NUMERICAL IMPLEMENTATION AND COMPUTATIONAL RESULTS OF NONLINEAR NETWORK OPTIMIZATION WITH LINEAR SIDE CONSTRAINTS

F. Javier Heredia & Narcís Nabona
Statistics and Operations Research Department
Universitat Politècnica de Catalunya
Pau Gargallo 5, 08028-Barcelona
e-mail : heredia@eio.upc.es

1. Introduction

This paper deals with the solution of the nonlinear network flow problem with linear side constraints. The problem to solve is the minimization of a nonlinear objective function (a cost function) whose variables are the values of the flows passing through the capacitated arcs of an oriented network. We expect also this flows to satisfy a set of linear inequality constraints, called the *side constrains* . The mathematical expression of this problem is :

$$\begin{array}{rlr} \min & f(x) & (1) \\ \text{subj. to :} & Ax + e_l a \qquad = r & (2) \\ & Tx \qquad + \mathbb{1} z = b & (3) \\ & 0 \le x \le u \; ; \; z \ge 0 \; ; \; 0 \le a \le 0 & (4) \end{array}$$

we will refer to this problem as the NNS problem. In this formulation we have that :

(1) is a nonlinear function $f\colon \mathbb{R}^n \to \mathbb{R}$. $f(x)$ is suposed to be twice continuously differenciable on the feasible set defined by the constraints (2) to (4). The variables $x \in \mathbb{R}^n$ represent the values of the arc flows in the network.

(2) represents the *Network Equations.* Matrix $A \in \mathbb{R}^{m\times n}$ is the *node-arc incidence matrix* and $r \in \mathbb{R}^m$ is the supply/demand vector. A root arc a has been introduced in order to have a full row rank constraint matrix.

(3) is a set of t side constrains . We supose that $t < m$. The variables $z \in \mathbb{R}$ are the slacks of the original inequality ("$\le$") side constraints.

(4) $u \in \mathbb{R}^n$ are the upper bounds assigned to the flows in each arc. Lower bounds $l \in \mathbb{R}^n$ can be assimilated easily in this notation by making $\hat{x} = x - l$.

We make the following assumptions about NNS :

1) The graph $\mathcal{G}$ the oriented network arises from is connected.
2) Matrix T has full row rank.
3) Total supply equals total demand.

Let $y' = (\, x' \mid z' \,)$, $y \in \mathbb{R}^{n+t}$, be a feasible point for NNS. We consider that the subindex of slack variables z_i denotes the number of the side constrains associated with the slack ($z_i \equiv y_{n+i}$, $i = 1, \ldots, t$). In such a point, consider the columns of the constraint matrix $\bar{A} = \begin{pmatrix} A & e_l & 0 \\ T & 0 & \mathbb{1} \end{pmatrix}$, and the associated variables, partitioned as usual into a basic ($\bar{\mathcal{B}}$), superbasic ($\bar{\mathcal{S}}$) and non basic ($\bar{\mathcal{N}}$) sets It is important to show explicitily the internal structure of matrix $\bar{A}$:

$$\bar{A} = (\,\bar{B} \mid \bar{S} \mid \bar{N}\,) = \begin{array}{|c|c|c|c|c|c|c|l} \multicolumn{1}{c}{m} & \multicolumn{2}{c}{t} & \multicolumn{2}{c}{s} & \multicolumn{2}{c}{(n-m-s)} & \\ \hline A^{\mathcal{B}} & A^{\mathcal{C}} & 0 & A^{\mathcal{S}} & 0 & A^{\mathcal{N}} & 0 & m \\ \hline T^{\mathcal{B}} & T^{\mathcal{C}} & E^{\mathcal{C}} & T^{\mathcal{S}} & E^{\mathcal{S}} & T^{\mathcal{N}} & E^{\mathcal{N}} & t \\ \hline \end{array} \tag{5}$$

The symbols $\bar{\mathcal{B}} = \mathcal{B} \cup \mathcal{C}$, $\bar{\mathcal{S}}$ and $\bar{\mathcal{N}}$ stands for the sets of indices of basic, superbasic and nonbasic variables respectively. These symbols, used as supraindex of matrices A and T, represent the matrix formed with the columns of A or T associated with variables in this sets. The column division of the basic matrix is the same proposed by Kennington & Helgason in [8] for the linear problem. Columns denoted by $\mathcal{B}$ are called *key columns* and are associated to the arcs of a spanning tree of the graph $\mathcal{G}$. Columns denoted by $\mathcal{C}$ are called *nonkey columns* and can be linked to basic arcs not belonging to the spanning tree, or to slacks. Matrix $E^{\mathcal{SUP}}$ denotes a matrix of t rows whose columns are the unitary vectors e_i of $\mathbb{R}^t$ associated to the slacks z_i with $(n+i) \in \mathcal{SUP} = \{\mathcal{B}, \mathcal{C}, \bar{\mathcal{S}}, \bar{\mathcal{N}}\}$.

