# Computational Differentiation and Algebraic Complexity Theory*

*Erich Kaltofen*

Department of Computer Science, Rensselaer Polytechnic Institute
Troy, NY 12180-3590; kaltofen@cs.rpi.edu

### Summary

The reverse mode of automatic differentiation allows within a constant cost factor the computation of the gradient of a multivariate, single-valued, function that is given by an arithmetic circuit. Indeed, a circuit can be constructed whose number of nodes does not exceed 4 times the number of nodes in the original circuit. Furthermore, it can be arranged that the depth of the circuit is within a constant of the original circuit as well (Kaltofen and Singer 1991, Kaltofen and Pan 1991). Griewank (1991) has also shown that the sequential space complexity can be kept within a logarithmic factor while increasing the time complexity by only a logarithmic factor. This result has been used for several algebraic complexity estimates:

1. Baur and Strassen (1983) show that the complexity of computing the determinant of an arbitrary non-singular matrix is asymptotically no less than that of the inverse, because for a square matrix $A$ we have

$$(-1)^{i+j}\frac{\partial\,\mathrm{Det}(A)}{\partial A_{j,i}} = \mathrm{Det}(A)\ (A^{-1})_{i,j}.$$

The recent so-called processor efficient parallel algorithms of poly-logarithmic time for computing the inverse of a non-singular matrix are based on this reduction (Kaltofen and Pan 1991 and 1992). Automatic differentiation is the only way known to me to compute inverses within the given time and processor count constraints.

2. Furthermore, Baur and Strassen employ the gradient contruction to show that the complexity of computing the sum $x_1^n + \cdots + x_n^n$ is within a constant of computing the individual $(n+1)^{\mathrm{st}}$ powers $x_1^{n+1}, \ldots, x_n^{n+1}$ which by the Strassen's (1973) degree bound is $\Theta(n \log n)$.

3. The transposition principle asserts that for any (possibly structured) matrix $A$ and any vector $b$ the problems of computing $A \cdot b$ and $A^{\mathrm{tr}} \cdot b$ are of the same asymptotic complexity. Proven explicitly in Kaminski, Kirkpatrick, and Bshouty (1988) by reversing the flow in the circuit for computing $A \cdot b$, the principle is also a simple consequence of reverse mode: for

$$f(x_1, \ldots, x_n) = ([\, x_1 \quad \ldots \quad x_n \,] \cdot A^{\mathrm{tr}}) \cdot b \qquad \text{we have} \qquad \begin{bmatrix} \partial_{x_1} f \\ \vdots \\ \partial_{x_n} f \end{bmatrix} = A^{\mathrm{tr}} b.$$

* Appears in Workshop Report on First Theory Institute on Computational Differentiation, C. H. Bischof and A. Griewank and P. M. Khademi editors, Tech. Rep. ANL/MCS-TM-183, Argonne National Laboratory, pp. 28–30, December 1993.

One application is when $A = V^{\mathrm{tr}}$ is a transposed Vandermonde matrix, a problem needed in sparse polynomial interpolation (Canny et al. 1989) and polynomial factoring (Shoup 1991). Shoup's explicit algorithm, however, is of linear space complexity and needs no divisions, unlike the one obtained from the fast multipoint polynomial evaluation problem $V \cdot b$ (see Aho et al. 1974, §6) and the transposition principle. Shoup (1993) also uses this principle in the construction of a fast method for computing the minimum polynomial of an element in an algebraic number field.

4. Similarly, the problems $A^{-1} \cdot b$ and $(A^{\mathrm{tr}})^{-1} \cdot b$ have the same asymptotic complexity. But again the explicitly derived algorithm for the Vandermonde case $(V^{\mathrm{tr}})^{-1} \cdot b$ by Kaltofen and Lakshman (1988) has the better linear space complexity.

As it turns out, higher derivatives are much more complex to compute. The following clever reduction of the product of two $n \times n$ matrices $B$ and $C$ to the trace of the Hessian has been communicated to me by T. Lickteig. Let $A$ be a third $n \times n$ matrix. Then

$$\mathrm{Trace}(ABC) = \left(\frac{\partial^2}{\partial x_1^2} + \cdots + \frac{\partial^2}{\partial x_n^2}\right)(\mathbf{x}^{\mathrm{tr}} ABC\mathbf{x}), \quad \text{where} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix};$$

note that the argument to the trace of the Hessian, $\mathbf{x}^{\mathrm{tr}}ABC\mathbf{x}$, can be computed in $O(n^2)$ time. However, by reverse mode, we can compute

$$\frac{\partial\, \mathrm{Trace}(ABC)}{\partial A_{i,j}} = (BC)_{j,i}$$

Therefore, any method for finding the trace of the Hessian within a factor $g(n)$ gives a matrix multiplication algorithm of $O(g(n)\, n^2)$ arithmetic steps.

