18	GANGES	1310	1681	7021	60191	-1.0958573612928e+05
19	GUB	930	3523	14173	92199	2.1851966989e+06
20	SEBA	516	1028	5252	39784	1.5711600000e+04

NOTES on the above: we have omitted extra right-hand side vectors from problems 5, 6, 7, and 16 and extra free rows from problems 3, 8, 13, 16, and 18. We also negated the cost coefficients in problems 11, 12, and 13. In their original form, these problems are usually maximized. In their modified form, all problems are to be minimized. Problem 25FV47 is sometimes called BP, and FFFFF800 is sometimes called POWELL. The names shown above are from the NAME line; the optimal values are from MINOS running on a VAX.

Notes from Michael Saunders describing experience with MINOS on problems 1-13 are available via the netlib request

```
send minos from lp/data
```


A Note on Interior-Point Methods for Linear Programming

Philip E. Gill†, Walter Murray†,
Michael A. Saunders†, J. A. Tomlin‡ and Margaret H. Wright†

1. Introduction

Within the past year, interest in linear programming has been intensified by the publication (Karmarkar, 1984) and discussion of an interior-point linear programming algorithm that is not only polynomial in complexity, but is also claimed to be *much faster* than the simplex method for practical problems.

In Section 2, we first examine the well known barrier-function approach to solving optimization problems with inequality constraints, and give a representation for the projected Newton search direction associated with applying a barrier transformation to a linear program. In Section 3, we state a formal equivalence between a projected Newton method and Karmarkar's (1984) algorithm. Section 4 gives some numerical results obtained with a preliminary implementation of a projected Newton barrier method. The implications of these results are discussed in Section 5. Further details about all aspects of this research are given in a technical report by Gill *et al.* (1985).

2. A Barrier-Function Approach

Barrier-function methods treat *inequality* constraints by creating a *barrier function*, which is a combination of the original objective function and a weighted sum of functions with a positive singularity at the constraint boundary. (We consider only the logarithmic barrier function, first suggested by Frisch, 1955.) As the weight assigned to the singularities approaches zero, the minimum of the barrier function approaches the minimum of the original constrained problem. Barrier-function methods require a strictly feasible starting point for each minimization, and generate a sequence of strictly feasible iterates. (For a complete discussion of barrier methods, see Fiacco, 1979; both barrier and penalty function methods are described in Fiacco and McCormick, 1968.)

Consider applying a barrier-function method to the following linear program:

$$\begin{array}{ll} \underset{x\in\Re^n}{\text{minimize}} & c^Tx \\ \text{subject to} & Ax=b, \qquad x\ge 0, \end{array} \tag{2.1}$$

where A is an $m\times n$ matrix with $m\le n$. The subproblem to be solved within a barrier-function method is:

$$\begin{array}{ll} \underset{x\in\Re^n}{\text{minimize}} & F(x)\equiv c^Tx-\mu\sum_{j=1}^{n}\ln x_j \\ \text{subject to} & Ax=b, \end{array} \tag{2.2}$$

where the scalar μ ($\mu>0$) is known as the *barrier parameter* and is specified for each subproblem. The equality constraints cannot be treated by a barrier transformation, and thus are handled

† Systems Optimization Laboratory, Department of Operations Research, Stanford University, Stanford, California 94305.
‡ Ketron Incorporated, 201 San Antonio Circle, Mountain View, California 94040.

directly. If $x^*(\mu)$ is the solution of (2.2), then $x^*(\mu) \to x^*$ as $\mu \to 0$, where x^* is a solution of (2.1) (see, e.g., Fiacco and McCormick, 1968).

In order to solve (2.2) by a projected Newton method, we assume that the current iterate x satisfies $Ax = b$. Let $D = \mathrm{diag}(x_j)$, $j = 1, \ldots, n$, and $e = (1, 1, \ldots, 1)^T$. Then p_B, the projected Newton barrier direction associated with the subproblem (2.2), is defined by

$$p_B = -(1/\mu) D r_B,$$

where π_B is the solution and r_B the optimal residual of the following linear least-squares problem:

$$\underset{\pi}{\text{minimize}} \ \|Dc - \mu e - DA^T\pi\|_2.$$

The next iterate of the projected Newton method is given by

$$\bar{x} = x + \alpha p_B$$

for some non-negative steplength α.

The barrier transformation and the associated Newton search direction can also be defined for linear programs with upper and lower bounds on the variables.

3. Relationship with Karmarkar's Projective Method

In the projective method (Karmarkar, 1984), the linear program is assumed to be of the special form

$$\begin{aligned} &\underset{x \in \Re^n}{\text{minimize}} && c^T x \\ &\text{subject to} && Cx = 0, \quad e^T x = 1, \quad x \geq 0. \end{aligned} \tag{3.1}$$

Let x_K^* denote a solution of (3.1). It is also assumed that $c^T x_K^* = 0$ and that $Ce = 0$. (These assumptions can always be assured by transforming the problem.)

Any strictly positive diagonal matrix D defines the following projective transformations, which relate any strictly feasible point x and the transformed point x':

$$x' = \frac{1}{e^T D^{-1} x} D^{-1} x, \qquad x = \frac{1}{e^T D x'} D x'. \tag{3.2}$$

In the projective method, given an iterate x, D is defined as $\mathrm{diag}(x_j)$, as in the barrier method. The next iterate in the *transformed* space is given by

$$\bar{x}' = x' - \alpha' r_K, \tag{3.3}$$

where $r_K = Dc - DC^T\pi_K - \phi e$ is the optimal residual of the linear least-squares problem

$$\underset{\pi, \phi}{\text{minimize}} \ \|Dc - (\, DC^T \quad e\,) \begin{pmatrix} \pi \\ \phi \end{pmatrix} \|_2.$$

The steplength α' in (3.3) is chosen to ensure strict feasibility of $\bar{x}'$ as well as descent in the transformed "potential function" (see Karmarkar, 1984, for details).

characters of lines stripped of trailing blanks).

It would be nice to provide more information about the test problems, such as further detail on their origins and comments about special structures they exhibit. As time permits, I hope to work with other COAL members on collecting such information and hope to make it available via netlib. This work is already benefitting from discussions with Bob Fourer.

