OPTIMIZACIÓN ECONÓMICA CON **GAMS**

Casasus, T.; Mocholi, M.; Sanchis, V. y Sala, R.[1]

1.- La docencia en los módulos de optimización económica.

La teoría de la optimización es una de las partes más importantes de la docencia de la matemática en los estudios de economía y empresa, y en particular, dentro de las directrices generales propias del titulo de Economía aparece en el descriptor de la materia de matemáticas como materia troncal. No obstante, en las restantes titulaciones (Administración y Dirección de Empresas y en la diplomatura de Empresariales) aunque no aparece dentro del descriptor como una parte especifica, su importancia para el estudio de otras materias han aconsejado su introducción en el plan de estudios, ya sea como parte de un módulo troncal o como un módulo optativo.

En la Universidad de Valencia, en particular, la teoría de la optimización aparece dentro del módulo troncal de Matemáticas para Economistas II, en la licenciatura de Economía, en los módulos optativos -de primer y segundo ciclo- de Programación Matemática y Modelización y Optimización, en la licenciatura de Administración y Dirección de Empresas, y en el módulo optativo de Investigación Operativa Básica en la diplomatura de Ciencias Empresariales. En los diferentes módulos se combinan, necesariamente, las cuestiones teóricas con lo ejercicios prácticos.

Para la realización de los ejercicios prácticos se hace recomendable el uso de los ordenadores para que los estudiantes puedan analizar los resultados de una forma rápida, ya que de otra forma les resulta tedioso el repetir mecánicamente los pasos para la obtención, por ejemplo en un problema de programación clásica sin restricciones, de los óptimos locales.

En este trabajo vamos a exponer la metodología seguida en nuestra Universidad para impartir optimización a los alumnos de economía y empresa, docencia que implica el uso de soporte informático para las clases prácticas.

Tomaremos como referencia los dos módulos de la licenciatura de ADE, ya que ambos módulos están muy interrelacionados, y suponen la continuación natural uno del otro.

---

[1] Departamento de Economía Financiera y Matemática. Universidad de Valencia.

Una versión más amplia de este trabajo se puede encontrar en la siguiente dirección electrónica:

http://www.sosig.ac.uk/cticce/cheer/ch102/ch102p02.htm

El módulo de Programación Matemática, es un módulo de 6 créditos (3 teóricos y 3 prácticos). En las clases de teoría se analizan los fundamentos matemáticos de la teoría de la optimización, es decir, comenzando por la convexidad (en conjuntos y funciones) y pasando por la programación no lineal, lineal y lineal entera. Junto a las clases teóricas se han de desarrollar las clases prácticas, que permiten complementar y consolidar los conocimientos adquiridos en las clases teóricas, tanto en base a ejercicios escritos como a ejercicios resueltos mediante ordenador. Para esta segunda parte del módulo practico es donde se le explica al alumno, aunque de forma muy básica el funcionamiento del software denominado GAMS, lo cual le permitirá resolver problemas de forma rápida y eficiente y centrar su atención en la interpretación económica de los resultados obtenidos.

En la segunda parte de este trabajo presentaremos algunas ventajas de GAMS para modelizar los problemas de mayores dimensiones, módulo de Modelización y Optimización, y finalizaremos esta exposición con algunas opciones más avanzadas que ya están más cerca de los trabajos de doctorado o de investigación que de la docencia del primer y segundo ciclo, aunque algunos de estos aspectos también pueden ser considerados dentro de este tipo de estudios.

2.- Introducción a GAMS

El nombre GAMS deriva de las iniciales de General Algebraic Modelling System, que como su propio nombre indica es un lenguaje de modelización, más que un programa para resolver problemas de optimización. La ventaja que presenta este programa GAMS, es que junto al módulo de modelización (base) incorpora diferentes solver (algoritmos de resolución de problemas) tanto de programación no lineal, como lineal y entera.

