Approximate Square-free Decomposition and Root-finding of Ill-conditioned Algebraic Equations

TATEAKI SASAKI* and MATU-TAROW NODA**

The exact square-free decomposition is generalized to polynomials with coefficients of floating-point numbers, and an algorithm of approximate square-free decomposition is presented. Given a polynomial $P(x)$ and a small positive number ε, $0<\varepsilon\ll 1$, the decomposition algorithm calculates polynomials $Q_1, Q_2, \ldots, Q_l$ such that $P(x)\simeq Q_1(x)Q_2^2(x)\ldots Q_l^l(x)$, where each root of $Q_m(x)=0$ is approximately equal to m multiple root or the average value of m close roots of $P(x)=0$. The decomposition is performed by using a generalized Euclidean algorithm, and the properties of approximate GCD (greatest common divisor), computed by the Euclidean algorithm, is investigated by developing a theory of approximate GCD. The equations $Q_i(x)=0$, $i=1, \ldots, l$, are much easier to solve numerically than $P(x)=0$. Hence, we apply the approximate square-free decomposition to solving ill-conditioned algebraic equations and propose an algorithm which finds not only multiple but also close roots nicely.

1. Introduction

A polynomial $Q(x)$ is called *square-free* if the equation $Q(x)=0$ has no multiple root. Any polynomial $P(x)$ with coefficients in a field or a Euclidean ring can be decomposed uniquely into square-free factors $Q_1, \ldots, Q_l$ as

$$P(x)=Q_1(x)Q_2^2(x)\ldots Q_l^l(x),$$

$Q_m^m(x)$ contains all the m-multiple factors of $P(x)$.

Finding Q_i, $i=1, \ldots, l$, for a given P is called the *square-free decomposition* [5, 8] and it is a very important operation in algebraic computation.

So far, only exact arithmetic has been used for coefficients in performing the square-free decomposition. In this paper, we generalize the decomposition to polynomials with coefficients of floating-point numbers so that approximately equal factors may be separated as approximately multiple factors. One may think that the generalization is trivial, but this is not true. When handling polynomials with coefficients of approximate numbers, we must investigate the accuracy of the coefficients carefully, which is completely unnecessary in the conventional exact decomposition. We introduce a concept of approximate GCD (*greatest common divisor*) in **2.**, and in **3.**, we investigate the Euclidean algorithm for calculating approximate GCD. Since the algorithm causes strong cancellation of numbers, the accuracy of polynomial coefficient is investigated in **4.** On the basis of theory of approximate GCD, we present an algorithm of approximate square-free decomposition and many examples in **5.**

We can apply the approximate square-free decomposition to solving algebraic equations nicely. As is well-known, an equation having multiple and/or close roots is not easy to solve numerically (*ill-conditioned equation*). Although many authors investigated this problem [1, 2, 3, 7], the ill-conditioned equation is still attracting many researchers. We note that, even if an equation is ill-conditioned, we can find a root easily so long as an approximate value and "multiplicity" of multiple/close roots are known. (By "multiplicity" of close roots, we mean the number of mutually close roots.) The approximate square-free decomposition gives us such information nicely as we will explain later. In **6.**, we present an effective algorithm for solving ill-conditioned equations numerically.

2. Approximate GCD (greatest common divisor)

Throughout this paper we treat only univariate polynomials with coefficients of floating-point numbers (may be complex). By ε and δ (with indices, if necessary), we mean small-magnitude numbers, with δ representing the distance between mutually close roots. The ε_M denotes the machine epsilon (if M bits are used to represent the mantissa of a floating-point number then $\varepsilon_M=2^{-M}$).

Definition 1 [*degree*, *leading coefficient* and *leading term*]. Let

$$P(x)=a_l x^l+\cdots+a_0,\ a_l\neq 0. \tag{1}$$

The l and a_l are called degree and leading coefficient, respectively, and written as $\deg(P)$ and $\mathrm{lc}(P)$. The term $a_l x^l$ is called leading term. //

Definition 2 [*maximum magnitude coefficient*]. The absolute value of the maximum magnitude coefficient of

*The Institute of Physical and Chemical Research, Wako-shi, Saitama 351-01, Japan.
**Faculty of Engineering, Ehime University, Matsuyama-shi, Ehime 790, Japan.

$P(x)$ is written as mmc(P):

$$\text{mmc}(P)=\text{Max}\{|a_l|,\ldots,|a_0|\}.// \tag{2}$$

Definition 3 [*numbers of similar magnitude*]. If a number g is such that $h/c\leqq|g|\leqq hc$, where h and c are positive numbers and c is not much different from 1, then we denote $g=O(h)$.//

(Note). Usage of "O" is not the same as Landau's symbol "O".

Definition 4 [*regular polynomial*]. The $P(x)$, defined in (1), is called regular if $\text{lc}(P)=O(1)$ and Max $\{|a_{l-1}|,\ldots,|a_0|\}$ = either $O(1)$ or 0.//

(Note 1). Any univariate polynomial can be made regular by scaling transformations $P\to\xi P$ and $x\to\eta x$, where ξ and η are numbers.

(Note 2). Let u be a root of $P(x)=0$, then $|u|\lesssim O(1)$ if $P(x)$ is regular. Hence, if the relative distance between some roots of $P(x)=0$ is much less than 1 then we call them *close roots*.

Definition 5 [*polynomial of small coefficients*]. Let ε be a small positive number, $0<\varepsilon\ll 1$. If $P(x)$, given in (1), is such that $\text{mmc}(P)=O(\varepsilon)$ then we write $P(x)=O(\varepsilon(x))$. If $|a_i|<\varepsilon$, $i=l,\ldots,0$, then we write $P(x)=0$ (cutoff ε).//

Definition 6 [*approximate GCD*]. Let ε be a small positive number, $0<\varepsilon\ll 1$. If polynomials $P_1(x)$ and $P_2(x)$ are such that $\text{mmc}(P_1)=O(1)$, $\text{mmc}(P_2)=O(1)$, and

$$\begin{cases}P_1(x)=D(x)\tilde{P}_1(x)+O(\varepsilon(x)),\\ P_2(x)=D(x)\tilde{P}_2(x)+O(\varepsilon(x)),\end{cases} \tag{3}$$

then D is called an *approximately common divisor* of P_1 and P_2 *with accuracy* ε. Among the approximately common divisors of P_1 and P_2 with accuracy ε, a polynomial of the largest degree is called *approximate GCD with accuracy* ε and written as $\text{GCD}(P_1, P_2; \varepsilon)$. We usually normalize D as $\text{lc}(D)=1$ or $\text{mmc}(D)=O(1)$.//

(Note). The number of $\text{GCD}(P_1, P_2; \varepsilon)$ is not one but infinite in general. In fact, $D(x)+O(\varepsilon(x))$, with D given in (3), is also an approximately common divisor of P_1 and P_2 with accuracy ε. Furthermore, if $\varepsilon'>\varepsilon$ then deg $(\text{GCD}(P_1, P_2; \varepsilon'))$ may be greater than $\deg(\text{GCD}(P_1, P_2; \varepsilon))$. Hence, the above definition of approximate GCD is quite vague for some value of ε. However, it does not cause any serious problem practically.

Let us investigate the relationship between the accuracy ε of $\text{GCD}(P_1, P_2; \varepsilon)$ and the distance δ between mutually close roots of $P_1=0$ and $P_2=0$.