The current index to `lp/data` follows:

```
===== LP/DATA index =====

To reduce transmission times, linear programming test problems
are stored in a compressed format; issue the netlib request

    send emps from lp/data

to obtain a Fortran 77 Subset program for expanding the test problems
into MPS-standard input form.  The program includes comments giving
test data.

The column and nonzero counts below exclude slack and surplus columns
and the right-hand side vector, but include the cost row.  We have
omitted other free rows and all but the first right-hand side vector,
as noted below.  The byte count is for the compressed file; it includes
a newline character at the end of each line.  These files start with
a blank initial line intended to prevent mail programs from discarding
any of the data.

The first 13 linear programming test problems are from the Systems
Optimization Laboratory at Stanford University (courtesy of Michael
Saunders).  The next 4 problems are from a tape that John Reid sent
me (David Gay) several years ago.  The last 3 were sent me by Linus
Schrage.
```

#	Name	Rows	Cols	Nonzeros	Bytes	Optimal Value
1	AFIRO	28	32	88	794	-4.6475314286e+02
2	ADLITTLE	57	97	465	3690	2.2549496316e+05
3	SHARE2B	97	79	730	4795	-4.1573224074e+02
4	SHARE1B	118	225	1182	8380	-7.6589318579e+04
5	BEACONFD	174	262	3476	17475	3.3592485807e+04
6	ISRAEL	175	142	2358	12109	-8.9664482186e+05
7	BRANDY	221	249	2150	14028	1.5185098965e+03
8	E226	224	282	2767	17749	-1.8751929066e+01
9	CAPRI	272	353	1786	15267	2.6900129138e+03
10	BANDM	306	472	2659	19460	-1.5862801845e+02
11	STAIR	357	467	3867	27435	-2.5126695119e+02
12	ETAMACRO	401	688	2489	21915	-7.5571522785e+02
13	PILOT	1442	3652	43220	278626	-5.5742017351e+02
14	25FV47	822	1571	11127	70477	5.5018458883e+03
15	CZPROB	930	3523	14173	92202	2.1851966989e+06
16	FFFFF800	525	854	6235	39637	5.5567996085e+05
17	SHELL	537	1775	4900	38049	1.2088253460e+09

Electronic Mail Distribution of Linear Programming Test Problems

David M. Gay

AT&T Bell Laboratories
Murray Hill, New Jersey 07974

Netlib is an experimental facility for distributing public-domain numerical software by electronic mail. Two machines, *research* at Bell Labs and *anl−mcs* at Argonne, currently provide *netlib* service. (This is natural, since *netlib* is the creation of Eric Grosse of Bell Labs and Jack Dongarra of Argonna National Laboratory.)

To use netlib, you send electronic mail to either *research!netlib* or *netlib@anl−mcs*, asking for what you want; if available, it will be sent back to you by return electronic mail. For example, to see what *research* currently offers via *netlib*, you would send the message "`send index`" to *netlib* at one of the following addresses:

Network	*netlib* address
USENET	`research!netlib`
CSNET	`netlib@anl-mcs.arpa`
ARPANET	`netlib@anl-mcs`
EDUNET	`netlib@anl-mcs.arpa`
ACSNET	`netlib@research`
BITNET	`netlib%anl-mcs.arpa@wiscvm`

The powers that be have consented to allow distribution of linear programming test problems via *netlib*. To see what problems are currently available, send the message "`send index from lp/data`" to *netlib*. The return mail that *research!netlib* would send you (at the time of this writing) is shown below. New problems will appear first on *research*, then (after an unpredictable length of time) on *anl−mcs*. (There is no guarantee that *netlib* service will continue to be available; on the other hand, there are presently no plans to withdraw this service.)

Anyone willing to make interesting linear programming test problems publicly available is encouraged to submit them for possible inclusion in *netlib*'s `lp/data` chapter. Contact David Gay electronically (`research!dmg`), by phone at (201) 582−5623, or in writing at the address shown above. You can save me some hassle by making sure the problems you send have a single right-hand side, a single (or no) BOUNDS set, and a single cost (free) row with signs reversed, if necessary, so that the LP is to be minimized. (Otherwise I will arbitrarily delete all but one right-hand side, free row, and BOUNDS set before putting the problems into `lp/data`. The obvious point of this exercise is to make as clear as possible what problem is to be solved.)

All of the LP test problems currently available from *netlib* are in MPS format. Source for portable programs that generate LP's would also be welcome.

Problems in MPS format can be large. To reduce transmission times, *netlib* sends a compressed version of the problems — and makes available a Fortran 77 program for expanding the problems back to MPS format. The compression and expansion involve no roundoff, so the problems should not be changed by the process. The compressed form contains only printing ASCII characters, and is broken into lines at most 72 characters long. There is one checksum character per line (sent somewhat later in a line full of checksums), so the expansion program is likely to warn you if a transmission error has occurred. Even with the checksum characters, problems are typically reduced to between 16% and 26% of their original size (where size is in

The new iterate $\bar{x}_K$ in the original parameter space is obtained by applying the transformation (3.2) to $\bar{x}'$, so that

$$\bar{x}_K = \frac{1}{e^T D(x' - \alpha' r_K)} D(x' - \alpha' r_K) = \gamma(x - \tilde{\alpha} D r_K),$$

where γ is chosen to make $e^T \bar{x}_K = 1$.

Let π_c be defined as the solution of the least-squares problem

$$\underset{\pi}{\text{minimize}}\ \|Dc - DC^T\pi\|_2,$$

and let

$$\mu_C = x^T(Dc - DC^T\pi_c).$$

If the projective method and the barrier method with $\mu = \mu_C$ are applied to problem (3.1) with the same initial point, and if the steplengths α and α' are suitably chosen, *the two algorithms generate identical sequences of iterates*. Thus, a special case of the barrier method would follow the same path as the projective method.

4. Numerical Results

In this section we illustrate the performance of a preliminary implementation of a projected Newton barrier algorithm on problems from an LP test set in use at the Systems Optimization Laboratory. All problems are in the form (2.1). To obtain constraints of the form $Ax = b$, any general inequality constraints are converted to equalities using slack variables. Details of the problems are given in Table 1. The value of "rows" refers to the number of general constraints, and "columns" to the number of variables, excluding slacks. The number "slacks" is defined above. The column "A" gives the number of nonzeros in the problem. This figure includes one for each slack but excludes the nonzeros in b and c.

