Approximate Greatest Common Divisors of Several Polynomials with Linearly Constrained Coefficients and Singular Polynomials*

Erich Kaltofen
Dept. of Mathematics
Massachusetts Institute of Technology
77 Massachusetts Ave.
Cambridge, Massachusetts 02139-4307, USA
kaltofen@math.mit.edu
http://www.kaltofen.us

Zhengfeng Yang and Lihong Zhi
Key Laboratory of Mathematics Mechanization
Academy of Mathematics and Systems Science
Beijing 100080, China
{zyang, lzhi}@mmrc.iss.ac.cn
http://www.mmrc.iss.ac.cn/~lzhi/

ABSTRACT

We consider the problem of computing minimal real or complex deformations to the coefficients in a list of relatively prime real or complex multivariate polynomials such that the deformed polynomials have a greatest common divisor (GCD) of at least a given degree k. In addition, we restrict the deformed coefficients by a given set of linear constraints, thus introducing the *linearly constrained approximate GCD* problem. We present an algorithm based on a version of the structured total least norm (STLN) method and demonstrate on a diverse set of benchmark polynomials that the algorithm in practice computes globally minimal approximations. As an application of the linearly constrained approximate GCD problem we present an STLN-based method that computes a real or complex polynomial the nearest real or complex polynomial that has a root of multiplicity at least k. We demonstrate that the algorithm in practice computes on the benchmark polynomials given in the literature the known globally optimal nearest singular polynomials. Our algorithms can handle, via randomized preconditioning, the difficult case when the nearest solution to a list of real input polynomials actually has non-real complex coefficients.

Categories and Subject Descriptors: I.2.1 [Computing Methodologies]: Symbolic and Algebraic Manipulation —Algorithms; G.1.2 [Mathematics of Computing]: Numerical Analysis—Approximation

General Terms: algorithms, experimentation

Keywords: multivariate polynomial gcd, approximate polynomial gcd, singular polynomial, approximate multiple root, linear constraint, symbolic/numeric hybrid method

*This research was supported in part by the National Science Foundation of the USA under Grants CCR-0305314 and CCF-0514585 (Kaltofen) and OISE-0456285 (Kaltofen, Yang and Zhi).
This research was partially supported by NKBRPC (2004CB318000) and the Chinese National Natural Science Foundation under Grant 10401035 (Yang and Zhi).
Kaltofen's permanent address: Dept. of Mathematics, North Carolina State University, Raleigh, North Carolina 27695-8205, USA, kaltofen@math.ncsu.edu.

1. INTRODUCTION

Symbolic-numeric algorithms accept as input polynomials and matrices with imprecise scalar coefficients, that is, the inputs do not yield a non-trivial result to the problem at hand. For instance, the polynomials have floating point coefficients such that if they were interpreted exactly, the polynomials would be relatively prime. One seeks minimal changes in the coefficients so that the perturbed inputs attain the desired property, for example, having a common factor of a given degree k. A classical example of such a problem formulation is that of computing a least squares solution to a linear system: one seeks a minimal change in the components of the right-side vector to make an inconsistent linear system have a solution.

Today, there is a significant body of results to solve those "approximate computer algebra" problems. This paper contributes to the approximate greatest common divisor (GCD) and the nearest singular polynomial problem. First, we formulate a generalization of the Sylvester matrix used for computing approximate GCDs of two polynomials [10, 34] that allows the computation of the approximate GCD of $s \geq 2$ multivariate polynomials over the real and complex numbers. The structured total least squares (STLS) and structured total least norm (STLN) [27] deformation techniques [18] are shown to produce accurate approximate GCDs. Second, we allow additional linear constraints on the coefficients of the deformed polynomials that yield the approximate GCD. Such constraints can enforce certain input coefficients to remain unchanged, preserving for example monicity or sparsity etc. They can also enforce linear relations (equations and inequalities) among the input coefficients. We present an STLN optimization formulation for the linearly constrained multivariate approximate GCD problem of several polynomials that has a reduced dimension compared with the unconstrained STLN formulation. Third, we apply the linearly constrained GCD problem to the problem of computing the nearest singular polynomial with a root of multiplicity at least $k \geq 2$ [37].

We show on a substantial body of benchmark data that our STLN algorithm is capable of computing the global optima for the approximate GCD and nearest singular polynomial problems. Although the STLN method, based on a least squares problem with penalty or least squares problems with linear constraints, is not guaranteed to converge to a global minimum, in our experiments it computes the solutions found by the global methods in [20, 13, 37, 36]. We note that the global minimum need not be unique [17, 19]. Our STLN implementation can handle inputs that have a large distance to the minimally deformed polynomials with a GCD or k-fold root. Such is especially the case when the input coefficients are far from satisfying the additional linear constraints. The approximate GCD and nearest singular polynomial problems have the exceptional property that minimal solution over the complex numbers to input polynomials with real coefficients can have non-real complex coefficients. Our STLN implementation computes the complex minimum via complex randomized pre-conditioning of the inputs. A new tool for obtaining these globally optimal results via STLN is the computation of initial vectors via the Lagrangian multiplier method.

Because of the page limit, we cannot give a complete review of previous work on the approximate GCD problem and must refer to the full version of this paper [19]. But we shall give a partial list of additional references. The question of how to deal with floating point coefficients in the Euclidean algorithm has been considered early on [8, 31, 30, 3]. Least squares and SVD-based total least squares methods were introduced in [6, 20]. When the approximate GCD is near the input polynomials, local minima are more accessible, and several algorithms have been proposed [7, 35, 10, 34]. The use of structure preserving total least squares algorithms is proposed in [15] in the setting of approximate polynomial factorization. In [17, 23, 4, 18] the approach was applied to the approximate GCD problem.

Finally, there is the issue of uniqueness of the global optimum. For certain structured total least squares problems the nearest solutions converge to an inconsistent system and a global optimum does not exist [13, Example 3]. The approximate GCD problem always has a globally nearest complex and a nearest real solution (in 2-norm) [17, Theorem 2.1]. Clearly, if real input polynomials possess a nearer complex optimum perturbation, the conjugate automatically becomes a second solution. In fact, multiple best approximations can exist and have been described for the nearest singular polynomial [37] and approximate factorization problems [15]. The examples in [37, Section 6] already exhibit polynomial inputs with real coefficients that have optimal approximations with complex coefficients. Even infinite families can occur for the nearest singular Toeplitz/ Hankel matrix problem [27, Section 4.3]. For the approximate GCD problem, the polynomials $f_1 = x + 1$ and $f_2 = x - 1$ have in 2-norm infinitely many nearest polynomials with a common root, namely $\tilde{f}_1 = \frac{-\sigma + i\tau + 1}{\sigma^2 + \tau^2 + 1}(x - \sigma - \boldsymbol{i}\tau)$ and $\tilde{f}_2 = \frac{\sigma - i\tau + 1}{\sigma^2 + \tau^2 + 1}(x - \sigma - \boldsymbol{i}\tau)$ with $\boldsymbol{i} = \sqrt{-1}$ and $\|\tilde{f}_1 - f_1\|_2^2 + \|\tilde{f}_2 - f_2\|_2^2 = 2$ for all $\sigma, \tau \in \mathbb{R}$.

2. TRANSFER TO LINEAR ALGEBRA

In order to apply structure preserving total least squares methods, we need a linear algebra formulation of our polynomial GCD problems. Fortunately, the approach in [17, 18, 24] can be generalized to s polynomials. We shall use total degree in our estimates, but as stated in [19] other degrees could be used.

