ornl

ORNL/TM-10929
(CESAR-88/52

OAK RIDGE
NATIONAL
LABORATORY

MARTIN MARIETTA

A New Algorithm for Linear Programming That Is Easy To Implement: Application to the Transportation Problem

J.-C. Culioli
V. Protopopescu

OAK RIDGE NATIONAL LABORATORY
CENTRAL RESEARCH LIBRARY
CIRCULATION SECTION
4500N ROOM 175
LIBRARY LOAN COPY
DO NOT TRANSFER TO ANOTHER PERSON
If you wish someone else to see this report, send in name with report and the library will arrange a loan.

OPERATED BY
MARTIN MARIETTA ENERGY SYSTEMS, INC.
FOR THE UNITED STATES
DEPARTMENT OF ENERGY

Printed in the United States of America. Available from
National Technical Information Service
U.S. Department of Commerce
5285 Port Royal Road, Springfield, Virginia 22161
NTIS price codes—Printed Copy: A03; Microfiche A01

This report was prepared as an account of work sponsored by an agency of the United States Government. Neither the United States Government nor any agency thereof, nor any of their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof.

ORNL/TM-10929
CESAR-88/52

Engineering Physics and Mathematics Division

A NEW ALGORITHM FOR LINEAR PROGRAMMING THAT IS EASY TO IMPLEMENT: APPLICATION TO THE TRANSPORTATION PROBLEM

J.-C. Culioli and V. Protopopescu

Published: September 1988

Prepared by the
OAK RIDGE NATIONAL LABORATORY
Oak Ridge, Tennessee 37831
operated by
MARTIN MARIETTA ENERGY SYSTEMS, INC.
for the
U.S. DEPARTMENT OF ENERGY
under contract DE–AC05–84OR21400

MARTIN MARIETTA ENERGY SYSTEMS LIBRARIES

3 4456 0282190 8

CONTENTS

ABSTRACT v
1. INTRODUCTION 1
2. THE SIGMOIDIC LINEAR-PROGRAMMING ALGORITHM 3
2.1. The Linear-Programming Problem 3
2.2. The Algorithm 4
2.3. Analysis of the Algorithm 6
3. APPLICATION TO THE TRANSPORTATION PROBLEM 11
4. NUMERICAL RESULTS 13
4.1. Example 1 13
4.2. Example 2 13
4.3. Example 3 17
5. CONCLUSIONS 23
ACKNOWLEDGEMENTS 25
REFERENCES 27
APPENDIX 29

ABSTRACT

We propose a new algorithm for linear-programming and apply it to solve some transportation problems. We prove the convergence of the algorithm under reasonable sufficient conditions. The main motivation of this new linear-programming solver is its extreme simplicity and its parallel structure which allows for straightforward implementations on personal computers as well as on more sophisticated machines. We give in the Appendix a version of the algorithm programmed in C in less than 30 lines. We also comment in the Appendix the relationship of our algorithm with some models of neural networks.

1. INTRODUCTION

Since Khachiyan [1] in 1978 and Karmarkar [2] in 1984 have revived the search for new linear-programming algorithms, many attempts have been made to surpass or at least equal the robust Simplex method introduced by Dantzig [3] in the late fourties. There are now so many competitors to the Simplex (Ref. [4] to [7] to cite a few) that one could wonder if it is necessary to add one more stone to an already solid edifice. Another reticence could be linked to the following remark: if the number of algorithms increases linearly, the number of versions on different machines and the number of numerical tests to compare them, increase almost exponentially and this amounts to a non negligible investment in terms of coding and testing for the linear-programming community.

Despite these possible caveats, we do propose here a new linear-programming algorithm and show its virtues both by theoretical analysis and by applying it to the Transportation Problem. We do not claim that the sigmoidic algorithm (the reason for this name will be clear in the next Section) is faster than any of the recent excellent algorithms that have been discovered in the late few years or that it can beat the Simplex in some instances. But we do claim and show that is is extremely simple to understand, analyze, and implement - three properties which are only rarely shared simultaneously by the above-mentioned algorithms. In addition, the method is easy to parallelize. Some very large-scale integration circuits could also be designed to mimic its parallel structure without many difficulties. The second part of the Appendix stresses this point in showing the connection of our algorithm with a neural network model.

In the next Section, we briefly describe the linear-programming problem, the sigmoidic algorithm, and conduct the theoretical analysis. We apply it to the Transportation Problem in Section 3 and give some numerical results in Section 4. The Appendix contains the listing of a C program that executes the algorithm for a sample program.

2. THE SIGMOIDIC LINEAR-PROGRAMMING ALGORITHM

2.1 . The Linear-Programming Problem

The standard linear programming problem can be stated as follows:

Find

$$\text{Max} \quad \Sigma_i \, c_i \, . \, x_i \qquad (1)$$

subject to the constraints

$$x_i \geq 0, \quad i = 1,2,\ldots, N \qquad (2\text{-a})$$

and

$$\Sigma_i \, a_j^i \, . \, x_i = b_j, \quad j = 1,2,\ldots,m. \qquad (2\text{-b})$$

Usually the vectors $(x_i) = x$ and $(c_i) = c$ are referred to as the primal vector and the cost vector. We shall denote by A the matrix $[a_j^i]$ and by b the constraint vector. In this notation, (1)-(2) can be written as the primal problem

Find Max $\langle c,x \rangle$ subject to $Ax = b$ (PP).

One can introduce the dual problem associated with (PP):

Find Min $\langle b,p \rangle$ subject to $A'p \geq c$ (DP).

Here, A' denotes the transpose of A and $p = (p_j)$ is the dual vector associated with the constraints. The equivalence of (LP) and (DP) is summarized [3] by the following statements:

- (LP) has a bounded solution x^* if and only if (DP) has a bounded solution p^*.
- The optimal couple (x^*, p^*) is such that

(i) $Ax^*=b$,

(ii) $A'p^* \geq c$

(iii) $\langle c,x^* \rangle = \langle b,p^* \rangle$

In the following, we assume that (1)-(2) has a bounded solution, denoted by x^*, and that we know beforehand a bound for each x_i, denoted X_i. Typically, X_i should be greater than x_i^*.

2.2 The Algorithm.

In order to present the sigmoidic algorithm, a few preliminary definitions are necessary. A one dimensional parameter-dependent mapping $g_T : R \to R$ defined by

$g_T(y) = 1/(1+\exp(-y/T))$ is referred here to as a sigmoidic function.

The typical graph of g_T (we will only consider $T > 0$) is given in Figure 1.

The most important properties of sigmoidic functions are the following:

for every y in R and any T in R_+^*, we have:

(S1) $0 < g_T(y) < 1$.

(S2) $g_T'(y) = \frac{1}{T} g_T(y)\, g_T(-y)$.

(S3) $0 < g_T'(y) < \frac{1}{T}$.

The inverse mapping of g_T is easily computed as

$g_T^{-1}(z) = T \operatorname{Log}[z/(1-z)]$ for z in (0,1).

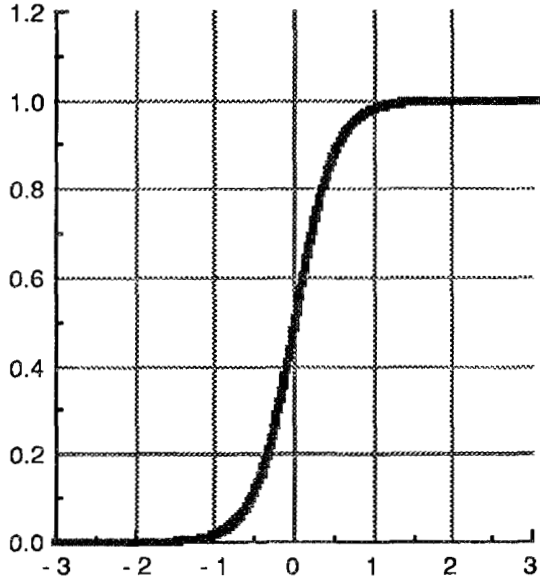


Figure 1. A sigmoidic function.

The sigmoidic function plays a crucial role in the design of the algorithm. The introduction of such "ad hoc" functions can be compared to the introduction of barrier functions like the "potential functions" of Karmarkar. As will be seen in Appendix A.1 (which describes Figure 2), the sigmoidic functions have the interesting geometric property of transforming a linear program into a parametrized non-linear optimization problem, enabling us to use non-linear search algorithms.

