR factorizations.

The Kogbetliantz algorithm (cnt.)

The sines/cosines of the angles are computed directly from G.

For example, if and the strategy is row-cyclic, and the matrix is upper triangular,

alg. on G:

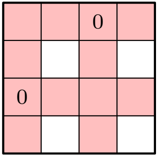


The Kogbetliantz algorithm (cnt.)

The sines/cosines of the angles are computed directly from G.

For example, if and the strategy is row-cyclic, and the matrix is upper triangular,

alg. on G:

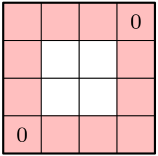


The Kogbetliantz algorithm (cnt.)

The sines/cosines of the angles are computed directly from G.

For example, if and the strategy is row-cyclic, and the matrix is upper triangular,

alg. on G:

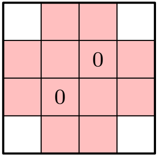


The Kogbetliantz algorithm (cnt.)

The sines/cosines of the angles are computed directly from G.

For example, if and the strategy is row-cyclic, and the matrix is upper triangular,

alg. on G:

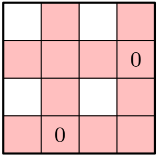


The Kogbetliantz algorithm (cnt.)

The sines/cosines of the angles are computed directly from G.

For example, if and the strategy is row-cyclic, and the matrix is upper triangular,

alg. on G:

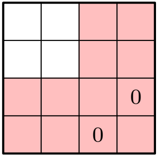


The Kogbetliantz algorithm (cnt.)

The sines/cosines of the angles are computed directly from G.

For example, if and the strategy is row-cyclic, and the matrix is upper triangular,

alg. on G:



Properties of the one-sided Jacobi algorithm

Favorable properties

1. Very accurate and fairly simple.
2. Very fast, provided all the tricks are used: `dgejsv` (Drmač).
3. Ideal for parallelization.
4. Can be generalized to work with the block-columns.
5. Output: matrix $\widehat{G} = U\Sigma$, accumulation of the eigenvectors unnecesary.

Shortcomings

1. It destroys the initial almost diagonality/triangularity of G.
2. In the final stages of the process, there are huge cancelations in computing the rotation parameters (dot products of almost orthogonal vectors).
3. Checking for convergence is very expensive.

Properties of the Kogbetliantz algorithm

Favorable properties

1. It further diagonalizes the starting almost diagonal triangular matrix (it preserves the triangular form).
2. It has very cheap and sound stopping criterion.
3. It is relatively accurate (Hari–Matejaš).
4. Some tricks can be borrowed from the one-sided Jacobi.
5. Algorithm can be parallelized (Hari–Zadelj-Martić).
6. Block version of the method can be designed (Bujanović).

Shortcomings

1. Algorithm is slower: transforms both rows and columns.
2. Less freedom in choosing the pivot strategy.
3. Eigenvector computation needs additional storage.

The algorithms

	one sided	two-sided
trigonometric	Jacobi	Kogbetliantz
hyperbolic	Jacobi	missing

Fill the missing algorithm

- all the existing algorithms are accurate in the relative sense,
- expectation: the missing one should be also accurate—proof harder than expected!

An alternative to the hyperbolic Jacobi algorithm

The main goals:

1. Provide an alternative to the hyperbolic one-sided Jacobi algorithm by the hyperbolic Kogbetliantz algorithm.
2. Find accurate 2×2 HSVD for triangular matrices.
3. Prove accuracy of the obtained algorithm.
4. Prove the global and the asymptotic convergence.

An alternative to the hyperbolic Jacobi algorithm

The hyperbolic Kogbetliantz algorithm:

- usually works in sweeps,
- in each step (according to a pivot strategy) a 2×2 pivot submatrix is chosen for diagonalization,
- computes (hyperbolic) sines/cosines of the angles,
- trigonometric transformations are applied to rows,
- trigonometric/hyperbolic transformations are applied to columns,
- pivot submatrix is updated (exact zeros are set to the off-diagonal).

