HOPDM, version 2.30: Benchmark Results: Solving Convex Quadratic Programming Problems

Jacek Gondzio

Logilab, HEC Geneva, Department of Management Studies
University of Geneva, 102 Bd Carl-Vogt, 1211 Geneva, Switzerland

gondzio@com7000.unige.ch or gondzio@ibspan.waw.pl
http://ecolu-info.unige.ch/~logilab/gondzio/

March 22, 1998

Abstract

This note presents computational results obtained with HOPDM interior point code applied to solve large scale convex quadratic programming problems.

Keywords: Large Scale QP, Benchmarks.

Table 1: Solving randomly generated quadratic programs (Pentium Pro, 200 MHz).

Name	m	n	nz	nzQ	nzL	it	time
qp100-1	220	100	2542	1660	7555	12	1.20
qp100-2	2020	100	22206	1660	29056	12	5.96
qp100-3	220	100	22100	1660	27150	12	6.06
qp1000-1	700	1000	15753	68762	491059	14	339.09
qp250-1	550	250	14638	16771	46010	14	11.53
qp250-2	550	250	69140	30699	100515	12	30.31
qp250-3	550	250	14638	16771	46006	15	12.29
qp500-1	1100	500	56750	89071	181999	13	70.43
qp500-2	1100	500	51788	89071	177037	17	83.24
qp500-3	100	500	5614	89071	128837	7	33.52
sqp2500-1	2000	2500	52321	738051	3124093	14	5380.93
sqp2500-2	2000	2500	52319	14345	3504910	17	7363.16
sqp2500-3	4500	2500	115073	738051	3216994	16	6332.87

Table 1 reports the results of solving large randomly generated convex quadratic programming problems. These relatively dense problems were generated by AMPL algebraic modeling language. We report the problem characteristics and the number of off-diagonal nonzero entries in the matrix Q and in the triangular factor of the augmented system matrix. Next, we report the solution statistics (iterations and CPU time) to reach 8-digit optimal solution.

The computations were done on a PC with a 200MHz Pentium Pro processor and 128 MB of RAM running under Linux operating system. HOPDM was compiled with the FORTRAN GNU compiler `fort77`; options `-O2 -m486` were used. All solution times were measured with the Linux `time` command. Solution times are given in seconds and include reading the MPS input file and writing MPS output report.

Large scale randomly generated quadratic programs have been provided to us by Hans Mittelmann, see e.g., `http://plato.la.asu.edu/guide.html`.

Table 2: Solving very large quadratic programs (IBM RS6000, Power 2, 58 MHz).

Name	m	n	nz	nz(Q)	nz(H)	nz(L)	iter	time
AUG2D	22500	45300	112500	0	112500	596270	8	66.77
AUG2DC	22500	45300	112500	0	112500	596270	9	75.02
AUG2DCQP	22500	45300	112500	0	112500	596270	10	73.06
AUG2DQP	22500	45300	112500	0	112500	596270	11	77.75
AUG3D	15625	52428	112981	0	112981	2851780	9	1405.07
AUG3DC	27000	89013	181320	0	181320	5641694	17	3628.78
AUG3DCQP	27000	89013	181320	0	181320	5641694	16	3310.65
AUG3DQP	15625	52428	112981	0	112981	2851780	13	1697.32
CVXQP1	7500	15000	22497	44981	67478	6619618	10	4186.08
CVXQP2	3750	15000	11249	44981	56230	5225664	9	3043.68
CVXQP3	11250	15000	33745	44981	78726	7872011	31*	15421.70
POWELL20	75000	75000	225000	0	225000	374997	7	58.05
UBH1	48000	72009	203982	0	203982	419664	15	80.86
YAO	60000	60002	239997	0	239997	499878	8	61.70

Table 2 reports the results of solving very large convex quadratic programs from the CUTE collection of nonlinear problems. We report the problem characteristics and the number of off-diagonal nonzero entries in the matrix Q, in the augmented system matrix, nz(H) and in the triangular factor of this matrix. Next, we report the solution statistics (iterations and CPU time) to reach 8-digit optimal solution. For the problem CVXQP3 5-digit optimal solution was reached.

The computations were done on an IBM RS6000 Power 2 workstation with a 58 MHz processor and 128 MB of RAM running under AIX 4.2 operating system. HOPDM was compiled with the FORTRAN compiler `xlf`; options `-O -qarch=pwr` were used. Solution times are given in seconds (they do not include reading the MPS input file and writing MPS output report).

Large scale quadratic programs coming from CUTE collection have been provided to us by Hans Mittelmann, see e.g., `http://plato.la.asu.edu/guide.html`.

Table 3: Multiple centrality corrections (IBM RS6000, Power 2, 58 MHz).

Corrs	AUG2DC		AUG3DC		CVXQP1	
	iters	time	iters	time	iters	time
0	11	87.2	25	4973.2	13	5221.6
1	10	80.1	23	4589.0	12	4812.6
2	**9**	75.0	23	4654.3	11	4456.4
4	9	75.6	21	4350.0	10	4105.0
6	9	75.7	19	3967.6	10	4119.4
8	9	76.6	18	3816.0	10	4097.0
10	9	78.3	**17**	3628.8	**10**	4186.1

Finally, we show the advantages of the use of multiple centrality correctors on three large problems AUG2DC, AUG3DC and CVXQP1. The Cholesky factors for these problems contain 596270, 5641694, and 6619618 subdiagonal entries, respectively. To see how the multiple corrections technique works, we let the maximum number of correctors go from 0 through 10. The maximum number of centrality correctors allowed (chosen automatically by HOPDM) is set to 2, 10 and 10, respectively. We marked in bold the number of iterations in runs corresponding to the default choice. All problems were solved to 8-digit optimal solution.

The computations were done on an IBM RS6000 Power 2 workstation with a 58 MHz processor and 128 MB of RAM running under AIX 4.2 operating system. HOPDM was compiled with the FORTRAN compiler `xlf`; options `-O -qarch=pwr` were used. Solution times are given in seconds (they do not include reading the MPS input file and writing MPS output report).