THEORY OF OPTIMAL ALGORITHMS

J. F. Traub

Department of Computer Science
Carnegie-Mellon University
Pittsburgh, Pa.
June 1973

Based on an invited paper presented at a Conference on Software for Numerical Mathematics, Loughborough, England.

This research was supported in part by the National Science Foundation under Grant GJ32111 and the Office of Naval Research under Contract N0014-67-A-0314-0010, NR 044-422.

1. INTRODUCTION

Recent progress in the theory of optimal algorithms has led to new algorithms as well as theoretical bounds on the efficiency of any possible algorithm.

I believe that historically there have been three major stages in the development of algorithmic analysis. They are:

1. Synthesis of an algorithm
2. Analysis of an algorithm
3. Analysis of a class of algorithms.

Initially the emphasis was on the synthesis of an algorithm. Around 1947 people started analyzing particular algorithms very carefully, which was the second stage. Within the last 10-15 years people have been looking at classes of algorithms and trying to find the best. This trend has recently accelerated and there is now tremendous interest in analyzing classes of algorithms, that is, in computational complexity.

There are many reasons for studying computational complexity. Those I consider most important are:

1. Constructing "good" new algorithms.
2. Filtering out "bad" algorithms.
3. Creating a theory of algorithms which will establish theoretical limits on computation.

To discuss optimal algorithms we need a measure of cost. The measure I'll use throughout this paper is the total number of arithmetic operations, +, -, ×, ÷. Gentleman [73] and Reddy [73] have discussed some of the other

components of the cost which might be included. Other properties of a numerical algorithm, such as stability and domain of convergence, are critical. Measures of cost deserve more refinement.

2. ALGEBRAIC AND ANALYTIC COMPUTATIONAL COMPLEXITY

I want to distinguish between two types of algorithms. The dichotomy depends on the nature of the underlying mathematical problem. A mathematical problem can be finite or infinite. Examples of finite problems are matrix multiplication and polynomial evaluation. Examples of infinite problems are the solution of an elliptic partial differential equation and the calculation of a polynomial zero.

I will refer to optimality theory for finite problems as algebraic computational complexity. Optimality theory for infinite problems I will refer to as analytic computational complexity. I give some examples of work from each domain.

3. RECENT RESULTS IN ALGEBRAIC COMPUTATIONAL COMPLEXITY

Borodin [73] gives a survey of the enormous recent activity in algebraic complexity. I will confine myself to some very recent results which deal with one set of related problems.

The problems are:

1. Polynomial multiplication. Given two polynomials of degree n, to find the product polynomial.

2. Polynomial division. Given two polynomials of degree n and $\frac{1}{2}n$, to find their quotient and remainder. More generally we divide a polynomial of degree n by a polynomial of degree m. The choice of $m = \frac{1}{2}n$ makes the "size" of the problem depend on just one parameter.

3. Polynomial interpolation. Given (x_i, y_i), $i = 0,1,\ldots,n$. Find $P(t)$ such that $P(x_i) = y_i$.

4. Evaluation of a polynomial at many points. Evaluate an nth degree polynomial at n+1 points given simultaneously.

5. Evaluation of a polynomial and all its derivatives. Evaluate an nth degree polynomial and all its derivatives at one point.

These problems take $O(n^2)$ operations classically. Using "fast" algorithms the first two problems can be done in $O(n \log n)$ operations while the next three problems can be done in $O(n \log^2 n)$ operations. Fast polynomial multiplication is done with the Fast Fourier Transform. Other fast algorithms are due to Moenck and Borodin [72], Strassen [72], and Kung [73]. Borodin [73] summarizes the state of the art in fast algorithms.

The results above are asymptotic. They are only significant for rather large values of n. For example n^2 is smaller than $n \log^2 n$ until n is somewhat greater than 30. (All logarithms are to base 2.) Furthermore, analyses ignore asymptotic constants which can prove significant if n is not too large (Borodin [73]).