Trigonometric vs. hyperbolic Kogbetliantz algorithm

Trigonometric annihilation relation in matrix form

$$\begin{bmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} g_{ii} & g_{ij} \\ g_{ji} & g_{jj} \end{bmatrix} \begin{bmatrix} \cos\psi & -\sin\psi \\ \sin\psi & \cos\psi \end{bmatrix} = \begin{bmatrix} g'_{ii} & 0 \\ 0 & g'_{jj} \end{bmatrix}.$$

Hyperbolic annihilation relation in matrix form

$$\begin{bmatrix} \cos\varphi & \sin\varphi \\ -\sin\varphi & \cos\varphi \end{bmatrix} \begin{bmatrix} g_{ii} & g_{ij} \\ g_{ji} & g_{jj} \end{bmatrix} \begin{bmatrix} \cosh\psi & -\sinh\psi \\ -\sinh\psi & \cosh\psi \end{bmatrix} = \begin{bmatrix} g'_{ii} & 0 \\ 0 & g'_{jj} \end{bmatrix}.$$

Trigonometric vs. hyperbolic Kogbetliantz algorithm

Trigonometric relations: left-hand side first (L-R)

$$
\begin{aligned}
(g_{ii}\cos\varphi + g_{ji}\sin\varphi)\cos\psi + (g_{ij}\cos\varphi + g_{jj}\sin\varphi)\sin\psi &= g'_{ii}\\
-(g_{ii}\cos\varphi + g_{ji}\sin\varphi)\sin\psi + (g_{ij}\cos\varphi + g_{jj}\sin\varphi)\cos\psi &= 0\\
-(g_{ii}\sin\varphi - g_{ji}\cos\varphi)\cos\psi - (g_{ij}\sin\varphi - g_{jj}\cos\varphi)\sin\psi &= 0\\
(g_{ii}\sin\varphi - g_{ji}\cos\varphi)\sin\psi - (g_{ij}\sin\varphi - g_{jj}\cos\varphi)\cos\psi &= g'_{jj}.
\end{aligned}
$$

Hyperbolic relations: left-hand side first (L-R)

$$
\begin{aligned}
(g_{ii}\cos\varphi + g_{ji}\sin\varphi)\cosh\psi - (g_{ij}\cos\varphi + g_{jj}\sin\varphi)\sinh\psi &= g'_{ii}\\
-(g_{ii}\cos\varphi + g_{ji}\sin\varphi)\sinh\psi + (g_{ij}\cos\varphi + g_{jj}\sin\varphi)\cosh\psi &= 0\\
-(g_{ii}\sin\varphi - g_{ji}\cos\varphi)\cosh\psi + (g_{ij}\sin\varphi - g_{jj}\cos\varphi)\sinh\psi &= 0\\
(g_{ii}\sin\varphi - g_{ji}\cos\varphi)\sinh\psi - (g_{ij}\sin\varphi - g_{jj}\cos\varphi)\cosh\psi &= g'_{jj}.
\end{aligned}
$$

Trigonometric vs. hyperbolic Kogbetliantz algorithm

Trigonometric case: L-R

$$\tan\psi = \frac{g_{ij} + g_{jj}\tan\varphi}{g_{ii} + g_{ji}\tan\varphi} = \frac{-g_{ii}\tan\varphi + g_{ji}}{g_{ij}\tan\varphi - g_{jj}},$$

$$\tau := \tan 2\varphi = \frac{2(g_{ii}g_{ji} + g_{ij}g_{jj})}{g_{ii}^2 - g_{jj}^2 + g_{ij}^2 - g_{ji}^2}.$$

