Universidad Nacional de Ingeniería

Facultad de Ciencias

ESTUDIO DEL ALGORITMO PROYECTIVO DE KARMARKAR Y SUS APLICACIONES EN LA INGENIERÍA

T E S I S

para obtener el grado de Maestro en Ciencias Mención Matemáticas Aplicadas

Autor:

JOSE FLORES SALINAS

Lima - Perú

2010

Dedicatoria:

Me es Grato Dedicar Este Trabajo a mi Familia muy en Especial al Chino Sami Flores

Agradecimientos:

Deseo aprovechar esta oportunidad para expresar mis más sinceros agradecimientos al profesor Willian Echegaray Castillo por su dedicación y paciencia en este trabajo de Tesis

Índice general

1

Algoritmo y Complejidad

El estudio de la complejidad de problemas relativos a algoritmos de programación lineal es uno de los objetivos de estudio del presente capítulo. En particular, se estudia la resolubilidad de problemas con algoritmos de complejidad polinomial. Para estudiar la complejidad de los problemas, el primer paso será describir que se entiende por conceptos tales como 'problema', 'tamaño', 'algoritmo', 'tiempo de ejecución', etc. Trabajaremos con $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}$ vector columna. En este Capítulo se demuestra que el método simplex, tiene convergencia exponencial en el peor de los casos, se estudia también la complejidad del método del elipsoide que al igual que el método de Karmarkar es un algorítmo de complejidad polinomial. Para el estudio de la complejidad del algoritmo del elipsoide se define el volumen de la envoltura convexa apartir de una medida, finalmente se expone brevemente como se aplica este método en problemas de programación lineal.

1.1. Letras, Tamaño, Palabras

Basicamente cuando se formaliza el problema de complejidad los objetos son simbolos y cadenas de simbolos.

Definición 1.1.1. *Sea $\sum$ un conjunto finito (a menudo $\sum = \{1, 0\}$). $\sum$ es llamado el alfabeto y sus elementos son llamados símbolos o letras.*

Definición 1.1.2. *Una sucesión finita y ordenada de simbolos de $\sum$ es llamada cadena (de simbolos) o palabra.*

Definición 1.1.3. *Denotemos por $\sum^*$ al conjunto de todas las cadenas de simbolos de $\sum$.*

Definición 1.1.4. *El tamaño de una cadena es el número de sus componentes.*

La cadena de tamaño 0 es la cadena vacia, denotada por Φ.

Las cadenas pueden tener la forma de sucesión finita de números racionales, vectores, matrices, grafos, sistemas de ecuaciones lineales o inecuaciones, etc. Hay varios métodos generales de transformación para codificar estos objetos de manera uniforme, como cadenas de símbolos de algún alfabeto prefijado con $\{1, 0\}$.
Dependiendo de estos métodos de transformación, esto induce el concepto de tamaño de dichos objetos. Los tamaños de un número racional $\alpha = \frac{p}{q}$ con p, q enteros primos entre sí, de un vector racional $\mathbf{c} = (\sigma_{\mathbf{1}}, \sigma_{\mathbf{2}}, ..., \sigma_{\mathbf{n}})$ y de una matriz racional $A = (a_{ij})_{m \times n}$ donde $i = 1...m, j = 1..n$; se definen como:

$$\begin{aligned} tam\,(\alpha) &= 1 + [\![\log(|p|) + 1]\!] + [\![\log(|q|) + 1]\!] \\ tam\,(c) &= n + tam\sigma_1 + ...tam\sigma_n \\ tam\,(A) &= mn + \sum_{i,j} tama_{ij} \end{aligned}$$

Nota: $[\![x]\!]$ denota el mayor número entero menor o igual que x.
El tamaño de una desigualdad lineal $ax \leqslant \beta$ ó una ecuación $ax = \beta$ es igual a $1 + tam\,(a) + tam\,(\beta)$. El tamaño de un sistema $Ax \leqslant b$ $(Ax = b)$ de inecuaciones lineales $1 + tam\,(A) + tam\,(b)$.

A menos que se indique lo contrario, *log* representa el logaritmo en base 2.

Definición 1.1.5. *Sean $f, g : \mathbb{N} \to \mathbb{R}$ funciones reales, decimos que $f(n) = O(g(n))$ si existe $c > 0$ y $n_0 \in \mathbb{N}$ tal que:*

$$| f(n) | \leqslant cg(n), \forall n \geqslant n_0.$$

La O que define esta expresión significa que f está acotada por g a partir de n_0.

1.2. Problemas

Definición 1.2.1. *Se define un problema como un subconjunto Π de $\sum^* \times \sum^*$ donde $\sum$ es algún alafabeto.*

Así un problema puede tener la forma de una interrogación o la forma de una tarea.

Ejemplo 1.2.1. *El problema matemático: "Dado $z \in \sum^*$, encontrar una $y \in \sum^*$ tal que $(z, y) \in \Pi$, o decidir que tal cadena y no existe".*

Aquí, la cadena z es llamada la entrada del problema, e y es la solución o salida.

Definición 1.2.2. *El problema* Π *es llamado problema de decisión o problema* **si/no** *si, para cada* $(z, y) \in \Pi$*, y es la cadena vacia* Φ*.*

En este caso, el problema es a menudo identificado como el conjunto χ de cadenas $z \in \sum^*$ para las cuales (z, Φ) pertenece a Π.

Un problema también puede indicarse de la siguiente forma:

"Dada una cadena $z \in \sum^*$, decidir si pertenece a χ", o "dado $z \in \sum^*$ ¿Pertenece a χ?".

Veamos algunos ejemplos de problemas:

Ejemplo 1.2.2. *Consideremos:*

1. $\{((A, \mathbf{b}), \Phi) / A$ *es una matriz de orden* $n \times n$*,* $\mathbf{b} \in \mathbb{R}^{\mathbf{n}}$*, tal que* $A\mathbf{x} \leqslant \mathbf{b}$ *para al menos un vector columna* $\mathbf{x} \in \mathbb{R}^{\mathbf{n}}\}$*.*
   Esto quiere decir:
   Dado un sistema $A\mathbf{x} \leqslant \mathbf{b}$ *de desigualdades lineales, ¿tiene solución?*

2. $\{((A, \mathbf{b}), \mathbf{x}) / A$ *es una matriz de orden* $n \times n$*,* $\mathbf{b}, \mathbf{x} \in \mathbb{R}^{\mathbf{n}}$*, tal que* $A\mathbf{x} \leqslant \mathbf{b}\}$*.*
   Esto quiere decir:
   Dado un sistema $A\mathbf{x} \leqslant \mathbf{b}$ *de desigualdades lineales, encontrar una solución si existe, caso contrario decir que no existe solución.*

1.3. Algoritmos y Tiempos de Ejecución

Definición 1.3.1. *Entenderemos por algoritmo a una lista de instrucciones que permiten resolver un Problema.*

Para una entrada $z \in \sum^*$, un algoritmo para el problema $\Pi \subseteq \sum^* \times \sum^*$ determina una salida "y" tal que $(z, y) \in \Pi$, o se detiene sin determinar ninguna salida si no existe tal "y".

Un algoritmo puede ser definido como una cadena finita, Ψ, de ceros y unos. Se dice que Ψ resuelve el problema Π o que es un algoritmo para Π, si para cualquier entrada z de Π, si introducimos la cadena (Ψ, z) en una máquina universal de turing; la máquina se detiene después de un número finito de pasos, proporcionando una cadena "y" con $(z, y) \in \Pi$ o no proporciona cadena alguna en caso de que no exista.

Hay varias formas de definir el concepto de tiempo de ejecución para indicar la cantidad de operaciones elementales de bit necesarias en la ejecución de un algoritmo por una computadora.

El tiempo de ejecución depende de la particular implementación del algoritmo.

Definición 1.3.2. *Sea Ψ un problema determinado con J un conjunto de índices, entonces Ψ_j es la familia de soluciones del problema Ψ, y $\Psi_j(n)$ la familia de soluciones de Ψ cuyo tamaño n, $T_j(n)$ el tiempo de ejecución del algoritmo para la solución $\Psi_j(n)$, se define como:*

$$T(n) = \max\{T_j(n), j \in J\}$$

Ejemplo 1.3.1. *Dado el polinomio $P(n) = 3n^2 + n + 4$ donde n es el tamaño del problema del algorítmo Ψ, el cual acota al tiempo de ejecución del algorítmo mencionado, decimos que el algorítimo Ψ queda polinomialmente acotado por $O(n^2)$ es decir $T(n) \leqslant cn^2$, donde $c \in \mathbb{R}$.*

Definición 1.3.3. *Un modelo en el cual el tiempo de ejecución de una instrucción de un algoritmo se considera constante se llama modelo aritmético.*

Definición 1.3.4. *El modelo bit es un modelo en el cual cada instrucción es descompuesta en un conjunto de instrucciones elementales que operan sobre números de un solo bit y se asume que el tiempo de ejecución cada instrucción elemental es una unidad de tiempo.*

1.4. Algoritmos Polinomiales

El tiempo de ejecución de un algoritmo dependerá, en general del problema al cual es aplicado. Sea $T(n)$ el peor caso de tiempo de ejecución de algún algoritmo sobre todos los problemas de tamaño n, de acuerdo al modelo bit. En otras palabras, sobre todos los problemas de tamaño n, consideremos un problema el cual toma el algoritmo mas grande.

Definición 1.4.1. *Un algoritmo se ejecuta en un tiempo polinomial si existe un entero k tal que $T(n) = O(n^k)$.*

En [3] en la sección 1.6 existe el ejemplo de la inversión de una matriz por eliminación gaussiana, en este caso el algoritmo converge polinomialmente tanto para el modelo bit como para el modelo aritmético, existen otros aspectos aún no estudiados como la elección del lenguaje de programación, la secuencia en la que se dan las instrucciones en estos lenguajes puede hacer que un algoritmo resulte polinomial para un lenguaje para otro no.

Las operaciones aritméticas elementales son: adición, sustracción, multiplicación, división y comparación de números. En aritmética racional, estas operaciones pueden ser ejecutadas por algoritmos polinomiales. Por tanto, para reducir la polinomialidad

de un algoritmo que incluya una secuencia de operciones aritméticas elementales es suficiente demostrar que el número total de estas operaciones está polinomialmente acotado por el tamaño de la entrada.

1.5. Tiempo de Ejecución del Algoritmo Simplex

Para probar que el método simplex con la regla de ejecución de pivote de Dantzig, se puede comportar como un algoritmo polinomial en muchos casos. Consideremos el siguiente problema propuesto por Klee y Minty en su artículo [10]:

$$
\begin{array}{lll}
\text{máx} & \{2^{n-1}x_1 + 2^{n-2}x_2 + ... + 2x_{n-1} + x_n\} & \\
\text{s.a.} & x_1 \leqslant 5 & \\
& 4x_1 + x_2 \leqslant 25 & \\
& 8x_1 + 4x_2 + x_3 \leqslant 125 & (LP_n) \\
& \vdots \quad \vdots \quad \vdots & \\
& 2^n x_1 + 2^{n-1}x_2 + 2^{n-2}x_3 + \cdots + 4x_{n-1} + x_n \leqslant 5^n & \\
& x_1, x_2, ..., x_n \geqslant 0 &
\end{array}
$$

Teorema 1.5.1. *Aplicando el método simplex al problema LP_n, con la regla de selección pivotal de Dantzig, se requiere la construcción de 2^n tablas.*

Demostración. Sean $y_1, y_2, ..., y_n$ las variables de holgura añadidas a LP_n. observamos que:

Para x_1 se verifica inmediatamente. El problema con $x_1, x_2, ..., x_{n-1}$ variables tiene como función objetivo

$$Obj_{n-1} = 2^{n-2}x_1 + 2^{n-3}x_2 + ... + 2x_{n-2} + x_{n-1}$$

y como valor máximo 5^{n-1}, se puede ver que

$$0 \leqslant Obj_{n-1} \leqslant 5^{n-1}.$$

La desigualdad

$$2^n x_1 + 2^{n-1}x_2 + 2^{n-2}x_3 + \cdots + 4x_{n-1} + x_n \leqslant 5^n$$

se introduce para el problema con $n - variables$ entonces

$$
\begin{aligned}
2^2\left(2^{n-2}x_1 + \cdots + 2x_{n-2} + x_{n-1}\right) + x_n &\leqslant 5^n \\
2^2\left(Obj_{n-1}\right) + x_n &\leqslant 5^n \\
x_n &\leqslant 5^n - 2^2\left(Obj_{n-1}\right)
\end{aligned}
$$

la función objetivo será

$$Obj_n = 2^{n-1}x_1 + 2^{n-2}x_2 + ... + 2x_{n-1} + x_n,$$

así

$$\begin{array}{lll} Obj_n = 2\left(Obj_{n-1}\right) + x_n & \leqslant & 2\left(Obj_{n-1}\right) + 5^n - 2^2\left(Obj_{n-1}\right) \\ Obj_n \leqslant 5^n - 2\left(Obj_{n-1}\right) & & \end{array}$$

Entonces el mayor valor de $Obj_n = 5^n$ de la última desiguldad del problema de n variables si $x_n = 5^n$, se cumple que $x_1 = x_2 = ... = x_{n-1} = 0$ además esta solución es única, en efecto supongamos que existe otra solución entonces:
$2^2\left(Obj_{n-1}\right) + x_n \leqslant 5^n$ esto es $Obj_n + 2Obj_{n-1} \leqslant 5^n$ por lo tanto $Obj_n \leqslant 5^n - 2Obj_{n-1}$ si $x_1, x_2, ..., x_{n-1}$ tienen valores diferentes de cero entonces $Obj_n < 5^n$ y esto es absurdo.

Ahora En el sistema de inecuaciones introduzcamos las variables de holgura $y_1, y_2, ..., y_{n-1}, y_n$ es decir:

$$\begin{array}{lcl} x_1 + y_1 & = & 5 \\ 4x_1 + x_2 + y_2 & = & 25 \\ 8x_1 + 4x_2 + x_3 + y_3 & = & 125 \\ \vdots & \vdots & \vdots \\ 2^n x_1 + 2^{n-1}x_2 + \cdots + 4x_{n-1} + x_n + y_n & = & 5^n \end{array}$$

en la última igualdad si $x_n = y_n = 0$ entonces

$$2^2\left(2^{n-2}x_1 + 2^{n-3}x_2 + \cdots + 2x_{n-2} + x_{n-1}\right) = 5^n .....(*)$$

pero Obj_{n-1} es como máximo igual a 5^{n-1} y de (*) Obj_{n-1} resulta que es $\frac{5^n}{4} > 5^{n-1}$, por lo tanto las dos variables son diferentes de cero o una es diferente de cero y la otra es cero. Como hay n variables básicas supongamos que x_n y y_n son básicas luego quedan $n-2$ variables básicas y $n-1$ pares (x_i, y_i) al menos un par sería $(0,0)$, lo cual no puede ser, entonces si x_i es básica y_i no es básica y viciversa.

De manera ilustrativa resolvamos el sistema con el método simplex

$$\begin{array}{lllll} \text{máx} & \{2^2x_1 + 2x_2 + x_3\} & & & \\ \text{s.a.} & x_1 & \leqslant & 5 & \\ & 4x_1 + x_2 & \leqslant & 25 & (1.1) \\ & 2^3x_1 + 4x_2 + x_3 & \leqslant & 5^3 & \\ & x_1, x_2, x_3 & \geqslant & 0 & \end{array}$$

se obtienen las siguientes 2^3 tablas

z	x_1	x_2	x_3	y_1	y_2	y_3	
	-2^2	-2	-1	0	0	0	0
y_1	1	0	0	1	0	0	5
y_2	4	1	0	0	1	0	5^2
y_3	8	4	1	0	0	1	5^3

Cuadro 1.1: Tabla del Simplex

z	x_1	x_2	x_3	y_1	y_2	y_3	
	-2^2	-2	-1	0	0	0	0
y_1	**1**	0	0	1	0	0	5
y_2	4	1	0	0	1	0	5^2
y_3	8	4	1	0	0	1	5^3

Cuadro 1.2: Tabla 1 del Simplex

z	x_1	x_2	x_3	y_1	y_2	y_3	
	0	-2	-1	4	0	0	20
x_1	1	0	0	1	0	0	5
y_2	0	**1**	0	-4	1	0	5
y_3	0	4	1	-8	0	1	85

Cuadro 1.3: Tabla 2 del Simplex

z	x_1	x_2	x_3	y_1	y_2	y_3	
	0	0	-1	-4	2	0	30
x_1	1	0	0	**1**	0	0	5
x_2	0	1	0	-4	1	0	5
y_3	0	0	1	8	-4	1	65

Cuadro 1.4: Tabla 3 del Simplex

z	x_1	x_2	x_3	y_1	y_2	y_3	
	4	0	-1	0	2	0	50
y_1	1	0	0	1	0	0	5
x_2	4	1	0	0	1	0	25
y_3	-8	0	**1**	0	-4	1	25

Cuadro 1.5: Tabla 4 del Simplex

z	x_1	x_2	x_3	y_1	y_2	y_3	
	-4	0	0	0	-2	1	75
y_1	**1**	0	0	1	0	0	5
x_2	4	1	0	0	1	0	25
x_3	-8	0	1	0	-4	1	25

Cuadro 1.6: Tabla 5 del Simplex

z	x_1	x_2	x_3	y_1	y_2	y_3	
	0	0	0	4	-2	1	95
x_1	1	0	0	1	0	0	5
x_2	0	1	0	-4	**1**	0	5
x_3	0	0	1	8	-4	1	65

Cuadro 1.7: Tabla 6 del Simplex

z	x_1	x_2	x_3	y_1	y_2	y_3	
	0	2	0	-4	0	1	105
x_1	1	0	0	**1**	0	0	5
y_2	0	1	0	-4	1	0	5
x_3	0	4	1	-8	0	1	85

Cuadro 1.8: Tabla 7 del Simplex

z	x_1	x_2	x_3	y_1	y_2	y_3	
	4	2	0	0	0	1	125
y_1	1	0	0	1	0	0	5
y_2	4	1	0	0	1	0	25
x_3	8	4	1	0	0	1	5^3

Cuadro 1.9: Tabla 8 del Simplex

De estas tablas se pueden tener las siguientes hipótesis inductivas, que la primera fila de las tablas tiene coeficientes enteros y las tablas 1 y 2^n para el problema LP_n

son:

$-c^t$	-1	$0_{1\times n-1}$	0	0
$A_{n-1\times n-1}$	$0_{n-1\times 1}$	$I_{n-1\times n-1}$	$0_{n-1\times 1}$	$b_{n-1\times 1}$
$2c^t$	1	$0_{1\times n-1}$	1	5^n

Cuadro 1.10: Tabla 1 del Simplex

c^t	0	$0_{1\times n-1}$	1	5^n
$A_{n-1\times n-1}$	$0_{n-1\times 1}$	$I_{n-1\times n-1}$	$0_{n-1\times 1}$	$b_{n-1\times 1}$
$2c^t$	1	$0_{1\times n-1}$	1	5^n

Cuadro 1.11: Tabla 2^n del Simplex

donde $c^t = (2^{n-1}\ 2^{n-1}\ \ldots\ 2^2\ 2)$, Si construimos el problema LP_{n+1} en tablas tenemos

		x_{n+1}			y_{n+1}	
$-2c^t$	-2	-1	$0_{1\times n-1}$	0	0	0
$A_{n-1\times n-1}$	$0_{n-1\times 1}$	$0_{n-1\times 1}$	$I_{n-1\times n-1}$	$0_{n-1\times 1}$	$0_{n-1\times 1}$	$b_{n-1\times 1}$
$2c^t$	1	0	$0_{1\times n-1}$	1	0	5^n
$4c^t$	4	1	$0_{1\times n-1}$	0	1	5^{n+1}

Cuadro 1.12: Tabla del Problema LP_{n+1}

Se puede ver que las columnas $n+1$ y $2n+2$ no son pivote para las 2^n-1 primeras tablas porque los coeficientes de la primera fila son enteros y como mínimo -2, y la última fila tampoco puede ser pivote ya que si esto ocurriera x_{n+1} y y_{n+1} serían diferente de cero ambas, además si se retiran las columnas $n+1$ y $2n+2$ con la última fila el problema tiene la misma estructura que LP_n entonces el problema se resuelve como LP_n hasta la tabla 2^n. Por la similaridad a LP_n luego de 2^n-1 tablas

$2c^t$	0	-1	$0_{1\times n-1}$	2	0	$2 * 5^n$
$A_{n-1\times n-1}$	$0_{n-1\times 1}$	$0_{n-1\times 1}$	$I_{n-1\times n-1}$	$0_{n-1\times 1}$	$0_{n-1\times 1}$	$b_{n-1\times 1}$
$2c^t$	1	0	$0_{1\times n-1}$	1	0	5^n
$-4c^t$	0	$\mathbf{1}$	$0_{1\times n-1}$	-4	1	5^n

Cuadro 1.13: Tabla 2^n del Simplex de LP_{n+1}

Se observa que el valor de la solución de la función objetivo es el doble ya que

los coeficientes de la primera fila son el doble, de igual forma el pivote es el indicado en negrita, entonces observando la tabla $2^n + 1$ del cuadro (1.14)

$-2c^t$	0	0	$0_{1\times n-1}$	-2	1	$3 * 5^n$
$A_{n-1\times n-1}$	$0_{n-1\times 1}$	$0_{n-1\times 1}$	$I_{n-1\times n-1}$	$0_{n-1\times 1}$	$0_{n-1\times 1}$	$b_{n-1\times 1}$
$2c^t$	1	0	$0_{1\times n-1}$	1	0	5^n
$-4c^t$	0	1	$0_{1\times n-1}$	-4	1	5^n

Cuadro 1.14: Tabla $2^n + 1$ del Simplex de LP_{n+1}

Por las mismas razones del caso anterior es decir, se puede ver que las columnas $n+1$ y $2n+2$ no son pivote para las siguientes $2^n - 1$ tablas, porque los coeficientes de la primera fila son enteros y como mínimo -2, entonces el problema tiene la misma estructura de LP_n solo que y_n está en vez de x_n y y_{n+1} está en vez de x_{n+1}, en $2^n - 1$ pasos llegamos a la última tabla entonces $2^n + 1 + 2^n - 1 = 2^{n+1}$ pasos, se observa que el simplex es exponencial en este caso. □

1.6. Tiempo de ejecución medio del Algoritmo Simplex

Recientemente, Borgwardt [1] ha demostrado que la media del tiempo ejecución del algoritmo del simplex está polinomialmente acotada por el tamaño del problema, en cierto modelo probabilistico natural usando determinada regla de selección del pivote. Borwardt demostro que el número medio de pasos al resolver

$$\text{máx}\left\{c^t\mathbf{x}/\ \mathbf{A}\mathbf{x} \leqslant \mathbf{b}\right\} \tag{I}$$

es $O\left(n^3 m^{\frac{1}{n-1}}\right)$, Borgwart [1] supone que $\mathbf{b}$ es positivo. (m y n denotan el número de filas y columnas, respectivamente, de A).