The sigmoidic algorithm represents a primal-dual approach similar to some non-linear saddle-point algorithms [8]. It consists in successive iterations on the primal and dual variables:

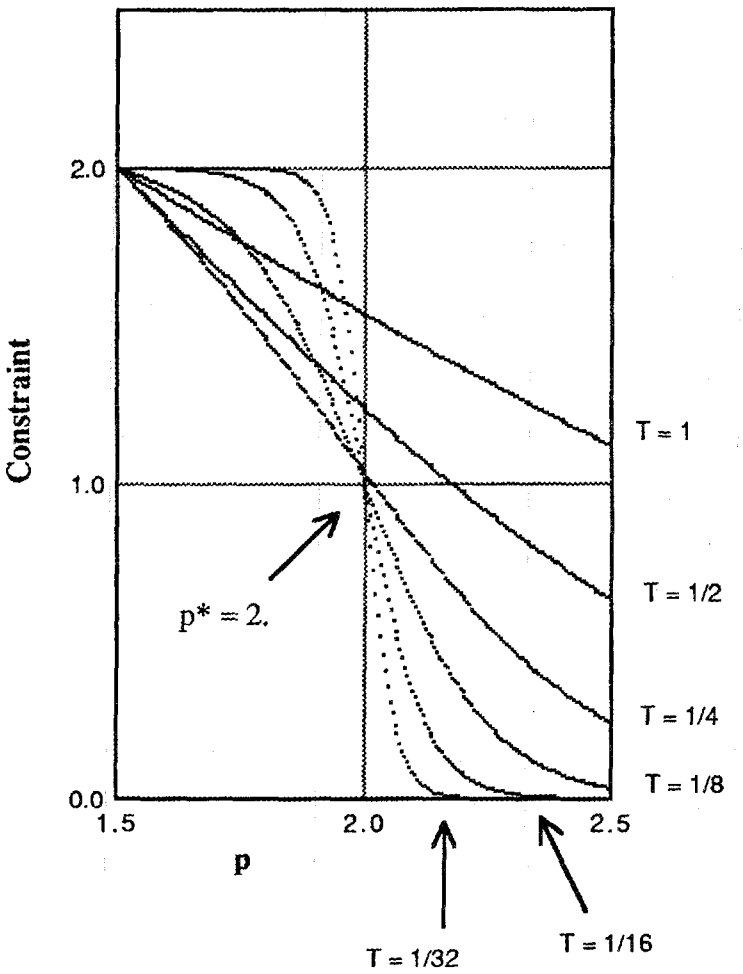


Figure 2. Non-linear interpretation (see Appendix A.1).

- initialize p_j^0 for $j = 1,2, ..., m$.
- for $k \geq 0$, update x_i and p_j according to:

for $i = 1, 2, ..., n$, $\quad x_i^{k+1} = X_i . g_T [c_i - \Sigma_j p_j^k . a_j^i]$ (3)

for $j = 1,2, ..., m$, $\quad p_j^{k+1} = p_j^k + \varepsilon .[\Sigma_i a_j^i . x_i^{k+1} - b_j]$ (4)

- stop when an optimality criterion (to be defined) is satisfied.

Here ε is a small positive parameter. Conditions on ε and T will be given in the next Section. Notice that the sigmoidic algorithm is already written in a parallel form.

2.3 Analysis of the Algorithm.

In order to prove convergence and optimality, we have to assume a set of conditions that are not always met by every linear-programming problem. These conditions are, however, satisfied by the transportation model (see Section 3). We also give in Section 4 a numerical example for which these conditions are not met but for which the algorithm works. Therefore, we think that our hypotheses are not the most general ones and weakening them would allow to include more general linear programming problems.

Now we are able to state the following

Theorem:

Assume that:

- all the components of b and the elements of A are non negative.
- for all $i \leq n$, there exists at least one j for which $a_j^i \neq 0$, so that we can define $X_i = 2 \, Max_j \{ b_j / a_j^i \mid a_j^i \neq 0 \}$. X_i is ≥ 0 and is the bound on x_i used in (3).
- the rank of the matrix A is m.

Then it follows that:

- for ε small enough, the sequence p^k converges to a finite vector $\hat{p}$ and consequently, the sequence x^k also converges to a limit denoted by $\hat{x}$.
- $\hat{x}$ is a feasible solution of (1) - (2) and $\hat{p}$ is a feasible solution of the dual problem corresponding to (1)-(2).
- The duality-gap $|\langle c,x\rangle - \langle b,p\rangle|$ is linearly bounded by T.

Proof:

Convergence.

By eliminating x_i^{k+1} between equations (3) and (4), we obtain a closed iteration formula for $p = (p_1, \ldots, p_m)$: $p^{k+1} = H(p^k)$, where H is applied componentwise to p, namely:

$$H(p_j) = p_j + \varepsilon \ \{ \Sigma_i \ a_j^i \ X_i \ g_T [c_i - \Sigma_h \ p_h \cdot a_h^i] - b_j \}$$

Let us compute the Jacobian of H:

We have $DH(j,j) = 1 - \varepsilon \ \{ \Sigma_i \ (a_j^i)^2 \ X_i \ g' [c_i - \Sigma_h \ p_h \cdot a_h^i] \}$

and for $k \neq j$, $DH(j,k) = - \varepsilon \ \{ \Sigma_i \ a_j^i a_k^i \ X_i \ g' [c_i - \Sigma_h \ p_h \cdot a_h^i] \}$.

According to property (S2), we have $g'[c_i - \Sigma_h \ p_h \ a_h^i] > 0$.

With the notations $g_i = \sqrt{g'[c_i - \Sigma_h \ p_h a_h^i] \ X_i}$

and $B_j^i = a_j^i \ g_i$, we have the simpler expressions:

$DH(j,j) = 1 - \varepsilon \ \Sigma_i \ (B_j^i)^2$ and $DH(j,k) = - \varepsilon \ \Sigma_i \ B_j^i B_k^i$.

Thus $DH = 1 - \varepsilon \ \ \mathrm{Gram}(u_1, \ldots, u_m)$ with $u_j = (B_j^1, \ldots, B_j^n)$.

Since rank(A) = m and every g_i is positive, then $B = [B_j^i] = [a_j^i \ g_i]$ is an mxn matrix of rank m. Gram(B) is an mxm symmetric and positive definite matrix of rank m (See [11], p. 155). Its spectrum contains m (possibly degenerate) real and positive eigenvalues

l_k, $1 \le k \le m$, which satisfy the following inequalities: $\|Gram(B)\| = l_1 \ge \ldots \ge l_m > 0$, where $\| . \|$ denotes the Euclidian norm.

Therefore, we can find $\varepsilon > 0$ such that $1 - \varepsilon\, l_1 < 1$ and $1 - \varepsilon\, l_m > -1$.

Thus, with $0 < \varepsilon < \frac{2}{l_1}$, $\|1 - \varepsilon\ Gram(B)\| < 1$ and the mapping H is a contraction, i.e. the sequence p^k converges to a fixed point.

Optimality.

Here, we refer to the properties of the optimal couple (x^*,p^*), given in Section 2. We prove that (i) and (ii) are satisfied by the limits of the x^k and p^k sequences. We then prove that (iii) can be satisfied as closely as desired, depending on the value of T. The derivation of the first optimality condition is conducted on a continuous version of the algorithm, i.e. we consider the updating equation for p as an ordinary differential equation ($0 < \varepsilon << 1$). With $dt = \frac{1}{\varepsilon}$, the evolution equations associated with (3)-(4) are

$$x(t) = X \,.\, g(c - A'p(t)) \qquad (5)$$

$$\frac{d}{dt} p(t) = Ax - b \qquad (6)$$

where $X \ge x^*$ and g_T is applied componentwise on its vector argument.

<u>Result (i) :</u>

We show that $\lim_{t \to \infty} Ax(t) = b$

To prove this, let us consider $V(t) = \frac{1}{2}\|Ax(t)-b\|^2 \ \ge\ 0$.

We have $\frac{d}{dt}V = \langle A\ \frac{d}{dt}x, Ax-b\rangle = \langle A\ \frac{d}{dt}x,\ \frac{d}{dt}p\rangle$ (with (6)).

If we take the derivative of (5), we get

$$\frac{d}{dt}x = - G A' \frac{d}{dt}p \qquad (7)$$

with $G = [\, X_i \,.\, g'(c_i - \Sigma_h\, p_h\, a_h^i)\, d_{ij}\,] = [\, g_i^2\, d_{ij}]$

Thus $\frac{d}{dt}V = - < A\, G\, A' \frac{d}{dt}p,\ \frac{d}{dt}p > = - < G\ A' \frac{d}{dt}p,\ A' \frac{d}{dt}p > \ \leq\ 0.$

The last inequality comes from the fact that rank(A') = rank(A) = m and G is a positive invertible matrix. The equality sign corresponds to the equilibrium situation, $\frac{d}{dt}p = 0$.

<u>Result (ii):</u>

Let us introduce new variables $y_i := c_i - \Sigma_j\, a_i^j\, p_j$.

We would like to prove that $\mathrm{Lim}_{k\to\infty}\ A'p^k \geq c$, which means that the variables y_i are negative whenever x^k and p^k have converged. We have $x_i = X_i \,.\, g_T\,(y_i)$.

