RAL-TR-2004-036

On implicit-factorization constraint preconditioners

H S Dollar N I M Gould A J Wathen

December 14, 2004

Council for the Central Laboratory of the Research Councils

Library and Information Services
CCLRC Rutherford Appleton Laboratory
Chilton Didcot
Oxfordshire OX11 0QX
UK
Tel: +44 (0)1235 445384
Fax: +44(0)1235 446403
Email: library@rl.ac.uk

CCLRC reports are available online at:
http://www.clrc.ac.uk/Activity/ACTIVITY=Publications;SECTION=225;

ISSN 1358-6254

RAL-TR-2004-036

On implicit-factorization constraint preconditioners

H. Sue Dollar[1,2,3], Nicholas I. M. Gould[4,5,6] and Andrew J. Wathen[1,2]

ABSTRACT

Recently Dollar and Wathen [14] proposed a class of incomplete factorizations for saddle-point problems, based upon earlier work by Schilders [40]. In this paper, we generalize this class of preconditioners, and examine the spectral implications of our approach. Numerical tests indicate the efficacy of our preconditioners.

---

[1] Oxford University Computing Laboratory, Numerical Analysis Group, Wolfson Building, Parks Road, Oxford, OX1 3QD, England, UK.
Email: Sue.Dollar@comlab.ox.ac.uk & Andy.Wathen@comlab.ox.ac.uk

[2] Current reports available from
"http://web.comlab.ox.ac.uk/oucl/publications/natr/index.html".

[3] This work was supported by the O.U.C.L. Doctoral Training Account.

[4] Computational Science and Engineering Department, Rutherford Appleton Laboratory, Chilton, Oxfordshire, OX11 0QX, England, UK.
Email: n.i.m.gould@rl.ac.uk

[5] Current reports available from
"http://www.numerical.rl.ac.uk/reports/reports.shtml".

[6] This work was supported by the EPSRC grant GR/S42170.

Computational Science and Engineering Department
Atlas Centre
Rutherford Appleton Laboratory
Oxfordshire OX11 0QX

December 14, 2004

1 Introduction

Given a symmetric n by n matrix H and a full-rank m ($\leq n$) by n matrix A, we are interested in solving structured linear systems of equations

$$\begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}. \tag{1.1}$$

Such "saddle-point" systems arise as stationarity (KKT) conditions for equality-constrained optimization [37, §18.1], in mixed finite-element approximation of elliptic problems [5], including in particular problems of elasticity [38] and incompressible flow [19], as well as other areas.

In this paper, we are particularly interested in solving (1.1) by iterative methods, in which so-called constraint preconditioners [33]

$$K_G = \begin{pmatrix} G & A^T \\ A & 0 \end{pmatrix} \tag{1.2}$$

are used to accelerate the iteration for some suitable symmetric G. In Section 2, we examine the spectral implications of such methods, and consider how to choose G to give favourable eigenvalue distributions. In Section 3, we then extend ideas by Dollar, Schilders and Wathen [14, 40] to construct "implicit" constraint preconditioners for which we can apply the eigenvalue bounds from Section 2. We demonstrate the effectiveness of such an approach in Section 4 and make broad conclusions in Section 5.

Notation

Let I by the (appropriately-dimensioned) identity matrix. Given a symmetric matrix M with, respectively, m_+, m_- and m_0 positive, negative and zero eigenvalues, we denote its inertia by $\text{In}(M) = (m_+, m_-, m_0)$.

2 Constraint preconditioners

2.1 General considerations

For K_G to be a meaningful preconditioner for certain Krylov-based methods [27], it is vital that its inertia satisfies

$$\text{In}(K_G) = (n, m, 0). \tag{2.1}$$

A key result concerning the use of K_G as a preconditioner is as follows.

Theorem 2.1. [33, Thm. 2.1] or, for diagonal G, [34, Thm. 3.3]. *Suppose that K_H is the coefficient matrix of (1.1), and N is any (n by $n-m$) basis matrix for the null-space of A. Then $K_G^{-1}K_H$ has $2m$ unit eigenvalues, and the remaining $n-m$ eigenvalues are those of the generalized eigenproblem*

$$N^T H N v = \lambda N^T G N v. \tag{2.2}$$

The eigenvalues of (2.2) are real since (2.1) is equivalent to $N^T G N$ being positive definite [7, 26].

Although we are not expecting or requiring that G (or H) be positive definite, it is well-known that this is often not a significant handicap.

Theorem 2.2. [1, Cor. 12.9, or 12, for example]. *The inertial requirement (2.1) holds for a given G if and only if there exists a positive semi-definite matrix $\bar{D}$ such that $G + A^T DA$ is positive definite for all D for which $D - \bar{D}$ is positive semi-definite.*

Since any preconditioning system

$$\begin{pmatrix} G & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} r \\ s \end{pmatrix} \tag{2.3}$$

may equivalently be written as

$$\begin{pmatrix} G + A^T DA & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} u \\ w \end{pmatrix} = \begin{pmatrix} r \\ s \end{pmatrix} \tag{2.4}$$

where $w = v - DAu$, there is little to be lost (save sparsity in G) in using (2.4), with its positive-definite leading block, rather than (2.3). This observation has allowed Golub, Greif and Varah [25, 31] to suggest[1] a variety of methods for solving (1.1) in the case that H is positive semi-definite, although the scope of their suggestions does not appear fundamentally to be limited to this case. Lukšan and Vlček [34] make related suggestions for more general G.

Note, however, that although Theorem 2.2 implies the existence of a suitable D, it alas does not provide a suitable value. In [31], the authors propose heuristics to use as few nonzero components of D as possible (on sparsity grounds) when G is positive semi-definite, but it is unclear how this extends for general G. Golub, Greif and Varah's methods aim particularly to produce well-conditioned $G + A^T DA$. Notice, though, that perturbations of this form do not change the eigenvalue distribution alluded to in Theorem 2.1, since if $H(D_H) = H + A^T D_H A$ and $G(D_G) = G + A^T D_G A$, for (possibly different) D_H and D_G,

$$N^T H(D_H) N = N^T H N v = \lambda N^T G N v = \lambda N^T G(D_G) N v.$$

and thus the generalized eigen-problem (2.2), and hence eigenvalues of $K_{G(D_G)}^{-1} K_{H(D_H)}$, are unaltered.

[1] They actually propose the alternative

$$\begin{pmatrix} G + A^T DA & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} r + A^T Ds \\ s \end{pmatrix}$$

although this is not significant.

2.2 Improved eigenvalue bounds with the reduced-space basis

In this paper, we shall suppose that we may partition the columns of A so that

$$A = (A_1 \;\; A_2),$$

and so that its leading m by m sub-matrix

A1 A_1 and its transpose are easily invertible.

Since there is considerable flexibility in choosing the "basis" A_1 from the rectangular matrix A by suitable column interchanges, assumption **A1** is often easily, and sometimes trivially, satisfied. Note that the problem of determining the "sparsest" A_1 is NP hard, [8,9], while numerical considerations must be given to ensure that A_1 is not badly conditioned if at all possible [23]. More generally, we do not necessarily assume that A_1 is sparse or has a sparse factorization, merely that there are effective ways to solve systems involving A_1 and A_1^T. For example, for many problems involving constraints arising from the discretization of partial differential equations, there are highly effective *iterative* methods for such systems [4].

Given **A1**, we shall be particularly concerned with the *reduced-space* basis matrix

$$N = \begin{pmatrix} R \\ I \end{pmatrix}, \quad \text{where} \quad R = -A_1^{-1} A_2. \tag{2.5}$$

Such basis matrices play vital roles in Simplex (pivoting)-type methods for linear programming [2,20], and more generally in active-set methods for nonlinear optimization [23,35,36].

Suppose that we partition G and H so that

$$G = \begin{pmatrix} G_{11} & G_{21}^T \\ G_{21} & G_{22} \end{pmatrix} \quad \text{and} \quad H = \begin{pmatrix} H_{11} & H_{21}^T \\ H_{21} & H_{22} \end{pmatrix}, \tag{2.6}$$

where G_{11} and H_{11} are (respectively) the leading m by m sub-matrices of G and H. Then (2.5) and (2.6) give

$$\begin{array}{rl} N^T G N = & G_{22} + R^T G_{21}^T + G_{21} R + R^T G_{11} R \\ \text{and} \;\; N^T H N = & H_{22} + R^T H_{21}^T + H_{21} R + R^T H_{11} R \end{array}.$$

In order to improve the eigenvalue distribution resulting from our attempts to precondition K_H by K_G, we consider the consequences of picking G to reproduce certain portions of H.

First, consider the case where

$$G_{22} = H_{22}, \quad \text{but} \quad G_{11} = 0 \quad \text{and} \quad G_{21} = 0. \tag{2.7}$$

Theorem 2.3. *Suppose that G and H are as in (2.6) and that (2.7) holds. Suppose furthermore that H_{22} is positive definite, and let*

$$\rho = \min\left[\text{rank}(A_2), \text{rank}(H_{21})\right] + \min\left[\text{rank}(A_2), \text{rank}(H_{21}) + \min[\text{rank}(A_2), \text{rank}(H_{11})]\right].$$

Then $K_G^{-1}K_H$ has at most

$$\mathrm{rank}(R^TH_{21}^T + H_{21}R + R^TH_{11}R) + 1 \leq \min(\rho, n-m) + 1 \leq \min(2m, n-m) + 1$$

distinct eigenvalues.

