UNIVERSITAT DE BARCELONA
Facultat de Matemàtiques i Informàtica

GRAU DE MATEMÀTIQUES

Treball final de grau

Introducción a la programación lineal y entera: métodos del Símplex y de B&B

Autor: Arturo Elizalde Masiá

Director: Dra. Susana Romano Rodríguez
Realitzat a: Departament de Matemàtiques i Informàtica (Matemàtiques i Informàtica)

Barcelona, 21 de junio de 2020

Abstract

Since Dantzig published *the simplex method* in 1949 for the solution of linear programming problems (area of optimization) a great interest was sparked that allowed a vast theoretical growth, opening the door to the development of new methods and algorithms that are widely used today.

The main goal of this work is to address, at an introductory level, the simplex method and the *Branch and Bound method* (or B&B). If $x = (x_B, x_N)$ is a basic feasible solution of a linear programming problem $\min\{f(x) \mid Ax = b, x \geq 0\}$ with no degenerate solutions, throughout the iterative algorithm of the simplex it is determined through a finite quantity of steps whether the problem is already within an optimal solution, whether a finite optimal solution does not exist, or whether it determines the optimal solution.

The Branch and Bound method is used for solving linear programming problems in which the solution (or part of it) must be an integer. The algorithm follows a *binary tree* structure and under ideal conditions finishes in a finite quantity of steps.

Resumen

Desde que Dantzig publicó el *método del símplex* en 1949 para la resolución de problemas de programación lineal (área de la optimización) despertó un fuerte interés, y permitió que se produjera un amplio crecimiento teórico dando pie al desarrollo de nuevos métodos y algoritmos muy utilizados hoy en día.

El objetivo principal del trabajo es abordar de manera introductoria el método del símplex y el *método de Branch and Bound* (o B&B). Si $x = (x_B, x_N)$ es una solución básica factible de un problema de programación lineal $\text{mín}\{f(x) \mid Ax = b, x \geq 0\}$ sin soluciones degeneradas, mediante el algoritmo iterativo del símplex se determina en una cantidad finita de pasos si el problema se encuentra ya en una solución óptima, si no existe solución óptima finita o bien determina la solución óptima.

El método de Branch and Bound, se utiliza para resolver problemas de programación lineal en donde la solución (o parte de esta), debe ser entera. El algoritmo sigue una estructura de *árbol binario* y bajo condiciones ideales termina en una cantidad finita de pasos.

2010 Mathematics Subject Classification. 90C05, 90C10, 90C11

Agradecimientos

A la Dra. Susana Romano por su ayuda, la paciencia y su tiempo.

A Joan y mi hermano Borja que me han ayudado siempre que lo he necesitado.

Índice

1. Introducción

Si $f : \mathcal{A} \longrightarrow \mathbb{R}$ es una función con $\mathcal{A} \subset \mathbb{R}^n$, mediante la optimización en general se pretende encontrar $y \in \mathcal{A}$ de forma que $f(x) \leq f(y)$, o bien $f(x) \geq f(y)$ para todo $y \in \mathcal{A}$. La programación lineal es un campo de la optimización en el cual la función denominada *función objetivo*, es lineal y respeta un conjunto de restricciones también lineales, de igualdad o desigualdad.

La historia de la programación lineal data sus inicios sobre una publicación en 1823 del matemático y físico *Jean-Baptiste Joseph Fourier*. No sería por eso hasta 1827 donde se considera que está formulado el primer problema de programación lineal, planteado como un sistema de inecuaciones a resolver junto al *método de eliminación de Fourier*; que de forma simplificada, es una extensión del método de *eliminación de Gauss* (usado para sistemas de ecuaciones lineales).
La siguiente publicación conocida data en 1911 escrito por el ingeniero y matemático *Charles-Jean de La Vallée Poussin*; Poussin diseño un método para la aproximación de *Chebyshev* que consistía en minimizar $\|Ax - b\|_\infty$ dada una matriz A y un vector b.
El matemático ruso *Leonid Vitaliyevich Kantorovich* publicó *Métodos matemáticos para la organización y la producción* en el año 1939. Su obra fue archivada por razones ideológicas en la URSS y no vio la luz hasta 1959.
Es por esto que es el matemático y físico *G. B. Dantzig* quien es considerado el padre de la programación lineal tal y como la conocemos hoy en día, debido a sus trabajos en 1947, cuando trabajaba como asesor de las Fuerzas Áereas de los Estados Unidos en el desarrollo de una herramienta para la automatización de la planificación de actividades como despliegue de tropas, entrenamiento y demás logística de las tropas aliadas entre otros.

El nombre de programación lineal viene precisamente de su trabajo como asesor donde cada una de estas planificaciones se denominaba "programación"("program", en inglés). Por tanto, el nombre se refiere al diseño óptimo de programas, como recoge el título de la primera publicación de Dantzig sobre el tema: *Programming in a linear structure*. El nombre final de programación lineal se le atribuye al economista y matemático americano, aunque de origen holandés, *T. C. Koopmans*.

En 1949 Dantzig publicó el *Método del Símplex* para resolver problemas de programación lineal, y que desde entonces ha sido el algoritmo de referencia para resolver este tipo de problemas, independientemente de su tamaño. Según el propio Dantzig, la idea que le llevo a creer que el Método del Símplex sería una técnica de solución muy eficiente, fue debida a que la geometría usada para su tesis fue asociada con las columnas de una matriz en lugar de sus filas. Desde su publicación la programación lineal creció en múltiples direcciones, desde desarrollos teóricos hasta aspectos más puramente computacionales.

"*La observación, en particular, de que varios sistemas económicos, industriales, financieros y militares pueden ser modelados (o aproximados razonablemente) por sistemas matemáticos de desigualdades y ecuaciones lineales ha dado lugar al desarrollo del campo de programación lineal.*"(Dantzig, 1997).

Los inicios de la programación lineal entera datan de 1958 con el trabajo de *Ralph E. Gomory* quien introdujo el método de *los planos de corte*. El algoritmo *Branch and Bound* (o B&B) fue propuesto por primera vez por *A. H. Land y A. G. Doig* en 1960 para la programación entera.

Aunque la programación entera es NP-hard en general, algoritmos como el de Branch and Bound han resultado ser de gran utilidad en la práctica. La investigación en esta área es actualmente muy activa.

Estructura de la Memoria

Este trabajo recoge los principales resultados teóricos que ayudan a comprender y desarrollar el método del símplex desde un punto de vista tanto geométrico como algebraico.

En el capítulo 2 establecemos las bases de la programación lineal. Se define matemáticamente lo que es un programa lineal así como sus representaciones equivalentes; se define el conjunto de soluciones posibles que admite este tipo problema y que a su vez son soluciones candidatas para optimizar el problema. Se darán condiciones necesarias y suficientes para poder encontrar la solución óptima. Además se verán algunas relaciones importantes entre las propiedades geométricas del conjunto de soluciones y el análisis convexo que nos permitirá comprender la lógica usada en el método del símplex. Completaremos la formulación del método del símplex con el uso de la tabla del método, que ayudará a organizar y afrontar mejor un problema dado. Para finalizar el capítulo trataremos la dualidad, donde podremos ver que todo programa lineal tiene asociado con otro programa lineal con el que esta ampliamente relacionado y ofrecer así una mayor perspectiva. Se tratará también el algoritmo primal-dual, un método alternativo para resolver problemas de programación lineal.

El capítulo 3 lo dedicaremos a la programación entera. Se analizará en detalle el método de Branch and Bound. Para finalizar, en el capítulo 4 ejemplificaremos la formulación de alguno de los problemas clásicos de la programación lineal y entera, y se establecen algunas relaciones lógicas que ayudaran a comprender la modelización de problemas de programación entera.

2. Programación Lineal

Las principales fuentes consultadas para este capítulo han sido, [1] y [2]. Se citaran fuentes más concretas en alguno de los apartados.

Un problema de programación[2] lineal es un problema de optimización en el cual tanto la función *(función objetivo)*[3] como las restricciones son lineales. Cualquier programa lineal se puede convertir a la siguiente *forma estándar:*

$$\begin{array}{lll} \text{minimizar} & c_1x_1 + c_2x_2 + \cdots + c_nx_n & \\ \text{sujeto a} & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n & = b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & = b_2 \\ & \vdots & \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & = b_m \\ \text{y} & x_i \geq 0, \;\; i \in \{1, \cdots, n\}. & \end{array}$$

donde las b_i, c_i y a_{ij} son constantes reales fijas, y las x_i son valores reales a determinar.

Podemos expresar el problema de forma más compacta como[4]

$$\begin{array}{ll} \text{minimizar} & c^T x \\ \text{sujeto a} & Ax = b \\ & x \geq 0 \end{array} \tag{2.1}$$

donde x, $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ y $A \in M_{m\times n}(\mathbb{R})$, de rango m, con $m \leq n$.

2.1. Representaciones equivalentes

En la siguiente tabla representamos dos formulaciones que son especialmente útiles.

	Problema de minimización	Problema de maximización
Forma Estándar	$\begin{array}{rl} \text{mín} & c^T x \\ \text{sujeto a} & Ax = b \\ & x \geq 0 \end{array}$	$\begin{array}{rl} \text{máx} & c^T x \\ \text{sujeto a} & Ax = b \\ & x \geq 0 \end{array}$
Forma Canónica	$\begin{array}{rl} \text{mín} & c^T x \\ \text{sujeto a} & Ax \geq b \\ & x \geq 0 \end{array}$	$\begin{array}{rl} \text{máx} & c^T x \\ \text{sujeto a} & Ax \leq b \\ & x \geq 0 \end{array}$

[2] El uso de la palabra programación en este contexto se refiere a "planifiación".

[3] $f : \mathbb{R}^n \longrightarrow \mathbb{R}$. Esta es la función que queremos minimizar o maximizar. Se suele expresar de la forma $c_1x_1 + c_2x_2 + \cdots + c_nx_n$ donde los coeficientes $c_1, c_2, \ldots, c_n$ son conocidos.

[4] Si un vector es un vector fila $(1 \times n)$ o columna $(n \times 1)$ debería quedar claro por contexto, pero para evitar confusión, si x es un vector, se denotará:

- x si representa un vector columna.
- x^T si representa un vector columna traspuesto, es decir un vector fila.

Forma Canónica. En caso de que el problema sea de minimización, todas las restricciones son de la forma " $\geq$ " y, si es de maximización, son de la forma "$\leq$". En ambos casos las variables han de ser no negativas.

Forma Estándar. todas las restricciones son de igualdad y todas las variables son no negativas. En general nos centraremos en esta última.

Vamos a ver distintas transformaciones que se pueden hacer en los elementos que componen un problema de programación lineal, y que nos permitirán pasar el problema a la forma estándar si fuera necesario.

Función objetivo. Todo problema de minimización puede transformarse en un problema de maximización, y viceversa, ya que se verifica la siguiente relación:

$$\text{máx} \sum_{j=1}^{n} c_j x_j = \text{-mín} \sum_{j=1}^{n} -c_j x_j.$$

Restricciones. Existen varias manipulaciones posibles:

- Para pasar de una restricción " $\leq$ "(" $\geq$ ") a una restricción " $\geq$ "(" $\leq$ "), es suficiente multiplicar todos los coeficientes por -1.
- Si no queremos restricciones de igualdad, podemos reemplazar cada restricción de la forma $\sum_{j=1}^{n} a_{ij}x_j = b_i$ por dos restricciones de desigualdad $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$ y $\sum_{j=1}^{n} a_{ij}x_j \leq b_i$.
- Si solo queremos restricciones de igualdad, podemos remplazar cada restriccion de la forma $\sum_{j=1}^{n} a_{ij}x_j \leq b_i$ por $\sum_{j=1}^{n} a_{ij}x_j + x_{i+n}^{s} = b_i$. A estas variables se las conoce como *variables de holgura* y se suelen denotar con la letra s por *slack*. Si la restricción es de la forma $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$, bastaría con restar la variable de holgura en vez de sumarla. Estas variables se denominan *variables excedentes*.

No negatividad. Si tenemos una variable x_j que puede tomar valores positivos y negativos y queremos solo valores ≥ 0, podemos definir $x_j := u_j - v_j$ imponiendo $u_j \geq 0$ y $v_j \geq 0$.

Observación 2.1. En caso de pasar las restricciones de un programa lineal de la forma $Ax \leq b$ ($Ax \geq b$) mediante variables de holgura (excedentes) a restricciones de igualdad, la matriz del programa lineal pasaría a ser de dimensión $m \times (n+m)$ de la forma $[A, I]x = b$, siendo I la matriz identidad de dimensión m.

2.2. Soluciones básicas

Dado el sistema de ecuaciones

$$Ax = b,$$

con $x \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ y $A \in M_{m\times n}(\mathbb{R})$ de rango m, con $m \leq n$, reordenamos la matriz de manera que las m primeras columnas sean linealmente independientes.

Ahora, sea $B \in M_{m\times m}(\mathbb{R})$ la matriz determinada por estas m primeras columnas. Entonces, B es una matriz invertible y $x_B \in \mathbb{R}^m$ formado por las variables asociadas a B y x_N las demás, podemos escribir el sistema como

$$Bx_B + Nx_N = b.$$

A menos que se indique explícitamente lo contrario, en este apartado se mantienen las dimensiones de las matrices y vectores introducidos.

Definición 2.2. *Dada cualquier submatriz B invertible formada por columnas de A, $x = (x_B, x_N)$ es una solución básica de un programa lineal en la forma estándar, si x_B es solución de $Bx_B = b$ y $x_N = 0$. Las variables de x_B se denominan variables básicas.*

Definición 2.3. *Si una, o más variables básicas son nulas, entonces la solución se dice solución básica degenerada.*

Consideremos ahora

$$\begin{aligned} Ax &= b \\ x &\geq 0, \end{aligned} \tag{2.2}$$

que representa las restricciones de un programa lineal en forma estándar.

Definición 2.4. *Un vector x que satisfaga (2.2), se dice que es factible. Una solución factible para (2.2) que también sea básica, se denomina solución básica factible.*

Observación 2.5. Al conjunto de soluciones factibles, se le conoce como *región factible* y se suele denotar por F. Es decir, $F = \{x \mid Ax = b, x \geq 0\}$.

Definición 2.6. *Una solución básica factible óptima és una solución básica factible que optimiza el problema.*

Observación 2.7. La solución básica factible óptima en caso de existir, puede ser única o existir más de una.

Mediante el siguiente teorema se establece la importancia primordial de las soluciones básicas factibles para resolver problemas de programación lineal. La demostración representa el inicio del desarrollo del método del símplex.

