GAMS Tutorial

Systems and Computing Seminar

Germán Montoya
ga.montoya44@uniandes.edu.co
Ph.D Student
Universidad de Los Andes
Bogotá D.C, Colombia
Oct.13/2015

Universidad de los Andes

Generalities

Universidad de los Andes

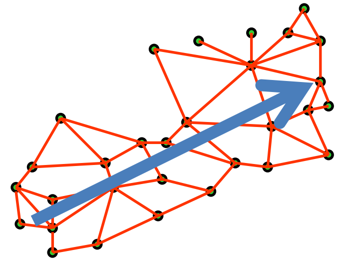
What is GAMS?

- The General Algebraic Modeling System (GAMS) is a high-level modeling system for mathematical optimization. GAMS is designed for modeling and solving linear, nonlinear, and mixed-integer optimization problems.

- GAMS contains an integrated development environment (IDE) and is connected to a group of third-party optimization solvers. Among these solvers are BARON, COIN-OR solvers, CONOPT, CPLEX, DICOPT, GUROBI, MOSEK, SNOPT, SULUM, and XPRESS.

General Purpose

Real-life problem



Mathematical Model

$$Min(d) = \sum_{(i,j)\in E} c_{ij} x_{ij}$$

$$S.A.$$

$$\sum_{(i,j)\in E} x_{ij} = 1, i = s$$

$$\sum_{(i,j)\in E} x_{ij} = 1, j = t$$

$$\sum_{(i,j)\in E} x_{ij} - \sum_{(j,i)\in E} x_{ji} = 0, i \neq s,t$$

$$x_{ij} \in \Im, 0/1$$

Implementation

```
Variables
  x(i,j)      Indica si el enlace i j es utilizado o no en el SP
  z           minimizacion ;
Binary Variable x;

Equations
Camino_Mas_Corto       Funcion Objetivo
nodo_origen(i)         nodo origen
nodo_destino(j)        nodo destino
nodo_intermedio        nodo intermedio;

Camino_Mas_Corto                    ..  z =e= sum((i,j), c(i,j) * x(i,j));

nodo_origen(i)$(ord(i) = 1)         ..  sum((j), x(i,j)) =e= 1;

nodo_destino(j)$(ord(j) = 5)        ..  sum((i), x(i,j)) =e= 1;

nodo_intermedio(i)$(ord(i) ne 1 and ord(i) ne 5)
                                    ..  sum((j), x(i,j)) - sum((j), x(i,j)) =e= 0;

Model Transport /all/ ;
option mip=CPLEX
Solve transport using mip minimizing z;

Display x.l
Display z.l
```

Download

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

Solution Architecture

Universidad de los Andes

Solution Architecture

- File types:
  - *.gms

```
Variables
  x(i,j)      Indicates if the link i-j is selected or not.
  z           Objective function  ;

Binary Variable x;

Equations
objectiveFunction        objective function
sourceNode(i)            source node
destinationNode(j)       destination node
intermediateNode         intermediate node;

objectiveFunction                               ..  z =e= sum((i,j), c(i,j) * x(i,j));

sourceNode(i)$(ord(i) = 1)                      ..  sum((j), x(i,j)) =e= 1;

destinationNode(j)$(ord(j) = 5)                 ..  sum((i), x(i,j)) =e= 1;

intermediateNode(i)$(ord(i) <> 1 and ord(i) ne 5)  ..  sum((j), x(i,j)) - sum((j), x(j,i)) =e= 0;

Model model1 /all/ ;
option mip=CPLEX
Solve model1 using mip minimizing z;

Display x.l;
Display z.l;
```

  - *.log

```
--- Job SP_Hops_v2.GMS Start 09/28/15 02:41:12 WEX-WEI 24.0.2 x86_64/MS Windows
GAMS Rev 240  Copyright (C) 1987-2013 GAMS Development. All rights reserved
Licensee: Ingenieria Quimica                              G120216:1205AP-WIN
          Universidad de los Andes                                  DC9561
          License for teaching and research at degree granting institutions
--- Starting compilation
--- SP_Hops_v2.GMS(44) 3 Mb
--- Starting execution: elapsed 0:00:00.003
--- SP_Hops_v2.GMS(40) 4 Mb
--- Generating MIP model Transport
--- SP_Hops_v2.GMS(41) 4 Mb
---   3 rows  26 columns  36 non-zeroes
---   25 discrete-columns
--- Executing CPLEX: elapsed 0:00:00.026

IBM ILOG CPLEX   Feb 14, 2013 24.0.2 WEX 38380.38394 WEI x86_64/MS Windows
Cplex 12.5.0.0
                                                     Double-Click to Open File
Reading data...
Starting Cplex...
Tried aggregator 2 times.
MIP Presolve eliminated 1 rows and 24 columns.
Aggregator did 2 substitutions.
All rows and columns eliminated.
Presolve time = 0.00 sec. (0.01 ticks)
MIP status(101): integer optimal solution
Cplex Time: 0.01sec (det. 0.02 ticks)
Fixing integer variables, and solving final LP...
Tried aggregator 1 time.
LP Presolve eliminated 3 rows and 26 columns.
```

Solution Architecture

- File types:
  - *.lst

```
----     47 VARIABLE x.L  Indicates if the link i-j is selected or not.

               n2          n5

n1          1.000
n2                      1.000


----     48 VARIABLE z.L                   =        2.000  Objective function


EXECUTION TIME       =        0.000 SECONDS      3 Mb  WEX240-240 Feb 14, 2013
```

Model Components

Example 1

- From 5 saleable goods, buy the ones that incur the minimal possible cost, taking into account our budget, that is, 10 monetary units.

  Goods' values: 12, 5, 9, 6 y 4 respectively.