Usando un modelo probabilistico diferente y con otra regla de selección de pivote, Smale [16], independientemente, ha demostrado para un m fijo, la siguiente cota superior: $O\left((\log n)^{m(m+1)}\right)$, al resolver el problema

$$\text{mín}\left\{c^t\mathbf{x}/\ \mathbf{x} \geqslant \mathbf{0};\ \mathbf{A}\mathbf{x} \leqslant \mathbf{b}\right\} \tag{II}$$

La cota de Smale no es de tipo polinomial, pero puede ser mejor que la de Borgwardt (si m es fijo y $n \to \infty$).

Heimovich [6], combinando la regla de selección de pivote dé Borgwardt [1] con una versión más general del modelo probabilistico de Smale, ha demostrado que el número medio de pasos es, de hecho, lineal y puede ser acotado por

$$\begin{array}{ll} \text{mín}\left\{\frac{n}{2}, \frac{m-n+1}{2}, \frac{m+1}{8}\right\} & \text{para problemas de tipo (I)} \\ \text{mín}\left\{\frac{n}{2}, \frac{m+1}{2}, \frac{n+m+1}{8}\right\} & \text{para problemas de tipo (II)} \end{array}$$

Para obtener estas cotas, no se ha supuesto $b > 0$.

Lo expuesto anteriormente coincide con la experiencia práctica observada al aplicar el método del simplex: El tiempo medio de resolución de problemas de programación lineal por el algoritmo del simplex es lineal en el número de variables. A pesar de estos buenos resultados "medios", los investigadores han intentado encontrar métodos de resolución que trabajaran en tiempo polinomial.

1.7. El Algoritmo de la Elipsoide

En el año 1979, el matemático soviético L.G. Khachiyan [9] publicó una demostración, de que cierto algoritmo para resolver problemas de programación lineal es polinomial, resolviendo asi, al menos teóricamente, una cuestión abierta desde bastante tiempo antes. El resultado de Khachian [9] se basa en los trabajos de otros matemáticos soviéticos sobre programación no lineal, en particular de Shor, Yudin y Nemirovskii [12].

Dicho algoritmo es conocido como algoritmo del elipsoide, el algoritmo es iterativo y, en principio, se aplica a un problema del siguiente tipo : Dado un conjunto de desigualdades lineales del tipo $Ax < b$, ¿tiene una solución?. Entonces, en cada iteración se construye un elipsoide que contiene siempre una solución del problema anterior (si la hay) . El elipsoide construido en cada iteración es siempre."más pequeño que el anterior, de manera que, después de un determinado número de iteraciones se encuentra una solución o se concluye que tal solución no existe. A continuación exponemos el algoritmo y una serie de resultados previos, que nos permitirán probar la convergencia polinomial del mismo.

Consideremos pues el siguiente sistema de desigualdades lineales estrictas con coeficientes enteros:

$$a_i^t \mathbf{x} < b_i \ (i = 1, 2, ..., m;\ a_i \in \mathbb{Z}^n,\ x \in \mathbb{R}^n,\ b_i \in \mathbb{Z}) \tag{1}$$

Las condiciones impuestas a los coeficientes c_i y b_i no son restrictivas puesto que si una ecuación tiene los coeficientes en $\mathbb{Q} \setminus \mathbb{Z}$, reducimos al caso anterior multiplicando

por el *m.c.m.*, y si fuese irracional, estos son truncados cuando se lleva el problema al computador, por lo que son tratados como racionales sea:

$$A\mathbf{x} < b,\ (A = [a_{ij}]_{m\times n}, x \in \mathbb{R}^n,\ b \in \mathbb{R}^m) \tag{2}$$

Se Define el tamaño del problema (2) de la siguiente forma:

$$L = [\![\sum_{\substack{ij \\ a_{ij}\neq 0}} (\log|a_{ij}| + 1)]\!] + [\![\sum_{\substack{i \\ b_i\neq 0}} (\log|b_i| + 1)]\!] + [\![\log(mn) + 1]\!]$$

1.7.1. El Algoritmo

Definimos una sucesión $x_0, x_1, ..., \in \mathbb{R}^n$ y una sucesión de matrices simétricas definidas positivas de orden n: $A_0, A_1, ...$ recursivamente

$$\begin{aligned} x_{k+1} &= x_k - \frac{1}{n+1}\cdot\frac{A_k a_{i_k}}{\sqrt{a_{i_k}^t A_k a_{i_k}}} \\ A_{k+1} &= \frac{n^2}{n^2-1}\left(A_k - \frac{2}{n+1}\cdot\frac{\left(A_k a_{i_k}\right)\left(A_k a_{i_k}\right)^t}{a_{i_k}^t A_k a_{i_k}}\right) \end{aligned}$$

con: $x_0 = 0, A_0 = 2^{2L}I$, $a_{i_k} \in \mathbb{R}^n, b_{i_k} \in \mathbb{R}$. Se comprueba si x_k es una solución de (1), si esta se cumple se detiene el proceso. Si no, se toma una cualquiera de las desigualdades de (1) que no se cumple

$$a_{i_k}^t x_k \geqslant b_{i_k}$$

como veremos más adelante.

Notar que el producto del vector $A_k a_i$ por su transpuesta $(A_k a_i)^t$ en la segunda igualdad da como resultado una matriz $n \times n$.

El resultado fundamental viene dado por el siguiente teorema.

Teorema 1.7.1. *Si el algoritmo se detiene, x_k es una solución de (1) Si el algoritmo no se detiene en $4(n+1)^2L$ pasos, entonces (1) no tiene solución.*

Demostración. veremos más adelante en la página (18). □

Nota 1. *Este teorema asegura la polinomialidad del algoritmo, pues en cada paso hay que realizar un número de operaciones elementales de orden polinomial.*

La primera afirmación del teorema es, por supuesto, una repetición de la regla de parada del algoritmo antes descrito. Para probar la segunda parte, definiremos previamente algunos conceptos, y daremos varios lemas.

Definición 1.7.1. *Dado $x_0 \in \mathbb{R}^n$ y una matriz $D_{n\times n}$ simétrica y definida positiva. Entonces:*

$$(x - x_0)^t D^{-1} (x - x_0) \leqslant 1$$

define un elipsoide $E(x_0, D)$ con centro x_0. Los vectores $x \in \mathbb{R}^n$ para los cuales se verifica la igualdad, definen la frontera de dicha elipsoide.

Lema 1.7.1. *Las matrices $A_0, ..., A_k, ...$ de orden $n \times n$ son simétricas y definidas positivas.*

Demostración. La simetria es evidente. Supongamos que A_k es definida positiva luego define un elipsoide $E(x_k, A_k)$. Mediante una transformación afín adecuada, podemos transformar este elipsoide en la esfera unidad y el vector a_{i_k} en $(\alpha, 0, 0, ..., 0)$ con $\alpha > 0$. De esta forma se obtiene:

$$\begin{aligned} x_{k+1} &= \left(\frac{1}{n+1}, 0, ..., 0\right) \\ A_{k+1} &= \frac{n^2}{n^2-1} diag\left(\frac{n-1}{n+1}, 1, ..., 1\right) \end{aligned}$$

Que es definida positiva. Así pues $E(x_k, A_k)$ define una sucesión de elipsoides. □

Definición 1.7.2. *Al elipsoide $E(x_k, A_k)$ lo denotaremos como E_k.*

Lema 1.7.2. *Se verifica que $2^L \geqslant \prod_{i=1}^{m}\prod_{j=1}^{n}(|a_{ij}| + 1)\prod_{i=1}^{m}(|b_i| + 1)(mn + 1)$ donde L es el tamaño según (2).*

Demostración. Basta probar que que si $n \in \mathbb{N} \setminus \{0\}$ entonces:
$2^{\log n+1} \geqslant (n + 1)$ o equivalentemente $\log(n+1) \leqslant \log n+1$ Consideremos el intervalo $I_p = [2^p, 2^{p+1} - 1]$ donde $p \in \mathbb{N}$. Si $p \in \mathbb{N}$ y $n \in I_p \cap \mathbb{N} \setminus \{0\}$, por ser $\log x$ una función monótona $\log(n + 1) \leqslant \log(2^{p+1} - 1 + 1) = p + 1$ y por otra parte:
$\log n + 1 \geqslant \log 2^p + 1 = p + 1$. □

Lema 1.7.3. *Si M es el valor de un determinante cuyos elementos son a_{ij} de la matriz $K = [a_{ij}]_{n\times n}$ que define el vértice de un poliedro, se verifica que:*
$M \leqslant \prod_i \prod_j (|a_{ij}| + 1)$.

Lema 1.7.4. *Sea $v = (v^1, ..., v^n)^t$ un vértice cualquiera del poliedro*

$$\Gamma = \left\{x \in \mathbb{R}^n / \, a_i^t x \leqslant b_i, x \geqslant 0, a_i \in \mathbb{R}^n, b_i \in \mathbb{R},\ i = 1, ..., m\right\}$$

se verifican:

a). Las coordenadas de v son números racionales de denominador menor que $\dfrac{2^L}{n}$.

b). $\|v\| < 2^L$. ($\|\|$ *denota la norma* 2)

Demostración.

a). Por ser v un vértice, sus coordenadas serán solución de un sistema de ecuaciones de orden n que con la regla de Cramer serán de la forma $v^i = \dfrac{M^i}{M}$ siendo M^i y M determinantes cuyos elementos son coeficientes de las desigualdades que definen al poliedro Γ, es decir ,$0, 1, a_{ij}, b_i$. Tendremos pues que M y M^i son enteros y además, por los lemas anteriores:

$$M \leqslant \prod_i \prod_j (|a_{ij}| + 1) \prod_i (|b_i| + 1) < \frac{2^L}{n}.$$

b). Por ser M entero no nulo, $|M| \geqslant 1$. Además, analogicamente al apartado anterior $M_i \leqslant \dfrac{2^L}{n}, \forall i$ Luego

$$v^i = \frac{M^i}{M} < \frac{2^L}{n}, \forall i \Rightarrow \|v\| = \sqrt{\sum (v^i)^2} < \sqrt{n \left(\frac{2^L}{n}\right)^2} \leqslant \frac{2^L}{\sqrt{n}} < 2^L.$$

□

En principio, vamos a suponer que el poliedro definido por (1) está acotado.

Definición 1.7.3. *Definimos el volumen de la envoltura convexa en* $\mathbb{R}^n$ *como*

$$Vol_n = abs \left| \sum_{k=0}^{n} (-1)^k \int_0^{y_k} Vol_{n-1,k} \frac{y^{n-1}}{y_k^{n-1}} dy \right|$$

donde y_k *es la última nupla del punto* (u_k^t, y_k)

$$Vol_{n-1,k} = \frac{1}{(n-1)!} \det \begin{pmatrix} 1 & \cdots & 1 & 1 & \cdots & 1 \\ u_0 & \cdots & u_{k-1} & u_{k+1} & \cdots & u_n \end{pmatrix} \qquad y_j \in \mathbb{R}, u_j \in \mathbb{R}^{n-1}$$

de esta manera

$$Vol_n = \frac{1}{n!} \left| \det \begin{pmatrix} 1 & \cdots & 1 & 1 & \cdots & 1 \\ u_0 & \cdots & u_{k-1} & u_k & \cdots & u_n \\ y_0 & \cdots & y_{k-1} & y_k & \cdots & y_n \end{pmatrix} \right|.$$

Lema 1.7.5. *Sea* $Q = \left\{ x \in \mathbb{R}^n / \, a_i^t x < b_i,\ a_i \in \mathbb{R}^n,\ \forall i \in \mathbb{N},\ \|x\| < 2^L \right\}$ *se verifican:*

a). $Q \subset E_0$.

b). $\forall\, k\, \exists\, i$ *tal que* $Q \subset \{x \in \mathbb{R}^n / -a_i^t(x - x_k) > 0,\ a_i, x_k \in \mathbb{R}^n\}$.

c). *Si* $Q \neq \Phi$, *entonces* $Vol(Q) > 2^{-(n+1)L}$.

Demostración.

a). Por definición, E_0 es la esfera centrada en el origen y de radio 2^L
$E_0 = E(x_0, A_0) = \{x \in \mathbb{R}^n / (x - x_0)^t A_0^{-1}(x - x_0) \leqslant 1,\ x_0 \in \mathbb{R}^n\}$ donde A_0 es una matriz simétrica de orden $n \times n$ según la definición.

b). Por la definición del algoritmo, x_k no verifica la restricción i_k, es decir,

$$\begin{array}{rcl} a_{i_k}^t x_k & \geqslant & b_{i_k} \\ \text{si } x \in Q & \Rightarrow & a_{i_k}^t x < b_i \end{array}$$

y combinando las dos desigualdades queda:

$$-a_{i_k}^t (x - x_k) > 0.$$

c). Si $Q \neq \Phi$, el poliedro definido por las restricciones (abierto) ha de contener necesariamente un punto interior. Se verifica que dicho poliedro tiene $n+1$ vértices linealmente independientes. En efecto, si todos los conjuntos de $n+1$ vértices fueran linealmente dependientes el poliedro estaría contenido en un hiperplano de dimensión menor que n. Sea x un punto interior y sea ε la mínima distancia de x.a las caras del poliedro .Evidentemente $\varepsilon > 0$ y la esfera de centro x y radio ε está contenida en el poliedro lo cual es absurdo.

Sean pues $v_0, v_1..., v_n$, $n+1$ vértices linealmente independientes. Denotemos por H la envoltura convexa definida por dichos vértices. Si x es un punto interior de H, por una parte, verifica $a_i^t x < b_i, \forall i$. Y además, por ser: $\|v\| < 2^L, \forall i$ (lema 1.7.4) y por ser x combinación convexa de $v_0, v_1..., v_n$ se tiene

$$\|x\| < 2^L$$

concluimos que $Int(H) \subset Q \Rightarrow Vol(Q) > Vol(H)$, por otra parte de la definición 1.7.3

$$Vol(H) = \frac{1}{n!} \left| \det \begin{pmatrix} v_0 & v_1 & \cdots & v_n \\ 1 & 1 & \cdots & 1 \end{pmatrix} \right|$$

y por ser $v_j^i = \frac{M_j^i}{M_j}$ tendremos

$$Vol(H) = \frac{1}{n!}\frac{1}{|M_0|...|M_n|}\left|\det\begin{pmatrix} M_0^1 & \cdots & M_n^1 \\ \vdots & \cdots & \vdots \\ M_0^n & \cdots & M_n^n \\ M_0 & \cdots & M_n \end{pmatrix}\right|$$

el último valor absoluto de la expresión anterior es mayor o igual que 1 ya que es entero y estrictamente mayor que cero. Y como

$$|M_j| < \frac{2^L}{n}, \forall j$$

tendremos

$$Vol(H) > \frac{1}{n!}\frac{1}{\left(\frac{2^L}{n}\right)^{n+1}} > 2^{-L(n+1)}$$

□

En caso de que el poliedro definido por (1) fuera no acotado, bastaría añadir las restricciones:

$$|x_i| < \frac{2^L}{n}, \quad i = 1, ..., n.$$

Lema 1.7.6. $\forall k, E_K \cap \left\{x \in \mathbb{R}^n / -a_{i_k}^t(x - x_k) > 0,\ a_{i_k}, x_k \in \mathbb{R}^n\right\} \subset E_{k+1}$.

Demostración. Mediante una transformación afín, E_k se puede transformar en la esfera de radio unidad centrada en el origen. Y mediante un giro de centro el origen (que deja dicha esfera invariante), el vector $-a_{i_k}$ se puede transformar en $(\alpha, 0, ..., 0)^t$ con $\alpha = \left\|a_{i_k}\right\| > 0$. Y por las propiedades de dichas transformaciones en $\mathbb{R}^n$ podemos realizar la demostración con dichos elementos. Así pues:

$$\begin{aligned} x_{k+1} &= \left(\frac{1}{n+1}, 0, ..., 0\right) \\ A_{k+1} &= \frac{n^2}{n^2-1} diag\left(\frac{n-1}{n+1}, 1, ..., 1\right) \end{aligned}$$

Sea $x \in E_k \cap \left\{x \in \mathbb{R}^n / -a_{i_k}^t(x - x_k) > 0,\ a_{i_k}, x_k \in \mathbb{R}^n\right\} \subset E_{k+1}$

$$\Rightarrow \begin{cases} \|x\|^2 \leqslant 1 \\ -a_{i_k}^t(x - x_k) = \alpha x_1 > 0 \end{cases}$$

$$
\begin{aligned}
\Rightarrow (x-x_{k+1})^t A_{k+1}^{-1}(x-x_{k+1}) &= x^t A_{k+1}^{-1}x - 2x^t A_{k+1}^{-1}x_{k+1} + x_{k+1}^t A_{k+1}^{-1}x_{k+1} \\
&= \frac{n^2}{n^2-1}\left[\|x\|^2 + \frac{2}{n-1}x_1^2\right] - \frac{2x_1(n^2-1)(n+1)}{n^2(n^2-1)} + \\
&\quad \frac{(n^2-1)(n+1)}{n^2(n+1)^2(n-1)} \\
&= \frac{n^2-1}{n^2}\|x\|^2 + \frac{2(n+1)}{n^2}(x_1^2 - x_1) + \frac{1}{n^2} \\
&= \frac{n^2-1}{n^2}\left[\|x\|^2 - 1\right] + \frac{2(n+1)}{n^2}x_1(x_1-1) + 1 \\
&\leqslant 1, \\
&\text{pues } \|x\|^2 \leqslant 1 \text{ y } 0 < x_1 \leqslant 1
\end{aligned}
$$

luego $x \in E_{k+1}$ □

Definición 1.7.4. *Sea $T : \mathbb{R}^n \to \mathbb{R}^n$ una transformación afín definida como:*

$$T(\Upsilon) = \{y \in \mathbb{R}^n / y = Ax + b \quad si \quad x \in \Upsilon\} \tag{1.2}$$

Definición 1.7.5. *El volumen de un conjunto $\Upsilon \subset \mathbb{R}^n$, lo denotamos por $Vol(\Upsilon)$ y se define como:$Vol(\Upsilon) = \int_{x\in\Upsilon} dx$*
de modo que

$$
\begin{aligned}
Vol(T(\Upsilon)) &= \int_{y\in T(\Upsilon)} dy \\
&= \int_{x\in\Upsilon} |\det(J(x))| dx \\
&= \int_{x\in\Upsilon} |\det(A)| dx \\
Vol(T(\Upsilon)) &= |\det(A)| \int_{x\in\Upsilon} dx
\end{aligned}
$$

de modo que $Vol(T(\Upsilon)) = |\det(A)| Vol(\Upsilon)$.

Si R es una matríz de rotación definida como

$$R = \frac{2(u + \|u\|e_1)(u + \|u\|e_1)^t}{u + \|u\|e_1} - I \quad , e_1 = (1, 0, 0, \ldots, 0), u \in \mathbb{R}^n$$

entonces $R^t R = I$, como D es simétrica y definida positiva define un elipsoide entonces existe D^{-1} que admite descomposición $D^{-\frac{1}{2}} D^{-\frac{1}{2}} = D^{-1}$ sea $D^{\frac{1}{2}} D^{\frac{1}{2}} = D$.

Sea

$$
\begin{aligned}
E_0 &= E\,(0, I) \\
E_1 &= E\,(x_0, D)
\end{aligned}
$$

Sea $S(\underline{x})$ una transformación afín tal que $S(\underline{x}) = D^{-\frac{1}{2}}(\underline{x} - \underline{x}_0)$

$$
\begin{aligned}
W(\underline{x}) &= RS(\underline{x}) \\
&= RD^{-\frac{1}{2}}(\underline{x} - \underline{x}_0)
\end{aligned}
$$

entonces $W(E_1) = E_0$.

Dada $F = \mathbb{R}^n \to \mathbb{R}^n$ Lineal y afín definida como

$$F(\underline{x}) = D^{-\frac{1}{2}}(\underline{x} - \underline{x}_0)\,; si A_1 = D$$

entonces $F(E_1) = E_0$ dado que

$$\begin{aligned}(\underline{x} - \underline{x}_0)^t A_1^{-\frac{1}{2}} A_1^{-\frac{1}{2}} (\underline{x} - \underline{x}_0) &\leqslant 1\\ F^t(\underline{x}) F(\underline{x}) &\leqslant 1\end{aligned}$$

entonces $F(\underline{x}) \in E_0$ verifica la desigualdad anterior.

Si $x \in E_1 \Rightarrow F(x) = A_1^{-\frac{1}{2}}\left(x - \frac{e_1}{n+1}\right)$ pertenece a E_0
$Vol(E_1) = det\,(A_1)^{\frac{1}{2}}\, Vol(E_0)$ entonces $Vol(E_1) = \sqrt{\det(A_1)}Vol(E_0)$.

Lema 1.7.7. .

a). $Vol(E_0) < 2^{(L+1)n}$.

b). $\forall k, Vol(E_{K+1}) = c_n Vol(E_k)$, *siendo* $c_n < 2^{-\frac{1}{2(n+1)}}$.

