COMPUTER SCIENCES
DEPARTMENT

University of Wisconsin-
Madison

A MODULAR GREATEST COMMON DIVISOR ALGORITHM FOR GAUSSIAN POLYNOMIALS

by

B. F. Caviness

and

Michael Rothstein

Computer Sciences Technical Report #245

March, 1975

A MODULAR GREATEST COMMON DIVISOR ALGORITHM FOR GAUSSIAN POLYNOMIALS*

by

B. F. Caviness

and

Michael Rothstein

Computer Sciences Department
University of Wisconsin
Madison, Wisconsin 53706

March, 1975

#295

*This research was supported in part by NSF Grant GJ-30125X and the University of Wisconsin Graduate School Research Committee.

Abstract.

In this paper the Brown-Collins modular greatest common divisor algorithm for polynomials in $Z[x_1,\ldots,x_v]$, where Z denotes the ring of rational integers, is generalized to apply to polynomials in $G[x_1,\ldots,x_v]$, where G denotes the ring of Gaussian integers, i.e., complex numbers of the form $a + ib$ where a,b are in Z.

Under certain simplifying assumptions, a function is found that dominates the maximum computing time of the new gcd algorithm.

Acknowledgement

The authors would like to express their thanks to Professor George E. Collins, Harvey I. Epstein, William Hibbard and Stuart Schaller for their aid in the development and testing of the work described herein.

1.1 Introduction

In this paper the Brown-Collins modular greatest common divisor algorithm for polynomials in $Z[x_1,\ldots,x_v]$, where Z denotes the ring of rational integers, is generalized to apply to polynomials in $G[x_1,\ldots,x_v]$ where G denotes the ring of Gaussian integers, i.e., complex numbers $a + ib$ where a,b are in Z. Polynomials in $G[x_1,\ldots,x_v]$ are called Gaussian polynomials.

Brown's description [BRO71] of the modular gcd algorithm for integral polynomials is excellent. In this paper we have used his definitions and notation to the extent possible. Furthermore we refer to his paper freely and the reader will undoubtedly find it necessary to have a copy of Brown's paper handy in order to understand our presentation. This method of presentation allows us to be brief but yet precise (hopefully). Our presentation also meshes nicely with Brown's because our modular algorithm for Gaussian polynomials is quite analogous to the algorithm for integral polynomials.

In section 2 we discuss some relevant properties of Gaussian integers. In section 3 we present the changes that are necessary to generalize the modular algorithm to Gaussian polynomials. This generalized algorithm has been programmed in the SAC-1 system [CAC75a].

Section 4 is devoted to bounding the number of unlucky primes and b-values that can occur in the Gaussian algorithm. These bounds assure us that the algorithm will terminate.

In section 5 the computing time for the Gaussian algorithm is analyzed. The computing time bound for the Gaussian algorithm is analogous to the bound obtained by Brown for polynomials with rational integer coefficients.

2. Gaussian Integers

In this section we briefly review some relevant properties of the Euclidean domain of Gaussian integers, henceforth denoted by G. For fuller information about G the reader can refer to a number theory source such as [HAW60].

The four units of G are ± 1 and $\pm i$. The <u>first-quadrant value</u> of a Gaussian integer is that associate $a + bi$ such that either $a = b = 0$ or $a > 0$ and $b \geq 0$. The first-quadrant value in each associate class will be taken to be unit normal.

Other than the prime $1 + i$, the unit normal primes of G fall into two classes. First if p is a prime in Z and $p \equiv 3 \pmod 4$ then p is prime in G. Second if p is a prime in Z and $p \equiv 1 \pmod 4$ then there exist unique positive a,b in Z such that $a^2 + b^2 = p$, $a + ib$ and $b + ia$ are primes in G. Except for associates there are no other Gaussian primes. The two classes of primes will be referred to as <u>real Gaussian primes</u> and <u>complex Gaussian primes</u> respectively.

