ORSA Journal on Computing

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Commentary—Major Cholesky Would Feel Proud

Michael A. Saunders,

To cite this article:
Michael A. Saunders, (1994) Commentary—Major Cholesky Would Feel Proud. ORSA Journal on Computing 6(1):23-27. https://doi.org/10.1287/ijoc.6.1.23

ORSA Journal on Computing
Vol 6, No. 1, Winter 1994

0899-1499/94/0601-0023 $01.25

COMMENTARY

Major Cholesky Would Feel Proud

MICHAEL A. SAUNDERS / *Department of Operations Research, Stanford University, Stanford, CA 94305-4022; Email: mike@SOL-Michael.stanford.edu*

Probably more than any other group, authors Lustig, Marsten, and Shanno have led the way in demonstrating the effectiveness of interior methods for solving large, real-world linear programs. For several years they have written at length about things they have actually *done*. There are no pies in the sky, no secrets. They have explored a multitude of algorithmic ideas and implementation strategies, and they have done more than could be asked to share their experience with the rest of the world.

Here, the authors look back on a decade's events that revived barrier methods for constrained optimization and eventually led to their production LP solver OB1. The results reported are indeed impressive. In this commentary, we focus mostly on the "engine"—the Cholesky factorizer —that has made OB1 such a practical success.

1. History

1.1. Solving with AD^2A^T

The principal operation at each step of an interior method has been viewed (by most authors from Karmarkar onward) as solving linear systems of the form

$$AD^2A^Tu = ADc \tag{1}$$

or

$$AD^2A^Tv = d. \tag{2}$$

These are "normal equations" for the *least-squares problem* and the *minimum-length problem*,

$$\min\|DA^Tu - c\| \tag{3}$$

and

$$\min\|w\| \text{ s.t. } ADw = d, \tag{4}$$

respectively. Interior methods vary in whether they need u,v, or w $(= DA^Tv)$. For example, the projective method and the primal barrier method both need least-squares solutions u,[13] whereas dual barrier needs a v each iteration. Primal-dual barrier can work with least-squares solutions once primal and dual feasibility are achieved. OB1 uses a min-length w to define its starting point x_0, but in general needs two vs each iteration.

We mention this because u is better defined than v when AD is ill-conditioned, and w is better defined than u or v. Because interior methods, unlike the simplex method, come increasingly close to numerical failure as the solution is approached, such considerations are vital for those concerned with implementation. We return to this issue in Section 3.

1.2. Major Cholesky

It was for the least-squares problems arising in geodesy that Major André-Louis Cholesky invented the triangular factorization $AA^T = LL^T$. (This occurred sometime before 1914 while Cholesky was attached to the Geodesic Section of the French Geographic Service, triangulating the island of Crete.[5])

In the days of hand computation, the dimension of L must have been quite modest. During the 1960s a dense matrix of order 100 was still considered "large," although order 1000 would have seemed large only a decade ago. Cholesky was noted for his "extraordinay intelligence, a great aptitude for mathematics and an enquiring mind," but how could he have imagined geodesic problems with millions of observations and variables? Or satellites and computing machines that could triangulate not just Crete but all of France and Europe in a matter of minutes?

In the feature article, the OB1 authors calmly mention Cholesky factors of order 50,000, part sparse and part dense, containing over 100 million numbers. By tortuous means these enormous Ls are helping to schedule air travel for millions of people world-wide. Other Earth-sized applications are in abundance. Surely the Major would feel proud! (And so, of course, would Sir Isaac Newton.)

1.3. Sparse Cholesky Factors

Sparse factorizations $M = LL^T$ are vital in engineering for the method of finite elements. A classic implementation is SPARSPAK,[4, 8] in which a positive-definite M is assumed to be given.

As it happens, finite-element problems can be formulated naturally as *least-squares* problems, such that $M = AA^T$ for some rectangular A. Indeed, the 1984 package SPARSPAK-B[9] was implemented to deal directly with A (using a sparse QR factorization as in [10]). It is fortunate that both AA^T and L remain sparse in typical finite-element applications. Who would have guessed that the same would be true for the A in linear programs?

For some reason, nobody concerned with LP was bold enough to try SPARSPAK on AA^T or SPARSPAK-B on A until long after 1984.