El lenguaje GAMS posee diferentes versiones (estudiante, profesional, workstation, mainframe, etc.) que se diferencian básicamente en las posibilidades de resolución de problemas de diferente tamaño, así por ejemplo, en la versión básica de estudiante existen una serie de limitaciones en cuanto al tamaño del problema que admite un máximo de 1000 elementos distintos de cero en los problemas lineales y no lineales, y de 20 variables enteras. Estas limitaciones no son importantes para los ejercicios prácticos que se desarrollan en las clases.

Una de las grandes ventajas de GAMS es la facilidad de migrar a versiones superiores ya que el formato general es idéntico tanto en un PC como en un supercomputador en paralelo, otra de las ventajas es también la capacidad de resolver diferentes versiones de un mismo modelo, tanto como problema no lineal, lineal y entero, y el poder usar diferentes solver. De entre ellos podemos citar los siguientes:

Problemas	Solver
NLP	CONOPT, MINOS, etc.
LP	OSL, CPLEX, MINOS, BDMLP, XA, etc.
MIP	OSL, ZOOM, CPLEX, XA; etc.
MINLP	DICOPT

Además, dentro de cada uno de estos solver, se pueden elegir diferentes opciones. Por ejemplo, para resolver los problemas lineales con la librería OSL, podemos elegir los siguientes algoritmos:

Primal Simplex
Dual Simplex
Network
Interior Point

Todo ello solamente con definir una línea dentro del fichero de datos, con lo cual no será necesario tener un conocimiento exhaustivo de cada uno de los diferentes solvers.

Quizá uno de los inconvenientes de GAMS de cara a su aplicación práctica con los estudiantes es lo poco "friendly" que resulta su uso las primeras veces, pero creemos que este inconveniente se ve superado, con creces, por la potencia y flexibilidad de este programa. Para poder resolver un problema con GAMS, es necesario generar un fichero de datos (en adelante lo denominaremos fichero GMS) que debe contener todas la instrucciones básicas y planteamiento del modelo que se desea resolver. Una vez generado el fichero, se ejecuta GAMS y el resultado se genera de forma automatica en un fichero distinto (fichero LST).

Si comparamos este procedimiento con la resolución de problemas de optimización con programas, como por ejemplo, EXCEL[2], LINDO, etc. podemos observar que resulta un poco más complicado, pero a medida que vayamos exponiendo los procedimientos y posibilidades de GAMS, nos daremos cuenta que no se trata de un grave inconveniente y que por el contrario, presenta una serie de ventajas como son el hecho de que con GAMS podemos usar un fichero de

[2] Es interesante el articulo de Conway y Ragsdale sobre la utilización de las hojas de calcúlo en los problemas de optimización.

datos que contenga todas las instrucciones y datos necesario del problema, o bien podemos recurrir a leer ficheros de datos externos, tanto ficheros ASCII como ficheros de hojas de calculo (Lotus, Excel, etc.). El fichero de resultados, también puede ser redireccionado (todo o parte) a otros ficheros con diferentes formatos. También, podemos introducir condicionales IF-ELSE, realizar bucles para resolver diferentes problemas de forma simultánea, etc. Aparentemente, puede resultar poco útil para los estudiantes, pero posteriormente veremos que se trata de procedimientos comunes en diferentes situaciones.

Todas estas ventajas, fueron las que nos decidieron a utilizar GAMS en nuestras clases prácticas. En los dos apartados siguientes expondremos la forma de funcionar con este tipo de programas, tanto en el módulo de Programación Matematica como en el módulo, más avanzado, de Modelización. Los dos casos, los hemos denominado: Lenguaje de Optimización y Lenguaje de Modelización.

3.- Primer nivel. Lenguaje de optimización.

En este apartado, comentaremos los usos básicos de GAMS para la resolución de los problemas no lineales y lineales de pequeña dimensión, dado que en ellos resulta mucho más sencillo el poder usar y definir todas las variables y ecuaciones. Posteriormente, al usar GAMS como lenguaje de modelización analizaremos que para problemas de mayores dimensiones resulta mucho más cómodo usar la notación "compacta".