Proof. Elementary bounds involving the products and sums of matrices show that the difference

$$N^THN - N^TGN = R^TH_{21}^T + H_{21}R + R^TH_{11}R$$

is a matrix of rank at most $\min(\rho, n-m)$. Since N^TGN is, by assumption, positive definite, we may write $N^TGN = W^TW$ for some non-singular W. Thus

$$W^{-1}N^THNW^{-T} = I + W^{-1}(R^TH_{21}^T + H_{21}R + R^TH_{11}R)W^{-T}$$

differs from the identity matrix by a matrix of rank at most $\min(\rho, n-m)$, and hence the generalized eigenproblem (2.2) has at most $\min(\rho, n-m)$ non-unit eigenvalues. □

As we have seen from Theorem 2.2, the restriction that H_{22} be positive definite is not as severe as it might first seem, particularly if we can entertain the possibility of using the positive-definite $H_{22} + A_2^TDA_2$ instead.

The eigenvalue situation may be improved if we consider the case where

$$G_{22} = H_{22} \text{ and } G_{11} = H_{11} \text{ but } G_{21} = 0. \tag{2.8}$$

Theorem 2.4. *Suppose that G and H are as in (2.6) and that (2.8) holds. Suppose furthermore that $H_{22} + R^TH_{11}^TR$ is positive definite, and that*

$$\nu = 2\min\left[\mathrm{rank}(A_2), \mathrm{rank}(H_{21})\right].$$

Then $K_G^{-1}K_H$ has at most

$$\mathrm{rank}(R^TH_{11}R) + 1 \leq \nu + 1 \leq \min(2m, n-m) + 1$$

distinct eigenvalues.

Proof. The result follows as before since now $N^THN - N^TGN = R^TH_{21}^T + H_{21}R$ is of rank at most ν. □

The same is true when

$$G_{22} = H_{22} \text{ and } G_{21} = H_{21} \text{ but } G_{11} = 0. \tag{2.9}$$

Theorem 2.5. *Suppose that G and H are as in (2.6) and that (2.9) holds. Suppose furthermore that $H_{22} + R^TH_{21}^T + H_{21}R$ is positive definite, and that*

$$\mu = \min\left[\mathrm{rank}(A_2), \mathrm{rank}(H_{11})\right].$$

Then $K_G^{-1}K_H$ has at most

$$\text{rank}(R^T H_{11} R) + 1 \leq \mu + 1 \leq \min(m, n-m) + 1$$

distinct eigenvalues.

Proof. The result follows, once again, as before since now $N^T HN - N^T GN = R^T H_{11} R$ is of rank at most μ. □

In Tables 2.1 and 2.2, we illustrate these results by considering the complete set of linear and quadratic programming examples from the Netlib [21] and CUTEr [29] test sets. All inequality constraints are converted to equations by adding slack variables, and a suitable "barrier" penalty term (in this case, 1.0) is added to the diagonal of H for each bounded or slack variable to simulate systems that might arise during an iteration of an interior-point method for such problems.

Given A, a suitable basis matrix A_1 is found by finding a sparse LU factorization of A^T using the HSL [32] packages `MA48` and `MA51` [17]. An attempt to correctly identify rank is controlled by tight threshold column pivoting, in which any pivot may not be smaller than a factor $\tau = 2$ of the largest entry in its (uneliminated) column [23, 24]. The rank is estimated as the number of pivots, $\rho(A)$, completed before the remaining uneliminated sub-matrix is judged to be numerically zero, and the indices of the $\rho(A)$ pivotal rows and columns of A define A_1—if $\rho(A) < m$, the remaining rows of A are judged to be dependent, and are discarded.[2] Although such a strategy may not be as robust as, say, a singular-value decomposition or a QR factorization with pivoting, both our and others' experience [23] indicate it to be remarkably reliable and successful in practice.

Having found A_1, the factors are discarded, and a fresh LU decomposition of A_1, with a looser threshold column pivoting factor $\tau = 100$, is computed in order to try to encourage sparse factors. All other estimates of rank in Tables 2.1 and 2.2 are obtained in the same way. The columns headed "iteration bounds" illustrate Theorems 2.1 ("any G"), 2.3 ("exact H_{22}") and 2.5 ("exact H_{22} & H_{21}"). Note that in the linear programming case, $H_{21} \equiv 0$, so that we have omitted the "exact H_{22}" statistics from Tables 2.1, since these would be identical to those reported as "exact H_{22} & H_{21}".

Table 2.1: NETLIB LP problems

							iteration bound		
			rank				any G	exact H_{22} & H_{21}	
name	n	m	A	A_2	H_{11}	H_{12}		$\mu+1$	upper
25FV47	1876	821	820	725	820	0	1057	726	822
80BAU3B	12061	2262	2262	2231	2262	0	9800	2232	2263
ADLITTLE	138	56	56	53	56	0	83	54	57
AFIRO	51	27	27	21	27	0	25	22	25
AGG2	758	516	516	195	516	0	243	196	243
AGG3	758	516	516	195	516	0	243	196	243
AGG	615	488	488	123	488	0	128	124	128

[2]Note that if this happens, the right-hand inequalities in Theorems 2.3–2.5 will depend on $n - \text{rank}(A)$ not $n - m$.

Table 2.1: NETLIB LP problems (continued)

			rank				iteration bound		
							any G	exact H_{22} & H_{21}	
name	n	m	A	A_2	H_{11}	H_{12}		$\mu+1$	upper
BANDM	472	305	305	161	305	0	168	162	168
BCDOUT	7078	5414	5412	1102	2227	0	1667	1103	1667
BEACONFD	295	173	173	116	173	0	123	117	123
BLEND	114	74	74	37	74	0	41	38	41
BNL1	1586	643	642	458	642	0	945	459	644
BNL2	4486	2324	2324	1207	2324	0	2163	1208	2163
BOEING1	726	351	351	314	351	0	376	315	352
BOEING2	305	166	166	109	166	0	140	110	140
BORE3D	334	233	231	73	231	0	104	74	104
BRANDY	303	220	193	98	193	0	111	99	111
CAPRI	482	271	271	144	261	0	212	145	212
CYCLE	3371	1903	1875	1272	1868	0	1497	1273	1497
CZPROB	3562	929	929	732	929	0	2634	733	930
D2Q06C	5831	2171	2171	2059	2171	0	3661	2060	2172
D6CUBE	6184	415	404	403	404	0	5781	404	416
DEGEN2	757	444	442	295	442	0	316	296	316
DEGEN3	2604	1503	1501	1052	1501	0	1104	1053	1104
DFL001	12230	6071	6058	5313	6058	0	6173	5314	6072
E226	472	223	223	186	223	0	250	187	224
ETAMACRO	816	400	400	341	400	0	417	342	401
FFFFF800	1028	524	524	290	524	0	505	291	505
FINNIS	1064	497	497	456	497	0	568	457	498
FIT1D	1049	24	24	24	24	0	1026	25	25
FIT1P	1677	627	627	627	627	0	1051	628	628
FIT2D	10524	25	25	25	25	0	10500	26	26
FIT2P	13525	3000	3000	3000	3000	0	10526	3001	3001
FORPLAN	492	161	161	100	161	0	332	101	162
GANGES	1706	1309	1309	397	1309	0	398	398	398
GFRD-PNC	1160	616	616	423	616	0	545	424	545
GOFFIN	101	50	50	50	0	0	52	1	51
GREENBEA	5598	2392	2389	2171	2389	0	3210	2172	2393
GREENBEB	5598	2392	2389	2171	2386	0	3210	2172	2393
GROW15	645	300	300	300	300	0	346	301	301
GROW22	946	440	440	440	440	0	507	441	441
GROW7	301	140	140	140	140	0	162	141	141
SIERRA	2735	1227	1217	768	1217	0	1519	769	1228
ISRAEL	316	174	174	142	174	0	143	143	143
KB2	68	43	43	25	43	0	26	26	26
LINSPANH	97	33	32	32	32	0	66	33	34
LOTFI	366	153	153	110	153	0	214	111	154
MAKELA4	61	40	40	21	40	0	22	22	22
MAROS-R7	9408	3136	3136	3136	3136	0	6273	3137	3137
MAROS	1966	846	846	723	846	0	1121	724	847
MODEL	1557	38	38	11	38	0	1520	12	39
MODSZK1	1620	687	686	667	684	0	935	668	688
BCDOUT	7078	5414	5412	1107	5028	0	1667	1108	1667
NESM	3105	662	662	568	662	0	2444	569	663
OET1	1005	1002	1002	3	1000	0	4	4	4
OET3	1006	1002	1002	4	1000	0	5	5	5
PEROLD	1506	625	625	532	562	0	882	533	626
PILOT4	1123	410	410	367	333	0	714	334	411
PILOT87	6680	2030	2030	1914	2030	0	4651	1915	2031
PILOT-JA	2267	940	940	783	903	0	1328	784	941
PILOTNOV	2446	975	975	823	975	0	1472	824	976
PILOT	4860	1441	1441	1354	1441	0	3420	1355	1442
PILOT-WE	2928	722	722	645	662	0	2207	646	723
PT	503	501	501	2	499	0	3	3	3
QAP8	1632	912	853	697	853	0	780	698	780

