Capítulo 13

Algunos trucos útiles

13.1 Introducción

Este capítulo está organizado en dos partes. La primera presenta algunos trucos genéricos que son útiles para problemas de programación lineal y lineal entera-mixta. En la segunda parte se sugieren algunos trucos útiles para la formulación de problemas en GAMS.

13.2 Algunos trucos genéricos

En este apartado se presentan algunos trucos que facilitan, por un lado, la formulación y resolución de gran variedad de problemas, y por otro, permiten plantear tanto problemas no lineales como problemas lineales. En particular, se describen trucos útiles para los siguientes casos:

1. Tratamiento de variables no acotadas que por necesidades de programación deben formularse como no negativas
2. Conversión de un conjunto de restricciones lineales de desigualdad en un conjunto equivalente de restricciones lineales de igualdad
3. Conversión de un conjunto de restricciones lineales de igualdad en un conjunto equivalente de restricciones lineales de desigualdad
4. Conversión de un problema de maximización en un problema de minimización
5. Conversión de una función objetivo no lineal en su equivalente lineal
6. Tratamiento de algunas funciones no lineales como lineales
7. Tratamiento de un espacio lineal como un cono
8. Tratamiento de conjuntos alternativos de restricciones

9. Tratamiento de restricciones condicionales
10. Tratamiento de funciones no continuas
11. Tratamiento de funciones no convexas a trozos

13.2.1 Tratamiento de variables no acotadas

Cualquier conjunto de variables no acotadas $\{x_1, x_2, \ldots, x_r\}$ se puede sustituir por otro conjunto formado por la diferencia entre dos variables no negativas

$$\{y_1 - z_1, y_2 - z_2, \ldots, y_r - z_r | y_1, y_2, \ldots, y_r \geq 0; z_1, z_2, \ldots, z_r \geq 0\}$$

Esta sustitución implica que se duplica el número de variables del conjunto inicial.

Para hacer la anterior transformación se considera que la variable x_i tiene una parte positiva y otra negativa, que se definen como

$$\begin{array}{rcrcl} y_i & = & x_i^+ & = & \max\{0, x_i\} \\ z_i & = & x_i^- & = & \max\{0, -x_i\} \end{array} \tag{13.1}$$

respectivamente. Así, resulta sencillo comprobar que $x = x_i^+ - x_i^-$, donde tanto x_i^+ como x_i^- son no negativos. Sin embargo, como también se puede escribir $x = (x_i^+ + k) - (x_i^- + k)$ para cualquier $k > 0$, se tiene un número infinito de descomposiciones.

Una mejora al planteamiento anterior consiste en sustituir el conjunto las r variables no acotadas $\{x_1, \ldots, x_r\}$ por el conjunto $r+1$ de variables no negativas $\{x_1^*, \ldots, x_r^*, x^*\}$, donde

$$x_i = x_i^* - x^*; \;\; i = 1, 2, \ldots, r \tag{13.2}$$

De esta forma se añade una sola variable en lugar de las r que se añaden con el primer método.

El lector puede comprobar sin dificultad que si se elige

$$x^* = -\min\{x_1, x_2, \ldots, x_r\}; \;\; x_i^* = x_i + x^*$$

entonces las variables $\{x_1^*, \ldots, x_r^*, x^*\}$ son no negativas. Nótese que si el conjunto $\{x_1^*, \ldots, x_r^*, x^*\}$ cumple la igualdad (13.2), entonces $\{x_1^*+k, \ldots, x_r^*+k, x^*+k\}$ también la cumple, para cualquier $k > 0$.

Ejemplo 13.1 (conversión a variables no negativas). Supóngase el siguiente conjunto de restricciones

$$\begin{array}{rrrcr} x_1 & +x_2 & +x_3 & = & 1 \\ x_1 & -x_2 & -x_3 & \leq & 1 \\ x_1 & & +x_3 & \geq & -1 \\ x_1 & & & \geq & 0 \end{array} \tag{13.3}$$

donde x_1 es la única variable no negativa. Si se necesita obtener un conjunto equivalente donde todas las variables sean no negativas, se tiene la siguiente transformación

$$\begin{array}{rcl} x_2 & = & y_2 - z_2 \\ x_3 & = & y_3 - z_3 \\ y_2, y_3, z_2, z_3 & \geq & 0 \end{array} \tag{13.4}$$

donde $y_2 = x_2^+$, $y_3 = x_3^+$, y $z_2 = x_2^-$, $z_3 = x_3^-$. Por tanto, el conjunto inicial de restricciones se sustituye por

$$\begin{array}{rrrrrcr} x_1 & +y_2 & -z_2 & +y_3 & -z_3 & = & 1 \\ x_1 & -y_2 & +z_2 & -y_3 & +z_3 & \leq & 1 \\ x_1 & & & +y_3 & -z_3 & \geq & -1 \\ x_1, & y_2, & z_2, & y_3, & z_3 & \geq & 0 \end{array} \tag{13.5}$$

■

Ejemplo 13.2 (conversión a variables no positivas: método alternativo). Una mejora respecto al método empleado en el ejemplo anterior consiste en realizar el siguiente cambio de variables:

$$\begin{array}{rcl} x_2 & = & y_2 - y \\ x_3 & = & y_3 - y \end{array} \tag{13.6}$$

y por tanto, el conjunto inicial de restricciones se sustituye por:

$$\begin{array}{rrrrcr} x_1 & +y_2 & +y_3 & -2y & = & 1 \\ x_1 & -y_2 & -y_3 & +2y & \leq & 1 \\ x_1 & & +y_3 & -y & \geq & -1 \\ x_1, & y_2, & y_3, & y, & \geq & 0 \end{array} \tag{13.7}$$

que contiene sólo una variable más que el conjunto (13.5). ■

13.2.2 Transformación de desigualdades en igualdades

Las restricciones de desigualdad pueden convertirse en restricciones de igualdad añadiendo lo que se conoce como *variables de holgura*:

- Si
$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \leq b_i$$
entonces existe una variable $x_{n+1} \geq 0$ de forma que
$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n + x_{n+1} = b_i$$
- Si
$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n \geq b_i$$
entonces existe una variable $x_{n+1} \geq 0$ de forma que
$$a_{i1}x_1 + a_{i2}x_2 + \cdots + a_{in}x_n - x_{n+1} = b_i$$

Ejemplo 13.3 (transformación de desigualdades en igualdades). Si se introducen las variables de holgura u_1 y u_2, entonces las restricciones de desigualdad del conjunto de restricciones (13.5) se transforman en el siguiente conjunto de restricciones de igualdad:

$$\begin{array}{rrrrrrrrr} x_1 & -y_2 & +z_2 & -y_3 & +z_3 & +u_1 & & = & 1 \\ x_1 & & & +y_3 & -z_3 & & -u_2 & = & -1 \end{array} \tag{13.8}$$

donde ahora se tiene que

$$x_1, y_2, y_3, z_2, z_3, u_1, u_2 \geq 0$$

■

13.2.3 Transformación de igualdades en desigualdades

Un conjunto de m igualdades siempre puede transformarse en un conjunto equivalente de $m+1$ desigualdades, como se demuestra en la siguiente proposición.