Thus $y_i = T\,\mathrm{Log}[\frac{x_i}{X_i - x_i}]$. In the limit, if $x_i \leq X_i/2$, then $y_i \leq 0$.

On the other hand, if $x_i > X_i/2$, this implies that $y_i > 0$, but then we are able to prove that at least one constraint in Ax = b is not satisfied: suppose there exists an index i_0 such that $x_{i_0} > X_{i_0}/2$, then $x_{i_0} > \mathrm{Max}_j\ \{b_j / a_j^i\,,\ \text{for } a_j^i \neq 0\}$.

This maximum is attained at $j = j_0$. Thus $\Sigma_i\, a_{j_0}^i \,.\, x_i > b_{j_0}$ and p_{j_0} could not have converged.

<u>Result (iii):</u>

Consider the quantity $\quad W = \Sigma_i\, W_i = \Sigma_i\ x_i\, y_i = \Sigma_i\, x_i\, (\, c_i - \Sigma_j\, a_i^j\, p_j\,)$

$$= <c,x> - \Sigma_j\, (\, \Sigma_i\ x_i\, a_i^j\,)\, p_j \ = <c,x> - <b,p>$$

W is the duality-gap.

We have $W_i = X_i \; y_i \; g_T(y_i) \; = \; T \; X_i \; \frac{y_i}{T} \; \frac{2}{1+\exp(-y/T)}$

With $u_i = - \frac{y_i}{T} > 0$, we have $W_i = h(u_i) = - T X_i \frac{u_i}{1 + \exp(u_i)}$

Since $\mathrm{Lim}_{u_i \to +\infty} \; h(u_i) \; = 0$ and $h(0) = 0$, h has a minimum.

This minimum is attained at u^0, solution of $h'(u^0) = 0$, that is:

$1 + \exp(u^0) - u^0 \exp(u^0) \; = 0.$

One finds easily that $1 \leq u^0 \leq 2$. Thus, for all i, we have

$$0 \geq \quad W_i \quad \geq - T X_i \frac{u^0}{1 + \exp(u^0)} = - T X_i \exp(-u_0) \geq - T \frac{X_i}{e}$$

which proves that

$$0 \geq \quad W \quad \geq - T \frac{n}{e} \mathrm{Sup}\{X_i\}$$

The lower bound for W can be made as small as needed if T is chosen small enough. Although in some situations this bound can be actually attained, in general it gives a rather pessimistic estimate, as is shown in the numerical results Section.

3. APPLICATION TO THE TRANSPORTATION PROBLEM

The Transportation Problem can be formulated in the following way [3]: given N sources indexed by i with stocks S_i, i =1, 2,..., N, and M destinations indexed by j with demands D_j, j =1, 2,..., M, a transportation cost $c_{ij}\ x_{ij}$ is associated with the shipment of the (positive) quantity x_{ij} from the source S_i to the destination D_j. One assumes there is no loss during the process, i.e. for every source i, $\Sigma_j x_{ij} = S_i$ (a "house-keeping" constraint) and also that the demand is met for every destination, that is $\Sigma_i x_{ij} = D_j$ (a "demand-satisfaction" constraint). Notice that these two assumptions lead to

$$\Sigma_i \Sigma_j\ x_{ij} = \Sigma_j\ D_j = \Sigma_i\ S_i.$$

The problem of minimizing the cost of the transportation leads to the linear program:

$$\text{Min}\ \Sigma\, c_{ij}\ x_{ij} \quad \text{with} \quad x_{ij} \geq 0, \quad \Sigma_j\ x_{ij} = S_i \quad \text{and} \quad \Sigma_i\ x_{ij} = D_j$$

We show here how to apply the algorithm to the above model which enters perfectly the theoretical scheme discussed in Section 2. The first step is to find natural boundaries for the primal variables. In our case, since the stocks S_i and the demands D_j are known data, they both constitute natural boundaries on x_{ij}. Thus $0 \leq x_{ij} \leq X_{ij} = \text{Min}\{S_i\ ,\ D_j\}$. The existence of well-scaled boundaries is important for a good behavior of the algorithm.

Now, we have to define a dual variable for each constraint.

We denote by p_i and by q_j the dual variable associated to $\Sigma_j\ x_{ij} = S_i$ and to $\Sigma_i\ x_{ij} = D_j$, respectively. The initialization of the algorithm consists in assigning preliminary values to the dual variables p_i and q_j. A good guess of these dual variables - which have the economic interpretation of prices associated to the constraints - speeds up the algorithm. However, $p_i^0 = q_j^0 = 0$, works well too.

Once the initialization is done, the variables x_{ij}, p_i and q_j are updated, for $k \geq 1$, according to:

$$x_{ij}^{k} = X_{ij}\ g_{\Gamma}\ (-\ c_{ij} - p_i^{k} - q_j^{k}), \quad \text{for every i and j.}$$

$$p_i^{k+1} = p_i^{k} + \varepsilon\ (\ \Sigma_j\ x_{ij}^{k} -\ S_i), \quad \text{for every i.}$$

$$q_j^{k+1} = q_j^{k} + \varepsilon\ (\ \Sigma_i\ x_{ij}^{k} -\ D_j), \quad \text{for every j.}$$

Intuitively, the updating of x_{ij} consists in allocating a portion of the total possible X_{ij} that depends on the associated costs. The updating of p or q is intended to meet the "house-keeping" and the "demand-satisfaction" constraints with correct prices. The economics analog is a "one goods buyer (destination) or seller (source) market": prices are adjusted until the constraint is satisfied.

4. NUMERICAL RESULTS

4.1 Example 1

We first consider a very simple problem for which the assumptions of the theorem are fullfilled. This example is the sample problem that appears in the listing given in the Appendix.

$$\text{Max } z = 2x_1 + 4x_2 + 4x_3 - 3x_4, \quad x_i \geq 0$$

$$\text{subject to the constraints} \quad x_1 + x_2 + x_3 = 4$$

$$\text{and} \quad x_1 + 4x_2 + x_4 = 8$$

The optimal solution is given by $x^* = (0,2,2,0)$, $p^* = (4,0)$ and $z^* = 16$.

We have chosen $T = 0.2$, $\varepsilon = 0.02$. The boundaries we have used are equal to 4 for every variable. In Figures 3 and 4 we give the trajectories of the primal, dual variables, primal cost $\langle c,x \rangle$ and dual cost $\langle b,p \rangle$. The convergence takes about a hundred iterations.

4.2 Example 2.

Now, we consider another problem which does not satisfy the conditions of the theorem. However, the algorithm finds the right solution in approximately the same number of iterations.

The problem is

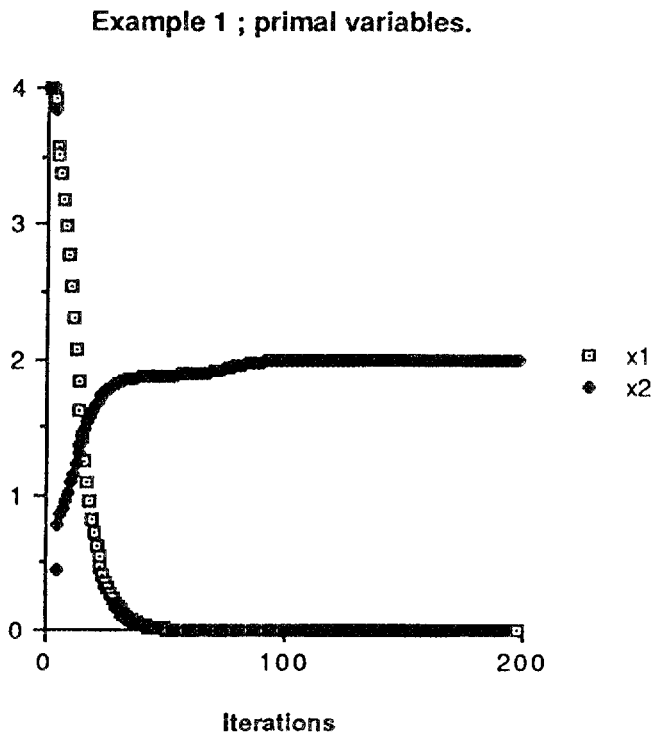
$$\text{Max } z = x_1 + 5x_2 + 3x_3, \quad x_i \geq 0$$

$$\text{subject to the constraints:} \quad x_1 + 2x_2 + x_3 = 3$$

$$\text{and} \quad 2x_1 - x_2 = 4$$

The optimal solution is given by $x^* = (2,0,1)$, $p^* = (3,-1)$ and $z^* = 5$.