Table 2.1: NETLIB LP problems (continued)

			rank				iteration bound		
							any G	exact H_{22} & H_{21}	
name	n	m	A	A_2	H_{11}	H_{12}		$\mu+1$	upper
QAP12	8856	3192	3089	2783	3089	0	5768	2784	3193
QAP15	22275	6330	6285	5632	6285	0	15991	5633	6331
QPBD_OUT	442	211	211	176	211	0	232	177	212
READING2	6003	4000	4000	2001	2001	0	2004	2002	2004
RECIPELP	204	91	91	78	91	0	114	79	92
SC105	163	105	105	58	105	0	59	59	59
SC205	317	205	205	112	205	0	113	113	113
SC50A	78	50	50	28	50	0	29	29	29
SC50B	78	50	50	28	50	0	29	29	29
SCAGR25	671	471	471	199	471	0	201	200	201
SCAGR7	185	129	129	56	129	0	57	57	57
SCFXM1	600	330	330	217	330	0	271	218	271
SCFXM2	1200	660	660	440	660	0	541	441	541
SCFXM3	1800	990	990	660	990	0	811	661	811
SCORPION	466	388	388	77	388	0	79	78	79
SCRS8	1275	490	490	341	490	0	786	342	491
SCSD1	760	77	77	77	77	0	684	78	78
SCSD6	1350	147	147	147	147	0	1204	148	148
SCSD8	2750	397	397	397	397	0	2354	398	398
SCTAP1	660	300	300	246	300	0	361	247	301
SCTAP2	2500	1090	1090	955	1090	0	1411	956	1091
SCTAP3	3340	1480	1480	1264	1480	0	1861	1265	1481
SEBA	1036	515	515	479	515	0	522	480	516
SHARE1B	253	117	117	72	117	0	137	73	118
SHARE2B	162	96	96	65	96	0	67	66	67
SHELL	1777	536	535	489	535	0	1243	490	537
SHIP04L	2166	402	360	343	360	0	1807	344	403
SHIP04S	1506	402	360	256	360	0	1147	257	403
SHIP08L	4363	778	712	679	712	0	3652	680	779
SHIP08S	2467	778	712	406	712	0	1756	407	779
SHIP12L	5533	1151	1042	828	1042	0	4492	829	1152
SHIP12S	2869	1151	1042	451	1042	0	1828	452	1152
SIERRA	2735	1227	1217	768	1217	0	1519	769	1228
SIPOW1M	2002	2000	2000	2	2000	0	3	3	3
SIPOW1	2002	2000	2000	2	1999	0	3	3	3
SIPOW2M	2002	2000	2000	2	2000	0	3	3	3
SIPOW2	2002	2000	2000	2	1999	0	3	3	3
SIPOW3	2004	2000	2000	4	1999	0	5	5	5
SIPOW4	2004	2000	2000	4	1999	0	5	5	5
SSEBLIN	218	72	72	72	72	0	147	73	73
STAIR	614	356	356	249	356	0	259	250	259
STANDATA	1274	359	359	283	359	0	916	284	360
STANDGUB	1383	361	360	281	360	0	1024	282	362
STANDMPS	1274	467	467	372	467	0	808	373	468
STOCFOR1	165	117	117	48	117	0	49	49	49
STOCFOR2	3045	2157	2157	888	2157	0	889	889	889
STOCFOR3	23541	16675	16675	6866	16675	0	6867	6867	6867
TFI2	104	101	101	3	100	0	4	4	4
TRUSS	8806	1000	1000	1000	1000	0	7807	1001	1001
TUFF	628	333	302	207	301	0	327	208	327
VTP-BASE	346	198	198	86	198	0	149	87	149
WOOD1P	2595	244	244	244	244	0	2352	245	245
WOODW	8418	1098	1098	1098	1098	0	7321	1099	1099

Table 2.2: CUTEr QP problems

							iteration bound				
			rank				any G	exact H_{22}		exact H_{22} & H_{21}	
name	n	m	A	A_2	H_{11}	H_{12}		$\rho+1$	upper	$\mu+1$	upper
AUG2DCQP	20200	10000	10000	10000	10000	0	10201	10001	10201	10001	10001
AUG2DQP	20200	10000	10000	10000	10000	0	10201	10001	10201	10001	10001
AUG3DCQP	27543	8000	8000	7998	8000	0	19544	7999	16001	7999	8001
AUG3DQP	27543	8000	8000	7998	8000	0	19544	7999	16001	7999	8001
BLOCKQP1	10011	5001	5001	5001	5001	5000	5011	5011	5011	5002	5002
BLOCKQP2	10011	5001	5001	5001	5001	5000	5011	5011	5011	5002	5002
BLOCKQP3	10011	5001	5001	5001	5001	5000	5011	5011	5011	5002	5002
BLOWEYA	4002	2002	2002	2000	2002	2000	2001	2001	2001	2001	2001
BLOWEYB	4002	2002	2002	2000	2002	2000	2001	2001	2001	2001	2001
BLOWEYC	4002	2002	2002	2000	2002	2000	2001	2001	2001	2001	2001
CONT-050	2597	2401	2401	192	2401	0	197	193	197	193	197
CONT-101	10197	10098	10098	99	10098	0	100	100	100	100	100
CONT-201	40397	40198	40198	199	40198	0	200	200	200	200	200
CONT5-QP	40601	40200	40200	401	40200	0	402	402	402	402	402
CONT1-10	10197	9801	9801	392	9801	0	397	393	397	393	397
CONT1-20	40397	39601	39601	792	39601	0	797	793	797	793	797
CONT-300	90597	90298	90298	299	90298	0	300	300	300	300	300
CVXQP1	10000	5000	5000	2000	5000	2000	5001	4001	5001	2001	5001
CVXQP2	10000	2500	2500	2175	2500	1194	7501	3370	5001	2176	2501
CVXQP3	10000	7500	7500	1000	7500	2354	2501	2001	2501	1001	2501
DEGENQP	125050	125025	125024	26	125024	0	27	27	27	27	27
DUALC1	223	215	215	8	215	0	9	9	9	9	9
DUALC2	235	229	229	6	229	0	7	7	7	7	7
DUALC5	285	278	278	7	278	0	8	8	8	8	8
DUALC8	510	503	503	7	503	0	8	8	8	8	8
GOULDQP2	19999	9999	9999	9999	9999	0	10001	10000	10001	10000	10000
GOULDQP3	19999	9999	9999	9999	9999	9999	10001	10001	10001	10000	10000
KSIP	1021	1001	1001	20	1001	0	21	21	21	21	21
MOSARQP1	3200	700	700	700	700	3	2501	704	1401	701	701
NCVXQP1	10000	5000	5000	2000	5000	2000	5001	4001	5001	2001	5001
NCVXQP2	10000	5000	5000	2000	5000	2000	5001	4001	5001	2001	5001
NCVXQP3	10000	5000	5000	2000	5000	2000	5001	4001	5001	2001	5001
NCVXQP4	10000	2500	2500	2175	2500	1194	7501	3370	5001	2176	2501
NCVXQP5	10000	2500	2500	2175	2500	1194	7501	3370	5001	2176	2501
NCVXQP6	10000	2500	2500	2175	2500	1194	7501	3370	5001	2176	2501
NCVXQP7	10000	7500	7500	1000	7500	2354	2501	2001	2501	1001	2501
NCVXQP8	10000	7500	7500	1000	7500	2354	2501	2001	2501	1001	2501
NCVXQP9	10000	7500	7500	1000	7500	2354	2501	2001	2501	1001	2501
POWELL20	10000	5000	5000	4999	5000	0	5001	5000	5001	5000	5001
PRIMALC1	239	9	9	9	9	0	231	10	19	10	10
PRIMALC2	238	7	7	7	7	0	232	8	15	8	8
PRIMALC5	295	8	8	8	8	0	288	9	17	9	9
PRIMALC8	528	8	8	8	8	0	521	9	17	9	9
PRIMAL1	410	85	85	85	85	0	326	86	171	86	86
PRIMAL2	745	96	96	96	96	0	650	97	193	97	97
PRIMAL3	856	111	111	111	111	0	746	112	223	112	112
PRIMAL4	1564	75	75	75	75	0	1490	76	151	76	76
QPBAND	75000	25000	25000	25000	25000	0	50001	25001	50001	25001	25001
QPNBAND	75000	25000	25000	25000	25000	0	50001	25001	50001	25001	25001
QPCBOEI1	726	351	351	314	351	0	376	315	376	315	352
QPCBOEI2	305	166	166	109	166	0	140	110	140	110	140
QPCSTAIR	614	356	356	249	356	0	259	250	259	250	259
QPNBOEI1	726	351	351	314	351	0	376	315	376	315	352
QPNBOEI2	305	166	166	109	166	0	140	110	140	110	140
QPNSTAIR	614	356	356	249	356	0	259	250	259	250	259
SOSQP1	5000	2501	2501	2499	2501	2499	2500	2500	2500	2500	2500
STCQP1	8193	4095	1771	0	1771	317	6423	1	6423	1	4096
STCQP2	8193	4095	4095	0	4095	1191	4099	1	4099	1	4096

Table 2.2: CUTEr QP problems (continued)

			rank				iteration bound				
							any G	exact H_{22}		exact H_{22} & H_{21}	
name	n	m	A	A_2	H_{11}	H_{12}		$\rho+1$	upper	$\mu+1$	upper
STNQP1	8193	4095	1771	0	1771	317	6423	1	6423	1	4096
STNQP2	8193	4095	4095	0	4095	1191	4099	1	4099	1	4096
UBH1	9009	6000	6000	3003	6	0	3010	7	3010	7	3010
YAO	4002	2000	2000	2000	2000	0	2003	2001	2003	2001	2001

We observe that in some cases there are useful gains to be made from trying to reproduce H_{22} and, less often, H_{21}. Moreover, the upper bounds on rank obtained in Theorems 2.3 and 2.5 can be significantly larger than even the estimates $\rho + 1$ and $\mu + 1$ of the number of distinct eigenvalues. However the trend is far from uniform, and in some cases there is little or no apparent advantage to be gained from reproducing portions of H. Nonetheless, since significant improvements are possible, we now investigate efficient ways of computing decompositions which are capable of reproducing sub-blocks of H.

