2.12.3 Hybrid Methods

2.12.3.1 Introduction. We here discuss an active research area: algorithms for symbolic/numeric computation. The main goal of hybrid symbolic-numeric computation is to extend the domain of efficiently solvable problems by combining the methods of numerical and symbolic computation.

One can take a broad point of view, and call any method a hybrid symbolic-numeric method provided that it solves a mathematical problem, and involves some aspects of numerical computing and some aspects of symbolic computing. This is similar to what Knuth calls a *seminumerical algorithm*, one that lies "on the borderline between numeric and symbolic calculation" [Knuth 1981, p. v]. This definition includes, then, such varied topics as:

- conversion of polynomial problems to eigenvalue problems,
- polynomial arithmetic (including GCD) with numerical coefficients,
- interval arithmetic (especially when correlations between intervals are tracked symbolically),
- symbolic pre-computation of expressions for later efficient and/or stable numerical evaluation (e.g. in C or Fortran compilers, not just computer algebra systems),
- code generation by computer algebra systems for later solution of numerical problems, for example PDE,
- automatic differentiation of formulas and programs,
- construction of special-purpose numerical methods for differential equations that automatically preserve invariants, and
- exact computation by intermediate use of floating-point arithmetic, for sake of speed.

The problem formulation can be, as above, entirely symbolic. An example of practical interest is the computation of the sign of the determinant of a rational matrix, and the result may be validated (see also section **??**). An interesting extension of this idea is the inverse symbolic calculator (`http://www.cecm.sfu.ca/projects/ISC/`), where for example a high precision floating point approximation of an algebraic number is computed first, and then a defining polynomial with integer coefficients is found via lattice basis reduction (see section **??**).

Clearly we will be unable to survey all possible hybrid symbolic/numeric methods in this short article, under this inclusive definition. We focus on Symbolic-Numeric Algorithms for Polynomials (SNAP), partly because of recent progress but also because polynomial problems play a historical role in the exposition and explication of difficulties characteristic for more general nonlinearities. See also the recent survey [Emiris 1999], for a detailed discussion of SNAP and many references.

2.12.3.1.1 History. We choose to date the nascence of SNAP, as a field of study, from the 1996 SNAP conference at INRIA (Sophia-Antipolis). Several papers presented at that conference were later published in a special issue of the Journal of Symbolic Computation edited by Hans J. Stetter and Stephen M. Watt, published in 1998.

One can find papers relevant to the SNAP field that appeared earlier than this conference. Some examples are [Kharitononv 1979; Kaltofen 1985; Schönhage 1985; Auzinger and Stetter 1988; Corless et al. 1995]. Many other papers have shown that certain algorithms for problems with exact data are unstable for problems with approximate data. Perhaps the best known of these is the Chauvenet-prize paper by Wilkinson [Wilkinson 1984], which shows very clearly that computing the characteristic polynomial and then finding its roots is not a stable method to find the eigenvalues of a matrix. This can be used to show the well-known result that trying to find the roots of a multivariate system by first using the Buchberger algorithm to compute a lexicographic-ordered Gröbner basis is, when implemented in floating-point arithmetic, unstable for approximate polynomials. To see this, simply consider the eigenproblem $Ax = \lambda x$ with the normalization $x_1^2 + x_2^2 + \cdots + x_s^2 = 1$ as an $s+1$-variate polynomial problem of total degree 2. Applying the Buchberger algorithm to this problem with the lexicographic ordering $x_1 > x_2 > \cdots > x_s > \lambda$, we see that the resulting Gröbner basis will contain the characteristic polynomial of A as its first element. Therefore, by the result of Wilkinson described above, this approach is numerically unstable as a method of finding roots.

2.12.3.2 Definitions and Techniques.

2.12.3.2.1 Continuity. One of the most important ideas in symbolic-numeric computation is the mathematical notion of continuity. Pure symbolic computation in algebraic domains does not concern itself with continuity or the lack thereof, and this is one of the main differences between pure symbolic computation and symbolic-numeric computation. Continuity is also important for pure symbolic computation with the elementary functions of analysis and for problems with parameters, but this has heretofore received less attention in the computer algebra community than the important algebraic aspects of such computation. But, in using numerical computation, we are forced to confront the issue. The solutions of many problems in this area are discontinuous with respect to changes in the data. It follows that applying exact methods to problems with approximate data does not always approximate the desired result. A key aspect of all symbolic-numeric methods and SNAP in particular is to define well-posed problems that may be solved by any mathematical method.

2.12.3.2.2 Approximate Polynomials. An important idea of this framework for computer algebraists working with polynomials, superficially similar to an idea from the field of interval arithmetic (see section **??**), is the notion of an *approximate polynomial*. An approximate polynomial is a polynomial with inexactly-known coefficients. More formally, consider the space of polynomials $\mathbb{R}[x]$ (or $\mathbb{C}[x]$), together with a metric $d(f, g)$ giving a "distance" from the polynomial f to the polynomial g. Usually we assume that the metric $d(f, g)$ is given by some norm $d(f, g) = \|f - g\|$. In Section 2.12.3.2.5 we take norms up further. Given some polynomial f_0 and scalar $\varepsilon > 0$, define a polynomial ε-neighbourhood $N_{f_0,\varepsilon} \subset \mathbb{R}[x]$ (or $\mathbb{C}[x]$) as the set $N_{f_0,\varepsilon} = \{f : d(f, f_0) \leq \varepsilon\}$. Later we generalize this to allow a vector of tolerances, some of which can be

zero. Every member of an ε-neighbourhood is considered indistinguishable as an approximate polynomial. This membership is not an equivalence relation, in general. Therefore we see that the notion of approximate polynomial is tied to ε and to the given polynomial f_0. The multivariate generalization is straightforward.

The reason this idea is important is that in some problem contexts, any exact solution to an approximate polynomial problem (relative to the originally stated problem $P(f_0)$) is precisely as useful as an exact solution to $P(f_0)$ itself; moreover, solution of the approximate problem may be more efficient.

One context where approximate polynomials are used is as an intermediate stage in the solution of exact problems, for example by embedding an exact problem into an approximate framework in order to take advantage of any existing fast (e.g. iterative) algorithms for approximate solution in the approximate domain; one then has to ensure that the final answer is then transferable back to the desired exact answer. One example of this kind of problem is the search for a certified sign of the determinant of a matrix [Clarkson 1992; Brönnimann, H. and Yvinec, M. 2000]. Another context is the solution of a discontinuous problem known to be singular but given with imprecise inputs, say floating point coefficients. An example here is the problem of computing the rank of a matrix. Clearly, an infinitesimal perturbation can change the rank, because the rank of a matrix is a discontinuous function of the entries of a matrix, so one may need to consider the set of approximate matrices in an ε-neighbourhood.