Demostración. :

a). $E_0 = E\left(0, 2^{2L} I\right)$ esto es, la esfera de centro el origen y el radio 2^L. Dicha esfera está incluida (estrictamente) en un cubo de lado 2^{L+1} cuyo volumen es $2^{n(L+1)}$.

b). Como las transformaciones afines conservan la relación entre volúmenes, podemos razonar, como en el lema anterior, como con E_k esfera centrada en el origen y de radio unidad, y $a_i^t = (\alpha, 0, ..., 0)$ Tendremos:

$$\begin{aligned}Vol(E_{k+1}) &= \left[\frac{\det(A_{k+1})}{\det(A_k)}\right]^{\frac{1}{2}} Vol(E_k)\\ &= \left[\det(A_{k+1})\right]^{\frac{1}{2}} Vol(E_k)\\ &= c_n Vol(E_k)\end{aligned}$$

puesto que

$$A_{k+1} = \frac{n^2}{n^2-1} diag\left(\frac{n-1}{n+1}, 1, ..., 1\right)$$

$$c_n = \left[\left(\frac{n^2}{n^2-1}\right)^{n-1} \frac{n^2}{(n+1)^2}\right]^{\frac{1}{2}} = \frac{n}{n+1}\left[\frac{n^2}{n^2-1}\right]^{\frac{n-1}{2}}$$

pero

$$\begin{aligned}\frac{n}{n+1} &= 1 - \frac{1}{n+1} < e^{-\frac{1}{n+1}}\\ \frac{n^2}{n^2-1} &= 1 + \frac{1}{n^2-1} < e^{\frac{1}{n^2-1}}\end{aligned}$$

luego

$$c_n < e^{-\frac{1}{n+1}} e^{\frac{n-1}{2(n^2-1)}} = e^{-\frac{1}{2(n+1)}} < 2^{-\frac{1}{2(n+1)}}.$$

□

A continuación pasamos a demostrar el Teorema 1.7.1

Demostración. Supongamos que el sistema (1) admite una solución y que el algoritmo no la ha proporcionado en k pasos con $k \geqslant 4\,(n+1)^2\,L$. Por el Lema 1.7.7.

$$\begin{aligned} Vol(E_K) = c_n Vol(E_{k-1}) = ... = c_n^k Vol(E_0) &< 2^{-\frac{k}{2(n+1)}} 2^{n(L+1)} \\ &\leqslant 2^{-\frac{4(n+1)^2 L}{2(n+1)}} 2^{n(L+1)} \\ &= 2^{-(n+1)L} 2^{-(n+1)L} 2^{n(L+1)} \\ &= 2^{-(n+1)L} 2^{n-L} \\ &< 2^{-(n+1)L} \text{ pues } n < L \end{aligned}$$

Por otra parte, por los lemas 1.7.5. y 1.7.6 y por recurrencia, se tiene que

$$Q \subset E_k \Rightarrow Vol(Q) < 2^{-(n+1)L}.$$

Ahora bien, $Q \neq \phi$ pues el sistema (1) admite solución, y por el lema 1.7.5, tendremos

$$Vol(Q) > 2^{-(n+1)L}$$

que es una contradicción. □

Nota 2. *Consideremos la matriz $A_{m\times n}$ los vectores columna $c, x \in \mathbb{R}^n, y \in \mathbb{R}^m, b \in \mathbb{R}^m$ Si queremos resolver el problema de programación lineal de la forma*

$$\begin{array}{ll} \text{mín:} & c^t x \\ s.a. & Ax \geqslant b \\ & x \geqslant 0 \end{array} \tag{P}$$

basta considerar el problema dual

$$\begin{array}{ll} \text{máx} & b^t y \\ s.a & A^t y \leqslant c \\ & y \geqslant 0 \end{array} \tag{D}$$

por teoria de dualidad, si $\overline{x}$ e $\overline{y}$ son soluciones factibles de P y D, y si $c^t\overline{x} = b^t\overline{y}$, entonces $\overline{x}$ e $\overline{y}$ son soluciones óptimas de P y D respectivamente. Considerando pues el siguiente sistema:

$$\begin{array}{l} c^t x = b^t y \\ Ax \geqslant b \\ A^t y \leqslant c \\ x \geqslant 0, y \geqslant 0 \end{array} \tag{C}$$

si en la solución de P existe y es finita, para hallarla basta calcular una solución de C. Finalmente observemos que el algoritmo del elipsoide se aplica a un sistema con desigualdades estrictas. En el caso de que esto no suceda, como por ejemplo en el problema C, el sistema se puede transformar en otro con desigualdades estrictas añadiendo perturbaciones. La validez de este procedimiento queda asegurada por el siguiente lema

Lema 1.7.8. *El sistema*

$$a_i^t x \leqslant b_i, (i = 1, ..., m,\ x \in \mathbb{R}^n, a_i \in \mathbb{Z}^n, b_i \in \mathbb{Z}) \tag{1}$$

tiene una solución si y solo si el sistema

$$a_i^t x < b_i + \varepsilon, (i = 1, ..., m,\ x \in \mathbb{R}^n, a_i \in \mathbb{Z}^n, b_i \in \mathbb{Z}) \tag{2}$$

tiene una solución. Donde $\varepsilon = 2^{-2L}$.

Demostración. Si el sistema (1) tiene una solución esta también satisface el sistema (2). Inversamente, supongamos que x_0 es una solución del sistema (2), consideremos el conjunto de vectores:

$$I = \left\{a_i \in \mathbb{Z}^n / b_i \leqslant a_i^t x_0 < b_i + \varepsilon,\ x_0 \in \mathbb{R}^n,\ b_i \in \mathbb{Z}\right\}$$

podemos suponer que

$$\forall j,\ a_j = \sum_{a_i \in I} \beta_{ji} a_i$$

siendo β_{ji} ciertos coeficientes. En efecto, si a_j es independiente de los vectores a_i de I, el sistema:

$$\begin{aligned} a_i^t z &= 0, i \in I,\ z \in \mathbb{R}^n \\ a_j^t z &= 1 \end{aligned}$$

tiene una solución z_0. Tomando $x_1 = x_0 + \lambda z_0$ para λ suficientemente pequeño tenemos otra solución x_1 de (2) y que tiene un vector más en el conjunto I. Y este procedimiento lo prodremos repetir como máximo m veces.
Así pues

$$\forall j, a_j = \sum_{a_i \in I'} \beta_{ji} a_i$$

para algún subconjunto I' de vectores linealmente independientes de I. Por la regla de cramer, los coeficientes β_{ji} son cocientes de determinantes de valores absolutos

menores que 2^L (ver lema 1.7.4). Escribamos $\beta_{ij} = \frac{M_{ij}}{|M|}$. Consideremos una solución "$\overline{x}$", del sistema:

$$a_i^t x = b_i, a_i \in I'$$

tendremos para cada j:

$$\begin{aligned}
|M|\left(a_j^t\overline{x} - b_j\right) &= \sum_{a_i \in I'} M_{ij} a_i^t \overline{x} - |M|\, b_j \\
&= \sum_{a_i \in I'} M_{ij} b_i - |M|\, b_j \\
&= -\sum_{a_i \in I'} M_{ij}\left(a_i^t\overline{x}_0 - b_i\right) + |M|\left(a_j^t\overline{x}_0 - b_j\right) \\
&< \varepsilon\left(\sum_{a_i \in I} |M_{ji}| + |M|\right) < 2^{-2L}(m+1)2^L \\
&< 1
\end{aligned}$$

ya que $2^L > m + 1$. Ahora, por la definición de x, los denominadores de sus componentes dividen a D por tanto $|M|\left(a_j^t\overline{x} - b_j\right)$ es entero (y menor que 1) luego $\left(a_j^t\overline{x} - b_j\right) \leqslant 0$, y (1) tiene una solución. Observamos que la construcción de x es a partir de x_0 se realiza en un número polinomial de pasos. □

2

Algoritmo de Karmarkar

El algoritmo de Karmarkar resuelve el problema de programación lineal cruzando a través de la región factible, ésta es la diferencia principal frente al método simplex. Al igual que el algoritmo del elipsoide encuentra la función objetivo con ayuda de elipsoides que garantizan que las aproximaciones de la solución se encuentran dentro de la región factible del problema, ambos algoritmos presentan complejidad polinomial, el algoritmo de Karmarkar a tenido exito en la disminución del tiempo de ejecución de un problema frente al simplex.

Básicamente el algoritmo de Karmarkar transforma el problema 2.1 en el problema:

$$\begin{array}{ll} \text{mín} & c^t Dy \\ s.a. & ADy = 0 \\ & e^t y = 1 \\ & \| \, y - \frac{e}{n} \, \| \leqslant \alpha r \, ; \, 0 < \alpha < 1 \end{array}$$

Esto indica que se busca la solución en una esfera de radio menor o igual al de la esfera inscrita en $e^t y = 1$. La interpretación geométrica del método será explicada en el ejemplo 2.3.2.

2.1. Descripción del Algoritmo

El algoritmo proyectivo de Karmarkar se aplica a un problema de programación lineal del tipo:

$$\begin{array}{ll} \text{mín} & c^t x \\ s.a. & Ax = 0 \\ & e^t x = 1 \\ & x \geqslant 0 \end{array} \tag{2.1}$$

Donde c, x, e son vectores columnas de $\mathbb{R}^n$: con $e^t = (1, ..., 1)$ y $A \in M_{m\times n}$. Suponemos que se verifican las siguientes hipótesis:

H1: El valor óptimo de la función objetivo es cero.

H2: A es el rango completo por filas.

H3: $\exists\ x^0 \in \mathbb{R}^n$ con $x^0 > 0$, factible.

La idea básica de este algoritmo es la siguiente: Dado un $\overline{x}$ factible con $\overline{x} = (x_1, \ldots, x_n)^t > 0$, realizar una transformación proyectiva del tipo:

$$T(x) = \frac{Ax}{e^t Ax} = y$$

de forma que el punto x se transforma en $z = \frac{e}{n}$. veamos a continuación una serie de definiciones y resultados que nos serán de utilidad.

Definición 2.1.1. *Definimos un simplex como el conjunto de puntos que cumple*

$$S = \left\{x \in \mathbb{R}^n / e^t x = 1, x \geqslant 0\right\}$$

Definición 2.1.2. *Definimos $T : S \to S$ de la siguiente forma:*

$$T(x) = \frac{D^{-1}x}{e^t D^{-1}x} \text{ con } D = \begin{pmatrix} x_1 & & \\ & \ddots & \\ & & x_n \end{pmatrix} \tag{2.2}$$

Lema 2.1.1. *La transformación T está bien definida y verifica $e^t T(x) = 1$. Además posee inversa $T^{-1} : S \to S$ que viene dada por $T^{-1}(y) = \frac{Dy}{e^t Dy}$, donde D es de tamaño $n \times n$*

Demostración. Veamos, de (2.2)

$$e^t T(x) = \frac{e^t Dy}{e^t Dy} = 1$$

Si llamamos $y = T(x) = \frac{D^{-1}x}{e^t D^{-1}x}$ entonces $Dy = \frac{x}{e^t D^{-1}x} \Rightarrow x = (e^t D^{-1}x)\, Dy$ pero $e^t Dy = \frac{e^t x}{e^t D^{-1}x} = \frac{1}{e^t D^{-1}x}$ entonces $e^t D^{-1}x = \frac{1}{e^t Dy}$ entonces $x = \frac{Dy}{e^t Dy} \Rightarrow T^{-1}(y) = \frac{Dy}{e^t Dy}$ □

Si aplicamos la transformación a nuestro problema inicial (2.1), obtenemos el

siguiente problema transformado:

$$\begin{array}{ll} \text{mín} & \dfrac{c^t Dy}{e^t Dy} \\ s.a. & \dfrac{ADy}{e^t Dy} = 0 \\ & e^t y = 1 \\ & \dfrac{Dy}{e^t Dy} \geqslant 0 \end{array}$$

Y por ser cero el valor óptimo de la función objetivo, y por la forma especial de la Transformación T, el problema anterior es equivalente a:

$$\begin{array}{ll} \text{mín} & c^t Dy \\ s.a. & ADy = 0 \\ & e^t y = 1 \\ & y \geqslant 0 \end{array}$$

si llamamos $z = \dfrac{e}{n}$ y por la hipótesis $H1$ tendremos que:

$$ADz = Ax = 0; \qquad e^t z = \frac{n}{n} = 1, \qquad z > 0$$

Luego z es un punto factible del problema transformado e interior. A dicho problema transformado le aplicamos el método del gradiente proyectado, tomando precisamente z como valor previo obteniendo un nuevo punto que vendrá dado en la forma:

$$y^1 = z - \alpha \; r \; \widehat{d} \qquad , \qquad \alpha \in (0,1) \;\; , \;\; r = \left(n\,(n-1)\right)^{-\frac{1}{2}} \tag{2.3}$$

$\widehat{d}$ es un vector unitario en la dirección del gradiente de la función objetivo "$c^t Dy$" proyectado sobre el espacio nulo de

$$B = \begin{pmatrix} AD \\ e^t \end{pmatrix}$$

Donde: $B_{(m+1)\times n}$, $A_{m\times n}$, $D_{n\times n}$ y $e \in \mathbb{R}^n$, con $e^t = (1, 1, ..., 1)$.

Por una parte, la elección de $\widehat{d}$ garantiza que el nuevo punto siga verificando las dos primeras restricciones. Por otra parte, la elección de α y r, ya que este último representa, geométricamente, el radio de la esfera inscrita en S, asegura que también cumpla la tercera restricción, es más, por ser $\alpha < 1$, y^1 será estrictamente positivo en todas sus componentes. Todas estas afirmaciones, basadas en ideas geométricas, se justificarán posteriormente.

Si ahora aplicamos a y^1 la Transformación T^{-1}, obtendremos un valor para $x^1 = T^{-1}(y^1)$ que mejora la función objetivo, y al que podemos aplicar nuevamente el procedimiento. Observamos que al ser $y^1 > 0$ también lo será x^1.

En suma, el algoritmo tiene la siguiente forma:

Algoritmo 2.1.1. *(Algoritmo De Karmarkar)*

1. *Hacer* $k = 0$, *si* $c^t x^0 = 0$ *parar* (x^0 *es solución), si no*

2. *Definir:*

$$\begin{aligned} D &= \begin{pmatrix} x_1^k & & \\ & \ddots & \\ & & x_n^k \end{pmatrix}, B = \begin{pmatrix} AD_k \\ e^t \end{pmatrix} \\ P_k &= I - B_k^t \left(B_k B_k^t\right)^{-1} B_k \\ b^k &= P_k D_k c \\ d^k &= \frac{b_k}{\|b_k\|} \end{aligned}$$

3. $y^{k+1} = \frac{e}{n} - \alpha r d^k$

4. $x^{k+1} = \dfrac{D_k y^{k+1}}{e^t D_k y^{k+1}}$

5. *(Terminación). Para un grado de precisión,* $p \in \mathbb{N}$, *prefijado.*

   Si $\dfrac{c^t x^{k+1}}{c^t x^0} \leqslant 2^{-p}$ *parar.*
   En caso contrario, $k+1 \to k$ *y volver al paso 2*

Ejemplo 2.1.1. *Resolver el problema*

$$\begin{array}{ll} \text{mín} & Z = -4x_1 + 4x_2 + 6x_3 + x_4 \\ s.a. & x_1 + x_2 - x_3 - x_4 = 0 \\ & 2x_1 + 3x_2 \quad - 5x_4 = 0 \\ & x_1 + x_2 + x_3 + x_4 = 1 \\ & x_1, x_2, x_3, x_4 \geqslant 0 \end{array}$$

Inicio identificar la matriz A y el vector c

c=$\begin{bmatrix} -4 \\ 4 \\ 6 \\ 1 \end{bmatrix}$; $A = \begin{bmatrix} 1 & 1 & -1 & -1 \\ 2 & 3 & 0 & 5 \end{bmatrix}$

Inicio del algoritmo

$x \leftarrow e$, $x = \begin{bmatrix} 0{,}25 \\ 0{,}25 \\ 0{,}25 \\ 0{,}25 \end{bmatrix}$

$c^t x = 1{,}75$

$$D = \begin{bmatrix} 0{,}25 & 0 & 0 & 0 \\ 0 & 0{,}25 & 0 & 0 \\ 0 & 0 & 0{,}25 & 0 \\ 0 & 0 & 0 & 0{,}25 \end{bmatrix}; \text{si } \widetilde{A} = AD; \widetilde{A} = \begin{bmatrix} 0{,}25 & 0{,}25 & -0{,}25 & -0{,}25 \\ 0{,}50 & 0{,}75 & 0 & -1{,}25 \end{bmatrix}$$

se tiene

$$P = \begin{bmatrix} -0{,}8413 \\ 0{,}8413 \\ -0{,}1683 \\ 0{,}1683 \end{bmatrix}; y = \begin{bmatrix} 0{,}4301 \\ 0{,}0699 \\ 0{,}286 \\ 0{,}214 \end{bmatrix}; \quad x = \begin{bmatrix} 0{,}431 \\ 0{,}0699 \\ 0{,}286 \\ 0{,}214 \end{bmatrix}$$

Primera Iteración

$c^t x = 0{,}489$

$$D = \begin{bmatrix} 0{,}4301 & 0 & 0 & 0 \\ 0 & 0{,}0699 & 0 & 0 \\ 0 & 0 & 0{,}286 & 0 \\ 0 & 0 & 0 & 0{,}214 \end{bmatrix}; \widetilde{A} = \begin{bmatrix} 0{,}431 & 0{,}0699 & -0{,}286 & -0{,}214 \\ 0{,}8603 & 0{,}2096 & 0 & -1.,0699 \end{bmatrix}$$

$$P = \begin{bmatrix} -0{,}1699 \\ 0{,}3556 \\ -0{,}1186 \\ 0{,}067 \end{bmatrix}; y = \begin{bmatrix} 0{,}3559 \\ 0{,}0285 \\ 0{,}3239 \\ 0{,}2918 \end{bmatrix}; \quad x = \begin{bmatrix} 0{,}4936 \\ 0{,}0064 \\ 0{,}2987 \\ 0{,}2013 \end{bmatrix}$$

Segunda Iteración

$c^t x = 0{,}0449$

$$P = \begin{bmatrix} -0{,}0116 \\ 0{,}0336 \\ -0{,}0112 \\ -0{,}0108 \end{bmatrix}; y = \begin{bmatrix} 0{,}3278 \\ 0{,}0250 \\ 0{,}3252 \\ 0{,}3220 \end{bmatrix}; \quad x = \begin{bmatrix} 0{,}4995 \\ 0{,}0005 \\ 0{,}2999 \\ 0{,}2001 \end{bmatrix}$$

Tercera Iteración

$c^t x = 0{,}0035$

$$P = \begin{bmatrix} -0{,}0009 \\ 0{,}0026 \\ -0{,}0009 \\ -0{,}0009 \end{bmatrix}; y = \begin{bmatrix} 0{,}3252 \\ 0{,}0250 \\ 0{,}3250 \\ 0{,}3248 \end{bmatrix}; \quad x = \begin{bmatrix} 0{,}50 \\ 0{,}00 \\ 0{,}30 \\ 0{,}20 \end{bmatrix}$$

Cuarta Iteración

$c^t x = 0{,}0002667$

$$P = 10^{-3}\begin{bmatrix} -0{,}0667 \\ 0{,}2000 \\ -0{,}0667 \\ -0{,}0667 \end{bmatrix};\ y = \begin{bmatrix} 0{,}3250 \\ 0{,}0250 \\ 0{,}3250 \\ 0{,}3250 \end{bmatrix};\ \ x = \begin{bmatrix} 0{,}50 \\ 0{,}00 \\ 0{,}30 \\ 0{,}20 \end{bmatrix} \text{ solución final}$$

2.2. Convergencia y Complejidad

Un resultado fundamental, obtenido por Karmarkar, asegura que α se puede escoger de manera que:

$$\ln(1+\alpha) - \frac{\beta^2}{1-\beta} > 0 \ \text{ donde } \quad \beta = \alpha\left(\frac{n}{n-1}\right)^{\frac{1}{2}} < 1$$

Teorema 2.2.1. *El algoritmo 2.1.1 necesita como máximo $O(np)$ iteraciones antes de detenerse para cualquier $p \geqslant 1$ y con $0 < \alpha < 0,7968...$*

Antes de exponer la demostración daremos una serie de lemas previos . Suponemos, sin pérdida de generalidad, que $x^0 = z = \frac{e}{n}$, pues si no fuera asi, bastaría tranformar el problema como se mostró en la sección anterior.

Lema 2.2.1. *La sucesión de puntos $x^k \in \mathbb{R}^n, k \in \mathbb{N} \cup \{0\}$ generada por el algoritmo 2.1.1, es positivo y factible para $0 \leqslant \alpha < \left(\frac{n-1}{n}\right)^{\frac{1}{2}}$.*

Demostración. Observemos que d^k es un vector unitario luego $-1 \leqslant d_j^k \leqslant 1$. Además de (2.3) $y_j^{k+1} = \frac{1}{n} - \frac{\alpha d_j^k}{(n(n-1))^{\frac{1}{2}}}$ pero

$$\left|\frac{\alpha d_j^k}{(n(n-1))^{\frac{1}{2}}}\right| \ < \ \frac{\left(\frac{n-1}{n}\right)^{\frac{1}{2}}}{(n(n-1))^{\frac{1}{2}}} = \frac{1}{n} \Rightarrow y_j^{k+1} > 0 \qquad , \qquad \forall\ j$$

$$\Rightarrow \ y^{k+1} > 0 \Rightarrow x^{k+1} = \frac{D_k y^{k+1}}{e^t D_k y^{k+1}} > 0$$

Por otra parte

$$e^t x^{k+1} = e^t \frac{D_k y^{k+1}}{e^t D_k y^{k+1}} = 1 \quad \text{(Segunda restricción)}$$

$$Ax^{k+1} = \frac{AD_k y^{k+1}}{e^t D_k y^{k+1}} = \frac{1}{e^t D_k y^{k+1}}\left[AD_k\left[z - \alpha r d_k\right]\right] = 0$$

ya que $AD_k z = Ax^k/n = 0$ y la aplicación del método del gradiente proyectado garantiza que la dirección de d_k pertenezca al espacio nulo de AD_k $\square$

Lema 2.2.2. *Sean* $H = \{x \in \mathbb{R}^n / e^t x = 1,\, e \in \mathbb{R}^n\}, \alpha \in [0,1\rangle, r = \dfrac{1}{(n(n-1))^{\frac{1}{2}}}$, $z \in \mathbb{R}^n$ *se verifican:*

a). $H \cap B[z,r] \subseteq \mathbb{R}^{n+} \subseteq H \cap B[z,(n-1)r]$

b). $x \in H \cap B[z,\alpha r] \Rightarrow \Pi x_j \geqslant (1-\alpha)\left(1+\dfrac{\alpha}{n-1}\right)^{n-1} n^{-n}$

Donde $B[z,r]$ representa la bola cerrada con centro en z y radio r, $\mathbb{R}^{+n}$ representa el vector con coordenadas positivas.