If p is a real Gaussian prime, then the residue classes of G modulo the ideal generated by p are isomorphic to the finite field $GF(p^2)$, the finite field containing p^2 elements. If $a \pm bi$ is a complex Gaussian prime then the finite field of residue classes is isomorphic to $Z_p = GF(p)$ where $p = a^2 + b^2$.

The elements of $G/(p)$, p a real Gaussian prime, may be represented by the set $G_p = \{x+iy: x,y \text{ in } Z_p\}$. The homomorphism $\phi: G \to G_p$ is defined by $\phi(a+ib) = (a \bmod p) + i(b \bmod p)$. The operations of addition, subtraction, and

multiplication in G_p are carried out like ordinary complex arithmetic except that operations in Z_p are substituted for operations in Z. The inverse of $x + iy$ in G_p may be computed in a number of ways. The extended Euclidean algorithm or Fermat's theorem may be used, or the problem may be reduced to computing inverses in Z_p as follows: To compute $(x+iy)^{-1}$ first compute $\bar{Z} = (x+iy)(x-iy)$, a number of Z_p. Compute $\bar{Z}^{-1}$ in Z_p. Then in G_p, $(x+iy)^{-1} = \bar{Z}^{-1}(x-iy)$. The latter method is used in [CAC75a], cf. algorithm CGIREC.

A polynomial F in $G_p[x_1,\ldots,x_v]$ may be expressed in the form $F = D + iE$ where D and E are in $Z_p[x_1,\ldots,x_v]$. For $\alpha = a + ib$ in G the notation α_p denotes $(a \bmod p) + i(b \bmod p)$. For $F = D + iE$ in $G[x_1,\ldots,x_v]$ the notation F_p denotes the homorphic image of F in $G_p[x_1,\ldots,x_v]$ in which each coefficient α of F is replaced by α_p.

The elements of $G/(a+ib)$, $a + ib$ a complex Gaussian prime, may be represented by the elements of Z_p. The homomorphism $\phi:G \to G/(a+ib)$ is defined by $\phi(x+iy) = \phi_p(\phi_p(x)+\hat{i}\phi_p(y))$ where ϕ_p is the homomorphism $\phi_p:Z \to Z_p$ and $\hat{i} = \phi(i)$ an element of Z_p. $\hat{i}$ may be computed as follows. Find b^{-1} in Z_p. Then $\hat{i} = -b^{-1}a$ in Z_p. For example if $a + ib = 3 + 2i$, $\hat{i} = -5$ in Z_{13} if z_{13} is represented by $\{-6,-5,\ldots,0,1,\ldots,6\}$.

However with this representation of $G/(a+ib)$ the authors know no method to reconstruct the correct coefficients of the GCD via the Chinese Remainder Algorithm in step (11) of the Gaussian version of algorithm M (see section 3 and section 4.3

of [BRO71]). There does not seem to be any other straightforward representation of $G/(a+ib)$. For this reason the use of complex Gaussian primes will not be considered further.

For the modular GCD algorithm it will be necessary to have a list of real Gaussian primes. Such a list can be reasonably computed using a simple sieve technique to generate all real (rational) primes in a given interval and then discard those rational primes that are congruent to 1 modulo 4. See, for example, algorithm GIGNPR in [CAC75a].

3. A Modular GCD Algorithm for Gaussian Polynomials

In this section will be discussed the generalizations for Gaussian polynomials of Brown's algorithm M, [BRO71, pp. 489-491] along with associated changes that are necessary in algorithms P([BRO71], pp. 493-494), u (p. 495), and the Chinese Remainder Algorithm (p. 496).