- Mathematical model:

$$value_i: parameter. Value\ of\ each\ article.$$

$$x_i: variable, where\ x_i = \begin{cases} 1\ if\ the\ article\ is\ bought \\ 0\ if\ not \end{cases}$$

$$\underbrace{value_1 * x_1 + value_2 * x_2 \ldots value_5 * x_5}$$

$$min\ (value_1 * x_1 + value_2 * x_2 \ldots value_5 * x_5)$$

$$value_1 * x_1 + value_2 * x_2 \ldots value_5 * x_5 = BUDGET$$

Example 1

- Mathematical model:

$value_i$: $parameter. Value\ of\ each\ article.$

$$x_i: variable, where\ x_i = \begin{cases} 1\ if\ the\ article\ is\ bought \\ 0\ if\ not \end{cases}$$

$$min\ (value_1 * x_1 + value_2 * x_2 \ldots value_5 * x_5) \Rightarrow min \sum_{i \in A} value_i * x_i$$

$$\underbrace{value_1 * x_1 + value_2 * x_2 \ldots value_5 * x_5 = BUDGET}_{\sum_{i \in A} value_i * x_i = BUDGET}$$

Universidad de los Andes

GAMS Implementation

- Mathematical model:

$value_i$: parameter. Value of each article.

x_i: variable, where $x_i = \begin{cases} 1 \text{ if the article is bought} \\ 0 \text{ if not} \end{cases}$

$$\min \sum_{i \in A} value_i * x_i$$ → *Objective Function*

$$\sum_{i \in A} value_i * x_i = BUDGET$$ → *Constraint*

- GAMS:

```
*Model1

Set i   articles / a1, a2, a3, a4, a5 /;

Scalar BUDGET budget /10/;

Parameter  value(i)   value of each article
                   /  a1 12, a2 5, a3 9, a4 6, a5 4  /;

Variables
  x(i)          Inidicates if the article is bought or not
  z             objective function;

Binary Variable x;

Equations
objectiveFunction                         objective function
budgetConstraint                          budget constraint;

objectiveFunction       ..      z =e= sum(i, value(i) * x(i));

budgetConstraint        ..      sum(i, value(i) * x(i)) =e= BUDGET;

Model Model1 /all/ ;

option mip=CPLEX
Solve Model1 using mip minimizing z

Display x.l;
Display z.l;
```

Model Components

```
*Model1

Set i   articles / a1, a2, a3, a4, a5 /;
```

Sets

```
Scalar BUDGET budget /10/;

Parameter  value(i)   value of each article
                    /  a1 12, a2 5, a3 9, a4 6, a5 4  /;
```

Parameters

```
Variables
  x(i)          Inidicates if the article is bought or not
  z             objective function;
```

Variables

```
Binary Variable x;

Equations
objectiveFunction                 objective function
budgetConstraint                  budget constraint;
```

Equations declaration

```
objectiveFunction       ..      z =e= sum(i, value(i) * x(i));

budgetConstraint        ..      sum(i, value(i) * x(i)) =e= BUDGET;
```

Equations definition

```
Model Model1 /all/ ;

option mip=CPLEX
Solve Model1 using mip minimizing z
```

Solver configuration

```
Display x.l;
Display z.l;
```

Results visualization in the *.lst file

Universidad de los Andes

Model Components

- *.lst file contents:
  - A copy of the mathematical model

```
*Modelo1

Set i   articles / a1, a2, a3, a4, a5 /;

Scalar BUDGET budget /10/;

Parameter  value(i)   value of each article
                    /  a1 12, a2 5, a3 9, a4 6, a5 4  /;

Variables
  x(i)          if the article is bought
  z             objective function;

Binary Variable x;

Equations
objectiveFunction               objective function
budgetConstraint                budget constraint;

objectiveFunction       ..      z =e= sum(i, value(i) * x(i));

budgetConstraint        ..      sum(i, value(i) * x(i)) =e= BUDGET;

Model Model1 /all/ ;

option mip=CPLEX
Solve Model1 using mip minimizing z

Display x.l;
Display z.l;
```

Model Components

- *.lst file contents:
  - Equations

```
---- objectiveFunction  =E=  objective function

objectiveFunction..  - 12*x(a1) - 5*x(a2) - 9*x(a3) - 6*x(a4) - 4*x(a5) + z =E=
     0 ; (LHS = 0)


---- budgetConstraint  =E=  budget constraint

budgetConstraint..  12*x(a1) + 5*x(a2) + 9*x(a3) + 6*x(a4) + 4*x(a5) =E= 10 ;

      (LHS = 0, INFES = 10 ****)
```

Model Components

- *.lst file contents:
  - Solver

```
IBM ILOG CPLEX   Feb 14, 2013 24.0.2 WEX 38380.38394 WEI x86_64/MS Windows
Cplex 12.5.0.0

MIP status(101): integer optimal solution
Cplex Time: 0.00sec (det. 0.01 ticks)
Fixing integer variables, and solving final LP...
Fixed MIP status(1): optimal
Cplex Time: 0.00sec (det. 0.00 ticks)
Proven optimal solution.

MIP Solution:          10.000000    (0 iterations, 0 nodes)
Final Solve:           10.000000    (0 iterations)
```

Model Components

- *.lst file contents:
  - Results

```
----      29 VARIABLE x.L  if the article is bought

a4 1.000,    a5 1.000


----      30 VARIABLE z.L                    =       10.000  objective function
```

Sets

Sets

- Five elements:

```
Set i   articles / a1, a2, a3, a4, a5 /;
Set i   articles / a1*a5 /;
Set i   articles / 1*5 /;
```

- Create a copy set:

```
Set i   articles / 1*5 /;

alias(j,i);
```

- Parameterize the number of elements:

```
$Set NARTICLES  10

Set i   articles / a1*a%NARTICLES% /;

Set i   articles / 1*%NARTICLES% /;
```

- Description of elements:

```
Set i articles
/a1   article 1
 a2   article 2
 a3   article 3
 a4   article 4
 a5   article 5
/;
```

Sets

- Subsets:

```
$Set TOTALARTICLES  10

Set k   articles / a1*a%TOTALARTICLES% /;
Set i(k) five articles /a1*a5/;
Set j(k) four articles /a2*a5/;
```

- One-to-one Mapping: $A = \{(b, d),\ (a, c),\ (c, e)\}.$

```
set c countries
/ jamaica
haiti
guyana
brazil / ;

set p ports
/ kingston
s-domingo
georgetown
belem / ;

set ptoc(p, c) port to country relationship
/ kingston .jamaica
s-domingo .haiti
georgetown .guyana
belem .brazil /;

Display c;
Display p;
Display ptoc;
```

```
----     25 SET c  countries

jamaica,    haiti  ,    guyana ,    brazil


----     26 SET p  ports

kingston  ,    s-domingo ,    georgetown,    belem


----     27 SET ptoc  port to country relationship

               jamaica       haiti       guyana      brazil

kingston           YES
s-domingo                      YES
georgetown                                  YES
belem                                                   YES
```