Demostración. Veamos por casos

a). Sea $x = (x_1, ..., x_n)^t \in H \cap B[z,r]$. Para demostrar que $x \in \mathbb{R}^{n+}$, supongamos, sin pérdida de generalidad, que $x_1 < 0$ tenemos que.

$$\begin{aligned}\left(x_2 - \frac{1}{n}\right)^2 + ... + \left(x_n - \frac{1}{n}\right)^2 &\leqslant r^2 - \left(x_1 - \frac{1}{n}\right)^2 < r^2 - \frac{1}{n^2} \\ &= \frac{1}{n(n-1)} - \frac{1}{n^2} = \frac{1}{n^2(n-1)}\end{aligned}$$

y por la desigualdad de Cauchy-Schwarz

$$\begin{aligned}\left(x_2 - \frac{1}{n}\right) + ... + \left(x_n - \frac{1}{n}\right) &\leqslant (n-1)^{\frac{1}{2}} \left(\left(x_2 - \frac{1}{n}\right)^2 + ... + \left(x_n - \frac{1}{n}\right)^2\right)^{\frac{1}{2}} \\ &< (n-1)^{\frac{1}{2}} \left(\frac{1}{n^2(n-1)}\right)^{\frac{1}{2}} = \frac{1}{n} \Rightarrow \\ x_2 + ... + x_n &< \frac{1}{n} + \frac{n-1}{n} = 1 \\ &\Rightarrow x_1 + ... + x_n < 1\end{aligned}$$

Lo que contradice el hecho de $x \in H$.
Ahora, si $x \in H \cap \mathbb{R}^{n+}$

$$\begin{aligned}\left(x_1 - \frac{1}{n}\right)^2 + ... + \left(x_n - \frac{1}{n}\right)^2 &= \left(x_1^2 + ... + x_n^2\right) + \frac{n}{n^2} - \frac{2}{n}(x_1 + ... + x_n) \\ &\leqslant (x_1 + ... + x_n)^2 + \frac{1}{n} - \frac{2}{n}(x_1 + ... + x_n) \\ &= 1 - \frac{2}{n} + \frac{1}{n} = 1 - \frac{1}{n} = r^2(n-1)^2\end{aligned}$$

de donde se deduce que $x \in B[z,(n-1)r]$

b). Primero probaremos lo siguiente:
Sea $\lambda, \mu \in \mathbb{R}$. Si Υ, Ψ, Ω son valores reales que resuelven el problema

$$\text{mín}\left\{\eta\gamma\varphi/\ \eta+\gamma+\varphi=\lambda,\ \eta^2+\gamma^2+\varphi^2=\mu\right\} \text{ con } \Upsilon \leqslant \Psi \leqslant \Omega \qquad (1)$$

entonces $\Psi = \Omega$.
En efecto. Reemplazando η, γ, φ por $\eta - \frac{\lambda}{3}, \gamma - \frac{\lambda}{3}, \varphi - \frac{\lambda}{3}$, podemos suponer que $\lambda = 0$. Es claro que el mínimo es no positivo se tiene $\Upsilon \leqslant 0 \leqslant \Psi \leqslant \Omega$ entonces por la desigualdad entre media aritmética y geométrica, y por ser $\Upsilon \leqslant 0$.

$$\Upsilon\Psi\Omega \geqslant \Upsilon\left(\frac{\Psi+\Omega}{2}\right)^2 = \frac{\Upsilon^3}{4}$$

por otra parte

$$\begin{aligned} \Upsilon^2+\Psi^2+\Omega^2=\mu \qquad & \Upsilon+\Psi+\Omega=0 \\ \Rightarrow\quad & \Upsilon=-(\Psi+\Omega) \\ \Rightarrow\quad & \Upsilon^2=(\Psi+\Omega)^2=\Psi^2+\Omega^2+2\Psi\Omega \\ \Rightarrow\quad & \Upsilon^2+\Upsilon^2-2\Psi\Omega=\mu \\ \Rightarrow\quad & \Upsilon^2-\Psi\Omega=\frac{\mu}{2} \end{aligned}$$

Pero

$$\begin{aligned} \Psi\Omega \leqslant \left(\frac{\Psi+\Omega}{2}\right)^2=\frac{\Upsilon^2}{4} \quad \Rightarrow\quad & \frac{\mu}{2}=\Upsilon^2-\frac{\Upsilon^2}{4} \geqslant \frac{3\Upsilon^2}{4} \\ \Rightarrow\quad & \Upsilon^2 \leqslant \frac{2\mu}{3} \\ \Rightarrow\quad & \Upsilon \geqslant -\frac{(6\mu)^{\frac{1}{2}}}{3} \\ \Rightarrow\quad & \Upsilon\Psi\Omega \geqslant -\mu\frac{(6\mu)^{\frac{1}{2}}}{18} \qquad (2) \end{aligned}$$

Por otra parte

$$(\Upsilon,\Psi,\Omega)=\left(-\frac{(6\mu)^{\frac{1}{2}}}{3}, \frac{(6\mu)^{\frac{1}{2}}}{6}, \frac{(6\mu)^{\frac{1}{2}}}{6}\right)$$

cuando satisface $\Upsilon+\Psi+\Omega=0, \Upsilon^2+\Psi^2+\Omega^2=\mu, \Upsilon\Psi\Omega=-\mu\frac{(6\mu)^{\frac{1}{2}}}{18}$ Es decir, produce la igualdad en (2) por lo tanto $\Psi = \Omega$.

probemos ahora b). El caso $n = 2$ es trivial. Supongamos que $n \geqslant 3$, además supongamos que $x = (x_1, ..., x_n)^t$ resuelve el problema:

$$\text{mín}\left\{\Pi x_j\ /\ x \in H \cap B\left[z, \alpha r\right]\right\}$$

sin pérdida de generalidad podemos suponer $x_1 \leqslant x_2 \leqslant ... \leqslant x_n$. Entonces $\forall i, j, k$ con $1 \leqslant i < j < k \leqslant n$ tendremos que el vector (x_i, x_j, x_k) resuelve el problema:

$$\text{mín}\left\{\eta\gamma\varphi \ / \ \eta + \gamma + \varphi = x_i + x_j + x_k \ , \ \eta^2 + \gamma^2 + \varphi^2 = x_i^2 + x_j^2 + x_k^2\right\}$$

Pues de otra manera, bastaría reemplazar sus valores por otros más apropiados. De aquí, por (1), se sigue que $x_i = x_k$. Por lo tanto tendremos x_1; $x_2 = x_3 = ... = x_n$. Y por ser $x \in H \cap B\left[z, \alpha r\right]$, un sencillo cálculo nos lleva a:

$$x_1 = \frac{(1-\alpha)}{n},\ x_2 = x_3 = ... = x_n = \left(1 + \frac{\alpha}{n-1}\right)\frac{1}{n}$$

Y el producto de tales x_i nos da la cota inferior de b). □

Corolario 2.2.1. *La sucesión de puntos $x^k, k \in \mathbb{N} \cup \{0\}$ generada por el algoritmo 2.1.1 es positiva y factible $\forall \alpha \in [0, 1\rangle$.*

Demostración. Basta tener en cuenta el apartado b) del lema anterior, pues, por la forma de obtener x^k este verifica $y^k \in H \cap B\left[z, \alpha r\right]$ luego tendremos

$$\Pi y_j^k > 0 \Rightarrow y^k > 0 \Rightarrow x^k > 0$$

donde $y \in S$ □

Lema 2.2.3. *El punto $y^{k+1} = z - \rho d^k$ resuelve el problema de optimización:*

$$\text{mín}\left\{c^t D_k y \ \ / \ AD_k y = 0 \ \ , \ \ e^t y = 1 \ \ , \ \ y \in B\left[z, \rho\right]\right\}$$

donde $D_k \in M_{n\times n}$ y $A_{m\times n}$, $y \in \mathbb{R}^n$, $c \in \mathbb{R}^n$, $e \in \mathbb{R}^n$, $z \in S$.

Demostración. Llamando $B = \begin{pmatrix} AD_k \\ e^t \end{pmatrix}_{(m+1)\times n}$, las restricciones $\left\{ \begin{array}{c} AD_k y = 0 \\ e^t y = 1 \end{array} \right\}$ pueden ser escritas como $By = b = (0, ..., 0, 1)^t$ y el problema:

$$\text{mín}\left\{c^t D_k y \ \ / \ By = b, \|z - y\| \leqslant \rho\right\}$$

Sea $\lambda = (\lambda_1, ...\lambda_n, \lambda_{n+1})^t$ el vector de multiplicadores de Lagrange asociados a las restricciones $By = b$. Y sea "y" un punto factible, tendremos:

$$\lambda^t By = \lambda^t b \qquad , \qquad \lambda^t Bz = \lambda^t b$$

luego

$$\begin{aligned} c^t D_k z - c^t D_k y &= \left(D_k c - B^t\lambda\right)^t \cdot (z - y) \\ &\leqslant \left\|D_k c - B^t\lambda\right\| \|z - y\| \leqslant \left\|D_k c - B^t\lambda\right\| \rho \end{aligned}$$

Conforme nos acerquemos al mínimo de $c^t D_k y$ más nos acercamos a la cota anterior. La igualdad se dará si se cumplen: $D_k c - B^t\lambda = \gamma(z-y)$ dado que los vectores son paralelos se tiene $\|z-y\| = \rho \Rightarrow \gamma = \dfrac{\|D_k c - B^t\lambda\|}{\rho}$ de donde se deduce que $y = z - \rho\dfrac{D_k c - B^t\lambda}{\|D_k^t c - B^t\lambda\|}$ nos dará el mínimo. Pero

$$\begin{aligned} B\left(D_k c - B^t\lambda\right) &= B\gamma(y-z) = \gamma(b-b) = 0 \\ &\Rightarrow BD_k c = BB^t\lambda \Rightarrow \lambda = \left(BB^t\right)^{-1} BD_k c \end{aligned}$$

y teniendo en cuenta que $b^k = \left(I - B^t\left(BB^t\right)^{-1}B\right)D_k c$ queda; $d^k = \frac{b^k}{\|b^k\|}$

$$y = z - \rho d^k$$

Donde: $I_{n\times n}$, $B_{(m+1)\times n}$, $c \in \mathbb{R}^n$. □

Definición 2.2.1. *Llamamos función potencial a la función $h : \mathbb{R}^+\setminus\{0\} \to \mathbb{R}$ definida de la siguiente forma:* $h(x) = \dfrac{c^t x}{(\Pi x_j)^{\frac{1}{n}}}$

Lema 2.2.4. *se cumple*

$$\frac{h(x^{k+1})}{h(x^k)} = \frac{c^t D_k y^{k+1}}{c^t D_k z}\frac{1}{\left(\Pi n y_j^{k+1}\right)^{\frac{1}{n}}}$$

Demostración. Recordemos que $x^{k+1} = \dfrac{D_k y^{k+1}}{e^t D_k\ y^{k+1}}$ y que $D_k e = x^k$

$$\begin{aligned} \frac{h(x^{k+1})}{h(x^k)} &= \frac{c^t x^{k+1}}{\left(\Pi x_j^{k+1}\right)^{\frac{1}{n}}}\frac{\left(\Pi x_j^k\right)^{\frac{1}{n}}}{c^t x^k} \\ &= \frac{c^t D_k y^{k+1}}{e^t D_k y^{k+1}}\frac{e^t D_k y^{k+1}}{\left(\Pi x_j^k y_j^{k+1}\right)^{\frac{1}{n}}}\frac{\left(\Pi x_j^k\right)^{\frac{1}{n}}}{c^t x^k} \\ &= \frac{c^t D_k y^{k+1}}{c^t D_k z}\frac{1}{\left(\Pi n y_j^{k+1}\right)^{\frac{1}{n}}} \end{aligned}$$

□

Lema 2.2.5. *Se cumple*

$$\frac{c^t D_k y^{k+1}}{c^t D_k z} \leqslant 1 - \frac{\alpha}{n-1}$$

donde α es del teorema 2.2.1

Demostración. Veamos

$$\begin{aligned} c^t D_k y^{k+1} &= c^t D_k \left(z - \alpha r d_k\right) \\ &= \left(1 - \frac{\alpha}{n-1}\right) c^t D_k z + \frac{\alpha}{n-1} c^t D_k \left(z - (n-1) r d_k\right) \end{aligned}$$

por el lema 2.2.3

$$c^t D_k \left(z - (n-1) r d_k\right) = \text{mín}\left\{c^t D_k y / A D_k y = 0, e^t y = 1, y \in B\left[z, (n-1) r\right]\right\}$$

y por el lema 2.2.2, apartado a), dicho mínimo será menor o igual que:

$$\text{mín}\left\{c^t D_k y \ / \ A D_k y = 0, e^t y = 1, y \geqslant 0\right\} = 0$$

entonces

$$c^t D_k y^{k+1} \leqslant \left(1 - \frac{\alpha}{n-1}\right) e^t D_k z$$

. □

Lema 2.2.6. *Se cumple*

$$\frac{1}{\left(\Pi n y_j^{k+1}\right)^{\frac{1}{n}}} \leqslant \left(\frac{1}{(1-\alpha)^{\frac{1}{n}}}\right) \left(\frac{1}{\left(1 + \frac{\alpha}{(n-1)}\right)^{\frac{n-1}{n}}}\right)$$

Demostración. Reemplazando x_j por y^{k+1} en el apartado b) del lema 2.2.2 resulta la desigualdad anterior. □

A continuación pasamos a demostrar el Teorema 2.2.1

Demostración. De los lemas , 2.2.4, 2.2.5 y 2.2.6 se sigue que:

$$\frac{h(x^{k+1})}{h(x^k)} \leqslant \frac{1 - \frac{\alpha}{n-1}}{1 + \frac{\alpha}{n-1}} \left(\frac{1 + \frac{\alpha}{n-1}}{1-\alpha}\right)^{\frac{1}{n}} = g(\alpha, n)$$

por otra parte:

$$\frac{h(x^m)}{h(x^0)} = \frac{h(x^m)}{h(x^{m-1})} \frac{h(x^{m-1})}{h(x^{m-2})} \cdots \frac{h(x^1)}{h(x^0)} \leqslant \left(g(\alpha, n)\right)^m$$

pero, denotamos $\Pi \frac{1}{n} = \Pi_{j=1}^n \left(\frac{1}{n}\right)$ se tiene: $\frac{h(x^m)}{h(x^0)} = \frac{c^t x^m}{c^t x^0} \frac{\left(\Pi \frac{1}{n}\right)^{\frac{1}{n}}}{\left(\Pi x_j^m\right)^{\frac{1}{n}}} \Rightarrow$

$$\begin{aligned} \frac{c^t x^m}{c^t x^0} &\leqslant n \left(\Pi x_j^m\right)^{\frac{1}{n}} \left(g(\alpha, n)\right)^m \\ &\leqslant n \left(\left(\frac{\sum x_j^m}{n}\right) g(\alpha, n)\right)^m = g^m(\alpha, n) \end{aligned}$$

Para obtener la última expresión se ha aplicado la desigualdad, entre media aritmética y media geométrica, y que $e^t x^m = 1$. Seguidamente vamos a buscar una cota para la función

$$\begin{aligned} g(\alpha, n) &= \frac{1 - \dfrac{\alpha}{n-1}}{1 + \dfrac{\alpha}{n-1}} \left(\frac{1 + \dfrac{\alpha}{n-1}}{1-\alpha} \right)^{\frac{1}{n}}, \alpha \in (0,1) \\ \ln g(\alpha, n) &= \ln \left(\frac{1 - \dfrac{\alpha}{n-1}}{1 + \dfrac{\alpha}{n-1}} \right) + \frac{1}{n} \ln \left(\frac{1 + \dfrac{\alpha}{n-1}}{1-\alpha} \right) \end{aligned}$$

Considerando el desarrollo $\ln\left(\dfrac{1+x}{1-x}\right) = 2\left(x + \dfrac{x^3}{3} + \dfrac{x^5}{5} + ...\right)$ válido para $|x| < 1$, tendremos para el primer sumando

$$\ln \left(\frac{1 - \dfrac{\alpha}{n-1}}{1 + \dfrac{\alpha}{n-1}} \right) = 2\left(\frac{-\alpha}{n-1} - \frac{\alpha^3}{3(n-1)^3} - ... \right) \leqslant \frac{-2\alpha}{n-1}$$

Por otra parte

$$1 - x \leqslant e^{-x} \text{ para } |x| \leqslant 1 \Rightarrow \ln(1-x) \leqslant -x \Rightarrow \ln\left(1 + \frac{\alpha}{n-1}\right) \leqslant \frac{\alpha}{n-1} \Rightarrow$$

$$\begin{aligned} \ln g(\alpha, n) &\leqslant \frac{-2\alpha}{n-1} + \frac{1}{n}\frac{\alpha}{n-1} - \frac{1}{n}\ln(1-\alpha) \\ &= \frac{-2\alpha}{n-1} - \frac{\alpha}{n} + \frac{\alpha}{n-1} - \frac{1}{n}\ln(1-\alpha) \\ &= \frac{-\alpha}{n-1} - \frac{1}{n}(\alpha + \ln(1-\alpha)) \\ &\leqslant \frac{-\alpha}{n-1} - \frac{1}{n-1}(\alpha + \ln(1-\alpha)) \\ &= \frac{-2\alpha}{n-1} - \frac{\ln(1-\alpha)}{n-1} \\ &= \ln\left(\frac{e^{-2\alpha}}{1-\alpha}\right)^{\frac{1}{n-1}} \end{aligned}$$

entonces

$$g(\alpha, n) \leqslant \left(\frac{e^{-2\alpha}}{1-\alpha}\right)^{\frac{1}{n-1}} \Rightarrow \frac{c^t x^m}{c^t x^0} \leqslant \left(\frac{e^{-2\alpha}}{1-\alpha}\right)^{\frac{m}{n-1}}$$

la función $f(\alpha) = \dfrac{e^{-2\alpha}}{1-\alpha}$ verifica $f(0) = 1, f'(0) < 0$ y tiene un mínimo para $\alpha = \dfrac{1}{2}$ luego existe $\alpha_0 \in (0,1)$ tal que: $f(\alpha_0) = 1$, donde se verifica que $0 < f(\alpha) < 1$ para

$\alpha \in (0, \alpha_0)$, siendo α_0 la raiz no nula de la ecuación $e^{-2\alpha} = 1 - \alpha$ que es

$$\alpha_0 = 0,7968...$$

y para $\alpha \in (0, \alpha_0)$ tendremos que el valor óptimo de la función objetivo se podrá hacer más pequeño como se quiera. Si tomamos $m \geqslant \dfrac{(n-1)\, p \ln 2}{2\alpha + \ln(1-\alpha)}$ tendremos

$$\frac{c^t x^m}{c^t x^0} \leqslant \left(\frac{e^{-2\alpha}}{1-\alpha}\right)^{\frac{m}{n-1}} \leqslant \left(\frac{e^{-2\alpha}}{1-\alpha}\right)^{\frac{p \ln 2}{2\alpha + \ln(1-\alpha)}} \leqslant 2^{-p}$$

luego en $O(np)$ iteraciones conseguimos la presición deseada. □

Oservación 2.2.1. *:*
En el algoritmo 2.1.1 se calcula en cada iteración $(B_k B_k^t)^{-1}$*, esto necesita un número* $O(n^3)$ *pasos en cada iteración, de esta forma el algoritmo tiene un orden de convergencia* $O(n^4 p)$.

Oservación 2.2.2. *:*
Podemos obtener una cota sencilla de $g(\alpha, n)$ *tomando* $\alpha = \dfrac{n-1}{2n-3}$*, con lo cual obtenemos:*

$$g(\alpha, n) \leqslant \left(\frac{2}{e}\right)^{\frac{1}{n}}$$

y tomando $m \geqslant np\left(\dfrac{\ln 2}{1 - \ln 2}\right)$ *se consigue la acotación deseada. Además, por ser* $\dfrac{2}{3} \leqslant \ln 2 \leqslant \dfrac{3}{4}$*, se cumple* $m \geqslant 3np$.

2.3. Transformaciones del Problema General

En general, un problema de programación lineal vendrá dado como:

$$\begin{array}{ll} \text{mín} & c^t x \\ s.a. & Ax = b \\ & x \geqslant 0 \end{array}$$

Donde: la matriz $A_{m \times n}$, x, c, e son vectores columna en $\mathbb{R}^n$, $b \in \mathbb{R}^m$. Pero la aplicación del algoritmo de Karmarkar requiere que se verifiquen una serie de condiciones que podemos resumir en :

1. Que el problema aparezca en la forma :

$$\begin{array}{ll} \text{mín} & c^t x \\ s.a. & Ax = 0 \\ & e^t x = 1 \\ & x \geqslant 0 \end{array}$$

2. Conocimiento de una solución inicial factible $x > 0$.

3. Valor óptimo de la función objetivo igual a cero.

Como veremos, esta última condición se puede sustituir por el requerimiento, menos restrictivo de que se conozca el valor óptimo de la función objetivo.

Numerosas transformaciones y modificaciones del algoritmo se han ideado para poder aplicarlo a un problema de tipo general. Hemos de mencionar, en primer lugar, la solución propuesta por el propio Karmarkar. Esta solución combina el problema primal con el dual, de la siguiente forma:

Dado el problema

$$\begin{array}{ll} \text{mín} & c^t x \\ s.a. & Ax = b \\ & x \geqslant 0 \end{array}$$

Donde: la matriz $A_{m\times n}$, x, c son vectores columna en $\mathbb{R}^n$, $b \in \mathbb{R}^m$
Consideremos su dual

$$\begin{array}{ll} \text{máx} & b^t \overline{u} \\ s.a. & A^t \overline{u} + w = c \\ & w \geqslant 0 \end{array}$$

Donde: la matriz $A_{m\times n}$, $\overline{u}, b$ son vectores columna en $\mathbb{R}^m$, $w \in \mathbb{R}^n$
Con el requerimiento que los valores óptimos de ambas funciones objetivos coinciden.

Reemplazando el vector $\overline{u}$ por la diferencia de dos vectores no negativos, $u - v$, con $u, v \in \mathbb{R}^m$ obtenemos

$$\begin{array}{l} Ax = b \\ A^t u - A^t v + w = c \\ c^t x - b^t u + b^t v = 0 \\ x \geqslant 0, u \geqslant 0, v \geqslant 0, w \geqslant 0 \end{array}$$

Y podemos plantear el siguiente problema de programación lineal:

$$\begin{array}{lllll} \text{mín:} & \lambda & & & \\ s.a. & Ax & & +g\lambda = b & \\ & A^t u - A^t v + w & & +h\lambda = c & \text{(Q)} \\ & c^t x - b^t u + b^t v & & +\beta\lambda = 0 & \\ & x \geqslant 0, u \geqslant 0, & & v \geqslant 0, w \geqslant 0, \lambda \geqslant 0 & \end{array}$$

cuya solución óptima ha de ser $\lambda = 0$, y de donde se han hecho las siguientes sustituciones:

$$\begin{array}{l} g = b - Ax_0 \\ h = c - (A^t u_0 - A^t v_0 + w_0) \\ \beta = -(c^t x_0 - b^t u_0 + b^t v_0) \end{array}$$

donde: b, g, u_0, v_0 son vectores columnas en $\mathbb{R}^m$, h, c, w_0 son vectores columnas en $\mathbb{R}^n$ y $\beta \in \mathbb{R}$.

Siendo $(x_0^t, u_0^t, v_0^t, w_0^t) > 0$ cualquier positivo arbitrario. Por ejemplo $\dfrac{e^t}{2n+2m}$, entonces el problema anterior tiene $(x_0^t, u_0^t, v_0^t, w_0^t, 1)$ como solución factible positiva, el orden del vector es $2(m+n+1)$ y de aquí en adelante será denotado por n. Consideremos la siguiente transformación:

$$\begin{aligned} y_i &= \frac{y_i'}{1+\sum_{j=1}^{n-1} y_j'}, i = 1, 2, ..., n-1 \\ y_n &= \frac{1}{1+\sum_{j=1}^{n-1} y_j'} \end{aligned}$$

Y observamos que $\sum_{i=1}^{n} y_i = 1$ y que $y_i = y_i' \cdot y_n, i = 1, 2, ...n-1$, siendo el orden del vector $2\,(m+n+1)\,, y' = (x^t, u^t, v^t, w^t, 1)^t$, luego el problema Q se convierte;

$$\begin{array}{ll} \text{mín} & \dfrac{y_{n-1}}{y_n} \\ s.a. & \overline{A}y = 0 \\ & e^t y = 1 \\ & y \geqslant 0 \end{array}$$

siendo

$$\overline{A} = \begin{pmatrix} A_{m\times n} & 0_{m\times m} & 0_{m\times m} & 0_{m\times n} & g_{m\times 1} & -b_{m\times 1} \\ 0_{n\times n} & A_{n\times m}^t & -A_{n\times m}^t & I_{n\times n} & h_{n\times 1} & -c_{n\times 1} \\ c_{1\times n}^t & -b_{1\times m}^t & b_{1\times m}^t & 0_{1\times n} & \beta_{1\times 1} & 0_{1\times 1} \end{pmatrix}$$

El problema anterior es equivalente a minimizar y_{n-1} considerando las mismas restricciones. $y^0 = \dfrac{e}{n}$ es un punto factible y en el óptimo las variables originales se obtienen como:

$$y_i' = \frac{y_i}{y_n}, i = 1, 2, ..., n-1$$

Con estas transformaciones de un problema general se obtiene un problema equivalente al que se puede aplicar el algoritmo de Karmarkar. Desgraciadamente el problema inicial se ve tremendamente aumentado.

