Trabajo Fin de Máster

Programación Lineal Multiobjetivo y Aplicaciones

Joaquín Carballo Mato

Máster en Técnicas Estadísticas

Curso 2018-2019

Propuesta de Trabajo Fin de Máster

Título en galego: Programación Lineal Multiobxectivo e Aplicacións
Título en español: Programación Lineal Multiobjetivo y Aplicaciones
English title: Multiobjective Linear Programming and Applications
Modalidad: Modalidad A
Autor: Joaquín Carballo Mato, Universidad de Santiago de Compostela
Directores: Balbina Virginia Casas Méndez, Universidad de Santiago de Compostela; David Rodríguez Penas, Universidad de Santiago de Compostela
Breve resumen del trabajo: En este trabajo se presentará una revisión de la programación lineal con un sólo objetivo con el fin de sentar las bases de la progamación multiobjetivo. Revisaremos el concepto de optimalidad de Pareto, distinguiéndola del concepto manejado en la optimización tradicional, y daremos diferentes mecanismos con los cuales podremos demostrar la optimalidad de ciertos puntos factibles. Además, se implementarán diferentes códigos, tanto en R (algoritmo del símplex monoobjetivo y multiobjetivo) como en AMPL, con el fin de ayudarnos en la resolución de los ejemplos presentados.
Recomendaciones: Es de interés el uso de herramientas informáticas para la obtención de soluciones, como por ejemplo AMPL o R.
Otras observaciones:

Doña Balbina Virginia Casas Méndez, profesora titular de la Universidad de Santiago de Compostela, don David Rodríguez Penas, contratado post-doctoral de la Universidad de Santiago de Compostela, informan que el Trabajo Fin de Máster titulado

Programación Lineal Multiobjetivo y Aplicaciones

fue realizado bajo su dirección por don Joaquín Carballo Mato para el Máster en Técnicas Estadísticas. Estimando que el trabajo está terminado, dan su conformidad para su presentación y defensa ante un tribunal.

En Madrid, a 04 de Septiembre de 2019.

La directora:

Doña Balbina Virginia Casas Méndez

El director:

Don David Rodríguez Penas

El autor:

Don Joaquín Carballo Mato

Índice general

Resumen

Resumen en español

La programación lineal nos permite resolver problemas de diversos campos utilizando diferentes algoritmos, siendo el método símplex uno de los más utilizados.

Se empezará el trabajo haciendo una revisión de la programación lineal con un sólo objetivo, centrándonos en el método símplex como principal algoritmo de resolución de este tipo de problemas. Se presentará la parte teórica y la acompañaremos de numerosos ejemplos con el fin de afianzar el entendimiento. El capítulo se cerrará con la presentación del algoritmo del símplex que se ha programado en el lenguaje R.

La segunda parte de este trabajo, y principal, se centrará en la programación multiobjetivo. Se revisará el concepto de optimalidad de Pareto, distinguiéndolo del concepto manejado en la optimización tradicional, y se explicará la escalarización, mostrando su aplicabilidad resolviendo (utilizando el lenguaje AMPL) un ejemplo real aplicado a la gestión de listas de espera. El capítulo se cerrará con la descripción del método símplex multiobjetivo, la tabla del símplex multiobjetivo y la programación del algoritmo del símplex multiobjetivo en el lenguaje R.

English abstract

Linear programming allows us to solve different type of problems in several fields by the use of various algorithms, being the simplex method the most commonly used.

This thesis will start presenting the multiobjective linear programming with a single objective, focusing our attention on simplex method. We will present the theoretical part along with a large number of examples. To finish the chapter the code of the simplex algorithm, which was programmed in the R language, will be presented.

The main part, and second chapter of this thesis, is focused on multiobjective lineal programming. We will examin the concept of Pareto optimality, distinguishing it from the classical concept of optimality used in traditional optimization. Moreover, scalarization will be explained, showing how effective it is by solving a real world example (using the AMPL language) applied to queuing problems in hospitals management operations. The chapter will be closed with the presentation of the multiobjetive simplex method, its table and an example solved with the multiobjective simplex algorithm which was programmed in the R language.

Prefacio

La investigación operativa es una rama de las matemáticas que busca la mejor solución a los problemas, una vez dado un cierto criterio, dentro de todas las soluciones factibles. Este tipo de problemas se escriben de la forma

$$\begin{aligned} \text{optimizar} \quad & f(x) \\ \text{sujeto a} \quad & x \in X \end{aligned}$$

siendo X un conjunto no vacío, denominado conjunto factible y f una función real definida en X, llamada función objetivo. Denominamos optimizar a dos tipos de opciones: maximizar o minimizar.

Este tipo de problemas nos permiten un "fácil" estudio de una gran variedad de problemas de la vida real; sin embargo, muchas veces nos encontramos ante problemas donde varios objetivos entran en conflicto, por lo que nuevas formas de enfrentarnos a este tipo de problemas tendrán que ser descritos. Algunos ejemplos pueden ser los siguientes:

- Diseño de un automóvil: A la hora del diseño de un automóvil tenemos que tener varias variables en cuenta. Por ejemplo, tendremos que minimizar el coste de producción o su consumo al mismo tiempo que maximizamos su seguridad y confort. Usualmente estos criterios entran en conflicto y no pueden ser optimizados al mismo tiempo.

- Compra de una vivienda: A la hora de tomar la decisión de comprar una casa hay que tener en cuenta diferentes criterios. Mínimo coste de compra, mínimo coste de mantenimiento, máxima comodidad o mejor situación geográfica. Y como es comprensible, maximizar la comodidad suele ir unido a aumentar el coste de la vivienda.

Como hemos visto en estos ejemplo, incluso en las situaciones más simples, nos podemos encontrar ante problemas con objetivos que no se pueden satisfacer simultáneamente. Debido a esto, los conceptos de optimización usuales no pueden ser utilizados y se necesita el desarrollo de técnicas que resuelvan problemas con varios objetivos. El economista italiano Pareto (1884-1923) introduce lo que se denominará *Óptimo de Pareto* y que se puede expresar como sigue: "No se alcanza la asignación óptima de recursos en una sociedad mientras que se pueda hacer más rico a un individuo manteniendo la riqueza del resto de individuos". Previo a Pareto, el economista Irlandés Edgeworth (1845-1926) ya había definido un óptimo para problemas con varias utilidades en un problema de dos consumidores P y Q de la siguiente manera: "Un punto (x, y) es óptimo siempre que en cualquier dirección que tomemos un paso suficientemente pequeño P y Q no crecen al mismo tiempo pero, mientras que uno crece, el otro decrece". Según la definición dada por Pareto, una alternativa es óptima si cualquier alternativa mejor que ella según un criterio, es peor respecto de algún otro criterio. Y esta es la base de la optimización multiobjetivo, que tiene su raíz en la teoría de espacios ordenados desarrollada por Cantor (1845-1918) y Hausdorff (1868-1942).

Un problema de optimización multiobjetivo se escribe de la siguiente manera:

$$\begin{aligned} \text{Maximizar} \quad & F(x) := (f_1(x), ..., f_k(x)) \\ \text{sujeto a} \quad & x \in X \end{aligned}$$

donde cada función $f_1, ..., f_k$ es una función real sobre X y "Maximizar" significa encontrar el elemento $\bar{x} \in X$ tal que ningún valor $F(x)$, con $x \in X$ sea mayor que $F(\bar{x})$. Es importante tener en cuenta que la solución verifica que, aunque la solución $\bar{x}$ no es peor que ninguna otra solución, no tiene porqué ser la mejor solución entre todas las $x \in X$; de hecho, en general no lo será. Por tanto, resolver un problema de programación multiobjetivo consistirá en encontrar el conjunto de soluciones "óptimas", o al menos una parte representativa de ellas; y esto puede llegar a ser un trabajo muy laborioso, lo que hace que la resolución de problemas de optimización multiobjetivo sea un gran reto teórico y práctico.

Capítulo 1

Programación lineal con un solo objetivo

La programación lineal es el campo de la programación matemática dedicado a maximizar o minimizar (optimizar) una función lineal, denominada función objetivo, de tal forma que las variables de dicha función estén sujetas a una serie de restricciones expresadas mediante un sistema de ecuaciones o inecuaciones también lineales.

En los siglos XVII y XVIII, grandes matemáticos como Newton, Leibnitz, Bernouilli y, sobre todo, Lagrange, que tanto habían contribuido al desarrollo del cálculo infinitesimal, se ocuparon de obtener máximos y mínimos condicionados de determinadas funciones. Posteriormente, el matemático francés Jean Baptiste-Joseph Fourier (1768-1830) fue el primero en intuir, aunque de forma imprecisa, los métodos de lo que actualmente llamamos programación lineal y la potencialidad que de ellos se deriva.

Si exceptuamos al matemático Gaspar Monge (1746-1818), quien en 1776 se interesó por problemas de este género, debemos remontarnos al año 1939 para encontrar nuevos estudios relacionados con los métodos de la actual programación lineal. En este año, el matemático ruso Leonodas Vitalyevich Kantoróvitch publica una extensa monografía titulada *Métodos matemáticos de organización y planificación de la producción*, en la que por primera vez se hace corresponder a una extensa gama de problemas una teoría matemática precisa y bien definida a la que actualmente llamamos programación lineal. El nombre de programación lineal no procede de la creación de programas de ordenador, sino de un término militar, programar, que significa "realizar planes o propuestas de tiempo para el entrenamiento, la logística o el despliegue de las unidades de combate", ya que en esta época, donde la Segunda Guerra Mundial estaba teniendo lugar, la programación lineal se planteaba como un modelo matemático desarrollado para planificar los gastos y los retornos, a fin de reducir los costos del ejército y aumentar las pérdidas del enemigo.

Kantoróvich recibiría el premio Nobel de economía en 1975 por sus aportaciones al problema de la asignación óptima de recursos humanos.

Tras finalizar la Segunda Guerra Mundial, en Estados Unidos se asumió que la eficaz coordinación de todas las energías y recursos de la nación era un problema, de tal complejidad, que su resolución y simplificación pasaba necesariamente por los modelos de optimización.

El gran impulso de la programación lineal para la industria y los negocios se identifica con el doctor George Dantzig, matemático de origen norteamericano que formula, en términos matemáticos muy precisos, el enunciado estándar al que cabe reducir todo problema de programación lineal. Dantzig desarrollaría este mismo año el algoritmo símplex, un método sistemático de resolución para problemas modelados con programación lineal. Esto ocurrió en 1947, una vez enrolado en el grupo de trabajo intensivo conocido como SCOOP (Scientific Computation of Optimum Programs) para la Fuerza Aérea de los EE.UU.

Desde que George B. Dantzig desarrolló el método símplex en 1947, que será explicado con de-

tenimiento en la Sección 1.2, la programación lineal se ha utilizado extensamente en el área militar, industrial, gubernamental y de planificación urbana entre otras. El desarrollo de la programación lineal se considera entre los avances científicos más importantes del siglo XX, pues su impacto ha sido extraordinario. Actualmente es una herramienta de uso común que ha beneficiado a muchas organizaciones en distintos países, consiguiendo ahorros de toda índole, por lo que su uso se está ampliando rápidamente a todos los sectores de la sociedad.

Los problemas de programación lineal pueden ser resueltos con diferentes algoritmos; y aunque el símplex es el método más utilizado no siempre es el más eficiente, ya que su complejidad en el peor caso puede ser exponencial, como podemos comprobar en Klee (1972). En 1979 el matemático ruso Leonid Khachiyan diseñó el llamado algoritmo del elipsoide, a través del cual demostró que los problema de programación lineal son resolubles de manera eficiente, es decir, en tiempo polinomial. Más tarde, en 1984, Narendra Karmarkar introduce un nuevo método (el método del punto interior), lo que constituiría un enorme avance en los principios teóricos y prácticos en el área.

Desde la creación del método símplex mucha gente ha contribuido al crecimiento de la programación lineal, ya sea desarrollando su teoría matemática, diseñando códigos y métodos computacionales eficientes, experimentando nuevas aplicaciones o utilizando la programación lineal como una herramienta auxiliar para resolver problemas más complejos como son programas enteros, programas discretos, programas no lineales, problemas combinatorios, problemas de programación estocástica o problemas de control óptimo.

En los últimos años lo notable y más prometedor parece ser: la programación lineal en números enteros por R. Gomory, el principio de descomposición de Dantzig y Wolfe, los programas lineales estocásticos...

1.1. Propiedades de los problemas de programación lineal

Empezaremos presentando una formulación matemática de los problemas de programación lineal y posteriormente veremos que todo problema de programación lineal puede ser expresado de esta forma.

$$\begin{array}{rll} \text{maximizar} & f(x) & \\ \text{sujeto a} & g_i(x) \leq 0 & i = 1, ..., m \\ & h_j(x) = 0 & j = 1, ..., l. \end{array}$$

Aquí $x = (x_1, ..., x_n) \in \mathbb{R}^n$ son las *variables de decisión*; $f : \mathbb{R}^n \to \mathbb{R}$ la *función objetivo*, que representa el beneficio o coste asociado a cada combinación de las variables de decisión; y $g_i(x) \leqq 0, i \in \{1, ..., m\}$ (con $g_i : \mathbb{R}^n \to \mathbb{R}$) y $h_j(x) = 0, j \in \{1, ..., l\}$ (con $h_j : \mathbb{R}^n \to \mathbb{R}$) son las *restricciones de desigualdad e igualdad*, que representan las configuraciones de los valores de las variables de decisión que son factibles. Además, tenemos que tener en cuenta que la relación de orden $x \geqq y$ significa que el vector $x - y$ no tiene elementos negativos y $x \geq y$ significa que $x - y$ no tiene elementos negativos y que además $x \neq y$.

Si queremos escribir el problema de programación lineal de una forma más compacta (usando notación matricial), tendremos la siguiente formulación:

$$\begin{array}{rl} \max & cx \\ \text{sujeto a} & Ax \leqq b \\ & x \geqq 0. \end{array} \qquad \text{(P)}$$

Aquí $c \in \mathbb{R}^n$ es el *vector de costes*; $A \in \mathbb{R}^{m \times n}$ la *matriz de restricciones* con elementos a_{ij}, donde a_i^f representará la fila i y a_j^c la columna j; y $b \in \mathbb{R}^m$ el *vector de lados derechos*. Las restricciones de la forma $x_i \geq 0$ se denominan *restricciones de no negatividad* y el conjunto $\{x : Ax \leqq b, x \geqq 0\}$ se

conoce como *región factible*.
Además, una vez que tenemos definido el problema, podemos hablar de soluciones óptimas. Una solución óptima es una solución factible, cumple las restricciones del problema, que da el valor más favorable de la función objetivo.

A la hora de resolver problemas de programación lineal trabajaremos con la *forma estándar*, donde todas las restricciones son de igualdad y las variables de decisión no negativas. Para ello, existen diferentes transformaciones de los elementos de los problemas de programación lineal que nos llevan a nuestro propósito:

- **Función Objetivo:** Todo problema de minimización se puede convertir en uno de maximización y viceversa llevando a cabo la siguiente transformación.

$$\min \sum_{j=1}^{n} c_j x_j = -max \sum_{j=1}^{n} -c_j x_j$$

- **Restricciones:**

  - Para cambiar una restricción de "$\geq$" por una de "$\leq$" o viceversa sólo tendremos que multiplicar todos los elementos de la restricción por -1.

  -Si sólo queremos restricciones de igualdad (como será en nuestro caso), cada restricción de la forma $\sum_{j=1}^{n} a_{ij}x_j \leq b_i$ la reemplazaremos por $\sum_{j=1}^{n} a_{ij}x_j + x_i^s = b_i$ donde x_i^s se denomina *variable de holgura* y $x_i^s \geq 0$. En caso de que la restricción sea de "$\geq$" tendremos que restar una variable de holgura en lugar de sumarla.

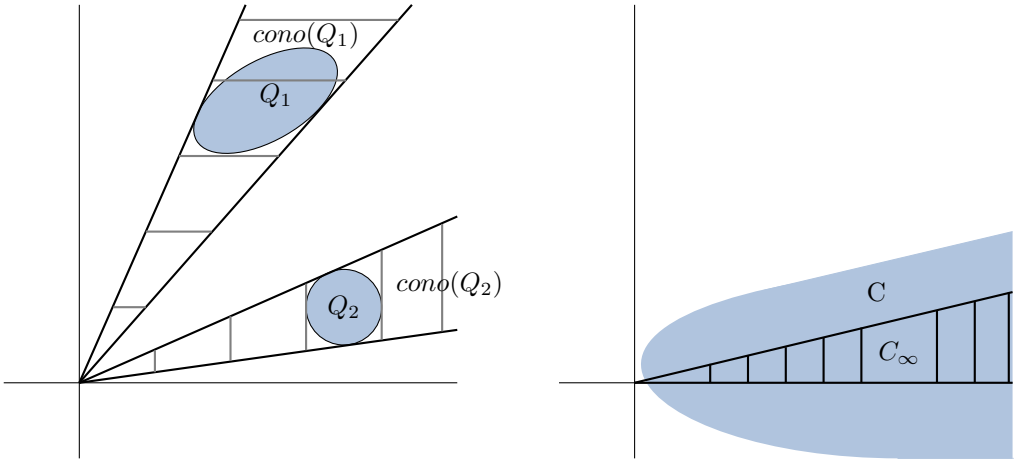
- **No negatividad:** Si lo que yo quiero para mi problema en forma estándar es que las variables solo tomen valores mayores o iguales que cero pero tengo una variable x_j que puede tomar valores positivos y negativos, lo que haré es substituir x_j por dos variables x_j' y x_j'', de tal forma que $x_j', x_j'' \geq 0$ y $x_j = x_j' - x_j''$.

Una vez visto como podemos representar cualquier problema de programación lineal en su forma estándar nos centraremos en las características de la región factible, que está definida por las restricciones de nuestro problema a resolver.
La región factible de un problema de programación lineal es un poliedro convexo (que puede ser acotado o no) ya que está definida por una intersección finita de semiespacios. Sea el poliedro definido por $S = \{x : Ax \leqq b, x \geqq 0\}$, S será la intersección de m semiespacios (que son las restricciones) de la forma $a_i^f x \leq b_i$ y n semiespacios de la forma $x_j \geq 0$. Cuando $b_i = 0$ para todo i, el conjunto factible S es un cono[1], que se denomina *cono convexo poliédrico*, y la solución del problema será el punto 0 o no tendrá óptimo finito.

Si ahora cogemos un punto $\bar{x} \in S$ que pertenezca al hiperplano dado por la restricción i, es decir $a_i^f \bar{x} = b_i$, entonces decimos que la restricción i está *activa* en $\bar{x}$ o que está *saturada* para $\bar{x}$. El conjunto que contiene los índices i de las restricciones saturadas en un punto x se denomina *conjunto de índices activos* y lo denotaremos por $I(x)$.
Sea un número $k \leq n$, el conjunto de puntos que pertenece a k de los $m + n$ hiperplanos que definen S (o que saturan a k de las $m + n$ restricciones) tiene dimensión $n - k$. Si ponemos como ejemplo $\mathbb{R}^3$, el conjunto de puntos que saturan una restricción tendrá dimensión 2 ($n - k = 3 - 1 = 2$), es decir, será una cara; los que saturan dos restricciones, una arista y el que satura tres, un punto.
También hay que tener en cuenta que existen un tipo de puntos llamados *degenerados* en los cuales se saturan más de n restricciones del poliedro.

Que la región factible sea un poliedro convexo nos será de gran utilidad, ya que como veremos más adelante los puntos extremos de estos poliedros tienen un papel fundamental en el método símplex. La siguiente proposición nos dará una caracterización de este tipo de puntos.

[1]Dado un conjunto no vacío $Q \subseteq \mathbb{R}^n$, se define el cono de Q (Fig. 1.1a) como el conjunto siguiente: $cono(Q) := \{ta : a \in Q, t \in \mathbb{R}, t \geq 0\}$.



(a) Cono de Q_1 y Q_2. (b) C_∞.

Figura 1.1

Proposición 1.1.1 *Sea $S \subseteq \mathbb{R}^n$ el poliedro dado por $\{x : Ax \leqq b, x \geqq 0\}$. Un punto $\bar{x} \in S$ es un punto extremo de S si y solo si en $\bar{x}$ se saturan n restricciones independientes de las que definen el conjunto S.*

A continuación definiremos un concepto de importancia vital en las proposiciones posteriores, las direcciones extremas.
Dado un conjunto convexo Q, un vector $d \neq 0$ es una dirección de Q si para cualquier punto $x \in Q$, el rayo de vértice x y dirección d está contenido en S. El conjunto de todas las direcciones lo denotaremos por C_∞ (Fig. 1.1b). Además, el conjunto C_∞ es un cono convexo al que denominaremos *cono asintótico*. Hay que tener en cuenta que un conjunto convexo acotado no tiene direcciones, ya que los rayos son no acotados.

Definición 1.1.1 *Dado un conjunto convexo Q, una dirección d de Q es una dirección extrema si no puede ser representada como una combinación cónica[2] estricta de dos direcciones distintas de Q.*

De la misma manera, un punto $x \in S$ es un punto extremos de S si no puede ser representado como una *combinación convexa*[3] estricta de dos puntos distintos de S.

Proposición 1.1.2 *El conjunto de óptimos de un problema de programación lineal es un conjunto convexo.*

Demostración. Supongamos que x^1 y x^2 son dos soluciones óptimas de nuestro problema de programación lineal; y sea $z = cx^1 = cx^2$ el valor de la función objetivo en los puntos óptimos. Definamos ahora un punto $\bar{x} = \lambda x^1 + (1-\lambda)x^2$, con $\lambda \in (0,1)$, que por tanto es combinación lineal de x^1 y x^2. Lo que queremos demostrar es que $\bar{x}$ es solución óptima; es decir, que $\bar{x}$ es factible y que $c\bar{x} = z$.

1. Para comprobar la factibilidad tenemos que probar que $A\bar{x} = b$ y $\bar{x} \geqq 0$.
   Por la linealidad del producto matricial tenemos que:
   $A\bar{x} = A(\lambda x^1 + (1-\lambda)x^2) = \lambda Ax^1 + (1-\lambda)Ax^2 = \lambda b + (1-\lambda)b = b,$ ya que x^1, x^2 son factibles.
   Además,
   $\bar{x} = \lambda x^1 + (1-\lambda)x^2 \geq \lambda 0 + (1-\lambda)0 = 0,$ ya que $x^1, x^2 \geq 0$ y $\lambda \in (0,1)$.

[2]Se denomina *combinación cónica* de una cantidad finita de puntos $x_1, ..., x_k$ a la suma $\sum_{i=1}^{k} \lambda_i x_i$, donde los λ_i son no negativos.

[3]Se denomina *combinación convexa* de una cantidad finita de puntos $x_1, ..., x_k$ a la suma $\sum_{i=1}^{k} \lambda_i x_i$, donde los λ_i son no negativos y $\sum_{i=1}^{k} \lambda_i = 1$. Se dice estricta si $0 < \lambda_i < 1$.

2. Por la linealidad del producto escalar tenemos que:
   $c\bar{x} = c(\lambda x^1 + (1-\lambda)x^2) = \lambda cx^1 + (1-\lambda)cx^2 = \lambda z + (1-\lambda)z = z.$

□

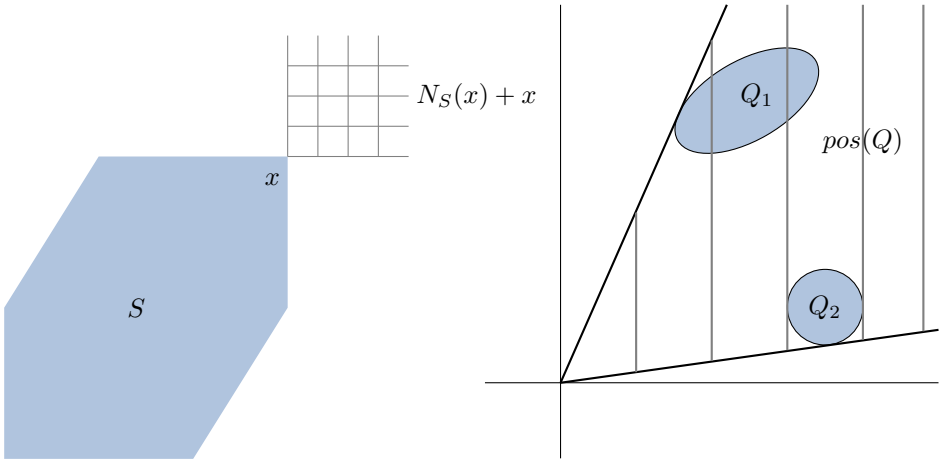
Esta proposición nos lleva ante cuatro tipo de soluciones posibles para nuestros problemas: solución única, múltiples soluciones óptimas, no existe solución óptima finita y que la región factible sea vacía.

Antes de demostrar los siguientes resultados, que nos dirán cuando una solución es óptima y nos asegurarán la existencia de soluciones óptimas finitas, definamos vector normal, cono normal, conjunto relativamente interior y mostremos un teorema que nos será de utilidad en capítulos posteriores.

Definición 1.1.2 *Sea $S \subseteq \mathbb{R}^n$ el poliedro convexo dado por $\{x : Ax \leqq b, x \geqq 0\}$ y sea $x \in S$. Decimos que v es un vector normal a S en x si*

$$v \cdot (y - x) \leq 0 \quad \forall y \in S.$$

El conjunto de vectores normales a S en x forman un cono convexo que se denomina *cono normal*(Fig. 1.2a) a x en S y se denota por $N_S(x)$.



(a) Cono normal. (b) Envoltura positiva de $Q = Q_1 \cup Q_2$.

Figura 1.2

El siguiente teorema nos dice como calcular el cono normal en un punto frontera de nuestra región factible.

Teorema 1.1.1 *El cono normal en un punto frontera x del poliedro S es la envoltura positiva*[4] *de los vectores a_i^f, siendo i un índice activo en el punto x.*

Definición 1.1.3 *Dado un conjunto no vacío y convexo $Q \subset \mathbb{R}^n$, el conjunto relativamente interior de Q se define como: $ri(Q) := \{x \in Q : B_n$*[5]*$(x, \epsilon) \cap aff(Q)$*[6]*$\subset Q$ para algún $\epsilon > 0\}$.*

La proposición que demostraremos a continuación nos dará una condición que nos asegura la optimalidad de una solución factible.