Lemma 2.1 *Let $f_1, \ldots, f_s \in F[y_1, \ldots, y_r] \setminus \{0\}$, where F is an arbitrary field, and let $d_i = \text{tdeg}(f_i)$ and $k \leq d_i$ for all i with $1 \leq i \leq s$. Then $\text{tdeg}(\gcd(f_1, ..., f_s)) \geq k$ if and only if there exist polynomials $u_1, \ldots, u_s \in F[y_1, \ldots, y_r]$ with*

$$\left.\begin{array}{r} u_1 \neq 0, \quad \forall j, 2 \leq j \leq s\colon u_j f_1 + u_1 f_j = 0, \\ \forall i, 1 \leq i \leq s\colon \text{tdeg}(u_i) \leq d_i - k. \end{array}\right\} \qquad (1)$$

Proof. The property (1) expresses the fact that the GCD can be cancelled in the fraction f_j/f_1 in the unique factorization domain $F[y_1, \ldots, y_r]$. $\square$

The condition (1) leads to homogeneous linear system in the unknown coefficients of u_i. One may choose any of the f_i as the left-side polynomial. We choose the one of lowest degree, so that the resulting coefficient matrix has minimal dimensions. We shall investigate the structure of this matrix. As in [26, 16] we introduce the convolution matrix $\mathrm{C}^{[l]}(f)$, which for the coefficient vector $\vec{u}$ of a polynomial u of degree l produces the coefficient vector of $u \cdot f$ as $\mathrm{C}^{[l]}(f) \cdot \vec{u}$. For instance,

$$\overrightarrow{(a_2 y^2 + a_1 y + a_0) \cdot (b_2 y^2 + b_1 y + b_0)} =$$

$$\mathrm{C}^{[2]}(a_2 y^2 + a_1 y + a_0) \cdot \begin{bmatrix} b_2 \\ b_1 \\ b_0 \end{bmatrix} = \begin{bmatrix} a_2 & 0 & 0 \\ a_1 & a_2 & 0 \\ a_0 & a_1 & a_2 \\ 0 & a_0 & a_1 \\ 0 & 0 & a_0 \end{bmatrix} \cdot \begin{bmatrix} b_2 \\ b_1 \\ b_0 \end{bmatrix}.$$

In the univariate case, the matrix is of Toeplitz form. In the multivariate case, the dimensions of $\mathrm{C}^{[l]}(f)$ with $\text{tdeg}(f) = m$ are $\binom{l+m+r}{r} \times \binom{l+r}{r}$. Then the matrix $S_k(f_1, \ldots, f_s)$ is

$$\begin{bmatrix} \mathrm{C}^{[d_2-k]}(f_1) & \mathbf{0} & \cdots & \mathbf{0} & \mathrm{C}^{[d_1-k]}(f_2) \\ \mathbf{0} & \mathrm{C}^{[d_3-k]}(f_1) & & \mathbf{0} & \mathrm{C}^{[d_1-k]}(f_3) \\ \vdots & & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \mathrm{C}^{[d_s-k]}(f_1) & \mathrm{C}^{[d_1-k]}(f_s) \end{bmatrix} \qquad (2)$$

for the coefficient matrix of (1). The matrix is essentially a multi-polynomial generalized Sylvester matrix. We will exploit the following fact.

Lemma 2.2 *Let $f_1, \ldots, f_s$ and k be as in Lemma 2.1. Then $\text{tdeg}(\gcd(f_1, \ldots, f_s)) \geq k$ if and only if $S_k(f_1, \ldots, f_s)$ in (2) has rank deficiency at least one.*

Proof. The "only if" part follows directly from Lemma 2.1, since $u_1 \neq 0$ yields a non-trivial column dependency in $S_k(f_1, \ldots, f_s)$. The "if" part is a consequence of the fact that the first $s - 1$ block columns in (2) form a set of linearly independent column vectors, hence any vector in the right nullspace of $S_k(f_1, \ldots, f_s)$ must have a non-zero component corresponding to u_1 in (1), and Lemma 2.1 again applies. $\square$

Next we reduce the problem of testing a polynomial for having a factor of multiplicity k to a polynomial GCD problem. The approach is classical and incorporates Lemma 2.2.

Lemma 2.3 *Let $f(y) \in F[y]$ be a polynomial of degree n over a field F of characteristic 0, and let k be a multiplicity with $2 \le k \le n$. Furthermore, denote by $f^{[i]} = \mathrm{d}^i f/\mathrm{d}y^i$ the i-th derivative of f. Then the following conditions are equivalent.*

(i) *There exists a polynomial $h(y) \in F[y]$ with $\deg(h) \ge 1$ such that h^k is a factor of f.*

(ii) $\deg(\gcd(f^{[0]}, \dots, f^{[k-1]})) \ge 1$.

(iii) *The matrix $S_k^{sing}(f) =$*

$$\begin{bmatrix} s_{n-1,k-1} & \mathbf{0} & \cdots & \mathbf{0} & s_{n-k,0} \\ \mathbf{0} & s_{n-2,k-1} & & \mathbf{0} & s_{n-k,1} \\ \vdots & & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & s_{n-k+1,k-1} & s_{n-k,k-2} \end{bmatrix} \quad (3)$$

where $s_{i,j} = \mathrm{C}^{[i]}(f^{[j]})$, has rank deficiency at least one.

Proof. The lemma immediately follows from the fact that if $f = h_1 h_2^2 \cdots h_m^m$ with h_i squarefree and pairwise relatively prime, then $\gcd(f, \mathrm{d}f/\mathrm{d}y) = h_2 h_3^2 \cdots h_m^{m-1}$ (see, e.g, [11]). □

In some approximate multivariate polynomial factorization algorithms [10, 16] approximate squared factors need to be removed separately Lemma 2.3 has a multivariate corollary, which allows us then to compute an approximate squarefree factorization via our STLN-based approach.

Corollary 2.4 *Let $f(y_1, \dots, y_r) \in F[y_1, \dots, y_r]$ be a polynomial of degree n in y_1 over a field F of characteristic 0, and let k be a multiplicity with $2 \le k \le n$. Assume that f is primitive in y_1, i.e., f has no factor in $F[y_2, \dots, y_r]$. Then there exists a polynomial $h(y_1, \dots, y_r) \in F[y_1, \dots, y_r]$ with $\deg(h) \ge 1$ such that h^k is a factor of f if and only if the matrix $S_{y_1,k}^{sing}(f) =$*

$$\begin{bmatrix} \bar{s}_{n-1,k-1} & \mathbf{0} & \cdots & \mathbf{0} & \bar{s}_{n-k,0} \\ \mathbf{0} & \bar{s}_{n-2,k-1} & & \mathbf{0} & \bar{s}_{n-k,1} \\ \vdots & & \ddots & \vdots & \vdots \\ \mathbf{0} & \mathbf{0} & \cdots & \bar{s}_{n-k+1,k-1} & \bar{s}_{n-k,k-2} \end{bmatrix} \quad (4)$$

where $\bar{s}_{i,j} = \mathrm{C}^{[i]}(\frac{\partial^j}{\partial y_1^j} f)$, has rank deficiency at least one.

Proof. We apply Lemma 2.3 with the coefficient field $L = F(y_2, \dots, y_r)$. Since f is assumed primitive in y_1, all factors in $L[y_1]$ are by Gauss's lemma [11] associates of factors in $F[y_1, \dots, y_r]$ and have degree ≥ 1 in y_1. □

3. A SOLUTION BASED ON STLN

Structure-preserving total least norm algorithms [21, 27, 5, 22] compute for a structured matrix A and a structured vector b outside the range of A a minimally perturbed matrix $\tilde{A}$ of the same structure of A and a minimally perturbed vector $\tilde{b}$ of the same structure of b such that the linear system $\tilde{A}x = \tilde{b}$ is consistent. A special case is to compute a structured approximation of a matrix S that is rank deficient, in which case b can be chosen a column of S and A the submatrix formed by the remaining columns. Commonly considered structured matrices are symmetric, sparse, circulant, Toeplitz and Hankel matrices. In our case, the matrices will have the generalized Sylvester structures (2), (3) and (4) of Section 2. In this paper, we apply the STLN algorithm of [27]. We shall briefly describe the method.