2.12.3.2.3 Metrics and Norms. Another important aspect of this framework is that the choice of metric is crucial to the success (and applicability) of the algorithm. Most of the algorithms developed so far have concentrated on metrics derived from the simple norms, for instance, (weighted) 2-norms of the coefficients, so $\|f\|^2 = \sum_{k=0}^{n} w_k f_k^* f_k$ (where the f_k are the coefficients of f in some basis, usually the monomial basis and w_k are non-negative weights); the 1-norm; or the max (infinity, component-wise) norm. In Chebychev's and Solotareff's approximation problems (their problems seek nearby polynomials of lower degrees), the distance between two univariate complex polynomials is measured by the maximum difference of values on the unit interval: $d(f,g) = \max\{|f(a) - g(a)| : -1 \le a \le 1\}$. This is the infinity norm, considered as a function norm, not a coefficient norm, on the interval $[-1,1]$. We do not use the symbol $\|\cdot\|_\infty$ for this because of the possibility of confusion with the corresponding coefficient norm $\|p\|_\infty = \max\{|p_k|, 0 \le k \le \deg p\}$. Victor Ya. Pan (see below) uses the "spectral metric" $d(f,g) = \max_i \min_j |\rho_i - \sigma_j|$, where the ρ_k and σ_k are the roots of f and g respectively (the case where deg f is not the same as deg g needs more care).

Finally, the choice of representation for f plays a role. We consider f fundamentally as a function $f\colon \mathbb{R}^s \to \mathbb{R}$ (or $\mathbb{C}^s \to \mathbb{C}$). Many norms can be understood better in this fashion. For example, by Parseval's identity, the 2-norm of the vector of coefficients of a univariate polynomial f is also the integral norm

$$\|f\|_2^2 = \frac{1}{2\pi} \int_C \overline{f(z)} f(z)\, dz$$

where C is the unit circle in $\mathbb{C}$. Therefore the size of f in the unit disk determines (and is determined by) the size of the 2-norm of the vector of coefficients of f. This shows clearly that the location of the origin and the scaling of the variable z (sometimes, but not always, at liberty in applications) matters to the algorithms that we will discuss here. Note that translating z to $z + a$ is ill-conditioned, amplifying errors in the coefficients by as much as $(1 + |a|)^n$, where n is the degree of f. Further, relative errors may be infinitely amplified [Corless et al. 1999]. This corresponds to loss of sparsity in the symbolic context and is thus to be avoided both symbolically and numerically.

2.12.3.2.4 Conditioning and Posedness. We now define some terms, commonly used in numerical analysis, that we find useful in this context. We follow standard usage in numerical analysis and say that a problem is *well-posed* if it has a unique solution that depends continuously on the problem parameters; a problem is *ill-posed* otherwise. Typically, in the SNAP context, a problem will be ill-posed not because of lack of existence or uniqueness, but rather because an important characteristic (such as rank, or degree of the GCD) fails to be continuously dependent on the problem parameters. We call the norm of the difference between a computed solution f and the true, exact solution f_{true} the *forward error* $\|f - f_{\text{true}}\|$. We call the difference between the problem that the computed f solves, and the problem that we originally wished to solve, the *residual*; we call the norm of the residual the *backward error*. For example, the backward error corresponding to the computed solution x of a linear system $Ax = b$ is the norm of the residual[1] $\|r\| = \|b - Ax\|$. Similarly, a residual (backward error) in the problem of finding the GCD of two approximate polynomials f and g is a set $(\Delta f, \Delta g)$ $(\|(\Delta f, \Delta g)\|)$ of perturbations to f and g that allows us to write the GCD as $h = s(f + \Delta f) + t(g + \Delta g)$. Such residuals may not be unique.

We call any linear measure of the asymptotic sensitivity of our stated problem to changes in its input arguments a *structured condition number* of the problem [Turing 1948; Higham 1996]. We emphasize that this linear measure does not help us to understand the effects of large changes in the data. We have, for small enough changes, that the forward error is roughly equal to the structured condition number times the backward error; but more to the point, the structured condition number measures the sensitivity of the problem to changes in its data.

The definition of structure that we are using is as follows. A problem that depends on more parameters and for which there exists a substitution for the parameters by functions in fewer parameters is more structured. For example, an $n \times n$ matrix whose entries depend only on $O(n)$ independent parameters has more structure than a general $n \times n$ matrix. The condition number of the structured problem is by the chain rule for differentiation equal to the product of the condition number of the unstructured problem times the condition number for the substitution, which is often less.

[1] Note that already here the deformation has a structure: the matrix A is taken exactly, while the vector b is approximate, as in the problem of least squares. The structured backward error for an approximate matrix can also be analyzed [Oettli and Prager 1964]. The corresponding notion for curve fitting is *total* least squares.

We say that a problem with a large[2] structured condition number is *ill-conditioned*, and *well-conditioned* otherwise. An ill-posed problem is of course ill-conditioned (with infinite condition number); but the notion "well-conditioned" is a significant, quantitative, refinement of the notion "well-posed"; it is perfectly possible for a well-posed problem to be hopelessly ill-conditioned.

Note that the conditioning of a problem is important even if there are no rounding errors whatever in a computation. Fundamentally, this is because the condition number of a problem is the reciprocal of the distance to the nearest singular problem (a fact noted in [Demmel 1987] as holding in a very general problem context). In the symbolic context, one may seek this nearest singular problem, for instance, a minimal perturbation of the inputs so that a curve factors. The condition number measures sensitivity to errors in the data, which the model of "approximate polynomial" assumes *a priori*. An ill-conditioned problem will also, of course, amplify any rounding errors that happen during computation, but this is often a secondary consideration.

Finally, we say that an algorithm is *numerically stable* if it produces the exact answer to a problem near to the one posed: that is, the answer it produces has small backward error. Therefore, a well-conditioned problem solved by a numerically stable algorithm will have small *forward* error. This nomenclature splits the difficulty of explanation of numerical behaviour into two useful components: *algorithms* are stable or unstable; *mathematical problems* are well-conditioned or ill-conditioned.