Sets

- Many-to-many Mapping:

```
set i / a, b /
j / c, d, e /
ij1(i,j) /a.c, a.d/
ij2(i,j) /a.c, b.c/
ij3(i,j) /a.c, b.c, a.d, b.d/ ;

Display i,j,ij1,ij2,ij3;
```

```
set i / a, b /
j / c, d, e /
ij1(i,j) /a.(c,d)/
ij2(i,j) /(a,b).c/
ij3(I,j) /(a,b).(c,d)/ ;
```

```
----     36 SET ij1

            c           d

a         YES         YES


----     36 SET ij2

            c

a         YES
b         YES


----     36 SET ij3

            c           d

a         YES         YES
b         YES         YES
```

Sets

- Many-to-many Mapping:
  - Exercise:

Construct	*Result*
(a,b).c.d	
(a,b).(c,d) .e	
(a.1*3).c	
1*3. 1*3. 1*3	

Sets

- Many-to-many Mapping:

  – Exercise:

Construct	*Result*
(a,b).c.d	a.c.d, b.c.d
(a,b).(c,d) .e	a.c.e, b.c.e, a.d.e, b.d.e
(a.1*3).c	(a.1, a.2, a.3).c or a.1.c, a.2.c, a.3.c
1*3. 1*3. 1*3	1.1.1, 1.1.2, 1.1.3, ..., 3.3.3

Data Entry

Parameters, Scalars &
Tables

Data Entry

- Scalars:

```
Scalar BUDGET budget /10/;

$Set BUDGET  10

Scalar BUDGET presupuesto /%BUDGET%/;
```

- Parameters:

```
Parameter  value(i)   value of each article
                    /  a1 12, a2 5, a3 9, a4 6, a5 4  /;
```

  - Parameter Data for Higher Dimensions:

```
parameter salaries(employee,manager,department)
         /anderson .murphy .toy = 6000
          hendry .smith .toy = 9000
          hoffman .morgan .cosmetics = 8000 / ;
```

  - Exercise:

```
Set row / row1*row10 /
    col / col1*col10 / ;

Parameter a(row, col)
        / (row1,row4) . col2*col7 12
          row10 . col10 17
          row1*row7 . col10 33 / ;
```

Universidad de los Andes

Data Entry

– Exercise:

```
Set row / row1*row10 /
    col / col1*col10 / ;

Parameter a(row, col)
        / (row1,row4) . col2*col7 12
          row10 . col10 17
          row1*row7 . col10 33 / ;
```

```
----      64 PARAMETER a

             col2        col3        col4        col5        col6        col7

row1       12.000      12.000      12.000      12.000      12.000      12.000
row4       12.000      12.000      12.000      12.000      12.000      12.000

    +       col10

row1       33.000
row2       33.000
row3       33.000
row4       33.000
row5       33.000
row6       33.000
row7       33.000
row10      17.000
```

Data Entry

- Tables:

$$\sum_{i\in P}\sum_{j\in B} C_{ij}X_{ij} \longrightarrow C_{ij}\ ?$$

```
Table c(i,j) link cost
                n1      n2      n3      n4      n5
n1              999     1       1       999     999
n2              999     999     999     999     1
n3              999     999     999     1       999
n4              999     999     999     999     1
n5              999     999     999     999     999;
```

```
----     45 PARAMETER c  link cost

            n1          n2          n3          n4          n5

n1     999.000       1.000       1.000     999.000     999.000
n2     999.000     999.000     999.000     999.000       1.000
n3     999.000     999.000     999.000       1.000     999.000
n4     999.000     999.000     999.000     999.000       1.000
n5     999.000     999.000     999.000     999.000     999.000
```

Data Entry

- Tables:
  - More than two dimensions?

```
table upgrade(strat,size,tech)
                   small.tech1 small.tech2 medium.tech1 medium.tech2
strategy-1               .05         .05          .05           .05
strategy-2               .2          .2           .2            .2
strategy-3               .2          .2           .2            .2
strategy-4                                        .2            .2
```

```
Parameter upgrade(strat,size,tech);
upgrade('strategy-1','small','tech1')=0.05;
upgrade('strategy-1','small','tech2')=0.05;
upgrade('strategy-1','medium','tech1')=0.05;
upgrade('strategy-1','medium','tech2')=0.05;
```

Variables

Universidad de los Andes

Variables

- Declaration:

```
Variables
  x(i)        Inidicates if the article is bought or not
  z           objective function;
```

- Variable type:

```
Binary Variable x;
```

- Variable types:

Keyword	*Default Lower Bound*	*Default Upper Bound*	*Description*
`free` (default)	`-inf`	`+inf`	No bounds on variable. Both bounds can be changed from the default values by the user
`positive`	`0`	`+inf`	No negative values are allowed for variable. The user can change the upper bound from the default value.
`negative`	`-inf`	`0`	No positive values are allowed for variables. The user can change the lower bound from the default value.
`binary`	`0`	`1`	Discrete variable that can only take values of 0 or 1
`integer`	`0`	`100`	Discrete variable that can only take integer values between the bounds. The user can change bounds from the default value.

Variables

```
Variables
  x(i)          Inidicates if the article is bought or not
  z             objective function;

Binary Variable x;
    .

    .

    .
Display x.l;
```

```
----     29 VARIABLE x.L  if the article is bought

a4 1.000,    a5 1.000


----     30 VARIABLE z.L                  =       10.000  objective function
```

Equations

Equations

- Declaration and definition:

```
Equations
objectiveFunction                    objective function
budgetConstraint                     budget constraint;

objectiveFunction        ..          z =e= sum(i, value(i) * x(i));
budgetConstraint         ..          sum(i, value(i) * x(i)) =e= BUDGET;
```

Declarations

Definitions

$$\sum_{i \in A} value_i * x_i = BUDGET$$

- Relational operators: =e=equal

  =g=greater than or equal to

  =l=less than or equal to

Equations

- Forall:

$$min \sum_{i \in P} \sum_{j \in B} C_{ij} X_{ij}$$

$$\sum_{j \in B} X_{ij} \leq a_i \quad \forall i \in P$$

$$\sum_{i \in P} X_{ij} \geq b_j \quad \forall j \in B$$

```
Equations
cost define objective function
supply(i) observe supply limit at plant i
demand(j) satisfy demand at market j ;

cost .. z =e= sum((i,j), c(i,j)*x(i,j)) ;
supply(i) .. sum(j, x(i,j)) =l= a(i) ;
demand(j) .. sum(i, x(i,j)) =g= b(j) ;
```