[4]Dado un conjunto no vacío $Q \subseteq \mathbb{R}^n$, se define la envoltura positiva (Fig. 1.2b) de Q como el conjunto siguiente: $pos(Q) := \{\sum_{i=1}^k t_i a_i : a_i \in Q, t_i \in \mathbb{R}, t_i \geq 0, i = 1, ..., k \text{ con } k \in \mathbb{N}\}$.

[5]Se define como bola cerrada en $\mathbb{R}^n$ de centro a y radio r ($B_n(a, r)$) como el conjunto $\{x \in \mathbb{R}^n : d(a, x) \leq r\}$.

[6]Dado un conjunto no vacío $Q \subseteq \mathbb{R}^n$, se define la envoltura afín de Q como el conjunto siguiente: $aff(Q) := \{\sum_{i=1}^k t_i a_i : a_i \in Q, t_i \in \mathbb{R}, i = 1, ..., k \text{ y } \sum_{i=1}^k t_i = 1 \text{ con } k \in \mathbb{N}\}$.

Proposición 1.1.3 *Dado un problema de programación lineal tal que*

$$\begin{array}{rll} max & cx & \\ sujeto\ a & Ax \leqq b & (P) \\ & x \geqq 0. & \end{array}$$

con región factible el poliedro convexo y no vacío $S = \{x : Ax \leqq b, x \geqq 0\}$*, las siguientes afirmaciones son equivalentes:*

I) $\bar{x}$ *es una solución óptima.*

II) El vector c *pertenece al cono normal del conjunto* S *en el punto óptimo* $\bar{x}$*.*

III) Toda cara que contiene al óptimo $\bar{x}$ *como punto relativamente interior es una cara óptima.*

Demostración. La implicación $III \Rightarrow I$ es trivial, así que demostremos que $I \Rightarrow II$ y que $II \Rightarrow III$.

($I \Rightarrow II$) Si $\bar{x}$ es una solución óptima, tenemos que

$$cx \leq c\bar{x} \text{ para todo } x \in S \Longleftrightarrow c(x - \bar{x}) \leq 0 \text{ para todo } x \in S.$$

Por definición, c es un vector normal a S en $\bar{x}$, por lo que se cumple II.

($II \Rightarrow III$) Asumamos II y sea x un punto de la cara que tiene a $\bar{x}$ como punto relativamente interior. Sea $\delta > 0$, los puntos $\bar{x} + \delta(\bar{x} - x)$ y $\bar{x} - \delta(\bar{x} - x)$ pertenecen a S. Por tanto, tenemos que

$$\begin{aligned} c(\bar{x} + \delta(\bar{x} - x) - \bar{x}) \leq 0 &\Longleftrightarrow c(\bar{x} - x) \leq 0 \Longleftrightarrow c\bar{x} \leq cx, \\ c(\bar{x} - \delta(\bar{x} - x) - \bar{x}) \leq 0 &\Longleftrightarrow c(-(\bar{x} - x)) \leq 0 \Longleftrightarrow cx \leq c\bar{x}. \end{aligned}$$

Por lo que $cx = c\bar{x}$ y queda demostrado que x también es un punto óptimo.

□

La siguiente proposición nos ofrece una manera de saber si nuestro problema tiene solución óptima finita.

Proposición 1.1.4 *Dado un problema de programación lineal con región factible no vacía, las siguientes afirmaciones son ciertas:*

I) Si el problema tiene solución óptima finita, al menos unos de los puntos extremos de la región factible será óptimo.

II) El problema tiene solución óptima finita si y sólo si no existe ninguna dirección extrema d *de la región factible tal que* $cd > 0$*.*

Corolario 1.1.1 *Si el conjunto factible del problema es acotado, entonces el problema tiene soluciones óptimas.*

Demostración. Si el conjunto factible está acotado, no tendrá direcciones extremas. Por tanto, la condición II de la Proposición 1.1.4 nos dice que el problema tendrá soluciones óptimas finitas. □

El método símplex que veremos en la sección 1.2 no calcula todas las direcciones extremas, pero encontrará una para la que $cd > 0$ si el problema no tiene solución óptima finita. Lo que nos dice esta condición geométricamente es que el problema no tendrá solución óptima finita si el vector de costes forma un ángulo menor de 90° con alguna dirección extrema.

En este trabajo no demostraremos algunas de las proposiciones y teoremas vistas hasta ahora ya que sólo queremos dar una introducción a la programación lineal con un sólo objetivo. Estas demostraciones pueden ser consultadas en González-Díaz (2017) y The Luc (2016). Además, un buen libro de referencia en caso de querer profundizar en la programación lineal es Bazaara et al. (1990).

1.2. El método símplex

En esta sección presentaremos el método símplex. Este método, desarrollado por G. B. Dantzig en 1947, consiste en la utilización de un algoritmo para optimizar el valor de la función objetivo teniendo en cuenta las restricciones planteadas. Partiendo de uno de los vértices de la región factible, por ejemplo el vértice A, y aplicando la propiedad: si la función objetivo no toma su valor máximo en el vértice A, entonces existe una arista que parte del vértice A y a lo largo de la cual la función objetivo aumenta hasta llegar a otro vértice. El procedimiento es iterativo, pues mejora los resultados de la función objetivo en cada etapa hasta alcanzar la solución buscada. Ésta se encuentra en un vértice del que no parta ninguna arista a lo largo de la cual la función objetivo aumente.

La primera prueba relevante fue realizada para resolver una versión del problema de la dieta (Dantzig 1990) con 77 variables y 9 restricciones. Un grupo de trabajo invirtió 1000 horas para desarrollar todos los cálculos necesarios. Sin embargo, hoy en día somos capaces de resolver este tipo de problemas de forma inmediata con la ayuda de ordenadores.

1.2.1. Bases y soluciones básicas factibles

El método símplex consiste en recorrer los distintos puntos extremos de la región factible de forma que, en general, sólo sea necesario visitar una relativamente pequeña parte de ellos antes de llegar al óptimo (si éste existe).

A la hora de trabajar con el método símplex, es necesario que nuestros problemas estén planteados en forma estándar, donde A es una matriz $m \times n$ de rango m, es decir, todas las filas de la matriz son independientes (no hay restricciones redundantes). La región factible es de la forma $S = \{x : Ax = b, x \geqq 0\}$; y por lo visto en la sección anterior, sus puntos extremos estarán formados por intersecciones de n hiperplanos de los que definen S. Además, ya que el problema está formulado en forma estándar, en cualquier solución factible habrá al menos m restricciones saturadas que provienen de $Ax = b$. Por tanto, dado un punto extremo $x \in S$, habrá necesariamente $(n - m)$ restricciones de no negatividad que se saturan en x, dando un total de $m + (n - m) = n$ restricciones saturadas.

Proposición 1.2.1 *Sea un problema de programación lineal en forma estándar con región factible* $S = \{x : Ax = b, x \geqq 0\}$ *donde* $A \in \mathbb{R}^{m\times n}$ *tiene rango* m*. Entonces, en todo punto extremo* x *de* S *hay al menos* $(n - m)$ *variables que toman valor* 0*.*

Como veremos a continuación, la definición de solución básica se basa en la proposición que acabamos de mostrar.

Sea el sistema $Ax = b, x \geqq 0$ con $A \in \mathbb{R}^{m\times n}$ y $b \in \mathbb{R}^m$ tales que rango(A)=rango(A, b)=m. La matriz A puede dividirse en dos matrices tale que $A = [B, N]$, donde $B \in \mathbb{R}^{m\times m}$ y $N \in \mathbb{R}^{m\times(n-m)}$. Supongamos también que B tiene rango m con el fin de permitir la existencia de su matriz inversa. Entonces, se denomina *solución básica* a la solución $x = (x_B, x_N)$ del sistema $Ax = B$ y $x \geqq 0$ dada por

$$x_B = B^{-1}b \quad \text{y} \quad x_N = 0.$$

Además, si $x_B \geqq 0$ se llama *solución básica factible*. De la misma forma, a cada submatriz cuadrada $B \in \mathbb{R}^{m\times m}$ de A de rango m se le puede asignar una solución básica, donde x_B contiene las componentes de x que se corresponden con las columnas de A presentes en B y x_N las restantes. La matriz B se denomina *matriz básica* o *base* ya que tiene rango m y por tanto, sus columnas y filas forman una base de $\mathbb{R}^m$. La matriz N se denomina *matriz no básica*. Las componentes de x_B se denominan *variables básicas* y las de x_N las *variables no básicas*. Por último, si alguna componente de x_B es igual a cero, entonces estaremos ante una *solución básica degenerada*, lo que implica que distintas bases representen al mismo punto extremo de la región factible.

Teorema 1.2.1 *Sea un problema de programación lineal en forma estándar con región factible* $S = \{x : Ax = b, x \geqq 0\}$*. Si* $S \neq \emptyset$*, entonces,*

I) Toda solución básica factible de S se corresponde con un punto extremos de S y cada punto extremo de S se corresponde con al menos una solución básica factible de S.

II) A cada punto extremos le corresponde una base (no necesariamente única) y a cada base le corresponde un único punto extremo.

El teorema que acabamos de presentar nos da una idea de algoritmo de búsqueda de óptimos en problemas de programación lineal. Esta idea consistiría en, bajo una formulación estándar, estudiar todas las soluciones básicas del problema. Mostremos este concepto de algoritmo en el siguiente ejemplo.

Ejemplo 1.2.1 *Sea el problema de programación lineal:*

$$\begin{array}{rrl} max & 2x_1 + x_2 & \\ sujeto\ a & x_1 + x_2 & \leq 4 \\ & x_1 & \leq 3 \\ & x & \geqq 0. \end{array}$$

Y ya que tenemos que trabajar con los problemas en forma estándar añadiremos dos variables de holgura, x_3^s y x_4^s. Por lo tanto, nuestro problema quedará formulado de la siguiente manera:

$$\begin{array}{rrl} max & 2x_1+x_2 & \\ sujeto\ a & x_1+x_2+x_3^s & = 4 \\ & x_1 \qquad +x_4^s & = 3 \\ & x & \geq 3 \end{array}$$

La región factible de este problema será representada en la siguiente figura.

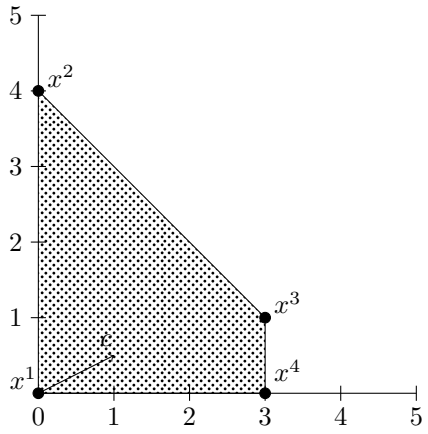


Figura 1.3: Región factible.

Ya que la matriz de restricciones A viene dada por

$$A = (A_1^c\ A_2^c\ A_3^c\ A_4^c) = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix} \tag{1.1}$$

sus soluciones básicas factibles se corresponderán con las bases 2×2 para las cuales $B^{-1}b$ sea no negativo. Veamos ahora todas las submatrices con capacidad de dar lugar a soluciones básicas factibles.

I) $B = (A_1^c\ A_2^c) = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$.

Por tanto, $x_B = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = B^{-1}b = \begin{pmatrix} 0 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 4 \\ 3 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \end{pmatrix}$ *y* $x_N = \begin{pmatrix} x_3^s \\ x_4^s \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

La solución básica es $x = (3, 1, 0, 0)$*, que es factible. El coste asociado a esta solución es* $cx = (2, 1, 0, 0)(3, 1, 0, 0) = 7$.

II) $B = (A_1^c\ A_3^c) = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$.

Por tanto, $x_B = \begin{pmatrix} x_1 \\ x_3^s \end{pmatrix} = B^{-1}b = \begin{pmatrix} 0 & 1 \\ 1 & -1 \end{pmatrix}\begin{pmatrix} 4 \\ 3 \end{pmatrix} = \begin{pmatrix} 3 \\ 1 \end{pmatrix}$ *y* $x_N = \begin{pmatrix} x_2 \\ x_4^s \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

La solución básica es $x = (3, 0, 1, 0)$*, que es factible. El coste asociado a esta solución es* $cx = 6$.

III) $B = (A_1^c\ A_4^c) = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}$.

Por tanto, $x_B = \begin{pmatrix} x_1 \\ x_4^s \end{pmatrix} = B^{-1}b = \begin{pmatrix} 1 & 0 \\ -1 & -1 \end{pmatrix}\begin{pmatrix} 4 \\ 3 \end{pmatrix} = \begin{pmatrix} 4 \\ -1 \end{pmatrix}$, $x_N = \begin{pmatrix} x_2 \\ x_3^s \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

La solución básica es $x = (4, 0, 0, -1)$*, que no es factible ya que* $x_4^s < 0$.

IV) $B = (A_2^c\ A_3^c) = \begin{pmatrix} 1 & 1 \\ 0 & 0 \end{pmatrix}$.

En este caso B *no es base ya que* $rango(B) = 1$ *y por tanto no tiene rango máximo. Por tanto,* B *no tiene ninguna solución básica asociada.*

V) $B = (A_2^c\ A_4^c) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

Por tanto, $x_B = \begin{pmatrix} x_2 \\ x_4^s \end{pmatrix} = B^{-1}b = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 4 \\ 3 \end{pmatrix} = \begin{pmatrix} 4 \\ 3 \end{pmatrix}$, $x_N = \begin{pmatrix} x_1 \\ x_3^s \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

La solución básica es $x = (0, 4, 0, 3)$*, que es factible. El coste asociado a esta solución es* $cx = 4$.

VI) $B = (A_3^c\ A_4^c) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$.

Por tanto, $x_B = \begin{pmatrix} x_3^s \\ x_4^s \end{pmatrix} = B^{-1}b = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 4 \\ 3 \end{pmatrix} = \begin{pmatrix} 4 \\ 3 \end{pmatrix}$ *y* $x_N = \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$.

La solución básica es $x = (0, 0, 4, 3)$*, que es factible. El coste asociado a esta solución es* $cx = 0$.

Por tanto, tenemos cuatro soluciones básicas factibles que, olvidándonos de las variables de holgura, dan lugar a los cuatro puntos extremos que podemos observar en la región factible. Además, una vez hechos estos cálculos, hemos comprobado que el mayor beneficio asociado es de 7 *unidades, que se corresponde con el punto óptimo* $x^3 = (3, 1)$.

Este método tiene la ventaja de que funciona para cualquier dimensión; sin embargo, calcular todas las bases y quedarse con la que tenga un mejor valor de la función objetivo entre todas las factibles tiene muchos inconvenientes, siendo los siguientes los más importantes:

- El algoritmo no es capaz de detectar que estamos ante un problema sin óptimo finito.
- El algoritmo obliga a invertir una matriz, lo que se vuelve muy exigente computacionalmente a medida que m crece.
- El número de posibles bases crece exponencialmente con las dimensiones de m y n, ya que son las combinaciones de n elementos tomados de m en m, es decir,

$$\frac{n!}{m!(n-m)!}.$$

Con el fin de solucionar estos problemas aparece el método símplex, que busca una forma más eficiente de saltar de un punto extremo a otro de la región factible, o lo que es lo mismo, una forma eficiente de tratar los cambios de base.

1.2.2. Descripción del método símplex. Ejemplo

La diferencia con el método anterior es que el método símplex se mueve entre extremos de la región factible con la seguridad de que en cada iteración se llega a un punto extremo donde la función objetivo no empeora. Este hecho nos permite recorrer sólo una cantidad "pequeña" de extremos. Además, el método símplex reconoce cuando se encuentra ante un problema con región factible vacía o sin solución óptima finita.
Presentaremos la idea de este método utilizando el Ejemplo 1.2.1.

Recordemos que nuestro problema en forma estándar es:

$$\begin{array}{rl} \max & 2x_1+x_2 \\ \text{sujeto a} & x_1+x_2+x_3^s = 4 \\ & x_1 + x_4^s = 3 \\ & x \geq 3 \end{array}$$

con matriz de restricciones

$$A = (A_1^c\ A_2^c\ A_3^c\ A_4^c) = \begin{pmatrix} 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \end{pmatrix}.$$

Las bases y sus soluciones básicas factibles asociadas (que se corresponden con los puntos extremos representados en la figura de la región factible del ejemplo 1.2.1) son las siguientes:

I) $B = (A_1^c\ A_2^c) = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$. Su solución básica asociada es $x^3 = (3, 1, 0, 0)$, con coste $cx^3 = 7$.

II) $B = (A_1^c\ A_3^c) = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$. Su solución básica asociada es $x^4 = (3, 0, 1, 0)$, con coste $cx^4 = 6$.

III) $B = (A_2^c\ A_4^c) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. Su solución básica asociada es $x^2 = (0, 4, 0, 3)$, con coste $cx^2 = 4$.

IV) $B = (A_3^c\ A_4^c) = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$. Su solución básica asociada es $x^1 = (0, 0, 4, 3)$, con coste $cx^1 = 0$.

Supongamos que desconocemos cuales son las soluciones básicas y sus costes asociados. Nuestra idea será movernos de punto extremo en punto extremo pero siempre mejorando el valor de la función objetivo. Empecemos situándonos en $x^2 = (0, 4, 0, 3)$, con base asociada $B = (A_2^c\ A_4^c)$, por tanto $N = (A_1^c\ A_3^c)$. Para desplazarnos a otro punto tendremos que reemplazar una de las columnas de B por una de las de N. Nuestro objetivo es saber qué variable queremos que entre en la base, es decir, qué cambio de variable hará que nuestra función objetivo mejore; y saber esto es la clave del método símplex.

Dada la base $B = (A_2^c\ A_4^c)$, tenemos que $b = Ax = Bx_B + Nx_N$, y despejando x_B tenemos $x_B = B^{-1}b - B^{-1}Nx_N$. Esta ecuación nos dice dos cosas: 1) Fijada la base, si $x_N = 0$ entonces $x_B = B^{-1}b$; 2) Nos aporta la información sobre como tiene que cambiar x_B si x_N es distinta de cero. Mostremos esta idea introduciendo A_1^c en la base. Esto implica, en general, que x_1 se hará positivo, por tanto, para mantener la factibilidad x_B deberá cambiar. Concretamente, x_B cambiará a razón de $-B^{-1}A_1^c x_1 = -\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}(1,1)x_1 = -(x_1, x_1)$. Es decir, para mantener la factibilidad x_2 y x_4^s tendrán que reducirse en una unidad por cada unidad que aumente x_1. La interpretación de esto es

la siguiente: $B^{-1}A_j^c$ devuelve los coeficientes resultantes de expresar A_j^c como combinación lineal de los vectores de B, aportando la información necesaria para saber como equilibrar las variaciones de x_j con los vectores de la base y así no perder factibilidad.

En el caso de nuestro ejemplo, si aumentamos x_1, las componentes de $x_B = (4, 3)$ decrecerán a la misma velocidad a la que crezca x_1.
Pasando a la función objetivo, si $z_B = c_B(B^{-1}b) = cx^2 = 4$ y definamos $z_1 = c_B B^{-1} A_1^c$:

$$\begin{aligned} z = cx &= c_B x_B + c_N x_N = c_B x_B + c_1 x_1 \\ &= c_B(B^{-1}b - B^{-1}A_1^c x_1) + c_1 x_1 \\ &= z_B - (c_B B^{-1} A_1^c - c_1)x_1 \\ &= z_B - (z_1 - c_1)x_1 \\ &= z_B + (c_1 - z_1)x_1. \end{aligned}$$

Esta ecuación nos dice que, como el problema es de maximizar, nos interesa aumentar el valor de x_1 siempre que $c_1 - z_1 \geq 0$. Si calculamos c_1 y z_1, tenemos que $z_1 = c_B B^{-1} A_1^c = c_B(1, 1) = (1, 0)(1, 1) = 1$ y $c_1 = 2$ y por tanto, $c_1 - z_1 = 2 - 1 = 1 > 0$. Esta expresión nos dice que será beneficioso meter A_1^c en la base.
Como resumen tenemos que: las variables de la base cambian a razón $B^{-1}A_1^c$ por cada unidad que aumenta x_1; la función objetivo, en lo que respecta a las variables de la base, empeora en una unidad (ya que $z_1 = 1$) por cada unidad que aumente x_1; la función objetivo mejora en dos unidades (ya que $c_1 = 2$) por cada unidad que aumenta x_1. Por tanto, por cada unidad que aumentemos x_1, la mejora total en la función objetivo será de $c_1 - z_1 = 1$ unidad.
Si sale de la base A_4^c, pasaremos de $x^2 = (0, 4, 0, 3)$ a $x^3 = (3, 1, 0, 0)$ y la función objetivo pasa de $cx^2 = 4$ a $cx^3 = 7$. Esto es lo esperado, ya que aumentamos x^1 en tres unidades, la mejora también es de tres unidades.

1.2.3. Descripción del método símplex. Formulación general

En esta sección desarrollaremos formalmente las ideas del apartado anterior con el fin de fijar los cimientos del método símplex, que será explicado en el siguiente apartado.

Sea un problema de programación lineal en forma estándar

$$\begin{aligned} \max \quad & cx \\ \text{sujeto a} \quad & Ax = b \\ & x \geqq 0 \end{aligned}$$

con $A \in \mathbb{R}^{m \times n}$ y de rango m. Sea B una base con solución básica factible asociada $x = \begin{pmatrix} x_B \\ x_N \end{pmatrix} = \begin{pmatrix} B^{-1}b \\ 0 \end{pmatrix}$ y cuya función objetivo z_B se define por

$$z_B = c \begin{pmatrix} x_B \\ x_N \end{pmatrix} = (c_B, c_N) \begin{pmatrix} B^{-1}b \\ 0 \end{pmatrix} = c_B B^{-1} b.$$

Denotamos por J a los índices de las variables no básicas, es decir, las columnas de A que no están en B, y definamos $y^j = B^{-1}A_j^c$ para cada columna j de A.

Ya que (x_B, x_N) es una solución básica factible, tenemos que $x_B \geqq 0$, $x_N \geqq 0$ y $b = Ax =$

$Bx_B + Nx_N$. Si despejamos x_B de la ecuación anterior tenemos que

$$\begin{aligned} x_B &= B^{-1}b - B^{-1}Nx_N \\ &= B^{-1}b - \sum_{j\in J} B^{-1}A_j^c x_j \\ &= B^{-1}b - \sum_{j\in J} y^j x_j. \end{aligned}$$

Esta ecuación plasma como se modifica x_B si alguna componente de x_N pasa a ser distinta de cero y queremos respetar las restricciones $Ax = b$. Si además queremos que nuestra solución sea factible y que mejore la función objetivo, también tendremos que respetar las restricciones de no negatividad. Definamos ahora $z_j = c_B B^{-1}A_j^c = c_B y^j$ y traslademos las ecuaciones anteriores a la función objetivo. Dado un $x \in \mathbb{R}^n$ cualquiera:

$$\begin{aligned} z = cx &= c_B x_B + c_N x_N \\ &= c_B(B^{-1}b - \sum_{j\in J} B^{-1}A_j^c x_j) + \sum_{j\in J} c_j x_j \\ &= z_B - \sum_{j\in J}(c_B y^j - c_j)x_j \\ &= z_B - \sum_{j\in J}(z_j - c_j)x_j \\ &= z_B + \sum_{j\in J}(c_j - z_j)x_j. \end{aligned}$$

Para conseguir que z sea mayor que z_B precisaremos que $c_j - z_j > 0$. Por tanto tenemos que:

- Si $c_j - z_j \leq 0$ para toda variable no básica, entonces (x_B, x_N) es una solución óptima.
- Si $c_j - z_j > 0$ para alguna variable no básica j, entonces esta variable es aspirante a entrar en la base. Si esto ocurre, el vector $y^j = B^{-1}A_j^c$ me permite saber cómo varían las variables de la base actual cuando aumento x_j.

A la cantidad $c_j - z_j$ se le denomina *coste reducido* de la variable j.
Veamos en el siguiente teorema que ocurre si la base B no es degenerada.

Teorema 1.2.2 *Sea B una base factible y $\bar{x}$ una solución básica factible asociada con B. La siguiente afirmación se cumple:*

- *Si la base B es no degenerada, $\bar{x}$ es un punto óptimo si y sólo si el vector de costes reducidos es negativo o cero.*

Esta forma de proceder nos permitirá reconocer cuando nos encontramos frente a un problema sin óptimo finito. Esto ocurrirá cuando tengamos una variable no básica $j \in J$ para la cual $(c_j - z_j) > 0$ y al mismo tiempo y^j tiene todas sus componentes menores o iguales que cero. Esto nos dice que independiente de lo que aumentemos el valor de x_j, el cambio inducido en las variables de la base es tal que ninguna de ellas reduce su valor ($B^{-1}b - \sum_{j\in J} y^j x_j \geq B^{-1}b \geq 0$), por lo que podemos aumentar indefinidamente x_j consiguiendo un aumento de la función objetivo sin perder factibilidad.

1.2.4. Algoritmo del método símplex

El esquema general del método símplex puede verse en la Figura 1.4.

Este método se presenta suponiendo el problema en forma estándar y que $A \in \mathbb{R}^{m\times n}$ tiene rango m. Esto se puede hacer sin pérdida de generalidad ya que hemos visto en la Sección 1.1 que todo

MÉTODO SÍMPLEX

INICIALIZACIÓN

Partimos de una base B que tiene asociada una solución básica factible $x = (x_B, x_N)$, con $x_B = B^{-1}b$ y $x_N = 0$. Además, $z_B = c_B B^{-1} b$.

PASO 1 (Criterio de entrada)

Calculamos los valores $c_j - z_j$ para todas las variables no básicas $j \in J$, donde $z_j = c_B B^{-1} A_j^c = c_B y^j$. Sea $k \in J$ el menor índice tal que $c_k - z_k = \max_{j \in J}\{c_j - z_j\}$, tenemos las siguientes posibilidades:

- Si $c_k - z_k < 0$, FIN. la solución básica factible $x = (x_B, x_N)$ es óptima.
- Si $c_k - z_k = 0$, la solución básica factible $x = (x_B, x_N)$ es óptima, ir al PASO 2 con k como variable de entrada..
- Si $c_k - z_k > 0$, $x = (x_B, x_N)$ no es una solución óptima, ir al PASO 2 con k como variable de entrada.