The runs summarized in Tables 2 and 3 were made in double precision on an IBM 3081K (relative precision 2.2×10^{-16}). The source code was compiled with the IBM Fortran 77 compiler VS Fortran, using `NOSDUMP`, `NOSYM` and `OPT(3)`. Table 2 gives the number of iterations and CPU-seconds required by the primal simplex method, as implemented in the Fortran code *MINOS 5.0* (May 1985). The default values of the parameters were used throughout (see Murtagh and Saunders, 1983).

Convergence for each subproblem (2.2) is measured by $\|r\|$, where r is a scaled form of the reduced gradient, which must tend to zero for any fixed μ. Reduction of μ is controlled by two parameters: `RGFAC` defines an initial "target value" for $\|r\|$; and `MUFAC` defines the factor by which μ is reduced when $\|r\|$ reaches the target value. `RGFAC` and `MUFAC` should lie in the range $(0, 1)$ to be meaningful. For example, the values `RGFAC` $= 0.99$, `MUFAC` $= 0.25$ allow a moderate reduction in μ almost every iteration, while `RGFAC` $=$ `MUFAC` $= 0.001$ requests more discernible progress towards optimality for each subproblem, with a substantial reduction in μ on rare occasions.

Many runs of the barrier method were made, incorporating different choices for `RGFAC` and `MUFAC`. One aim was to find a set of values that could be used reliably on all problems. Table 3 summarizes the performance of the barrier method with `RGFAC` $= 0.1$ and `MUFAC` $= 0.1$. The second and third columns of the table give the number of iterations to obtain a feasible point and the total iterations required. The fourth column gives the total CPU time (in seconds) to solve the problem. The underlined digits in the fifth column show the correct figures in the objective function on termination. The final two columns indicate the degree of feasibility and optimality of the final point.

Table 1
Problem Statistics

Problem	Rows	Slacks	Columns	A	$\|x^*\|$	$\|\pi^*\|$
Afiro	27	19	32	102	$9.7\ 10^2$	$3.9\ 10^1$
ADLittle	56	41	97	424	$6.1\ 10^2$	$6.2\ 10^3$
Share2b	96	83	79	777	$1.8\ 10^2$	$3.8\ 10^2$
Share1b	117	28	225	1179	$1.3\ 10^6$	$7.7\ 10^1$
Beaconfd	173	33	262	3408	$1.6\ 10^5$	$1.2\ 10^2$
Israel	174	174	142	2443	$9.1\ 10^5$	$5.6\ 10^2$
BrandY	220	54	249	2202	$6.5\ 10^4$	$8.7\ 10^1$
E226	223	190	282	2768	$9.6\ 10^2$	$4.1\ 10^1$
BandM	305	0	472	2494	$1.5\ 10^3$	$3.0\ 10^1$

Table 2
Results from the primal simplex code *MINOS 5.0*

	Optimal	No scaling		
	objective	Phase 1	Total	Time
Afiro	−464.75314	2	6	0.5
ADLittle	225494.96	28	123	1.3
Share2b	−415.73224	59	91	1.3
Share1b	−76589.319	135	296	3.4
Beaconfd	33592.486	8	38	1.9
Israel	−896644.82	109	345	5.0
BrandY	1518.5099	176	292	4.9
E226	−18.751929	109	570	9.4
BandM	−158.62802	167	362	7.6

Table 3
Barrier method
No scaling, RGFAC = 0.1, MUFAC = 0.1

Problem	Phase 1	Total	Time	Objective	$\frac{\|b-Ax\|}{\|b\|}$	$\frac{\|D(c-A^T\pi)\|}{\|x\|\|\pi\|}$
Afiro	4	20	0.4	−464.75314	$3.6\ 10^{-12}$	$1.9\ 10^{-9}$
ADLittle	13	36	1.1	225494.96	$8.8\ 10^{-10}$	$2.4\ 10^{-10}$
Share2b	7	22	1.5	−415.73224	$2.1\ 10^{-8}$	$9.3\ 10^{-11}$
Share1b	11	66	4.9	−76589.319	$9.8\ 10^{-8}$	$4.5\ 10^{-11}$
Beaconfd	20	40	9.9	33592.486	$8.4\ 10^{-9}$	$4.5\ 10^{-11}$
Israel	17	54	22.6	−896644.82	$8.6\ 10^{-6}$	$9.4\ 10^{-12}$
BrandY	19	40	8.5	1518.5099	$3.8\ 10^{-8}$	$3.7\ 10^{-11}$
E226	18	45	9.8	−18.751929	$8.0\ 10^{-8}$	$1.4\ 10^{-10}$
BandM	19	41	9.3	−158.62802	$3.7\ 10^{-8}$	$5.6\ 10^{-11}$

(a) In the first place more concerned with intrinsic properties of algorithms ("self-comparison") than with cross-comparisons.

(b) I don't know to what extent, but I think it calls mainly for experiments which—while necessarily performed in computers—deal with issues which are independent of the particular hardware and software.

The implications of (a) and (b) are certainly not that there is no need for scientific standards with regard to test problems, reproducibility, etc.; but that referring to the guidelines of Crowder et al. seems to confuse the issue. There is probably no more sophisticated organization in existence than Bell Labs on such matters. I rather hope that along with the positive note in your letter concerning prospects of co-operation on test problems, there will also be some interaction on differentiation between basic scientific experimentation on MP algorithms and testing/experimentation which is part of product development. Clarification along such lines—which might involve further guidelines—could be a valuable outcome.

Sincerely,

Alex Orden
Professor of Applied Mathematics
University of Chicago

UNIVERSITY OF CHICAGO

26 July 1985

Dr. Karla Hoffman
Operations Research Division
National Bureau of Standards
Building 101, Room A415
Gaithersburg, Maryland 20899

Dear Karla:

The purpose of this letter is to examine the circumstances in the relationship between the NBS/COAL group and Bell Labs concerning experimentation on the Karmarkar algorithm. Although the first line of the letter of July 18th from Boggs, Jackson, and yourself refers to a *brief* report, I am grateful that in fact the letter plus attachments provides a good review of what has occurred.