Hyperbolic case: L-R

$$\tanh\psi = \frac{g_{ij} + g_{jj}\tan\varphi}{g_{ii} + g_{ji}\tan\varphi} = \frac{g_{ii}\tan\varphi - g_{ji}}{g_{ij}\tan\varphi - g_{jj}},$$

$$\tau := \tan 2\varphi = \frac{2(g_{ii}g_{ji} - g_{ij}g_{jj})}{g_{ii}^2 + g_{jj}^2 - g_{ij}^2 - g_{ji}^2}.$$

Trigonometric vs. hyperbolic Kogbetliantz algorithm

Two solutions for $\tan\varphi$, trigonometric case: L-R

Both tangents can be taken for annihilation.

$$(\tan\varphi)_1 = -\frac{1+\sqrt{1+\tau^2}}{\tau}, \quad (\tan\varphi)_2 = \frac{\tau}{1+\sqrt{1+\tau^2}}.$$

Two solutions for $\tan\varphi$, hyperbolic case: L-R

At most one solution is suitable, since it has to give $|\tanh\psi| < 1$ in the later computation. Which one? Not clear immediately...

$$(\tan\varphi)_1 = -\frac{1+\sqrt{1+\tau^2}}{\tau}, \quad (\tan\varphi)_2 = \frac{\tau}{1+\sqrt{1+\tau^2}}.$$

Hyperbolic Kogbetliantz algorithm

Hyperbolic case: L-R

Note that $|\tau|$ can be infinity or zero. For example, let

$$G = \begin{bmatrix} 1 & 2 \\ 2 & 1 \end{bmatrix}.$$

In this case $\tan 2\varphi = 0$, and if we take $\tan\varphi = 0$ we obtain

$$\tanh\psi = 2 \qquad (!!!)$$

The other solution is $\tan\varphi = \pm\infty$, and in this case

$$\tanh\psi = \frac{1}{2}.$$

Hyperbolic Kogbetliantz algorithm

Theorem

In the L-R algorithm, if both $(\tan\varphi)_{1,2}$ *are well-defined, exactly one is suitable for the definition of the hyperbolic tangent.*

Note that

$$(\tan\varphi)_1 = -\frac{1+\sqrt{1+\tau^2}}{\tau}, \quad (\tan\varphi)_2 = \frac{\tau}{1+\sqrt{1+\tau^2}}$$

can be accurately computed (no subtractions!).

Previous example with $\tan\varphi = \pm\infty$ motivates us to try the right-hand side first algorithm.

Trigonometric vs. hyperbolic Kogbetliantz algorithm

Trigonometric relations: right-hand side first (R-L)

$$
\begin{aligned}
(g_{ii}\cos\psi + g_{ij}\sin\psi)\cos\varphi + (g_{ji}\cos\psi + g_{jj}\sin\psi)\sin\varphi &= g'_{ii}\\
-(g_{ii}\sin\psi - g_{ij}\cos\psi)\cos\varphi - (g_{ji}\sin\psi - g_{jj}\cos\psi)\sin\varphi &= 0\\
-(g_{ii}\cos\psi + g_{ij}\sin\psi)\sin\varphi + (g_{ji}\cos\psi + g_{jj}\sin\psi)\cos\varphi &= 0\\
(g_{ii}\sin\psi - g_{ij}\cos\psi)\sin\varphi - (g_{ji}\sin\psi - g_{jj}\cos\psi)\cos\varphi &= g'_{jj}.
\end{aligned}
$$

Hyperbolic relations: right-hand side first (R-L)

$$
\begin{aligned}
(g_{ii}\cosh\psi - g_{ij}\sinh\psi)\cos\varphi + (g_{ji}\cosh\psi - g_{jj}\sinh\psi)\sin\varphi &= g'_{ii}\\
-(g_{ii}\sinh\psi - g_{ij}\cosh\psi)\cos\varphi - (g_{ji}\sinh\psi - g_{jj}\cosh\psi)\sin\varphi &= 0\\
-(g_{ii}\cosh\psi - g_{ij}\sinh\psi)\sin\varphi + (g_{ji}\cosh\psi - g_{jj}\sinh\psi)\cos\varphi &= 0\\
(g_{ii}\sinh\psi - g_{ij}\cosh\psi)\sin\varphi - (g_{ji}\sinh\psi - g_{jj}\cosh\psi)\cos\varphi &= g'_{jj}.
\end{aligned}
$$