Kennington & Helgason show in [8] the procedure to compute systems $\bar{B}y = z$ and $z' = y'\bar{B}$, each one of dimension $m + t$ solving systems of equations with matrix $A^{\mathcal{B}}$, that can be efficiently computed exploiting the data structure of the spanning tree, and systems of equations with coefficient matrix $Q = (\, T^{\mathcal{C}} \mid E^{\mathcal{C}} \,) - T^{\mathcal{B}} A^{\mathcal{B}^{-1}} (\, A^{\mathcal{C}} \mid 0 \,)$, of dimension $t \times t$. Matrix Q is called the *working basis*, and plays a central role. These procedures are applied in [8] to solve the linear network flow problem with linear side constraints with an specialized simplex algorithm . The authors of this paper have extended the method developed by Kennington & Helgason for the linear problem to the nonlinear case. In this extension, the management of the basic matrix $\bar{B}$ introduced in [8] and the special structure of the constraint matrix $\bar{A}$ has been used to allow the treatment of the side constrains within the general framework of the pure nonlinear network flow algorithms. The algorithm proposed by the authors has been presented in [6]. An implementation of this algorithm has been coded in FORTRAN. In the following sections the main features of the resulting program, called NOXCB, are described, together with the very first computational results obtained.

2. Computation of a feasible solution.

Program NOXCB finds a feasible solution for NNS in the following way :

1. (*Phase 0*) Find $\hat{x}$ feasible for the network equations ($A\hat{x} = r$)
2. (*Phase 1*) Define $\mathrm{I} = \{i : T^i\hat{x} > b_i\}$
3. Add excess variables to the side constrains : $Tx + \mathbb{1}z - \mathbb{1}f = b$
4. Define the starting point $\hat{y}$ as :

$$\hat{y}' = (\ \hat{x}' \ \mid \ \hat{z}' \ \mid \ \hat{f}'\ ),\ \hat{z}_i = \begin{cases} b_i - T^i\hat{x} & i \notin \mathrm{I} \\ 0 & i \in \mathrm{I} \end{cases},\ \hat{f}_i = \begin{cases} 0 & i \notin \mathrm{I} \\ T^i\hat{x} - b_i & i \in \mathrm{I} \end{cases}$$

5. Solve :

$$\begin{array}{ll} \min & \sum_{i\in \mathrm{I}} f_i \\ \text{subj. to :} & Ax + e_I a \qquad\qquad = r \\ & Tx \qquad + \mathbb{1}z - \mathbb{1}f = b \\ & 0 \le x \le u\ ;\ z \ge 0\ ;\ a = 0\ ;\ f \ge 0 \end{array}$$

with the feasible initial point $\hat{y}$.
If $\sum_{i\in\mathrm{I}} f_i = 0$ at the optimal point y^* then $y^{*\prime} = (\ x^{*\prime} \ \mid \ z^{*\prime}\ )$ is feasible for NNS.

Remarks :

1) Step **1.**, called here phase 0, consists on the phase 1 of a pure linear network flow problem. It is solved with the LEXA routines (Nabona, [11]).
2) The problem stated in step **5.** is a linear network flow problem with linear side constraints. It is solved with an implementation of the algorithm proposed by Kennington & Helgason (see Heredia & Nabona [7])

3. The algorithm

The structure of the proposed algorithm is outlined in this section. Assume that in each iteration we have the following :