Model and Solve Statements

```
Model Model1 /all/ ;

Model Model1 /cost, supply, demand/ ;

Model Model1 /cost, supply/ ;
```

```
cost .. z =e= sum((i,j), c(i,j)*x(i,j)) ;
supply(i) .. sum(j, x(i,j)) =l= a(i) ;
demand(j) .. sum(i, x(i,j)) =g= b(j) ;
```

```
option lp=CPLEX

Solve Model1 using lp minimizing z;
```

```
Display x.l;

Display z.l;
```

Solution for ‘x’ and ‘z’.
‘l’ : level

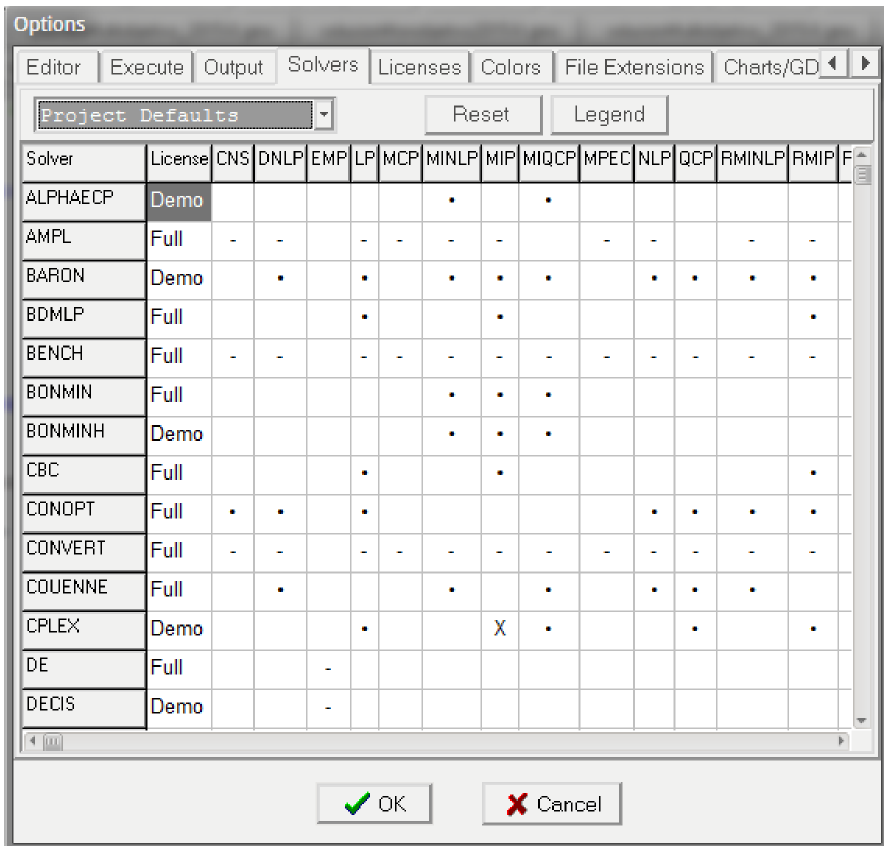
Model and Solve Statements

- Problem type:

lp	for linear programming
qcp	for quadratic constraint programming
nlp	for nonlinear programming
dnlp	for nonlinear programming with discontinuous derivatives
mip	for mixed integer programming
rmip	for relaxed mixed integer programming
miqcp	for mixed integer quadratic constraint programming
rmiqcp	for relaxed mixed integer quadratic constraint programming
minlp	for mixed integer nonlinear programming
rminlp	for relaxed mixed integer nonlinear programming
mcp	for mixed complementarity problems
mpec	for mathematical programs with equilibrium constraints
rmpec	for relaxed mathematical program with equilibrium constraints
cns	for constrained nonlinear systems
emp	for extended mathematical programming

Model and Solve Statements

- Problem type:
  - File>Options>Solvers



Solver	License	CNS	DNLP	EMP	LP	MCP	MINLP	MIP	MIQCP	MPEC	NLP	QCP	RMINLP	RMIP	F
ALPHAECP	Demo						•		•						
AMPL	Full	-	-		-	-	-	-		-	-		-	-	
BARON	Demo		•		•		•	•	•		•	•	•	•	
BDMLP	Full				•			•						•	
BENCH	Full	-	-		-	-	-	-	-	-	-	-	-	-	
BONMIN	Full						•	•	•						
BONMINH	Demo						•	•	•						
CBC	Full				•			•						•	
CONOPT	Full	•	•		•						•	•	•	•	
CONVERT	Full	-	-		-	-	-	-	-	-	-	-	-	-	
COUENNE	Full		•				•		•		•	•	•		
CPLEX	Demo				•			X	•			•		•	
DE	Full			-											
DECIS	Demo			-											

Solver Selection Legend

X Current selection

• Available selection

- Not available (Option statement only)

Conditionals

Conditionals

- a$(b > 1.5) = 2 ;
  - if (b > 1.5), then a = 2

- Relational operators:

  le, <= less than or equal to

  lt, < strictly less than

  eq, = equal to

  ne, <> not equal to

  ge, >= greater than or equal to

  gt, > strictly greater than

Conditionals

- $ operator in parameters:

```
b= sum(i$(a(i) ne 1), a(i)) ;
```

```
rho(i)$(sig(i) ne 0) = sig(i) - 1;
```

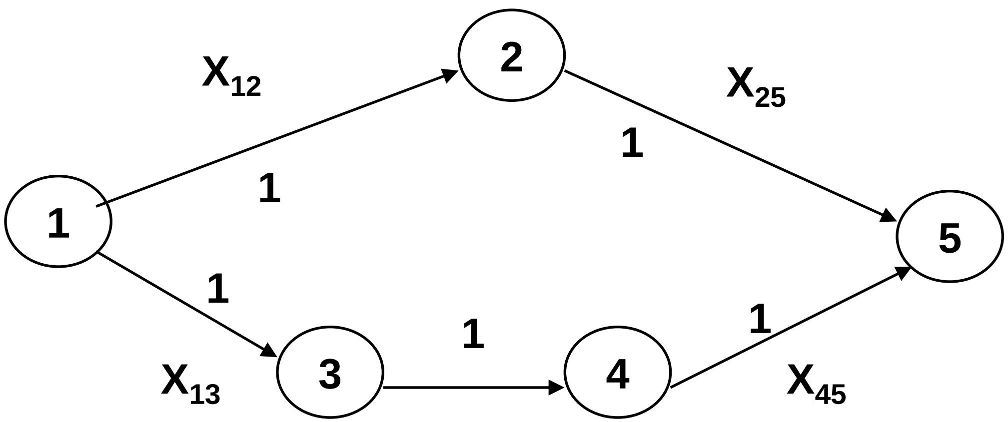
  - It is equivalent to: `rho(i)$(sig(i)) = sig(i) - 1`