G and G_p respectively denote the Gaussian integers and the Gaussian integers modulo a real Gaussian prime p. The inputs to algorithm M are two nonzero polynomials F_1' and F_2' in $G[x_1,\ldots,x_v]$. Algorithm M computes their GCD G' and their cofactors H_1' and H_2'. Define $c_1,c_2,c,F_1,F_2,G,H_1,H_2,f_1,f_2,g,h_1,h_2,\bar{g},\bar{G},\bar{F}_1,\bar{F}_2,\bar{H}_1$ and $\bar{H}_2$ as is done on page 489 of [BRO71]. However the capital letters now denote elements of $G[x_1,\ldots,x_v]$ and the small letters elements of G. Then as in the real case $G' = cG$, $H_1' = (c_1/c)H_1$, $H_2' = (c_2/c)H_2$, $\bar{F}_1 = \bar{G}\bar{H}_1$, $\bar{F}_2 = \bar{G}\bar{H}_2$, $pp(\bar{G}) = G$, $lc(\bar{G}) = \bar{g}$, $lc(\bar{H}_1) = gh_1 = f_1$ and $lc(\bar{H}_2) = gh_2 = f_2$. Also let $\underline{d} = \underline{\partial}(G)$.

At any given time in the execution of algorithm M there is a set of real Gaussian primes $p_1,\ldots,p_n$ that have been used and not discarded. Furthermore for $i = 1,\ldots,n$ the algorithm has computed $\tilde{F}_1^{(i)}$, $\tilde{F}_2^{(i)}$, $\tilde{G}^{(i)}$, $\tilde{H}_1^{(i)}$ and $\tilde{H}_2^{(i)}$ in $G_p[x_1,\ldots,x_v]$ such that Brown's equations on p. 489 (31) - (34) hold with the new interpretations.

The same argument that is used for the real case shows that $\underline{\partial}(\tilde{G}^{(i)}) \geq \underline{\partial}(G_{pi}) = \underline{\partial}(G) = \underline{d}$. As in the real case the algorithm keeps only those p_i for which $\underline{\partial}(\tilde{G}^{(i)}) = \underline{e}$ is minimal to date. Eventually $\underline{e} = \underline{d}$ and

$$\tilde{G}^{(i)} \equiv \bar{G} \mod p_i$$
$$\tilde{H}_1^{(i)} \equiv \bar{H}_1 \mod p_i$$
$$\tilde{H}_2^{(i)} \equiv \bar{H}_2 \mod p_i$$

for $i = 1,\ldots,n$.

As in the real case instead of preserving the quadruples $(p_i,\tilde{G}^{(i)},\tilde{H}_1^{(i)},\tilde{H}_2^{(i)})$ only the (rational) integer

$$q = \prod_{i=1}^{n} p_i$$

is maintained along with the unique polynomials G^*, H_1^*, and H_2^* with Gaussian integer coefficients whose real and imaginary parts have magnitude less than $q/2$ such that

$$G^* \equiv \tilde{G}^{(i)} \mod p_i$$
$$H_1^* \equiv \tilde{H}_1^{(i)} \mod p_i$$
$$H_2^* \equiv \tilde{H}_2^{(i)} \mod p_i$$

for $i = 1,\ldots,n$. As soon as $\underline{e} \equiv \underline{d}$ we have that

$$G^* \equiv \bar{G} \mod q$$
$$H_1^* \equiv \bar{H}_1 \mod q$$
$$H_2^* \equiv \bar{H}_2 \mod q.$$

when $q > \mu = 2 \max |\psi|$ where ψ ranges over the real and imaginary parts of the coefficients of $\bar{G}$, $\bar{H}_1$ and $\bar{H}_2$. It follows that $G^* = \bar{G}$, $H_1^* = \bar{H}_1$, and $H_2^* = \bar{H}_2$. The final results are obtained from the same set of relationships used in the real case. As in the real case, the (rational) integers modulo r can be represented as integers with magnitude less than $r/2$. In the implementation in [CAC75a]

Z_p is actually represented by the non-negative integers less than p.

The algorithm given on page 491 of [BRO71] remains valid for Gaussian polynomials F_1' and F_2' with the following changes:

1. The polynomial and coefficient operations are carried out with $G[x_1,\ldots,x_v]$ (or $G_p[x_1,\ldots,x_v]$) and G(or G_p) instead of $Z[x_1,\ldots,x_v]$ (or $Z_p[x_1,\ldots,x_v]$) and Z (or Z_p).

