A Verification Method of Generalized Eigenvalue Problems and Its Applications to Constructive Error Estimate

Nobito Yamamoto[†], Yoshitaka Watanabe[‡], Yasunori Tanaka[†] and Mitsuhiro T.Nakao[†]

[†]Graduate School of Mathematics, Kyushu University 33
[‡]Computer Center, Kyushu University 33
Fukuoka 812-81, JAPAN

Slide 1

一般化固有値問題の精度保証付き解法

山本 野人[†], 渡部 善隆[‡],
田中 得登[†], 中尾 充宏[†]

[†] 九州大学大学院数理学研究科
[‡] 九州大学大型計算機センター

Generalized Eigenvalue Problems

PROBLEM

Find the maximum abusolute value of λ such that

$$A\,x = \lambda\,B\,x, \tag{1}$$

$x \in \mathbb{R}^n$,
A : symmetric in $\mathbb{R}^{n\times n}$,
B : symmetric and positive definite in $\mathbb{R}^{n\times n}$.

○ Motivation ○

- As conserns the numerical verification methods for PDEs which we have developed, we need an upper bound of λ as sharp as possible.
- Aside from the numerical verification, this situation with the problem (1) often appears in many cases in mathematical science and engineering.
- Thus we believe that it is significant to investigate calculations for (1) with guaranteed accuracy.

Interval Arithmetic and Reliable Computation

Interval arithmetic may be used for considering the rounding error occuring in the calculation.

- **The arithmetical operations**

For $X := [\underline{x}, \overline{x}]$ and $Y := [\underline{y}, \overline{y}]$,

$$
\begin{aligned}
X + Y &:= [\underline{x} + \underline{y}, \overline{x} + \overline{y}], \\
X - Y &:= [\underline{x} - \overline{y}, \overline{x} - \underline{y}], \\
X \cdot Y &:= [\min\{\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}\}, \max\{\underline{x}\underline{y}, \underline{x}\overline{y}, \overline{x}\underline{y}, \overline{x}\overline{y}\}], \\
1/Y &:= [1/\overline{y}, 1/\underline{y}], \text{ if } 0 \notin Y, \\
X/Y &:= X \cdot (1/Y).
\end{aligned}
$$

- **In order to enclose the rounding error**

Bit operations (addition) : $A + B \subset [\underline{a+b}, \overline{a+b}]$,

$\underline{a+b}$: nearest floating number (lower) to $\underline{a} + \underline{b}$ in the machine,

$\overline{a+b}$: nearest floating number (upper) to $\overline{a} + \overline{b}$ in the machine.

For example, in Fortran 90, `NEAREST` function is provided.

Slide 4

Software Packages for Interval Arithmetic

C-XSC : Provides calculation for inner products of two vectors with no cancellarion of mantissa. There are other packages for different languages which provides precise inner products : FORTRAN-SC, PASCAL-XSC.

PROFIL : A very fast library for interval arithmetic based on C++. Many people use it, and everybody is welcome to join. PROFIL is growing, currently available for many PCs and workstations.

Range Solver : An Add-In package that generalizes MS Excel 5.0 (Windows NT) and MS Excel 7.0 (Windows 95) for extended interval arithmetic and constraint satisfaction.

CALC : A free software of a calculator for rational computations. This has an operator-overloading language, so one can easily construct a library for interval arithmetic with rational numbers.

CLP(BNR), Delisoft Ltd, INTBLAS Project, INTLIB, Interval Software in Maple, Karlsruhe ftp site, RVInterval, UniCalc, VerGO etc.

URL : http://interval-computations.home.ml.org/

Slide 5

Methods for Generalized Eigenvalue Probrems

We explain four kinds of methods :

PROBLEM

Method 1 : Approximate Diagonalization Method

Method 2 : Approximate Diagonalization Method Advanced

Method 3 : Rump's Method

Method 4 : Generalized Rump's Method

○ Remarks ○

- Method 1 and Method 3 treat problems tranformed to

$$\lambda = \sup_{y \neq 0} \left| \frac{y^T E y}{y^T y} \right|, \tag{2}$$

$$E := C^{-1} A C^{-T},$$

C : $n \times n$ lower triangular matrix obtained by the Cholesky decomposition of B, namely,

$$B = CC^T.$$

○ **Remarks**(continuation) ○

- Note that this transformation should be executed by a rigorous calculation using interval arithmetic considering the rounding error.

  Thus E is given as an interval matrix.