When I read the letter I felt puzzled by the turn of events wherein Bell labs was initially receptive, but later rejected collaboration on computer experiments which could have been useful to them. As the "shock" wore off I began to wonder whether my being surprised implied that there was something to learn from the episode. A question which came to mind was:

> *Q*: Are we muddying the waters by casually assuming that standards for proper experimentation on algorithms should be about the same when a particular algorithm is being investigated as when alternative algorithms for solving some class of problems are being compared?

To indicate that the assumption stated in *Q* may or may not be sensible I will refer to *self-comparison*—meaning (experimental) determination of properties of an algorithm by itself—and to *cross-comparison*.

In either case the main objective may be in the basic math./science domain or in that of application-in-practice/proprietary knowledge. In the latter category a "product evaluation" may consist primarily of self-comparison (when an algorithm has no close competitors), or of cross-comparison when it does. Similarly, on the academic side, one may investigate properties of an algorithm per se, or differences between algorithms.

Essentially the paper by Crowder, Dembo, and Mulvey ("On Reporting Computational Experiments With Mathematical Software")—to which the NBS/COAL group has frequently referred as a guide to proper computer experimentation on algorithms—does not distinguish between self-comparison and cross-comparison; nor does it between experimentation for basic scientific purposes and those for effectiveness in practice. Although there are indications in the paper that it is directed toward cross-comparison, the material which actually presents guidelines is written completely as though there is no difference at all between self-comparison and cross-comparison.

Admittedly the real world jumbles together self- and cross-comparison, and also basic research and product development. The situation with which you have dealt at Bell Labs is a shining example. Nevertheless I feel that at least insofar as you represent the Math Programming Society—which is oriented mainly toward basic math/science issues—the distinctions above need to be made, and are relevant to the situation.

I am dubious with regard to the paper by Crowder et al., not only on the matter of self-comparison *v.* cross-comparison, but also whether it is a good set of guidelines for the computer experimentation side of basic research in mathematical programming. That area of experimentation is

In all cases, the number of iterations required by the barrier algorithm appears to be qualitatively similar to that reported for various implementations of the projective method (*cf.* Tomlin, 1985, and Lustig, 1985). The computation time in two thirds of the examples is comparable to that required by the simplex method.

When tried on a second test set of three models characterized by extreme degeneracy, the barrier algorithm became relatively less efficient than the simplex method as problem size increased—probably because of the increasing cost of solving the least-squares subproblem in the barrier method. However, it is obvious that for certain structures in A, the relative increase in cost of the linear algebra will remain constant as problem size increases, and that the performance of the barrier algorithm will consequently *improve* relative to the simplex method on such problems as size increases.

Because of the properties of the iterative sequence generated by the barrier method, a "nearly optimal" solution can be obtained by early termination. In contrast, it is well known that early termination of the simplex method does not necessarily produce a "good approximation" to the optimal solution. (This observation emphasizes the fundamental difference in the iterative sequences generated by a combinatorial algorithm like the simplex method and a nonlinear algorithm like the barrier method.) Our experiments indicate that almost one-half the work of the barrier algorithm can be saved by terminating early, *if an inaccurate solution is acceptable.* As many authors have noted, this suggests the possibility of using a barrier algorithm to attempt to identify the correct active set, and then switching to the simplex method (say) to obtain the solution.

5. Conclusions

Our experience with the barrier method suggests several conclusions.
On the positive side:

- Substantial computational evidence indicates that for *some* non-trivial linear programs, a general barrier method can be comparable in speed to the simplex method;
- The barrier method is likely to be faster—perhaps substantially so—than the simplex method for problems in which the least-squares subproblems can be solved rapidly. Furthermore, since we deliberately did not attempt to obtain maximum efficiency of the barrier method on each problem, there is much scope for further "tuning" of the algorithm on particular problem classes;
- The mathematical and qualitative relationship between the projective and barrier methods places this approach to linear programming in a well understood context of nonlinear programming, and provides an armory of known theoretical and practical techniques useful in convergence analysis and implementation.

On the negative side:

- The barrier method has not been consistently faster than the simplex method on general unstructured problems, and has been considerably slower on certain examples. Its efficiency relative to the simplex method seems to decrease with size on some problem classes;
- "Nonlinearizing" a linear problem leads to difficulties in developing a robust general-purpose algorithm. Furthermore, nonlinear algorithms typically display wide variations in performance, depending on the selection of various parameters.

Finally, we make the following general observations:

- Most qualitative aspects of Karmarkar's projective method can be found in the projected Newton barrier method;
- No attempt has been made to date to obtain a proof of polynomial complexity for any version of the barrier algorithm;

- The efficiency of the projective and barrier methods depends critically on fast, stable techniques for solving large-scale least-squares problems. Modern sparse-matrix technology is absolutely crucial in any further development of these methods for large-scale linear programming;
- There is much promise for interior-point approaches to linear programming, particularly for specially-structured problems.

Acknowledgement

The research of the Stanford authors was supported by the U.S. Department of Energy Contract DE-AM03-76SF00326, PA No. DE-AT03-76ER72018; National Science Foundation Grants DCR-8413211 and ECS-8312142; the Office of Naval Research Contract N00014-85-K-0343; and the U.S. Army Research Office Contract DAAG29-84-K-0156.

References

Fiacco, A. V. (1979). "Barrier methods for nonlinear programming", in *Operations Research Support Methodology* (A. Holzman, ed.), pp. 377–440, Marcel Dekker, Inc., New York.

Fiacco, A. V. and McCormick, G. P. (1968). *Nonlinear Programming: Sequential Unconstrained Minimization Techniques*, John Wiley and Sons, New York and Toronto.

Frisch, K. R. (1955). The logarithmic potential method of convex programming, Memorandum of May 13, 1955, University Institute of Economics, Oslo, Norway.

Gill, P. E., Murray, W., Saunders, M. A., Tomlin, J. A. and Wright, M. H. (1985). On projected Newton barrier methods for linear programming and an equivalence to Karmarkar's projective method, Report SOL 85-11, Department of Operations Research, Stanford University, Stanford, California.

