CAPITULO 14: GAMS

14.1.- INSTALACION DE GAMS

El programa GAMS (General Algebraic Modeling System) es un software desarrollado por A. Brooke, D. Kendrick y A. Meeraus. A diferencia de otros paquetes de software de implementación de algoritmos matemáticos que permiten resolver los problemas de optimización, el programa GAMS presenta la ventaja de plantear un lenguaje de modelización que permite el poder escribir en un editor la formulación matemática del problema y posteriormente aplicarle una serie de "solvers" o programas de resolución.

Este programa fue desarrollado a finales de la década de los años 80 en el World Bank por un grupo de economistas, aprovechando la experiencia de su trabajo sobre programas de desarrollo económico, que requieren en primer lugar una modelización exhaustiva y posteriormente la aplicación de los correspondientes programas de optimización para poder hallar la solución numérica a los modelos propuestos.

Para este texto usaremos una de las ultimas versiones de GAMS. Se trata de una versión STUDENT o reducida que tiene disponibles todos los solvers disponibles, pero se encuentre limitada por el numero de variables o el numero de elementos distintos de cero en el modelo. La practica totalidad de los ficheros disponibles en el texto se pueden resolver son esta versión de demostración.

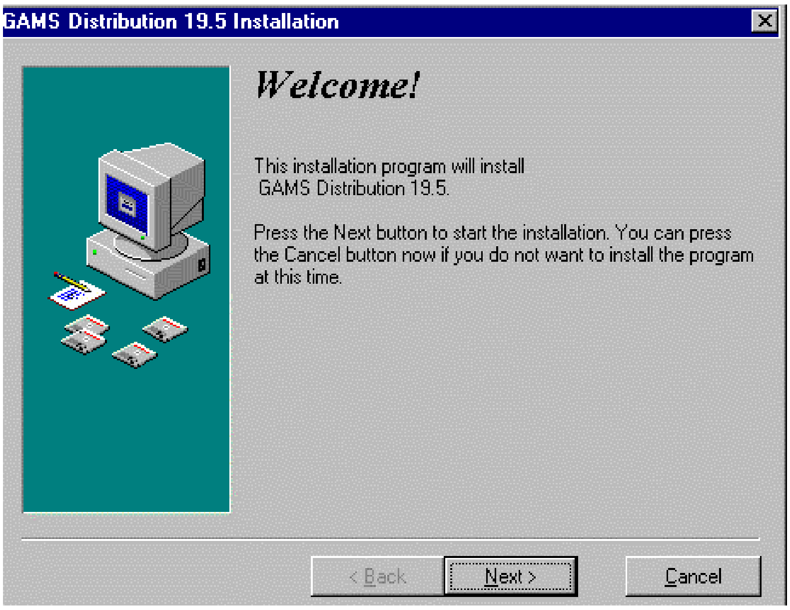
Para poder instalar esta versión necesitamos disponer de un ordenador con sistema operativo Windows 95 o superior, con 40 Mb. libres en el disco duro, y se recomienda al menos 32 Mb. de memoria RAM para ejecutar modelos de tamaño medio, aunque la versión de demostración tiene unas serie de limitaciones en cuanto al tamaño de los modelos como:

Máximo numero de filas:	300
Máximo numero de columnas:	300
Máximo numero de elementos distintos de cero:	2000
Máximo numero de elementos no lineales:	1000
Máximo numero de variables discretas:	50

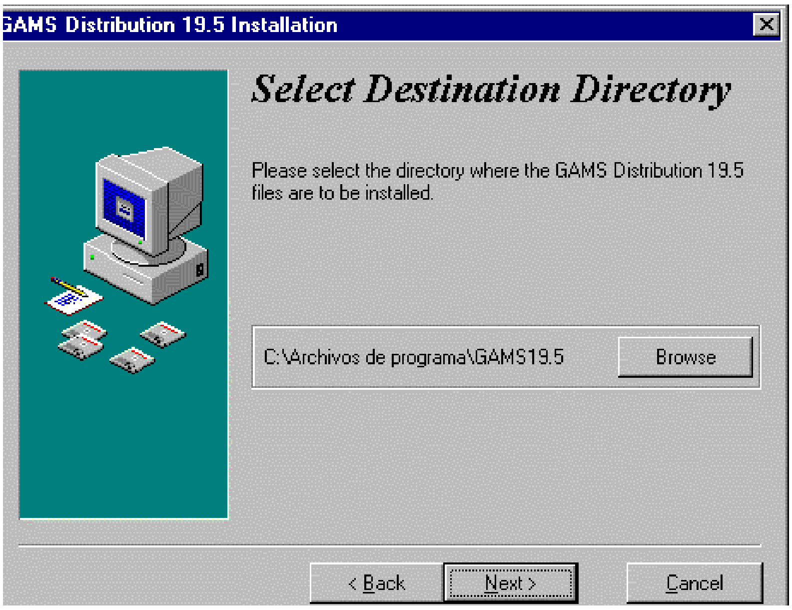
Por ello, puede ser suficiente con 16 Mb. de RAM.

En el CD se puede obtener el programa de instalación (GAMS 19.5). La dirección es: d\GAMS\setup.exe. Ejecutando este fichero se desarrolla un programa de autoinstalción que nos guiara a la instalación completa.

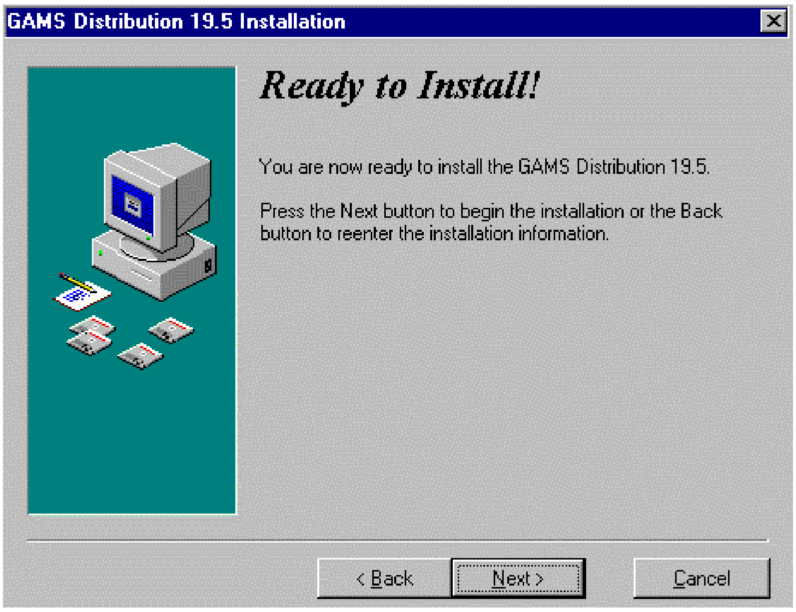
Al ejecuatarlo aparace una pantalla como la siguiente:



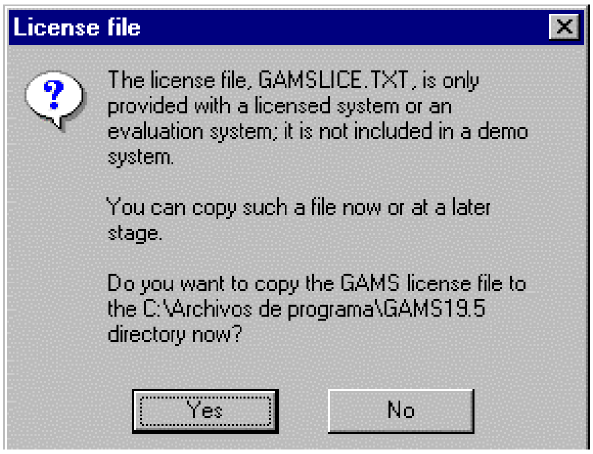
A continuación nos pregunta en que directorio del disco deseamos instalar el programa: (por ejemplo, en C:\Archivos de programas\GAMS 19.5)



Una vez instalado pregunta donde encontrar una licencia de GAMS (si no se dispone de una versión profesional de GAMS), lo indicado en indicarle que NO.



Una vez completada la instalación nos aparece una pantalla como lo siguiente:



Es posible que GAMS nos pregunte donde deseamos situar el "project", es decir la dirección donde -por defecto- buscará los ficheros de input y depositara los ficheros de output. En ese caso podemos indicarle la dirección más conveniente.

Una vez instalado el programa, y cuando lo ejecutemos nos aparecerá una pantalla como la siguiente:

Eligiendo la opción **File-New,** obtenemos una pantalla como la siguiente, en donde podemos comenzar a escribir las instrucciones correspondiente:

También podemos acceder a la ultima versión de GAMS en la siguiente dirección:

http://www.gams.com/download/

En esa pagina web (web site) es necesario registrarse con una dirección de correo electrónico. Una vez registrados se recibe una contraseña para poder acceder a la pagina de descarga de los archivos necesarios. En la misma pagina de registro se informa de las limitaciones que se han comentado anteriormente.

Una vez copiados los archivos en el disco, se procede a la instalación automática del programa, como acabamos de exponer previamente. En esa instalación se crean el fichero de ayuda necesarias, así como una serie de manuales de ayuda (todos ellos en formato PDF). Estos manuales incluyen una guía de usuario de casi 300 paginas. Además incluye un “Tutorial” de 30 paginas, además de una breve referencia en la ayuda del programa. Pero no solo tiene ayuda sobre el programa, sino también tiene referencias y guías sobre los solvers que se usan como MINOS, CPLEX; OSL, etc.

Una vez instalado se puede ejecutar alguno de los modelos de la librería de programas, ya sea lineal, no lineal, entero, PNL no diferenciable, Programación complementaria, etc.

14.2.- GAMS: REFERENCIAS BASICAS

En el epígrafe anterior hemos visto la instalación de GAMS, y ahora vamos a explicar los rudimentos básicos para poder iniciar el trabajo con este programa.

Este programa permite muchas interrelaciones con otros lenguajes de alto nivel (Fortran, C etc.,), así como la aplicación de diversos algoritmos de resolución. En la actualidad se están desarrollando nuevas versiones de GAMS que permiten importar y exportar datos con hojas de cálculo, que son herramientas de uso muy común en las tareas informatizadas.

En este capitulo queremos recoger un resumen de los elementos básicos de funcionamiento con GAMS, en la versión para Windows, es decir GAMS-IDE

GAMS, presenta la ventaja de la potencia de este lenguaje de modelización, así como la capacidad para resolver problemas lineales, enteros y no lineales, sin olvidar las posibilidades de crecimiento del mismo programa como lenguaje de modelización, aunque aquí no se exploraran todas las posibilidades.

Para poder ejecutar el programa GAMS es necesario crear un fichero de datos donde recoger toda la información necesaria del problema, aunque todo ello introducido en un formato particular. La versión actual de GAMS-IDE incorpora un editor que facilita la escritura y resolución de los problemas.

En los ficheros de modelos, hay que organizar una serie de bloques que son obligatorios y otros bloques que son opcionales.

Nos centraremos en los bloques obligatorios, pues lo que se pretende en este anexo es sólo recopilar las instrucciones de funcionamiento de GAMS.

Los bloques obligatorios son:

Variables	VARIABLES
Ecuaciones	EQUATIONS
Modelo	MODEL
Solución	SOLVE

Los bloques optativos son:

Conjuntos	SET
Datos	DATA
Visualización	DISPLAY

Líneas de comentario.

En todos los ficheros que generemos siempre es conveniente introducir líneas de comentario que, si bien no forman parte del modelo y por lo tanto no van a ser compiladas, nos pueden facilitar la lectura posterior tanto del fichero de datos como el de la solución.

Las líneas de comentarios pueden ser introducidas de dos formas distintas:

a) Comenzando cada línea con un asterisco (*), en este caso hay que tomar en consideración que ciertos símbolos están prohibidos, como por ejemplo los acentos, la letra ñ, etc.

b) Cuando hay que escribir varias líneas, para no tener que escribir cada vez un asterisco (*), es conveniente utilizar el comando $ONTEXT, a continuación de él podemos escribir tantas líneas como queramos y además utilizar cualquier carácter (acentos, ñ, etc.), para indicar que han finalizado las líneas de comentarios hay que utilizar el comando $OFFTEXT.

Bloque de variables.

Este bloque debe comenzar con la palabra VARIABLES. Dentro de este bloque se han de definir las variables que se van a usar en el modelo, indicando de que clase son, que tipo de restricciones presentan, si tienen o no cotas y el punto de partida.

Bloque de ecuaciones.

Este bloque ha de comenzar con el título EQUATIONS. En este bloque hay que declarar y definir las ecuaciones que se van a usar en el modelo.

Bloque de modelo.

En este grupo se han de definir las ecuaciones que componen el modelo. No es obligatorio incluir todas las ecuaciones utilizadas. Este bloque tiene que comenzar con el nombre de MODEL.

Bloque de solución.

En este bloque hay que indicar que tipo de algoritmo deseamos usar para poder resolver el modelo que se ha definido previamente. A la hora de inicializar este bloque ha de aparecer la palabra SOLVE.

Además de estos cuatro bloques obligatorios y como ya se ha indicado con anterioridad, se pueden definir otros tres bloques de carácter opcional:

Bloque de Conjuntos, SET. Consiste en definir una serie de conjuntos, por lo general índices y asignarles unos valores a estos conjuntos.

Bloque de Datos, DATA. No se trata de un único bloque, sino que puede contener diferentes grupos. Se usa para definir una serie de datos fijos dentro del modelo, así podemos definir parámetros (PARAMETERS), tablas (TABLES) y escalares (SCALARS).

Bloque de visualización, DISPLAY. Este bloque permite indicar la clase de salida de datos y formato que deseamos para el problema. En principio nos limitaremos a comentar la salida estándar (por defecto) que proporciona GAMS.

Vamos a comenzar por explicar la construcción de un fichero de datos, con los bloques obligatorios, ya que en un anexo posterior usaremos el mismo ejercicio con los bloques opcionales.

El fichero lo crearemos utilizando un editor y le vamos a dar el nombre (para este ejemplo) de LINEAL.GMS. La extensión GMS es la que por defecto se usa para identificar a los ficheros de datos de GAMS.

Comenzaremos por introducir las líneas de comentario correspondientes:

```
$ONTEXT

Se trata de introducir los datos en forma matricial, para poderlo
resolver mediante un programa lineal.

        Min F(x) = 3*x1 + 2*x2 +x3 + 2*x4 +3*x5
        s.a:
               2*x1 + 5*x2 + x4 + x5 >= 6
               4*x2 -2*x3 +2*x4 + 3*x5 >= 5
               x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 >= 7
              x1, x2, x3, x4, x5 no negativas

$OFFTEXT
```

Vamos a exponer los bloques o grupos que son susceptibles de añadir en un fichero GMS:

Bloque de variables:

Como se trata de un bloque obligatorio, en el incluiremos las siguientes partes:

- Nombre de las variables
- Clase de las variables (positivas, enteras, etc.)
- Cotas de las variables (superiores, inferiores, valores fijos, etc.)
- Punto de partida. Especialmente en los problemas no lineales.

En nuestro problema el bloque de variables es el siguiente:

```
VARIABLES
X1, X2, X3, X4, X5, F;
POSITIVE VARIABLES
X1, X2, X3, X4, X5;
```

Bloque de ecuaciones:

En este bloque hemos de incorporar dos partes:

- Nombre de las ecuaciones
- Definición de las variables

En nuestro ejemplo seria:

```
EQUATIONS
OBJ, R1, R2, R3;

OBJ..   F =E= 3*x1 + 2*x2 +x3 + 2*x4 +3*x5 ;
R1..    2*x1 + 5*x2 + x4 + x5 =G= 6;
R2..    4*x2 -2*x3 +2*x4 + 3*x5 =G= 5;
R3..    x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 =G= 7;
```

Bloque de model y solve:

En estos dos bloques hemos de definir que variables forman parte del modelo, y a continuación definir que tipo de modelos es y que dirección de optimización debemos seguir.

