Performance of an Augmented System Approach for Solving Least-Squares Problems in an Interior-Point Method for Linear Programming

Robert Fourer and Sanjay Mehrotra

Department of Industrial Engineering and Management Sciences,
Northwestern University, Evanston, IL 60208-3119, U.S.A.

1. Introduction

Interior point methods for linear programming solve a weighted least-squares problem,

$$\min_w \|DAw - v\|, \tag{1.1}$$

to compute the search direction. Specifically, an iteration first generates a new weight matrix D, and solves (1.1) for one or more right hand sides v. The obtained solutions are combined to construct a search direction, and a step is taken to generate a new iterate and a new D.

It is well understood that solution of the least-squares problem represents a large part of the work of interior point methods. Many prior implementations [1, 3, 9, 11, 13, 14] first form the associated normal equations,

$$AD^2A^Tw = ADv, \tag{1.2}$$

and then compute the Cholesky factorization $LL^T = AD^2A^T$ by elimination. A major advantage of this approach is that all elimination pivot orders for finding L are stable. A computationally suitable pivot order can be determined (for example, by using a minimum degree or minimum local fill heuristic) at the outset, and can then be used for all iterations of the algorithm.

In this note we document our experience with an alternative, augmented system approach for solving the least-squares problem. Although this approach has been well studied in other contexts [2, 7], its suitability in the context of interior point methods is, to the best of our knowledge, still an open question. *Our objective here is to quickly disseminate our results; a subsequent report will give detailed findings.*

Our interest in the augmented system approach is motivated by the difficulties encountered in solving the least squares problem when A contains "dense" columns. Because the matrix AD^2A^T is then much denser than A, the normal equations approach must be modified in some way for this case. One idea is to remove the dense columns from A, and then to correct for their absence by an iterative method [1], a direct method [6], or some combination of the two [11]. All of these modifications suffer from numerical instability, which may be traced to singularity or near-singularity of AD^2A^T after the dense columns have been removed. A more numerically sound alternative is based on an augmented linear system formed by "splitting" dense columns [12, 19]; we do not yet have firm evidence as to the efficiency of this kind of modification, however.

All of the above approaches have the drawback of requiring that dense columns be identified in A in some arbitrary way. The approach that we describe below

handles dense columns naturally as part of the elimination of the augmented system. As a bonus, this approach also provides for a natural and stable handling of free variables as proposed by Mehrotra [16].

2. The Augmented System Approach

Problem (1.1) can also be solved by considering the augmented system

$$C_\alpha \begin{pmatrix} u \\ w \end{pmatrix} \equiv \begin{pmatrix} \alpha I & DA^T \\ AD & 0 \end{pmatrix} \begin{pmatrix} u \\ w \end{pmatrix} = \begin{pmatrix} v \\ 0 \end{pmatrix}, \tag{2.1}$$

where I is an $n \times n$ identity matrix and α is the scaling constant recommended by Björck [4] to improve the condition number of C_α. In particular Skeel's condition number [18], $\kappa_S(C_\alpha) = \| \, |C_\alpha^{-1}| \, |C_\alpha| \, \|$, where $|C_\alpha|$ is a matrix of entries $|(C_\alpha)_{ij}|$, is minimized by some $\alpha \in [0, \|AD\|_\infty]$ (see [2]). Our tests use $\alpha = 16^{-2}\|AD\|_\infty$.

If the first n rows and columns are eliminated from C_α, then the above system reduces to the normal equations (1.2); a somewhat different ordering gives the so-called Schur complement modification [6]. One can hope to obtain a sparser and more stable factorization, however, by applying a more general elimination method to C_α. Since C_α is a symmetric indefinite matrix, it can be symmetrically factored by using 1×1 and 2×2 block pivots [5]; but in contrast to the positive definite case, a tentative pivot sequence obtained from the nonzero pattern of C_α may not be numerically stable.

Mehrotra [16] shows that, if one of the diagonal entries of αI in C_α is taken as zero rather than α, then the associated variable is free rather than nonnegative in the underlying linear program. Since C_α remains symmetric and positive semi-definite, this adjustment offers a natural approach to handling free variables that has no direct analogue in the normal equations or Schur complement approaches.

Duff *et al.* [7] have recently reported very encouraging results for solving sparse symmetric indefinite systems of the kinds just described. They consider a variety of strategies for promoting sparsity in a series of 1×1 and 2×2 pivots, and propose a new stability test. Although their work is still in an experimental stage, initial computational results show significant improvements over the earlier Harwell multifrontal code, MA27 [8].