3 Implicit-factorization constraint preconditioners

It has long been common practice (at least in optimization circles) [3,6,10,18,22,34,39,42] to use preconditioners of the form (1.2) by specifying G and factorizing K_G using a suitable symmetric, indefinite package such as MA27 [16] or MA57 [15]. While such techniques have often been successful, they have usually been rather *ad hoc*, with little attempt to improve upon the eigenvalue distributions beyond those suggested by the Theorem 2.1.

Recently, Dollar and Wathen [14] have suggested using a preconditioner of the form

$$K_G = PBP^T, \tag{3.1}$$

where solutions with each of the matrices P, B and P^T are easily obtained. In particular, rather than obtaining P and B from a given K_G, *K_G is derived implicitly from specially chosen P and B.* In this section, we examine a broad class of methods of this form.

3.1 Structural considerations

In general, we may write

$$P = \begin{pmatrix} P_1 & A^T \\ P_2 & 0 \end{pmatrix} \text{ and } B = \begin{pmatrix} B_1 & B_2^T \\ B_2 & B_{33} \end{pmatrix} \tag{3.2}$$

where B_1 and B_{33} are symmetric and P_2 is of full rank; the zero block in P is selected so as to mimic that in K_G. Given this form, we have

$$K_G = \begin{pmatrix} P_1B_1P_1^T + A^TB_2P_1^T + P_1B_2^TA + A^TB_{33}A & P_1B_1P_2^T + A^TB_2P_2^T \\ P_2B_1P_1^T + P_2B_2^TA & P_2B_1P_2^T \end{pmatrix}$$

and since we wish (1.2) to hold, we require that

$$P_2 B_1 P_1^T + P_2 B_2^T A = A \text{ and } P_2 B_1 P_2^T = 0. \tag{3.3}$$

As A and P_2 are of full rank, we write

$$A = (A_1 \;\; A_2) \text{ and } P_2 = (P_{31} \;\; P_{32})$$

for nonsingular m by m matrices A_1 and P_{31}, and shall likewise write

$$P_1 = \begin{pmatrix} P_{11} & P_{12} \\ P_{21} & P_{22} \end{pmatrix}, \;\; B_1 = \begin{pmatrix} B_{11} & B_{21}^T \\ B_{21} & B_{22} \end{pmatrix} \text{ and } B_2 = (B_{31} \;\; B_{32}).$$

The second requirement in (3.3) is then that

$$P_{31} B_{11} P_{31}^T + P_{32} B_{21} P_{31}^T + P_{31} B_{21}^T P_{32}^T + P_{32} B_{22} P_{32}^T = 0.$$

Although there are a number of ways of guaranteeing this,[3] the simplest is to insist that

$$P_{32} = 0 \text{ and } B_{11} = 0.$$

The first requirement in (3.3) may be satisfied if

$$P_2 B_2^T = I \text{ and } P_2 B_1 P_1^T = 0, \tag{3.4}$$

although again there are other (more complicated) possibilities. It then follows that

$$B_{31} = P_{31}^{-T} \text{ and } P_{31} B_{21}^T (P_{12}^T \;\; P_{22}^T) = 0$$

and the second of these implies that

$$B_{21} = 0$$

since P_{31} is non singular and $(P_{12}^T \;\; P_{22}^T)$ must be of full rank.[4] Thus

$$P = \begin{pmatrix} P_{11} & P_{12} & A_1^T \\ P_{21} & P_{22} & A_2^T \\ B_{31}^{-T} & 0 & 0 \end{pmatrix} \text{ and } B = \begin{pmatrix} 0 & 0 & B_{31}^T \\ 0 & B_{22} & B_{32}^T \\ B_{31} & B_{32} & B_{33} \end{pmatrix}, \tag{3.5}$$

where B_{31} and B_{22} are non-singular. Furthermore, it follows trivially from Sylvester's law of inertia (see, for example, [11]) that

$$B_{22} \text{ must be positive definite} \tag{3.6}$$

if (2.1) is to hold.

[3] In general $B_{11} = -P_{31}^{-1}\left(P_{32} B_{21} P_{31}^T + P_{31} B_{21}^T P_{32}^T + P_{32} B_{22} P_{32}^T\right) P_{31}^{-T}$ for any P_{32}.

[4] The latter follows since $P_{32} = 0$ and P is required to be non-singular.

3.2 Solution considerations

3.2.1 Solves involving P and its transpose

Suppose that B_{31} is chosen to be easily invertible—Dollar and Wathen [14] suggest picking $B_{31} = I$, but other simple choices are possible. Then, in order to solve systems involving the block (reverse) triangular matrix P and its transpose, it suffices to be able to do so for systems involving the sub-matrix

$$\begin{pmatrix} P_{12} & A_1^T \\ P_{22} & A_2^T \end{pmatrix}.$$

Although **A1** allows a general (Schur-complement) pivot, in which such systems may be solved knowing factors of A_1 and $P_{22} - R^T P_{12}$, perhaps the easiest possibility is, again, to follow [14] and pick

$$P_{12} = 0. \tag{3.7}$$

This then presupposes that P_{22} is non-singular.

One further saving here in the solution of (2.3) via forward and backward substituting from (3.1) in the usual (preconditioning) case for which $s = 0$ is that the the block zero component of the right-hand-side may trivially be exploited in the initial forward substitution

$$\begin{pmatrix} P_{11} & 0 & A_1^T \\ P_{21} & P_{22} & A_2^T \\ B_{31}^{-T} & 0 & 0 \end{pmatrix} \begin{pmatrix} p_1 \\ p_2 \\ q \end{pmatrix} = \begin{pmatrix} r_1 \\ r_2 \\ 0 \end{pmatrix}$$

for which $p_1 = 0$.

3.2.2 Solves involving B

It follows from (3.5) that solving systems of equations whose coefficient matrix is B relies on being able to solve systems with coefficient matrices B_{31}, B_{22} and B_{31}^T. The choice $B_{31} = I$ made by Dollar and Wathen [14] is again ideal from this perspective.

3.3 Considerations relating to preconditioning

So far, we simply require that P and B satisfy (3.5) in order to ensure K_G is of the form (1.2), but additionally that (3.6) holds for K_G to be a useful preconditioner. Note that without (3.6) we could choose the components of P and B to factorize K_G in the case where $H = G$, but if

$$\text{In} \begin{pmatrix} H & A^T \\ A & 0 \end{pmatrix} \neq (n, m, 0)$$

it will not be possible to find B_{22} satisfying (3.6) in this case.

3.3.1 Recovering G

The leading diagonal block G of K_G is

$$G = P_1 B_1 P_1^T + A^T B_2 P_1^T + P_1 B_2^T A + A^T B_{33} A. \tag{3.8}$$

In what remains, we shall thus assume that P and B_2 are given by (3.5), and that (3.7) holds, that is that

$$P = \begin{pmatrix} P_{11} & 0 & A_1^T \\ P_{21} & P_{22} & A_2^T \\ B_{31}^{-T} & 0 & 0 \end{pmatrix} \text{ and } B = \begin{pmatrix} 0 & 0 & B_{31}^T \\ 0 & B_{22} & B_{32}^T \\ B_{31} & B_{32} & B_{33} \end{pmatrix}. \tag{3.9}$$

It follows immediately from (2.6), (3.8) and (3.9) that

$$\begin{array}{rcl} G_{11} & = & P_{11} B_{31}^T A_1 + A_1^T B_{31} P_{11}^T + A_1^T B_{33} A_1 \\ G_{21} & = & A_2^T B_{31} P_{11}^T + P_{21} B_{31}^T A_1 + P_{22} B_{32}^T A_1 + A_1^T B_{33} A_2 \text{ and} \\ G_{22} & = & P_{22} B_{22} P_{22}^T + P_{21} B_{31}^T A_2 + P_{22} B_{32}^T A_2 + A_2^T B_{31} P_{21}^T + A_2^T B_{32} P_{22}^T + A_2^T B_{33} A_2 \end{array}.$$

Notice that we have not as yet determined P_{11}, P_{21}, P_{22}, B_{22}, B_{31}, B_{32} and B_{33}, but that G involves significantly less information, and thus there is likely to be considerable freedom in our remaining choices even if we wish to recover a particular G.