Let $S(\zeta) = [A_1(\zeta) \mid b(\zeta) \mid A_2(\zeta)]$ and let $A(\zeta) = [A_1(\zeta) \mid A_2(\zeta)]$. Here the matrices S and A and the vector b are parametrized via the vector ζ. In the case of $S = S_k(f_1, \dots, f_s)$ in (2), the parameter vector ζ contains the coefficients of $f_1, \dots, f_s$, and in the case $S = S_k^{sing}(f)$ or $S = S_{y_1,k}^{sing}(f)$, the parameter vector is the coefficient vector of f. We wish to solve the two structure-preserving total least norm problems

$$\min_{\triangle\mathbf{c} \in \mathbb{R}^\nu} \|\triangle\mathbf{c}\| \text{ or } \min_{\triangle\mathbf{c} \in \mathbb{C}^\nu} \|\triangle\mathbf{c}\|$$
$$\text{with } A(\mathbf{c} + \triangle\mathbf{c})\mathbf{x} = b(\mathbf{c} + \triangle\mathbf{c}) \text{ for some vector } \mathbf{x}, \quad (5)$$

where $\mathbf{c}$ is fixed to the initial coefficient vector.[*] Here $\|\cdot\|$ can be 2-, 1- and ∞-norm, hence the notion "least norm" rather than "least squares," which is the 2-norm case. The choice of which column of S is moved to the right side depends on whether the nearest singular matrix contains that column in a linear column relation. For our problems, the condition is whether the corresponding co-factor polynomial u_j in Lemma 2.1 contains a corresponding non-zero term. As suggested in [18] we choose the column corresponding to the absolutely largest component in the first singular vector.

The STLN algorithm first initializes $\mathbf{x}$ as the unstructured least squares solution $A(\mathbf{c})\mathbf{x} \approx b(\mathbf{c})$ for the input parameters $\mathbf{c}$ and $\triangle\mathbf{c} = \mathbf{z} = 0$, and then refines both $\mathbf{x}$ and $\mathbf{z}$ simultaneously by iteration: the updated $\mathbf{x} + \triangle\mathbf{x}$ and $\mathbf{z} + \triangle\mathbf{z}$ satisfy (5), namely

$$\min \|\mathbf{z}{+}\triangle\mathbf{z}\| \text{ with } A(\mathbf{c}{+}\mathbf{z}{+}\triangle\mathbf{z})\ (\mathbf{x}{+}\triangle\mathbf{x}) = b(\mathbf{c}{+}\mathbf{z}{+}\triangle\mathbf{z}).$$

However, our optimization problems have many local suboptimal minima, and the standard initialization [27] is insufficient. We therefore extend the new initialization method in [21, Section 4.5.3] based on Lagrangian multipliers to our approximate GCD problems. Because the matrix-vector product $S(\zeta)\xi$ encodes sums of polynomial products, there is a Sylvester-like matrix $H(\xi)$ such that $H(\xi)\zeta = S(\zeta)\xi$. Suppose the first singular vector of the matrix $S(\mathbf{c})$ is $\mathbf{v}$; then we compute $\mathbf{z}$ as:

$$\mathbf{z} = -H(\mathbf{v})^{Tr}(H(\mathbf{v})H(\mathbf{v})^{Tr})^{-1}S(\mathbf{c})\mathbf{v}. \quad (6)$$

Suppose $b(\mathbf{c})$ is the t-th column corresponding to the absolutely largest component in $\mathbf{v}$; we compute the vector $\mathbf{x}$ by normalizing the vector $\mathbf{v}$ to make $\mathbf{v}[t] = -1$, i.e., we initialize $\mathbf{x}$ as

$$\mathbf{x} = \left[-\frac{\mathbf{v}[1]}{\mathbf{v}[t]}, \dots, -\frac{\mathbf{v}[t-1]}{\mathbf{v}[t]}, -\frac{\mathbf{v}[t+1]}{\mathbf{v}[t]}, \dots\right]^{Tr}. \quad (7)$$

Plugging in (6) we have $-S(\mathbf{z})\mathbf{v} = -H(\mathbf{v})\mathbf{z} = S(\mathbf{c})\mathbf{v}$, hence $S(\mathbf{c} + \mathbf{z})\mathbf{v} = 0$, or $A(\mathbf{c} + \mathbf{z})\mathbf{x} = b(\mathbf{c} + \mathbf{z})$ as required.

Since our parameterization is linear, we obtain as a first order approximation of the residue

$$\begin{aligned}
&\mathbf{r}(\mathbf{z} + \triangle\mathbf{z}, \mathbf{x} + \triangle\mathbf{x}) \\
&= b(\mathbf{c} + \mathbf{z} + \triangle\mathbf{z}) - A(\mathbf{c} + \mathbf{z} + \triangle\mathbf{z})\ (\mathbf{x} + \triangle\mathbf{x}) \\
&= b(\mathbf{c} + \mathbf{z}) + b(\triangle\mathbf{z}) - (A(\mathbf{c} + \mathbf{z}) + A(\triangle\mathbf{z}))(\mathbf{x} + \triangle\mathbf{x}) \\
&\approx b(\mathbf{c} + \mathbf{z}) - A(\mathbf{c} + \mathbf{z})\mathbf{x} + b(\triangle\mathbf{z}) - A(\mathbf{c} + \mathbf{z})\triangle\mathbf{x} - A(\triangle\mathbf{z})\mathbf{x} \\
&= \mathbf{r}(\mathbf{z}, \mathbf{x}) + b(\triangle\mathbf{z}) - A(\mathbf{c} + \mathbf{z})\triangle\mathbf{x} - A(\triangle\mathbf{z})\mathbf{x}.
\end{aligned}$$

Because the entries in $b(\zeta)$ are components of ζ we have a constant matrix P with $b(\zeta) = P\zeta$. Furthermore, changing the elements in $H(\xi)$ that correspond to the right-side $-b$

[*]In [27] the matrix $A(\mathbf{c})$ is denoted by A, and the "error" matrix $A(\triangle\mathbf{c})$ by E, and the perturbation vector $\triangle\mathbf{c}$ by η.

vector from -1 to 0, we obtain a second Sylvester-like matrix $Y(\xi)$ such that $Y(\xi)\zeta = A(\zeta)\xi$, in particular $A(\triangle\mathbf{z})\mathbf{x} = Y(\mathbf{x})\triangle\mathbf{z}$. Hence a first order approximation of the new residue can be expressed as

$$\begin{aligned}&\mathbf{r}(\mathbf{z}+\triangle\mathbf{z},\mathbf{x}+\triangle\mathbf{x})\\ &\qquad\approx \mathbf{r}(\mathbf{z},\mathbf{x}) + P\triangle\mathbf{z} - A(\mathbf{c}+\mathbf{z})\triangle\mathbf{x} - Y(\mathbf{x})\triangle\mathbf{z}.\end{aligned} \tag{8}$$

Using a penalty $w \gg 1$ on the residue [1] the minimization problem

$$\min_{\triangle\mathbf{z},\triangle\mathbf{x}} \left\| \begin{matrix} w\mathbf{r}(\mathbf{z}+\triangle\mathbf{z},\mathbf{x}+\triangle\mathbf{x}) \\ \mathbf{z}+\triangle\mathbf{z}\end{matrix} \right\|$$

has then the first order iterative update

$$\begin{aligned}\min_{\triangle\mathbf{x},\,\triangle\mathbf{z}} \Bigg\| &\begin{bmatrix} w(Y(\mathbf{x})-P) & wA(\mathbf{c}+\mathbf{z}) \\ \mathbf{I} & \mathbf{0}\end{bmatrix} \begin{bmatrix}\triangle\mathbf{z}\\ \triangle\mathbf{x}\end{bmatrix}\\ &\qquad + \begin{bmatrix}-w\mathbf{r}(\mathbf{z},\mathbf{x})\\ \mathbf{z}\end{bmatrix}\Bigg\|\end{aligned} \tag{9}$$