Lemma 1. Let δ be a small positive number and let

$$\begin{cases}D_1=(x-u-\delta_1)\ldots(x-u-\delta_m),\\ \qquad |\delta_i|\leqq O(\delta),\ i=1,\ldots,m,\\ D_2=(x-u-\delta_1')\ldots(x-u-\delta_n'),\\ \qquad |\delta_j'|\leqq O(\delta),\ j=1,\ldots,n,\end{cases} \tag{4}$$

where $m\geqq n$. Let $D=\text{GCD}(D_1, D_2; \varepsilon)$ and $\deg(D)=n$, then

$$\begin{cases}\varepsilon\geqq O(m\delta_{av}-n\delta_{av}'),\\ \delta_{av}=(\delta_1+\cdots+\delta_m)/m,\ \delta_{av}'=(\delta_1'+\cdots+\delta_n')/n.\end{cases} \tag{5}$$

(Proof). Let the approximate GCD be

$$D=(x-u-\bar{\delta}_1)\ldots(x-u-\bar{\delta}_n),$$

then $|\bar{\delta}_j|\leqq O(\delta)$, $j=1,\ldots,n$. We can express D as

$$D=(x-u)^n-(x-u)^{n-1}(\bar{\delta}_1+\cdots+\bar{\delta}_n)+O(\delta^2(x)).$$

Expressing D_1 and D_2 similarly, we find

$$D_1=D\times(x-u)^{m-n}-(x-u)^{m-1}[(\delta_1+\cdots+\delta_m)-(\bar{\delta}_1+\cdots+\bar{\delta}_n)]+O(\delta^2(x)),$$

$$D_2=D-(x-u)^{n-1}[(\delta_1'+\cdots+\delta_n')-(\bar{\delta}_1+\cdots+\bar{\delta}_n)]+O(\delta^2(x)).$$

By determining $\bar{\delta}_1,\ldots,\bar{\delta}_n$ so that coefficients of $(x-u)^{m-1}$ term of D_1 and $(x-u)^{n-1}$ term of D_2 become smallest simultaneously, we obtain Eq. (5).//

(Note). If $m\delta_{av}=n\delta_{av}'$ then terms of coefficients of $O(\delta^2)$ become dominant in D_1 and D_2, hence $\varepsilon=O(m\delta_{av}-n\delta_{av}')$ so long as $|m\delta_{av}-n\delta_{av}'|>O(\delta^2)$.

Theorem 1. Let δ be a small positive number and let

$$\begin{cases}P_1=(x-u-\delta_1)\ldots(x-u-\delta_m)\tilde{P}_1,\\ \qquad |\delta_i|\leqq O(\delta),\ i=1,\ldots,m,\\ P_2=(x-u-\delta_1')\ldots(x-u-\delta_n')\tilde{P}_2,\\ \qquad |\delta_j'|\leqq O(\delta),\ j=1,\ldots,n,\end{cases} \tag{6}$$

where $m\geqq n$ and $\tilde{P}_1$ and $\tilde{P}_2$ are regular polynomials having no root in the δ''-neighborhood of $x=u$ with δ'' sufficiently larger than δ. Let $D=\text{GCD}(P_1, P_2; \varepsilon)$ and D contain $(x-u)^n$ as an approximate factor. If every factor in D is an approximately common divisor of P_1 and P_2 with accuracy $\lesssim O(\delta)$, then $\varepsilon=O(\delta)$ unless a special relation holds among close roots of relative distance $\lesssim O(\delta)$. If, however, $P_2(x)=dP_1(x)/dx$, then $\varepsilon=O(\delta^2)$ unless a special relation holds among close roots of relative distance $\lesssim O(\delta)$.

(Proof). The first claim of the theorem is a direct consequence of Lemma 1. Next, putting $\delta_{av}=(\delta_1+\cdots+\delta_m)/m$, we have

$$P_1=[(x-u)^m-(x-u)^{m-1}(\delta_1+\cdots+\delta_m)+O(\delta^2(x))]\tilde{P}_1$$
$$=[(x-u-\delta_{av})^m+O(\delta^2(x))]\tilde{P}_1. \tag{7}$$

Hence, if $P_2=dP_1/dx$, then

$$P_2=(x-u-\delta_{av})^{m-1}\tilde{P}_2'+O(\delta^2(x)). \tag{8}$$

That is, $(x-u-\delta_{av})^{m-1}$ is an approximately common divisor of P_1 and P_2 and its accuracy is $O(\delta^2)$ unless the coefficients of $O(\delta^2)$ terms in Eq. (7) cancel accidentally.//

3. Euclidean Algorithm for Approximate GCD

We must mention that the theory developed in this section is not applicable to some polynomials of high

degrees. By restricting ourselves to handling only polynomials of low or medium degrees, we are able to prove beautiful and strong theorems. Note, however, that the restriction is only to satisfy some weak conditions. In particular, we often use the fact: "if degrees of polynomials F and G are not large, then

$$[\mathrm{mmc}(F)\times\mathrm{mmc}(G)]/\mathrm{mmc}(FG)=O(1).\text{''}$$

We will calculate an approximate GCD by the generalized Euclidean algorithm which computes remainders successively. We impose the following three rules for the remainder computation.

Rule 1 [*leading term elimination*]. Let $F(x)$ and $G(x)$ be

$$\begin{cases}F(x)=f_l x^l+\cdots+f_0, f_l\neq 0,\\ G(x)=g_m x^m+\cdots+g_0, g_m\neq 0, l\geqq m.\end{cases}\tag{9}$$

We eliminate the leading term (x^l term) as

$$\tilde{F}(x)=[F(x)-f_l x^l]-x^{l-m}(f_l/g_m)[G(x)-g_m x^m]$$
$$=\sum_{i=1}^{l}[f_{l-i}-(f_l/g_m)g_{m-i}]x^{l-i}.\ //\tag{10}$$

Rule 2 [*zero coefficient*]. If x^{l-i} term of the above equation satisfies

$$|f_{l-i}-(f_l/g_m)g_{m-i}|=O(\varepsilon_M|f_{l-i}|),\tag{11}$$

then we discard the term as a zero coefficient term.//

Rule 3 [*normalization of remainder*]. Let Q be the quotient of the division of F by G, then we normalize the remainder R as

$$F=QG+\mathrm{Max}\{1, \mathrm{mmc}(Q)\}\times R.//\tag{12}$$

With Rule 1, we are free from error in $f_l-(f_l/g_m)g_m$ which must be 0 theoretically but may not be 0 actually. If $|g_m|=O(\varepsilon_M|f_l|)\neq 0$, which may happen in the floating-point arithmetic, then the elimination is meaningless ($\tilde{F}$ in Eq. (10) is almost proportional to G). Rule 2 is imposed to avoid such hazardous situation. As will see in **4.**, in the calculation of approximate GCD by the Euclidean algorithm, the magnitude of the coefficient of polynomial contains very important information. The above normalization in Rule 3 is determined so as to preserve the information during the calculation. We will explain the meaning of the normalization in **4.**

With the above rules for polynomial and coefficient arithmetic, we calculate the so-called *polynomial remainder sequence* (abbreviated to PRS).