Teorema 2.8. *Dado un programa lineal en la forma estándar (2.1), donde A es una matriz de $m \times n$ de rango m,*

1. *si existe una solución factible, existe una solución básica factible;*
2. *si existe una solución factible óptima, existe una solución básica factible óptima.*

Dem.: 1.) Sean $a_1, a_2, \ldots, a_n$ las columnas de A. Supongamos que $x^T = (x_1, x_2, \ldots, x_n)$ es una solución factible. Entonces, en función de las columnas de A, esta solución satisface:

$$x_1a_1 + x_2a_2 + \cdots + x_na_n = b.$$

Supongamos que exactamente p de las variables x_i son mayores que cero, y por conveniencia, que sean las primeras p variables. Entonces,

$$x_1a_1 + x_2a_2 + \cdots + x_pa_p = b. \tag{2.3}$$

Separamos en dos casos, dependiendo si el conjunto $a_1, a_2, \ldots, a_p$ es linealmente independiente o no:

Caso 1: Si tenemos $a_1, a_2, \ldots, a_p$ linealmente independientes, entonces es evidente que $p \leq m$.

- Si $p = m$, la solución es básica factible y por tanto hemos acabado.
- Si $p < m$, como $rang(A) = m$, podemos completar con $m - p$ vectores de los $n - p$ restantes por lo que el conjunto resultante de m vectores es linealmente independientes. La asignación del valor cero a las $m - p$ variables correspondientes da una solución básica factible (degenerada).

Caso 2: Si $a_1, a_2, \ldots, a_p$ linealmente dependientes, entonces existen $\lambda_1, \lambda_2, \ldots, \lambda_p \in \mathbb{R}$, no todas nula y al menos una de ellas positiva tal que

$$\lambda_1 a_1 + \lambda_2 a_2 + \cdots + \lambda_p a_p = 0. \tag{2.4}$$

Consideremos ahora $\alpha \in \mathbb{R}$, entonces, obtenemos

$$(x_1 - \alpha\lambda_1)a_1 + (x_2 - \alpha\lambda_2)a_2 + \cdots + (x_p - \alpha\lambda_p)a_p = b. \tag{2.5}$$

como resultado de multiplicar (2.4) por α y restándola de (2.3). Observamos que para todo α las componentes $x_i - \alpha\lambda_i$ corresponden a una solución de las ecuaciones lineales, pero la solución puede ser no factible ($i.e$, $x_j - \alpha\lambda_j < 0$ para algún $j \in \{1, \ldots, p\}$).
Haciendo $\lambda^T = (\lambda_1, \lambda_2, \ldots, \lambda_p, 0, 0, \ldots, 0)$, se observa que para cualquier α

$$x - \alpha\lambda \tag{2.6}$$

es una solución de las igualdades. Si $\alpha = 0$ tenemos la solución factible original. Como tenemos que al menos una de las λ_i es positiva, como mínimo una componente de $x - \alpha\lambda$ decrecerá a medida que α crece. Incrementamos α hasta que una o más componentes son cero. Específicamente, seleccionamos $\alpha = \bar{\alpha}$, donde

$$\bar{\alpha} = \text{mín}\left\{\frac{x_i}{\lambda_i} : \lambda_i > 0\right\}.$$

Sea k un índice para el cual $\bar{\alpha} = x_k/\lambda_k$, y consideramos

$$x_j - \bar{\alpha}\lambda_j \tag{2.7}$$

- Si $\lambda_j \leq 0$, OK.
- Si $\lambda_j > 0$, entonces $x_j - \bar{\alpha}\lambda_j \geq x_j - (x_j/\lambda_j)\lambda_j \geq 0$.

Acabamos de ver así que la solución dada por (2.6) es factible y tiene a lo sumo $p - 1$ variables positivas.
Al repetir este proceso, en caso necesario, se podrían eliminar las variables positivas hasta tener una solución factible con las columnas correspondientes que son linealmente independientes. En este punto es aplicable el Caso 1.

2.) Sea $x^T = (x_1, x_2, \ldots, x_n)$ una solución factible óptima. Como en 1.) suponemos que hay exactamente p variables positivas $x_1, x_2, \ldots, x_p$. De nuevo hay dos casos; el caso 1, correspondiente a la independencia lineal, es exactamente igual que antes. El caso 2 también se desarrolla igual que el anterior, pero además debemos demostrar que la solución (2.6) es óptima.
Para α suficientemente pequeña la solución dada por (2.6) es una solución factible para valores positivos o negativos de α[5]. Consideremos entonces el valor de la solución (2.6):

$$c^T x - \alpha c^T \lambda.$$

Veamos ahora que $c^T \lambda = 0$. Puesto que si $c^T \lambda \neq 0$, esto sería:

- si $c^T \lambda > 0$, para $\alpha = \bar{\alpha}$, $c^T x - \bar{\alpha} c^T \lambda < c^T x$.
- si $c^T \lambda < 0$, podemos escoger $\alpha < 0$ tal que, $c^T x - \alpha c^T \lambda < c^T x$.

Estaríamos así ante una contradicción por la hipótesis de optimalidad de x. Por tanto necesariamente $c^T \lambda = 0$.
Una vez establecido que la nueva solución factible con menos componentes positivos también es óptima, el resto de la demostración se puede completar exactamente como en la parte 1). □

Observación 2.9. De la demostración del teorema que acabamos de presentar, se deduce que para buscar una solución óptima en un problema de programación lineal, es suficiente con tener en cuenta las soluciones básicas factibles, puesto que en dicho tipo de solución siempre se consigue el valor óptimo.

2.3. Conjuntos convexos

Hasta ahora, nos hemos basado en propiedades elementales de sistemas de ecuaciones lineales. A continuación vamos a ver como interviene el *análisis convexo*[6] en la programación lineal. El vinculo principal entre las teorías algebraica y geométrica es la relación formal existente entre las soluciones básicas factibles y los puntos extremos de los poliedros.

Definición 2.10. *Se dice que un conjunto $C \subset \mathbb{R}^n$ es convexo si $\forall x, y \in C$, $[x, y] \subset C$ donde $[x, y] = \{\lambda x + (1 - \lambda) y \mid 0 \leq \lambda \leq 1\}$.*

Definición 2.11. *$x \in \mathbb{R}^n$ es combinación lineal convexa de $x_1, \cdots, x_k \in \mathbb{R}^n$ si existen $\lambda_1, \cdots, \lambda_k \in \mathbb{R}$ no negativos con $\sum_{i=1}^k \lambda_i = 1$ y $x = \sum_{i=1}^k \lambda_i x_i$.*

Definición 2.12. *Sea $S \subset \mathbb{R}^n$. La envoltura convexa de S, $conv(S)$, se define como el conjunto de todas las combinaciones lineales convexas de puntos de S.*

[5] Para $-\bar{\alpha}$:

- Si $\lambda_j \geq 0$, OK.
- Si $\lambda_j < 0$, entonces puede pasar que $x_j + \bar{\alpha}\lambda_j < 0$. En ese caso seleccionaríamos $\alpha = \text{mín}\left\{\frac{x_i}{\lambda_i} : \lambda_i < 0\right\}$. Volviendo a empezar el análisis, tenemos que si $\lambda_j \geq 0$, OK. Si $\lambda_j < 0$, entonces $x_j - \alpha\lambda_j \geq x_j - (x_j/\lambda_j)\lambda_j \geq 0$.

Hemos encontrado así un $\alpha < 0$ para el cual $x - \alpha\lambda$ es factible.

[6] El análisis convexo es la rama de las matemáticas que se dedica al estudio de las *funciones convexas* y de los conjuntos convexos.

Proposición 2.13. *(La intersección de conjuntos convexos es convexa). Dada una colección de conjuntos convexos $\{C_i\}_{i\in I}$, entonces el conjunto $C^* = \bigcap_{i\in I} C_i$ es un conjunto convexo*

Dem.: Supongamos $C^* \neq \emptyset$ (en el caso contrario, $\emptyset$ es convexo por definición). Entonces, sean x e y dos elementos de C^* y sea $z = \lambda x + (1-\lambda)y$ con $\lambda \in [0,1]$. Tenemos que $\forall i \in I$, $x \in C_i$, $y \in C_i$, entonces, por la convexidad de C_i, $z \in C_i$. Por tanto, $z \in \bigcap_{i\in I} C_i = C^*$. □

Definición 2.14. *Dado un conjunto convexo C, Decimos que $z \in C$ es un vértice o punto extremo de C si no existen dos puntos distintos x, y en C tales que $z = \lambda x + (1-\lambda)y$ con $\lambda \in (0,1)$. En particular, $z = x = y$.*

Definición 2.15. *Un poliedro convexo es una intersección finita de semiespacios cerrados.*

Observación 2.16. Es fácil de demostrar que, dada una matriz $A \in M_{m\times n}(\mathbb{R})$, $b \in \mathbb{R}^m$, la región factible $F = \{x \mid Ax = b, x \geq 0\}$ de un problema de programación lineal, es un poliedro convexo.

Definición 2.17. *Se denomina politopo a la envoltura convexa de un número finito de puntos.*

Observación 2.18. Por definición, todo politopo es un conjunto acotado y es fácil de demostrar que todo poliedro acotado es un politopo y viceversa.

Teorema 2.19. *(Equivalencia de puntos extremos y soluciones básicas). Sea A una matriz $m \times n$ de rango m, $b \in \mathbb{R}^m$. Sea K el poliedro convexo formado por todos los vectores $x \in \mathbb{R}^n$ que satisfacen*

$$\begin{aligned} Ax &= b \\ x &\geq 0. \end{aligned} \tag{2.8}$$

Un vector x es un punto extremo de K si, y sólo si, x es una solución básica factible de (2.8).

Dem.: Si $x^T = (x_1, x_2, \ldots, x_m, 0, 0, \ldots, 0)$ una solución básica fatible de 2.8. Entonces,

$$x_1a_1 + x_2a_2 + \cdots + x_ma_m = b,$$

donde $a_1, a_2, \ldots, a_m$, las m primeras columnas de A, son linealmente independientes. Supongamos ahora que x puede formularse como una combinación convexa de dos puntos distintos en K, por ejemplo, $x = \alpha y + (1-\alpha)z$, $0 < \alpha < 1$, $y \neq z$. Como todas las componentes de x, y, z son ≥ 0 y además $0 < \alpha < 1$, el resultado inmediato es que las últimas $n - m$ componentes de y y z son cero. Así, se tiene en especial

$$y_1a_1 + y_2a_2 + \cdots + y_ma_m = b$$

y

$$z_1a_1 + z_2a_2 + \cdots + z_ma_m = b.$$

Sin embargo, como los vectores $a_1, a_2, \ldots, a_m$ son linealmente independientes, se deduce que $x = y = z$ y, en consecuencia, x es un punto extremo de K.

Recíprocamente, sea x un punto extremo de K. Supongamos que las componentes distintas de cero de x son las p primeras componentes. Entonces,

$$x_1a_1 + x_2a_2 + \cdots + x_pa_p = b,$$

con $x_i > 0$, $i = 1, 2, \ldots, p$.

Para demostrar que x es una solución básica factible, debe demostrarse que los vectores $a_1, a_2, \ldots, a_p$ son linealmente independientes.

Esto se hace por contradicción. Supongamos por tanto que $a_1, a_2, \ldots, a_p$ son linealmente dependientes. Entonces existe una combinación lineal no trivial igual a cero:

$$y_1a_1 + y_2a_2 + \cdots + y_pa_p = 0.$$

Siendo $y^T = (y_1, y_2, \ldots, y_p, 0, 0, \ldots, 0)$, $y \in \mathbb{R}^n$. Como $x_i > 0$, $1 \leq i \leq p$, podemos seleccionar ϵ tal que

$$x + \epsilon y \geq 0, \qquad x - \epsilon y \geq 0.$$

Se tiene entonces que $x = \frac{1}{2}(x + \epsilon y) + \frac{1}{2}(x - \epsilon y)$, que describe a x como una combinación convexa de dos puntos distintos de K, lo que no puede ser, pues x es un punto extremo de K. Así, $a_1, a_2, \ldots, a_p$ son linealmente independientes y x es una solución básica factible. (Aunque si $p < m$, es una solución básica factible degenerada.)

□

Corolario 2.20. *Si existe una solución óptima finita en un problema de programación lineal, existe una solución óptima en un punto extremo.*

Dem.: En particular una solución óptima es una solución básica factible. □

Corolario 2.21. *Si el conjunto convexo K correspondiente a (2.8) es no vacío, entonces tiene al menos un punto extremo.*

Dem.: Esto resulta de la primera parte del teorema (2.8) y del teorema de la equivalencia anterior. □

Corolario 2.22. *El conjunto K convexo correspondiente a (2.8) tiene a lo sumo un número finito puntos extremos.*

Dem.: Obviamente al seleccionar m vectores básicos de las n columnas de A, se obtiene un número finito de soluciones básicas. Los vértices de K son un subconjunto de estas soluciones básicas. □

2.4. El método del símplex

Para la elaboración de este apartado principalmente se han seguido los capítulos 3 y 4 de [1]. Alguno de los subapartados se han completado con ideas extraídas de [3], [4], [5], [6] y [7].

El método del símplex se basa en ir pasando de una solución básica factible (i.e un punto extremo) a otra, hasta encontrar un punto extremo óptimo. Los resultados vistos hasta ahora aseguran que para buscar una solución óptima es suficiente con considerar sólo las soluciones básicas factibles.

Observación 2.23. Para un problema con n variables y m restricciones hay a lo sumo

$$\binom{n}{m} = \frac{n!}{m!(n-m)!}$$

soluciones básicas (que corresponden al número de formas de seleccionar m de n columnas). Es por eso que dado un problema grande, una solución de búsqueda exhaustiva haría el problema inabordable.

Por tanto, es conveniente buscar una forma más eficiente de movernos entre puntos extremos de la región factible, o lo que es lo mismo, una forma eficiente de gestionar los cambios de base. Esto es precisamente lo que intenta el método del símplex.