[27, Equation (2.9)]. The iterative update $\mathbf{x} = \mathbf{x} + \triangle\mathbf{x}$ and $\mathbf{z} = \mathbf{z} + \triangle\mathbf{z}$ is stopped when $\|\triangle\mathbf{x}\|$ and/or $\|\triangle\mathbf{z}\|$ becomes smaller than a given tolerance. The minimization problem (9) is for 2-norm and large penalty values, for instance $w = 10^8$, a stiff least squares problem which requires special care [2]. If the coefficient matrix in (9) is not of full rank, a solution via QR decomposition with column pivoting can yield good results (see also [19]). Alternatively, (9) can be formulated as least squares problem with linear equational constraints; see end of Section 3.1 and Section 5. For 1- and ∞-norm (on the combined vector of linear and imaginary parts), the problem can be solved by linear programming (see [13, Section 8.3]). However, if then the optimization problem is over the complex numbers (second case in (5)), real and complex parts need to be separated first. By writing $A(\zeta) = A(\zeta_R + \boldsymbol{i}\zeta_I) = A(\zeta_R) + \boldsymbol{i}A(\zeta_I)$, where ζ_R and ζ_I are the real and imaginary parts of ζ and $\boldsymbol{i} = \sqrt{-1}$, and splitting the residual and incremental vectors similarly, the updated residue (8) can be written as

$$\begin{aligned}&\mathbf{r}(\mathbf{z}+\triangle\mathbf{z},\mathbf{x}+\triangle\mathbf{x})\\ &\quad\approx \mathbf{r}_R(\mathbf{z},\mathbf{x}) + \boldsymbol{i}\mathbf{r}_I(\mathbf{z},\mathbf{x}) + P\triangle\mathbf{z}_R + \boldsymbol{i}P\triangle\mathbf{z}_I\\ &\qquad - (A(\mathbf{c}_R+\mathbf{z}_R) + \boldsymbol{i}A(\mathbf{c}_I+\mathbf{z}_I))(\triangle\mathbf{x}_R + \boldsymbol{i}\triangle\mathbf{x}_I)\\ &\qquad - (Y(\mathbf{x}_R) + \boldsymbol{i}Y(\mathbf{x}_I))(\triangle\mathbf{z}_R + \boldsymbol{i}\triangle\mathbf{z}_I)\\ &\quad= \mathbf{r}_R(\mathbf{z}+\triangle\mathbf{z},\mathbf{x}+\triangle\mathbf{x}) + \boldsymbol{i}\mathbf{r}_I(\mathbf{z}+\triangle\mathbf{z},\mathbf{x}+\triangle\mathbf{x}).\end{aligned}$$

The iterative update (9) can then be formulated as the real optimization problem

$$\begin{aligned}\min_{\triangle\mathbf{x}_R,\triangle\mathbf{x}_I,\triangle\mathbf{z}_R,\triangle\mathbf{z}_I} &\Big\| M \begin{bmatrix}\triangle\mathbf{z}_R & \triangle\mathbf{z}_I & \triangle\mathbf{x}_R & \triangle\mathbf{x}_I\end{bmatrix}^{Tr}\\ &+ \begin{bmatrix}-w\mathbf{r}_R(\mathbf{z},\mathbf{x}) & -w\mathbf{r}_I(\mathbf{z},\mathbf{x}) & \mathbf{z}_R & \mathbf{z}_I\end{bmatrix}^{Tr}\Big\|,\end{aligned} \tag{10}$$

where M is

$$\begin{bmatrix} w(Y(\mathbf{x}_R)-P) & -wY(\mathbf{x}_I) & wA(\mathbf{c}_R+\mathbf{z}_R) & -wA(\mathbf{c}_I+\mathbf{z}_I)\\ wY(\mathbf{x}_I) & w(Y(\mathbf{x}_R)-P) & wA(\mathbf{c}_I+\mathbf{z}_I) & wA(\mathbf{c}_R+\mathbf{z}_R)\\ \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0}\\ \mathbf{0} & \mathbf{I} & \mathbf{0} & \mathbf{0}\end{bmatrix} \tag{11}$$

[27, Equation (2.12)].

3.1 Solution With Linearly Constrained Input Coefficients

The matrix $S^{sing}(f)$ in (3) is a specialization $S(f_1,\ldots,f_k)$ in (2), where the input coefficients are restricted by linear constraints. The constraints are of the form $z_j = \lambda z_i$ where λ is an integer. The case is handled by making the appropriate substitution in the least squares problem (9), in particular since the minimization is on the vector of the remaining free z_i. Preserving monicity or sparsity of the input polynomials can be enforced by constraints of the type $z_j =$ constant.

In [13, Section 6.3] we have observed that the algorithms in [6, 20] allow incorporation of arbitrary linear constraints on the coefficients of the approximations of the input polynomials. We now show how to accomplish optimization under linear constraints for the STLN approach.

Let ζ be the symbolic coefficient vector (of dimension ν) of the structured problem, and let $\Gamma\zeta = \gamma$ be the system of fixed linear constraints on the goal coefficients. We do not assume that $\Gamma\mathbf{c} \approx \gamma$ for the initial input coefficient vector $\mathbf{c}$. By Gaussian elimination we construct for the linear system $\Gamma\zeta = \gamma - \Gamma\mathbf{c}$ a matrix C, a vector $\mathbf{d}$ and a sub-vector of the free parameters $[\zeta_{i_1},\ldots,\zeta_{i_\mu}]^{Tr}$ such that

$$\zeta = C\zeta^- + \mathbf{d}, \text{ where } \zeta^- = \begin{bmatrix}\zeta_{i_1}\\ \vdots\\ \zeta_{i_\mu}\end{bmatrix}; \text{ note that } \Gamma C = 0. \tag{12}$$

Following our earlier initialization approach, we compute

$$\mathbf{z} = \mathbf{d} - C\ (H(\mathbf{v})C)^{Tr}(H(\mathbf{v})C\ (H(\mathbf{v})C)^{Tr})^{-1}S(\mathbf{c}+\mathbf{d})\mathbf{v},$$

where $\mathbf{v}$ is the first singular vector of the matrix $S(\mathbf{c}+\mathbf{d})$. Again, we have $S(\mathbf{c}+\mathbf{z})\mathbf{v} = 0$. Normalizing $\mathbf{v}$ with respect to the absolutely largest component, we obtain the initialization of $\mathbf{x}$ (see (7)). The iterative update (9) can now be written as

$$\begin{aligned}\min_{\triangle\mathbf{x},\,\triangle\mathbf{z}^-} \Bigg\| &\begin{bmatrix} w(Y(\mathbf{x})-P)C & wA(\mathbf{c}+\mathbf{z}) \\ C & \mathbf{0}\end{bmatrix} \begin{bmatrix}\triangle\mathbf{z}^-\\ \triangle\mathbf{x}\end{bmatrix}\\ &\qquad + \begin{bmatrix}-w\mathbf{r}(\mathbf{z},\mathbf{x})\\ \mathbf{z}\end{bmatrix}\Bigg\|.\end{aligned} \tag{13}$$

The new coefficient values are $\mathbf{c} + \mathbf{z} + \triangle\mathbf{z} = \mathbf{c} + \mathbf{z} + C\triangle\mathbf{z}^-$ and satisfies $\Gamma(\mathbf{c}+\mathbf{z}+\triangle\mathbf{z}) = \gamma$ provided $\Gamma(\mathbf{c}+\mathbf{z}) = \gamma$. Our initialization $\Gamma(\mathbf{c}+\mathbf{z}) = \Gamma(\mathbf{c}+\mathbf{d}) = \Gamma\mathbf{c} + (\gamma - \Gamma\mathbf{c}) = \gamma$ guarantees that throughout the iteration. If the scalars are complex numbers, (13) can again be expressed as a real optimization problem:

$$\begin{aligned}\min_{\triangle\mathbf{x}_R,\triangle\mathbf{x}_I,\triangle\mathbf{z}_R^-,\triangle\mathbf{z}_I^-} &\Big\| M^- \begin{bmatrix}\triangle\mathbf{z}_R^- & \triangle\mathbf{z}_I^- & \triangle\mathbf{x}_R & \triangle\mathbf{x}_I\end{bmatrix}^{Tr}\\ &+ \begin{bmatrix}-w\mathbf{r}_R(\mathbf{z},\mathbf{x}) & -w\mathbf{r}_I(\mathbf{z},\mathbf{x}) & \mathbf{z}_R & \mathbf{z}_I\end{bmatrix}^{Tr}\Big\|,\end{aligned}$$

where

$$M^- = \begin{bmatrix} M_{1,1}^- & M_{1,2}^- & wA(\mathbf{c}_R+\mathbf{z}_R) & -wA(\mathbf{c}_I+\mathbf{z}_I)\\ M_{2,1}^- & M_{2,2}^- & wA(\mathbf{c}_I+\mathbf{z}_I) & wA(\mathbf{c}_R+\mathbf{z}_R)\\ C_R & -C_I & \mathbf{0} & \mathbf{0}\\ C_I & C_R & \mathbf{0} & \mathbf{0}\end{bmatrix}$$

with

$$\left.\begin{aligned} M_{1,1}^- &= w(Y(\mathbf{x}_R)C_R - Y(\mathbf{x}_I)C_I - PC_R),\\ M_{1,2}^- &= -w(Y(\mathbf{x}_I)C_R + Y(\mathbf{x}_R)C_I - PC_I),\\ M_{2,1}^- &= w(Y(\mathbf{x}_I)C_R + Y(\mathbf{x}_R)C_I - PC_I),\\ M_{2,2}^- &= w(Y(\mathbf{x}_R)C_R - Y(\mathbf{x}_I)C_I - PC_R),\end{aligned}\right\} \tag{14}$$

and the linear constraint is split into real and imaginary parts as $\zeta_R + \boldsymbol{i}\zeta_I = (C_R + \boldsymbol{i}C_I)(\zeta_R^- + \boldsymbol{i}\zeta_I^-) + \mathbf{d}_R + \boldsymbol{i}\mathbf{d}_I$.