1) A feasible vector y : $\bar{A}y = \bar{r}$, $0 \leq x \leq u$, $0 \leq z$, $\bar{r}' = (\, r' \;\; | \;\; b' \,)$.
2) $f(x)$ and $g(x)' = \left(g^{\bar{\mathcal{B}}'} \;\; | \;\; g^{\bar{\mathcal{S}}'} \;\; | \;\; g^{\bar{\mathcal{N}}'} \right)$
3) The tree structure of the matrix $A^{\mathcal{B}}$.
4) The product form of the inverse Q^{-1}
5) The Lagrange multipliers π satifying : $\pi' = g^{\bar{\mathcal{B}}'}\bar{B}^{-1}$
6) The reduced gradient $g_z = Z'g$, $g_z \in \mathrm{IR}^s$:
7) Depending on the method selected for the minimization on the null space, the Cholesky factors $R'R$ of a quasi-Newton aproximation to the reduced hessian H_z, $H_z = Z'HZ$, $H_z \in \mathrm{IR}^{s\times s}$, $H = \nabla^2 f(x)$

The evaluation of the Lagrange multipliers $\pi' = (\, \pi^{1'} \;\; | \;\; \pi^{2'} \,) = \left(g^{\mathcal{B}'} \;\; | \;\; g^{\mathcal{C}'} \right) \bar{B}^{-1}$ and the reduced gradient can take profit of the procedures explained in [8] and [6].

We consider that phase 0 and phase 1 have been performed, providing a feasible vector y. The sets $\mathcal{C}$ and $\bar{\mathcal{S}}$ with a subindex x or z denote the subset of indices associated with arcs or slacks respectively. The proposed algorithm goes through the following steps :

```
0. INITIALIZATIONS
1. If $\|g_z\| \geq$ TGR go to 3..
2. CHANGE OF THE ACTIVE SET
    2.1. Select, if possible, $q \in \bar{\mathcal{N}}_z$ to leave $\bar{\mathcal{N}}$. Go to 2.4.
    2.2. Select, if possible, $q \in \bar{\mathcal{N}}_x$ to leave $\bar{\mathcal{N}}$. Go to 2.4.
    2.3. Go to 11.
    2.4. Update $\bar{\mathcal{N}}$,$\bar{\mathcal{S}}$,$\pi$,$g_z$ and $R$ if QNM.
3. COMPUTING A FEASIBLE DESCENT DIRECTION
    3.1. Find $p_z$
    3.2. Find $p' = \left( p^{\bar{\mathcal{B}}'} \;|\; p^{\bar{\mathcal{S}}'} \;|\; p^{\bar{\mathcal{N}}'} \right)$
4. LINESEARCH
    4.1. Find $\bar{\alpha} = \min\{\alpha^{\bar{\mathcal{B}}}, \alpha^{\bar{\mathcal{S}}}\}$. If $\bar{\alpha} = 0$, set $\alpha^* = 0$ and go to 5.
    4.2. Estimate $\alpha^*$ such that $f(y + \alpha^* p) = \min_{0<\alpha<\bar{\alpha}} f(y + \alpha p)$
    4.3. Update : $g_z$,$\pi$ and $R$ if QNM.
5. IF $\alpha^* < \bar{\alpha}$ , GO TO 8.
6. BASIS CHANGE : VARIABLE $p$ LEAVE $\bar{B}$
    6.1. If $\alpha^* = \alpha^{\bar{\mathcal{S}}}$ go to 7.
    6.2. Select, if possible, $q \in \bar{\mathcal{S}}_z$ to become basic.
         Otherwise, select $q \in \bar{\mathcal{S}}_x$ to become basic.
    6.3. If $\bar{\alpha} = \alpha^{\mathcal{B}}$ : $p \in \mathcal{B} \leftrightarrow q \in \bar{\mathcal{S}}$ , update : $\mathcal{B}$,$\bar{\mathcal{S}}$, $Q^{-1}$, $g_z$,$\pi$ and $R$ if QNM.
    6.4. If $\bar{\alpha} = \alpha^{\mathcal{C}_x}$ : $p \in \mathcal{C}_x \leftrightarrow q \in \bar{\mathcal{S}}$ , update : $\mathcal{B}$,$\bar{\mathcal{S}}$, $Q^{-1}$, $g_z$,$\pi$ and $R$ if QNM.
    6.5. If $\bar\alpha = \alpha^{\mathcal{C}_z}$ : $p \in \mathcal{C}_z \leftrightarrow q \in \bar{\mathcal{S}}$ , update : $\bar{\mathcal{B}}$,$\bar{\mathcal{S}}$, $Q^{-1}$, $g_z$,$\pi$. and $R$ if QNM.
7. ELIMATION OF DEGENERATED S.B. VARIABLES
    7.1. Transfer degenerated variables from $\bar{\mathcal{S}}$ to $\bar{\mathcal{N}}$
    7.2. Update $g_z$. If QNM, update $R$
8. ELIMINATION OF SUPERBASIC SLACKS IF $\bar{\mathcal{S}}_z \neq \emptyset$
    [8.1] $q \in \bar{\mathcal{S}}_z \leftrightarrow p \in \mathcal{C}_x$
    [8.2] Update $\mathcal{C}$,$\bar{\mathcal{S}}$,$g_z$,$\pi$ and $R$ if QNM.
9. REINVERSION OF $Q$.
10. GO TO 1.
11. STOP : OPTIMAL SOLUTION FOUND.
```

