QÜESTIIÓ, vol. 22, 1, p. 117-142, 1998

UN ALGORITMO DE PUNTO INTERIOR PARA PROGRAMACIÓN CUADRÁTICA A TRAVÉS DE PROBLEMAS EQUIVALENTES SEPARABLES*

J. CASTRO*
Universitat Rovira i Virgili

Se presenta un algoritmo de punto interior para la solución de problemas cuadráticos simétricos y definidos positivos, mediante su transformación en problemas equivalentes separables (esto es, la matriz de coeficientes cuadráticos es diagonal y no existen términos cruzados). El algoritmo difiere de otros ya existentes (como el implementado en el sistema LoQo) en el hecho de que soluciona las denominadas «ecuaciones normales en forma primal» (LoQo soluciona el denominado «sistema aumentado») y en que no requiere ningún tratamiento específico para las variables libres creadas durante la obtención del problema equivalente separable. Se presenta una implementación del algoritmo y su eficiencia es comparada con los sistemas LoQo y Minos 5.3. Para la comparación se utilizan 80 problemas cuadráticos derivados de problemas lineales de la colección Netlib (Gay (1985)) (una batería estándar de problemas de programación lineal). Se obtienen los problemas cuadráticos mediante un generador ad-hoc de problemas cuadráticos.

An interior-point algorithm for quadratic programming through separable equivalent problems.

Palabras clave: Algoritmo primal-dual, métodos de punto interior, método predictor-corrector, programación cuadrática

⋆ Este trabajo ha sido subvencionado por la Ayuda Iberdrola a la Investigación Científica y al Desarrollo Tecnológico 95-005, y por el proyecto CICYT TAP96-1044-J02-93.

* J. Castro. Estadística i Investigació Operativa. Dept. d'Enginyeria Química. Universitat Rovira i Virgili. Autovia de Salou, s/n. 43006 Tarragona.

– Recibido en noviembre de 1996.
– Aceptado en julio de 1997.

1. INTRODUCCIÓN

Desde la aparición del algoritmo de Karmarkar (Karmarkar (1984)) se ha dedicado un gran esfuerzo al estudio de los métodos de punto interior, debido fundamentalmente a la gran eficiencia que han mostrado tener en la resolución de ciertos tipos de problemas de programación lineal. Ver en Kranich (1991) una extensa recopilación bibliográfica sobre métodos de punto interior. Ver en Terlaky (1996) los últimos avances conseguidos en este campo.

Durante los primeros años de estudio, los métodos de punto interior fueron aplicados con éxito a problemas de programación lineal. Ver en Monteiro y Adler (1989) un estudio sobre la convergencia de estos métodos para la solución de problemas cuadráticos. Existen diversas implementaciones de carácter práctico, tales como el sistema LoQo (Linear Optimization, Quadratic Optimization), ver Vanderbei (1992).

El algoritmo presentado en este trabajo difiere del implementado en LoQo en tres puntos fundamentalmente. Primero, LoQo utiliza técnicas para matrices simétricas casi-definidas, dado que resuelve el denominado «sistema aumentado» (que no es más que una simplificación del sistema de ecuaciones de Karush-Kuhn-Tucker del problema). Por su parte, nuestro algoritmo utiliza las denominadas «ecuaciones normales en forma primal» (consistentes en una simplificación adicional del «sistema aumentado»); ver en Vanderbei (1996) una descripción de ambas técnicas. En contra del sentir general en la literatura (ver Vanderbei y Carpenter (1993)) que considera que la utilización del «sistema aumentado» reduce el incremento de la degradación de la esparsidad (i.e., poca densidad) del sistema de ecuaciones, en este trabajo se muestra como el uso de las «ecuaciones normales en forma primal» puede dar lugar a mejores rendimientos computacionales. Esto es debido a que, a pesar de tener un aumento en la degradación de la esparsidad, este hecho se ve compensado por la solución de un sistema más simplificado. Asimismo, las «ecuaciones normales en forma primal» también son utilizadas en otras implementaciones eficientes, como OB1 (Lustig *et al.* (1992)).

El segundo punto en donde ambos métodos difieren es en el preproceso previo a la solución del problema, dado que cada método resuelve un sistema de ecuaciones distinto. La utilización de las «ecuaciones normales en forma primal» en nuestro método, fuerza a transformar el problema original en uno equivalente separable. De otro modo, el coste computacional sería excesivo y no podrían obtenerse implementaciones competitivas. Por su parte, la estrategia implementada en LoQo permite tanto transformar el problema en uno equivalente separable como solucionar el problema en su forma original. En este sentido, ofrece una mayor flexibilidad. Sin embargo, tal y como se mostrará en los resultados obtenidos, la implementación de nuestro algoritmo obtiene, en general, mejores rendimientos que LoQo, tanto si éste soluciona el problema original o uno transformado.

El tercer punto a destacar es el tratamiento de las variables cuadráticas libres (éstas aparecen al transformar el problema en uno equivalente separable). LoQo (ver Vanderbei y Carpenter (1993)) efectúa un tratamiento específico para este tipo de variables. El método propuesto en este trabajo, sin embargo, no requiere dicho tratamiento, lo cual permite agilizar su rendimiento.

Este trabajo es una extensión de la implementación de un algoritmo primal-dual para problemas lineales, utilizando un método predictor-corrector descrito en Castro (1998) (que a su vez es una extensión del presentado en Castro (1995)). El trabajo tiene la siguiente organización: la sección 2 recoge la extensión del algoritmo presentado en Castro (1998) al caso cuadrático; la sección 3 detalla la transformación del problema original en uno equivalente y separable; la sección 4 presenta el generador de problemas cuadráticos, la sección 5 discute los resultados computacionales obtenidos en la solución de una batería de 80 problemas cuadráticos; finalmente, la sección 6 ofrece las conclusiones obtenidas de los resultados de este trabajo.

2. ALGORITMO PRIMAL-DUAL PREDICTOR-CORRECTOR PARA PROBLEMAS CUADRÁTICOS

En este apartado se describe el algoritmo primal-dual para problemas cuadráticos como extensión del algoritmo presentado en Castro (1998). De nuevo, la principal diferencia entre el algoritmo aquí detallado y el utilizado por LoQo reside en la solución de las «ecuaciones normales en forma primal», en vez del «sistema aumentado». Para el caso cuadrático se verá cómo este hecho tiene unas repercusiones mayores que para el caso lineal, forzando a transformar el problema cuadrático original en uno equivalente separable.

2.1. Formulación de los problemas primal y dual

Sea la siguiente formulación general del problema cuadrático (en forma primal):

$$(1) \qquad (P) \qquad \begin{array}{ll} \min_x & c'x + \frac{1}{2}x'Qx \\ \text{sujeto a} & A_u x_u + A_l x_l + A_f x_f = b \\ & \underline{0} \leq x_u \leq \overline{x}_u \\ & \underline{0} \leq x_l \\ & x_f \text{ libre} \end{array}$$

donde $x \in \mathbb{R}^n$ representa el vector de variables, el cual puede ser particionado como $x' = (x'_u \; x'_l \; x_f)$ ($x_u \in \mathbb{R}^{n_u}$ corresponde a las variables con límites superiores e inferiores, $x_l \in \mathbb{R}^{n_l}$ denota las variables sólo acotadas inferiormente, y $x_f \in \mathbb{R}^{n_f}$ representa las variables libres —por lo tanto, $n_u + n_l + n_f = n$). $Q \in \mathbb{R}^{n \times n}$ es una matriz simétrica y definida positiva con los costes cuadráticos, y $c \in \mathbb{R}^n$ es el vector de costes lineales. La matriz de restricciones $A \in \mathbb{R}^{m \times n}$ se encuentra particionada según los tres tipos

de variables en $A_u \in \mathbb{R}^{m\times n_u}$, $A_l \in \mathbb{R}^{m\times n_l}$ y $A_f \in \mathbb{R}^{m\times n_f}$. b es el vector de términos independientes, y $\overline{x}_u$ es el vector de los límites superiores de las variables x_u.

Añadiendo variables de holgura $f \in \mathbb{R}^{n_u}$ para las variables acotadas superiormente, sea la siguiente expresión del problema (1) en la forma estándar:

$$(2) \qquad (P) \qquad \begin{array}{ll} \min_{x,f} & c'x + \frac{1}{2}x'Qx \\ \text{sujeto a} & A_u x_u + A_l x_l + A_f x_f = b \\ & x_u + f = \overline{x}_u \\ & x_u \geq \underline{0},\ x_l \geq \underline{0},\ f \geq \underline{0},\ x_f \text{ libre} \end{array}$$

Considerando una partición del vector c y de la matriz Q de acuerdo con los tres tipos de variables $x' = (x'_u\ x'_l\ x_f)$, resulta que $c' = (c'_u\ c'_l\ c_f)$ y

$$(3) \qquad Q = \begin{pmatrix} Q_u & Q'_{ul} & Q'_{uf} \\ Q_{ul} & Q_l & Q'_{lf} \\ Q_{uf} & Q_{lf} & Q_f \end{pmatrix}$$

El dual de (2) tiene la expresión:

$$(4) \qquad (D) \qquad \begin{array}{ll} \max_{y,x,z,w} & b'y - \overline{x}'_u w - \frac{1}{2}x'Qx \\ \text{sujeto a} & A'_u y - (Q_u x_u + Q'_{ul} xl + Q'_{uf} x_f) + z_u - w = c_u \\ & A'_l y - (Q_{ul} x_u + Q_l x_l + Q'_{lf} x_f) + z_l = c_l \\ & A'_f y - (Q_{uf} x_u + Q_{lf} x_l + Q_f x_f) = c_f \\ & z_u \geq \underline{0},\ z_l \geq \underline{0},\ w \geq \underline{0},\ y \text{ libre} \end{array}$$

donde $y \in \mathbb{R}^m$ son las variables duales asociadas con el primer grupo de restricciones de (2), y $z_u \in \mathbb{R}^{n_u}$, $z_l \in \mathbb{R}^{n_l}$ y $w \in \mathbb{R}^{n_u}$ son las variables de holgura duales.