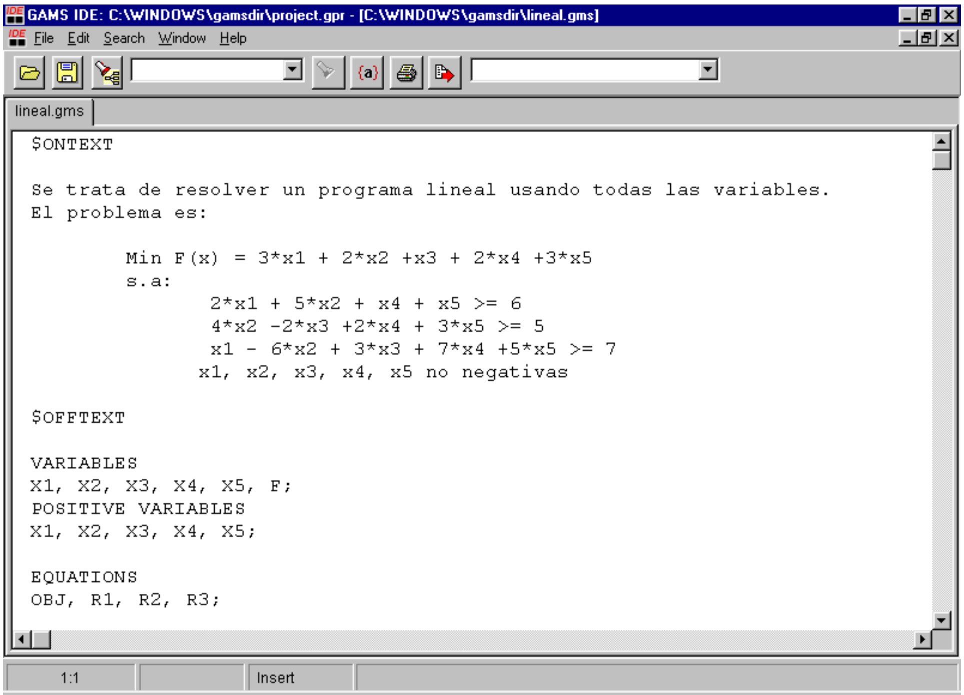
En nuestro caso será:

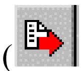
```
MODEL LINEAL /OBJ, R1, R2, R3/;

SOLVE LINEAL USING LP MINIMIZING F;
```

Con todo ello el fichero completo (lineal.gms) será:

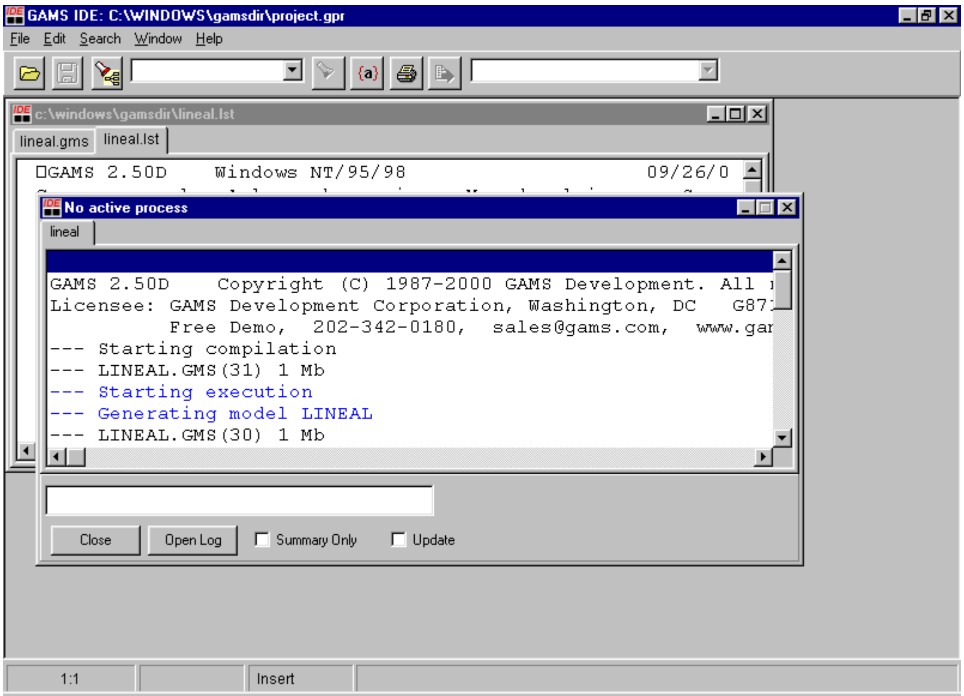
```
$ONTEXT

Se trata de resolver un programa lineal usando todas las variables.
El problema es:
        Min F(x)  = 3*x1 + 2*x2 +x3 + 2*x4 +3*x5
        s.a:
               2*x1 + 5*x2 + x4 + x5 >= 6
               4*x2 -2*x3 +2*x4 + 3*x5 >= 5
               x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 >= 7
              x1, x2, x3, x4, x5 no negativas
$OFFTEXT
VARIABLES
X1, X2, X3, X4, X5, F;
POSITIVE VARIABLES
X1, X2, X3, X4, X5;

EQUATIONS
OBJ, R1, R2, R3;

OBJ..   F =E= 3*x1 + 2*x2 +x3 + 2*x4 +3*x5 ;
R1..    2*x1 + 5*x2 + x4 + x5 =G= 6;
R2..    4*x2 -2*x3 +2*x4 + 3*x5 =G= 5;
R3..    x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 =G= 7;

MODEL LINEAL /OBJ, R1, R2, R3/;
SOLVE LINEAL USING LP MINIMIZING F;
```

Con ello la pantalla de ejecución será:



Para ejecutar el fichero debemos elegir RUN dentro de la opción FILE o pulsa F9 o el boton con la flecha roja del editor ()

Con ello, lo que se obtiene es:



Para analizar la solución debemos abrir el fichero LST (lineal.lst), que en nuestro caso es el siguiente:
(Los saltos de pagina están indicados como: =============)

```
GAMS Rev 116  Windows NT/95/98                    12/12/00 16:37:17  PAGE      1
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
C o m p i l a t i o n

      Se trata de resolver un programa lineal usando todas las variables.
      El problema es:
              Min F(x) = 3*x1 + 2*x2 +x3 + 2*x4 +3*x5
              s.a:
                     2*x1 + 5*x2 + x4 + x5 >= 6
                     4*x2 -2*x3 +2*x4 + 3*x5 >= 5
                     x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 >= 7
                    x1, x2, x3, x4, x5 no negativas
  12  VARIABLES X1, X2, X3, X4, X5, F;
  13  POSITIVE VARIABLES X1, X2, X3, X4, X5;
  14
  15  EQUATIONS OBJ, R1, R2, R3;
  16
  17  OBJ..   F =E= 3*x1 + 2*x2 +x3 + 2*x4 +3*x5 ;
  18  R1..    2*x1 + 5*x2 + x4 + x5 =G= 6;
  19  R2..    4*x2 -2*x3 +2*x4 + 3*x5 =G= 5;
  20  R3..    x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 =G= 7;
  21
  22  MODEL LINEAL /OBJ, R1, R2, R3/;
  23  SOLVE LINEAL USING LP MINIMIZING F;
  24

COMPILATION TIME     =        0.550 SECONDS    0.7 Mb      WIN194-116
=========================================================================================
GAMS Rev 116  Windows NT/95/98                    12/12/00 16:37:17  PAGE      2
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
Equation Listing    SOLVE LINEAL USING LP FROM LINE 23

---- OBJ  =E=

OBJ..  - 3*X1 - 2*X2 - X3 - 2*X4 - 3*X5 + F =E= 0 ; (LHS = 0)

---- R1  =G=

R1..  2*X1 + 5*X2 + X4 + X5 =G= 6 ; (LHS = 0, INFES = 6 ***)

---- R2  =G=

R2..  4*X2 - 2*X3 + 2*X4 + 3*X5 =G= 5 ; (LHS = 0, INFES = 5 ***)

---- R3  =G=
```

```
R3..  X1 - 6*X2 + 3*X3 + 7*X4 + 5*X5 =G= 7 ; (LHS = 0, INFES = 7 ***)
===========================================================================================

GAMS Rev 116  Windows NT/95/98                    12/12/00 16:37:17  PAGE      3
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
Column Listing      SOLVE LINEAL USING LP FROM LINE 23

---- X1


X1
                (.LO, .L, .UP = 0, 0, +INF)
       -3       OBJ
        2       R1
        1       R3

---- X2


X2
                (.LO, .L, .UP = 0, 0, +INF)
       -2       OBJ
        5       R1
        4       R2
       -6       R3

---- X3


X3
                (.LO, .L, .UP = 0, 0, +INF)
       -1       OBJ
       -2       R2
        3       R3

---- X4


X4
                (.LO, .L, .UP = 0, 0, +INF)
       -2       OBJ
        1       R1
        2       R2
        7       R3

---- X5


X5
                (.LO, .L, .UP = 0, 0, +INF)
       -3       OBJ
        1       R1
        3       R2
        5       R3
===========================================================================================
```

```
GAMS Rev 116  Windows NT/95/98                    12/12/00 16:37:17  PAGE      4
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
Column Listing      SOLVE LINEAL USING LP FROM LINE 23

---- F

F
                (.LO, .L, .UP = -INF, 0, +INF)
        1       OBJ
=========================================================================================

GAMS Rev 116  Windows NT/95/98                    12/12/00 16:37:17  PAGE      5
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
Model Statistics    SOLVE LINEAL USING LP FROM LINE 23


MODEL STATISTICS

BLOCKS OF EQUATIONS       4     SINGLE EQUATIONS        4
BLOCKS OF VARIABLES       6     SINGLE VARIABLES        6
NON ZERO ELEMENTS        19

GENERATION TIME      =        0.390 SECONDS    1.4 Mb      WIN194-116

EXECUTION TIME       =        0.440 SECONDS    1.4 Mb      WIN194-116
=========================================================================================

GAMS Rev 116  Windows NT/95/98                    12/12/00 16:37:17  PAGE      6
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m

               S O L V E      S U M M A R Y

     MODEL   LINEAL              OBJECTIVE  F
     TYPE    LP                  DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  23

**** SOLVER STATUS     1 NORMAL COMPLETION
**** MODEL STATUS      1 OPTIMAL
**** OBJECTIVE VALUE               5.1707

 RESOURCE USAGE, LIMIT          2.800     1000.000
 ITERATION COUNT, LIMIT         2        10000

GAMS/Cplex    Aug  7, 2000 WIN.CP.NA 19.4 016.015.038.WAT For Cplex 6.6
Cplex 6.6.1, GAMS Link 16, Using a GAMS/Cplex demo license installed at runtime.

Optimal solution found.

Objective :           5.170732
```

```
                       LOWER     LEVEL     UPPER    MARGINAL

---- EQU OBJ              .         .         .        1.000
---- EQU R1             6.000     6.000     +INF      0.634
---- EQU R2             5.000     6.878     +INF       .
---- EQU R3             7.000     7.000     +INF      0.195

                       LOWER     LEVEL     UPPER    MARGINAL

---- VAR X1               .         .       +INF      1.537
---- VAR X2               .       0.854     +INF       .
---- VAR X3               .         .       +INF      0.415
---- VAR X4               .       1.732     +INF       .
---- VAR X5               .         .       +INF      1.390
---- VAR F              -INF      5.171     +INF       .

**** REPORT SUMMARY :        0     NONOPT
                             0 INFEASIBLE
                             0  UNBOUNDED

EXECUTION TIME       =        0.110 SECONDS    0.7 Mb      WIN194-116


USER: GAMS Development Corporation, Washington, DC   G871201:0000XX-XXX
      Free Demo,  202-342-0180,  sales@gams.com,  www.gams.com   DC9999

**** FILE SUMMARY

INPUT      C:\RAMON\PROVALLIBRE\1401.GMS
OUTPUT     C:\RAMON\GMS\1401.LST
```

Como comentario general, el fichero de salida contiene varias partes:

- Compilación. Es decir la transformación de las instrucciones originales en código legible por el solver.
- Listado de ecuaciones. Allí se recogen todas las ecuaciones del modelo que se han escrito. Siempre es conveniente repasarlo para detectar los posibles errores, ya que los errores de lenguaje los detecta GAMS directamente mientras que los numéricos tiene que ser el propio usuario. Así, por ejemplo, para el caso de la función objetivo, GAMS escribe todas las variables en el primer miembro de la ecuación, por eso aparecen con coeficientes negativos las variables principales del problema. El termino `(LHS = 0)` significa que el termino de la izquierda toma el valor cero. Eso es así porque al no definir un punto de partida inicial, se toma por defecto el cero.

```
---- OBJ  =E=
OBJ..  - 3*X1 - 2*X2 - X3 - 2*X4 - 3*X5 + F =E= 0 ; (LHS = 0)
```

- Listado de columnas o de variables. Aparecen relacionadas todas las variables y los coeficientes que incorporan en cada ecuación. Así por ejemplo, para la variable X2, tenemos:

```
X2
                (.LO, .L, .UP = 0, 0, +INF)
       -2       OBJ
        5       R1
        4       R2
       -6       R3
```

Podemos observar que la variable X2, tiene una cota inferior (LO) de 0, el punto de partida (L) es cero, y la cota superior (UP) es +INF.

Esta variable tiene un coeficiente de **- 2** en la función objetivo, tal como explicado anteriormente. Si analizamos las restantes ecuaciones, vemos que el valor asociado al nombre de cada restricción se corresponde con el coeficiente de la variable X2 en cada ecuación.

- Estadísticas del modelo:

```
MODEL STATISTICS

BLOCKS OF EQUATIONS        4     SINGLE EQUATIONS         4
BLOCKS OF VARIABLES        6     SINGLE VARIABLES         6
NON ZERO ELEMENTS         19
```

Nos señalan el numero de variables y ecuaciones que contiene el modelo.

- El resumen de la solución. En este apartado podemos distinguir dos partes diferenciadas: a) referida al proceso de solución, y b) la referida al valor de las variables y al comportamiento de las ecuaciones.

```
                S O L V E      S U M M A R Y

     MODEL   LINEAL              OBJECTIVE  F
     TYPE    LP                  DIRECTION  MINIMIZE
     SOLVER  OSL                 FROM LINE  30

**** SOLVER STATUS     1 NORMAL COMPLETION
**** MODEL STATUS      1 OPTIMAL
**** OBJECTIVE VALUE                5.1707

 RESOURCE USAGE, LIMIT          2.800     1000.000
 ITERATION COUNT, LIMIT         2        10000

GAMS/Cplex    Aug  7, 2000 WIN.CP.NA 19.4 016.015.038.WAT For Cplex 6.6
Cplex 6.6.1, GAMS Link 16, Using a GAMS/Cplex demo license installed at
runtime.

Optimal solution found.
```

En el cuadro anterior no señala que el modelo es lineal, y queremos minimizar la función objetivo (F) y el solver que se usa es CPLEX. También nos indica que se ha encontrado una solución optima, y con un valor de 5.1707, y para ello se han realizado 2 iteraciones de las 10.000 posibles.

Con relación a la solución obtenida tenemos:

```
                      LOWER     LEVEL     UPPER    MARGINAL
---- EQU OBJ            .         .         .        1.000
---- EQU R1            6.000     6.000     +INF      0.634
---- EQU R2            5.000     6.878     +INF       .
---- EQU R3            7.000     7.000     +INF      0.195

                      LOWER     LEVEL     UPPER    MARGINAL
---- VAR X1             .         .        +INF      1.537
---- VAR X2             .        0.854     +INF       .
---- VAR X3             .         .        +INF      0.415
---- VAR X4             .        1.732     +INF       .
---- VAR X5             .         .        +INF      1.390
---- VAR F            -INF       5.171     +INF       .
```

La información facilitada se refiere primero a las ecuaciones. Allí se analiza la existencia de las cotas (tipo de restricción ≥, tiene cota inferior). Y el comportamiento de la restricción (saturada o no). Asimismo, para las restricciones saturadas nos informa del valor del multiplicador o marginal, que nos indica en cuanto variaría el valor de la función al variar el termino independiente. En este caso las dos restricciones activas son: R1 y R3.

Lo mismo ocurre con las variables, siendo la solución: X2=0.854 y X4=1.732, con un valor de la función objetivo de 5.171 (redondeo de5.1707).

- Informe resumen: Todas las instrucciones que tengan cuatro asteriscos (****) son muy importantes. En este caso, no hay soluciones no optimas, ni infactibles, ni no acotadas, es decir, la solución es normal y optima.

```
**** REPORT SUMMARY :        0     NONOPT
                             0 INFEASIBLE
                             0  UNBOUNDED
```

- Resumen de origen y final de los fichero: Es decir el nombre de los ficheros de entrada y salida, asi como su localización.

```
**** FILE SUMMARY

INPUT      C:\RAMON\1401.GMS
OUTPUT     C:\RAMON\1401.LST
```

Como final vamos a comentar brevemente el significado del elemento MARGINAL o multiplicador de las restricciones activas.

Para ello modificaremos ligeramente el términos independiente de la primera restricción:

```
R1..    2*x1 + 5*x2 + x4 + x5 =G= 6;
```

Por:

```
R1..    2*x1 + 5*x2 + x4 + x5 =G= 6.1;
```

Al resolver el problema nos encontramos con la siguiente solución:

```
**** MODEL STATUS      1 OPTIMAL
**** OBJECTIVE VALUE                5.2341
```

Si recordamos el significado del multiplicador, tenemos que:

$$\frac{\partial L}{\partial b} = \lambda$$

Por tanto, al incrementar el termino independiente (b), el incremento de la función es:

$$\Delta F \cong \Delta b.\lambda$$

En nuestro caso particular tenemos:

$$\lambda = 0.634$$

$$\Delta b=0.1$$

$$\Delta F \cong 0.634 * 0.1 = 0.0634$$

Si comparamos las diferencias de los valores de las dos funciones objetivo tenemos:

$$F(6) = 5.1707$$

$$F(6.1)= 5.2341$$

Es decir:

$$\Delta F = F(6.1) - F(6) = 5.2341 - 5.1707 = 0.0634$$

que es precisamente el resultado que habíamos obtenido anteriormente.

