The BPMPD interior point solver for convex quadratic problems *

Csaba Mészáros
Laboratory of Operations Research and Decision Systems,
Hungarian Academy of Sciences

WP 98–8

Abstract

The paper describes the convex quadratic solver BPMPD Version 2.21. The solver is based on the infeasible–primal–dual algorithm extended by the predictor–corrector and target–following techniques. The discussion includes topics related to the implemented algorithm and numerical algebra employed. We outline the presolve, scaling and starting point stategies used in BPMPD, and special attention is given for sparsity and stability issues. Computational results are given on a demonstrative set of convex quadratic problems.

1 Introduction

The success of interior point methods in the linear programming (LP) practice resulted in an increased interest for the application of the IPM technology in the nonlinear optimization. It is now widely accepted that on large–scale problems interior point methods can significantly outperform the algorithms based on the generalizations of the simplex approach (see the benchmark results in [20]). The paper describes the IPM solver, BPMPD version 2.21,

*This work was supported in part by EPSRC grant No. GR/J52655 and Hungarian Research Fund OTKA T-016413.

which is an efficient implementation of the infeasible–primal–dual algorithm for solving convex quadratic (QP) problems. Without loss of generality, the convex QP problem is usually assumed to be in the following form:

$$\begin{array}{rl} \min & c^T x + \frac{1}{2} x^T Q x, \\ & Ax = b, \\ & x \geq 0, \end{array} \tag{1}$$

where $A \in R^{m \times n}$ is of full row rank, $Q \in R^{n \times n}$ is symmetric positive semidefinite and $c, x \in R^n$, $b \in R^m$. The Wolfe–dual associated with this problem can be written as

$$\begin{array}{rl} \max & b^T y - \frac{1}{2} x^T Q x, \\ & A^T y + z - Qx = c, \\ & z \geq 0, \end{array} \tag{2}$$

where $z \in R^n$ and $y \in R^m$. The constrains of the above problems define convex polyhedrons which are nonempty if a feasible primal–dual solution exists.

BPMPD version 2.21 is entirely written in C and is believed to be portable, and can be compiled with any Ansi C compiler system. BPMPD supports the industry–standard MPS format with the extension for quadratic problems [10]. In the MPS input file A, b and c, furthermore the bounds on individual variables and rows are specified by the usual ROWS, COLUMNS, RHS, BOUNDS, RANGES records of the MPS format [23]. For specifying the quadratic part of the problem, a new section, "QUADOBJ", is introduced which describes the lower triangular part of the quadratic objective in that relative column order as it has been given in the COLUMNS section. Otherwise the data records in the QUADOBJ section are of the same structure as that of the COLUMNS section. In addition, BPMPD is also available as callable library and is connected to AMPL and MATLAB via interface routines.

The paper is organized as follows: Section 2 presents the infeasible–primal–dual method on which our implementation is based. In section 3 we outline some of the techniques used in BPMPD, including presolve, scaling, starting point and sparsity issues. Section 4 describes the methods related to the numerical stability which make our implementation robust and reliable. Numerical results are shown in section 5 on large–scale quadratic programming test problems.

2 The infeasible–primal–dual method

After Karmarkar introduced the novel interior point method for linear programming, a number of different interior point algorithms appeared in the literature (see [25]). Among those, the infeasible–primal–dual log barrier methods are considered to be the most effective for solving large–scale problems [28]. Whereas the results of Monterio and Zhou [22] show the superlinear convergence behavior of these methods, computational practice indicates their usefulness in real life [21, 4].

Primal–dual interior point algorithms are iterative approaches which seek to satisfy the Karush–Kuhn–Tucker type optimality conditions for the primal–dual problem (1-2):

$$\begin{aligned} Ax &= b && (3)\\ A^T y + z - Qx &= c, && (4)\\ Xz &= 0, && (5)\\ (x, z) &\geq 0, && (6) \end{aligned}$$

where $X = diag(x_1, ..., x_n)$. The infeasible–primal–dual interior point methods generate a sequence of iterates in both the primal and dual spaces

$$(x^k, y^k, z^k) \qquad k = 0, 1, 2, ...$$

which fulfil the strict positivity condition $(x^k, z^k) > 0$, but feasibility (3,4) and optimality (5) is reached as k increases.