We have chosen $T = 1$, $\varepsilon = 0.01$. The boundaries are 3., 3. and 1.5 respectively (computed from the first equation). In Figure 5, we present the trajectories of the primal and



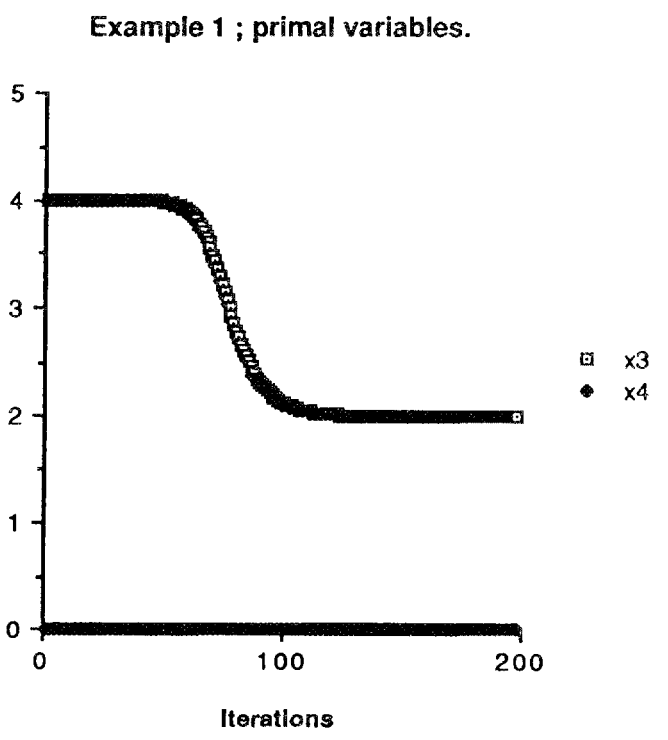
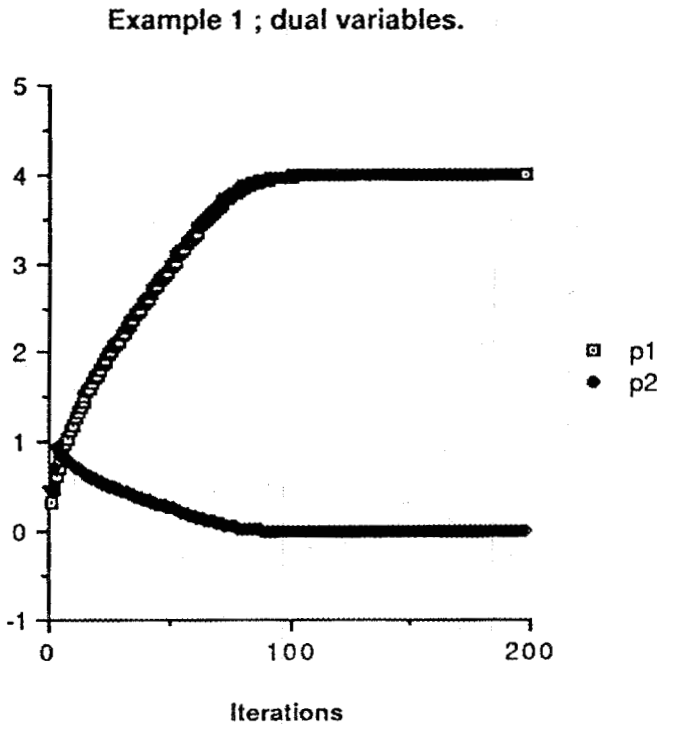


Figure 3. Trajectories of the primal variables for example 1.



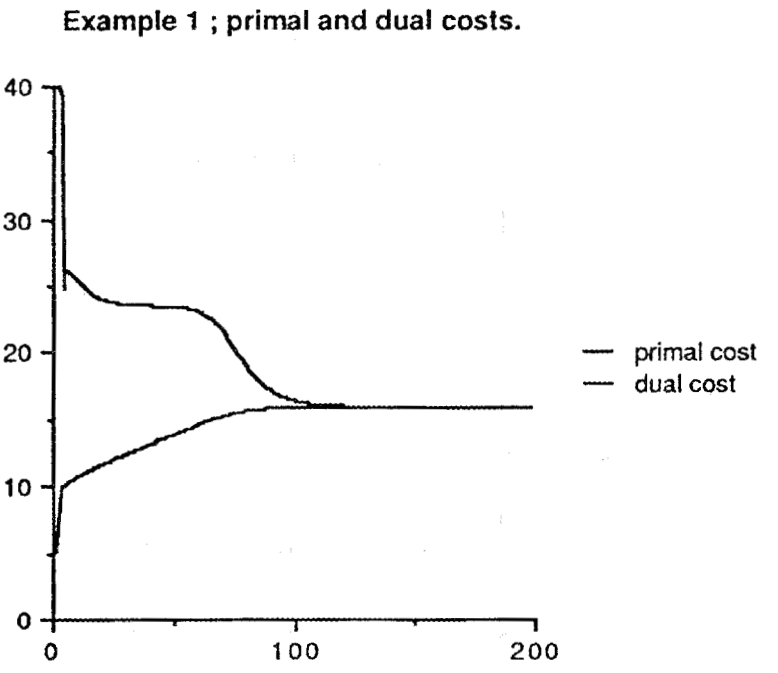
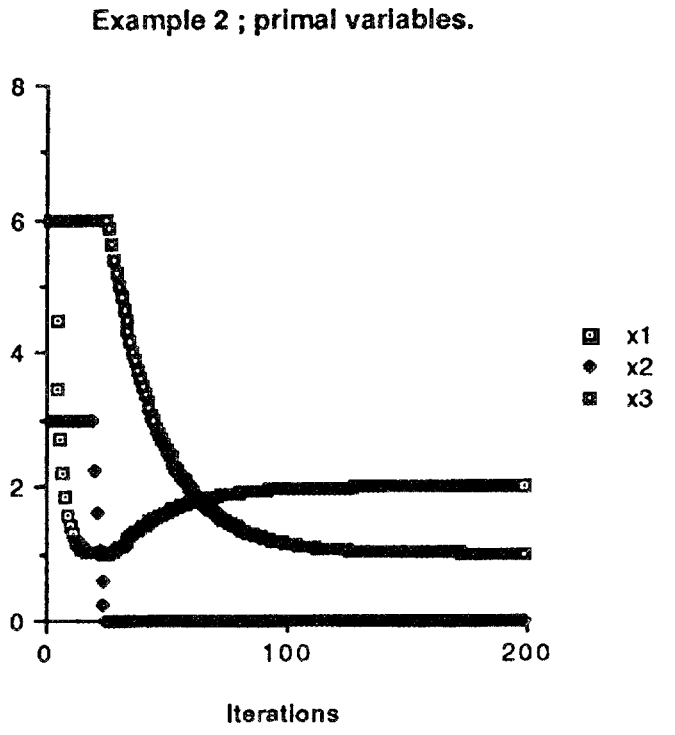


Figure 4. Trajectories of the dual variables and of the costs for example 1.



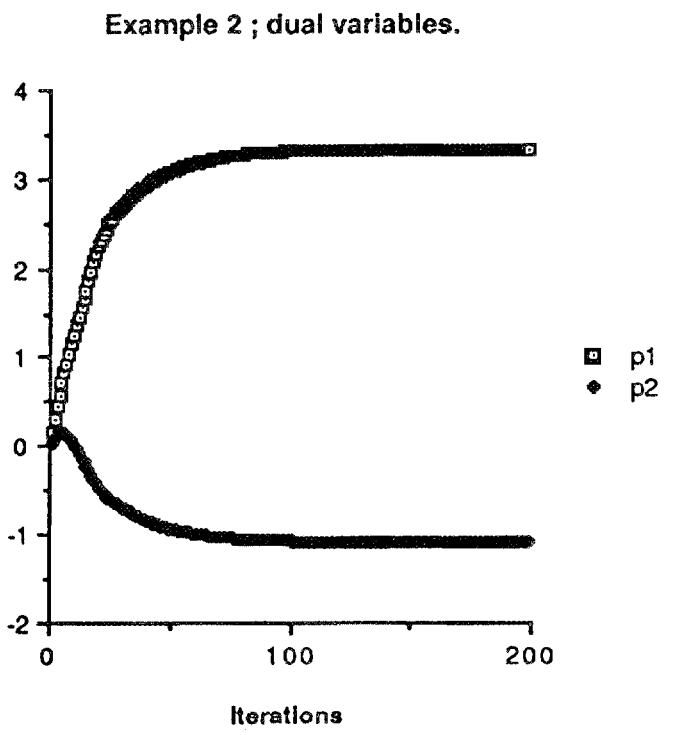


Figure 5. Trajectories of the primal dual variables for example 2.

dual variables. In order to illustrate that the duality gap is still linearly bounded by T, we have also run the algorithm with T = 0.2. Figure 6 shows clearly the consequent reduction of the duality-gap.

