数理解析研究所講究録
第1976巻 2015年 1-7

近似GCDにおける逐次的なQR分解法とその実装について*

長坂耕作†
KOSAKU NAGASAKA
神戸大学人間発達環境学研究科
GRADUATE SCHOOL OF HUMAN DEVELOPMENT AND ENVIRONMENT, KOBE UNIVERSITY

1 近似GCDと本講演の背景

本講演では，近似GCDを構造の入った行列を分解することで求めるQRGCD[2]やUVGCD[5]などのアルゴリズムで使われる行列のQR分解に着目する。特に，近年研究されている構造化された行列のQR分解に関する高速算法を改良する試みについて，その中間報告を行う。なお，本講演で取り上げる近似GCDは，以下で定義されるものとする。

定義 1 (Degree-Revealing Approximate GCD)
$\|f\|_2 = \|g\|_2 = 1$ なる $f(x), g(x) \in \mathbb{C}[x]$ と $\varepsilon \in \mathbb{R}_{\geq 0}$ に対して，次式を満たす $d(x) \in \mathbb{C}[x]$ を求めよ。

$$\exists \Delta_f, \Delta_g, f_1, g_1 \in \mathbb{C}[x],\ \max\{\|\Delta_f\|_2, \|\Delta_g\|_2\} < \varepsilon,\ f + \Delta_f = d \cdot f_1,\ g + \Delta_g = d \cdot g_1,\ \|d\|_2 = 1$$

◁

例 1 (近似GCDの例)
$f(x) = 0.999x^2 + 1.999x + 1.001$ と $g(x) = 1.001x^2 - 0.999$ は，通常の意味で互いに素であり，その厳密な GCD は当然であるが $\gcd(f, g) = 1.0$ となる。しかし，これら与式の係数が（何らかの原因で含まれてしまった）誤差を含むと考えることで，その近似 GCD の1つとして，$\gcd(f, g) = 1.00063x + 0.999375$ を取ることができる。誤差を排除した真の多項式は確定できないが，その可能性の一つとして考えられるのが近似 GCD となる。 ◁

本講演では，行列分解による近似GCDアルゴリズムを対象とするが，主なアルゴリズムとQR分解との関係を述べておく。QRGCD[2]やその改良版であるExQRGCD[4]では，Sylvester行列のQR分解により，次数候補や因子候補の決定を行う。UVGCDでは，部分終結式行列の特殊な逐次型のQR分解により，次数候補や因子候補の決定を行い，QR分解によるGauss-Newton法で許容度の改善を行う。一方，FastGCD[1]では，これらのQR分解をPLUQ分解で代用することで，高速化を図っている。

なお，FastGCDの利用する構造化行列に対するPLUQ分解は $O(n^2)$ で，一般のQR分解やLU分解の $O(n^3)$ よりも高速であるが，QR分解がユニタリ変換でノルムを変化させないのに対し，LU分解やPLUQ分解では，ノルムが変化してしまうデメリットも存在する。本講演で改良しようとしている構造化行列に対するFQR分解は，Delvauxら[3]が提案しているもので，$O(n^2)$ の計算量（後述の r も加味すると，$O(rn^2)$）になっている。本講演では，このFQR分解の正方でない行列への拡張と，数値精度の改善に関する試みの中間報告を行っていく。

*This work was supported by JSPS KAKENHI Grant Number 15K00016.
†nagasaka@main.h.kobe-u.ac.jp

2 構造化行列と Delvaux らのアルゴリズム

下記のように，行列の要素に一定の構造が入ったものは，構造化行列 (Structured Matrix) と呼ばれる。

Hessenberg matrix

$$\begin{pmatrix} \times & \times & 0 & 0 \\ \times & \times & \times & 0 \\ \times & \times & \times & \times \\ \times & \times & \times & \times \end{pmatrix}, \begin{pmatrix} \times & \times & \times & \times \\ \times & \times & \times & \times \\ 0 & \times & \times & \times \\ 0 & 0 & \times & \times \end{pmatrix}$$

Toeplitz matrix

$$\begin{pmatrix} a & b & c & d \\ g & a & b & c \\ f & g & a & b \\ e & f & g & a \end{pmatrix}$$

Cauchy matrix

$A = (a_{ij}) \in \mathbb{C}^{n\times m}$,
$a_{ij} = \frac{1}{x_i - y_j}$, $x_i - y_j \neq 0$
for (x_i) and (y_j)