It follows from (3.8) that

$$N^T G N = N^T P_1 B_1 P_1^T N$$

for any null-space basis matrix N, since $AN = 0$ It also follows from the required form (3.9) of P and B that

$$P_1 B_1 P_1^T = \begin{pmatrix} 0 & 0 \\ 0 & P_{22} B_{22} P_{22}^T \end{pmatrix}$$

and in the case of the reduced-space basis matrix (2.5) we have that

$$N^T G N = P_{22} B_{22} P_{22}^T.$$

3.4 Particular choices of P and B

3.4.1 Existing proposals

Schilders [40] sets $B_{31} = I$ and $B_{32} = 0$, and uses P_{11} and P_{22} as free parameters to determine P_{21}, B_{22} and B_{33} from G. Dollar and Wathen [14] consider the same choices for B_{31} and B_{32}, and use P_{11} and P_{22} and B_{33} as free parameters to determine P_{21}, B_{22} and G_{22} from G_{11} and G_{21}. So for example, if

$$P_{11} = 0, P_{21} = 0, \; P_{22} = I, \; B_{31} = I, \; B_{22} = I, \; B_{32} = 0 \text{ and } B_{33} = 0$$

then

$$G_{11} = 0, \; G_{21} = 0 \text{ and } G_{22} = I.$$

3.4.2 Reproducing H_{22}

The simplest option is to set as many of free components of P and B as possible to zero; this corresponds to setting

$$P_{11} = 0, \quad P_{21} = 0, \quad B_{32} = 0 \text{ and } B_{33} = 0, \tag{3.10}$$

and results in

$$G_{11} = 0, \quad G_{21} = 0 \text{ and } G_{22} = P_{22} B_{22} P_{22}^T.$$

Thus the requirement (3.6) forces G_{22} to be positive definite, and any positive-definite G_{22} may be accommodated by the choice (3.10). In particular, if H_{22} is positive-definite, Theorem 2.3 shows that picking $G_{22} = H_{22}$ leads to an improved eigenvalue bound over that for generic G. In this case P_{22} and B_{22} could accommodate (sparse) Cholesky or LDL^T factors of H_{22}.

3.4.3 Reproducing H_{21} and H_{22}

The choice

$$P_{11} = 0 \text{ and } B_{33} = 0 \tag{3.11}$$

gives

$$\begin{aligned} G_{11} &= 0, \; G_{21} = P_{21} B_{31}^T A_1 + P_{22} B_{32}^T A_1 \text{ and} \\ G_{22} &= P_{22} B_{22} P_{22}^T + P_{21} B_{31}^T A_2 + P_{22} B_{32}^T A_2 + A_2^T B_{31} P_{21}^T + A_2^T B_{32} P_{22}^T. \end{aligned}$$

while choosing

$$P_{11} = 0, \quad B_{32} = 0 \text{ and } B_{33} = 0 \tag{3.12}$$

gives

$$G_{11} = 0, \; G_{21} = P_{21} B_{31}^T A_1 \text{ and } G_{22} = P_{22} B_{22} P_{22}^T + P_{21} B_{31}^T A_2 + A_2^T B_{31} P_{21}^T.$$

Both of these possibilities allow us to choose $G_{22} = H_{22}$ and $G_{21} = H_{21}$, and Theorem 2.5 indicates that such choices lead to further improved eigenvalue bounds. Moreover, in both cases,

$$P_{22} B_{22} P_{22}^T = G_{22} + R^T G_{21}^T + G_{21} R$$

regardless of how we choose P_{21}, B_{31} and B_{32}.

3.4.4 Ensuring that G is positive definite

The role of the matrix B_{33} is interesting. For Theorem 2.2 and (3.8) suggest that by picking B_{33} sufficiently negative definite, the remaining terms

$$P_1 B_1 P_1^T + A^T B_2 P_1^T + P_1 B_2^T A$$

will be positive definite. However, since any significantly dense rows of A will result in dense blocks in $A^T B_{33} A$, it may well be wise to keep $B_{33} = 0$.

3.5 Factors in other orders

We have seen that specifying decompositions of the form (3.1) in which P and B have the block form (3.2) is an extremely flexible approach. A natural question is: are there other block forms which are equally useful? The most obvious alternative is to seek a decomposition

$$K_G = QEQ^T, \tag{3.13}$$

where

$$Q = \begin{pmatrix} Q_1 & Q_2 \\ A & 0 \end{pmatrix} \text{ and } E = \begin{pmatrix} E_1 & E_2^T \\ E_2 & E_{33} \end{pmatrix} \tag{3.14}$$

where E_1 and E_{33} are symmetric and Q_2 is of full rank; here again the zero block in Q is selected so as to mimic that in K_G. In this case

$$K_G = \begin{pmatrix} Q_1E_1Q_1^T + Q_2E_2Q_1^T + Q_1E_2^TQ_2^T + Q_2E_{33}Q_2^T & Q_1E_1A^T + Q_2E_2A^T \\ AE_1Q_1^T + AE_2^TQ_2^T & AE_1A^T \end{pmatrix}. \tag{3.15}$$

But now we see a strong disadvantage of (3.13) compared with (3.1), namely that requiring that the 2,1 and 2,2 blocks of (3.15) reproduce A and 0 respectively place strong restrictions on E_1, E_2, Q_1 and Q_2. In particular, E_1A^T must lie in the null-space of A. Since this seems to limit the scope of (3.13)–(3.14) we do not pursue this further.

4 Numerical experiments

In this section we indicate that, in some cases, the implicit-factorization preconditioners proposed in Section 3 are very effective in practice.

We consider the set of quadratic programming examples from the CUTEr test set examined in Section 2. For each, we use the projected preconditioned conjugate-gradient method [27] to solve the resulting quadratic programming problem

$$\text{EQP:} \quad \underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; q(x) = \tfrac{1}{2}x^THx + c^Tx \;\; \text{subject to} \;\; Ax = b.$$

Firstly a feasible point $x = x_0$ is determined. Thereafter, iterates $x_0 + s$ generated by the conjugate-gradient method are constrained to satisfy $As = 0$ by means of the preconditioning system (2.3). Since, as frequently happens in practice, $q(x_0+s)$ may be unbounded from below, a trust-region constraint $\|s\| \leq \Delta$ is also imposed, and the Generalized Lanczos Trust-Region (GLTR) method [28], as implemented in the GALAHAD library [30], is used to solve the resulting problem

$$\underset{x \in \mathbb{R}^n}{\text{minimize}} \;\; q(x_0 + s) \;\; \text{subject to} \;\; As = 0 \;\; \text{and} \;\; \|s\| \leq \Delta; \tag{4.1}$$

a large value of $\Delta = 10^{10}$ is used so as not to cut off the unconstrained solution for convex problems.

In Tables 4.1 and 4.2, we compare four preconditioning strategies for (approximately) solving the problem (4.1). We consider both low and high(er) accuracy solutions. For the former, we terminate as soon as the norm of the (preconditioned) gradient of $q(x_0 + s)$ has been reduced more than 10^{-2} from that of $q(x_0)$, while the latter requires a 10^{-8} reduction; these are intended to simulate the levels of accuracy required within a nonlinear programming solver in early (global) and later (asymptotic) phases of the solution process.

We consider two explicit factorizations, one using exact factors ($G = H$), and the other using a simple projection ($G = I$). The HSL package `MA57` [15] (version 2.2.1) is used to factorize K_G and subsequently solve (2.3); by way of comparison, we also include times for exact factorization with the earlier `MA27` [16], since this is still widely used. Two implicit factorizations of the form (3.1) with factors (3.9) are also considered. In the first, we use the method in Section 3.4.1 to get $G_{22} = I$. The second follows Section 3.4.2 and aims to reproduce $G_{22} = H_{22}$, and uses `MA57` to compute its factors. In particular, we exploit one of `MA57`'s options to make modest modifications [41] of the diagonals of H_{22} to ensure that G_{22} is positive definite if H_{22} fails to be—this proved only to be necessary for the `BLOWEY*` problems.

All of our experiments were performed using a single processor of a 3.05Mhz Dell Precision 650 Workstation with 4 Gbytes of RAM. Our codes were written in double precision fortran 90, compiled using the Intel ifort 8.1 compiler, and wherever possible made use of tuned ATLAS BLAS [43] for core computations. A single iteration of iterative refinement is applied, as necessary, when applying the preconditioner (2.3) to try to ensure small relative residuals.

For each option tested, we record the time taken to compute the (explicit or implicit) factors, the number of GLTR iterations performed (equivalently, the number of preconditioned systems solved), and the total time taken to solve the quadratic programming problem EQP (including the factorization). The initial feasible point x_0 is found by solving

$$\begin{pmatrix} G & A^T \\ A & 0 \end{pmatrix} \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} 0 \\ b \end{pmatrix}$$

using the factors of K_G. Occasionally—in particular when $c = 0$ and $G = H$—such a point solves EQP, and the resulting iteration count is zero. In a few cases, the problems are so ill-conditioned that the trust-region constraint is activated, and more than one GLTR iteration is required to solve EQP even when $G = H$. Furthermore, rank deficiency of A occasionally resulted in unacceptably large residuals in (2.3) and subsequent failure of GLTR when $G = H$, even after iterative refinement.

In many cases, the use of an "exact" preconditioner $G = H$ is cost effective, particularly when the newer factorization package `MA57` is used to compute the factors. For those problems for which the exact preconditioner is expensive—for example, the `CVXQP*` and `NCVXQP*` problems—the "inexact" preconditioners are often more effective, particularly when low accuracy solutions are required. The explicit preconditioner with $G = I$ is often a good compromise, although this may reflect the fact that H is often (almost) diagonal.