Trigonometric vs. hyperbolic Kogbetliantz algorithm

Trigonometric case: R-L

$$\tan\varphi = -\frac{g_{ii}\tan\psi - g_{ij}}{g_{ji}\tan\psi - g_{jj}} = \frac{g_{ji} + g_{jj}\tan\psi}{g_{ii} + g_{ij}\tan\psi},$$

$$\sigma := \tan 2\psi = \frac{2(g_{ii}g_{ij} + g_{ji}g_{jj})}{g_{ii}^2 - g_{ij}^2 + g_{ji}^2 - g_{jj}^2}.$$

Hyperbolic case: R-L

$$\tan\varphi = -\frac{g_{ii}\tanh\psi - g_{ij}}{g_{ji}\tanh\psi - g_{jj}} = \frac{g_{ji} - g_{jj}\tanh\psi}{g_{ii} - g_{ij}\tanh\psi},$$

$$\sigma := \tanh 2\psi = \frac{2(g_{ii}g_{ij} + g_{ji}g_{jj})}{g_{ii}^2 + g_{ij}^2 + g_{ji}^2 + g_{jj}^2}.$$

Hyperbolic Kogbetliantz algorithm

Theorem

In the R-L algorithm, $|\sigma| \leq 1$ since

$$(|g_{ii}| - |g_{ij}|)^2 + (|g_{ji}| - |g_{jj}|)^2 \geq 0.$$

Equality holds if and only if $g_{ii} = g_{ij}$ and $g_{ji} = g_{jj}$, or $g_{ii} = -g_{ij}$ and $g_{ji} = -g_{jj}$, i.e., if and only if the pivot matrix is singular.

Note that if pivot matrix is triangular and nonsingular, $\tanh 2\psi$ is always well-defined. Additionally, it is always computed accurately (no subtractions!).

This motivates us to try the triangular R-L algorithm.

Trigonometric vs. hyperbolic Kogbetliantz algorithm

Two solutions for $\tan\psi$, trigonometric case: R-L

Both tangents can be taken for annihilation.

$$(\tan\psi)_1 = \frac{\sigma}{1+\sqrt{1+\sigma^2}}, \quad (\tan\psi)_2 = -\frac{1+\sqrt{1+\sigma^2}}{\sigma}.$$

Two solutions for $\tanh\psi$, hyperbolic case: R-L

At most one solution is suitable, and it is always $|\tanh\psi_1| < 1$.

Note that we have unpleasant subtractions!

$$(\tanh\psi)_1 = \frac{\sigma}{1+\sqrt{1-\sigma^2}}, \quad (\tanh\psi)_2 = \frac{1+\sqrt{1-\sigma^2}}{\sigma}.$$

Advantages of the triangular Kogbetliantz algorithm

Update of the diagonal elements for upper triangular G:

$$g'_{ii} = \frac{g_{ii}\cos\varphi}{\cos\psi} = \frac{g_{jj}\sin\psi}{\sin\varphi}, \quad g'_{jj} = \frac{g_{ii}\sin\varphi}{\sin\psi} = \frac{g_{jj}\cos\psi}{\cos\varphi}.$$

Update of the diagonal elements for upper triangular G:

$$g'_{ii} = \frac{g_{ii}\cos\varphi}{\cosh\psi} = -\frac{g_{jj}\sinh\psi}{\sin\varphi}, \quad g'_{jj} = -\frac{g_{ii}\sin\varphi}{\sinh\psi} = \frac{g_{jj}\cosh\psi}{\cos\varphi}.$$

Formulæ for lower triangular G are similar.