- $ operator in constraints:

```
Eq2(i)$(cost(i)=5) .. x(i)=l=a(i) ;

Eq3(i)$(ord(i)<>1) .. x(i)=l=a(i) ;
```

Conditionals

- Example 2:



$$min \sum_{i \in N} \sum_{j \in N} x_{ij}$$

Subject to:

$$\sum_{j \in N} x_{ij} = 1, i = s$$

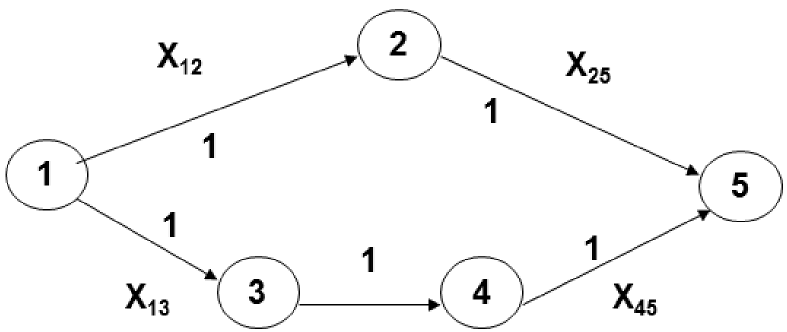
$$\sum_{i \in N} x_{ij} = 1, j = d$$

$$\sum_{j \in N} x_{ij} - \sum_{j \in N} x_{ji} = 0, i \neq s, d$$

$$x_{ij} \in Z, 0/1$$

Conditionals

- Example 2:

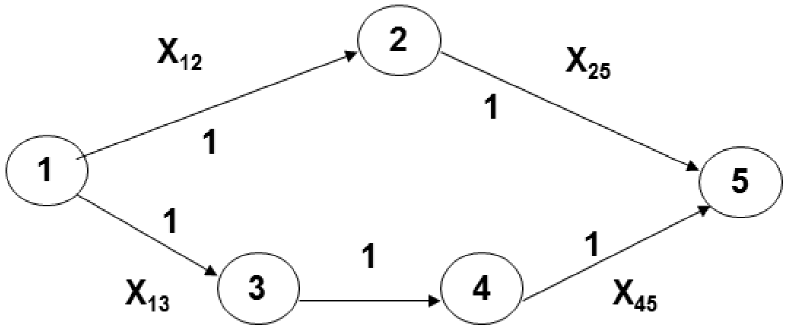


s=1
d=5

```
Sets
  i   network nodes / n1, n2, n3, n4, n5 /

alias(j,i);

Table c(i,j) link cost
                  n1        n2        n3        n4        n5
n1                999       1         1         999       999
n2                999       999       999       999       1
n3                999       999       999       1         999
n4                999       999       999       999       1
n5                999       999       999       999       999;


Variables
  x(i,j)       Indicates if the link i-j is selected or not.
  z            Objective function  ;

Binary Variable x;
```

Conditionals

- Example 2:



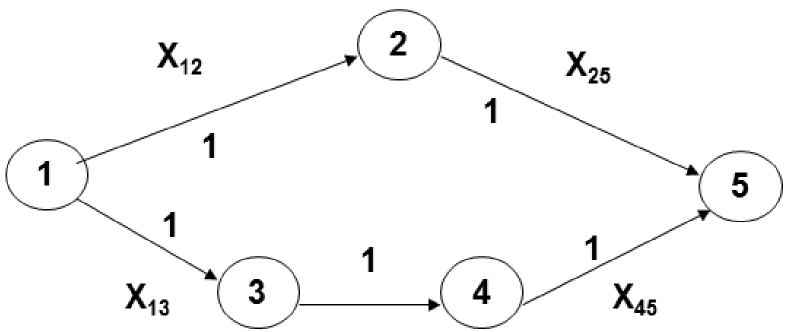
s=1

d=5

```
Equations
objectiveFunction        objective function
sourceNode               source node
destinationNode          destination node
intermediateNode2        intermediate node2
intermediateNode3        intermediate node3
intermediateNode4        intermediate node4;

objectiveFunction    ..  z =e= sum((i,j), c(i,j) * x(i,j));

sourceNode           ..  sum((j), x('n1',j)) =e= 1;

destinationNode      ..  sum((i), x(i,'n5')) =e= 1;

intermediateNode2    ..  sum((i,j), x(i,'n2')) - sum((i,j), x('n2',j)) =e= 0;
intermediateNode3    ..  sum((i,j), x(i,'n3')) - sum((i,j), x('n3',j)) =e= 0;
intermediateNode4    ..  sum((i,j), x(i,'n4')) - sum((i,j), x('n4',j)) =e= 0;

Model Model1 /all/ ;
option mip=CPLEX
Solve Model1 using mip minimizing z;

Display x.l
Display z.l
```

Conditionals

- Example 2: generic version



s=1

d=5

```
Equations
objectiveFunction          objective function
sourceNode(i)              source node
destinationNode(j)         destination node
intermediateNode           intermediate node;

objectiveFunction                                   ..  z =e= sum((i,j), c(i,j) * x(i,j));

sourceNode(i)$(ord(i) = 1)                          ..  sum((j), x(i,j)) =e= 1;

destinationNode(j)$(ord(j) = 5)                     ..  sum((i), x(i,j)) =e= 1;

intermediateNode(i)$(ord(i) <> 1 and ord(i) ne 5)  ..  sum((j), x(i,j)) - sum((j), x(j,i)) =e= 0;

Model model1 /all/ ;
option mip=CPLEX
Solve model1 using mip minimizing z;

Display x.l, c;
Display z.l;
```

```
----     45 VARIABLE x.L  Indicates if the link i-j is selected or not.

            n2          n5

n1       1.000
n2                   1.000


----     46 VARIABLE z.L                   =        2.000  Objective function
```

Programming Flow Control Features

Programming Flow Control Features

- Loop:

```
set t / 1985*1990 /
parameter pop(t) / 1985 3456 /
growth(t) / 1985 25.3, 1986 27.3, 1987 26.2
            1988 27.1, 1989 26.6, 1990 26.6 /;
loop(t,
        pop(t+1) = pop(t) + growth(t) ) ;
```

- Conditional loop:

```
loop(i$(curacc > reltol),
        value(i+1) = 0.5*(value(i) + target/value(i));
        sqrtval = value(i+1);
        curacc = abs (value(i+1)-value(i))/(1+abs(value(i+1)))
) ;
```

  - One cannot make declarations or define equations inside a loop statement.
  - It is illegal to modify any controlling set inside the body of the loop.