The computational results that we present here were obtained by using our own implementation of the Duff *et al.* approach, adapted for use within an interior point code. We next briefly describe the selection criterion and stability test under this approach, as we have implemented them.

Selection criterion

Assume that at some stage of the elimination we have the reduced (symmetric) matrix $C \in \Re^{k \times k}$. We must next select either a 1×1 pivot $c_{ii} \neq 0$, or a 2×2 pivot

$$\begin{pmatrix} c_{ii} & c_{ij} \\ c_{ji} & c_{jj} \end{pmatrix} \tag{2.2}$$

such that $c_{ij} \neq 0$. Following the terminology in Duff *et al.* [7] we call a 2×2 pivot *oxo*, if $c_{ii} = c_{jj} = 0$; *tile*, if either c_{ii} or $c_{jj} = 0$, but not both; or *full*, if $c_{ii} \neq 0$ and $c_{jj} \neq 0$.

Following the "Markowitz-b" approach of [7], we let n_i be the number of nonzero rows in column i of C (the *degree* of column i) and let $\hat{n}_{ij} = \min(n_i + n_j - 4, k - 2)$. For each kind of 2×2 pivot we define a "cost" as follows:

$$\begin{array}{ll} \textit{oxo} & (n_i - 1)(n_j - 1) \\ \textit{tile} & (n_i - 1)(\hat{n}_{ij} + 1) \quad \text{if } c_{ii} = 0 \\ & (n_j - 1)(\hat{n}_{ij} + 1) \quad \text{if } c_{jj} = 0 \\ \textit{full} & (\hat{n}_{ij})(\hat{n}_{ij}) \end{array}$$

We seek to find, in an approximate sense, the "sparsest" pivot that is numerically acceptable. For each $r = 1, 2, \ldots$, we first consider all 1×1 pivots of degree r, and then all 2×2 pivots of the form (2.2) whose cost is less than a certain threshold value. This threshold is taken to be $(r-1)^2$, $(r-1)(2r-3)$, or $(2r-4)^2$ for an *oxo*, *tile*, or *full* pivot, respectively.

Once a pivot order has been determined in this way, we try to re-use it at subsequent iterations of the interior point method. If it becomes unstable, then we generate a new pivot sequence.

Stability test

A 1×1 pivot c_{ii} is accepted if it satisfies the usual partial-pivoting condition,

$$|c_{ii}^{-1}| \cdot \|c_i\|_\infty \leq \beta^{-1},$$

where $\|c_i\|_\infty$ denotes the largest element in row (or column) i of C. A 2×2 pivot (2.2) is accepted if it satisfies an analogous condition,

$$\left| \begin{pmatrix} c_{ii} & c_{ij} \\ c_{ji} & c_{jj} \end{pmatrix}^{-1} \right| \begin{pmatrix} \|c_i\|_\infty \\ \|c_j\|_\infty \end{pmatrix} \leq \begin{pmatrix} \beta^{-1} \\ \beta^{-1} \end{pmatrix}.$$

In our tests, we initially take $\beta = 16^{-6}$, a value much lower than that used in [7]. If at any iteration the current β is inadequate for a numerically stable elimination, it is increased by a factor of 16.

3. Computational Results

Our tests use the primal-dual path-following algorithm [10, 17, 13, 11] as in the implementation described by Mehrotra [14], with the addition of facilities for bounded variables [15, Algorithm 2.1].

Timings are reported for both the original normal equations version of the implementation, and for a new version that incorporates the augmented system approach as described above. For the normal equations the pivot order is chosen by a minimum degree heuristic, which is the 1×1 analogue of the Markowitz-b approach. All times are in seconds, using the same Fortran compiler on a Sun-4/110.

The accompanying table gives iterations and timings for linear programs from the `netlib` test set. Whereas the augmented system approach achieves eight digits of accuracy in all cases, the normal equations approach sometimes fails; failures are indicated in the table by the iteration at which failure occurs, and the number

of digits of accuracy in the solution at that stage (in place of the timing). We believe that these failures are primarily due to a well-known numerical difficulty associated with the treatment of free variables (see, for example, [20]). Indeed, the augmented system version succeeded on problems *greenbea* and *greenbeb* only after a small number of free variables, which had been modeled as differences of nonnegative variables, were identified explicitly to the code.