3.1. Change of the active set

The evaluation of the Lagrange multipliers $\sigma' = g^{\bar{\mathcal{N}}'} - \pi'\bar{N}$ is easily carried out taking into account the partition of $g^{\bar{\mathcal{N}}'} = (\, g_x^{\bar{\mathcal{N}}'} \;\mid\; g_z^{\bar{\mathcal{N}}'} \,)$, $\bar{N}$ and $\pi' = (\, \pi^{1'} \;\mid\; \pi^{2'} \,)$. In step **2.** slacks are priced out before than arcs. If a slack is found to be a good candidate, then it becomes superbasic, but only temporarily, because it is interchanged with a nonkey arc at the end of the current iteration (step **8.**). This strategy tries to improve the stability of the solution of the systems of equations with coefficient matrix Q introducing in it as many identity columns as possible.

3.2. Descent directions

A descent direction on the null space p_z is obtained solving the system :

$$H_z p_z = -g_z \tag{6}$$

Two different techniques to solve system (6) have been implemented :

1) A truncated Newton method (TNM).
2) A quasi–Newton method (QNM).

The truncated–Newton method follows the strategy exposed by Dembo & Steihaug in [4]. It is based on the solution of system (6) by a *conjugated gradient* (CG) method. With regards to the use of the second derivatives, the CG algorithm only needs the aproximation to the Hessian H to compute the products $Z'HZd$, where the vectors d are the directions generated by the CG method in each iteration. This products are computed actually using a forward finite difference to approximate the product $H(Zd)$. Proceeding in this way it is neither necessary to compute, nor to store, any approximation to Hessian matrix.

When the quasi–Newton method is selected, program NOXCB follows the methodology exposed by Murtagh & Saunders in [9]. The Cholesky factors R of an approximation to the reduced Hessian ($R'R \approx H_z$) must be stored, updated and retriangularized whenever a change in the matrix H_z occurs. Matrix H_z is modified when $\bar{\mathcal{B}}$ or $\bar{\mathcal{S}}$ change and when a nonzero step is performed in the real variables x. In the first case, after the addition or removal of certain number of rows/columns of R, the factor R is retriangularized via succesive Givens rotations. When a change in the variables is made , factors $R'R$ are updated with a complementary DFP formula. In order to update the factor R it is necessary to perform a set of operations involving matrices $\bar{B}$, Z and $\bar{S}$ wich can take advantage of the structure exhibited by $\bar{A}$ in (5).

Once p_z has been found, a feasible descent direction $p' = \left(p^{\bar{\mathcal{B}}'} \;\mid\; p^{\bar{\mathcal{S}}'} \;\mid\; p^{\mathcal{N}'} \right)$ can be obtained making $p = Zp_z$, that can be efficiently computed following the procedures exposed in [6].

3.3. Linesearch

If $\alpha^{\bar{\mathcal{B}}}$ and $\alpha^{\bar{\mathcal{S}}}$ denote the maximal step length allowed by variables in $\bar{\mathcal{B}}$ and $\bar{\mathcal{S}}$, respectively, a limited linesearch with $0 \le \alpha \le \bar{\alpha} = \min\{\alpha^{\bar{\mathcal{B}}}, \alpha^{\bar{\mathcal{S}}}\}$ must be executed in order to the basic and superbasic bounds not to be violated by the iterated point. In fact, we must solve a nonlinear problem with simple constraints.