Programming Flow Control Features

- If-elseif-else:

```
if (f <= 0,
    p(i) = -1 ;
    q(j) = -1 ;
elseif ((f > 0) and (f < 1)),
    p(i) = p(i)**2 ;
    q(j) = q(j)**2 ;
else
    p(i) = p(i)**3 ;
    q(j) = q(j)**3 ;
    ) ;
```

  - One cannot make declarations or define equations inside an if statement.

Programming Flow Control Features

- while:

```
scalar count ; count = 1 ;
scalar globmin ; globmin = inf ;
while((count le 1000),
    a(j) = uniform(0,1) ;
    solve ml using nlp minimizing obj ;
    if (obj.l le globmin,
       globmin = obj.l ;
       ) ;
    count = count+1 ;
    ) ;
```

  - One cannot make declarations or define equations inside a while statement.

Programming Flow Control Features

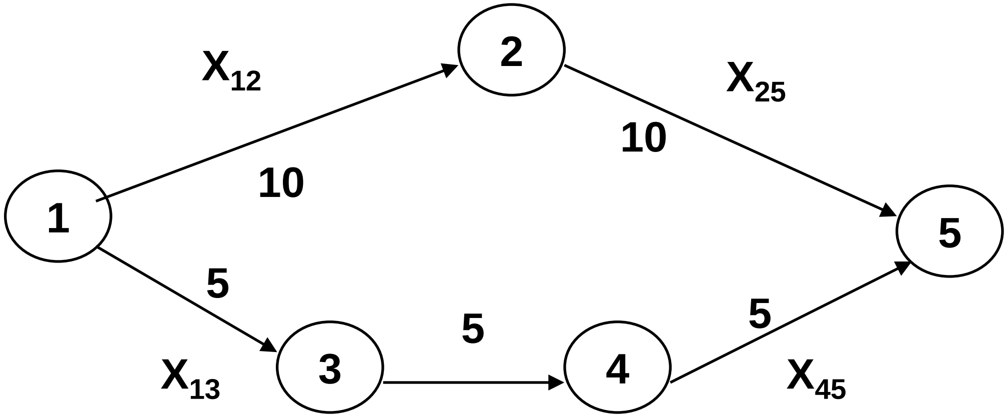
- for:

```
scalar i ;
scalar globmin ; globmin = inf ;
for (i = 1 to 1000,
   a(j) = uniform(0,1) ;
   solve ml using nlp minimizing obj ;
   if (obj.l le globmin,
     globmin = obj.l ;
     globinit(j) = x.l(j) ;
   );
);
```

  – One cannot make declarations or define equations inside a for statement.

Multiobjective Optimization

- Concept: Example
  - Function 1: minimize hops
  - Function 2: minimize cost



Multiobjective Optimization

- Theoretical basis:

Optimize [minimize/maximize]

$$F(X) = \{f_1(X), f_2(X), \ \ldots, f_n(X)\}$$

subject to

$$H(X) = 0$$

$$G(X) \geq 0$$

Multiobjective Optimization

- Weighted Sum Method:

Optimize [minimize/maximize]

$$F'(X) = \sum_{i=1}^{n} r_i * f_i(X)$$

subject to

$$H(X) = 0$$

$$G(X) \geq 0$$

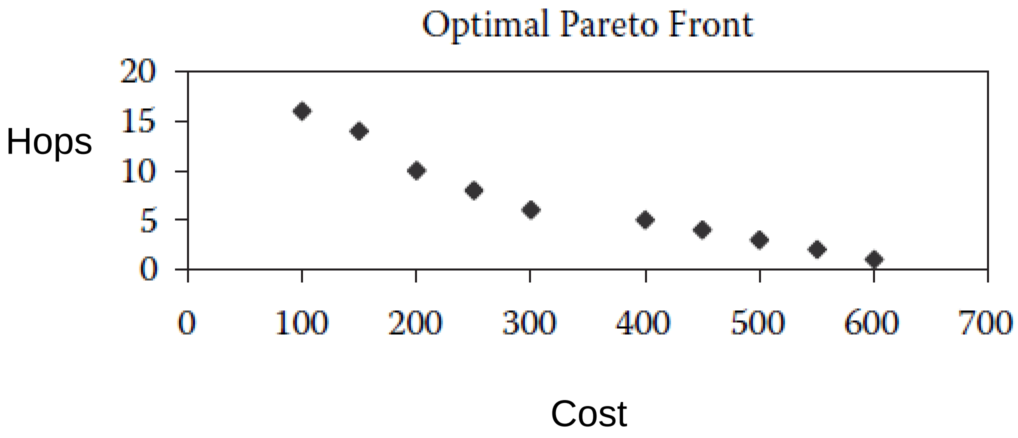
$$0 \leq r_i \leq 1,\ i = \{1, \ldots, n\}$$