4.3 Example 3: a Transportation Problem.

We consider the following transportation problem (Figures 7 to 9):

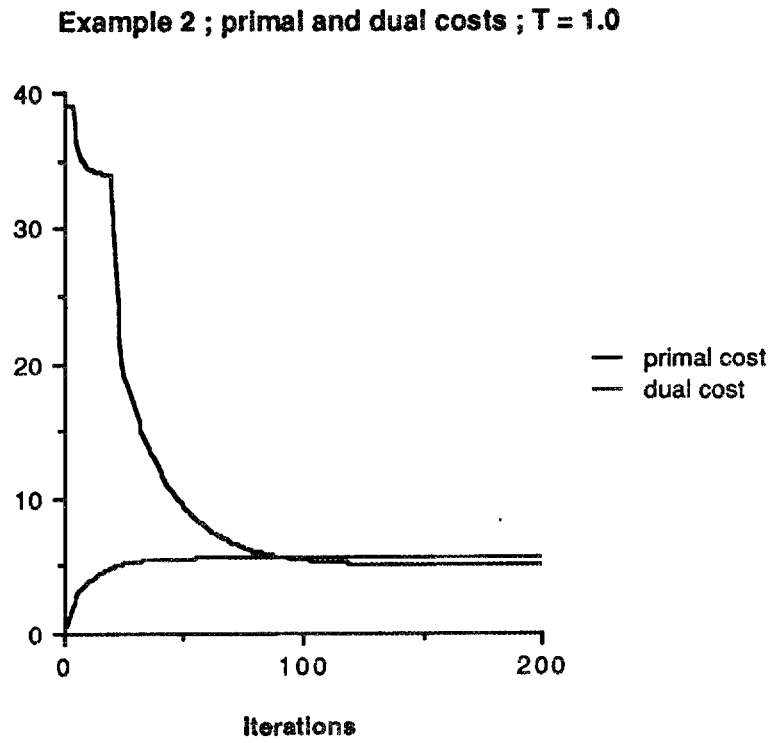
$$\text{Min} \quad z = 464\, x_{11} + 513\, x_{12} + 654\, x_{13} + 867\, x_{14} + 352\, x_{21} + 416\, x_{22} + 690\, x_{23} + 791\, x_{24} + 995\, x_{31} + 682\, x_{32} + 388\, x_{33} + 685\, x_{34}$$

subject to the standard constraints $\Sigma_j\ x_{ij} = S_i$ and $\Sigma_i\ x_{ij} = D_j$

with $S_i = 75, 125, 100,$ respectively and $D_j = 80, 65, 70, 85.$

We have taken $\varepsilon = 0.1$ and T = 12. The optimum value is 152, 535.

After one hundred and fifty iterations (Figure 9), z stabilizes at 153, 075, which is at 0.3 % of the optimum. Reducing T would - according to the above theorem - decrease this lack of optimality.



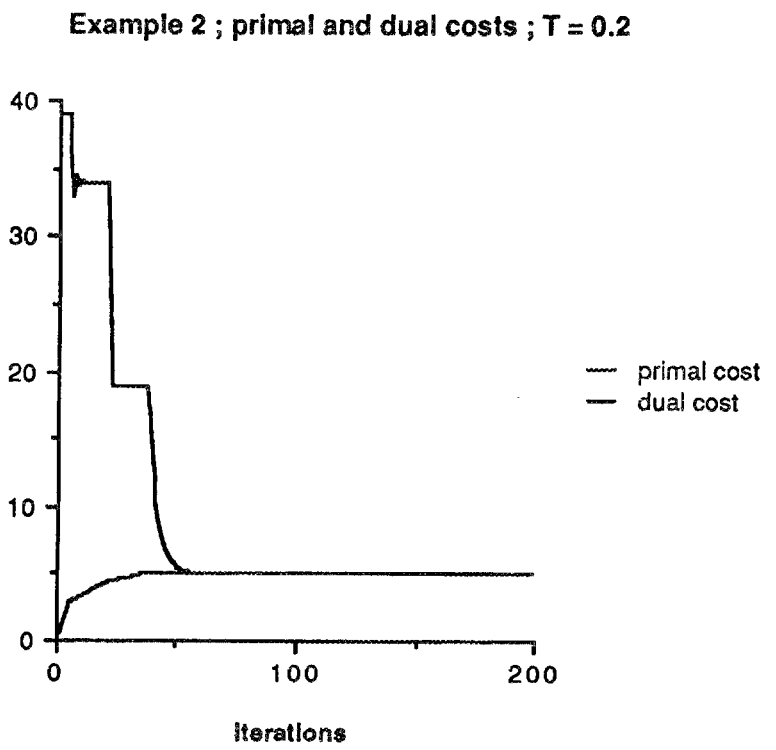
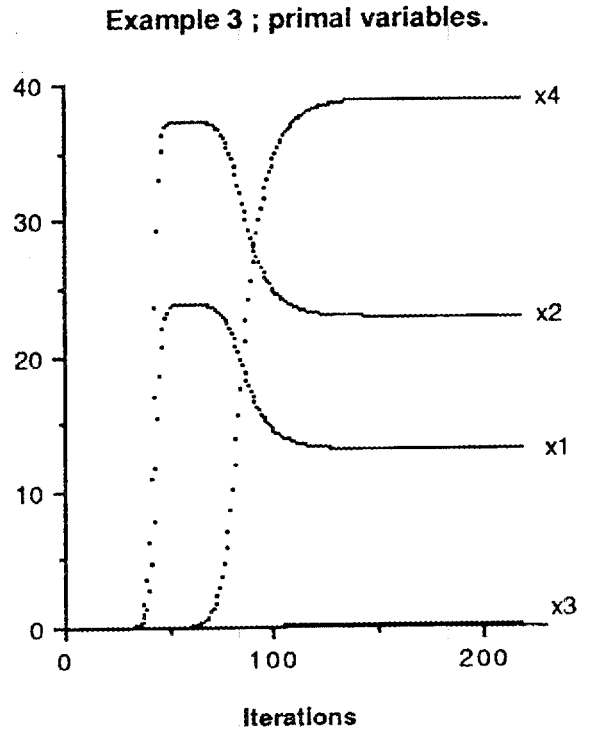


Figure 6. Primal and dual costs for example 2.



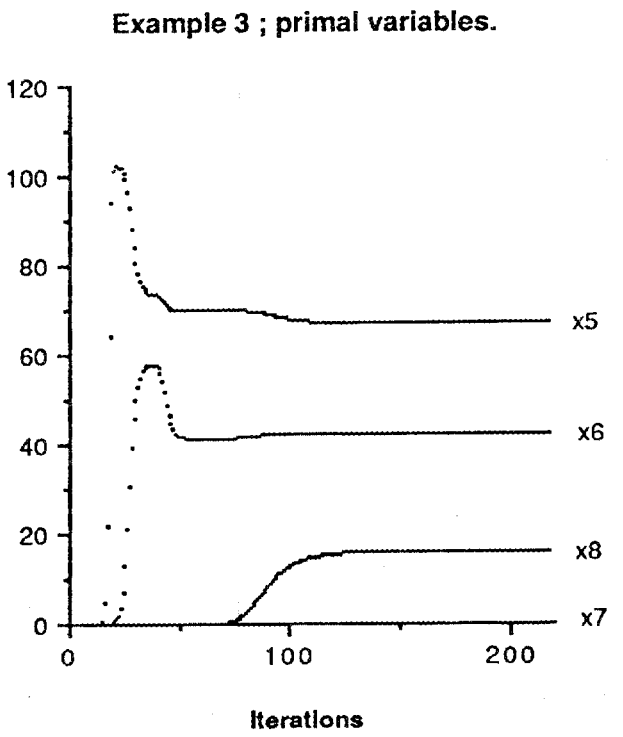
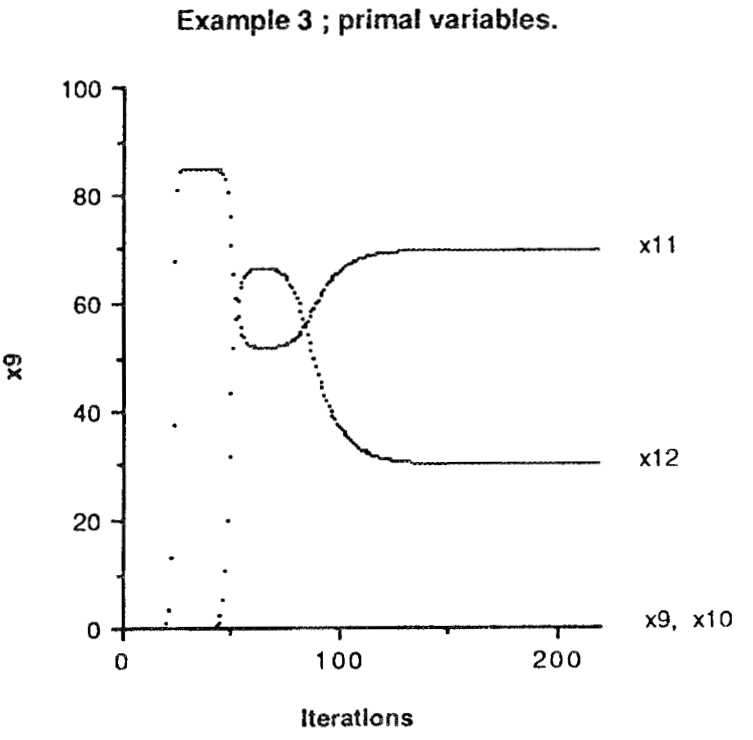


Figure 7. Trajectories of the primal variables for example 3.