La exposición del funcionamiento de GAMS la realizaremos sobre un ejemplo concreto y particular. Consideremos en primer lugar un problema clásico de optimización. Sea el problema de maximizar una función de utilidad sujeta a una restricción presupuestaria:

Max U(x,y) = (x+2) * (y+1)

s.a: 4*x + 6*y = 130;

Para resolver este problema es necesario preparar un "input file" o un fichero de datos[3] (en adelante lo denominaremos fichero GMS) como se muestra en la tabla 1, en donde se han definido todos los elementos básicos: VARIABLES; EQUATIONS; MODEL y SOLVE:

[3] Para generar un fichero de datos podemos utilizar cualquier editor de líneas (Edit, Kedit,etc.) e incluso procesadores de texto (Word, Wordperfect, etc) pero en este ultimo caso hay que guardar el fichero en formato de solo texto, para evitar los caracteres internos de cada programa.

```
VARIABLES
X
Y
U;
X.L=1;
Y.L=1;

EQUATIONS

OBJ     FUNCION DE UTILIDAD
RP      RESTRICCION PRESUPUESTARIA;

OBJ.. U =E= (X+2)*(Y+1);
RP.. 4*X + 6*Y =E= 130;

MODEL MAXUTIL/OBJ,RP/;

SOLVE MAXUTIL USING NLP MAXIMIZING U;
```

Tabla 1

Para construir el fichero GMS es necesario declarar :

1. Las variables que aparecen en el problema (X,Y,U) y si tienen alguna consideración de clase (no negativas, enteras, etc.). No es necesario fijar un punto de partida, pero si este punto no se declara GAMS fija como punto inicial el (0,0), por lo cual, con el fin de facilitar la obtención de una solución, en muchos casos es aconsejable dar un punto de partida distinto del (0,0). En el ejemplo anterior, el punto de partida elegido ha sido el punto (1,1).
2. Las ecuaciones (nombre y expresión), en nuestro ejemplo serán OBJ y RP y la expresión matemática de estas ecuaciones.
3. El modelo (MAXUITL) y las ecuaciones (OBJ y RP) que forman el modelo.
4. La declaración de resolución, el solver (NLP, LP, MIP, ...) y la dirección de optimización (Maximización o minimización).

Para ejecutar el programa, una vez el programa ha sido instalado en un PC, workstation o mainframe, se procesará el programa siguiendo las instrucciones adecuadas para cada tipo de sistema, por ejemplo, en un PC bajo DOS[4], seria:

C:\GAMS>GAMS INPUT.GMS OUTPUT OUTFILE.LST

El output file es un fichero extenso, pero la parte más importante es el resumen de la solución buscada aparece en la Tabla 2 :

[4] En Windows, basta con asociar las extensiones GMS al fichero gamsw.bat.

```
EXIT -- OPTIMAL SOLUTION FOUND

                       LOWER     LEVEL     UPPER    MARGINAL
---- EQU OBJ             .         .         .       -1.000
---- EQU RP           130.000   130.000   130.000     3.000

  OBJ         FUNCION DE UTILIAD
  RP          RESTRICCION PRESUPUESTARIA

                       LOWER     LEVEL     UPPER    MARGINAL
---- VAR X              -INF     16.000     +INF       .
---- VAR Y              -INF     11.000     +INF      EPS
---- VAR U              -INF    216.000     +INF       .
```

Tabla 2

La solución óptima buscada es X=16 (level), Y=11 y U=216, además otra información relevante de este tipo de problemas es el valor del multiplicador de Lagrange asociado con la ecuación RP, con valor marginal 3, que aparece en la segunda linea correspondiente a la ecuación RP.