Proposición 13.1 (transformación de igualdades en desigualdades). *El conjunto de igualdades*

$$\mathbf{a}_i^T \mathbf{x} = b_i \tag{13.9}$$

es equivalente al conjunto de desigualdades

$$\begin{array}{rcl} \mathbf{a}_i^T \mathbf{x} & \leq & b_i,\ i = 1, \ldots, m \\ \left(\sum\limits_{i=1}^{m} \mathbf{a}_i^T \right) \mathbf{x} & \geq & \sum\limits_{i=1}^{m} b_i \end{array} \tag{13.10}$$

■

Demostración. Es claro que si se cumple (13.9) también se cumple (13.10). Inversamente, considérese que $k \in \{1, \ldots, m\}$. Entonces

$$\mathbf{a}_k^T \mathbf{x} = \left(\sum_{i=1}^{m} \mathbf{a}_i^T \right) \mathbf{x} - \left(\sum_{i \neq k} \mathbf{a}_i^T \right) \mathbf{x} \geq \sum_{i=1}^{m} b_i - \sum_{i \neq k} b_i = b_k$$

y teniendo en cuenta que $\mathbf{a}_k^T \mathbf{x} \leq b_k$, entonces

$$\mathbf{a}_k^T \mathbf{x} = b_k$$

■

Ejemplo 13.4 (transformación de igualdades en desigualdades). El conjunto de igualdades

$$\begin{array}{rrrrr} x_1 & +x_2 & +x_3 & = & 0 \\ x_1 & -x_2 & -x_3 & = & 2 \\ x_1 & & +x_3 & = & -1 \end{array}$$

es equivalente al conjunto de desigualdades

$$\begin{array}{rrrcr} x_1 & +x_2 & +x_3 & \leq & 0 \\ x_1 & -x_2 & -x_3 & \leq & 2 \\ x_1 & & +x_3 & \leq & -1 \\ 3x_1 & & +x_3 & \geq & 1 \end{array}$$

■

13.2.4 Transformación de maximización a minimización

Cualquier problema de maximización equivale a uno de minimización cuya función objetivo tiene signo contrario. Concretamente, maximizar

$$Z_{max} = \mathbf{c}^T\mathbf{x}$$

equivale a minimizar

$$Z_{min} = -\mathbf{c}^T\mathbf{x}$$

donde ambos problemas están sujetos al mismo conjunto de restricciones. Obsérvese que $Z_{max} = -Z_{min}$, aunque el valor óptimo de las variables de ambos problemas coincide.

13.2.5 Transformación de funciones no lineales en lineales

El problema de minimización

$$Z = f(\mathbf{x})$$

sujeto a

$$\begin{array}{rcl} \mathbf{h}(\mathbf{x}) & = & \mathbf{0} \\ \mathbf{g}(\mathbf{x}) & \leq & \mathbf{0} \end{array} \tag{13.11}$$

es equivalente al problema de minimización

$$Z = y$$

sujeto a

$$\begin{array}{rcl} \mathbf{h}(\mathbf{x}) & = & \mathbf{0} \\ \mathbf{g}(\mathbf{x}) & \leq & \mathbf{0} \\ f(\mathbf{x}) & \leq & y \end{array} \tag{13.12}$$

La demostración de esta equivalencia es sencilla.

Este truco resulta útil para aquellas técnicas de resolución que requieran una función objetivo lineal.

13.2.6 Tratamiento de funciones no lineales como lineales

Los PPL se restringen a problemas cuya función objetivo es lineal y también lo son sus restricciones. Cuando la función objetivo o alguna de las restricciones no es lineal, se tiene un PPNL. Sin embargo, para ciertos casos, es posible transformar un PPNL en un PPL equivalente.

Tratamiento de valores absolutos

La función valor absoluto es una función que se utiliza con cierta frecuencia en problemas de optimización. Esta función, además de ser no lineal, es una función no diferenciable.

El PPNL que minimiza

$$Z = |\mathbf{c}^T\mathbf{x}|$$

sujeto a

$$\mathbf{Ax} = \mathbf{b}$$

equivale al PPL que minimiza

$$Z = y$$

sujeto a

$$\begin{array}{rcl} \mathbf{Ax} & = & \mathbf{b} \\ \mathbf{c}^T\mathbf{x} & \leq & y \\ -\mathbf{c}^T\mathbf{x} & \leq & y \\ y & \geq & 0 \end{array}$$

Genéricamente, el PPNL que minimiza

$$Z = \sum_{i=1}^{m} |\mathbf{c}_i^T\mathbf{x}|$$

sujeto a

$$\mathbf{Ax} = \mathbf{b}$$

equivale al PPL que minimiza

$$Z = \sum_{i=1}^{m} y_i$$

sujeto a

$$\begin{array}{rcll} \mathbf{Ax} & = & \mathbf{b} & \\ \mathbf{c}_i^T\mathbf{x} & \leq & y_i, & i = 1, 2, \ldots, m \\ -\mathbf{c}_i^T\mathbf{x} & \leq & y_i, & i = 1, 2, \ldots, m \\ y_i & \geq & 0, & i = 1, 2, \ldots, m \end{array}$$

Tratamiento de la función máximo

Otra función de interés es la función máximo que también es no lineal y no diferenciable. A continuación, se presenta un truco para el tratamiento de esta función.

El PPNL que minimiza

$$Z = \max_{i=1,2,\ldots,m} |\mathbf{c}_i^T\mathbf{x}|$$

sujeto a

$$\mathbf{Ax} = \mathbf{b}$$

equivale al PPL que minimiza

$$Z = y$$

sujeto a

$$\begin{array}{rcll} \mathbf{Ax} & = & \mathbf{b} & \\ \mathbf{c}_i^T\mathbf{x} & \leq & y, & i = 1, 2, \ldots, m \\ -\mathbf{c}_i^T\mathbf{x} & \leq & y, & i = 1, 2, \ldots, m \\ y & \geq & 0 & \end{array}$$

El Problema de la programación lineal fraccional

El *problema de la programación fraccional* consiste en optimizar el cociente de dos funciones lineales sujeto a un conjunto de restricciones también lineales. Este problema puede plantearse como sigue. Minimizar

$$\frac{\mathbf{p^T x} + \alpha}{\mathbf{q^T x} + \beta} \tag{13.13}$$

sujeto a

$$\begin{array}{rcl} \mathbf{Ax} & \leq & \mathbf{b} \\ \mathbf{x} & \geq & \mathbf{0} \end{array} \tag{13.14}$$

donde $\mathbf{p}$ y $\mathbf{q}$ son vectores de dimensión n, $\mathbf{b}$ es un vector de dimensión m, $\mathbf{A}$ es una matriz de dimensión $m \times n$, y α y β son escalares.

Este problema puede transformarse en un PPL mediante el siguiente teorema debido a Charnes y Cooper [25].