Nota: observar que las variables x_u tienen asociadas las holguras duales z_u y w, las variables x_l únicamente tienen asociadas las holguras duales z_l, y las variables libres x_f no tienen ninguna holgura dual asociada.

2.2. Obtención de las funciones Lagrangianas a través de una barrera logarítmica

Una vez formulados los problemas primal y dual, se pueden eliminar las restricciones de no-negatividad de las variables añadiendo una barrera logarítmica. El problema primal tendrá la siguiente expresión:

$$(5) \qquad \begin{array}{ll} \min & c'x + \frac{1}{2}x'Qx - \mu\sum_{i=1}^{n_u+n_l} \ln x_i - \mu\sum_{i=1}^{n_u} \ln(\overline{x}_{u_i} - x_{u_i}) \\ \text{sujeto a} & Ax = b \\ & \mu \geq 0 \end{array}$$

mientras que el dual puede formularse como:

$$
\begin{array}{ll}
\max & b'y - \frac{1}{2}x'Qx - \overline{x}_u' w + \mu \sum_{i=1}^{n_u+n_l} \ln z_i + \mu \sum_{i=1}^{n_u} \ln w_i \\
\text{sujeto a} & A_u' y - (Q_u x_u + Q_{ul}' x_l + Q_{uf}' x_f) + z_u - w = c_u \\
& A_l' y - (Q_{ul} x_u + Q_l x_l + Q_{lf}' x_f) + z_l = c_l \\
& A_f' y - (Q_{uf} x_u + Q_{lf} x_l + Q_f x_f) = c_f \\
& y \text{ libre}, \quad \mu \geq 0
\end{array} \tag{6}
$$

A continuación, asociando los multiplicadores de Lagrange $y \in \mathbb{R}^m$ a las restricciones de igualdad de (5) se obtiene la función Lagrangiana L_p del problema primal:

$$
L_p(x,y,\mu) = c'x + \frac{1}{2}x'Q - \mu \sum_{i=1}^{n_u+n_l} \ln x_i - \mu \sum_{i=1}^{n_u} \ln(\overline{x}_{u_i} - x_{u_i}) - y'(Ax - b) \tag{7}
$$

Análogamente, asociando los multiplicadores $x_u \in \mathbb{R}^{n_u}$, $x_l \in \mathbb{R}^{n_l}$ y $x_f \in \mathbb{R}^{n_f}$ a las restricciones de igualdad de (6), se obteniene la función Lagrangiana del dual L_d:

$$
\begin{array}{rl}
L_d(x,y,z,w,\mu) & = b'y - \frac{1}{2}x'Qx - \overline{x}_u' w + \mu \sum_{i=1}^{n_u+n_l} \ln z_i + \mu \sum_{i=1}^{n_u} \ln w_i \\
& - x_u'(A_u' y - (Q_u x_u + Q_{ul}' x_l + Q_{uf}' x_f) + z_u - w - c_u) \\
& - x_l'(A_l' y - (Q_{ul} x_u + Q_l x_l + Q_{lf}' x_f) + z_l - c_l) \\
& - x_f'(A_f' y - (Q_{uf} x_u + Q_{lf} x_l + Q_f x_f) - c_f)
\end{array} \tag{8}
$$

2.3. Condiciones de optimalidad de Karush-Kuhn-Tucker de primer orden

Sea e_l el vector l-dimensional de 1's y X_u, X_l, X_f, X, Z_u, Z_l, W, F las matrices diagonales, definidas como:

$$
\begin{array}{rl}
e_l & = (1_1, \ldots, 1_l)' \\
X_u & = \operatorname{diag}(x_{u_1}, \ldots, x_{u_{n_u}}) \\
X_l & = \operatorname{diag}(x_{l_1}, \ldots, x_{l_{n_l}}) \\
X_f & = \operatorname{diag}(x_{l_1}, \ldots, x_{l_{n_f}}) \\
X & = \begin{pmatrix} X_u & & \\ & X_l & \\ & & X_f \end{pmatrix} \\
Z_u & = \operatorname{diag}(z_{u_1}, \ldots, z_{u_{n_u}}) \\
Z_l & = \operatorname{diag}(z_{l_1}, \ldots, z_{l_{n_l}}) \\
W & = \operatorname{diag}(w_1, \ldots, w_{n_u}) \\
F & = \operatorname{diag}(\overline{x}_{u_1} - x_{u_1}, \ldots, \overline{x}_{u_{n_u}} - x_{u_{n_u}})
\end{array} \tag{9}
$$

Teniendo en cuenta la estructura de Q presentada en (3), las condiciones de optimalidad necesarias de primer orden de L_p (7) tienen la siguiente expresión:

$$\frac{\partial L_p}{\partial x_u} = c_u - \mu X_u^{-1} e_{n_u} + \mu F^{-1} e_{n_u} - A'_u y + (Q_u x_u + Q'_{ul} x_l + Q'_{uf} x_f) := 0 \tag{10}$$

$$\frac{\partial L_p}{\partial x_l} = c_l - \mu X_l^{-1} e_{n_l} - A'_l y + (Q_{ul} x_u + Q_l x_l + Q'_{lf} x_f) := 0 \tag{11}$$

$$\frac{\partial L_p}{\partial x_f} = c_f - A'_f y + (Q_{uf} x_u + Q_{lf} x_l + Q_f x_f) := 0 \tag{12}$$

$$\frac{\partial L_p}{\partial y} = -(Ax - b) := 0 \tag{13}$$

Por otro lado, las condiciones de optimalidad de primer orden de L_d (8) tienen la expresión:

$$\frac{\partial L_d}{\partial x_u} = c_u - A'_u y - z_u + w + (Q_u x_u + Q'_{ul} x_l + Q'_{uf} x_f) := 0 \tag{14}$$

$$\frac{\partial L_d}{\partial x_l} = c_l - A'_l y - z_l + (Q_{ul} x_u + Q_l x_l + Q'_{lf} x_f) := 0 \tag{15}$$

$$\frac{\partial L_d}{\partial x_f} = c_f - A'_f y + (Q_{uf} x_u + Q_{lf} x_l + Q_f x_f) := 0 \tag{16}$$

$$\frac{\partial L_d}{\partial y} = b - Ax := 0 \tag{17}$$

$$\frac{\partial L_d}{\partial z_u} = \mu Z_u^{-1} e_{n_u} - X_u e_{n_u} := 0 \tag{18}$$

$$\frac{\partial L_d}{\partial z_l} = \mu Z_l^{-1} e_{n_l} - X_l e_{n_l} := 0 \tag{19}$$

$$\frac{\partial L_d}{\partial w} = \mu W^{-1} e_{n_u} - F e_{n_u} := 0 \tag{20}$$

Las condiciones (13) y (17) son las mismas e imponen la factibilidad primal. Las condiciones (14)–(16) imponen la factibilidad dual. Las condiciones (18)–(20), denominadas condiciones de complementariedad, garantizan que la solución verifique las condiciones de holgura complementaria. Finalmente, puede observarse que, garantizándose la factibilidad primal, la factibilidad dual y la complementariedad, automáticamente quedan satisfechas las condiciones restantes (10)–(12). Por ejemplo, para (10) se puede observar que:

$$\begin{aligned} 0 &\stackrel{?}{=} c_u - \mu X_u^{-1} e_{n_u} + \mu F^{-1} e_{n_u} - A'_u y + (Q_u x_u + Q'_{ul} x_l + Q'_{uf} x_f) \\ &= -\mu X_u^{-1} e_{n_u} + \mu F^{-1} e_{n_u} + z_u - w && \text{[usando (14)]} \\ &= 0 && \text{[usando (18) y (20)]} \end{aligned}$$

De igual forma puede verificarse que se satisfacen (11) y (12).