1.4. Karmarkar's Contribution

The authors rightly give prominence to Karmarkar's 1984 projective method for its polynomial complexity and the originality of the proof. In practical terms, we must regard one of Karmarkar's contributions to be his insistence that

an interior method could be efficient for large linear programs, *even though it depended* (at the time) *on the sparsity of the Cholesky factors of* AA^T.

A further contribution that may yet prove practical was the idea of *updating* the Cholesky L an arbitrary number of times.

Subsequent implementations such as OB1 have confirmed that AA^T and L are indeed acceptably sparse for many real-life LPs. (Newton's method is thereby primed to compete with simplex.)

The exceptions remain of great practical importance. We pursue this difficulty in the following sections.

2. Computational Results

The authors' test problems are distinctly larger than those we are accustomed to seeing in the literature. They clearly provide a demanding test for any LP solver. It is commendable that the authors ran the OSL simplex code eight times on each of the first eight models, in an effort to deduce same favorable options and show the variability possible.

The models are listed in a seemingly arbitrary order. Unfortunately, the obvious reorderings according to model size (rows, columns, or nonzeros) do not reveal a clear pattern.

We do see that preprocessing reduced the model size substantially in all but one case (Model *car*). If anyone feels a need for more demanding models, the preprocessing could clearly be bypassed. Would numerical difficulties become more evident in OB1?

Probably so, according to the authors' candid hints in earlier papers. The simplex method may be more robust in this respect, yet we cannot overlook the fact that primal simplex produced the only failure reported (one run on Model *initial*). Which is more admirable in terms of reliability: eight of eight for OB1 with default settings, or 63 of 64 for OSL simplex?

We cannot really say because both performances are excellent, but it is implied that certain simplex codes have failed on the two Delta models, giving 10 of 10 for OB1 and 63 of 66 for simplex. The balance begins to sway.

In terms of run-time, OB1's performance is undeniably good: 9 of 10 compared to simplex, and mostly by a wide margin. The authors' conclusions therefore seem correctly stated: "For many large models, interior point codes are significantly faster than simplex." Can Major Cholesky rest in peace?

2.1. Dense Cholesky Factors

The efficiency of OB1 depends directly on $|L|$, the number of nonzeros in the Cholesky factorization $PAD^2A^TP^T = LL^T$ (where P is a judicious row permutation). As already noted, one of the marvels is that for certain choices of P and for many real-world matrices A, the factor L is sparse. This is true for six of the present examples, but Models *schedule* and *initial* give an initial tremor of alarm. Table I lists the models in order of increasing $|L|/|A|$.

Note that *initial* is the only model with more rows than columns, and OSL dual simplex is by far the most efficient code. Perhaps OB1 would show a significant improvement on the dual of *initial*, but the question remains: Why is L so dense on the last three problems?

Table I. The Test Problems Listed in Order of Increasing $|L|/|A|$. m = Number of rows; Time ratio = (Best OSL simplex time) / (OB1 time)

Model	$\|L\|/\|A\|$	m ×1000	$\|A\|$ ×1000	Time ratio
car	1	43	190	13.3
continent	2	7	184	2.5
energy1	2	12	82	16.5
energy3	2	10	192	6.1
energy2	3	7	143	27.5
fuel	7	13	186	8.4
schedule	23	5	37	7.6
initial	80	16	65	0.1
Deltal	137	32	240	
Delta2	308	45	342	

The Cray Y-MP used for the Delta models evidently has an abundance of memory. (It would be useful to know how much.) Because Cholesky factors with 33 million and 105 million nonzeros are beyond comprehension for most of us, one wonders how the authors can report those figures *almost without comment*? The dense linear algebra may parallelize beautifully, and we can share the authors' sense of accomplishment in solving these enormous problems, but there must be a hint of concern at the apparent trend. (Indeed, for Model *initial* the authors describe 5 million Cholesky nonzeros as "catastrophic," but only by comparison to the simplex method's modest requirements. For the Delta results, the alarm takes on both relative and absolute proportions.)

In a similar vein, Forrest and Tomlin[6] reported solving some very large models on IBM RS/6000 workstations. Like OB1, the OSL barrier code[16] treats part of the Cholesky L as a dense matrix. Although RS/6000s have virtual memory, 256MB of real memory was not enough for one case to solve efficiently. The "easy answer" was to move to a machine with 512MB.