A good numerical analysis of an algorithm for approximate polynomials will take both forward and backward error into account. The algorithm itself should provide an estimate of the structured condition number of the problem.

2.12.3.2.5 Dual Norms and a Useful Inequality. We follow the approach of the survey [Stetter 1999]. The key observation is that some recent SNAP results are straightforward implications of the Hölder inequality. In finite-dimensional linear spaces, we have that the equality sign is attained in the Hölder inequality [Hardy et al. 1951, pp. 24–26] at explicitly-known "witness" vectors. These witness vectors allow us to solve useful minimax problems explicitly.

Following the notation of [Stetter 1999], v^T denotes the transposition of a column vector v into a row vector, without complex conjugation. Suppose $\mathbb{C}^n$ is equipped with a norm $\|\cdot\|$. The space of linear functionals v^T on $\mathbb{C}^n$ may then be equipped with the dual norm $\|\cdot\|^*$ defined by

$$\|v^T\|^* := \max_{\|u\|=1} \left|v^T u\right| \ . \tag{1}$$

For example, it is well-known that the p-norm is dual to the q-norm if $1/p+1/q = 1$, for $1 \le p, q \le \infty$.

[2] "Large" means large relative to the problem context; in some cases a condition number of 5 is "large", whilst in others a condition number of 10^{10} might be acceptably small. A condition number should be provided; it is up to the user to do something intelligent with it.

Proposition 1 in [Stetter 1999] is: For each $u \in \mathbb{C}^n$ with $\|u\| = 1$, there exist vectors $v \in \mathbb{C}^n$ with $\|v^T\|^* = 1$ such that $|v^T u| = 1$. That is, *the maximum value is attained*. Moreover, for the associated p and q norms, the vector v in this proposition is given explicitly for $p < \infty$ ([Stetter 1999] also gives formulas for $p = \infty$) by

$$v_k = \gamma |u_k|^{p-2} \overline{u_k} \,, \text{ for } 1 \le k \le n \,. \tag{2}$$

where $\gamma \in \mathbb{C}$ is an arbitrary unimodular constant (that is, $|\gamma| = 1$). The proof is a straightforward application of the Hölder inequality.

This general result allows one to derive useful bounds and formulae for such problems as explicitly finding the nearest polynomial with a given zero. Corollary 4 of [Stetter 1999] states that if $p(x) = \sum_{k=0}^{n} a_k x^k = \mathbf{a}^T \mathbf{x}$, where $\mathbf{x} = [1, x, x^2, \ldots, x^n]^T$, then $(p + \Delta p)(z) = 0$ with $\Delta p(x) = \sum_{k=0}^{n} \Delta a_k x^k$ requires

$$\|\Delta a\|^* \ge \frac{|p(z)|}{\|\mathbf{z}\|} \,, \tag{3}$$

and moreover equality is attained at perturbations Δa known explicitly from equation (2). This is a generalization of the work reported in [Manocha and Demmel 1995; Corless et al. 1995; Hitz and Kaltofen 1998; Zhi and Wu 1998; Hitz et al. 1999], observed by Hitz in 1999. In words, pseudozeros, which are values of z for which $p(z)$ is small, are the roots of explicitly-known approximate polynomials.

2.12.3.2.6 Restrictions on the Coefficients. Restrictions of coefficients are also addressed in [Karmarkar and Lakshman Y. N. 1995; Hitz and Kaltofen 1998], for example keeping the perturbed polynomials monic. In [Stetter 1999] the following definition is given. If we want to restrict our approximate polynomials so that only certain coefficients are allowed to vary by $|\Delta a_k| \le \varepsilon_k$, for $k \in \mathcal{K} \subset \{1, 2, \ldots, n\}$, we must use a weighted max-norm:

$$\|\Delta a^T\|_\varepsilon^* := \max_{k \in \mathcal{K}} \frac{|a_k|}{\varepsilon_k} \,. \tag{4}$$

The dual norm is then

$$\|\mathbf{z}\|_\varepsilon := \sum_{k \in \mathcal{K}} \varepsilon_k |z|^k \,. \tag{5}$$

Setting individual $\varepsilon_k = 0$ prevents variation in the kth coefficient of p. Theorem 8 from [Stetter 1999] states that for a given $z \in \mathbb{C}^s$, then the neighbourhood $N_\varepsilon(p) := \{\tilde{p} : \|p - \tilde{p}\|_\varepsilon^* \le 1\}$ contains polynomials $\tilde{p}$ with $\tilde{p}(z) = 0$ iff $|p(z)| \le \|\mathbf{z}\|_\varepsilon$.

2.12.3.2.7 Local Versus Global Solutions. Currently, SNAP algorithms can be divided in two categories: local solutions (such as quotient-divisor iteration or Newton iteration to find approximate GCD [Chin et al. 1998]) and global solutions (such as parameterizing exactly nearest polynomial pairs by an unknown common root (using equation (2)), setting up polynomial equations for the unknown, and globally solving them [Karmarkar and Lakshman Y. N.

1995]). Local algorithms are simpler to find than global algorithms, and correspond naturally to iterative solutions for local minima of optimization problems. Local optimization, because of its limited scope, can be more efficient than global optimization. For example, factoring bivariate approximate polynomials that are near to exactly factorable polynomials is possible in this fashion [Huang et al. 2000; Corless et al. 2001b]. Local solution methods do not find the absolute nearest factorable polynomial, however, if the given polynomial is far from a factorable one. Global methods are more mathematically satisfying, and, if efficient, more useful in practice because they come with a guarantee that they produce the best of all possible answers.

2.12.3.2.8 A Note on Asymptotically Valid Algorithms. It is very common in numerical analysis, and applied mathematics generally, to rely on the validity or utility of a formula or algorithm that can be proved to be valid only in an asymptotic limit. For example, one uses a condition number as an estimate of the sensitivity of a problem to small changes, when technically this is only valid for "infinitesimal" changes; that is, in modern pure analytical terminology, for all $\delta > 0$ there exists an $\varepsilon_0 > 0$ for which the difference between the forward error E and the estimate $C\varepsilon$, namely $\|E - C\varepsilon\|$, is less than δ if the backward error ε is less than ε_0. This is summarized by saying that the condition number C is an asymptotically valid estimate as $\varepsilon \to 0$. Notice that no prescription for computing ε_0 from δ is given; *but experience with many practical problems shows* that the mere existence of asymptotic validity is often enough to ensure the practical utility of the estimate: in practice, we have $\|E - C\varepsilon\| = O(\varepsilon^2)$ and moreover that the constant of proportionality is of moderate size, unless we are near a singularity.