The first step to obtain the infeasible–primal–dual algorithm is replacing the nonnegativity constraints with logarithmic penalty terms, which gives the following logarithmic barrier problem:

$$\begin{array}{ll} \text{minimize} & c^T x + \frac{1}{2} x^T Q x - \mu \sum_{j=1}^{n} \ln x_j \\ \text{subject to} & Ax = b. \end{array} \qquad (7)$$

The Lagrangian for (7) is

$$L(x, s, y, w, \mu) = c^T x + \frac{1}{2} x^T Q x - \mu \sum_{j=1}^{n} \ln x_j - y^T (Ax - b) \qquad (8)$$

and the first order optimality conditions are:

$$\begin{array}{rcl} \nabla_x L = & c - \mu X^{-1} e - A^T y + Qx & = 0, \\ \nabla_y L = & b - Ax & = 0. \end{array}$$

By introducing $Z = \mu X^{-1}$ one can write the first order optimality conditions of (8) as

$$\begin{array}{rl} Ax & = b, \\ A^T y + z - Qx & = c, \\ XZe & = \mu e. \end{array} \tag{9}$$

Let us observe that the last nonlinear equation in (9) becomes the complementarity condition for $\mu = 0$, which together with the feasibility constraints provides the optimality.

One iteration of the primal–dual algorithm takes one step of Newton's method applied to the first order optimality conditions (9) with a given μ. The algorithm terminates when the infeasibility and the complementarity gap are reduced below predetermined tolerances or infeasibility is detected.

For a given $x, z \in \mathcal{R}^n_+$ and $y \in \mathcal{R}^m$ Newton's direction is obtained by solving the following system of linear equations

$$\begin{bmatrix} A & 0 & 0 \\ Q & A^T & I \\ Z & 0 & X \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix} = \begin{bmatrix} \xi_b \\ \xi_c \\ \mu e - XZe \end{bmatrix}, \tag{10}$$

where

$$\begin{array}{rrl} & \xi_b & = b - Ax, \\ \text{and} & \xi_c & = c - A^T y - z + Qx \end{array}$$

denote the violations of the primal and the dual constraints, respectively. We call the linear system (10) as the Newton equations system.

Once the system (10) has been solved, the maximum stepsizes in primal space ($\alpha_{\bar{P}}$) and dual space ($\alpha_{\bar{D}}$) are computed such that the nonnegativity of variables is preserved:

$$\begin{array}{rcl} \alpha_{\bar{P}} & = & \dfrac{1}{\max\limits_{k=1\ldots n} \left\{1, -\frac{\Delta x_k}{x_k}\right\}}, \\ \alpha_{\bar{D}} & = & \dfrac{1}{\max\limits_{k=1\ldots n} \left\{1, -\frac{\Delta z_k}{z_k}\right\}}. \end{array}$$

For quadratic problems, using the common allowable steplength, defined as $\alpha = \min(\alpha_P, \alpha_D)$, is a natural idea since this technique guarantees that both the primal and dual infeasibilities are decreased by factor $(1-\alpha)$. On the other hand, as the linear programming practice showed, reducing the larger stepsize to the value of the smaller one may not be the most efficient choice. A new approach was introduced in [15] which considers the quadratic multiobjective problem as

$$\begin{array}{lc}
\min & \|b - A(x + \alpha_P \Delta x)\|^2, \\
\min & \|c - A^T(y + \alpha_D \Delta y) - (z + \alpha_D \Delta z) + Q(x + \alpha_P \Delta x)\|^2, \\
 & \bar{\alpha}_P \geq \alpha_P \geq 0, \\
 & \bar{\alpha}_D \geq \alpha_D \geq 0,
\end{array} \tag{11}$$

and chooses the steplength (α_P, α_D) from the efficient set of problem (11) such that the infeasibility is reduced faster than by using the common allowable steplength. The new iterate is then computed as

$$\begin{array}{rcl}
x & \leftarrow & x + \alpha_P \alpha \Delta x, \\
y & \leftarrow & y + \alpha_D \alpha \Delta y, \\
z & \leftarrow & z + \alpha_D \alpha \Delta z,
\end{array} \tag{12}$$

where $0 < \alpha_0 < 1$ is a given factor which prevents to hit the boundary. After making the Newton step, the barrier parameter μ is decreased and the process is repeated.

BPMPD is equipped with the predictor–corrector and target following techniques, the reader should refer to [12, 5] for the detailed development of these extensions.