Hankel matrix

$$\begin{pmatrix} a & b & c & d \\ b & c & d & e \\ c & d & e & f \\ d & e & f & g \end{pmatrix}$$

近似 GCD で頻出する Sylvester 行列や部分終結式行列は，どちらも一定の構造の入った構造化行列となっているが，そのままではよく知られている Toeplitz や Hankel に似ているだけで，これらの行列には分類されない。一方，変異構造化行列 (Displacement Structured Matrix) と呼ばれる構造もあり，これは次のように定義される。

定義 2 (Displacement Structure （変異構造化行列）)
行列 $S \in \mathbb{C}^{m\times n}$ が次式を満たすならば，S は **displacement structure** を持つという。

$$\exists V \in \mathbb{C}^{m\times r},\ H \in \mathbb{C}^{r\times n},\ \underline{O_L S - S O_R = VH}$$

ここで，O_L, O_R は変異作用素 *(displacement operators)*，r は変異階数 *(displacement rank)*，V, H は生成行列 *(generators)* と呼ばれる。 ◁

Delvaux ら [3] の論文で対象としている行列は複数あるが，本講演に関係があるのは，Toeplitz-like 行列と呼ばれるもので，次のような Displacement Equation を満たすものである。

$$\exists A \in \mathbb{C}^{m\times r},\ B \in \mathbb{C}^{r\times n},\ Z_m^1 S - S Z_n^0 = AB,\ \ Z_\ell^\alpha = \begin{pmatrix} & I_{\ell-1} \\ \alpha & \end{pmatrix} \in \mathbb{C}^{\ell\times\ell}$$

一般に Toeplitz-like 行列と呼ばれるものは，これに限らず類似の Displacement Operator に対し，この種の Displacement Equation を満たす。実際，近似 GCD で頻出する Sylvester 行列や部分終結式行列は，どちらも Toeplitz-like 行列となっている。

さらに，Delvaux ら [3] の論文では，これらの Toeplitz-like 行列が，次のように定義される階数が r の階数構造化行列になっていることに着目し，QR 分解を（理論上）高速に行うアルゴリズムを提案している。

定義 3 (Rank Structured Matrix（階数構造化行列）)
行列 $A \in \mathbb{C}^{n\times m}$ が **semiseparable** であるとは，次式を満たすことをいう。

$$\forall j_2 \leq i_1 \text{ or } i_2 \leq j_1,\ \mathrm{rank}(A(i_1 : i_2, j_1 : j_2)) \leq r$$

ここで，r を **semiseparable rank** という。上記の式を言い換えると，対角成分を含まないどのブロック（要素を連続的に取り出す部分行列）の階数も r を越えない，となる。また，行列 $A \in \mathbb{C}^{n\times m}$ の下三角部分のみ *semiseparable* な場合は，**lower semiseparable** と呼ばれる。 ◁

2.1 Delvaux らのアルゴリズムの基礎

以下，Toeplitz-like 行列に対する Delvaux らのアルゴリズムを簡単に紹介する。まず，QR 分解の対象である Toeplitz-like 行列 $\tilde{A} \in \mathbb{C}^{n\times n}$ は次式を満たしているとする。すなわち，$\tilde{A} = \tilde{Q}R$ なる QR 分解を計算するのが目的である。

$$Z_{n,1}\tilde{A} - \tilde{A}Z_{n,0} = \tilde{V}H,\ V \in \mathbb{C}^{n\times r},\ H \in \mathbb{C}^{r\times n}$$

この関係式に，逆 Fourier 行列 F_n^h を左から積を取ることにより，式変形を行う（$\cdot^h$ は複素共役転置）。

$$F_n^h(Z_{n,1}F_nF_n^h\tilde{A}) - F_n^h(\tilde{A}Z_{n,0}) = F_n^h(\tilde{V}H) \Rightarrow \boxed{D_yA - AZ_{n,0} = VH}$$

ここで，それぞれの行列は，次のようになっている。

$$F_n = (f_{ij}) \in \mathbb{C}^{n\times n},\ f_{ij} = \frac{1}{\sqrt{n}}\left(e^{2\pi \mathbf{i}(i-1)(j-1)/n}\right),$$
$$A = F_n^h\tilde{A},\ V = F_n^h\tilde{V},\ D_y = \mathrm{diag}(y_{ii}) = F_n^hZ_{n,1}F_n \mathrm{with} |y_{ii}| = 1$$