Finalmente, las condiciones necesarias de Karush-Kuhn-Tucker de primer orden del problema cuadrático (1) (condiciones de factibilidad primal, factibilidad dual y complementariedad) a satisfacer por la solución óptima tienen la siguiente expresión:

$$f_1 \equiv X_u Z_u e_{n_u} - \mu e_{n_u} = 0 \tag{21}$$

$$f_2 \equiv X_l Z_l e_{n_l} - \mu e_{n_l} = 0 \tag{22}$$

$$f_3 \equiv FW e_{n_u} - \mu e_{n_u} = 0 \tag{23}$$

$$f_4 \equiv Ax - b = 0 \tag{24}$$

$$f_5 \equiv A'_u y + z_u - w - (Q_u x_u + Q'_{ul} x_l + Q'_{uf} x_f) - c_u = 0 \tag{25}$$

$$f_6 \equiv A'_l y + z_l - (Q_{ul} x_u + Q_l x_l + Q'_{lf} x_f) - c_l = 0 \tag{26}$$

$$f_7 \equiv A'_f y - (Q_{uf} x_u + Q_{lf} x_l + Q_f x_f) - c_f = 0 \tag{27}$$

2.4. Solución del sistema no lineal

Se resuelve el sistema no lineal de ecuaciones (21)–(27) mediante un método iterativo. Una buena elección consistiría en usar el método de Newton, partiendo del punto actual $\xi'_i = (x'_i,\ y'_i,\ z'_i, w'_i)$ a un nuevo punto $\xi_i + d\xi_i$, donde las sucesivas direcciones $d\xi_i$ se obtienen como solución del sistema $\nabla f(\xi_i) d\xi_i = -f(\xi_i)$. Sin embargo, tal y como se describe para el caso lineal en Castro (1998), puede aprovecharse información de segundo orden y realizar una aproximación cuadrática de $f_j(\xi_i + + d\xi_i) = 0 \quad j = 1,\ldots,7$ (f_j hace referencia a cada una de las ecuaciones de (21)–(27)), obteniendo:

$$f_j(\xi_i) + \nabla f_j(\xi_i)' d\xi_i + \frac{1}{2}\, d\xi'_i \nabla^2 f_j(\xi_i) d\xi_i = 0 \quad j = 1,\ldots,7 \tag{28}$$

Aplicando (28) al sistema (21–27), el nuevo sistema será:

$$\begin{aligned}
Z_u dx_u + X_u dz_u + dZ_u dx_u &= -f_1(\xi_i) \\
Z_l dx_l + X_l dz_l + dZ_l dx_l &= -f_2(\xi_i) \\
-W dx_u + F dw - dW dx_u &= -f_3(\xi_i) \\
A_u dx_u + A_l dx_l + A_f dx_f &= -f_4(\xi_i) \\
-Q_u dx_u - Q'_{ul} dx_l - Q'_{uf} dx_f + A'_u dy + dz_u - dw &= -f_5(\xi_i) \\
-Q_{ul} dx_u - Q_l dx_l - Q'_{lf} dx_f + A'_l dy + dz_l &= -f_6(\xi_i) \\
-Q_{uf} dx_u - Q_{lf} dx_l - Q_f dx_f + A'_f dy &= -f_7(\xi_i)
\end{aligned} \tag{29}$$

Si se hubiera usado el método de Newton sin introducir información de segundo orden, la única diferencia hubiera sido la no aparación de los términos no lineales $dZ_u dx_u$, $dZ_l dx_l$ y $dW dx_u$. Sin estos términos, el sistema resultante es lineal y puede

ser directamente solucionado. Sin embargo, al introducir la información de segundo orden es preciso utilizar una estrategia alternativa. La forma clásica de solucionar este sistema (propuesta inicialmente en Mehrotra (1990)) consiste en realizar dos pasos, denominados el paso predictor y el paso corrector, tal y como se detalla en Castro (1998). En el paso predictor (tal y como indica su nombre) se obtiene una dirección $\hat{d\xi}_i$ aproximada. Dicha dirección se obtiene como solución del sistema (29) eliminando las no linealidades y los términos μ de los términos independientes de las tres primeras ecuaciones:

$$
\begin{aligned}
Z_u\hat{dx}_u + X_u\hat{dz}_u &= -X_uZ_ue_{n_u} \\
Z_l\hat{dx}_l + X_l\hat{dz}_l &= -X_lZ_le_{n_l} \\
-W\hat{dx}_u + F\hat{dw} &= -WFe_{n_u} \\
A_u\hat{dx}_u + A_l\hat{dx}_l + A_f\hat{dx}_f &= -f_4(\xi_i) \\
-Q_u\hat{dx}_u - Q'_{ul}\hat{dx}_l - Q'_{uf}\hat{dx}_f + A'_u\hat{dy} + \hat{dz}_u - \hat{dw} &= -f_5(\xi_i) \\
-Q_{ul}\hat{dx}_u - Q_l\hat{dx}_l - Q'_{lf}\hat{dx}_f + A'_l\hat{dy} + \hat{dz}_l &= -f_6(\xi_i) \\
-Q_{uf}\hat{dx}_u - Q_{lf}\hat{dx}_l - Q_f\hat{dx}_f + A'_f\hat{dy} &= -f_7(\xi_i)
\end{aligned}
\tag{30}
$$

Se utiliza la dirección aproximada $\hat{d\xi}_i$ para aproximar los términos no lineales $\hat{dZ}_u\hat{dx}_u$, $\hat{dZ}_l\hat{dx}_l$ y $\hat{dW}\hat{dx}_u$ de (29). Entonces, la dirección $d\xi_i$ a utilizar viene dada por la solución del paso corrector:

$$
\begin{aligned}
Z_udx_u + X_udz_u &= -f_1(\xi_i) - \hat{dZ}_u\hat{dx}_u \\
Z_ldx_l + X_ldz_l &= -f_2(\xi_i) - \hat{dZ}_l\hat{dx}_l \\
-Wdx_u + Fdw &= -f_3(\xi_i) + \hat{dW}\hat{dx}_u \\
A_udx_u + A_ldx_l + A_fdx_f &= -f_4(\xi_i) \\
-Q_udx_u - Q'_{ul}dx_l - Q'_{uf}dx_f + A'_udy + dz_u - dw &= -f_5(\xi_i) \\
-Q_{ul}dx_u - Q_ldx_l - Q'_{lf}dx_f + A'_ldy + dz_l &= -f_6(\xi_i) \\
-Q_{uf}dx_u - Q_{lf}dx_l - Q_fdx_f + A'_fdy &= -f_7(\xi_i)
\end{aligned}
\tag{31}
$$

Puede observarse cómo los términos de la izquierda de los sistemas (30) y (31) son iguales, y tan sólo varían los términos independientes. Por lo tanto únicamente debe solucionarse el mismo sistema de ecuaciones lineal dos veces con distintos términos independientes.

2.5. Solución del sistema lineal

Sea el siguiente sistema donde los términos de la izquierda son los correspondientes de (30) y (31):

$$
\begin{aligned}
Z_u dx_u + X_u dz_u &= b_{1_u} \\
Z_l dx_l + X_l dz_l &= b_{1_l} \\
-W dx_u + F dw &= b_2 \\
A_u dx_u + A_l dx_l + A_f dx_f &= b_3 \\
-Q_u dx_u - Q'_{ul} dx_l - Q'_{uf} dx_f + A'_u dy + dz_u - dw &= b_{4_u} \\
-Q_{ul} dx_u - Q_l dx_l - Q'_{lf} dx_f + A'_l dy + dz_l &= b_{4_l} \\
-Q_{uf} dx_u - Q_{lf} dx_l - Q_f dx_f + A'_f dy &= b_{4_f}
\end{aligned} \tag{32}
$$

Para simplificar la obtención de la solución de (32), sea el sistema equivalente:

$$\tilde{Z}dx + X\tilde{dz} = \tilde{b}1 \tag{33}$$

$$-\tilde{W}dx + \tilde{F}\tilde{dw} = \tilde{b}_2 \tag{34}$$

$$Adx = b_3 \tag{35}$$

$$-Qdx + A'dy + \tilde{dz} - \tilde{dw} = b_4 \tag{36}$$

donde ahora todos los términos (matrices o vectores) se encuentran constituidos por tres partes (submatrices o subvectores) asociadas a los tres tipos de variables ($_u$,$_l$,$_f$), de forma que:

(37)

$$dx = \begin{pmatrix} dx_u \\ dx_l \\ dx_f \end{pmatrix} \quad \tilde{dz} = \begin{pmatrix} dz_u \\ dz_l \\ \mathbf{0} \end{pmatrix} \quad \tilde{dw} = \begin{pmatrix} dw \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} \quad \tilde{b}_1 = \begin{pmatrix} b_{1_u} \\ b_{1_l} \\ \mathbf{0} \end{pmatrix} \quad \tilde{b}_2 = \begin{pmatrix} b_2 \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix}$$

$$b_4 = \begin{pmatrix} b_{4_u} \\ b_{4_l} \\ b_{4_f} \end{pmatrix} \quad \tilde{Z} = \begin{pmatrix} Z_u & & \\ & Z_l & \\ & & \mathbf{0} \end{pmatrix} \quad \tilde{W} = \begin{pmatrix} W & & \\ & \mathbf{0} & \\ & & \mathbf{0} \end{pmatrix} \quad \tilde{F} = \begin{pmatrix} F & & \\ & \mathbf{1} & \\ & & \mathbf{1} \end{pmatrix}$$