So, reasoning by analogy, one would expect that a *local algorithm* that is asymptotically valid as the size of the necessary regularization goes to zero would be useful in practice, giving the desired answers even when we cannot *a priori* guarantee that it will. Classical examples of this include Newton's method, which often converges even when we can't prove ahead of time that it will. A further point is that these local algorithms are self-validating: when they work, you can detect convergence and verify that the residual is small. When they fail, which is detectable, we cannot say whether the problem has a solution or not. We see that finding an efficient global algorithm for a SNAP problem is both harder and more valuable. An example from statistics is the method of least squares curve fitting that finds the global minimum. More generally, whenever a problem is convex, a local method will give a global minimum.

2.12.3.3 Selecta. We present here discussions relating to a number of approximate polynomial problems. Because the GCD is one of the most fundamental operations one encounters beyond ring arithmetic, it has received the most study. In the approximate setting, it gives us our first interesting algorithms. We therefore devote considerable attention to the GCD here. We then discuss algorithms for other fundamental operations, including matrix eigenproblems, approximate polynomial decomposition and factoring.

2.12.3.3.1 Approximate GCD. We call the GCD of approximate polynomials an "approximate GCD". The precise definition is as follows.

Definition: Approximate GCD. Given two approximate polynomials f and g, a polynomial metric d, and a tolerance $\varepsilon > 0$, we say that h is an *approximate* GCD of f and g if there exist polynomials Δf and Δg such that the exact GCD$(f + \Delta f, g + \Delta g) = h$ and $d(f, f + \Delta f) \leq \varepsilon$ and $d(g, g + \Delta g) \leq \varepsilon$. [Variations on this definition include, for example, the case where the last condition is replaced by $\|\Delta g\|_2^2 + \|\Delta f\|_2^2 \leq \varepsilon^2$.]

Call the GCD *nontrivial* if the degree of h is *larger than* 1; moreover if the degree of h is the maximum over all approximate GCDs of f and g in the ε-neighbourhood, then we call h a *maximal degree approximate* GCD or just maximal approximate GCD. Approximate GCDs (even maximal approximate GCDs) are not necessarily unique. If there is no approximate GCD with degree h larger than 1, we say by abuse of notation that there is no approximate GCD; this is really a convenient shorthand for saying that there is no *nontrivial* approximate GCD. Note that this is not a "quasi-GCD" in the sense of [Schönhage 1985]; that work assumes the input polynomials are not known to complete precision, but are exact, not approximate, and that more digits of the input can be obtained on demand. Furthermore, the paper [Corless et al. 2001a] uses an example from [Schönhage 1985] to show that a step of the Euclidean algorithm does not preserve approximate GCD, irrespective of the arithmetic system the step is carried out in. Further, the result of the Euclidean algorithm is not necessarily of a degree that is a lower bound on the degree of the approximate GCD: sometimes it can be spuriously too high. This fact contradicts statements by several authors.

The problem of computing an approximate GCD in polynomial-time, the minimax problem, is solvable in exponential time by methods from the existential theory of the reals (see section **??**) and was finally solved in [Karmarkar and Lakshman Y. N. 1995]. We will give their solution below. Two attempts at the solution are noteworthy, although they do not provide a complete solution.

The first is based on the use of the Singular Value Decomposition or SVD [Corless et al. 1995; Gianni et al. 1998]. The smallest singular value of the Sylvester matrix for f and g gives a lower bound on the distance to the nearest singular Sylvester matrix. The SVD method can give overly weak lower bounds, because the smallest singular value is the 2-norm distance to the nearest singular matrix of the same dimension [Eckart and Young 1936], but not the 2-norm distance to the nearest singular Sylvester matrix, which must necessarily be at least as far away, and can sometimes be much farther away. The partial algorithm given in that paper for solving the optimization problem for approximate GCD relied on the SVD clearly separating singular values, and that separation is prone to fail for even moderately large problems. Nonetheless, the paper [Corless et al. 1995] gives a precise definition of approximate GCD, and phrases the problem of finding it as an optimization problem.

A second attempt is based on root-matching. Instead of working with the polynomial coefficients and trying to generalize the rational algorithms so successful in the exact computation case, Victor Ya. Pan proposed in [Pan 1998]

to compute approximate roots of univariate approximate polynomials, efficiently sort to identify the roots that allow us to compute the spectral metric, and decide which nearby roots are to be considered equivalent. Once these decisions are taken, it is relatively easy to construct the GCD by recovering its coefficients from the root representation.

An example of why this approach is interesting is the "multiplicity problem" $f = z^n - a^n$ compared to $g = z^n$. It is clear that for $|a| < 1$ and large enough n, any coefficient-based metric will imply that these two polynomials have nontrivial approximate GCD. However, in the spectral metric, since the roots of f are on the circle of radius $|a|$ and the roots of g are zero, the spectral distance between these two polynomials is independent of n. This implies that for problems where the approximate GCD has roots with very high multiplicity, this approach may give more satisfactory results than coefficient-metric based methods. The strength of this approach is the possibility that the root locations can be very sensitive to small changes in the coefficients of a polynomial, as Wilkinson has observed. Therefore, constraining the roots instead of the coefficients may give better results.

A Global Algorithm for Approximate GCD. Using the Hölder inequality, one can derive explicit formulas for the nearest polynomials with a common root. These can be used to give a global algorithm to compute the nearest polynomials with a non-trivial GCD. This approach was introduced and shown to be polynomial time in [Karmarkar and Lakshman Y. N. 1995]. Using the 2-norm, the algorithm runs as follows.

If $f(x) + \Delta f(x)$ and $g(x) + \Delta g(x)$ have a common zero, say $x = \alpha$, then by the Hölder inequality formula we have

$$0 = f(\alpha) + \Delta f(\alpha)$$

and

$$\|\Delta f\| \geq |f(\alpha)| / \|[1, \alpha, \alpha^2, \ldots, \alpha^n]\| \,.$$

Equality is attained (i.e. the minimum $\|\Delta f\|$ is attained) at a vector of coefficients for Δf that is explicitly known as a rational polynomial in α. A similar formula holds for Δg. Therefore the minimum value of $P(\alpha) = \|\Delta f\|^2 + \|\Delta g\|^2$ occurs at one of the finitely many zeros of the derivative $P'(\alpha)$. The search for the zeros of the numerator of this rational polynomial and comparison of the resulting values of $P(\alpha)$ can be carried out in polynomial time.