$$\sum_{i=1}^{n} r_i = 1$$

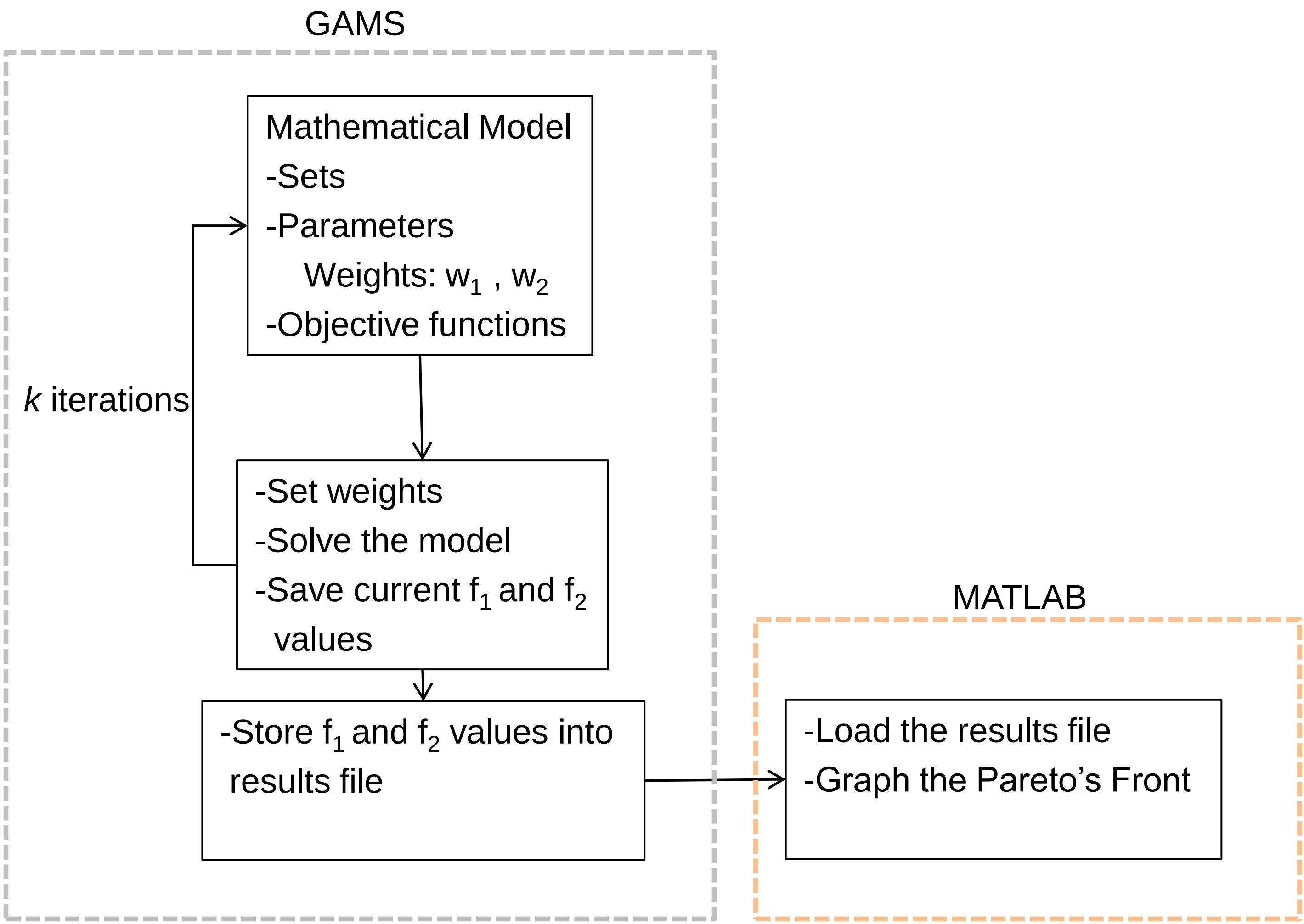
Multiobjective Optimization

- Weighted Sum Method:

$$F(X) = r_1 \cdot f_1(X) + r_2 \cdot f_2(X)$$



Multiobjective Optimization



Multiobjective Optimization

- Example 3:

Suppose there are two types of packets in a network: without priority and with priority. The network has three source nodes and four destination nodes. From the source nodes it is required to send 60, 80 and 50 packets without priority, respectively, and 20, 20 and 30 packets with priority respectively. At the destination nodes are requested 50, 90, 40 and 10 packets without priority respectively, and 10, 20, 10 and 30 packets with priority, respectively. The sending costs from source nodes to destination nodes are presented in the following table:

Source node	Destination 1	Destination 2	Destination 3	Destination 4
1	10	9	10	11
2	9	10	11	10
3	11	9	10	10

In addition, it is necessary to take into account the sending delay from source nodes to destination nodes:

Source node	Destination 1	Destination 2	Destination 3	Destination 4
1	12	14	10	11
2	11	8	7	13
3	6	11	4	15

Propose a multiobjective optimization model for minimizing the sending cost and the sending delay, finding the Pareto front using the Weighted Sum method.

Multiobjective Optimization

- ## Example 3:

Suppose there are two types of packets in a network: without priority and with priority. The network has three source nodes and four destination nodes. From the source nodes it is required to send 60, 80 and 50 packets without priority, respectively, and 20, 20 and 30 packets with priority respectively. At the destination nodes are requested 50, 90, 40 and 10 packets without priority respectively, and 10, 20, 10 and 30 packets with priority, respectively. The sending costs from source nodes to destination nodes are presented in the following table:

Source node	Destination 1	Destination 2	Destination 3	Destination 4
1	10	9	10	11
2	9	10	11	10
3	11	9	10	10

In addition, it is necessary to take into account the sending delay from source nodes to destination nodes:

Source node	Destination 1	Destination 2	Destination 3	Destination 4
1	12	14	10	11
2	11	8	7	13
3	6	11	4	15

Propose a multiobjective optimization model for minimizing the sending cost and the sending delay, finding the Pareto front using the Weighted Sum method.

Sets:

i: set of packet types
j: set of source nodes
k: set of destination nodes

Parameters:

c_{jk}	d_1	d_2	d_3	d_4
s_1	10	9	10	11
s_2	9	10	11	10
s_3	11	9	10	10

t_{jk}	d_1	d_2	d_3	d_4
s_1	12	14	10	11
s_2	11	8	7	13
s_3	6	11	4	15

inv_{ij}	s_1	s_2	s_3
p_1	60	80	50
p_2	20	20	30

dem_{ik}	d_1	d_2	d_3	d_4
p_1	50	90	40	10
p_2	10	20	10	30

Multiobjective Optimization

- Example 3:

Suppose there are two types of packets in a network: without priority and with priority. The network has three source nodes and four destination nodes. From the source nodes it is required to send 60, 80 and 50 packets without priority, respectively, and 20, 20 and 30 packets with priority respectively. At the destination nodes are requested 50, 90, 40 and 10 packets without priority respectively, and 10, 20, 10 and 30 packets with priority, respectively. The sending costs from source nodes to destination nodes are presented in the following table:

Source node	Destination 1	Destination 2	Destination 3	Destination 4
1	10	9	10	11
2	9	10	11	10
3	11	9	10	10

In addition, it is necessary to take into account the sending delay from source nodes to destination nodes:

Source node	Destination 1	Destination 2	Destination 3	Destination 4
1	12	14	10	11
2	11	8	7	13
3	6	11	4	15

Propose a multiobjective optimization model for minimizing the sending cost and the sending delay, finding the Pareto front using the Weighted Sum method.