Se inicia la solución de (33)–(36) aislando $\tilde{d_w}$ de (34) y $\tilde{d_z}$ de (36):

$$
\begin{aligned}
\tilde{dw} &= \tilde{F}^{-1}(\tilde{b}_2 + \tilde{W}dx) \\
\tilde{dz} &= b_4 + \tilde{dw} + Qdx - A'dy
\end{aligned} \tag{38}
$$

Sustituyendo (38) en (33) resulta:

(39)

$$
\begin{aligned}
\tilde{Z}dx + X\tilde{dz} &= \tilde{b}1 \\
\tilde{Z}dx + X(b_4 + \tilde{dw} + Qdx - A'dy) &= \tilde{b}1 \\
(\tilde{Z} + XQ)dx &= \tilde{b}1 - Xb_4 + XA'dy - X[\tilde{F}^{-1}(\tilde{b}_2 + \tilde{W}dx)] \\
(\tilde{Z} + XQ)dx &= \tilde{b}1 - Xb_4 + XA'dy - X\tilde{F}^{-1}\tilde{b}_2 - X\tilde{F}^{-1}\tilde{W}dx \\
(\tilde{Z} + XQ + X\tilde{F}^{-1}\tilde{W})dx &= \tilde{b}1 - Xb_4 + XA'dy - X\tilde{F}^{-1}\tilde{b}_2 \\
(Q + \tilde{Z}X^{-1} + \tilde{F}^{-1}\tilde{W})dx &= X^{-1}\tilde{b}1 - b_4 + XA'dy - \tilde{F}^{-1}\tilde{b}_2
\end{aligned}
$$

Ahora definiendo:

$$\begin{aligned} \Theta &= (Q + \tilde{Z}X^{-1} + \tilde{F}^{-1}\tilde{W}) \\ r &= \tilde{F}^{-1}\tilde{b}_2 + b_4 - X^{-1}\tilde{b}1 \end{aligned} \tag{40}$$

la expresión (39) puede escribirse como:

$$\Theta dx = A'dy - r \tag{41}$$

Dado que Θ es invertible (ya que es simétrica y definida positiva, puesto que Q también lo es y $\tilde{Z}X^{-1}$, $\tilde{F}^{-1}\tilde{W}$ no son más que matrices diagonales con elementos no negativos), se puede sustituir (41) en (35):

$$\begin{aligned} Adx &= b_3 \\ A[\Theta^{-1}(A'dy - r)] &= b_3 \\ (A\Theta^{-1}A')dy &= b_3 + A\Theta^{-1}r \end{aligned} \tag{42}$$

Por lo tanto, la solución del sistema equivalente (33)–(36) se obtiene calculando por este orden:

$$\begin{aligned} (A\Theta^{-1}A')dy &= b_3 + A\Theta^{-1}r \\ dx &= \Theta^{-1}(A'dy - r) \\ \tilde{d}w &= \tilde{F}^{-1}(\tilde{b}_2 + \tilde{W}dx) \\ \tilde{d}z &= b_4 + \tilde{d}w + Qdx - A'dy \end{aligned} \tag{43}$$

Una vez obtenida la solución del sistema equivalente, es preciso reescribir las ecuaciones de (43) en función de las variables originales, usando (37).

En primer lugar sean las expresiones de Θ y r (40) en base a (3),

(44)

$$\begin{aligned} \Theta &= (Q + \tilde{Z}X^{-1} + \tilde{F}^{-1}\tilde{W}) \\ &= Q + \begin{pmatrix} Z_u & & \\ & Z_l & \\ & & \mathbf{0} \end{pmatrix} \begin{pmatrix} X_u^{-1} & & \\ & X_l^{-1} & \\ & & X_f^{-1} \end{pmatrix} + \begin{pmatrix} F^{-1} & & \\ & \mathbf{1} & \\ & & \mathbf{1} \end{pmatrix} \begin{pmatrix} W & & \\ & \mathbf{0} & \\ & & \mathbf{0} \end{pmatrix} \\ &= \begin{pmatrix} Q_u + Z_u X_u^{-1} + F^{-1}W & Q'_{ul} & Q'_{uf} \\ Q_{ul} & Q_l + Z_l X_l^{-1} & Q'_{lf} \\ Q_{uf} & Q_{lf} & Q_f \end{pmatrix} \end{aligned}$$

$$
\begin{aligned}
r &= \tilde{F}^{-1}\tilde{b}_2 + b_4 - X^{-1}\tilde{b}1 \\
&= \begin{pmatrix} F^{-1} & & \\ & \mathbf{1} & \\ & & \mathbf{1} \end{pmatrix} \begin{pmatrix} b_2 \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} + \begin{pmatrix} b_{4_u} \\ b_{4_l} \\ b_{4_f} \end{pmatrix} - \begin{pmatrix} X_u^{-1} & & \\ & X_l^{-1} & \\ & & X_f^{-1} \end{pmatrix} \begin{pmatrix} b_{1_u} \\ b_{1_l} \\ \mathbf{0} \end{pmatrix} \\
&= \begin{pmatrix} r_u \\ r_l \\ r_f \end{pmatrix} = \begin{pmatrix} F^{-1}b_2 + b_{4_u} - X_u^{-1}b_{1_u} \\ b_{4_l} - X_l^{-1}b_{1_l} \\ b_{4_f} \end{pmatrix}
\end{aligned} \tag{45}
$$

La expresión de $\tilde{dw}$ (38) en las variables originales será:

$$
\begin{aligned}
\tilde{dw} &= \tilde{F}^{-1}(\tilde{b}_2 + \tilde{W}dx) \\
&= \begin{pmatrix} F^{-1} & & \\ & \mathbf{1} & \\ & & \mathbf{1} \end{pmatrix} \left[\begin{pmatrix} b_2 \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} + \begin{pmatrix} W & & \\ & \mathbf{0} & \\ & & \mathbf{0} \end{pmatrix} \begin{pmatrix} dx_u \\ dx_l \\ dx_f \end{pmatrix} \right] \\
&= \begin{pmatrix} F^{-1}(b_2 + W dx_u) \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} \\
&\Downarrow \\
dw &= F^{-1}(b_2 + W dx_u)
\end{aligned} \tag{46}
$$

La expresión de $\tilde{dz}$ en las variables originales será:

$$
\begin{aligned}
\tilde{dz} &= b_4 + Qdx - A'dy + \tilde{dw} \\
&= \begin{pmatrix} b_{4_u} \\ b_{4_l} \\ b_{4_f} \end{pmatrix} + Q \begin{pmatrix} dx_u \\ dx_l \\ dx_f \end{pmatrix} - \begin{pmatrix} A'_u \\ A'_l \\ A'_f \end{pmatrix} dy + \begin{pmatrix} dw \\ \mathbf{0} \\ \mathbf{0} \end{pmatrix} \\
&= \begin{pmatrix} b_{4_u} + (Q_u dx_u + Q'_{ul}dx_l + Q'_{uf}dx_f) - A'_u dy + dw \\ b_{4_l} + (Q_{ul}dx_u + Q_l dx_l + Q'_{lf}dx_f) - A'_l dy \\ b_{4_f} + (Q_{uf}dx_u + Q_{lf}dx_l + Q_f dx_f) - A'_f dy \end{pmatrix} \\
&\Downarrow
\end{aligned}
$$

$$dz_u = b_{4_u} + (Q_u dx_u + Q'_{ul}dx_l + Q'_{uf}dx_f) - A'_u dy + dw \tag{47}$$

$$dz_l = b_{4_l} + (Q_{ul}dx_u + Q_l dx_l + Q'_{lf}dx_f) - A'_l dy \tag{48}$$

$$0 = b_{4_f} + (Q_{uf}dx_u + Q_{lf}dx_l + Q_f dx_f) - A'_f dy \tag{49}$$

Se puede observar fácilmente que la ecuación (49) se satisface. Para ello basta observar que, usando la definición de Θ en (44), el término $Q_{uf}dx_u + Q_{lf}dx_l + Q_f dx_f$

es igual a las n_f últimas componentes de Θdx. Usando ahora la relación $\Theta dx = A'dy - r$ de (41), directamente se tiene que las últimas n_f componentes de Θdx son igual a $A'_f dy - r_f$, donde $r_f = b_{4_f}$ tal y como fue definido en (45). Por lo tanto

$$\begin{aligned} 0 &\stackrel{?}{=} b_{4_f} + (Q_{uf}dx_u + Q_{lf}dx_l + Q_f dx_f) - A'_f dy \\ &= b_{4_f} + (A'_f dy - r_f) - A'_f dy \\ &= b_{4_f} - r_f \\ &= 0 \end{aligned}$$

Finalmente, la solución del sistema lineal (32) tiene la siguiente expresión:

$$(A\Theta^{-1}A')dy = b_3 + A\Theta^{-1}r \tag{50}$$

$$dx = \Theta^{-1}(A'dy - r) \tag{51}$$

$$dw = F^{-1}(b_2 + Wdx_u) \tag{52}$$

$$dz_u = b_{4_u} + (Q_u dx_u + Q'_{ul}dx_l + Q'_{uf}dx_f) - A'_u dy + dw \tag{53}$$

$$dz_l = b_{4_l} + (Q_{ul}dx_u + Q_l dx_l + Q'_{lf}dx_f) - A'_l dy \tag{54}$$

donde Θ y r se recogen en (44) y (45), respectivamente.