The following is an example of a new algorithm which is better than the best previously known algorithm, not just asymptotically, but for all n. Given

$$P(t) = \sum_{j=0}^{n} a_{n-j}\, t^j,$$

and a number x, the problem is to calculate the normalized derivatives $\frac{P^{(j)}(x)}{j!}$, $j = 0,\ldots,n$. The standard algorithm is some 150 years old and appears in most numerical methods texts. It is known as the iterated Horner rule or a synthetic division. This algorithm can be written as

$$T_i^{-1} = a_{i+1},\ i = 0,1,\ldots,n-1,$$

$$T_j^j = a_0,\ j = 0,1,\ldots,n,$$

$$T_i^j = T_{i-1}^{j-1} + xT_{i-1}^j,\ j = 0,1,\ldots,n-1,\ i = j+1,\ldots,n.$$

It is not difficult to verify that

$$\frac{P^{(j)}(x)}{j!} = T_n^j,\ j = 0,1,\ldots,n.$$

Observe that the first two lines of the algorithm define initial conditions. All the work is done in the recursion of the last line. The recursion is done $\frac{1}{2}n(n+1)$ times and there is one addition and one multiplication per step. Thus the iterated Horner algorithm requires $\frac{1}{2}n(n+1)$ multiplications and $\frac{1}{2}n(n+1)$ additions.

Consider now the following algorithm.

$$T_i^{-1} = a_{i+1}x^{n-i-1},\ i = 0,1,\ldots,n-1,$$

$$T_j^j = a_0x^n,\ j = 0,1,\ldots,n, \tag{3.1}$$

$$T_i^i = T_{i-1}^{j-1} + T_{i-1}^j,\ j = 0,1,\ldots,n-1,\ i = j+1,\ldots,n.$$

It may be shown (Shaw and Traub [72]) that

$$\frac{P^{(j)}(x)}{j!} = x^{-j}T_n^j,\ j = 0,1,\ldots,n-1.$$

In this algorithm all the multiplications are done as part of the initial conditions. The recursion involves additions only. The normalized derivatives are obtained by division using the x^j calculated as part of the initialization.

Thus this algorithm, which is just as simple as the iterated Horner rule, yields the normalized derivatives in 3n-2 multiplications and divisions and $\frac{1}{2}n(n+1)$ additions. The algorithm is of practical utility. It is also of theoretical interest since it demonstrates that only a <u>linear</u> number of multiplications and divisions are needed.

The problem posed here is a special case of the problem of calculating m derivatives of an nth degree polynomial. The algorithm presented above is a member of a one-parameter family of algorithms (Shaw and Traub [72]). The optimal choice of the parameter as a function of m and n is discussed by Shaw and Traub [73]. Stability of these algorithms is established by Wozniakowski [73].

4. AN EFFICIENCY MEASURE

The remainder of this paper deals with analytic computational complexity. Recent research includes the complexity of elliptic partial differential equations (Schultz [73]) and the complexity of systems of non-linear equations (Brent [72]). A more extensive bibliography may be found in Traub [72].

I confine myself here to the problem of calculating a real simple zero α of a real function f. This zero-finding problem may seem rather specialized, but it is equivalent to the fixed-point problem, a ubiquitous problem in mathematics and applied mathematics. It may be formulated in an abstract setting and covers partial differential equations, integral equations, and many other important problems, Traub [72] and Kung and Traub [73a, 73b] may be consulted for the results reported in the rest of this paper and for proofs of the theorems.

Consider iteration algorithms for approximating α. Let the x_i be generated by an iteration function φ,

$$x_{i+1} = \varphi(x_i)$$

To define an efficiency measure for φ we need measures of goodness and cost. As the measure of goodness we use the order p defined as follows. If

$$\lim_{x_i \to \alpha} \frac{\varphi(x_i)-\alpha}{(x_i-\alpha)^p} = S \neq 0$$

then $p = p(\varphi)$ is the order of convergence.

The cost consists of two parts: the evaluation cost and the combinatory cost. Let φ use v_i evaluations of $f^{(i)}$. If $f^{(i)}$ is rational, let $c(f^{(i)})$

denote the number of arithmetic operations for one evaluation of $f^{(i)}$; otherwise let $c(f^{(i)})$ denote the number of arithmetic operations used in the rational subroutine which approximates $f^{(i)}$. Then

$$\text{Evaluation cost} = \sum_{i\geq 0} v_i \, c(f^{(i)}).$$

Let $a(\varphi)$ be the minimum number of arithmetic operations to combine the $f^{(i)}$ to form φ by any procedure λ. Then

$$\text{Combinatory cost} = a(\varphi).$$

Finally, the cost of performing one iteration step is

$$\sum_{i\geq 0} v_i \, c(f^{(i)}) + a(\varphi).$$

We define the efficiency $e(\varphi,f)$ of the iteration φ with respect to the problem f by

$$(4.1) \quad e(\varphi,f) = \frac{\log p(\varphi)}{\sum_{i\geq 0} v_i \, c(f^{(i)}) + a(\varphi)}.$$

A discussion of this efficiency measure, including its relation to other efficiency measures, is given by Kung and Traub [73b]. Here I will only point out that earlier measures (Traub [72]) did not include the combinatory cost $a(\varphi)$ and that inclusion of combinatory cost is crucial.