14.3.- GAMS: REFERENCIAS AVANZADAS

En este epígrafe vamos a exponer algunos procedimientos más avanzados sobre GAMS, aunque para los conceptos más elementales y ampliación de los mismos aconsejamos también remitirse a la Guía de Usuario de GAMS[1].

Con el objetivo de ilustrar estos elementos vamos a proceder a la construcción del problema de programación lineal, en forma matricial para la introducción de los elementos necesarios para ello.

Comenzaremos por introducir las líneas de comentario correspondientes:

```
$ONTEXT

Se trata de introducir los datos en forma matricial, para poderlo
resolver mediante un programa lineal.

        Min F(x) = 3*x1 + 2*x2 +x3 + 2*x4 +3*x5
        s.a:
               2*x1 + 5*x2 + x4 + x5 >= 6
               4*x2 -2*x3 +2*x4 + 3*x5 >= 5
               x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 >= 7
              x1, x2, x3, x4, x5 no negativas

$OFFTEXT
```

Vamos a exponer los bloques o grupos que son susceptibles de añadir en un fichero GMS:

GRUPO ***SET***

Este bloque o grupo nos sirve para definir conjuntos, aunque lo normal es definir con el índices, tanto de variables como de restricciones.

Como todos los grupos debe comenzar con el nombre del bloque: SET y a continuación definir el nombre y los elementos que lo forman. Por ejemplo:

SET S /1, 2, 3, 4/

o también podemos escribirlo con nombre de comentario:

SET S Los cuatro primeros números

[1] Brooke, A., Kendrick, D. y Meeraus, A.: "GAMS: A user's guide". Scientific Press. California.

```
/ 1        uno
  2        dos
  3        tres
  4        cuatro/
```

Pueden definirse más de un conjunto de forma simultánea, es decir:

```
SET
    I    /I1, I2, I3/
    J    /J1, J2, J3, J4, J5 /
```

A veces, cuando hemos de introducir algunos elementos con nombres comunes se pueden introducir con un asterisco entre el primer elemento y el último. Así los conjuntos anteriores también los podemos definir como:

```
SET
I    /I1*I3/
J    /J1*J5 /
```

con ello GAMS identifica todos los elementos desde el J1 hasta el J5, es decir J1, J2, J3, J4, J5.

En determinados problemas pueden existir índices de longitud y elementos iguales, para estos casos no es necesarios definirlos dos veces. Así por ejemplo si los conjuntos I y J son:

```
SET    I    /1, 2, 3, 4/
SET    J    /1, 2, 3, 4/
```

podemos definir solo uno, y asociar los elementos de este conjunto a otro conjunto mediante la declaración de un ALIAS. Por ejemplo:

```
SET    I    /1, 2, 3, 4/
ALIAS (I,J).
```

Otros elementos que se pueden crear dentro de este grupo, son los subconjuntos (SUBSETS) y los conjuntos de varias dimensiones (MULTIDIMENSIONAL SETS). Estos elementos se pueden crear como combinación de conjuntos anteriores. Para mayores referencias puede consultarse la guía de usuario de GAMS.

Del problema anterior de programación lineal, los elementos correspondientes al grupo SET son:

```
SET
        J       VARIABLES    / 1*5/
        I       RESTRICCIONES/ 1*3/;
```

*GRUPO **DATA***

La introducción de los datos en los modelos es siempre una fuente de errores, y por esto es muy importante ser cuidadosos en el manejo de los datos. Así que siempre es preferible introducir los datos de la forma más cómoda y fácil posibles, así como introducirlos una sola vez.

Dentro del grupo DATA, podemos distinguir tres tipos de elementos: PARAMETER, SCALAR y TABLES

El grupo PARAMETER suele estar asociado a los elementos de los conjuntos (SET), y sirven normalmente para indicar disponibilidades o requerimientos de estos.

Veamos el ejemplo que hemos descrito inicialmente, en donde es necesario definir los coeficientes de la función objetivo y el vector de términos independientes de las restricciones en el problema lineal:

```
PARAMETERS

C(J)     COEF F /1       3.
                2        2.
                3        1.
                4        2.
                5        3./

B(I)     RHS    /1       6.
                2        5.
                3        7/;
```

Como podemos observar en el grupo PARAMETERS hemos definido los parámetros (vectores en este caso) C(J) asociado a las variables (J) y B(I) asociado con las restricciones (I).

El grupo C(J) denominado coeficientes de la función objetivo, se componen de los pares de valores que asocian a cada elemento del conjunto J el correspondiente coeficiente de la función objetivo, es decir, a la variable 1 se le asocia el coeficiente 3, mientras que a la variable 2 el coeficiente 2, y así sucesivamente.

En el caso de los términos independiente B(I) de las restricciones, a la restricción 1 se le ha asociado el límite de 6, mientras que a la restricción 2 el termino independiente es 5, y en la tercera restricción 7. Nótese que al final del bloque se ha incorporado la señal de final de bloque (;).

También es posible asociar parámetros a elementos de grupos multidimensionales, pero en este caso también nos remitimos al manual de GAMS para poder realizar este análisis.

El grupo <u>SCALAR</u> se usa para declarar e inicializar los parámetros de dimensión cero, es decir, que no están asociados a ninguna clase de conjuntos. Algunos ejemplos de este grupo serían:

SCALARS	V	Version /2/
	D	Dia del mes de Noviembre
	NUN	Numero de unidades /20/

D = 10;

En el grupo de escalares hemos declarado el símbolo D, pero no se ha definido, como los símbolos V y NUN, en otra parte del fichero se ha asignado un valor de 10 a D.

El grupo TABLE proporciona una forma fácil de introducir los datos del problema lo habitual es trabajar con tablas de dos dimensiones, ya que en el caso de una dimensión estaríamos hablando de PARAMETER. Dentro de este bloque podemos distinguir tres tipos de tablas: tablas simples (dos dimensiones), tablas continuadas (tablas de dos dimensiones pero con ancho o largo mayor que una pantalla) y tablas de más de dos dimensiones.

Las *tablas simples*, es decir, las tablas de dos dimensiones se realizan declarando el nombre de la tabla, así como los conjuntos que forman parte de ella, y a continuación escribiendo en la primera fila y la primera columna, los correspondientes elementos de las columnas y las filas correspondientes.

Comenzaremos, reproduciendo la tabla que corresponde a la matriz técnica del problema lineal:

```
TABLE   A(I,J)  MATRIZ

                        1       2       3       4       5
                1       2       5       0       1       1
                2       0       4       -2      2       3
                3       1       -6      3       7       5;
```

Como podemos observar junto a la identificación de TABLE hemos añadido el nombre correspondiente A seguido de los conjuntos que están incluidos en la matriz, en este caso los conjunto I y J. Pero en cualquier caso la primera fila contiene los nombres de los elementos del conjunto J, convenientemente espaciados mediante los tabuladores correspondientes.

La primera columna contiene a los elementos del conjunto I, y a partir de ahí incorporamos los correspondientes valores de la tabla, debajo de cada una de la columnas. Al final de la tabla se incorpora el identificador de final de bloque (;).

Las *tablas continuadas* son también tablas simples, pero de un mayor numero de columnas o de filas para poder ser visualizadas en una pantalla. Dado que es conveniente no incorporar elementos a lo largo de todas las posiciones que permite el editor (256) sino que es preferible usar solo las primeras 80 columnas o menos. Cuando tenemos que incorporar tablas de grandes dimensiones, se pueden escribir todas las filas y las correspondientes columnas sin más que añadir el signo "+" a la izquierda de la columna de los nombres.

El caso anterior escrito como tabla continuada sería:

```
TABLE   A(I,J)  MATRIZ

                        1       2
                1       2       5
                2       0       4
                3       1       -6
            +
                        3       4       5
                1       0       1       1
                2       -2      2       3
                3       3       7       5;
```

Como puede observarse solamente hemos escrito debajo de la columna indicativa de las filas el signo "+" y hemos repetido las filas, pero hemos cambiando el encabezamiento de las columnas.

Las *tablas de dimensiones mayores que dos* se pueden escribir como el los casos anteriores, pero teniendo la precaución de asociar las identificaciones a las correspondientes filas o columnas.

Un ejemplo en este caso podría ser:

```
      SET   I     /I1*I2/
            J     /J1*J3/
            K     /K1*K4/

      TABLE A(I,J,K)    TABLA DE TRES CONJUNTOS.
                  K1    K2    K3    K4
      I1.J1       5     -2    4     2
      I1.J2       -1    0.5   2     -5
      I1.J3       4           2     -7
      I2.J1             3     -5    89
      I2.J2       -1.5  2     5     6
      I2.J3       2     5     -8    -.3
```

La introducción de datos es un fuente inagotable de errores, por eso conviene insistir nuevamente en que esta es una de las fases esenciales de la construcción de los ficheros GMS. El fichero de salida (LST) nos informa de los errores que se comenten en el fichero original, pero hay que prestar mucha atención a esta fase. Uno de los errores muy comunes es la repetición de los nombres en los diferentes grupos, por ejemplo definir un parámetro como A, y posteriormente etiquetar una tabla con el nombre A(I,J), que aunque aparentemente tienen nombre diferente, en realidad le estamos asociando el mismo A, ya que en el caso de la tabla solo le añadimos la identificación de los conjuntos que la componen. Por ello hay que reiterar el cuidado en la construcción del fichero, si no se quiere repetir muchas veces la ejecución del problema.

Como siempre, una mayor explicación de estas operaciones puede verse en el manual del usuario de GAMS.

Dentro del bloque de DATA, aunque también es posible realizarlas en otros bloques -por ejemplo, EQUATIONS- se pueden realizar operaciones de asignación de valores a diferentes parámetros. Estas pueden ser operaciones simples, operaciones con índices , operaciones con funciones, operaciones relacionales, etc.

Las operaciones simples, son las que incluyen los símbolos aritméticos elementales:

Suma	+
Diferencia o resta	-
Producto	*
Cociente	/
Exponenciación	**

estas operaciones se pueden realizar tanto con parámetros , escalares, tablas o funciones.

Las operaciones con índices más comunes son:

Sumatorio: $\sum_i \sum_j A_{ij}$ lo podemos expresar como: SUM((I,J), A(I,J))

Productorio: $\prod_i a_i^{b_i}$ lo podemos expresar como: PROD(I, a(I)**b(I))

Máximo de un conjunto: Máx $\{a_i\}$ lo podemos expresar como:SMAX(I, a(I))

Mínimo de un conjunto: Min $\{a_i\}$ lo podemos expresar como:SMIN(I, a(I))

Las operaciones con funciones más comunes son las siguientes:

ABS(x)	valor absoluto de x
CEIL(x)	parte entera por exceso de x
FLOOR(x)	parte entera por defecto de x
EXP(x)	exponente base e, e^x.
LOG(x)	logaritmo neperiano, ln x
LOG10(x)	logaritmo decimal de x, log x.
MAX(x,y,...z)	Máximo de x, y,..., z
MIN(x,y,...z)	Mínimo de x, y,..., z
NORMAL(x,y)	Distribución normal de media x y desviación y.
POWER(x,y)	potenciación, x^y , siendo y entero.
SIGN(X)	signo de x: +1 si x > 0, -1 si x < 0, 0 si x = 0.
SQR(x)	cuadrado de x, x^2.
SQRT(X)	raíz cuadrada de x, $\sqrt{x}$

Con las diferentes expresiones de un modelo también se pueden establecer operaciones relacionales entre ellas, así se usan las expresiones:

LT	Menor (<)
LE	Menor-igual (≤)
EQ	Igual (=)
NE	No igual (≠)
GE	Mayor-igual (≥)
GT	Mayor (>)
NOT	No
AND	y
OR	o

Estas operaciones relacionales, tienen un operador fundamental (operador Dólar, $) también conocido como la asignación condicional, cuando aparece en alguna relación.

Vamos a ver a continuación algunos ejemplo del uso de este operador condicional.

Para expresar la suma de un conjunto de parámetros, todos ellos positivos, podemos utilizar diversas expresiones:

SUMATOT = SUM(I, ABS(X(I)));

esta expresión nos indica la suma de los valores absolutos de las variables X(I), o también:

SUMATOT = SUM(I$(X(I) GE 0), X(I)) + SUM(I$(X(I) LT 0), -X(I));

esta expresión descompone la suma en dos partes, en primer lugar la suma de las variables, si estas variables son mayores o iguales que cero (GE 0), y un segundo sumando para las variables menores que cero (LT 0), y en este caso suma las variables cambiadas de signo, o también:

SUMATOT = SUM(I$(X(I) GE 0), X(I)) + SUM(I$(NOT(X(I) GE 0)), -X(I));

esta expresión es igual a la anterior en el primer sumando, mientras que el segundo sumando significa que si X(I) <u>no</u> (NOT) es mayor-igual que cero, sume los X(I) cambiados de signo.

Las posibilidades de este operador relacional son muy amplias, y por ello es preferible ver distintas posibilidades a lo largo de los diferentes problemas, para ver la incidencia de este tipo de operaciones. No obstante, otras aplicaciones pueden verse en la Guía de Usuario de GAMS.

Otra forma de introducir los datos es mediante operaciones, es decir, asignar valores que se obtienen mediante el calculo de alguna expresión. Vamos a introducir un ejemplo de esta forma de introducción de datos, que además nos pueda servir para ilustrar el funcionamiento de GAMS sin necesidad de incluir un solver para resolver un problema de optimización, sino simplemente como calculadora o como hoja de calculo.

Vamos a considerar el calculo del valor actual neto (VAN) de dos proyectos de inversión con varios periodos de tiempo.

El fichero GMS para calcular el VAN se recoge a continuación, en donde previamente se definen como PARAMETER el valor de VAN(PR), que se obtiene con posterioridad al desarrollar la expresión correspondiente, y por ultimo se visualiza mediante un DISPLAY el valor de los datos obtenido.

```
$TITLE VAN1
$ONTEXT

        Calculo del VAN de un proyecto de inversion

$OFFTEXT


SET      P /PER0*PER7/;
SET      PR/PRO1*PRO2/;

PARAMETER       VAN(PR);
PARAMETER       FAC(P);
PARAMETER       FACTOR(P);

SCALAR  TI      /.1/;

TABLE A(PR,P)
        PER0    PER1    PER2    PER3    PER4    PER5    PER6    PER7
PRO1    -100    10      15      20      30      40      15      50
PRO2    -1000   150     250     350     450     500     0       0;


FAC(P) = POWER((1+TI),ORD(P));

FACTOR(P) = 1 / FAC(P);

VAN(PR) = SUM(P,A(PR,P)*FACTOR(P));

DISPLAY FAC, FACTOR, VAN;
```

El fichero LST -completo- para este problema seria el siguiente:

```
GAMS Rev 116  Windows NT/95/98                    12/12/00 16:58:09  PAGE      1
VAN1


              Calculo del VAN de un proyecto de inversion

   7
   8
   9  SET      P /PER0*PER7/;
  10  SET      PR/PRO1*PRO2/;
  11
  12  PARAMETER        VAN(PR);
  13  PARAMETER        FAC(P);
  14  PARAMETER        FACTOR(P);
  15
  16  SCALAR  TI       /.1/;
  17
  18  TABLE A(PR,P)
  19           PER0     PER1     PER2     PER3     PER4     PER5     PER6     PER7
  20  PRO1     -100     10       15       20       30       40       15       50
  21  PRO2     -1000    150      250      350      450      500      0        0;
  22
  23
  24  FAC(P) = POWER((1+TI),ORD(P));
  25
  26  FACTOR(P) = 1 / FAC(P);
  27
  28  VAN(PR) = SUM(P,A(PR,P)*FACTOR(P));
  29
  30  DISPLAY FAC, FACTOR, VAN;
  31


COMPILATION TIME     =        0.390 SECONDS    0.7 Mb      WIN194-116

GAMS Rev 116  Windows NT/95/98                    12/12/00 16:58:09  PAGE      2
VAN1
E x e c u t i o n


----     30 PARAMETER FAC

PER0 1.100,    PER1 1.210,    PER2 1.331,    PER3 1.464,    PER4 1.611
PER5 1.772,    PER6 1.949,    PER7 2.144


----     30 PARAMETER FACTOR

PER0 0.909,    PER1 0.826,    PER2 0.751,    PER3 0.683,    PER4 0.621
PER5 0.564,    PER6 0.513,    PER7 0.467


----     30 PARAMETER VAN

PRO1  14.515,    PRO2 203.411


EXECUTION TIME       =        0.220 SECONDS    1.4 Mb      WIN194-116


USER: GAMS Development Corporation, Washington, DC   G871201:0000XX-XXX
      Free Demo,  202-342-0180,  sales@gams.com,  www.gams.com   DC9999

**** FILE SUMMARY

INPUT      C:\RAMON\PROVALLIBRE\1402.GMS
OUTPUT     C:\RAMON\GMS\1402.LST
```

GRUPO ***VARIABLES***

El bloque de variables es un bloque obligatorio, y por ello vamos a remitirnos a lo expuesto con anterioridad sobre los bloques obligatorios, en cuanto a la forma de definición de las variables, la clase de variable, etc.