Nota: Una vez calculada la dirección de movimiento, el resto de pasos del algoritmo (cálculo del paso y actualización del nuevo punto) se realiza de forma análoga al caso del problema lineal. Únicamente hay que tener en cuenta que no se debe hacer intervenir las variables libres al calcular la longitud de paso que preserve la no-negatividad de las variables (puesto que por ser libres no tienen esta restricción).

2.6. Notas sobre la solución obtenida

- Si no existiera la matriz Q (se tiene, por tanto, un problema lineal con variables libres), y según la definición (44), se puede observar que en este caso la matriz Θ_l (que representa la matriz Θ para el caso de sólo disponer de variables lineales) vendría dada por:

$$\Theta_l = \begin{pmatrix} Z_u X_u^{-1} + F^{-1}W & & \\ & Z_l X_l^{-1} & \\ & & \mathbf{0} \end{pmatrix} \tag{55}$$

  y, claramente, no sería invertible. Por lo tanto, el método descrito anteriormente no sería viable para problemas lineales con variables libres. En el caso de problemas

cuadráticos, podrá invertirse la matriz Θ siempre que al sumar Q a la matriz Θ_l anterior se rompa la singularidad. Si Q es simétrica y definida positiva este hecho siempre se garantizará. No hace falta, sin embargo, imponer una condición tan restrictiva. Por ejemplo, una matriz Q diagonal con ceros en las $n_u + n_l$ primeras posiciones diagonales y valores positivos en las n_f últimas posiciones diagonales, también garantizaría la invertibilidad de Θ. Ver en el apartado siguiente una aplicación de una matriz con estas características.

- De nuevo y como en el caso lineal, la operación más costosa es la solución de un sistema simétrico y definido positivo: $(A\Theta^{-1}A')dy = b_3 + A\Theta^{-1}r$. Sin embargo, a diferencia del caso lineal donde Θ era una matriz diagonal, en el caso cuadrático hay que factorizar Θ^{-1} en cada iteración. Y una vez factorizada, utilizar esta factorización para obtener $A\Theta^{-1}A'$ y poder realizar su posterior descomposición de Cholesky. El coste computacional de estas operaciones es elevado. Naturalmente, si la matriz Q fuera diagonal, el coste computacional del sistema anterior sería equivalente al de un problema lineal (ver la siguiente sección). Vanderbei y Carpenter (1993) proponen la solución del ya citado «sistema aumentado» (método implementado en LoQo) en vez del sistema de ecuaciones «normales en forma primal» $(A\Theta^{-1}A')dy = b_3 + A\Theta^{-1}r$.
- Una posible forma de proceder en un problema lineal con variables libres es como sigue, ver Vanderbei y Carpenter (1993). Se observó en (55) que en este caso la matriz Θ_l no era invertible, debido a la no existencia de las holguras duales z_f. Por tanto, se pueden considerar existentes dichas holguras z_f, asociadas con las variables libres x_f. Sin embargo, $z_f = 0$ en el óptimo forzosamente (ya que teóricamente no deberían existir estas holguras duales). Dado que siempre se debe garantizar que $z_{f_i} x_{f_i} = 0 \ \ \forall i = 1, \ldots, n_f$, y se pretende que $z_{f_i} = 0$, se debe asegurar que $x_{f_i} \neq 0$. Se obtendrá entonces el paso de las variables primales sin tener en cuenta las variables libres x_f. Si con este paso alguna de las variables libres x_{f_i} pasa a tener un valor cercano a 0 o negativo (inicialmente todas las x_f son positivas), entonces se realiza el cambio de variable $x'_{f_i} = k - x_{f_i}$ donde k es un valor positivo suficientemente alejado de 0 (Vanderbei y Carpenter (1993) proponen $k = 2$) modificando convenientemente la estructura del problema (negar una columna de A y modificar el término de la derecha b). En el nuevo espacio de variables, x'_{f_i} tendrá un valor suficientemente positivo, y la z'_{f_i} asociada deberá valer 0. Este proceso deberá repetirse a lo largo del algoritmo tantas veces como sea necesario.

3. TRANSFORMACIÓN DEL PROBLEMA CUADRÁTICO EN UNO EQUIVALENTE SEPARABLE

El algoritmo descrito en la sección anterior para problemas cuadráticos tiene el inconveniente computacional de tener que factorizar $A\Theta^{-1}A'$ en cada iteración. Si la matriz Θ fuera diagonal (el problema se denomina en este caso «separable») el esfuer-

zo computacional sería equivalente al realizado para problemas lineales. La estrategia a usar consistirá en transformar el problema original en uno equivalente donde la matriz Q sea diagonal. Vanderbei y Carpenter (1993), entre otros, utilizan también esta estrategia para el «sistema aumentado». En cambio, en este trabajo se observará el rendimiento logrado usando la transformación del problema cuando se solucionan las ecuaciones «normales en forma primal».

Sea el siguiente problema cuadrático:

$$\begin{array}{ll} \min_x & c'x + \frac{1}{2}x'Qx \\ \text{sujeto a} & A_u x_u + A_l x_l = Ax = b \\ & \underline{0} \leq x_u \leq \overline{x}_u \\ & \underline{0} \leq x_l \end{array} \tag{56}$$

Factorizando la matriz Q de forma que:

$$Q = LL' \tag{57}$$

la nueva expresión de la función objetivo será:

$$c'x + \frac{1}{2}x'Qx = c'x + \frac{1}{2}x'LL'x = c'x + \frac{1}{2}x'_f x_f \qquad \text{donde } x_f = L'x$$

El problema original (56) será:

$$\begin{array}{ll} \min_x & c'x + \frac{1}{2}x'_f x_f \\ \text{sujeto a} & Ax = b \\ & L'x - x_f = 0 \\ & \underline{0} \leq x_u \leq \overline{x}_u \\ & \underline{0} \leq x_l \\ & x_f \text{ libre} \end{array} \tag{58}$$

La matriz del nuevo problema, que será denotada por $\tilde{Q}$, es ahora:

$$\tilde{Q} = \begin{pmatrix} \mathbf{0} & & \\ & \mathbf{0} & \\ & & \mathbf{1} \end{pmatrix} \tag{59}$$

donde la matriz identidad corresponde a las variables libres x_f, y los bloques diagonales asociados con x_u y x_l son $\mathbf{0}$. El nuevo problema (58) tiene la ventaja de que la matriz $\tilde{Q}$ es diagonal, lo cual reducirá el esfuerzo computacional. El inconveniente es que tiene n_f nuevas variables y restricciones, donde $n_f = \text{rango}(Q) = \text{rango}(L)$. Sin embargo, si la esparsidad de la matriz L definida en (57) es grande, puede esperarse una mayor rapidez en la resolución del problema equivalente (58) que optimizando directamente el original (56).

Para poder aplicar el algoritmo desarrollado en el apartado anterior, se vió que se debía garantizar que la matriz Θ definida en (44) fuera invertible. Sin embargo, usando la definición (59) de $\tilde{Q}$ se observa que:

$$\Theta = \tilde{Q} + \begin{pmatrix} Z_u X_u^{-1} & & \\ & Z_l X_l^{-1} & \\ & & \mathbf{0} \end{pmatrix} + \begin{pmatrix} F^{-1}W & & \\ & \mathbf{0} & \\ & & \mathbf{0} \end{pmatrix}$$

$$= \begin{pmatrix} Z_u X_u^{-1} + F^{-1}W & & \\ & Z_l X_l^{-1} & \\ & & \mathbf{1} \end{pmatrix}$$

y, por tanto, claramente es una matriz invertible. Además, dado que las únicas variables libres son las creadas en el proceso de transformación de un problema a otro, y que la submatriz de Θ asociada con estas variables será siempre $\mathbf{1}$, no es necesario realizar ningún tratamiento específico para variables libres, con el consiguiente ahorro computacional.

4. GENERACIÓN DE PROBLEMAS TESTS CUADRÁTICOS

Para poder verificar el rendimiento computacional de la implementación realizada del algoritmo anterior y su comparación con implementacines alternativas, se ha desarrollado un generador automático de problemas cuadráticos con Q simétrica y definida positiva. A diferencia del caso lineal, donde existe la batería estándard de problemas Netlib, para el caso cuadrático no se conoce de ninguna colección de problemas simétricos y definidos positivos usados como batería de tests. A pesar de existir ya algunos generadores aleatorios de problemas cuadráticos (como el descrito en Calamai *et al.* (1993)), estos generan problemas con una estructura totalmente aleatoria. Otros autores (como Vanderbei y Carpenter (1993)) han generado problemas cuadráticos a partir de los problemas lineales de la Netlib, obteniendo la matriz Q como $Q = A_r' A_r$, donde A_r representa las r% primeras filas de la matriz de restricciones lineales A (siendo r un parámetro preestablecido).