Definition 7 [*PRS with cutoff* ε]. Given regular polynomials $P_1(x)$ and $P_2(x)$, $\deg(P_1)\geqq\deg(P_2)$, and a small positive number ε, $0<\varepsilon\ll 1$, we calculate a sequence of polynomials

$$(P_1, P_2, \ldots, P_k\neq 0 \text{ (cutoff } \varepsilon), P_{k+1}=0 \text{ (cutoff } \varepsilon))\tag{13}$$

by the iteration formula

$$P_{i-1}=Q_i P_i+\mathrm{Max}\{1, \mathrm{mmc}(Q_i)\}\times P_{i+1},\ i=2, \ldots, k,\tag{14}$$

as well as Rules 1 and 2. We call the sequence (13) a PRS of P_1 and P_2 with cutoff ε, and we write P_k as

$$P_k=\mathrm{GCD}'(P_1, P_2; \varepsilon).//\tag{15}$$

(Note). Although the length of PRS depends on ε, polynomials $P_3, \ldots, P_k$ are not dependent on ε and $\mathrm{GCD}'(P_1, P_2; \varepsilon)$ is uniquely determined.

As we have noted below Def. 6, $D=\mathrm{GCD}(P_1, P_2; \varepsilon)$ is not unique: when $\deg(D)$ is fixed, the accuracy ε will be small for some D while ε will be large for other D. Then, a question arises: what is the accuracy of $D'=\mathrm{GCD}'(P_1, P_2; \varepsilon')$? The following theorem shows that D' is an approximate GCD of almost the smallest accuracy for a given $\deg(D')$.

Lemma 2. For each P_i in PRS (13), there exist polynomials A_i and B_i such that

$$\begin{cases}P_i=A_i P_1+B_i P_2\ (+O(\varepsilon_M(x))),\\ \deg(A_i)<\deg(P_2)-\deg(P_i),\\ \deg(B_i)<\deg(P_1)-\deg(P_i).\end{cases}\tag{16}$$

Furthermore, if $\deg(P_1)$ and $\deg(P_2)$ are not large, then

$$\mathrm{mmc}(A_i)\leqq O(1),\ \mathrm{mmc}(B_i)\leqq O(1).\tag{17}$$

(Proof). The A_i and B_i satisfying Eq. (16) are calculated by the extended Euclidean algorithm: put $A_1=1, B_1=0, A_2=0, B_2=1$, and calculate A_{i+1} and B_{i+1} by the iteration formula

$$A_{i+1}=(A_{i-1}-Q_i A_i)/c_i,\ B_{i+1}=(B_{i-1}-Q_i B_i)/c_i,\tag{18}$$

where Q_i is the quotient in Eq. (14) and $c_i=\mathrm{Max}\{1, \mathrm{mmc}(Q_i)\}$. Hence, $\mathrm{mmc}(Q_i)/c_i\leqq 1$. Since we assumed that $\deg(P_1)$ is not large, the interation in Eq. (18) does not cause the coefficients of A_{i+1} and B_{i+1} large and we obtain Eq. (17).//

(Note). Since $\deg(A_i)>\deg(A_{i-1})$ and $\deg(B_i)>\deg(B_{i-1})$, formulas in (18) show that $\mathrm{mmc}(A_{i+1})=O(1)$ and $\mathrm{mmc}(B_{i+1})=O(1)$ in most cases even if $\mathrm{mmc}(P_{i+1})$ becomes small.

Theorem 2. Let P_1 and P_2 be regular polynomials and let $D=\mathrm{GCD}(P_1, P_2; \varepsilon)$ be an approximate GCD of the smallest accuracy ε among approximate GCD's of degree $d=\deg(D)$, where we set $\mathrm{lc}(D)=1$. Let $(P_1, P_2, \ldots, P_k, \ldots)$ be a PRS calculated by formula (14). If $\deg(P_k)=d$ and $\deg(P_1)$ is not large then

$$P_i=D\tilde{P}_i+O(\varepsilon(x)).\tag{19}$$

That is, P_i, $2<i\leqq k$, contains D as an approximate factor of accuracy $\varepsilon/\mathrm{mmc}(P_i)$ at greatest. In particular,

$$P_k=\text{constant}\times D+O(\varepsilon(x)).\tag{19'}$$

(Proof). By assumption, there exist polynomials $\tilde{P}_1$ and $\tilde{P}_2$ such that

$$P_1=D\tilde{P}_1+\varepsilon_1(x),\ \varepsilon_1(x)=O(\varepsilon(x)),$$
$$P_2=D\tilde{P}_2+\varepsilon_2(x),\ \varepsilon_2(x)=O(\varepsilon(x)).$$

Substituting these into Eq. (16), we have

$$P_i = A_i P_1 + B_i P_2$$
$$= D(A_i \tilde{P}_1 + B_i \tilde{P}_2) + A_i \varepsilon_1(x) + B_i \varepsilon_2(x).$$

By division, we rewrite the last two terms of this equation as

$$A_i \varepsilon_1(x) + B_i \varepsilon_2(x) = D\xi_i(x) + \eta_i(x), \ \deg(\eta_i) < \deg(D).$$

Thus, we have

$$P_i = D \times [A_i \tilde{P}_1 + B_i \tilde{P}_2 + \xi_i(x)] + \eta_i(x).$$

Since P_1 and P_2 are regular polynomials of degrees not large and $\mathrm{lc}(D)=1$, we have $\mathrm{mmc}(D)=O(1)$. Furthermore, Lemma 2 tells that $\mathrm{mmc}(A_i) \leqq O(1)$ and $\mathrm{mmc}(B_i) \leqq O(1)$. Hence, $\eta_i(x)=O(\varepsilon(x))$ and the main part of the theorem is proved. For $i=k$, we have $[A_k \tilde{P}_1 + B_k \tilde{P}_2 + \xi_k(x)] = \text{constant}$ because $\deg(P_k)=\deg(D)$. Hence, we have Eq. (19'). //

(Note). In the division by D, we must be careful about the magnitude of $\mathrm{lc}(D)$. If, for example, $|\mathrm{lc}(D)| \ll \mathrm{mmc}(D)$ then the above theorem does not hold, because the division may give quotient and remainder with coefficients of much larger magnitude than those of dividend.

Theorem 2 tells us that Euclidean algorithm calculates GCD of almost the smallest accuracy. Next, we investigate the magnitude of coefficients of polynomials in PRS. Below, by $\mathrm{quo}(F, G)$ and $\mathrm{rem}(F, G)$ we denote the quotient and remainder, respectively, of the division of F by G.

Lemma 3. Let ε be a small positive number, $0<\varepsilon \ll 1$, F and G be polynomials such that $\mathrm{mmc}(F)=O(1)$, $\mathrm{mmc}(G)=O(1)$, $\mathrm{lc}(G)=O(1)$, and

$$\begin{cases} F = D\tilde{F} + \varepsilon_1(x), \ \varepsilon_1(x) = O(\varepsilon(x)), \\ G = D\tilde{G} + \varepsilon_2(x), \ \varepsilon_2(x) = O(\varepsilon(x)). \end{cases} \tag{20}$$

If $|\mathrm{lc}(D)| = O(1)$ and degrees of F and G are not large, then

$$\mathrm{rem}(F, G) = D \times \mathrm{rem}(\tilde{F}, \tilde{G}) + O(\varepsilon(x)). \tag{21}$$