From the standpoint of computational time, the augmented system approach achieves a predictably great advantage in solving those problems (*fit1p, fit2p, israel, seba*) that are well known to have dense columns. We also observe substantial improvement in another group of problems (*agg, capri, d2q06c, ganges*) whose dense columns are not so readily identified. For other problems the normal equations are acceptably sparse, and hence the augmented system is more expensive to solve; even so, performance is rarely worse by more than a factor of 2.

These preliminary results suggest that the augmented system approach offers a significant advantage in robustness of implementation for interior point methods, at an acceptable cost in speed for the easier problems.

Acknowledgements

This work has been supported in part by: National Science Foundation grants DDM-8908818 (Fourer) and CCR-8810107 (Mehrotra); and grants from GTE Laboratories (Mehrotra).

References

[1] İ. Adler, N. Karmarkar, M.G.C. Resende and G. Veiga, "An implementation of Karmarkar's algorithm for linear programming." *Mathematical Programming* **44** (1989) 297–336.

[2] M. Arioli, I.S. Duff and P.P.M. de Rijk, "On the augmented system approach to sparse least-squares problems." *Numerische Mathematik* **55** (1989) 667–684.

[3] P.T. Boggs, P.D. Domich, J.R. Donaldson and C. Witzgall, "Optimal 3-dimensional methods for linear programming." MISTIR 89-4225, Center for Computing and Applied Mathematics, National Institute of Standards and Technology, Gaithersburg, MD (1989).

[4] A.A. Björck, "Iterative refinement of linear least-squares solutions." BIT **7** (1967) 257–278.

[5] J.R. Bunch and B.N. Parlett, "Direct methods for solving symmetric indefinite systems of linear equations." *SIAM Journal on Numerical Analysis* **8** (1971) 639–655.

[6] I.C. Choi, C.L. Monma and D.F. Shanno, "Further Development of a Primal-Dual Interior Point Method." *ORSA Journal on Computing* **2** (1990) 304–311.

[7] I.S. DUFF, N.I.M. GOULD, J.K. REID, J.A. SCOTT and K. TURNER, "The factorization of sparse symmetric indefinite matrices." Report RAI-90-066, Rutherford Appleton Laboratory, Oxon (1990).

[8] I.S. DUFF and J.K. REID, "The multifrontal solution of indefinite sparse symmetric linear equations." *ACM Transactions on Mathematical Software* **9** (1983) 302–325.

[9] P.E. GILL, W. MURRAY, M.A. SAUNDERS, J.A. TOMLIN and M.H. WRIGHT, "On projected Newton barrier methods for linear programming." *Mathematical Programming* **36** (1986) 183–209.

[10] M. KOJIMA, S. MIZUNO and A. YOSHISE, "A $O(\sqrt{n}L)$ potential reduction algorithm for linear complementarity problems." Research Report B-217, Department of Information Sciences, Tokyo Institute of Technology (1988).

[11] I. LUSTIG, R.E. MARSTEN and D.F. SHANNO, "On Implementing Mehrotra's Predictor-Corrector Interior Point Method for Linear Programming." Technical Report SOR 90-03, Department of Civil Engineering and Operations Research, Princeton University (1990).

[12] I. LUSTIG, J.M. MULVEY and T.J. CARPENTER, "Formulating Stochastic programs for interior point methods." Technical Report SOR 89-16, Department of Civil Engineering and Operations Research, Princeton University (1989).

[13] K.A. MCSHANE, C.L. MONMA and D.F. SHANNO, "An implementation of a primal-dual interior point method for linear programming." *ORSA Journal on Computing* **1** (1989) 70–83.

[14] S. MEHROTRA, "On the implementation of a (primal-dual) interior point method." Technical Report 90-03, Department of Industrial Engineering and Management Sciences, Northwestern University (1990).

[15] S. MEHROTRA, "Performance of higher order methods." Technical Report 90-16R, Department of Industrial Engineering and Management Sciences, Northwestern University (1990).

[16] S. MEHROTRA, "Handling free variables in interior methods." Technical Report 91-06, Department of Industrial Engineering and Management Sciences, Northwestern University (1991).

[17] R.C. MONTEIRO and I. ADLER, "An $O(n^3L)$ primal-dual interior point algorithm for linear programming." *Mathematical Programming* **44** (1989) 43–66.