- The Method 2 and Method 4 do not need the tranformation from (1) to (2) (ie. $Ax = \lambda Bx \implies (C^{-1}AC^{-T})x = \lambda x$), thus we will find

$$\lambda = \sup_{x \neq 0} \left| \frac{x^T A x}{x^T B x} \right| .$$

Approximate Diagonalization Method

Algorithm

1. $B = CC^T$ — rigorously
2. $E := C^{-1}AC^{-T}$ — rigorously
3. Compute a $n \times n$ matrix $\tilde{T}$ which approximately diagonalizes $\mathrm{mid}(E)$ by some suitable method (eg. QR method). — approximately
4. $\tilde{D} := \tilde{T}^T E \tilde{T}$, — rigorously

   $\tilde{\lambda} := \max\limits_{1 \le i \le n} \sum\limits_{j=1}^{n} |\tilde{D}_{ij}|$ — rigorously
5. $\tilde{I} := (\tilde{T}\tilde{T}^T)^{-1}$, — rigorously

   $\lambda_T := \max\limits_{1 \le i \le n} \sum\limits_{j=1}^{n} |\tilde{I}_{ij}|$ — rigorously
6. $\lambda \le \tilde{\lambda}\lambda_T$ — rigorously

Approximate Diagonalization Method

Proof of Algorithm

$$
\begin{aligned}
\lambda &= \sup_{y \neq 0} \left| \frac{y^T E y}{y^T y} \right| \\
&= \sup_{z \neq 0} \left| \frac{z^T \tilde{T}^T E \tilde{T} z}{z^T \tilde{T}^T \tilde{T} z} \right| && \longleftarrow \quad z = \tilde{T}^{-1} y \\
&\leq ||\tilde{T}^T E \tilde{T}||_2 \; ||(\tilde{T}^T \tilde{T})^{-1}||_2 && \text{where} \quad ||A||_2 := \sup_{x \neq 0} \frac{||Ax||}{||x||} \\
&= ||\tilde{D}||_2 \; ||\tilde{I}||_2 \\
&\leq \tilde{\lambda} \lambda_T && \longleftarrow \quad ||A||_2 \leq \max_{1 \leq i \leq n} \sum_{j=1}^{n} |A_{ij}|
\end{aligned}
$$

(Corollary of Gerschgorin's thorem)

Slide 9

Approximate Diagonalization Method Advanced

Algorithm

1.	$B \approx CC^T$	approximately
2.	$\tilde{E} := \tilde{C}^{-1} A \tilde{C}^{-T}$	approximately
3.	Compute a $n \times n$ matrix $\tilde{T}$ which approximately diagonalizes $\tilde{E}$ by some suitable method (eg. QR method).	approximately
4.	$\tilde{P} := \tilde{T}^T \tilde{C}^{-1}$	approximately
5.	$X_1 := \tilde{P} A \tilde{P}^T$	rigorously
	$\tilde{\lambda_1} := \max_{1 \le i \le n} \sum_{j=1}^{n} \lvert (X_1)_{ij} \rvert$	rigorously
6.	$X_2 := (\tilde{P} B \tilde{P}^T)^{-1}$	rigorously
	$\tilde{\lambda_2} := \max_{1 \le i \le n} \sum_{j=1}^{n} \lvert (X_2)_{ij} \rvert$	rigorously
7.	$\lambda \le \lambda_1 \lambda_2$	rigorously

Approximate Diagonalization Method Advanced

Proof of Algorithm

$$
\begin{aligned}
\lambda &= \sup_{x\neq 0}\left|\frac{x^TAx}{x^TBx}\right| \\
&= \sup_{y\neq 0}\left|\frac{y^T\tilde{C}^{-1}A\tilde{C}^{-T}y}{y^T\tilde{C}^{-1}B\tilde{C}^{-T}y}\right| && \longleftarrow \quad y=\tilde{C}^Tx \\
&= \sup_{z\neq 0}\left|\frac{z^T\tilde{T}^T\tilde{C}^{-1}A\tilde{C}^{-T}\tilde{T}z}{z^T\tilde{T}^T\tilde{C}^{-1}B\tilde{C}^{-T}\tilde{T}z}\right| && \longleftarrow \quad z=\tilde{T}^{-1}y \\
&= \sup_{z\neq 0}\left|\frac{z^T\tilde{P}A\tilde{P}^Tz}{z^T\tilde{P}B\tilde{P}^Tz}\right| && \longleftarrow \quad z=\tilde{P}^{-T}x \\
&\leq ||\tilde{P}A\tilde{P}^T||_2\ ||(\tilde{P}B\tilde{P}^T)^{-1}||_2 && \text{where} \quad ||A||_2 := \sup_{x\neq 0}\frac{||Ax||}{||x||} \\
&= ||X_1||_2\ ||X_2||_2 \\
&\leq \lambda_1\lambda_2 && \longleftarrow \quad ||A||_2 \leq \max_{1\leq i\leq n}\sum_{j=1}^{n}|A_{ij}| \\
& && \text{(Corollary of Gerschgorin's thorem)}
\end{aligned}
$$