(Proof). Put $\tilde{Q}=\mathrm{quo}(\tilde{F}, \tilde{G})$ and $\tilde{R}=\mathrm{rem}(\tilde{F}, \tilde{G})$, then $\tilde{F}=\tilde{Q}\tilde{G}+\tilde{R}$. Hence, $D\tilde{F}=D\tilde{G}\tilde{Q}+D\tilde{R}$ and we can rewrite F as

$$F = D\tilde{G}\tilde{Q} + D\tilde{R} + \varepsilon_1(x)$$
$$= \tilde{Q} \times [D\tilde{G} + \varepsilon_2(x)] + D\tilde{R} - \tilde{Q}\varepsilon_2(x) + \varepsilon_1(x).$$

By division, we rewrite the last two terms of this equation as

$$-\tilde{Q}\varepsilon_2(x) + \varepsilon_1(x) = G\xi(x) + \eta(x), \ \deg(\eta) < \deg(G).$$

Thus, we have

$$F = [\tilde{Q} + \xi(x)] \times G + D\tilde{R} + \eta(x).$$

Since $\mathrm{mmc}(G)=O(1)$, $\mathrm{lc}(G)=O(1)$ and $|\mathrm{lc}(D)|=O(1)$, we have $\mathrm{mmc}(\tilde{Q})=O(1)$. Hence, $\eta(x)=O(\varepsilon(x))$ and we obtain Eq. (21). //

Theorem 3. Let P_1 and P_2 be regular polynomials and $D=\mathrm{GCD}(P_1, P_2; \varepsilon)$, with ε a small positive number. Let $(P_1, P_2, \ldots, P_k, \ldots)$ be a PRS of P_1 and P_2. If $\deg(P_k)=\deg(D)$ and $\deg(P_1)$ is not large, then

$$P_{k+1} = O(\varepsilon(x)). \tag{22}$$

(Proof). Put $P_i' = P_i / \mathrm{mmc}(P_i)$, then Theorem 2 tells us that P_{k-1}' and P_k' contain D as an approximate factor of accuracies $\varepsilon/\mathrm{mmc}(P_{k-1})$ and $\varepsilon/\mathrm{mmc}(P_k)$, respectively. First, consider the case $\mathrm{mmc}(P_{k-1}) \geqq \mathrm{mmc}(P_k)$. In this case, we may think that the accuracy of D is $\hat{\varepsilon} \equiv \varepsilon/\mathrm{mmc}(P_k)$. Let

$$P_{k-1}' = Q_k' P_k' + P_{k+1}', \ \deg(P_{k+1}') < \deg(P_k'),$$

then $\mathrm{mmc}(Q_k')=O(1)$. Comparing this equation with Eq. (14), we obtain

$$c_k P_{k+1} = \mathrm{mmc}(P_{k-1}) \times P_{k+1}',$$

where $c_k = [\mathrm{mmc}(P_{k-1})/\mathrm{mmc}(P_k)] \times \mathrm{mmc}(Q_k') \approx \mathrm{mmc}(P_{k-1})/\mathrm{mmc}(P_k)$. Now, Lemma 3 is applicable to the case $F=P_{k-1}'$ and $G=P_k'$, hence $P_{k+1}'=O(\hat{\varepsilon}(x))$ and we obtain Eq. (22). Next, consider the case $\mathrm{mmc}(P_{k-1}) < \mathrm{mmc}(P_k)$. In this case, we may think that the accuracy of D is $\hat{\varepsilon} \equiv \varepsilon/\mathrm{mmc}(P_{k-1})$, and we see $c_k = \mathrm{Max}\{1, [\mathrm{mmc}(P_{k-1})/\mathrm{mmc}(P_k)] \times \mathrm{mmc}(Q_k')\} \approx 1$. Hence, we obtain Eq. (22) again. //

Using Theorems 1, 2 and 3, we can calculate approximately common factor, of root-separation $\lesssim \delta$, of regular polynomials P_1 and P_2 as follows. Let $\varepsilon = O(\delta)$ if P_1 and P_2 are arbitrary or $\varepsilon = O(\delta^2)$ if $P_2 = dP_1/dx$. (We set $\varepsilon=\delta$ or $\varepsilon = 2 \times \delta^2$ in our program.) Calculate PRS of P_1 and P_2 with cutoff ε, then the last element of PRS is the required PRS.

4. Accuracy of Coefficient and Examples of PRS

This section has two aims, one is to investigate the accuracy of coefficients in PRS, and the other is to check the theory developed above by examples.

Theorem 3 tells us that some polynomials in PRS are such that their coefficients are of very small magnitude if P_1 and P_2 have approximately common factor, and this magnitude reduction of coefficient is caused by cancellation of almost equal numbers. The cancellation of numbers leads to erroneous result in the floating-point arithmetic. Therefore, investigation of accuracy of coefficients in PRS is indispensable. Let us first define the accuracy of number.

Definition 8 [*accuracy of number*]. Let a and b be floating-point numbers accurate to the last bit and $|a-b| = O(\varepsilon \times \mathrm{Max}\{|a|, |b|\})$, with $0<\varepsilon<1$, then we say that accuracy of the number $a-b$ is ε. (Fully accurate number is of accuracy 1, and a number of accuracy ε_M is wholly erroneous and insignificant.) //

Now, we consider the meaning of Rule 3. The division is a successive application of leading term elimination, so consider Eq. (10) as well as Eq. (12). If $|f_l| \leqq |g_m|$ then elimination of x^l term in F gives (f_l/g_m) x^{l-m} term in Q, and this term does not affect the normalization of R. In this case, G is multiplied by f_l/g_m to

cancel the leading term. On the other hand, if $|f_l| > |g_m|$ then the magnitude of the coefficients of $\tilde{F}$ is reduced by $|g_m/f_l|$ to give R. In this case, we can regard F as multiplied by g_m/f_l to cancel the leading term (the result is the normalized remainder). Summarizing both the above cases, we may say that R in Eq. (12) can be calculated by adjusting the polynomial with larger magnitude leading coefficient to the other and canceling the leading term.

Suppose that the accuracy of a number is nearly equal to the magnitude of the number. Let a, b, c_a and c_b be numbers such that $|a| \leqq 1$ and $|b| \leqq 1$, and consider $c_a a - c_b b = d$. As we have discussed above, we may regard Rule 3 as if requiring $c_a = 1 \geqq |c_b|$ when $|b| \geqq |a|$ and $c_b = 1 > |c_a|$ when $|a| > |b|$. Then, the accuracy of d is almost equal to the magnitude of d regardless of cancellation in $c_a a - c_b b$. We recall that $\mathrm{mmc}(P_1) = O(1)$ and $\mathrm{mmc}(P_2) = O(1)$ in PRS (13). Hence, Rule 3 leads to the following important results.

Property 1. Let P_1 and P_2 be regular polynomials, with their coefficients accurate to the last bit. Let $(P_1, P_2, \ldots, P_i, \ldots)$ be a PRS calculated by formula (14), then $\mathrm{mmc}(P_i)$ is nearly equal to the accuracy of coefficients of P_i.

Property 2. Let P_1 and P_2 contain approximately common factor D_1 of accuracy ε_1, D_2 of accuracy ε_2, and so on, where ε_1 is enough larger than ε_2, ε_2 is enough larger than ε_3, and so on. Let $P_{k1}, P_{k2}, \ldots$, be such that $\deg(P_{k1}) = \deg(D_1)$, $\deg(P_{k2}) = \deg(D_2)$, ... Then, the accuracy of coefficients of P_i decreases to $O(\varepsilon_1)$ at $i = k1+1$, decreases to $O(\varepsilon_2)$ at $i = k2+1$, and so on.