En este caso solo haremos referencias a como introducir algunas de las etiquetas sobre las variables, cuando estas están indexadas. Por ejemplo, si las variables X(J) son variables positivas, previamente las habremos declarado como tales, pero la variable X(2) tiene una cota superior ,mientras que las restantes variables no estan acotadas superiormente.

Lo anterior se puede escribir como sigue:

X.UP(“2”) = 100. ;

esto nos indica que la cota superior de solamente la variable X(2) es 100.

De la misma forma se puede proceder con el resto de etiquetas y conjuntos. Por ejemplo, la expresión:

X.L(I,”5”) = 1.;

nos indica un punto de partida para todas las variables de dos argumentos siempre que el segundo argumento sea 5, es decir, para las variables X(1,5), X(2,5), X(3,5), etc..

Las líneas correspondientes del problema lineal a variables asociadas con índices sería:

```
VARIABLES

X(J)    VARIABLES PRINCIPALES
F;

POSITIVE VARIABLES X;
```

Aquí podemos observar que no se han definido las variables auxiliares -variables de holgura- asociadas a cada una de las restricciones para que nos indiquen de forma inmediata la diferencia respecto al término independiente. Esta información se puede obtener las analizar en la solución el comportamiento de las ecuaciones, en este caso simplemente por diferencia entre el valor de la restricción (LEVEL) y el valor del término independiente o cota superior de la restricción (UPPER).

GRUPO ***EQUATIONS***

Se trata de otro grupo también obligatorio, y por ello, los fundamentos básicos de funcionamiento de este bloque ya los hemos referido con anterioridad. La única diferencia que hemos de hacer notar a la hora de declarar las ecuaciones asociadas con índices, es que debemos definirlas con el índice correspondiente, tal y como era de esperar siempre que aparezcan índices.

Como ya conocemos las ecuaciones que podemos definir son de dos clases: ecuaciones indexadas (dependientes de algún/algunos índices) y las ecuaciones no indexadas.

En el ejemplo del problema lineal al que nos estamos refiriendo, tenemos las dos clases de ecuaciones, las no indexadas (función objetivo) y las indexadas (ecuaciones de restricción).

```
EQUATIONS

OBJ     FUNCION OBJETIVO
R(I)    RESTRICCIONES;

OBJ.. F =E= SUM(J,C(J)*X(J));

R(I).. SUM(J,A(I,J)*X(J))  =G= B(I);
```

En este ejemplo podemos observar que las restricciones están asociadas a un índice I, y por tanto existirán tantas restricciones como elementos tenga el conjunto I

La definición de las dos ecuaciones. función objetivo y restricciones se hace de la misma forma que la expresión matricial de un problema lineal.

Así la expresión de la función objetivo lineal: $F = \sum_{j} c_j \, x_j$ se indica de la forma: SUM(J,C(J)*X(J)) es decir la suma para las distintas variables (J) de los productos del coeficiente por el valor de la variable. Como podemos observar esta ecuación es una ecuación no indexadas, es decir, que el resultado es un “escalar” que no depende de ningún índice.

Las ecuaciones indexadas, como la expresión de las distintas restricciones (I) del problema , que en términos matemáticos seria: $\sum_{j} a_{ij} \, x_j \leq b_i \; , \forall i$, se corresponde con la expresión: SUM(J,A(I,J)*X(J)) =L= B(I), es importante resaltar que estamos sumando para todas y cada una de las restricciones, por tanto el índice a usar en el operador SUM es el de J, y no el de I, ya que este se corresponde con el total de restricciones. Es importante resaltar que el nombre de las restricciones tiene que estar asociado al índice (en este caso R(I)), y en la expresión matemática de la ecuación los dos miembros deben estar referidos al mismo índice, es decir, el termino de la izquierda debe tener como índice “libre” el I, y el termino de la derecha debe estar referido al mismo índice I.

Dentro de este bloque también es posible establecer operaciones como las que hemos visto en el grupo DATA, tanto en lo que se refiere a las operaciones con funciones como con el operador condicional para las expresiones relacionales.

Es importante hacer notar que el uso de funciones modifica el tipo de problema a resolver, así por ejemplo si usamos el operador ABS - valor absoluto - el problema se convierte en un problema no lineal y no *smooth*, y para ello tendremos que usar un solver del tipo DNLP. En otros casos no es posible usar ciertas funciones, como por ejemplo: CEIL, FLOOR, NORMAL, SIGN, etc. en el bloque de funciones ya que hacen imposible la resolución de los problemas con los solvers de GAMS.

GRUPOS ***MODEL & SOLVE***

Como ya sabemos se trata de dos grupos obligatorios, y por tanto lo único que podemos referir en este caso son el uso de diferentes modelos dentro de un mismo problema, y los diferentes solvers que existen para GAMS.

En el caso de tener que resolver un mismo problema varias veces con un pequeña variación el alguno de los datos, esto se pueden incorporar en el mismo fichero. Supongamos que queremos resolver un problema de optimización con una única restricción y para diferentes valores del termino independiente. En este caso lo que deberemos hacer es definir tantas restricciones como términos independientes (evidentemente, con distintos nombres) y a la hora de definir el modelo incorporar cada vez una de las restricciones.

Por ejemplo:

Supongamos que una restricción es : $4\ x + 5\ y \leq 50$, y hemos de resolver el problema para los siguientes términos independientes: 60, 75, 90y 120.

Para ello definiremos en el bloque de ecuaciones, además de OBJ (función objetivo) las restricciones: RO, RM1, RM2, RM3 y RM4. Las restricciones las declaramos como sigue:

```
RO.. 4*X + 5*Y =L= 50;
RM1.. 4*X + 5*Y =L= 60;
RM2.. 4*X + 5*Y =L= 75;
RM3.. 4*X + 5*Y =L= 90;
RM4.. 4*X + 5*Y =L= 120;
```

Una vez declaradas y definidas las restricciones definimos dentro del bloque de modelo cinco nombres diferentes para el modelo, como sigue:

```
MODEL    MODORIG     /OBJ, RO/
         MODMOD1     /OBJ, RM1/
         MODMOD2     /OBJ, RM2/
         MODMOD3     /OBJ, RM3/
         MODMOD4     /OBJ, RM4/
```

Como ya conocemos, una vez declarados los modelos hemos de proceder a su solución dentro del bloque SOLVE, con las instrucciones habituales

```
SOLVE MODORIG USING NLP MAXIMIZING F;
SOLVE MODMOD1 USING NLP MAXIMIZING F;
```

SOLVE MODMOD2 USING NLP MAXIMIZING F;

SOLVE MODMOD3 USING NLP MAXIMIZING F;

SOLVE MODMOD4 USING NLP MAXIMIZING F;

Ya conocemos algunos de los tipos de modelos que podemos resolver: programas lineales (LP), problemas no lineales (NLP) y problemas enteros (MIP).

No obstante, GAMS dispone de la posibilidad de poder resolver otros tipos de modelos, aunque hay que advertir que no todos los solvers están disponibles para los programas de uso estándar.

DNLP	Problemas no lineales con discontinuidades.
RMIP	Problemas enteros donde se puede relajar la condición de integridad .
MIDNLP	Problemas enteros mixtos no lineales con derivadas discontinuas.
RMIDNLP	Problemas enteros mixtos no lineales con derivadas discontinuas en los que se puede relajar la condición de integridad.

No obstante conviene advertir que para algunos de estos problemas los algoritmos de solución carecen de la seguridad de proporcionar buenas soluciones ya que algunos se encuentran en fases de experimentación. Para mayores detalles puede consultarse la Guía de Usuario de GAMS.

Antes pasar al grupo de options, hemos de recordar que podemos estudiar los grupos de DISPLAY y OUTPUT, pero con las nociones elementales que ya hemos visto será suficiente para el manejo habitual de los programas.

GRUPO ***OPTIONS***

El programa GAMS permite varias opciones para poder mejorar el funcionamiento de los algoritmos, tanto no lineales como enteros. Estas opciones, excepto las propias del solver se incorporan dentro del fichero GMS inicializadas como OPTIONS. Las opciones del solver se incorporan como un fichero *.OPT, en donde se indican las opciones propias del solver.

Las opciones en GAMS pueden afectar a varios elementos:

- Detalle del output
- Parámetros del solver
- Opciones propias del solver
- Control del input

Así por ejemplo para especificar detalles del output tenemos:

DECIMALS

Significa el numero de decimales que se especifican para la solución. Valor por defecto 3. Se puede especificar la opción entre 0 y 8.

LIMCOL

Proporciona en el fichero LST el listado las variables (columns). Valor por defecto 3.

Cuando no se tenga la seguridad de haber escrito bien todas las variables, se recomienda modificar su valor para visionar todas las variables.

LIMROW

Proporciona en el fichero LST el listado las ecuaciones (rows). Valor por defecto es 3.

Cuando no se tenga la seguridad de haber escrito bien todas las ecuaciones, se recomienda modificar su valor para visionar todas las ecuaciones.

SOLPRINT

Controla la impresión de la solución. Las opciones son ON y OFF. Por defecto ON. La opción ON imprime la solución de una columna o una fila en cada línea. La opción OFF, no imprime la solución.

SOLVEOPT

Controla la vía por la cual son devueltos los valores de la solución a GAMS. Las opciones son: MERGE o REPLACE. Por defecto MERGE. La opción MERGE combina los nuevos y los antiguos valores. La opción REPLACE los antiguos valores de una variable o ecuación son inicializados con los valores por defecto antes de que los valores de la nueva solución sean devueltos.

SYSOUT

Controla las opciones de impresión Las opciones posibles son ON y OFF. El valor por defecto es OFF. Con la opción ON se imprime en el fichero las opciones del solver. Con OFF, ello no ocurre.

Una explicación más amplia de las diferentes opciones puede encontrarse en la Guía del Usuario.

Para finalizar incorporamos el planteamiento completo del problema lineal que hemos venido usando para poder explicar algunos de elementos del programa GAMS

Opciones de parámetros del solver

BRATIO

La aceptación o no de una base en los problemas lineales o no lineales, depende de esta opción. El valor por defecto de esta opción es 0.25. Los valores admisibles son los reales entre cero y uno.

El uso de una base es rechazado si el numero de variables básicas es menor que (0.25 = valor opción) veces el tamaño de la base. Para la opción 0, se acepta cualquier base, mientras que para la opción 1, se rechaza siempre la base.

Esta opción no es aplicable en GAMS/ZOOM.

DOMLIM

Establece el numero de errores en la función o en las derivadas en un subsistema que se va a resolver antes de detener el proceso. El valor por defecto es 0. No conviene alterar este numero.

ITERLIM

Establece el numero máximo de iteraciones que puede realizar GAMS, antes de detenerse. Valor por defecto 1000

OPTCA

Es el criterio de optimalidad absoluta para problemas enteros. El proceso de solución se detiene cuando el criterio de solución asegura que la distancia entre la solución global y la actual es de OPTCA veces. El valor por defecto de esta opción es 0.

OPTCR

Es el criterio de optimalidad relativa de las soluciones enteras. EL proceso de solución se detiene si se puede garantizar que la mejor solución encontrada no dista mas de 100*OPTCR de la solución global. El valor por defecto es 0.1

Puede tomar valores reales entre 0 y 1. Siempre es conveniente reducir esta opción, aunque ello significa aumentar el numero de iteraciones del branch and bound.

RESLIM

Establece el limite máximo de recursos usados por el sistema. Superado este limite se detiene el programa. El valor por defecto de esta opción es 1000. Se puede modificar este valor, admitiendo cualquier valor real positivo.

Opciones propias del solver:

Los diferente solvers tienen susporpias opciones para aceleración y control del algoritmos. A efectos ilustartivos usaremo dos ejemplos de ficheros de opciones: MINOS.OPT y OSL.OPT

Un ejemplo de fichero de opciones MINOS5, es:

```
BEGIN GAMS/MINOS options
Scale 2
Linesearch tolerance 0.9
Iterations limit 100000
END GAMS/MINOS options
```

La explicación es:

`Scale 2:` Significa que se considere a todas las variables y restricciones con la misma magnitud de los coeficientes (escalamiento).

`Linesearch tolerance 0.9`: Exactitud de la longitud del paso de búsqueda de un punto a otro.

`Iterations limit 100000`: Establece el numero máximo de iteraciones a realizar.

Un ejemplo de fichero de opciones OSL, es:

```
*method interior3
bbpreproc 2
strategy 8
```

`* method interior3`: Uso de un método interior, de barrera predictor-corrector dual.

`bbpreproc 2`: Uso del método de Branch and Bound con copia de la matriz en cada iteración

`strategy 8`: Usa la información de los precios duales (método heurístico) para determinar la variable a ramificar en cada paso.

Para especificar opciones que afectan al input:

SEED

Inicializa la semilla generadora de números pseudo-aleatorios. Valor por defecto es 3141.

SOLVEOPT

Controla la vía por la cual son devueltos los valores de la solución a GAMS. Las opciones son: MERGE o REPLACE. Por defecto MERGE. La opción MERGE combina los nuevos y los antiguos valores. La opción REPLACE los antiguos valores de una variable o ecuación son inicializados con los valores por defecto antes de que los valores de la nueva solución sean devueltos.

Una explicación más amplia de las diferentes opciones puede encontrarse en la Guía del Usuario.

Para finalizar incorporamos el planteamiento completo del problema lineal que hemos venido usando para poder explicar algunos de elementos del programa GAMS

```
$ONTEXT

Se trata de introducir los datos en forma matricial, para poderlo
resolver mediante un programa lineal.

        Min F(x) = 3*x1 + 2*x2 +x3 + 2*x4 +3*x5
        s.a:
               2*x1 + 5*x2 + x4 + x5 >= 6
               4*x2 -2*x3 +2*x4 + 3*x5 >= 5
               x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 >= 7
              x1, x2, x3, x4, x5 no negativas

$OFFTEXT
SET
        J       VARIABLES    / 1*5/
        I       RESTRICCIONES/ 1*3/;
PARAMETERS
C(J)    COEF F /1      3.
                2      2.
                3      1.
                4      2.
                5      3./
B(I)    RHS    /1      6.
                2      5.
                3      7/;

TABLE   A(I,J)  MATRIZ
                       1       2       3       4       5
                1      2       5       0       1       1
                2      0       4       -2      2       3
                3      1       -6      3       7       5;
VARIABLES
X(J)    VARIABLES PRINCIPALES
F;
POSITIVE VARIABLES X;

EQUATIONS
OBJ     FUNCION OBJETIVO
R(I)    RESTRICCIONES;

OBJ.. F =E= SUM(J,C(J)*X(J));
R(I).. SUM(J,A(I,J)*X(J))  =G= B(I);
MODEL LINMATR2/ALL/;
SOLVE LINMATR2 USING LP MINIMIZING F;
```