Karmarkar, N. (1984). A new polynomial-time algorithm for linear programming, *Proceedings of the 16th Annual ACM Symposium on the Theory of Computing*, pp. 302–311.

Khachiyan, L. G. (1979). A polynomial algorithm in linear programming, *Doklady Akademiia Nauk SSSR Novaia Seriia* **244**, pp. 1093–1096. [English translation in *Soviet Mathematics Doklady* **20**, (1979), pp. 191–194.]

Lustig, I. J. (1985). A practical approach to Karmarkar's algorithm, Report SOL 85-5, Department of Operations Research, Stanford University, California.

Murtagh, B. A. and Saunders, M. A. (1983). *MINOS 5.0* user's guide, Report SOL 83-20, Department of Operations Research, Stanford University, California.

Tomlin, J. A. (1985). An experimental approach to Karmarkar's projective method for linear programming, Manuscript, Ketron Inc., Mountain View, California.

be "sanitized" before they are disseminated. As you know, COAL has tried previously to perform this task, but has been unsuccessful. This task has thus been left to dedicated individuals who collected test problems for their own use and were willing to shoulder the burden of dissemination. Although some test problem collections have been created in this way, they do not always contain the range of problems desired, nor is it always true that they are well-publicized and readily available. This makes it difficult to perform high quality testing of new ideas.

There is hope, however. Even though Bell Labs does not feel the time is right for a cooperative test effort, they graciously agreed to help create a basic set of LP test problems and disseminate them over their experimental software distribution system, netlib. In fact, during the special session on the new method that was held at the Boston Meeting of TIMS/ORSA, D. Gay of AT&T Bell Labs announced his willingness to collect, reformat if necessary, and add new test problems to the collection. This is an offer that is too good to refuse, and we will be pursuing it with Bell Labs in the near future, perhaps at the Boston Meeting of the Mathematical Programming Society. We will keep you informed of our progress in this matter.

Sincerely,

Paul T. Boggs   Karla L. Hoffman   Richard H.F. Jackson the proposal and transmitted it to A. Fronistas, AT&T's designated liaison for this test effort, on 7 January 1985. Copies were sent to you and to other prominent members of the professional community. In the proposal, we addressed ourselves to scientific thoroughness in the design of the experiment and the subsequent publication of the results. To guarantee the scientific community that the evaluation procedures met the highest standards of the profession, we were careful to follow the guidelines set down in 1979 by the COAL and published in Crowder, H.P., Dembo, R.S., and Mulvey, J.M., "On reporting computational experiments with mathematical software," *ACM TOMS*, Vol. 5, No. 2, June 1979, pp. 193-203. In subsequent phone conversations with the AT&T representatives, we attempted to assure them that we would protect AT&T's proprietary interests in the method and the code they had developed. We proposed signing non-disclosure agreements, treating any documents received as "confidential," and/or performing the experiment at their site. We were open to other suggestions as well.

As a result of the publication of our proposal, project participants became engaged in a number of other related activities. R. Jackson was interviewed by a reporter from the Wall Street Journal for an article on Karmarkar's method that appeared 5 May 1985. S. Gass and R. Jackson were asked by L. Bodin, the chairman of ORSA's Technical Section on Education, to organize a session on efforts to test the new method that was held at the Boston Meeting of ORSA. P. Boggs and R. Jackson were interviewed by a reporter from the IEEE Spectrum for an article that will appear shortly. R. Jackson was invited by W. Stewart, editor of ORSA's Computer Science Technical Section Newsletter, to prepare an article on some of the philosophical issues that had been raised at the Dallas session on Karmarkar's method. P. Boggs, K. Hoffman, and R. Jackson prepared an ARPANET announcement on the proposed test effort, which generated other ARPANET communications. And, of course, the three of us have received numerous phone calls from interested members of the professional community seeking information on the new method and on the progress of our test effort.

AT&T's response to our proposal came in the enclosed letter from M. Garey and D. Gay, dated 24 April 1985. For essentially two reasons, our proposal did not seem feasible to them at this time. The first reason has to do with their concerns over the handling of company proprietary information. As mentioned above, however, we believe there are many ways to surmount this problem, and that a concerted effort among the involved parties could have been successful. The second problem was that they consider their code to be too experimental to be subjected to the type of evaluation we had proposed. For our part, we intended only to test the code they used in reporting speeds of 50 to 100 times faster than MPSX370. If their code is too experimental for an independent body to replicate their experiments, we feel that it was probably dangerous to report comparative speeds in the first place. In fact, it was because of these beliefs that our proposal did not include any comparative testing. The guidelines referenced above are clear on the amount and type of testing that is required to support comparative claims, and it is much more than we had in mind.

Although this attempt at a cooperative test effort was not successful, the experience illustrates the continuing need for the existence of a group like COAL to educate the community with regard to availability of testing methodologies, and to continue to ensure the existence of both sound techniques for testing mathematical software and large sets of documented test problems. Indeed, had there existed a well-documented, readily available set of linear programming test problems, AT&T might have been able to report on this test set initially, and perhaps none of the events discussed here would have occurred.

The creation, updating, maintenance, and dissemination of a collection of test problems is a grubby, unrewarding, and expensive exercise. This is especially true for large-scale problems, not simply because they are large, but also because they frequently arise in practice, are proprietary, and must

EMP: AN EXPERT SYSTEM FOR MATHEMATICAL PROGRAMMING

Klaus Schittkowski

EMP is an interactive programming system that supports model building, numerical solution and data processing of mathematical programming problems, i.e. of problems in the form

```
minimize      f(x)
subject to    g(j,x)  =  0 ,  j=1,...,r
              g(j,x)  >= 0 ,  j=r+1,...,m
              a  <=  x  <=  b
```

where x is an n-dimensional vector and where all problem functions are continuously differentiable or are at least composed of continuously differentiable functions. Various options are available in EMP to facilitate the formulation of problem functions. The objective function e.g. may be a linear or quadratic function, a data fitting function, a sum or maximum of functions, or a general function without a structure that could be exploited. More precisely the following mathematical models are available for facilitating the formulation of objective function and exploiting special structures mathematically whenever possible:

- Linear function
- Linear least squares function
- Quadratic function
- L1-data fitting with general nonlinear functions
- L1-data fitting with one parameter estimation model function
- L2- or least squares data fitting with general nonlinear functions
- L2- or least squares data fitting with one parameter estimation model function
- Maximum-norm data fitting with general nonlinear functions
- Maximum-norm data fitting with one parameter estimation model function
- Sum of nonlinear functions
- Sum of 'simple' nonlinear functions differing at most by an index
- Maximum of nonlinear functions
- Maximum of realizations of a parameter estimation model function
- General function

Independently from the objective function, the user has to declare the constraints in form of a sequence of linear or nonlinear functions, respectively. In both cases it is possible to proceed from two-sided bounds for the restrictions. The input of 'simple' constraints is facilitated, if they differ at most by an index.