Teorema 13.1 (programación fraccional). *Si el conjunto factible*

$$\mathbf{X} = \{\mathbf{x} | \mathbf{Ax} \leq \mathbf{b} \ y \ \mathbf{x} \geq \mathbf{0}\}$$

es acotado y no vacío, y si $\mathbf{q}^T\mathbf{x} + \beta > 0$ *para cada* $\mathbf{x} \in \mathbf{X}$*, el problema (13.13)–(13.14) equivale al PPL: Minimizar*

$$\mathbf{p}^T\mathbf{y} + \alpha z \tag{13.15}$$

sujeto a

$$\begin{array}{rcll} \mathbf{Ay} - \mathbf{b}z & \leq & \mathbf{0} & (13.16) \\ \mathbf{q}^T\mathbf{y} + \beta z & = & 1 & (13.17) \\ \mathbf{y} & \geq & \mathbf{0} & (13.18) \\ z & \geq & 0 & (13.19) \end{array}$$

que tiene sólo una variable y una restricción adicionales. ∎

Este teorema se basa en la siguiente trasformación:

$$z = \frac{1}{\mathbf{q}^T\mathbf{x} + \beta} \quad \text{and} \quad \mathbf{y} = z\mathbf{x}, \tag{13.20}$$

Ejemplo 13.5 (programación fraccional). Supóngase el siguiente problema de programación lineal fraccional. Minimizar

$$\frac{x_1 + 1}{x_2 + 2}$$

sujeto a

$$\begin{aligned} x_1 + x_2 &\leq 1 \\ x_1, x_2 &\geq 0 \end{aligned}$$

Para transformar este problema de programación lineal fraccional en uno lineal, se realiza la transformación (13.20)

$$z = \frac{1}{x_2 + 1};\ \mathbf{y} = z\mathbf{x},$$

y se obtiene el siguiente PPL. Minimizar

$$y_1 + z$$

sujeto a

$$\begin{aligned} y_1 + y_2 - z &\leq 0 \\ y_2 + 2z &= 1 \\ y_1, y_2, z &\geq 0 \end{aligned}$$

■

13.2.7 Espacio vectorial como cono

En ocasiones, puede resultar interesante tratar sólo con las componentes no negativas de un vector. La siguiente proposición puede ser útil en estos casos.

Proposición 13.2 (espacio lineal como cono). *Un espacio lineal* $\mathbf{A}_\rho$ *es un caso particular del cono*

$$\mathbf{A}_\rho \equiv (\mathbf{A} : -\mathbf{A})_\pi, \tag{13.21}$$

donde $-\mathbf{A}$ *es la parte negativa de* $\mathbf{A}$*. En otras palabras, un espacio lineal es el cono generado por sus generadores y los vectores opuestos de los mismos.* ■

Obsérvese que esto equivale a duplicar el número de generadores. Sin embargo, existe una solución más eficiente que la que muestra el siguiente teorema.

Teorema 13.2 (espacio vectorial como cono). *Dado un espacio lineal* $\mathbf{A}_\rho$ *y un vector* $\mathbf{x} \in \mathbf{A}_\rho$ *con todas sus componentes positivas respecto al conjunto de generadores, entonces* $\mathbf{A}_\rho \equiv (\mathbf{A} : -\mathbf{x})_\pi$. ∎

Demostración. Como $(\mathbf{A} : -\mathbf{x})_\pi \subseteq \mathbf{A}_\rho$, sólo se necesita demostrar que $\mathbf{A}_\rho \subseteq (\mathbf{A} : -\mathbf{x})_\pi$. Puesto que $\mathbf{x}$ puede expresarse como:

$$\mathbf{x} = \sum_{i=1}^{m} \sigma_i \mathbf{a}_i; \;\; \sigma_i > 0; i = 1, \ldots, m \tag{13.22}$$

si $\mathbf{y} \in \mathbf{A}_\rho$, para cualquier real π, se tiene

$$\mathbf{y} = \sum_{i=1}^{m} \rho_i \mathbf{a}_i = \sum_{i=1}^{m} \rho_i \mathbf{a}_i + \pi\mathbf{x} - \pi\mathbf{x} = \sum_{i=1}^{m} (\rho_i + \pi\sigma_i)\, \mathbf{a}_i + \pi(-\mathbf{x}) \tag{13.23}$$

Si se elige

$$\pi = \max_{i=1,\ldots,m} \left| \frac{\rho_i}{\sigma_i} \right| \geq 0 \tag{13.24}$$

se tiene $\rho_i + \pi\sigma_i \geq 0; i = 1, \ldots, m$, y finalmente $\mathbf{y} \in (\mathbf{A} : -\mathbf{x})_\pi$. ∎

Nota 13.1 *El teorema anterior implica en la práctica que cualquier espacio lineal generado por* m *vectores puede considerarse como un cono generado por* $m + 1$ *vectores. Esto mejora el resultado de la proposición 13.2, expresión (13.21), que requiere* $2m$ *generadores.*

Para expresar un espacio lineal como un cono, se puede elegir $\mathbf{x}$ *de muchas formas, sin embargo,* $\mathbf{x} = \sum_{i=1}^{m} \mathbf{a}_i$*, conduce a* $\pi = \max_{i=1,\ldots,m} |\rho_i|$*, que es una elección muy conveniente de* $\mathbf{x}$. ∎

Ejemplo 13.6 (espacio vectorial como un cono). Considérese el espacio vectorial $\mathbf{A}_\rho$ donde

$$\mathbf{A} = \begin{pmatrix} 1 & -1 \\ 2 & 0 \\ 3 & 1 \end{pmatrix}$$

De acuerdo con la nota 13.1, se puede aplicar el teorema 13.2, con

$$\mathbf{x} = \sum_{i=1,2} \mathbf{a}_i = (1, 2, 3)^T + (-1, 0, 1)^T = (0, 2, 4)^T$$

y obtener $\mathbf{A}_\rho \equiv (\mathbf{A} : -\mathbf{x})_\pi \equiv \mathbf{B}_\pi$, con

$$\mathbf{B} = \begin{pmatrix} 1 & -1 & 0 \\ 2 & 0 & -2 \\ 3 & 1 & -4 \end{pmatrix}$$

Ahora bien, dado el vector $\mathbf{y}$, de $\mathbf{A}_\rho$, como

$$\mathbf{y} = -(1, 2, 3)^T - 2(-1, 0, 1)^T = (1, -2, -5)^T$$

se puede elegir

$$\pi = \max_{i=1,2} |\rho_i| = \max(|-1|, |-2|) = 2$$

y luego

$$\begin{array}{rcl} \mathbf{y} & = & (1,-2,-5)^T = (-1+2)(1,2,3)^T + (-2+2)(-1,0,1)^T + 2(0,-2,-4)^T \\ & = & (1,2,3)^T + 2(0,-2,-4)^T = \mathbf{a}_1 + 2(-\mathbf{x}) \end{array} \tag{13.25}$$

que es una combinación lineal no negativa del vector $\mathbf{a}_1$ y del vector $-\mathbf{x}$. ■

13.2.8 Restricciones alternativas

Las variables binarias constituyen una poderosa herramienta para modelar no linealidades de diversa naturaleza, que aparecen con frecuencia en el mundo de la ingeniería. Una aplicación posible de las variables binarias es modelar el cumplimiento de al menos uno de dos conjuntos de restricciones. Para conseguirlo, se necesita la siguiente proposición.