For the Delta models here, we can deduce that the dense part of L was a triangle of order $k \approx 12{,}000$. Unless mass storage was cleverly used, the Cray Y-MP must have had about 1000MB of main memory to store A and all of L. Crays normally have many tasks competing for memory. Did the operating system really allocate that much to a single job for 22 hours?!

The authors probably would not fear even larger problems. However, the storage and work for computing dense Cholesky factors grows at a quadratic and cubic rate, as shown in Table II. It is hard to imagine that such figures will be commonplace when the next "computational state of the art" review appears.

2.2. Dense Columns

The usual reason for dense Cholesky factors is the presence of relatively dense columns in A. Strangely, the authors do

Table II. Storage and Work for Computing Dense Cholesky Factors of Order k. Storage = $\frac{1}{2}k^2 \times 8 / 10^6$ Megabytes; Work = $\frac{1}{6}k^3 / 10^9$ Gigaflops

k	Storage (Megabytes)	Work (Gigaflops)
10000	400	167
20000	1600	1333
30000	3600	4500
40000	6400	10667
50000	10000	20833

not touch on this question here, although they have indeed been concerned in earlier papers. Did any of the present problems contain dense columns? Were they treated specially? It would be good to know.

At present, probably the most effective way to help Cholesky is to "stretch" dense columns into two or more pieces, as described for stochastic programs by Lustig et al.,[20] for general LPs by Vanderbei[22] and even more generally by Grcar.[14]

Otherwise, non-Cholesky methods must be considered; see Section 3.

2.3. Other Tricks

The question of *robustness* must arise in a computational review. Although no failures were reported for OB1, the authors could offer some clues on how this was achieved in the face of inevitable ill-conditioning of the Cholesky factors. Preprocessing is one device already mentioned. Two other possibilities are regularization and iterative refinement; see Gill et al.[12]

Regularization involves perturbing the LP problem to make it "easier"—the net effect being to add small diagonal matrices to D^{-2} and AD^2A^T. Because eight digits of solution accuracy are ample on a 16-digit machine, perturbations of order 10^{-8} or 10^{-9} should be justifiable. Do the authors prefer not to perturb?

Refinement means extra solves with the Cholesky factors, but a benefit is that it helps determine if the linear systems are being solved with reasonable accuracy. (If not, it may be time to stop.)

In general, a robust code knows when to "fail," as opposed to struggling on with undetected erroneous search directions. Newton methods generally have a merit function that must decrease noticeably at each iteration. This aspect of global convergence is covered in the primal-dual implementations of Mehrotra,[21] Gill et al.,[12] and Lustig et al.[19] The results for the Delta models quote eight and six digits of accuracy. It would be interesting to know: How was OB1 terminated in the second case?

3. Alternatives to Cholesky

If there are tests to determine that one's engine is about to fail, it would be ideal if a spare engine were available.

The simplex method enjoys certain such luxuries. If the basis update seems to be unstable, it can be abandoned in favor of a fresh factorization $PBQ = LU$. If that too shows signs of failure, it can be repeated with stricter tolerances to alter the permutations P and Q, trading sparsity for stability. In effect, the LU factorizer is its own backup.

In Cholesky-based barrier methods, there is little one can do if failure appears imminent because altering P does not materially alter the stability of factorizing a positive-definite PAD^2A^TP. Also, if the preferred P causes catastrophic fill-in, it is unlikely that a better P can be found.

What is the state of the art regarding alternatives?

3.1. Improving Stability

One way to obtain better stability with essentially the same Cholesky factor is to compute it by orthogonal factorization. This means $DA^T = QR$, where $Q^TQ = I$ and $R = L^T$ with exact arithmetic.

Even if Q is unavailable (as in SPARSPAK-B), Björck[1] has shown that good least-squares solutions can be obtained by the method of *corrected seminormal equations*. This involves solving $R^TRu = ADc$ and performing one step of refinement. Björck[2] has also shown that a similar method can solve the min-length problem accurately without the aid of Q.

A disadvantage is that R may take two or three times as long to compute as the highly optimized Cholesky L. Also, it helps (1), (3), and (4), but apparently not (2).

3.2. Improving Stability and Sparsity

QR factorization does not remove the risk of catastrophic fill-in in $L = R^T$. As a remedy, various authors have advocated factorizing larger systems of the form

$$K = \begin{pmatrix} -D^{-2} & A^T \\ A & \end{pmatrix}. \tag{5}$$

Two practical implementations warrant mention, given their reported efficiency on the full NETLIB collection.