Veamos a continuación otros tipos de transformaciones que nos permiten superar los requerimientos anteriores.

2.3.1. Transformación del Problema General Cuando se Conoce una Solución Factible Estrictamente Positiva.

Consideremos el problema

$$\begin{array}{ll} \text{mín} & c^t y \\ s.a. & Ay = b \\ & y \geqslant 0 \end{array} \tag{P}$$

Donde: la matriz $A_{m\times n}$, los vectores columna y, c en $\mathbb{R}^n$, $b \in \mathbb{R}^m$

Y sea $\overline{y} = (y_1^0, y_2^0, ...y_n^0,)^t > 0$ una solución factible. Suponemos que el valor óptimo de la función objetivo vale cero. Consideremos la siguiente Transformación:

$$x_j = \frac{\frac{y_j}{y_j^0}}{1 + \sum_{j=1}^{n} \frac{y_j}{y_j^0}}, j = 1, 2, ..., n \qquad ; x_{n+1} = 1 - \sum_{j=1}^{n} x_j \tag{T1}$$

que está bien definida en el conjunto factible. Si llamamos

$$\begin{aligned} K &= 1 + \sum_{j=1}^{n} \frac{y_j}{y_j^0} \Rightarrow x_j = \frac{\frac{y_j}{y_j^0}}{K} \Rightarrow y_j = x_j y_j^0 K \Rightarrow K x_j = \frac{y_j}{y_j^0} \\ &\Rightarrow K \sum_{j=1}^{n} x_j = \sum_{j=1}^{n} \frac{y_j}{y_j^0} = K - 1 \Rightarrow K\left(1 - x_{n+1}\right) = K - 1 \\ &\Rightarrow K x_{n+1} = 1 \Rightarrow K = \frac{1}{x_{n+1}} \Rightarrow y_j = \frac{x_j y_j^0}{x_{n+1}}; j = 1, 2, ..., n \end{aligned}$$

y sustituyendo en el problema (P), queda:

$$\begin{array}{ll} \text{mín} & \sum_{j=1}^{n} \frac{c_j y_j^0 x_j}{x_{n+1}} \\ s.a. & \sum_{j=1}^{n} \frac{a_{ij} y_j^0 x_j}{x_{n+1}} = b_i; i = 1, 2, ..., n \\ & \sum_{j=1}^{n+1} x_j = 1 \\ & x_j \geqslant 0; j = 1, 2, ..., n+1 \end{array} \tag{P'}$$

Observemos que T1 transforma el problema P en el P' de manera que:

a). A cada solución factible "y ", de P, le corresponde una solución factible, "x ", de P'

b). La solución óptima de P se transforma en la solución óptima de P' y en ambos casos el óptimo vale cero.

De esta forma, el problema anterior es equivalente a:

$$\begin{array}{ll} \text{mín} & \sum_{j=1}^{n} c_j y_j^0 x_j \\ s.a. & \sum_{j=1}^{n} a_{ij} y_j^0 x_j - b_j x_{n+1} = 0; i = 1, 2, ..., m \\ & \sum_{j=1}^{n+1} x_j = 1 \\ & x_j \geqslant 0; j = 1, 2, ..., n+1 \end{array} \qquad (P'')$$

Y llamamos $x^t = (x_1, x_2, ...x_n)\,, D = \begin{pmatrix} y_1^0 & & \\ & \ddots & \\ & & y_n^0 \end{pmatrix}; \overline{A} = [AD - b]\,; \overline{c} = \begin{pmatrix} Dc \\ 0 \end{pmatrix}$

donde: $A_{m\times n}, D_{n\times n}$, los vectores columna x, c, e en $\mathbb{R}^n$, $b \in \mathbb{R}^m$ de manera que P'' queda en la forma

$$\begin{array}{ll} \text{mín} & \overline{c}^t x \\ s.a. & \overline{A} x = 0 \\ & e^t x = 1; x \geqslant 0 \end{array}$$

que es la requerida por el algoritmo de Karmarkar. Además el punto

$$x^0 = \left(\frac{1}{n+1}, ..., \frac{1}{n+1}\right)^t > 0$$

es factible y se puede usar como punto inicial.

2.3.2. Obtención de una Solución Factible Estrictamente Positiva.

Consideremos nuevamente el problema

$$\begin{array}{ll} \text{mín} & c^t x \\ s.a. & Ax = b \\ & x \geqslant 0 \end{array} \qquad (P)$$

donde: $A_{m\times n}$, los vectores columna x, c en $\mathbb{R}^n$, $b \in \mathbb{R}^m$

La aplicación del algoritmo de Karmarkar requiere el conocimiento de un punto inicial factible estrictamente positivo, para comenzar las iteraciones. Ya vimos como la transformación originalmente propuesta por Karmarkar nos proporciona un punto inicial de este tipo. Veamos ahora otro método para generar puntos factibles interiores, si los hay.

Consideremos un vector arbitrario $\overline{x} > 0$ y sea $R = A\overline{x} - b$. Planteamos el siguiente problema de programación lineal:

$$\begin{array}{ll} \text{mín} & \lambda \\ s.a. & Ax - \lambda R = b \\ & x \geqslant 0, \lambda \geqslant 0 \end{array} \qquad (Q')$$

Obviamente $x = \overline{x}, \lambda = 1$ es una solución factible del problema anterior verificando:

a). $(\overline{x}, \lambda)$ es estrictamente positiva.

b). Si existe una solución factible "x"para el problema Q', el valor óptimo de la función objetivo de este problema Q' es $\lambda = 0$ y proporciona una solución estrictamente positiva para el problema P.

Tenemos pues un problema que podemos transformar en la forma requerida por el algoritmo de Karmarkar. Como además, dicho algoritmo genera soluciones factibles positivas, la solución obtenida en esta primera fase puede ser utilizada como punto inicial para el algoritmo de Karmarkar propiamente dicho. De esta forma, la resolución del problema P pasa por dos fases, una primera, llamada de factibilidad, en la que se genera un punto factible estrictamente positivo, y una segunda fase, de optimalidad, en la que se aplica el algoritmo al problema original. Es importante observar que este método de dos fases es usado, no necesariamente en la misma forma, por otras variantes del algoritmo de Karmarkar.

Ejemplo 2.3.1. *Usando el algoritmo de Karmarkar Resolver el problema*

$$\begin{array}{ll} \text{mín} & Z = -x_1 - 1{,}4x_2 \\ s.a. & x_1 + x_2 \geqslant 400 \\ & x_1 + 2x_2 \geqslant 580 \\ & x_1 \qquad \geqslant 300 \\ & x_1, x_2 \geqslant 0 \end{array}$$

Convirtiendo a la forma canónica de minimización e identificando la matriz A y los vectores c, b se tiene:

$$c = \begin{bmatrix} -1 \\ -1{,}4 \end{bmatrix}; \; A = \begin{bmatrix} -1 & -1 \\ -1 & -2 \\ -1 & 0 \end{bmatrix}; \; b = \begin{bmatrix} -400 \\ -580 \\ -300 \end{bmatrix}$$

los coeficientes de la matriz son:

$$A' = \begin{bmatrix} -1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ -1 & -2 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & -1 & -2 & 0 & 0 & 0 & 0 & 0 & 1 \\ -1 & -1{,}4 & 400 & 580 & 300 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$b' = \begin{bmatrix} -400 & -580 & -300 & -1 & -1.4 & 0 \end{bmatrix}^t$

$c' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}^t$

Los coeficientes de la matriz A'' son:

$$\begin{bmatrix} -1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -397 \\ -1 & -2 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -596 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -298 \\ 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & -1 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0{,}6 \\ -1 & -1{,}4 & 400 & 580 & 300 & 0 & 0 & 0 & 0 & 0 & -1227{,}6 \end{bmatrix}$$

$b'' = \begin{bmatrix} -400 & -580 & -300 & -1 & -1.4 & 0 \end{bmatrix}^t$

Finalmente

$c'' = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}^t$

los coeficientes de la matriz A''' son:

$$\begin{bmatrix} -1 & -1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & -397 & 400 \\ -1 & -2 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & -596 & 580 \\ -1 & 0 & 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & -298 & 300 \\ 0 & 0 & -1 & -1 & -1 & 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 0 & -1 & -2 & 0 & 0 & 0 & 0 & 0 & 1 & 0{,}6 & 1{,}4 \\ -1 & -1{,}4 & 400 & 580 & 300 & 0 & 0 & 0 & 0 & 0 & -1227{,}6 & 0 \end{bmatrix}$$

la solución es:

Y'$= \begin{bmatrix} 0{,}45643055 & 0{,}37344311 & \dots & 0{,}00207469 \end{bmatrix}^t$

y dividiendo las dos primeras componentes por 0,00207469 se obtiene.

$X = \begin{bmatrix} 2{,}1999999 & 1{,}7999999 & \dots & \dots \end{bmatrix}^t$

Los valores de la solución exacta del problema inicial son :

$x_1 = 2{,}2$

$x_2 = 1{,}8$

Ejemplo 2.3.2. *Resolver el problema*

$$\begin{aligned} \text{mín} \quad & Z = -x_1 + 2x_2 \\ s.a. \quad & x_1 - 2x_2 + x_3 = 0 \\ & x_1 + x_2 + x_3 = 1 \\ & x_1, x_2, x_3 \geqslant 0 \end{aligned}$$

En el presente ejemplo se aplica el algoritmo de Karmarkar ilustrando su interpretación geométrica.

Definiendo valores iniciales

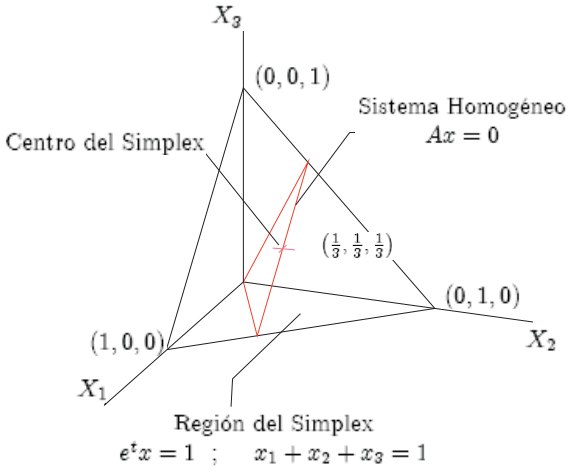
$\alpha = 0{,}9$;

$e = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ centro del simplex.

Dado un punto factible x que cumpla las condiciones

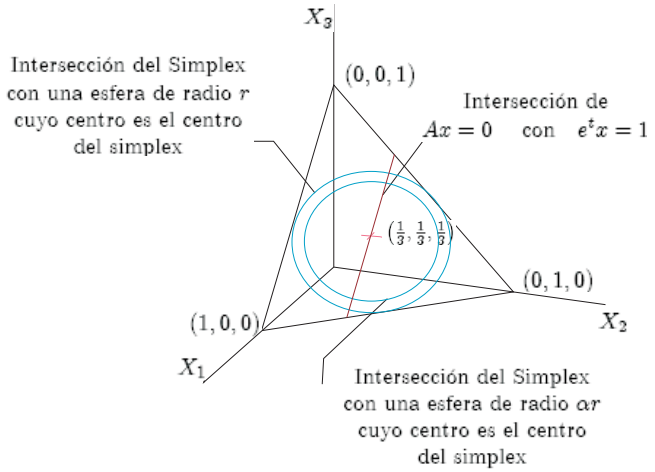
$Ax = 0$, $x_1 - 2x_2 + x_3 = 0$ y $x_1 + x_2 + x_3 = 1$ tal como $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$ que es el punto factible buscado entonces.

$$x = \begin{bmatrix} 0{,}3333 \\ 0{,}3333 \\ 0{,}3333 \end{bmatrix}; \; r = \sqrt{\frac{1}{n(n-1)}} = \sqrt{\frac{1}{6}} = 0{,}40825, \text{ entonces } Z = 0{,}33333$$



La intersecección de los sistemas $Ax = 0$ y $e^t x = 1$ será región factible donde se encuentra la solución.

La idea principal de Karmarkar es empezar desde un punto interior representado por el centro del simplex y después avanzar en dirección del gradiente proyectado para determinar un nuevo punto de aproximación. El punto debe ser estrictamente un punto interior, no debe estar en los limites del simplex.



Para garantizar este resultado, se inscribe una esfera con su centro coincidiendo con el centro del simplex, una esfera más pequeña con radio $\alpha r \quad (o < \alpha < 1)$ será un subconjunto de la esfera y cualquier punto en la intersección de la esfera más pequeña con el sistema homogéneo $Ax = 0$ será un punto interior. Por consiguiente podemos avanzar en este espacio restringido a lo largo del gradiente proyectado, para determinar el nuevo punto de aproximación, necesariamente mejorado.

Para que el procedimiento sea iterativo, necesitamos encontrar una forma de llevar el punto de la solución hacia el centro del simplex. Karmarkar satisface este requerimiento proponiendo una transformación proyectiva $y = T(x) = \dfrac{D^{-1}x}{e^t D^{-1}x}$. El problema transformado tiene el mismo formato que el problema original.

Primera iteración:

$$D = \begin{bmatrix} x_1 & 0 & 0 \\ 0 & x_2 & 0 \\ 0 & 0 & x_3 \end{bmatrix} \Rightarrow D = \begin{bmatrix} 0{,}3333 & 0 & 0 \\ 0 & 0{,}3333 & 0 \\ 0 & 0 & 0{,}3333 \end{bmatrix}; A = \begin{bmatrix} 1 & -2 & 1 \end{bmatrix}$$

$$\widetilde{A} \leftarrow AD \Rightarrow \widetilde{A} = \begin{bmatrix} 0{,}3333 & 0{,}3333 & 0{,}3333 \end{bmatrix}$$

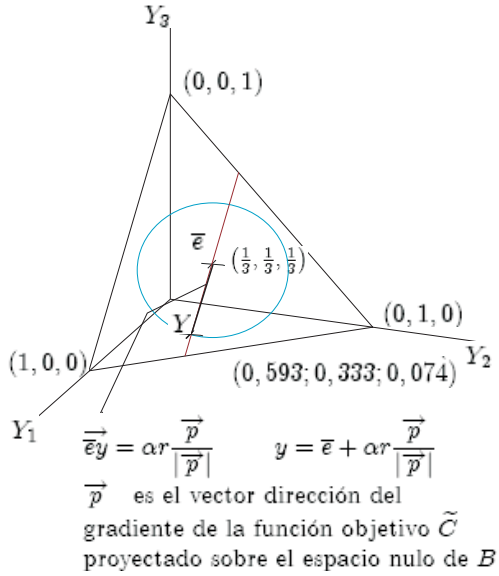
$$\widetilde{C} \leftarrow Dc \Rightarrow \widetilde{C} = \begin{bmatrix} -0{,}33333 & 0{,}66666 & 0 \end{bmatrix}^t$$

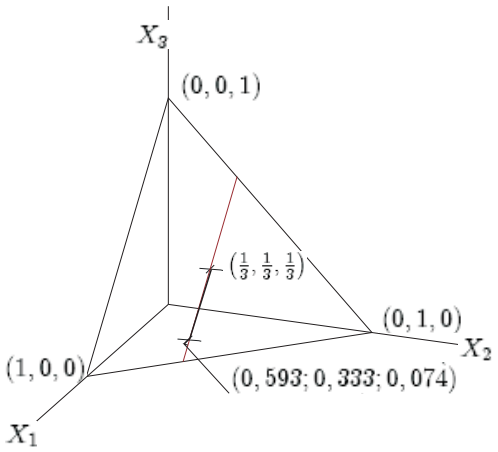
$$B \leftarrow \begin{bmatrix} \widetilde{A} \\ e^t \end{bmatrix} \Rightarrow B = \begin{bmatrix} 0{,}33333 & -0{,}66666 & 0{,}33333 \\ 1{,}00000 & 1{,}00000 & 1{,}00000 \end{bmatrix}$$

$$p \leftarrow \left(I - B^t \left(BB^t\right)^{-1} B\right) \widetilde{C} \Rightarrow p = \begin{bmatrix} 0{,}166666 \\ 0{,}000000 \\ -0{,}166666 \end{bmatrix}$$

Después de cada iteración, podemos calcular los valores de las variables x originales a partir de la solución y, realizando una transformación inversa, obtendremos un valor x que mejora la función objetivo Z.

$$x = \frac{Dy}{e^t Dy} \quad y = \begin{bmatrix} 0{,}593141 \\ 0{,}333333 \\ 0{,}073525 \end{bmatrix} \quad x = \begin{bmatrix} 0{,}593141 \\ 0{,}333333 \\ 0{,}073526 \end{bmatrix} \quad Z = 0{,}073526$$





El nuevo punto de la solución ya no estará en el centro del simplex. Para que el procedimiento sea iterativo, necesitamos llevar el nuevo punto de la solución hacia el centro del simplex. Karmarkar satisface este requerimiento proponiendo una transformación proyectiva $y = T(x) = \frac{D^{-1}x}{e^t D^{-1}x}$, la transformación traza el espacio x sobre el espacio y únicamente

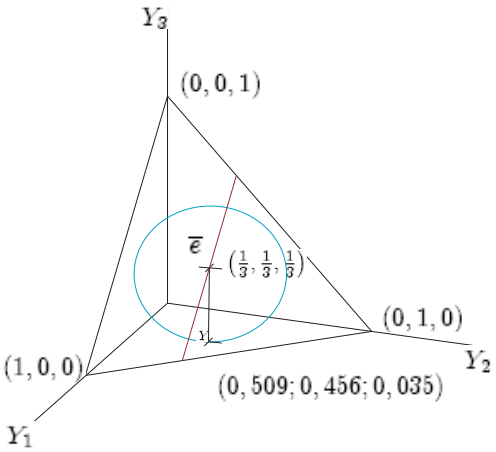
Segunda iteración

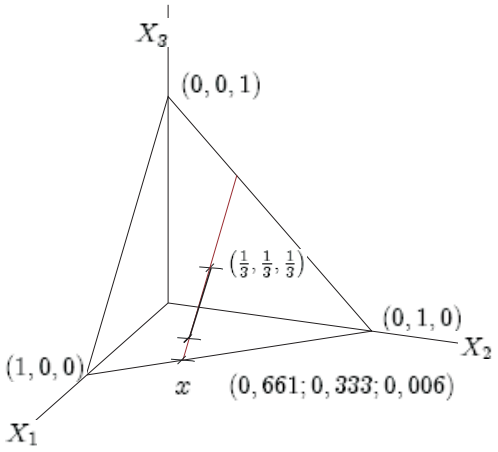
$$D = \begin{bmatrix} 0{,}593141 & 0 & 0 \\ 0 & 0{,}33333 & 0 \\ 0 & 0 & 0{,}073526 \end{bmatrix}; \widetilde{A} = \begin{bmatrix} 0{,}593141 & -0{,}66666 & 0{,}073526 \end{bmatrix}$$

$$\widetilde{C} = \begin{bmatrix} -0{,}593141 & 0{,}66666 & 0 \end{bmatrix}^t$$

$$B = \begin{bmatrix} 0{,}593141 & -0{,}66666 & 0{,}07352 \\ 1{,}00000 & 1{,}00000 & 1{,}00000 \end{bmatrix}$$

$$p = \begin{bmatrix} 0{,}028508 \\ 0{,}020013 \\ -0{,}048521 \end{bmatrix}; y = \begin{bmatrix} 0{,}508703 \\ 0{,}456443 \\ 0{,}034855 \end{bmatrix}; x = \begin{bmatrix} 0{,}661052 \\ 0{,}333333 \\ 0{,}005615 \end{bmatrix}; Z = 0{,}0056145$$





Tercera Iteración

$$D = \begin{bmatrix} 0{,}6610522 & 0 & 0 \\ 0 & 0{,}33333 & 0 \\ 0 & 0 & 0{,}005615 \end{bmatrix};$$

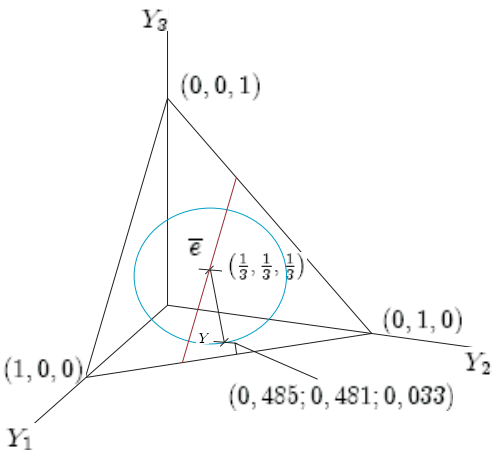
$$\widetilde{A} = \begin{bmatrix} 0{,}6610522 & -0{,}666666 & 0{,}005615 \end{bmatrix}$$

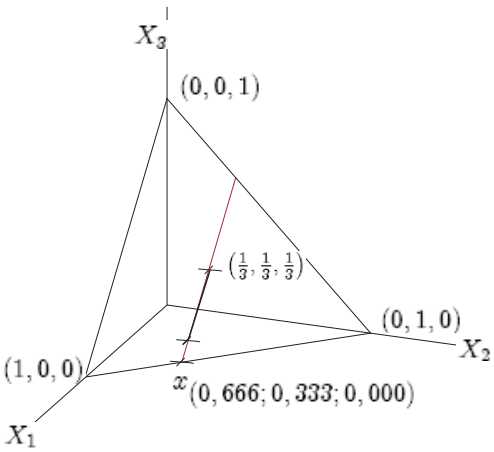
$$\widetilde{C} = \begin{bmatrix} -0{,}6610522 & 0{,}666666 & 0 \end{bmatrix}^t$$

$$B = \begin{bmatrix} 0{,}661052 & -0{,}66666 & 0{,}00561 \\ 1{,}00000 & 1{,}00000 & 1{,}00000 \end{bmatrix}$$

$$p = \begin{bmatrix} 0{,}001895 \\ 0{,}001848 \\ -0{,}003743 \end{bmatrix}; \; y = \begin{bmatrix} 0{,}4852322 \\ 0{,}4814265 \\ 0{,}0333414 \end{bmatrix}; \; x = \begin{bmatrix} 0{,}6662778 \\ 0{,}3333333 \\ 0{,}0003888 \end{bmatrix}$$

$$Z = 3{,}888347779250623 * 10^{-4}$$





Cuarta Iteración

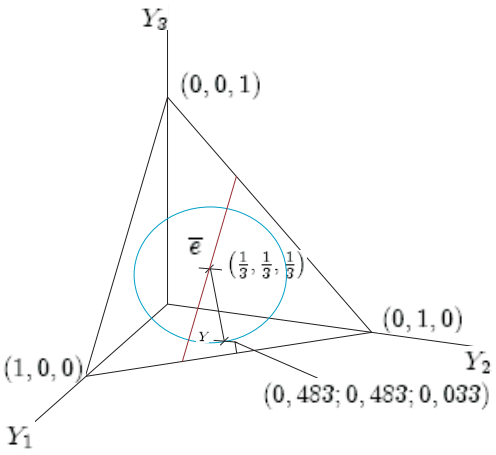
$$D = \begin{bmatrix} 0{,}666277 & 0 & 0 \\ 0 & 0{,}33333 & 0 \\ 0 & 0 & 0{,}00038 \end{bmatrix}; \widetilde{A} = \begin{bmatrix} 0{,}666277 & -0{,}666666 & 0{,}00038 \end{bmatrix}$$

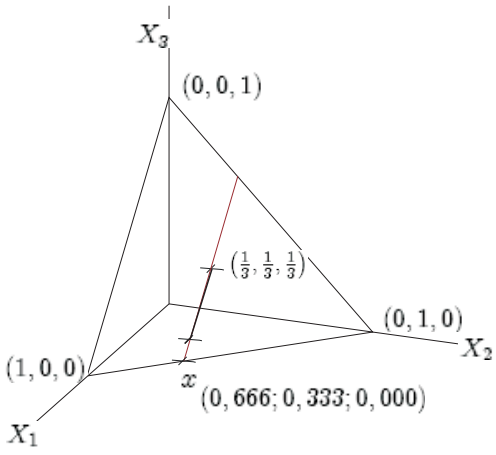
$$\widetilde{C} = \begin{bmatrix} -0{,}666277 & 0{,}666666 & 0 \end{bmatrix}^t$$

$$B = \begin{bmatrix} 0{,}666277 & -0{,}66666 & 0{,}00038 \\ 1{,}00000 & 1{,}00000 & 1{,}00000 \end{bmatrix}$$

$$p = 1{,}0 * 10^{-0{,}003} \begin{bmatrix} 0{,}12972 \\ 0{,}12949 \\ -0{,}25922 \end{bmatrix}; y = \begin{bmatrix} 0{,}483465 \\ 0{,}483202 \\ 0{,}033333 \end{bmatrix}; x = \begin{bmatrix} 0{,}666665 \\ 0{,}3333333 \\ 0{,}000002 \end{bmatrix}$$

$Z = 2{,}682350849925186 * 10^{-6}$.





los valores de x son

N	x
1	$6{,}666648 * 10^{-3}$
2	$3{,}333333 * 10^{-3}$
3	$1{,}849932 * 10^{-6}$

y la función objetivo tiende a cero.