Delvaux らのアルゴリズムは，$\tilde{A}$ に変えて VH の QR 分解をその階数構造に着目し行うものである。$\tilde{A}$ に対する $\tilde{Q} = F_nQ$ の計算は，コストが高くなるものの近似 GCD では使用しないため特に問題とはならない。

2.2 Delvaux らのアルゴリズムの実際

実際の分解は，さらに次のような変形を行った上で行われる。なお，この式変形は A が正則であることを求めているが，正則でない場合も問題がないことが間接的に論文内で言及されている。

$$\begin{aligned}
& D_yA - AZ_{n,0} = VH && (A = QR \text{を代入})\\
\Leftrightarrow\ & D_y(QR) - (QR)Z_{n,0} = VH && (Q^h \text{を左から})\\
\Leftrightarrow\ & (Q^hD_yQ)R - RZ_{n,0} = Q^hVH && (R^{-1} \text{を右から})\\
\Leftrightarrow\ & (Q^hD_yQ) - RZ_{n,0}R^{-1} = Q^hVHR^{-1} && (RZ_{n,0}R^{-1} \text{を移項})\\
\Leftrightarrow\ & (Q^hD_yQ) = RZ_{n,0}R^{-1} + Q^hVHR^{-1}
\end{aligned}$$

結果として得られる Q^hD_yQ は階数が r の **lower semiseparable** となり，Q がユニタリ行列で，$D_y = \mathrm{diag}(y_{ii})$，$|y_{ii}| = 1$ もユニタリ性を持つため，Q^hD_yQ もユニタリ行列となる。この性質（Q^hD_yQ が **lower semiseparable unitary**）により，具体的に上三角行列 R を計算せずに QR 分解を行うのが，Delvaux らのアルゴリズムの特徴となる。

実際の分解は，A の左下から右上方向へ逐次的に行われる。$\lfloor_kM\rfloor$ で行列 M の左下 $(k+1)\times(k+1)$ のブロックを表し，同様に，$\lfloor M\rfloor_k$ で右下，$\lceil_kM\rceil$ で左上を表すとする。このとき，既に，$A_k = Q_kR_k$，$A_k = \lfloor_kA\rfloor$ まで分解が終わっていると仮定すると，次の関係式が成立している。

$$Q_k^h\lfloor D_y\rfloor_kQ_k = R_k\lceil_kZ_{n,0}\rceil R_k^{-1} + Q_k^h\lfloor_kVH\rfloor R_k^{-1}$$

ここから，A_{k+1} の分解を直接求めるのではなく，$Q_{k+1}^h\lfloor D_y\rfloor_{k+1}Q_{k+1}$ を求める方法を説明する。まず，$Q_k^h\lfloor D_y\rfloor_kQ_k$ を，$(k+2)\times(k+2)$ のサイズに埋め込み，その上で，$G_1\cdots G_k$ を逐次的に計算する。

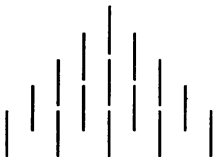
$$Q_{k+1}^h\lfloor D_y\rfloor_{k+1}Q_{k+1} \Leftarrow G_k^h\cdots G_1^h\begin{pmatrix}1 & 0\\ 0 & Q_k^h\end{pmatrix}\lfloor D_y\rfloor_{k+1}\begin{pmatrix}1 & 0\\ 0 & Q_k\end{pmatrix}G_1\cdots G_k$$

問題は，A_{k+1} を用いずに $G_1\cdots G_k$ を計算する方法であるが，それを以下で順次説明する。

2.2.1 Givens product representation graph

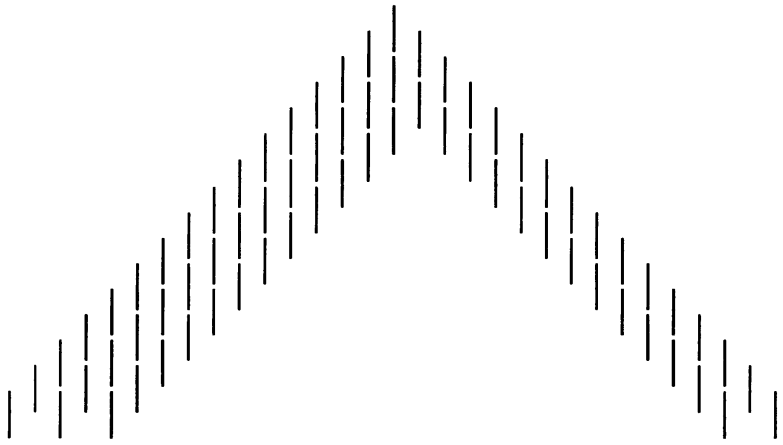
QR分解で求めるユニタリ行列は，Givens回転を使ったアルゴリズムの場合，その積として通常は計算される。Delvauxらのアルゴリズムでは，これらの積をそのままグラフとして表現する方法が取られている。例えば，$G_{i,j}$で，i行とj行に影響を与えるGivens回転に対応する行列とする。このとき，$G_{5,6}G_{4,5}G_{3,4}G_{2,3}G_{1,2}G_{3,4}$というユニタリ行列は，その影響を与える範囲を縦線で表すことで，次のグラフで表現される。