Proposición 13.3 (restricciones alternativas). *Considérense dos conjuntos de restricciones:*

$$\mathbf{A}_1^T \mathbf{x} \leq \mathbf{b}_1 \tag{13.26}$$
$$\mathbf{A}_2^T \mathbf{x} \leq \mathbf{b}_2 \tag{13.27}$$

El conjunto de restricciones que asegura el cumplimiento de al menos una de las restricciones anteriores es:

$$\mathbf{A}_1^T \mathbf{x} - y_1 \mathbf{d}_1 \leq \mathbf{b}_1 \tag{13.28}$$
$$\mathbf{A}_1^T \mathbf{x} - y_2 \mathbf{d}_2 \leq \mathbf{b}_2 \tag{13.29}$$
$$y_1 + y_2 \leq 1 \tag{13.30}$$
$$y_1, y_2 \in \{0,1\} \tag{13.31}$$

Las matrices columna $\mathbf{d}_1$ *y* $\mathbf{d}_2$ *deben satisfacer para todo* $\mathbf{x}$ *que:*

$$\mathbf{A}_1^T \mathbf{x} \leq \mathbf{b}_1 + \mathbf{d}_1 \Leftrightarrow \mathbf{d}_1 \geq \mathbf{A}_1^T \mathbf{x} - \mathbf{b}_1 \tag{13.32}$$
$$\mathbf{A}_2^T \mathbf{x} \leq \mathbf{b}_2 + \mathbf{d}_2 \Leftrightarrow \mathbf{d}_2 \geq \mathbf{A}_2^T \mathbf{x} - \mathbf{b}_2. \tag{13.33}$$

Por otro lado, si se necesita que se cumpla tan sólo una restricción de las dos posibles, entonces la condición $y_1 + y_2 \leq 1$ *se debe sustituir por* $y_1 + y_2 = 1$. ■

Demostración. Puesto que se cumplen (13.30) y (13.31), se tiene tres casos posibles.

Caso 1. $y_1 = y_2 = 0$. En este caso, (13.28) y (13.29) se transforman en (13.26) y (13.27), respectivamente; por tanto se cumplen las dos restricciones.

Caso 2. $y_1 = 0;\ \ y_2 = 1$. Puesto que (13.33) se cumple para todo x, y como (13.29) se transforma en (13.33), (13.29) no está activa, es decir, se puede eliminar.

Caso 3. $y_1 = 1;\ \ y_2 = 0$. Puesto que (13.32) se cumple para todo x, y como (13.28) se transforma en (13.32), (13.28) no está activa, es decir, se puede eliminar.

■

Ejemplo 13.7 (restricciones alternativas). Considérese el siguiente conjunto de restricciones:

$$\begin{array}{rcl} -x_1 & \leq & 0 \\ x_2 & \leq & 2 \\ 2x_1 - x_2 & \leq & 0 \end{array} \Leftrightarrow \begin{pmatrix} -1 & 0 \\ 0 & 1 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 0 \\ 2 \\ 0 \end{pmatrix} \tag{13.34}$$

y

$$\begin{array}{rcl} -x_1 & \leq & 0 \\ -x_2 & \leq & 0 \\ x_1 + x_2 & \leq & 1 \end{array} \Leftrightarrow \begin{pmatrix} -1 & 0 \\ 0 & -1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{13.35}$$

que delimitan los conjuntos factible A y B, que aparecen en la figura 13.1 con sombreado claro y oscuro, respectivamente.

Para calcular los límites correspondientes a (13.32) y (13.33) se puede buscar aquella región que con las menos modificaciones posibles tenga unos límites paralelos a los límites de A que contienen a B. Esta región es OST. Análogamente, aquella región que con las menos modificaciones posibles tenga sus límites paralelos a los límites de B y que contiene a A, es la región PQR.

Estas dos nuevas regiones pueden definirse mediante el siguiente conjunto de restricciones

$$\begin{array}{rcl} -x_1 & \leq & 0 \\ x_2 & \leq & 2 \\ 2x_1 - x_2 & \leq & 2 \end{array} \Leftrightarrow \begin{pmatrix} -1 & 0 \\ 0 & 1 \\ 2 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - y_1 \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix} \leq \begin{pmatrix} 0 \\ 2 \\ 0 \end{pmatrix} \tag{13.36}$$

$$\begin{array}{rcl} -x_1 & \leq & 0 \\ -x_2 & \leq & 0 \\ x_1 + x_2 & \leq & 3 \end{array} \Leftrightarrow \begin{pmatrix} -1 & 0 \\ 0 & 1 \\ 1 & 1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} - y_2 \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix} \leq \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{13.37}$$

de lo que se obtiene que

$$\mathbf{d}_1 = \mathbf{d}_2 = \begin{pmatrix} 0 \\ 0 \\ 2 \end{pmatrix} \tag{13.38}$$

Sin embargo, basta con elegir $\mathbf{d}_1$ y $\mathbf{d}_2$ suficientemente grandes.

■

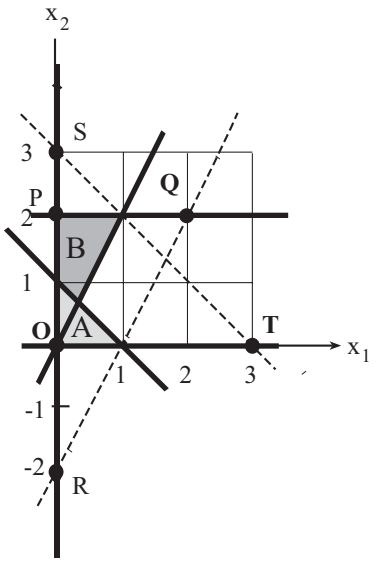

Figura 13.1: Representación gráfica de las restricciones alternativas del ejemplo 13.7.

13.2.9 Tratamiento de restricciones condicionales

Una condición del tipo

$$f_1(x_1,\ldots,x_n) > b_1 \qquad \text{implica que} \qquad f_2(x_1,\ldots,x_n) \leq b_2 \tag{13.39}$$

es equivalente al siguiente conjunto de restricciones

$$f_1(x_1,\ldots,x_n) \leq b_1 \qquad \text{y/o} \qquad f_2(x_1,\ldots,x_n) \leq b_2 \tag{13.40}$$

Las restricciones alternativas se analizaron en el apartado 13.2.8.

A continuación se demuestra la equivalencia anterior. El conjunto original de condiciones únicamente se incumple cuando

$$f_1(x_1,\ldots,x_n) > b_1 \qquad \text{y} \qquad f_2(x_1,\ldots,x_n) > b_2 \tag{13.41}$$

y, por tanto, se cumple cuando

$$f_1(x_1,\ldots,x_n) \leq b_1 \qquad \text{y/o} \qquad f_2(x_1,\ldots,x_n) \leq b_2 \tag{13.42}$$

13.2.10 Tratamiento de funciones no continuas

Las variables binarias también permiten el tratamiento de funciones no continuas, lo que se demuestra en la siguiente proposición.

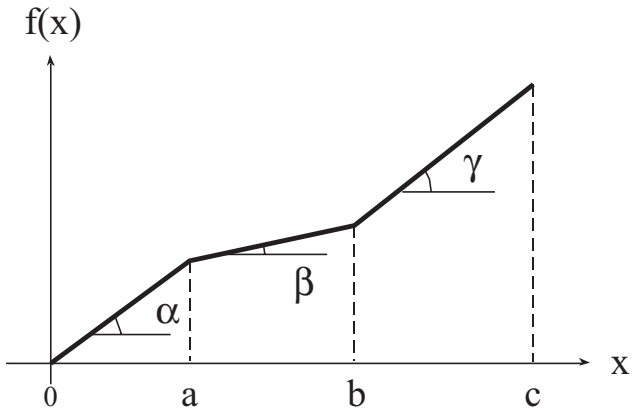

Figura 13.2: Función no convexa a trozos.