Un aspecto muy importante es el análisis del significado de esta *utilidad marginal*, y por esta razón es muy útil construir un nuevo archivo con las ecuaciones originales y una nueva ecuación con 131 como valor para el termino independiente de la restricción. Un ejemplo de este archivo viene dado en la tabla 3, y la parte relevante del fichero LST aparece en la tabla 4:

```
VARIABLES
X
Y
U;
X.L=1;
Y.L=1;

EQUATIONS

OBJ     FUNCION DE UTILIDAD
RP      RESTRICCION PRESUPUESTARIA
RP1     NUEVO TERMINO INDEPENDIENTE;

OBJ.. U =E= (X+2)*(Y+1);
RP.. 4*X + 6*Y =E= 130;
RP1.. 4*X + 6*Y =E= 131;

MODEL MAXUTIL/OBJ,RP/;
MODEL MAXUTIL1/OBJ,RP1/;

SOLVE MAXUTIL USING NLP MAXIMIZING U;
SOLVE MAXUTIL1 USING NLP MAXIMIZING U;
```

Tabla 3

```
EXIT -- OPTIMAL SOLUTION FOUND
                       LOWER     LEVEL     UPPER    MARGINAL
---- EQU OBJ             .         .         .       -1.000
---- EQU RP           130.000   130.000   130.000     3.000

  OBJ         FUNCION DE UTILIDAD
  RP          RESTRICCION PRESUPUESTARIA

                       LOWER     LEVEL     UPPER    MARGINAL
---- VAR X              -INF     16.000     +INF        .
---- VAR Y              -INF     11.000     +INF       EPS
---- VAR U              -INF    216.000     +INF        .

 EXIT -- OPTIMAL SOLUTION FOUND
                       LOWER     LEVEL     UPPER    MARGINAL
---- EQU OBJ             .         .         .       -1.000
---- EQU RP1          131.000   131.000   131.000     3.021

  OBJ         FUNCION DE UTILIDAD
  RP1         NUEVO TERMINO INDEPENDIENTE

                       LOWER     LEVEL     UPPER    MARGINAL
---- VAR X              -INF     16.125     +INF       EPS
---- VAR Y              -INF     11.083     +INF        .
---- VAR U              -INF    219.010     +INF        .
```

Tabla 4

Como puede observarse, el nuevo valor de la función de utilidad es U=219.01 y este valor es, aproximadamente, el valor anterior (216) más la utilidad marginal (3). La pequeña diferencia existente (0.01) aparece debido a que el incremento de la nueva RHS no es , exactamente infinitesimal.

Cuando añadimos condiciones de no negatividad en las variables y las restricciones de desigualdad, estamos ante un problema de programación no lineal. En este caso no es posible resolver el problema vía optimización clásica y debemos hacer uso de las condiciones de Kuhn-Tucker. Con GAMS su planteamineto y resolución es igual que el problema anterior de programación clásica y por tanto muy facil para el alumno cuando se ha resuelto el anterior.Veamos un ejemplo:

Consideremos el problema de minimización de costes

Min C=4*K +5*L

s.a: $10 * K^{0.5} * L^{0.5} \geq 1000$

$K \geq 0, L \geq 0$

El archivo de entrada se construye de manera similar al problema de Maximización de la Utilidad, cambiando el nombre y la clase de las variables y el sentido de las desigualdades en las restricciones.

En el archivo de entrada visto anteriormente añadimos las condiciones de no negatividad para las variables K y L (positive variables). Para obtener la *tasa marginal de sustitución técnica* añadimos una nuevo valor de Q_O, (1001) que sustituye el valor original (1000). El archivo se presenta en la Tabla 5:

```
VARIABLES
K
L
C;
POSITIVE VARIABLES K,L;
K.L=10;
L.L=10;

EQUATIONS
OBJ     FUNCION DE COSTES
P       CANTIDAD A PRODUCIR
P1      NUEVA CANTIDAD;

OBJ.. C =E= 4*K + 5*L;
P.. 10*(K**(0.5) * L**(0.5)) =G= 1000;
P1.. 10*(K**(0.5) * L**(0.5)) =G= 1001;

MODEL MINCOS/OBJ,P/;
MODEL MINCOS1/OBJ,P1/;

SOLVE MINCOS USING NLP MINIMIZING C;
SOLVE MINCOS1 USING NLP MINIMIZING C;
```