Although $S_k^{sing}(f)$ of (3) is a linearly constrained version of $S_k(f_1, \ldots, f_k)$ of (2), the actual optimization problems are not the same. For the nearest k-fold singular polynomial one optimizes $\|\triangle f\|$, while the GCD problem with the corresponding constraints on the coefficients optimizes $\sum \|\mathrm{d}^i \triangle f / \mathrm{d}y^i\|$, which has a different minimum. Already in [27] a (diagonal) weight matrix D is allowed in the minimization problem (5), namely

$$\min_{\triangle\mathbf{c}\in\mathbb{R}^\nu} \|D\triangle\mathbf{c}\| \text{ or } \min_{\triangle\mathbf{c}\in\mathbb{C}^\nu} \|D\triangle\mathbf{c}\|$$
$$\text{with } A(\mathbf{c}+\triangle\mathbf{c})\mathbf{x} = b(\mathbf{c}+\triangle\mathbf{c}) \text{ for some vector } \mathbf{x}.$$

Then (13) becomes

$$\min_{\triangle\mathbf{x},\,\triangle\mathbf{z}^-} \left\| \begin{bmatrix} w(Y(\mathbf{x})-P)C & wA(\mathbf{c}+\mathbf{z}) \\ DC & \mathbf{0} \end{bmatrix} \begin{bmatrix} \triangle\mathbf{z}^- \\ \triangle\mathbf{x} \end{bmatrix} + \begin{bmatrix} -w\mathbf{r}(\mathbf{z},\mathbf{x}) \\ D\mathbf{z} \end{bmatrix} \right\|.$$

Not assuming that the weights are positive real numbers, the matrix D can now be chosen as to optimize the norm of the perturbation in the first polynomial in the GCD problem, thus allowing the nearest singular polynomial question to be formulated as a GCD problem with linearly constrained coefficients. For general $D = D_R + \boldsymbol{i} D_I$, the real matrix given above now becomes the matrix M^-:

$$\begin{bmatrix} M^-_{1,1} & M^-_{1,2} & wA(\mathbf{c}_R+\mathbf{z}_R) & -wA(\mathbf{c}_I+\mathbf{z}_I) \\ M^-_{2,1} & M^-_{2,2} & wA(\mathbf{c}_I+\mathbf{z}_I) & wA(\mathbf{c}_R+\mathbf{z}_R) \\ D_RC_R-D_IC_I & -D_RC_I-D_IC_R & \mathbf{0} & \mathbf{0} \\ D_RC_I+D_IC_R & D_RC_R-D_IC_I & \mathbf{0} & \mathbf{0} \end{bmatrix} \quad (15)$$

where $M^-_{1,1}, \ldots, M^-_{2,2}$ are as in (14).

As an entirely alternative approach, one can add the constraints $\Gamma(\mathbf{z}+\triangle\mathbf{z}) = \gamma$ directly to (9) and iterate on

$$\min_{\triangle\mathbf{x},\,\triangle\mathbf{z}} \left\| \begin{bmatrix} w(Y(\mathbf{x})-P) & wA(\mathbf{c}+\mathbf{z}) \\ w\Gamma & \mathbf{0} \\ D & \mathbf{0} \end{bmatrix} \begin{bmatrix} \triangle\mathbf{z} \\ \triangle\mathbf{x} \end{bmatrix} + \begin{bmatrix} -w\mathbf{r}(\mathbf{z},\mathbf{x}) \\ w(\Gamma(\mathbf{c}+\mathbf{z})-\gamma) \\ D\mathbf{z} \end{bmatrix} \right\|. \quad (16)$$

However, that least squares problem has a larger dimension than (13). Nonetheless, a variant of (16) can be used together with the linear programming techniques mentioned above (cf. [13, Section 8.3]) to minimize for 1- and ∞-norm with linear component-wise *inequality* constraints $\Gamma(\mathbf{c}+\mathbf{z}+\triangle\mathbf{z}) \geq \gamma$. For instance, for ∞-norm and for real inputs and outputs one minimizes $\min_{\triangle\mathbf{z}} \|D(\mathbf{z}+\triangle\mathbf{z})\|_\infty$ under the linear constraints $(Y(\mathbf{x})-P)\triangle\mathbf{z} + A(\mathbf{c}+\mathbf{z})\triangle\mathbf{x} = \mathbf{r}(\mathbf{z},\mathbf{x})$ and $\Gamma\triangle\mathbf{z} \geq \gamma - \Gamma(\mathbf{c}+\mathbf{z})$. The corresponding linear program in the unknowns δ and the entries of $\triangle\mathbf{z}$ and $\triangle\mathbf{x}$ is

$$\begin{array}{lrcl} \text{minimize:} & \delta & & \\ \text{subject to:} & \delta - D_i\triangle\mathbf{z} & \geq & D_i\mathbf{z}, \\ & \delta + D_i\triangle\mathbf{z} & \geq & -D_i\mathbf{z}, \\ & (Y(\mathbf{x})-P)\triangle\mathbf{z} + A(\mathbf{c}+\mathbf{z})\triangle\mathbf{x} & = & \mathbf{r}(\mathbf{z},\mathbf{x}), \\ & \Gamma\triangle\mathbf{z} & \geq & \gamma - \Gamma(\mathbf{c}+\mathbf{z}), \end{array}$$

where D_i is the i-th row of D, $i = 1, 2, \ldots$.

3.2 Computing Complex Optima for Real Inputs

In [17] we observe that the STLN iterations given above can only produce the real local optimum when executed on inputs with real coefficients. However, the approximate GCD and nearest singular polynomial problems can in those cases have a nearer approximation when permitting complex coefficients in the solution [17, 19]. Here we employ the change in the initialization of [17] in order to escape from the entirely real arithmetic process. Suppose the first singular vector of the matrix $S(\mathbf{c}+\boldsymbol{i}\triangle\mathbf{c}_{rand})$ is $\mathbf{v}$, where $\triangle\mathbf{c}_{rand}$ is a random real vector of small noise. Then we compute $\mathbf{z}$ as:

$$\mathbf{z} = -H(\mathbf{v})^{Tr}(H(\mathbf{v})H(\mathbf{v})^{Tr})^{-1}S(\mathbf{c}+\boldsymbol{i}\triangle\mathbf{c}_{rand})\mathbf{v}.$$

We compute the $\mathbf{x}$ by normalizing the vector $\mathbf{v}$ with respect to the absolutely largest component so that we have

$$A(\mathbf{c}+\boldsymbol{i}\triangle\mathbf{c}_{rand}+\mathbf{z})\mathbf{x} = b(\mathbf{c}+\boldsymbol{i}\triangle\mathbf{c}_{rand}+\mathbf{z}). \quad (17)$$

However, we do not add $\boldsymbol{i}\triangle\mathbf{c}_{rand}$ to the real optimization problem (10).