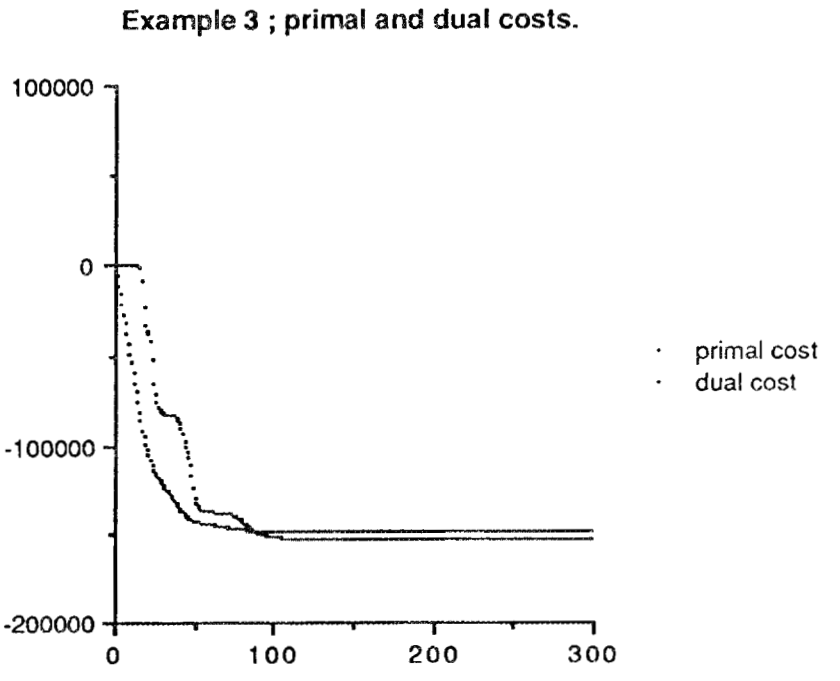
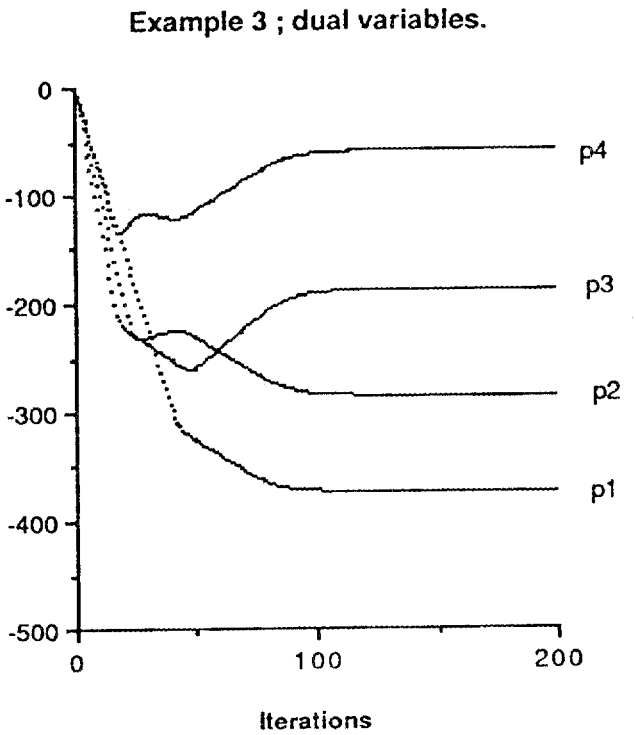


Figure 8. Trajectories of the primal variables and of the costs (example 3).



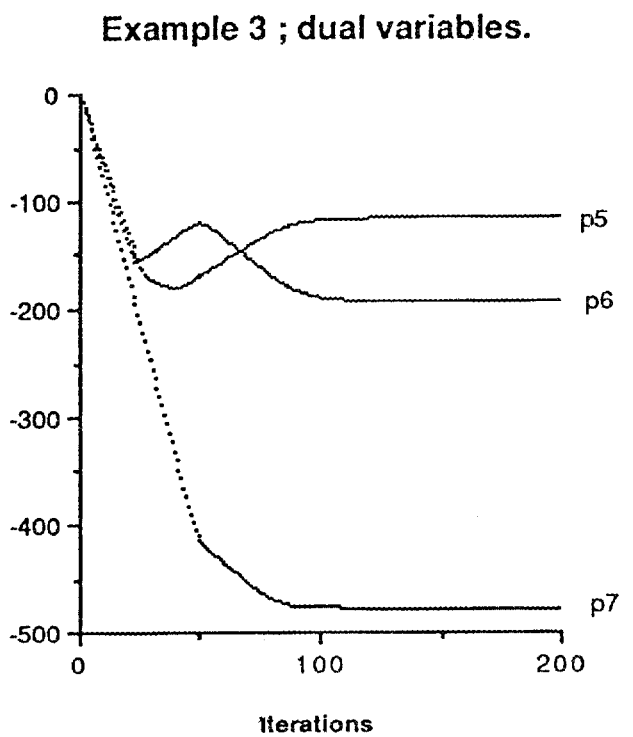


Figure 9. Trajectories of the dual variables (example 3).

5. CONCLUSIONS

In this preliminary theoretical and numerical study of the new sigmoidic linear-programming algorithm, we have shown that it is extremely simple to understand and to implement. It does not rely on any large numerical package nor does it use any complex algebraic transformation. It is fully parallel and can be implemented on very high speed parallel computers. We have shown its connection with some neural networks models.

We are currently in the process of testing the sigmoidic algorithm on some large-scale transportation models, of weakening the conditions in the theorem, and of extending the class of sigmoidic algorithms to non-linear optimization [13].
These algorithms have very interesting non-linear properties that, according to us, could be used in order to speed-up many optimization algorithms [14].

ACKNOWLEDGMENTS

This research was partially sponsored by the Office of Military Applications and the Office of Basic Energy Sciences, U.S. Department of Energy, under contract DE-AC05-84OR21400 with Martin Marietta Energy Systems, by OJCS, U.S. Department of Defense under Interagency Agreement DOE #1612-1657-A1 with Martin Marietta Energy Systems, and by an ORNL Postgraduate Research Appointment administered by Oak Ridge Associated Universities.

We thank R. C. Mann for his critical reading of the manuscript and his continuous support.

REFERENCES

[1] L. G. Khachiyan, "A polynomial algorithm in linear programming", Soviet Mathematics Doklady 20, 191-194, 1979.

[2] N. K. Karmarkar, ``A new polynomial-time algorithm for linear-programming", Combinatorica 4 (4), December 1984.

[3] G. B. Dantzig, Linear programming and extensions, Princeton University Press, 1964.

[4] E. Tardos, "A strongly-polynomial algorithm to solve combinatorial linear programs", Operations Research, Vol. 34, N. 2, March-April, 1986.

[5] C. L. Monma and A. J. Morton, "Computational experience with a dual-affine variant of Karmarkar's method for linear-programming", Operations Research Letters 6 (December, 1987), 261-267.

[6] K. A. McShane, C. L. Monma, and D. Shanno, "An Implementation of a Primal-Dual Interior Point Method for Linear-Programming", March 22, 1988.

[7] R. H. Leary, "L1 solutions of overdetermined linear equations by an affine scaling linear programming algorithm", presented at the SIAM National Meeting, July 11, 1988, Minneapolis, MN.

[8] K. J. Arrow, L. Hurwicz, and H. Uzawa, Studies in Linear and Non-linear Programming, Stanford University Press, 1972.

[9] L. Mirsky, An Introduction to Linear Algebra, Dover Publications Inc, New York, 1982.

[10] J.J. Hopfield and D. W. Tank, `` Simple ``neural" optimization networks: an A/D converter, signal decision circuit and a linear programming circuit", IEEE Trans. on Circuits and Systems, Vol. CAS-33, n.5, May 1986.

[11] G. V. Wilson and G. S. Pawley, ``On the stability of the travelling salesman problem algorithm of Hopfield and Tank", Biol. Cyber. 58, 63-70 (1988).

[12] D. E. Rumelhart, J.L. McClelland and the PDP Research Group, Parallel Distributed Processing: Explorations in the Microstructure of Cognitions, Vol. 1 and 2, The MIT Press, Cambridge, MA, 1986.

