

The logo graphic consists of a vertical black line intersecting a horizontal black line. To the left of the vertical line, there are three overlapping rectangular blocks: a blue one at the top, a red one in the middle, and a yellow one at the bottom. The text 'GAMS' is positioned to the right of the vertical line, centered vertically between the blue and yellow blocks.

GAMS



Contenido

- Estructura de un modelo en GAMS. Ejemplo a través del cual se ilustra el lenguaje
- Conjuntos
- Datos
- Variables
- Ecuaciones
- Sumatoria y Productoria
- Definición de ecuaciones
- Función Objetivo
- Sentencias "Model" y "Solve"
- Procedimientos disponibles
- Visualización de sentencias
- Bases de datos en GAMS
- Asignación de límites y valores iniciales



Estructura de un modelo GAMS (1/3)

Entradas

- **Sets**
Declaración.
Asignación de miembros
- **Data** (Parámetros, tablas, escalar)
Declaración.
Asignación de valores
- **Variables**
Declaración.
Asignación de tipo
- **Asignación de límites y/o valores iniciales** (opcional)
- **Ecuaciones**
Declaración.
Definición
- **Sentencias del modelo y del Solve**
- **Impresión de sentencias** (opcional)



Estructura de un modelo GAMS (2/3)

Salidas

- Impresión
- Mapas de Referencia
- Listado de Ecuaciones
- Reportes de estado
- Resultados



Estructura de un modelo GAMS (3/3)

Algunas reglas

- Un modelo es una colección de sentencias en el lenguaje de GAMS
- Se permiten varias sentencias en una misma línea, o varias líneas para una misma sentencia, líneas en blanco
- El nombre de una entidad tiene que comenzar con una letra y puede tener hasta 9 dígitos o letras
- Cada sentencia termina con un punto y coma
- El compilador no distingue entre mayúsculas y minúsculas
- La línea que comience con un asterisco en la primera columna se considera comentario
- La creación de una entidad en GAMS lleva dos pasos:
 1. Declaración (declarar la existencia ...)
 2. Asignación o Definición (asignar un valor)



Ejemplo de un modelo de Transporte (1/2)

Planteamiento del problema

Se tiene el suministro de una cierta mercancía proporcionado por algunas plantas y las demandas de ciertos supermercados de dicha mercancía. Se conoce el costo por unidad del reparto de esta mercancía de las plantas a los supermercados. La cuestión económica es: ¿Cómo debería ser el reparto, de manera que el costo total de transporte sea mínimo?

	Distancia			Suministros
	Supermercados			
Plantas	León	Bilbao	Madrid	
Burgos	2.5	1.7	1.8	350
Salamanca	2.5	1.8	1.4	600
Demandas	325	300	275	

Ejemplo de un modelo de Transporte (2/2)

Índices:

i = plantas

j = supermercados

Datos:

a_i = suministro de mercancía de la planta i

b_j = demanda de la mercancía en el supermercado j

c_{ij} = costo por unidad del transporte entre la planta i y el supermercado j

VARIABLES DE DECISIÓN:

x_{ij} = cantidad de mercancía que se transporta de la planta i al supermercado j . Donde $x_{ij} \geq 0$, para todas las i, j

Restricciones:

$$\sum_j x_{ij} \leq a_j \quad \forall i \longleftarrow \text{Límite de suministro}$$

$$\sum_i x_{ij} \geq b_j \quad \forall j \longleftarrow \text{Satisfacer la demanda}$$

Función Objetivo:

$$\text{Min } \sum_i \sum_j c_{ij} x_{ij}$$



Sets

Corresponde a los índices en la representación algebraica de un modelo

Set

i plantas /Burgos, Salamanca/
j supermercados /Leon, Bilbao, Madrid/ ;

Set t /1991*2000/;

Set m /mach1*mach2/;

Set

i = {Burgos, Salamanca}
j = {Leon, Bilbao, Madrid};

Set i plantas /Burgos, Salamanca/;

Set j supermercados /Leon, Bilbao, Madrid/ ;



Data (1/3)

Existen tres formatos de entrada de datos: Listas, tablas y asignaciones directas

Entrada con formato de Listas

Parameters

a(i) capacidad de la planta i

/Burgos 350

Salamanca 600/

b(j) demanda de los supermercados j

/Leon 325

Bilbao 300

Madrid 275/;



Data (2/3)

Formato por Tablas

Table $d(i,j)$ "distancia en cientos de kilometros"

	Leon	Bilbao	Madrid
Burgos	2.5	1.7	1.8
Salamanca	2.5	1.8	1.4 ;



Data (3/3)

Formato por Asignación directa

Scalar f /90/;

Parameter $c(i,j)$ coste de transporte en miles por caso ;

$$c(i,j) = f * d(i,j) / 1000;$$



Variables

Cada variable tiene que tener un nombre, un dominio y un comentario (opcional)

Variables

$x(i,j)$ cantidades de mercancías
 z coste total de transporte ;

Tipo de Variable	Rango permitido
Free	$-\infty, +\infty$
Positive	$0, +\infty$
Negative	$-\infty, 0$
Binary	0 ó 1
Integer	$0, 1, \dots, 100$