In the linearly constrained case, we initialize

$$\mathbf{z} = \mathbf{d} - C\,(H(\mathbf{v})C)^{Tr}(H(\mathbf{v})C\,(H(\mathbf{v})C)^{Tr})^{-1} \times S(\mathbf{c}+\mathbf{d}+\boldsymbol{i}\triangle\mathbf{c}_{rand})\mathbf{v},$$

where $\mathbf{v}$ is the first singular vector of the matrix $S(\mathbf{c}+\mathbf{d}+\boldsymbol{i}\triangle\mathbf{c}_{rand})$. We initialize the vector $\mathbf{x}$ by normalizing the vector $\mathbf{v}$ with respect to the absolutely largest component in $\mathbf{v}$ (see (7)). Then we have

$$A(\mathbf{c}+\boldsymbol{i}\,C\triangle\mathbf{z}^-_{rand}+\mathbf{z})\mathbf{x} = b(\mathbf{c}+\boldsymbol{i}\,C\triangle\mathbf{z}^-_{rand}+\mathbf{z}), \quad (18)$$

where $\triangle\mathbf{z}^-_{rand}$ is a random real vector of small free parameter values. As is exhibited in Section 4, the preconditioners (17) and (18) take the STLN iteration away from the real local optimum for our problems. Unfortunately, $\mathbf{x}, \mathbf{z}$ may not be near the complex optimum and the algorithm then searches through many iterations to find a suitable place from which convergence begins. Nonetheless, the method can compute global complex minima and appears significantly faster than the universal polynomial-time algorithms in [20, 37, 36]. Furthermore, for our approximate problems we currently do not know of any viable alternative in this difficult case.

4. ALGORITHMIC DETAILS AND EXPERIMENTS

We have implemented algorithms for real and complex inputs and real and complex optima for

– computing the approximate GCD of several multivariate polynomials

– computing the nearest univariate polynomial with a k-fold root

– computing the approximate GCD of several univariate or multivariate polynomials with linearly constrained coefficients

We first present a worked example of the latter procedure, which constitutes a generalization of the nearest singular polynomial problem to several input polynomials.

Example 4.1 Consider the polynomials

$$f = y(y-\boldsymbol{i})^2 + 0.01 \quad \text{and} \quad g = (y+\boldsymbol{i})(y-\boldsymbol{i})^2 - 0.01\boldsymbol{i}. \quad (19)$$

We seek to compute the nearest pair of complex polynomials $\tilde{f}$ and $\tilde{g}$ that have a *common* 2-fold root (cf. [28]). The linearly constrained GCD problem is for the four polynomials f, g, $\mathrm{d}f/\mathrm{d}y$, $\mathrm{d}g/\mathrm{d}y$ (cf. proof of Lemma 2.3) and restricting the distance measure to $\|f-\tilde{f}\|_2^2 + \|g-\tilde{g}\|_2^2$. Writing the coefficients as a single 14-dimensional vector ζ, we obtain the parameterized constraint matrix

$$C = \begin{bmatrix} 1&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&1&0&0&0&0\\ 0&0&0&0&1&0&0&0\\ 0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&1&0\\ 0&0&0&0&0&0&0&1\\ 3&0&0&0&0&0&0&0\\ 0&2&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0\\ 0&0&0&0&3&0&0&0\\ 0&0&0&0&0&2&0&0\\ 0&0&0&0&0&0&1&0 \end{bmatrix}, \ \zeta^- = \begin{bmatrix}\zeta_1\\ \vdots\\ \zeta_8\end{bmatrix} \text{ and } \mathbf{d} = \mathbf{0},$$

(see Section 3.1). The weight matrix is $D = \mathrm{diag}(I_8, \mathbf{0}_6)$. We use the real formulation (15) of the STLN algorithm with constrained coefficients and a penalty weight of $w = 10^8$. After 6 iterations we stop the algorithm at $\triangle\mathbf{x} = 0.974\cdot 10^{-6} < 10^{-6}$. The needed deformation for the input coefficients is computed as $C\mathbf{z}^-$ yielding the deformed inputs

$$\begin{aligned}\tilde{f} &= 0.0039 - 0.0018\,\boldsymbol{i} + 0.0013\,y^2 - 2.0006\,\boldsymbol{i}y^2\\ &\quad +0.0037\,\boldsymbol{i}y - 1.0006\,y + 0.0011\,\boldsymbol{i}y^3 + 0.9982\,y^3,\\ \tilde{g} &= 0.0018 - 1.0066\,\boldsymbol{i} + 0.0006\,y^2 - 1.0022\,\boldsymbol{i}y^2\\ &\quad -0.0006\,\boldsymbol{i}y + 1.0028\,y - 0.0018\,\boldsymbol{i}y^3 + 0.9984\,y^3.\end{aligned}$$

The co-factors are derived from $\mathbf{x}$ as

$$\begin{aligned}\tilde{u}_1 &= 0.0018 + 0.0039\,\boldsymbol{i} + 0.9982\,y^2 + 0.0011\boldsymbol{i}y^2\\ &\quad -0.0013\,y - 0.9994\,\boldsymbol{i}y,\\ \tilde{u}_2 &= 1.0036 + 0.0003\,\boldsymbol{i} + 0.9984\,y^2 - 0.0018\,\boldsymbol{i}y^2\\ &\quad +0.0008999\,y - 0.00081\,\boldsymbol{i}y.\end{aligned}$$

Performing an approximate polynomial division [35, 16] we obtain as the double factor $\tilde{h} = 0.001504 - 1.003015\,\boldsymbol{i} + 1.000000\,y$. Finally, the backward error is

$$\|f - \tilde{h}^2\,\tilde{u}_1\|_2^2 + \|g - \tilde{h}^2\,\tilde{u}_2\|_2^2 = 0.947\cdot 10^{-4}.$$

For comparison the unconstrained approximate GCD of f, g, $\mathrm{d}f/\mathrm{d}y$, $\mathrm{d}g/\mathrm{d}y$ is $\bar{h} = -6.504887\cdot 10^{-8} - 1.000011\,\boldsymbol{i} + 1.00000\,y$, which was found by our STLN implementation after 2 iterations. Although $\bar{h}$ seems a more accurate result w.r.t. the construction (19), the backward error of the combined approximate polynomial divisions of f by $\bar{h}^2$ and g by $\bar{h}^2$ is 0.000140.

Example 4.2 Consider the polynomials

$$f = 1000\,y^{10} + y^3 - 1 \quad\text{and}\quad g = y^2 - \frac{1}{100}.$$

We seek to compute the nearest pair of polynomials $\tilde{f}$ and $\tilde{g}$ that have a non-trivial GCD. We add random small complex noise to the initialization as in Section 3.2. After about ten iterations in the average, the algorithm converges to the following local minima:

$$0.0421579, 0.0463113, 0.0474087, 0.0493292, \ldots$$

for different initializations. Among solutions, the polynomials

$$\begin{aligned}\tilde{f} &= 1000.0\,y^{10} + 0.0000147908\,y^9 + 0.0000297998\,y^8\\ &\quad +0.0000604355\,y^7 + 0.000122287\,y^6 + 0.000247491\,y^5\\ &\quad +0.000500837\,y^4 + 1.00101\,y^3 + 0.00205103\,y^2\\ &\quad +0.00415059\,y - 0.991601,\\ \tilde{g} &= 0.956139\,y^2 - 0.0887590\,y - 0.189618,\end{aligned}$$

have a common divisor $y - 0.4941547$, and the backward error is

$$\|f-\tilde{f}\|_2^2 + \|g-\tilde{g}\|_2^2 = 0.0421579.$$

It is the non-monic global minimum found by the global methods in [20, 13].

The polynomials f and g also provide an example for a small *structured* condition number of a Sylvester matrix with a large unstructured condition number. Let S be the Sylvester matrix of f and g and $\hat{S}$ be the nearest singular Sylvester matrix to S measured in Frobenius matrix norm, which we shall denote by $\|\cdot\|_F$. First assume that $\hat{S}$ corresponds to two complex polynomials $\hat{f}$ and $\hat{g}$ of degrees 10 and 2, respectively. By virtue of the singularity of $\hat{S}$, $\hat{f}$ and $\hat{g}$ have a common root. Therefore

$$\begin{aligned}\|S-\hat{S}\|_F^2 &= 2\|f-\hat{f}\|_2^2 + 10\|g-\hat{g}\|_2^2\\ &\geq 2\|f-\hat{f}\|_2^2 + 2\|g-\hat{g}\|_2^2\\ &\geq 2\|f-\tilde{f}\|_2^2 + 2\|g-\tilde{g}\|_2^2 \geq 0.084315.\end{aligned}$$

Second, one can easily rule out that $\hat{S}$ corresponds to polynomials with zero coefficients in the degree 10 and 2 terms. The above lower bound can be scaled to other matrix norms by well-known inequalities [32, Section 4.2, Example 13]. The nearest unstructured singular matrix $\hat{A}$ has $\|S-\hat{A}\|_F^2$ equal the square of the smallest singular value of S [9; 32, Theorem 6.7], which is 0.000000098975. For other matrix norms, similar explicit values can be computed [14, p. 775]. Such a difference between structured and unstructured distances to singularity is impossible for Toeplitz matrices [29]. □

In the following three tables, we show the performance of our algorithms for computing the approximate GCDs and nearest singular polynomials on Pentium 4 at 2.0 Ghz for *Digits* = 14 in Maple 10 under Windows.