For objective function and constraints, the input of quadratic or linear functions reduces to definition of some vectors and matrices, respectively, where sparsity can be exploited. Gradients of nonlinear and nonquadratic functions are approximated numerically, but can also be provided by the user in analytical form. Further mathematical models are in preparation, e.g. multicriteria optimization.

Only the problem relevant data need to be provided by a user in an interactive way. Functions must be defined by sequences of FORTRAN statements addressing a numerical value to a user-provided function name. All generated problems are stored in form of a data base system, so that they are easily retrieved, modified, or deleted on request. EMP selects a suitable mathematical algorithm and writes a complete FORTRAN source program. The system executes this program and stores the numerical results in the data base, so that they are available for further processing. Since individual names for functions and variables can be provided by a user, it is possible to get a problem dependable output of the achieved solution.

The user has the option to choose between two solution levels. The standard level will execute a sequential quadratic programming algorithm or a suitable variant to take advantage of special problem structures. If this method fails or if a suitable starting point is not available, a user could proceed to another solution level which is based on an implementation of the ellipsoid-method.

The user will be asked whether he wants to link the generated FORTRAN program with some of his own files or whether he wants to insert additional subroutines, declaration and executable statements to formulate the problem. Moreover it is possible to provide keywords and to define 'experience fields', so that the system is capable to learn from previous solution ways. A failure analysis explains some reasons for possible false terminations and proposes remedies to overcome numerical difficulties.

All actions of EMP are controlled by self-explained commands which are displayed in form of menues. Step by step the user will be informed how to supply new data. Whenever problem data are generated or altered, the corresponding information will be saved on a user-provided file. Besides commands to generate, solve or edit a problem, there are others to transfer data from one problem to another, to delete a problem, to sort problems, to update the known experience, to get a report on problem or solution data, to halt the system and to get some information on the system, the mathematical models and the available algorithms.

The main intention of EMP is to prevent the organisational 'ballast' otherwise required to solve a nonlinear programming problem with a special algorithm. Once the system is implemented, it is not necessary

- to define a special file for each problem to be solved,
- to select a suitable mathematical algorithm,
- to read any documentation of the used mathematical programming algorithms,
- to write long lists of declaration statements, e.g. for dimensioning auxiliary arrays required by the algorithm, or to

UNITED STATES DEPARTMENT OF COMMERCE
National Bureau of Standards
Gaithersburg, Maryland 20899

July 18, 1985

Dr. Alex Orden, Chairman
Mathematical Programming Society
Graduate School of Business
University of Chicago
1101 East 58th Street
Chicago, IL 60637

Dear Alex:

The purpose of this letter is to provide a brief report to you on the activities of the Committee on Algorithms (COAL) of the Mathematical Programming Society with regard to the testing of the new projective method for linear programming proposed by N. Karmarkar of AT&T Bell Laboratories.

Shortly after the special session on the new method that was held at the Dallas meeting of ORSA/TIMS in November 1984, A. Rinooy Kan arranged a meeting of representatives of Bell Labs, COAL, and the National Bureau of Standards (NBS). The purpose was to explore the possibility of a collaborative test effort among experienced computational testing staff of these organizations. The idea was to perform a thorough computational evaluation of the new method and to publish the results in the open literature.

During that meeting, R. Graham of Bell Labs evidenced an interest in a collaborative effort, and, on behalf of Bell Labs, agreed to participate, subject to corporate approval. We of course were delighted to participate. This proposed effort was reported to you shortly thereafter and you sanctioned COAL's involvement. The project was also endorsed by the presidents of ORSA and TIMS, and by many respected members of the operations research and mathematical programming communities. Our charge was to perform a thorough, scientific, computational evaluation of the new method. Our hope was that we could shed some light on its computational characteristics and thereby resolve some of the issues that were raised during the Dallas ORSA session.

The issues that were raised and the discussion that ensued resulted from a sincere and professional interest in the computational performance of the new method. Efforts by audience members to obtain more information, however, were somewhat frustrated by the fact that Karmarkar did not, even after specific requests, reveal enough details of their testing (experiment design, test environment, test problems, starting values, and convergence tolerances) for others to replicate their results. At the Dallas Meeting, Karmarkar reported that his method was 50 to 100 times faster than the simplex method, a report that naturally caused great interest. It was our intention to fashion a computational experiment that would have provided the scientific community with the needed details from which such results could be fully understood.

The first step in this effort, it was agreed, was for NBS/COAL to propose an experiment that would, due to its design, provide the desired information on the computational characteristics of the new method. As you will see from the enclosed collection of correspondence, we developed

UNITED STATES DEPARTMENT OF COMMERCE
National Bureau of Standards
Gaithersburg, Maryland 20899

July 15, 1985

Dr. Michael R. Garey
Dr. David M. Gay
At&T Bell Laboratories
600 Mountain Avenue
Murray Hill, New Jersey 07974

Dear Drs. Garey and Gay:

We received your response of April 24, 1985 to our proposal of January 7, 1985 to test the projective algorithm for linear programming of N. Karmarkar. Needless to say, we were disappointed that we will not be able to do this testing.