The efficiency measure has the following two properties:

1. It is invariant under composition.
2. It is inversely proportional to total cost.

The first property can be written as

$$e(\varphi\cdot\varphi,f) = e(\varphi,f),$$

where $\varphi\cdot\varphi$ denotes performing the iteration φ twice. This says that a sequence and a subsequence have the same efficiency. The second property is stated more precisely as follows. Let φ_1, φ_2 be two iterations used to approximate α to within a certain accuracy. Let the total cost of φ_j be W_j. Then

$$\frac{e(\varphi_1,f)}{e(\varphi_2,f)} \sim \frac{W_2}{W_1}.$$

Let

$$c_f = \min_{i\geq 0} c(f^{(i)}).$$

In this paper, we refer to c_f as the problem complexity. Let

$$v(\varphi) = \sum_{i\geq 0} v_i(\varphi).$$

Clearly, $v(\varphi)$ is the total number of evaluations used in φ. Then by (4.1),

$$(4.2)\quad e(\varphi,f) \leq \frac{\log p(\varphi)}{v(\varphi)\, c_f + a(\varphi)}.$$

This will be useful for obtaining upper bounds for $e(\varphi,f)$.

The optimal efficiency depends on the family Φ to which φ belongs. Our classification for Φ depends on the information required by Φ. We can distinguish between iterations with or without memory. We restrict ourselves here to iterations without memory. That is, the new iterate x_{i+1} is computed using information only at the current iterate x_i. For iterations without

memory we distinguish between one-point iteration and multipoint iteration. Roughly speaking, if f or its derivatives require evaluation at k points in order to generate a new iterate by the iteration φ, then φ is a k-point iteration. In particular, if k = 1 we call φ a one-point iteration and if k > 1 and the value of k is not important we call φ a multipoint iteration. This terminology was introduced by Traub [64]. Precise definitions are given by Kung and Traub [73a].

The following two examples illustrate the definitions.

Example 4.1. (Newton-Raphson Iteration)

$$\varphi(f)(x) = x - \frac{f(x)}{f'(x)} .$$

This is a one-point iteration with $p(\varphi) = 2$, $v_0(\varphi) = v_1(\varphi) = 1$, and $a(\varphi) = 2$. Hence

$$e(\varphi,f) = \frac{1}{c(f)+c(f')+2},$$

$$e(\varphi,f) \leq \frac{1}{2c_f+2} .$$

Example 4.2.

$$z_0 = x,$$

$$z_1 = z_0 - \frac{f(z_0)}{f'(z_0)},$$

$$\varphi(f)(x) = z_1 - \frac{f(z_1)f(z_0)}{[f(z_1)-f(z_0)]^2} \cdot \frac{f(z_0)}{f'(z_0)} .$$

This is a two-point iteration with $p(\varphi) = 4$, $v_0(\varphi) = 2$, $v_1(\varphi) = 1$ and $a(\varphi) = 8$.

Hence

$$e(\varphi,f) = \frac{2}{2c(f)+c(f')+8} ,$$

$$e(\varphi,f) \leq \frac{2}{3c_f+8} .$$

Given an algorithm φ and a problem f, we can use $e(\varphi,f)$ as defined by (4.1) to calculate efficiency. We are also interested in the optimal efficiency of a class of algorithms. This motivates the following definitions.

It is natural to ask for a given problem f what is the optimal value of $e(\varphi,f)$ for all φ belonging to some family Φ. Define

$$E_n(\Phi,f) = \sup_{\varphi\in\Phi}\{e(\varphi,f) \mid v(\varphi) = n\}.$$

Thus $E_n(\Phi,f)$ is the optimal efficiency over all $\varphi \in \Phi$ which use n evaluations. Define

$$E(\Phi,f) = \sup\{E_n(\Phi,f) \mid n = 1,2,\ldots\}.$$

Thus $E(\Phi,f)$ is the optimal efficiency for all $\varphi \in \Phi$. We will establish lower and upper bounds for $E_n(\Phi,f)$ and $E(\Phi,f)$ with respect to different families of iterations. When there is no ambiguity, we write $E_n(\Phi,f)$ and $E(\Phi,f)$ as $E_n(f)$ and $E(f)$, respectively. Since in practice we are more concerned with efficiency for problems f with higher complexity, we are particularly interested in the asymptotic behavior of these bounds as $c_f \to \infty$.