2. GCD's of Gaussian integers may be computed efficiently by using the generalized Lehmer algorithm as given in [CAC75a or CAC75b].

3. In step (5) set $\bar{\bar{\mu}} = 2 \max|\phi|$ where ϕ ranges over the real and imaginary parts of $\bar{g}$ and the real and imaginary parts of the coefficients of F_1 and F_2.

4. In step (6), p should be a new real Gaussian prime not dividing f_1 and f_2.

5. In step (8) algorithm P must be altered to apply to polynomials in $G_p[x_1,\ldots,x_v]$.

6. In step (11) the Chinese remainder algorithm must be altered to apply to polynomials in $G_p[x_1,\ldots,x_v]$. Note comments below.

7. In step (13) chose μ^* such that $\mu^*/2$ is an integer bound on the magnitudes of the real and imaginary parts of the coefficients of $G^*H_1^*$ and $G^*\ H_2^*$.

Since p is a real (rational) prime as well as a Gaussian prime it is not necessary to actually change the Chinese remainder algorithm in step (11). It is sufficient to apply twice the same algorithm used in the real case. First apply it to update the quadruple $(q,rp(G^*),rp(H_1^*),rp(H_2^*))$ to

include $(p,rp(\tilde{G}),rp(\tilde{H}_1),rp(\tilde{H}_2))$ where $rp(G^*)$ etc. denotes the real part of the polynomial G^*,i.e.,G^* can be written in the form $G_1^* + G_2^*$ where G_1^*, G_2^* are polynomials over Z. $rp(G^*) = G_1^*$. Next apply the real Chinese remainder algorithm to update the quadruple $(q,ip(\tilde{G}^*),ip(\tilde{H}_1^*),ip(\tilde{H}_2^*))$ (here we must use the same q that was used as input in the first application of the Chinese remainder algorithm and not the updated q) to include $(p,ip(G),ip(H_1),ip(H_2))$ where $ip(G^*)$, etc. denotes the imaginary part of G^*, i.e., G_2^*. Then the updated G^* = the updated $rp(G^*)$ + i times the updated $ip(G^*)$. The updated H_1^* and H_2^* are constructed analogously.

Algorithm P on page 494 of [BRO71] will work correctly for inputs F_1' and F_2' in $G_p[x_1,\ldots,x_v]$ with the following changes:

1. In the Gaussian case $\mathcal{F}$ denotes the domain $G_p[x_v]$.

2. Algorithm U should be the Euclidean algorithm for univariate polynomials over the field G_p.

3. In step (8), $\mathcal{F}_b[x_1,\ldots,x_{v-1}] = G_p[x_1,\ldots,x_{v-1}]$.

Note that we can still choose b from Z_p in step (6). This enables us to update the quadruple (q,G^*,H_1^*,H_2^*) to include $(x_v-b,\tilde{G},\tilde{H}_1,\tilde{H}_2)$ by applying the same interpolation form of the Chinese algorithm used for the real case to the respective real and imaginary parts of the polynomials G^*,H_1^*, and H_2^* in a manner that is completely analogous to that previously discussed for algorithm M.

4. Unlucky Primes and b-Values

The determination of upper bounds on the numbers of unlucky primes and unlucky b-values that can occur in the Gaussian case closely follows Brown's analysis for the real case (pp. 491-2 and 495). Brown's analysis depends on certain bounds for the coefficients of the subresultants of F_1 and F_2.

The bounds for the subresultant coefficients are obtained in section 3.5 (pp. 485-6) of [BRO71]. For the Gaussian case these bounds must be altered slightly. Let F_1 and F_2 be in $G[x]$ with coefficients, the real and imaginary parts of which are bounded in magnitude by C. Then by Hadamard's theorem the coefficients of T_i are bounded in magnitude by $(4m_iC^2)^{m_i}$. This is the generalization for the Gaussian case of Brown's bound (18).