3 Implementation

The efficient and robust implementation of a numerical algorithm is a highly complex task. The recent version of BPMPD is a result of four years development and consists of approximately 15 thousand C source lines. The implementation contains many parts of different theoretic and implementation complexity which is demonstrated by the following table, showing the different parts of the implementation and their dividend relative to the whole 15 thousand source lines:

Implementation part	dividend
Memory management, others	5 %
Input/output	10 %
Presolve/postsolve	27 %
Symbolical investigations	18 %
Numerical procedures	25 %
Primal–dual algotithm with extensions	15 %

The numeric computations are the most involved steps of any implementation, making them efficient is of extraordinary importance. Even the symbolical investigations are closely related to the numerical procedures, because they are used for reducing the number of necessary arithmetical operations as possible. Therefore during the development of BPMPD a significant effort was concentrated on making the "numerical algebra kernel" robust and fast as possible.

The main feature of BPMPD is that it includes both of the normal equation and augmented system approaches, furthermore a symbolic analyzer which is able to determine which of them is the more beneficial for a given problem. This unique feature makes BPMPD very flexible and able to handle special problems efficiently [16].

3.1 Presolve and scaling

Investigations of interior point implementations indicated that presolve techniques which reduce the original problem size are particularly important in interior point implementations. BPMPD uses all basic techniques [3, 6], including singleton row and column reductions, primal and dual feasibility tests based on upper and lower limits on primal and dual constraints. Beyond the reduction methods which do not transform the original problem algebraically, BPMPD also eliminates free variables . To increase the room for these elimination steps, BPMPD generates free variables in doubleton rows [8] and uses bound relaxation techniques [14]. During the elimination both numerical and structural properties are taken into account which results in a reduction in the problem size without losing advantageous sparsity and numerical properties [14].

A very important presolve step for interior point methods is removing duplicated columns. This step often improves significantly the numerical behavior of IPMs, especially if free variables are presented in spited form in the optimization problem. Since this presolve investigation can also detect hidden free variables, it is advantageous to apply it before the elimination steps.

Before the elimination steps BPMPD uses scaling to improve the numerical behavior of the optimization problem. BPMPD scales each row and column by the geometric mean of the nonzero values in the row and column. This scaling is performed in successive passes and the measure

$$\sum_{\substack{i=1,..,m,\\ j=1,...n,\\ a_{ij}\neq 0}} \log |a_{ij}| \tag{13}$$

is monitored in between. The scaling is stopped if the maximum number of passes is reached or if expression (13) does not decrease. Table 1 presents the effect of the default preprocessing of BPMPD on quadratic programming problems. Figures given include the number of rows, columns and nonzeros in the constraint matrix furthermore the number of nonzeros in the lower off–diagonal part of Q before and after the preprocessing of BPMPD. The results indicate that in several cases the original problem size of quadratic problems can be reduced significantly. For example, problems stcqp1 and stcqp2 were reduced to simple bound constrained optimization problems while the presolve reductions completely removed the nonlinearity of problem qbore3d.

3.2 Starting point

BPMPD determines the starting point by computing

$$\begin{aligned}
\hat{x} &= \arg\min_{x\in R^n} \|Ax-b\|^2 + \frac{\gamma}{2}x^TQx,\\
\hat{y} &= \arg\min_{y\in R^m} \left\|A^Ty-c\right\|^2 + \frac{\gamma}{2}(A^Ty-c)^TQ(A^Ty-c),\\
\hat{z} &= c - A^Ty.
\end{aligned}$$

Since there are possibly negative or small components in the computed $\hat{x}$ and $\hat{z}$, BPMPD uses the method described by Mehrotra [12] to push all