First, Fourer and Mehrotra[7] compute LBL^T factorizations (B block-diagonal) of matrices of the form

$$H = \begin{pmatrix} -\kappa I & DA^T \\ AD & \end{pmatrix}, \tag{6}$$

using a Markowitz-type strategy to choose pivot orderings. Two parameters κ and δ are used to balance stability and sparsity. On 87 NETLIB problems, the number of nonzeros in L was never more than twice that for Cholesky on AD^2A^T, and in several cases was much less. (Not just when A contained dense columns.) Because reliability was uniformly good, this could be the spare engine that OB1 needs.

Second, Vanderbei[23, 24] applies an "indefinite" Cholesky factorization $\bar{L}\bar{D}\bar{L}^T$ to a somewhat modified K. ($\bar{D}$ is allowed to have both positive and negative elements.) It is hard to make any theoretical claims about stability, but on 88 NETLIB problems, the number of nonzeros in $\bar{L}$ was very similar to Cholesky on AD^2A^T,[25] and no numerical difficulties were reported.[23, 25] As with Cholesky itself, this

success remains difficult to explain, but difficult to overlook.

3.3. Some Pies in the Sky

Vavasis[26] warns that most existing methods for solving with K in (5) are unstable! He proposes a stable algorithm that is intriguingly different from others. Initial results on sparse finite-element problems are given in [27]. The method seems worthy of future consideration.

A further possibility is to work with *unsymmetric Jacobians* (cf. Wright[28]). For example,

$$\bar{K} = \begin{pmatrix} Z & -XA^T \\ A & \end{pmatrix} \tag{7}$$

arises naturally in the primal-dual method. An apparent advantage is that no large numbers are present (in contrast to K). The condition of $\bar{K}$ tends to be lower. However, primal or dual degeneracy does cause $\bar{K}$ to become ill-conditioned.

Returning to the primal-dual equations,

$$XZe = \mu e, \tag{8}$$

$$z + A^T y = c, \tag{9}$$

$$Ax = b, \tag{10}$$

we might ask: Should these be treated directly, and are they the right equations?

As the authors note, they may be derived from the first-order conditions for the primal or dual barrier problems. Lagrangians are not really needed (since the Karush-Kuhn-Tucker conditions may be written directly), but a sleight of hand *is* required. The primal barrier problem really suggests the equation $z = \mu X^{-1}e$, whereas dual barrier suggests $x = \mu Z^{-1}e$. Convexity and uniqueness arguments are needed to arrive at $XZe = \mu e$. A fourth possibility is $X^{-1}Z^{-1}e = (1/\mu)e$.

If we apply Newton's method to (8)–(10) with the first equation replaced in turn by those options, we obtain Jacobians of the form

$$J_1 = \begin{pmatrix} I & \mu X^{-2} & \\ I & & A^T \\ & A & \end{pmatrix}, \tag{11}$$

$$J_2 = \begin{pmatrix} \mu Z^{-2} & I & \\ I & & A^T \\ & A & \end{pmatrix}, \tag{12}$$

$$J_3 = \begin{pmatrix} X & Z & \\ I & & A^T \\ & A & \end{pmatrix}, \tag{13}$$

$$J_4 = \begin{pmatrix} X^{-1}Z^{-2} & X^{-2}Z^{-1} & \\ I & & A^T \\ & A & \end{pmatrix}. \tag{14}$$

Some of the implications are discussed by Gill et al.,[11] along with global convergence proofs and assurances of nonsingularity in each J. A few additional comments follow. Conceivably some future solver will be able to handle one of these forms efficiently.

- J_2 is the only symmetric Jacobian.
- For J_1, it is safe to pivot on either I. Eliminating Δz in that way gives K in (5), which should therefore be a satisfactory symmetric form.
- For J_3, I is again the only block pivot that is obviously stable. Eliminating Δz gives the unsymmetric system (7), whose condition should therefore be as good as that of J_3.
- J_4 seems unlikely (until its top row is scaled by X^2Z^2), but it is the only one for which the usual damped Newton method maintains both $x > 0$ and $z > 0$ automatically.