PASO 2 (Criterio de salida)

Calculamos $y^k = B^{-1} A_k^c$. Tenemos las siguientes posibilidades:

- Si $y^k \leq 0$, FIN. El problema no tiene óptimo finito. Podemos movernos indefinidamente a lo largo del rayo
$$R = \{(x_B, x_N) + (-y^k, e^k)x_k : x_k \geq 0\}$$
cuyos puntos tienen todas sus componentes no negativas, por lo que serán factibles. A medida que $x_k \to +\infty$, $z_B + (c_k - z_k)x_k$ tiende a ∞.
- Si $y^k \not\leq 0$, entonces A_k^c entra en la base y x_k pasa a ser una variable básica. Abandona la base la variable i, aquella con el índice más bajo de entre las que primero se anulen en el rayo R:
$$\frac{(x_B)_i}{y_i^k} = \text{mín}\left\{\frac{(x_B)_l}{y_l^k} : y_l^k > 0\right\}.$$
Recalcular B, $x = (x_B, x_N)$, z_B y los vectores y^j. Volver al PASO 1.

Figura 1.4: Esquema del método símplex.

problema de maximización se puede pasar a uno de minimización y viceversa. En cuanto a que A tenga rango máximo m, presentaremos en la sección 1.2.5 un método para conseguir una solución inicial que asegurará que la matriz de restricciones tiene rango máximo.

En cuanto a la existencia de B^{-1} la tenemos asegurada en todas las iteraciones, ya que si queremos meter un vector en la base será suficiente con representarlo como combinación de los vectores de la base y substituirlo por alguno de los vectores con coeficiente distinto de cero en dicha representación. Esto lo tenemos siempre asegurado ya que en el Paso 2 del método símplex observamos que el vector saliente siempre es uno con coeficiente positivo ($y_i^k > 0$). Aunque acabamos de ver que la existencia de B^{-1} está garantizada en todos los pasos, la inversión de matrices suele tener un alto costo computacional. La inversión de una matriz $m \times m$, como es el caso de nuestra base B, es de $O(m^3)$ si lo hacemos por el método de eliminación de Gauss-Jordan. Para resolver este inconveniente presentaremos en la sección 1.2.7 la tabla del símplex. Este método utiliza gran parte de la información existente, aprovechándose de que sólo cambia una columna de B entre cada iteración, para hacer m^2 operaciones por iteración, lo que convierte a este método en una de las formas más eficientes de implementar el método símplex.

El método símplex también identifica todas las soluciones óptimas en caso de que existan múltiples.

Una condición necesaria para que el óptimo no sea único es que haya alguna variable no básica con $c_j - z_j = 0$ en la solución final. En este caso el método generaría todas las soluciones metiendo sucesivamente en la base tales variables no básicas. Esto permite detectar también los rayos extremos óptimos. El conjunto de soluciones óptimas del problema será la envoltura convexa de tales puntos y rayos extremos óptimos.

Por último, en la sección 1.2.6 discutiremos si tenemos asegurado que el algoritmo termina en una cantidad finita de pasos.

Terminaremos esta sección con un ejemplo de aplicación del método símplex siguiendo lo detallado en la Figura 1.4. El ejemplo que resolveremos será el trabajado a lo largo de esta sección.

Ejemplo 1.2.2 *Sea el problema de programación lineal*

$$\begin{aligned} max \quad & 2x_1 + x_2 \\ sujeto\ a \quad & x_1 + x_2 \leq 4 \\ & x_1 \leq 3 \\ & x \geqq 0. \end{aligned}$$

Tenemos que trabajar con los problemas en forma estándar, por añadiremos dos variables de holgura, x_3^s y x_4^s. Por lo tanto, nuestro problema quedará formulado de la siguiente manera:

$$\begin{aligned} max \quad & 2x_1+x_2 \\ sujeto\ a \quad & x_1+x_2+x_3^s = 4 \\ & x_1 + x_4^s = 3 \\ & x \geqq 0. \end{aligned}$$

Ya que hemos agregado una variable de holgura por restricción, nuestra matriz de restricciones tendrá rango 2 (tiene una submatriz 2×2 que es la identidad). Además, ya que los lados derechos son no negativos, tendremos una base inicial $B = (A_3^c, A_4^c)$, para la cual B y B^{-1} son la matriz identidad. La solución básica asociada es $x_B = B^{-1}b = (4, 3)$ y $x_N = (0, 0)$, que es factible ya que cumple las restricciones de no negatividad.
A continuación desarrollaremos el algoritmo utilizando dicha base como base inicial.

Inicialización: $B = (A_3^c, A_4^c)$, $x_B = B^{-1}b = (4, 3)$, $x_N = (0, 0)$ *y* $z_B = c_B x_B = (0, 0)(4, 3) = 0$.

Iteración 1. Paso 1 (Criterio de entrada). *Calculamos los $c_j - z_j$:*

- $B^{-1}A_1^c = y^1 = (1, 1)$. *Por tanto,* $c_1 - z_1 = c_1 - c_B y^1 = 2 - 0 = 2 > 0$.
- $B^{-1}A_2^c = y^2 = (1, 0)$. *Por tanto,* $c_2 - z_2 = c_2 - c_B y^2 = 1 - 0 = 1 > 0$.

El mayor valor para $c_j - z_j$ se obtiene cuando $j = 1$ y es mayor que cero. Por tanto, x_1 es la variable candidata a entrar.

Iteración 1. Paso 2 (Criterio de salida). *El vector y^1 tiene ambas componentes positivas y tenemos que mín$\{\frac{4}{1}, \frac{3}{1}\} = 3$, asociado con la variable x_4^s, que saldrá de la base. La nueva base es $B = (A_3^c, A_1^c)$, con lo que*

$$B = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad y \quad B^{-1} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix}.$$

Además, $x_B = (x_3^s, x_1) = B^{-1}b = (1, 3)$, $x_N = (0, 0)$ y $z_B = c_B x_B = (0, 2)(1, 3) = 6$.

Iteración 2. Paso 1 (Criterio de entrada). *Calculamos los $c_j - z_j$:*

- $B^{-1}A_2^c = y^2 = (1,0)$. *Por tanto,* $c_2 - z_2 = c_2 - c_B y^2 = 1 - 0 = 1 > 0$.
- $B^{-1}A_4^c = y^4 = (-1,1)$. *Por tanto,* $c_4 - z_4 = c_4 - c_B y^4 = 0 - 2 = -2 < 0$.

El máximo se alcanza para $j = 2$ *y es mayor que cero. Por tanto* x_2 *es la variable candidata a entrar.*

Iteración 2. Paso 2 (Criterio de salida). *El vector* y^2 *tiene sólo una componente positiva, asociada a la variable* x_3^s*, que saldrá de la base. La nueva base es* $B = (x_2, x_1)$*, con lo que*

$$B = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} \quad y \quad B^{-1} = \begin{pmatrix} 1 & -1 \\ 0 & 1 \end{pmatrix}.$$

$x_B = (x_2, x_1) = B^{-1}b = (1,3)$, $x_N = (0,0)$ *y* $z_B = c_B x_B = (1,2)(1,3) = 7$.

Iteración 3. Paso 1 (Criterio de entrada). *Calculamos los* $c_j - z_j$*:*

- $B^{-1}A_3^c = y^3 = (1,0)$. *Por tanto,* $c_3 - z_3 = c_3 - c_B y^3 = 0 - 1 = -1 < 0$.
- $B^{-1}A_4^c = y^4 = (-1,1)$. *Por tanto,* $c_4 - z_4 = c_4 - c_B y^4 = 0 - 1 = -1 < 0$.

El máximo se alcanza para $j = 3$ *y* $j = 4$*, pero es menor que cero.*

Por tanto hemos alcanzado la solución óptima que, expresada en las variables del problema original, será $x = (3,1)$ *con coste asociado* 7.

1.2.5. Construcción de una solución inicial

El algoritmo presentado en la Figura 1.4 parte de una solución factible pero no explica como conseguirla. Uno de los métodos más utilizados para la consecución de esta solución inicial del método símplex es el *método de la M grande*. Ya hemos visto en el ejemplo 1.2.2 que bajo un problema en forma estándar con vector de lados derechos no negativo y matriz de restricciones conteniendo a la matriz identidad $m \times m$ como submatriz, conseguir una base inicial se hace de forma inmediata. Si esto no ocurre, la idea del método de la M grande radica en agregar variables artificiales en tantas restricciones como sea necesario para conseguir que la nueva matriz de restricciones contenga a la matriz identidad como submatriz, lo que nos sitúa en los supuestos del ejemplo anterior. El papel que ejercen las variables artificiales es muy diferente al de las variables de holgura, ya que aumentan el conjunto de soluciones factibles del problema. Por tanto, cuando añadimos variables artificiales lo que queremos es que el método símplex las expulse de la base, para así movernos por la región factible del problema original. Para lograr este objetivo se le asigna a estas variables un coste M tan grande como deseemos[7]. Si el algoritmo remata con alguna de las variables artificiales con valor positivo, entonces estaremos ante un problema sin soluciones factibles.

Mostraremos este método mediante un ejemplo.

Ejemplo 1.2.3 *Sea el problema de programación lineal*

$$\begin{array}{rl} max & 3x_1+3x_2-3x_3-4x_4 \\ sujeto\ a & -x_1+\ x_2-6x_3 \qquad \geq 0 \\ & 5x_1+2x_2+2x_3-\ x_4 \geq 7 \\ & -x_1+3x_2-\ x_3+2x_4 \leq 7 \\ & x \geqq 0. \end{array}$$

Si lo pasamos a forma estándar tenemos que

[7] A la hora de elegir el tamaña de la M hay que tener bastante cuidado. Si el método se implementa a mano se puede trabajar con la propia M; sin embargo, a la hora de implementarlo a ordenador habrá que elegir un valor suficientemente grande, para lo que hay que prestar cierta atención.

$$\begin{array}{rl} max & 3x_1{+}3x_2{-}3x_3{-}4x_4 \\ sujeto\ a & -x_1{+}\ x_2{-}6x_3 \qquad -x_5^s \qquad\qquad = 0 \\ & 5x_1{+}2x_2{+}2x_3{-}\ x_4 \qquad -x_6^s \qquad = 7 \\ & -x_1{+}3x_2{-}\ x_3{+}2x_4 \qquad\quad +x_7^s = 7 \\ & \qquad\qquad\qquad\qquad\qquad\quad x \geq 0. \end{array}$$

Vemos que aún no hemos conseguido una submatriz de A que sea la identidad, solamente tenemos el vector $(0,0,1)$ que se corresponde con la última columna de A. Para conseguir esta submatriz deseada añadiremos dos variables artificiales. El resultado es el siguiente:

$$\begin{array}{rl} max & 3x_1{+}3x_2{-}3x_3{-}4x_4 \qquad\qquad -Mx_8^a{-}Mx_9^a \\ sujeto\ a & -x_1{+}\ x_2{-}6x_3 \qquad -x_5^s \qquad + \quad x_8^a \qquad = 0 \\ & 5x_1{+}2x_2{+}2x_3{-}\ x_4 \qquad -x_6^s \qquad\quad + \quad x_9^a = 7 \\ & -x_1{+}3x_2{-}\ x_3{+}2x_4 \qquad +x_7^s \qquad\qquad = 7 \\ & \qquad\qquad\qquad\qquad\qquad\qquad\qquad\quad x \geq 0. \end{array}$$

De esta manera hemos obtenido un problema en el que la matriz de restricciones tiene una submatriz identidad, con lo que podemos aplicar el método símplex tomando $B = (A_7^c\ A_8^c\ A_9^c)$, con solución asociada $x_B = (7,0,7)$ y $x_N = 0$, que en el problema ampliado con las variables artificiales es factible. Por último, habría que aplicar el método símplex y comprobar si en la solución óptima todas las variables artificiales se han hecho cero. Si este es el caso, se habría encontrado una solución óptima del problema original; en caso contrario, el problema original no tendrá soluciones factibles.

1.2.6. Convergencia y ciclado del algoritmo

El interés en la convergencia del método símplex se vincula con la existencia o no de soluciones básicas degeneradas. En ausencia de este tipo de soluciones el siguiente resultado nos asegura que el método símplex converge.

Teorema 1.2.3 *Sea (P) un problema de programación lineal de forma estándar sin soluciones básicas degeneradas. Dada una solución básica factible inicial de (P), el método símplex termina en una cantidad finita de pasos, bien encontrando una solución factible óptima o concluyendo que no tiene solución óptima finita.*

Demostración. Notemos que el número de vértices del poliedro $S = \{x : Ax = b, x \geq 0\}$ es finito, digamos que es igual a p. Además, en el método símplex, la función objetivo siempre aumenta su valor cuando cambiamos de vértice; por tanto, después de un número finito de iteracciones, como mucho p, estaremos en un vértice que: o bien es óptimo o la función objetivo aumenta a lo largo de un rayo. □

En caso de existencia de soluciones básicas degeneradas esto puede suponer que el algoritmo se mueva entre las distintas bases que representan un punto extremo no óptimo sin conseguir abandonarlo; esto es lo que se denomina problemas de ciclado. En la práctica, para solucionar este problema, muchas implementaciones del método símplex realizan los cambios de base trabajando sobre pequeñas perturbaciones sobre los parámetros del problema. La idea es la siguiente: si trabajamos con un ejemplo en $\mathbb{R}^2$, una degeneración se ocasiona cuando las rectas que definen 3 semiespacios se intersecan en un mismo punto. Si llevamos a cabo una pequeña perturbación de los lados derechos del problema, las rectas se moverán levemente de tal manera que se intersecarían dos a dos en puntos muy próximos entre si pero no coincidentes con alta probabilidad. De esta manera disminuiríamos la probabilidad de degeneración de forma notable.

1.2.7. Tabla del método símplex

El procedimiento de resolución de problemas de programación lineal mediante la *tabla del símplex* permitirá realizar las actualizaciones de $x_B = B^{-1}b$ e $y^j = B^{-1}A_j^c$ de una forma más eficiente. Para ello se guardará en una tabla toda la información relevante en cada iteración, con lo que la actualización de la misma se llevará a cabo de una forma más sencilla.

Supongamos una base B para la cual sus m columnas se corresponden con las primeras m variables, N la matriz formada por las columnas no básicas e $I_{m\times m}$ la matriz identidad $m \times m$. Además, denotaremos por $\bar{b}$ al vector $B^{-1}b$. Entonces, la tabla del símplex quedará de la siguiente forma:

	x_B	x_N	$\bar{z}$
$c_j - z_j$	0 $c_B - c_B B^{-1}B$	$c_N - c_B B^{-1}N$	$-c_B B^{-1}b$
x_B	$I_{m\times m}$ $B^{-1}B$	$B^{-1}N$	$B^{-1}b$

Cuadro 1.1: Tabla del método símplex.

Vemos que si nos olvidamos de los $c_j - z_j$ en el Cuadro 1.1, las columnas de la tabla se corresponden con las columnas de la matriz $B^{-1}A$.

Si ahora escribimos la tabla de forma expandida (Cuadro 1.2) tenemos que:

	x_{B_1}	$\cdots$	x_{B_r}	$\cdots$	x_{B_m}	$\cdots$	x_j	$\cdots$	x_k	$\cdots$	$\bar{z}$
$z_j - c_j$	0	$\cdots$	0	$\cdots$	0	$\cdots$	$c_j - z_j$	$\cdots$	$c_k - z_k$	$\cdots$	$-c_B b$
x_{B_1}	1	$\cdots$	0	$\cdots$	0	$\cdots$	y_1^j	$\cdots$	y_1^k	$\cdots$	$\bar{b}_1$
$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$
x_{B_r}	0	$\cdots$	1	$\cdots$	0	$\cdots$	y_r^j	$\cdots$	y_r^k	$\cdots$	$\bar{b}_r$
$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$
x_{B_m}	0	$\cdots$	0	$\cdots$	1	$\cdots$	y_m^j	$\cdots$	y_m^k	$\cdots$	$\bar{b}_m$

Cuadro 1.2: Tabla del método símplex expandida.

Vemos que la tabla posee toda la información para realizar una iteración del método símplex. Supongamos que es la variable k la que tiene que entrar en la base y la variable r la que debe salir. A continuación describiremos los pasos a seguir para llevar a cabo una iteración del método símplex y obtener la tabla asociada a la nueva base. Lo que tendremos que hacer será:

- Actualizar las variables básicas (x_B) y sus valores ($\bar{b}$).
- Actualizar los costes reducidos ($c_j - z_j$).
- Actualizar las columnas y^j.

El procedimiento se denomina *pivotado*, ya que todos los cálculos pivotan alrededor del elemento y_r^k, y consiste en lo siguiente:

Coeficientes y_l^j. Para obtener $B^{-1}N$ se realiza la siguiente actualización de la tabla::

- Si $l = r$ se actualiza a $\frac{y_r^j}{y_r^k}$. Es decir, la fila r se actualiza dividiéndola por el pivote.

- Si $l \neq r$, y_l^j se actualiza a $y_l^j - \frac{y_r^j \cdot y_l^k}{y_r^k}$. La forma más fácil de llevar a cabo estas operaciones es mirando el cuadrado que forma en la tabla y_l^j y el pivote. Tendríamos que:

$$\begin{array}{ccc} y_l^j & \cdots & y_l^k \\ \vdots & \swarrow\nearrow & \vdots \\ y_r^j & \cdots & \boxed{y_r^k} \end{array} \qquad y_l^j \xrightarrow{\text{se actualiza a}} y_l^j - \frac{y_r^j \cdot y_l^k}{y_r^k}$$

Coeficientes $\bar{b}_l$. La actualización de la tabla para obtener $B^{-1}b$ se hace de la siguiente forma:

- Si $l = r$, $\bar{b}_r$ se actualiza a $\frac{\bar{b}_r}{y_r^k}$.

- Si $l \neq r$, $\bar{b}_l$ se actualiza a $\bar{b}_l - \frac{\bar{b}_r \cdot y_l^k}{y_r^k}$. Si aplicamos la misma regla utilizada anteriormente, tenemos que:

$$\begin{array}{ccc} y_l^k & \cdots & \bar{b}_l \\ \vdots & \swarrow\nearrow & \vdots \\ \boxed{y_r^k} & \cdots & \bar{b}_r \end{array} \qquad \bar{b}_l \xrightarrow{\text{se actualiza a}} \bar{b}_l - \frac{\bar{b}_r \cdot y_l^k}{y_r^k}$$

Costes reducidos en negativo $c_j - z_j$. Aplicando la regla del cuadrado tenemos que:

$$\begin{array}{ccc} c_j - z_j & \cdots & c_k - z_k \\ \vdots & \swarrow\nearrow & \vdots \\ y_r^j & \cdots & \boxed{y_r^k} \end{array} \qquad c_j - z_j \xrightarrow{\text{se actualiza a}} (c_j - z_j) - \frac{y_r^j \cdot (c_k - z_k)}{y_r^k}$$

Función objetivo $\bar{z}$. La actualización es análoga a las anteriores:

$$\begin{array}{ccc} c_k - z_k & \cdots & -c_B\bar{b} \\ \vdots & \swarrow\nearrow & \vdots \\ \boxed{y_r^k} & \cdots & \bar{b}_r \end{array} \qquad -c_B\bar{b} \xrightarrow{\text{se actualiza a}} -c_B\bar{b} - \frac{\bar{b}_r \cdot (c_k - z_k)}{y_r^k}$$

En cuanto a las columnas asociadas a las variables básicas, sus columnas se corresponden con las de la matriz identidad y sus costes reducidos son cero. A esta conclusión también podemos llegar realizando los cálculos que acabamos de describir para las columnas de las variables no básicas.

En cuanto a la matriz B^{-1}, la estamos calculando implícitamente en cada iteración. En cada iteración las columnas de la tabla representan a $B^{-1}A$, y ya que comenzamos el método con una base formada por los vectores de la matriz identidad, siempre tendremos debajo de los vectores que forman la primera base a B^{-1}, ya que $B^{-1}I_{m\times m} = B^{-1}$.

1.2.8. Ejemplo

En esta sección será resuelto mediante la tabla del símplex el problema modelo con el que estamos trabajando durante todo el capítulo.

Ejemplo 1.2.4 *Sea el problema de programación lineal*

$$\begin{array}{rl} max & 2x_1 + x_2 \\ sujeto\ a & x_1 + x_2 \leq 4 \\ & x_1 \quad\ \leq 3 \\ & x \geqq 0. \end{array}$$

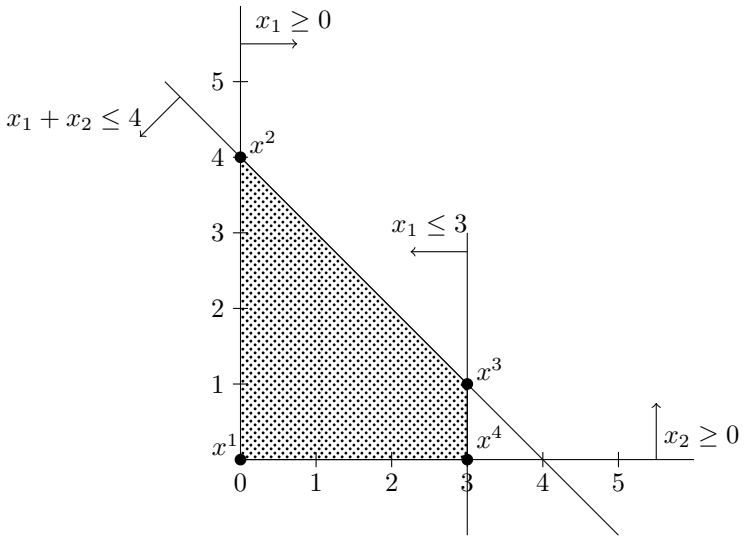


Figura 1.5: Región factible.

La región factible asociada a este problema se puede ver en la Figura 1.5.

Si añadimos las variables de holgura con el fin de tener el problema en forma estándar, tenemos que:

$$\begin{array}{rl} max & 2x_1+x_2 \\ sujeto\ a & x_1+x_2+x_3^s = 4 \\ & x_1 + x_4^s = 3 \\ & x \geq 0. \end{array}$$

Ya que este ejemplo tiene dos desigualdades ahora tenemos un problema en $\mathbb{R}^4$, por lo que no podemos dibujar su región factible. Lo que sí podemos pensar es que el polígono de la región factible representada anteriormente se convierte en el que mostramos en la Figura 1.6, donde la desigualdad $x_1 + x_2 \leq 4$ se convierte en $x_3^s \geq 0$, la $x_1 \leq 3$ en $x_4^s \geq 0$ y el resto de desigualdades no cambian.

Con esta transformación será más fácil conseguir los puntos que nos interesan (los vértices). Vemos que para conseguir el punto x^3 tenemos que intersecar las ecuaciones $x_1 + x_2 = 4$ y la $x_1 = 3$, cuyo costo computacional es grande; sin embargo, para conseguir el punto x'^3 tenemos que intersecar $x_3^s = 0$ y $x_4^s = 0$, cuyo costo computacional es mucho menor.

Pasemos ahora a la tabla del símplex:

	x_1	x_2	x_3^s	x_4^s	$\bar{z}$
$c_j - z_j$	2	1	0	0	0
					x_B
x_3^s	1	1	1	0	4
x_4^s	1	0	0	1	3

Cuadro 1.3

El punto de la región factible en la que estamos situados en esta situación es el $x = x'^1 = (0, 0, 4, 3)$, con $\bar{z} = 0$. Nuestro objetivo es movernos al vértice adyacente a x'^1 que mejore más nuestra función objetivo. Para ello tendremos que fijarnos en los $c_j - z_j$ y elegir el mayor de ellos, que en este caso es el que se corresponde con la variable x_1, que por tanto entrará en la base. Lo que nos queda por saber es la variable que tiene que salir de la base. Para ello podemos encontrar en la tabla del símplex

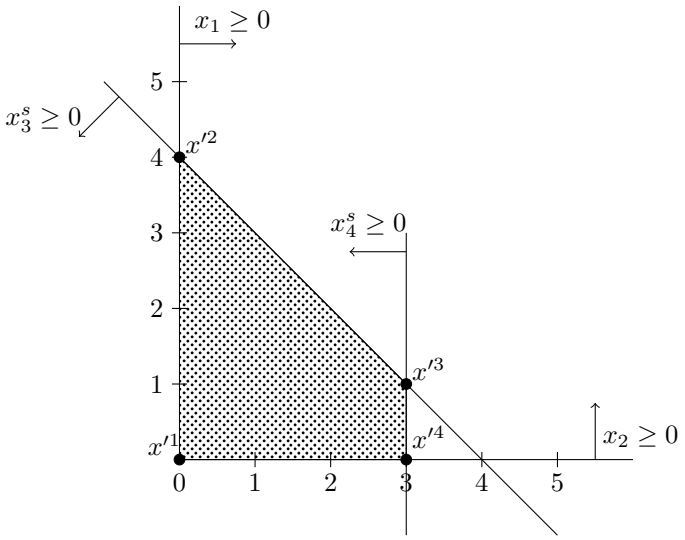


Figura 1.6: Región factible.

los vectores directores de las dos "aristas"que salen del punto x'^1 *en el que nos encontramos; siendo el vector* $(1, 0, -1, -1)$ *el que hace crecer* x_1 *y el* $(0, 1, -1, 0)$ *el que hace crecer* x_2 *a partir del punto en el que nos encontramos. En la Figura 1.7 lo podemos ver claramente.*

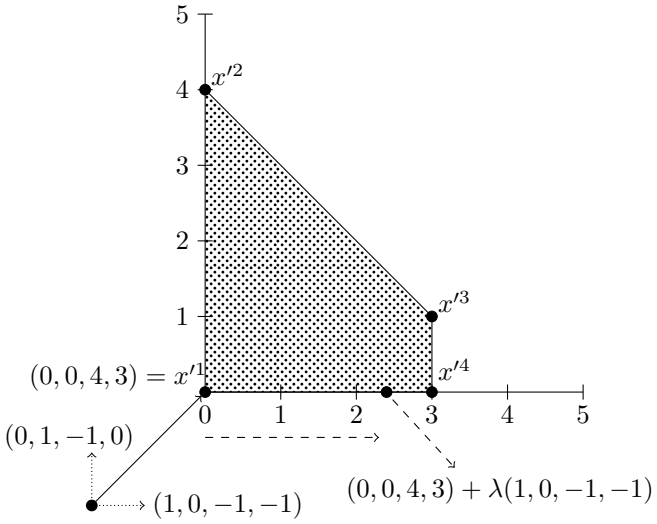


Figura 1.7: Región factible.