In Table 1 we show the performance of our algorithm for computing the *monic* nearest singular polynomials. Here m denote the degrees of polynomials; k is the multiplicity of roots; whereas *it. (STLN)* denotes the number of iterations in our C-multiple-root algorithm; *error (STLN)* and *error (ZNKW)* are the minimal perturbation $\|\tilde{f}-f\|_2^2$ computed by the algorithm in [36] and our C-multiple-root procedure, respectively, such that $\tilde{f}$ has a k-fold root.

Example 1 and 2 are from [37]. Example 1 is a real polynomial. Due to the geometry of the zeros of the polynomial, there are four nearest singular complex polynomials with a k-fold root, for $k = 2, 3$. Our algorithm can compute all four globally minimal solutions through randomization. Example 2 is a polynomial with complex coefficients. Examples 3 to 8 are from [36]. Note that the minima $\mathcal{N}_m$ in Tables 2 and 5 in [37] and the minimum $\mathcal{N}_m$ in Example 5 in [36] are incorrectly stated. Here we give the corrected minima computed with the original Maple procedures of [37, 36]. Except Example 7, all other examples have real coefficients.

Ex.	*m*	*k*	*it.* *(STLN)*	*error* *(ZNKW)*	*error* *(STLN)*
1	4	2	12	.1763296120	.1763296118
		3	34	.6261127476	.6261127498
2	4	2	4	.1552760123e–12	.1552723415e–12
		3	11	.8834609009e–9	.9814886696e–9
		4	4	.2021848972e–4	.1958553174e–4
3	4	2	4	.1645037985e–10	.16450617515e–10
		3	4	.4144531274e–6	.4144531274e–6
		4	12	.1049993144	.1049993152
4	5	2	1	.2460987981e–8	.2461467456e–8
	5	3	20	.3681785214	.3681784856
5	6	2	2	.3231668276e–5	.3231668277e–5
6	6	2	3	.3009788845e–11	.3009789157e–11
		3	3	.7453849284e–6	.7453849284e–6
		4	24	.4449023547	.4449023547
7	5	2	8	.8565349347	.8565327605
8	21	2	2	.190477e–8	.1893347157e–8
		3	6	.963776e–4	.9637591989e–4

Table 1: Algorithm performance on benchmarks (univariate singular polynomial case)

Ex.	m_i	*k*	*e*	*it.*	*error* *(Zeng)*	*error* *(GKMYZ)*	*error* *(STLN)*
1	7,7	4	3	2	2.44360e–4	2.59476e–4	6.50358e–5
2	7,7	4	5	1	2.44404e–8	2.59194e–8	6.50357e–9
3	7,7	4	7	1	2.44405e–12	2.59191e–12	6.50357e–13
4	7,7	4	9	1	2.44396e–16	2.59187e–16	6.50361e–17
5	6,6	3	2	3	2.26617	1.49524	4.80154e–1
6	10,10	5	4	2		2.74672e–3	1.84914e–3
7	8,8	4	5	2	7.09371e–5	2.38059e–5	2.01393e–5
8	40,40	30	5	2	1.39858e–3	4.83931e–4	4.39489e–4
9	10,9,8	5	3	2			6.21772e–2
10	8,7,8,6	4	5	2			4.04458e–6

Table 2: Algorithm performance on benchmarks (multivariate polynomials case)

In fact, we have been able to match the ZNKW backward error with both formulations of our STLN-method, one explicitly based on the matrix S_k^{sing} of (3) on page and one based on approximate GCDs with linear constraints.

In Table 2 we show the performance of our algorithm for computing the approximate GCDs of multivariate polynomials. Here m_i denote the total degrees of polynomials; k is the total degree of the approximate GCD; whereas *error (Zeng)*, *error (GKMYZ)* and *error (STLN)* are the minimal perturbation $\sum_i \|\tilde{f}_i - f_i\|_2^2$ computed by the algorithms in [34, 10] and our new algorithm, respectively. As in Table 1, *it.* is again the number of iterations performed by our STLN algorithm. Examples 1 to 4 in Table 2 correspond to the Example 4 in [34] for different perturbations. We note that following our results, Zhongang Zeng has shown us improvements to his code that yield backward errors comparable to ours for those examples. Examples 5 and 7–10 were constructed by choosing polynomials with random integer

	$j=3$	$j=4$	$j=5$	$j=6$	$j=7$	$j=8$
it.	1	1	1	3	4	5
err.	2.48e–12	3.33e–10	3.39e–8	3.76e–6	2.53e–4	2.93e–2

Table 3: Algorithm performance on two 4-variate polynomials

coefficients in the range $-25 \le c \le 25$ and having a nontrival GCD, then adding perturbations to each polynomial; for perturbation we randomly choose a polynomial that has the same degree as the unperturbed polynomial and coefficients in $[-10^e, 10^e]$; finally, we scale the perturbation so that the relative error is 10^{-e}. Example 6 is an example with complex coefficients, where the real and imaginary parts of the coefficients were integers in the range $-25 \le c \le 25$. Perturbations as stated before were added to both real and imaginary parts of polynomials.

In Table 3 we show the performance of our algorithm for computing the approximate linear common factor with multiplicity 2 of two degree 4 and 4-variate polynomial pair f_1, f_2 defined in Example 5 in [34]. The case is handled by computing GCD of f_1, ∂f_1, f_2, ∂f_2 and adding linear constraints which are generated by comparing the coefficients of $f_1, \partial f_1$ and $f_2, \partial f_2$, respectively. Here ∂f_i is the partial derivative of f_i w.r.t. one variable. The perturbation is of order 10^{j-10}.

Our 36 test cases and Maple implementation is available at `http://www.mmrc.iss.ac.cn/~lzhi/Research/hybrid/manystln/` and `http://www.math.ncsu.edu/~kaltofen/software/manystln/`.

5. CONCLUDING REMARKS

We have shown that the structured total least norm approach to approximate computer algebra problems can be applied when the coefficients of the deformed polynomials are also to satisfy a set of linear constraints.

When the input polynomials are within a relative error of no more than 10^{-2}, we have demonstrated that our structured total least norm (STLN) based algorithms converge quickly to the minimal approximate solutions, needing no more than about 10 iterations. However, for both the approximate GCD and nearest singular polynomial problems special cases arise where the nearest solution to a list of polynomials with real inputs are complex polynomials. In addition, our introduction of linear constraints on the input coefficients can move the nearest solution satisfying those constraints to a substantial distance from the input. These are difficult cases for all iterative algorithms that we know. Nonetheless, our STLN-based algorithms can compute optimal solutions. We have presented a new approach to choose better starting points, but in some difficult cases it still takes significantly many iterations of search before reaching a point of convergence.

In our experiments, we apply the penalty approach in [27]. The Constrained Total Least Squares (CTLS) and the Riemannian SVD are possible alternatives, which under mild conditions are equivalent [21]. We have extended the CTLS algorithm in [25] to our approximate GCD problems. Our initial experiments show that the CTLS approach can also achieve globally optimal backward errors. We will continue to investigate the numerical stability of the various structure preserving approximation techniques, including linear programming, on larger inputs.

So far, we are focusing on speed of convergence and accuracy of approximation. We use standard linear algebra algorithms for our arising least squares problem. For univariate polynomials, the displacement structure of matrices which arise has been exploited to speed the cost of each individual iteration without loss of accuracy [35, 24]. Those results carry over to our coefficient matrices at least in the univariate case; in the multivariate case we hope to develop efficient displacement operators in the near future.