Proposición 13.4 (discontinuidad). *La función discontinua a minimizar*

$$f(x) = \begin{cases} 0 & x = 0 \\ k + cx & 0 < x \leq b;\ k > 0 \end{cases} \tag{13.43}$$

se puede formular como

$$\begin{array}{rcl} f(x) & = & ky + cx \\ x & \leq & by \\ -x & \leq & 0 \\ y & \in & \{0, 1\} \end{array} \tag{13.44}$$

■

Demostración. Se analizan las dos únicas posibilidades:

Caso 1. Si $y = 0$ por la segunda y tercera ecuación en (13.44), se tiene que $0 \leq x \leq 0$; es decir, $x = 0$, y sustituyendo $x = y = 0$ en la primera ecuación se tiene que $f(x) = 0$.

Caso 2. Si, por el contrario $y = 1$, se tiene que $0 \leq x \leq b$ y $f(x) = k + cx$. Obsérvese que la discontinuidad en cero ($f(0) = 0, f(0^+) = k$) deja de existir al minimizar $f(x)$, pues esto implica que $f(0^-) = 0$. ■

13.2.11 Tratamiento de funciones no convexas a trozos

En este apartado se explica cómo usar variables binarias para tratar funciones no convexas a trozos.

Proposición 13.5 (función no convexa a trozos). *La función no convexa a trozos* (*véase la figura 13.2*)

$$f(x) = \begin{cases} \alpha x & 0 \le x \le a \\ \alpha a + \beta(x-a) & a < x \le b \\ \alpha a + \beta(b-a) + \gamma(x-b) & b < x \le c \end{cases}$$

donde

$$\beta < \alpha < \gamma; \;\; \alpha, \beta, \gamma > 0; \;\; a < b < c; \;\; a, b, c > 0$$

puede expresarse como

$$\begin{aligned} f(x) &= \alpha x_1 + \beta x_2 + \gamma x_3 && (13.45) \\ x &= x_1 + x_2 + x_3 && (13.46) \\ a w_1 &\le x_1 \le a && (13.47) \\ w_2(b-a) &\le x_2 \le (b-a) w_1 && (13.48) \\ 0 &\le x_3 \le (c-b) w_2 && (13.49) \\ w_2 &\le w_1 && (13.50) \\ w_1, w_2 &\in \{0, 1\} && (13.51) \end{aligned}$$

■

Demostración. Como consecuencia de las dos últimas restricciones (13.50) y (13.51), se tiene tres casos posibles:

Caso 1. $w_2 = w_1 = 0$. Lo anterior, junto con (13.47)–(13.49), implica que $x_2 = x_3 = 0$ y $0 \le x_1 \le a$, y, por tanto, (13.46) lleva a $x = x_1$ y $0 \le x \le a$, y $f(x) = \alpha x$.

Caso 2. $w_2 = 0; \;\; w_1 = 1$. Lo anterior, junto con (13.54)–(13.56), implica que $x_3 = 0$ y $0 \le x_2 \le b - a$ y $x_1 = a$, y, por tanto, (13.53) lleva a $x = a + x_2 \Rightarrow a \le x \le b$ y $f(x) = \alpha a + \beta(x - a)$.

Caso 3. $w_2 = w_1 = 1$. Lo anterior, junto con (13.54)–(13.56), implica que $0 \le x_3 \le c - b$, $x_2 = b - a$ y $x_1 = a$, y, por tanto, (13.53) lleva a $x = b + x_3 \Rightarrow b \le x \le c$ y $f(x) = \alpha a + \beta(b-a) + \gamma(x-b)$. ■

Proposición 13.6 (función no convexa a trozos con discontinuidad inicial). *La función no convexa a trozos con discontinuidad inicial* (*véase la figura 13.3*)

$$f(x) = \begin{cases} 0 & 0 \le x < a \\ f_0 + \alpha(x-a) & a < x \le b \\ f_0 + \alpha(b-a) + \beta(x-b) & b < x \le c \\ f_0 + \alpha(b-a) + \beta(c-b) + \gamma(x-c) & c < x \le d \end{cases}$$

donde

$$0 < \beta < \alpha < \gamma; \;\; 0 < a < b < c < d$$

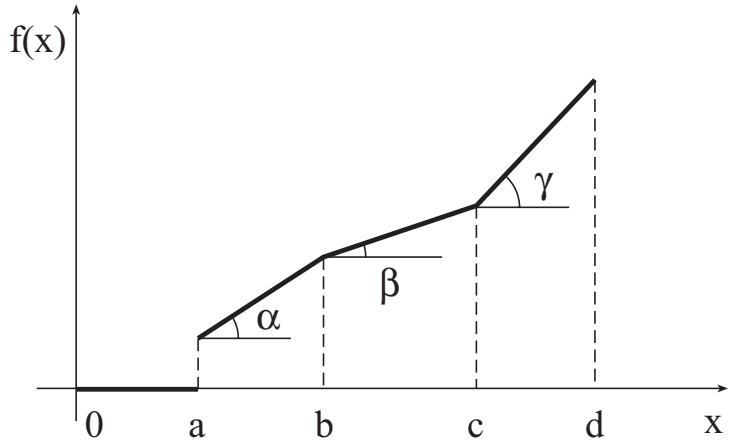

Figura 13.3: Función no convexa a trozos con discontinuidad inicial.

puede expresarse como

$$
\begin{aligned}
f(x) &= vf_0 + \alpha x_1 + \beta x_2 + \gamma x_3 && (13.52)\\
x &= va + x_1 + x_2 + x_3 && (13.53)\\
w_1(b-a) &\le x_1 \le (b-a)v && (13.54)\\
w_2(c-b) &\le x_2 \le (c-b)w_1 && (13.55)\\
0 &\le x_3 \le (d-c)w_2 && (13.56)\\
w_2 &\le w_1 \le v && (13.57)\\
v, w_1, w_2 &\in \{0,1\} && (13.58)
\end{aligned}
$$

■

Demostración. Como consecuencia de las dos últimas restricciones (13.57) y (13.58), se tiene cuatro casos posibles:

Caso 1. $w_2 = w_1 = v = 0$. Lo anterior, junto con (13.54)–(13.56), implica que $x_1 = x_2 = x_3 = 0$, y, por tanto, (13.53) lleva a $x = 0$ y $f(x) = 0$.

Caso 2. $w_2 = w_1 = 0$; $v = 1$. Lo anterior, junto con (13.54)–(13.56), implica que $x_2 = x_3 = 0$ y $0 \le x_1 \le b - a$, y por tanto, (13.53) lleva a $x = a + x_1 \Rightarrow a \le x \le b$ y $f(x) = f_0 + \alpha(x - a)$.

Caso 3. $w_2 = 0$; $w_1 = v = 1$. Lo anterior, junto con (13.54)–(13.56), implica que $x_3 = 0$, $0 \le x_2 \le c - b$ y $x_1 = b - a$, y por tanto (13.53) lleva a $x = b + x_2 \Rightarrow b \le x \le c$ y $f(x) = f_0 + \alpha(b - a) + \beta(x - b)$.