3.4. The Outlook

The preceding sections have raised more questions than answers. To help justify them, let us recall the simplex method and the classic Product-Form (PF) update to the basis. After many years of sterling service in commercial codes, the PF update has finally given way to the Forrest–Tomlin LU update. Ironically, this is not for stability reasons but because it is more efficient.

Similarly, the potential for catastrophic fill-in with Cholesky may lead to a completely different implementation of interior methods. The authors' complacency with Cholesky is doubtlessly only an illusion. Their vectorized and parallelized implementation is certainly a good start.

4. Final Comments

No state-of-the-art review can include every reference that has bearing on the subject. Nor can it make every fine point known to other researchers. Here we list a few missing items that seem important.

- An exceptionally thorough and readable review of interior methods has been given by Wright.[28] This covers much of the theoretical development underlying any computational work, and also explores some of the linear algebraic issues touched on here.
- In terms of significant computational developments, we cannot forget the advent of the KORBX® system of AT&T.[3, 15, 17] The primary equation solver was based on Cholesky factors of AD^2A^T, with an option to use a conjugate-gradient method with Cholesky preconditioning. Extremely large models were reported solved, comparable to those solved here by OB1.
- Conversely, we should not forget that the authors' efforts to develop and market OB1 were hampered for several years by some remarkably over-reaching pressure from AT&T's lawyers. Karmarkar's work and the KORBX® implementation undoubtedly provided inspiration in broad terms, but to what extent is inspiration patentable? Should George Dantzing sue IBM and others for royalties from their implementations of the simplex method? ("The" simplex method?)
- Perhaps we are now beyond a passing lapse in company policy. The lawyers may have moderated their view, just as the company's TV ads may have risen at last above ridiculing other telephone companies. Much excellence

shines forth from AT&T to earn respect. "The greater an institution, the more noble it can afford to be;" is that principle too idealistic? It could soon become "The more noble an institution, the greater it will be."

- Another principle: "Emulation is the highest form of praise." The authors were too modest to mention that their published work on OB1 has led directly to effective barrier implementations within IBM's OSL.[16] The existence and good performance of the OSL barrier code certainly merits mention.
- The alternative solvers of Fourer and Mehrotra[7] and Vanderbei[23,24] have also demonstrated good performance worthy of mention. In (5) and (6) we should determine which matrix K or H is more "friendly."
- We did not discuss iterative methods for solving the Newton equations. One of the more promising possibilities is being pursued by Kim and Nazareth.[18]

In summary, the OB1 authors have covered much ground in their review to give us a feeling for what has been accomplished and how. We trust that their dynamic spirit will live on.

The topics they list for future work are warm starts, detecting infeasibility, and crossover to simplex. In this commentary we have recommended that eyes be kept open for a backup to Cholesky factorization, in the event of numerical failure or catastrophic fill-in.

In the meantime, Major Cholesky's invention has been honed to a peak of efficiency for the current generation of machines, and some of us can't help wondering: Why is it so hard to do better? Let's see where we are in 2004 (a further decade after the revolution).