Property 3. If P_1 and P_2 have the same roots but no close root, then the common factor can be separated very accurately.

Let us show several examples of PRS calculation. The calculation was performed by double-precision arithmetic and the 9 MSD (*most significant digits*) of each number are printed (this unfamiliar output style is due to a host Lisp system on which our program runs). In the examples, we show not only the PRS but also polynomials A_i and B_i defined in Eq. (16). This will help the reader's deeper understanding of PRS. Below, we classify the PRS into normal and abnormal, as follows.

Definition 9 [*abnormal PRS*]. If the PRS (13) is such that

$$|\mathrm{lc}(P_i)| \ll O(\mathrm{mmc}(P_i)) \text{ for some } i,\ 3 \leqq i \leqq k,$$

then the PRS is called abnormal, otherwise the PRS is normal. //

Example 1. Normal PRS.

$P_1 := (X-0.5)*(X-0.502)*(X+1)*(X-2)*(X-1.5)$

$P_2 := (X-0.501)*(X-0.503)*(X-1)*(X+2)*(X+1.5)$

$P_3 = -4.998*X**4+5.013997*X**3+4.7414925*X**2-6.0174985*X+1.509009$
$A_3 = 1,\ B_3 = -1$

$P_4 = 6.97880794\mathrm{E}-1*X**3-7.01037162\mathrm{E}-1*X**2+1.78930204\mathrm{E}-1*X-1.4439101\mathrm{E}-3$
$A_4 = 2.00080032\mathrm{E}-1*X+5.00040152\mathrm{E}-1$
$B_4 = -2.00080032\mathrm{E}-1*X+4.99959848\mathrm{E}-1$

$P_5 = 8.40067492\mathrm{E}-1*X**2-8.41442765\mathrm{E}-1*X+2.10704053\mathrm{E}-1$
$A_5 = 2.00080032\mathrm{E}-1*X**2+5.0030468\mathrm{E}-1*X+1.40293119\mathrm{E}-1$
$B_5 = -2.00080032\mathrm{E}-1*X**2+4.9969532\mathrm{E}-1*X-1.3897101\mathrm{E}-1$

$P_6 = 1.87196957\mathrm{E}-3*X-9.38795693\mathrm{E}-4$
$A_6 = -1.6621523\mathrm{E}-1*X**3-4.15145325\mathrm{E}-1*X**2+8.47317694\mathrm{E}-2*X+5.00376473\mathrm{E}-1$
$B_6 = 1.6621523\mathrm{E}-1*X**3-4.15598395\mathrm{E}-1*X**2-8.34328336\mathrm{E}-2*X+4.99626697\mathrm{E}-1$

$P_7 = -1.39801471\mathrm{E}-9$
$A_7 = 1.6621523\mathrm{E}-1*X**4+3.32015185\mathrm{E}-1*X**3-2.91914844\mathrm{E}-1*X**2$
$\quad -4.56884248\mathrm{E}-1*X+2.50568671\mathrm{E}-1$
$B_7 = -1.6621523\mathrm{E}-1*X**4+4.98728536\mathrm{E}-1*X**3-1.24868536\mathrm{E}-1*X**2$
$\quad -5.40240922\mathrm{E}-1*X+2.49571382\mathrm{E}-1$

The approximately common factor of P_1 and P_2 is $\sim(X-0.5015)^2$ and its accuracy is ~ 0.001 by Theorem 1. Hence, Theorems 2 and 3 predict that if $P_k \cong (x-0.5015)^2$ then $\mathrm{mmc}(P_{k+1}) = O(0.001)$. We see that $P_k = P_5$ and the theory is well consistent with actual computation.

Example 2. PRS of P_1 and $P_2 = (dP_1/dX)/\mathrm{lc}(P_1)$.

$P_1 := (X+1)*(X-2)*(X-0.5)*(X-0.501)*(X-0.503)$

$P_2 := (dP_1/dX)/5$

$P_3 = -9.0000136\mathrm{E}-1*X**3+1.35432204*X**2-6.79323541\mathrm{E}-1*X+1.13581429\mathrm{E}-1$
$A_3 = 1,\ B_3 = -X+5.008\mathrm{E}-1$

$P_4 = -1.21499582*X**2 + 1.21823582*X - 3.05370171E-1$
$A_4 = X - 4.98400006E-1$
$B_4 = -X**2 + 9.99200006E-1*X + 6.50402637E-1$

$P_5 = 3.49999695E-6*X - 1.75299848E-6$
$A_5 = -7.40744406E-1*X**2 + 7.41139463E-1*X + 8.14618898E-1$
$B_5 = 7.40744406E-1*X**3 - 1.11210426*X**2 - 1.11012723*X + 7.42718852E-1$

$P_6 = 1.92857883E-12$
$A_6 = -7.40744406E-1*X**3 + 1.11285207*X**2 + 4.42710932E-1*X - 4.08784963E-1$
$B_6 = 7.40744406E-1*X**4 - 1.48381686*X**3 - 5.5206559E-1*X**2 + 1.29979415*X - 3.72701526E-1$

The distance between close roots around $X=0.5$ is ~ 0.001, so Theorem 1 tells us that P_1 and P_2 have approximately common factor of degree 2 with accuracy $\sim (0.001)^2$, and Theorem 3 predicts that $\mathrm{mmc}(P_{k+1}) \sim (0.001)^2$ where $\deg(P_{k+1}) = 1$. This is observed in the actual computation. Furthermore, we see that $P_5 \propto [X - (0.5 + 0.501 + 0.503)/3]^2$, as predicted by Theorems 1 and 2.

Example 3. Abnormal PRS.

$P_1 := (X - 0.5)*(X + 0.502)*(X + 1)*(X - 2)*(X - 1.5)$

$P_2 := (X - 0.501)*(X + 0.503)*(X - 1)*(X + 2)*(X + 1.5)$

$P_3 = -5.0*X**4 - 8.997E-3*X**3 + 7.2575075*X**2 + 1.14985E-2*X - 1.509009$
$A_3 = 1,\ B_3 = -1$

$P_4 = 6.99999639E-1*X**3 + 3.37121201E-4*X**2 - 1.76050589E-1*X + 1.44395856E-3$
$A_4 = 2.0E-1*X + 5.0004012E-1$
$B_4 = -2.0E-1*X + 4.9995988E-1$

$P_5 = 8.40000382E-1*X**2 + 2.82174861E-3*X - 2.11259248E-1$
$A_5 = 2.0E-1*X**2 + 5.0030368E-1*X + 1.4065888E-1$
$B_5 = -2.0E-1*X**2 + 4.9969632E-1*X - 1.39341082E-1$

$P_6 = 5.38066233E-6*X + 9.37355668E-4$
$A_6 = -1.66666505E-1*X**3 - 4.16439726E-1*X**2 + 8.39841165E-2*X + 5.00377422E-1$
$B_6 = 1.66666505E-1*X**3 - 4.16892799E-1*X**2 - 8.26842652E-2*X + 4.99625738E-1$

$P_7 = 9.373479E-4$
$A_7 = 9.56727001E-4*X**4 - 1.64275988E-1*X**3 - 4.16921818E-1*X**2$
$\quad + 8.11117845E-2*X + 5.00377427E-1$
$B_7 = -9.56727001E-4*X**4 + 1.69059623E-1*X**3 - 4.16418169E-1*X**2$
$\quad - 8.55522822E-2*X + 4.99625733E-1$

We see that $|\mathrm{lc}(P_6)| \approx 5 \times 10^{-6} \ll 9 \times 10^{-4} \approx \mathrm{mmc}(P_6)$. The division of P_5 by P_6 gives quotient and remainder with coefficients of much larger magnitude than those of P_5. However, formula (14) gives a reasonable P_7, and we see that the accuracy condition (Property 1 given above) is satisfied even by abnormal PRS. Note further that $\mathrm{mmc}(A_i) = O(1)$ and $\mathrm{mmc}(B_i) = O(1)$ regardless of whether the PRS is normal or not.