In developing our proposal, we had conscientiously tried to suggest a means of testing the algorithm which both satisfied the needs of the community for relevant details and, at the same time, protected AT&T's proprietary interests. Our proposal was intended to be a point of departure for discussion and negotiation and thus we think that the problems you raised could be satisfactorily resolved. First, we did not require a full-scale implementation of the algorithm for the series of tests we suggested. We certainly realize that the production of such a code is premature until further testing and refinements are completed. We did, however, assume that there was a research version of a code which could be used for the purpose. Indeed, we fully expected to use the program, or slight modification, which was employed to generate reported results. Secondly, as we discussed on the phone, NBS has a history of signing non-disclosure agreements in order to protect the proprietary interests of the private sector and we were willing to explore other means to guard against public disclosure.

We are still willing to discuss a joint testing effort with you, but we agree that the delay has necessarily limited its value. We are, however, interested in pursuing your offer to cooperate in forming and distributing a set of test problems for linear programming. We continue to believe that such an effort will be in the best interest of the mathematical community. Perhaps we could discuss this matter at the XIIth International Symposium on Mathematical Programming to be held in Boston on August 5-9, 1985.

Sincerely,

Paul T. Boggs　　Karla L. Hoffman　　Richard H. F. Jackson

call the mathematical programming code with a long list of parameters that are to be defined initially,
- to provide the problem functions and their gradients in a special form required by the mathematical programming algorithm,
- to define an individual output file for each set of results,
- to interprete the results for a decision maker.

Thus the domain of application of EMP is summarized as follows:

(i) Programming neighbourhood for developing a first executable program version solving a specific practical problem (or class of problems).
(ii) Investigation of different model variants fitting best to a given real world situation.
(iii) Testing certain types or modifications of mathematical programming algorithms for solving a class of problems.
(iv) Collecting numerical experience on solution methods for optimization problems.
(v) Teaching students on model building (e.g. structural optimization courses in engineering science) or on numerical behavior of optimization algorithms (e.g. optimization courses in mathematics).

EMP allows a user to concentrate all his efforts on the problem he wants to solve and takes over the additional work to select a suitable algorithm and to organize the data, the execution of the problem functions and the program structure. It should be possible to solve optimization problems of the class under consideration within a minimal fraction of time needed otherwise.

A more extensive report is available from the author which contains also a detailed example in form of a complete copy of the corresponding terminal in- and output.

Klaus Schittkowski
Mathematisches Institut
Universitaet Bayreuth
8580 Bayreuth, Germany F.R.

APPLICATION FOR MEMBERSHIP

Mail to: Mathematical Programming Society, Inc.
c/o International Statistical Institute
428 Prinses Beatrixlaan
2270 AZ Voorburg, The Netherlands

Cheques or money orders should be made payable to The Mathematical Programming Society, Inc. in one of the currencies listed below.

The dues, which cover subscription to volumes 34-36 of MATHEMATICAL PROGRAMMING, are: Dfl. 125.00 (or $45.00 U.S. or British Pounds 28.00 or Sw.Fr. 90.00 or FF 340.00 or DM 112.00) (*).

Subscription to volumes 25-26 of MATHEMATICAL PROGRAMMING STUDIES (available to members only) is: Dfl. 45.00 (or $15.00 U.S. or British Pounds 10.00 or Sw.Fr. 33.00 or FF 120.00 or DM 40.00).

I wish to enroll as a member of the Society. My subscription(s) is (are) for my personal use and not for the benefit of any library or other institution. I enclose payments as follows:

Dues for 1986 (*): ....
Subscription to the STUDIES (volumes 25-26): ....
Total: ....

(*) NEW MEMBERSHIP APPLICATIONS: For 1986 only, new members pay dues at one-half the above rates. Subscription to the STUDIES is at full rates.

New member ? (check here) ....

STUDENT APPLICATIONS: Dues are one-half the above rates. Subscription to the STUDIES is at full rates. Have a faculty member verify your student status and send application with dues to the above address.

Faculty verifying status .... Institution ....

AT&T
Bell Laboratories
600 Mountain Avenue
Murray Hill, New Jersey 07974

April 24, 1985

Dr. Paul T. Boggs
Dr. Karla L. Hoffman
Dr. Richard H. F. Jackson
National Bureau of Standards
Gaithersburg, Maryland 20899

Dear Drs. Boggs, Hoffman, and Jackson:

This is in response to the proposal you made to A. Fronistas for a cooperative effort, involving AT&T Bell Laboratories, the National Bureau of Standards, and the Committee on Algorithms of the Mathematical Programming society, to evaluate the performance of the new linear programming method of N. Karmarkar.

We are delighted about the prospect of having others involved in studying this method and very much appreciate your interest in exploring ways to make this come about. The rather formal arrangement you have proposed, however, does not seem to us the most sensible way to proceed. In fact, for various reasons, such an arrangement does not even seem feasible at this point. For example, the proposal seems to require a full-scale implementation of the method that goes well beyond the more limited version that we are using for purposes of research. Moreover, any consideration of releasing a detailed implementation of the method outside of AT&T raises a host of difficult issues involving the handling of company proprietary information. In addition, it is not clear that the best interests of such independent groups as the NBS and COAL would be served by a cooperative arrangement that might be interpreted by some as a "product evaluation," despite the fact that what would be evaluated is by no means a product at this point.

We would prefer to cooperate more informally, in particular, by offering our enthusiastic support for the efforts of COAL toward developing a basic set of LP test problems that might serve in general for "benchmarking" LP software. We would be most interested in using such test problems for studying the Karmarkar method. We should also be able to help in this effort by providing some candidates for the test set, as well as pointers to other individuals who might also be willing to provide such candidates. Moreover, we are willing to help distribute test problems by making them available over "netlib," the experimental software distribution system put together by Eric Grosse and Jack Dongarra. (We cannot guarantee to offer netlib service indefinitely, but it seems likely that we can offer it for a goodly length of time.)

Despite the initial difficulties that many people have had in understanding and effectively implementing the Karmarkar method (which is quite understandable in view of the substantial differences between a theoretical algorithm description and what is needed for a good implementation–as well as the need for time to digest the mathematics of the new approach), it now appears that others in the community are beginning to make progress with regard to both implementations and theoretical improvements. Thus it seems likely that the kind of evaluation of the method that we all seek will occur naturally in the scientific community.

