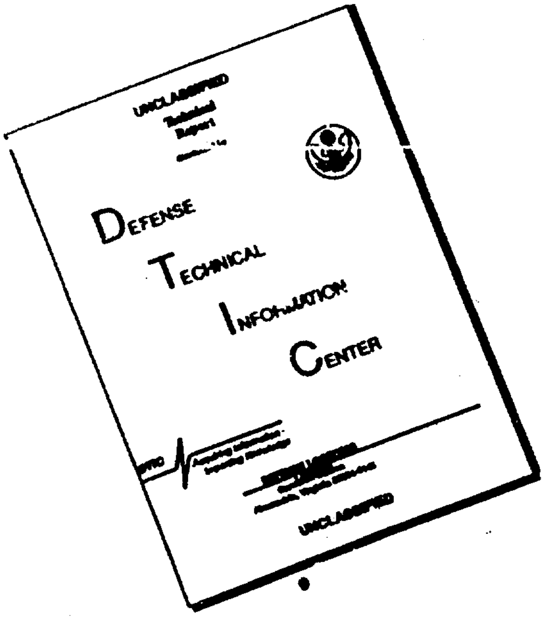
DISCLAIMER NOTICE



UNCLASSIFIED

Technical

Report

DEFENSE

TECHNICAL

INFORMATION

CENTER

DTIC

UNCLASSIFIED

THIS DOCUMENT IS BEST QUALITY AVAILABLE. THE COPY FURNISHED TO DTIC CONTAINED A SIGNIFICANT NUMBER OF PAGES WHICH DO NOT REPRODUCE LEGIBLY.

ACCELERATING LP ALGORITHMS

B. L. Fox[*]

The RAND Corporation, Santa Monica, California

Winograd ~~[3]~~ has given a new method for computing inner products. Under certain circumstances, when a series of inner products must be calculated, using Winograd's scheme is more efficient than the standard (naive) method. Winograd points out that his method does matrix multiplication up to twice as fast as the usual scheme and notes similar acceleration for matrix inversion and the solution of linear equations.

Here we point ~~out~~ how Winograd's method can speed up linear programming ~~(LP)~~ algorithms, in particular the revised simplex method, (see, e.g., Simonnard [2]).

Winograd's algorithm:

Let $x = (x_n \dots, x_n)$, $y = (y_1, \dots, y_n)$,

$$\xi = \sum_{j=1}^{[n/2]} x_{2j-1} x_{2j}$$

$$\eta = \sum_{j=1}^{[n/2]} y_{2j-1} y_{2j}$$

[*] Any views expressed in this paper are those of the author. They should not be interpreted as reflecting the views of The RAND Corporation or the official opinion or policy of any of its governmental or private research sponsors. Papers are reproduced by The RAND Corporation as a courtesy to members of its staff.

$$\gamma = \sum_{j=1}^{[n/2]} (x_{2j-1} + y_{2j})(x_{2j} + y_{2j-1}),$$

where [t] denotes the integer part of t. The inner product (x,y) is then given by

$$(x,y) = \begin{cases} \gamma - \xi - \eta, \text{ if n is even} \\ \gamma - \xi - \eta + x_n y_n, \text{ if n is odd.} \end{cases}$$

In the sequel, we denote (x,y) simply by xy.

Pricing out in LPs:

At a given iteration, let π be the vector of simplex multipliers and let J be the set of nonbasic columns in the constraint matrix. A majcr task is to determine which column should enter the basis. Let a^j stand for a generic column. To price out, we need to compute πa^j, $j \in J$. Assuming that there are m constraints and k columns in J, the conventional procedure requires km multiplications and k(m - 1) additions.

Now we apply Winograd's method. For simplicity, suppose that m is even. At the first iteration <u>only</u>, compute

$$\lambda_j = \sum_{i=1}^{m/2} a^j_{2i-1} \, a^j_{2i}$$

for all j, basic or not. Of course, if a^j is a unit vector, $\lambda_j = 0$. (The initial basis generally consists of unit vectors.) Since the λ_j's need to be computed only once, we neglect the multiplications and additions involved. Thus, pricing out with Winograd's scheme requires $(k + 1)m/2$ multiplications and $k(3m/2 + 1) + m/2 - 1$ additions per iteration. Typically, $k >> m$. If the multiply and add times were identical, the two procedures would be roughly equivalent. If the multiply time is signigicantly slower than the add time, as is generally the case, the acceleration using Winograd's procedure can be dramatic. Winograd's procedure requires roughly half as many multiplications as the standard procedure and roughly 50 per cent more additions.

Improvement factor:

If p is the fraction of time spent pricing out with the standard method and r is the ratio of multiply time to add time, then using Winograd's algorithm is approximately $2(r + 1)/[p(r + 3) + 2(1 - p)(r + 1)]$ times as fast as using a standard LP code. For $r = 3$, a fairly representative number for current computers, the improvement factor is $8/(8 - 2p)$.

Now we estimate p. A pivot takes about m^2 multiplications and m^2 additions. By comparison, the number of operations to find the representation of the pivot column in terms of the current basis is of the same order and the number to determine the pivot row is negligible. We do not count input/output time. Thus p is about $k/(2m + k)$, which is typically near one. For $r = 3$, we can expect an improvement of almost 25 percent.

Other LP applications:

We do not see where Winograd's procedure would be helpful in general in pivoting. However, sometimes a good starting basis is known (often, in Markov renewal programming [1]) and Winograd's procedure can then be applied to efficiently compute its inverse. See [3] for details.

Sparse matrices:

Various programming tricks for handling sparse matrices are available. When using these tricks, the relative efficiency of Winograd's procedure should be reexamined. We believe that Winograd's procedure will generally turn out to be faster.

REFERENCES

1. Denardo, E. V., and B. L. Fox, "Multichain Markov Renewal Programs," SIAM Journal Appl. Math. 16 (1968), 468-487.

2. Simonnard, M., Linear Programming, Prentice-Hall, Englewood Cliffs, New Jersey, 1966.

3. Winograd, S., "A New Algorithm for Inner Product," IEEE Trans. on Computers C-17 (1968), 693-694.