Rump's Method

Algorithm

1.	$B = CC^T$	rigorously
2.	$E := C^{-1}AC^{-T}$	rigorously
3.	Compute an approximate value $\gamma \approx \lambda$ by some favorite method.	approximately
4.	$\tilde{\gamma} := (1+\delta)\gamma \quad$ for some $0 < \delta \ll 1$	rigorously
5.	$X_1 := \tilde{\gamma}I - E, \quad X_2 := \tilde{\gamma}I + E$	rigorously
6.	$\mathrm{mid}(X_1) \approx \tilde{C}_1 \tilde{C}_1^{\,T}, \quad \mathrm{mid}(X_2) \approx \tilde{C}_2 \tilde{C}_2^{\,T}$	approximately
7.	$Y_1 := \tilde{C}_1 \tilde{C}_1^{\,T} - X_1, \quad \lambda_1 := \max\limits_{1 \le i \le n} \sum\limits_{j=1}^{n} \lvert(Y_1)_{ij}\rvert$	rigorously
8.	$Y_2 := \tilde{C}_2 \tilde{C}_2^{\,T} - X_2, \quad \lambda_2 := \max\limits_{1 \le i \le n} \sum\limits_{j=1}^{n} \lvert(Y_2)_{ij}\rvert$	rigorously
9.	$\lambda \le \tilde{\gamma} + \max\{\lambda_1, \lambda_2\}$	rigorously

Rump's Method

Proof of Algorithm

For any vector $x \in \mathbb{R}^n$ under $x^T x = 1$,

$$|x^T(\tilde{C}_1\tilde{C}_1{}^T - \tilde{\gamma}I + E)x| = |x^T Y_1 x| \leq \lambda_1,$$

$$|x^T(\tilde{C}_2\tilde{C}_2{}^T - \tilde{\gamma}I - E)x| = |x^T Y_2 x| \leq \lambda_2.$$

Since $\tilde{C}_1\tilde{C}_1{}^T$ and $\tilde{C}_2\tilde{C}_2{}^T$ are positive definite, we obtain

$$-\tilde{\gamma} - \lambda_2 \leq x^T E x \leq \tilde{\gamma} + \lambda_1.$$

Then, $\lambda \leq \tilde{\gamma} + \max\{\lambda_1, \lambda_2\}$ holds.

○ Note ○

If A is known as positive definite, the process for X_2 can be omitted.

Generalized Rump's Method

Algorithm

1. Compute an approximate value $\gamma \approx \lambda$ by some favorite method. — approximately
2. $\tilde{\gamma} := (1 + \delta_1)\gamma \quad$ for some $0 < \delta_1 \ll 1$ — rigorously
3. $X_1 := \tilde{\gamma}B - A, \quad X_2 := \tilde{\gamma}B + A$ — rigorously

   We will verify each X_k $(k = 1, 2)$ is **positive definite**
4. Compute an approximate value ρ_k to the minimum absolute value of the eigenvalues of X_k. — approximately
5. $\tilde{\rho_k} := (1 - \delta_2)\rho_k \quad$ for some $0 < \delta_2 \ll 1$ — rigorously
6. $Y_k := X_k - \tilde{\rho}_k I$ — rigorously
7. $Y_k \approx \tilde{C}_k \tilde{C}_k^T$ — approximately
8. $Z_k := \tilde{C}_k \tilde{C}_k{}^T - Y_k, \quad \lambda_k := \max\limits_{1 \le i \le n} \sum\limits_{j=1}^{n} |(Z_k)_{ij}|$ — rigorously
9. If $\tilde{\rho_k} - \lambda_k > 0$ then X_k is positive definite and $\lambda \le \tilde{\gamma}$.

Generalized Rump's Method

Proof of Algorithm

For any vector $x \in \mathbb{R}^n$ under $x^T x = 1$,

$$|x^T(\tilde{C}_k \tilde{C}_k{}^T - X_k + \tilde{\rho}_k I)x| = |x^T Z_k x| \leq \lambda_k,$$

Since $\tilde{C}_k \tilde{C}_k{}^T$ positive definite, we obtain

$$x^T X_k x \geq \tilde{\rho}_k - \lambda_k > 0$$

Then, X_k is positive definite.