Ecuaciones

Declaración de Ecuaciones

Equations


costo	define la función objetivo
suministro(i)	garantiza el límite de suministro en la planta i
demanda(j)	demanda para satisfacer al supermercado j ;

costo..	$z = e = \sum ((i,j), c(i,j) * x(i,j)) ;$
suministro(i)..	$\sum(j, x(i,j)) = l = a(i) ;$
demanda(j)..	$\sum(i, x(i,j)) = g = b(j) ;$

Definición



Notación de SUMA (PRODUCTO) en GAMS

Sum (índice de sumatoria  sumandos)

$$\sum_j x_{ij} \quad \Rightarrow \quad \text{Sum}(j, x(i,j))$$

$$\sum_i \sum_j c_{ij} x_{ij}$$



$$\text{Sum}((i,j), c(i,j) * x(i,j))$$

$$\text{Sum}(i, \text{Sum}(j, c(i,j) * x(i,j)))$$

$$\prod_j x_{ij}$$



$$\text{prod}(j, x(i,j))$$



Definición de Ecuaciones

Los componentes de la definición de una ecuación son, por orden:

- El nombre de la ecuación que se está definiendo
- El dominio
- Condición de restricción del dominio (opcional)
- El símbolo '..'
- Parte izquierda de la expresión
- Operadores de relación: =l=, =e=, =g=
- Parte derecha de la ecuación

costo..	$z =e= \text{sum} ((i,j), c(i,j)*x(i,j)) ;$
suministro(i)..	$\text{sum}(j, x(i,j)) =l= a(i) ;$
demanda(j)..	$\text{sum}(i, x(i,j)) =g= b(j) ;$



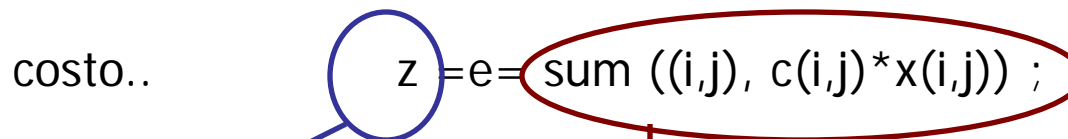
Función Objetivo

En GAMS no existe una entidad que se llame explícitamente "Función Objetivo".

Para especificar la función que se quiere optimizar, hay que crear una variable de tipo "Free" (sin restricciones sobre el signo), y su valor tiene que ser un "escalar", o sea, sin dominio.

Esta variable debe ser igualada en la sección de "Definición de ecuaciones" a la función que se quiere optimizar.

costo.. $z = e = \text{sum}((i,j), c(i,j)*x(i,j)) ;$



Esta es la variable que se iguala a la función objetivo

Esta es la ecuación que se quiere optimizar



Sentencia "Model"

En GAMS, *Model* es una colección de ecuaciones.

El formato para declarar un modelo es, la palabra clave *Model*, seguida de su nombre, y seguida de la lista de nombres de las ecuaciones que lo componen entre barras //.

model transporte /all/ ; 

En este caso se incluyen
todas las ecuaciones
definidas

model transporte /costo, suministro, demanda/ ;

Ahora se debe llamar al Solver para la solución del modelo ...



Sentencia "Solve"

El formato de la sentencia *Solve* es el siguiente (por orden):

- La palabra clave "*solve*"
- El nombre del modelo que se desea resolver
- La palabra clave "*using*"
- El procedimiento de solución disponible
- La palabra clave "*minimizing*" o "*miximizing*"
- El nombre de la variable que se va a optimizar

solve transporte *using* lp *minimizing* z ;



Procedimientos disponibles

- **lp** para Programación Lineal
- **nlp** para Programación No Lineal
- **mip** para Programación Mixta Entera
- **rmip** para Programación Mixta Entera Relajada
- **minlp** para Programación No Lineal Mixta Entera
- **mcp** para Problemas Complementarios Mixtos
- **cns** para Sistemas No lineales con Restricciones



Visualización de Sentencias

Cuando el **solver** se está ejecutando ocurren cosas ...

1. Se generan instancias específicas del modelo
2. Se crean las estructuras de datos de entrada al solver
3. Las salidas del solver se guardan a un archivo

Para obtener los valores óptimos de las variables, o visualizar los resultados podemos hacer lo siguiente:

display x.l , x.m ; valor final y marginal de x(i,j)



Las bases de datos: “.LO, .L, .UP, .M”

GAMS ha sido diseñada con un pequeño sistema de bases de datos, en el cual los “records” se mantienen para las ecuaciones y las variables

- .lo** = límite inferior
- .l** = valor de nivel o primario
- .up** = límite superior
- .m** = valor marginal o dual

El formato para hacer referencia a estas cantidades es poner el *nombre de la variable* o de la *ecuación* seguido del *campo del nombre*, y si es necesario también *del dominio*, o de uno de los elementos del dominio

display **x.l** , **x.m** ;

Asignación de límites y valores iniciales

Aunque los límites de las variables se toman automáticamente dependiendo del tipo de la variable, el usuario puede especificarlos:

x.up(i,j) = capacity(i,j) ;

x.lo(i,j) = 10.0 ;

x.up('Burgos', 'Leon') = 1.2*capacity('Burgos' , 'Leon') ;

En estos casos capacity(i,j) es un parámetro que ha sido declarado y cuyos valores han sido asignados previamente

Estas sentencias tienen que aparecer después de la zona de declaración de las variables, y antes de la sentencia solve

Los campos **.up** y **.lo** están bajo el control del usuario, mientras que **.l** y **.m**, pueden ser inicializados por el usuario, pero son controlados por el solver