If the coefficients of F_1 and F_2 are in $G[x_1,\ldots,x_v]$ with degree at most e_j in x_j the coefficients of T_i have degree at most $2m_ie_j$ in v_j as given by Brown in (20). If the polynomial coefficients of F_1 and F_2 have at most t terms each and have Gaussian integer coefficients, the real and imaginary parts of which are bounded in magnitude by C, then the real and imaginary parts of the Gaussian integer coefficients of the polynomial coefficients of T_i are bounded in magnitude by $(4m_iC^2t^2)^{m_i}$. This is the generalization of Brown's bound (21).

Brown's theorem 1 (p. 492) is still true if, in the statement, Z is changed to G and "integer" is changed to "Gaussian integer." The same proof, with the obvious modifications, holds for the new statement. Brown's theorem 2 becomes:

Theorem. Let u be the number of unlucky primes $p > \alpha$ in algorithm M applied to F_1, F_2 in $G[x_1,\ldots,x_v]$; where $\alpha \geq 2$ is a given integer. Let $C = \max |\phi|$ where ϕ ranges over the real and imaginary parts of the Gaussian integer coefficients of F_1 and F_2. Let

$m = (1/2) \max_{1\leq l\leq v} (\partial_i(F_1)+\partial_i(F_2))$ and $t = \max_{1\leq i\leq v} t_i$ where

t_i is the maximum number of terms in any polynomial coefficient of F_1 or F_2 viewed as univariate polynomials in x_i. Then

$$u < mv \log_\alpha(4mC^2t^2).$$

Similarly the total number of unlucky b-values for algorithm P applied to inputs F_1, F_2 in $G_p[x_1,\ldots,x_v]$ is at most $2me(v-1)$ where $m =$

$(1/2) \max_{1\leq i<v} (\partial_i(F_1)+\partial_i(F_2))$ and $e = \max(\partial_v(F_1),\partial_v(F_2))$.

5. Computing Time Analysis

The computing time analysis of the algorithms for Gaussian polynomials is quite similar to the analysis for real polynomials. We shall make the analogous assumptions that Brown made, namely: (1) In algorithms M and P no unlucky primes nor b-values occur. (2) In algorithm M it is assumed that the integer-length (defined below) of an exact quotient of Gaussian polynomials does not exceed the integer-length of the dividend. (3) In algorithm M the number of primes required is a linear function of ℓ where ℓ bounds the integer-lengths of the inputs.

The concepts of length and integer-length must be extended to members of G and $G[x_1,\ldots,x_v]$ respectively. Define the length of a Gaussian integer $a + ib$ to be the maximum of the length of a and the length of b. Let F be a nonzero member of $G[x_1,\ldots,x_v]$. The integer-length of F is defined to be the maximum of the integer-lengths of the Gaussian integer coefficients of F.

Now we present bounds for the maximum computing times of various operations in G and $G[x_1,\ldots,x_v]$. Let $T_G(op)$ denote the maximum computing time for the Gaussian integer operation op and let x_i denote a Gaussian integer of length ℓ_i. Then

$$T_G(x_3 \leftarrow x_1 \pm x_2) \le \ell_1 + \ell_2.$$

$$T_G(x_3 \leftarrow x_1 x_2) \le \ell_1 \ell_2 .$$

If $\ell_1 > \ell_2$, $T_G(x_3 \leftarrow x_1/x_2, x_4 \leftarrow x_1 - x_2 x_3) \le \ell_2(\ell_1 - \ell_2) \le \ell_2 \ell_3$.

If $\ell_1 > \ell_2$; $T_G(x_3 \leftarrow \gcd(x_1,x_2)) \le \ell_2(\ell_1 - \ell_3)$.

An algorithm for the division-remainder operation whose maximum computing time is dominated by $\ell_2\ell_3$ is given in [CAC75b]. The straightforward algorithm that multiplies dividend and divisor by the conjugate of the divisor and then performs two rational integer divisions has a maximum computing time that is not dominated by $\ell_2\ell_3$. The gcd operation can be carried out by the Euclidean algorithm and the above bound holds although as was pointed out earlier a more efficient algorithm can be found in [CAC75b].