Table 1: Preprocessing results on quadratic programming problems

Problem	Original				Preprocessed			
Name	M	N	Nz (A)	Nz (Q)	M	N	Nz (A)	Nz (Q)
dualc2	229	7	1603	21	9	7	51	21
dualc5	278	8	2224	28	1	8	8	28
dualc8	503	8	4024	28	15	8	105	28
q25fv47	820	1571	10400	59053	687	1456	10489	57674
qbandm	305	472	2494	16	162	317	1421	5
qbore3d	233	315	1429	50	28	53	241	0
qbrandy	220	249	2148	49	93	170	1582	27
qcapri	271	353	1767	838	148	214	1448	824
qe226	223	282	2578	897	149	250	2161	737
qffffff80	524	854	6227	1638	276	617	4649	1628
qgfrdxpn	616	1092	2377	108	432	908	2005	108
qpilotno	975	2172	13057	391	731	1754	11833	163
qscagr25	471	500	1554	100	229	380	1296	89
qscfxm1	330	457	2589	677	238	390	2195	674
qscfxm2	660	914	5183	1057	473	777	4433	1049
qscfxm3	990	1371	7777	1132	712	1168	6559	1109
qscrs8	490	1169	3182	88	221	905	2542	60
qscsd8	397	2750	8584	2370	397	2750	8584	2370
qsctap1	300	480	1692	117	284	480	1638	117
qsctap2	1090	1880	6714	636	1033	1880	6489	636
qsctap3	1480	2480	8874	861	1408	2480	8595	861
qseba	515	1028	4352	550	55	110	343	433
qshell	536	1775	3556	34385	293	1280	2517	2010
qship12l	1151	5427	16170	60205	638	4175	10513	43156
qship12s	1151	2763	8178	16361	323	1902	4777	8715
qstair	356	467	3856	952	254	282	5087	703
stcqp1	2052	4097	13338	22506	0	3158	0	1525
stcqp2	2052	4097	13338	22506	0	2045	0	5898

components of $\hat{x}$ and $\hat{z}$ away from zero. Finally, the starting procedure of BPMPD increases the components of $\hat{x}$ and $\hat{z}$ if necessary until the condition

$$\hat{x}_i \hat{z}_i < \beta \frac{\sum_{i=1}^{n} \hat{x}_j \hat{z}_j}{n} \quad \text{for} \quad i = 1, ..., n$$

is satisfied. The parameter values used by default are $\gamma = 10^{-2}$ and $\beta = 10^{-3}$. Let us observe that the computation of the described starting point needs the factorization of the matrix

$$\begin{bmatrix} -Q - \gamma I & A^T \\ A & 0 \end{bmatrix}$$

and two backsolve operations, thus its cost is nearly the same as of one interior point iteration.

As alternative to the "cold–start" BPMPD is equipped with a warm–start procedure which can take advantage to an available "close optimal" solution. This feature of BPMPD was successfully applied in linear and nonlinear optimization practice [7, 2].

3.3 Stopping criteria

The BPMPD interior point solver terminates with optimal status if the first order optimality conditions (9) are satisfied with a predetermined tolerance:

$$\frac{||Ax - b||}{1 + ||b||} \leq 10^{-p}, \tag{14}$$

$$\frac{||A^T y + z - w - c||}{1 + ||c + Qx||} \leq 10^{-p}, \tag{15}$$

$$\frac{\left| c^T x + \frac{1}{2} x^T Q x - (b^T y - \frac{1}{2} x^T Q x) \right|}{1 + \left| b^T y + \frac{1}{2} x^T Q x \right|} \leq 10^{-p}, \tag{16}$$

where p is the number of accurate digits in the solution. By default, BPMPD stops at an 8–digits exact solution ($p = 8$) which is typically required in the literature. Since the infeasible–primal–dual interior point algorithm diverges if the problem is infeasible, a more interesting question is to determine the infeasible status of an optimization problem. BPMPD terminates with infeasible status if

$$\frac{(b^Ty - \frac{1}{2}x^TQx) - \left(c^Tx + \frac{1}{2}x^TQx\right)}{1 + \left|b^Ty + \frac{1}{2}x^TQx\right|} > 10^t \max\{1, d\},$$

where d is the minimal relative duality gap during the preceding iterations and $t = 4$. Regarding to our experiments the above criterion is very reliable in practice. BPMPD detects the infeasibility in all infeasible NETLIB linear programming problems except of cplex2. Let us note, however, that the relative infeasibility of cplex2 is smaller than 10^{-8} thus a solution satisfying (14–16) exists, therefore the problem can not be declared as a "true" infeasible problem.