5. Approximate Square-free Decomposition

Let $P(x)$ be a polynomial such that

$$\begin{cases} P(x) = Q_1(x)Q_2^2(x) \cdots Q_l^l(x), \\ Q_m^m(x) \text{ contains all the } m\text{-multiple factors of } P(x). \end{cases} \quad (23)$$

The square-free decomposition is based on the following well-known relation

$$\mathrm{GCD}(P(x), dP(x)/dx) = Q_2(x)Q_3^2(x) \cdots Q_l^{l-1}(x). \quad (24)$$

Diving Eq. (23) by Eq. (24), we obtain

$$QQ \equiv \text{Eq. (23)} / \text{Eq. (24)} = Q_1(x)Q_2(x) \cdots Q_l(x). \quad (25)$$

Combining these relations, we can perform the decomposition as follows.

Algorithm Exact-SQFR (exact square-free decomposition)

Input: A polynomial $P(x)$ with exact number coefficients;

Output: A list $((Q_1, 1), (Q_2, 2), \ldots, (Q_l, l))$, where $Q_1, \ldots, Q_l$ satisfy (23);

Step 0: $QPP \leftarrow P$; $m \leftarrow 1$; ANS $\leftarrow$ nil;

Step 1: [A] $PP \leftarrow \mathrm{GCD}(QPP, dQPP/dx)$;
[B] $QQ \leftarrow QPP/PP$;
Append (QQ, m) to ANS;
If $PP = 1$ then go to **Step 2**;
$m \leftarrow m + 1$; $QPP \leftarrow PP$; Go to **Step 1**;

Step 2: Let ANS be $((QQ_1, 1), (QQ_2, 2), \ldots, (QQ_l, l))$;
For $m = 1$ to $l - 1$ replace (QQ_m, m) in ANS by $(QQ_m/QQ_{m+1}, m)$;

Return ANS.//

Extending Eq. (23), we define the approximate square-free decomposition.

Definition 10 [*"multiplicity" of close roots*]. The "multiplicity" of close roots is the number of mutually close roots.//

Definition 11 [*approximate square-free decomposition*]. Let ε be a small positive number, $0<\varepsilon\ll 1$, and $P(x)$ be a regular polynomial such that

$$P(x)=Q_1(x)Q_2^2(x)\ldots Q_l^l(x)+O(\varepsilon(x)), \qquad (26)$$

where each $Q_m^m(x)$ contains all the multiple/close factors, of "multiplicity" m, of $P(x)$. Finding Q_i, $i=1, \ldots, l$, for given P is called approximate square-free decomposition.//

(Note). Let δ be the average distance of mutually close roots of $P(x)=0$, then Theorem 1 tells us that $\varepsilon=O(\delta^2)$.

We can perform the approximate square-free decomposition by modifying Exact-SQFR slightly. The GCD operation in step [A] of Exact-SQFR may give a polynomial of wrong degree if approximate arithmetic is employed. On the other hand, if the algorithm works well, we have the following relation (in the notation in Exact-SQFR)

$$\deg(QQ_1)\geqq\deg(QQ_2)\geqq\ldots\geqq\deg(QQ_l). \qquad (27)$$

So long as this relation holds, the above algorithm gives us a reasonable answer regardless of exactness of the arithmetic used. Therefore, in each iteration in **Step 1** of Exact-SQFR, we check the relation (27) and restore it if violated. The restoration is done as follows: if deg $(QQ_{m-1})<\deg(QQ_m)$ then replace QQ_m by QQ_{m-1} and replace PP by quo(QPP/QQ_{m-1}). Thus, our algorithm is as follows.

Algorithm Approx-SQFR (approximate square-free decomp.)

Input: A polynomial $P(x)$ and a small number δ, $0<\delta\ll 1$;

Output: A list $((Q_1, 1), (Q_2, 2), \ldots, (Q_l, l))$, where $Q_1, \ldots, Q_l$ satisfy Eq. (26) with $\varepsilon=O(\delta^2)$;

***Same as Exact-SQFR except for the following two points:

1) interprete the division U/V as quotient(U, V),
2) change steps [A] and [B] as follows.

[A] $PP\leftarrow$GCD'$(QPP, dQPP/dx; 2\times\delta{**}2)$;

[B] $QQ\leftarrow QPP/PP$;
/* $QQQ\leftarrow QQ_{m-1}$, $QQ\leftarrow QQ_m$ */
If $m\geqq 2$ and $\deg(QQQ)<\deg(QQ)$ then do
begin $QQ\leftarrow QQQ$; $PP\leftarrow QPP/QQQ$;
end;
$QQQ\leftarrow QQ$;//

(Note). Eauations (24) and (25) give

$$\text{GCD(Eq. (24), Eq. (25))}=Q_2(x)Q_3(x)\ldots Q_l(x). \qquad (28)$$

Hence, we can calculate Q_1 as $Q_1=$Eq. (25)/Eq. (28). One may think that this method is more efficient than the above method, and this is true for algorithm Exact-SQFR. However, in the case of using approximate arithmetic, the above algorithm is better than using Eq. (28). The reason is as follows. According to the theory in **2.** and **3.**, we must calculate Eq. (28) as

$$\text{GCD'(Eq. (24), Eq. (25); } \delta)$$

while our algorithm calculates only

$$\text{GCD'}(P(x), dP(x)/dx; 2\times\delta^2)$$

and the former GCD is much more erroneous than the latter GCD.

Let us show the performance of algorithm Approx-SQFR for various kinds of polynomials having multiple/close roots. The algorithm was implemented on an algebraic computation system, constructed by one of the authors (T.S.), running on a Lisp system. The calculations were made by using double-precision arithmetic, and the 9 MSD of each number are printed. The reader can read the following output as follows: if only Q_3 and Q_2 are printed, for example, then $P(X)=Q_2^2(X)Q_3^3(X)+O(\delta^2(X))$. The value of δ is written before Q_i.