Bertsekas in [1] introduces a method to solve this kind of problems where more than one variable can be set to one of its bound simultaneously. Program NOXCB follows the scheeme presented by Toint & Tuyttens in [12] to execute this special linesearch. In particular, the present version of NOXCB makes use of the *quasi–active bounds* strategy for finding the search direction p. In this strategy the algorithm for finding the descent direction p acts only over the components of p associated with superbasic variables without quasi–active bounds, and takes for the other components the reduced gradient direction. We will refer to this linesearch as LSBE. When a Bertsekas step cannot be performed, a cubic fit with safeguards is made between $\alpha = 0$ and $\alpha = \bar{\alpha}$. A cubic fit is also used when $\bar{\mathcal{S}} \equiv \bar{\mathcal{S}}_z$, that is, the only superbasic variable existing is a slack. We will refer to this linesearch as LSCF. Both subroutines, LSBE and LSCF provide, not only and estimation of α^*, but also the new values of y, $f(x)$ and $g(x)$.

3.4. Pivot operations

The simultaneous presence in the network of arcs $\mathcal{C}_x$ and arcs $\bar{\mathcal{S}}_x$ makes the interpretation and the management of the pivot operation more difficult than in the pure nonlinear network flow problems. In presence of side constraints, it is possible for an arc $p \in \bar{\mathcal{B}}$ to change its value without forming cycle with any superbasic arc $q \in \bar{\mathcal{S}}_x$. The nonkey arcs can also change its value without being in a superbasic cycle. This special behaviour of the arcs of the spanning tree must be taken into account in the pivot operations.

4. Update of Q^{-1}

Matrix Q^{-1}, the inverse of the working basis is found once at the beginning of the first iteration and updated when a change in the set $\bar{\mathcal{B}}$ occurs. Kennington & Helgason explain in [8] the way to update matrix Q^{-1} after a simplex pivot in the linear network case. In the nonlinear problem the procedure to update Q^{-1} when variables $p \in \bar{\mathcal{B}}$ and $q \in \bar{\mathcal{S}}$ are interchanged is similar as in the linear case. The only difference is that the algorithm implemented can reorder the nonkey columns when $q \in \bar{\mathcal{S}}_z$. This reordering forces a new update of the type $Q_{i+1}^{-1} = Q_i^{-1} P_{pk}$, with P_{pk} an elementary permutation matrix. Reinversion takes place after a certain number of updates. Matrix Q^{-1} is stored in product form as an eta file with a set of complementary vectors. This eta file holds the coefficients of the eta columns of the last reinversion and the information of all the updatings made since the last reinversion. This information is either coefficients of column/row eta or two numbers denoting a nonkey column permutation. The reinversion routines implements the algorithm P^3 proposed by Hellerman & Rarick in [5] and partial pivoting to ensure numerical stability.

5. Input files and user-written subroutines

In order to define the NNS problem, some information must be given to program NOXCB at the beginning of the optimization process. This information consists on an input data file and a subroutine. The data file contains :

1) The value of m, n, t and the number of non zero elements of the side constrains .
2) Two vectors describing the oriented network.
3) The vector of arc capacities
4) The supply/demand vector
5) The right-hand side vector of the side constraints.
6) The vectors containing the sparse representation of the side constrains .

The actual version of NOXCB stores the supply/demand vector, the capacity vector and the elements of the side constraints as integer arrays. The user must provide a subroutine that computes the value of the objective function $f(x)$ and the gradient vector $g(x) = \nabla f(x)$ at a given point x.

A set of tolerances and control parameters are stored in a COMMON area. Its default values are read from an input file that users can easily modify . Some other operational parameters are requested interactively at the beginning of the execution.

6. Computational results

6.1. Short Term Hydro-Thermal Coordination of Electricity Generation

The developement of program NOXCB was motivated by the work of the authors in the field of the Hydro-Thermal Coordination of Electricity Generation. This problem deals with the study of a reservoir system for hydroelectric generation that must supply certain amounts of electric energy at a time intervals of a given period of time. The evolution in time of the reservoir system is modelled by dividing the whole period of time into a certain number of intervals. At each interval the state of each reservoir is described by the initial and final volume of water, the water inflows (natural inflow and water discharges from upstream reservoirs) , and the water discharges. A replicated network representing the reservoir system at each interval can be constructed. The nonlinear objective function is the sum of thermal generation at each interval, being the thermal generation equivalent to the load minus hydro generation. Hydro generation is the sum of generations at each reservoir, which is a nonlinear function of initial and final volume and water discharges. Side constraints come either from hydro generation limitations or from irrigation constraints. Generation limits are generally imposed at each interval. The sum of generations at each reservoir (here simplified to be a linear function of water discharges) must be within limits. Irrigation constraints are linear combinations of some discharges of reservoirs, at the same or in several time intervals, that must also be within limits.