5. EFFICIENCY OF ONE-POINT ITERATION

The iterations most used in practice are one-point iterations. We derive lower and upper bounds on the efficiency of any one-point iteration.

We consider a particular family of one-point iterations $\{\gamma_n\}$. The first three members of this family are given by

$$\gamma_1 = x$$

$$\gamma_2 = \gamma_1 - \frac{f(x)}{f'(x)}$$

$$\gamma_3 = \gamma_2 - \frac{f''(x)}{f'(x)}\left[\frac{f(x)}{f'(x)}\right]^2 .$$

The family γ_n has been thoroughly studied (Traub [64], Section 5.1). Its important properties from our point of view are summarized in the following

Theorem 5.1

1. $v_i(\gamma_n) = 1$, $i = 0,1,\ldots,n-1$, $v_i(\gamma_n) = 0$, $i > n-1$. Hence $v(\gamma_n) = n$.

2. $p(\gamma_n) = n$.

It can be shown (Kung and Traub [73b]) that

$$a(\gamma_n) \leq \rho\, n^2 \log n \tag{5.1}$$

for some positive constant ρ. By (5.1) and Theorem 5.1,

$$e(\gamma_n, f) \geq \frac{\log n}{\sum_{i\geq 0}^{n-1} c(f^{(i)}) + \rho n^2 \log n} \tag{5.2}$$

For n small, $a(\gamma_n)$ can be calculated by inspection. Thus $a(\gamma_3) = 7$ and

$$(5.3) \quad e(\gamma_3,f) = \frac{\log 3}{c(f)+c(f')+c(f'')+7} .$$

I now turn to general one-point iterations. Let φ be any one-point iteration, with $v(\varphi) = n$, which satisfies a mild smoothness condition. Then by Traub [64, Section 5.4], Kung and Traub [73b, Theorem 6.1], $v_i(\varphi) \geq 1$, $i = 0,\ldots,p(\varphi)-1$ and hence $p(\varphi) \leq n$. Since at least n-1 arithmetic operations are needed to combine n evaluations of f and its derivatives, $a(\gamma_n) \geq n-1$. Hence, from (4.2),

$$(5.4) \quad e(\varphi,f) \leq \frac{\log n}{nc_f+n-1} = h(n).$$

It may be verified that

$$(5.5) \quad h(n) \leq h(3) = \frac{\log 3}{3c_f+2}, \text{ for all n, for } c_f > 4.$$

Since it is important to solve "difficult" problems efficiently, the condition $c_f > 4$ is not restrictive. Since

$$h(2) = \frac{\log 2}{2c_f+1} = \frac{1}{2c_f+1}$$

and $\frac{1}{3} \log 3 \doteq .52$, there is little difference between the bounds on the second and third order iteration.

One of the pieces of folk wisdom of numerical mathematics is that for most problems it is better to use a fairly low order method more often than to use a higher order method less often. The above result gives a theoretical

justification in the case of one-point iterations. We shall see this does not hold for multipoint iterations.

From (5.2), (5.3), (5.4), (5.5) we obtain the theorem giving lower and upper bounds on the efficiency of one-point iterations.

Theorem 5.2

For the family Φ of one-point iterations,

$$(5.6)\quad \frac{\log n}{\sum_{i=0}^{n-1} c(f^{(i)})+\rho n^2 \log n} \le E_n(f) \le \frac{\log n}{nc_f+n-1}, \text{ for a constant } \rho > 0, \forall n,$$

$$(5.7)\quad \frac{\log 3}{c(f)+c(f')+c(f'')+7} \le E(f) \le \frac{\log 3}{3c_f+2}, \text{ for } c_f > 4.$$

6. EFFICIENCY OF MULTIPOINT ITERATION

In the previous section it was shown that the order of a one-point iteration is at most linear in the number of evaluations it requires. This restriction does not apply for multipoint iterations. Furthermore a one-point iteration of order p requires the evaluation of at least the first p-1 derivatives of f. This restriction also does not apply to multipoint iterations. A high order multipoint iteration can be constructed that requires no derivative evaluations at all.