Example 4.1 [multiple roots and no close roots].

```
P(X):=(X+1)**3*(X-2.0/3.0)**2*(X+4.0/3.0)
      **2*(X-2)
***δ=0.01***
  Q3=X+1.0
  Q2=X**2+6.66666667E-1*X-8.88888889E-1
  Q1=X-2.0
P(X):=(X+1)**4*(X-1)**3*(X+0.555)**3
      *(X-2)*(X-3)
***δ=0.01***
  Q4=X+1.0
  Q3=X**2-4.45E-1*X-5.55E-1
  Q1=X**2-5.0*X+6.0
```

Example 4.2 [close roots and no multiple roots].

```
P(X):=(X+1)*(X-2)*(X-0.5)*(X-0.501)
      *(X-0.6)*(X-0.601)
***δ=0.01***
  Q2=X**2-1.10100009*X+3.00551047E-1
  Q1=X**2-9.99999819E-1*X-2.00000208
P(X):=(X+1)*(X-2)*((X-0.5)**4+0.00000001)
***δ=0.01***
  Q4=X-5.0E-1
  Q1=X**2-X-2.00000002
```

Example 4.3 [close roots as well as multiple roots].

```
P(X):=(X+1)*(X-1)**2*(X-0.5)*(X-0.501)
      *(X-2)
***δ=0.01***
  Q2=X**2-1.50050165*X+5.0050165E-1
  Q1=X**2-9.99996699E-1*X-2.0000019
P(X):=(X+1)**2*(X-2)**2*(X-0.5)*(X-0.501)
      *(X-0.503)
```

```
***δ=0.01***
  Q3=X-5.0133334E-1
  Q2=X**2-9.99999987E-1*X-2.00000156
```

Example 4.4 [close roots of different distances].

```
P(X):=(2*X**2-1)*(X-29.0/41.0)*(X-70.0/99.0)
***δ=0.01***
  Q3=X-7.07164833E-1
  Q1=X+7.07106719E-1
***δ=0.0001***
  Q2=X-7.07094337E-1
  Q1=X**2-1.99106643E-4*X-5.00140792E-1
P(X):=(X+1)*(X-1)*(X-1.1)*(X-1.01)
      *(X-1.001)*(X-2)
***δ=0.1***
  Q4=X-1.02937394
  Q1=X**2-9.93504252E-1*X-2.00131081
***δ=0.01***
  Q3=X-1.00421665
  Q1=X**3-2.09835006*X**2-9.01823625E-1*X
      +2.19670084
***δ=0.001***
  Q2=X-1.00062163
  Q1=X**4-3.10975675*X**3+1.22048665*X**2
      +···
***δ=0.0001***
  Q1=X**6-5.111*X**5+8.44511*X**4-···
```

Example 4.5 [close roots some of which are multiple].

```
P(X):=(X+1)*(X-2)*(X-0.5)**2*(X-0.501)
      *(X-0.503)
***δ=0.01***
  Q4=X-5.01000003E-1
  Q1=X**2-9.99999988E-1*X-2.00000151
***δ=0.0001***
  Q2=X-4.99999991E-1
  Q1=X**4-2.00400002*X**3-7.43996973E-1*X
      **2+···
P(X):=(X+1)*(X-2)
      *(X-0.5)**2*(X-0.501)*(X-0.503)
      *(X+0.5)**2*(X+0.501)*(X+0.503)
***δ=0.01***
  Q4=X**2+5.63119694E-6*X-2.5100488E-1
  Q1=X**2-1.00002242*X-1.99998004
***δ=0.0001***
  Q2=X**2-4.6684407E-8*X-2.5000001E-1
  Q1=X**6-9.99999907E-1*X**5-2.50401007*X
      **4+···
```

Summarizing the above results, we may say that the approximate square-free decomposition is quite successful. In fact, although the Approx-SQFR is equipped with a mechanism of recovering from failure of GCD operation (step [B] in Approx-SQFR), the mechanism is not used for all of the above examples. Furthermore, we can observe the following points: 1) Approx-SQFR separates the multiple factors very accurately if there is no close root factor; 2) Approx-SQFR works well even if $P(x)=0$ contains close roots of various distances.

6. Root-finding Algorithm for Ill-conditioned Equations

Now, we state how to solve ill-conditioned algebraic equations by using approximate square-free decomposition. We first note that, when applying the approximate square-free decomposition to solving ill-conditioned equations, we had better choose a mediumly small number as δ (root-separation number). If δ is very small then PRS may become erroneous when the equation contains close roots of various distances, as Property 2 in **4.** tells. In the following algorithms, we choose $\delta=0.01$. Our algorithm is composed of main procedure FIND-Roots and sub-procedure FIND-CloseRoots, where we use the notation

$$P^{(m)}(u)=d^mP(x)/dx^m|_{x=u}.$$

Procedure FIND-Roots($\hat{P}(\hat{x})$, $\hat{\delta}$)

Input: $\hat{P}(\hat{x})$=a polynomial in $\hat{x}$, may be irregular;
$\hat{\delta}$=a small positive number such that close roots of distance $<\hat{\delta}$ are treated as multiple roots;

Output: All the roots of $\hat{P}(\hat{x})=0$ and their multiplicities;

Step 0: $P(x)\leftarrow$regularize $\hat{P}(\hat{x})$, where $x=c\hat{x}$ with c a number;

Step 1: Apply Approx-SQFR to $P(x)$ and find $Q_1, \ldots,$ Q_l such that
[C] $P(x)=Q_1(x)Q_2^2(x)\ldots Q_l^l(x)+O(\varepsilon(x))$, $\varepsilon=2\times10^{-4}$;

Step 2: For each Q_m, $m=1, \ldots, l$, solve $Q_m(x)=0$ by low precision arithmetic, and let the roots obtained be $u_{m1}, u_{m2}, \ldots$;

Step 3: For each u in $\{u_{11}, u_{12}, \ldots\}$, solve $P(x)=0$ accurately by Newton's method with initial approximation $x=u$;

Step 4: For each u in $\{u_{m1}, u_{m2}, \ldots\}$, $m=2, \ldots, l$, determine the multiple/close roots around $x=u$ accurately by FIND-CloseRoots($P(x)$, u, m, $\hat{\delta}$);
/* close roots around $x=u$ */

Step 5: Let the roots and their multiplicities be $\{(u_1, m_1), (u_2, m_2), \ldots\}$;
Return $\{(u_1/c, m_1), (u_2/c, m_2)\ldots\}$.//

Let us explain the above procedure. The square-free decomposition [C] is such that the close roots with relative distance less than ~0.01 are factored out as approximately equal roots. Hence, the roots of $Q_m(x)=0$, $m=1, \ldots, l$, are well distant from each other and we can find them by conventional Newton's method. The m of Q_m means the "multiplicity" of multiple and/or mutually close roots of $P(x)=0$. Thus, approximate positions and "multiplicities" of the multiple and close roots are known, and the roots are determined by FIND-CloseRoots as follows.

Sub-procedure FIND-CloseRoots($P(x)$, u, m, $\hat{\delta}$)
Input: $P(x)$=regular polynomial having m close roots around $x=u$;
$\hat{\delta}$=minimum distance of close roots to be separated;
Output: m close roots $u_1, \ldots, u_m$ (with accuracy $\hat{\delta}$) around $x=u$, some of $\{u_1, \ldots, u_m\}$ may be counted as multiple roots;
Step 1: Putting $x=u+\delta$, expand $P(x+\delta)$ up to δ^m terms:
[D] $P^{(m)}(u)\delta^m/m!+\cdots+P^{(1)}(u)\delta/1!+P(u)=0$;
Step 2: Regularize [D] by putting $z=c\delta$, with c a large number, and let the regularized equation in z be $\tilde{P}(z)$;
Step 3: Perform the approximate square-free decomposition of $\tilde{P}(z)$:
$\tilde{P}(z)=\tilde{Q}_1(z)\tilde{Q}_2^2(z)\ldots\tilde{Q}_m^m(z)+O(\varepsilon(z))$, $\varepsilon=2\times10^{-4}$;
Solve $\tilde{Q}_i(z)=0$, $i=1, \ldots, m$, by low precision arithmetic;
Step 4: For each root $\tilde{z}$ of $\tilde{Q}_1(z)=0$, solve $P(x)=0$ accurately by Newton's method with initial approximation $x=u+\tilde{z}/c$;
Step 5: For each root $\tilde{z}$ of $\tilde{Q}_\mu(z)=0$, $\mu=2, \ldots, m$, do if $\hat{\delta}\geqq 1/100c$ then let $x=u+\tilde{z}/c$ be a μ-multiple root of $P(x)$, else call FIND-CloseRoots($P(x)$, $u+\tilde{z}/c$, μ, $\hat{\delta}$);
/* μ very-close roots around $x=u+\tilde{z}/c$ */
Step 6: Return the roots and multiplicities obtained. //

(Note 1). Equation [D] is not regular because $P^{(i)}(u)\approx 0$, $i=0, 1, \ldots, m-1$. Regularization in **Step 2** is equivalent to magnifying the area around $x=u$ locally by the factor c. By this, close roots of relative distance $\geqq 1/c$ are converted to well-distant roots, see an example below.
(Note 2). We need not solve $\tilde{Q}_i(z)=0$ accurately in **Step 3**. High precision arithmetic is necessary only in **Step 1**, **Step 4** and **Step 5**.