Note that the method determines the nearest pair of polynomials with a common root. That pair may share another root in common. It is possible to force the GCD to be of higher degree, but then the method becomes exponential in the number of common roots, see [Karmarkar and Lakshman Y. N. 1995] for details. The parametric optimization approach by Karmarkar and Lakshman has led to several other polynomial-time solutions for global optimization problems:

- Compute the nearest polynomial in Euclidean distance with a root on a piecewise rational parametric curve [Hitz and Kaltofen 1998].

- Compute the nearest polynomial in coefficient-wise distance with a real root [Hitz et al. 1999].
- Compute the nearest polynomial in Euclidean distance with a k-fold root, where k is arbitrary [Zhi and Wu 1998].

The task of proving that two given approximate polynomials are relatively prime, i.e. that they have no nontrivial approximate GCD, is likely to be very common in practice. The algorithm of Beckermann and Labahn [Beckermann and Labahn 1998] is currently the fastest known. It uses fast matrix techniques, with look-ahead to skip ill-conditioned blocks, to compute a better bound than the unstructured condition number for the distance to the nearest singular Sylvester matrix. This is, at the time of writing, being implemented by Claude-Pierre Jennerod and George Labahn.

Certified Approximate GCD. André Galligo, and his co-workers David Rupprecht, Henri Lombardi, Ioannis Emiris and Laureano Gonzalez-Vega, have developed a refined point of view on resolution of approximate polynomial systems. Their work is based both on geometry (stratification of the parameter space of the coefficients of the inputs) and *a priori* inequalities obtained by a refined analysis of the SVD of Sylvester-like matrices [Galligo and Rupprecht 2001; Emiris et al. 1996, 1997; Rupprecht 1999; Galligo et al. 2001].

For the computation of approximate GCD of two univariate polynomials, here as above, the coefficients of these polynomials are known with a limited precision, governed by a tolerance ε. They proved so-called gap theorems and described an algorithm which produces a partition into intervals of tolerance, say $d = 0$ if $\varepsilon < 5 \cdot 10^{-6}$, $d = 3$ if $4 \cdot 10^{-5} < \varepsilon < 2 \cdot 10^{-3}$, and $d = 6$ if $9 \cdot 10^{-2} < \varepsilon < 1$, where d is the maximum of the degrees of the GCD of two near-by polynomials, with respect to the given tolerance. Notice that there are gaps in these bounds where the maximal degree d is not certified; nonetheless this information is useful and may allow deductions to be made about the underlying problem.

This means that in the corresponding gap of tolerances, the algorithm judges that the situation is too costly (in term of effective computations) to be rigorously solved. They try to keep these interval gaps as small as possible. Geometrically this situation corresponds in the parametric space to the case where the point (attached to the set of coefficients of the two polynomials) is almost equidistant to several strata. These strata are singular varieties in high dimensional spaces and the optimization process will be ill-conditioned.

At the time of writing, implementations of this algorithm are able to compute a certified approximate GCD of two polynomials of degree 100 in under one minute of cpu time.

2.12.3.3.2 Interval Polynomials and the Kharitonov Theorem. The Kharitonov theorem from control theory [Kharitononv 1979; Minnichelli et al. 1989] predates most SNAP work, and leads to a powerful stability criterion of an interval polynomial. Recall that an interval polynomial includes all approximate polynomials in a neighbourhood. Therefore, results about interval polynomials imply results about approximate polynomials. We state the simplest form of the

Kharitonov theorem. Given are $2n$ rational numbers $\underline{a}_i, \bar{a}_i$. Let P be the *interval* polynomial

$$P = \{x^n + a_{n-1}x^{n-1} + \cdots + a_0 \mid \underline{a}_i \le a_i \le \bar{a}_i \text{ for all } 0 \le i < n\}.$$

Then every polynomial in P is *Hurwitz* (all roots have negative real parts), if and only if the four "corner" polynomials

$$g_k(x) + h_l(x) \in P, \quad \text{where } k = 1, 2 \text{ and } l = 1, 2,$$

with

$$\begin{aligned} g_1(x) &= \underline{a}_0 + \bar{a}_2x^2 + \underline{a}_4x^4 + \cdots, \quad h_1(x) = \underline{a}_1x + \bar{a}_3x^3 + \underline{a}_5x^5 + \cdots, \\ g_2(x) &= \bar{a}_0 + \underline{a}_2x^2 + \bar{a}_4x^4 + \cdots, \quad h_2(x) = \bar{a}_1x + \underline{a}_3x^3 + \bar{a}_5x^5 + \cdots \end{aligned}$$

are Hurwitz.

The corner polynomials are easily tested for the Hurwitz condition, for example by a variant of Sturm sequences, and the condition constitutes the stability criterion for the corresponding differential equations (see [Gantmacher 1960, Ch. XV]). There now exist many generalizations of Kharitonov's theorem.

2.12.3.3.3 Polynomial Systems and Matrix Eigenvalues. In the paper [Auzinger and Stetter 1988] it is shown how to reduce the solution of a system of multivariate polynomials to an eigenvalue problem for a commuting family of matrices. As pointed out in [Stetter 1996], this is a key step forward, because given an eigenproblem we may use efficient and stable numerical methods to solve it, while computing a triangular basis (see sections **??** and **??**) may not be numerically stable. This approach has also been used for resultant formulations [Manocha and Demmel 1995; Emiris 1999].

The paper [Corless et al. 1997] gives a theorem showing that a nearly-commuting family of matrices can be simultaneously placed in nearly-upper triangular form by a unitary transformation (and therefore this process is numerically stable). Using clustering heuristics [Manocha and Demmel 1995] this method can then be used to solve problems with multiple roots numerically. Using a generic linear combination of the multiplication matrices to remove apparent but not actual multiplicities is recommended.

A Simple Example. Consider the problem of finding the intersections of x^2+y^2-1 and $x^2 - y^2 = 1/2$. A short computation shows that the normal set is $[1, y, x, xy]$ and the multiplication matrices corresponding to x and y are

$$M_x = \begin{bmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ \frac{3}{4} & 0 & 0 & 0 \\ 0 & \frac{3}{4} & 0 & 0 \end{bmatrix} \qquad \text{and} \qquad M_y = \begin{bmatrix} 0 & 1 & 0 & 0 \\ \frac{1}{4} & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & \frac{1}{4} & 0 \end{bmatrix}.$$

These matrices commute, as can be verified by direct computation. Both of these matrices have multiple eigenvalues, because of the symmetry in the problem.