[18] R.D. SKEEL, "Scaling for numerical stability in Gaussian elimination." *Journal of the Association for Computing Machinery* **26** (1979) 494–526.

[19] R.J. VANDERBEI, "Splitting dense columns in sparse linear systems." Manuscript, AT&T Bell Laboratories, Murray Hill, NJ (1990).

[20] R.J. VANDERBEI, "ALPO: another linear program solver." Manuscript, AT&T Bell Laboratories, Murray Hill, NJ (1990).

problem	*Augmented* it	time	*Normal* it	time
25fv47	26	259.84	26	164.66
80bau3b	48	814.16	46	444.44
adlittle	10	1.20	10	.82
afiro	7	.29	7	.23
agg	25	15.12	25	33.83
agg2	22	101.61	22	71.15
agg3	20	90.35	20	77.88
bandm	17	11.53	17	8.36
beaconfd	7	4.04	7	2.55
blend	10	1.53	10	1.38
bnl1	29	82.94	29	54.89
bnl2	36	1092.99	36	1113.57
boeing1	25	32.51	26	23.93
boeing2	19	8.26	19	6.84
bore3d	17	3.76	17	4.10
brandy	20	14.25	20	7.99
capri	20	16.11	31	22.54
cycle	31	354.44	24	370.79
czprob	36	73.93	35	42.83
d2q06c	30	1770.20	30	2967.94
degen2	12	57.93	12	34.70
degen3	16	1056.70	16	634.72
e226	20	19.21	20	10.08
etamacro	30	64.41	32	60.12
ffff800	37	104.11	38	87.80
finnis	26	24.87	25	23.07
fit1d	18	36.30	21	23.58
fit1p	18	33.58		∞
fit2d	23	447.15	24	266.52
fit2p	22	260.21		∞
forplan	24	22.68	23	16.03
ganges	18	60.73	19	92.23
gfrd-pnc	16	16.19	17	9.98
greenbea	40	439.75	39	(6)
greenbeb	40	385.11	36	(5)
grow15	12	21.59	12	13.82
grow22	13	37.60	14	21.92
grow7	12	9.81	12	6.27
israel	24	15.02	24	57.06
kb2	20	1.87	20	1.39
lotfi	14	4.54	14	3.43
nesm	34	170.54	35	152.00
perold	32	216.78	27	(3)

problem	*Augmented* it	time	*Normal* it	time
pilotnov	25	369.50	25	350.62
pilot.ja	44	665.84	34	(3)
pilot.we	51	225.83	45	(3)
pilot4	43	95.50	32	(3)
pilot	44	4505.81	43	3942.65
recipe	10	2.12	11	1.53
sc105	9	1.31	9	.89
sc205	11	3.00	11	2.15
sc50a	8	.60	8	.41
sc50b	6	.38	6	.33
scagr25	17	9.15	17	7.39
scagr7	13	1.97	13	1.55
scfxm1	18	16.14	18	10.84
scfxm2	20	36.26	20	24.49
scfxm3	20	52.48	20	38.29
scorpion	12	9.51	12	4.17
scrs8	21	24.10	21	16.69
scsd1	8	5.35	8	2.97
scsd6	10	11.92	10	6.10
scsd8	9	21.45	9	12.07
sctap1	15	9.55	15	5.83
sctap2	13	45.56	13	33.10
sctap3	14	55.92	14	42.28
seba	18	16.16	22	493.91
share1b	22	6.34	22	3.88
share2b	12	2.40	12	1.70
shell	20	24.90	20	15.57
ship04l	12	19.71	12	11.02
ship04s	13	18.89	13	7.77
ship08l	14	38.63	14	20.89
ship08s	14	21.39	14	10.94
ship12l	18	63.93	18	33.18
ship12s	16	27.83	16	15.14
sierra	21	125.67	22	55.41
stair	16	30.17	12	(2)
standata	13	12.73	14	8.86
standmps	20	23.72	20	16.58
stocfor1	16	2.49	16	1.92
stocfor2	24	93.51	24	77.79
tuff	20	31.39	19	23.39
vtp.base	18	2.66	19	2.51
woodlp	29	289.03	29	175.23
woodw	29	287.30	29	187.71

Iteration counts and timings for the augmented system and normal equation approaches. In the column for normal equation timings, ∞ indicates that the equations were too large and dense to solve on the computer used; (k) means that the algorithm failed after k digits of accuracy were achieved.