2.4. Variantes del Algoritmo de Karmarkar

Desde la publicación del algoritmo de Karmarkar, varios autores han propuesto métodos que de alguna forma se basan en las ideas de aquél. La principal motivación de estos métodos es el poder obviar los requerimientos del algoritmo original de Karmarkar y también disminuir el volumen de los cálculos a realizar en cada iteración.

Muchas de estas variantes introducen realmente pocas modificaciones con respecto a dicho algoritmo. Acontinuación estudiamos la variante de Barnes.

2.4.1. Variante de Barnes

Esta variante se aplica a problemas de la forma:

$$\begin{array}{ll} \text{mín} & c^t x \\ s.a. & Ax = b \\ & x \geqslant 0 \end{array} \tag{P1}$$

donde: $A_{m\times n}$, los vectores columna x, c en $\mathbb{R}^n$, $b \in \mathbb{R}^m$
y no requiere que el valor óptimo de la función objetivo sea conocido, pero si es necesario disponer de un punto inicial factible estrictamente positivo. Como es usual, c y x son vectores n-dimensionales y A una matriz de orden $m \times n$ y de rango m, cuya j-ésima columna se denotará a_j. Suponemos que P1 tiene soluciones básicas no degeneradas y que su dual:

$$\begin{array}{ll} \text{mín:} & b^t \lambda \\ s.a. & A^t \lambda \leqslant c \end{array} \tag{D1}$$

tiene soluciones básicas no degeneradas. Esto significa que b no puede ser expresado como una combinación positiva de menos de m columnas de A y que como máximo m de las ecuaciones:

$$c_i - a_i^t \lambda = 0 \quad , \quad i = 1, ..., n$$

pueden ser satisfechas simultaneamente. Claramente, P1 y D1 permanecen no degenerados si b y c son sometidos a pequeñas perturbaciones. De hecho, si P1 es no degenerado existe un número $\varepsilon_1 > 0$ tal que cualquier solución factible de P1 tiene al menos m componentes mayores que ε_1. Y análogamente, si D1 es no degenerado, existe un número $\varepsilon_2 > 0$ tal que como máximo m de las desigualdades:

$$\left| c_i - a_i^t \lambda \right| < \varepsilon_2 \quad , \quad i = 1, ..., n$$

Puede verificarse simultaneamente. Los números ε_1 y ε_2 pueden tomarse de forma que las condiciones anteriores se sigan verificando cuando b y c son sometidos a pequeñas perturbaciones.

Sea $y = (y_1, ..., y_n)^t$ una solución factile de P1 tal que $y > 0$. Sea $0 < R < 1$. Se verifica que el elipsoide

$$\sum_{i=1}^{n} \frac{(x_i - y_i)^2}{y_i^2} \leqslant R^2 \tag{2.4}$$

está contenido en el interior de $\{x \in \mathbb{R}^n \ / \ x \geqslant 0\}$. En efecto, si $x_j \leqslant 0$ para algún j, tendriamos que:

$$\sum_{i=1}^{n} \frac{(x_i - y_i)^2}{y_i^2} \geqslant \frac{(x_j - y_j)^2}{y_j^2} \geqslant 1 > R^2.$$

Esto implica que podemos obtener una solución factible de P1, satisfaciendo $c^t x < c^t y$, resolviendo el siguiente problema:

$$\begin{aligned} &\text{mín} \quad c^t x \\ &s.a. \quad Ax = b \end{aligned}$$

$$\sum_{i=1}^{n} \frac{(x_i - y_i)^2}{y_i^2} \leqslant R^2 \qquad \text{(P1')}$$

Para resolver P1', consideremos $\lambda = (\lambda_1, ..., \lambda_m)^t$ vector de multiplicadores de Lagrange correspondiente a las restricciones $Ax = b$, (con $A_{m\times n}$)y sea $D = diag\,(\ y_1, ..., y_n)$. Tendremos que:

$$\begin{aligned} c^t y - c^t x &= \left\{c - A^t\lambda\right\}(y - x) = \left[D\left(c - A^t\lambda\right)\right]^t D^{-1}(x - y) \\ &\leqslant \left\|D\left(c - A^t\lambda\right)\right\| \left\|D^{-1}(x - y)\right\| \leqslant \left\|D\left(c - A^t\lambda\right)\right\| .R \end{aligned} \qquad (2.5)$$

Las últimas desigualdades se deducen de las desigualdades de Shwartz y de la segunda restricción de P1'. La igualdad será si se verifica:

$$D\left(c - A^t\lambda\right) = \gamma D^{-1}(x - y) \qquad (2.6)$$

para alguna constante, y si

$$\left\|D^{-1}(x - y)\right\| = R$$

condiciones que implican:

$$\gamma = \frac{\|D(c - A^t\lambda)\|}{R}$$

y sustituyendo en 2.6 llegamos a:

$$x = y - \frac{RD^2(c - A^t\lambda)}{\|D(c - A^t\lambda)\|}$$

por otra parte, la condición $Ax = Ay = b$ implica , a partir de 2.6 que

$$AD^2\left(c - A^t\lambda\right) = \gamma A(x - y) = b - b = 0$$

de donde deducimos

$$\lambda = \left(AD^2A^t\right)^{-1} AD^2c$$

Si ahora escribimos 2.5 como $c^t x \geqslant c^t y - R\left\|D(c - A^t\lambda)\right\|$, el mínimo se dará precisamente para la igualdad, con lo cual, x y λ vendrán dados por las expresiones anteriores.

De esta forma, podemos construir el siguiente algoritmo:

Algoritmo 2.4.1. *Sea* $x^0 > 0$, *satisfaciendo* $Ax^0 = b$, *con* $A_{m\times n}, x^0 \in \mathbb{R}^n$. *En general, dado* $x^k > 0$, *definimos*

$$D_k = diag\left(x_1^k, ..., x_n^k \right)$$

y hallamos $x^{k+1} > 0$ *(por la segunda restricción de P1'), de la siguiente forma:*

$$x^{k+1} = x^k - \frac{RD_k^2 \left(c - A^t\lambda_k\right)}{\left\|D_k \left(c - A^t\lambda_k\right)\right\|} \tag{2.7}$$

Siendo $\lambda_k = \left(AD_k^2A^t\right)^{-1} AD_k^2c$.

A continuación se resuelve el ejemplo 2.1.1 con el algoritmo de Barnes

Iteración 1		Iteración 2	
Barnes	Karmarkar	Barnes	Karmarkar
0,4060	0,431	0,4879	0,4936
0,0939	0,0699	0,0120	0,0064
0,2812	0,286	0,2975	0,2987
0,2187	0,214	0,2024	0,2013

Iteración 3		Iteración 4	
Barnes	Karmarkar	Barnes	Karmarkar
0,4987	0,4995	0,4998	0,50
0,0012	0,0005	0,0001	0,00
0,2997	0,2999	0,2999	0,30
0,2002	0,2001	0,2000	0,20

Cuadro 2.1: Comparación de los Algoritmos de Karmarkar y Barnes

En este ejemplo el algoritmo de Karmarkar se muestra más efectivo que el algoritmo de Barnes sin embargo la implementación en código de programa del algoritmo de Barnes es más sencilla que la implementación de Karmarkar. El algoritmo de Barnes es sencible a la elección del punto inicial, en este caso ambos algoritmos tienen como punto inicial $(0,25\ 0,25\ 0,25\ 0,25)^t$.

Teorema 2.4.1. *Si el problema P1 tiene solución óptima acotada, la sucesión* $\left\{x^k\right\}$ *producida por el algoritmo 2.4.1 converge a una solución óptima de P1, esto es, a un punto extremo de las restricciones definidas por* $Ax = b, x \geqslant 0$, *con* $A_{m\times n}$, $x, b \in \mathbb{R}^m$

Demostración. Ya hemos visto anteriormente que, en la forma en que se definen las iteraciones sucesivas del algoritmo 2.4.1, se ha de cumplir que:

$$c^t x^{k+1} = c^t x^k - R \left\| D_k \left(c - A^t \lambda_k \right) \right\|, \quad \forall\ k$$

Como los números $c^t x^k$ forma una sucesión decreciente y acotada inferiormente, tendremos que $\left\{ c^t x^k \right\}$ converge. Esto implica que:

$$\lim_{k \to \infty} \left\| D_k \left(c - A^t \lambda_k \right) \right\| = 0.$$

Sea ahora $\varepsilon > 0$ satisfaciendo $\varepsilon < \varepsilon_1$ y $\varepsilon < \varepsilon_2$ y escojamos k lo suficientemente grande para que

$$\left\| D_r \left(c - A^t \lambda_r \right) \right\| < \varepsilon^2 \text{ para } r \geqslant k$$

Existen entonces $n - m$ valores de i para los cuales

$$\left| c_i - a_i^t \lambda_r \right| > \varepsilon_2 > \varepsilon \tag{2.8}$$

Supongamos que $r \geqslant k$ es fijo. Sin pérdida de generalidad podemos suponer que la desigualdad anterior se verifica para $i = m+1, ..., n$.
Para cada i tenemos:

$$\begin{aligned} x_i^r \left| c_i - a_i^t \lambda_r \right| &\leqslant \left[\sum_{j=1}^{n} \left(x_j^r \right)^2 \left(c_j - a_j^t \lambda_r \right)^2 \right]^{\frac{1}{2}} \\ &= \left\| D_r \left(c - A^t \lambda_r \right) \right\| < \varepsilon^2 \end{aligned} \tag{2.9}$$

y por 2.8 tendremos

$$r_i^r < \varepsilon \ , \ i = m+1, ..., n \tag{2.10}$$

Como P1 es no degenerado y $Ax^r = b$, x^r tiene al menos m componentes mayores estrictamente que ε_1 y por ello, mayores estrictamente que ε, luego, teniendo en cuenta 2.9 se ha de verificar:

$$x_i^r > \varepsilon_1 \quad , \quad i = 1, ..., m \tag{2.11}$$

Por ser $\varepsilon_1 > \varepsilon$ y por 2.9, se cumplirá también que:

$$\left| c_i - a_i^t \lambda_r \right| < \varepsilon \quad , \quad i = 1, ..., m$$

Esta condición implica que los vectores $a_1, ...a_m$ son linealmente independientes si ε es lo suficientemente pequeño, ya que si fueran dependientes, el problema dual D1 con c sometido a una pequeña perturbación tendría una solución básica degenerada.

Denotemos por B la base factible $(a_1, ...a_m)$ y llamemos $x_B^r = (x_1^r, ..., x_m^r)^t$. Tendremos entonces:

$$x_B^r - B^{-1}b = -\sum_{i=m+1}^{n} x_i^r B^{-1} a_i \tag{2.12}$$

y por 2.10, se sigue que x_B^r está muy próximo a la solución factible básica $B^{-1}b$ para valores pequeños de ε, o, de manera equivalente, para k suficientemente grande y $r \geqslant k$. De esta forma, podemos asociar cada una de las soluciones factibles $x^k, r \geqslant k$, con una solución factible básica del problema P1. Si z e y son puntos extremos del conjunto definido por $Ax = b, x \geqslant 0$, tendremos que $\|z - y\| > \sqrt{2}\varepsilon_1$.por la hipótesis de no degeneración. Entonces, para ε suficientemente pequeño x^r se aproxima a un único punto extremo del conjunto definido por las restricciones del problema P1. Ahora, a partir de 2.4 tendremos

$$\frac{x_i^{r+1} - x_i^r}{\left(x_i^r\right)^2} \leqslant R^2$$

lo que significa que

$$x_i^{r+1} \geqslant (1 - R)\, x_i^r \quad \text{para cada} \quad i \text{ y cada } r$$

Esto significa que que si ε es suficientemente pequeño y $x_i^r > \varepsilon_1$, no podemos tener que $x_i^{r+1} < \varepsilon$. Entonces desde 2.10 y 2.11, deducimos que la solución factible básica asociada con x^r también estará asociada con x^{r+1} para r suficientemente grande. Así pues, 2.10 y 2.11 se cumplen para todo r suficientemente grande y para alguna base B que que seguiremos suponiendo $B = \{a_1, ..., a_m\}$. A las variables que satisfacen 2.10 la llamaremos no básicas a las que satisfacen 2.11 básicas. Ya que ε en 2.10 puede ser escogido arbitrariamente pequeño, tendremos que:

$$\lim_{r \to \infty} x_i^r = 0. \text{ para cada variable no básica } x_i^r \tag{2.13}$$

Sea ahora $c_B = (c_1, ...c_m)^t$ el vector m-dimencional correspondiente a las variables básicas, y sea $\overline{D} = diag\,(x_1^r, ..., x_m^r)$. Tenemos que:

$$AD_r^2 A^t = \sum_{i=1}^{n} \left(x_i^r\right)^2 a_i a_i^t = B\overline{D}_r^2 B^t + \sum_{i=m+1}^{n} \left(x_i^r\right)^2 a_i a_i^t = B\overline{D}_r^2 B^t + M_1$$

donde $\|M_1\| = \theta\left(\varepsilon^2\right)$ luego

$$\begin{aligned}
\left(AD_r^2 A^t\right)^{-1} &= \left(I + \left(B\overline{D}_r^2 B^t\right)^{-1} M_1^{-1}\right)^{-1} \left(B\overline{D}_r^2 B^t\right)^{-1} \\
&= \left[I - \left(B\overline{D}_r^2 B^t\right)^{-1} M_1 + \left(\left(B\overline{D}_r^2 B^t\right)^{-1} M_1\right)^2 - ...\right] \left(B\overline{D}_r^2 B^t\right)^{-1} \\
&= \left(B\overline{D}_r^2 B^t\right)^{-1} + M_2
\end{aligned}$$

donde $\|M_2\| = \theta(\varepsilon^2)$.
Denotemos por N la matriz de orden $m \times (n-m)$ siguiente:
$N = \left[\left(x_{m+1}^r\right)^2 a_{m+1}, ..., \left(x_n^r\right)^2 a_n \right]$ y sea $c_N = (c_{m+1}, ...c_n)$ tendremos que
$AD_r^2 c = B\overline{D}_r^2 c_B + Nc_N$ y

$$\lambda_r = \left(AD_r^2 A^t\right)^{-1} AD_r^2 c = \left(B\overline{D}_r^2 B^t\right)^{-1} B\overline{D}_r^2 c_B + e_r = \left(B^t\right)^{-1} c_B + e_r$$

donde $\|e_r\| = \theta(\varepsilon^2)$. Y ya que podemos tomar $\varepsilon \to 0$ cuando $r \to \infty$ tendremos que

$$\lim_{r\to\infty} \lambda_r = \left(B^t\right)^{-1} c_B \tag{2.14}$$

Consideremos ahora la i-ésima componente en 2.7

$$x_i^{r+1} = x_i^r - \frac{\left(x_i^r\right)^2 (c_i - a_i^t \lambda_r)}{\|D_r (c - A^t \lambda_r)\|}$$

que prueba

$$\begin{cases} x_i^{r+1} > x_i^r & \text{si } c_i - a_i^t \lambda_r < 0 \\ x_i^{r+1} < x_i^r & \text{si } c_i - a_i^t \lambda_r > 0 \end{cases} \tag{2.15}$$

Si x_i^r es una variable no básica, es imposible que $c_i - a_i^t \lambda_r$ cambie de signo cuando aumenta r, para r suficientemente grande. En efecto, de la hipótesis de no degeneración junto con el hecho de que $c_i - a_i^t \lambda_r$ no puede cambiar de signo sin hacerse muy próximo a cero. Luego tenemos, a partir de 2.13 y 2.15 que $c_i - a_i^t \lambda_r > 0$ para cada variable no básica si r es suficientemente grande. Sea ahora $x_B = B^{-1}b$. De 2.12 y 2.13 se tiene

$$x_B^r \to x_B \text{ cuando } r \to \infty.$$

Para completar la demostración del teorema, probaremos que x_B es una solución básica factible y óptima de P1. Tenemos que:

$$c_B^t x_B = c_B^t B^{-1} b = b^t \lambda \text{ donde } \quad \lambda = \left(B^t\right)^{-1} c_B$$

Entonces el vectores $x = (x_B, 0, ..., 0)^t$ y λ originan que las funciones objetivo primal y dual (P1 y D1) coincidan. Además se cumple $x^t (c - A^t \lambda) = 0$. Finalmente, tenemos:

$$c_i - a_i^t \lambda = \lim_{r\to\infty} \left(c_i - a_i^t \lambda_r\right) \geqslant \varepsilon_2 > 0$$

para x_i no básica.
Luego $A^t \lambda \leqslant c$ entonces λ es factible para el problema dual. Es decir, x y λ satisfacen la condición necesaria y suficiente para que x sea una solución básica de P1. □

Teorema 2.4.2. *Sea x^* una solución del problema P1. La sucesión $\left\{x^k\right\}$ generada por el algoritmo 2.4.1 satisface:*

$$c^t x^{k+1} - c^t x^* \leqslant \left[1 - \frac{R}{(n-m+\varepsilon_k)^{\frac{1}{2}}}\right]\left(c^t x^k - c^t x^*\right)$$

donde $\{\varepsilon_k\}$ es una sucesión de números positivos que converge a cero.

Demostración. Por la suposición de no degeneración, sabemos que $x^* = \lim x^k$ tiene $n-m$ componentes iguales a cero. Por simplicidad, supondremos que $x^* = (x_1^*, ..., x_m^*, 0, ..., 0)^t$. Entonces:

$$\begin{aligned} c^t x^k - c^t x^* &= \left[D_k\left(c - A^t\lambda_k\right)\right]^t D_k^{-1}\left(x^k - x^*\right) \\ &\leqslant \left\|D_k\left(c - A^t\lambda_k\right)\right\|(n-m+\varepsilon_k)^{\frac{1}{2}} \\ &= \frac{1}{R}\left(c^t x^k - c^t x^{k+1}\right)(n-m+\varepsilon_k)^{\frac{1}{2}} \end{aligned}$$

donde $\varepsilon_k = \sum_{i=1}^{m}\left[\frac{x_i^k - x_i^*}{x_i^k}\right]^2 \to 0$ cuando $k \to \infty$
La desigualdad puede ser escrita como:

$$c^t x^{k+1} - c^t x^* - \left(c^t x^k - c^t x^*\right) \leqslant -\frac{R}{(n-m+\varepsilon_k)^{\frac{1}{2}}}\left(c^t x^k - c^t x^*\right)$$

que es equivalente a la tesis del teorema. □

Notemos que el anterior teorema proporciona información sobre la rapidez de convergencia del método, pero no prueba que tenga complejidad polinomial.

3

Aplicación

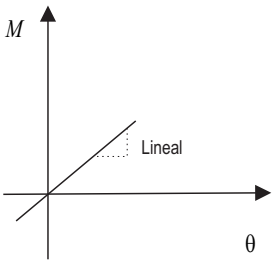
La programación lineal es empleada para formular y resolver problemas de diversos campos de la ingeniería. Por ejemplo en el campo de los recursos hidraúlicos, en el planeamiento de distribución urbana de agua, operación de reservorios, dotación de agua de cultivo (riego), minimizar el costo y cantidad de materiales en la ingeniería estructural.

En este capítulo presentamos dos ejemplos relativos al campo de la ingeniería, en ellos se aplica la programación lineal para su solución. La primera aplicación está referida a la obtención del peso mínimo de una estructura aporticada sujeta a fuerzas externas, este es un problema de interés en la ingeniería estructural;la segunda aplicación está relacionada a la distribución de dotación de agua de cultivo, este último es un problema de interés tanto en la ingeniería hidráulica como agrícola.