3.4 Matrix computations

The reduction of system (10) to the augmented system, whose matrix is

$$M = \begin{bmatrix} -Q - D & A^T \\ A & 0 \end{bmatrix}, \tag{17}$$

where $D \in R^{n \times n}$ is diagonal and

$$D_{ii} = \frac{z_i}{x_i} \text{ for } i = 1, ..., n,$$

is a common step of all interior point implementations and discussions [9, 1]. For solving the augmented system, BPMPD uses the direct symmetric decomposition

$$PMP^T = L\hat{D}L^T, \tag{18}$$

where $\hat{D} \in R^{(n+m)\times(n+m)}$ is diagonal, $L \in R^{(n+m)\times(n+m)}$ is lower triangular and $P \in R^{(n+m)\times(n+m)}$ is a permutation. The decomposition can be obtained in two ways which differ in the pivot selection rule. The *"normal equations"* approach uses two Cholesky factorizations:

$$\hat{F}\hat{F}^T = P_1 (Q + D) P_1^T, \tag{19}$$

where $\hat{F} \in R^{n \times n}$ is lower triangular, $P_1 \in R^{n \times n}$ is a permutation and

$$\hat{L}\hat{L}^T = P_2 \left(A(\hat{F}\hat{F}^T)^{-1} A^T \right) P_2^T, \tag{20}$$

where $\hat{L} \in R^{m \times m}$ is lower triangular and $P_2 \in R^{m \times m}$ is a permutation, which results in the decomposition in the following form:

$$L\hat{D}L^T = \begin{bmatrix} \hat{F} & \\ -A\hat{F}^{-T} & \hat{L} \end{bmatrix} \begin{bmatrix} -I & \\ & I \end{bmatrix} \begin{bmatrix} \hat{F}^T & -\hat{F}^{-1}A^T \\ & \hat{L}^T \end{bmatrix}, \tag{21}$$

while P is

$$P = \begin{bmatrix} P_1 & \\ & P_2 \end{bmatrix}.$$

The "*augmented system*" approach uses a more general pivot selection without describing the elimination of the $-Q - D$ block as first step. In other words, it is not required by the augmented system approach that P is decomposable to an $n \times n$ and $m \times m$ block. This gives more freedom for the pivot selection which can be advantageous in several cases [11, 18].

3.5 Sparsity issues

On large and sparse problems the computational efficiency of the numerical computations are related to the efficient handling of sparsity. This problem is equivalent to the determination of permutation P in (18) in such a way that the fill–in in L is kept as low as possible. This problem involves two steps. The first one is to decide that the normal equation or the augmented system approach is the more beneficial for the given problem. The second step is to determine the proper permutations corresponding to the decision made in the first step. By this latter BPMPD uses the extended tiebreaking variant of the minimum degree ordering [19] which gives nearly the same ordering quality than the minimum local fill–in heuristic but at significantly less computational effort. It is to be noted that by the normal equations approach P_1 and P_2 can be determined in independent steps because the investigation in [18] showed that the sparsity pattern of $A(\hat{F}\hat{F}^T)^{-1}A^T$ is unrelated to the permutation used to obtain $\hat{F}$.

BPMPD uses the adaptive heuristic described in [11] to determine which of the augmented system or the normal equation approach has to be used.

Whereas the heuristic was originally developed for linear programming, in [18] it was shown that one can easily attribute the non–separable quadratic programming case to that of linear programming by building a symbolic matrix from the sparsity patterns of A and Q.

Once the P permutation has been determined, the factorization pattern is fixed for the interior point iterations. BPMPD uses the left–looking numerical factorization and exploits the supernode structure of the factors by applying dense computations and loop unrollings [13]. Dense computations and loop unrollings are also applied during the backsolve operations.

4 Stability issues

The logarithmic barrier interior point methods are based on minimizing a series of barrier subproblems (7). It is easy to see that the eigenvalues of the Hessian of the barrier functions converge either to infinity or to zero, which makes the solution of these subproblems numerically difficult. The solution of these barrier problems is approached by the Newton method, therefore, the structural ill–conditioning of the subproblems translates to the computation of the Newton steps and manifests as ill–conditioning in the Newton system (10). In such a way, ill–conditioning is always present as the interior point method approaches the optimal solution. Theoretical results show [26] that the ill-conditioning of (10) does not influence considerably the accuracy if the optimization problem has nonempty primal and dual interior and the primal problem is non–degenerate. Although the numerical experience confirms the above statement, numerical troubles can arise under several circumstances. In [17] the following three cases were identified which result in numerical problems for the standard factorization approaches in the linear programming case:

- Primal degeneracy.
- The positive components of the primal optimal solution are of different orders of magnitude.
- The columns of A corresponding to the components of x having positive value at the optimum is ill–conditioned.