To illustrate these points we consider the family of iterations $\{\Psi_n\}$ defined by Kung and Traub [73a, Section 4]. The important properties of $\{\Psi_n\}$ from our point of view are summarized in

Theorem 6.1

1. $v_0(\Psi_n) = n.\quad v_i(\Psi_n) = 0,\ i > 0.$
   Hence $v(\Psi_n) = n.$

2. $p(\Psi_n) = 2^{n-1}.$

Thus Ψ_n requires just n evaluations of f, and no derivative evaluations, and is of order 2^{n-1}. In particular

$$v(\Psi_4) = 4,\ p(\Psi_4) = 8.$$

The best previously known result for four evaluations was order five.

Kung and Traub [73a, Appendix I] give a procedure for computing $\Psi_n(x)$ in $\frac{3}{2}n^2 + \frac{3}{2}n-7$ arithemtic operations. Hence

$$a(\Psi_n) \leq \frac{3}{2}n^2 + \frac{3}{2}n-7.$$

More generally, we assume that

(6.1) $a(\Psi_n) \le r(n)$,

where $r(n) = r_2 n^2 + r_1 n + r_0$, $r_2 > 0$.

Then by (6.1) and Theorem 6.1,

(6.2) $e(\Psi_n, f) \ge \frac{n-1}{nc(f)+r(n)}$.

We choose n so as to maximize the righthand side of (6.2). The maximum is achieved when n=t where

$$t = 1 + \sqrt{\frac{c(f)}{r_2} + \delta}, \quad \delta = \frac{r_0+r_1+r_2}{r_2} .$$

Let

(6.3) $M = \text{round}(t)$.

Then from (6.2) we can easily prove

Theorem 6.2

There exists a constant $\zeta < 0$ such that if $M = M(f)$ is chosen by (6.3) then

$$e(\Psi_M, f) \ge \frac{1}{c(f)}\left(1 + \frac{\zeta}{\sqrt{c(f)}}\right), \text{ for } c(f) \text{ large.}$$

From (6.2) and Theorem 6.2, we have

Corollary 6.1

For the family Φ of one-point or multipoint iterations,

$E_n(f) \ge \frac{n-1}{nc(f)+r(n)}$, where $r(n) = r_2 n^2 + r_1 n + r_0$, $r_2 > 0$; and

$E(f) \ge \frac{1}{c(f)}\left(1 + \frac{\zeta}{\sqrt{c(f)}}\right)$, for a constant $\zeta < 0$, for $c(f)$ large.

Can still higher order be achieved with n evaluations of f? An upper bound is provided by the following theorem proven by Kung and Traub [73a, Theorem 7.2].

Theorem 6.3

Let φ be a multipoint iteration with $v_0(\varphi) = n$, $v_i(\varphi) = 0$, $i > 0$. Then $p(\varphi) \le 2^n$.

As the conjecture at the end of this paper shows, we don't believe that the bound of 2^n can be achieved. Since $a(\varphi) \ge n-1$,

$$(6.4) \quad e(\varphi,f) \le \frac{n}{nc(f)+n-1} \le \frac{1}{c(f)} .$$

Since Ψ_n is a multipoint iteration which uses evaluations of f only, from (6.4) and Corollary 6.1, we have

Theorem 6.4

For the family of Φ of multipoint iterations using values of f only,

$$\frac{n-1}{nc(f)+r(n)} \le E_n(f) \le \frac{n}{nc(f)+n-1}, \ \forall n,$$

$$\frac{1}{c(f)}\left(1 + \frac{\zeta}{\sqrt{c(f)}}\right) \le E(f) \le \frac{1}{c(f)} ,$$

for c(f) large, where $r(n) = r_2 n^2 + r_1 n + r_0$, $r_2 > 0$, and $\zeta < 0$.

We can now give a lower bound on the ratio of the optimal efficiency of multipoint iteration to the optimal efficiency of one-point iteration.