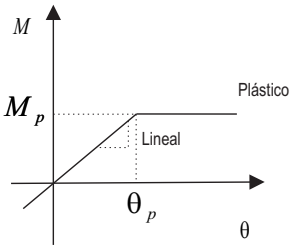
3.1. Aplicación a la Ingeniería Estructural

Definimos algunos términos que se usarán en el desarrollo de esta aplicación

- **Comportamiento Lineal:** Las secciones transversales de los elementos de la edificación vuelven a su posición inicial, cuando dejan de actuar las fuerzas externas, la relación momento - giro es lineal (ver figura 3.1(a)).
- **Comportamiento Plástico:** cuando una sección transversal del pórtico gira un valor mayor o igual al giro plástico el comportamiento es como se indica en la figura 3.1(b).
- **Rótula Plástica:** La sección de una viga tiene el comportamiento plástico a partir de una deformación conocida como deformación de fluencia (giro plástico de la sección). Esta deformación se representa por θ_p.



(a) Comportamiento Lineal



(b) Comportamiento Plástico

Figura 3.1: Comportamiento del material

- **Momento flector:** Es el momento que se produce en una sección transversal de un elemento de una estructura debido a las fuerzas que actuan en ella. Cuando este momento produce el giro θ_p toma el nombre de momento plástico.

Se considera que el trabajo de deformación interna en un punto de la estructura se calcula como $U_j = M_j\theta_j$ y el trabajo realizado por una carga externa se obtiene multiplicando la carga por el desplazamiento que produce.

Se debe cumplir que

$$\sum_{i=1}^{n} U_{int} \geqslant \sum_{j=1}^{m} E_{ext}.$$

Donde:
$\sum_{i=1}^{n} U_{int}$: denota la suma de los trabajos producidos por fuerzas internas.
$\sum_{i=1}^{m} E_{ext}$: Denota la suma de los trabajos producidos por fuerzas externas.

El ejemplo de aplicación, ilustra la obtensión del peso mínimo de una edificación cuando esta cumple las siguientes hipótesis:

a) Se considera unicamente cargas puntuales.

b) Los puntos en que pueden producirse rótulas plásticas son fijos y corresponden a nudos de la estructuras o a puntos de aplicación de cargas.

c) Las cargas crecen monotónicamente hasta el colapso.

d) La geometría de la estructura se considera inalterable.

e) Se considera que el colapso se produce únicamente por efecto de los esfuerzos de flexión.

f) Se considera que el área de la sección transversal es proporcional al momento plástico.

Si se considera que el momento plástico M_p es proporcional al área de la sección transversal se tiene

$$F_{obj} = \rho \sum_{i=1}^{n} A_i l_i = \rho k \sum_{i=1}^{n} M_{p_i} l_i,$$

entonces el problema consiste en minimizar

$$F_{obj} = \sum_{i=1}^{n} M_{p_i} l_i$$

Donde:

F_{obj} : función objetivo.

M_{p_i} : Momento plástico de la sección i.

l_i : longitud del elemento i.

Se tiene un pórtico rígido como el que se muestra en la figura (3.2). El momento plástico en la viga es representado por $M_b = x_2$ y el momento en la columna es denotado por $M_c = x_1$, las fuerzas $F_1 = P$ y $F_2 = 2P$, consideremos $P = 1$.

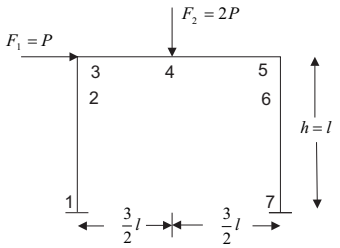


Figura 3.2: Pórtico Viga Columna

Por las hipótesis mencionadas anteriormente, los puntos de desarrollo del momento plástico en la estructura son enumerados desde 1 hasta 7 en la figura (3.2). En las figuras (3.3) (a, b. c, d, e, f) se indican cuando el pórtico será inestable, a estos estados se les conoce como mecanismo de colapso. El trabajo interno (U) que puede soportar el pórtico debe ser mayor que el trabajo externo (E) para los diversos mecanismos de colapso de la estructura. Los casos de Colapso plástico que deben considerarse y las ecuaciones asociadas a ellos son

$$\begin{aligned}
&\text{a).} && 4x_1 && \geqslant 1 \\
&\text{b).} && 4x_1 + 2x_2 && \geqslant 4 \\
&\text{c).} && 2x_1 + 2x_2 && \geqslant 3 \\
&\text{d).} && 4x_2 && \geqslant 3 \\
&\text{e).} && 2x_1 + 4x_2 && \geqslant 4 \\
&\text{f).} && 2x_1 + 2x_2 && \geqslant 1
\end{aligned}$$

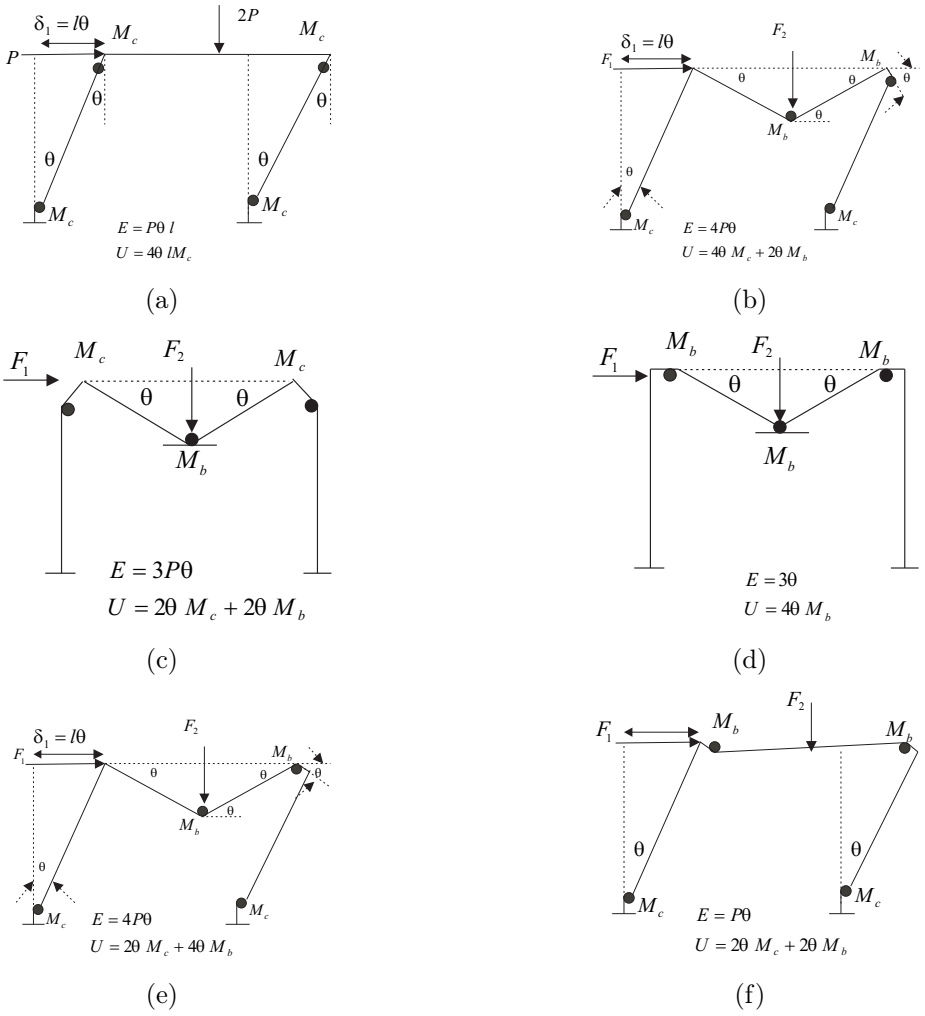


Figura 3.3: Posibles mecanismos de colapso del Pórtico

La función objetivo que permite determinar el peso mínimo vendrá definida por la ecuación $F = 2x_1 + 3x_2$. Por tanto el problema lineal será:

$$\begin{array}{lll} \text{mín} & F = 2x_1 + 3x_2 \\ s.a. & 4x_1 & \geqslant 1 \\ & 4x_1 + 2x_2 & \geqslant 4 \\ & 2x_1 + 2x_2 & \geqslant 3 \\ & 4x_2 & \geqslant 1 \end{array}$$

Las desigualdades, se obtienen comparando los trabajos producidos por las fuerzas internas y externas del pórtico, los resultados se indican en las figuras 3.3 (a,b,c,d,e,f).

Así tenemos:

$$\begin{array}{ll} \text{mín} & c^t x \\ s.a. & Ax \geqslant b \\ & x \geqslant 0 \end{array} \qquad \begin{array}{ll} \text{mín} & F = 2x_1 + 3x_2 \\ s.a. & 4x_1 \qquad\quad \geqslant 1 \\ & 4x_1 + 2x_2 \geqslant 4 \\ & 2x_1 + 2x_2 \geqslant 3 \\ & \qquad\quad 4x_2 \geqslant 1 \\ & x_1, x_2 \geqslant 0 \end{array}$$

Acondicionando el problema determinamos el problema dual,

$$\begin{array}{ll} \text{máx} & b^t u \\ s.a. & A^t u \leqslant c \\ & u \geqslant 0 \end{array} \qquad \begin{array}{ll} \text{máx} & F' = y_1 + 4y_2 + 3y_3 + y_4 \\ s.a. & 4y_1 + 4y_2 + 2y_3 \quad \leqslant 2 \\ & \qquad 2y_2 + 2y_3 + 4y_4 \leqslant 3 \\ & y_1, y_2, y_3, y_4 \geqslant 0 \end{array}$$

Introduciendo las variables de holgura a fin de conseguir una base inicial, se obtiene:

$$\begin{array}{ll} \text{máx} & F' = y_1 + 4y_2 + 3y_3 + y_4 + y_5 + y_6 \\ s.a. & 4y_1 + 4y_2 + 2y_3 \quad + y_5 \qquad = 2 \\ & \qquad 2y_2 + 2y_3 + 4y_4 \quad + y_6 = 3 \\ & y_1, y_2, y_3, y_4, y_5, y_5 \geqslant 0 \end{array}$$

Resolviendo el problema haciendo uso del método Simplex

	y_1	y_2	y_3	y_4	y_5	y_6	c
f	-1	-4	-4	-1	0	0	0
y_5	4	4	**2**	0	1	0	2
y_6	0	2	2	4	0	1	3

Cuadro 3.1: Tabla 1 del ejemplo

entra en la base la variable y_3 y sale y_5

	y_1	y_2	y_3	y_4	y_5	y_6	c
f	7	4	0	-1	2	0	4
y_3	2	2	1	0	1/2	0	1
y_6	-4	-2	0	**4**	-1	1	1

Cuadro 3.2: Tabla 2 del ejemplo

entra en la base la variable y_4 y sale y_6 y se obtiene la tabla

	y_1	y_2	y_3	y_4	y_5	y_6	c
F	6	7/2	0	0	7/4	1/4	17/4
y_3	2	2	1	0	1/2	0	1
y_4	−1	−1/2	0	1	−1/4	1/4	1/4

Cuadro 3.3: Tabla 3 del ejemplo

como en la función objetivo no hay coeficientes negativos el proceso se da por terminado. Así el valor mínimo es

$$F = 17/4.$$

Para resolver el problema del pórtico por el Algoritmo de Karmarkar debemos acondicionar previamente el problema asi se tiene:

$$\begin{array}{rr} \text{mín} & c^t x \\ s.a. & Ax \geqslant b \\ & x \geqslant 0 \end{array} \qquad \begin{array}{rr} \text{máx} & b^t \lambda \\ s.a. & A^t \lambda \leqslant c \\ & \lambda \geqslant 0 \end{array}$$

que en el òptimo se verifica

$$\text{mín}\, c^t x = \text{máx}\, b^t \lambda$$

así

$$\begin{array}{rr} \text{mín} & c^t x - b^t \lambda \\ s.a. & Ax \geqslant b \\ & A^t \lambda \leqslant c \\ & x, \lambda \geqslant 0 \end{array} \qquad \begin{array}{rr} \text{mín} & c^t x - b^t \lambda \\ s.a. & Ax - x' = b \\ & A^t \lambda + \lambda' = c \\ & x, x', \lambda, \lambda' \geqslant 0 \end{array}$$

el problema consiste en resolver:

$$\begin{array}{rrrrrrrrr} \text{mín} & 2x_1 & + 3x_2 & - \lambda_1 & & - \lambda_2 & & & \\ s.a. & 2x_1 & + 2x_2 & - x_3 & & & & = & 1 \\ & 4x_1 & & & - x_4 & & & = & 1 \\ & & & & & 2\lambda_1 & + 4\lambda_2 + \lambda_3 & = & 2 \\ & & & & & 2\lambda_1 & + \lambda_4 & = & 3 \end{array}$$

que haciendo la transformación indicada en (2.3) (transformación del problema general a la forma canónica de Karmarkar) se tiene:

$$\begin{array}{lllll} \text{mín} & 2y'_1 + 3y'_2 - y'_5 & & & \\ s.a. & 2y'_1 + 3y'_2 + y'_3 & & & y'_9 = 0 \\ & 4y'_1 \quad - 3y'_4 & & & y'_9 = 0 \\ & & & 2y'_5 & -2y'_9 = 0 \\ & & & 2y'_5 + 2y'_8 & y'_9 = 3 \end{array}$$

aplicando el algoritmo de Karmarkar se tiene como solución factible estrictamente positiva a 4,85. como se muestra en la tabla

N_{ite}	y'_1	y'_2	x_1	x_2	F
1	0,104	0,075	1,0	1,0	5
10	0,123	0,076	0,488	0,023	1,045
20	0,112	0,082	0,500	0,000	1,001

Cuadro 3.4: Tabla de Iteración del Algoritmo de Karmarkar

3.2. Aplicación a la dotación de agua de riego

Esta aplicación está referida a la producción agrícola del sub sector de riego bajo Caplina situado en Tacna, que en adelante será llamado valle Caplina la formulación del problema toma en cuenta un patrón de cultivo heterogéneo, conformado por cultivos permanentes (vid negra corriente), transitorios (papa, maíz amiláceo, hortalizas, maíz choclo, ají, tomate, haba y arveja), forrajeros (alfalfa y maíz chala) y forestales.

El valle Caplina se tipifica como una zona de menor desarrollo relativo con tendencia al estancamiento principalmente en las áreas de minifundio; sin embargo, su potencial agro ecológico es un factor determinante para impulsar complejos frutihorticolas con fines agroindustriales y para el consumo humano directo.

La programación Lineal es un método de asignación de recursos que busca la optimización de recursos limitados a un numero de actividades en competencia. El objetivo de esta aplicación es optimizar la cedula de cultivos acorde con las condiciones agro climáticas y económicas del valle Caplina, mediante la formulación de modelos de programación lineal, utilizando para esto el algoritmo de programación lineal de Karmarkar programado en MATLAB.

Con el modelo se pretende maximizar los beneficios del área del proyecto, sujeto a diferentes restricciones:

a). Agua

b). Mano de obra

c). Mercado

d). Suelo

Obteniendo una cedula de cultivo optima; con cultivos económicamente rentables y la cantidad de áreas a ser instaladas en una campaña agrícola.

La metodología empleada en para la optimización de la cédula de cultivo toma en cuenta:

a). Recopilación de información agroeconómica, suelo, disponibilidad de agua, requerimientos agroclimáticos de los cultivos.

b). Definición de cultivos a considerar teniendo en cuenta su adecuación al clima, demanda del producto en el mercado, rentabilidad, requerimiento de agua. A cada cultivo se le asigna una variable.

c). Formulación de las funciones objetivo de los modelos que maximicen los beneficios de la producción, mediante la rentabilidad económica de los cultivos.

d). Definición de la restricciones de los modelos como demanda y disponibilidad de agua, mano de obra requerida, demanda de mercado y disponibilidad de suelo.

e). Formulación de los Modelos.

f). Obtención de la cedula optimizada mediante la técnica de Karmarkar en programación lineal en MATLAB.

La función objetivo ha sido definida mediante la sumatoria de los productos de los Valores Netos de la Producción VNP_i y el área correspondiente a cada cultivo x_i.

$$\text{máx} \sum (VNP_i \, x_i)$$

La función objetivo maximiza los beneficios de la producción en ($US\$$) esta es representada por:

$$\begin{aligned}\text{máx } Z = \quad & 980x_1 + 759x_2 + 882x_3 + 847x_4 + \\ & 768x_5 + 169x_6 + 1077x_7 + 213x_8 + \\ & 186x_9 + 459x_{10} + 561x_{11} + 405x_{12}\end{aligned}$$

Las Restricciones del Modelo se elaboran teniendo en cuenta los siguientes criterios

a). AGUA. La restricción del agua es la sumatoria del producto del volumen de agua expresado en m3/mes-Ha y el área de cada cultivo(Ha), esta sumatoria debe ser menor o igual al volumen de agua disponible del mes correspondiente.

- DEMANDA. La demanda bruta de agua ha sido elaborada teniendo en cuenta la condición actual de un 31 % de eficiencia de riego y las proyecciones del Plan de Desarrollo Agropecuario Tacna 1995-2015, en la cual se espera una eficiencia de riego de 46 % con el sistema de riego por gravedad mejorado.
- DISPONIBILIDAD. La disponibilidad de agua para los fines de restricción se ha definido como la suma de las descargas del río Caplina al 75 % de persistencia, menos la demanda poblacional (0,05 m3/s) y las condiciones de aporte.

b). MANO DE OBRA

-. REQUERIMIENTO. Según el censo de población y vivienda de 1993 la población urbana entre 20 y 24 años de edad se estima en 22239 personas. Este dato se ha tomado como restricción de la disponibilidad de mano de obra para la producción de los cultivos.

c). MERCADO.

-. AREAS MÁXIMAS. Las superficies de cultivo por restricciones de mercado es la cantidad máxima a sembrarse en hectáreas y ha sido determinado en el Plan de Desarrollo Agropecuario Tacna 1995-2015, considerando las demandas locales, regionales e industriales y la no saturación del mercado interno.

d). SUELO. La restricción de suelo es la sumatoria de todas las áreas de los diferentes cultivos a sembrarse y debe ser menor o igual al área disponible del terreno, que según el padrón de regantes el área regable es de 1253 Ha, esta área como máximo puede duplicar su uso en un año(2506 Ha).

La matriz de restricciones en MATLAB muestra en el Anexo 5.2.2, a continuación se entrega en forma tabular la matriz de restricciones

RESTRICCIONES	Nº	X1	X2	X3	X4	X5	X6	X7	X8	X9	X10	X11	X12		
	1	2200	3000	3400	0	0	0	3400	0	0	0	0	0	< =	2127721
A	2	1987	2710	3071	0	0	0	2890	0	0	0	0	0	< =	2195666
	3	2090	2850	3230	0	0	0	2850	0	0	0	0	1520	< =	2058083
G	4	1597	2177	2468	0	1161	0	0	1161	1161	0	0	2032	< =	1879701
	5	1320	1800	2040	0	1680	0	0	1992	1680	960	1080	2280	< =	2021121
U	6	1011	1379	1563	0	1931	0	0	1931	1655	1287	1103	1563	< =	2092262
	7	1100	1500	1700	0	1700	0	0	1800	2000	2100	1800	1400	< =	2160397
A	8	1320	1800	2040	0	1680	0	0	1440	1440	1920	2520	0	< =	2173789
	9	1544	2105	2385	2245	0	842	842	0	0	1684	2806	0	< =	2143584
(m^3/mes-Ha)	10	1980	2700	3060	3780	0	2520	2160	0	0	0	2520	0	< =	2452879
	11	2076	2831	3208	3963	0	3963	3585	0	0	0	0	0	< =	2993242
	12	2310	3150	3570	4410	0	4200	3780	0	0	0	0	0	< =	3155155
	13	20	13	4	0	0	0	18	0	0	0	0	0	< =	22239
MANO	14	23	15	4	0	0	0	26	0	0	0	0	0	< =	22239
	15	14	12	2	0	0	0	26	0	0	0	0	23	< =	22239
	16	9	5	4	0	41	0	0	29	21	0	0	21	< =	22239
DE	17	8	4	3	0	37	0	0	28	20	28	40	22	< =	22239
	18	7	4	2	0	29	0	0	20	16	26	34	45	< =	22239
	19	10	5	4	0	34	0	0	39	21	19	27	46	< =	22239
OBRA	20	16	7	3	0	32	0	0	38	20	21	21	0	< =	22239
	21	8	5	2	13	0	29	28	0	0	38	36	0	< =	22239
(Jor/Ha)	22	9	4	3	13	0	33	28	0	0	0	40	0	< =	22239
	23	17	11	2	17	0	19	25	0	0	0	0	0	< =	22239
	24	8	10	2	16	0	29	19	0	0	0	0	0	< =	22239
	25	1	0	0	0	0	0	0	0	0	0	0	0	< =	120
M	26	0	1	0	0	0	0	0	0	0	0	0	0	< =	80
E	27	0	0	1	0	0	0	0	0	0	0	0	0	< =	190
R	28	0	0	0	1	0	0	0	0	0	0	0	0	< =	40
C	29	0	0	0	0	1	0	0	0	0	0	0	0	< =	260
A	30	0	0	0	0	0	1	0	0	0	0	0	0	< =	110
D	31	0	0	0	0	0	0	1	0	0	0	0	0	< =	125
O	32	0	0	0	0	0	0	0	1	0	0	0	0	< =	200
	33	0	0	0	0	0	0	0	0	1	0	0	0	< =	150
(Ha)	34	0	0	0	0	0	0	0	0	0	1	0	0	< =	50
	35	0	0	0	0	0	0	0	0	0	0	1	0	< =	130
	36	0	0	0	0	0	0	0	0	0	0	0	1	< =	190
SUELO(Ha)	37	1	1	1	1	1	1	1	1	1	1	1	1	< =	2506

Cuadro 3.5: Tabla de Restricciones

La solución que nos brinda el algoritmo corresponde a la cédula optimizada. Esta cédula optimizada representa el máximo beneficio para los usuarios del valle Caplina, considerando la rentabilidad económica de los cultivos y la asignación equitativa de los recursos (agua, mano de obra, suelo).

Los resultados obtenidos por el algoritmo presentado son:

N	x	F
1	120.00	117600.00
2	0.00	0.00
3	88.93	78440.14
4	14.34	12149.36
5	260.00	199680.00
6	0.00	0.00
7	125.00	134625.00
8	0.00	0.00
9	0.00	0.00
10	50.00	22950.00
11	129.28	72930.00
12	160.92	65175.79

La valor función óptima es :$703146,781

4

Conclusiones

El algoritmo del elipsoide tiene una convergencia polinomial si se considera cada instrucción como un paso del algoritmo, sin embargo no ha sido posible asegurar en este trabajo que el tiempo de ejecución sea polinomial, si se considera al tiempo de ejecución de cada caracter como paso del problema.