Hence,

$$x^T(\tilde{\gamma}B - A)x \geq 0, \quad x^T(\tilde{\gamma}B + A)x \geq 0,$$

then

$$-\tilde{\gamma}x^T Bx \leq x^T Ax \leq \tilde{\gamma}x^T Bx.$$

Since B is positive definite, we conclude that

$$\lambda = \sup_{x^T x = 1} \left| \frac{x^T Ax}{x^T Bx} \right| \leq \tilde{\gamma}.$$

Slide 15

Numerical Experiments

○ **Note** ○

- The results are obtained by rigorous calculation taking account of the effects of the rounding error with interval arithmetic.
  In practice, the interval library PROFIL is used.

- When B is diagonal dominant, there is no significant difference among the four methods, although Rump's Method (Method 3) is slightly better.
  - For example, with random matries A, B ($n = 100$),

Householder-Bisection method(floating)	1.46678030490876
Method 3 (Rump's Method)	1.46678045158692
Method 2 (Approximate Diagonalization Method Advanced)	1.46692583274310
Method 1 (Approximate Diagonalization Method)	1.46692583274691
Method 4 (Generalized Rump's Method)	1.46824708521367

$\delta = 10^{-7}$(Method 3), $\delta_1 = 10^{-3}$, $\delta_2 = 10^{-1}$(Method 4)

  - Diagonal dominant matrices are still easy to deal with even if they include interval elements.

- "An interval matrix is symmetric or positive definite." means "We will consider a set of symmetric or positive definite (non interval) matrices within the interval matrix."

To check the cases that B is not diagonal dominant

1. Give a list of eigenvalues arbitrarily, and a diagonal matrix D which has these eigenvalues as its diagonal elements.

2. Take a vector $v \in \mathbb{R}^n$ with random elements,
   and set $u := v/\|v\|$, where $\|v\| = \sqrt{v_1^2 + v_2^2 + \cdots + v_n^2}$.

3. Calculate $M := I - 2uu^T$.
   Then M is an orthogonal matrix : $M^T = M^{-1}$.

4. Determine a lower triangular matrix Q by : Q_{ij} $(i \neq j)$: random numbers within $[-1, 1]$,
   Q_{ii} : random numbers within $[a, b]$ $(1 < a < b)$.

5. Take $A := QM^T DMQ^T$ and $B := QQ^T$.

Result 1

- If $Q_{ii} < 2.5$, the Cholesky decomposition of $B := QQ^T$ fails in all cases, because of the inflation of intervals caused by the rounding error. Thus Method 2 and Method 4 are available.

Rate of Success in Cholesky Decomposition

$Q_{ii}(i = 1, \cdots, n)$	success/trial
$Q_{ii} \in [1, 2]$	0/10
$Q_{ii} \in [2, 3]$	0/10
$Q_{ii} \in [2.5, 3.5]$	10/10
$Q_{ii} \in [3, 4]$	10/10
$Q_{ii} \in [4, 5]$	10/10

$n = 100$, $Q_{ij} \in [-1, 1]$

Result 2

- For $Q_{ii} \in [3, 4]$, it is seen that Method 2 gives the best (however, slightly better than Method 4).

  For example, with $Q_{ii} \in [3, 4]$, $Q_{ij} \in [-1, 1]$ and $n = 100$, we obtain the following results.

Maximum Eigenvalue	0.99865718558305
Householder-Bisection method(floating)	0.99866527318954
Method 2 (Approximate Diagonalization Method Advanced)	0.99874050465301
Method 4 (Generalized Rump's Method)	0.99966393846273
Method 3 (Rump's Method)	1.28195979497205
Method 1 (Approximate Diagonalization Method)	1.67007007110778

$\delta = 10^{-7}$(Method 3), $\delta_1 = 10^{-3}$, $\delta_2 = 10^{-1}$(Method 4)

Result 3

- Method 1 and Method 3 are getting better as Q_{ii} glows, and for $Q_{ii} > 4$, the Method 3 comes to be slightly better than the other methods.

  For example, with $Q_{ii} \in [5, 6]$, $Q_{ij} \in [-1, 1]$ and $n = 100$, we obtain the following results.

Maximum Eigenvalue	0.995330668050172
Householder-Bisection method(floating)	0.995335459709168
Method 3 (Rump's Method)	0.995341292611985
Method 2 (Approximate Diagonalization Method Advanced)	0.995384115831106
Method 1 (Approximate Diagonalization Method)	0.995385194006698
Method 4 (Generalized Rump's Method)	0.996330795168877

$\delta = 10^{-7}$(Method 3), $\delta_1 = 10^{-3}$, $\delta_2 = 10^{-1}$(Method 4)

Result 4

- For a problem of large marices, Method 4 is available. But there seems to be some **critical value of** δ_1.