Table 4.1: CUTEr QP problems—residual decrease of at least 10^{-2}

	Explicit factors									Implicit factors					
	$G = H$						$G = I$			$G_{22} = I$			$G_{22} = H_{22}$		
	MA27			MA57			MA57			MA57			MA57		
name	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total
AUG2DCQP	0.08	1	0.13	0.47	1	0.54	0.46	1	0.53	0.04	125	1.54	0.25	125	2.01
AUG2DQP	0.08	1	0.13	0.47	1	0.54	0.46	2	0.53	0.04	120	1.49	0.25	125	2.03
AUG3DCQP	1.56	1	1.66	1.54	1	1.67	1.45	1	1.57	0.05	41	0.71	0.79	41	1.59
AUG3DQP	1.59	1	1.69	1.29	1	1.42	1.46	2	1.59	0.05	43	0.71	0.78	40	1.56
BLOCKQP1	0.06	0	0.08	0.21	0	0.23	0.23	1	0.26	0.33	2	0.35	0.39	2	0.41
BLOCKQP2	0.06	0	0.08	0.21	0	0.23	0.23	2	0.26	0.33	2	0.36	0.39	2	0.41
BLOCKQP3	0.06	0	0.08	0.21	0	0.23	0.23	1	0.25	0.33	2	0.35	0.38	2	0.41
BLOWEYA	26.50	1	26.60	0.04	1	0.05	0.05	35	0.21	0.03	50	0.13	0.04	50	0.15
BLOWEYB	26.29	1	26.39	0.04	1	0.05	0.05	13	0.11	0.03	32	0.09	0.04	32	0.11
BLOWEYC	26.27	1	26.36	0.04	1	0.05	0.05	36	0.21	0.03	50	0.12	0.04	50	0.15
CONT-050	0.17	1	0.19	0.12	1	0.14	0.12	1	0.14	0.09	3	0.10	0.09	3	0.11
CONT-101	3.03	1	3.18	0.73	2	0.85	0.70	2	0.82	0.86	2	0.91	0.86	2	0.91
CONT-201	35.96	4	38.38	5.78	5	6.99	5.63	6	7.04	10.14	2	10.41	10.10	2	10.37
CONT5-QP	33.89	1	34.59	3.37	1	3.83	3.35	2	3.80	20.01	39	22.36	19.94	37	22.20
CONT1-10	2.81	1	2.95	0.68	1	0.80	0.66	1	0.77	0.90	3	0.97	0.91	3	0.99
CONT1-20	30.94	1	31.65	6.85	1	7.46	6.67	2	7.28	10.83	3	11.22	10.86	3	11.26
CONT-300	140.10	9	146.23	19.33	5	22.26	18.33	5	21.25	40.82	2	41.46	41.00	2	41.64
CVXQP1	579.20	0	580.15	3.99	0	4.11	0.20	3	0.24	0.21	57	0.56	0.24	55	0.69
CVXQP2	139.11	0	139.48	1.70	0	1.78	0.10	3	0.12	0.01	14	0.07	0.10	14	0.23
CVXQP3	1353.52	0	1355.13	9.93	0	10.13	0.32	3	0.38	0.33	44	0.64	0.34	43	0.68
DEGENQP	3.85	1	4.14	14.36	1	14.72	0.01	2	0.01	2.43	3	2.87	2.45	3	2.89
DUALC1	0.01	5	0.01	0.00	2	0.01	0.00	1	0.00	0.00	8	0.00	0.00	8	0.00
DUALC2	0.01	9	0.01	0.00	1	0.01	0.01	2	0.01	0.00	6	0.00	0.00	6	0.01
DUALC5	0.01	8	0.02	0.01	1	0.01	0.01	2	0.01	0.00	6	0.01	0.00	6	0.01
DUALC8	0.11	5	0.13	0.01	2	0.01	0.20	0	0.23	0.01	7	0.01	0.01	7	0.01
GOULDQP2	0.05	0	0.07	0.23	0	0.27	0.20	2	0.25	0.03	0	0.05	0.08	0	0.10
GOULDQP3	0.07	1	0.11	0.32	1	0.40	0.05	5	0.06	0.03	6	0.11	0.08	6	0.17
KSIP	0.01	1	0.02	0.05	1	0.06	0.04	3	0.05	0.02	3	0.03	0.02	3	0.03
MOSARQP1	0.02	1	0.03	0.04	1	0.04	0.20	3	0.24	0.06	6	0.07	0.07	6	0.08
NCVXQP1	573.69	0	574.65	4.10	0	4.22	0.20	3	0.24	0.21	55	0.54	0.24	55	0.68
NCVXQP2	584.17	0	585.14	4.02	0	4.14	0.20	3	0.24	0.20	55	0.54	0.24	56	0.70
NCVXQP3	573.04	0	573.98	4.15	0	4.28	0.11	3	0.13	0.20	54	0.53	0.23	55	0.69
NCVXQP4	138.52	0	138.90	1.71	0	1.79	0.10	3	0.12	0.01	14	0.07	0.10	13	0.22
NCVXQP5	130.26	0	130.64	1.69	0	1.76	0.10	3	0.13	0.01	14	0.06	0.10	14	0.24
NCVXQP6	139.37	0	139.75	1.70	0	1.79	0.32	3	0.38	0.01	14	0.06	0.10	14	0.24
NCVXQP7	1363.85	0	1365.49	10.03	0	10.23	0.33	3	0.39	0.33	43	0.64	0.34	43	0.67
NCVXQP8	1386.80	0	1388.45	10.07	0	10.26	0.33	3	0.38	0.33	43	0.63	0.34	43	0.67
NCVXQP9	1357.68	0	1359.31	10.12	0	10.32	0.09	2	0.11	0.33	44	0.64	0.34	43	0.67
POWELL20	0.03	0	0.05	0.09	0	0.11	0.00	5	0.01	0.01	2	0.03	0.07	2	0.08
PRIMALC1	0.00	1	0.00	0.00	1	0.01	0.00	3	0.00	0.00	11	0.00	0.00	6	0.00
PRIMALC2	0.00	1	0.00	0.00	1	0.01	0.00	6	0.01	0.00	5	0.00	0.00	5	0.00
PRIMALC5	0.00	1	0.00	0.00	1	0.01	0.01	4	0.01	0.00	6	0.00	0.00	5	0.00
PRIMALC8	0.01	1	0.01	0.01	1	0.01	0.01	8	0.02	0.00	11	0.01	0.00	7	0.01
PRIMAL1	0.01	1	0.01	0.01	1	0.02	0.03	5	0.03	0.00	15	0.01	0.00	27	0.02
PRIMAL2	0.01	1	0.01	0.03	1	0.03	0.06	4	0.07	0.00	13	0.01	0.01	21	0.02
PRIMAL3	0.03	1	0.03	0.06	1	0.06	0.03	3	0.04	0.01	18	0.04	0.01	26	0.06
PRIMAL4	0.04	1	0.04	0.03	1	0.03	14.34	2	14.69	0.01	12	0.03	0.02	15	0.04
QPBAND	0.16	1	0.30	1.08	1	1.28	1.84	2	1.99	0.09	2	0.19	0.40	2	0.54
QPNBAND	0.17	1	0.30	1.07	1	1.27	1.83	3	2.03	0.09	3	0.24	0.41	2	0.55
QPCBOEI1	0.01	1	0.01	0.02	2	0.02	0.01	3	0.01	0.00	12	0.01	0.00	12	0.01
QPCBOEI2	0.00	1	0.01	0.00	1	0.01	0.00	3	0.01	0.00	12	0.00	0.00	12	0.00
QPCSTAIR	0.01	1	0.01	0.02	1	0.02	0.01	3	0.02	0.00	12	0.01	0.00	14	0.01
QPNBOEI1	0.01	1	0.01	0.02	2	0.02	0.01	3	0.01	0.01	12	0.01	0.00	12	0.01
QPNBOEI2	0.00	1	0.00	0.00	1	0.01	0.00	3	0.01	0.00	12	0.00	0.00	12	0.00
QPNSTAIR	0.01	1	0.01	0.02	1	0.02	0.01	3	0.02	0.00	12	0.01	0.00	12	0.01
SOSQP1	0.01	0	0.01	0.04	0	0.04	0.04	0	0.05	0.03	1	0.04	0.05	1	0.05
STCQP1	rank deficient A			rank deficient A			20.67	3	21.01	0.02	3	0.04	0.09	1	0.10
STCQP2	9.76	0	9.84	0.87	0	0.92	0.14	3	0.17	0.03	3	0.05	0.11	1	0.13
STNQP1	113.27	0	113.59	rank deficient A			20.75	3	21.09	0.02	3	0.04	0.09	1	0.11
STNQP2	9.64	0	9.72	0.87	0	0.92	0.14	3	0.17	0.03	3	0.05	0.11	1	0.13
UBH1	0.02	0	0.03	0.12	0	0.14	0.11	0	0.13	0.02	0	0.03	0.04	0	0.05
YAO	0.01	1	0.01	0.03	1	0.04	0.03	6	0.05	0.01	21	0.04	0.02	21	0.06