La solución a este problema, así como la representación de todas las ecuaciones de las restricciones, es el siguiente:

```
GAMS Rev 116  Windows NT/95/98                    12/12/00 17:04:30  PAGE      1
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
C o m p i l a t i o n

      Se trata de introducir los datos en forma matricial, para
poderlo
      resolver mediante un programa lineal.

              Min F(x) = 3*x1 + 2*x2 +x3 + 2*x4 +3*x5
              s.a:
                     2*x1 + 5*x2 + x4 + x5 >= 6
                     4*x2 -2*x3 +2*x4 + 3*x5 >= 5
                     x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 >= 7
                    x1, x2, x3, x4, x5 no negativas

  14  SET
  15          J       VARIABLES    / 1*5/
  16          I       RESTRICCIONES/ 1*3/;
  17  PARAMETERS
  18  C(J)    COEF F /1      3.
  19                  2      2.
  20                  3      1.
  21                  4      2.
  22                  5      3./
  23  B(I)    RHS    /1      6.
  24                  2      5.
  25                  3      7/;
  26
  27  TABLE   A(I,J)  MATRIZ
  28                          1       2       3       4       5
  29                  1       2       5       0       1       1
  30                  2       0       4       -2      2       3
  31                  3       1       -6      3       7       5;
  32  VARIABLES
  33  X(J)    VARIABLES PRINCIPALES
  34  F;
  35  POSITIVE VARIABLES X;
  36
  37  EQUATIONS
  38  OBJ     FUNCION OBJETIVO
  39  R(I)    RESTRICCIONES;
  40
  41  OBJ.. F =E= SUM(J,C(J)*X(J));
  42  R(I).. SUM(J,A(I,J)*X(J))  =G= B(I);
  43  MODEL LINMATR2/ALL/;
  44  SOLVE LINMATR2 USING LP MINIMIZING F;
  45

COMPILATION TIME     =        0.280 SECONDS    0.7 Mb      WIN194-116
======================================================================
GAMS Rev 116  Windows NT/95/98                    12/12/00 17:04:30  PAGE      2
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
Equation Listing    SOLVE LINMATR2 USING LP FROM LINE 44

---- OBJ  =E=  FUNCION OBJETIVO

OBJ..  - 3*X(1) - 2*X(2) - X(3) - 2*X(4) - 3*X(5) + F =E= 0 ;(LHS = 0)
```

```
---- R  =G=  RESTRICCIONES

R(1)..  2*X(1) + 5*X(2) + X(4) + X(5) =G= 6 ; (LHS = 0, INFES = 6 ***)

R(2)..  4*X(2) - 2*X(3) + 2*X(4) + 3*X(5) =G= 5 ; (LHS = 0, INFES = 5
***)

R(3)..  X(1) - 6*X(2) + 3*X(3) + 7*X(4) + 5*X(5) =G= 7 ; (LHS = 0,
INFES = 7 ***)

======================================================================
GAMS Rev 116  Windows NT/95/98                12/12/00 17:04:30  PAGE      3
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
Column Listing      SOLVE LINMATR2 USING LP FROM LINE 44

---- X  VARIABLES PRINCIPALES

X(1)
                (.LO, .L, .UP = 0, 0, +INF)
       -3       OBJ
        2       R(1)
        1       R(3)

X(2)
                (.LO, .L, .UP = 0, 0, +INF)
       -2       OBJ
        5       R(1)
        4       R(2)
       -6       R(3)

X(3)
                (.LO, .L, .UP = 0, 0, +INF)
       -1       OBJ
       -2       R(2)
        3       R(3)

REMAINING 2 ENTRIES SKIPPED

---- F

F
                (.LO, .L, .UP = -INF, 0, +INF)
        1       OBJ
======================================================================
GAMS Rev 116  Windows NT/95/98                12/12/00 17:04:30  PAGE      4
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
Model Statistics    SOLVE LINMATR2 USING LP FROM LINE 44


MODEL STATISTICS

BLOCKS OF EQUATIONS       2     SINGLE EQUATIONS        4
BLOCKS OF VARIABLES       2     SINGLE VARIABLES        6
NON ZERO ELEMENTS        19

GENERATION TIME      =        0.380 SECONDS    1.4 Mb      WIN194-116

EXECUTION TIME       =        0.380 SECONDS    1.4 Mb      WIN194-116
```

```
GAMS Rev 116  Windows NT/95/98                    12/12/00 17:04:30  PAGE      5
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m

               S O L V E      S U M M A R Y

     MODEL   LINMATR2            OBJECTIVE  F
     TYPE    LP                  DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  44

**** SOLVER STATUS     1 NORMAL COMPLETION
**** MODEL STATUS      1 OPTIMAL
**** OBJECTIVE VALUE               5.1707

 RESOURCE USAGE, LIMIT          1.920     1000.000
 ITERATION COUNT, LIMIT         2        10000

GAMS/Cplex    Aug  7, 2000 WIN.CP.NA 19.4 016.015.038.WAT For Cplex 6.6
Cplex 6.6.1, GAMS Link 16, Using a GAMS/Cplex demo license installed at
runtime.

Optimal solution found.

Objective :           5.170732

                       LOWER     LEVEL     UPPER    MARGINAL

---- EQU OBJ             .         .         .        1.000

  OBJ  FUNCION OBJETIVO

---- EQU R  RESTRICCIONES

     LOWER     LEVEL     UPPER    MARGINAL

1     6.000     6.000     +INF      0.634
2     5.000     6.878     +INF       .
3     7.000     7.000     +INF      0.195

---- VAR X  VARIABLES PRINCIPALES

     LOWER     LEVEL     UPPER    MARGINAL

1      .         .        +INF      1.537
2      .       0.854      +INF       .
3      .         .        +INF      0.415
4      .       1.732      +INF       .
5      .         .        +INF      1.390

                       LOWER     LEVEL     UPPER    MARGINAL

---- VAR F              -INF      5.171     +INF       .

**** REPORT SUMMARY :        0     NONOPT
                             0 INFEASIBLE
                             0  UNBOUNDED
```

```
GAMS Rev 116  Windows NT/95/98                    12/12/00 17:04:30  PAGE      6
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m

EXECUTION TIME       =        0.160 SECONDS    0.7 Mb      WIN194-116

USER: GAMS Development Corporation, Washington, DC   G871201:0000XX-
XXX
      Free Demo,  202-342-0180,  sales@gams.com,  www.gams.com
DC9999

**** FILE SUMMARY

INPUT      C:\RAMON\1403.GMS
OUTPUT     C:\RAMON\1403.LST
```

Solamente se han representado tres variables, ya que no se ha modificado la opción LIMCOL, y toma el valor por defecto de 3, tampoco se ha modificado LIMROW, pero en este caso solo hay tres restricciones, y por ello se visualizan todas.

Vamos a presentar a continuación un ejercicio de aplicación, con el fin de ilustrar el proceso de modelización y optimización de un problema "real", así como las diferentes fases que conlleva este proceso hasta la determinación de la solución global general, e indicar la forma de construir un fichero de datos (fichero GMS) que permite resolver el problema con GAMS, vamos a realizarlo sobre un problema particular, para posteriormente ir analizando las diferentes variantes que sobre este problema se pueden introducir.

El servicio de bomberos de la ciudad de Valencia, necesita dar cobertura a las posibles necesidades de actuación mediante la disposición de un numero de bomberos cada día de la semana. Las necesidades diarias (estimadas en función de la experiencia del Jefe de Bomberos) son las siguientes: Lunes, 47 personas. Martes, 53 personas. Miércoles, 65 personas. Jueves, 59 personas. Viernes, 44 personas. Sábados, 46 personas. Domingos, 31 personas.

Para poder atender el servicio dispone de bomberos de plantilla, que trabajan durante cinco días a la semana de forma continuada y descansan dos días por semana, es decir, es posible trabajar de lunes a viernes y descansar sábado y domingo, o bien trabajar de martes a sábado y descansar domingo y lunes, etc.

Se pide:

Determinar el numero de trabajadores que atienden el servicio, de forma que el numero total sea mínimo.

En primer lugar debemos analizar el problema, es decir, que elementos lo componen, asi como llegar a determinar las variables, las restricciones, la función objetivo, etc.

Se trata de un problema de dar cobertura a un servicio que requiere la presencia de una serie de trabajadores durante cada día de la semana (numero prefijado) y de forma que el numero total de trabajadores sea el menor posible.

Un forma irreflexiva de abordar el problema seria la siguiente:

Denominar x_i (i = lunes, martes,..., domingo) a la variable que representar el numero de trabajadores que cubren el servicio cada día, con lo que las restricciones que deben cumplirse son:

$$x_i \geq n_i \qquad n_i = \text{numero de trabajadores necesarios}$$

Con esto tendríamos determinadas las restricciones del problema[2], y ahora quedaría definir la función objetivo y las variables del problema.

Con relación a la función objetivo, y dado que se plantear el utilizar el menor numero de trabajadores, la función será:

$$\text{Min } F(x) = \sum_{i=1}^{7} x_i$$

La variables x_i deberan ser enteras (imposibilidad de trabajar una fracción de bombero) y no negativas, con lo cual el problema, ya modelizado en GAMS, será:

```
OPTION LIMROW = 100;
SET     I        /LU,MA,MI,JU,VI,SA,DO/;
ALIAS   (I,J);
PARAMETER        B(I)
        /LU      47
         MA      53
         MI      65
         JU      59
         VI      44
         SA      46
         DO      31 /;
VARIABLES
X(J),Z;
INTEGER VARIABLES X(J);
EQUATIONS
OBJ,R(I);
OBJ.. Z =E= SUM(J, X(J) );
R(I).. X(I) =G= B(I);
MODEL SEIS /ALL/;
SOLVE SEIS USING MIP MINIMIZING Z;
DISPLAY X.L, Z.L;
```

[2] Con posterioridad se discutirá si la restricción es una igualdad o una desigualdad, y que implicaciones tiene esto sobre la solución del problema.

Las ecuaciones que genera este problema son las siguientes:

```
---- OBJ  =E=

OBJ..  - X(LU) - X(MA) - X(MI) - X(JU) - X(VI) - X(SA) - X(DO) + Z =E= 0 ;
      (LHS = 0)

---- R  =G=

R(LU)..  X(LU) =G= 47 ; (LHS = 0, INFES = 47 ***)

R(MA)..  X(MA) =G= 53 ; (LHS = 0, INFES = 53 ***)

R(MI)..  X(MI) =G= 65 ; (LHS = 0, INFES = 65 ***)

R(JU)..  X(JU) =G= 59 ; (LHS = 0, INFES = 59 ***)

R(VI)..  X(VI) =G= 44 ; (LHS = 0, INFES = 44 ***)

R(SA)..  X(SA) =G= 46 ; (LHS = 0, INFES = 46 ***)

R(DO)..  X(DO) =G= 31 ; (LHS = 0, INFES = 31 ***)
```

Como puede observarse se trata de la transcripción de las ecuaciones planteadas originalmente.

Con esto el resultado que se obtiene, tanto de como se satisfacen las restricciones, como el valor de las variables, es el siguiente:

a) Restricciones

```
---- EQU R

      LOWER     LEVEL     UPPER    MARGINAL

LU    47.000    47.000     +INF       .
MA    53.000    53.000     +INF       .
MI    65.000    65.000     +INF       .
JU    59.000    59.000     +INF       .
VI    44.000    44.000     +INF       .
SA    46.000    46.000     +INF       .
DO    31.000    31.000     +INF       .
```

b) Valor de las variables y de la función objetivo:

```
---- VAR X

      LOWER     LEVEL     UPPER    MARGINAL

LU      .       47.000   100.000     1.000
MA      .       53.000   100.000     1.000
MI      .       65.000   100.000     1.000
JU      .       59.000   100.000     1.000
VI      .       44.000   100.000     1.000
SA      .       46.000   100.000     1.000
DO      .       31.000   100.000     1.000
```

Como era previsible, las restricciones se satisfacen como igualdad, y el total de trabajadores necesarios para poder atender el servicio es de 345 trabajadores. Obviamente, este resultado es erróneo, y por ello cabe analizar el porque de este resultado.

La primera observación importante deriva de que no se ha tenido en cuenta que la plantilla trabaja cinco días de forma consecutiva y descansa dos días, con ello, parte de los trabajadores que trabajan el lunes, ya han trabajado otros días, con lo cual los 47 trabajadores del lunes no son diferentes del alguno de los 31 del domingo o 46 del sábado. Por lo tanto, el problema es un error de planteamiento del modelo original, por ello cabe replantearse la formulación matemática del problema.

Las variables del problema (x_i) no deben representar el numero de trabajadores que trabajan el día i, sino que cabe redefinirlas como el numero de trabajadores que *comienzan su semana laboral* el día i, y a partir de aquí estos trabajadores trabajan cinco días consecutivos.

Con esta nueva redefinición de las variables, vamos a plantear como serian las restricciones.

<u>Restricción de los trabajadores para el lunes:</u>

El lunes trabajaran los que comienzan su semana laboral el lunes. ¿Trabajaran los que comienzan el martes su semana?, la respuesta es no. ¿ Y los que comienzan el miércoles?, la respuesta sigue siendo no. ¿Y los del jueves?, en este caso la respuesta es SI, ya que los que comienzan a trabajar el jueves, para cumplir la semana de cinco días trabajaran: jueves, viernes, sábado, domingo y LUNES, es decir, el lunes será su quinto día de trabajo. De la misma forma, los que comienzan a trabajar los viernes, sábado y domingo.

Por tanto, formalmente la restricción será de la siguiente forma[3]:

$$x_L + x_J + x_V + x_S + x_D = 47$$

De igual forma se plantearían las restricciones para los restantes días de la semana. Así, por ejemplo, la restricción del jueves seria de la siguiente forma:

$$x_J + x_D + x_L + x_M + x_X = 59$$

Una vez, tenemos definidas las restricciones, la función objetivo y las variables deberíamos definirlas como en el problema anterior.

[3] Vamos a considerar que esta restricción sea de igualdad, ya que el numero de trabajadores que exceden de las necesidades mínimas, nos generan unos costes adicionales, y por ello, es preferible ajustar los trabajadores a las necesidades reales.

Con relación a la función objetivo, y dado que se plantear el utilizar el menor numero de trabajadores, la función será:

$$\text{Min } F(x) = \sum_{i=1}^{7} x_i$$

La variables x_i deberán ser enteras (imposibilidad de trabajar una fracción de bombero) y no negativas.

A la hora de plantear las restricciones del problema mediante GAMS tendremos el problema de como excluir dos de los subíndices en las variables. La forma más cómoda de incorporar este tipo de condiciones es a través de una matriz de transición entre las variables y las restricciones, de forma que la matriz A, solamente contenga 0 y 1, que significa que si la variable no esta en la restricción el elemento es 0, mientras que si lo esta el elemento sea un 1.

Esta matriz que proponemos para este caso, será una matriz de 7*7, es decir, de 7 filas (una por restricción) y 7 columnas (una por variable). La idea básica es que al multiplicar una matriz A por un vector X, el resultado es un vector :

$$A * X = R$$
$$(7*7)*(7*1) = (7*1)$$

Dado que el vector X, es el vector de variables (x_L, x_M, ..., x_D) el resultado R_L (restricción del lunes) se obtendrá de multiplicar la primera fila de la matriz A por el vector X, por tanto si queremos que R_L sea: $x_L + x_J + x_V + x_S + x_D$. La primera fila de A debe ser:

$$(1 \quad 0 \quad 0 \quad 1 \quad 1 \quad 1 \quad 1)$$

De la misma forma, se haría con las restantes restricciones, hasta llegar a que la matriz A tiene la siguiente forma:

$$A = \begin{pmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Una vez definida esta matriz de transición, solamente queda incorporarla a la hora de definir las restricciones del problema.

No obstante, como en la mayoría de los problemas "complejos" es conveniente realizar una versión preliminar para ver como evoluciona la solución antes de realizar la versión definitiva. Esta consideración, aplicada a este problema seria la de elaborar una versión LP, es decir, una versión con las variables no negativas , sin declararlas como enteras, que es muy fácil de resolver, simplemente para comprobar que las ecuaciones están correctamente definidas, y que no existen otros problemas adicionales.

Por tanto este problema se correspondería con el siguiente fichero de GAMS:

```
OPTION LIMROW = 100;
SET      I        /LU,MA,MI,JU,VI,SA,DO/;
ALIAS    (I,J);
PARAMETER         B(I)
        /LU       47
         MA       53
         MI       65
         JU       59
         VI       44
         SA       46
         DO       31 /;
TABLE    A(I,J)
        LU       MA       MI       JU       VI       SA       DO
LU      1                          1        1        1        1
MA      1        1                          1        1        1
MI      1        1        1                          1        1
JU      1        1        1        1                          1
VI      1        1        1        1        1
SA               1        1        1        1        1
DO                        1        1        1        1        1;
VARIABLES
X(J),Z;
POSITIVE VARIABLES X(J);
EQUATIONS
OBJ,R(I);
OBJ.. Z =E= SUM(J, X(J) );
R(I)..  SUM(J, A(I,J)*X(J) ) =E= B(I);
MODEL SEIS /ALL/;
SOLVE SEIS USING LP MINIMIZING Z;
DISPLAY X.L, Z.L;
```

Al ejecutar este fichero, nos tenemos que fijar, tanto en las restricciones como en la solución.