Una vez que sabemos esto, tenemos que calcular cuanto nos podemos desplazar en la dirección de x_1 *sin incumplir ninguna restricción. Lo que haremos será máximizar* λ *hasta el límite en que alguna de las componentes en* $(0, 0, 4, 3) + \lambda(1, 0, -1, -1)$ *se haga negativa. En este ejemplo vemos que cuando* $\lambda = 3$ *nos chocamos con la restricción* $x_4^s \geq 0$*, y cuando* $\lambda = 4$ *con la restricción* $x_3^s \geq 0$*; y ya que tenemos que respetar todas las restricciones nos quedaremos con el mínimo de los* λ *y será* x_4^s *la variable saliente. Lo que acabamos de hacer no es más que el mín* $\left\{ \frac{(x_B)_l}{y_l^k} : y_l^k > 0 \right\}$*. Una vez que sabemos que* x_1 *debe entrar en la base y* x_4^s *salir, llevaremos a cabo el pivotado consiguiendo la tabla del símplex siguiente (Cuadro 1.4).*

	x_1	x_2	x_3^s	x_4^s	$\bar{z}$
$z_j - c_j$	0	1	0	-2	-6
					x_B
x_3^s	0	1	1	-1	1
x_1	1	0	0	1	3

Cuadro 1.4

Utilizado el mismo argumento vemos en esta nueva tabla símplex que será la variable x_2 la que deba entrar en la base y la x_3^s la que debe salir, obteniendo una nueva tabla (Cuadro 1.5).

	x_1	x_2	x_3^s	x_4^s	$\bar{z}$
$z_j - c_j$	0	0	-1	-1	-7
					x_B
x_2	0	1	1	-1	1
x_1	1	0	0	1	3

Cuadro 1.5

Para finalizar podemos ver que en la Tabla 1.5 todos los $c_j - z_j$ son menores que 0, por lo que estaremos ante una solución óptima, donde $x = (3, 1, 0, 0)$ y $\bar{z} = 7$. Si nos quedamos con las variables del problema original el punto óptimo será el $x = (x_1, x_2) = (3, 1)$, siendo 7 el valor de la función objetivo en este punto.

Además, este ejemplo también ha sido resuelto utilizando el código descrito en el Apéndice A, dando lugar a los mismos resultados.
Los argumentos que se han utilizado en la función son los siguientes:

- $a = \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$
- b $= (4, 3)$
- c $= (2, 1)$
- operacion_opt = "max"
- signos = (" $\leq$ "," $\leq$ ").

Capítulo 2

Programación lineal multiobjetivo

La teoría de toma de decisiones multicriterio es una disciplina relativamente joven que nace de la interacción entre las matemáticas y la economía. Su origen se sitúa en los estudios que Jean-Charles de Borda y el Marqués de Condorcet llevan a cabo en el siglo XVIII sobre política y la preferencia en las votaciones. Posteriormente, ya en el siglo XIX, economistas como Antoine Cournot y Leon Walras analizarán los conceptos de conflicto y equilibrio y Wilfredo Pareto introducirá el concepto de eficiencia u optimalidad.

Una vez terminada la Segunda Guerra Mundial, la investigación operativa en general y la programación multiobjetivo en particular toman un gran impulso, apareciendo investigadores como John von Neumann, que realiza trabajos en teoría de juegos o Kuhn y Tucker, que formulan las condiciones de optimalidad para problemas no lineales y la existencia de soluciones eficientes en problemas multiobjetivos.

En 1955 Charnes, Cooper y Fergurson publican un artículo que contiene la esencia de la programación por metas. Además, otros investigadores como Bruno Contini y Stan Zionts se unieron a Cooper para desarrollar en 1968 el modelo de negociación multicriterio, lo que llevaría al inicio de la investigación que acabaría con el desarrollo del método iterativo para resolver problemas lineales multiobjetivo.

2.1. Propiedades de los problemas lineales multiobjetivos

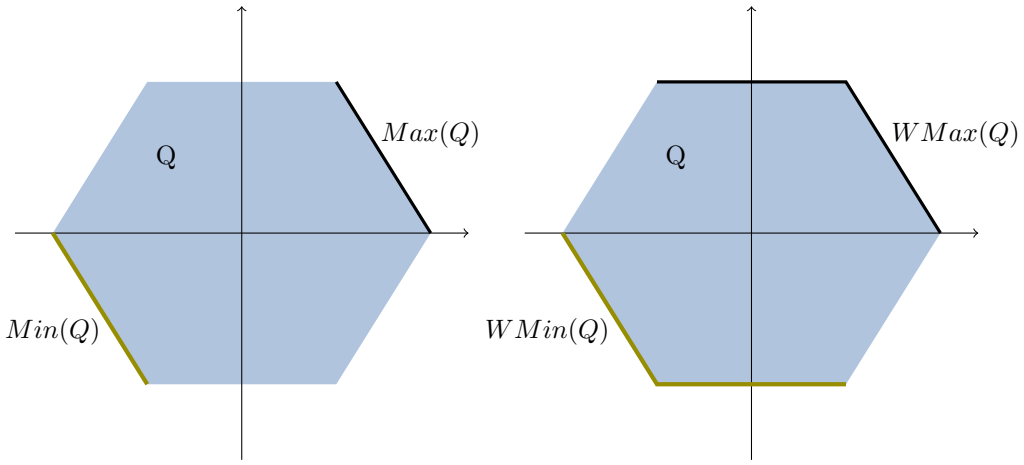
Durante esta sección desarrollaremos las principales propiedades de la programación lineal multiobjetivo, que será la base teórica para la introducción del método símplex multiobjetivo que se llevará a cabo en el siguiente capítulo.
Con el fin de facilitar la lectura, denominaremos S a los poliedros convexos y Q a cualquier subconjunto de $\mathbb{R}^k$.

2.1.1. Optimalidad de Pareto

Cuando nos situamos en el espacion $\mathbb{R}^n$, la relación de orden $x \geqq y$ significa que el vector $x - y$ no tiene elementos negativos. Este orden no es total, ya que existen vectores que no pueden ser comparables y por tanto, no podremos trabajar con la noción usual de máximos y mínimos. Notemos que $x > y$ significa que todos los elementos del vector $x - y$ son estrictamente positivos y $x \geq y$ significa que $x - y$ no tiene elementos negativos y que además $x \neq y$.

Esta relación de orden que acabamos de definir nos lleva al concepto de optimalidad según Pareto, que es la base fundamental de la investigación multiobjetivo, teoría que presentaremos en este capítulo.

Definición 2.1.1 *Sea Q un subconjunto de $\mathbb{R}^k$ no vacío. Un punto $x \in Q$ se denomina máximo según Pareto si no existe otro punto $x' \in Q$ tal que $x' \geq x$. Este punto se dice máximo débilmente si no*



(a) $Max(Q)$ y $Min(Q)$ según Pareto. (b) $WMax(Q)$ y $WMin(Q)$ según Pareto.

Figura 2.1: Puntos eficientes y débilmente eficientes.

existe $x' \in Q$ *tal que* $x' > x$.

El conjunto de punto máximos y máximos débilmente se denotarán por $Max(Q)$ y $WMax(Q)$ (Figura 2.1). De forma análoga, el conjunto de puntos mínimos se denotará por $Min(Q)$ y el de mínimos débilmente por $WMin(Q)$ (Figura 2.1). En caso de no existir confusión entre los puntos máximos y mínimos estos puntos serán denominados eficientes y débilmente eficientes, y los conjuntos que los contienen, conjuntos eficientes y conjuntos débilmente eficientes. Una cara de un poliedro convexo es eficiente si todos sus puntos lo son.

Geométricamente, un punto $x \in Q$ es eficiente si la intersección del conjunto Q con el ortante positivo movido a x consiste en sólo x, es decir

$$Q \cap (x + \mathbb{R}^k_+) = \{x\}$$

y es débilmente eficiente si la intersección de Q con el interior del ortante positivo movido a y es vacía, es decir,

$$Q \cap (x + \text{int}(\mathbb{R}^k_+)) = \emptyset$$

Además, es claro que los puntos eficientes son débilmente eficientes; sin embargo, lo contrario no se cumple. Veámoslo en los siguientes ejemplos.

Ejemplo 2.1.1 *Sea* S *un triángulo de vértices* $x_1 = (0,0)$, $x_2 = (1,0)$ *y* $x_3 = (0,1)$ *en* $\mathbb{R}^2$. *Tenemos que* $Max(S) = WMax(S) = [x_2, x_3]$, $Min(S) = \{x_1\}$ *y* $WMin(S) = [x_1, x_2] \cup [x_1, x_3]$. *Los puntos que acabamos de describir se muestran en la Figura 2.2a.*

Ejemplo 2.1.2 *Sea* S *el conjunto en* $\mathbb{R}^3$ *determinado por las desigualdades*

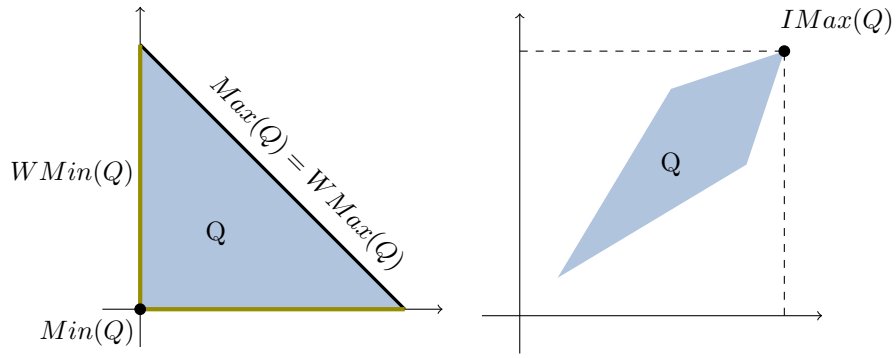
$$\begin{aligned} x_2 + x_3 &\geqq 0 \\ x_3 &\geqq 0. \end{aligned}$$

Entonces, $Max(S) = WMax(S) = \emptyset$, $Min(S) = \emptyset$ *y* $WMin(S) = S \setminus int(S)$.

Otro tipo de punto interesante es el denominado punto de utopía.

Definición 2.1.2 *Dado un conjunto* Q, *un punto* $x \in Q$ *es un punto ideal o punto de utopía si*

$$x \geq x' \text{ para todo } x' \in Q.$$



(a) Elementos importantes de Q. (b) $IMax(Q)$.

Figura 2.2

El punto de utopía suele ser inalcanzable y, en caso de existir, será único y lo denotaremos por $IMax(Q)$ (Fig. 2.2b).

Como hemos visto en el Ejemplo 2.1.2, podemos tener poliedros sin puntos máximos débilmente; esto ocurre cuando algunas componentes del conjunto no son acotadas. La existencia de puntos eficientes y débilmente eficiente puede ser comprobada fácilmente con la ayuda de funcionales[1].

Teorema 2.1.1 *Sea Q un conjunto no vacío y $\lambda \in \mathbb{R}^k$ un vector distinto de cero. Asumamos que $x \in Q$ maximiza el funcional $< \lambda, \cdot >$ en Q. Entonces,*

I) x es débilmente eficiente si λ es un vector positivo.

II) x es eficiente si: λ es un vector estrictamente positivo o, λ es un vector positivo y x es el único vector de Q que maximiza $< \lambda, \cdot >$.

En particular, si Q es compacto, entonces tiene puntos eficientes.

Demostración.

I) Sea $\lambda \geq 0$. Si x no fuese débilmente eficiente, entonces existiría otro vector $x' \in Q$ tal que $x' - x > 0$. Esto nos lleva a que $\lambda x' > \lambda x$, con lo que llegamos a una contradicción.

II) Sea $\lambda > 0$. Si x no fuese eficiente, existiría un vector $x' \in Q$ tal que $x' \geq x$, por lo que $\lambda x' > \lambda x$, con lo que llegamos a una contradicción.
Cuando $\lambda \geq 0$ la anterior desigualdad pasará a ser una desigualdad no estricta ($\lambda x' \geq \lambda x$). De hecho, ya que x es maximizador, tendremos una igualdad. Pero en este caso x' también sería un maximizador del funcional $< \lambda, \cdot >$ en Q, lo que contradice la hipótesis.

Cuando Q es compacto, cualquier vector estrictamente positivo λ produce un maximizador en Q, por tanto, un punto eficiente según Pareto. □

El vector λ que acabamos de mencionar se denomina *vector escalar débil* si es positivo y *vector escalar* si es estrictamente positivo. Además, un punto eficiente de Q puede maximizar varios vectores escalares independientes, y un vector escalar puede determinar varios maximizadores.

En el siguiente ejemplo veremos un caso donde maximizadores de un funcional $< \lambda, \cdot >$ con $\lambda \geq 0$, pero no estrictamente positivo, pueden no producir puntos eficientes.

Ejemplo 2.1.3 *Sea un conjunto S en $\mathbb{R}^3$ definido por los vectores $x = (x_1, x_2, x_3)$ con $x_3 \leq 0$. Sea $\lambda = (0, 0, 1)$. Entonces, cualquier elemento x de S con $x_3 = 0$ maximiza el funcional $< \lambda, \cdot >$ en S; sin embargo, estos maximizadores son puntos débilmente eficientes pero no eficientes. De hecho, el conjunto S no tiene puntos eficientes.*

[1] Un funcional lineal es un operador lineal en el que el rango es unidimensional.

Dado un punto a en el espacio, el conjunto de todos los elementos de Q mayores que a forman un subconjunto dominante, denominado *sección* de Q en a. Con el siguiente lema demostraremos que los puntos eficientes de una sección también serán puntos eficientes del conjunto dado.

Lema 2.1.1 *Sea un conjunto $Q \subset \mathbb{R}^k$ no vacío. Entonces, para todo punto de $\mathbb{R}^k$ tenemos que*

$$Max(Q \cap (a + \mathbb{R}^k_+)) \subseteq Max(Q)$$
$$WMax(Q \cap (a + \mathbb{R}^k_+)) \subseteq WMax(Q).$$

Demostración. Sea x un punto eficiente según Pareto de la sección $Q \cap (a + \mathbb{R}^k_+)$. Si el punto x no fuese eficiente del conjunto Q, entonces existiría un punto $x' \in Q$ tal que $x' \geq x$. Pero como $x' \geq x$, esto implica que x' pertenece a la sección $Q \cap (a + \mathbb{R}^k_+)$, por lo que llegamos a una contradicción.
La segunda inclusión se prueba análogamente. □

En el siguiente teorema veremos que la existencia de puntos eficientes en poliedros convexos se caracteriza por la posición de las direcciones con respecto al ortante positivo del espacio.

Teorema 2.1.2 *Sea S un poliedro convexo en $\mathbb{R}^k$. Las siguientes afirmaciones se cumplen:*

I) S tiene puntos eficientes si y sólo si

$$S_\infty \cap \mathbb{R}^k_+ = \{0\}.$$

II) S tiene puntos débilmente eficientes si y sólo si

$$S_\infty \cap int(\mathbb{R}^k_+) = \emptyset.$$

En particular, todo politopo[2] tiene vértices eficientes.

Demostración. Sea x un punto eficiente de S y d una dirección distinta de cero. Ya que $x + d$ pertenece a S y $S \cap (x + \mathbb{R}^k_+) = \{x\}$ por ser x eficiente, deducimos que d no pertenece a $\mathbb{R}^k_+$, y por tanto se cumple que $S_\infty \cap \mathbb{R}^k_+ = \{0\}$.
Para demostrar la otra implicación supongamos que S no tiene direcciones positivas. Entonces, fijado un vector $y \in Q$, la sección $Q \cap (y + \mathbb{R}^k_+)$ está acotada; si esto no fuese así, cualquier dirección asintótica distinta de cero de esta intersección cerrada y convexa, que existe ya que todo poliedro convexo es la suma de un poliedro acotado y su cono asintótico, sería un vector asintótico positivo de Q. Por tanto, dado $x \in S$, la sección $S \cap (x + \mathbb{R}^k_+)$ es acotada. Si ahora tenemos en cuenta el Teorema 2.1.1, el conjunto compacto $S \cap (x + \mathbb{R}^k_+)$ posee puntos eficientes, y por el Lema 2.1.1 también los poseerá S.
El punto II) no lo demostraremos ya que para su demostración hay que echar mano de un teorema de separación que no aparece en este trabajo. Su demostración puede ser consultada en The Luc (2016).
□

En el Ejemplo 2.1.3 hemos visto un caso donde maximizadores de un funcional no estrictamente positivo pueden no producir puntos eficientes. Sin embargo, en caso de tener asegurada la existencia de puntos eficientes, esto no ocurrirá.

Corolario 2.1.1 *Sea S un poliedro convexo y $\lambda \in \mathbb{R}^k$ un vector positivo distinto de cero. Si S tiene puntos eficientes y el funcional λ maximizadores en S, entonces alguno de los maximizadores será un punto eficiente de S.*

Demostración. Denotemos por S_0 al poliedro convexo definido por la intersección no vacía del conjunto S con el hiperplano $\{x \in \mathbb{R}^k : \lambda x = d\}$, donde d es el máximo de $< \lambda, \cdot >$ en S. Ya que S tiene puntos eficientes por definición, en vista del Teorema 2.1.2 tenemos que $S_\infty \cap \mathbb{R}^k_+ = \{0\}$, lo que implica que $(S_0)_\infty \cap \mathbb{R}^k_+ = \{0\}$. Por el mismo teorema, S_0 tiene al menos un punto eficiente, digamos x_0. Lo que

[2] Un politopo se define como la envoltura convexa de una cantidad finita de puntos.

nos queda por demostrar es que x_0 también es un punto eficiente de S.
Supongamos que x_0 no es un punto eficiente en S. Entonces existiría un punto $\hat{x} \in S$ tal que $\hat{x} \geq x_0$; y ya que $\lambda \geq 0$, tendríamos que $\lambda\hat{x} \geqq \lambda x_0 = d$. Pero por definición, los $x \in S_0$ maximizan λ en S, y ya que $\hat{x} \notin S_0$ tenemos que $\lambda\hat{x} < d$, por lo que llegamos a una contradicción. □

Otra forma de comprobar si un punto es eficiente es echando mano de las direcciones normales.

Teorema 2.1.3 *Sea $S \subset \mathbb{R}^k$ un poliedro convexo. Las siguientes afirmaciones se cumplen:*

I) $x \in S$ es un punto eficiente si y sólo si el cono normal a x en S ($N_S(x)$) contiene un vector estrictamente positivo.

II) $x \in S$ es un punto débilmente eficiente si y sólo si el cono normal a x en S ($N_S(x)$) contiene un vector positivo distinto de cero.

Demostración. Sea x un punto de S. Si el cono normal a S en x contiene un vector estrictamente positivo, llamémoslo λ, entonces por definición de vector normal tenemos que $\lambda(x' - x) \leq 0$ para todo $x' \in S$. Por lo tanto $\lambda x' \leq \lambda x$, y ya que $\lambda > 0$, si $x \neq x'$, la desigualdad será estricta ($\lambda x' < \lambda x$). Por tanto, λ alcanza su máximo en x y, por el Teorema 2.1.1, x será un punto eficiente de S.
La demostración del "sólo si" no la haremos en este trabajo ya que es necesaria la utilización del Teorema de Farkas. Esta parte de la demostración y el punto II), que se demuestra de forma análoga al I), puede ser consultada en The Luc (2016). □

A continuación mostraremos un ejemplo en el que se utiliza el teorema anterior con el fin de comprobar si ciertos puntos de un poliedro son eficientes.

Ejemplo 2.1.4 *Sea un poliedro convexo $S \subset \mathbb{R}^3$ determinado por el sistema*

$$\begin{aligned} x_1+x_2+x_3 &\leqq 1 \\ x_2+x_3 &\leqq 1 \\ x_1 \quad +x_3 &\leqq 1 \\ -x_3 &\leqq 0 \\ x_3 &\leqq 1 \end{aligned}$$

Hagamos la comprobación de si ciertos puntos son eficientes:

i) Sea un punto $y = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3}) \in S$. Su conjunto de índices activos es $I(y) = \{1\}$; y según el Teorema 1.1.1, el cono normal a S en el punto y es generado por el vector $(1, 1, 1)$. Si ahora tenemos en cuenta el Teorema 2.1.3, y es un punto eficiente de S.

ii) Sea el punto $z = (-1, 0, 1) \in S$. Su conjunto de índices activos consiste en los índices 2 y 5 ($I(z) = \{2, 5\}$). Por tanto, el cono normal será la envoltura positiva de los vectores $(0, 1, 1)$ y $(0, 0, 1)$. Y se ve de forma clara que este cono normal no contiene vectores estrictamente positivos, por lo que el punto z no será eficiente. Sin embargo, este punto sí es débilmente eficiente, ya que las direcciones normales a z son positivas.

iii) Para finalizar comprobemos el punto $w = (0, 0, 0)$, que tiene como conjunto de índices activos a $I(w) = \{4\}$. El cono normal a S en w es generado por el vector $(0, 0, -1)$, por lo que no contiene vectores positivos y, por tanto, w no es un punto débilmente eficiente.

Para finalizar esta sección mostraremos una proposición y un teorema que nos ayudará a entender la estructura de los conjuntos eficientes.
Tenemos que tener en cuenta que el conjunto de puntos eficientes no es un conjunto sencillo, de hecho normalmente no es convexo, e incluso una arista del conjunto puede no ser eficiente aún siendo sus dos puntos extremos vértices eficientes (Figura 2.3).

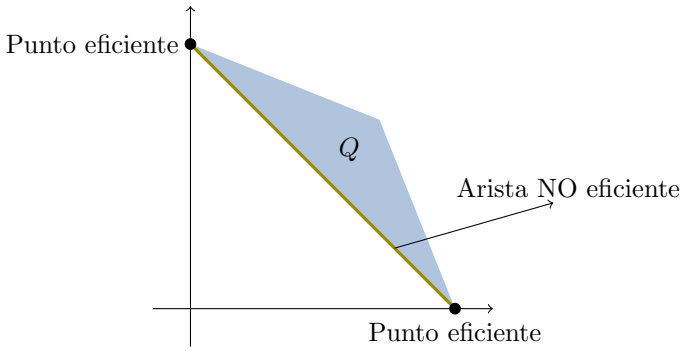


Figura 2.3

Proposición 2.1.1 *Sea S un poliedro convexo en $\mathbb{R}^k$. Las siguientes afirmaciones se cumplen.*

I) Si un punto relativamente interior de una cara de S es eficiente o débilmente eficiente, entonces también lo serán el resto de puntos de la cara.

II) Si S tiene vértices, entonces tendrá vértices eficientes/ débilmente eficientes siempre que S tenga puntos eficientes/ débilmente eficientes.

Demostración. Ya que el cono normal a S en cualquier punto de una cara contiene al cono normal en un punto relativamente interior de esa misma cara, el Teorema 2.1.3 nos asegura la veracidad de la afirmación I).
En cuanto a II), sea x un punto eficiente de S. Por ser x eficiente, existe $\lambda > 0$ tal que x es un maximizador de λ. Si x es un vértice ya tendremos probada la afirmación. En otro caso, x será un punto relativamente interior y eficiente, por tanto el resto de punto de la cara también serán eficientes. Además, esta cara tendrá algún vértice, que por tanto también será eficiente.
El caso débilmente eficiente se prueba de forma análoga. □

Teorema 2.1.4 *El conjunto de puntos eficientes y débilmente eficientes de un poliedro convexo consiste en caras del poliedro y es cerrado y conexo por caminos* [3].

Demostración. Probaremos el teorema para conjuntos eficientes, siendo la demostración análoga para los débilmente eficientes.
Según la proposición anterior, si un punto x es eficiente, todos los puntos de la cara que contiene a x como punto relativamente interior también lo serán. Por tanto $Max(S)$, si es no vacío, consistirá en caras de S. Además, las caras son cerradas, por lo que su unión también lo será.
La demostración de que el conjunto es conexo por caminos puede ser consultada en The Luc (2016). □

2.1.2. Problemas lineales multiobjetivo

Los problemas lineales multiobjetivos (MOLP) que estudiaremos a lo largo de este capítulo se formulan de la siguiente manera

$$\begin{aligned} \max \quad & Cx \\ \text{sujeto a} \quad & x \in S \end{aligned}$$

siendo S un poliedro convexo no vacío en $\mathbb{R}^n$ y C una matriz real de dimensión $k \times n$. El objetivo de este tipo de problemas será encontrar soluciones eficientes $\bar{x}$ tales que $C\bar{x} \in Max(C(S))$. Y esto

[3] Un conjunto $Q \in \mathbb{R}^k$ se dice conexo por caminos si dados dos puntos $y, z \in Q$, existe un número finito de puntos $y^0, ..., y^l$ dentro de Q tal que $y^0 = y$, $y^l = z$ y los segmentos $[y^i, y^{i+1}]$ con $i = 0, ..., l-1$ pertenecen a Q

significa que no existirá otra solución factible $x \in S$ tal que

$$C\bar{x} \leq Cx.$$

El conjunto de soluciones eficientes del MOLP se denotará por S(MOLP). Cuando x es una solución/punto eficiente, el vector Cx se llamará valor maximal o eficiente. De forma similar denotaremos por WS(MOLP) al conjunto de soluciones débilmente eficientes; es decir, el conjunto de soluciones cuya imagen mediante C pertenece a $WMax(C(S))$. Como hemos visto en secciones anteriores, es claro que soluciones eficientes serán débilmente eficientes pero lo inverso no tiene porqué cumplirse. A la hora de trabajar con este tipo de problemas los modelaremos en forma estándar, siendo ésta análoga a la utilizada en los problemas con un sólo objetivo

$$\begin{aligned} Ax &= B \\ x &\geqq 0 \end{aligned}$$

con $A^{m\times n}$ una matriz real, $b \in \mathbb{R}^m$ y $x \in \mathbb{R}^n$.