  For example, with $Q_{ii} \in [9, 10]$, $Q_{ij} \in [-1, 1]$ and $n = 1000$, we obtain the following results.

Maximum Eigenvalue	0.997131260109256
Householder-Bisection method(floating)	0.997196376323700
Method 4 (Generalized Rump's Method) $\delta_1 = 10^{-3}$	*failure*
Method 4 (Generalized Rump's Method) $\delta_1 = 5 \times 10^{-3}$	*failure*
Method 4 (Generalized Rump's Method) $\delta_1 = 10^{-2}$	*failure*
Method 4 (Generalized Rump's Method) $\delta_1 = 5 \times 10^{-2}$	1.04705619513989

$\delta_2 = 10^{-1}$

Conclusion

- When the condition of the matrix B is not so good, Method 2 and Method 4 give desiable results.
- In view of the cost of computation (CPU time), Method 1, Method 2 and Method 3 are expensive because of the rigorous calculation of inverse matrices.

$\Downarrow$

Generalized Rump's Method(Method 4) is the best of the four if it works with a sufficiently small parameter δ_1.

Application to the Stokes Problems

• Stokes Equations

$$\begin{cases} -\nu\Delta u + \nabla p = f & \text{in} \quad \Omega, \\ \operatorname{div} u = 0 & \text{in} \quad \Omega, \\ u = 0 & \text{on} \quad \partial\Omega, \end{cases}$$

where Ω is a convex polygon in $\mathbb{R}^2$.

○ Weak Formulation

$$\nu(\nabla u, \nabla v) - (p, \operatorname{div} v) - (q, \operatorname{div} u) = (f, v) \qquad \forall [v, q] \in H_0^1(\Omega)^2 \times L_0^2(\Omega).$$

- $X_h \subset H_0^1(\Omega) \cap C(\bar{\Omega})$: approximation of the velocity u satisfies

$$\inf_{\xi \in X_h} |v - \xi|_1 \leq C_0 h |v|_2, \quad \forall v \in H_0^1(\Omega) \cap H^2(\Omega),$$

where C_0 is a positive constant numerically determined.

- $Y_h \subset L_0^2(\Omega) \cap C(\bar{\Omega})$: approximation of the pressure p.

○ Finite Element Solution

$$\nu(\nabla u_h, \nabla v_h) - (p_h, \operatorname{div}\, v_h) - (q_h, \operatorname{div}\, u_h) = (f, v_h) \quad \forall [v_h, q_h] \in X_h^2 \times Y_h.$$

A Priori Error Estimates for the Stokes Equations

THEOREM

$$|u - u_h|_1 \leq C(h) \left(\frac{1}{\nu^2} + \frac{1}{\beta^2} \right)^{\frac{1}{2}} |f|_0,$$

$$|p - p_h|_0 \leq C(h) \left(\frac{1}{\beta} + \frac{\nu}{\beta^2} \right) |f|_0.$$

where

$$C(h) \equiv \sqrt{(K_1 + C_0 h K_2)^2 + (C_0 h)^2},$$

$$|v|_1 \leq \frac{1}{\beta} |\mathrm{div}\, v|_0 \quad v \in \{u \in H_0^1(\Omega)^2; \mathrm{div}\, u = 0\}^{\perp},$$

$$K_1 := \left(\sup_{x \in R^n} \frac{x^T A_1 x}{x^T L x} \right)^{1/2},$$

$$K_2 := \left(\sup_{x \in R^n} \frac{x^T A_2 x}{x^T L x} \right)^{1/2}.$$

Numerical Example

- $\Omega = (0,1) \times (0,1)$.
- uniform partition. L denotes the number of partitions for the interval $(0,1)$, i.e. $h = 1/L$.
- X_h : continuous piecewise quadratic polynomials, Y_h : piecewise linear.
- $\nu = 1$, $C_0 = 1/(2\pi)$ and $1/\beta^2 = 4 + 2\sqrt{2}$.

Upper bounds obtained by Generalized Rump's Method

L	K_1	K_2	$C(h)$	dimension
5	0.051035	1.268929	0.0968082	162×162
(*fl.*)	(0.051008)	(1.268295)	(0.0967652)	
10	0.027955	1.238731	0.0502564	722×722
(*fl.*)	(0.027941)	(1.238111)	(0.0502339)	

$\delta_1 = 10^{-3}$, $\delta_2 = 10^{-1}$.

"*fl.*"s are computed on FUJITSU VP2600/10 vector processor by IBM floating format with double precision and Householder-Bisection method.