For a given problem f let $E'(f)$, $E''(f)$ be the optimal efficiency achievable by one-point iteration and multipoint iteration, respectively. By

Theorem 5.2 and Corollary 6.1,

$$E'(f) \leq \frac{\log 3}{3c_f+2},$$

$$E''(f) \geq \frac{1}{c(f)}\left[1 + \frac{\zeta}{\sqrt{c(f)}}\right], \ \zeta < 0, \text{ for } c(f) \text{ large.}$$

Hence

$$\frac{E''(f)}{E'(f)} \geq \frac{3c_f+2}{(\log 3)c(f)}\left[1 + \frac{\zeta}{\sqrt{c(f)}}\right] \sim \frac{3}{\log 3} \cdot \frac{c_f}{c(f)}, \text{ for } c(f) \text{ large.}$$

In particular, if f is a problem such that $c_f = c(f)$ and c_f is large, then the ratio between optimal efficiencies achievable by multipoint iteration and one-point iteration is at least $3/\log 3 \sim 1.89$.

7. TWO CONJECTURES

Section 6 showed that an iteration of order 2^{n-1} can be constructed using n evaluations of f. Kung and Traub [73a] conjecture that this order is optimal for any iteration (without memory) using n evaluations of f and its derivatives.

Conjecture 7.1

For any one-point or multipoint iteration φ with $v(\varphi) = n$, $p(\varphi) \leq 2^{n-1}$.

The conjecture is very general. φ may use any n values of f or its derivatives evaluated at any points. This conjecture is one of the major open questions in analytic complexity.

If Conjecture 7.1 is true, it implies the truth of the following conjecture (Kung and Traub [73b]).

Conjecture 7.2

For the family Φ of one-point or multipoint iterations,

$$E_n(f) \leq \frac{n-1}{nc_f+n-1},$$

$$E(f) \leq \frac{1}{c_f+1}.$$

This conjecture states, essentially, that the optimal efficiency for solving the problem f with respect to all one-point or multipoint iterations is bounded by the reciprocal of the problem complexity.

ACKNOWLEDGMENTS

I would like to thank H. T. Kung, Pamela McCorduck, and Mary Shaw for their comments on this paper.

BIBLIOGRAPHY

Borodin [73] — Borodin, A., "On the Number of Arithmetics Required to Compute Certain Functions - Circa May 1973." Appears in Complexity of Sequential and Parallel Numerical Algorithms, edited by J. F. Traub, Academic Press, 1973.

Gentleman [73] — Gentleman, W. M., "On the Relevance of Various Cost Models of Complexity." Appears in Complexity of Sequential and Parallel Numerical Algorithms, edited by J. F. Traub, Academic Press, 1973.

Kung [73] — Kung, H. T., Fast Evaluation and Interpolation. Report, Department of Computer Science, Carnegie-Mellon University, 1973.

Kung and Traub [73a] — Kung, H. T. and Traub, J. F., Optimal Order of One-Point and Multipoint Iteration. Report, Department of Computer Science, Carnegie-Mellon University, 1973.

Kung and Traub [73b] — Kung, H. T. and Traub, J. F.,"Computational Complexity of One-Point and Multipoint Iteration." To appear in Complexity of Real Computation, edited by R. Karp, American Mathematical Society, 1973.

Moenck and Borodin [72] — Moenck, R. and Borodin, A., "Fast Modular Transformations via Division," Proceedings IEEE SWAT Conference, 1972.

Reddy [73] — Reddy, D. R., "Some Numerical Problems in Artificial Intelligence." Appears in Complexity of Sequential and Parallel Numerical Algorithms, edited by J. F. Traub, Academic Press, 1973.

Shaw and Traub [72] — Shaw, M. and Traub, J. F., On the Number of Multiplications for the Evaluation of a Polynomial and Some of its Derivatives. Report, Computer Science Department, Carnegie-Mellon University. To appear in JACM.

Shaw and Traub [73] — Shaw, M. and Traub, J. F., "The Analysis of a Family of Algorithms for the Evaluation of a Polynomial and its Derivatives." To appear.

Strassen [73] — Strassen, V., "Die Berechnungskomplexität von Elementarsymmetrischen Funktionen und von Interpolationskoeffizienten," Numer. Math. 20 (1973), pp. 238-251.

Traub [64] Traub, J. F., Iterative Methods for the Solution of Equations, Prentice-Hall, 1964.

Traub [72] Traub, J. F., "Computational Complexity of Iterative Processes," SIAM J. Comp., Vol. 1 (1972), pp. 167-179.

Traub [72] Traub, J. F., editor, Complexity of Sequential and Parallel Numerical Algorithms, Academic Press, 1973.

Wozniakowski [73] Wozniakowski, H., "Rounding Error Analysis for the Evaluation of a Polynomial and Some of its Derivatives." To appear.