Caso 4. $w_2 = w_1 = v = 1$. Lo anterior, junto con (13.54)–(13.56), implica que $0 \leq x_3 \leq d - c$, $x_2 = c - b$ y $x_1 = b - a$, y por tanto (13.53) lleva a $x = c + x_3 \Rightarrow c \leq x \leq d$ y $f(x) = f_0 + \alpha(b - a) + \beta(c - b) + \gamma(x - c)$.

■

13.3 Algunos trucos en GAMS

En este apartado, se presentan los siguientes trucos en GAMS:

- Asignación de valores a una matriz
- Definición de matrices simétricas
- Definición de matrices cuasi-vacías
- Descomposición de un problema en subproblemas
- Adición iterativa de restricciones
- Tratamiento de los estados inicial y final
- Análisis de sensibilidad
- Dependencia del flujo del programa con respecto a la resolución de un problema

13.3.1 Asignación de valores a una matriz

La mayoría de los lenguajes de programación utilizan bucles para asignar valores a los elementos de una matriz. En GAMS, esto no es necesario, el usuario de GAMS puede asignar un valor inicial a todos los elementos de la matriz con una sola sentencia. Esto se debe a que en GAMS la asignación se hace en 'paralelo'. El siguiente ejemplo ilustra cómo realizar esta asignación. Sean las siguientes matrices:

$$\mathbf{m_1} = \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \end{pmatrix}, \quad \mathbf{m_2} = \begin{pmatrix} 2 & 3 & 4 \\ 3 & 4 & 5 \end{pmatrix}$$

El código para definir ambas matrices puede ser el siguiente:

```
SETS   I  indice de filas    /1*2/
       J  indice de columnas /1*3/;

PARAMETER m1(I,J),m2(i,J);

** Asignacion en paralelo para todos los elementos
 m1(I,J)=1;
 m2(I,J)=ord(I)+ord(J);

DISPLAY m1,m2;
```

La anterior sentencia `DISPLAY` produce:

```
----      9 PARAMETER M1
            1           2           3
1       1.000       1.000       1.000
2       1.000       1.000       1.000

----      9 PARAMETER M2
            1           2           3
1       2.000       3.000       4.000
2       3.000       4.000       5.000
```

13.3.2 Definición de una matriz simétrica

Para representar una matriz simétrica basta con definir la submatriz triangular superior. Los valores de la submatriz triangular inferior se pueden asignar posteriormente utilizando una expresión condicional. Para ilustrar este truco, se considera la siguiente matriz simétrica:

$$\mathbf{a} = \begin{pmatrix} 1 & 0 & 3 \\ 0 & 2 & 4 \\ 3 & 4 & 5 \end{pmatrix}$$

El código GAMS que permite definir esta matriz es el siguiente:

```
SET   I  indice de filas y columnas  /1*3/
ALIAS(I,J);

PARAMETER a(I,J) matriz de datos
          /1.1  1
           2.2  2
           3.3  5
           1.3  3
           2.3  4
          /;

** Primero se muestra, en la salida, la submatriz triangular superior
DISPLAY a;

** Se asignan valores a la submatriz triangular inferior
a(I,J)$(ord(I) gt ord(J))=a(J,I);

** Se muestra en la salida la matriz completa
DISPLAY a;
```

El resultado que produce la sentencia `DISPLAY` es el siguiente:

```
----     13 PARAMETER A             matriz de datos
            1           2           3
1       1.000                   3.000
2                   2.000       4.000
3                               5.000

----     19 PARAMETER A             matriz de datos
```

```
          1           2           3
1     1.000                   3.000
2                 2.000       4.000
3     3.000       4.000       5.000
```

En el apartado 11.4.10 se presenta un ejemplo alternativo.

13.3.3 Definición de una matriz cuasi-vacía

Las matrices cuasi-vacías se pueden definir de distintas maneras en GAMS. La opción más sencilla es definirlas por enumeración de los elementos distintos de cero. Esta opción es apropiada cuando la posición de los elementos distintos de cero no sigue ningún patrón determinado. Sin embargo, las matrices cuasi-vacías de gran dimensión suelen tener una distribución determinada de sus elementos distintos de cero. En estos casos, puede resultar ineficiente la enumeración de los elementos, y es mejor aprovechar el conocimiento de la distribución que siguen estos. Si la posición de los elementos distintos de cero sigue un patrón, las asignaciones pueden realizarse imponiendo condiciones sobre los índices de sus posiciones. De lo contrario, es necesario definir un conjunto especial que establezca la correspondencia entre elementos que tengan el mismo valor. Sea la siguiente matriz

$$\mathbf{s} = \begin{pmatrix} 0 & 0 & 1 & 2 & 0 & 0 \\ 0 & 1 & 2 & 0 & 0 & 1 \\ 1 & 2 & 0 & 0 & 1 & 0 \\ 2 & 0 & 0 & 1 & 0 & 2 \\ 0 & 0 & 1 & 0 & 2 & 0 \\ 0 & 1 & 0 & 2 & 0 & 1 \end{pmatrix}$$

Un código que aprovecha la distribución de los elementos distintos de cero puede ser

```
SET  I  indice de filas y columnas /I1*I6/;

ALIAS(I,J);

PARAMETER s(I,J) matriz cuasi-vacia;
 s(I,J)$(mod((ord(I)+ord(J)),4) eq 0)=1;
 s(I,J)$(mod((ord(I)+ord(J)),5) eq 0)=2;
DISPLAY s;
```

El siguiente código define la matriz **s** mediante la enumeración de las posiciones que ocupan los elementos distintos de cero.

```
SETS  I  indice de filas y columnas /I1*I6/
      MAP1(I,I)     /I1.I3,I2.I2,I2.I6,I3.I5,I4.I4,I6.I6/
      MAP2(I,I)     /I1.I4,I2.I3,I4.I6,I5.I5 /;
ALIAS(I,J);

PARAMETER s(I,J) matriz cuasi-vacia;

 s(I,J)$MAP1(I,J)=1;
```

```
 s(I,J)$MAP2(I,J)=2;

 s(I,J)$(ord(I) gt ord(J))=s(J,I);

DISPLAY s;
```

Los dos códigos anteriores producen la misma matriz:

```
----     10 PARAMETER S             matriz cuasi-vacia
           I1          I2          I3          I4          I5          I6
I1                                1.000       2.000
I2                    1.000       2.000                                 1.000
I3         1.000      2.000                                 1.000
I4         2.000                                 1.000                 2.000
I5                                1.000                     2.000
I6                    1.000                     2.000                 1.000
```