2.4.1. Formulación general del método del símplex.

consideremos el siguiente problema de programación lineal en forma estándar:

$$\begin{array}{ll} \text{minimizar} & c^T x \\ \text{sujeto a} & Ax = b \\ & x \geq 0 \end{array} \tag{2.9}$$

donde x, $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ y $A \in M_{m\times n}(\mathbb{R})$ de rango m, con $m \leq n$.
Dada una base B con una solución básica factible asociada $x = (x_B, x_N) = (B^{-1}b, 0)$ y cuya función objetivo está dada por z_B, donde

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

Denotemos por J al conjunto de los índices de las variables no básicas, es decir, las columnas de A que no están en la base B, y para cada columna a_j de A, definimos $y_j := B^{-1}a_j$. Entonces, si ponemos a_j como combinación lineal de las columnas de las base B, obtenemos que el coeficiente de la columna i de B es precisamente y_{ij}.
Dado que (x_B, x_N) es una solución basica factible tenemos que $x_B \geq 0$, $x_N = 0$ y $b = Ax = Bx_B + Nx_N$. Despejando x_B en la ecuación anterior obtenemos

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

Esta ecuación refleja como cambiará $x_B = B^{-1}b$ si alguna componente de x_N pasa a ser distinta de cero y al mismo tiempo queremos respetar las restricciones $Ax = b$. Además si queremos encontrar una solución factible que mejore la función objetivo, también deberemos asegurar que respetamos las restricciones de no negatividad.

Entonces, trasladando esto a la función objetivo y definiendo $z_j := c_B^T y_j$ tenemos que, dado $x \in \mathbb{R}^n$ cualquiera,

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

Por tanto para que $z = z_B - \sum_{j \in J} (z_j - c_j) x_j$ sea menor que z_B necesitamos que algún $(z_j - c_j) > 0$ [7]. Es decir, dada una solución básica factible (x_B, x_N), tenemos lo siguiente:

- si $z_j - c_j \leq 0$ para toda variable no básica j, entonces $x = (x_B, x_N)$ és una solución óptima.
- $z_j - c_j > 0$ para alguna variable no básica j, entonces la variable j es candidata a entrar a formar parte de la base. En dicho caso, el vector $y_j = B^{-1} a_j$ me permite saber cómo varían las variables de la base actual a medida que aumento el valor de x_j.

El análisis que acabamos de desarrollar también nos permite identificar cuando estamos ante un problema sin óptimo finito. Supongamos que tenemos una variable no básica $k \in J$ para la cual $z_k - c_k > 0$ y que, al mismo tiempo, el vector $y_k = B^{-1} a_k$ tiene todas sus componentes menores o iguales que cero. Esto nos estaría diciendo que, a medida que aumentamos el valor de x_k, el cambio inducido en las variables de la base (para mantener factibilidad) es tal que ninguna de ellas reduce su valor ($B^{-1} b - y_k x_k \geq B^{-1} b \geq 0$). Por tanto podemos aumentar x_k indefinidamente sin perder factibilidad y mejorando la función objetivo a cada paso.

2.4.2. Esquema general del método del símplex

Presentamos el método del símplex suponiendo que estamos ante un problema de programación lineal en forma estándar.

Algoritmo (Símplex).

Paso 0. *En la forma estándard, determinamos 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, $z_B = c_B^T B^{-1} b$.*

Paso 1. *Calculamos los valores $z_j - c_j$ para todas las variables no básicas $j \in J$, donde $z_j = c_B^T B^{-1} a_j = c_B^T y_j$. Tenemos las siguientes posibilidades:*

- *si $z_j - c_j \leq 0 \ \forall j$, FIN. La solución básica factible $x = (x_B, x_N)$ és óptima.*

[7] En general, la cantidad $c_j - z_j$ se denomina coste reducido de la variable j, pues en caso que $z_j - c_j > 0$ nos indica cuánto tendría que reducirse el coste c_j para que nos interesase que j entrase en la base

- *Si $\exists\ j$ tal que $z_j - c_j > 0$. Sea $k \in J$ el menor índice tal que $z_k - c_k = máx_{j\in J}\{z_j - c_j\}$. Ir al Paso 2 con k como variable de entrada.*

Paso 2. Calculamos $y_k = B^{-1}a_k$. Tememos las siguientes posibilidades:

- *si $y_k \leq 0$, FIN. El problema no tiene óptimo finito.*
- *Si $y_k \not\leq 0$, entonces a_k entra en la base y x_k pasa a ser una variable básica. Abandona la base la variable i con el índice menor:*

$$\frac{(x_B)_i}{y_{ik}} = \min_{1\leq r\leq m}\left\{\frac{(x_B)_r}{y_{rk}} : y_{rk} > 0\right\}.$$

Recalcular B, $x = (x_B, x_N)$, z_B y los vectores y_j. Volver al Paso 1.

2.4.3. Convergencia del método del símplex

El estudio de la convergencia del método del símplex está relacionada con la existencia o no de soluciones básicas degeneradas. Cuando una de las variables básicas toma el valor cero, implica que distintas bases representan al mismo punto extremo de la región factible y por tanto cuando se aplica el algoritmo (símplex) hay distintos cambios de base que pueden no mejorar la función objetivo.

El siguiente resultado nos muestra la convergencia del método en ausencia de soluciones degeneradas.[8]

Teorema 2.24. *Dado un problema de programación lineal en forma estándar sin soluciones básicas degeneradas, y dada una solución básica factible inicial , el método del símplex termina en una cantidad finita de pasos, bien encontrando una solución óptima o concluyendo que no tiene solución óptima finita.*

Dem.: En cada iteración del algoritmo pueden suceder tres cosas:

- Si $z_j - c_j \leq 0\ \forall j$, el algoritmo termina pues estamos en una solución básica factible óptima.
- Si existe k tal que $z_k - c_k > 0$ y, además, $y_k \leq 0$, entonces el algoritmo termina concluyendo que no existe solución óptima finita.
- si existe k tal que $z_k - c_k > 0$ y, además, $y_k \not\leq 0$, se genera una nueva solución básica factible $x^T = (x_B, x_N)$ con $x_B = B^{-1}b$ y $x_N = 0$ siendo B la nueva base resultante de la entrada de a_k y la salida de x_{Bi}. En ausencia de degeneración x_B tiene todas sus componentes estrictamente mayores que cero, y por tanto, $\min_{1\leq r\leq m}\left\{\frac{(x_B)_r}{y_{rk}} : y_{rk} > 0\right\} > 0$. Esta desigualdad, combinada con $z_k - c_k > 0$, implica que la función objetivo decrece estrictamente en cada iteración y por tanto las soluciones básicas factibles obtenidas por el algoritmo son todas distintas entre sí. Como el número total de dichas soluciones es finito el algoritmo terminará en una cantidad finita de iteraciones.

□

[8] En el apartado 2.4.6, se explica que se puede hacer bajo la presencia de soluciones degeneradas.

Observación 2.25. Si $x \in \mathbb{R}^n$ es una solución degenerada de un problema de programación lineal con las condiciones habituales y, x tiene $p < m$ componentes positivas, entonces podrían haber $\binom{n-p}{m-p}$ soluciones básicas factibles diferentes correspondientes a x.

2.4.4. La tabla del método del símplex

Vamos ahora a ilustrar un ejemplo del método del símplex. Para hacerlo usaremos la *tabla del método del símplex*. La idea consiste en guardar en una tabla toda la información relevante de cada iteración a modo de facilitar la actualización de la misma. Veamos primero cómo funciona la tabla.

Supongamos que partimos de un problema en la forma estándar con una solución básica factible asociada $x = (x_B, x_N)$. Además, para facilitar la exposición, si $A = [B, N]$ supongamos que las m columnas de B se corresponden con las primeras m variables. Recordamos que N denota la matriz formada por las columnas de las variables no básicas. Por comodidad $B^{-1}b = \overline{b}$. Entonces representaremos la tabla del símplex de la siguiente forma:

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

Observamos que tenemos la información necesaria para realizar una iteración del método del símplex. En forma menos compacta, tenemos

		x_{B_1}	$\cdots$	x_{B_r}	$\cdots$	x_{B_m}	$\cdots$	x_i	$\cdots$	x_k	$\cdots$	$\overline{z}$
$z_j - c_j$		0	$\cdots$	0	$\cdots$	0	$\cdots$	$z_i - c_i$	$\cdots$	$z_k - c_k$	$\cdots$	$c_B^T \overline{b}$
x_{B_1}	c_{B_1}	1	$\cdots$	0	$\cdots$	0	$\cdots$	y_{1i}	$\cdots$	y_{1k}	$\cdots$	$\overline{b}_1$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$
x_{B_r}	c_{B_r}	0	$\cdots$	1	$\cdots$	0	$\cdots$	y_{ri}	$\cdots$	y_{rk}	$\cdots$	$\overline{b}_r$
$\vdots$	$\vdots$	$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$		$\vdots$
x_{B_m}	c_{B_m}	0	$\cdots$	0	$\cdots$	1	$\cdots$	y_{mi}	$\cdots$	y_{mk}	$\cdots$	$\overline{b}_m$

Supongamos entonces que decidimos que la variable x_k debe entrar en la base y la variable x_{B_r} debe salir de la misma. Se asocia la nueva base a $\tilde{B}$. Las operaciones a realizar sobre la tabla actual, deberían de ser las siguientes:

- se actualiza el vector c_B, y los valores $\overline{b}$ de las variables básicas x_B.
- Se actualizan las columnas y_i y la matriz $I_{m\times m}$ pasa a ser $\tilde{B}^{-1}$.
- se actualizan los valores $z_j - c_j$ y el valor de la función objetivo $c_B^T\overline{b}$.

Obtendríamos de este modo la tabla asociada a la base $\tilde{B}$, y poder así iterar el algoritmo en caso de ser necesario.

veamos ahora un ejemplo del funcionamiento del algoritmo del símplex mediante la tabla.

Ejemplo 2.26. Consideremos el siguiente problema de programación lineal:

$$\begin{array}{ll} \text{minimizar} & -4x_1 - 2x_2 + 6x_3 \\ \text{sujeto a} & -x_1 + x_2 + 2x_3 \leq 8 \\ & 6x_1 + x_2 + 7x_3 \leq 6 \\ & -5x_1 + 6x_3 \leq 1 \\ & x \geq 0 \end{array}$$

el primer paso debe ser pasar el problema a forma estándar, lo que resulta en el problema:

$$\begin{array}{lllllll} \text{minimizar} & -4x_1 - 2x_2 + 6x_3 & & & & & \\ \text{sujeto a} & -x_1 + x_2 + 2x_3 + x_4^s & & & & = 8 \\ & +6x_1 + x_2 + 7x_3 & & + x_5^s & & = 6 \\ & -5x_1 & + 6x_3 & & + x_6^s & = 1 \\ & & & & & x \geq 0 \end{array}$$

<u>*Paso 0.*</u> (Inicialización).
Determinamos $B = (a_4, a_5, a_6) = I_{m\times m}$. $x = (x_B, x_N)^T$ es $x_B = B^{-1}b = (8, 6, 1)^T$ y $x_N^T = (0, 0, 0)$, siendo x una solución básica factible.
Se calcula además el valor de la función objetivo $z_B = c_B^T x_B = (0, 0, 0)x_B = 0$.

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

- $B^{-1}a_1 = y_1 = (-1, 6, -5)^T$. Por tanto, $z_1 - c_1 = c_B^T y_1 - c_1 = 0 - (-4) = 4 > 0$.
- $B^{-1}a_2 = y_2 = (1, 1, 0)^T$. Por tanto, $z_2 - c_2 = c_B^T y_2 - c_2 = 0 - (-2) = 2 > 0$.
- $B^{-1}a_3 = y_3 = (2, 7, 6)^T$. Por tanto, $z_3 - c_3 = c_B^T y_3 - c_3 = 0 - (6) = -6 < 0$.

Entonces la tabla quedaría de la siguiente forma:

[Tabla 1](entra x_1, sale x_5^s)

		x_1	x_2	x_3	x_4^s	x_5^s	x_6^s	$\overline{z}$
$z_j - c_j$		$+4$	$+2$	-6	0	0	0	0
	c_B							x_B
x_4^s	0	-1	1	2	1	0	0	8
x_5^s	0	6	1	7	0	1	0	6
x_6^s	0	-5	0	6	0	0	1	1

El máximo se alcanza para $k = 1$ y es $4 > 0$. Por tanto, tenemos que x_1 es la candidata a entrar en la base.

Paso 2. (criterio de salida). como el vector $y_1 = (-1, 6, -5)^T$ tiene una única componente positiva, que esta asociada con la variable x_5^s, entonces esta sale de la base. La columna a_1 entra en la base y x_1 pasa a ser una variable básica.
La nueva base es $B_1 = (a_4, a_1, a_6)$, con lo que tenemos:

$$B_1 = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 6 & 0 \\ 0 & -5 & 1 \end{pmatrix} \quad \text{y} \quad B_1^{-1} = \begin{pmatrix} 1 & 0.17 & 0 \\ 0 & 0.17 & 0 \\ 0 & 0.83 & 1 \end{pmatrix},$$

y entonces $x_B = B_1^{-1}b = (9, 1, 6)^T$, $x_N = (0, 0, 0)^T$ y $z_{B_1} = c_{B_1}x_B = (0, -4, 0)^T x_B = -4$.

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

- $B_1^{-1}a_2 = y_2 = (1.17, 0.17, 0.83)^T$ y $z_2 - c_2 = c_B^T y_2 - c_2 = -0.67 - (-2) = 1.33 > 0$.
- $B_1^{-1}a_3 = y_3 = (3.17, 1.17, 11.83)^T$ y $z_3 - c_3 = c_B^T y_3 - c_3 = -4.67 - (-6) = -10.67 < 0$.
- $B_1^{-1}a_5 = y_5 = (0.17, 0.17, 0.83)^T$ y $z_5 - c_5 = c_B^T y_5 - c_5 = -0.67 - (0) = -0.67 < 0$.

La tabla quedaría de la siguiente forma:

[Tabla 2](entra x_2, sale x_1)

		x_1	x_2	x_3	x_4^s	x_5^s	x_6^s	$\overline{z}$
$z_j - c_j$		0	+1.33	−10.67	0	−0.67	0	−4
	c_B							x_B
x_4^s	0	0	1.17	3.17	1	0.17	0	9
x_1	−4	1	0.17	1.17	0	0.17	0	1
x_6^s	0	0	0.83	11.83	0	0.83	1	6

El máximo se alcanza para $k = 2$ y es $1.33 > 0$. Por tanto, tenemos que x_2 es la candidata a entrar en la base.

Paso 2.(criterio de salida). como el vector $y_2 = (1.17, 0.17, 0.83)^T$ tiene todas las componente positivas, tenemos que mín$\{\frac{9}{1.17}, \frac{1}{0.17}, \frac{6}{0.83}\} = \frac{1}{0.17}$. Por tanto x_1 sale de la base. La nueva base es $B_2 = (a_4, a_2, a_6)$, con lo que tenemos:

$$B_2 = \begin{pmatrix} 1 & 1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} \quad \text{y} \quad B_2^{-1} = \begin{pmatrix} 1 & -1 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix},$$

y entonces $x_B = B_2^{-1}b = (2, 6, 1)^T$, $x_N = (0, 0, 0)^T$ y $z_{B_2} = c_{B_2}^T x_B = (0, -2, 0)^T x_B = -12$.