[13] J.-C. Culioli and V. Protopopescu, "Linear and Non-Linear Sigmoidic Algorithms", ORNL TM, in preparation.

[14] J.-C. Culioli and V. Protopopescu, "A New Class of Optimization Algorithms with Bifurcations and Chaotic Behavior", presented at the SIAM National Meeting, July 11, 1988, Minneapolis, MN.

APPENDIX

A.1 Non-Linear Interpretation.

In order to make the notations easier, we consider the simple problem

$$\text{Max } z = c_1 x_1 + c_2 x_2 \text{ with } c_1 > c_2, \qquad \text{(I)}$$

and x_1 and x_2 subject to the constraints $x_1 + x_2 = 1$, and $x_i \geq 0$,

the solution of which is $x^* = (1,0)$, $p^* = c_1$.

We assume a parametrization of x_i as a function of p and transform this problem into

$$\text{Max } z(p) = c_1 x_1(p) + c_2 x_2(p) \qquad \text{(II)}$$

subject to the constraint $x_1(p) + x_2(p) = 1$, and $x_i(p) \geq 0$.

The associated Lagrangian is (with q the dual variable associated with the constraint in p)

$$L(p,q) = c_1 x_1(p) + c_2 x_2(p) - q\,[\,x_1(p) + x_2(p) - 1\,]$$

and the optimality conditions are, if we assume that $x_i(p)$ is differentiable:

$$\frac{\partial L}{\partial p} = c_1 x_1'(p) + c_2 x_2'(p) - q\,[\,x_1'(p) + x_2'(p)\,] = 0 \qquad \text{(CO1)}$$

$$\frac{\partial L}{\partial q} = x_1(p) + x_2(p) - 1 = 0. \qquad \text{(CO2)}$$

These two conditions give p as a solution of (CO2) and, with a parametrization $x_i(p)$ such that $x_i'(p) \neq 0$ for every p, we get $q = \dfrac{c_1 x_1'(p) + c_2 x_2'(p)}{x_1'(p) + x_2'(p)}$.

Now, if $x_i(p) = 2\ g_T(c_i - p)$, which the choice of the sigmoidic algorithm, after a few algebraic manipulations, we obtain

$$q = c_1 + (c_2 - c_1)\frac{1}{1+\{r(T)\}^2} \quad \text{with } r(T) = \frac{\cosh[\frac{p - c2}{T}]}{\cosh[\frac{p - c1}{T}]}.$$

In Figure 2, we have represented (with $c_1 = 2$ and $c_2 = 1$) the left handside of (CO2) for several values of T. As T decreases, the values of p(T) tend rapidly to the optimal value $p^* = c_1 = 2$.

Thus we infer that $\mathrm{Lim}_{T->0}\ p(T) = c_1$, as could be easily derived from (CO2). This also implies that $\mathrm{Lim}_{T->0}\ q(T) = c_1 = p^*$. In conclusion, we have shown that the primal and dual solutions of the linear problem (I) can be approximated as closely as needed by the solutions of the parametrized problem (II). This is exactly what the sigmoidic algorithm does.

A.2 Listing of the Sigmoidic Algorithm.

```
#include <stdio.h>
#include<math.h>
 main()
{ double x[4],xbar[4],p[2],constraint[2],y[4],criprim,cridual;
 double c[4],a[4][2],b[2],eps,temp;
 int nmax,nvar,ncont,i,j,k;
 FILE *outdata ;

 outdata = fopen("sigmo_LP","w+");

      /*   Data for the linear program     */
c[0] = 2.00 ; c[1] = 4.00; c[2] = 4.00 ; c[3] = -3.00;
a[0][0] = 1.00  ; a[1][0] =1.00;  a[2][0] = 1.00 ;   a[3][0] =  0.00   ;
```

```
a[0][1] = 1.00 ; a[1][1] = 4.00 ;  a[2][1] = 0.00 ;  a[3][1] =  1.00   ;
b[0] = 4.00 ; b[1] = 8.00;


        /*   Data for the algorithm   */
nmax =199;   nvar = 3; ncont = 1;
p[0] = 0.00 ;  p[1] = 0.00 ; eps = .02 ; temp =.2 ;
xbar[0] = 4.00 ; xbar[1] = 4.00 ; xbar[2] = 4.00 ; xbar[3] = 4.00;
        /* The algorithm */
for  (i = 1;i<=nmax;i = i+1){
        /* iteration on primal variables */
      for  (j=0;j<=nvar;j=j+1){
       x[j]=xbar[j]/(1.0+exp(-(c[j]-p[0]*a[j][0] - p[1]*a[j][1]) /temp));}
        /* computation of the constraints  */
      constraint[0] = - b[0];   constraint[1] = - b[1];
      for  (j=0;j<=nvar;j=j+1){
              constraint[0] =  constraint[0] +  a[j][0]*x[j] ;
              constraint[1] =  constraint[1] +  a[j][1]*x[j] ; }
        /* iteration on dual variables */
      p[0] = p[0] +  eps*constraint[0] ;
      p[1] = p[1] +  eps*constraint[1] ;
        /* primal and dual criterions */
      criprim = c[0]*x[0]+c[1]*x[1]+c[2]*x[2]+c[3]*x[3];
      cridual = b[0]*p[0]+b[1]*p[1];
fprintf(outdata,"%f\t%f\t%f\t%f\t%f\t%f\t%f\t%f\n",
        x[0],x[1],x[2],x[3],p[0],p[1],criprim,cridual) ; }
fclose(outdata);}
```

A.3 Connection with Neural Networks.

The sigmoidic algorithm described in this paper has some common features with neural network models [10,11]. However, the ``computation energy" of our algorithm is not quadratic, but linear, which greatly simplifies the model.

To model a linear program with a neural network, we introduce two sets of neurons: ``primal" and ``dual". The primal neurons correspond to the x_i. Their states take values in the interval $[0,X_i]$, where X_i has been defined above. By analogy, we simply denote the state of the primal neuron i by x_i. We also define dual neurons. The dual neuron j models the cost associated to the satisfaction of the constraint j. Its state, denoted p_j, can take any value between -P and P, with P a given positive constant.

Our computation energy is, by analogy with non-linear optimization, the Lagrangian function L:

$$L(x,p) = \Sigma_i c_i \cdot x_i + \Sigma_j p_j \cdot [\Sigma_i a_j^i \cdot x_i - b_j].$$

The updating rules for the primal and the dual neurons are given by the sigmoidic algorithm. They conduct the network to local extrema of the Lagrangian function.

One notices that the sigmoidic algorithm is expressed in a parallel distributed processing way [12]. The separation of the neurons in two classes allows the dual neurons to provide information about the constraints to the primal neurons. Each dual neuron acts as a coordinator that announces to all the primal neurons what is the price associated to the constraint. A representation of the model is given in Figure 10. The updating of the primal neurons is similar to the discrete updating of many neural network models. The sigmoid function g on which the algorithm is based is reminiscent of the neuron firing function with its ``best efficiency" and ``saturation" regions. The variables c_i and b_j are respectively the ``threshold" associated to neuron x_i and p_j, and the matrix coefficients a_j^i are the

``connection strengths" between primal and dual neurons. Since the connection matrix is given by the optimization problem, it is not - in general - symmetric. Yet, when m = N, one can establish an analogy with Hopfield's quadratic energy. In that case, the connection matrix must be symmetric, all the neurons must be primal, and there is no collective information conveyed by any dual variable. In that case, one can only minimize quadratic functions without constraints.

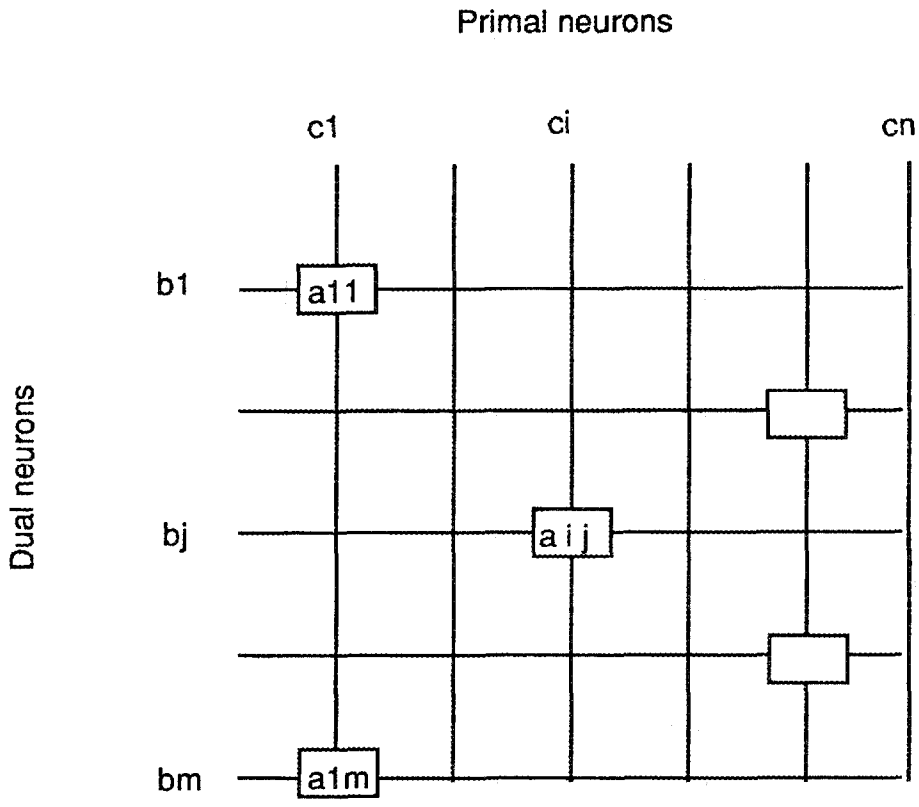


Figure 10. Architecture of a neural network based on the sigmoidic algorithm.