13.3.4 Descomposición de un problema separable

En algunas ocasiones, desde una perspectiva computacional, puede resultar útil descomponer un problema en varios subproblemas. En estos casos, se puede formular el problema igual que si no fuese separable, y luego, mediante un bucle, se resuelven secuencialmente los distintos subproblemas. Por ejemplo, dado el siguiente problema de optimización. Minimizar

$$Z = \sum_{j=1}^{n} \sum_{i=1}^{n} c_{ij} x_{ij}$$

sujeto a

$$\sum_{i=1}^{n} a_i x_{ij} \geq b_j \quad j = 1, \ldots, n$$

Este problema se puede descomponer en tantos subproblemas como el cardinal del intervalo de variación del índice j (un total de n subproblemas). Cada uno de estos subproblemas tiene la forma siguiente: Minimizar

$$Z = \sum_{i=1}^{n} c_{ij} x_{ij}$$

sujeto a

$$\sum_{i=1}^{n} a_i x_{ij} \geq b_j$$

Para comprobar cómo GAMS resuelve los n subproblemas, se sugiere estudiar el fichero de salida GAMS que genera el siguiente código:

```
SETS   I  indice de filas    /1*2/
       J  indice de columnas /1*3/

       DIN(J) conjunto dinamico;

ALIAS(J,J1);
```

```
PARAMETERS c(I,J),a(I),b(J);
 c(I,J)=1; a(I)=1; b(J)=1;

POSITIVE VARIABLE x(I,J);
VARIABLE z;

EQUATIONS
 Cost(J)  funcion objetivo
 Rest(J)  restriccion;


Cost(J)$DIN(J)..    SUM(I,c(I,J)*x(I,J)) =e= z;
Rest(J)$DIN(J)..    SUM(I,a(I)*x(I,J))   =g=  b(J);

Model descompone /all/;

DIN(J)=NO;
loop(J1,
     DIN(J1)=YES;
     Solve descompone using lp minimizing z;
     DIN(J1)=NO;
);
```

13.3.5 Adición iterativa de restricciones a un problema

Este truco es de utilidad para algoritmos que resuelven un problema de forma iterativa. La solución óptima del problema de partida se consigue iterativamente mediante la resolución de problemas intermedios a los que se van añadiendo restricciones. Las restricciones que se añaden tienen la misma estructura pero difieren en los datos. Para programar este tipo de algoritmos es necesario el uso de bucles y conjuntos dinámicos. En primer lugar, se inicia el conjunto dinámico al conjunto vacío. Luego, este conjunto se actualiza añadiendo el índice que referencia la nueva restricción en cada iteración. Considérese la siguiente secuencia de problemas.

Minimizar

$$Z = cx$$

sujeto a

$$ax \geq b^{(i)}, \;\; i = 1, \ldots, t$$

Supóngase que el número máximo de iteraciones que necesita el algoritmo es 5, $(t = 5)$, y también que el escalar $b^{(i)}$ se incrementa en uno tras cada iteración; entonces se puede resolver el problema anterior en 5 iteraciones utilizando el siguiente código:

```
SETS   I contador de iteraciones  /1*5/
       DIN(I) conjunto dinamico;

ALIAS(I,I1);
```

```
SCALARS c,a;

PARAMETER b(I);
 c=1; a=1; b(I)=1;

POSITIVE VARIABLE x;
VARIABLE z;

EQUATIONS
 Cost     funcion objetivo
 Rest(I)  restriccion;

Cost..              c*x =e= z;
Rest(I)$DIN(I)..    a*x =g= b(I);

Model adicion /all/;

DIN(I)=NO;
loop(I1,
     DIN(I1)=YES;
     Solve adicion using lp minimizing z;
     b(I+1)=b(I)+1;
);
```

Para comprender mejor el funcionamiento del código anterior, el lector debe estudiar el fichero de salida GAMS.

13.3.6 Tratamiento de los estados inicial y final

En muchos problemas de ingeniería es importante establecer los estados inicial y final de ciertas variables. En GAMS, estos estados se pueden tratar mediante condiciones sobre las variables y restricciones que los modelan. Como siempre, el programador debe definir un conjunto ordenado para recorrer las variables en cuestión. En ese conjunto ordenado se puede distinguir mediante distintos identificadores un índice para el estado inicial y otro para el estado final. En GAMS, siempre es posible referirse al primer y último elemento de un conjunto que representarán respectivamente, el estado inicial y final. Por ejemplo, dado el conjunto K de períodos de tiempo, las condiciones para referirse a distintos subconjuntos son:

- `$(ord(K) EQ 1)` para el instante inicial
- `$(ord(K) EQ card(K))` para el instante final
- `$((ord(K) GT 1) AND (ord(K) LT card(K)))` para todos los instantes, excepto el inicial y el final.

Para más detalle, véanse los ejemplos de planificación de la producción y de la programación horaria de centrales térmicas en los apartados 11.2.2 y 11.3.6, respectivamente.

13.3.7 Análisis de sensibilidad

En ciertas aplicaciones, puede ser de interés establecer la dependencia de la solución respecto a algunos parámetros. Para ello, el problema se resuelve repetidas veces cambiando algunos datos de entrada o límites de sus variables.

A continuación se muestra, a modo de ejemplo, la dependencia de la solución de un problema respecto a un coeficiente de una restricción. El valor del coeficiente se modifica en cada iteración. Minimizar

$$Z = cx$$

sujeto a

$$a^{(t)}x \geq b.$$

El coeficiente a se incrementa en una unidad en cada iteración, es decir, $a^{(t+1)} = a^{(t)} + 1$. El siguiente código resuelve el problema 5 veces variando el valor de a.

```
SET   I  contador de iteraciones /1*5/

SCALARS a,b,c;
 a=1; b=1; c=1;

POSITIVE VARIABLE x;

VARIABLE z;

EQUATIONS
 Cost      funcion objetivo
 Rest      restriccion;

Cost..    c*x =e= z;
Rest..    a*x =g= b;

Model analisis /all/;

loop(I,
     Solve analisis using lp minimizing z;
     a=a+1;
);
```

13.3.8 Dependencia del flujo del programa

En ciertos casos, es interesante variar la estructura de un modelo según la información que se obtiene de problemas resueltos con anterioridad. Por ejemplo, si tras la resolución de un problema se concluye que éste no está acotado, se puede acotar añadiendo una nueva restricción al problema anterior y resolviéndolo de nuevo. El siguiente problema muestra un ejemplo sencillo. Minimizar

$$Z = cx$$

Si la solución del problema anterior está acotada se termina la ejecución del programa; si no, se añade un nueva restricción, de forma que el problema anterior se reescribe como sigue. Minimizar

$$Z = cx$$

sujeto a

$$ax \geq b$$

El siguiente código refleja la resolución condicionada de este último problema. En el código se incluyen algunos comentarios aclaratorios.

```
SCALARS c,a,b,cond;
 c=1; a=1; b=1;

VARIABLE x,z;

EQUATIONS
 COST      funcion objetivo
 REST      restriccion adicional;

COST..                c*x =e= z;
REST$(cond eq 1)..    a*x =g= b;

MODEL depende /ALL/;

** Si cond es 0 no se incluye en el modelo la restriccion REST
   cond=0;

** Se resuelve el modelo sin la restriccion REST
   SOLVE depende USING lp MINIMIZING z;

** Si el problema no esta acotado se agrega la restriccion REST
** y se vuelve a resolver

   if(depende.modelstat eq 3,

**   Se incluye en el modelo la restriccion REST
     cond=1;

**   El modelo se resuelve con la nueva restriccion
     SOLVE depende USING lp MINIMIZING z;
   );
```

Para comprender mejor el funcionamiento del programa, el lector debe estudiar el fichero de salida GAMS