Let T_{GP} denote the maximum computing time for the Gaussian polynomial operation op. As in the real case, a dimension vector for a nonzero F in $G[x_1,\ldots,x_v]$ is a pair (ℓ,d) where ℓ is the integer-length of F and $d \ge \partial_i(F)$, $1 \le i \le v$. Let F_i denote a Gaussian polynomial in v variables with dimension vector (ℓ_i,d_i) with $\ell_i > 0$. Then $T_{GP}(F_3 \leftarrow F_1 \pm F_2) \le \ell_1(d_1+1)^v + \ell_2(d_2+1)^v$.

$$T_{GP}(F_3 \leftarrow F_1 F_2) \le \ell_1\ell_2(d_1+1)^v(d_2+1)^v.$$

If $F_2 | F_1$ then $T_{GP}(F_3 \leftarrow F_1/F_2) \le \ell_2\ell_3(d_2+1)^v(d_3+1)^v$.

For polynomials in $G_p[x]$ we assume that all coefficient operations can be done in time dominated by 1. Hence for the Euclidean algorithm (algorithm U) applied to F_1, F_2 in $G_p[x]$ we have

$$T_{GP}(U) \le d_2(d_1 - d_3)$$

where $d_1 = \partial(F_1)$, $d_2 = \partial(F_2)$ and $d_3 = \partial(F_3)$ where $F_3 = \gcd(F_1,F_2)$. Here we assume that $d_1 \ge d_2 > 0$.

The analysis of the computing time for algorithm P applied to inputs F_1', F_2' in $G_p[x_1,\ldots,x_v]$ is completely analogous to the analysis for the real case since the only difference between the two versions of the algorithm is the

domain in which the coefficient arithmetic is carried out. But in both cases the time to do any of the required calculations in the coefficient domains is assumed to be dominated by 1.

However it should be noted that there seems to be a slight error in Brown's analysis of algorithm P. Formula (89) on page 501 while, strictly speaking, not incorrect, should nonehtless be replaced by

$$P(v,d) \le (2d+1)P(v-1,d) + c(d+1)^{v+1}$$

where c is a positive real constant. Solving this recurrence with the initial condition $P(1,d) \le cd^2$, yields $P(v,d) \lesssim (2d+1)^{v+1}$. Hence (91) on page 501 should be replaced by this dominance relationship.

Thus if $P(v,d)$ denotes the maximum computing time of algorithm P applied to F_1', F_2' in $G_p[x_1,\ldots,x_v]$ with $\partial_i(F_1')$, $\partial_i(F_2') \le d$, $1 \le i \le v$, then

$$P(v,d) \lesssim (2d+1)^{v+1}.$$

When applying algorithm M to nonzero polynomials F_1', F_2' in $G[x_1,\ldots,x_v]$, let (ℓ,d) bound the dimension vectors of F_1' and F_2'. Once again the computing time analysis for the Gaussian case is analogous to the real case since the bounds for $T_G(op)$ and $T_I(op)$ are completely analogous as are the bounds for $T_{GP}(op)$ and $T_P(op)$. Thus if $M(v,\ell,d)$ denotes the maximum computing time for algorithm M applied to F_1' and F_2' specified as above, then

$$M(v,\ell,d) \lesssim \ell^2(d+1)^v + \ell(2d+1)^{v+1} .$$

This bound reflects the correction to the bound for the computing time of algorithm P.

References

[BRO71] W. S. Brown, On Euclid's Algorithm and the computation of polynomial greatest common divisors, J. ACM 18 (Oct. 1971), 478-504.

[CAC75a] B. F. Caviness, G. E. Collins, H. I. Epstein, M. Rothstein, and S. C. Schaller, The SAC-1 Gaussian Integer and Gaussian Polynomial System, University of Wisconsin Computer Sciences Dept. Tech. Report (In preparation).

[CAC75b] B. F. Caviness and G. E. collins, Algorithms for Gaussian integer arithmetic (In preparation).

[HAW60] G. H. Hardy and E. M. Wright, An Introduction to the Theory of Numbers, Oxford University Press, London, 1960.