Tabla 5

Parte de la solución del archivo de salida aparece en la Tabla 6:

```
 EXIT -- OPTIMAL SOLUTION FOUND
                      LOWER     LEVEL     UPPER    MARGINAL
---- EQU OBJ            .         .         .        1.000
---- EQU P          1000.000  1000.000     +INF      0.894

  OBJ         FUNCION DE COSTES
  P           CANTIDAD A PRODUCIR

                      LOWER     LEVEL     UPPER    MARGINAL
---- VAR K              .       111.803     +INF       .
---- VAR L              .        89.443     +INF      EPS
---- VAR C             -INF    894.427     +INF       .

 EXIT -- OPTIMAL SOLUTION FOUND
                      LOWER     LEVEL     UPPER    MARGINAL
---- EQU OBJ            .         .         .        1.000
---- EQU P1         1001.000  1001.000     +INF      0.894

  OBJ         FUNCION DE COSTES
  P1          NUEVA CANTIDAD

                      LOWER     LEVEL     UPPER    MARGINAL
---- VAR K              .       111.915     +INF      EPS
---- VAR L              .        89.532     +INF       .
---- VAR C             -INF    895.322     +INF       .
```

Tabla 6

Vemos, al igual que en el caso anterior, que el nuevo valor de la función de coste (895.322) es aproximadamente igual al antiguo (894.427) más el valor del multiplicador de Lagrange de la ecuación P (0.894).

De igual forma que hemos resuelto dos problemas de programación clásica y no lineal, se puede usar GAMS para resolver problemas lineales. Quizá sea en este aspecto uno de los puntos débiles de GAMS para la docencia, ya que en la versión estudiantes de GAMS no aparece el análisis de sensibilidad de la solución. En las versiones profesionales con los solvers, por ejemplo, OSL, si que es posible realizar análisis de sensibilidad de la solución, tanto de los coeficientes de la función objetivo como de los términos independientes de la restricciones.

Una forma, rudimentaria de superar este inconveniente es recurrir al análisis de los valores de las variables duales, y conocido este valor (para las variables no básicas) determinar el campo de variación. Otra alternativa, seria el plantear un problema con diferentes términos independiente y resolverlos de forma simultánea.

Pero, evidentemente, estas alternativas no sustituyen al inconveniente de la falta de análisis de sensibilidad.[5]

4.- Segundo Nivel. Lenguaje de Modelización

Una vez conocidos los fundamentos básicos para la creación de ficheros para usar GAMS, así como la lectura e interpretación de los resultados de los diferentes problemas, se hace necesario ampliar los conocimientos sobre el manejo del programa y de otros conceptos. Por esta razón en el módulo de Modelización y Optimización, en el tercer curso de la licenciatura de ADE se introducen una serie de conceptos básicos sobre modelización, tanto de problemas lineales, enteros y no lineales, así como una introducción a la programación multiobjetivo y con incertidumbre.

Es en este módulo donde se avanza un poco más en el uso de GAMS para plantear y resolver complejos problemas económicos. Así, a titulo de ejemplo, vamos a modelizar un problema de transporte con el objetivo de ilustrar el funcionamiento de otras partes de GAMS, como SET, DISPLAY u OPTIONS.

[5] Posiblemente en la próxima versión 3 de GAMS, se incluya el análisis de sensibilidad en los problemas lineales.