En este trabajo se ha optado por una opción intermedia entre las dos anteriores. Por un lado, los problemas cuadráticos serán generados a partir de un problema lineal ya existente (se han usado los problema Netlib). Así, el problema generado no será totalmente aleatorio y de estructura cualquiera. Sin embargo, sí ha sido obtenida aleatorimente la estructura y coeficientes de la matriz Q, en vez de definirla a partir de la matriz de restricciones lineales A. De hecho, si se usara la técnica antes comentada de definir $Q = A_r' A_r$, claramente al convertir el problema cuadrático en uno equivalente separable, la nueva matriz de restricciones $\tilde{A}$ que aparecería, según (58), vendría dada

por

$$\tilde{A} = \begin{array}{|c|c|} \hline A & \mathbf{0} \\ \hline A_r & \mathbf{1} \\ \hline \end{array}$$

Dado que A_r no es más que una submatriz de A, este hecho influiría al construir la matriz $\tilde{A}\tilde{\Theta}\tilde{A}$, la cual estaría formada por submatrices que serían unas subconjuto de las otras (resultando, por tanto, patrones de esparsidad repetidos).

El generador desarrollado, por tanto, genera la matriz Q de forma aleatoria a partir de tres parámetros: p_1, p_2 y s. El primer parámetro (con rango de 1 a 100) es usado para calcular la dimensión de la matriz Q como $n_q = n \cdot p_1/100$, siendo n el número de variables del problema, donde n_q denota la dimensión de Q. Las n_q variables que intervendrán en la parte cuadrática se escogen aleatoriamente de entre las $n_u + n_l$ variables lineales del problema (56). El segundo parámetro (también expresado como un porcentaje) es usado para obtener el número de elementos diferentes de 0 de la parte sobrediagonal de Q (la parte subdiagonal es simétrica y la diagonal siempre tiene un coeficiente diferente de 0). El número de elementos diferentes de 0, denotado por nz_q, viene dado por $nz_q = n_q^2 \cdot p_2/100$. La distribución de estos nz_q elementos se hace de forma aleatoria sobre la parte sobrediagonal de Q. Los valores que pueden tomar los coeficientes sobrediagonales de Q se obtienen aleatoriamente de una distribución uniforme $[0, M]$, donde $M = \sum_{i=1}^{n} c_i/n$, siendo c_i el coste lineal de la variable x_i (por tanto, M es la media del vector de costes lineales c). Para obtener de forma aleatoria el patrón de esparsidad y los coeficientes cuadráticos se utiliza el generador pseudo-aleatorio descrito por Schrage (1979), el cual es alimentado inicialmente con la semilla proporcionada por el tercer parámetro s. Una vez se han obtenido los coeficientes sobrediagonales, se obtienen los términos diagonales $Q_{ii},\ i = 1, \ldots, n$ aleatoriamente de una distribución uniforme $[\sum_{j=1}^{n} Q_{ij}, \sum_{j=1}^{n} Q_{ij} + M]$, garantizando de esta forma que la matriz será diagonal dominante y definida positiva.

Se han utilizado los 80 problemas lineales de la colección Netlib para la obtención de una batería de problemas tests cuadráticos, según la técnica descrita anteriormente. Las características de los problemas lineales Netlib se muestran en la tabla 1 donde *m*, *n* y *nel* son el número de condiciones, variables y elementos no nulos de la matriz de restricciones. Las características de la matriz Q generada en cada caso (dimensión y número de elementos sobrediagonales no nulos) puede inferirse a partir de las definiciones anteriores de n_q y nz_q, de los parámetros p_1, p_2 y s (concretamente, se han usado en todas las pruebas $p_1 = 5$, $p_2 = 5$ y $s = 3141592$) y del número de variables de cada problema según se recoge en la tabla 1.[1]

[1]Contactar con el autor para obtener una copia del generador (programado en ANSI-C) en jcastro@etse.urv.es.

Tabla 1. *Dimensiones de los problemas Netlib*

Problema	*m*	*n*	*nel*
25fv47	822	1571	11127
80bau3b	2263	9799	29063
adlittle	57	97	465
afiro	28	32	88
agg	489	163	2541
agg2	517	302	4515
agg3	517	302	4531
bandm	306	472	2659
beaconfd	174	262	3476
blend	75	83	521
bnl1	644	1175	6129
bnl2	2325	3489	16124
boeing1	351	384	3865
boeing2	167	143	1339
bore3d	234	315	1525
brandy	221	249	2150
czprob	930	3523	14173
d2q06c	2172	5167	35674
d6cube	416	6184	43888
degen2	445	534	4449
degen3	1504	1818	26230
e226	224	282	2767
etamacro	401	688	2489
fffff800	525	854	6235
finnis	498	614	2714
fit1d	25	1026	14430
fit1p	628	1677	10894
fit2d	26	10500	138018
fit2p	3001	13525	60784
ganges	1310	1681	7021
gfrd-pnc	617	1092	3467
greenbea	2393	5405	31499
grow15	301	645	5665
grow22	441	946	8318
grow7	141	301	2633
israel	175	142	2358
kb2	44	41	291
lotfi	154	308	1086
maros	847	1443	10006
maros-r7	3137	9408	151120

Problema	*m*	*n*	*nel*
nesm	663	2923	13988
pilot	1442	3652	43220
pilot87	2031	4883	73804
pilotnov	976	2172	13129
recipe	92	180	752
sc105	106	103	281
sc205	206	203	552
sc50a	51	48	131
sc50b	51	48	119
scagr25	472	500	2029
scagr7	130	140	553
scfxm1	331	457	2612
scfxm2	661	914	5229
scfxm3	991	1371	7846
scorpion	389	358	1708
scrs8	491	1169	4029
scsd1	78	760	3148
scsd6	148	1350	5666
scsd8	398	2750	11334
sctap1	301	480	2052
sctap2	1091	1880	8124
sctap3	1481	2480	10734
seba	516	1028	4874
share1b	118	225	1182
share2b	97	79	730
shell	537	1775	4900
ship04l	403	2118	8450
ship04s	403	1458	5810
ship08l	779	4283	17085
ship08s	779	2387	9501
ship12l	1152	5427	21597
ship12s	1152	2763	10941
sierra	1228	2036	9252
standata	360	1075	3038
standgub	362	1184	3147
standmps	468	1075	3686
stocfor1	118	111	474
stocfor2	2158	2031	9492
wood1p	245	2594	70216
woodw	1099	8405	37478

Tabla 2. *Efectividad de los sistemas IPQ, LoQo y Minos en los problemas cuadráticos*

	IPPC		IP		LoQo		Minos 5.3
Problema	niter	t	niter	t	niter	t	t
25fv47	26	31.7	31	57.9	32	135.0	312.5
80bau3b	35	900.7	(a)		146	6675.9	1211.7
adlittle	15	0.1	18	0.5	18	0.6	0.4
afiro	10	0.0	13	0.3	13	0.3	0.1
agg	41	8.4	27	4.2	26	5.2	2.0
agg2	42	15.1	26	15.5	27	21.1	4.1
agg3	36	13.3	24	13.9	24	18.8	3.9
bandm	21	1.7	22	2.8	22	3.7	4.0
beaconfd	14	1.3	17	2.0	18	2.3	1.4
blend	15	0.2	17	0.5	17	0.6	0.4
bnl1	50	18.0	65	28.3	64	147.4	29.8
bnl2	37	244.5	(b)		70	3104.7	424.7
boeing1	34	4.8	43	8.1	41	28.1	9.3
boeing2	23	0.9	32	1.9	22	1.5	1.3
bore3d	21	1.0	21	1.7	23	1.9	1.0
brandy	16	1.0	23	2.8	22	3.1	2.1
czprob	45	35.2	61	24.4	86	52.8	109.4
d2q06c	37	636.4	53	742.2	(b)		4777.0
d6cube	53	316.4	36	247.8	(a)		1058.5
degen2	14	6.1	20	11.8	20	17.6	11.3
degen3	20	170.9	23	197.5	23	233.4	245.2
e226	23	1.6	22	3.1	23	4.4	4.8
etamacro	31	6.6	42	15.6	42	18.3	7.2
fffff800	55	18.1	37	25.9	101	125.5	5.8
finnis	27	3.4	29	4.3	36	9.2	10.1
fit1d	27	4.1	23	8.3	25	8.7	23.3
fit1p	15	237.1	30	8.4	30	25.2	25.6
fit2d	(a)		26	341.8	29	374.3	3921.0
fit2p	(a)		41	142.0	43	698.6	2285.2
ganges	38	21.6	32	18.9	34	41.2	12.4
gfrd-pnc	52	3.9	(b)		(b)		7.5
greenbea	125	832.0	(b)		63	1588.4	1209.0
grow15	21	3.0	28	7.3	31	93.7	15.3
grow22	24	6.4	32	13.1	43	305.4	26.2
grow7	20	1.0	25	2.9	26	12.4	3.7
israel	46	8.8	28	2.4	28	4.2	3.6
kb2	15	0.1	16	0.3	16	0.3	0.2
lotfi	23	0.6	38	1.9	18	3.3	1.5
maros	(b)		45	68.1	76	251.4	96.4
maros-r7	(a)		(a)		29	7981.5	(b)

(a) Memoria insuficiente. (b) Problemas de convergencia.