However, the roots (intersections) of the original problem are all simple. Taking $M = \alpha M_x + (1-\alpha)M_y$ for some random α in $(0,1)$, we find a matrix whose eigenvalues are all simple. The eigenvectors are necessarily (from the normal set) of the form $[1, y, x, xy]$ where (x, y) are the roots of the original system. Computation of the (all simple) eigenvectors of M gives us (to complete accuracy) the four roots $(\pm 1/2, \pm\sqrt{3}/2)$.

A Larger Example. In the paper [Li et al. 1989] we find the following problem, used to show the effectiveness of a particular trick for homotopy methods (the trick was called the 'cheater's homotopy' in that paper). The example is the following system of two polynomial equations in two variables, x and y:

$$\begin{aligned} x^3y^2 + c_1x^3y + y^2 + c_2x + c_3 &= 0 \\ c_4x^4y^2 - x^2y + y + c_5 &= 0\,. \end{aligned} \tag{6}$$

The symbols c_k, $k = 1, \ldots, 5$, represent parameters, which can take values in $\mathbb{C}$. Generically, as noted in [Li et al. 1989], there are 10 complex roots of this system—by "generically" one means for almost all sets of values of the parameters; for some exceptional values (which are of course interesting) the number of roots may be different.

A lexicographic order Gröbner basis computation fails for this example (the answer is too large to be useful). A total-degree order basis is computable in seconds, and is small enough that the normal set can be computed again in seconds, and 10 by 10 multiplication matrices for x and y can be found, with entries rational in the parameters, from which eigenvalues can be found easily once the parameters are specified.

One can compare the Gröbner basis approach with the resultant approach, see [Emiris 1999], or with the homotopy approach, see [Sommese et al. 2001]. At the time of writing there is no one "best" method for all problems.

2.12.3.3.4 Functional Decomposition. Writing a polynomial f as a composition of two smaller polynomials, say $f(x) = g(h(x))$, can dramatically simplify working with that polynomial. Polynomial-time algorithms for computing the exact decomposition of exactly-known polynomials, when such decompositions exist, have been known for some time [Kozen and Landau 1989; von zur Gathen 1990; von zur Gathen and Weiss 1995]. Recently, these algorithms have been extended to the approximate polynomial case [Corless et al. 1999]. In the exact case, the algorithm is polynomial-time, and is guaranteed to succeed in finding a decomposition if such exists. For approximate polynomials, the algorithm is a local algorithm, and thus is guaranteed to find a decomposition only if the given approximate polynomial is sufficiently close to being a decomposable polynomial.

The algorithm of the paper is an iterative one. It takes as a starting point the $h(x)$ computed by the series reversion technique of the exact algorithm [von zur Gathen 1990] (here normalizing so that $\|f\|_2 = 1$ and g is monic) and then solves a linear least-squares problem to identify the corresponding g. Thereafter an alternating sequence of linear least-squares problems is solved to improve the (g, h) pair. At every stage the residual $\Delta f = f - g \circ h$ is available so the

convergence of the algorithm can be monitored. This technique of solving a nonlinear optimization problem by optimizing over alternating subsets of the problem parameters iteratively is well-known, and is linearly convergent at best. A Newton iteration can be used to speed up convergence, but as usual at the cost of more complicated steps at each iteration. According to the tests of the paper, the linearly-convergent method is expected to be faster in practice, as usually only a few iterations are sufficient to give approximate decompositions in the cases where the given polynomials are close to decomposable polynomials (which is likely to be the only case of practical interest).

2.12.3.3.5 Bivariate Factoring. Already in 1992, Kaltofen stated the problem of finding the nearest bivariate polynomial with nontrivial complex factors [Kaltofen 1992]. Subsequent work includes [Galligo and Watt 1997; Hitz et al. 1999; Rupprecht 2001]. In fact, the results in [Hitz et al. 1999] give a polynomial-time algorithm that finds the globally nearest polynomial with a complex factor of a fixed given degree. The method is based on the parametric optimization of [Karmarkar and Lakshman Y. N. 1995]. If the degree of a factor can be arbitrary, no such polynomial-time algorithm is known. Therefore, several heuristic numerical algorithms have been proposed. Those algorithms suppose that the input polynomial is near a polynomial that factors.

The paper [Huang et al. 2000] reports a local algorithm, apparently numerically stable, to find factors of bivariate approximate polynomials. The algorithm is of complexity exponential in the degree of the input. The paper [Corless et al. 2001b] gives another local algorithm to factor bivariate approximate polynomials. This algorithm is also apparently numerically stable, and is of polynomial complexity in the degree of the input (modulo an unproven conjecture). The algorithm uses numerical path following in the Riemann surface of one factor only and numerical implicitization to recover the factor from the path.

The papers [Kaltofen 1985; Sasaki et al. 1991, 1992; Sasaki 2001] discuss another interesting algorithm based on zero-sum identities of power-series solutions of $f(x, y) = 0$. In [Sasaki and Sasaki 1993] this algorithm is shown to be quite general, also being applicable to the problem of factoring over algebraic number fields and algebraic function fields. This algorithm is also apparently numerically stable and of polynomial complexity.

In all cases, when they succeed the algorithms produce answers which can, of course, be verified a posteriori to be factors of an explicitly constructible polynomial, which can be examined to see how near to the original it was. Moreover, the computed factors can be used as the starting point for a Newton refinement.

2.12.3.3.6 Matrix Spectra under Perturbation There is now a substantial theory of how matrix spectra and matrix canonical forms behave near a given matrix. For brevity, we only give three references [Edelman et al. 1997; Jeanerrod and Pflügel 1999; Hitz et al. 1999].

2.12.3.4 Outlook. Hybrid symbolic-numeric methods have proved their worth in applications such as Computer-Aided Geometric Design. Good implementa-

tions of many symbolic-numeric algorithms for polynomials are now publicly available. Nonetheless, much work remains to be done.