また，Givens回転によるQR分解アルゴリズムの結果を$A = QR$とすると，Q^hは次のグラフで表される。



$$\Longleftarrow Q^hA = \begin{pmatrix} \times & \times & \times & \times & \times & \times \\ 0 & \times & \times & \times & \times & \times \\ 0 & 0 & \times & \times & \times & \times \\ 0 & 0 & 0 & \times & \times & \times \\ 0 & 0 & 0 & 0 & \times & \times \\ 0 & 0 & 0 & 0 & 0 & \times \end{pmatrix}$$

この表現を用いると，Q^hD_yQ が lower semiseparable unitary である性質から，$n = 17$で$r = 3$の場合，そのグラフは次のようになる。

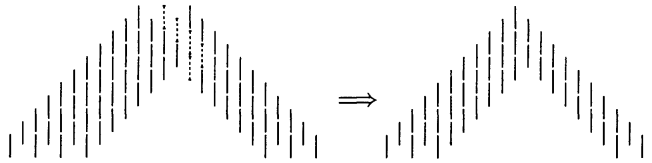


結果として，Q^hD_yQに関する逐次的な処理は次のような流れで表現される。

(1) $Q_k^h \lfloor D_y \rfloor_k Q_k$

(2) $\begin{pmatrix} 1 & 0 \\ 0 & Q_k^h \end{pmatrix} \lfloor D_y \rfloor_{k+1} \begin{pmatrix} 1 & 0 \\ 0 & Q_k \end{pmatrix}$

(3) $G_k^h \cdots G_1^h \begin{pmatrix} 1 & 0 \\ 0 & Q_k^h \end{pmatrix} \lfloor D_y \rfloor_{k+1} \begin{pmatrix} 1 & 0 \\ 0 & Q_k \end{pmatrix} G_1 \cdots G_k$



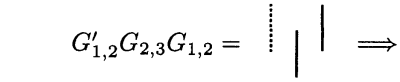
ところが，このままでは $G_1 \cdots G_k$ の計算が必要となってしまい，Hessenberg QR アルゴリズムと違いがなくなってしまう。そこで使われるのが，Pull-Through Lemma とそれによる連鎖的な $G_1 \cdots G_k$ の生成計算となる。

2.2.2 Pull-Through Lemma と Chasing

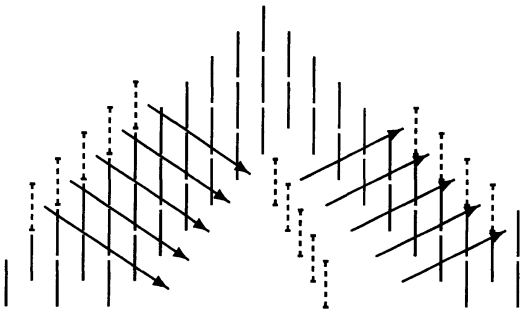
Delvaux らのアルゴリズムで中心的な役割を担うのが次の補題である。

補題 4 (Pull-Through)
$Q = G'_{1,2}G_{2,3}G_{1,2}$ と分解されるユニタリ行列 $Q \in \mathbb{C}^{3\times 3}$ に対し，$Q = \tilde{G}'_{2,3}\tilde{G}_{1,2}\tilde{G}_{2,3}$ なる分解が存在する。◁

この補題をグラフで表現すると次のようになる。

$G'_{1,2}G_{2,3}G_{1,2} =$  $= \tilde{G}'_{2,3}\tilde{G}_{1,2}\tilde{G}_{2,3}$

なお，この補題の逆も成り立つことと，この再分解は一意でないことに留意が必要となる。最終的に，この補題を用いて，$G_1 \cdots G_k$ の計算は次のように行われる（最初の r 個と最後の $r-1$ 個は，A_k から計算する必要がある）。