Sincerely yours,

Michael R. Garey   David M. Gay

MESSAGE FROM THE CHAIRMAN

From the point of view of COAL the Mathematical Programming Symposium in Boston was a successful one. A vivid and growing interest in computational mathematical programming and other COAL related matters was evident from the attention given to the papers (around 30) presented in the COAL sessions. Also about 200 people attended the ceremony in which George Dantzig as chairman of the prize committee awarded the Orchard Hays prize for excellence in computational mathematical programming to Michael Saunders.

On the organizational side many changes can be noted. In addition to the many new COAL members (see inside cover) I was nominated as the new chairman. Our old chairman, Karla Hofmann, was elected to the MPS council, where she joins a substantial group of ex-COAL members. As you have noticed Bob Meyer is willing to continue as COAL Newsletter editor with the assistance of Jens Clausen as European co-editor. I hope all of us will serve very fruitful terms.

Currently we are exploring the possibilities of cooperation in meetings and publications with ACM's SIGMAP. Although no final decision has been made, it seems like the Newsletter is the most convenient way to get the cooperation started. To stay with publications, the editor reports on the resolution of the problem of sending the COAL Newsletter to the "Friends of COAL". The MP Study on Computational Mathematical Programming edited by Ric Jackson, Karla Hofmann and myself, is well underway. We expect to be able to send off for printing the first 8 papers later this year. The Proceedings of the Bad Windsheim meeting (NATO-ASI on Computational Mathematical Programming) have been published by Springer Verlag. The editor, Klaus Schittkowski is to be congratulated on the nice job he did.

As far as the future is concerned, preparations for the next COAL meeting (winter 86/87 or summer 87) are well underway. It will not be a "summer school" or "study institute" but more like a one week advanced research seminar. We are currently investigating which one of 3 proposed sites to hold the meeting at. Also I have requested strong council support for an update and revision of the guidelines for reporting computational experiments. Ric Jackson, Susan Powell, Steve Nash and Paul Boggs have volunteered to undertake this task. They will start as soon as council has accepted a motion expressing its willingness to consider the guidelines to be developed as a standard to be imposed for all publications of the Mathematical Programming Society.

Jan Telgen

PETITION FOR THE FORMATION OF

SIGCMP
SIG on Computational Mathematical Programming

(A reincarnation of SIGMAP)

SIGCMP will be concerned with innovations in hardware and software involved with the implementation of mathematical programming algorithms and solutions to mathematical programming models.

It will be concerned with innovations in hardware and software which affect the speed, robustness, accuracy, reliability and portability of mathematical programming algorithms.

It will be concerned with innovations in hardware and software which affect the expression and solution of mathematical programming models.

It is expected that, if formed, SIGCMP would undertake joint activities with the Committee on Algorithms (COAL) of the Mathematical Programming Society. Tentative agreements have already been made for joint newsletters and conferences.

If you would like to see this SIG formed with the above goals, please sign this petition and return, no later than February 28, 1986, to:

Jerome L. Kreuser
World Bank, Rm. M-816
1818 H Street, N. W.
Washington D.C. 20433

______________________________ ____________
Signature Date

____
ACM Membership Number
(An ACM membership number is required for a valid ballot)

NOTES FROM THE U.S. CO-EDITOR

Thanks are due to Jan Telgen for his fine work as past co-editor. Jan gave generously of his time, collected many good articles and handled actual publication and distribution of the COAL Newsletter. The Newsletter has served as a vehicle for the rapid dissemination of information on new software and computational results, and with the assistance of the new co-editor, Jens Clausen of the University of Copenhagen, this tradition will be continued. Jan Telgen has been appointed as the new COAL chairman, and, as he reports on the following page, he is assisting with the planning of a COAL meeting for 1987. Jens and I invite short contributions appropriate for the Newsletter, which will be issued in April and October, 1986. While on the topic of re-organization, it should be noted that non-MPS-members (Friends of COAL) are invited to subscribe to the COAL Newsletter for 1986 for the annual fee of $10 (US), which may be sent to me in the form of a check made payable to *The Mathematical Programming Society*. Discussions are continuing with ACM officers on possible joint activities with SIGCMP, the Special Interest Group on Computational Mathematical Programming, which Jerry Kreuser is seeking to organize as the successor to SIGMAP, which was dissolved. ACM members are urged to sign and mail the petition for the formation of SIGCMP that is in this Newsletter.

In this issue we are fortunate to have a summary of the results presented at the August MPS International Symposium on the log barrier function method for linear programming. With the permission of the authors, we have also reproduced the correspondence between COAL members at the National Bureau of Standards, researchers at AT&T Bell Laboratories, and Alex Orden, president of MPS, on the subject of computational testing, especially as this relates to the software by N. Karmarkar. Although negotiations for *outside* testing are no longer continuing, one positive result of the discussions has been the establishment of the distribution mechanism for linear programming test problems described in the article below by David Gay.

This coming year should be a particularly exciting one for computational mathematical programming, and I look forward to issuing some interesting Newsletters in 1986.

Robert R. Meyer

COAL OBJECTIVES

The Committee on Algorithms is involved in computational developments in mathematical programming. There are three major goals: (1) ensuring a suitable basis for comparing algorithms, (2) acting as a focal point for computer programs that are available for general calculations and for test problems, and (3) encouraging those who distribute programs to meet certain standards of portability, testing, ease of use and documentation.

NEWSLETTER OBJECTIVES

The newsletter's primary objective is to provide a vehicle for the rapid dissemination of new results in computational mathematical programming. To date, our profession has not developed a clear understanding of the issues of how computational tests should be carried out, how the results of these tests should be presented in the literature, or how mathematical programming algorithms should be properly evaluated and compared. These issues will be addressed in the newsletter.

Mathematical Programming Society

Committee on Algorithms Newsletter

NO. 13 — ROBERT R. MEYER, JENS CLAUSEN — EDITORS

DECEMBER 1985

CONTENTS

ROBERT R. MEYER
UNIVERSITY OF WISCONSIN-MADISON
COMPUTER SCIENCES DEPARTMENT
1210 WEST DAYTON STREET
MADISON, WISCONSIN 53706