As it was pointed out, the standard Cholesky factorization in the normal equation approach can break down in case if a row in AF^{-T} lies close to the linear space spanned by the preceding rows in the pivot order. It is easy to see that in the quadratic programming case the same situation can also happen during the Cholesky decomposition (19) if the problem is non–separable and Q is semidefinite.

BPMPD uses relative pivot tolerance [17] during the factorization (18). The transformations are organized such that during the computation of the elements of $\hat{D}$ the negative and positive terms are accumulated separately and added together at the end:

$$\begin{aligned}
\hat{D}_{ii}^{-} &= -\sum_{\substack{j=1,\dots,i-1\\ \hat{D}_{jj}>0}} \frac{L_{ij}^2}{\hat{D}_{jj}}, \\
\hat{D}_{ii}^{+} &= \left(PMP^T\right)_{ii} - \sum_{\substack{j=1,\dots,i-1\\ \hat{D}_{jj}<0}} \frac{L_{ij}^2}{\hat{D}_{jj}}, \\
\hat{D}_{ii} &= \hat{D}_{ii}^{+} + \hat{D}_{ii}^{-}.
\end{aligned}$$

The result is qualified by the Orchard–Hayes relative acceptability measure [24] that is $\left|\hat{D}_{ii}\right|$ is compared to $\epsilon \hat{D}_{ii}^{+}$, where $\epsilon > 0$ is the pivot tolerance. It was pointed out that the Orchard–Hayes relative acceptability measure gives a scaling–independent information about the reliability of the computed pivot value and can be interpreted as relative Euclidean distance between a vector and a linear subspace [17]. BPMPD uses a modified Cholesky factorization where the modifications are done on pivot elements not satisfying the relative tolerance. If the unacceptable pivot indicates that a row in AF^{-T} lies close to the space spanned by the preceding rows the procedure omits the factor by setting the pivot value to infinity. Otherwise the pivot value is "*regularized*" that is its absolute value is increased to $\epsilon \hat{D}_{ii}^{+}$ while its sign is determined corresponding to the position of the pivot.

Regularization is also used for handling free variables. If there is no nonnegativity restriction for the i-th variable then D_{ii} in (17) is computed as

$$\frac{\mu}{\left(|x_i| + 1.0\right)^2} \vartheta,$$

where $1 \geq \vartheta > 0$ increases as the complementarity gap decreases during iterations. If the augmented system approach is selected, BPMPD uses a similar regularization on the free variables of the dual problem. In this case $M_{j+n,j+n}$ is initialized as

$$\frac{(|y_j| + 1.0)^2}{\mu}\vartheta^{-1}$$

if row $1 \leq j \leq m$ is equality. In such a way the augmented system becomes quasidefinite which allows the application of symbolic ordering methods [27].

Numerical algebra investigation shows that the perturbation of the regularization in the solution of system (10) is the same order of magnitude than the regularization itself. Additionally, the perturbation of the solution can be always reduced by the standard iterative refinement [14].

5 Numerical results

In this section we present numerical results obtained by BPMPD version 2.21. Our tests were performed on a set of QP problems from the QP library [10] on an IBM PC compatible machine equipped with one Intel Pentium 200 MMX processor and 64 Mbyte of memory.

In the first experiment we show the performance of BPMPD. Table 2 collects the results. Figures given include the number of nonzeros in L, the number of operations (given in thousands) to compute one decomposition (18), the number of iterations and the total execution time in seconds. One operation is meant a pair of one multiplication and addition.

In the next experiment, presented by table 3, we compared the costs of the fundamental processes of BPMPD relative to the total execution time. Figures given include the cost of the presolve and scaling (together with descaling and postsolve), the time spent for symbolic analysis, for computing the starting point and the accumulated time of the factorizations during the iterations. Since the algorithm logic requires very limited computational effort, the remaining part of the execution time was spent mostly on backsolve operations and on iterative refinements.