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

- $B_2^{-1}a_1 = y_1 = (-7, 6, -5)^T$. Por tanto, $z_1 - c_2 = c_B^T y_1 - c_1 = -12 - (-4) = -8 < 0$.
- $B_2^{-1}a_3 = y_3 = (-5, 7, 6)^T$. Por tanto, $z_3 - c_3 = c_B^T y_3 - c_3 = -14 - (-6) = -20 < 0$.
- $B_2^{-1}a_5 = y_5 = (-1, 1, 0)^T$. Por tanto, $z_5 - c_5 = c_B^T y_5 - c_5 = 2 - (0) = -2 < 0$.

Entonces la tabla quedaría de la siguiente forma:

[Tabla 3](óptima)

		x_1	x_2	x_3	x_4^s	x_5^s	x_6^s	$\overline{z}$
$z_j - c_j$		-8	0	-20	0	-2	0	-12
	c_B							x_B
x_4^s	0	-7	0	-5	1	-1	0	2
x_2	-2	6	1	7	0	1	0	6
x_6^s	0	-5	0	6	0	0	1	1

El máximo se alcanza para $k = 5$ y es $-2 < 0$. Por tanto, estamos ante una solución óptima.
La solución óptima, expresada únicamente con las variables del problema original, viene dada por $(0, 6, 0)$ con valor óptimo -12.

2.4.5. Construcción de una solución básica factible inicial

Pasamos a ver ahora como generar una solución básica factible dado un programa lineal. Si las condiciones del problema son de la forma $Ax = b$, $x \geq 0$, con $b \geq 0$ para cada una de sus componentes, y la matriz de restricciones A contiene la matriz identidad $I_{m\times m}$ entonces el problema de encontrar una solución básica factible $x = (x_B, x_N)$ es inmediato. Basta con seleccionar las columnas asociadas a dicha submatriz identidad y la solución inicial será entonces $x_B = B^{-1}b = b \geq 0$ y $x_N = 0$.

Observación 2.27. Si las condiciones del problema son $Ax \leq b$ y $x \geq 0$, si pasamos el problema a la forma estándar añadiendo variables de holgura como hemos visto en la sección 2.1, entonces estaríamos ante el mismo caso, siendo $[A, I]x = b$ el problema equivalente en forma estándar y $x_B^T = (x_{n+1}^s, \cdots, x_{n+m}^s)$.

Suponiendo que no estamos ante las condiciones anteriores, podemos usar el llamado *método de la M grande* (o "*Big-M*").
En la forma estándar, la idea del método consiste en añadir *variables artificiales* en tantas restricciones como sea necesario para que la nueva matriz de restricciones contenga la matriz identidad como submatriz, lo que nos pone nuevamente en las condiciones del caso anterior. En la función objetivo se añade además una penalización de las variables artificiales dado por una constante M suficientemente grande.

consideremos el problema en la forma estándar

$$\begin{array}{ll} \text{minimizar} & c^T x \\ \text{sujeto a} & Ax = b \\ & x \geq 0, \end{array} \tag{2.10}$$

donde para simplificar y sin perdida de generalidad, suponemos $b \geq 0$, y le añadimos un vector de variables artificiales x_a y una penalización M a la función objetivo, de manera que

$$\begin{array}{ll} \text{minimizar} & \sum_{i=1}^{n} c_i x_i + M \sum_{j=n+1}^{m} x_{a_j} \\ \text{sujeto a} & Ax + x_a = b \\ & x, x_a \geq 0. \end{array} \tag{2.11}$$

Una vez introducida la penalización, mediante el método del símplex, buscamos una solución óptima para (2.11). Si el algoritmo termina y alguna de las variables artificiales tienes valor positivo en la solución óptima, entonces sabremos que el problema original (2.10) no tenía soluciones factibles.

Observación 2.28. Las variables artificiales aumentan el conjunto de soluciones factibles del problema original. De hecho, toda solución factible del problema original se puede representar como una solución factible del problema ampliado en el que todas las variables artificiales valen 0.

Veamos a continuación un ejemplo de como encontrar una solución básica factible usando el método de la *M grande*.

Ejemplo 2.29. Supongamos que tenemos el siguiente problema de programación lineal

$$\begin{array}{lll} \text{minimizar} & 3x_1 - 3x_2 + 3x_3 - 4x_4 & \\ \text{sujeto a} & -x_1 + x_2 - 6x_3 & \geq 1 \\ & 5x_1 + 2x_2 + 2x_3 - x_4 & \geq 7 \\ & -x_1 + 3x_2 - x_3 + 2x_4 & \leq 7 \\ & & x \geq 0. \end{array}$$

Si lo pasamos a forma estándar, obtenemos

$$\begin{array}{lllll} \text{minimizar} & 3x_1 - 3x_2 + 3x_3 - 4x_4 & & & \\ \text{sujeto a} & -x_1 + x_2 - 6x_3 - x_5^s & & & = 1 \\ & 5x_1 + 2x_2 + 2x_3 - x_4 & - x_6^s & & = 7 \\ & -x_1 + 3x_2 - x_3 + 2x_4 & & + x_7^s & = 7 \\ & & & & x \geq 0. \end{array}$$

Si consideramos la matriz de coeficientes $A = (a_1, \cdots, a_7)$, observamos que la columna a_7 se corresponde con el vector $(0, 0, 1)$, pero todavía nos faltan las columnas $(1, 0, 0)$ y $(0, 1, 0)$ para que se cumpla que la matriz identidad es una submatriz de A. Siguiendo el método de la *M grande*, añadiendo las *variables artificiales* x_8^a y x_9^a nos queda el problema

$$\begin{array}{lllllll} \text{minimizar} & 3x_1 - 3x_2 + 3x_3 - 4x_4 & & & + Mx_8^a & + Mx_9^a & \\ \text{sujeto a} & -x_1 + x_2 - 6x_3 - x_5^s & & & + x_8^a & & = 1 \\ & 5x_1 + 2x_2 + 2x_3 - x_4 & - x_6^s & & & + x_9^a & = 7 \\ & -x_1 + 3x_2 - x_3 + 2x_4 & & + x_7^s & & & = 7 \\ & & & & & & x \geq 0. \end{array}$$

De este modo podríamos tomar como base inicial $B = (a_7, a_8, a_9)$ con solución asociada $x_B^T = (7, 1, 7)$ y $x_N = 0$.
Llegados ha este punto, habría que iterar el método del símplex y ver si en la solución óptima todas las variables artificiales se han hecho cero o no para decidir si se ha encontrado una solución óptima del problema original o éste no tenia soluciones factibles.

2.4.6. Reglas de prevención de ciclo

Hasta ahora hemos podido ver la convergencia del método del símplex bajo condiciones de no degeneración. Ante la presencia de soluciones básicas degeneradas, existen métodos diversos para evitar un posible ciclo durante el desarrollo del símplex, garantizando así la convergencia del método. En este apartado nos centraremos en la *regla lexicográfica*; consiste en establecer un criterio para escoger la variable de salida de la base en cada iteración del método del símplex, asegurando que no exista la posibilidad de repetir ninguna base a lo largo del proceso.

Sea $x = (x_B, x_N)$ una solución básica factible con base B asociada. supongamos que x_k es la variable escogida para entrar en la base. Entonces, para determinar la variable que abandona la base calculan los índices de las variables candidatas con el criterio ya expuesto en el *Paso 2.* del algoritmo (símplex):

$$I_0 = \left\{ i : \frac{(x_B)_i}{y_{ik}} = \min_{1 \leq r \leq m} \left\{ \frac{(x_B)_r}{y_{rk}} : y_{rk} > 0 \right\} \right\}.$$

Si de aquí se deduce que solo hay una variable candidata, se elige ésta. En caso de existir más de una candidata, se calculan los índices:

$$I_1 = \left\{ i : \frac{y_{i1}}{y_{ik}} = \min_{r \in I_0} \left\{ \frac{y_{r1}}{y_{rk}} \right\} \right\}.$$

De nuevo, si podemos deducir que solo hay una variable candidata, se elige esta. De otra forma, se repite el proceso. Generalizando tenemos

$$I_j = \left\{ i : \frac{y_{ij}}{y_{ik}} = \min_{r \in I_{j-1}} \left\{ \frac{y_{rj}}{y_{rk}} \right\} \right\},$$

con $j \leq m$. Llegando a un único índice para alguno de los I_j.

Nos aseguraríamos así de no repetir ninguna base en el desarrollo del símplex . No vamos a ha hacer una demostración formal de este hecho.

2.5. Dualidad

A todo problema de programación lineal le podemos asociar un nuevo problema de programación lineal, conocido como *problema dual*. Este nuevo problema tiene importantes propiedades. En particular, puede ser usado para resolver el problema original, que llamaremos primal durante esta sección.
Empezamos presentando la formulación dual de un problema de programación lineal en forma estándar

$$\begin{array}{lll} & \boldsymbol{Primal} & \\ \text{minimizar} & c^T x & \\ \text{sujeto a} & Ax = b & \\ & x \geq 0 & \end{array} \qquad \begin{array}{ll} & \boldsymbol{Dual} \\ \text{maximizar} & \lambda^T b \\ \text{sujeto a} & \lambda^T A \leq c^T \end{array} \tag{2.12}$$

Donde x, $c \in \mathbb{R}^n$, b, $\lambda \in \mathbb{R}^m$ y $A \in M_{m\times n}(\mathbb{R})$, con $m \leq n$. x es la variable del problema primal y λ es la variable del problema dual.

En esta sección no se supone que A sea de rango completo. Veamos a continuación algunas relaciones entre el problema primal y su dual:

- Si el primal es un problema de minimización, entonces el dual lo es de maximización (y viceversa).
- La matriz del problema dual es la matriz traspuesta A^T del problema primal. ($\lambda^T A = c^T$ es equivalente a $A^T\lambda = c$).
- Cada variable del primal corresponde con una restricción del dual.
- Cada restricción del primal se corresponde con una variable del dual.
- El vector b de términos independientes del primal pasa a ser el vector de coeficientes de la función objetivo del dual.
- El vector c de coeficientes de la función objetivo del primal pasa a ser el vector de términos independientes del dual.
- Restricciones de igualdad pasan a variables no restringidas y restricciones de mayor o igual pasan a variables no negativas.

La siguiente tabla resume las principales relaciones entre el primal y el dual:

	Primal		*Dual*	
Variables	≥ 0	$\longleftrightarrow$	$\leq$	Restricciones
	≤ 0	$\longleftrightarrow$	$\geq$	
	no restr.	$\longleftrightarrow$	$=$	
Restricciones	$\geq$	$\longleftrightarrow$	≥ 0	Variables
	$\leq$	$\longleftrightarrow$	≤ 0	
	$=$	$\longleftrightarrow$	no restr.	

Proposición 2.30. *El dual del dual coincide con el primal.*

Dem.: Sea el dual de un problema de programación lineal en forma estándar, es decir:

$$\begin{array}{ll} \text{máx} & \lambda^T b \\ \text{sujeto a} & \lambda^T A \leq c^T \end{array}$$

Se expresa de forma equivalente

$$\begin{array}{ll} -\text{mín} & -b^T(u-v) \\ \text{sujeto a} & -A^T(u-v) \geq -c \\ & u \geq 0 \\ & v \geq 0 \end{array}$$

donde hemos considerado que $\lambda = u - v$. Además podemos expresar

$$-b^T(u-v) = \left(-b^T, b^T\right)\begin{pmatrix} u \\ v \end{pmatrix}$$

y

$$-A^T(u-v) = \left(-A^T, A^T\right)\begin{pmatrix} u \\ v \end{pmatrix}$$

Entonces, el dual correspondiente es

$$\begin{array}{ll} -\text{máx} & -y^T c \\ \text{sujeto a} & y^T(-A^T, A^T) \leq (-b^T, b^T) \\ & y \geq 0, \end{array}$$

que a su vez es equivalente a

$$\begin{array}{ll} \text{mín} & c^T y \\ \text{sujeto a} & Ay = b \\ & y \geq 0 \end{array}$$

que efectivamente coincide con el primal en su forma estándar. □

Para facilitar la exposición presentaremos siempre el problema primal como un problema de minimización y el dual como un problema de maximización.

2.5.1. Dualidad débil y dualidad fuerte

Empezamos este apartado presentando otra relación importante entre el primal y el dual. Supongamos sin pérdida de generalidad, que tenemos la formulación dual de un problema primal expresado en forma estándar como en (2.12).

Proposición 2.31 (**Dualidad débil**). *Sean x, λ factibles para el problema primal y dual respectivamente de (2.12), entonces se verifica $c^T x \geq \lambda^T b$.*

Dem.: Se tiene

$$\lambda^T b = \lambda^T A x \leq c^T x,$$

siendo válida la última desigualdad, porque $x \geq 0$ y $\lambda^T A \leq c^T$. □

Este resultado tiene varias implicaciones prácticas. La primera de ellas es que un vector factible para uno de los problemas da lugar a una cota en el valor del otro. Los valores asociados al primal son mayores que los asociados al dual.
De esta proposición se deriva de forma inmediata el siguiente corolario.

Corolario 2.32. *Si x^* y λ^* son soluciones factibles del primal y del dual tales que $c^T x^* = (\lambda^*)^T b$, entonces x^* y λ^* son soluciones optimas del primal y dual, respectivamente.*

A continuación vamos a ver la demostración del teorema de dualidad fuerte, utilizando las características del método del símplex.

Teorema 2.33 (**Dualidad fuerte**)**.** *Dada la formulación dual de un problema de programación lineal en forma estándar, Si uno de los problemas primal o dual tiene una solución optima finita, también la tiene el otro, y los valores correspondientes de las funciones objetivo coinciden.*
Si la función objetivo de uno de ellos es no acotada, el otro no tiene soluciones factible.

Dem.: Observemos que el segundo enunciado es una consecuencia directa de la dualidad débil, ya que si el primal no está acotado y λ es factible para el dual, se debe tener $\lambda^T b \leq -M$ para una M arbitrariamente grande, lo cual evidentemente, es imposible.