Table 4.2: CUTEr QP problems—residual decrease of at least 10^{-8}

	Explicit factors									Implicit factors					
	$G = H$						$G = I$			$G_{22} = I$			$G_{22} = H_{22}$		
	MA27			MA57			MA57			MA57			MA57		
name	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total	fact.	iter.	total
AUG2DCQP	0.08	1	0.13	0.47	1	0.54	0.46	1	0.53	0.04	866	10.35	0.25	872	12.50
AUG2DQP	0.08	1	0.13	0.47	1	0.54	0.46	4	0.60	0.04	882	10.67	0.25	855	12.33
AUG3DCQP	1.56	1	1.66	1.54	1	1.67	1.45	1	1.57	0.05	378	6.04	0.79	377	8.18
AUG3DQP	1.59	1	1.69	1.29	1	1.42	1.46	5	1.75	0.05	381	5.90	0.78	380	8.27
BLOCKQP1	0.06	0	0.08	0.21	0	0.23	0.23	1	0.26	0.33	3	0.37	0.39	3	0.43
BLOCKQP2	0.06	0	0.08	0.21	0	0.23	0.23	2	0.26	0.33	3	0.37	0.39	3	0.43
BLOCKQP3	0.06	0	0.08	0.21	0	0.23	0.23	1	0.25	0.33	5	0.39	0.38	4	0.44
BLOWEYA	26.50	1	26.60	0.04	1	0.05	$>$ 10000 iterations			$>$ 10000 iterations			$>$ 10000 iterations		
BLOWEYB	26.29	1	26.39	0.04	1	0.05	0.05	216	0.99	0.03	668	1.23	$>$ 10000 iterations		
BLOWEYC	26.27	1	26.36	0.04	1	0.05	$>$ 10000 iterations			$>$ 10000 iterations			$>$ 10000 iterations		
CONT-050	0.17	1	0.19	0.12	1	0.14	0.12	1	0.14	0.09	7	0.12	0.09	7	0.13
CONT-101	3.03	1	3.18	0.73	4	1.11	0.70	5	1.15	0.86	10	1.09	0.86	10	1.10
CONT-201	35.96	4	38.38	5.78	8	9.39	5.63	13	11.26	10.14	11	11.48	10.10	11	11.43
CONT5-QP	33.89	1	34.59	3.37	1	3.83	3.35	2	3.95	20.01	113	26.81	19.94	98	25.89
CONT1-10	2.81	1	2.95	0.68	1	0.80	0.66	1	0.77	0.90	10	1.13	0.91	10	1.16
CONT1-20	30.94	1	31.65	6.85	1	7.46	6.67	5	9.08	10.83	12	12.29	10.86	12	12.34
CONT-300	140.10	27	174.66	19.33	26	45.80	18.33	40	58.01	40.82	15	44.98	41.00	15	45.16
CVXQP1	579.20	0	580.15	3.99	0	4.11	0.20	5	0.27	0.21	211	1.49	0.24	207	1.94
CVXQP2	139.11	0	139.48	1.70	0	1.78	0.10	5	0.14	0.01	51	0.21	0.10	51	0.59
CVXQP3	1353.52	0	1355.13	9.93	0	10.13	0.32	5	0.42	0.33	183	1.62	0.34	178	1.71
DEGENQP	3.85	1	4.14	14.36	1	14.72	0.01	11	0.01	2.43	3	3.00	2.45	7	3.52
DUALC1	0.01	5	0.01	0.00	11	0.01	0.00	1	0.00	0.00	8	0.01	0.00	8	0.00
DUALC2	0.01	9	0.01	0.00	1	0.01	0.01	4	0.01	0.00	6	0.00	0.00	6	0.01
DUALC5	0.01	145	0.20	0.01	1	0.01	0.01	5	0.01	0.00	7	0.01	0.00	7	0.01
DUALC8	0.11	5	0.13	0.01	7	0.02	0.20	0	0.23	0.01	7	0.01	0.01	7	0.01
GOULDQP2	0.05	0	0.07	0.23	0	0.27	0.20	5	0.31	0.03	0	0.05	0.08	0	0.10
GOULDQP3	0.07	1	0.11	0.32	1	0.40	0.05	21	0.08	0.03	1614	18.95	0.08	1579	23.38
KSIP	0.01	1	0.02	0.05	1	0.06	0.04	5	0.05	0.02	18	0.05	0.02	10	0.04
MOSARQP1	0.02	1	0.03	0.04	1	0.04	0.20	5	0.27	0.06	36	0.10	0.07	35	0.13
NCVXQP1	573.69	0	574.65	4.10	0	4.22	0.20	5	0.27	0.21	215	1.51	0.24	204	1.89
NCVXQP2	584.17	0	585.14	4.02	0	4.14	0.20	6	0.28	0.20	212	1.50	0.24	212	2.00
NCVXQP3	573.04	0	573.98	4.15	0	4.28	0.11	5	0.14	0.20	210	1.46	0.23	204	1.92
NCVXQP4	138.52	0	138.90	1.71	0	1.79	0.10	5	0.14	0.01	51	0.20	0.10	51	0.60
NCVXQP5	130.26	0	130.64	1.69	0	1.76	0.10	6	0.15	0.01	51	0.20	0.10	50	0.59
NCVXQP6	139.37	0	139.75	1.70	0	1.79	0.32	5	0.42	0.01	51	0.21	0.10	51	0.61
NCVXQP7	1363.85	0	1365.49	10.03	0	10.23	0.33	5	0.43	0.33	189	1.69	0.34	176	1.67
NCVXQP8	1386.80	0	1388.45	10.07	0	10.26	0.33	5	0.42	0.33	191	1.69	0.34	176	1.70
NCVXQP9	1357.68	0	1359.31	10.12	0	10.32	0.09	20	0.23	0.33	193	1.69	0.34	179	1.71
POWELL20	0.03	0	0.05	0.09	0	0.11	0.00	11	0.01	0.01	40	0.21	0.07	40	0.31
PRIMALC1	0.00	1	0.00	0.00	1	0.01	0.00	4	0.00	0.00	25	0.01	0.00	12	0.01
PRIMALC2	0.00	1	0.00	0.00	1	0.01	0.00	10	0.01	0.00	9	0.00	0.00	9	0.00
PRIMALC5	0.00	1	0.00	0.00	1	0.01	0.01	7	0.01	0.00	15	0.01	0.00	10	0.01
PRIMALC8	0.01	1	0.01	0.01	1	0.01	0.01	14	0.02	0.00	20	0.01	0.00	10	0.01
PRIMAL1	0.01	1	0.01	0.01	1	0.02	0.03	8	0.03	0.00	153	0.08	0.00	158	0.09
PRIMAL2	0.01	1	0.01	0.03	1	0.03	0.06	6	0.07	0.00	86	0.06	0.01	92	0.08
PRIMAL3	0.03	1	0.03	0.06	1	0.06	0.03	5	0.04	0.01	74	0.14	0.01	80	0.15
PRIMAL4	0.04	1	0.04	0.03	1	0.03	14.34	2	14.80	0.01	41	0.07	0.02	44	0.09
QPBAND	0.16	1	0.30	1.08	1	1.28	1.84	5	2.19	0.09	7	0.46	0.40	5	0.78
QPNBAND	0.17	1	0.30	1.07	1	1.27	1.83	6	2.24	0.09	8	0.51	0.41	6	0.84
QPCBOEI1	0.01	1	0.01	0.02	5	0.02	0.01	5	0.02	0.00	47	0.03	0.00	47	0.03
QPCBOEI2	0.00	1	0.01	0.00	1	0.01	0.00	5	0.01	0.00	38	0.01	0.00	37	0.01
QPCSTAIR	0.01	1	0.01	0.02	1	0.02	0.01	8	0.02	0.00	40	0.02	0.00	52	0.03
QPNBOEI1	0.01	1	0.01	0.02	5	0.03	0.01	5	0.01	0.01	48	0.03	0.00	47	0.03
QPNBOEI2	0.00	1	0.00	0.00	1	0.01	0.00	5	0.01	0.00	37	0.01	0.00	37	0.01
QPNSTAIR	0.01	1	0.01	0.02	1	0.02	0.01	8	0.02	0.00	40	0.02	0.00	56	0.03
SOSQP1	0.01	0	0.01	0.04	0	0.04	0.04	0	0.05	0.03	1	0.04	0.05	1	0.05
STCQP1	rank deficient A			rank deficient A			20.67	6	21.35	0.02	6	0.05	0.09	1	0.10
STCQP2	9.76	0	9.84	0.87	0	0.92	0.14	7	0.20	0.03	7	0.07	0.11	1	0.13
STNQP1	113.27	0	113.59	rank deficient A			20.75	6	21.43	0.02	6	0.05	0.09	1	0.11
STNQP2	9.64	0	9.72	0.87	0	0.92	0.14	8	0.22	0.03	8	0.08	0.11	1	0.13
UBH1	0.02	0	0.03	0.12	0	0.14	0.11	0	0.13	0.02	0	0.03	0.04	0	0.05
YAO	0.01	1	0.01	0.03	1	0.04	0.03	26	0.11	0.01	107	0.18	0.02	106	0.23

The implicit factors are sometimes but not always cheaper to compute than the explicit ones. The cost of finding a good basis A_1 using `MA48` is higher than we would have liked, and is usually the dominant cost of the overall implicit factorization. Nonetheless, for problems like the `DUAL*`, `PRIMAL*` and `ST*` examples, the implicit factors seem to offer a good alternative to the explicit ones. We must admit to being slightly disappointed that the more sophisticated implicit factors using $G_{22} = H_{22}$ seemed to show few advantages over the cheaper $G_{22} = I$, but again this might reflect the nature of H in our test set.