Antes de presentar el siguiente teorema, que nos asegurará la existencia de soluciones eficientes bajo ciertas condiciones, mostraremos una proposición que utilizaremos en la demostración de tal teorema.

Proposición 2.1.2 *Sea S un poliedro definido por $S = \{x : Ax \leqq b, x \geqq 0\}$, y sea L un operador lineal de $\mathbb{R}^n$ en $\mathbb{R}^m$. Entonces,*

$$L(S_\infty) = [L(S)]_\infty.$$

Pasemos ahora al teorema en cuestión.

Teorema 2.1.5 *Sea un MOLP con soluciones factibles. Las siguientes afirmaciones se cumplen.*

I) El problema tendrá soluciones eficientes si y sólo si

$$C(S_\infty) \cap \mathbb{R}^k_+ = \{0\}.$$

II) El problema tendrá soluciones débilmente eficientes si y sólo si

$$C(S_\infty) \cap int(\mathbb{R}^k_+) = \emptyset.$$

En particular, si todas las direcciones de S pertenecen al núcleo de C, es decir $C(S_\infty) = 0$, entonces el MOLP tendrá soluciones eficientes.

Demostración. Por definición, un MOLP tiene soluciones eficientes si y sólo si $C(S)$ tiene puntos eficientes; lo que, teniendo en cuenta el Teorema 2.1.2, es equivalente a que

$$[C(S)]_\infty \cap \mathbb{R}^k_+ = \{0\}.$$

Además, teniendo en cuenta la proposición anterior, el cono de $C(S)$ coincide con $C(S_\infty)$.
El punto II) se prueba de forma análoga al I). □

A continuación presentaremos un ejemplo con el fin de clarificar los conceptos que se acaban de presentar.

Ejemplo 2.1.5 *Sea el problema lineal multiobjetivo definido por el conjunto factible*

$$\begin{aligned} x_1 + x_2 - x_3 &= 5 \\ x_1 - x_2 &= 4 \\ x &\geqq 0. \end{aligned}$$

El conjunto factible S que acabamos de definir se puede escribir de forma paramétrica de la siguiente manera

$$S = \left\{ \begin{pmatrix} t+4 \\ t \\ 2t-1 \end{pmatrix} \quad : \quad t \geq 1/2 \right\}.$$

Por tanto, el cono de dicho conjunto se describe mediante el conjunto

$$S_\infty = \left\{ \begin{pmatrix} t \\ t \\ 2t \end{pmatrix} \quad : \quad t \geq 0 \right\}.$$

Una vez que tenemos descrito el conjunto S_∞ pongamos dos ejemplos de matriz C y comprobemos si tales MOLP tienen puntos eficientes.

i) Sea $C = \begin{pmatrix} 1 & 0 & 1 \\ -2 & -4 & 0 \end{pmatrix}$, entonces la imagen de S_∞ por la función C es el conjunto $C(S_\infty) = \left\{ \begin{pmatrix} 3t \\ -6t \end{pmatrix} \quad : \quad t \geq 0 \right\}$, que solo tiene al vector cero en común con el ortante positivo; por tanto, en vista del Teorema 2.1.5 el problema tendrá soluciones eficientes.

ii) Sea ahora $S' = \begin{pmatrix} -1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$. La imagen de S_∞ por C será el conjunto $C(S'_\infty) = \left\{ \begin{pmatrix} 0 \\ 2t \end{pmatrix} \quad : \quad t \geq 0 \right\}$, que no tiene puntos en común con el interior del ortante positivo, por lo que el problema tendrá puntos débilmente eficientes. Sin embargo, no poseerá puntos eficientes ya que la intersección de $C(S'_\infty)$ con el ortante positivo contiene vectores positivos.

Definición 2.1.3 *La función objetivo de un MOLP se dice acotada (respectivamente débilmente acotada) superiormente si no existe $d \in S_\infty$ distinto de cero tal que*

$$Cd \geq 0 \ (\textit{respectivamente } Cd > 0).$$

En caso de que la función objetivo sea acotada, podemos decir simplemente que el MOLP es acotado. Además, es obvio que todo problema acotado es débilmente acotado pero lo contrario no se cumple.

Utilicemos el siguiente ejemplo para aplicar la definición de MOLP acotado.

Ejemplo 2.1.6 *Sea el siguiente problema con dos objetivos*

$$\begin{array}{rrl} max & -3x_1+x_2+x_3 & \\ max & x_2 & \\ sujeto\ a & x_1-x_2 & =0 \\ & x_3 & =1 \\ & x & \geq 0 \end{array}$$

El conjunto factible y su cono pueden ser dados de la siguiente forma:

$$S = \left\{ \begin{pmatrix} t \\ t \\ 1 \end{pmatrix} \quad : \quad t \geq 0 \right\}$$

y

$$S_\infty = \left\{ \begin{pmatrix} t \\ t \\ 0 \end{pmatrix} \quad : \quad t \geq 0 \right\}.$$

Por tanto, para la dirección $d = (t, t, 0) \in S_\infty$ *tenemos que*

$$Cd = \begin{pmatrix} -2t \\ t \end{pmatrix} \not> 0.$$

Además, Cd *sólo es igual a cero si* $t = 0$ *y, en tal caso,* d *también sería igual a cero, por lo que la función objetivo es acotada.*

Corolario 2.1.2 *Un problema de programación lineal multiobjetivo tiene soluciones eficientes (respectivamente débilmente eficientes) si y sólo su función objetivo es acotada (respectivamente débilmente acotada).*

Pasemos ahora a demostrar un teorema que nos aporta un criterio para saber si un punto es eficiente teniendo en cuenta los vectores normales en este punto.

Teorema 2.1.6 *Sea* $\bar{x}$ *una solución factible del MOLP. Entonces,*

I) $\bar{x}$ *es una solución eficiente si y sólo si el cono normal* $N_S(\bar{x})$ *a* S *en* $\bar{x}$ *contiene algún vector* λC *con* $\lambda \in \mathbb{R}^k$ *un vector estrictamente positivo.*

II) $\bar{x}$ *es un punto débilmente eficiente si y sólo si el cono normal* $N_S(\bar{x})$ *a* S *en* $\bar{x}$ *contiene algún vector* λC *con* $\lambda \in \mathbb{R}^k$ *un vector positivo distinto de cero.*

Demostración.
Punto I):

"$\Leftarrow$" Sea el vector λC con $\lambda \in \mathbb{R}^k$ un vector estrictamente positivo y normal a S en $\bar{x}$, Entonces

$$\lambda C(x - \bar{x}) \leq 0 \text{ para todo } x \in S$$

que es equivalente a que

$$\lambda Cx \leq \lambda C\bar{x} \text{ para todo } x \in S$$

Por tanto, por el Teorema 2.1.1 el vector $C\bar{x}$ es un punto eficiente del conjunto $C(S)$; y por definición, $\bar{x}$ es un punto eficiente del MOLP.

"$\Rightarrow$" Sea $C\bar{x}$ un valor eficiente del conjunto $C(S)$, entonces, por el Teorema 2.1.3, el cono normal a $C(S)$ en $C\bar{x}$ contiene un vector estrictamente positivo que denotaremos por λ. Por tanto se tienen las siguientes equivalencias:

$$\lambda(Cx - C\bar{x}) \leq 0 \Leftrightarrow \lambda C(x - \bar{x}) \leq 0 \text{ para todo } x \in S.$$

Lo que demuestra que λC es normal a S en $\bar{x}$.

El punto II) se prueba de una forma similar. □

En la siguiente sección hablaremos sobre la escalarización, que es una de las principales formas de buscar soluciones de los MOLP.

2.1.3. Escalarización

Dado un vector $\lambda \in \mathbb{R}^k$ distinto de cero, definamos el problema lineal, que denotaremos por LP_λ y llamaremos problema escalarizado del MOLP, como

$$\begin{aligned} \max \quad & \lambda Cx \\ \text{sujetos a} \quad & x \in S \end{aligned}$$

En el resto de este capítulo mostraremos como de útil es la escalarización a la hora de resolver un MOLP.

Teorema 2.1.7 *Sea $\bar{x}$ una solución factible del MOLP. Entonces,*

I) $\bar{x}$ es eficiente si y sólo si existe un vector $\lambda \in \mathbb{R}^k$ estrictamente positivo tal que $\bar{x}$ es una solución óptima del problema escalarizado LP_λ.

II) $\bar{x}$ es un punto débilmente eficiente si y sólo si existe un vector $\lambda \in \mathbb{R}^k$ distinto de cero y positivo tal que $\bar{x}$ es una solución óptima del problema escalarizado LP_λ.

Demostración.

"⇐" Si $\bar{x}$ resuelve el problema LP_λ entonces, por la Proposición 1.1.3, λC pertenecerá al cono normal de S en $\bar{x}$; y ya que λ es estrictamente positiva, por el Teorema 2.1.6, el punto $\bar{x}$ será solución eficiente del MOLP.

"⇒" Sea $\bar{x}$ una solución eficiente del MOLP. Entonces, por el Teorema 2.1.6, existirá un vector estrictamente positivo λ tal que λC es un vector normal a S en $\bar{x}$. Esto implica que

$$\lambda C(x - \bar{x}) \leq 0 \Leftrightarrow \lambda Cx \leq \lambda C\bar{x} \text{ para todo } x \in S$$

por lo que $\bar{x}$ es una solución óptima del problema LP_λ.

El punto II) se prueba de una forma similar. □

Corolario 2.1.3 *Asumamos que dado un vector λ positivo y distinto de cero, el conjunto $C\bar{x}$, donde $\bar{x}$ es solución eficiente de LP_λ, tiene un sólo punto. Entonces, cualquier solución óptima de LP_λ es solución eficiente del MOLP.*

Demostración. Sea $\bar{x}$ una solución óptima del LP_λ y sea y una solución factible del MOLP tal que $Cy \geqq C\bar{x}$. Ya que λ es positivo tenemos que $\lambda C\bar{x} \leqq \lambda Cy$. Pero ya que $\bar{x}$ es solución óptima de LP_λ, lo que tendremos en realidad será una igualdad ($\lambda C\bar{x} = \lambda Cy$). Y ya que por hipótesis tenemos que $C\bar{x} = Cy$, hemos demostrado que $\bar{x}$ es una solución eficiente del MOLP. □

A continuación demostraremos un teorema que nos asegura que con la resolución de un número finito de LP_λ se pueden generar todas las soluciones eficientes y débilmente eficientes de un MOLP.

Proposición 2.1.3 *Existe un número finito de vectores λ^i, con $i = 1, ..., p$, estrictamente positivos (respectivamente positivos) tales que*

$$S(MOLP) = \bigcup_{i=1}^{p} S(LP_{\lambda^i})$$

$$\textit{respectivamente } WS(MOLP) = \bigcup_{i=1}^{p} S(LP_{\lambda^i})$$

siendo $S(LP_\lambda)$ el conjunto de soluciones óptima del problema LP_λ.

Demostración. Teniendo en cuenta la Proposición 1.1.3, tenemos que si una solución óptima del problema LP_λ es un punto relativamente interior de una cara del poliedro factible, entonces toda la cara será óptima. Si además, tenemos que λ es un vector estrictamente positivo, ya que el número de caras es finito, un número finito de tales vectores serán suficientes para generar todas las soluciones eficientes del MOLP.
El caso donde buscamos soluciones débilmente eficientes se tratará de forma análoga. □

Mostremos ahora ciertas propiedades interesantes de los conjuntos eficientes de los MOLP. En estas propiedades podemos ver cierta similitud con las vistas para el caso con un solo objetivo.

Teorema 2.1.8 *Los conjuntos eficientes de un MOLP tienes las siguientes propiedades:*

I) Si un punto relativamente interior de una cara de S es una solución eficiente o débilmente eficiente, entonces también lo será cualquier punto de la cara.

II) Si S tiene un vértice y el MOLP tiene soluciones eficientes (débilmente eficientes), entonces tendrá un vértice eficiente (débilmente eficiente).

III) El conjunto de soluciones eficientes y débilmente eficientes del MOLP está formado por caras del poliedro factible y es cerrado y conexo por caminos.

2.1.4. Una aplicación: los problemas de listas de espera quirúrgicas

Para demostrar la utilidad de la escalarización a la hora de encontrar puntos eficientes de los problemas lineales multiobjetivos resolveremos el problema planteado en Cerdá et al. (2001) utilizando el lenguaje AMPL.

En 1998 se estableció un máximo de seis meses en lista de espera para una intervención quirúrgica. De esta forma, un paciente que debiera ser intervenido antes de julio de 1998 tendría un límite en lista de espera de nueve meses, sin embargo, los pacientes que entren en lista de espera tras el 1 de Julio sólo podrán estar en lista de espera un máximo de seis meses. Cierto hospital en la Comunidad de Madrid tenía largas listas de espera en cuatro procesos quirúrgicos: cataratas (dependiente de oftalmología), y hallux valgus, desgarro interno de rodilla y osteoartrosis (dependiente de traumatología). En el hospital había suficientes camas y personal, sin embargo el número de quirófanos no era suficiente; por lo que, con el fin de cumplir la nueva normativa de listas de espera planteó el siguiente problema.

Problema 2.1.9 *El problema consiste en decidir el número de intervenciones de los tipos anteriormente citados se llevarán a cabo en* 1998 *en horario ordinario, extraordinario y en centros concertados de forma que se cumplan todas las restricciones.*

Tenemos que tener en cuenta algunos conceptos para resolver el problema:

- *Las intervenciones de cataratas se pueden hacer en horario ordinario y extraordinario.*
- *Hallux valgus y desgarro interno de rodilla se puede hacer en horario ordinario y en concertación.*
- *Osteoartrosis sólo se puede intervenir en horario ordinario.*

Las variables con las que trabajaremos en la resolución del problema son las siguientes:

- CL_k*: número de pacientes en lista de espera de cataratas el día* 1 *del mes* k*.*
- HL_k*: número de pacientes en lista de espera de hallux valgus el día* 1 *del mes* k*.*
- KL_k*: número de pacientes en lista de espera de desgarro interno de rodilla el día* 1 *del mes* k*.*
- OL_k*: número de pacientes en lista de espera de osteoartrosis el día* 1 *del mes* k*.*

Siendo CL_1*,* HL_1*,* KL_1*,* OL_1 *datos conocidos a día* 1 *de enero de* 1998*. Tenemos que tener en cuenta que* $k = 1, ..., 13$*, donde el mes* 13 *se refiere a enero de* 1999*.*

En cuanto a las intervenciones tenemos las siguientes variables:

- CR_i*: número de intervenciones de cataratas a realizar en el mes* i *en horario ordinario.*
- HR_i*: número de intervenciones de hallux valgus a realizar en el mes* i *en horario ordinario.*
- KR_i*: número de intervenciones de desgarro interno de rodilla a realizar en el mes* i *en horario ordinario.*
- OR_i*: número de intervenciones de osteoartrosis a realizar en el mes* i *en horario ordinario.*

- CO_i: *número de intervenciones de cataratas a realizar en el mes i en horario extraordinario.*
- HP_i: *número de intervenciones de hallux valgus a realizar en el mes i mediante concertación.*
- KP_i: *número de intervenciones de desgarro interno de rodilla a realizar en el mes i mediante concertación.*

La lista de espera en el mes k es la suma de los pacientes en lista de espera el mes $k-1$*, más los nuevos pacientes que entran en lista de espera menos el número de pacientes intervenidos menos el número de pacientes que salen de la lista de espera sin intervención. Las previsiones a un año de estos datos pueden ser consultadas en Cerdá et al. (2001) (páginas* 102-104*).*

Las restricciones del problema a resolver son las siguientes:

- Pacientes en lista de espera en el mes $i = 1, ..., 12$*.*

$$\begin{aligned} CL_{i+1} &= CL_i + CA_i - CE_i - CR_i - CO_i \\ HL_{i+1} &= HL_i + HA_i - HE_i - HR_i - HP_i \\ KL_{i+1} &= KL_i + KA_i - KE_i - KR_i - KP_i \\ OL_{i+1} &= OL_i + OA_i - OE_i - OR_i \end{aligned}$$

Siendo las condiciones iniciales $CL_1 = 480$*,* $HL_1 = 199$*,* $KL_1 = 132$ *y* $OL_1 = 128$*.*

- Quirófanos asignados a cada servicio:

$$\begin{aligned} &\textit{Oftalmología: } 80CR_i \leq OCQ_i \ , \ i = 1, ..., 12 \\ &\textit{Oftalmología } 85HR_i + 120KR_i + 160OR_i \leq TEQ_i \ , \ i = 1, ..., 12 \end{aligned}$$

La primera desigualdad expresa una cota superior de los minutos que pueden ser utilizados los quirófanos para intervenir cataratas y la segunda una cota superior de los minutos que pueden ser utilizados los quirófanos para intervenir hallux valgus, desgarro interno de rodilla y osteoartrosis. Los valores de OCQ_i*,* TEQ_i *pueden ser consultados en Cerdá et al. (2001) (página* 106*).*

- Cotas al número de procesos fuera de horario normal en el hospital:

$$\begin{aligned} CO_i &\leq l_i \\ HP_i &\leq m_i \\ KP_i &\leq n_i \end{aligned}$$

Donde $i = 1, ..., 12$ *y los valores de* l_i*,* m_i*,* m_i *pueden ser consultados en Cerdá et al. (2001) (página* 107*).*

- No más de 9 *meses en lista de espera:*

$$\begin{aligned} \sum_{i=1}^{k} (CR_i + CO_i) &\geq a_k \\ \sum_{i=1}^{k} (HR_i + HP_i) &\geq b_k \\ \sum_{i=1}^{k} (KR_i + KP_i) &\geq c_k \\ \sum_{i=1}^{k} (OR_i + CO_i) &\geq d_k \end{aligned}$$

donde a_k, b_k, c_k y d_k pueden ser consultados en Cerdá et al. (2001) (página 108).

Lo que quieren decir estas restricciones es que en cada mes, el número de intervenciones debe ser mayor que la suma de los pacientes en lista de espera de los meses anteriores.

-No más de 6 meses en lista de espera al final de 1998:

$$\begin{aligned} CL_{13} &\leq 395 \\ HL_{13} &\leq 69 \\ KL_{13} &\leq 77 \\ OL_{13} &\leq 57 \end{aligned}$$

Estos datos se obtienen a partir de la suma de las entradas menos las salidas sin intervención correspondientes a los últimos 6 meses del año.

- Todas las variables tienen que ser enteras, no negativas.

Las funciones objetivo que tenemos que optimizar son las siguientes:

$$\begin{aligned} Minf_1 &= 80CL_{13} + 85HL_{13} + 120KL_{13} + 160OL_{13} \\ Minf_2 &= 110852(\sum_{i=1}^{12} CR_i) + 125899(\sum_{i=1}^{12} HR_i) + 287973(\sum_{i=1}^{12} KR_i) + 853338(\sum_{i=1}^{12} OR_i) \\ &+ 123733(\sum_{i=1}^{12} CO_i) + 106605(\sum_{i=1}^{12} HP_i) + 141120(\sum_{i=1}^{12} KP_i) + 90584CL_{13} \\ &+ 58035HL_{13} + 148157KL_{13} + 537603OL_{13} \end{aligned}$$

donde la primera función objetivo minimiza la lista de espera al final de 1998, en concreto se pretende minimizar el tiempo de quirófano que queda libre a final de año; y la segunda función pretende minimizar costes, teniendo en cuenta las cirugías en horario ordinario, extraordinario, en concertación y el coste de las intervenciones que quedan pendientes para el siguiente año.

Como hemos visto a lo largo de esta sección, si dado un $\lambda > 0$, si $\bar{x}$ es una solución óptima del problema escalarizado, $\bar{x}$ será un punto eficiente del MOLP. Nosotros lo que haremos es resolver el problema escalarizado dándole un peso del 80 % a la primera función objetivo y un peso del 20 % a la segunda. De está forma, la función objetivo a optimizar es la siguiente:

$$\begin{aligned} Min\ &22170{,}4(\sum_{i=1}^{12} CR_i) + 25179{,}8(\sum_{i=1}^{12} HR_i) + 57594{,}6(\sum_{i=1}^{12} KR_i) + 170668(\sum_{i=1}^{12} OR_i) + 24746{,}6(\sum_{i=1}^{12} CO_i) \\ &+ 21321(\sum_{i=1}^{12} HP_i) + 28224(\sum_{i=1}^{12} KP_i) + 18181CL_{13} + 11675HL_{13} + 29727KL_{13} + 107649OL_{13} \end{aligned}$$

Una vez resolvemos el problema utilizando el solver CPLEX en el servidor NEOS de optimización, encontramos que no existe solución factible. Para solucionar este problema lo que haremos será asignar 5 consultas de oftalmología a traumatología, de esta forma conseguimos la factibilidad deseada. El número de pacientes en lista de espera y el número de operaciones realizadas se muestran en el cuadro 2.1.

El código AMPL utilizado para la resolución de este ejercicio puede ser visto en el apéndice B.

Mes/Año	CL	HL	KL	OL	CR	CO	HR	HP	KR	KP	OR
01/1998	480	199	132	128	64	0	11	0	0	0	17
02/1998	492	212	150	116	68	0	5	20	0	0	21
03/1998	496	207	167	115	69	0	1	25	1	8	23
04/1998	493	190	172	98	56	40	0	35	0	21	21
05/1998	475	171	163	84	68	64	0	34	2	21	22
06/1998	401	161	160	83	68	72	0	35	0	20	24
07/1998	328	140	144	76	27	0	0	35	2	20	17
08/1998	373	114	133	57	33	0	0	35	3	20	15
09/1998	370	88	122	42	25	44	0	35	2	20	15
10/1998	360	68	117	37	79	0	0	0	1	10	26
11/1998	359	80	115	32	68	0	0	0	29	10	2
12/1998	368	90	96	42	52	0	0	35	27	0	1
01/1999	395	69	77	57							

Cuadro 2.1: Número de pacientes en lista de espera y número de operaciones realizadas.

2.2. Método símplex multiobjetivo

En este capítulo vamos a explicar el método símplex multiobjetivo que, como en el caso de problemas con un sólo objetivo, es uno de los métodos más populares a la hora de resolver problemas lineales. Este capítulo nos servirá de introducción teórica a la realización de un paquete de R que incluirá este método.

Presentaremos el método símplex para problemas MOLP definidos en forma estándar

$$\begin{aligned} \max \quad & Cx \\ \text{sujeto a} \quad & Ax = b \\ & x \geqq 0, \end{aligned}$$

donde $C \in \mathbb{R}^{k\times n}$, $A \in \mathbb{R}^{m\times n}$ y $b \in \mathbb{R}^m$.

2.2.1. Descripción del método símplex multiobjetivo

El objetivo del método símplex, como explicamos para el caso con un sólo objetivo, es encontrar vértices eficientes y direcciones eficientes. Pudiendo dividir las fases del método en 4.

1º) Encontrar una solución factible $\bar{x}$. Este punto se llevará a cabo según hemos explicado en la sección 1.2.5.

2º) Comprobar si $\bar{x}$ es una solución eficiente.

3º) Moverse a algún vértice adyacente para encontrar vértices adyacentes eficientes o direcciones eficientes.

4º) Guardar los vértices eficientes y no eficientes y parar cuando todos los vértices hayan sido visitados.

Tenemos que tener en cuenta que ahora el elemento que multiplica a x en la función objetivo es una matriz C. Por tanto, dado una base B y separando la matriz objetivo C en C_B (submatriz que contiene a las columnas de C que están en la base B) y C_N (submatriz que contiene a las columnas no básicas de C), tenemos que

$$x_B = B^{-1}b$$
$$C_j - Z_j = C_j - C_B B^{-1} A_j$$

donde $j \in J$ denota a los índices de las variables no básicas. Con el fin de facilitar la notación, denotaremos a la matriz $C_N - Z_N$ como $\bar{C}_N$.

A continuación demostraremos un teorema que tendrá gran importancia a la hora de resolver un MOLP utilizando el método símplex.

Teorema 2.2.1 *Sea $\bar{x}$ una solución básica factible. Las siguientes condiciones son equivalentes.*

I) $\bar{x}$ es una solución eficiente.

II) La solución $\bar{x}$ resuelve el problema

$$\begin{array}{ll} max & < c^1 + \cdots + c^k, x > \\ sujeto\ a & Ax = b \\ & Cx \geqq C\bar{x} \\ & x \geqq 0. \end{array}$$

Además, si la solución $\bar{x}$ es no degenerada, I) es equivalente a la siguiente condición.

III) El siguiente sistema tiene solucion

$$\begin{array}{l} -\lambda \bar{C}_N \geqq 0 \\ \lambda > 0, \lambda \in \mathbb{R}^k. \end{array}$$

Demostración. Probemos primero la equivalencia entre I) y II):

"$I) \Rightarrow II$)" Si $\bar{x}$ es una solución eficiente, entonces para toda solución factible x del sistema en II) tendremos que $Cx = C\bar{x}$, lo que demuestra que $\bar{x}$ es una solución óptima.

"$II) \Rightarrow I$)" Supongamos que $\bar{x}$ no es una solución eficiente, entonces existirá $x \geqq 0$ con $Ax = b$ y $Cx \geq C\bar{x}$. Por tanto, tendremos que

$$< c^1 + \cdots + c^k, x > \ > \ < c^1 + \cdots + c^k, \bar{x} >$$

y por tanto $\bar{x}$ no será solución óptima del problema en II).

Demostremos a continuación la equivalencia entre I) y III).

"$I) \Leftrightarrow III$)" Según el Teorema 2.1.7, $\bar{x}$ es eficiente si y solo si existe un vector $\lambda \in \mathbb{R}^k$ estrictamente positivo tal que $\bar{x}$ es solución óptima del problema escalarizado siguiente

$$\begin{array}{ll} \max & \lambda Cx \\ \text{sujeto a} & Ax = b \\ & x \geqq 0. \end{array}$$

Asumamos que $\bar{x}$ es no degenerada. Teniendo el cuenta el Teorema 1.2.2 el problema anterior tendrá a $\bar{x}$ como solución óptima si y solo si el vector de costes reducidos $(\lambda \bar{C})_N$ contiene sólo valores no estrictamente positivos. Se tiene:

$$\begin{array}{l} (\lambda \bar{C})_N \leq 0 \Leftrightarrow (\lambda C)_N - (\lambda C)_B B^{-1} N \leq 0 \Leftrightarrow \lambda C_N - \lambda C_B B^{-1} N \leq 0 \Leftrightarrow \\ \Leftrightarrow \lambda \bar{C}_N \leq 0 \end{array}$$

Por tanto $-\lambda \bar{C}_N \geqq 0$, con $\lambda > 0$ y $\lambda \in \mathbb{R}^k$, llegando a que el sistema del punto III) tiene solución.