The results in table 3 indicate that it is a very rare case when presolve and scaling requires more than 10% of the total execution time. It is to be observed that on problems with sparse factors the numerical factorizations

Table 2: Computational results on QP problems

Problem name	Nonzeros in L	Operations	Iterations	Solution time
cvxqp1-l	3106765	154267	10	1459.16
cvxqp1-m	66521	540	9	5.62
cvxqp2-l	2400979	112100	10	1040.90
cvxqp2-m	47190	297	10	3.74
cvxqp3-l	3694095	211355	10	2163.34
cvxqp3-m	72540	584	10	7.26
mosarqp1	20851	27	8	1.21
mosarqp2	19539	31	9	0.95
q25fv47	95712	775	18	18.73
qetamacr	17518	47	16	1.81
qfffff80	21138	46	20	2.15
qforplan	8304	10	19	0.88
qgrow15	12336	12	19	1.37
qgrow22	17614	16	22	2.21
qpilotno	54808	222	23	8.19
qscfxm2	11389	9	23	1.51
qscfxm3	17110	14	24	2.26
qscsd8	18451	15	11	1.50
qsctap2	20938	26	14	1.80
qsctap3	29147	42	15	2.68
qshell	7711	7	24	1.88
qship08l	116273	782	9	10.23
qship08s	26164	61	10	2.13
qship12l	168732	1030	11	15.39
qship12s	26618	37	12	2.38
qsierra	16763	10	15	1.81

Table 3: Relative costs of the major procedures in BPMPD

Problem	Presolve	Symb. anal.	IPM start	Factorization	Other
cvxqp1-l	0.03 %	3.11 %	8.68 %	85.28 %	2.89 %
cvxqp1-m	1.42 %	14.77 %	7.83 %	58.72 %	17.26 %
cvxqp2-l	0.04 %	1.99 %	8.77 %	86.12 %	3.09 %
cvxqp2-m	1.87 %	12.03 %	6.95 %	56.15 %	22.99 %
cvxqp3-l	0.03 %	3.70 %	11.23 %	82.78 %	2.25 %
cvxqp3-m	1.24 %	16.94 %	6.61 %	56.47 %	18.73 %
mosarqp1	8.26 %	9.09 %	5.79 %	16.53 %	60.33 %
mosarqp2	7.37 %	11.58 %	6.32 %	21.05 %	53.68 %
q25fv47	1.76 %	5.66 %	3.42 %	52.86 %	36.31 %
qetamacr	5.52 %	9.94 %	4.42 %	38.67 %	41.44 %
qfffff80	6.98 %	4.19 %	5.12 %	37.21 %	46.51 %
qforplan	13.64 %	5.68 %	5.68 %	11.36 %	63.64 %
qgrow15	7.30 %	11.68 %	3.65 %	14.60 %	62.77 %
qgrow22	6.33 %	11.31 %	2.71 %	18.10 %	61.54 %
qpilotno	3.66 %	4.88 %	3.30 %	48.84 %	39.32 %
qscfxm2	9.93 %	3.97 %	3.31 %	13.25 %	69.54 %
qscfxm3	9.29 %	3.98 %	2.65 %	22.12 %	61.95 %
qscsd8	14.00 %	7.33 %	4.00 %	13.33 %	61.33 %
qsctap2	8.89 %	6.67 %	4.44 %	22.22 %	57.78 %
qsctap3	7.46 %	7.84 %	3.73 %	26.12 %	54.85 %
qshell	7.98 %	4.26 %	2.13 %	10.64 %	75.00 %
qship08l	3.42 %	15.64 %	6.74 %	50.83 %	23.36 %
qship08s	7.04 %	19.72 %	5.16 %	23.47 %	44.60 %
qship12l	2.21 %	9.42 %	5.72 %	52.63 %	30.02 %
qship12s	6.72 %	15.13 %	3.78 %	21.01 %	53.36 %
qsierra	10.50 %	5.52 %	4.97 %	16.57 %	62.43 %
qstair	5.98 %	5.98 %	5.98 %	48.91 %	33.15 %

may require significantly less effort than the backsolve operations, while on dense problems the factorization is clearly the most expensive process.

References

[1] E.D. Andersen, J. Gondzio, Cs. Mészáros, and X. Xu. Implementation of interior point methods for large scale linear programs. In T. Terlaky, editor, *Interior point methods of mathematical programming*, pages 189–252. Kluwer Academic Publishers, 1996.

[2] D. Boyd. Calibrating and extrapolating weighted sample data sets by choosing new weights – an application of large-scale nonlinear programming. Applied Topics in Operations Research # 092-6940-02, Rensselaer Polytechnic Institute, 1998.