Computational complexity is an important consideration. Some of the problems in this area are computationally intractable (for example, the problem of computing the nearest singular matrix in entry-wise distance is NP-hard [Poljak and Rohn 1993]), or even recursively undecidable (for example, computing the GCD over the computable reals [Schönhage 1985]). Therefore, there is not much hope to attack these problems by numerical or any methods. Cook's hypothesis states that most instances of this problem will be computationally intractable. If a polynomial time algorithm becomes known that computes a singular matrix whose nearness to the input matrix is within a certain factor to the nearest one, the situation would change for that problem. Such is the case, for example, for lattice basis reduction problems (see section **??**). The recent polynomial-time algorithms for computing the globally optimal approximate GCD or approximate factors show, however, that some of the problems are computationally tractable. The complexity class of others remains to be discovered. Full use of randomization has not yet been made; combinatorial methods exploiting sparsity are being developed. Other mathematical areas, such as optimization and perturbation theory, are being scrutinized for useful results. The challenge of hybrid symbolic numeric algorithms is to explore the effects of imprecision, discontinuity, and algorithmic complexity by applying mathematical optimization, perturbation theory, and inexact arithmetic and other tools in order to solve mathematical problems that today are not solvable by numerical or symbolic methods alone.

Robert M. Corless (U. Western Ontario),
Erich Kaltofen (North Carolina State U.),
Stephen M. Watt (U. Western Ontario)

Note added on September 25, 2005: In section 2.12.3.3.1, references to the papers [Sasasaki and Noda 1989; Noda and Sasaki 1991; Ochi et al. 1991] were inadvertently omitted.

References

Auzinger, W. and Stetter, H. J. An elimination algorithm for the computation of all zeros of a system of multivariate polynomial equations. In Agarwal, Ravi P., Chow, Y. M., and Wilson, S. J., editors, *Numerical Mathematics*, volume 86 of *ISNM*, pages 11–30. Birkhäuser, 1988.

Beckermann, Bernhard and Labahn, George. When are two polynomials relatively prime? *Journal of Symbolic Computation*, 26:677–689, 1998.

Brönnimann, H. and Yvinec, M. Efficient exact evaluation of signs of determinant. *Algorithmica*, 27:21–56, 2000.

Chin, Paulina, Corless, Robert M., and Corliss, George F. Optimization strategies for the approximate GCD problem. In Gloor [1998], pages 228–235. ISBN 1-58113-002-3.

Clarkson, Kenneth L. Safe and efficient determinant evaluation. In *Proc. 33rd Ann. Symp. Foundations Comput. Sci.*, pages 387–395, Los Alamitos, California, 1992. IEEE Computer Society Press.

Corless, Robert M., Gianni, Patrizia M., and Trager, Barry M. A reordered Schur factorization method for zero-dimensional polynomial systems with multiple roots. In Küchlin [1997], pages 133–140.

Corless, Robert M., Gianni, Patrizia M., and Trager, Barry M. Approximate GCD, or, schönhage's algorithm revisited. Technical report, Ontario Research Centre for Computer Algebra, 2001a.

Corless, Robert M., Gianni, Patrizia M., Trager, Barry M., and Watt, Stephen M. The Singular Value Decomposition for polynomial systems. In Levelt, A. H. M., editor, *ISSAC '95, International Symposium on Symbolic and Algebraic Computation, Montreal, Canada*, pages 195–207, New York, 1995. ACM.

Corless, Robert M., Giesbrecht, Mark W., Jeffrey, David J., and Watt, Stephen M. Approximate polynomial decomposition. In Dooley [1999], pages 213–219.

Corless, Robert M., Giesbrecht, Mark W., Kotsireas, Ilias S., and Watt, Stephen M. Towards factoring bivariate approximate polynomials. In Mourrain [2001], pages 85–92.

Demmel, J. On condition numbers and the distance to the nearest ill-posed problem. *Numerische Mathematik*, 51:251–289, 1987.

Dooley, S., editor. *ISSAC 99 Proc. 1999 Internat. Symp. Symbolic Algebraic Comput.*, New York, N. Y., 1999. ACM Press.

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

Edelman, Alan, Elmroth, Erik, and Kågström, Bo. A geometric approach to perturbation theory of matrices and matrix pencils. part I: Versal deformations. *SIAM J. Matrix Anal. Appl.*, 18(3):653–692, 1997.

Emiris, Ioannis, Galligo, André, and Lombardi, Henri. Numerical univariate polynomial GCD. In Renegar, J., Shub, M., and Smale, S., editors, *Proc. 1995 AMS-SIAM Summer Seminar on the Mathematics of Numerical Analysis*, volume 32 of *Lectures in Applied Math*, pages 323–343, Park City, Utah, 1996.

Emiris, Ioannis, Galligo, André, and Lombardi, Henri. Certified approximate univariate GCD. *Journal of Pure and Applied Algebra*, 117 & 118:229–251, 1997.

Emiris, Ioannis Z. *Symbolic-numeric algebra for polynomials*, volume 39, pages 261–281. Marcel Dekker, New York, 1999.

Galligo, André, Gonzalez-Vega, Laureano, and Lombardi, Henri. Continuity properties for flat families of polynomials (I). *Journal of Pure and Applied Algebra*, page to appear, 2001.

Galligo, André and Rupprecht, David. Semi-numerical determination of irreducible branches of a reduced space curve. In Mourrain [2001], pages 137–142.

Galligo, André and Watt, Stephen M. A numerical absolute primality test for bivariate polynomials. In Küchlin [1997], pages 217–224.

Gantmacher, F. R. *The Theory of Matrices*, volume 2. Chelsea Publ. Co., New York, N. Y., 1960.

von zur Gathen, J. Functional decomposition of polynomials: the tame case. *Journal of Symbolic Computation*, 9(3):281–299, 1990.

von zur Gathen, J. and Weiss, J. Homogeneous bivariate decompositions. *J. Symbolic Computation*, 19:409–434, 1995.

Gianni, P., Seppälä, M., Silhol, R., and Trager, B. Riemann surfaces, plane algebraic curves and their period matrices. *Journal of Symbolic Computation*, 26 (6):789–803, 1998. Special issue on Symbolic Numeric Algebra for Polynomials S. M. Watt and H. J. Stetter, editors.

Gloor, O., editor. *ISSAC 98 Proc. 1998 Internat. Symp. Symbolic Algebraic Comput.*, New York, N. Y., 1998. ACM Press. ISBN 1-58113-002-3.

Hardy, G. H., Littlewood, J. E., and Polya, G. A. *Inequalities*. Cambridge University Press, 2nd edition, 1951.

Higham, Nicholas J. *Accuracy and Stability of Numerical Algorithms*. SIAM, 1996.