Finally, computing multiple partial derivatives is known to be as hard as counting the number of satisfying assignments in a Boolean formula (Valiant 1982): Consider

$$P(x_1, \ldots, x_n, Z_{1,1}, \ldots, Z_{n,n}) = \prod_{i=1}^{n} \left( \sum_{j=1}^{n} x_j Z_{i,j} \right)$$

Then

$$\frac{\partial^n P}{\partial x_1 \cdots \partial x_n} = \mathrm{Permanent}(Z).$$

Therefore, given a transformation that computes $\partial^n/(\partial x_1 \cdots \partial x_n)$ within a factor of $h(n)$ leads to an algorithm to compute the permanent with $O(h(n)\, n^2)$ arithmetic operation. By Valiant's (1979) proof that the permanent is $\#\mathcal{P}$-complete, $h(n)$ is likely exponential in $n$.

## Literature Cited

Aho, A., Hopcroft, J., and Ullman, J., *The Design and Analysis of Algorithms*; Addison and Wesley, Reading, MA, 1974.

Baur, W. and Strassen, V., "The complexity of partial derivatives," *Theoretical Comp. Sci.* **22**, pp. 317–330 (1983).

Canny, J., Kaltofen, E., and Lakshman Yagati, "Solving systems of non-linear polynomial equations faster," *Proc. ACM-SIGSAM 1989 Internat. Symp. Symbolic Algebraic Comput.*, pp. 121–128 (1989).

Griewank, A., “Achieving logarithmic growth of temporal and spatial complexity in reverse automatic differentiation,” *Preprint* **MCS-P228-0491**, Argonne National Lab., Math. and Comput. Sci. Div., Argonne, Illinois, May 1991.

Kaltofen, E. and Lakshman Yagati, “Improved sparse multivariate polynomial interpolation algorithms,” *Proc. ISSAC ’88, Springer Lect. Notes Comput. Sci.* **358**, pp. 467–474 (1988).

Kaltofen, E. and Pan, V., “Processor efficient parallel solution of linear systems over an abstract field,” in *Proc. 3rd Ann. ACM Symp. Parallel Algor. Architecture*; ACM Press, pp. 180–191, 1991.

Kaltofen, E. and Pan, V., “Processor-efficient parallel solution of linear systems II: the positive characteristic and singular cases,” *Proc. 33rd Annual Symp. Foundations of Comp. Sci.*, pp. 714–723 (1992).

Kaltofen, E. and Singer, M. F., “Size efficient parallel algebraic circuits for partial derivatives,” in *IV International Conference on Computer Algebra in Physical Research*, edited by D. V. Shirkov, V. A. Rostovtsev, and V. P. Gerdt; World Scientific Publ., Singapore, pp. 133–145, 1991.

Kaminski, M., Kirkpatrick, D. G., and Bshouty, N. H., “Addition requirements for matrix and transposed matrix products,” *J. Algorithms* **9**, pp. 354–364 (1988).

Shoup, V., “A fast deterministic algorithm for factoring polynomials over finite fields of small characteristic,” in *Proc. 1991 Internat. Symp. Symbolic Algebraic Comput.*, edited by S. M. Watt; ACM Press, pp. 14–21, 1991.

Shoup, V., “Fast construction of irreducible polynomials over finite fields,” in *Proc. 4th Annual ACM-SIAM Symp. on Discrete Algor.*; ACM and SIAM, New York, N.Y., and Philadelphia, PA, pp. 484–492, 1993.

Strassen, V., “Die Berechnungskomplexität von elementarsymmetrischen Funktionen und von Interpolationskoeffizienten,” *Numer. Math.* **20**, pp. 238–251 (1973). In German.

Valiant, L., “The complexity of computing the permanent,” *Theoretical Comp. Sci.* **8**, pp. 189–201 (1979).

Valiant, L., “Reducibility by algebraic projections,” *L’Enseignement mathématique* **28**, pp. 253–268 (1982).