Advantages of the triangular Kogbetliantz algorithm

Hari and Matejaš have proved that choice L-R or R-L transformations (in the trigonometric case) depends on

- size of the elements in a triangle,
- structure of the matrix (lower or upper triangular).

In the hyperbolic case,

- examples indicate that always at least one transformation L-R or R-L has accurately computed angles,
- it is not easy to say which transformation has this property,
- the algorithm works well with almost orthogonal factor.

Pathological examples

Example 1

$$G = \begin{bmatrix} 10^{-6} & 1 \\ & 1 \end{bmatrix}, \quad J = \mathrm{diag}(1, -1).$$

Exact values:

$$(\tan\varphi)_1 = -0.99999999999995, \quad (\tanh\psi)_1 = 4.99999999999875 \cdot 10^{-7},$$
$$(\tan\varphi)_2 = 1.00000000000005, \quad (\tanh\psi)_2 = 2.00000000000005 \cdot 10^{6}.$$

L-R algorithm:

$$(\tan\varphi)_1 = -1, \quad (\tanh\psi)_1 = 5.00044 \cdot 10^{-7},$$
$$(\tan\varphi)_2 = 1, \quad (\tanh\psi)_2 = 2 \cdot 10^{6}.$$

R-L algorithm:

$$(\tanh\psi)_1 = 5.0 \cdot 10^{-7}, \quad (\tan\varphi)_1 = -1,$$
$$(\tanh\psi)_2 = 1.99982 \cdot 10^{6}, \quad (\tan\varphi)_2 = 0.999822.$$

Pathological examples

Example 2

$$G = \begin{bmatrix} 1 & 1 \\ & 10^{-6} \end{bmatrix}, \quad J = \mathrm{diag}(1, -1).$$

Exact values:

$$(\tan\varphi)_1 = -0.999999500000125, \quad (\tanh\psi)_1 = 0.9999990000005,$$
$$(\tan\varphi)_2 = 1.000000500000125, \quad (\tanh\psi)_2 = 1.0000010000005.$$

L-R algorithm:

$$(\tan\varphi)_1 = -1, \quad (\tanh\psi)_1 = 0.999999,$$
$$(\tan\varphi)_2 = 1, \quad (\tanh\psi)_2 = 1.$$

R-L algorithm:

$$(\tanh\psi)_1 = 0.999999, \quad (\tan\varphi)_1 = -0.999988,$$
$$(\tanh\psi)_2 = 1, \quad (\tan\varphi)_2 = 0.999989.$$

Pathological examples

Example 3

$$G = \begin{bmatrix} 10^{-6} & 1 \\ & 10^{-6} \end{bmatrix}, \quad J = \mathrm{diag}(1, -1).$$

Exact values:

$$(\tan\varphi)_1 = -999999.999999, \qquad (\tanh\psi)_1 = 9.99999999999 \cdot 10^{-7},$$
$$(\tan\varphi)_2 = 1.000000000001 \cdot 10^{-6}, \qquad (\tanh\psi)_2 = 1.000000000001 \cdot 10^{6}.$$

L-R algorithm:

$$(\tan\varphi)_1 = -1.00002 \cdot 10^{6}, \qquad (\tanh\psi)_1 = -22.1222,$$
$$(\tan\varphi)_2 = 1 \cdot 10^{-6}, \qquad (\tanh\psi)_2 = 1 \cdot 10^{6}.$$

R-L algorithm:

$$(\tanh\psi)_1 = 1 \cdot 10^{-6}, \qquad (\tan\varphi)_1 = -1 \cdot 10^{6},$$
$$(\tanh\psi)_2 = 999967, \qquad (\tan\varphi)_2 = -33.3883.$$

Conclusion

Future work

- decision procedure when L-R/R-L algorithm for 2×2 matrix is better than the other,
- proof of global and asymptotic convergence of the algorithm (off-norm and norm of the matrix can increase in a sweep),
- blocking and parallelization.