Las restricciones son las siguientes:

```
---- R  =E=

R(LU)..  X(LU) + X(JU) + X(VI) + X(SA) + X(DO) =E= 47 ; (LHS = 0,
INFES = 47 ***)

R(MA)..  X(LU) + X(MA) + X(VI) + X(SA) + X(DO) =E= 53 ; (LHS = 0,
INFES = 53 ***)

R(MI)..  X(LU) + X(MA) + X(MI) + X(SA) + X(DO) =E= 65 ; (LHS = 0,
INFES = 65 ***)

R(JU)..  X(LU) + X(MA) + X(MI) + X(JU) + X(DO) =E= 59 ; (LHS = 0,
INFES = 59 ***)

R(VI)..  X(LU) + X(MA) + X(MI) + X(JU) + X(VI) =E= 44 ; (LHS = 0,
INFES = 44 ***)

R(SA)..  X(MA) + X(MI) + X(JU) + X(VI) + X(SA) =E= 46 ; (LHS = 0,
INFES = 46 ***)

R(DO)..  X(MI) + X(JU) + X(VI) + X(SA) + X(DO) =E= 31 ; (LHS = 0,
INFES = 31 ***)
```

Podemos comprobar, que efectivamente están escritas correctamente las ecuaciones que queríamos representar. Ahora, vamos a fijarnos en la solución:

```
               S O L V E      S U M M A R Y

     MODEL   SEIS                OBJECTIVE  Z
     TYPE    LP                  DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  29

**** SOLVER STATUS     1 NORMAL COMPLETION
**** MODEL STATUS      4 INFEASIBLE
**** OBJECTIVE VALUE               10.0000

 RESOURCE USAGE, LIMIT          1.920     1000.000
 ITERATION COUNT, LIMIT         1        10000

Problem infeasible.

                       LOWER     LEVEL     UPPER    MARGINAL
---- EQU OBJ             .         .         .         EPS

---- EQU R

      LOWER     LEVEL     UPPER    MARGINAL
LU    47.000    47.000    47.000     0.400
MA    53.000    53.000    53.000    -0.600  NOPT
MI    65.000    65.000    65.000     0.400
JU    59.000    59.000    59.000     0.400
VI    44.000    44.000    44.000    -0.600  NOPT
SA    46.000    46.000    46.000     0.400
DO    31.000    31.000    31.000    -0.600  NOPT
```

```
---- VAR X

      LOWER      LEVEL      UPPER     MARGINAL
LU      .        18.000      +INF        .
MA      .        20.000      +INF        .
MI      .         2.000      +INF        .
JU      .        14.000      +INF        .
VI      .       -10.000      +INF      1.000 INFES
SA      .        20.000      +INF        .
DO      .         5.000      +INF        .

                       LOWER      LEVEL      UPPER     MARGINAL

---- VAR Z              -INF      69.000      +INF        .
```

Como puede observarse, se trata de una solución infactible, es decir, no existe ninguna solución que satisfaga todas las condiciones. Por ello, una alternativa es considerar, las restricciones: A * X = R como restricciones de desigualdad, es decir: A * X ≥ R. Con ello el nuevo fichero de datos seria el siguiente:

```
OPTION LIMROW = 100;
SET      I        /LU,MA,MI,JU,VI,SA,DO/;
ALIAS    (I,J);
PARAMETER         B(I)
        /LU      47
         MA      53
         MI      65
         JU      59
         VI      44
         SA      46
         DO      31 /;
TABLE    A(I,J)
        LU      MA      MI      JU      VI      SA      DO
LU      1                       1       1       1       1
MA      1       1                       1       1       1
MI      1       1       1                       1       1
JU      1       1       1       1                       1
VI      1       1       1       1       1
SA              1       1       1       1       1
DO                      1       1       1       1       1;
VARIABLES
X(J),Z;
POSITIVE VARIABLES X(J);
EQUATIONS
OBJ,R(I);
OBJ.. Z =E= SUM(J, X(J) );
R(I)..  SUM(J, A(I,J)*X(J) ) =G= B(I);
MODEL SEIS /ALL/;
SOLVE SEIS USING LP MINIMIZING Z;
DISPLAY X.L, Z.L;
```

La solución a este problema, analizando el cumplimiento de las restricciones y el valor de las variables, es el siguiente:

a) Cumplimiento de las restricciones:

```
---- EQU R

      LOWER     LEVEL     UPPER    MARGINAL

LU    47.000    47.000     +INF      0.333
MA    53.000    54.667     +INF        .
MI    65.000    65.000     +INF      0.333
JU    59.000    59.000     +INF      0.333
VI    44.000    59.000     +INF        .
SA    46.000    46.000     +INF      0.333
DO    31.000    31.000     +INF       EPS
```

Como puede observarse solamente la restricción del martes presenta una ligera desviación respecto al objetivo del numero necesario (diferencia entre LEVEL y LOWER):

Martes: 54.667 - 53 = 1.667

Esto representa que se cumplen todas las restricciones como igualdad, excepto la del martes que presenta una ligera desviación.

b) Valor de las variables:

```
---- VAR X

      LOWER     LEVEL     UPPER    MARGINAL

LU      .       26.333     +INF        .
MA      .       15.000     +INF        .
MI      .       10.333     +INF        .
JU      .        7.333     +INF        .
VI      .         .        +INF      0.333
SA      .       13.333     +INF        .
DO      .         .        +INF  -5.55E-17
```

La solución nos indica que solamente es necesario que los bomberos comiencen a trabajar cinco días a la semana ya que el viernes y el domingo están cubiertos con la gente de los otros días.

No obstante, podemos observar que se trata de una solución imposible ya que el lunes, por ejemplo, no pueden comenzar a trabajar 26.33 bomberos, por lo que esta solución no es valida, pero ,esta solución ya nos indica que es posible obtener una solución entera y para ello en el fichero de datos anterior solamente sustituyendo la condición de variables positivas por variables enteras encontraremos una solución entera.

El nuevo fichero es:

```
OPTION LIMROW = 100;
SET      I         /LU,MA,MI,JU,VI,SA,DO/;
ALIAS    (I,J);
PARAMETER          B(I)
         /LU       47
          MA       53
          MI       65
          JU       59
          VI       44
          SA       46
          DO       31 /;
TABLE    A(I,J)
         LU       MA       MI       JU       VI       SA       DO
LU       1                          1        1        1        1
MA       1        1                          1        1        1
MI       1        1        1                          1        1
JU       1        1        1        1                          1
VI       1        1        1        1        1
SA                1        1        1        1        1
DO                         1        1        1        1        1;
VARIABLES
X(J),Z;
INTEGER VARIABLES X(J);
EQUATIONS
OBJ,R(I);
OBJ.. Z =E= SUM(J, X(J) );
R(I)..  SUM(J, A(I,J)*X(J) ) =G= B(I);
MODEL SEIS /ALL/;
SOLVE SEIS USING MIP MINIMIZING Z;
DISPLAY X.L, Z.L;
```

Ejecutando este fichero , podemos comprobar las dos condiciones fundamentales de la solución, es decir, primero si existe solución (y si esta es optima) y en segundo lugar como se satisfacen las restricciones.

a) La satisfacción de las restricciones es la siguiente:

```
---- EQU R

      LOWER      LEVEL      UPPER     MARGINAL

LU    47.000     47.000      +INF         .
MA    53.000     55.000      +INF         .
MI    65.000     66.000      +INF         .
JU    59.000     59.000      +INF       1.000
VI    44.000     59.000      +INF         .
SA    46.000     47.000      +INF         .
DO    31.000     32.000      +INF         .
```

Como podemos observar algunas de las restricciones se satisfacen por exceso (diferencia entre LEVEL y LOWER):

Martes (Ma)	55 - 53 = 2
Miercoles (Mi)	66 - 65 = 1
Viernes (Vi)	59 - 44 = 15
Sabado (Sa)	47 - 46 = 1
Domingo (Do)	32 - 31 = 1

Como puede observarse las diferencias en estos días son notables (un total de 20), pero no obstante vamos a analizar como se comporta la solución.

b) Valor de las variables y de la función objetivo:

```
E x e c u t i o n

----      30 VARIABLE  X.L

LU 26.000,    MA 15.000,    MI 11.000,    JU  7.000,    SA 14.000

----      30 VARIABLE  Z.L                   =       73.000
```

La solución nos indica que son necesarios 73 bomberos para poder atender al servicio.

Si nos fijamos que clase de solución nos indica GAMS:

```
               S O L V E      S U M M A R Y

     MODEL   SEIS                OBJECTIVE  Z
     TYPE    MIP                 DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  29

**** SOLVER STATUS     1 NORMAL COMPLETION
**** MODEL STATUS      1 OPTIMAL
**** OBJECTIVE VALUE               73.0000

 RESOURCE USAGE, LIMIT          2.920     1000.000
 ITERATION COUNT, LIMIT         5        10000

GAMS/Cplex    Aug  7, 2000 WIN.CP.NA 19.4 016.015.038.WAT For Cplex 6.6
Cplex 6.6.1, GAMS Link 16, Using a GAMS/Cplex demo license installed at runtime.

Proven optimal solution.

MIP Solution  :           73.000000    (5 iterations, 0 nodes)
Final LP      :           73.000000    (0 iterations)

Best integer solution possible :          73.000000
Absolute gap          :                  0
Relative gap          :                  0
```

Una de las ventajas comentadas de GAMS es la posibilidad de resolución de varios problemas que tengan las misma estructura y que se diferencien en unos pocos datos. Anteriormente hemos visto como en un mismo fichero podíamos resolver dos problemas de maximización de la utilidad con dos términos independientes que se diferenciaban en una unidad, con el fin de ilustrar el significado del multiplicador de Lagrange.

Ahora vamos a exponer como resolver 10 problemas de las mismas características que el expuesto en el apartado de programación clásica, pero donde los términos independientes varían en una unidad, desde 130 hasta 139, y con un formato de salida que sea fácilmente interpretable.

Esto resulta sencillo usando las instrucciones de LOOP, así solamente cabe añadir al fichero anterior las instrucciones que aparecen en negrilla en la tabla 10.

```
OPTION LIMCOL = 0;OPTION LIMROW = 0;
SCALAR M        /130/;
VARIABLES
X, Y, U;
X.L=1;Y.L=1;
EQUATIONS
OBJ, RP;
OBJ.. U =E= (X+2)*(Y+1);
RP.. 4*X + 6*Y =L= M;
MODEL MAXUTIL/OBJ,RP/;
SOLVE MAXUTIL USING NLP MAXIMIZING U;
* INTRODUCCION DE UN BUCLE
SET RHS /1*10/;
PARAMETER TI(RHS)
/1       130
 2       131
 3       132
 4       133
 5       134
 6       135
 7       136
 8       137
 9       138
 10      139/;
OPTION SOLPRINT = OFF;
PARAMETER OUTPUT(*,RHS)
LOOP (RHS, M=TI(RHS);
           SOLVE MAXUTIL USING NLP MAXIMIZING U;
           OUTPUT("M",RHS)= M;
           OUTPUT("FUNCION",RHS) = U.L;
           OUTPUT("LAGRANGE",RHS) = RP.M;
           OUTPUT("X",RHS) = X.L;
           OUTPUT("Y",RHS) = Y.L;
          );
DISPLAY OUTPUT;
```

Con este bucle que permite resolver todos los problemas con una sola ejecución, las soluciones que obtenemos son:

```
E x e c u t i o n

----     40 PARAMETER OUTPUT

                 1           2           3           4           5           6

M          130.000     131.000     132.000     133.000     134.000     135.000
FUNCION    216.000     219.010     222.042     225.094     228.167     231.260
LAGRANGE     3.000       3.021       3.042       3.062       3.083       3.104
X           16.000      16.125      16.250      16.375      16.500      16.625
Y           11.000      11.083      11.167      11.250      11.333      11.417

      +          7           8           9          10

M          136.000     137.000     138.000     139.000
FUNCION    234.375     237.510     240.667     243.844
LAGRANGE     3.125       3.146       3.167       3.187
X           16.750      16.875      17.000      17.125
Y           11.500      11.583      11.667      11.750
```

En la tabla anterior podemos observar que para cada valor del parámetro M, la suma del valor de la función y del multiplicador de Lagrange, es aproximadamente igual al valor de la función para el parámetro siguiente. Así mismo, podemos observar los cambios que se producen en el valor del multiplicador, dado que se trata de una derivada que toma diferentes valores para cada uno de los puntos en que esta evaluada.

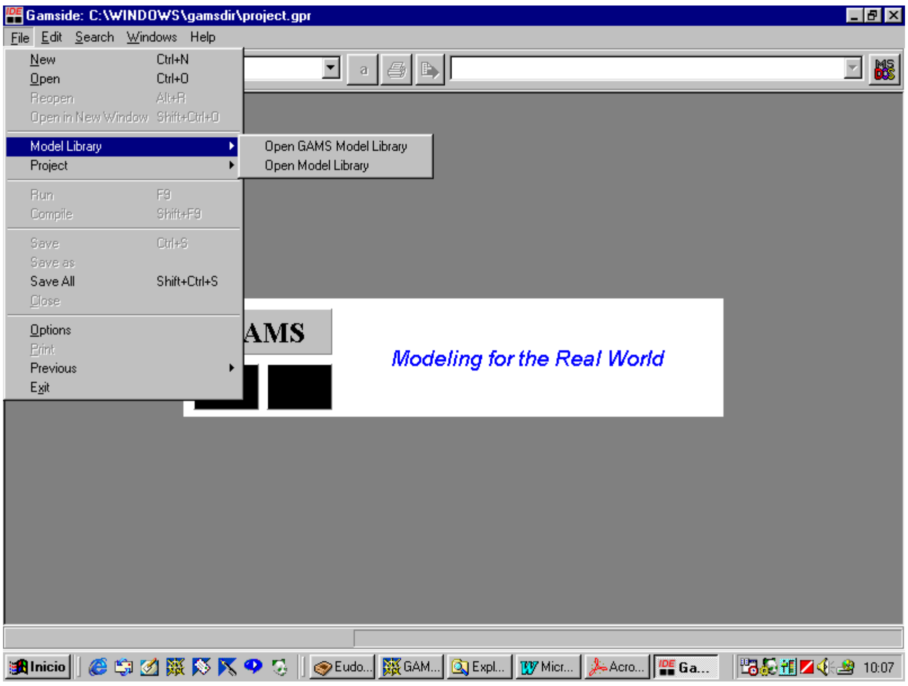
14.4.- OTRAS REFERENCIA CON GAMS

Las posibilidades GAMS no se agotan con la propio programa y los ficheros que podamos crear, así en este epígrafe vamos a comentar una serie de elemento a tener en cuenta con GAMS como son el uso de la librería de modelos que incorpora este texto o los que contiene el programa GAMS, como se puede trabajar con GAMS en al red Internet, y como podemos subsanar los errores que con frecuencia se cometen en la creación y resolución de los problemas en los ficheros de input.

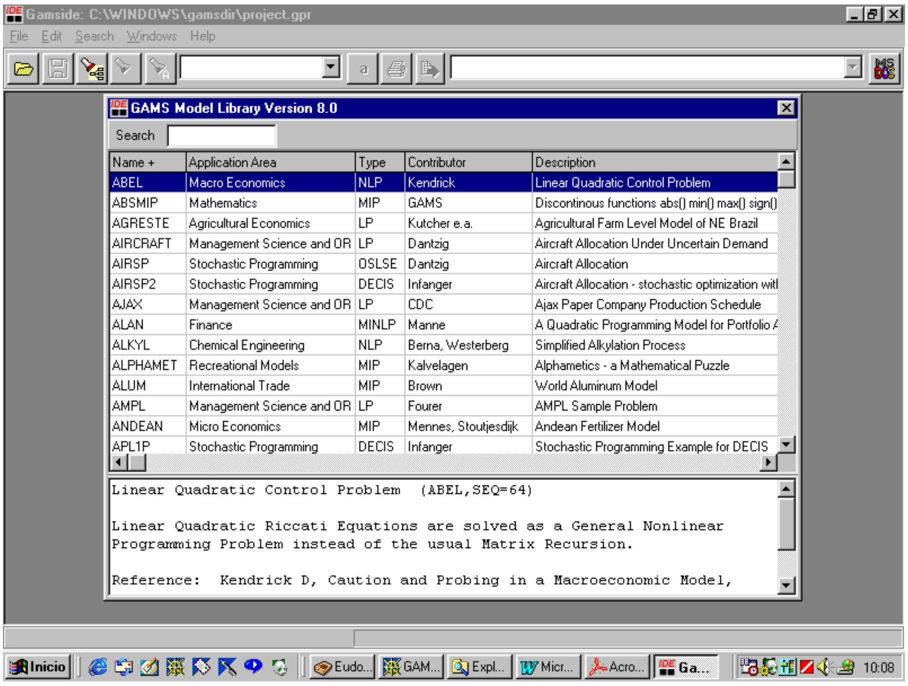
Librería de GAMS

Junto con al versión de GAMS se distribuye una librería de modelos, de diferentes tipos que pueden ayudar a construir un modelo y dar unas ideas de cómo introducir las características que deseemos del modelo.