INTERNAL DISTRIBUTION

1-5. J. C. Culioli
6. G. de Saussure
7. C. W. Glover
8. J. Han
9. M. R. Hilliard
10. E. L. Hillsman
11. J. P. Jones
12. R. M. Leed
13. G. E. Liepins
14. F. C. Maienschein
15–19. R. C. Mann
20. C. March-Leuba
21. F. G. Perey
22. R. B. Perez
23. F. G. Pin
24–28. V. Protopopescu
29. R. S. Solanki
30. E. Wacholder
31. C. R. Weisbin
32. J. J. Dorning (consultant)
33. R. Haralick (consultant)
34. Central Research Library
35. Y-12 Technical Library
36–37. Laboratory Records
38. EPMD Reports Office
39. ORNL Patent Office
40. Laboratory Records–RC

EXTERNAL DISTRIBUTION

41. Dr. P. G. Anderson, School of Computer Science and Technology, Rochester Institute of Technology, One Lomb Memorial Drive, P.O. Box 9887, Rochester, New York 14623-0887

42. Professor J. P. Aubin, CEREMADE, Universite de Paris-Dauphine, Place du Marechal Delattre de Tassigny, 75017 Paris, France

43. D. C. Bond, BP America Inc., Research and Development, 4440 Warrensville Center Road, Cleveland, OH 44128-2837

44. Dr. Wayne Book, Department of Mechanical Engineering, J. S. Coon Bldg., Cherry Street, Georgia Institute of Technology, Atlanta, GA 30332

45. Richard Caasi, University Computer Center, San Diego State University, San Diego, CA 92182-0531

46. Dr. H. R. A. Cardon, Shell Oil Research, Cognitive Systems, Volmerlaan 6, Rijswijk, Holland

47. C. K. Cheng, Department of Electrical Engineering and Computer Sciences, University of California San Diego, La Jolla, CA 92093 (C-014)

48. Professor Guy Cohen, ENSMP-Centre d'Automatique, 35, rue St. Honore, 77305 Fontainebleau, France

49. N. E. Cotter, Department of Electrical Engineering, The University of Utah, 3262 Merrill Engineering Bldg., Salt Lake City, Utah 84112

50. Dr. A. DeCegama, GTE, 333 Lexington Road, Concord, MA 01742
51. Gregory J. Denault,Computer Engineering Artificial Intelligence Program, General Electric Co., P. O. Box 8, Bldg. K-1, Room 5C29A, Schenectady, NY 12301
52. Dr. A. L. Dia, Department of Computer Science, University of Southern Mississippi, Southern Station, Box 5106, Hattiesburg, MS 39406
53. J. B. Fau, 1409 Upland Drive, Houston, TX 77043
54. Professor Walter Freeman, University of California, Berkeley, CA 94720.
55. Dr. M. G. Fuhrman, Equipment Development Division, Alcoa Technical Center, Aluminum Company of America, Alcoa Center, PA 15069
56. G. Giandonato, CSELT, Via Reiss Romoli 279, 10148 – Torino, Italy
57. Ewald Heer, Heer Associates, 5329 Crown Avenue, LaCanada, CA 91011
58. Professor D. M. Himmelbau, Department of Chemical Engineering, The University of Texas at Austin, Austin, TX 7812-1062
59. Professor Morris W. Hirsch, Department of Mathematics, University of California, Berkeley, CA 94720
60. Paul Hollis, Department of Electrical and Computer Engineering, North Carolina State University, Raleigh, NC 27695-7911
61. John Ip, 9465 McNaught Road, Chilliwack, B.C., Canada V2P 6E9
62. John Jadik, Westinghouse D&EL, Box 746, MS-5370, Baltimore, MD 21803
63. Andrew B. Kahng, Department of Computer Science and Engineering, 3715-C Miramar Street, La Jolla, CA 92037
64. Avi Kak, Department of Electrical Engineering, Purdue University, Lafayette, Indiana 47907
65. Roxane Kamen, Code 421, NOSC, San Diego, CA 92152
66. Dr. Arthur Karler, Karler Laboratories, P.O. Box 373, Berkeley, CA 94701
67. George Komen, Sandia National Laboratory, Division 1274, P.O. Box 5800, Albequerque, NM 87185
68. Professor J. F. Magnan, Department of Mathematics, The Florida State University, Tallahassee, FL 32306
69. Dr. T. J. McAvoy, Department of Chemical Engineering, University of Maryland, College Park, MD 90742
70. A. J. Mimms, Douglas Aircraft Company, IMC–35-29, 3855 Lakewood Blvd., Long Beach, CA 90846
71. Joseph Naft, Computer Aided Design Laboratory, The University of Maryland, Engineering Research Center, College Park, MD 20742

72. Dr. Larry Orwig, Norden Systems, MS-RV04, P.O. Box 5300, Norwalk, CT 06856

73. Dr. John F. Palmer, NCUBE Corporation, 915 E. La Vieve Lane, Tempe, AZ 85284

74. Dr. Charles M. Patton, Bldg. 3U-L9, Hewlett-Packard Company, 100 NE Circle Blvd., Corvallis, OR 97331

75. Dr. Harry E. Pople, Decision Systems Laboratory, University of Pittsburgh, School of Medicine, Pittsburgh, PA 15261

76. Dr. Michael Rabins, Department of Mechanical Engineering, The Texas A&M University, 100 Engineering Physics Building, College Station, TX 77843-3123

77. Gary Ramet, Department of Computer Science, Chapman College, Orange, CA 9266

78. Karl N. Reid, College of Engineering, Architecture, and Technology, Oklahoma State University, Stillwater, Oklahoma 74074

79. Robert F. Richbourg, Department of Geography and Computer Science, United State Military Academy, OAIAE (MADN-B), West Point, NY 10996-1695

80. Dr. Steven K. Rogers, Neural Networks, Artificial Intelligence, AFITENG, WPAFB, OH 45433

81. Dr. Asim Roy, Department of Decision and Information Systems, Arizona State University, 1401 E. Brentrup Drive, Tempe, AZ 85287

82. M. Saridereli, EE 217, UMR, University of Missouri-Rolla, Rolla, MO 65401

83. Dr. John Shepanski, Data Systems Laboratory, One Space Park, Redondo Beach, CA 90278

84. Patrick K. Simpson, Artificial Intelligence Center, General Dynamics, P.O. Box 85310, San Diego, CA 92138 MZ 7202K

85. Peter Steinmetz, 2675 University #202, St. Paul, MN

86. Gene Tagliarini, Clemson University, Department of Computer Science, 435 Nursing Building, Clemson, SC 29634-1906

87. Barrett Tomlinson, Spectra-Physics, Autolab Division, 3333 North First Street, San Jose, CA 95134-1995

88. Dr. B. W. Wah, National Science Foundation, 1800 G Street, NW, Washington, DC 20550

89. Dr. James A. White, Honeywell, Inc., 3660 Technology Drive, Minneapolis, MN 55418

90. Dr. George W. Whitson, III, Department of Computer Science, The University of Texas at Tyler, 3900 University Blvd., Tyler, TX 75701

91. Dr. Jack H. Winters, AT&T Bell Laboratories, Research Department, Holmdel, NJ 07733

92. Dr. H. C. Yee, NASA – Ames Research Center, MS: 202A-1, Moffett Field, CA 94035

93. Dr. J. M. Zurada, Department of Electrical Engineering, University of Louisville, Louisville, KY 40292

94. D. Zwillinger, The MITRE Corporation, Burlington Road, MS: E025, Bedford, MA 01730-0208

95. Office of the Assistant Manager, Energy Research and Development, DOE-ORO, Oak Ridge, TN 37831

96–105. Office of Scientific and Technical Information, US/DOE, Oak Ridge, TN 37831