□

Corolario 2.2.1 *Dado un MOLP y sea $\bar{x}$ una solución factible, las siguientes afirmaciones son ciertas:*

i) Si todos los elementos de la matriz de costes reducidos $\bar{C}_N$ son negativos o cero, entonces el vértice actual $\bar{x}$ es una solución ideal.

ii) Si la matriz de costes reducidos tiene una fila estrictamente negativa, entonces $\bar{x}$ es eficiente.

iii) Si la matriz de costes reducidos tiene una columna positiva distinta de cero, entonces el vértice actual $\bar{x}$ no es eficiente.

Demostración.

"*i*)" Si todos los elementos de $\bar{C}_N$ son no estrictamente positivos, entonces para cualquier $\lambda \in \mathbb{R}^k$ estrictamente positivo, el sistema del punto III) del Teorema 2.2.1 se cumple, por lo que $\bar{x}$ será un punto eficiente. Además, ya que cada solución factible x del MOLP se puede descomponer en su parte básica y su parte no básica, tenemos que $Ax = B \Leftrightarrow Bx_B + Nx_N = b$. Por tanto, $x_B = B^{-1}b - B^{-1}Nx_N$ y

$$Cx = Cx_B + C_N x_N = C_B(B^{-1}b - B^{-1}Nx_N) + C_N x_N = C_B B^{-1}b + (C_N - c_B B^{-1}N)x_N = \\ = C_B B^{-1}b + \bar{C}_N x_N.$$

Ya que x_N es un vector positivo, tenemos que $\bar{C}_N x_N \leqq 0$, por lo que $C\bar{x} \geqq Cx$, y por tanto $\bar{x}$ es una solución ideal del problema.

"*ii*)" Sea $k \in \mathbb{R}^k$ un vector estrictamente positivo con todas las componentes iguales a uno excepto las que se corresponden con la fila negativa, que tomará un valor suficientemente grande; entonces, se cumplirá el sistema del punto III) del teorema anterior.

"*iii*)" Si existe una columna de $\bar{C}_N$ con elementos positivos y distinta de cero, entonces el sistema del punto III) de Teorema 2.2.1 no tiene solución, por lo que $\bar{x}$ no puede ser eficiente.

□

A continuación mostraremos un ejemplo en el que comprobaremos que si la matriz de costes reducidos tiene una fila negativa pero no estrictamente negativa, la solución $\bar{x}$ en tal punto no tiene porqué ser eficiente.

Ejemplo 2.2.1

$$\begin{array}{lrl} max & x_1 & \\ max & x_2 & \\ sujeto\ a & x_1 - x_2 + x_3 & = 0 \\ & x_2 + x_4 & = 1 \\ & x & \geq 0. \end{array}$$

La base que se corresponde con las variables x_2 y x_3 será dada por $B = \begin{pmatrix} -1 & 1 \\ 1 & 0 \end{pmatrix}$ *siendo* $B^{-1} = \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$. *La solución básica asociada es $\bar{x} = (0, 1, 1, 0)$ y la matriz de costes reducidos*

$$\begin{aligned} \bar{C}_{1,4} &= C_{1,4} - C_{2,3}B^{-1}A_{1,4} \\ &= \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} - \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \end{aligned}$$

Podemos ver claramente que ya que la primera columna de la matriz de costes reducidos $\bar{C}_{1,4}$ es positiva, la solución $\bar{x}$ no será eficiente. Sin embargo también tenemos que la segunda fila es negativa, pero al no serlo estrictamente negativa acabamos de ver que no nos asegura que el punto $\bar{x}$ sea eficiente.

Una vez que hemos comprobado si el punto en el que nos encontramos es eficiente o no, debemos mostrar que ocurre cuando nos movemos a otro vértice factible o a lo largo de cierta dirección. La forma de movernos a lo largo de la región factible es igual al caso donde sólo tenemos una función objetivo.

Sea k una variable no básica, si la columna $y^k = B^{-1}A_k$ es negativa tenemos que el rayo $R = \{(x_B, x_N) + (-y^k, e^k)x_k : x_k \geq 0\}$ es factible. En otro caso, A_k^c entra en la base y x_k pasa a ser una variable básica. Abandona la base la variable i con el índice más bajo de entre las que primero se anulen en el rayo R:

$$x_k = \frac{(x_B)_i}{y_i^k} = \text{mín}\left\{\frac{(x_B)_l}{y_l^k} : y_l^k > 0\right\}.$$

El siguiente teorema nos dirá que ocurre cuando nos movemos en la dirección de la variable k siendo y^k negativo.

Teorema 2.2.2 *Sea y^k una columna negativa, las siguientes afirmaciones son verdaderas.*

I) Si $\bar{x}$ no es eficiente, entonces todo elemento del rayo R no es eficiente.

II) Si $\bar{x}$ es eficiente y la columna $\bar{C}_k$ de la matriz de costes reducidos es cero, entonces la dirección R es eficiente.

III) Sea $\bar{x}$ un punto eficiente y la columna $\bar{C}_k$ negativa, entonces cualquier punto del rayo R es dominado por $\bar{x}$.

IV) Sea $\bar{x}$ un punto no degenerado y eficiente. Si la columna $\bar{C}_k$ tiene elementos tanto positivos como negativos, entonces el rayo R es eficiente si y sólo si el siguiente sistema tiene solución:

$$\begin{aligned} -\lambda\bar{C}_N &\geqq 0 \\ \lambda\bar{C}_N e^k &= 0 \\ \lambda \in \mathbb{R}^k, y &> 0, \end{aligned}$$

V) Si $\bar{x}$ no es eficiente y dos columnas no básicas $\bar{C}_k$ y $\bar{C}_r$ de la matriz de costes reducidos $\bar{C}_N$ correspondientes con columnas negativas y^k e y^r tiene componentes tanto positivas como negativas con $\alpha\bar{C}_k \geq \bar{C}_r$ con $a > 0$, entonces el rayo R correspondiente a la columna y^r está dominado.

Demostración.

I) Demostrémoslo por reducción al absurdo. Supongamos que $\bar{x}$ no es eficiente pero el rayo R si lo es. Ya que $\bar{x}$ es un vértice de una cara, el punto $(x_B, x_N) + (-y^k, e^k)x_k$ para algún $x_k > 0$ será un punto relativamente interior y además eficiente de una de las caras que contiene a $\bar{x}$. Si ahora tenemos en cuenta el Teorema 2.1.8, tenemos que $\bar{x}$ también debe ser eficiente, por lo que llegamos a una contradicción. Por tanto el rayo R no podrá ser eficiente.

Para demostrar lo que resta del teorema expresaremos $C(\bar{x} + (-y^k, e^k)x_k)$ en términos de $C\bar{x}$ y $\bar{C}_k$. De esta manera tendremos que

$$\begin{aligned} C(\bar{x} + (-y^k, e^k)x_k) &= (C_B, C_N)\begin{pmatrix} \bar{x}_B - y^k x_k \\ e^k x_k \end{pmatrix} \\ &= C_B x_B + (C_N e^k - C_B y^k)x_k \\ &= C\bar{x} + \bar{C}_k x_k. \end{aligned}$$

II) Asumamos que $\bar{x}$ es eficiente; por lo que si $\bar{C}_k = 0$, tendremos que $C(\bar{x} + (-y^k, e^k)x_k) = C\bar{x}$ y por tanto el rayo R será eficiente.

III) Sea $\bar{C}_k \leqq 0$, entonces $C(\bar{x} + (-y^k, e^k)x_k) \leqq C\bar{x}$, por lo que cualquier punto del rayo R será dominado por $\bar{x}$.

IV) La demostración de este punto puede ser consultada en The Luc (2016). No la demostraremos en este trabajo ya que para su demostración son necesarios conceptos de dualidad que no aparecen en este trabajo.

V) Por cada elemento $\bar{x} + (-y^k, e^k)x_k$ del rayo factible R correspondiente con la columna negativa y^k, elegiremos el elemento $\bar{x} + \alpha(-y^r, e^r)x_k$ del rayo correspondiente con la columna negativa y^r. Si ahora calculamos

$$\begin{aligned} C(\bar{x} + \alpha(-y^k, e^k)x_k) - C(\bar{x} + (-y^r, e^r)x_k) &= C(\bar{x}) + \alpha\bar{C}_k x_k - C(\bar{x}) - \bar{C}_r x_k \\ &= (\alpha\bar{C}_k - \bar{C}_r)x_k \geq 0 \end{aligned}$$

por lo que queda demostrado que el rayo $\bar{x} + (-y^r, e^r)x_k$ con $x_k \geqq 0$ es dominado.

□

Como ocurre con el caso con un sólo objetivo, en caso de que y^k no sea negativo, el rayo R no estará totalmente contenido en la región factible del MOLP. La parte factible del rayo será la que une el punto $\bar{x}$ con un nuevo punto $\hat{x} = \bar{x} + (-y^k, e^k)x_k$.

El teorema que demostraremos a continuación nos asegurará hacia donde vale la pena moverse si nos encontramos ante un vértice no eficiente.

Teorema 2.2.3 *Sean dos puntos $\bar{x}$ y $\hat{x} = \bar{x} + (-y^k, e^k)x_k$, y asumamos que y^k tiene elementos positivos. Las siguientes afirmaciones son ciertas.*

I) Si la columna $\bar{C}_k$ es negativa, entonces el nuevo vértice $\hat{x}$ está dominado por $\bar{x}$.

II) Sean otro punto $x' = \bar{x} + (-y^r, e^r)x_r$. Si tenemos que $\bar{C}_k x_k - \bar{C}_r x_r \geq 0$, entonces el vértice x' que entra en la base es no eficiente.

Demostración.

I) Ya que $C\hat{x} = C\bar{x} + \bar{C}_k x_k$, tenemos que si $\bar{C}_k$ es negativo, entonces $C\hat{x} \leqq C\bar{x}$, y por tanto $\hat{x}$ es un vértice dominado.

II) Sean $x_k, x_r > 0$, donde $\hat{x}$ y x' son dos puntos adyacentes a $\bar{x}$. Y ya que tenemos que $C\hat{x} - Cx' = \bar{C}_k x_k - \bar{C}_r x_r \geq 0$, entonces $C\hat{x} \geq Cx'$, por lo que x' no será un punto eficiente.

□

A continuación (Figura 2.4) mostraremos el esquema general del método símplex multiobjetivo.

Teorema 2.2.4 *Si todas las bases factibles de un MOLP son no degeneradas, entonces el algoritmo del símplex termina en un número finito de iteraciones.*

Demostración. El algoritmo del símplex recorre, como mucho, todos los vértices del conjunto factible; y ya que estamos trabajando con conjuntos factibles que son poliedros, siempre que no tengamos degeneración, el algoritmo terminará en un número finito de iteraciones. □

ESQUEMA MÉTODO SÍMPLEX MULTIOBJETIVO

INICIALIZACIÓN

Partimos de una base B que tiene asociada una solución básica factible $v^i = \bar{x} = (\bar{x}_B, \bar{x}_N)$ con $i = 1$, con $\bar{x}_B = B^{-1}b$ y $\bar{x}_N = 0$. Además, $Z_B = C_B B^{-1} b$.

PASO 1

Calculamos los valores $\bar{C}_N = C_j - Z_j$ e y^j para todas las variables no básicas $j \in J$, donde $Z_j = C_B B^{-1} A_j^c = C_B y^j$.

- Si $\bar{C}_N \leqq 0$, FIN. $\bar{x}$ es una solución ideal.
- Otro caso: ir al paso 2.

PASO 2

Comprobamos si alguna fila de $\bar{C}_N$ es estrictamente negativa o la suma de las filas de $\bar{C}_N$ es negativa.

- Si se cumple alguna de las dos condiciones, $\bar{x}$ es una solución eficiente. Ir al Paso 4.
- En otro caso: ir al paso 3.

PASO 3

Comprobamos si el punto factible $\bar{x}$ es eficiente o encontramos otro vértice que domina el punto actual resolviendo el siguiente problema lineal.

$$\begin{aligned} \max \quad & (c^1 + \cdots + c^k)x \\ \text{sujeto a} \quad & Ax = b \\ & Cx \geqq Cv^i \\ & x \geqq 0. \end{aligned}$$

- Si el sistema no tiene óptimo finito, FIN. El MOLP no tiene soluciones eficientes.
- Si el valor óptimo es finito y v^i es una solución óptima, entonces v^i es una solución eficiente del MOLP. Ir a Paso 4.
- Si $\hat{x}$ es un vértice óptimo con $C\hat{x} \geq Cv^i$, entonces v^i no es una solución eficiente y $\hat{x}$ sí lo es. Recalcular B, $\hat{x} = (\hat{x}_B, \hat{x}_N)$, Z_B y los vectores y^j. Ir al PASO 4.

PASO 4

Guardar el vértice eficiente en un conjunto que denominaremos V y los no eficiente en un conjunto que llamaremos NV.

PASO 5 Comprobar si para algún índice no básico k, $\bar{C}_k = 0$ e $y^k \leqq 0$, entonces el rayo $R = \{\bar{x} + (-y^k, e^k)x_k : x_k \geq 0\}$ es un rayo eficiente.

PASO 6 Encontrar todos los índices no básicos j del último vértice guardado que satisfagan que

i) $\bar{C}_j$ contiene tanto valores positivos como negativos.

ii) No hay otro índice no básico r cumpliendo *i)* tal que $\bar{C}_r x_r \geq \bar{C}_j x_j$.

PASO 7 Guardar todas la bases inexploradas al introducir las variables con índices j cumpliendo el PASO 6.

- Si no existe ninguna de estas bases, FIN.
- En otro caso, la variable con el índice j entra en la base y obtenemos v^{i+1}.

PASO 8 Actualizamos $i = i + 1$ y volvemos al PASO 1.

Figura 2.4: Esquema del método símplex multiobjetivo.

	x_B	x_N	$\bar{z}$
$C_j - Z_j$	0 $C_B - C_B B^{-1} B$	$\bar{C}_N = C_N - C_B B^{-1} N$	$-C_B B^{-1} b$
x_B	$I_{m\times m}$ $B^{-1}B$	$B^{-1}N$	$B^{-1}b$

Cuadro 2.2: Tabla del método símplex multiobjetivo.

2.2.2. Tabla del método símplex multiobjetivo

Al igual que en el caso con un sólo objetivo, trabajar con la tabla sel símplex nos permitirá resolver los MOLP de una forma más eficiente.

La tabla del símplex tiene la forma presentada en el Cuadro 2.2.

Podemos ver que esta tabla posee toda la información necesaria para llevar a cabo un iteración del método símplex. El único cambio con respecto al caso con un solo objetivo es que ahora los costes de la matriz objetivo forman una matriz y no un vector.

Una vez tenemos la tabla, comprobamos si nos encontramos ante un vértice $\bar{x}$ eficiente y, en caso de no poder comprobarlo mirando la matriz $\bar{C}_N$, tendremos que resolver el sistema II) del Teorema 2.2.1. Recordemos que el sistema a resolver es el siguiente:

$$\begin{aligned} \max \quad & (c^1 + \cdots + c^k)x \\ \text{sujeto a} \quad & Cx \geqq C\bar{x} \\ & Ax = b \\ & x \geqq 0. \end{aligned}$$

o lo que es lo mismo

$$\begin{aligned} \max \quad & (c^1 + \cdots + c^k)x \\ \text{sujeto a} \quad & yI - Cx = -C\bar{x} \\ & Ax = b \\ & y \geqq 0, x \geqq 0. \end{aligned}$$

La solución básica factible $(0\ \bar{x})$ tiene como base asociada a $B = \begin{pmatrix} I_{k\times k} & -C_B \\ 0 & B \end{pmatrix}$ y por tanto, $B^{-1} = \begin{pmatrix} I_{k\times k} & C_B B^{-1} \\ 0 & B^{-1} \end{pmatrix}$.

La tabla del método símplex asociada al problema se consigue multiplicando B^{-1} por la matriz $\begin{pmatrix} I & -C_B & -C_N & -C_B\bar{x} \\ 0 & B & N & b \end{pmatrix}$ y calculando los $z_j - c_j$.

Por tanto tenemos la siguiente (Cuadro 2.3):

Trabajando con el Cuadro 2.2 y con el Cuadro 2.3 tendremos todas las herramientas para llevar a cabo los distintos pasos del método símplex ya que, a la hora de hacer los cambios de base, estos se llevan a cabo de la mismo forma que en el caso con un sólo objetivo (Subsección 1.2.7).

2.2.3. Ejemplo

Para finalizar el capítulo resolvamos un ejemplo de MOLP utilizando el método símplex multiobjetivo.

	x_B	x_B	x_N	$\bar{z}$
$c_j - z_j$	$0_{1\times k}$	$0_{1\times m}$	$c_N - c_B B^{-1} N$	$-c_B B^{-1} b$
x_B	$I_{k\times k}$	$0_{k\times m}$	$-C_N + C_B B^{-1} N$	$0_{k\times 1}$
x_B	$0_{m\times k}$	$I_{m\times m}$	$B^{-1} N$	$B^{-1} b$

Cuadro 2.3

Ejemplo 2.2.2 *Sea el siguiente MOLP*

$$\begin{aligned} max \quad & 6x_1+4x_2+5x_3 \\ max \quad & x_3 \\ sujeto\ a \quad & x_1+ x_2+2x_3 \leqq 12 \\ & x_1+2x_2+ x_3 \leqq 12 \\ & 2x_1+ x_2+ x_3 \leqq 12 \\ & x \geqq 0. \end{aligned}$$

Con el fin de conseguir una base inicial, introduciremos 3 variables de holgura, quedando el problema de la siguiente manera:

$$\begin{aligned} max \quad & 6x_1+4x_2+5x_3 \\ max \quad & x_3 \\ sujeto\ a \quad & x_1+ x_2+2x_3 + x_4^s = 12 \\ & x_1+2x_2+ x_3 + x_5^s = 12 \\ & 2x_1+ x_2+ x_3 + x_6^s = 12 \\ & x \geq 0. \end{aligned}$$

Pasemos ahora a escribir la tabla del símplex, en la cual tendremos una base evidente formada por las variables de holgura.

	x_1	x_2	x_3	x_4^s	x_5^s	x_6^s	$\bar{z}$
$C_j - Z_j$	6	4	5	0	0	0	0
	0	0	1	0	0	0	0
							x_B
x_4^s	1	1	2	1	0	0	12
x_5^s	1	2	1	0	1	0	12
x_6^s	2	1	1	0	0	1	12

El punto de la región factible en el que nos encontramos es el $v^1 = x = (0, 0, 0, 12, 12, 12)$ *con* $\bar{z} = (0\ 0)$.
Este punto no es, claramente, un punto ideal , ya que no se cumple que $\bar{C}_N \leqq 0$. *Tampoco se cumple que alguna de las filas o la suma de las filas de* $\bar{C}_N$ *sea estrictamente negativa, por lo que para comprobar si* v^1 *es eficiente o encontrar otro punto eficiente que domine a* v^1 *tendremos que resolver el sistema del Paso 3 del esquema del método símplex.*
Una de las restricciones del sistema es $Cx \geqq Cv^1$, *es decir*

$$\begin{aligned} 6x_1 + 4x_2 + 5x_3 &\geqq 0 \\ x_3 &\geqq 0 \end{aligned}$$

	x_1	x_2	x_3	x_4^s	x_5^s	x_6^s	$\bar{z}$
$c_j - z_j$	6	4	6	0	0	0	0
$C_j - Z_j$	6	4	5	0	0	0	0
	0	0	1	0	0	0	0
							x_B
x_4^s	1	1	2	1	0	0	12
x_5^s	1	2	1	0	1	0	12
x_6^s	2	1	1	0	0	1	12

Cuadro 2.4

Pero como ya tenemos que $x \geqq 0$, estas ecuaciones son redundantes y el sistema a resolver será un problema lineal con un sólo objetivo en el que substituiremos las dos filas de las funciones objetivo del MOLP por su suma, es decir,

$$\begin{aligned} max \quad & 6x_1+4x_2+6x_3 \\ sujeto\ a \quad & x_1+ x_2+2x_3 + x_4^s = 12 \\ & x_1+2x_2+ x_3 + x_5^s = 12 \\ & 2x_1+ x_2+ x_3 + x_6^s = 12 \\ & x \geq 0. \end{aligned}$$

A continuación escribiremos la tabla del símplex (Tabla 2.4) y haremos una iteración del método.

Como los $c_j - z_j$ del problema lineal con un sólo objetivo no son menores o iguales que cero, llevaremos a cabo una iteración del método símplex. Escogemos la columna j tal que $c_j - z_j$ sea mayor, es decir, la columna de x_1 o la de x_3. Escojamos la de x_3 y veamos que variable es la que tiene que salir de la base. Para ello, tenemos que calcular mín $\left\{ \frac{(x_B)_l}{y_l^k} : y_l^k > 0 \right\}$, es decir, el mín $\left\{ \frac{12}{2}, \frac{12}{1}, \frac{12}{1} \right\} = \frac{12}{2}$ asociado a la variable x_4^s, que saldrá de la base. Si ahora llevamos a cabo una iteración del método símplex, donde la nueva base será $B = (A_3^c, A_5^c, A_6^c)$, la tabla quedará de la siguiente forma:

	x_1	x_2	x_3	x_4^s	x_5^s	x_6^s	$\bar{z}$
$c_j - z_j$	3	1	0	-3	0	0	-36
$C_j - Z_j$	7/2	3/2	0	$-5/2$	0	0	-30
	$-1/2$	$-1/2$	0	$-1/2$	0	0	-6
							x_B
x_3	1/2	1/2	1	1/2	0	0	6
x_5^s	1/2	3/2	0	$-1/2$	1	0	6
x_6^s	3/2	1/2	0	$-1/2$	0	1	6

La solución básica factible asociada a esta base es $v^2 = (0, 0, 6, 0, 6, 6)$. Como podemos ver, esta solución no es óptima para el problema monoobjetivo pero sí lo es para el MOLP, ya que la segunda fila de la matriz de costes reducidos tiene todos sus elementos estrictamente negativos; por tanto $v^2 \in V$ y $v_1 \in VN$.

En este momento nos encontramos en el paso 6 del esquema del símplex, por lo que buscaremos los índices no básicos tales que $\bar{C}_j$ contenga valores positivos y negativos, cumpliendo además que no hay otro índice no básico r tal que $\bar{C}_r x_r \geq \bar{C}_j x_j$. Estas dos columnas serán las de las variables x_1 y x_2. Calculemos x_1 y x_2: $x_1 = min \left\{ \frac{6}{1/2}, \frac{6}{1/2}, \frac{6}{3/2} \right\} = 4$ y $x_2 = min \left\{ \frac{6}{1/2}, \frac{6}{3/2}, \frac{6}{1/2} \right\} = 4$, y ya que $\bar{C}_1 = (7/2, -1/2)$ y $\bar{C}_2 = (3/2, -1/2)$, tenemos que $\bar{C}_1 x_1 \geq \bar{C}_2 x_2$, por lo que la solución básica obtenida desde v^2 introduciendo x_2 en la base está dominada por la solución obtenida desde v^2 introduciendo en la base la variable x_1. Introduzcamos por tanto x_1 en la base y hagamos una iteración en la tabla del símplex.

	x_1	x_2	x_3	x_4^s	x_5^s	x_6^s	$\bar{z}$
$c_j - z_j$	0	0	0	-2	0	-2	-48
$C_j - Z_j$	0	$1/3$	0	$-4/3$	0	$-7/3$	-44
	0	$-1/3$	0	$-2/3$	0	$1/3$	-4
							x_B
x_3	0	$1/3$	1	$2/3$	0	$-1/3$	4
x_5^s	0	$4/3$	0	$-1/3$	1	$-1/3$	4
x_1	1	$1/3$	0	$-1/3$	0	$2/3$	4

siendo la nueva base $B = (A_3^c, A_5^c, A_1^c)$.
Vemos claramente que aunque la matriz de costes reducidos no es estrictamente negativa, la suma de las filas de $\bar{C}_N$ *es negativa, por lo que teniendo en cuenta el Paso 2 del esquema del símplex, el punto* $v^3 = (4, 0, 4, 0, 4, 0)$ *es un punto eficiente, por lo que* $v^3 \in V$.

Si ahora volvemos a realizar los mismos pasos que en la iteración anterior podríamos introducir en la base las variables x_2 *o* x_6 *pero, si introducimos la variable* x_6 *estaríamos en el vértice* v^2*, que ya habíamos comprobado que es eficiente, por lo que introduciremos la variable* x_2. $x_2 = min\left\{\frac{4}{1/3}, \frac{4}{4/3}, \frac{4}{1/3}\right\} = 3$*, siendo* x_5^s *la variable que debe salir de la base. Introduciendo esta variable en la base tendremos la siguiente tabla del símplex con base* $B = (A_3^c, A_2^c, A_1^c)$.

	x_1	x_2	x_3	x_4^s	x_5^s	x_6^s	$\bar{z}$
$c_j - z_j$	0	0	0	-2	0	-2	-48
$C_j - Z_j$	0	0	0	$-5/4$	$-1/4$	$-9/4$	-45
	0	0	0	$-3/4$	$1/4$	$1/4$	-3
							x_B
x_3	0	0	1	$3/4$	$-1/4$	$-1/4$	3
x_2	0	1	0	$-1/4$	$3/4$	$-1/4$	3
x_1	1	0	0	$-1/4$	$-1/4$	$3/4$	3

Como en el caso de v^3*, este punto es eficiente ya que la suma de las filas de* $\bar{C}_N$ *es negativa. Por tanto,* $v^4 = (3, 3, 3, 0, 0, 0) \in V$.