Para el problema básico de transporte[6] vamos a considerar el siguiente ejemplo: Winston(1991) :

	DESTINO				
ORIGEN	Ciudad 1	Ciudad 2	Ciudad 3	Ciudad 4	OFERTA
Planta 1	8	6	10	9	35
Planta 2	9	12	13	7	50
Planta 3	14	9	16	5	40
DEMANDA	45	20	30	30	

Figura 1

Para este problema, en GAMS es posible definir un SET (índice) para plantas y ciudades, y a partir de estos índices declarar las variables y las ecuaciones en función de ellos con el fin de poder utilizar los símbolos de sumatorio para un índice. En este caso el fichero GMS tendría el siguiente formato:

[6] Como se sabe se trata de un programa entero que por sus características matemáticas puede resolverse como un problema de programación lineal.

```
$TITLE   PROBLEMA DE TRANSPORTE
OPTIONS LIMROW = 10;
  SETS
       I   plantas          / PLANTA1, PLANTA2, PLANTA3 /
       J   mercados         / CIUDAD1*CIUDAD4 / ;
  PARAMETERS
       A(I)  oferta
         /    PLANTA1      35
              PLANTA2      50
              PLANTA3      40  /
       B(J)  demanda
         /    CIUDAD1     45
              CIUDAD2     20
              CIUDAD3     30
              CIUDAD4     30  / ;
  TABLE C(I,J)  Costes de Transporte
                CIUDAD1 CIUDAD2 CIUDAD3 CIUDAD4
  PLANTA1       8       6       10      9
  PLANTA2       9       12      13      7
  PLANTA3       14      9       16      5 ;
 VARIABLES
       X(I,J)  cantidades transportadas desde i hasta j
       Z       coste total de transporte;
  POSITIVE VARIABLE X ;
  EQUATIONS
       COSTE       función objetivo
       OFERTA(I)   oferta de la planta i
       DEMANDA(J)  demanda del mercado j;

  COSTE ..       Z  =E=  SUM((I,J), C(I,J)*X(I,J)) ;
  OFERTA(I) ..   SUM(J, X(I,J))  =L=  A(I) ;
  DEMANDA(J) ..  SUM(I, X(I,J))  =G=  B(J) ;

  MODEL TRANSPORT /ALL/ ;
  SOLVE TRANSPORT USING LP MINIMIZING Z ;
  DISPLAY X.L, Z.L ;
```

Tabla 7

La formulación del problema generada directamente por GAMS, incluyendo todas las ecuaciones del mismo es :

```
---- COSTE       =E=  funcion objetivo

COSTE..  - 8*X(PLANTA1,CIUDAD1) - 6*X(PLANTA1,CIUDAD2) -
10*X(PLANTA1,CIUDAD3) - 9*X(PLANTA1,CIUDAD4) - 9*X(PLANTA2,CIUDAD1) -
12*X(PLANTA2,CIUDAD2) - 13*X(PLANTA2,CIUDAD3) - 7*X(PLANTA2,CIUDAD4) -
14*X(PLANTA3,CIUDAD1) - 9*X(PLANTA3,CIUDAD2) - 16*X(PLANTA3,CIUDAD3) -
5*X(PLANTA3,CIUDAD4) + Z =E= 0 ; (LHS = 0)

---- OFERTA      =L=  oferta de la planta i

OFERTA(PLANTA1)..  X(PLANTA1,CIUDAD1) + X(PLANTA1,CIUDAD2)
        + X(PLANTA1,CIUDAD3) + X(PLANTA1,CIUDAD4) =L= 35 ; (LHS = 0)

OFERTA(PLANTA2)..  X(PLANTA2,CIUDAD1) + X(PLANTA2,CIUDAD2)
      + X(PLANTA2,CIUDAD3) + X(PLANTA2,CIUDAD4) =L= 50 ; (LHS = 0)

OFERTA(PLANTA3)..  X(PLANTA3,CIUDAD1) + X(PLANTA3,CIUDAD2)
      + X(PLANTA3,CIUDAD3) + X(PLANTA3,CIUDAD4) =L= 40 ; (LHS = 0)

---- DEMANDA     =G=  demanda del mercado j

DEMANDA(CIUDAD1)..  X(PLANTA1,CIUDAD1) + X(PLANTA2,CIUDAD1)
      + X(PLANTA3,CIUDAD1) =G= 45 ; (LHS = 0 ***)

DEMANDA(CIUDAD2)..  X(PLANTA1,CIUDAD2) + X(PLANTA2,CIUDAD2)
      + X(PLANTA3,CIUDAD2) =G= 20 ; (LHS = 0 ***)

DEMANDA(CIUDAD3)..  X(PLANTA1,CIUDAD3) + X(PLANTA2,CIUDAD3)
      + X(PLANTA3,CIUDAD3) =G= 30 ; (LHS = 0 ***)

DEMANDA(CIUDAD4)..  X(PLANTA1,CIUDAD4) + X(PLANTA2,CIUDAD4)
      + X(PLANTA3,CIUDAD4) =G= 30 ; (LHS = 0 ***)
```