2.2.3 Delvaux らのアルゴリズムのまとめ

QR 分解の計算量が $O(n^3)$ なのは，消去すべき要素（左下部分）の総数が $O(n^3)$ なためであるが，Pull-Through Lemma を用いて，それまでの Q から次の Q を逐次的に計算することで，計算量を抑えている。結果として，計算量は $O(rn^2)$ となる。Sylvester 行列では $r = 2$ で，部分終結式行列では $r = 3$ なので，実質的に $O(n^2)$ のアルゴリズムとなる。

一方，問題点としては，実際に必要となる演算回数が多い（Pull-Through Lemma の操作は計算量に影響を及ぼさないが，演算回数はかなり多い）ことと，Delvaux ら [3] の論文内で「the accuracy of computed results can be seriously worse」と述べているように，計算精度が著しく悪いことが挙げられる。ただし，

Delvaux らの実験によれば，Cauchy-like 行列では一定の計算精度が実現されており，この違いは今後の課題とされている。

3 本講演での改善点

本講演では，計算精度を改善する可能性のある方法を 2 つ提案している。

3.1 Givens 回転の選択

複素行列に対する Givens 回転は，実数行列と異なり選択の幅が広くなっている。ここでは，その選択として以下の $G, U_{\mathbb{R}}, U_{\mathbb{C}}$ を取り上げる。Delvaux らが実験で使用している Givens 回転については不明であるが，Pull-Through Lemma による操作では，$r \in \mathbb{R}$ となることが部分的に必要なため，$U_{\mathbb{R}}$ であると推測する。

- G: $r \in \mathbb{C}$

$$G = \begin{pmatrix} c & s \\ -\bar{s} & c \end{pmatrix},\ c \in \mathbb{R}, s \in \mathbb{C}\ \text{ s.t. }\ \begin{pmatrix} c & s \\ -\bar{s} & c \end{pmatrix}^h \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix}$$

- $U_{\mathbb{R}}$: $r \in \mathbb{R}$

$$U_{\mathbb{R}} = \begin{pmatrix} c & s \\ -\bar{s} & \bar{c} \end{pmatrix},\ c, s \in \mathbb{C}\ \text{ s.t. }\ \begin{pmatrix} c & s \\ -\bar{s} & \bar{c} \end{pmatrix}^h \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix}$$

- $U_{\mathbb{C}}$: $r \in \mathbb{C}$

$$U_{\mathbb{C}} = \begin{pmatrix} c & s \\ -\bar{s} & \bar{c} \end{pmatrix},\ c, s \in \mathbb{C}\ \text{ s.t. }\ \begin{pmatrix} c & s \\ -\bar{s} & \bar{c} \end{pmatrix}^h \begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} r \\ 0 \end{pmatrix}$$

どの Givens 回転が適切であるか議論するために，Pull-Through Lemma に基づく計算を述べておく。

1. 通常の行列表現 Q を $Q = G'_{1,2} G_{2,3} G_{1,2}$ から求める。
2. Givens 回転を求め，必要な要素にのみそれを適用する。

$$\begin{pmatrix} \times & \times & \times \\ \times & \times & \times \\ \times & \times & \times \end{pmatrix} \xrightarrow[\tilde{G}'_{2,3} = U_{\mathbb{R}}]{} \begin{pmatrix} \times & \times & \times \\ \boxed{\begin{matrix} \times & \times \\ 0 & \times \end{matrix}} & \times \\ \end{pmatrix} \xrightarrow[\tilde{G}_{1,2} = U_{\mathbb{R}}]{} \begin{pmatrix} \boxed{\begin{matrix} 1 & 0 \\ 0 & \times \end{matrix}} & 0 \\ 0 & \times & \times \end{pmatrix}$$

$$\begin{pmatrix} 1 & 0 & 0 \\ 0 & \boxed{\begin{matrix} \times & \times \\ \times & \times \end{matrix}} \\ 0 & \end{pmatrix} \xrightarrow[\tilde{G}_{2,3} = \text{just conjugate transpose}]{} \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

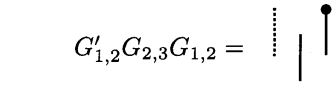
3. 結果として，$Q = \tilde{G}'_{2,3} \tilde{G}_{1,2} \tilde{G}_{2,3}$ が得られる。