5 Comments and conclusions

We have developed a class of implicit-factorization constraint preconditioners for the iterative solution of symmetric linear systems arising from saddle-point problems. These preconditioners are flexible, and allow for improved eigenvalue distributions over traditional approaches. Numerical experiments indicate that these methods hold promise for solving large-scale problems, and suggest that such methods should be added to the arsenal of available preconditioners for saddle-point and related problems. A fortran 90 package which implements methods from our class of preconditioners will shortly be available as part of the GALAHAD library [30]. We are currently generalizing implicit-factorization preconditioners to cope with problems for which the 2,2 block in (1.1) may be nonzero [13].

One issue we have not really touched on—aside from the need for stable factors—is the effect of partitioning of the columns of A to produce a non-singular sub-matrix A_1. Consider the simple example

$$A = \begin{pmatrix} \times & 0 & \times & 0 \\ 0 & \times & \times & \times \end{pmatrix},$$

where each $\times$ is non-zero. If we chose A_1 as the sub-matrix corresponding to the first two columns of A, A_2 has rank two, while if A_1 were made up of columns one and three, A_2 then has rank one. This simple example indicates how the choice of A_1 may effect the iteration bounds obtained in Theorems 2.3–2.5, and significantly, leads us to ask just how much we can reduce the bounds indicated in these theorems by judicious choice of A_1. We plan to investigate this issue in future.

Acknowledgment

Thanks are due to Mario Arioli, Iain Duff and John Reid for fruitful discussions on various aspects of this work.

References

[1] M. Avriel. *Nonlinear Programming: Analysis and Methods*. Prentice-Hall, Englewood Cliffs, New Jersey, 1976.

[2] R. H. Bartels and G. H. Golub. The simplex method of linear programming using lu decompositions. *Communications of the ACM*, 12:266–268, 1969.

[3] L. Bergamaschi, J. Gondzio, and G. Zilli. Preconditioning indefinite systems in interior point methods for optimization. *Computational Optimization and Applications*, 28:149–171, 2004.

[4] G. Biros and O. Ghattas. A Lagrange-Newton-Krylov-Schur method for pde-constrained optimization. *SIAG/Optimiztion Views-and-News*, 11(2):12–18, 2000.

[5] J. H. Bramble and J. E. Pasciak. A preconditioning technique for indefinite systems resulting from mixed approximations of elliptic problems. *Mathematics of Computation*, 50:1–17, 1988.

[6] R. H. Byrd, M. E. Hribar, and J. Nocedal. An interior point algorithm for large scale nonlinear programming. *SIAM Journal on Optimization*, 9(4):877–900, 2000.

[7] Y. Chabrillac and J.-P. Crouzeix. Definiteness and semidefiniteness of quadratic forms revisited. *Linear Algebra and its Applications*, 63:283–292, 1984.

[8] T. F. Coleman and A. Pothen. The null space problem I: complexity. *SIAM Journal on Algebraic and Discrete Methods*, 7(4):527–537, 1986.

[9] T. F. Coleman and A. Pothen. The null space problem II: algorithms. *SIAM Journal on Algebraic and Discrete Methods*, 8(4):544–563, 1987.

[10] A. R. Conn, N. I. M. Gould, D. Orban, and Ph. L. Toint. A primal-dual trust-region algorithm for non-convex nonlinear programming. *Mathematical Programming*, 87(2):215–249, 2000.

[11] R. W. Cottle. Manifestations of the Schur complement. *Linear Algebra and its Applications*, 8:189–211, 1974.

[12] G. Debreu. Definite and semidefinite quadratic forms. *Econometrica*, 20(2):295–300, 1952.

[13] H. S. Dollar, N. I. M. Gould, W. H. A. Schilders, and A. J. Wathen. On implicit-factorization preconditioners for symmetric, block systems of linear equations. Technical Report RAL-TR-2004-037, Rutherford Appleton Laboratory, Chilton, Oxfordshire, England, 2004.

[14] H. S. Dollar and A. J. Wathen. Incomplete factorization constraint preconditioners for saddle point problems. Technical Report 04/01, Oxford University Computing Laboratory, Oxford, England, 2004.

[15] I. S. Duff. MA57 - a code for the solution of sparse symmetric definite and indefinite systems. *ACM Transactions on Mathematical Software*, 30(2):118–144, 2004.

[16] I. S. Duff and J. K. Reid. The multifrontal solution of indefinite sparse symmetric linear equations. *ACM Transactions on Mathematical Software*, 9(3):302–325, 1983.

[17] I. S. Duff and J. K. Reid. The design of MA48: a code for the direct solution of sparse unsymmetric linear systems of equations. *ACM Transactions on Mathematical Software*, 22(2):187–226, 1996.

[18] C. Durazzi and V. Ruggiero. Indefinitely constrained conjugate gradient method for large sparse equality and inequality constrained quadratic problems. *Numerical Linear Algebra with Applications*, 10(8):673–688, 2002.

[19] H. C. Elman, D. J. Silvester, and A. J. Wathen. *Finite-Elements and Fast Iterative Solvers: with applications in Incompressible Fluid Dynamics*. Oxford University Press, Oxford, 2005, to appear.

[20] J. J. H. Forrest and J. A. Tomlin. Updating triangular factors of the basis to maintain sparsity in the product form simplex method. *Mathematical Programming*, 2(3):263–278, 1972.

[21] D. M. Gay. Electronic mail distribution of linear programming test problems. Mathematical Programming Society COAL Newsletter, December 1985. See `http://www.netlib.org/lp/data/`.

[22] P. E. Gill, W. Murray, D. B. Ponceleón, and M. A. Saunders. Preconditioners for indefinite systems arising in optimization. *SIAM Journal on Matrix Analysis and Applications*, 13(1):292–311, 1992.

[23] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12(4):979–1006, 2002.

[24] P. E. Gill and M. A. Saunders. private communication, 1999.

[25] G. H. Golub and C. Greif. On solving block-structured indefinite linear systems. *SIAM Journal on Scientific Computing*, 24(6):2076–2092, 2003.

[26] N. I. M. Gould. On practical conditions for the existence and uniqueness of solutions to the general equality quadratic-programming problem. *Mathematical Programming*, 32(1):90–99, 1985.

[27] N. I. M. Gould, M. E. Hribar, and J. Nocedal. On the solution of equality constrained quadratic problems arising in optimization. *SIAM Journal on Scientific Computing*, 23(4):1375–1394, 2001.

[28] N. I. M. Gould, S. Lucidi, M. Roma, and Ph. L. Toint. Solving the trust-region subproblem using the Lanczos method. *SIAM Journal on Optimization*, 9(2):504–525, 1999.

[29] N. I. M. Gould, D. Orban, and Ph. L. Toint. CUTEr (and SifDec), a Constrained and Unconstrained Testing Environment, revisited. *ACM Transactions on Mathematical Software*, 29(4):373–394, 2003.

[30] N. I. M. Gould, D. Orban, and Ph. L. Toint. GALAHAD—a library of thread-safe fortran 90 packages for large-scale nonlinear optimization. *ACM Transactions on Mathematical Software*, 29(4):353–372, 2003.

[31] C. Greif, G. H. Golub, and J. M. Varah. Augmented Lagrangian techniques for solving saddle point linear systems. Technical report, Computer Science Department, University of British Columbia, Vancouver, Canada, 2004.

[32] HSL. A collection of Fortran codes for large-scale scientific computation, 2004. See `http://hsl.rl.ac.uk/`.

[33] C. Keller, N. I. M. Gould, and A. J. Wathen. Constraint preconditioning for indefinite linear systems. *SIAM Journal on Matrix Analysis and Applications*, 21(4):1300–1317, 2000.

[34] L. Lukšan and J. Vlček. Indefinitely preconditioned inexact Newton method for large sparse equality constrained nonlinear programming problems. *Numerical Linear Algebra with Applications*, 5(3):219–247, 1998.

[35] B. A. Murtagh and M. A. Saunders. Large-scale linearly constrained optimization. *Mathematical Programming*, 14(1):41–72, 1978.

[36] B. A. Murtagh and M. A. Saunders. A projected Lagrangian algorithm and its implementation for sparse non-linear constraints. *Mathematical Programming Studies*, 16:84–117, 1982.

[37] J. Nocedal and S. J. Wright. *Large sparse numerical optimization*. Series in Operations Research. Springer Verlag, Heidelberg, Berlin, New York, 1999.

[38] L. A. Pavarino. Preconditioned mixed spectral finite-element methods for elasticity and Stokes problems. *SIAM Journal on Scientific Computing*, 19(6):1941–1957, 1998.

[39] I. Perugia and V. Simoncini. Block-diagonal and indefinite symmetric preconditioners for mixed finite element formulations. *Numerical Linear Algebra with Applications*, 7(7-8):585–616, 2000.

[40] W. Schilders. A preconditioning technique for indefinite systems arising in electronic circuit simulation. Talk at the one-day meeting on preconditioning methods for indefinite linear systems, TU Eindhoven, December 9, 2002.

[41] R. B. Schnabel and E. Eskow. A revised modified Cholesky factorization algorithm. *SIAM Journal on Optimization*, 9(4):1135–1148, 1999.

[42] R. J. Vanderbei and D. F. Shanno. An interior point algorithm for nonconvex nonlinear programming. *Computational Optimization and Applications*, 13:231–252, 1999.

[43] R. C. Whaley, A. Petitet, and J. Dongarra. Automated empirical optimization of software and the ATLAS project. *Parallel Computing*, 27(1–2):3–35, 2001.