Supongamos ahora, que para el programa lineal en forma estándar

$$\begin{array}{ll} \text{minimizar} & c^T x \\ \text{sujeto a} & Ax = b \\ & x \geq 0, \end{array} \tag{2.13}$$

se tiene la solución óptima $x^* = (x_B, 0)$ con la base B correspondiente.
Entonces, queremos determinar una solución del problema dual

$$\begin{array}{ll} \text{maximizar} & \lambda^T b \\ \text{sujeto a} & \lambda^T A \leq c^T \end{array} \tag{2.14}$$

en función de B.
Se divide A como $A = [B, N]$. Como la solución básica factible $x_B = B^{-1}b$ es óptima, entonces de la sección (2.4.1) sabemos que $z_j - c_j \leq 0$ para toda variable no básica j, que, es equivalente a $c_B^T B^{-1} a_j - c_j \leq 0$ y por tanto resulta $c_B^T B^{-1} N \leq c_N^T$.

Ahora, definimos $\lambda^T = c_B^T B^{-1}$. Demostramos que esta elección de λ resuelve el problema dual. Tenemos

$$\lambda^T A = \left[\lambda^T B, \lambda^T N\right] = \left[c_B^T, c_B^T B^{-1} N\right] \leq \left[c_B^T, c_N^T\right] = c^T.$$

Así, como $\lambda^T A \leq c^T$, λ es factible para el dual. Por otro lado,

$$\lambda^T b = c_B^T B^{-1} b = c_B^T x_B,$$

y el valor de la función objetivo dual para esta λ es igual al valor del problema primal. Por tanto por la *dualidad débil*, queda establecida la optimalidad de λ para el dual.

□

De este teorema, se deriva de forma inmediata el siguiente corolario:

Corolario 2.34. *Sea x^* solución óptima con base B asociada de un programa lineal en forma estándar como el de (2.12), entonces*

$$\lambda^T = c_B^T B^{-1}$$

es una solución óptima del problema dual.

Observación 2.35. De la demostración del teorema de dualidad fuerte, se deduce que si una solución cumple las condiciones de optimalidad del primal (*i.e*, $z_j - c_j \leq 0$), automáticamente es factible en el dual (sea factible o no para el primal).

2.5.2. Holguras complementarias

Pasamos a ver ahora la relación existente entre las soluciones óptimas de los problemas primal y dual. Presentamos entonces el teorema de holguras complementarias, ofreciendo así una caracterización de las soluciones óptimas de dichos problemas.
Para el teorema nos basaremos la formulación canónica de un problema de programación lineal y su dual, es decir en el par:

$$\begin{array}{llll} & \boldsymbol{Primal} & & \boldsymbol{Dual} \\ \text{minimizar} & c^T x & \text{maximizar} & \lambda^T b \\ \text{sujeto a} & Ax \geq b & \text{sujeto a} & \lambda^T A \leq c^T \\ & x \geq 0 & & \lambda^T \geq 0. \end{array} \tag{2.15}$$

Teorema 2.36. *(Teorema de las holguras complementarias). Sean x y λ soluciones factibles de un problema de programación lineal en forma canónica y su formulación dual (como por ej. (2.15)) , respectivamente. Entonces, x y λ forman un par de soluciones óptimas si y sólo si*

$$\lambda_i(a^{(i)}x - b_i) = 0 \ \forall i \in \{1, \ldots, m\},^9 \tag{2.16}$$

$$(c_j - \lambda^T a_j)x_j = 0 \ \forall j \in \{1, \ldots, n\}. \tag{2.17}$$

Dem.: Definimos,

$$u_i := \lambda_i(a^{(i)}x - b_i) \ \forall i \in \{1, \ldots, m\},$$

y

$$v_j := (c_j - \lambda^T a_j)x_j \ \forall j \in \{1, \ldots, n\}.$$

Claramente de las relaciones entre el primal y el dual tenemos que $\forall i \in \{1, \ldots, m\}$ $u_i \geq 0$ y $\forall j \in \{1, \ldots, n\}$ $v_j \geq 0$.
Definimos ahora $U := \sum_{i=1}^m u_i$ y $V := \sum_{i=1}^n v_i$. Entonces, $U = V = 0$ si y sólo se cumplen (2.16) y (2.17). Además,

$$U + V = \sum_{i=1}^m u_i + \sum_{i=1}^n v_i = c^T x + \lambda^T Ax - \lambda^T Ax - \lambda^T b = c^T x - \lambda^T b.$$

Finalmente, se verifican (2.16) y (2.17) si y sólo si $U + V = 0$ ($\Leftrightarrow c^T x = \lambda^T b$), que, por el corolario (2.32), sabemos que es una condición necesaria y suficiente tales que tanto x como λ sean soluciones óptimas del primal y dual, respectivamente. □

[9] $a^{(i)}$ representa la fila i de la matriz $A_{m \times n}$.

Observación 2.37. El resultado que acabamos de ver, implica que dadas un par de soluciones óptimas x y λ, si una restricción del dual no se satura[10], entonces la correspondiente variable del primal es 0 ($i.e\ c_j - \lambda^T a_j > 0 \Rightarrow x_j = 0$).

2.5.3. Análisis de sensibilidad

El análisis de sensibilidad nos permite estudiar como variaciones en los parámetros del problema de partida se traducen en variaciones sobre la solución óptima y función objetivo del problema asociado.

Supongamos que dado el problema (2.15), conocemos una base óptima B del problema primal. Además, para facilitar la exposición, asumiremos que la solución básica óptima asociada, x es no degenerada. Del corolario (2.34) sabemos que $\lambda^T = c_B^T B^{-1}$ es una solución óptima del dual. Si denotamos J como el conjunto de variables no básicas, sabemos también de la sección (2.4.1) que

$$z = c_B^T B^{-1} b - \sum_{j \in J} (z_j - c_j)\, x_j = \lambda^T b - \sum_{j \in J} (z_j - c_j)\, x_j.$$

Como estamos asumiendo que la solución es no degenerada, tenemos que $x_B = B^{-1}b > 0$, con lo que si perturbásemos ligeramente el vector b, seguiríamos teniendo una solución básica factible asociada a la base B. Para ver el cambio de dicha perturbación en b, si definimos $\bar{z} = \lambda^T b = c_B^T B^{-1} b$, tenemos que

$$\frac{\partial \bar{z}}{\partial b_i} = \lambda_i.$$

Es decir, λ_i nos da la tasa de variación de la función objetivo ante una perturbación en b_i. Como $\lambda_i > 0$, aumentando el valor de b_i la función objetivo empeora (aumenta), y análogamente mejora si reducimos b_i.

Relacionando la presentación que acabamos de hacer con el teorema de holguras complementarias, podemos interpretar que cuando una restricción no se satura, *no tenemos nada que ganar* y por eso la variable dual asociada es 0.

El estudio de las perturbaciones sobre el vector c es similar a la exposición que acabamos de ver; de hecho el vector c en el dual se comporta como el vector b en el primal. Por lo general si perturbamos c en el primal podemos tener pérdida de optimalidad, pero nunca a pérdida de factibilidad. Una manera de ver como ha cambiado el valor óptimo $c^T x$ después de perturbar el vector c, sería iterando el método del símplex en el propio primal a partir de la solución óptima anterior.

Aunque no vamos a ofrecer un análisis detallado, supongamos ahora que quisiéramos realizar alguna perturbación sobre la matriz A de restricciones. Distinguimos dos casos:

- **Cambios en una columna no básica.** Si perturbamos una de las columnas no básicas de la matriz A, por ejemplo cambiando a_i por $\hat{a}_j$, solamente se vería afectada

[10] Decimos que una restricción esta saturada en x, si se cumple $a^{(i)}x = b_i$ donde $x \in F$ con F la región factible de un programa lineal.

la columna correspondiente de la tabla[11] incluyendo el $z_j - c_j$ correspondiente. Para este caso sería suficiente con recalcular la columna $B^{-1}\hat{a}_j$ y obtener $\hat{z}_j - c_j = C_B^T B^{-1}\hat{a}_j - c_j$. Si $\hat{z}_j - c_j \leq 0$ mantenemos optimalidad y nada cambia. En caso contrario habría que iterar el método del s´simplex hasta recuperar optimalidad (la columna j entraría en la base).

- **Cambios en una columna básica**. El cambió sobre una columna básica, puede implicar que tengamos dependencia de la matriz B, y por tanto dejemos de tener una base. Pero aun sin perder la independencia de la base, cambiaría B^{-1} y eso podría provocar que cambien todas las columnas de la tabla del símplex.

Ilustramos con un ejemplo simple los cambios que se producen en un programa lineal tras aplicar alguna perturbación en el vector b:

Ejemplo 2.38. Consideremos el problema

$$\begin{array}{lrl} \text{minimizar} & -2x_1 - 3x_2 - 4x_3 & \\ \text{sujeto a} & x_1 + 2x_2 + 3x_3 & \leq 11 \\ & 2x_1 + 3x_2 + 2x_3 & \leq 10 \\ & x & \geq 0. \end{array}$$

Si lo resolviendo mediante la tabla del símplex, obtendremos la tabla óptima:

[Tabla](óptima)

		x_1	x_2	x_3	x_4^s	x_5^s	$\overline{z}$
$z_j - c_j$		0	$-\frac{1}{2}$	0	-1	$-\frac{1}{2}$	-16
	c_B						x_B
x_3	-4	0	$\frac{1}{4}$	1	$\frac{1}{2}$	$-\frac{1}{4}$	3
x_1	-2	1	$\frac{5}{4}$	0	$-\frac{1}{2}$	$\frac{3}{4}$	2

con $x_B = (3, 2)$ y valor óptimo $z = -16$.

De forma general, supongamos que perturbamos una de las componentes del vector b de forma que $\hat{b} = b + (0, \ldots, 0, \Delta b_i, 0, \ldots, 0)^T$. Si B_i^{-1} es la columna i de B^{-1}, queremos encontrar las condiciones que nos aseguren,

$$\hat{x}_B = B^{-1}\hat{b} = B^{-1}b + B_i^{-1}\Delta b_i \geq 0.$$

Es decir que mantenemos la factibilidad y, por lo tanto, como los $z_j - c_j$ no cambian, mantenemos también optimalidad.

Veamos ahora entonces qué pasa si en el problema dado realizamos cambios en b_1:

de la tabla, tenemos que $B^{-1} = \begin{pmatrix} \frac{1}{2} & \frac{-1}{4} \\ \frac{-1}{2} & \frac{3}{4} \end{pmatrix}$, y por lo tanto

$$B^{-1}b + B_i^{-1}\Delta b_i = (3, 2) + (1/2, -1/2)\Delta b_1 \geq 0 \Rightarrow \begin{cases} 3 + \Delta b_1/2 \geq 0 \Rightarrow \Delta b_1 & \geq -6 \\ 2 - \Delta b_1/2 \geq 0 \Rightarrow \Delta b_1 & \leq 4. \end{cases}$$

[11] Nos referimos a la tabla del método del símplex; se puede encontrar información mas detallada en la sección 2.4.4.

Es decir, si $\Delta b_1 \in [-6, 4]$ se mantiene la factibilidad para $\hat{x}_B$. Equivalentemente, podemos variar b_1 en el intervalo $[11 - 6, 11 + 4] = [5, 15]$.
Observamos además, si λ^T es una solución óptima del dual, entonces

$$\lambda^T = c_B^T B^{-1} = (-4, -2)B^{-1} = (-1, -1/2).$$

Por cada unidad en que aumentemos b_1 dentro de $[5, 15]$, la función objetivo se verá reducida en $\lambda_1 = -1$ unidad; si por ejemplo tenemos $\hat{b} = (12, 10)$, entonces el valor óptimo de la función objetivo pasa a ser $\hat{z} = -17$.
Realizando el mismo análisis para b_2, la tasa de variación de la función objetivo sería $\lambda_2 = -1/2$.[12]

2.5.4. Formulación del algoritmo primal-dual

El algoritmo primal-dual es una variante del método del símplex, que consiste en resolver un problema de programación lineal operando de forma simultanea en los problemas primal y dual.
Consideremos un problema de programación lineal en forma estándar y su formulación dual como el que sigue, con las condiciones habituales:

	Primal		***Dual***
minimizar	$c^T x$	maximizar	$\lambda^T b$
sujeto a	$Ax = b$	sujeto a	$\lambda^T A \leq c^T$
	$x \geq 0$		

Para facilitar la exposición nos referiremos a los problemas primal y dual como P y D respectivamente. En el algoritmo primal-dual, primero es necesario disponer de una solución factible λ para D, que se puede obtener del siguiente modo:

- Si $c \geq 0$, entonces $\lambda = 0$ solución factible de D.
- Si $\exists j$ con $c_j < 0$, entonces añadimos al problema P una variable x_{n+1} junto a la restricción $x_1 + x_2 + \cdots + x_n + x_{n+1} = b_{m+1}$; donde $b_{m+1} > \text{máx}_{1 \leq i \leq n} \{x_i\}$ para cualquier posible valor de las variables $x_1, \ldots, x_n$.[13]
  El nuevo problema dual tendrá una variable λ_{m+1} y se formulará de la siguiente forma:

$$\begin{array}{lrll} \text{maximizar} & \lambda^T b + \lambda_{m+1} b_{m+1} & & \\ \text{sujeto a} & \lambda^T a_j + \lambda_{m+1} & \leq c_j & (j \in \{1, \ldots, n+1\}) \\ & \lambda_{m+1} & \leq 0. & \end{array}$$

  Escogiendo $\lambda = (0, 0, \ldots, 0, \hat{c})^T$ con $\hat{c} = \text{mín}\{c_i \mid c_i < 0\}$, tenemos que λ es una solución factible de este problema (pues $\hat{c} < 0$).

Supongamos entonces que disponemos de una solución λ factible para D. Si encontramos una solución x factible[14] de P, tal que $x_j = 0 \ \forall j$ con $c_j - \lambda^T a_j > 0$, por el teorema de las holguras complementarias, tanto x como λ serían óptimos[15].

[12] Los $z_j - c_j$ asociados a las variables de holgura coinciden con la solución óptima del problema dual, pues $z_j = c_B^T B^{-1} e_j$ y $c_j = 0$.

[13] una cota para este valor se puede encontrar fácilmente a partir de la matriz A.

[14] Obsérvese que si x es factible para P, automáticamente se cumpliría la ecuación $\lambda_i(a^{(i)}x - b_i) = 0$ del teorema de las holguras complementarias.

[15] El algoritmo primal-dual, se basa precisamente buscar una solución x que cumpla dichas condiciones.