Las variables con posibilidad de entrar en la base a estas alturas, ya que tienen en sus vectores de costes reducidos elementos positivos y negativos, son x_5 *y* x_6*. Si calculamos* x_5 *y* x_6 *vemos que ambas son iguales a* 4*, y ya que* $x_5\bar{C}_5 \geq x_6\bar{C}_6 \Leftrightarrow (-1, 1) \geq (-9, 1)$*, introducir* x_6 *nos llevaría a una solución dominada. Por tanto tendremos que introducir la variable* x_5*. Pero si introducimos* x_5 *la única variable que puede salir de la base es* x_2*, ya que el único elemento de* y^5 *positivo es el que se corresponde con la variable básica* x_2*, pero si hacemos este cambio de variable nos volveremos a encontrar en la base* $B = (A_3^c, A_5^c, A_1^c)$*, que se corresponde con el punto* v^3 *que ya habíamos comprobado que es eficiente. Por tanto hemos llegado al fin del algoritmo encontrando tres puntos eficientes que, escritos en función de las variables iniciales, son:* $(0, 0, 6)$*,* $(4, 0, 4)$ *y* $(3, 3, 3)$.

Además, este ejemplo también ha sido resuelto utilizando el código descrito en el Apéndice C, dando lugar a los mismos resultados.
Los argumentos que se han utilizado en la función son los siguientes:

- $a = \begin{pmatrix} 6 & 4 & 5 \\ 0 & 0 & 1 \end{pmatrix}$
- b $= (12, 12, 12)$
- $a = \begin{pmatrix} 1 & 1 & 2 \\ 1 & 2 & 1 \\ 2 & 1 & 1 \end{pmatrix}$

- operacion_opt = (”max”, ”max”)
- signos = (” $\leq$ ”,” $\leq$ ”,” $\leq$ ”).

Apéndice A

Código símplex

simplex_monoobjetivo_latex.r

```r
simplex3 <- function(a, b, c , operacion_opt, signos){
  if(operacion_opt == "min"){
    c <- -c
  }
  n_col <- dim(a)[2]
  n_row <- dim(a)[1]
  n_col_h <- n_col
  for(i in 1:length(signos)){
    if(signos[i] == "<="){
      n_col_h <- n_col_h + 1
    }else if(signos[i] == "="){
      n_col_h <- n_col_h + 1
    }else{
      n_col_h <- n_col_h + 2
    }
  }
  a_h <- matrix(nrow=n_row, ncol=n_col_h)
  n_col2 <- n_col
  var_artificiales <- c()
  var_basicas <- numeric(n_row)
  for(i in 1:n_row){
    if(signos[i] == "<="){
      a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), 1, rep(0, n_col_h - n_col2 - 1)))
      n_col2 <- n_col2 + 1
      var_basicas[i] <- n_col2
    }else if(signos[i] == ">="){
      a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), -1, 1, rep(0, n_col_h - n_col2 - 2)))
      var_artificiales <- c(var_artificiales, n_col2 + 2)
      n_col2 <- n_col2 + 2
      var_basicas[i] <- n_col2
    }else{
      a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), 1, rep(0, n_col_h - n_col2 - 1)))
      var_artificiales <- c(var_artificiales, n_col2 + 1)
      n_col2 <- n_col2 + 1
      var_basicas[i] <- n_col2
    }
  }
  var_totales <- c(1:n_col_h)
  c_h <- numeric(n_col_h)
  c_h[1:n_col] <- c(c)
  c_h[var_artificiales] <- -1111
  x_B <- b
  c_B <- c_h[var_basicas]
  z <- sum(c_B * b)
```

```
  conjunto_resultados <- list()
  l = 0
  c_hh <- c_h
  if(sum(c_B != 0) != 0){
    for(i in 1:length(var_artificiales)){
      c_hh <- c_hh - (a_h[which(var_artificiales[i] == var_basicas),] *
              c_h[var_artificiales[i]] / a_h[which(var_artificiales[i] == var_basicas)
              , var_artificiales[i]])
    }
    c_j <- c_hh
    z_j <- integer(n_col_h)
    z_j[var_basicas] <- c_hh[var_basicas] * b
    bar_c <- c_j - z_j
    bar_cc <- bar_c[-var_basicas]
    a_hh <- matrix(nrow = n_row, ncol = n_col_h)
    bb <- numeric(n_row)
  }else{
    c_j <- c_hh
    z_j <- integer(n_col_h)
    z_j[var_basicas] <- c_h[var_basicas] * b
    bar_c <- c_j-z_j
    bar_cc <- bar_c[-var_basicas]
    a_hh <- matrix(nrow = n_row, ncol = n_col_h)
    bb <- numeric(n_row)
  }
  while(max(bar_cc) > 0){
    var_entrada <- min(which(bar_c == max(bar_c)))
    y_k <- a_h[,var_entrada]
    if(sum(y_k > 0) > 0){
      var_salida <- min(var_basicas[which(x_B / y_k == min(x_B[y_k > 0] /
                    y_k[y_k > 0]) & y_k > 0)])
      var_salida_posicion <- min(which(var_salida == var_basicas))
      pivote <- a_h[var_salida_posicion, var_entrada]
      a_hh[var_salida_posicion, ] <- a_h[var_salida_posicion, ] / pivote
      if(class(a_h[-var_salida_posicion, ]) == "numeric"){
        a_hh[-var_salida_posicion, ] <- a_h[-var_salida_posicion, ] -
                                        (a_h[var_salida_posicion, ] *
                                        a_h[-var_salida_posicion, var_entrada] / pivote)
      }else{
        for(i in 1:(n_row - 1)){
          a_hh[-var_salida_posicion, ][i, ] <- a_h[-var_salida_posicion, ][i, ] -
                                    (a_h[var_salida_posicion, ]*
                                    a_h[-var_salida_posicion, var_entrada][i] / pivote)
        }
      }
      bb[var_salida_posicion] <- b[var_salida_posicion] / pivote
      bb[-var_salida_posicion] <- b[-var_salida_posicion] - (b[var_salida_posicion] *
      a_h[-var_salida_posicion, var_entrada] / pivote)
      b <- bb
      x_B <- b
      bar_c <- bar_c - (a_h[var_salida_posicion, ] * bar_c[var_entrada] / pivote)
      a_h[var_salida_posicion,] = a_hh[var_salida_posicion, ]
      a_h[-var_salida_posicion,] = a_hh[-var_salida_posicion, ]
      var_basicas[which(var_basicas == var_salida)] <- var_entrada
      var_basicas
      bar_cc <- bar_c[-var_basicas]
      c_B <- c_h[var_basicas]
      z <- sum(c_B * b)
    }else{
      punto_optimo_todas_variables <- numeric(n_col_h)
      punto_optimo_todas_variables[var_basicas] <- x_B
      direccion_optima <- numeric(n_col_h)
      direccion_optima[var_basicas] <- -y_k
      direccion_optima[var_entrada] <- 1
      resultados <- list( c("Estamos ante un rayo con valores que aumentan
```

```
                infinitamente", punto_optimo_todas_variables, "+" , direccion_optima))
      return(resultados)
    }
  }
  punto_optimo <- numeric(n_col)
  for(i in 1:n_col){
    if(sum(var_basicas == i) == 0){
      punto_optimo[i] <- 0
    }else{
      punto_optimo[i] <- x_B[which(var_basicas == i)]
    }
  }
  l <- l + 1
  conjunto_resultados[[l]] <- punto_optimo
  while(max(bar_cc) == 0){
    var_entrada <- var_totales[-var_basicas][min(which(bar_c[-var_basicas] ==
                   max(bar_c[-var_basicas])))]
    y_k <- a_h[ ,var_entrada]
    if(sum(y_k > 0) > 0){
      var_salida <- min(var_basicas[which(x_B / y_k == min(x_B[y_k > 0]
                    / y_k[y_k > 0]))])
      var_salida_posicion <- min(which(var_salida == var_basicas))
      pivote <- a_h[var_salida_posicion, var_entrada]
      a_hh[var_salida_posicion, ] <- a_h[var_salida_posicion, ] / pivote
      if(class(a_h[-var_salida_posicion,]) == "numeric"){
        a_hh[-var_salida_posicion, ] <- a_h[-var_salida_posicion, ] -
                                        (a_h[var_salida_posicion, ] *
                                        a_h[ -var_salida_posicion, var_entrada] / pivote)
      }else{
        for(i in 1:(n_row - 1)){
          a_hh[ -var_salida_posicion, ][i, ] <- a_h[-var_salida_posicion, ][i, ] -
                                    (a_h[var_salida_posicion, ] *
                                     a_h[-var_salida_posicion, var_entrada][i] / pivote)
        }
      }
      bb[var_salida_posicion] <- b[var_salida_posicion] / pivote
      bb[-var_salida_posicion] <- b[-var_salida_posicion] - (b[var_salida_posicion] *
      a_h[-var_salida_posicion, var_entrada] / pivote)
      b <- bb
      x_B <- b
      bar_c <- bar_c - (a_h[var_salida_posicion, ] * bar_c[var_entrada] / pivote)
      a_h[var_salida_posicion, ] = a_hh[var_salida_posicion, ]
      a_h[-var_salida_posicion, ] = a_hh[-var_salida_posicion, ]
      var_basicas[which(var_basicas == var_salida)] <- var_entrada
      bar_cc <- bar_c[-var_basicas]
      c_B <- c_h[var_basicas]
      z <- sum(c_B * b)
      punto_optimo <- numeric(n_col)
      for(i in 1 : n_col){
        if(sum(var_basicas == i) == 0){
          punto_optimo[i] <- 0
        }else{
          punto_optimo[i] <- x_B[which(var_basicas == i)]
        }
      }
      l <- l + 1
      conjunto_resultados[[l]] <- punto_optimo
      if(is.element(conjunto_resultados[l], conjunto_resultados) == TRUE){
        resultados <- list("El valor optimo es" = z,
                           "El punto optimo" = conjunto_resultados)
        return(resultados)
      }
    }else{
      punto_optimo_todas_variables <- numeric(n_col_h)
      punto_optimo_todas_variables[var_basicas] <- x_B
```

```
175       direccion_optima <- numeric(n_col_h)
176       direccion_optima[var_basicas] <- -y_k
177       direccion_optima[var_entrada] <- 1
178       resultados <- list( "El valor optimo es" = z, c("Estamos ante un rayo optimo
179                   factible", punto_optimo_todas_variables, "+" , direccion_optima))
180       return(resultados)
181     }
182   }
183   if(length(which(var_basicas %in% var_artificiales)) != 0){
184     return("Problema sin soluciones factibles")
185   }else{
186     if(operacion_opt == "max"){
187       resultados <- list("El valor optimo es" = z,
188                   "El punto optimo" = conjunto_resultados)
189       return(resultados)
190     }else{
191       resultados <- list("El valor optimo es" = -z,
192                   "El punto optimo" = conjunto_resultados)
193       return(resultados)
194     }
195   }
196 }
```

Listing A.1: Código r

Apéndice B

Código AMPL

Los diferentes archivos que aquí se presentan contienen el código utilizado para la resolución de problema 2.1.9.

datos_espera.dat

```
set meses_control := Enero_1998 Febrero_1998 Marzo_1998 Abril_1998 Mayo_1998
    Junio_1998 Julio_1998 Agosto_1998 Septiembre_1998 Octubre_1998 Noviembre_1998
    Diciembre_1998;
set meses_estado := Enero_1998 Febrero_1998 Marzo_1998 Abril_1998 Mayo_1998 Junio_1998
     Julio_1998 Agosto_1998 Septiembre_1998 Octubre_1998 Noviembre_1998 Diciembre_1998
    Enero_1999;

# Pacientes incorporados a la lista de espera en el mes i
param :              CA    HA    KA    OA :=
Enero_1998           84    28    21    10
Febrero_1998         85    28    22    22
Marzo_1998           82    22    18    15
Abril_1998           94    22    15    14
Mayo_1998            78    34    30    30
Junio_1998           104   45    18    24
Julio_1998           125   31    15     5
Agosto_1998          42    12    12     5
Septiembre_1998      78    20    24    17
Octubre_1998         98    24    18    34
Noviembre_1998       94    12    21    14
Diciembre_1998       86    33    13    21
;

# Pacientes excluidos la lista de espera en el mes i
param :              CE    HE    KE    OE :=
Enero_1998            8     4     3     5
Febrero_1998         13     8     5     2
Marzo_1998           16    13     4     9
Abril_1998           16     6     3     7
Mayo_1998            20    10    10     9
Junio_1998           37    31    14     7
Julio_1998           53    22     4     7
Agosto_1998          12     3     0     5
Septiembre_1998      19     5     7     7
Octubre_1998         20    12     9    13
Noviembre_1998       17     2     1     2
Diciembre_1998        7    19     5     5
;

# Cotas al número de procesos fuera del horario normal (asignamos 5 consultas de
    oftalmología a traumatologia por mes).
```

```
param :              OCQ     TEQ   :=
Enero_1998           5120    3655
Febrero_1998         5440    3792
Marzo_1998           5520    3886
Abril_1998           4480    3382
Mayo_1998            5440    3792
Junio_1998           5440    3886
Julio_1998           2160    2972
Agosto_1998          2640    2784
Septiembre_1998      2000    2647
Octubre_1998         6320    4292
Noviembre_1998       5440    3834
Diciembre_1998       4160    3424
;

param :              l    m    n   :=
Enero_1998            0    0    0
Febrero_1998          0   20    0
Marzo_1998           68   25    8
Abril_1998           40   35   21
Mayo_1998            64   35   21
Junio_1998           72   35   20
Julio_1998            0   35   20
Agosto_1998           0   35   20
Septiembre_1998      44   35   20
Octubre_1998         52   35   10
Noviembre_1998       48   35   10
Diciembre_1998       24   35    0
;

param :              a    b    c    d :=
Enero_1998           11   9    4    3
Febrero_1998         26   24   23   17
Marzo_1998           50   62   35   34
Abril_1998          116   92   49   38
Mayo_1998           153  103   59   42
Junio_1998          224  130   82   61
Julio_1998          309  153  100   94
Agosto_1998         398  165  119  107
Septiembre_1998     480  199  132  128
Octubre_1998        556  223  150  133
Noviembre_1998      628  243  167  153
Diciembre_1998      694  252  181  159
;
```

Listing B.1: Código AMPL

modelo_espera.mod

```
# ********** C O N J U N T O S **********#

set meses_control ordered;
set meses_estado ordered;

# ********** P A R A M E T R O S **********#

param CA{meses_control} integer;
param HA{meses_control} integer;
param KA{meses_control} integer;
param OA{meses_control} integer;

param CE{meses_control} integer;
param HE{meses_control} integer;
param KE{meses_control} integer;
param OE{meses_control} integer;
```

```
17
18 param OCQ{meses_control} integer;
19 param TEQ{meses_control} integer;
20
21 param l{meses_control} integer;
22 param m{meses_control} integer;
23 param n{meses_control} integer;
24
25 param a{meses_control} integer;
26 param b{meses_control} integer;
27 param c{meses_control} integer;
28 param d{meses_control} integer;
29
30 # *************** V A R I A B L E S ***************************#
31
32 # Variables de control
33 var CR{meses_control}>=0 integer;
34 var HR{meses_control}>=0 integer;
35 var KR{meses_control}>=0 integer;
36 var OR{meses_control}>=0 integer;
37 var CO{meses_control}>=0 integer;
38 var HP{meses_control}>=0 integer;
39 var KP{meses_control}>=0 integer;
40
41 # Variables de estado
42 var CL{meses_estado}>=0 integer;
43 var HL{meses_estado}>=0 integer;
44 var KL{meses_estado}>=0 integer;
45 var OL{meses_estado}>=0 integer;
46
47 # *************** F U N C I O N   O B J E T I V O *************#
48
49 minimize fobj:22170.4*sum{i in meses_control}CR[i]+25179.8*sum{i in meses_control}HR[i
       ]+57594.6*sum{i in meses_control}KR[i] + 170668*sum{i in meses_control}OR[i] +
       24746.6*sum{i in meses_control}CO[i] + 21321*sum{i in meses_control}HP[i] + 28224*
       sum{i in meses_control}KP[i] + 18181*CL['Enero_1999'] + 11675*HL['Enero_1999']  +
       29727*KL['Enero_1999'] + 107649*OL['Enero_1999'];
50
51 # *************** R E S T R I C C I O N E S ******************#
52
53 ec_estado_cataratas_1{j in 1..12}:     CL[member(j+1,meses_estado)] = CL[member(j,
       meses_estado)] + CA[member(j,meses_estado)] - CE[member(j,meses_estado)] - CR[
       member(j,meses_estado)]- CO[member(j,meses_estado)];
54
55 ec_estado_hallux_1{j in 1..12}: HL[member(j+1,meses_estado)] = HL[member(j,
       meses_estado)] + HA[member(j,meses_estado)] - HE[member(j,meses_estado)] - HR[
       member(j,meses_estado)] - HP[member(j,meses_estado)];
56
57 ec_estado_rodilla_1{j in 1..12}: KL[member(j+1,meses_estado)] = KL[member(j,
       meses_estado)] + KA[member(j,meses_estado)] - KE[member(j,meses_estado)] - KR[
       member(j,meses_estado)] - KP[member(j,meses_estado)];
58
59 ec_estado_osteoartrosis_1{j in 1..12}: OL[member(j+1,meses_estado)] = OL[member(j,
       meses_estado)] + OA[member(j,meses_estado)] - OE[member(j,meses_estado)] - OR[
       member(j,meses_estado)] ;
60
61 # Quirófanos asignados a cada servicio
62 # Oftalmología
63 quirofanos_asignados_oftalmologia{i in meses_control}:   80*CR[i] <= OCQ[i];
64
65 # Traumatología
66 quirofanos_asignados_traumatologia{i in meses_control}:  85*HR[i] + 120*KR[i] + 160*OR
       [i] <= TEQ[i];
67
68 # Cotas al número de procesos fuera del horario normal
```

```
69
70 cota_cataratas{i in meses_control}: CO[i] <= l[i];
71 cota_hallux{i in meses_control}:    HP[i] <= m[i];
72 cota_rodilla{i in meses_control}:   KP[i] <= n[i];
73
74 # Permanencia máxima menor de 9 meses
75
76 permanencia_cataratas_1{j in 1..12}: sum{i in meses_control: ord(i)<=j} (CR[i]+CO[i])
       >=  a[member(j,meses_control)];
77
78 permanencia_hallux_1{j in 1..12}: sum{i in meses_control: ord(i)<=j}(HR[i]+HP[i]) >=
       b[member(j,meses_control)];
79
80 permanencia_rodilla_1{j in 1..12}: sum{i in meses_control: ord(i)<=j}(HR[i]+HP[i]) >=
        c[member(j,meses_control)];
81
82 permanencia_osteoartrosis_1{j in 1..12}: sum{i in meses_control: ord(i)<=j}(HR[i]+HP[i
       ]) >=  d[member(j,meses_control)];
83
84 # No más de 6 meses en lista de espera
85
86 no_mas_6_meses_cataratas:       CL["Enero_1999"]  <= 395;
87 no_mas_6_meses_hallux:          HL["Enero_1999"]  <= 69;
88 no_mas_6_meses_rodilla:         KL["Enero_1999"]  <= 77;
89 no_mas_6_meses_osteoartrosis:   OL["Enero_1999"]  <= 57; #con 81 funciona
```

Listing B.2: Código AMPL

run_espera.run

```
1 # Cargamos el modelo
2 model modelo_espera_final.mod;
3
4 # Cargamos los datos
5 data datos_espera.dat;
6
7 # Elegimos a CPLEX como solver
8 option solver cplex;
9
10 # Condiciones iniciales
11 fix CL["Enero_1998"] := 480;
12 fix HL["Enero_1998"] := 199;
13 fix KL["Enero_1998"] := 132;
14 fix OL["Enero_1998"] := 128;
15
16 # Resolvemos el problema
17 solve;
18
19 # Mostramos los resultados deseados.
20 display CL,HL,KL,OL;
21 display CR,CO,HR,HP,KR,KP,OR;
22
23 display fobj;
```

Listing B.3: Código AMPL

Apéndice C

Código símplex multiobjetivo

El archivo principal y el cual llamará al resto de scripts, los cuales resuelven cada uno de los pasos del método símplex multiobjetivo, es el llamado *multiobjetivo.r*. Este archivo devolverá las bases eficientes y sus puntos asociados.

multiobjetivo.r

```
multiobjetivo <- function(a_inicial, b_inicial, C_inicial, signos, operacion_opt){
  library(gtools)
  options(digits=2)
  C <- C_inicial
  a <- a_inicial
  b <- b_inicial
  C_row <- dim(C)[1]
  C_col <- dim(C)[2]
  n_col <- dim(a)[2]
  n_row <- dim(a)[1]
  n_col_h <- n_col
  for(i in 1:C_row){
    if(operacion_opt[i] == "min"){
      C[i,] <- -C[i,]
    }
  }
  operacion_opt<- "max"
  for(i in 1:length(signos)){
    if(signos[i] == "<="){
      n_col_h <- n_col_h + 1
    }else if(signos[i] == "="){
      n_col_h <- n_col_h + 1
    }else{
      n_col_h <- n_col_h + 2
    }
  }
  a_h <- matrix(nrow=n_row, ncol=n_col_h)
  n_col2 <- n_col
  var_artificiales <- c()
  var_basicas <- numeric(n_row)
  for(i in 1:n_row){
    if(signos[i] == "<="){
      a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), 1, rep(0, n_col_h - n_col2 - 1)))
      n_col2 <- n_col2 + 1
      var_basicas[i] <- n_col2
    }else if(signos[i] == ">="){
      a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), -1, 1, rep(0, n_col_h - n_col2 - 2)))
      var_artificiales <- c(var_artificiales, n_col2 + 2)
      n_col2 <- n_col2 + 2
      var_basicas[i] <- n_col2
```

```
  }else{
    a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), 1, rep(0, n_col_h - n_col2 - 1)))
    var_artificiales <- c(var_artificiales, n_col2 + 1)
    n_col2 <- n_col2 + 1
    var_basicas[i] <- n_col2
  }
}
var_totales <- c(1:n_col_h)
C_h <- matrix(0, nrow=C_row, ncol=n_col_h)
C_h[1:C_row, 1:n_col] <- C
C_h[,var_artificiales] <- -1111
x_B <- b
C_B <- C_h[, var_basicas]
z <- rowSums(C_B * b)
conjunto_resultados <- list()
l = 0
C_hh <- C_h
b_h <- numeric(n_col_h)
b_h[var_basicas] <- b
if(sum(C_B != 0) != 0){
  for(j in 1:C_row){
    for(i in 1:length(var_artificiales)){
      C_hh[j,] <- C_hh[j,] - (a_h[which(var_artificiales[i] == var_basicas),] *
                C_h[j,][var_artificiales[i]] /
                a_h[which(var_artificiales[i] == var_basicas), var_artificiales[i]])
    }
  }
  C_j <- C_hh
  z_j <- matrix(0, nrow=C_row, ncol=n_col_h)
  z_j[,var_basicas] <- C_hh[, var_basicas] * b
  bar_C <- C_j - C_j
  bar_CC <- bar_C[,-var_basicas]
  a_hh <- matrix(nrow = n_row, ncol = n_col_h)
  bb <- numeric(n_row)
}else{
  C_j <- C_hh
  z_j <- matrix(0, nrow=C_row, ncol=n_col_h)
  z_j[, var_basicas] <- C_h[, var_basicas] * b
  bar_C <- C_j-z_j
  bar_CC <- bar_C[,-var_basicas]
  a_hh <- matrix(nrow = n_row, ncol = n_col_h)
  bb <- numeric(n_row)
}
conjunto_vertices_eficientes <- list()
conjunto_vertices_no_eficientes <- list()
variables_vertices_eficientes <- list()
rayos_eficientes <- list()
if(all(bar_C <= 0)){
  punto_ideal <- numeric(n_col)
  for(i in 1:n_col){
    if(sum(var_basicas == i) == 0){
      punto_ideal[i] <- 0
    }else{
      punto_ideal[i] <- x_B[which(var_basicas == i)]
    }
  }
  resultados <- list( c("Estamos ante un punto ideal" , punto_ideal))
  return(resultados)
}else if(any(rowSums(bar_C[,-var_basicas] < 0) == n_col) | (all(colSums(bar_C) <= 0)
  )){
  punto_optimo <- numeric(n_col)
  for(i in 1:n_col){
    if(sum(var_basicas == i) == 0){
      punto_optimo[i] <- 0
    }else{
```

```
104         punto_optimo[i] <- x_B[which(var_basicas == i)]
105       }
106     }
107     l <- l + 1
108     conjunto_vertices_eficientes[[l]] <- punto_optimo
109     variables_vertices_eficientes[[l]] <- sort(var_basicas)
110   }else{
111     C_escalarizacion <- colSums(C)
112     a_escalarizacion <- rbind(a_inicial, C)
113     vector_paso_3 <- numeric(length(1:C_col))
114     posiciones <- var_basicas[var_basicas %in% 1:C_col]
115     vector_paso_3[posiciones] <- x_B[var_basicas %in% 1:C_col]
116     Cv <- C %*% vector_paso_3
117     b_escalarizacion <- c(b_inicial, Cv)
118     signos_escalarización <- c(signos, rep(">=", C_row))
119     simplex3_ejecutar = simplex3(a_escalarizacion, b_escalarizacion, C_escalarizacion,
                          operacion_opt, signos_escalarización)
120     if(typeof(simplex3_ejecutar) == "character"){
121       return("El MOLP no tiene soluciones eficientes")
122     }
123     punto_optimo <- simplex3_ejecutar[[2]][[1]]
124     matriz_optima <- simplex3_ejecutar[[3]]
125     variables_entrada <- simplex3_ejecutar[[4]]
126     valor_variables_basicas<- simplex3_ejecutar[[5]]
127     variables_basicas <- simplex3_ejecutar[[6]]
128     numero_variables_escal <- simplex3_ejecutar[[7]]
129     eliminar_var_problem_escal <- (n_col_h+1) :numero_variables_escal
130     var_basicas=variables_basicas[!variables_basicas %in% eliminar_var_problem_escal]
131     variables_basicas_ordenadas = sort(variables_basicas)
132     conjunto_variables_eficientes <- var_totales[(1:n_col_h) %in%
133                                      variables_basicas_ordenadas]
134     varibales_basicas_pronlema_multi <- variables_basicas  %in%
135                                         conjunto_variables_eficientes
136     l <- l + 1
137     conjunto_vertices_eficientes[[l]] <- punto_optimo
138     variables_vertices_eficientes[[l]] <- conjunto_variables_eficientes
139     a_h <- round(matriz_optima[varibales_basicas_pronlema_multi, 1:n_col_h],5)
140     x_B <- round(valor_variables_basicas[varibales_basicas_pronlema_multi],4)
141     b <- x_B
142     variables_entrada <- unique(variables_entrada[variables_entrada %in% var_totales])
143     for (i in 1:C_row){
144       c = C_inicial[i,]
145       a = a_inicial
146       signos
147       b =b_inicial
148       operacion_opt<-"max"
149       una_iteracion_ejecutar = una_iteracion(a, b, c, operacion_opt, signos,
150                                variables_entrada)
151       filas_C = una_iteracion_ejecutar
152       bar_C[i,] <- round(filas_C,4)
153     }
154   }
155   var_entrada <- 0
156   while(typeof(var_entrada) != "character"){
157     bar_CC <- bar_C[,-var_basicas]
158     if(any(colSums(bar_CC == 0) == C_row)){
159       if(a_h[,-var_basicas][, which(colSums(bar_CC == 0) == C_row)] <= 0){
160         punto_optimo_todas_variables <- numeric(n_col_h)
161         punto_optimo_todas_variables[var_basicas] <- x_B
162         direccion_optima <- numeric(n_col_h)
163         pos_rayo_efi_en_var_no_basicas = which(colSums(bar_CC == 0) == C_row)==T
164         direccion_optima[var_basicas] <- -a_h[,-var_basicas][,pos_rayo_efi]
165         pos_rayo_efi_en_var_totales = var_totales[-var_basicas][pos_rayo_efi]
166         direccion_optima[pos_rayo_efi_en_var_totales] <- 1
167         resultados_rayos <- list( c("Estamos ante un rayo con valores que aumentan
```

```
168                 infinitamente", punto_optimo_todas_variables, "+" , direccion_optima))
169       }
170     }
171     posicion_var_posible_entrada <- which(colSums(bar_C[, -var_basicas] >= 0) == 1)
172     posicion_var_posible_entrada = var_totales[-var_basicas][
        posicion_var_posible_entrada]
173     var_posible_entrada <- bar_C[, posicion_var_posible_entrada]
174     if(length(posicion_var_posible_entrada) > 1){
175       ee = x_B/a_h[, posicion_var_posible_entrada]
176       ee[which((ee<0)==T)] = 100000
177       multiplicadores_comprobacion_quien_entra_base <- apply(ee,2, min)
178       multiplicacion <- t(t(var_posible_entrada) *
179                           multiplicadores_comprobacion_quien_entra_base)
180       if(length(posicion_var_posible_entrada) > 1){
181         numero_combinaciones <- dim(combinations(
182                                 length(posicion_var_posible_entrada), 2 ))[1]
183       }
184       o = 0
185       while(length(posicion_var_posible_entrada) > 1){
186         for(i in 1:(length(posicion_var_posible_entrada)-1)){
187           if(sum(multiplicacion[,i] > multiplicacion[,i+1]) == C_row){
188             posicion_var_posible_entrada <- posicion_var_posible_entrada[-(i+1)]
189           }else if(sum(multiplicacion[,i] < multiplicacion[,i+1]) == C_row){
190             posicion_var_posible_entrada <- posicion_var_posible_entrada[-i]
191           }else{
192             posicion_var_posible_entrada <- posicion_var_posible_entrada
193           }
194         }
195         o = o + 1
196         if(o == numero_combinaciones){
197           break
198         }
199       }
200     }
201     numero_posibles_variables_entrantes <- length(posicion_var_posible_entrada)
202     posicion_var_posible_entrada_inicial <- posicion_var_posible_entrada
203     for(i in 1:numero_posibles_variables_entrantes){
204       var_entrada <- elegir_var_entrada(posicion_var_posible_entrada_inicial[i],
205                       var_basicas, a_h, x_B,
206                       variables_vertices_eficientes, conjunto_vertices_eficientes)
207       if(typeof(var_entrada)=="character"&length(posicion_var_posible_entrada)==1){
208         devolver<-list( "puntos eficientes" = conjunto_vertices_eficientes,
209         "variables eficientes" = variables_vertices_eficientes)
210         return(devolver)
211       }else if(typeof(var_entrada) != "character"){
212         paso_7_paso3_ejecutar = paso_7_paso3(var_entrada, bar_C, a_h, a_hh, x_B,
213                     n_col,var_basicas, l, conjunto_vertices_eficientes,
214                     variables_vertices_eficientes, C, C_inicial, a_inicial, C_col,
215                     b_inicial, signos, C_row, operacion_opt, n_col_h,
216                     var_totales,n_row,bb)
217         if(typeof(paso_7_paso3_ejecutar) == "character"){
218           posicion_var_posible_entrada = posicion_var_posible_entrada[
219                           -which(posicion_var_posible_entrada == var_entrada)]
220           if(length(posicion_var_posible_entrada)==0){
221             devolver<-list(  "puntos eficientes" = conjunto_vertices_eficientes,
222                     "variables eficientes" = variables_vertices_eficientes)
223             return(devolver)
224           }
225         }else{
226           l <- paso_7_paso3_ejecutar[[5]]
227           conjunto_vertices_eficientes <- paso_7_paso3_ejecutar[[6]]
228           variables_vertices_eficientes <- paso_7_paso3_ejecutar[[7]]
229           posicion_var_posible_entrada = posicion_var_posible_entrada[
230                     -which(posicion_var_posible_entrada == var_entrada)]
231         }
```

```
      }else{
        posicion_var_posible_entrada = posicion_var_posible_entrada[
                -which(posicion_var_posible_entrada == var_entrada)]
      }
    }
    bar_C <- paso_7_paso3_ejecutar[[1]]
    a_h <- paso_7_paso3_ejecutar[[2]]
    x_B <- paso_7_paso3_ejecutar[[3]]
    var_basicas <- paso_7_paso3_ejecutar[[4]]
    l <- paso_7_paso3_ejecutar[[5]]
    conjunto_vertices_eficientes <- paso_7_paso3_ejecutar[[6]]
    variables_vertices_eficientes <- paso_7_paso3_ejecutar[[7]]
    a_hh <- paso_7_paso3_ejecutar[[8]]
    bb <- paso_7_paso3_ejecutar[[9]]
  }
}
```