References

1. A. BJÖRCK, 1987. Stability Analysis of the Method of Seminormal Equations for Linear Least Squares Problems, *Linear Algebra and its Applications 88 / 89*, 31–48.
2. Å. BJÖRCK, 1990. Error Analysis of the Modified Gram-Schmidt Method for Solving Underdetermined Linear Systems, Report, Department of Mathematics, Linköping University, Linköping, Sweden.
3. Y-C. CHENG, D. J. HOUCK, JR., J-M. LIU, M. S. MEKETON, L. SLUTSMAN, R. J. VANDERBEI AND P. WANG, 1989. AT&T KORBX® System, *AT&T Technical Journal 68:3*, 7–19.
4. E. CHU, A. GEORGE, J. LIU, AND E. NG, 1984. SPARSPAK: Waterloo Sparse Matrix Package User's Guide for SPARSPAK-A, Research Report CS-84-36, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada.
5. R. W. COTTLE, 1975. Major Cholesky, Manuscript, Department of Operations Research, Stanford University, Stanford, CA. Translated from the French *Bulletin Géodésique* (1922).
6. J. J. FORREST AND J. A. TOMLIN, 1992. What Makes a Linear Program Difficult? Presented at ORSA/TIMS Joint National Meeting, San Francisco, CA.
7. R. FOURER AND S. MEHROTRA, 1992. Solving Symmetric Indefinite Systems in an Interior-Point Method for Linear Programming, Technical Report 92–01, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL.
8. A. GEORGE AND J. W-H. LIU, 1981. *Computer Solution of Large Sparse Positive Definite Systems*, Prentice-Hall, Inc., Englewood Cliffs, NJ.
9. A. GEORGE AND E. NG, 1984. SPARSPAK: Waterloo Sparse Matrix Package User's Guide for SPARSPAK-B. Research Report CS-84-37, Department of Computer Science, University of Waterloo, Waterloo, Ontario, Canada.
10. J. A. GEORGE AND M. T. HEATH, 1980. Solution of Sparse Linear Least Squares Problems Using Givens Rotations, *Linear Algebra and its Applications 34*, 69–83.
11. P. E. GILL, W. MURRAY, D. B. PONCELEÓN AND M. A. SAUNDERS, 1991. Primal-Dual Methods for Linear Programming, Report SOL 91-3, Department of Operations Research, Stanford University, Stanford, CA.
12. P. E. GILL, W. MURRAY, D. B. PONCELEÓN AND M. A. SAUNDERS, 1991. Solving Reduced KKT Systems in Barrier Methods for Linear and Quadratic Programming, Report SOL 91-7, Department of Operations Research, Stanford University, Stanford, CA.
13. P. E. GILL, W. MURRAY, M. A. SAUNDERS, J. A. TOMLIN AND M. H. WRIGHT, 1986. On Projected Newton Methods for Linear Programming and an Equivalence to Karmarkar's Projective Method, *Mathematical Programming 36*, 183–209.
14. J. F. GRCAR, 1990. Matrix Stretching for Linear Equations, Report SAND90-8723, Sandia National Laboratories, Albuquerque, NM.
15. E. C. HOUSOS, C. C. HUANG, AND J.-M. LIU, 1989. Parallel Algorithms for the AT&T KORBX® Systems, *AT&T Technical Journal 68:3*, 37–47.
16. IBM CORPORATION, 1991. *OSL Guide and Reference, Release 2*, IBM Corporation, Kingston, NY.
17. N. K. KARMARKAR, J. C. LAGARIAS, L. SLUTSMAN AND P. WANG, 1989. Power Series Variants of Karmarkar-type Algorithms, *AT&T Technical Journal 68:3*, 20–36.
18. K. KIM AND J. L. NAZARETH, 1994. A Primal Null-space Affine Scaling Method, *ACM Transactions on Mathematical Software*. To appear.
19. I. J. LUSTIG, R. E. MARSTEN AND D. F. SHANNO, 1992. Computational Experience with a Globally Convergent Primal-Dual Predictor-Corrector Algorithm for Linear Programming, Report SOR 92-10, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.
20. I. J. LUSTIG, J. M. MULVEY, AND T. J. CARPENTER, 1990. Formulating Stochastic Programs for Interior Point Methods, Report SOR 89-16, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.
21. S. MEHROTRA, 1992. On the Implementation of a Primal-Dual Interior Point Method, *SIAM Journal on Optimization*, 2, 575–601.
22. R. J. VANDERBEI, 1991. Splitting Dense Columns in Sparse Linear Systems, *Linear Algebra and its Applications 152*, 107–117.
23. R. J. VANDERBEI, 1991. Symmetric Quasi-definite Matrices, Report SOR 91-10, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.
24. R. J. VANDERBEI, 1992. LOQO User's Manual, Report SOR 92-5, Department of Civil Engineering and Operations Research, Princeton University, Princeton, NJ.
25. R. J. VANDERBEI AND T. J. CARPENTER, 1993. Symmetric Indefinite Systems for Interior Point Methods, *Mathematical Programming 58*, 1–32.
26. S. A. VAVASIS, 1993. Stable Numerical Algorithms for Equilibrium Systems, *SIAM Journal on Matrix Analysis and Applications*. To appear.
27. S. A. VAVASIS, 1993. Stable Finite Elements for Problems with Wild Coefficients, Report TR 93-1364, Department of Computer Science, Cornell University, Ithaca, NY.
28. M. H. WRIGHT, 1992. Interior Methods for Constrained Optimization, pp. 341–407 in *Acta Numerica 1992*, A. Iserles, (ed.), Cambridge University Press, New York, NY.