En este caso, para nuestra solución λ tendremos que algunas de las restricciones de la forma $\lambda^T a_i \leq c_i$ estarán saturadas. Definimos el conjunto

$$K_A := \{k \in N \mid \lambda^T a_k = c_k\}, \quad \text{para } N = \{1, \ldots, n\}.$$

Entonces una solución factible x de P será óptima si y sólo si para todo $k \notin K_A,\ x_k = 0$. Por tanto, sera suficiente encontrar una solución factible verificando

$$\begin{aligned} \sum_{k \in k_A} a_k x_k &= b \\ x_k &\geq 0 \qquad k \in K_A \\ x_j &= 0 \qquad j \notin K_A. \end{aligned} \tag{2.18}$$

Nos referiremos a K_A como el conjunto de *columnas activas*[16]. Para buscar una solución de (2.18) consideremos una versión restringida de P que llamaremos RP, y que es de la forma:

$$\begin{aligned} \text{minimizar} \quad & \sum_{i=1}^{m} x_i^a \\ \text{sujeto a} \quad & \sum_{k \in K_A} a_k x_k + x^a = b \\ & x_k \geq 0 \qquad k \in K_A \\ & x_j = 0 \qquad j \notin K_A \\ & x_i^a \geq 0 \qquad i \in \{1, \ldots, m\}. \end{aligned}$$

En el cual se ha introducido un vector de *variables artificiales* $x^a = (x_1^a, \ldots, x_m^a)$, de dimensión m. Una componente en cada restricción de P. Partiendo de $x_i^a = b_i$ optimizamos.

- Si el valor óptimo de RP es 0, entonces x verifica (2.18) y consecuentemente x es solución óptima de P.
- Si el valor óptimo de RP es mayor que 0, en este caso no existe solución en el primal que verifique las condiciones de holguras complementarias, y es necesario cambiar λ. se procede del siguiente modo con la formulación del dual de RP, que llamaremos DRP. Entonces DRP será:

  $$\begin{aligned} \text{maximizar} \quad & \lambda^T b \\ \text{sujeto a} \quad & \lambda^T a_k \leq 0 \qquad k \in K_A \\ & \lambda_i \leq 1 \qquad i \in \{1, \ldots, m\}. \end{aligned}$$

  Denotemos por μ el óptimo de DRP. Entonces se repite el procedimiento, pero modificando λ por una combinación lineal de μ con el propio λ, esto es:

  $$\bar{\lambda} = \lambda + \omega\mu.$$

  Queremos ω que nos mantenga factibilidad en D, y que además mejore el valor objetivo, $\bar{\lambda}^T b = \lambda^T b + \omega\mu^T b$. Como RP y RPD son un par primal-dual, tenemos

[16] Observamos que las igualdades usan sólo columnas de A correspondientes con elementos de K_A, que provienen de las restricciones saturadas en D (para la solución λ).

que $\mu^T b > 0$ (ya que ha de coincidir con el óptimo de RP). Entonces, si $\omega > 0$ el valor objetivo en D mejora.
Por otro lado para que $\bar{\lambda}$ sea factible, necesariamente $\forall i \in \{1, \ldots, n\}$,

$$\bar{\lambda}^T a_i = \lambda^T a_i + \omega \mu^T a_i \leq c_i.$$

Observamos que si $\forall i \in \{1, \ldots, n\}$, $\mu^T a_i \leq 0$, entonces podemos aumentar ω indefinidamente, y por tanto como D sería no acotado, consecuentemente P no tendrá solución factible.
En otro caso, seleccionamos $\omega = \hat{\omega}$ donde

$$\hat{\omega} = \min_{i \notin K_A, \mu^T a_i > 0} \left\{ \frac{c_i - \lambda^T a_i}{\mu^T a_i} \right\}.$$

Se reemplaza λ por $\bar{\lambda}$ y así sucesivamente hasta que lleguemos a un problema en el que el óptimo de RP es 0 o que nos permita asegurar que P no tiene soluciones factibles.

Observación 2.39. Bajo las condiciones del algoritmo primal-dual, se mantiene constantemente la factibilidad de λ en el dual.

2.5.5. Algoritmo primal-dual

El Algoritmo primal-dual, resulta de gran utilidad para situaciones en las que no disponemos de una solución básica factible en el primal.

Para finalizar esta sección y teniendo en cuenta la exposición anterior, presentamos el algoritmo primal-dual y posteriormente un breve análisis de la convergencia del mismo seguido por un ejemplo.

Algoritmo (Primal-Dual).

Paso 1. *Disponer de una solución factible λ del problema dual D.*

Paso 2. *Se define el conjunto K_A asociado a λ y se resuelve el problema relajado primal RP.*

- *(i) Si el valor óptimo de RP es 0, la solución correspondiente x es óptima para el problema primal P.*
- *(ii) Si el valor óptimo de $RP > 0$ y, $\forall j \notin K_A$, $\mu^T a_j \leq 0$, P no tiene soluciones factibles.*
- *(iii) En otro caso, λ pasa a ser $\bar{\lambda}$, y se vuelve a empezar el Paso 2.*

El método descrito converge puesto que si $\hat{k}$ es el índice que determina la selección de $\hat{\omega}$, entonces, tendremos que la columna $\hat{k}$ pasará a ser activa puesto que $\bar{\lambda}^T a_k = c_{\hat{k}}$. Suponiendo además ausencia de degeneración[17], el problema dual va mejorando a cada iteración, con lo que eventualmente, se recorrerán todas las posibles combinaciones de columnas activas y no activas.

Ilustramos el algoritmo con el siguiente ejemplo:

[17] En caso de degeneración se podría aplicar alguna regla que asegure que no se repite ninguna base como ya se ha visto para el símplex.

Ejemplo 2.40. Dado el problema (P):

$$\begin{array}{lrl} \text{minimizar} & 2x_1 + x_2 + 4x_3 & \\ \text{sujeto a} & x_1 + x_2 + 2x_3 & = 3 \\ & 2x_1 + x_2 - 3x_3 & = 5 \\ & & x \geq 0. \end{array}$$

Iteración 1.

Lo primero que necesitamos es una solución factible del problema dual. Como tenemos que $c = (2, 1, 4)^T \geq 0$, podemos considerar $\lambda = (0, 0)^T$.

Paso 2.

- $K_A = \emptyset$. Para la versión restringida RP_1, tenemos que la solución óptima es $x = (0, 0, 0, 3, 5)^T$ con valor óptimo $z_{RP1} = 8 > 0$.
- La solución factible de DRP es $\mu = (1, 1)^T$. como se cumple $\exists j \notin K_A$, $\mu^T a_j > 0$, seguimos.
  $\omega = \text{mín}\{\frac{2}{3}, \frac{1}{2}\} = 1/2 \Rightarrow \bar{\lambda} = (0, 0)^T + \frac{1}{2}(1, 1)^T = (\frac{1}{2}, \frac{1}{2})^T$.
- Actualizamos $\lambda = (\frac{1}{2}, \frac{1}{2})^T$. Volvemos a iterar el *Paso 2*.

Iteración 2.

Paso 2.

- $K_A = \{2\}$. Para la versión restringida RP_2, tenemos que la solución óptima es $x = (0, x_2, 0, x_1^a, x_1^a)^T$ con valor óptimo $z_{RP2} > 0$.
- La solución factible de DRP es $\mu = (-1, 1)^T$. como se cumple $\exists j \notin K_A$, $\mu^T a_j > 0$, seguimos.
  $\omega = \text{mín}\{\frac{1}{2}, \frac{3}{2}\} = 1/2 \Rightarrow \bar{\lambda} = (\frac{1}{2}, \frac{1}{2})^T + \frac{1}{2}(-1, 1)^T = (0, 1)^T$.
- Actualizamos $\lambda = (0, 1)^T$. Volvemos a iterar el *Paso 2*.

Iteración 3.

Paso 2.

- $K_A = \{1, 2\}$. Para la versión restringida RP_3, tenemos que la solución óptima es $x = (2, 1, 0, 0, 0)^T$ con valor óptimo $z_{RP3} = 0$. Se termina; la solución óptima correspondiente al primal es $x^* = (2, 1, 0)^T$.

3. Programación Entera

Las principales fuentes consultadas para este capítulo han sido [8] y [9].

Hasta ahora hemos visto modelos de programación lineal donde las variables podían tomar cualquier valor real.
Vamos a considerar ahora, el problema de la forma:

$$\begin{array}{ll} \text{minimizar} & c^T x \\ \text{sujeto a} & Ax = b \\ & x \geq 0 \\ & x \in \mathbb{Z}^n \end{array} \tag{3.1}$$

donde $c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ y $A \in M_{m\times n}(\mathbb{R})$ de rango m, con $m \leq n$.
A diferencia de un problema de programación lineal, las variables se restringen a valores enteros. Este tipo de problemas se conocen como *problemas de programación (lineal) entera*. Cuando algunas variables están restringidas a valores enteros y otras no, hablamos de problemas de *programación entera mixta.*

Observación 3.1. La región factible de un problema de programación entera, es no convexa.

Observación 3.2. Se podría pensar en resolver el problema lineal asociado resultante de relajar/eliminar la condición de que las variables sean enteras, para posteriormente redondear la solución obtenida. Pero esta técnica rara vez ofrece buenos resultados.

Ilustramos la observación en el siguiente ejemplo:

Ejemplo 3.3.

$$\begin{array}{ll} \text{minimizar} & -5x_1 - 8x_2 \\ \text{sujeto a} & x_1 + x_2 \leq 6 \\ & 5x_1 + 9x_2 \leq 45 \\ & x \geq 0 \\ & x \in \mathbb{Z}^2 \end{array}$$

La solución óptima del problema relajado es $x_r^* = (2.25, 3.75)$ con valor objetivo $z_r = -41.25$. La solución óptima entera es $x^* = (0, 5)$, con valor objetivo $z = -40$. En este caso, si intentásemos buscar una solución entera a partir de la solución óptima x_r^* podríamos probar de redondear la solución, obteniendo así el punto $(2, 4)$, que no es factible pues $2 \cdot 5 + 9 \cdot 4 = 46 > 45$.

3.1. El método de Branch and Bound

El método *Branch and Bound* o *Ramificación y Acotación*, consiste en subdividir sucesivamente la región factible del problema de partida y resolver las versiones relajadas de los problemas resultantes, hasta encontrar la solución óptima del problema. La subdivisión se realiza de forma que sigue una estructura de *árbol binario*[18].

[18] De la teoría de grafos, sabemos que: *Un árbol binario es un árbol no dirigido tal que el grado de cada vértice es menor o igual a* 3. Intuitivamente, es una estructura de datos formada por un nodo inicial, y en la cual cada nodo puede tener hasta "dos hijos", que pasarían a ser dos nodos más.

3.1.1. Descripción del método Branch and Bound

Consideremos el siguiente problema de programación entera mixta:

$$\begin{array}{ll} \text{minimizar} & c^T x + d^T y \\ \text{sujeto a} & Ax + Gy = b \\ & x, y \geq 0 \\ & x \in \mathbb{Z}^n \end{array} \tag{3.2}$$

donde $d, y \in \mathbb{R}^p$, $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$, $A \in M_{m\times n}(\mathbb{R})$ y $G \in M_{m\times p}(\mathbb{R})$ ambas de rango m, con $m \leq n$. Para facilitar la exposición, vamos a denotar el problema como MILP (*Mixed Integer Linear Program*),

$$\text{MILP}: \quad \text{mín}\left\{c^T x + d^T y \mid (x, y) \in S\right\}$$

con $S = \left\{(x, y) \in \mathbb{Z}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy = b\right\}$.
Supongamos que el problema MILP dado admite la solución óptima (x^*, y^*) con valor óptimo z^*.
Consideremos entonces la relajación del problema MILP, es decir

$$\text{LP}_0: \quad \text{mín}\left\{c^T x + d^T y \mid (x, y) \in P_0\right\}$$

con $P_0 = \left\{(x, y) \in \mathbb{R}^n_+ \times \mathbb{R}^p_+ \mid Ax + Gy = b\right\}$ y supongamos que LP_0 admite una solución óptima (x, y) con valor óptimo z_0.
Como $S \subseteq P_0$, inmediatamente tenemos $z* \geq z_0$. Además, si x es un vector entero, entonces $(x, y) \in S$ y por tanto $z^* = z_0$; en este caso quedaría resuelto el MILP.
De no ser así, y siendo $x^T = (x_1, \cdots, x_n)$ al menos una de las variables x_j con $j \in \{1, \cdots, n\}$ sería no entera. Seleccionamos $x_j = f$ la variable no entera con j mínimo de mejor potencial[19]. Definimos entonces los conjuntos

$$S_1 := S \cap \{(x, y) \mid x_j \leq \lfloor f \rfloor\}, \quad S_2 := S \cap \{(x, y) \mid x_j \geq \lceil f \rceil\}$$

donde $\lfloor f \rfloor$ denota al mayor entero menor o igual que f y $\lceil f \rceil$ denota al menor entero mayor o igual que f. Tenemos que (S_1, S_2) es una partición de S.
Consideramos ahora los dos siguientes problemas enteros mixtos basados en esta partición,

$$\text{MILP}_1: \quad \text{mín}\left\{c^T x + d^T y \mid (x, y) \in S_1\right\}, \quad \text{MILP}_2: \quad \text{mín}\left\{c^T x + d^T y \mid (x, y) \in S_2\right\}.$$

La solución original del problema se ha reducido a resolver estos dos nuevos subproblemas. Denotamos por P_1, P_2 las relajaciones resultantes de S_1 y S_2 respectivamente. Esto es

$$P_1 := P_0 \cap \{(x, y) \mid x_j \leq \lfloor f \rfloor\}, \quad P_2 := P_0 \cap \{(x, y) \mid x_j \geq \lceil f \rceil\},$$

y consideramos,

$$\text{LP}_1: \quad \text{mín}\left\{c^T x + d^T y \mid (x, y) \in P_1\right\}, \quad \text{LP}_2: \quad \text{mín}\left\{c^T x + d^T y \mid (x, y) \in P_2\right\}.$$

(i) Si uno de los LP_i es infactible, i.e., $P_i = \emptyset$, entonces también tenemos $S_i = \emptyset$ ya que $S_i \subseteq P_i$. Y por tanto como MILP_i es infactible no lo necesitamos considerar más. Decimos que el problema *queda podado por infactibilidad*.

[19] Cconsideramos que x_j tiene mejor potencial, si se cumple que $c_j f$ es mínimo)

(ii) Sea (x^i, y^i) una solución óptima de LP_i con valor óptimo z_i, para $i = 1, 2$.