The problem of computing approximate factorizations of multivariate complex polynomials can also be solved by the STLN approach [16], and therefore we can again introduce additional linear constraints on the coefficients of the minimally deformed and factorizable polynomial. We hope to investigate the performance of the linearly constrained structured total least norm methods applied to the approximate factorization and related problems in the future.

Acknowledgement: We thank Ivan Markovsky for his comments on the STLN iteration. Example 4.2 answers a question posed to us by James Demmel at the BIRS "Challenges" workshop in October 2005. We thank Rong Xiao and Bican Xia for helping us compute the non-monic global minimum of Example 4.2. We also thank Mark Giesbrecht and John May for their comments on structure-preserving methods, the referees for their helpful comments, and Zhonggang Zeng for sending us his code for computing approximate GCDs of polynomials.

6. REFERENCES

[1] Anda, A. A., and Park, H. Fast plane with dynamic scaling. *SIAM J. Matrix Anal. Applic. 15* (1994), 162–174.

[2] Anda, A. A., and Park, H. Self-scaling fast rotations for stiff and equality-constrained linear least squares problems. *Linear Algebra and Applications 234* (1996), 137–161.

[3] Beckermann, B., and Labahn, G. A fast and numerically stable Euclidean-like algorithm for detecting relative prime numerical polynomials. *J. Symbolic Comput. 26* (1998), 691–714.

[4] Botting, B., Giesbrecht, M., and May, J. Using Riemannian SVD for problems in approximate algebra. In Wang and Zhi [33], pp. 209–219.

[5] Chu, M. T., Funderlic, R. E., and Plemmons, R. J. Structured low rank approximation. *Linear Algebra and Applications 366* (2003), 157–172.

[6] Corless, R. M., Gianni, P. M., Trager, B. M., and Watt, S. M. The singular value decomposition for polynomial systems. In *Proc. 1995 Internat. Symp. Symbolic Algebraic Comput. ISSAC'95* (New York, N. Y., 1995), A. H. M. Levelt, Ed., ACM Press, pp. 96–103.

[7] Corless, R. M., Watt, S. M., and Zhi, L. QR factoring to compute the GCD of univariate approximate polynomials. *IEEE Transactions on Signal Processing 52* (Dec. 2004), 3394–3402.

[8] Dunaway, D. K. Calculation of zeros of a real polynomial through factorization using Euclid's algorithm. *SIAM J. Numer. Anal. 11*, 6 (1974), 1087–1104.

[9] Eckart, C., and Young, G. The approximation of one matrix by another of lower rank. *Psychometrika 1*, 3 (Sept. 1936), 211–218.

[10] Gao, S., Kaltofen, E., May, J. P., Yang, Z., and Zhi, L. Approximate factorization of multivariate polynomials via differential equations. In Gutierrez [12], pp. 167–174.

[11] von zur Gathen, J., and Gerhard, J. *Modern Computer Algebra*. Cambridge University Press, Cambridge, New York, Melbourne, 1999. Second edition 2003.

[12] Gutierrez, J., Ed. *ISSAC 2004 Proc. 2004 Internat. Symp. Symbolic Algebraic Comput.* (New York, N. Y., 2004), ACM Press.

[13] Hitz, M. A., and Kaltofen, E. Efficient algorithms for computing the nearest polynomial with constrained roots. In *Proc. 1998 Internat. Symp. Symbolic Algebraic Comput. (ISSAC'98)* (New York, N. Y., 1998), O. Gloor, Ed., ACM Press, pp. 236–243.

[14] Kahan, W. Numerical linear algebra. *Canadian Math. Bull. 9* (1966), 757–801.

[15] Kaltofen, E., and May, J. On approximate irreducibility of polynomials in several variables. In *ISSAC 2003 Proc. 2003 Internat. Symp. Symbolic Algebraic Comput.* (New York, N. Y., 2003), J. R. Sendra, Ed., ACM Press, pp. 161–168.

[16] Kaltofen, E., May, J., Yang, Z., and Zhi, L. Approximate factorization of multivariate polynomials using singular value decomposition. Manuscript, 22 pages. Submitted, Jan. 2006.

[17] Kaltofen, E., Yang, Z., and Zhi, L. Structured low rank approximation of a Sylvester matrix. Manuscript, 15 pages, Oct. 2005. Preliminary version in SNC 2005 Proceedings, Dongming Wang and Lihong Zhi eds., pp. 188–201, distributed at the International Workshop on Symbolic-Numeric Computation in Xi'an, China, July 19–21, 2005.

[18] Kaltofen, E., Yang, Z., and Zhi, L. Structured low rank approximation of a generalized Sylvester matrix. In *Proc. of the Seventh Asian Symposium on Computer Mathematics* (Seoul, South Korea, 2005), S. Pae and H. Park, Eds., Korea Institute for Advanced Study, pp. 219–222. Extended abstract.

[19] Kaltofen, E., Yang, Z., and Zhi, L. Approximate greatest common divisors of several polynomials with linearly constrained coefficients and singular polynomials. Manuscript, 16 pages, Apr. 2006.

[20] Karmarkar, N. K., and Lakshman Y. N. On approximate GCDs of univariate polynomials. *J. Symbolic Comput. 26*, 6 (1998), 653–666.

[21] Lemmerling, P. *Structured total least squares: analysis, algorithms and applications.* Dissertation, Katholieke Universiteit Leuven, Belgium, 1999.

[22] Lemmerling, P., Mastronardi, N., and Van Huffel, S. Fast algorithm for solving the Hankel/Toeplitz Structured Total Least Squares problem. *Numerical Algorithms 23* (2000), 371–392.

[23] Li, B., Liu, Z., and Zhi, L. Fast low rank approximation of a Sylvester matrix. In Wang and Zhi [33], pp. 202–208.

[24] Li, B., Yang, Z., and Zhi, L. Fast low rank approximation of a Sylvester matrix by structured total least norm. *J. JSSAC (Japan Society for Symbolic and Algebraic Computation) 11*, 3,4 (2005), 165–174.

[25] Mastronardi, N., Lemmerling, P., and Van Huffel, S. Fast structured total least squares algorithm for solving the basic deconvolution problem. *SIAM J. Matrix Anal. Applic. 22*, 2 (2000), 533–553.

[26] May, J. P. *Approximate factorization of polynomials in many variables and other problems in approximate algebra via singular value decomposition methods.* PhD thesis, North Carolina State Univ., Raleigh, North Carolina, Aug. 2005.

[27] Park, H., Zhang, L., and Rosen, J. B. Low rank approximation of a Hankel matrix by structured total least norm. *BIT 39*, 4 (1999), 757–779.

[28] Pope, S., and Szanto, A. Nearest multivariate system with given root multiplicities. Manuscript available at `http://www.math.ncsu.edu/~aszanto/papers.html`, 2005.

[29] Rump, S. M. Structured perturbations part I: Normwise distances. *SIAM J. Matrix Anal. Applic. 25*, 1 (2003), 1–30.

[30] Sasasaki, T., and Noda, M. T. Approximate square-free decomposition and root-finding of ill-conditioned algebraic equations. *J. Inf. Process. 12* (1989), 159–168.

[31] Schönhage, A. Quasi-gcd computations. *Journal of Complexity 1* (1985), 118–137.

[32] Stewart, G. W. *Introduction to Matrix Computations.* Academic Press, Inc., New York, 1973.

[33] Wang, D., and Zhi, L., Eds. *Proc. 2005 International Workshop on Symbolic-Numeric* (July 2005). Distributed at the Workshop in Xi'an, China.

[34] Zeng, Z., and Dayton, B. H. The approximate GCD of inexact polynomials part II: a multivariate algorithm. In Gutierrez [12], pp. 320–327.

[35] Zhi, L. Displacement structure in computing approximate GCD of univariate polynomials. In *Proc. Sixth Asian Symposium on Computer Mathematics (ASCM 2003)* (Singapore, 2003), Z. Li and W. Sit, Eds., vol. 10 of *Lecture Notes Series on Computing*, World Scientific, pp. 288–298.

[36] Zhi, L., Noda, M.-T., Kai, H., and Wu, W. Hybrid method for computing the nearest singular polynomials. *Japan J. Industrial and Applied Math. 21*, 2 (June 2004), 149–162.

[37] Zhi, L., and Wu, W. Nearest singular polynomial. *J. Symbolic Comput. 26*, 6 (1998), 667–675.