[3] A.L. Brearley, G. Mitra, and H.P. Williams. Analysis of mathematical programming problems prior to applying the simplex algorithm. *Math. Programming*, 15:54–83, 1975.

[4] T.J. Carpenter, I.J. Lustig, J.M. Mulvey, and D.F. Shanno. Higher order predictor-corrector interior point methods with application to quadratic objectives. *SIAM Journal on Optim.*, 3:696–725, 1993.

[5] B. Jansen, C. Roos, T. Terlaky, and J. P. Vial. Primal–dual target following algorithms for linear programming. *Annals of Operations Research*, 62:197–231, 1996.

[6] M.H. Karwan, V. Lotfi, J. Telgen, and S. Zionts. *Redundancy in Mathematical Programming.* Springer-Verlag, 1983.

[7] A.H. Lobbrecht. *Dymanic Water-System Control, Design and Operation of Regional Water-Resources Systems.* PhD thesis, TU Delft, Department of Civil Engineering, 1997.

[8] I. J. Lustig, R. E. Marsten, and D. F. Shanno. Computational experience with a primal-dual interior point method for linear programming. *Linear Algebra Appl.*, 20:191–222, 1991.

[9] I. J. Lustig, R. E. Marsten, and D. F. Shanno. Interior point methods for linear programming: Computational state of the art. *ORSA Journal on Comput.*, 6(1):1–15, 1994.

[10] I. Maros and Cs. Mészáros. A repository of convex quadratic programming problems. Technical Report DOC 97/6, Imperial College, Department of Computing, London, 1997. submitted to Optimization Methods and Software.

[11] I. Maros and Cs. Mészáros. The role of the augmented system in interior point methods. *European Journal on Operational Research*, 107(3):720–736, 1998.

[12] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optim.*, 2(4):575–601, 1992.

[13] Cs. Mészáros. Fast Cholesky factorization for interior point methods of linear programming. *Computers & Mathematics with Applications*, 31(4/5):49–54, 1996.

[14] Cs. Mészáros. On free variables in interior point methods. *Optimization Methods and Software*, 9:121–139, 1997.

[15] Cs. Mészáros. Steplengths in infeasible primal–dual interior point algorithms of convex quadratic programming. Technical Report DOC 97/7, Imperial College, Department of Computing, London, 1997.

[16] Cs. Mészáros. The augmented system variant of IPMs in two–stage stochastic linear programming. *European Journal on Operational Research*, 101(2):317–327, 1997.

[17] Cs. Mészáros. On a property of the Cholesky factorization and its consequences in interior piont methods. Working paper WP 98–7, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1998.

[18] Cs. Mészáros. On the sparsity issues of interior point methods for quadratic programming. Working paper WP 98–4, Computer and Automation Institute, Hungarian Academy of Sciences, Budapest, 1998.

[19] Cs. Mészáros. Ordering heuristics in interior point LP methods. In F. Gianessy, S. Komlósi, and T. Rapcsák, editors, *New trends in mathematical programming*, pages 203–221. Kluwer Academic Publishers, 1998.

[20] H.D. Mittelmann and P. Spellucci. Decision tree for optimization software. World Wide Web, http://plato.la.asu.edu/guide.html, 1998.

[21] R. D. C. Monteiro and I. Adler. Interior path following primal-dual algorithms. Part II: Convex quadratic programming. *Math. Programming*, 44:43–66, 1989.

[22] R.D.C. Monteiro and F. Zhou. On superlinear convergence of infeasible–interior–point algorithms for linearly constrained convex programs. *Computational Optimization and Applications*, (8):245–262, 1997.

[23] J.L. Nazareth. *Computer Solution of Linear Programs*. Oxford University Press, New York, 1987.

[24] W. Orchard-Hays. *Advanced Linear Programming Computing Techniques*. McGraw Hill, 1968.

[25] C Roos, T. Terlaky, and J.-Ph. Vial. *Theory and Algorithms for Linear Optimization: An Interior Point Approach*. John Wiley & Sons, 1996.

[26] G. W. Stewart. On scaled projections and pseudoinverses. *Linear Algebra Appl.*, 112:189–193, 1989.

[27] R. J. Vanderbei. Symmetric quasi-definite matrices. *SIAM Journal on Optim.*, 5(1):100–113, February 1995.

[28] S. J. Wright. *Primal–dual interior point methods*. SIAM, T57.74.W75, University Science Center, Philadelphia, 1997.