6.2. Numerical results

The numerical results for a set of real Hydro-Thermal Coordination problems are presented in this last section. These problems represents several reservoirs systems located in Spain. Times are in seconds for executions on a VAXstation 3200 (VEGA) and a VAX 6410 (NEFTIS).

			#	%	it.	time	it.	time	it.	time
	nodes	arcs	s.c.	s.c.	Ph.0	Ph.0	Ph.1	Ph.1	Ph.2	Ph.2
TER01	49	228	24	49	82	2.9	16	0.3	171	10.7
TER02	49	228	5	10	82	1.2	2	2.2	320	28.3
EBRE01	79	390	26	33	282	8.1	1	0.2	601	52.1
EBRE02 (NEFTIS)	79	390	4	5	282	8.1	1	0.2	585	42.5

Due to the high cost of the evaluation of $f(x)$ for this kind of problems, QNM seems to be preferable than TNM, as shown in the next table :

	it. Ph.2	time Ph.2	time/ iter.	calls $f(x)$	calls ite.	time $p^{\bar{S}}$	
TER01	161	37.8	0.235	440	2.73	44%	TNM
(VEGA)	168	30.2	0.180	313	1.86	13%	QNM
EBRE01	1443	399.0	0.277	7272	5.04	63%	TNM
(NEFTIS)	1403	136.5	0.097	2796	1.99	9%	QNM

The use of a Bertsekas like linesearch does not seem to improve the behaviour of program NOXCB. The following table shows the results of the numerical experiments with (LSBE) and without (LSCF) the Bertsekas linesearch :

	it. Ph.2	time Ph.2	time iter.	calls $f(x)$	calls ite.	time α^*	# $\bar{S}$	
TER01	172	27.7	0.161	322	1.87	51%	8	LSBE
(VEGA)	198	30.5	0.154	355	1.79	49%	8	LSCF
TER02	810	244.1	0.301	1849	2.28	58%	45	LSBE
(VEGA)	908	239.0	0.263	1686	1.86	49%	45	LSCF
EBRE01	1545	159.4	0.103	3592	2.32	70%	18	LSBE
(NEFTIS)	1403	136.6	0.097	2796	1.99	64%	18	LSCF

The current version of program NOXCB does not produce total execution times lower than those of the general purpuse program MINOS ([10]). However, additional computational experiments have still to be performed to compare the behaviour of both programs starting at the same feasible point.

REFERENCES

[1] D.P. Bertsekas, “Constrained optimization and Lagrange multiplier methods” , (Academic Press, London, 1982).

[2] G.H. Bradley, G.G. Brown and G.W. Graves,“Design and implementation of large scale transshipment algorithms” Management Science 24 (1977) 1–34.

[3] R.S. Dembo,“A primal truncated newton algorithm with application to large-scale nonlinear network optimization”, Mathematical Programming Studies 31 (1987) 43–71.

[4] R.S. Dembo and T. Steihaug, "Truncated–Newton algorithms for large–scale unconstrained optimization", Mathematical Programming 26 (1983) 190–212.

[5] E. Hellerman and D. Rarick,"Reinversion with the preassigned pivot procedure", Mathematical Programming 1 (1971) 195–216.

[6] F.J. Heredia and N. Nabona,"Large scale nonlinear network optimization with linear side constraints", presented at the EURO XI, 11th European Congress on Operational Research, (Aachen, july 1991)

[7] F.J. Heredia and N. Nabona,"Programa FXCB de fluxos lineals en xarxes amb constriccions a banda lineals", research report RR 90/06 Facultat d'Informàtica de Barcelona, 1990.

[8] J.L. Kennington and R.V. Helgason , "Algorithm for network programming" , (John Wiley & Sons, New York, 1980).

[9] B.A. Murtagh and M.A. Saunders,"Large–scale linearly constrained optimization", Mathematical Programming 14 (1978) 41–72.

[10] B.A. Murtagh and M.A. Saunders,"MINOS 5.0 User's Guide." Dept. of Operations Research, Standford University, CA 9430, USA.

[11] N. Nabona, "Descripció del programa LEXA", research report of the Facultat d'Informàtica de Barcelona (to appear)

[12] Ph.L. Toint and D.Tuyttens, "On large scale nonlinear network optimization", Mathematical Programming 48 (1990) 125–159.