Tabla 2. (cont.) *Efectividad de los sistemas IPQ, LoQo y Minos en los problemas cuadráticos*

	IPPC		IP		LoQo		Minos 5.3
nesm	39	47.2	38	40.9	25	193.9	103.9
pilot	46	850.0	51	1049.6	[b]		2886.4
pilot87	50	3312.9	71	6291.6	70	14929.4	8679.1
pilotnov	42	82.1	40	88.7	[b]		317.2
recipe	22	0.3	16	0.7	16	0.8	0.4
sc105	15	0.1	14	0.4	17	0.6	0.3
sc205	16	0.3	53	2.1	20	1.1	0.5
sc50a	13	0.1	15	0.4	15	0.4	0.2
sc50b	12	0.1	18	0.3	18	0.3	0.2
scagr25	28	1.7	41	3.6	43	5.6	6.3
scagr7	19	0.2	20	0.6	20	0.8	0.5
scfxm1	26	2.2	26	3.4	29	6.0	3.2
scfxm2	25	5.7	30	8.1	33	13.8	12.7
scfxm3	33	14.6	35	16.6	38	26.2	24.2
scorpion	15	0.8	25	2.0	27	3.0	1.7
scrs8	25	4.1	34	6.2	34	24.9	8.2
scsd1	18	0.8	17	1.5	18	127.7	1.9
scsd6	16	2.0	18	2.9	26	715.5	4.5
scsd8	18	10.3	23	9.0	25	1083.9	22.2
sctap1	18	1.1	32	2.7	31	10.3	2.4
sctap2	22	18.1	28	16.7	31	325.3	22.0
sctap3	21	34.5	28	22.2	30	479.7	36.3
seba	30	55.2	59	8.3	58	14.2	4.3
share1b	55	0.9	22	1.2	28	1.7	1.1
share2b	13	0.2	22	0.8	23	0.9	0.5
shell	47	6.9	33	5.9	34	27.4	5.1
ship04l	14	5.6	29	7.1	30	21.5	6.2
ship04s	14	2.7	29	4.6	33	9.4	3.6
ship08l	16	26.0	38	22.3	36	72.1	25.9
ship08s	18	8.2	35	10.2	36	18.1	8.8
ship12l	17	50.6	44	41.1	47	114.0	50.6
ship12s	18	12.1	41	15.5	40	21.1	13.2
sierra	42	37.9	48	31.5	55	1989.4	13.2
standata	21	2.7	22	3.5	21	6.2	2.7
standgub	22	3.3	20	3.1	21	7.0	2.9
standmps	29	4.8	23	4.5	23	17.7	4.0
stocfor1	19	0.2	20	0.6	20	0.9	0.3
stocfor2	31	49.4	38	36.8	37	38.2	30.6
wood1p	24	67.5	35	135.1	35	207.0	31.0
woodw	30	465.4	80	359.5	75	7254.9	100.8
Promedio	25	71.7	30	111.9	31	427.9	148.7

[a] Memoria insuficiente. [b] Problemas de convergencia.

5. RESULTADOS COMPUTACIONES PARA LOS PROBLEMAS CUADRÁTICOS

En este apartado se presenta una comparación computacional de la implementación realizada del método presentado en los apartados anteriores, sea IPQ, y los sistemas LoQo y Minos 5.3 (Murtagh y Saunders (1983)). Dado que LoQo permite abordar los problemas cuadráticos tanto directamente en su forma original como a través de la transformación en equivalentes separables, se utilizan ambas alternativas en la comparación. De esta forma se observa si la eficiencia de IPQ respecto a LoQo se debe al hecho de transformar el problema en uno equivalente separable o a la utilización de las «ecuaciones normales en forma primal».

La tabla 2 recoge los resultados obtenidos con los 80 problemas de la colección Netlib presentados en la tabla 1, una vez transformados en problemas cuadráticos mediante el algoritmo descrito en el apartado anterior (usando, tal y como ya se ha indicado, los valores de $p_1 = 5$, $p_2 = 5$ y $s = 3141592$). La tabla recoge el número de iteraciones (niter) y tiempo de computación requerido (t), en segundos de CPU, para IPC y las alternativas de LoQo para el problema original y el problema equivalente separable (denotado por LoQo(sep)), y Minos. Para Minos únicamente se muestra el tiempo total de computación (no tiene sentido comparar su número de iteraciones con el de los métodos de punto interior). No se muestra el valor de función objetivo puesto que para los cuatro códigos se obtuvieron los mismos resultados. Las ejecuciones han sido realizadas sobre una estación de trabajo SunSparc 10/41 de 64 Mbytes de memoria (32 reales y 32 mapeadas en disco) y aproximadamente 10 Mflops. Cuando la ejecución no ha podido ser realizada se indica el motivo: $^{(a)}$ indica memoria insuficiente, y $^{(b)}$ problemas de convergencia.

Teniendo en cuenta los resultados recogidos en la tabla 2 puede observarse, en primer lugar, que Minos es el sistema más robusto, solucionando 79 de los 80 casos posibles. IPQ resuelve 76 casos. LoQo no pudo ejecutar cinco casos y LoQo(sep) tampoco pudo ejecutar cinco casos.

En lo que se refiere al rendimiento (sólo se tendrán en cuenta los 68 problemas que han podido ser solucionados por los cuatro códigos), se observa como IPQ ha sido el más eficiente en 44 problemas, LoQo en 9 y Minos en 15. LoQo(sep) no tuvo un mejor rendimiento que los otros sistemas en ningún caso. La diferencia de rendimiento entre IPQ y los otros sistemas es especialmente significativa para los problemas de grandes dimensiones (como «pilot87», donde IPQ es mucho más eficiente que LoQo y Minos). También puede observarse cómo en algunos casos las técnicas clásicas de optimización no lineal de gradiente reducido en que se basa Minos son más eficientes que las de punto interior (como, por ejemplo, en los problemas «wood1p» y «woodw»). Hay que tener en cuenta, sin embargo, que IPQ no incluye ningún tratamiento específico para columnas densas (en la versión actual) lo cual repercute negativamente en su

rendimiento (este hecho explica la gran diferencia de rendimiento entre IPQ y los otros sistemas en los problemas fit1p y seba). Añadiendo un tratamiento para este tipo de columnas su eficiencia se vería claramente aumentada.

Otro de los aspectos a destacar es el comportamiento de LoQo y LoQo(sep). En general, LoQo es más eficiente solucionando el problema en su formulación original que una vez transformado. Estos resultados difieren de los obtenidos y presentados en Vanderbei y Carpenter (1993), probablemente debido a la forma en que se construyeron los problemas cuadráticos. Sería necesario, pues, un estudio mucho más amplio con un mayor número de problemas tests (reales y generados) para poder concluir si es mejor transformar el problema o solucionarlo en su forma original. En el método descrito en este trabajo e implementado en IPQ, sin embargo, no hay ninguna duda, puesto que la transformación es prácticamente la única alternativa computacionalmente efectiva.

La última fila de la tabla recoge los promedios de número de iteraciones y tiempo de computación (sólo se han tenido en cuenta los problemas que se resolvieron por los cuatro códigos). Se puede observar que IPQ es el más eficiente. Sin embargo, esta observación está muy influenciada por los resultados del problema «pilot87», y porque algunos de los problemas más costosos («d2q06», «d6cube», «pilot») no han sido contemplados (no fueron solucionados por los cuatro sistemas).

6. CONCLUSIONES

Basados en la experiencia computacional bastante fuerte cuyos resultados principales se han obtenido en el apartado anterior, el método presentado en este trabajo se ha mostrado como una alternativa eficiente al sistema LoQo para la solución de problemas cuadráticos. La diferencia entre ambos enfoques es el hecho de solucionar un sistema simétrico e indefinido («sistema aumentado» en LoQo) o la solución de un sistema simétrico y definido positivo («ecuaciones normales en forma primal» en IPQ). En este último caso se ha mostrado cómo es necesario transformar el problema en un problema equivalente separable. Sin embargo, IPQ es más eficiente en el problema transformado que lo es LoQo en el problema original. A la vista de los resultados obtenidos, también se observa que las técnicas de punto interior para problemas cuadráticos son, en general, una alternativa más eficiente a las técnicas clásicas de programación no lineal de gradiente reducido (sistema Minos entre otros).

7. REFERENCIAS

[1] **Arbel, A.** (1993). *Exploring Interior-Point Linear Programming. Algorithms and Software*. The MIT Press, Cambridge, Massachusetts.

[2] **Barnes, E.R.** (1986). «A variation on Karmarkar's algorithm for solving linear programming problems». *Mathematical Programming*, **36**, 174–182.