El simplex busca el máximo de la función objetivo en los vértices de la región factible, esto hace que el método sea lento a medida que se incrementa el tamaño del problema; el método de karmarkar encuentra la solución atravezando la región factible, lo que hace que el método se aproxime con mayor rapidez a la solución a medida que crece el tamaño del problema.

El tiempo de ejecución del algoritmo Karmarkar es polinomialmente acotable $O(np)$ en el peor de los casos el simplex es exponencial como queda demostrado en el capítulo 1 sección 2.5 .

Muchos problemas del campo de la ingeniería son modelados con métodos de programación lineal y tienen un tamaño del problema considerable, esto pone en vigencia algoritmos polinomiales como el de Karmarkar.

5

Anexos

5.1. Programación en Matlab del Algoritmo de Karmarkar

La programación del algoritmo de Karmarkar se realizo en Matlab 2007, dicho algoritmo lleva por nombre karjfs, el programa consta de seis (6) partes o estructuras:

- karjfs.m
- optkarc.m
- maxkarc.m
- minkarc.m
- kare.m
- karfo.m

Siendo el eje principal del algoritmo el archivo **karjfs.m** en el cual se encuentra estructurado los diferentes casos en que se presentan los problemas de programación lineal, encontrándose las llamadas a los demás archivos.

El archivo **optkar.m** contiene los valores de las constantes α y *eps*; α es una constante entre 0 y 1 ($0 < \alpha < 1$), eps es el valor del error donde se detiene el algoritmo.

El archivo **maxkarc.m** contiene la solución del caso particular de maximización en el cual el problema se encuentre directamente en la forma canónica de karmarkar $A * x = 0$.

El archivo **minkarc.m** contiene la solución del caso particular de minimización en el cual el problema se encuentre directamente en la forma canónica de karmarkar $A * x = 0$.

El archivo **kare.m** contiene la solución del caso particular de maximización y minimización en el cual el problema se encuentre en la forma estándar $A * x >= b$.

El archivo **karfo.m** contiene la solución del caso particular de maximización y minimización en el cual el problema se encuentre en la forma estándar de óptimo finito $A * x = b$.

El programa se ejecuta escribiendo karjfs en "≫ "karjfs

A continuación se indican el código de los algoritmos en matlab

5.1.1. Archivo karjfs

Listing 5.1: Programa karjfs.m

```
1
2  %******************
3  %*       karjfs.m
4  %*
5  %*
6  %*
7  %******************
8  clc
9  warning off;
10 fprintf('        Metodo de Karmarkar para "n" variables\n\n');
11 fprintf('Si esta en la forma canónica de Karmarkar     A*x=0 F=0 \n');
12 fprintf('Si esta en la forma                           A*x>=b F=1 \n');
13 fprintf('Si esta en la forma estandar de optimo finito A*x=0 F=0 \n\n');
14
15 F = input('Ingrese F:         \n');
16
17 fprintf('Para Minimizar P=0   \n');
18 fprintf('Para Minimizar P=0   \n');
19
20 P = input('Ingrese P:          ');
21
22 if F==0
23     A= input('nIngrese la matriz A:          \n\n');
24     c= input('nIngrese el vector columna de restricciones c:   \n\n');
25     optkar;
26     if P~=0
27         c=-c;
28         maxkarc;
29     else
30         minkarc;
31     end
32 elseif F==1
33     %Ingrese de esta manera "max c'*x<=b"
34     %Ingrese de esta manera "minx c'*x>=b"
35
```

```
36      A=input('nIngrese la matriz A:          \n\n');
37      b=input('\nIngrese el vector columna de constantes b:   \n\n');
38      c=input('\nIngrese el vector columna de restricciones c:   \n\n');
39
40      optkar;
41
42      if P~=0
43          c=-c;
44          A=-A;
45          b=-b
46      end
47          kare;
48      else
49          A=input('nIngrese la matriz A:          \n\n');
50          b=input('\nIngrese el vector columna de constantes b:   \n\n');
51          c=input('nIngrese el vector columna de restricciones c:   \n\n');
52
53      optkar;
54
55      if P~=0
56          c=-c;
57
58      end
59          karfo;
60      end
```

5.1.2. Archivo kare

Listing 5.2: Kare.m

```
1
2  % kare.m
3  % %
4  % %
5  % ADAPTACION AL ALGORITMO DE KARMARKAR
6  % %
7  %  TRANSFORMAR POR EJEMPLO:
8  %
9  % max Z=c'*x      ---------->          max Z=c'*y
10 %        A*x<=b                          A*y=0
11 %        x>=0                          e'*y=1
12 %                                     y>=0
13 %
14 %  EMPECEMOS CON EL PROBLEMA PL
15 %  PL
16 %        min c'* x
17 %        PL
18 %        A*x<=b
19 %        x> = 0
20 % Utilizando la teoria del primal y dual
21 % transformamos el problema PL en un problema de factibilidad estand % llamado PL1
22 %PL1 APL1*x=bPL1
23 %       x> = 0
24
25 APL1=[A zeros (m)   -eye (m)   zeros (m, n.) ; zeros (n) A'   zeros(n,m)
26      eye(n) c'   -b1   zeros(1,m)   zeros(1,n)];
27
28 bPL1=[b;c;0] ;
29
30 %        Como necesitamos un punto inicial factible estrictamente positivo
31 %        para comenzar las iteraciones, -generamos dicho punto realizando
32 %        un artificio entonces transformamos el problema PL1 en un problema
33 %        de minimización llamado PL2 factible y con óptimo nulo.
34 %
35 %        min cPL2'*x
36 %        ' ...
37 %         PL2 APL2*x=bPL1
38 %              x> = 0
39 % %
40 APL2=[APL1 bPL1-APL1*ones(2*m+2*n,1)]; cPL2=[zeros(2*m+2*n, 1);  1] ;
41 % PL2 se puede convertir en un problema en la FCK (A*x=0) realizando
42 % una transformación proyectiva y agrupando adecuadamente nos resulta
43 % PL3 que esta la FCK.
44 o
45 %        min cPL3'*y
46 %
47 % PL3 APL3*y=0  % e'*y=1  % y>=0  %
48 ao=ones(2*n+2*m+1,1); d=diag(ao); APL3=[APL2*d -bPL1]; cPL3=[d*cPL2;0];
49
50 %DEFINIENDO VALORES INICIALES
51 e=ones(2*m+2*n+2, 1) ; ee=e/(2*m+2*n+2) ; x=ee*1; N=0;
52 I=eye(2*n+2*m+2);
53 r=(1/sqrt((2*m+2*n+2)*(2*m+2*n+2-1)));
```

```
54 %INICIO DEL ALGORITMO
55 while (l*cPL3')*x>eps
56 dd=diag(x) ; % (1)
57 cc=[dd*cPL3]; % (2)
58 AA=APL3*dd ; % (3)
59 w=[AA;e'] ; % ( 4 )
60 p=(I-w'*inv(w*w')*w)*cc; % (5)
61 y=(ee-alfa*r*(p/norm(p)))*10; % . . . (6)
62 x=(dd*y/(e'*dd*y))*100; % (7)
63 N=N+1;
64 end
65 %FIN DEL ALGORITMO
66 % Obtenida una buena aproximación de la solución óptima de este
67 % problema de karmarkar, es necesario utilizar una transformación
68 % inversa para regresar al PL2 ; para esto basta con dividir las
69 % primeras 2n+2m+l componentes de la solución final por la ultima
70 % componente (2n+2m+2). Las primeras n componentes serán la solución
71 % del problema inicial (PL).
72
73 for j=l:n
74 xx(j)=(ao(j)*x(j))/x(2*m+2*n+2);
75 X=xx';
76 end
77 elapsedtime = etime(clock,tstart);
78 c=abs(c);
79 fprintf('\n\nEl numero de iteraciones es: %d\n\n',N)
80 disp('  N. x  F  ')
81 for i=l:n
82 ff=c(i)'*X(i);
83 fprintf('\n %d %d !sd\n\i,X(i),ff) end
84 F=c'*X;
85 fprintf('\n\nLa función objetivo optimizada F es: %d\n',F) FF=cPL3'*x;
86 fprintf('\n\nLa función objetivo dual es: %d\n\n',FF)
87 fprintf('\nALGORITMO KARMARKAR: Tiempo de ejecución = %9.3f segundos...
88 \n',elapsedtime);
```

5.1.3. Archivo karfo

Listing 5.3: karfo.m

```
1
2  % karfo.m
3
4  %ADAPTACION AL ALGORITMO DE KARMARKAR
5
6  %TRANSFORMAR POR EJEMPLO:
7  %
8  %   max Z=c'*x  ------->          max Z=c'*y
9  %
10 %        A*x=b                       A*y=0
11 %        x>=0                       e'*y=l
12 %                                  y>=0
13 % %
14 % EMPECEMOS CON EL PROBLEMA PL
15 %
16 %   PL         min c'*x
17 %
18 %                    A*x=b
19 %                     x>=0
20 %
21 %   Utilizando la teoria del primal y dual
22 %       transformamos el problema PL en un problema de factibilidad estándar
23 %       llamado PL1
24 %
25 %       PL1       APLl*x=bPLl
26 %                     x> = 0
27
28 APL1= [A zeros(m) zeros(m) zeros(m,n);...
29      zeros(n) A' -A' eye(n);...
30      c' -b' b' zeros(1,n)];
31
32 bPLl=[b;c;0];
33
34 %       Como necesitamos un punto inicial factible estrictamente positivo
35 %       para comenzar las iteraciones, generamos dicho punto realizando
36 %       un artificio entonces transformamos el problema PL1 en un problema
37 %       de minimización llamado PL2 factible y con óptimo nulo.
38 % 'i
39 %       min cPL2'*x
40 % ' %        PL2 APL2*x=bPLl % x>=0
41 % %
42 APL2=[APL1 bPLl-APLl*ones(2*m+2*n,1)];
43 cPL2=[zeros(2*m+2*n; 1);  1] ;
44 %PL2 se puede convertir en un problema en la FCK (A*x=0) realizando
45 %una transformación proyectiva y agrupando adecuadamente nos resulta
46 %PL3 que esta la FCK.
47 %     PL3      min cPL3'*y
48 % %              APL3*y=0
49 %                 e'*y=l
50 %                   y>=0
51
52 ao=ones(2*n+2*m+l, 1);
53 d=diag(ao) ;
```

```
54 APL3=[APL2*d -bPL1];
55 cPL3=[d*cPL2;0] ;
56
57 %DEFINIENDO VALORES INICIALES
58 e=ones(2*m+2*n+2,1);
59 ee=e/(2*m+2*n+2);
60 x=ee; N=0;
61 I=eye(2*n+2*m+2);
62 r=(1/sqrt((2*m+2*n+2)*(2*m+2*n+2-1)));
63 %INICIO DEL ALGORITMO
64 while cPL3'*x>eps
65 dd=diag(x);        %      (1)
66 cc=[dd*cPL3];          %   (2)
67 AA=APL3*dd;                %       (3)
68 w= [AA;e'] ;          %    ( 4 )
69 p=(I-(W'*inv(w*w')*w) )*cc;          %       (5)
70 y=(ee-alfa*r*(p/norm(p)))*10;        %     (6)
71 x=(dd*y/(e'*dd*y))*100;        % (7)
72 N=N+1;
73 end
74 %FIN DEL ALGORITMO
75 % Obtenida una buena aproximación de la solución óptima de este
76 % problema de karmarkar,   es necesario utilizar una transformación
77 % inversa para regresar al PL2;  para esto basta con dividir las
78 % primeras 2n+2m+l componentes de la solución final por la ultima
79 % componente   (2n+2m+2). Las primeras n componentes serán la solución
80 % del problema inicial   (PL).
81 for j=1:n
82 xx(j)=(ao(j)*x(j))/x(2*m+2*n+2);
83 X=xx' ;
84 end
85 elapsedtime = etime(clock,tstart);
86 fprintf('\n\nEl numero de iteraciones es: %d\n\n',N)
87 disp('  N.       x       F ')
88 for i=1:n
89 ff=c(i)'*X(i);
90 fprintf('\n %d      %d      %d\n',i, X(i),ff)
91 end
92 F=c'*X;
93 fprintf('\n\nLa función optimizada F es: %d\n\n',F)
94 disp('ALGORITMO KARMARKAR ')
95 fprintf('\nALGORITMO KARMARKAR: Tiempo de ejecución = %9.3f segundos...
96 \n',elapsedtime);
```

5.1.4. Archivo maxkarc

Listing 5.4: maxkarc.m

```
%maxkarc.m
%Definiendo valores iniciales
I=eye(n);
e=ones(n,1);
ee=e/n;
x=ee;
N=0;
r=(1/sqrt(n*(n-1)));
foi=c'*x;
D=diag(x);
AA=A*D;
cc=D*c;
B=[AA;e'];
p=(I-(B'*inv(B*B')*B))*cc;
y=(ee-alfa*r*(p/norm(p)));
x=(D*y/(e'*D*y));
N=N+1;
fof=c'*x;
%Inicio del algoritmo
while abs(fof-foi)>eps
    D=diag(x);          %.....(1)
    AA=A*D;             %.....(2)
    cc=D*c;             %.....(3)
    W=[AA;e'];          %.....(4)
    p=(I-(W'*inv(W*W')*W))*cc;   %..(5)
    y=(ee-alfa*r*(p/norm(p)))*10;    %.(6)
    x=(D*y/(e'*D*y))*100;        %.....(7)
    N=N+1;
    foi=fof;
    fof=c'*x;
end
%Fin algoritmo

elapsedtime=etime(clock,tstart);
F=c'*x;
fprintf('\n\nEl numero de iteraciones es : %d\n\n',N)
disp('N.          x          F')
for i=1:n
    xx=x(i);
    ff=c(i)'*x(i);
    fprintf('\n  %d    %d    %d\n',i,xx,ff)
end
fprintf('\n\nLa funcion optimizada F es: %d\n\n',F)
fprintf('\nAlgoritmo karmarkar: tiempo de ejecución = %-9.3f segundos ...
\n',elapsedtime);
```

5.1.5. Archivo minkarc

Listing 5.5: minkarc.m

```
1
2  %minkarc.m
3  %Definiendo valores iniciales
4  I=eye(n);
5  e=ones(n,1);
6  ee=e/n;
7  x=ee;
8  N=0;
9  r=(1/sqrt(n*(n-1)));
10 %Inicio del algoritmo
11 while c'*x>eps
12     D=diag(x);          %.....(1)
13     AA=A*D;             %.....(2)
14     cc=D*c;             %.....(3)
15     W=[AA;e'];          %.....(4)
16     p=(I-(W'*inv(W*W')*W))*cc   %..(5)
17     y=(ee-alfa*r*(p/norm(p)))*1  %.(6)
18     x=(D*y/(e'*D*y))*1     %.....(7)
19     N=N+1;
20 end
21 %Fin algoritmo
22
23 %elapsedtime=etime(clock,tstart);
24 F=c'*x;
25 fprintf('\n\nEl numero de iteraciones es : %d\n\n',N)
26 disp('N.          x          F')
27 for i=1:n
28     xx=x(i);
29     ff=c(i)'*x(i);
30     fprintf('\n  %d    %f    %f',i,xx,ff)
31 end
32 fprintf('\n\nLa funcion optimizada F es: %d\n\n',F)
33 %fprintf('\nAlgoritmo karmarkar: tiempo de ejecución =%-9.3f segundos
34 %\n',elapsedtime);
```

5.1.6. Archivo optkarc

Determina el tamaño de la matriz A

Listing 5.6: optkarc.m

```
1 alfa=0.7968;
2 eps=10^-8;
3 [m,n]=size(A);
```

5.2. Algoritmo de Barnes.

Es una variante del método que resuelve un problema de la forma de la sección (2.4.1), en este algoritmo los datos de ingreso son:

x0: vector inicial / $Ax = b$

A: matriz de restricciones

c: coeficientes de la función objetivo

R: radio del elipsoide

La programación del algoritmo de Barnes se realizo en Matlab 2007, dicho algoritmo lleva por nombre Barnes.m, y La matriz de ingreso de datos se tiene por nombre datosb.m

5.2.1. Archivo barnes

Listing 5.7: barnes.m

```
1
2  function [x1,it]=barnes(A,x0,c,tol,R)
3  E=2*tol;
4  it=0;
5  while E>tol
6       D=diag(x0);            %.....(1)
7       L=inv(A*D*D*A')*A*D*D*c;                %.....(2)
8       x1=x0-R/norm(D*(c-A'*L),2)*D*D*(c-A'*L);              %.....(3)
9       E=norm(x1-x0,2);          %.....(4)
10      x0=x1
11      it=it+1
12 end
13  %Fin algoritmo
```

5.2.2. Tabla de Datos de Ingreso

Listing 5.8: datosb.m

```
1
2  %MODELO 7    :EFICIENCIA=31 %, SIN DERIVACION DE UCHUSUMA, SIN ROTACION DE CULTIVOS
3            Mo=7;
4            %                max  c'*X
5
6            %                  A*X<=b
7
8
9  A=[2200  3000     3400        0        0        0     3400        0        0        0        0        0
10 1987     2710     3071        0        0        0     2890        0        0        0        0        0
11 2090     2850     3230        0        0        0     2850        0        0        0        0     1520
12  1597    2177     2468        0     1161        0        0     1161     1161        0        0     2032
13  1320    1800     2040        0     1680        0        0     1992     1680      960     1080     2280
14  1011    1379     1563        0     1931        0        0     1931     1655     1287     1103     1563
15  1100    1500     1700        0     1700        0        0     1800     2000     2100     1800     1400
16  1320    1800     2040        0     1680        0        0     1440     1440     1920     2520        0
17  1544    2105     2385     2245        0      842      842        0        0     1684     2806        0
18  1980    2700     3060     3780        0     2520     2160        0        0        0     2520        0
19  2076    2831     3208     3963        0     3963     3585        0        0        0        0        0
20  2310    3150     3570     4410        0     4200     3780        0        0        0        0        0
21  20        13        4        0        0        0       18        0        0        0        0        0
22  23        15        4        0        0        0       26        0        0        0        0        0
23  14        12        2        0        0        0       26        0        0        0        0       23
24  9          5        4        0       41        0        0       29       21        0        0       21
25  8          4        3        0       37        0        0       28       20       28       40       22
26  7          4        2        0       29        0        0       20       16       26       34       45
27  10         5        4        0       34        0        0       39       21       19       27       46
28  16         7        3        0       32        0        0       38       20       21       21        0
29  8       5           2       13        0       29       28        0        0       38       36        0
30  9          4        3       13        0       33       28        0        0        0       40        0
31  17        11        2       17        0       19       25        0        0        0        0        0
32  8         10        2       16        0       29       19        0        0        0        0        0
33  1          0        0        0        0        0        0        0        0        0        0        0
34  0          1        0        0        0        0        0        0        0        0        0        0
35  0          0        1        0        0        0        0        0        0        0        0        0
36  0          0        0        1        0        0        0        0        0        0        0        0
37  0          0        0        0        1        0        0        0        0        0        0        0
38  0          0        0        0        0        1        0        0        0        0        0        0
39  0          0        0        0        0        0        1        0        0        0        0        0
40  0          0        0        0        0        0        0        1        0        0        0        0
41  0          0        0        0        0        0        0        0        1        0        0        0
42  0          0        0        0        0        0        0        0        0        1        0        0
43  0          0        0        0        0        0        0        0        0        0        1        0
44  0          0        0        0        0        0        0        0        0        0        0        1
45  1          1        1        1        1        1        1        1        1        1        1        1];
46
47
48
49           c=[980 ; 759 ; 882 ;847 ; 768 ; 169 ; 1077 ; 213 ; 186 ; 459 ; 561 ; 405];
50
51
52           b=[1582934 ; 1874880 ; 1909699 ; 1462406 ; 1331165 ; 1319328 ; 1312416 ; ...
53              1272240 ; 1174176 ; 1159747 ; 1039392 ; 1157069 ; 22239 ; 22239 ; 22239 ; ...
54              22239 ; 22239 ; 22239 ; 22239 ; 22239 ; 22239 ; 22239 ; 22239 ; 22239 ; ...
55              120 ; 80 ; 190 ; 40 ; 260 ; 110 ; 125 ; 200 ; 150 ; 50 ; 130 ; 190 ; 2506];
```

Bibliografía

[1] K. Borgwardt . The Simplex Method, Springler Verlag. New York, 1987.

[2] Dantzig G. B.- Aplication of the Simplex Method to a Transportation Problem. Monografía n.º 13 de la Cowles Commision. Jhon Wiley, New York 1951

[3] D. Bertsimas and N. Tsitsiklis Introduction to Liner Optimization, pag. 32, editorial. 1997

[4] D. Golfarb y M.J. Todd: Modifications and Implementation of the Ellipsoid Algorithm for Linear Programing. Mathematical Programing, 23,pag 1-19, 1982.

[5] Gutierrez, M. "Estudio hidrogeológico detallado Valle Caplina - La Yarada". Universidad Nacional Agraria, Facultad de Ingeniería Agrícola, tesis para optar el titulo de Ingeniero Agrícola. Lima. 1 986. 339 p.

[6] Haimovich. The simplex algorithm is very good. On the expected number of pivot steps and related properties of random linear programs. 1983.

[7] Hernandez, S. "Método de Diseño Optimo de Estructuras". Colegio de Ingenieros de caminos, canales y puertos de Marcación de Aragón. Colección Senior N°8. España 1 998.

[8] N. Karmarkar, A New Polinomial-time Algorithm for Linear Programing, Combinatórica 4, pag. 373-395, 1984.

[9] L. G. Khachiyan . Polinomian Algorithm i Linear Programing. U.R.S.S. Computational Mathematics y Mathematical Physics, 20,pag 53-72, 1980.

[10] V. Klee Minty: How good is the Simplex Agorithm?. O. Shisha, e.d. in Inequalities Academic Press, New York. pag 159 - 175, 1972

[11] Mora, H. "El Método de Karmarkar". 8vo Coloquio Distrital de Matemáticas y Estadística. Universidad Nacional de Colombia. Colombia. 1992. 62 p.

[12] Nemirovskii A., A new polynomial algorithm for linear programming, Soviet Math. Dokl., 37, 1988, 264-269

[13] Oficina de Información Agraria (OIA). Ministerio de Agricultura, Republica del Perú . Boletín Hidrometeorologico 1999. 140p.

[14] R. Pérez Cupe: Algoritmos polinomiales para problemas de flujo máximo. Tesis. Lima Perú. 2001.

[15] D. Nagesh Kumar, IOptimization Methods: Linear Programming Aplications-Structural, IISc, Bangalore 1992.

[16] The problem of the average speed of the simplex method. Mathematical Programming, 27, 1983, 241-262.

[17] Sosa, W. "Introducción a la Optimización, Programación Lineal". 18º Coloquio, Sociedad Matemática Peruana, 3-7 de Julio del 2000 56 p.

[18] G. W. Stewart, Matrix Algorithms. Volume I: Basic Decompositions, Siam 1998.