Example 5. Performance of FIND-CloseRoots on $P(x)=(x-0.99)^2(x-1.02)(x-2)$. Suppose we know that $x=1.0$ is an approximate root of $P(x)=0$ with "multiplicity" 3. Then, equation [D] becomes

$$-\delta^3-0.03\times10^{-2}\delta^2+2.98\times10^{-4}\delta+2.0\times10^{-6}=0.$$

This equation is not regular as we have noted above, and after the regularization $z=100\times\delta$ we have

$$z^3+0.03z^2-2.98z-2.0=0.$$

By the approximate square-free decomposition, we find that this equation has two close roots around $z=-1.0018$ and single root at $z\approx 1.9825$. We see that, although the regularized equation still has close roots corresponding to multiple roots of $P(x)=0$, the close roots of $P(x)=0$ are well separated. With the initial approximation $x=1.019825$, it is easy to calculate the single root at $x=1.02$ accurately. The two roots around $x=1.0-0.010018$ are investigated accurately by calling FIND-CloseRoots recursively.

The procedure FIND-Roots is effective for equations having close roots of any distances as well as multiple roots. As a result, it is not fast. On the other hand, many ill-conditioned equations to be solved actually have no close root but multiple roots which are not close to each other. Noting Property 3 in **4.**, we can solve such ill-conditioned equations much faster than FIND-Roots by modifying the procedure as follows.

Procedure FIND-Roots (modified version)
***Only the **Step 1** is modified as follows.
Step 1: Apply Approx-SQFR to $P(x)$ and find $Q_1, \ldots, Q_l$ such that
[C] $P(x)=Q_1(x)Q_2^2(x)\ldots Q_l^l(x)+O(\varepsilon(x))$, $\varepsilon=2\times10^{-4}$;
Check the accuracy $\hat{\varepsilon}$ of decomposition [C], and if $\hat{\varepsilon}\lesssim\hat{\delta}^2$ then solve each $Q_m(x)=0$, $m=1, \ldots, l$, as m-multiple roots and return the result; //

The accuracy check for the above decomposition [C] is made easily by checking the magnitude of $\mathrm{mmc}(P_{k+1})$: if $P_k\approx\mathrm{GCD}(P_1, P_2; \hat{\delta}^2)$ then $P_{k+1}=O(\hat{\delta}^2(x))$. If we want to determine the multiple roots accurately, with accuracy $\ll\hat{\delta}$, we may apply the conventional method for determining multiple roots, described in literature such as [4], by using the roots of $Q_m(x)$ as initial approximations.

7. Final Remarks

Computation of PRS's of various initial polynomials shows that much deeper study is necessary to explain the magnitude of coefficients of polynomials in PRS. In particular, knowing about the distance of neighboring roots in $\mathrm{GCD}'(P_1, P_2; \varepsilon)$ gives us a good information on PRS. Such a study is apparently fruitful because approximate GCD is useful not only for the approximate square-free decomposition but also for many kinds of calculations. Furthermore, analysis of PRS is directly applicable to Sturm sequence.

Another problem is the treatment of high degree polynomials. As we have noted in **3.** our theory of approximate GCD may not hold for high degree polynomials. Besides this point, high degree polynomials may cause another problem. If the degree of a regular polynomial $P(x)$ is high, say $\deg(P)=100$, then many roots are distributed within a circle of radius $O(1)$. Hence, the average distance between the neighboring roots is considerably small and the separation of close root factor seems to be not easy. As for high degree polynomials, more investigation, not only theoretical but also experimental, is apparently necessary.

The algorithms we have described in this paper are somewhat different from conventional numerical algorithms in that not only numeric computation but also algebraic computation are desirable. If different styles of computation are used combinedly in a single calculation, we call it *hybrid computation*. Although our algorithms can be implemented in a language for numeric computation, such as FORTRAN, they will be

most nicely performed on a hybrid algebraic-numeric system. The hybrid computation has been quite unfamiliar so far, but it will surely become quite important in future. For an example of hybrid computation and system, see [6, 9], for example.

The approximate GCD and square-free decomposition are direct generalizations of conventional algebraic algorithms. We think that many algebraic algorithms can be generalized to apply for approximate expressions, and we call such algorithms *approximate algebraic algorithms*. We want to emphasize the importance and possible fruitfulness of approximate algebraic algorithms as well as hybrid computation.

References

1. BAREISS, E. H. The numerical Solution of Polynomial Equations and The Resultant Procedure, in *Mathematical Methods for Digital Computers*, 2, John Wiley (1967).
2. GARSIDE, G. R., JARRAT, P. and MACK, C. A New Method for Solving Polynomial Equations, *Comput. J.*, **11** (1968), 87–89.
3. HITOTUMATU, S. A Method of Successive Approximation based on the Expansion of Second Order, *Mathematica Japonicae* (1966), 31–50.
4. IRI, M. *Numerical Computation*, Asakura Publishing Co., Tokyo (1981).
5. KALTOFEN, E. Factorization of Polynomials, in *Computer Algebra: Symbolic and Algebraic Computation* (Edited by B. Buchberger, *et al.*), Springer-Verlag (1982).
6. NODA, M. and IWASHITA, H. Solving Ordinary Differential Equations on a Hybrid Computation System, *J. IPS Japan* **28** (1987), 689–696.
7. POMENTALE, T. A Class of Iterative Method for Holomorphic Functions, *Numer. Math.*, **18** (1971), 193–203.
8. SASAKI, T. *Formula Manipulation*, *IPS Japan* (sold by Ohm Publishing Co.), Tokyo (1981).
9. SUZUKI, M., SASAKI, T., SATO, M. and FUKUI, Y. A Hybrid Algebraic-Numeric System ANS and its Preliminary Implementation, *Proc. of SYMSAC'87* (Lecture Notes Comp. Sci.), Springer-Verlag (1989).

(Received June 21, 1988; revised November 24, 1988)

Note added in Proof:

After submitting the paper, the authors found that Schöhage: "Quasi-GCD Computations", J. Complexity 1, pp. 118–137 (1985), discussed computing GCD of polynomials with real or complex coefficients up to a given accuracy. Schönhage introduced "pivoting" to the Euclidean algorithm to avoid numerical instability (which may happen in abnormal sequences, if conventional algorithm is used), and he discussed mostly the time-complexity of the algorithm. His approach to the approximate GCD is considerably different from ours.