[3] **Calamai, P.H, Vicente, L.N.** and **Júdice, J.J.** (1993). «A new technique for generating quadratic programming test problems», *Mathematical Programming*, **61(2)**, 215–231.

[4] **Castro, J.** (1995). «Implementació d'un algorisme primal-dual de punt interior amb fites superiors a les variables», *Qüestiió*, **19, 1, 2, 3,** 233–257.

[5] **Castro, J.** (1998). «Implementación de un algoritmo primal-dual de orden superior mediante el uso de un método predictor-corrector para programación lineal», *Qüestiió*, **22, 1**, 107–120.

[6] **Duff, I.S., A.M. Erisman** y **J.K. Reid.** (1986). *Direct Methods for Sparse Matrices*. Oxford University Press, New York.

[7] **Gay, D.M.** (1985). «Electronic mail distribution of linear programming test problems». *Mathematical Programming Society COAL Newsletter*, **13**, 10–12.

[8] **George, J.A.** y **J.W.H. Liu** (1981). *Computer Solution of Large Sparse Positive Definite Systems*. Prentice-Hall, Englewood Cliffs, NJ, USA.

[9] **Gill, P.E., W. Murray** y **M.H. Wright** (1981). *Practical Optimization*. Academic Press, London, UK.

[10] **Karmarkar, N.K.** (1984). «A new polynomial time algorithm for linear programming». *Combinatorica*, **4**, 373–395.

[11] **Khachiyan, G.** (1979). «A polynomial algorithm in linear programming». *Doklady Akademii Nauk SSSR*, **244(S)**, 1093–1096, traduït en *Soviet Mathematics Doklady*, **20(1)**, 191–194.

[12] **Kranich, E.** (1991). «Interior point methods for mathematical programming: a bibliography». *Diskussionbeitrag*, Nr. **171**, Dept. of Mathematics, Universität Wuppertal, Germany.

[13] **Lustig, I.J., R.E Marsten** and **D.F. Shanno** (1992). «On implementing Mehrotra's predictor-corrector interior-point method for linear programming», *SIAM Journal on Optimization*, **2(3)**, (1992), 435–449.

[14] **Mehrotra, S.** (1990). «On the implementation of a (primal-dual) interior point method», *Technical Report*, **90-03**. Dept. of Industrial Engineering and Management Science, Northwestern University, Evanston, IL.

[15] **Monma, C.L** y **A.J Morton** (1987). «Computational experience with a dual affine variant of Karmarkar's method for linear programming». *Operations Research Letters*, **6**, 261–267.

[16] **Monteiro, R.D.C** y **I. Adler** (1989). «Interior path following primal-dual algorithms. Part I: linear programming». *Mathematical Programming*, **44**, 27–41.

[17] **Monteiro, R.D.C.** and **Adler, I.** (1989). «Interior path following primal-dual algorithms. Part II: quadratic programming», *Mathematical Programming*, **44**, (1989) 43–66.

[18] **Murtagh, B.A.** y **M.A. Saunders** (1983). «MINOS 5.0. User's guide». Dept. of Operations Research, Stanford University, CA, USA.

[19] **Schrage, L.** (1979). «A More Portable FORTRAN Random Number Generator», *ACM Transactions on Mathematical Software*, June.

[20] **Terlaky, T.** (1996). (ed.) «Interior Point Methods of Mathematical Programming», Kluwer Academic Publishers, The Netherlands.

[21] **Vanderbei, R.J.** (1992). «LOQO User's Manual». Princeton University, Princeton, NJ, USA.

[22] **Vanderbei, R.J.** (1994). «An interior point code for quadratic programming». Princeton University, Princeton, NJ, USA.

[23] **Vanderbei, R.J.** (1996). *Linear Programming: Foundations and Extensions*. Kluwer Academic Publishers, Boston.

[24] **Vanderbei, R.J.** y **T.J. Carpenter** (1993). «Symmetric indefinite systems for interior point methods». *Mathematical Programming*, **58**, 1-32.

[25] **Vanderbei, R.J., M.S. Meketon** y **B.A. Freedman** (1986). «A modification of Karmarkar's linear programming algorithm». *Algorithmica*, **1**, 395–407.

[26] **Wright, M.H.** (1991). «Interior methods for constrained optimization». *Acta Numerica*, 341–407.

ENGLISH SUMMARY

AN INTERIOR-POINT ALGORITHM FOR QUADRATIC PROGRAMMING THROUGH SEPARABLE EQUIVALENT PROBLEMS*

J. CASTRO*

Universitat Rovira i Virgili

This paper presents an interior point algorithm for the solution of symmetric and positive definite quadratic programming problems. Instead of solving the original problem, the algorithm transforms it into an equivalent separable one, thus having a diagonal quadratic coefficients matrix. The main differences between this algorithm and others, like that implemented in the LoQo package, is that it solves the «normal equations in primal form» instead of the «augmented system», and that it does not require an explicit treatment for the quadratic free variables, i.e. those created when obtaining the separable equivalent problem. This algorithm is implemented in the IPQ package, and its efficiency is compared with that of the LoQo and Minos 5.3 packages. The comparison is performed through the solution of 80 problems of the Netlib collection (a standard suite for linear programming). The quadratic problems are generated from the linear ones through an ad-hoc generator for quadratic programming. Though it has been stated in the literature that solving the «augmented system» can be more efficient that solving the «normal equations in primal form» the computational results presented show that IPQ is competitive against LoQo and Minos 5.3, specially in some of the largest instances.

Keywords: Interior point methods, predictor-corrector method, primal-dual algorithm, quadratic programming.

⋆ This work has been supported by Iberdrola grant 95-005 and by CICYT project TAP96-1044-J02-93.

* J. Castro. Estadística i Investigació Operativa. Dept. d'Enginyeria Química. Universitat Rovira i Virgili. Autovia de Salou, s/n. 43006 Tarragona.

– Received November 1996.
– Accepted July 1997.

This paper presents a variation of the primal-dual algorithm for quadratic problems. The main features of the algorithm are as follows:

i) it considers a partition of the set of variables depending on they are just lower bounded, lower and upper bounded, and free variables (free variables are only allowed to be quadratic ones), leading us to the solution of the following linear programming problem

$$(P) \qquad \begin{array}{ll} \min_x & c'x + \frac{1}{2}x'Qx \\ \text{subject to} & A_u x_u + A_l x_l + A_f x_f = b \\ & \underline{0} \leq x_u \leq \overline{x}_u \\ & \underline{0} \leq x_l \\ & x_f \text{ libre} \end{array}$$

ii) it is not required an explicit treatment for the free quadratic variables.

iii) the algorithm solves the commonly called «normal equations in primal form», instead of the «augmented system».

In this last point the algorithm presented differs substantially from other quadratic interior point solvers, like LoQo, which solves the «augmented system». Though it has been stated in the literature that solving the indefinite symmetric augmented system can be more efficient that solving the positive definite primal normal equations, the computational results presented in the paper show that an accurate implementation based on the last technique can outperform a solver based on the augmented system.

The algorithm presented solves at each iteration the following «normal equations system»

$$(A\Theta^{-1}A')dy = \overline{b}$$

where matrix Θ is defined as follows

$$\Theta = (Q + \tilde{Z}X^{-1} + \tilde{F}^{-1}\tilde{W}) = \begin{pmatrix} Q_u + Z_u X_u^{-1} + F^{-1}W & Q'_{ul} & Q'_{uf} \\ Q_{ul} & Q_l + Z_l X_l^{-1} & Q'_{lf} \\ Q_{uf} & Q_{lf} & Q_f \end{pmatrix}$$

Q being the quadratic coefficients matrix. The computational cost of the above system is prohibitive for non-diagonal Q matrices. Therefore, instead of attempting the solution of the original problem, the algorithm transforms it to a separable equivalent one with a diagonal $\tilde{Q}$ matrix. This increases the number of variables and constraints, but reduces considerably the cost of the «normal equations in primal form» system to be solved at each iteration of the algorithm.

The structure of the paper is as follows. The first section of the document presents a brief overview of the state of the art, and outlines the main differences between

the algorithm presented and that implemented in the LoQo package (as stated in the former paragraph).

The second section fully details the algorithm presented for solving quadratic programming problems. The process of reduction of the Karush-Kuhn-Tucker necessary optimality conditions to the positive definite primal normal equations is showed, and the main characteristics of the system $(A\Theta A')dy = b_3 + A\Theta^{-1}r$ obtained are commented. It is also shown that the solution of the previous system is computationally prohibitive if the matrix Θ is not a diagonal one. This forces us to transform the original problem into another equivalent and separable, where the matrix Θ satisfies the property of being diagonal. This is presented in section three.

Last two sections focuses on the computational experiments. In section four the quadratic problems generator employed in the work is detailed. It creates a quadratic problem from a linear one, adding a positive definite matrix Q to the objective function. Section five reports the computational results obtained, comparing the implementation developed (named IPQ) with LoQo, and Minos5.3 over a battery test of 80 problems. Looking at the results, it can be concluded that IPQ is more efficient than the other codes, specially when the dimension of the quadratic problem tends to be large.