(iia) Si x^i es un vector entero, entonces (x^i, y^i) es una solución óptima del problema MILP_i y una solución factible del problema MILP. El problema MILP_i queda solucionado, y decimos que *queda podado por optimalidad*. Como $S_i \subseteq S$, le sigue que $z_i \geq z^*$, esto es, z_i es una cota superior del valor de MILP.

(iib) Si x^i no es un vector entero, y z_i es mayor o igual que el valor de la mejor cota superior conocida del MILP, entonces S_i no puede contener una solución mejor y el problema *queda podado por acotación*.

(iic) Si x^i no es un vector entero y z_i es menor que el valor de la mejor cota superior conocida, entonces S_i aun puede contener una mejor solución óptima para el MILP. Sea x_{ij}, la componente no entera de la solución x_i tal que $c_j x_{ij}$ es mínimo. Se define $f_i := x_{ij}$ y los conjuntos $S_{i_1} := S_i \cap \{(x, y) \mid x_{ij} \leq \lfloor f_i \rfloor\}$, $S_{i_2} := S_2 \cap \{(x, y) \mid x_{ij} \geq \lceil f_i \rceil\}$ y se repite el proceso.

3.1.2. Esquema general del método Branch and Bound

El metodo de *Branch and Bound* que presentamos a continuación es para problemas de minimización con región factible no vacía y acotada[20].
Cada Programa Lineal relajado, corresponde con un nodo del árbol de binario. Para un nodo N_{i_j} [21], se le asocia el valor óptimo z_{i_j} del programa lineal LP_{i_j}.
Denotaremos como $\mathcal{L}$ la lista de nodos que deban resolverse.

Algoritmo (Branch and Bound).

Paso 0. INICIALIZACIÓN.
Definimos $\mathcal{L} := \{N_0\}$, la cota superior $UB := +\infty$

Paso 1. PRINCIPAL.

- *Selecciona un nodo N_{i_j} en $\mathcal{L}$ y eliminarlo ($\mathcal{L} \to \mathcal{L} \setminus N_{i_j}$).*
- *Resolver LP_{i_j}. Si es infactible ir al Paso 2. De otro modo, sea (x^i, y^i) la solución óptima de LP_{i_j} con valor óptimo z_{i_j}.*
- *Si x no es entera y $UB \leq z_{i_j}$ descartamos nodo por acotación. Ir al Paso 2.*
- *Si (x^i, y^i) es factible para MILP y $z_{i_j} < UB$ definimos:*

$$UB := z_{i_j}$$

$$(x^*, y^*) := (x^i, y^i).$$

Ir al Paso 2.

- *Desde LP_i se construyen los dos problemas lineales LP_{k_j}, $LP_{k_{j+1}}$ y se añaden a $\mathcal{L}$ los nodos N_{k_j} y $N_{k_{j+1}}$ correspondientes, donde $k = i + 1$. Ir al paso 1.*

[20] En caso de no tener región factible no vacía y acotada, el método puede no converger. Esto se puede ver en el ejemplo (3.8).

[21] El índice i representa cada *nivel* del árbol binario, mientras j representa la posición. Para cada *nivel* $i \geq 0$, pueden haber 2^i nodos distintos.

Paso 2. CONTROL

- *Si $\mathcal{L} \neq \emptyset$ ir al Paso 1.*
- *Si $\mathcal{L} = \emptyset$, la solución (x^*, y^*) es óptima. Se termina.*

Observación 3.4. En el *Paso 1.* la selección del nodo N_i queda abierta. Lo mas recomendado es tomar aquel con mayor potencial, es decir aquel donde el valor óptimo del nodo z_i sea mínimo.

3.1.3. Convergencia del método Branch and Bound

Teorema 3.5. *(Convergencia del Algorimo). Sea MILP un problema de programación entera mixta en donde la región factible es no vacía y acotada, entonces el algoritmo de Branch and Bound descrito anteriormente encuentra la solución óptima en una cantidad finita de pasos.*

Dem.: Presentamos la idea de la demostración.
Como la región factible es acotada, el número total de ramificaciones es finito. Esto, unido a que el algoritmo no repite dos veces la misma ramificación, asegura la convergencia finita del mismo (se puede ramificar varias veces sobre la misma variable, pero nunca será una ramificación ya hecha).
Por otro lado, para ver que la convergencia se produce a un óptimo del problema entero, es importante notar que los pasos de ramificación dividen la región factible del subproblema actual de tal manera que solo se dejan fuera soluciones no enteras: x_i en el intervalo $(\lfloor xi \rfloor, \lceil x_i \rceil)$. Por tanto nunca dejamos fuera soluciones factibles del MILP original. El algoritmo termina cuando todo subproblema N_{i_j} creados en el transcurso del mismo está en alguna de las siguientes situaciones:

(I) El subproblema LP_{i_j} asociado a N_{i_j} ha sido resuelto y se ha encontrado el óptimo entero del mismo.

(II) El potencial z_{i_j} del subproblema asociado a N_{i_j} es tal que $UB < z_{i_j}$ con lo que ninguna solución factible de N_{i_j} puede mejorar el valor UB obtenido.

Esto asegura que, entre las regiones factibles de todos los subproblemas explorados, no hay ninguna solución que pueda mejorar UB. Lo cual, unido a que ninguna solución factible del problema entero queda fuera de la unión de las regiones factibles de estos subproblemas, asegura la optimalidad de UB. □

Observación 3.6. A pesar de tener garantizada la convergencia finita del algoritmo, se trata de un algoritmo exponencial, y eso puede elevar considerablemente el tiempo de resolución.
Si consideramos LB la cota mínima encontrada en un problema relajado, podemos tener controlado en todo momento el porcentaje máximo de desviación con respecto al óptimo mediante el cociente $(UB - LB)/LB$.

Veamos a continuación como se resolvería el problema del ejemplo (3.3) con el algoritmo de *Branch and Bound*.

Ejemplo 3.7.

$$\begin{array}{ll} \text{minimizar} & -5x_1 - 8x_2 \\ \text{sujeto a} & x_1 + x_2 \leq 6 \\ & 5x_1 + 9x_2 \leq 45 \\ & x \geq 0 \\ & x \in \mathbb{Z}^2 \end{array}$$

Paso 0. Inicialización.
Definimos $\mathcal{L} = \{N_0\}$, $LB = -\infty$, $UB = +\infty$.

Iteración 1 *Paso 1*. Principal.

- $\mathcal{L} \to \mathcal{L} \setminus N_0$ ($\mathcal{L} = \emptyset$).
- Al resolver LP_0, obtenemos $(x_1, x_2) = (2.25, 3.75)^T$ y $z_0 = -41.25 < UB = +\infty$.
- Como $c_2x_2 = -30 < c_1x_1 = -11.25$, construimos los dos problemas correspondientes LP_{1_1} y LP_{1_2} añadiendo las restricciones $x_2 \leq 3$ y $x_2 \geq 4$ respectivamente. Actualizamos $\mathcal{L} = \{N_{1_1}, N_{1_2}\}$.

Iteración 2 *Paso 1*. Principal. Como el potencial de N_{1_1} y N_{1_2} es el mismo (-41.11). Escogemos N_{1_1} por ser el de índice más bajo.

- $\mathcal{L} \to \mathcal{L} \setminus N_{1_1}$ ($\mathcal{L} = \{N_{1_2}\}$).
- Al resolver LP_{1_1}, obtenemos $(x_1, x_2) = (2, 3)^T$ y $z_{1_1} = -34 < UB = +\infty$. Actualizamos $UB = -34$.

Paso 2. Control.

- como $\mathcal{L} \neq \emptyset$, volvemos al Paso 1.

Iteración 3 *Paso 1*. Principal.

- $\mathcal{L} \to \mathcal{L} \setminus N_{1_2}$ ($\mathcal{L} = \emptyset$).
- Al resolver LP_{1_2}, obtenemos $(x_1, x_2) = (1.8, 4)^T$ y $z_{1_2} = -41 < UB = -34$.
- construimos los dos problemas correspondientes LP_{2_3} y LP_{2_4} añadiendo las restricciones $x_1 \leq 1$ y $x_1 \geq 2$ respectivamente.. Actualizamos $\mathcal{L} = \{N_{2_3}, N_{2_4}\}$.

Iteración 4 *Paso 1*. Principal. Como el potencial de N_{2_3} y N_{2_4} es el mismo (-41). Escogemos N_{2_3} por ser el de índice más bajo.

- $\mathcal{L} \to \mathcal{L} \setminus N_{2_3}$ ($\mathcal{L} = \{N_{2_4}\}$).
- Al resolver LP_{2_3}, obtenemos $(x_1, x_2) = (1, 4.44)^T$ y $z_{2_3} = -40.56 < UB = -34$.

- construimos los dos problemas correspondientes LP_{3_5} y LP_{3_6} añadiendo las restricciones $x_2 \leq 4$ y $x_2 \geq 5$ respectivamente.. Actualizamos $\mathcal{L} = \{N_{2_4}, N_{3_5}, N_{3_6}\}$.

Iteración 5 *Paso 1*. Principal. Escogemos N_{2_4} por ser el nodo de mejor potencial (-41).

- $\mathcal{L} \to \mathcal{L} \setminus N_{2_4}$ $(\mathcal{L} = \{N_{3_5}, N_{3_6}\})$.
- Al resolver LP_{2_4}, obtenemos que el problema es infactible.

Paso 2. Control.

- como $\mathcal{L} \neq \emptyset$, volvemos al Paso 1.

Iteración 6 *Paso 1*. Principal. Como el potencial de N_{3_5} y N_{3_6} es el mismo (-40.56). Escogemos N_{3_5} por ser el de índice más bajo.

- $\mathcal{L} \to \mathcal{L} \setminus N_{3_5}$ $(\mathcal{L} = \{N_{3_6}\})$.
- Al resolver LP_{3_5}, obtenemos $(x_1, x_2) = (1, 4)^T$ y $z_{3_5} = -37 < UB = -34$. Actualizamos $UB = -34$.

Paso 2. Control.

- como $\mathcal{L} \neq \emptyset$, volvemos al Paso 1.

Iteración 7 *Paso 1*. Principal.

- $\mathcal{L} \to \mathcal{L} \setminus N_{3_6}$ $(\mathcal{L} = \emptyset)$.
- Al resolver LP_{3_6}, obtenemos $(x_1, x_2) = (0, 5)^T$ y $z_{3_6} = -40 < UB = -37$. Actualizamos $UB = -40$.

Paso 2. Control.

- como $\mathcal{L} = \emptyset$, la solución $(x_1^*, y_2^*) = (0, 5)^T$ es óptima. Fin

---

El siguiente diagrama muestra la evolución de los distintos nodos del algoritmo:

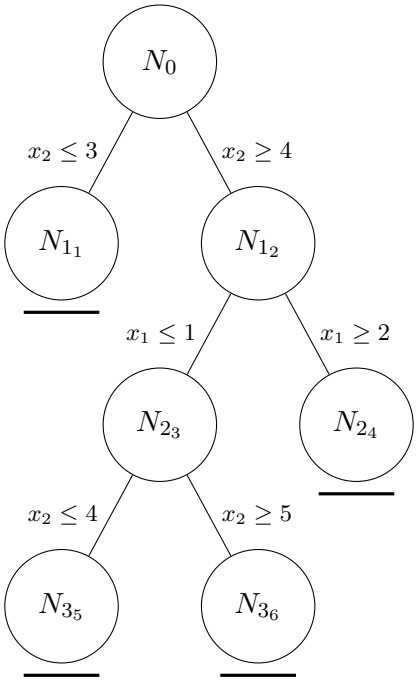

Figura 1: esquema del algoritmo Branch and Bound del ejemplo 3.7

Para finalizar esta sección vamos a ver un ejemplo donde la región factible del problema relajado no cumple las condiciones del teorema (3.5) y entonces el algoritmo no converge.

Ejemplo 3.8. Consideremos el siguiente problema de programación entera.

$$\begin{array}{lrl} \text{minimizar} & x_1 + x_2 & \\ \text{sujeto a} & 3x_1 - 3x_2 & \leq 2 \\ & 3x_1 - 3x_2 & \geq 1 \\ & x & \geq 0 \\ & & \in \mathbb{Z}^2 \end{array} \tag{3.3}$$

Si aplicamos el algoritmo, obtenemos que la solución del problema relajado LP_0, es $(x_1, x_2) = (\frac{1}{3}, 0)^T$ con $z_0 = \frac{1}{3}$. Posteriormente se derivan los problemas LP_{1_1} y LP_{1_2} resultantes de añadir las restricciones $x_1 \leq 0$ y $x_1 \geq 1$ respectivamente.

- LP_{1_1} es infactible por ser $x_1 \leq -1$.
- Para LP_{1_2}, la solución óptima del problema es $(x_1, x_2) = (1, \frac{1}{3})^T$ con $z_{1_2} = \frac{4}{3}$.

Si seguimos iterando, se derivan los problemas LP_{2_3} y LP_{2_4} resultantes de añadir las restricciones $x_2 \leq 0$ y $x_2 \geq 1$ respectivamente.

- LP_{2_3} es infactible.
- Para LP_{2_4},la solución óptima del problema es $(x_1, x_2) = (\frac{4}{3}, 1)^T$ con $z_{2_4} = \frac{7}{3}$.

A medida que vayamos iterando, iremos obteniendo soluciones de la forma $(n + \frac{1}{3}, n)$ y $(n+1, n+\frac{1}{3})$ para los problemas relajados, con valor creciente para la función objetivo, pero sin que el algoritmo pueda identificar que la región factible es vacía; consecuentemente podríamos seguir iterando el algoritmo indefinidamente.

4. Problemas aplicados

Las principales fuentes consultadas para este capítulo han sido [8] y [9].

En esta sección vamos a mostrar algunos ejemplos de problemas clásicos cuya modelización matemática dan pie a la formulación de problemas de programación lineal y programación entera.

4.1. Ejemplos de problemas de programación lineal

4.1.1. Problema de la dieta

Una de las primeras aplicaciones conocidas del algoritmo del Simplex fue la determinación de una dieta adecuada que fuera de menor coste. El problema de la dieta consiste en determinar la dieta más económica que satisfaga las necesidades nutritivas mínimas básicas para una persona o un grupo de personas.