Tabla 8

La solución del problema, usando la opción DISPLAY, es :

```
----     36 VARIABLE  X.L           cantidades transportadas desde i
hasta j

            CIUDAD1     CIUDAD2     CIUDAD3     CIUDAD4
PLANTA1                  10.000      25.000
PLANTA2       45.000                  5.000
PLANTA3                  10.000                  30.000

----     36 VARIABLE  Z.L                  =     1020.000 coste total
de
                                                        transporte
```

Tabla 9

5.- Opciones avanzadas: Bucles.

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 ilustrar como podemos 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        BIEN X
Y        BIEN Y
U;X.L=1;Y.L=1;
EQUATIONS
OBJ      FUNCION DE UTILIDAD
RP       RESTRICCION PRESUPUESTARIA;
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;
```

Tabla 10

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

```
----     68 PARAMETER OUTPUT


                   1           2           3           4           5
M            130.000     131.000     132.000     133.000     134.000
FUNCION      216.000     219.010     222.042     225.094     228.167
LAGRANGE       3.000       3.021       3.042       3.062       3.083
X             16.000      16.125      16.250      16.375      16.500
Y             11.000      11.083      11.167      11.250      11.333

       +           6           7           8           9          10
M            135.000     136.000     137.000     138.000     139.000
FUNCION      231.260     234.375     237.510     240.667     243.844
LAGRANGE       3.104       3.125       3.146       3.167       3.187
X             16.625      16.750      16.875      17.000      17.125
Y             11.417      11.500      11.583      11.667      11.750
```

Tabla 11

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.

Otra consideración importante en GAMS es el poder utilizar ficheros de datos separados de los ficheros del modelo, de forma que una vez construido el modelo básico es posible probar el modelo con diferentes estructuras de datos. Esto hace referencia a la facilidad de lectura y escritura en ficheros diferentes del fichero GMS original. Vamos a ilustrar esta consideración retomando el problema del transporte que hemos explicado anteriormente, y para este problema utilizaremos un fichero con los datos del problema y otro fichero con la estructura del modelo.

Así vamos a considerar en la tabla 12, un fichero de datos que denominaremos “asp1.inc”, que no es ni mas ni menos que la primera parte del fichero anterior, en donde se han definido los conjuntos de índices, así como los parámetros y las tablas necesarias. El fichero será el siguiente:

```
OPTIONS LIMROW = 10;
  SETS
       I   plantas           / PLANTA1, PLANTA2, PLANTA3 /
       J   mercados          / CIUDAD1*CIUDAD4 / ;
  PARAMETERS
       A(I)  oferta
         /    PLANTA1      35
              PLANTA2      50
              PLANTA3      40  /
       B(J)  demanda
         /    CIUDAD1      45
              CIUDAD2      20
              CIUDAD3      30
              CIUDAD4      30  / ;
  TABLE C(I,J)  Costes de Transporte
                CIUDAD1 CIUDAD2 CIUDAD3 CIUDAD4
  PLANTA1       8       6       10      9
  PLANTA2       9       12      13      7
  PLANTA3       14      9       16      5 ;
```

Tabla 12

El fichero GMS contiene una llamada a este fichero de datos, pero además hemos incluido otra consideración con relación a la solución, y es que la solución la pueda guardar en un fichero diferente del fichero LST de salida. Con ello el fichero del modelo del transporte es el siguiente:

```
$INCLUDE "ASP1.INC"
 VARIABLES
       X(I,J)  cantidades transportadas desde i hasta j
       Z       coste total de transporte;
  POSITIVE VARIABLE X ;
  EQUATIONS
       COSTE       funcion objetivo
       OFERTA(I)   oferta de la planta i
       DEMANDA(J)  demanda del mercado j;
  COSTE ..       Z  =E=  SUM((I,J), C(I,J)*X(I,J)) ;
  OFERTA(I) ..   SUM(J, X(I,J))  =L=  A(I) ;
  DEMANDA(J) ..  SUM(I, X(I,J))  =G=  B(J) ;
  MODEL TRANSPORT /ALL/ ;
  SOLVE TRANSPORT USING LP MINIMIZING Z ;
  FILE RES /RESUL.DAT/;
  PUT RES;
  LOOP( (I,J),
             PUT I.TL:11, J.TL:11,  X.L(I,J):15:7 /;
       );
```

Tabla 13

En el fichero anterior se ha resaltado en negrilla las llamadas al fichero de datos y al fichero de solución. El fichero RESUL.DAT, tiene el siguiente formato de salida:

```
PLANTA1    CIUDAD1          0.0000000
PLANTA1    CIUDAD2         10.0000000
PLANTA1    CIUDAD3         25.0000000
PLANTA1    CIUDAD4          0.0000000
PLANTA2    CIUDAD1         45.0000000
PLANTA2    CIUDAD2          0.0000000
PLANTA2    CIUDAD3          5.0000000
PLANTA2    CIUDAD4          0.0000000
PLANTA3    CIUDAD1          0.0000000
PLANTA3    CIUDAD2         10.0000000
PLANTA3    CIUDAD3          0.0000000
PLANTA3    CIUDAD4         30.0000000
```

Tabla 14

En este fichero tenemos la solución de las cantidades transportadas desde cada origen a cada destino.

Como se podrá comprobar la única consideración importante es la definición de los formatos de salida de los datos, ya que no presenta mayor dificultad el leer y escribir en otros ficheros.

6. Resumen y consideraciones generales.

Con este trabajo intentamos introducir lo que nosotros pensamos es un programa muy útil para la resolución de problemas en Matemáticas Empresariales y Económicas, y más específicamente, en problemas de Programación en Economía.

Aunque el uso de GAMS no está extendido como una herramienta de uso común entre los profesores y estudiantes en nuestras Facultades, nuestra opinión es que, una vez los estudiantes conocen los instrumentos básicos de este programa, GAMS permite la resolución de muy variados tipos de problemas (problemas lineales, no lineales, enteros, etc.) y también el estudio de los diferentes aspectos del problema , con solo pequeños cambios en el archivo de entrada. Por todo esto, nuestra experiencia como profesores de distintos grupos en las licencituras de Economía y Administración y Dirección de Empresas nos confirma que el programa GAMS es de gran ayuda, tanto entre los estudiantes, como entre los profesores, para la correcta resolución e interpretación de los problemas de Programación en Economía.

El hecho de que los estudiantes deban construir su propio archivo de entrada no consideramos que sea un problema, sino un modo para ayudar al estudiante a reforzar su comprensión del problema que está tratando de resolver.

Referencias bibliográficas:

Chiang, A. C. (1984): “Fundamental Methods of Mathematical Economics”. McGraw-Hill Tokio, Japan.

Brooke, A and alt. (1.992):“GAMS: User’s guide”Release 2.25. The Scientific Press. USA.

Conway, D.G. & Ragsdale, C.T. (1.997): “ Modeling optimization problems in the unstructured world of spreadsheets”. *Omega* Vol 25, pp 313-322..

Mocholí, M. y Sala, R. (1.996): "Decisiones de optimización". Ed. Tirant Lo Blanc. Valencia.

Winston, W. L. (1.991): “Introduction to Mathematical Programming: Applications & Algorithms”. PWS-KENT, Boston.