Ejercicios

13.1 Un problema de estimación se puede formular como

$$\text{Minimizar}_{\Theta \in C} \; F(\hat{\mathbf{y}}, \mathbf{y}(\Theta))$$

donde $\hat{\mathbf{y}}$ son los datos observados, e $\mathbf{y}(\Theta)$ son los valores predichos por un modelo en función del vector de parámetros que se quiere estimar. F puede ser cualquier relación entre los datos observados y los datos predichos. Si F es la relación que viene dada por la norma euclídea ($\|\cdot\|_2$), el método de estimación se denomina método de los mínimos cuadrados (MMC), y F será igual a $\sum_{i=1}^{n}(\hat{y}_i - y_i)^2$. Si F viene dada por la norma ℓ_1 ($\|\cdot\|_1$), es decir, $F(\hat{\mathbf{y}}, \mathbf{y}) = \sum_{i=1}^{n} |\hat{y}_i - y_i|$, el método se denomina el *método del mínimo valor absoluto* (MVA). Otra posibilidad es expresar F como la norma del supremo ($\|\cdot\|_\infty$) lo cual implica que $F(\hat{\mathbf{y}}, \mathbf{y}) = \max\{|\hat{y}_i - y_i| \; i = 1, \cdots, n\}$. Este método se denomina *método de estimación min-max* (MM). Considérense los dos modelos teóricos que describen la variable y_i como una función de los parámetros a y b

$$\begin{array}{lrcl} \text{Modelo lineal} & y_i & = & a + bx_i \\ \text{Modelo no lineal} & y_i & = & (x_i - a)^b \end{array}$$

Estimar los parámetros a y b suponiendo que $(a, b) \in \mathbb{R}^2$ en ambos modelos, según los datos de entrada de la Tabla 13.1, mediante los métodos MMC, MVA y MM.

Tabla 13.1: Datos de entrada para el problema de estimación

x_i	$\hat{y}_i$
0	1.4
1	3.8
1.5	6.7
2	9.2
3	15.2
4	24.3
5	30.2

13.2 Considérese la siguiente secuencia de problemas. Minimizar

$$Z = x$$

sujeto a

$$a^{(i)}x \geq b^{(i)} \quad i = 1, \ldots, t$$

Supóngase que $(t = 5)$, y también que los parámetros $(a^{(i)}, b^{(i)})$ vienen dados por la Tabla 13.2. Escribir un código en GAMS para resolver la secuencia de problemas anteriores.

Tabla 13.2: Parámetros para la secuencia de problemas

$a^{(i)}$	$b^{(i)}$
1	1.7
2	2.4
3	3.8
4	4.2
5	5.7

13.3 Sea un circuito eléctrico con 3 nudos y 3 líneas. El límite de capacidad y la susceptancia de las líneas (véase la sección 1.8 del capítulo 1) vienen dados por la siguiente tabla:

Línea	Susceptancia	Límite
1–2	2.5	0.3
1–3	3.5	0.7
2–3	3.0	0.7

La demanda eléctrica en el nudo 3 es de 0.85 MW. Existen dos generadores ubicados en los nudos 1 y 2, respectivamente. Los límites de producción de estos generadores son los siguientes:

Generador	Límite superior	Límite inferior
1	0.9	0
2	0.9	0

El coste de producción de los generadores se expresa como

$$c_i = \begin{cases} 0 & \text{if } p_i = 0 \\ F_i + V_i p_i & \text{if } p_i > 0 \end{cases}$$

donde F_i es el coste fijo, V_i es el coste variable, y p_i es la potencia de salida.

Los costes fijo y variable de los dos generadores viene dados por la siguiente tabla:

Generador	F_i	V_i
1	10	6
2	5	7

Determínese la producción de cada generador de forma que el coste total al suministrar la demanda sea mínimo. Repítase el problema tomando $F_2 = 10$.

13.4 Sea el siguiente problema de programación lineal. Minimizar

$$Z = x_1$$

sujeto a

$$\begin{array}{rcl} -2x_1 + x_2 & \leq & 0 \\ x_1 - x_2 & \leq & 2 \\ x_1 + x_2 & \leq & 6 \end{array}$$

(a) Resuélvase gráficamente

(b) Exprésese el problema en forma estándar considerando las siguientes alternativas:

(i) Tratamiento de las variables no acotadas como la diferencia entre dos variables no negativas

(ii) Tratamiento de las variables no acotadas añadiendo una sola variable

(c) Empléese GAMS para resolver el problema formulado con las dos alternativas y comparar los resultados

13.5 Sea el siguiente problema de programación lineal fraccional. Minimizar

$$Z = \frac{-2x_1 - 2x_3 + 3}{x_1 + x_2 + x_3 + 1}$$

sujeto a

$$\begin{array}{rcl} -2x_1 + x_2 & \leq & 0 \\ - x_2 & \leq & 0 \\ x_1 - x_2 & \leq & 2 \\ x_1 + x_2 & \leq & 6 \end{array}$$

(a) Empléese GAMS para resolver el problema de programación lineal fraccional

(b) Formúlese este problema como un problema de programación lineal

(c) Empléese GAMS para resolver el problema de programación lineal y comparar los resultados con los obtenidos en (a).

13.6 Sean los dos conjuntos de restricciones

$$A = \left\{ \begin{array}{rcl} -2x_1 + x_2 & \leq & 0 \\ x_1 - x_2 & \leq & 2 \\ x_1 + x_2 & \leq & 6 \end{array} \right.$$

$$B = \left\{ \begin{array}{rcl} -x_1 + x_2 & \leq & -6 \\ -x_1 + x_2 & \leq & -2 \\ 3x_1 - x_2 & \leq & 18 \end{array} \right.$$

Tabla 13.3: Distribución de la temperatura dentro de la pared

Distancia	0	0.2	0.4	0.6	0.8	1.0
Temperatura	400	350	250	175	100	50

(a) Exprésese matemáticamente el siguiente problema. Minimizar

$$Z = x_1 - 3x_2$$

sujeto a

$$(x_1, x_2) \quad \in \quad A \cup B$$

(b) Resuélvase gráficamente este problema

(c) Empléese GAMS para resolver este problema. Modifíquese el código GAMS para obtener una solución óptima alternativa

13.7 Sea un problema de programación lineal con restricciones de igualdad y desigualdad. Escríbase un problema equivalente con sólo restricciones de desigualdad.

13.8 Sea un problema de programación matemática con una función objetivo no lineal. Escríbase un problema equivalente con una función objetivo lineal.

13.9 La Tabla 13.3 muestra la variación de temperatura dentro de una pared, que se calienta por una de sus caras, respecto a la distancia de la zona en la que se aplica el calor. Se quiere ajustar estos datos a un modelo lineal que exprese la temperatura como una función de la distancia

$$T = a + bx$$

donde a y b son constantes. Si los parámetros se estiman minimizando los errores siguientes, determínense estos coeficientes, por medio de GAMS y los trucos que se describen en este capítulo.

(a)

$$\sum_i \|a + bx_i - T_i\|$$

(b)

$$\max_i \|a + bx_i - T_i\|$$

donde las parejas (x_i, T_i) vienen dadas por la Tabla 13.3.

13.10 Dada una matriz $T(i, j)$, escribir una sola sentencia de GAMS que permita contar los elementos de la diagonal principal que son distintos de 10.

13.11 En GAMS, una vez se fija el valor de una variable, ésta permanece constante a lo largo del tiempo de ejecución. Supóngase que se requiere resolver un problema mediante un algoritmo iterativo en que el valor de la variable es fijo, pero distinto en cada iteración. Escríbase el código GAMS para abordar este caso.