Ejemplo 4.1. Supongamos que disponemos de n alimentos distintos con un precio c_i la unidad del alimento i-ésimo. Además hay m ingredientes nutritivos básicos y, para conseguir una dieta equilibrada, cada persona debe recibir al menos b_j unidades del j-ésimo elemento nutritivo por día. Finalmente, se supone que cada unidad del alimento i contiene a_{ji} unidades del j-ésimo elemento nutritivo. Entonces, denominamos x_i al número de unidades del alimento i de la dieta. El problema consistirá en seleccionar las x_i para minimizar el coste total. Esto es:

si,
a_{ji} = Cantidad del nutriente j en el alimento i, para $j \in \{1, \dots, m\}$, $i \in \{1, \dots, n\}$
b_j = Cantidad necesaria del nutriente j, para $j \in \{1, \dots, m\}$
c_i = Coste unitario del alimento i, para $i \in \{1, \dots, n\}$
x_i = número de unidades del alimento i, para $i \in \{1, \dots, n\}$.

Entonces el problema a resolver se corresponde con el programa lineal,

$$\begin{array}{ll} \text{minimizar} & c_1x_1 + c_2x_2 + \cdots + c_nx_n \\ \text{sujeto a} & a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \geq b_1 \\ & a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \geq b_2 \\ & \vdots \qquad\qquad\qquad\qquad \vdots \\ & a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \geq b_m \\ \text{y} & x_i \geq 0, \;\; i \in \{1, \dots, n\}. \end{array}$$

El problema de la dieta tiene su particular interpretación en el dual. Si consideramos el problema que acabamos de describir como el primal, es decir con las restricciones $Ax \geq b$ y $x \geq 0$, el dual de este problema sería:

$$\begin{array}{ll} \text{máx} & \lambda^T b \\ \text{sujeto a} & \lambda^T A \leq c^T \\ & \lambda \geq 0. \end{array}$$

Este problema se corresponde por ejemplo, en el caso en que un farmacéutico que dispone de m complementos vitamínicos distintos y una cantidad b_j de cada complemento.

El precio unitario del nutriente j es λ_j. El farmacéutico quiere maximizar los beneficios asociados a la venta de complementos vitamínicos, pero debe asegurarse que los precios sean competitivos con el de los alimentos que contienen al nutriente. Es decir, para cada alimento i, el precio en complementos vitamínicos de sus nutrientes, $\sum_{j\in\{1,\dots,m\}}\lambda_j a_{ji}$ no puede ser superior al coste del alimento, c_i.

Observación 4.2. Además podríamos dar una interpretación del *teorema de holguras complementarias*. Supongamos por ejemplo que tenemos un par de óptimos x, λ^T del primal y dual respectivamente; entonces, si en el primal hay un nutriente que se produce por encima de su nivel mínimo ($a^{(j)}x > b_j$), entonces el precio del complemento correspondiente en el dual será 0. Se podría interpretar como que ese nutriente se consigue "gratis". De modo similar, si un complemento vitamínico tiene un precio > 0, la restricción en el primal se verificará con igualdad ya que no puede ser que haya un exceso de nutriente en el primal (el dual nos dice que "no es gratis").
Por otro lado, si compramos un alimento en el primal, no puede ser que en el dual lo "produzcamos más baratoçon los complementos; así pues, si en el dual replicamosün alimento por debajo de su coste primal, entonces en el primal no será óptimo comprarlo.

4.2. Relaciones lógicas con variables binarias

Antes de presentar alguno de los problemas clásicos de la programación entera vamos a introducir el uso de variables binarias, a fin de modelar muchos tipos de relaciones lógicas entre variables y restricciones.

4.2.1. Producción acotada

Supongamos que tenemos que decidir la cantidad a fabricar de un producto x_j. Podemos decidir si fabricarlo o no. En caso de hacerlo queremos que la cantidad este entre l_j y u_j, con $0 < l_j < u_j$. Una manera de modelar esto sería considerando la variable binaria $y_j \in \{0, 1\}$ tal que:

$$y_j = \begin{cases} 1 & \text{si se fabrica el producto } j \\ 0 & \text{si no se fabrica el producto } j. \end{cases}$$

Las restricciones de producción vendrán dadas por

$$l_j y_j \leq x_j$$

$$x_j \leq u_j y_j.$$

Entonces si $y_j = 0$, $x_j = 0$. Si $y_j = 1$ se cumple $l_j \leq x_j \leq u_j$.

4.2.2. Costes fijos

Supongamos que tenemos una actividad x_j que podemos realizar o no y que en caso de realizar-la tiene un coste fijo f_j y un coste variable c_j. Por lo tanto el coste asociado a x_j es 0 si $x_j = 0$ y $f_j + c_j x_j$ si $x_j > 0$. De nuevo una manera de modelar esto seria considerando la variable binaria $y_j \in \{0, 1\}$ tal que:

$$y_j = \begin{cases} 1 & \text{si se realiza la actividad } j \\ 0 & \text{si no se realiza la actividad } j. \end{cases}$$

Entonces el sumando correspondiente de la función objetivo sería $f_j y_j + c_j x_j$.

Observación 4.3. Hay que evitar soluciones en las que $y_j = 0$ y $x_j > 0$. Para eso se puede añadir la restricción $x_j \leq M y_j$, para M suficientemente grande. De esta forma nos aseguramos que si $y_j = 1$, x_j no tenga ninguna restricción adicional, y que si $y_j = 0$, entonces se impondrá $x_j = 0$

4.2.3. Variables con una cantidad finita de valores

Supongamos que tenemos una variable x_j que puede tomar una cantidad finita de valores $\{v_1, \cdots, v_l\}$. Entonces consideramos y_{ij} variables binarias, con $i \in \{1, \cdots, l\}$ tales que

$$y_{ij} = \begin{cases} 1 & \text{si } x_j = v_i \\ 0 & \text{si } x_j \neq v_i. \end{cases}$$

como x_j sólo puede tomar uno de estos valores tendremos

$$\sum_{i=1}^{l} y_{ij} = 1.$$

Por último, la variable x_j vendrá dada por

$$x_j = \sum_{i=1}^{l} v_i y_{ij}$$

4.2.4. Restricciones sustitutas

Supongamos ahora que tenemos un problema en el que ha de cumplirse al menos una de las dos siguientes restricciones

$$\begin{array}{rcl} \sum_{j=1}^{n} a_{1j} x_j & \leq & b_1 \\ \sum_{j=1}^{n} a_{2j} x_j & \leq & b_2 \end{array}$$

Esto se puede modelar a través de una variable binaria y tal que

$$y = \begin{cases} 0 & \text{si imponemos que se cumpla la primera restricción} \\ 1 & \text{si imponemos que se cumpla la segunda restricción,} \end{cases}$$

y se introducen en el problema las siguientes dos restricciones:

$$\begin{array}{rcl} \sum_{j=1}^{n} a_{1j} x_j & \leq & b_1 + My \\ \sum_{j=1}^{n} a_{2j} x_j & \leq & b_2 + M(1-y), \end{array}$$

donde $M > 0$ es una constante lo suficientemente grande.

Veamos ahora el caso que dadas m restricciones queremos que se cumplan al menos k. La formulación pasaría por introducir m variables binarias y_i, donde

$$y_i = \begin{cases} 0 & \text{si imponemos que se cumpla la i-ésima restricción} \\ 1 & \text{si imponemos que se cumpla la i-ésima restricción,} \end{cases}$$

En el problema habría que introducir las siguientes m restricciones

$$\begin{array}{rcl} \sum_{j=1}^{n} a_{1j}x_j & \leq & b_1 + My_1 \\ \sum_{j=1}^{n} a_{2j}x_j & \leq & b_2 + My_2 \\ \vdots & \vdots & \vdots \\ \sum_{j=1}^{n} a_{mj}x_j & \leq & b_m + My_m, \end{array}$$

acompañada de la restricción

$$\sum_{i=1}^{m} y_i = m - k.$$

4.3. Ejemplos de problemas de programación Entera

4.3.1. El problema de la mochila

El problema de la mochila se puede formular como sigue:
Un excursionista debe decidir que cosas incluir en su mochila. Tiene n objetos entre los que elegir y cada objeto tiene un peso p_j y una utilidad u_j. El excursionista quiere maximizar la utilidad de la mochila que se encuentras sujeta a una restricción, el peso de la mochila $P \geq 0$. Se expresa el problema, de la forma que sigue:

$$\begin{array}{rll} \text{maximizar} & \displaystyle\sum_{j=1}^{n} u_j x_j & \\ \text{sujeto a} & \displaystyle\sum_{j=1}^{n} p_j x_j & \leq \ P \\ & & x \in \{0,1\}^n \end{array}$$

x_j es una variable binaria que indica si el objeto entra en la mochila o no.

Observación 4.4. Existen problemas relevantes de programación entera que resultan ser (casi) equivalente al problema de la mochila, como por ejemplo *el problema de la suma de subconjuntos*. Dicho problema plantea la siguiente cuestión:

Dado un conjunto de números $S = \{a_1, a_2, \ldots, a_n\}$ y otro número b, ¿existe algún subconjunto de S cuyos números sumen exactamente b? Por ejemplo, si $S = \{2, 5, 16, 4, 7, 28\}$ y $b = 29$, la respuesta és afirmativa puesto que $16 + 7 + 4 + 2 = 29$. La formulación del problema es como sigue:

$$\begin{array}{rll} \text{maximizar} & \displaystyle\sum_{j=1}^{n} a_j x_j & \\ \text{sujeto a} & \displaystyle\sum_{j=1}^{n} a_j x_j & \leq \ b \\ & & x \in \{0,1\}^n \end{array}$$

Por tanto la equivalencia es casi inmediata[22], y existirá dicho subconjunto de S cuyos elementos sumen b si y sólo si la solución óptima del problema toma el valor b.

[22]Es suficiente considerar que los valores de los números representan tanto su peso como su utilidad y que la mochila admite un peso b.

4.3.2. Problemas de localización

Uno de los problemas clásicos de la programación entera, es el problema de localización. Su desarrollo a lo largo del tiempo ha sido tal, que existen múltiples versiones de este problema. Presentamos a continuación un ejemplo.

Ejemplo 4.5. Supongamos que disponemos de una empresa con m potenciales ubicaciones distintas en donde situar un almacén de distribución para atender las demandas de n clientes. El problema que se plantea, es decidir qué almacenes deberían abrirse y qué cantidad de mercancías se deben enviar desde cada almacén a cada cliente.

Usando las variables binarias vistas en el apartado 4.2.2, tenemos:

$$y_j = \begin{cases} 1 & \text{se abre el almacen de distribución } j \\ 0 & \text{si no se abre el almacen de distribución } j. \end{cases}$$

Por otro lado x_{ij} denota la cantidad de mercancías enviadas del almacén i al cliente j, y c_{ij} es el coste asociado a enviar una unidad de mercancía de i a j, y f_i, el coste fijo por abrir el almacén i.

Además, se debe tener en cuenta, que para cada cliente j, su demanda d_j debe ser realizada y no se pueden hacer envíos desde almacenes no abiertos. Así pues, con todo lo descrito anteriormente, la formulación del problema es la siguiente:

$$\begin{aligned}
\text{minimizar} \quad & \sum_{i=1}^{m}\sum_{j=1}^{n} c_{ij}x_{ij} + \sum_{i=1}^{m} f_i y_i \\
\text{sujeto a} \quad & \sum_{i=1}^{m} x_{ij} = d_j \qquad j \in \{1,\dots,n\} \\
& \sum_{j=1}^{n} x_{ij} - y_i\left(\sum_{j=1}^{n} d_j\right) \leq 0 \qquad i \in \{1,\dots,m\} \\
& x_{ij} \geq 0 \qquad j \in \{1,\dots,m\}, \quad j \in \{1,\dots,n\} \\
& y_i \in \{0,1\} \qquad i \in \{1,\dots,m\}
\end{aligned}$$

La función objetivo a minimizar contempla tanto los costes fijos asociados a los almacenes abiertos como los costes de transporte entre almacén y cliente. Con la restricción $\sum_{i=0}^{m} x_{ij} = d_j$, se impone la demanda del cliente d_j. Veamos ahora que pasa con las variables vinarias:

- Si $y_i = 0$, el almacén i no abre, y entonces $\sum_{j=1}^{n} x_{ij} \leq 0$, no se envía nada desde el almacén i.
- Si $y_i = 1$, la restricción establece que desde el almacén i no se enviarán más que la suma de todas las demandas.

5. Conclusiones

Una de las motivaciones principales que me ha llevado a realizar este trabajo es la inmediata conexión entre lo práctico y teórico.

La profunda comprensión de la naturaleza algebraica del problema que plantea la programación lineal, ha permitido idear métodos para una resolución muy eficiente en general. El método del símplex es un claro ejemplo de ello, y que además a día de hoy sigue siendo el gran referente para abordar este tipo de problemas no solo por su gran utilidad sino por que también proporciona mucha información adicional relativa a la solución encontrada. Con el método del símplex se ha podido demostrar el teorema de dualidad fuerte, que nos ha servido en gran medida para demostrar el teorema de las holguras complementarias, y para el análisis de sensibilidad con el cual se ha visto como pueden afectar las distintas variaciones de un problema de programación lineal.
La importancia de disponer de un método como el del símplex, también se ha visto reflejada en la programación entera, ya que como hemos podido comprobar el método de Branch and Bound resuelve versiones relajadas del problema inicial.

Este trabajo, de manera introductoria, alcanza el objetivo de establecer una base rigurosa de conocimientos sobre la programación lineal y entera a partir de la cual poder seguir profundizando a nivel teórico o desarrollar y implementar en la práctica.

Referencias

[1] Bazaraa, M., Jarvis, J. and Sherali, H., 2013. *Linear Programming And Network Flows*. Hoboken: Wiley.

[2] Luenberger, D. and López Mateos, M., 1989. *Programación lineal y no lineal*. Wilmington, Del.: Addison-Wesley Iberoamericana.

[3] Dantzig, G. and Thapa, M., 1997. *Linear Programming, 1:Introduction*. New York: Springer.

[4] Fuente O'Connor, J., 1998. *Técnicas De Cálculo Para Sistemas De Ecuaciones, Programación Lineal Y Programación Entera*. 2nd ed. Barcelona: Reverté.

[5] Basart i Muñoz, J., 1998. *Programació Lineal*. Bellaterra: Universitat Autònoma de Barcelona, Servei de Publicacions.

[6] Pan, P., 2014. *Linear Programming Computation*. Berlin: Springer.

[7] Matousek, J. and Gärtner, B., 2007. *Understanding And Using Linear Programming*. Berlin: Springer.

[8] Conforti, M., Cornuejols, G. and Zambelli, G., 2014. *Integer Programming*. Springer.

[9] Bradley, S., Hax, A. and Magnanti, T., 1992. *Applied Mathematical Programming*. Reading, Mass: Addison-Wesley.