Hitz, M. A. and Kaltofen, E. Efficient algorithms for computing the nearest polynomial with constrained roots. In Gloor [1998], pages 236–243. ISBN 1-58113-002-3.

Hitz, M. A., Kaltofen, E., and Lakshman Y. N. Efficient algorithms for computing the nearest polynomial with a real root and related problems. In Dooley [1999], pages 205–212.

Huang, Yuzhen, Wu, Wenda, Stetter, Hans J., and Zhi, Lihong. Pseudofactors of multivariate polynomials. In Traverso, Carlo, editor, *Proceedings of the 2000 International Symposium on Symbolic and Algebraic Computation*, pages 161–168. ACM Press, August 2000.

Jeanerrod, C.-P. and Pflügel, E. A reduction algorithm for matrices depending on a parameter. In Dooley [1999], pages 121–128.

Kaltofen, E. Fast parallel absolute irreducibility testing. *Journal of Symbolic Computation*, 1(1):57–67, 1985. Misprint corrections: *J. Symbolic Comput.* vol. 9, p. 320 (1989).

Kaltofen, E. Polynomial factorization 1987-1991. In Simon, I., editor, *Proc. LATIN '92*, volume 583 of *Lect. Notes Comput. Sci.*, pages 294–313, Heidelberg/New York, 1992. Springer Verlag.

Karmarkar, N. K. and Lakshman Y. N. Approximate polynomial greatest common divisors and nearest singular polynomials. In Lakshman, Y. N., editor, *ISSAC '96, International Symposium on Symbolic and Algebraic Computation, Zurich, Switzerland*, pages 35–42, New York, 1995. ACM.

Kharitononv, V. L. Asymptotic stability of an equilibrium of a family of systems of linear differential equations. *Differential Equations*, 14:1483–1485, 1979.

Knuth, Donald E. *The Art of Computer Programming*, volume 2: Seminumerical Algorithms. Addison-Wesley, 2nd edition, 1981.

Kozen, D. and Landau, S. Polynomial decomposition algorithms. *Journal of Symbolic Computation*, 7:445–456, 1989.

Küchlin, W. W., editor. *ISSAC '97, International Symposium on Symbolic and Algebraic Computation, Maui, Hawaii*, New York, 1997. ACM.

Li, T. Y., Sauer, T., and Yorke, J. A. The cheater's homotopy. *SIAM J. Num. Anal.*, 26(5):1241–1251, 1989.

Manocha, D. and Demmel, J. Algorithms for intersecting parametric and algebraic curves II: multiple intersections. *Computer Vision, Graphics and Image Processing: Graphical Models and Image Processing*, pages 81–100, 1995.

Minnichelli, R. J., Anagnost, J. J., and Desoer, C. A. An elementary proof of Kharitonov's stability theorem with extensions. *IEEE Trans. Automatic Control*, 34(9):995–998, 1989.

Mourrain, B., editor. *ISSAC 2001 Proc. 2001 Internat. Symp. Symbolic Algebraic Comput.*, New York, N. Y., 2001. ACM Press.

Noda, M. T. and Sasaki, T. Approximate GCD and its application to ill-conditioned algebraic equations. *J. Comput. Appl. Math.*, 38:335–351, 1991.

Ochi, M., Noda, M. T., and Sasaki, T. Approximate greatest common divisor of multivariate polynomialsand its application to ill-conditioned system of algebraic equations. *J. Inf. Process.*, 12:292–300, 1991.

Oettli, W. and Prager, W. Compatibility of approximate solution of linear equations with given error bounds for coefficients and right-hand sides. *Numerische Mathematik*, 6:405–409, 1964. ISSN 0029-599X.

Pan, Victor Ya. Approximate polynomial GCDs, Pade approximation, polynomial zeros, and bipartite graphs. In *Proc. 9th ACM-SIAM Symposium on Discrete Algorithms*, pages 68–77, Philadelphia, 1998. SIAM Publications.

Poljak, S. and Rohn, J. Checking robust nonsingularity is NP-hard. *Math. Control Signals Systems*, 6:1–9, 1993.

Rupprecht, David. An algorithm for computing certified approximate GCD of n univariate polynomials. *Journal of Pure and Applied Algebra*, 139, 1999.

Rupprecht, David. Semi-numerical absolute factorization of polynomials with integer coefficients. *Journal of Symbolic Computation*, page to appear, 2001.

Sasaki, Tateaki. Approximate multivariate polynomial factorization based on zero-sum relations. In Mourrain [2001], pages 284–291.

Sasaki, Tateaki, Saito, Tomokatsu, and Hilano, Teruhiko. Analysis of approximate factorization algorithm i. *Japan J. Industrial and Applied Math*, 9(3): 351–368, October 1992.

Sasaki, Tateaki and Sasaki, Mutsuko. A unified method for multivariate polynomial factorization. *Japan J. Industrial and Applied Math*, 10(1):21–39, February 1993.

Sasaki, Tateaki, Suzuki, Masayuki, r, Miroslav Kolá¸, and Sasaki, Mutsuko. Approximate factorization of multivariate polynomials and absolute irreducibility testing. *Japan J. Industrial and Applied Math*, 8(3):357–375, October 1991.

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

Schönhage, A. Quasi-gcd computations. *Journal of Complexity*, 1:118–137, 1985.

Sommese, Andrew J., Verschelde, Jan, and Wampler, Charles W. Numerical decomposition of the solution sets of polynomial systems into irreducible components. *SIAM Journal of Numerical Analysis*, 38(6):2022–2046, 2001.

Stetter, Hans J. Matrix eigenproblems are at the heart of polynomial system solving. SIGSAM *Bulletin: Communications on Computer Algebra*, 30(4):22–25, December 1996.

Stetter, Hans J. The nearest polynomial with a given zero, and similar problems. SIGSAM BULLETIN*: Communications on Computer Algebra*, 33(4): 2–4, December 1999.

Turing, Alan M. Rounding errors in matrix processes. *Quarterly J. Mech. Appl. Math.*, 1:287–308, 1948.

Wilkinson, James H. *The Perfidious Polynomial*, pages 1–28. Mathematical Association of America, 1984.

Zhi, Lihong and Wu, Wenda. Nearest singular polynomials. *Journal of Symbolic Computation*, 26(6):667–675, 1998. Special issue on Symbolic Numeric Algebra for Polynomials S. M. Watt and H. J. Stetter, editors.