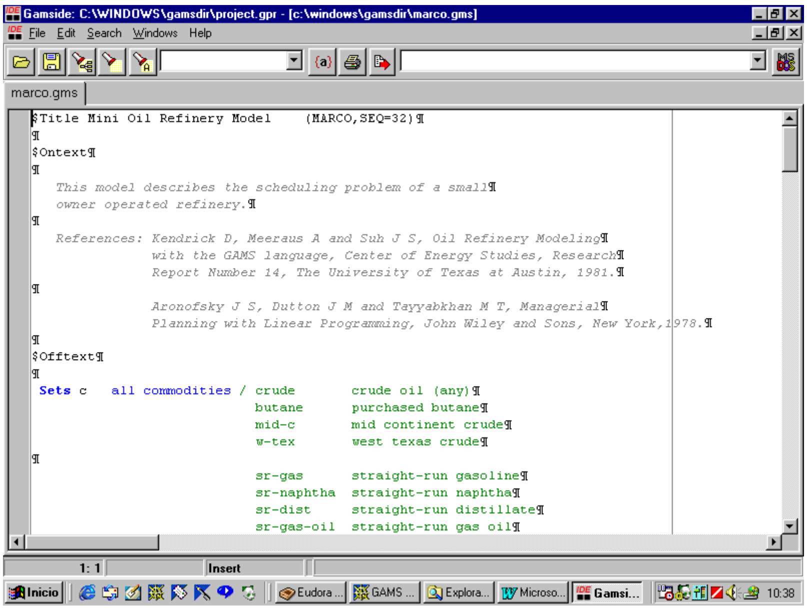
Para usar la librería, dentro de la opción **File-Model Library**:



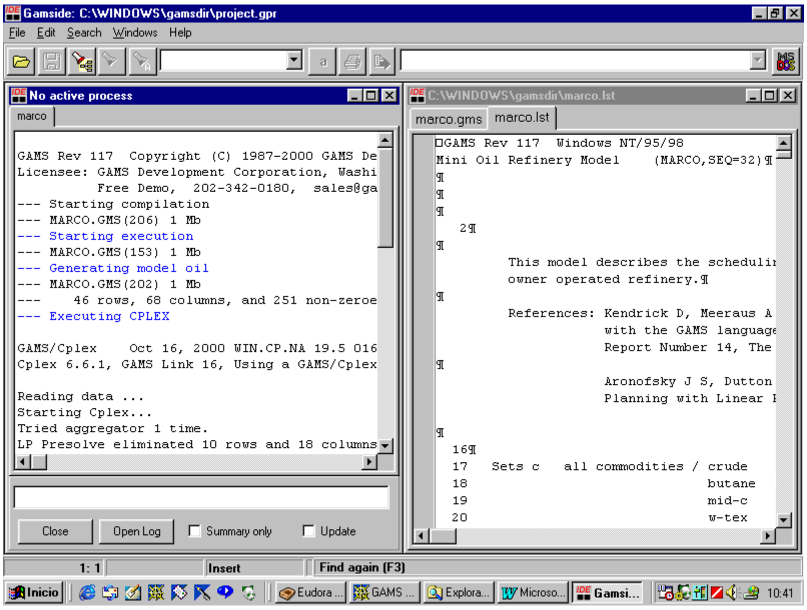
Con ello, obtenemos la siguiente descripción de los modelos:



Si elegimos cualquiera de ellos, por ejemplo el denominado MARCO, un modelo lineal simplificado de una refinería:



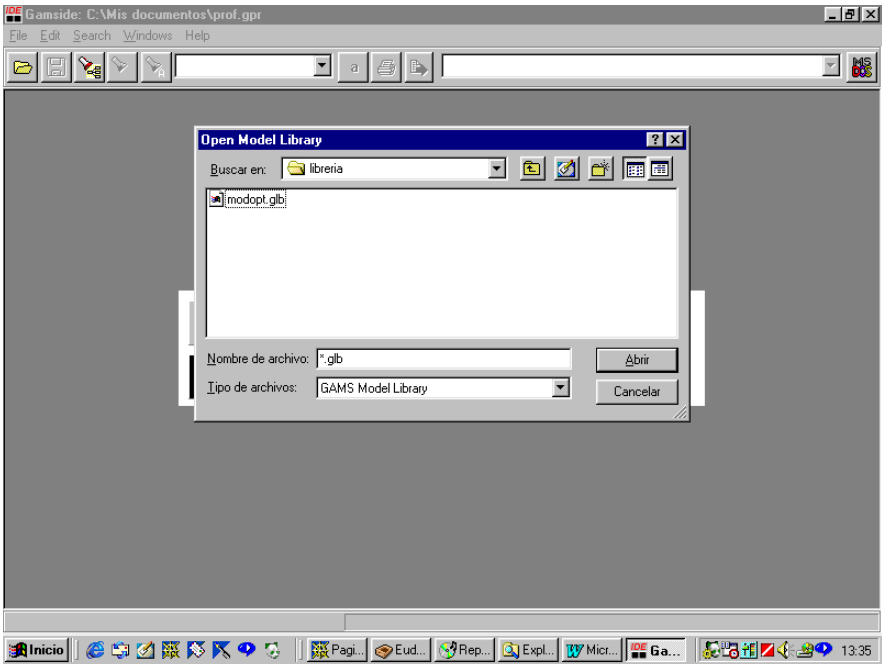
podemos obtener su solución de forma inmediata sin más que ejecutar el problema:



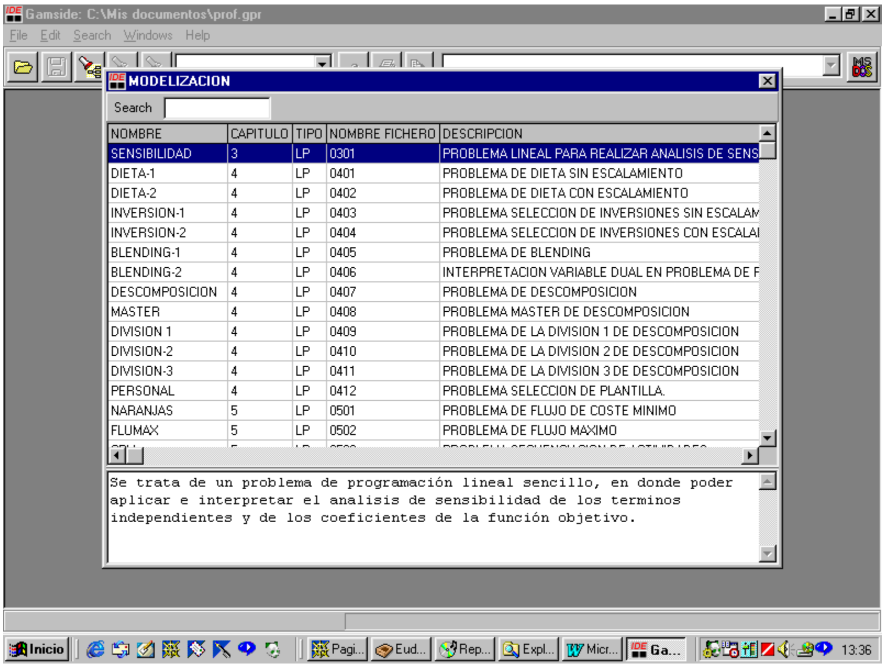
Podemos analizar su solución y su comportamiento como cualquier otro modelo.

Librería de ficheros de Modelización y Optimización.

Si por el contrario queremos usar una librería definida por el usuario, siguiendo el mismo proceso, pero al seleccionar USER LIBRARY tenemos que indicar cual es la dirección del fichero de librería (*.glb), es decir:

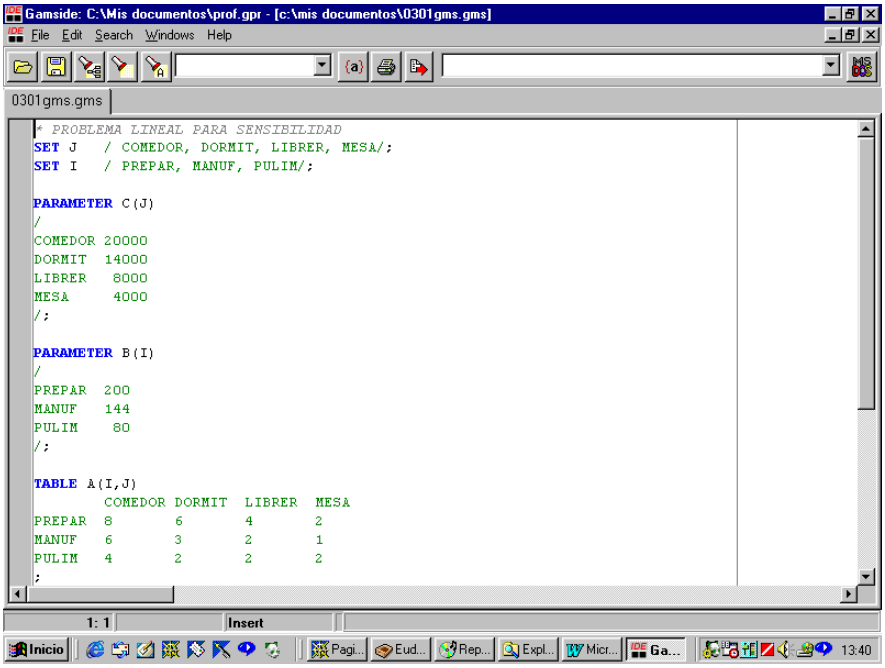


Si seleccionamos (buscar ubicación) la librería ***modopt.glb***, nos aparecerá relacionada toda la librería de modelos y ejemplos usado en modelización y optimización, que es la siguiente relación:



Junto al nombre aparece el capitulo al que pertenece el fichero, el nombre y una breve descripción. Algunos fichero incorporan en la parte inferior de la ventana una descripción más detallada.

Pulsando dos veces sobre la línea resaltada, es decir, sobre el fichero que nos interesa, este fichero se abre en el edito de GAMS, es decir:

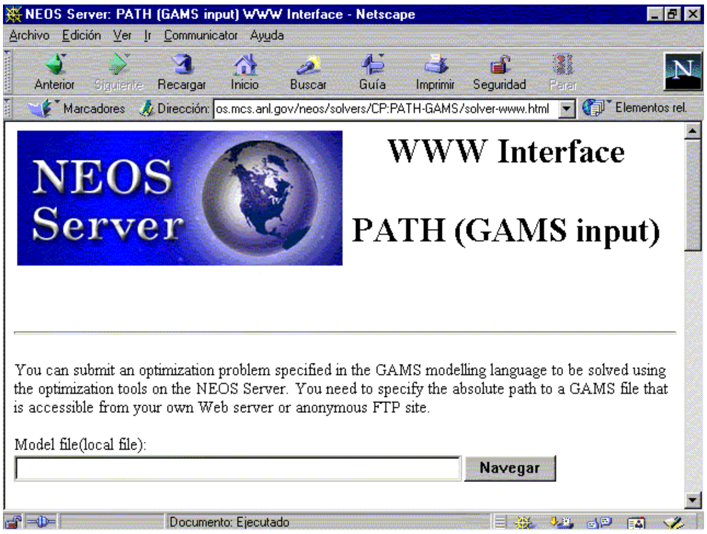


Este fichero ya esta en condiciones de ser ejecutado.

GAMS en Internet.

Cuando deseamos resolver un problemas de grandes dimensiones y las limitaciones de la versión que disponemos nos lo impiden, una de las alternativas es crear el fichero con el editor del GAMS-IDE, y una vez creado (con todas las opciones necesarias) se puede enviar vía web a la siguiente dirección:

http://www-neos.mcs.anl.gov/neos/solvers/CP:PATH-GAMS/solver-www.html



Se trata de una pagina del NEOS Server que admite la posibilidad de resolver en sus ordenadores (del tipo Sun Solaris) los ficheros GAMS que no es posible resolver con la versión de estudiante. La ventaja es que las instrucciones para crear los ficheros son las mismas en entorno Windows o UNIX, y por tanto la única diferencia es la forma de crear el fichero. El fichero es enviado, junto con el fichero de opciones (sí es necesario, este fichero de opciones es necesario si, por ejemplo, se quiere hacer una análisis de sensibilidad de los coeficientes del modelo). Junto con el fichero hay que adicionar una dirección de correo electrónico, en donde recibir el fichero solución.

La pagina de NEOS, permite además el enviar los ficheros vía e-mail o con has aplicaciones en JAVA (disponibles en el servidor como instalaciones cliente).

Una vez resuelto el programa, se recibe vía web (o mejor, vía e-mail) la solución en un fichero LST, tal como se hubiera resuelto en el ordenador local.

Esta es pues, una ventaja adicional de GAMS respecto de otros programas, pero al final debe ser el usuario-destinatario quien debe decidir la bondad o no este programa frente a otros de similares prestaciones y características. No hace falta reiterar que el que suscribe esta recensión es un usuario de este software, pero en todo caso, debe ser el usuario final el que debe analizar las ventajas e inconvenientes de este tipo de programas de ayuda a la docencia, y que si se pretende explicar o ayudar a la resolución de problemas de optimización clásica, no lineal, lineal y entera, hay pocos programas que permitan realizarlo con un mismo software, y además que sean de contrastada solvencia y garantía.

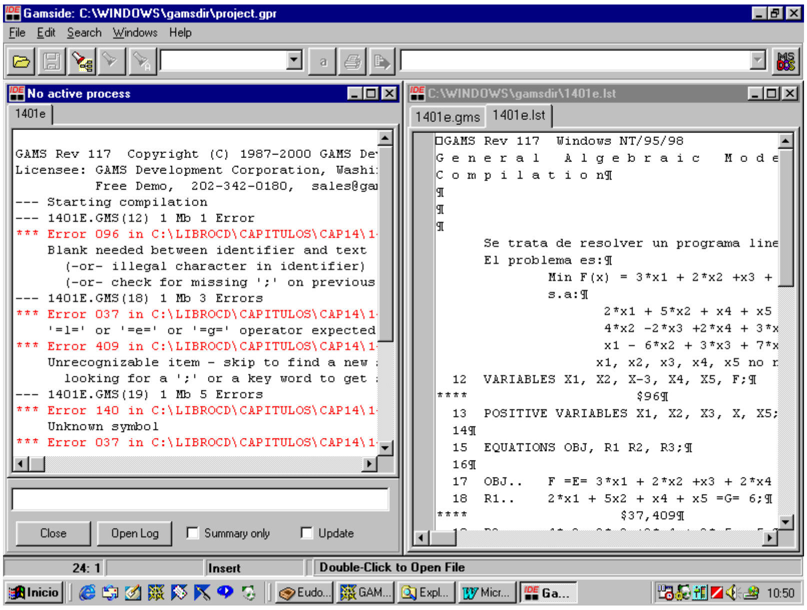
Corrección de errores en GAMS

Una de las practicas más comunes, al menos las primeras veces es cometer errores en la creación de los ficheros de entrada de datos. Afortunadamente GAMS, nos proporciona información de donde se encuentran los errores, y que ha provocado el error.

A continuación vamos a ver como detectar y corregir algunos errores en los ficheros. Para ello vamos a tomar el siguiente fichero GMS:

```
$ONTEXT

Se trata de resolver un programa lineal usando todas las variables.
El problema es:
        Min F(x) = 3*x1 + 2*x2 +x3 + 2*x4 +3*x5
        s.a:
               2*x1 + 5*x2 + x4 + x5 >= 6
               4*x2 -2*x3 +2*x4 + 3*x5 >= 5
               x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 >= 7
              x1, x2, x3, x4, x5 no negativas
$OFFTEXT
VARIABLES X1, X2, X-3, X4, X5, F;
POSITIVE VARIABLES X1, X2, X3, X, X5;

EQUATIONS OBJ, R1 R2, R3;

OBJ..   F =E= 3*x1 + 2*x2 +x3 + 2*x4 +3*x5 ;
R1..    2*x1 + 5x2 + x4 + x5 =G= 6;
R2..    4*x2 -2*x3 +2*x4 + 3*x5 = 5;
R3..    x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 =G= 7

MODEL LINEAL /OBJ, R1, R2, /;
SOLVE LINEAL USING NLP MINIMIZING F;
```

Si ejecutamos el fichero, tenemos que en la ventana de ejecución aparecen lineas en rojo, indicativas de los errores cometidos. No obstante, el fichero de salida LST tiene indicados cuales son esos errores:



El fichero LST, es:

```
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
C o m p i l a t i o n


         Se trata de resolver un programa lineal usando todas las
variables.
      El problema es:
              Min F(x) = 3*x1 + 2*x2 +x3 + 2*x4 +3*x5
              s.a:
                     2*x1 + 5*x2 + x4 + x5 >= 6
                     4*x2 -2*x3 +2*x4 + 3*x5 >= 5
                     x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 >= 7
                    x1, x2, x3, x4, x5 no negativas
  12  VARIABLES X1, X2, X-3, X4, X5, F;
****                     $96
  13  POSITIVE VARIABLES X1, X2, X3, X, X5;
  14
  15  EQUATIONS OBJ, R1 R2, R3;
  16
  17  OBJ..   F =E= 3*x1 + 2*x2 +x3 + 2*x4 +3*x5 ;
  18  R1..    2*x1 + 5x2 + x4 + x5 =G= 6;
****                   $37,409
  19  R2..    4*x2 -2*x3 +2*x4 + 3*x5 = 5;
****   $140                              $37
  20  R3..    x1 - 6*x2 + 3*x3 + 7*x4 +5*x5 =G= 7
```

```
  21
  22  MODEL LINEAL /OBJ, R1, R2, /;
****                            $2
  23  SOLVE LINEAL USING NLP MINIMIZING F;
****                                   $257
GAMS Rev 117  Windows NT/95/98                12/14/00 10:50:54  PAGE      2
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
Error Messages

  2  Identifier expected
 37  '=l=' or '=e=' or '=g=' operator expected
 96  Blank needed between identifier and text
       (-or- illegal character in identifier)
       (-or- check for missing ';' on previous line)
140  Unknown symbol
257  Solve statement not checked because of previous errors
409  Unrecognizable item - skip to find a new statement
       looking for a ';' or a key word to get started again

**** 7 ERROR(S)   0 WARNING(S)
```

En el fichero anterior se han resaltado en rojo, tanto los errores como los mensajes que ello origina. Sin embargo vamos a analizar uno por uno todos ellos indicando cual podría ser la corrección a aplicar. Afortunadamente GAMS numera las líneas, para facilitar esta labor, así tenemos:

```
12  VARIABLES X1, X2, X-3, X4, X5, F;
****                         $96
Con el mensaje:
96  Blank needed between identifier and text
       (-or- illegal character in identifier)
       (-or- check for missing ';' on previous line)
```

Se trata de que hemos usado un carácter ilegal para identificar una variable `X-3`, la corrección seria usar el nombre de `X3`.

```
18  R1..    2*x1 + 5x2 + x4 + x5 =G= 6;
****                  $37,409
Con el mensaje:
37  '=l=' or '=e=' or '=g=' operator expected
409  Unrecognizable item - skip to find a new statement
       looking for a ';' or a key word to get started again
```

Se trata de que hemos usado un carácter no reconocido: `5x2`, en lugar de: `5*x2`, para indicar el producto del coeficiente por la variable.

```
19  R2..    4*x2 -2*x3 +2*x4 + 3*x5 = 5;
****    $140                             $37

37  '=l=' or '=e=' or '=g=' operator expected
140  Unknown symbol
```

En esta línea hay dos errores, el primero (140) no marca que el símbolo R2, es desconocido. Si nos fijamos donde hemos definido el símbolo, en EQUATIONS, podemos observar que hemos omitido la coma entre R1 y R2, con lo cual GAMS solo considera valido R1 y como comentario R2, y por ello no es un símbolo adecuado.
El segundo error (37) es que hemos usado un símbolo de igual (=) en lugar de los símbolos necesarios para indicar el comportamiento de las restricciones, en este caso mayor-igual (=G=).

```
22  MODEL LINEAL /OBJ, R1, R2, /;
****                            $2
2  Identifier expected
```

En este caso se trata de un error un poco más complejo, ya que aunque indica que falta algún símbolo, en particular, después de R2 hay una coma y el final del bloque de model: (`/)` se trata de que falta un punto y coma (;) en el final de la línea anterior.