Listing C.1: Código r

simplex_monoobjetivo.r

```
simplex3 <- function(a, b, c , operacion_opt, signos){
  n_col <- dim(a)[2]
  n_row <- dim(a)[1]
  n_col_h <- n_col
  for(i in 1:length(signos)){
    if(signos[i] == "<="){
      n_col_h <- n_col_h + 1
    }else if(signos[i] == "="){
      n_col_h <- n_col_h + 1
    }else{
      n_col_h <- n_col_h + 2
    }
  }
  a_h <- matrix(nrow=n_row, ncol=n_col_h)
  n_col2 <- n_col
  var_artificiales <- c()
  var_basicas <- numeric(n_row)
  for(i in 1:n_row){
    if(signos[i] == "<="){
      a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), 1, rep(0, n_col_h - n_col2 - 1)))
      n_col2 <- n_col2 + 1
      var_basicas[i] <- n_col2
    }else if(signos[i] == ">="){
      a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), -1, 1, rep(0, n_col_h - n_col2 - 2)))
      var_artificiales <- c(var_artificiales, n_col2 + 2)
      n_col2 <- n_col2 + 2
      var_basicas[i] <- n_col2
    }else{
      a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), 1, rep(0, n_col_h - n_col2 - 1)))
      var_artificiales <- c(var_artificiales, n_col2 + 1)
      n_col2 <- n_col2 + 1
      var_basicas[i] <- n_col2
    }
  }
  var_totales <- c(1:n_col_h)
  c_h <- numeric(n_col_h)
  c_h[1:n_col] <- c(c)
  c_h[var_artificiales] <- -1111
  x_B <- b
  c_B <- c_h[var_basicas]
  z <- sum(c_B * b)
  conjunto_resultados <- list()
  l = 0
  c_hh <- c_h
```

```
  if(sum(c_B != 0) != 0){
    for(i in 1:length(var_artificiales)){
      c_hh <- c_hh - (a_h[which(var_artificiales[i] == var_basicas),] *
          c_h[var_artificiales[i]] / a_h[which(var_artificiales[i] == var_basicas),
          var_artificiales[i]])
    }
    c_j <- c_hh
    z_j <- integer(n_col_h)
    z_j[var_basicas] <- c_hh[var_basicas] * b
    bar_c <- c_j - z_j
    bar_cc <- bar_c[-var_basicas]
    a_hh <- matrix(nrow = n_row, ncol = n_col_h)
    bb <- numeric(n_row)
  }else{
    c_j <- c_hh
    z_j <- integer(n_col_h)
    z_j[var_basicas] <- c_h[var_basicas] * b
    bar_c <- c_j-z_j
    bar_cc <- bar_c[-var_basicas]
    a_hh <- matrix(nrow = n_row, ncol = n_col_h)
    bb <- numeric(n_row)
  }
  variables_entrada = c()
  while(max(bar_cc) > 0){
    var_entrada <- min(which(bar_c == max(bar_c)))
    variables_entrada <- c(variables_entrada, var_entrada)
    y_k <- a_h[,var_entrada]
    if(sum(y_k > 0) > 0){
      var_salida <- min(var_basicas[which(x_B / y_k == min(x_B[y_k > 0] /
                    y_k[y_k > 0]) & y_k > 0)])
      var_salida_posicion <- min(which(var_salida == var_basicas))
      pivote <- a_h[var_salida_posicion, var_entrada]
      a_hh[var_salida_posicion, ] <- a_h[var_salida_posicion, ] / pivote
      if(class(a_h[-var_salida_posicion, ]) == "numeric"){
        a_hh[-var_salida_posicion, ] <- a_h[-var_salida_posicion, ] -
        (a_h[var_salida_posicion, ] * a_h[-var_salida_posicion, var_entrada] / pivote)
      }else{
        for(i in 1:(n_row - 1)){
          a_hh[-var_salida_posicion, ][i, ] <- a_h[-var_salida_posicion, ][i, ] -
                    (a_h[var_salida_posicion, ]*
                    a_h[-var_salida_posicion, var_entrada][i] / pivote)
        }
      }
      bb[var_salida_posicion] <- b[var_salida_posicion] / pivote
      bb[-var_salida_posicion] <- b[-var_salida_posicion] - (b[var_salida_posicion] *
      a_h[-var_salida_posicion, var_entrada] / pivote)
      b <- bb
      x_B <- b
      bar_c <- bar_c - (a_h[var_salida_posicion, ] * bar_c[var_entrada] / pivote)
      a_h[var_salida_posicion,] = a_hh[var_salida_posicion, ]
      a_h[-var_salida_posicion,] = a_hh[-var_salida_posicion, ]
      var_basicas[which(var_basicas == var_salida)] <- var_entrada
      bar_cc <- bar_c[-var_basicas]
      c_B <- c_h[var_basicas]
      z <- sum(c_B * b)
    }else{
        punto_optimo_todas_variables <- numeric(n_col_h)
        punto_optimo_todas_variables[var_basicas] <- x_B
        direccion_optima <- numeric(n_col_h)
        direccion_optima[var_basicas] <- -y_k
        direccion_optima[var_entrada] <- 1
        resultados <- list( c("Estamos ante un rayo con valores que aumentan
              infinitamente" , punto_optimo_todas_variables, "+" , direccion_optima))
        return(resultados)
      }
```

```
    }
  punto_optimo <- numeric(n_col)
  for(i in 1:n_col){
    if(sum(var_basicas == i) == 0){
      punto_optimo[i] <- 0
    }else{
      punto_optimo[i] <- x_B[which(var_basicas == i)]
    }
  }
  l <- l + 1
  conjunto_resultados[[l]] <- punto_optimo
  if(length(which(var_basicas %in% var_artificiales)) != 0){
    return("Problema sin soluciones factibles")
  }else{
    if(operacion_opt == "max"){
      resultados <- list("El valor optimo es" = z, "El punto optimo" =
              conjunto_resultados, "La matriz óptimo" = a_hh,
              "Las variables de entrada utilizadas son" = variables_entrada,
              "El valor de las variables básicas son" = x_B,
              "Las variables básicas son" = var_basicas, "Numero variables matriz
              aumentada simplex3" = n_col_h)
      return(resultados)
    }else{
      resultados <- list("El valor optimo es" = -z,  "El punto optimo" =
              conjunto_resultados, "La matriz óptimo" = a_hh,
                "Las variables de entrada utilizadas son" = variables_entrada,
                "El valor de las variables básicas son" = x_B,
                "Las variables básicas son" = var_basicas, "Numero variables matriz
                aumentada simplex3" = n_col_h)
      return(resultados)
    }
  }
}
```

Listing C.2: Código r

cambio_variable_multiobjetivo.r

```

cambio_variable_multiobjetivo <- function(a_h, b, c, var_entrada, var_basicas, x_B,
                                  a_hh,n_row,bb){
  bar_c <- c
  y_k <- a_h[,var_entrada]
  var_salida <- min(var_basicas[which(x_B / y_k == min(x_B[y_k > 0] / y_k[y_k > 0]) &
    y_k > 0)])
  var_salida_posicion <- min(which(var_salida == var_basicas))
  pivote <- a_h[var_salida_posicion, var_entrada]
  a_hh[var_salida_posicion, ] <- a_h[var_salida_posicion, ] / pivote
  if(class(a_h[-var_salida_posicion, ]) == "numeric"){
    a_hh[-var_salida_posicion, ] <- a_h[-var_salida_posicion, ] -
                    (a_h[var_salida_posicion, ] *
                    a_h[-var_salida_posicion, var_entrada] / pivote)
  }else{
    for(i in 1:(n_row - 1)){
      a_hh[-var_salida_posicion, ][i, ] <- a_h[-var_salida_posicion, ][i, ] -
                    (a_h[var_salida_posicion, ]*
                    a_h[-var_salida_posicion, var_entrada][i] / pivote)
    }
  }
  bb[var_salida_posicion] <- b[var_salida_posicion] / pivote
  bb[-var_salida_posicion] <- b[-var_salida_posicion] - (b[var_salida_posicion] *
  a_h[-var_salida_posicion, var_entrada] / pivote)
  b <- bb
  x_B <- b
  bar_c <- bar_c - (a_h[var_salida_posicion, ] * bar_c[var_entrada] / pivote)
```

```
  a_h[var_salida_posicion,] = a_hh[var_salida_posicion, ]
  a_h[-var_salida_posicion,] = a_hh[-var_salida_posicion, ]
  var_basicas[which(var_basicas == var_salida)] <- var_entrada
  resultados <- list(bar_c, a_h, x_B, var_basicas, a_hh)
  return(resultados)
}
```

Listing C.3: Código r

comprobacion_eficiencia.r

```

paso_7_paso3 <- function(var_entrada, bar_C, a_h, a_hh, x_B, n_col, var_basicas, l,
    conjunto_vertices_eficientes, variables_vertices_eficientes, C, C_inicial,
    a_inicial, C_col, b_inicial, signos, C_row, operacion_opt, n_col_h, var_totales,
    n_row, bb){
  for (i in 1:C_row){
    c = bar_C[i,]
    a_h
    b = x_B
    cambio_variable_multiobjetivo_ejecutar = cambio_variable_multiobjetivo(a_h, b, c,
              var_entrada, var_basicas, x_B, a_hh,n_row,bb)
    filas_C <- cambio_variable_multiobjetivo_ejecutar[[1]]
    bar_C[i,] <- filas_C
  }
  a_h <- cambio_variable_multiobjetivo_ejecutar[[2]]
  x_B <- cambio_variable_multiobjetivo_ejecutar[[3]]
  var_basicas <- cambio_variable_multiobjetivo_ejecutar[[4]]
  a_hh <- cambio_variable_multiobjetivo_ejecutar[[5]]
  if(any(rowSums(bar_C[,-var_basicas] < 0) == n_col) | (all(colSums(bar_C) <= 0))){
    punto_optimo <- numeric(n_col)
    for(i in 1:n_col){
      if(sum(var_basicas == i) == 0){
        punto_optimo[i] <- 0
      }else{
        punto_optimo[i] <- x_B[which(var_basicas == i)]
      }
    }
    l <- l + 1
    conjunto_vertices_eficientes[[l]] <- punto_optimo
    variables_vertices_eficientes[[l]] <- sort(var_basicas)
  }else{
    C_escalarizacion <- colSums(C)
    a_escalarizacion <- rbind(a_inicial, C)
    vector_paso_3 <- numeric(length(1:C_col))
    posiciones <- var_basicas[var_basicas %in% 1:C_col]
    vector_paso_3[posiciones] <- x_B[var_basicas %in% 1:C_col]
    Cv <- C %*% vector_paso_3
    b_escalarizacion <- c(b_inicial, Cv)
    signos_escalarización <- c(signos, rep(">=", C_row))
    simplex3_ejecutar = simplex3(a_escalarizacion, b_escalarizacion, C_escalarizacion,
     operacion_opt, signos_escalarización)
    if(simplex3_ejecutar == "Problema sin soluciones factibles"){
      return("Esta variable no entra en la base")
    }
    punto_optimo <- simplex3_ejecutar[[2]][[1]]
    matriz_optima <- simplex3_ejecutar[[3]]
    variables_entrada <- simplex3_ejecutar[[4]]
    valor_variables_basicas<- simplex3_ejecutar[[5]]
    variables_basicas <- simplex3_ejecutar[[6]]
    numero_variables_escal <- simplex3_ejecutar[[7]]
    eliminar_var_problem_escal <- (n_col_h+1) :numero_variables_escal
    var_basicas=variables_basicas[!variables_basicas %in% eliminar_var_problem_escal]
    variables_basicas_ordenadas = sort(variables_basicas)
```

```
    conjunto_variables_eficientes <- var_totales[(1:n_col_h) %in%
    variables_basicas_ordenadas]
    varibales_basicas_pronlema_multi <- variables_basicas  %in%
    conjunto_variables_eficientes
    l <- l + 1
    conjunto_vertices_eficientes[[l]] <- punto_optimo
    variables_vertices_eficientes[[l]] <- conjunto_variables_eficientes
    a_h <- round(matriz_optima[varibales_basicas_pronlema_multi, 1:n_col_h],5)
    x_B <- round(valor_variables_basicas[varibales_basicas_pronlema_multi],4)
    b <- x_B
    variables_entrada <- unique(variables_entrada[variables_entrada %in% var_totales])
    for (i in 1:C_row){
      c = C_inicial[i,]
      a = a_inicial
      signos
      b =b_inicial
      operacion_opt<-"max"
      una_iteracion_ejecutar = una_iteracion(a, b, c, operacion_opt, signos,
                               variables_entrada)
      filas_C = una_iteracion_ejecutar
      bar_C[i,] <- round(filas_C,4)
    }
  }
  resultados_paso7 <- list(bar_C, a_h, x_B, var_basicas, l,
                conjunto_vertices_eficientes, variables_vertices_eficientes, a_hh,bb)
  return(resultados_paso7)
}
```

Listing C.4: Código r

elegir_var_entrada.r

```

elegir_var_entrada<- function(posicion_var_posible_entrada, var_basicas, a_h, x_B,
                    variables_vertices_eficientes, conjunto_vertices_eficientes){
  for(i in 1:length(posicion_var_posible_entrada)){
    variable_repetida = 0
    var_basicas_comprobacion <- var_basicas
    var_entrada1 <- posicion_var_posible_entrada[i]
    y_k <- a_h[,var_entrada1]
    if(sum(y_k > 0) > 0){
      var_salida <- min(var_basicas[which(x_B / y_k == min(x_B[y_k > 0] /
                    y_k[y_k > 0]) & y_k > 0)])
    }
    var_basicas_comprobacion[which(var_basicas == var_salida)] <- var_entrada1
    for(i in 1:length(variables_vertices_eficientes)){
      if(sum(variables_vertices_eficientes[[i]] %in% sort(var_basicas_comprobacion))
      == length(var_basicas_comprobacion)){
        variable_repetida = 1
        break
      }
    }
    if(variable_repetida == 0){
      break
    }
  }
  if(variable_repetida == 1){
    return("Esta base ya se ha comprobado")
  }
  var_entrada <- var_entrada1
  return(var_entrada)
}
```

Listing C.5: Código r

una_iteracion_simplex.r

```
1
2  una_iteracion <- function(a, b, c, operacion_opt, signos, variables_entrada){
3    n_col <- dim(a)[2]
4    n_row <- dim(a)[1]
5    n_col_h <- n_col
6    for(i in 1:length(signos)){
7      if(signos[i] == "<="){
8        n_col_h <- n_col_h + 1
9      }else if(signos[i] == "="){
10       n_col_h <- n_col_h + 1
11     }else{
12       n_col_h <- n_col_h + 2
13     }
14   }
15   a_h <- matrix(nrow=n_row, ncol=n_col_h)
16   n_col2 <- n_col
17   var_artificiales <- c()
18   var_basicas <- numeric(n_row)
19   for(i in 1:n_row){
20     if(signos[i] == "<="){
21       a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), 1, rep(0, n_col_h - n_col2 - 1)))
22       n_col2 <- n_col2 + 1
23       var_basicas[i] <- n_col2
24     }else if(signos[i] == ">="){
25       a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), -1, 1, rep(0, n_col_h - n_col2 - 2)))
26       var_artificiales <- c(var_artificiales, n_col2 + 2)
27       n_col2 <- n_col2 + 2
28       var_basicas[i] <- n_col2
29     }else{
30       a_h[i,] <- c(a[i,], c(rep(0,n_col2-n_col), 1, rep(0, n_col_h - n_col2 - 1)))
31       var_artificiales <- c(var_artificiales, n_col2 + 1)
32       n_col2 <- n_col2 + 1
33       var_basicas[i] <- n_col2
34     }
35   }
36   var_totales <- c(1:n_col_h)
37   c_h <- numeric(n_col_h)
38   c_h[1:n_col] <- c(c)
39   c_h[var_artificiales] <- -1111
40   x_B <- b
41   c_B <- c_h[var_basicas]
42   z <- sum(c_B * b)
43   conjunto_resultados <- list()
44   l = 0
45   c_hh <- c_h
46   if(sum(c_B != 0) != 0){
47     for(i in 1:length(var_artificiales)){
48       c_hh <- c_hh - (a_h[which(var_artificiales[i] == var_basicas),] *
49             c_h[var_artificiales[i]] / a_h[which(var_artificiales[i] == var_basicas),
50             var_artificiales[i]])
51     }
52     c_j <- c_hh
53     z_j <- integer(n_col_h)
54     z_j[var_basicas] <- c_hh[var_basicas] * b
55     bar_c <- c_j - z_j
56     bar_cc <- bar_c[-var_basicas]
57     a_hh <- matrix(nrow = n_row, ncol = n_col_h)
58     bb <- numeric(n_row)
59   }else{
60     c_j <- c_hh
61     z_j <- integer(n_col_h)
62     z_j[var_basicas] <- c_h[var_basicas] * b
63     bar_c <- c_j-z_j
```

```
    bar_cc <- bar_c[-var_basicas]
    a_hh <- matrix(nrow = n_row, ncol = n_col_h)
    bb <- numeric(n_row)
  }
  for(i in variables_entrada){
    var_entrada <- i
    y_k <- a_h[,var_entrada]
    if(sum(y_k > 0) > 0){
      var_salida <- min(var_basicas[which(x_B / y_k == min(x_B[y_k > 0] /
                    y_k[y_k > 0]) & y_k > 0)])
      var_salida_posicion <- min(which(var_salida == var_basicas))
      pivote <- a_h[var_salida_posicion, var_entrada]
      a_hh[var_salida_posicion, ] <- a_h[var_salida_posicion, ] / pivote
      if(class(a_h[-var_salida_posicion, ]) == "numeric"){
        a_hh[-var_salida_posicion, ] <- a_h[-var_salida_posicion, ] -
        (a_h[var_salida_posicion, ] * a_h[-var_salida_posicion, var_entrada] / pivote)
      }else{
        for(i in 1:(n_row - 1)){
          a_hh[-var_salida_posicion, ][i, ] <- a_h[-var_salida_posicion, ][i, ] -
                (a_h[var_salida_posicion, ]*a_h[-var_salida_posicion,
                var_entrada][i] / pivote)
        }
      }
      bb[var_salida_posicion] <- b[var_salida_posicion] / pivote
      bb[-var_salida_posicion] <- b[-var_salida_posicion] - (b[var_salida_posicion] *
      a_h[-var_salida_posicion, var_entrada] / pivote)
      b <- bb
      x_B <- b
      bar_c <- bar_c - (a_h[var_salida_posicion, ] * bar_c[var_entrada] / pivote)
      a_h[var_salida_posicion,] = a_hh[var_salida_posicion, ]
      a_h[-var_salida_posicion,] = a_hh[-var_salida_posicion, ]
      var_basicas[which(var_basicas == var_salida)] <- var_entrada
      var_basicas
      bar_cc <- bar_c[-var_basicas]
      c_B <- c_h[var_basicas]
      z <- sum(c_B * b)
    }else{
      punto_optimo_todas_variables <- numeric(n_col_h)
      punto_optimo_todas_variables[var_basicas] <- x_B
      direccion_optima <- numeric(n_col_h)
      direccion_optima[var_basicas] <- -y_k
      direccion_optima[var_entrada] <- 1
      resultados <- list( c("Estamos ante un rayo con valores que aumentan
      infinitamente" , punto_optimo_todas_variables, "+" , direccion_optima))
      return(resultados)
    }
  }
  return(bar_c)
}
```

Listing C.6: Código r

Bibliografía

[1] Bazaraa MS, Jarvis JJ, Sherali HD (1990) Linear Programming and Network Flows. John Wiley and Sons, New York.

[2] Dantzig GB (1990) The Diet Problem. Interfaces 20:4, July/Aug, 430?47.

[3] Cerdá-Tena E, de Pablos-Escobar L, Rodríguez-Uría MV (2001) La gestión de las listas de espera quirúrgica en España. Instituto de estudios fiscales.

[4] González-Díaz J (2017) Programación Lineal y Entera. Apuntes Máster en Técnicas Estadísticas (UDC, USC y UVIGO).

[5] Klee V, Minty GJ (1972) How Good is the Simplex Algorithm? Academic Press, New York, 159-175.

[6] The Luc D (2016) Multiobjective Linear Programming. Springer International Publishing, Switzerland.