ここで，$\tilde{G}_{1,2}$ は対角成分を 1 にする必要があるため，$U_{\mathbb{R}}$ の Givens 回転であることが求められる。一方で，$\tilde{G}'_{2,3}$ については，どの Givens 回転を用いても問題ない。しかしながら，$\tilde{G}'_{2,3}$ に $U_{\mathbb{R}}$ を用いると，途中経過の行列に実数が現れることになる。これは実部と虚部の情報が，実部のみに圧縮されたと考えることもでき，桁精度が僅かながら失われている可能性を否定できない。そこで，$\tilde{G}'_{2,3}$ には $U_{\mathbb{C}}$ を用いてはどうか，というのが本講演での提案である。手元の実験では，僅かながら精度の向上が見られている。

3.2 Unimodularity の維持

Givens 回転に基づく QR 分解アルゴリズムでは，Givens 回転を表す行列の行列式が 1 であるため，計算全体を通してユニタリ行列 Q の Unimodularity が保たれている。ところが，Delvaux らのアルゴリズムでは，Q 自体が Unimodularity を持っていたとしても，計算途中で現れる $Q^h D_y Q$ に含まれる対角行列 D_y は，部分を取り出すと Unimodular ではないため，$Q^h D_y Q$ も Unimodularity を持たない行列となる。本講演では，$Q^h D_y Q$ の Unimodularity が維持されないことも計算精度を失う原因の一つであると考え，その Unimodularity を保つよう Delvaux らのアルゴリズムを拡張する。

まず，既存の Pull-Through Lemma による副作用を説明する。$G'_{1,2}G_{2,3}G_{1,2}$ の再分解を行う際に，$G_{1,2}$ が Unimodular でない場合（これをドットで表す），次のように Unimodular でない性質が移動していく。

$$G'_{1,2}G_{2,3}G_{1,2} =$$  $$= \tilde{G}'_{2,3}\tilde{G}_{1,2}\tilde{G}_{2,3}$$

アルゴリズムにおいて Unimodularity が崩れるのは，D_y の対角成分を $Q^h D_y Q$ のグラフ表現に取り入れる際に，それをユニタリ行列として一括して表現してしまうためである。本講演では，次のように，D_y の対角成分はユニタリ行列に含めず，当該行をスカラー倍する要素として分離したまま保持することを提案する（これをドットで表している）。

$$G'_{1,2}G_{2,3}G_{1,2} =$$  $$= \tilde{G}'_{2,3}\tilde{G}_{1,2}\tilde{G}_{2,3}$$

Pull-Through Lemma はこの表現に対しても有効であり，実際，$G'_{1,2}S_{2,3}G_{2,3}S_{1,2}G_{1,2} = S_{2,3}S_{1,2}G''_{1,2}G_{2,3}G_{1,2}$ と，D_y の対角成分の絶対値が 1 である性質から，積 2 回と複素共役 1 回の演算で変形できる。結果として，$G''_{1,2}G_{2,3}G_{1,2} = \tilde{G}'_{2,3}\tilde{G}_{1,2}\tilde{G}_{2,3}$ なる Pull-Through 操作に帰着される。精度への影響については，講演時点では実験が終わっておらず不明である。なお，正方でない行列への拡張については，時間の関係で細かく講演しなかったため，本原稿においては割愛した。

参 考 文 献

[1] Bini, D. A., Boito, P., 2007. Structured matrix-based methods for polynomial ϵ-gcd: Analysis and comparisons. In: Proceedings of the 2007 International Symposium on Symbolic and Algebraic Computation. ISSAC '07. ACM, New York, NY, USA, pp. 9–16.

[2] Corless, R. M., Watt, S. M., Zhi, L., 2004. QR factoring to compute the GCD of univariate approximate polynomials. IEEE Trans. Signal Process. 52 (12), 3394–3402.

[3] Delvaux, S., Gemignani, L., Van Barel, M., 2010. QR-factorization of displacement structured matrices using a rank structured matrix approach. In: Numerical methods for structured matrices and applications. Vol. 199 of Oper. Theory Adv. Appl. Birkhäuser Verlag, Basel, pp. 229–254.

[4] Nagasaka, K., Masui, T., 2013. Extended qrgcd algorithm. In: Proceedings of the 15th International Workshop on Computer Algebra in Scientific Computing - Volume 8136. CASC 2013. Springer-Verlag New York, Inc., New York, NY, USA, pp. 257–272.

[5] Zeng, Z., 2011. The numerical greatest common divisor of univariate polynomials. In: Randomization, relaxation, and complexity in polynomial equation solving. Vol. 556 of Contemp. Math. Amer. Math. Soc., Providence, RI, pp. 187–217.