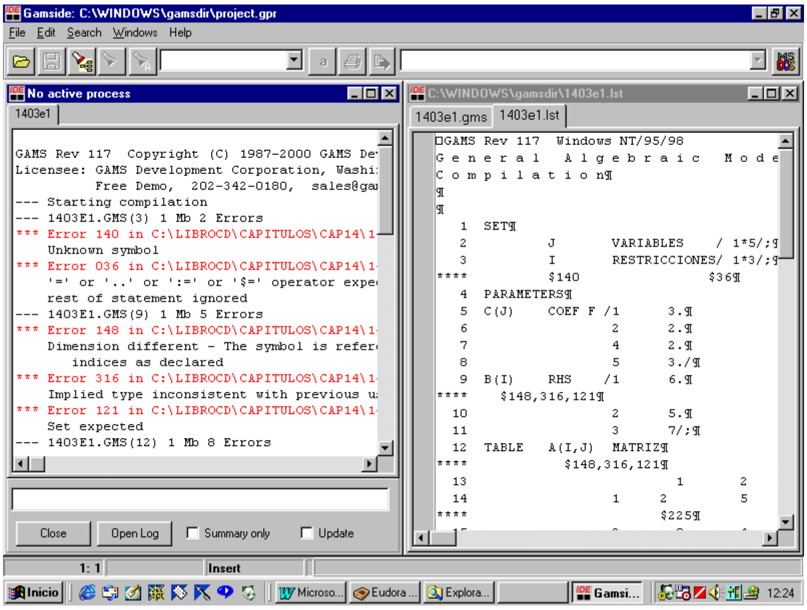
```
23  SOLVE LINEAL USING NLP MINIMIZING F;
****                                    $257
257  Solve statement not checked because of previous errors
```

Se trata de indicar que no sea podido ejecutar el fichero debido a que se han detectado errores anteriores.

Vamos a ver otros tipos de errores que se producen en los ficheros con los bloques opcionales. Para ello consideremos el siguiente fichero GMS:

```
SET
        J       VARIABLES    / 1*5/;
        I       RESTRICCIONES/ 1*3/;
PARAMETERS
C(J)    COEF F /1      3.
                2      2.
                4      2.
                5      3./
B(I)    RHS    /1      6.
                2      5.
                3      7/;
TABLE   A(I,J)  MATRIZ
                 1       2       3       4       5
                1     2         5       0       1       1
                2       0       4       -2      2       3
                3       1       -6      3       7       5;
VARIABLES
X(J)    VARIABLES PRINCIPALES
F;
POSITIVE VARIABLES ;

EQUATIONS
OBJ     FUNCION OBJETIVO
R(I)    RESTRICCIONES;

OBJ.. F =E= SUM(J,C(J)*X(J));
R(J).. SUM(J,A(I,J*X(J))  =G= B(I);
MODEL LINMATR2/ALL/;
SOLVE LINMATR2 USING LP MINIMIZING F;
```

La ejecución del fichero da origen a una pantalla como la siguiente:



Como podemos observar se indican los errores con las lineas rojas correspondientes. Así, el fichero LST es de la forma:

```
GAMS Rev 117  Windows NT/95/98                12/14/00 12:24:16  PAGE      1
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
C o m p i l a t i o n


   1  SET
   2          J       VARIABLES    / 1*5/;
   3          I       RESTRICCIONES/ 1*3/;
****          $140                   $36
   4  PARAMETERS
   5  C(J)    COEF F /1       3.
   6                  2       2.
   7                  4       2.
   8                  5       3./
   9  B(I)    RHS    /1       6.
****    $148,316,121
  10                  2       5.
  11                  3       7/;
  12  TABLE   A(I,J)  MATRIZ
****            $148,316,121
```

```
                        1         2         3         4         5
               1      2           5         0         1         1
****                  $225
               2         0        4        -2         2         3
               3         1       -6         3         7         5;
     VARIABLES
     X(J)     VARIABLES PRINCIPALES
     F;
     POSITIVE VARIABLES ;
****                $2

     EQUATIONS
     OBJ      FUNCION OBJETIVO
     R(I)     RESTRICCIONES;
****    $148,316,121

     OBJ.. F =E= SUM(J,C(J)*X(J));
     R(J).. SUM(J,A(I,J*X(J))  =G= B(I);
****              $125$316,121$148,8  $316,121$148
     MODEL LINMATR2/ALL/;
     SOLVE LINMATR2 USING LP MINIMIZING F;
****                                $257
GAMS Rev 117  Windows NT/95/98                12/14/00 12:24:16  PAGE     2
G e n e r a l   A l g e b r a i c   M o d e l i n g   S y s t e m
Error Messages

  2  Identifier expected
  8  ')' expected
 36  '=' or '..' or ':=' or '$=' operator expected
     rest of statement ignored
121  Set expected
125  Set is under control already
140  Unknown symbol
148  Dimension different - The symbol is referenced with more/less
        indices as declared
225  Floating entry ignored
257  Solve statement not checked because of previous errors
316  Implied type inconsistent with previous use of symbol

**** 22 ERROR(S)    0 WARNING(S)
```

```
3           I        RESTRICCIONES/ 1*3/;
****          $140                    $36

36  '=' or '..' or ':=' or '$=' operator expected
     rest of statement ignored
140  Unknown symbol
```

El error cometido es que GAMS no identifica que significa I, no tiene ningun indicador de referencias de conjunto, variable, etc. Ello se debe a que en la linea anterior se ha "cerrado" el bloque de SET con un punto y como (;). La solución consiste en eliminar el punto y coma correspondiente. Además, esto va a provocar que muchos de los errores sucesivos se deban a este error inicial, y por ello, más adelante haremos referencia es ete tipo de error.

```
9  B(I)     RHS     /1       6.
****    $148,316,121
148  Dimension different - The symbol is referenced with more/less
        indices as declared
316  Implied type inconsistent with previous use of symbol
121  Set expected
```

Estos tipos de error están motivado por el desconocimiento de I como conjuneto (error anterior).

```
12  TABLE   A(I,J)  MATRIZ
****              $148,316,121
148  Dimension different - The symbol is referenced with more/less
        indices as declared
316  Implied type inconsistent with previous use of symbol
121  Set expected
```

La explicación, y los tipos de errores son los mismo que el anterior.

```
13                          1       2       3       4       5
  14                   1    2         5       0       1       1
****                        $225
225  Floating entry ignored
```

Este error se debe al desajuste existente entre las cabeceras identificativas y los valores correspondientes, por eso en este caso hemos incluido las lineas 13 y 14.

```
20  POSITIVE VARIABLES ;
****                     $2
2  Identifier expected
```

El error aquí cometido es obviar que tipo de variables han de ser positivas.

```
24  R(I)     RESTRICCIONES;
****    $148,316,121
148  Dimension different - The symbol is referenced with more/less
        indices as declared
316  Implied type inconsistent with previous use of symbol
121  Set expected
```

Se trata de los mismos errores que se han cometido anteriormente, la no definición del conjunto I.

```
27  R(J).. SUM(J,A(I,J*X(J))  =G= B(I);
****             $125$316,121$148,8  $316,121$148
8  ')' expected
121  Set expected
125  Set is under control already
148  Dimension different - The symbol is referenced with more/less
        indices as declared
316  Implied type inconsistent with previous use of symbol
```

En esta línea, y aunque parezca que se han cometido múltiples errores, básicamente solamente son tres: a) el original de la no definición de I como conjunto, b) al omisión de un paréntesis para cerrar la expresión (`A(I,J)`), y c) la diferente dimensión entre el contador de la ecuación: `R(J)` con el termino independiente `B(I)`. Estos dos índices, siempre deben ser iguales.

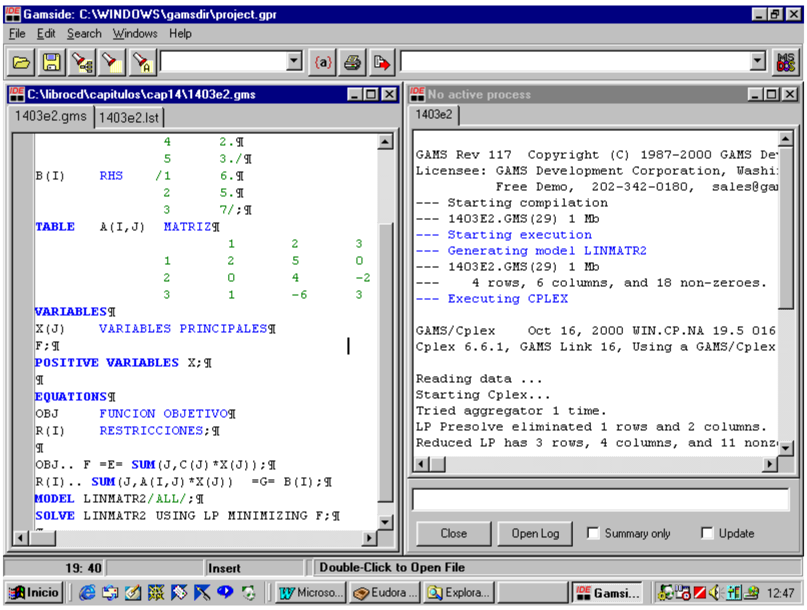
```
29  SOLVE LINMATR2 USING LP MINIMIZING F;
****                                     $257
257  Solve statement not checked because of previous errors
```

Como se ha explicado anteriormente, se trata de indicar que no sea podido ejecutar el fichero debido a que se han detectado errores anteriores.

Sin embargo, hay que resaltar que aunque GAMS puede detectar errores, hay otros "errores que GAMS no puede detectar, y son errores en los datos. Así, si corregimos los errores indicado por GAMS, tenemos el siguiente fichero de datos:

```
SET
        J       VARIABLES    / 1*5/
        I       RESTRICCIONES/ 1*3/;
PARAMETERS
C(J)    COEF F /1      3.
                2      2.
                4      2.
                5      3./
B(I)    RHS    /1      6.
                2      5.
                3      7/;
TABLE   A(I,J)  MATRIZ
                1       2       3       4       5
        1       2       5       0       1       1
        2       0       4       -2      2       3
        3       1       -6      3       7       5;
VARIABLES
X(J)    VARIABLES PRINCIPALES
F;
POSITIVE VARIABLES X;
EQUATIONS
OBJ     FUNCION OBJETIVO
R(I)    RESTRICCIONES;
OBJ.. F =E= SUM(J,C(J)*X(J));
R(I).. SUM(J,A(I,J)*X(J))   =G= B(I);
MODEL LINMATR2/ALL/;
SOLVE LINMATR2 USING LP MINIMIZING F;
```

Que una vez ejecutado, no nos indica ningún error, al mostrarnos una pantalla como la siguiente:



No obstante si comparamos el listado de ecuaciones de los dos problemas tenemos:

```
---- OBJ  =E=  FUNCION OBJETIVO

OBJ..  - 3*X(1) - 2*X(2) - 2*X(4) - 3*X(5) + F =E= 0 ; (LHS = 0)

---- R  =G=  RESTRICCIONES

R(1)..  2*X(1) + 5*X(2) + X(4) + X(5) =G= 6 ; (LHS = 0, INFES = 6 ***)

R(2)..  4*X(2) - 2*X(3) + 2*X(4) + 3*X(5) =G= 5 ; (LHS = 0, INFES = 5
***)

R(3)..  X(1) - 6*X(2) + 3*X(3) + 7*X(4) + 5*X(5) =G= 7 ; (LHS = 0,
INFES = 7 ***)
```

Con las ecuaciones anteriores:

```
Equation Listing    SOLVE LINMATR2 USING LP FROM LINE 44

---- OBJ  =E=  FUNCION OBJETIVO

OBJ..  - 3*X(1) - 2*X(2) - X(3) - 2*X(4) - 3*X(5) + F =E= 0 ;(LHS = 0)

---- R  =G=  RESTRICCIONES

R(1)..  2*X(1) + 5*X(2) + X(4) + X(5) =G= 6 ; (LHS = 0, INFES = 6 ***)

R(2)..  4*X(2) - 2*X(3) + 2*X(4) + 3*X(5) =G= 5 ; (LHS = 0, INFES = 5
***)

R(3)..  X(1) - 6*X(2) + 3*X(3) + 7*X(4) + 5*X(5) =G= 7 ; (LHS = 0,
INFES = 7 ***)
```

Podemos observar que se ha omitido el coeficiente en la función objetivo de la variable 3, al definir el parámetro correspondiente:

```
PARAMETERS
C(J)    COEF F /1      3.
                2      2.
                4      2.
                5      3./
```

Con ello GAMS interpreta que el citado coeficiente es cero. Aunque aparentemente se trataría de un error sin importancia, podemos observar que la diferencia que ello produce en la solución optima es notable. Así la solución de este problema es:

```
               S O L V E      S U M M A R Y

     MODEL   LINMATR2            OBJECTIVE  F
     TYPE    LP                  DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  29

**** SOLVER STATUS     1 NORMAL COMPLETION
**** MODEL STATUS      1 OPTIMAL
**** OBJECTIVE VALUE               4.7200

 RESOURCE USAGE, LIMIT          0.060     1000.000
 ITERATION COUNT, LIMIT         3        10000



Optimal solution found.

Objective :           4.720000

                       LOWER     LEVEL     UPPER    MARGINAL
---- EQU OBJ             .         .         .        1.000

  OBJ  FUNCION OBJETIVO

---- EQU R  RESTRICCIONES

     LOWER     LEVEL     UPPER    MARGINAL
1     6.000     6.000     +INF      0.400
2     5.000     5.000     +INF      0.240
3     7.000     7.000     +INF      0.160

---- VAR X  VARIABLES PRINCIPALES

     LOWER     LEVEL     UPPER    MARGINAL
1      .         .        +INF      2.040
2      .       0.910      +INF        .
3      .       0.770      +INF        .
4      .       1.450      +INF        .
5      .         .        +INF      1.080

                       LOWER     LEVEL     UPPER    MARGINAL
---- VAR F              -INF      4.720     +INF        .
```

Mientras que la solución optima "real" del problema es:

```
               S O L V E      S U M M A R Y

     MODEL   LINMATR2            OBJECTIVE  F
     TYPE    LP                  DIRECTION  MINIMIZE
     SOLVER  CPLEX               FROM LINE  44

**** SOLVER STATUS     1 NORMAL COMPLETION
**** MODEL STATUS      1 OPTIMAL
**** OBJECTIVE VALUE                5.1707

 RESOURCE USAGE, LIMIT          1.920     1000.000
 ITERATION COUNT, LIMIT         2        10000

GAMS/Cplex    Aug  7, 2000 WIN.CP.NA 19.4 016.015.038.WAT For Cplex 6.6
Cplex 6.6.1, GAMS Link 16, Using a GAMS/Cplex demo license installed at
runtime.

Optimal solution found.

Objective :           5.170732

                       LOWER     LEVEL     UPPER    MARGINAL

---- EQU OBJ             .         .         .        1.000

  OBJ  FUNCION OBJETIVO

---- EQU R  RESTRICCIONES

     LOWER     LEVEL     UPPER    MARGINAL

1     6.000     6.000     +INF      0.634
2     5.000     6.878     +INF       .
3     7.000     7.000     +INF      0.195

---- VAR X  VARIABLES PRINCIPALES

     LOWER     LEVEL     UPPER    MARGINAL

1      .         .        +INF      1.537
2      .       0.854      +INF       .
3      .         .        +INF      0.415
4      .       1.732      +INF       .
5      .         .        +INF      1.390

                       LOWER     LEVEL     UPPER    MARGINAL

---- VAR F              -INF      5.171     +INF       .
```

Por ello debemos insistir en la necesidad de revisar todo los datos del problema, ya que ahí es donde de genera el mayor numero de errores que GAMS no es capaz de detectar y que pueden producir unas diferencias notables en los resultados.