Variables: $x_{ijk} \in Re^+$

$$f_1 = \sum_{ijk} c_{jk} * x_{ijk} \qquad f_2 = \sum_{ijk} t_{jk} * x_{ijk}$$

Objective function:

$$\min\,(w_1 * f_1 + w_2 * f_2)$$

$$where\ w_1 + w_2 = 1$$

Constraints:

$$\sum_k x_{ijk} \le inv_{ij} \quad \forall i,j$$

$$\sum_j x_{ijk} = dem_{ik} \quad \forall i,k$$

Multiobjective Optimization

- Example 3:

Sets:

i: set of packet types
j: set of source nodes
k: set of destination nodes

Definition of weights:

Parameters:

c_{jk}	d_1	d_2	d_3	d_4
s_1	10	9	10	11
s_2	9	10	11	10
s_3	11	9	10	10

t_{jk}	d_1	d_2	d_3	d_4
s_1	12	14	10	11
s_2	11	8	7	13
s_3	6	11	4	15

inv_{ij}	s_1	s_2	s_3
p_1	60	80	50
p_2	20	20	30

dem_{ik}	d_1	d_2	d_3	d_4
p_1	50	90	40	10
p_2	10	20	10	30

```
Set i   packet types / p1, p2 /;
Set j   source nodes  / s1, s2, s3 /;
Set k   destination nodes / d1, d2, d3, d4 /;

set iter iterations /it1*it11/;
scalar w1 weight 1 / 0 /;
scalar w2 weight 2 / 0 /;

parameter w1_vec(iter) w1 values
                  /it1 1, it2 0.9, it3 0.8, it4 0.7, it5 0.6, it6 0.5,
                   it7 0.4, it8 0.3, it9 0.2, it10 0.1, it11 0/;
parameter w2_vec(iter) w2 values;

Table c(j,k) sending cost
                  d1        d2       d3        d4
s1                10         9       10        11
s2                 9        10       11        10
s3                11         9       10        10;

Table t(j,k) sending delay
                 d1        d2       d3        d4
s1               12        14       10        11
s2               11         8        7        13
s3                6        11        4        15;

Table inv(i,j) inventory
                  s1        s2        s3
p1                60        80        50
p2                20        20        30;

Table dem(i,k) demand
                 d1       d2       d3        d4
p1               50       90       40        10
p2               10       20       10        30;
```

Multiobjective Optimization

- Example 3:

Variables: $x_{ijk} \in Re^+$

$$f_1 = \sum_{ijk} c_{jk} * x_{ijk} \qquad f_2 = \sum_{ijk} t_{jk} * x_{ijk}$$

Objective function:

$$\min\left(w_1 * f_1 + w_2 * f_2\right)$$

$$where\ w_1 + w_2 = 1$$

Constraints:

$$\sum_k x_{ijk} \leq inv_{ij} \quad \forall i, j$$

$$\sum_j x_{ijk} = dem_{ik} \quad \forall i, k$$

```
Variables
  x(i,j,k)      Amount of i type packets sent from the source node j
                to the destination node k.
  z             minimization
  f1            function 1
  f2            function 2;

Positive Variable x;

Equations
funObj                          Objective Function

invConstraint(i,j)              inventory constraint

demConstraint(i,k)              demand constraint

f1_value                        f1 value
f2_value                        f2 value;

f1_value              ..        f1=e= sum((i,j,k), c(j,k) * x(i,j,k));

f2_value              ..        f2=e= sum((i,j,k), t(j,k) * x(i,j,k));

funObj                ..        z =e= w1*f1 + w2*f2;

invConstraint(i,j)    ..        sum((k), x(i,j,k)) =l= inv(i,j);

demConstraint(i,k)    ..        sum((j), x(i,j,k)) =e= dem(i,k);


Model Model1 /all/ ;
```

Multiobjective Optimization

- Example 3:

***z_res*, *f1_res*, *f2_res* and *x_res*:**
Parameters to store the values of z, f1, f2 and x at each iteration. These parameters are arrays to save a specified value at each iteration.

Loop statement:
Instruction for solving the problem at each iteration, according to certain values of w_1 and w_2.

Display parameters:
Once all iterations have been performed, in the *.lst file are shown the results of these parameters.

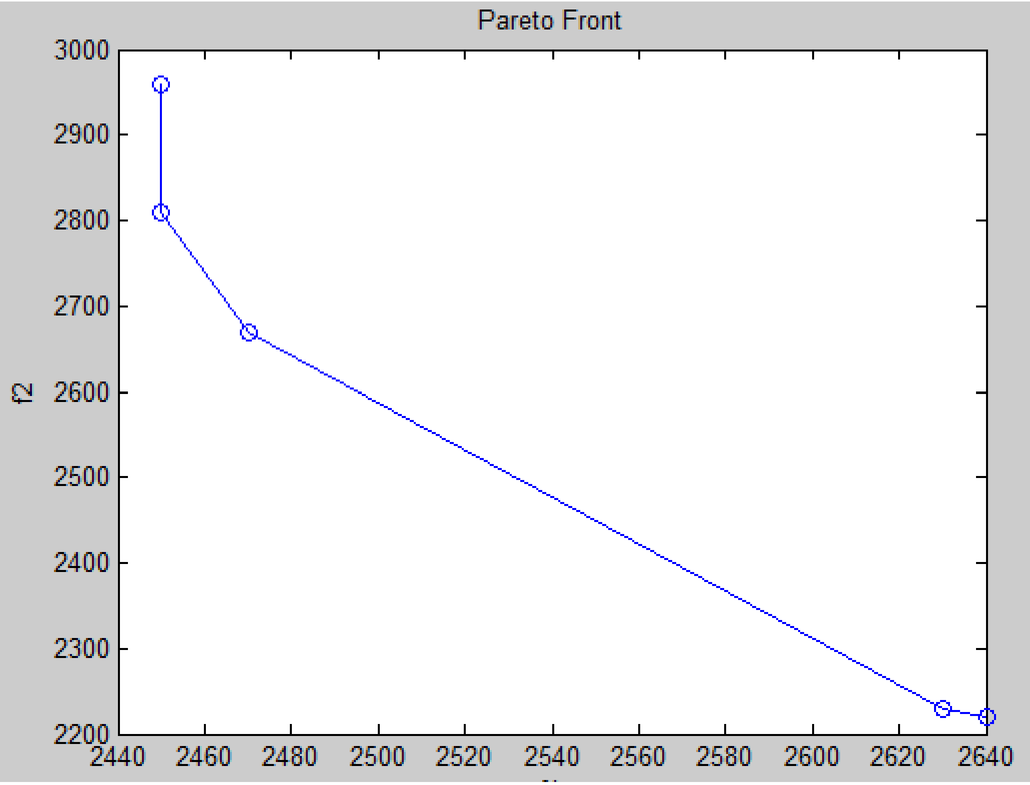
Using the put writing instruction:
In the “result.dat” file are stored the values of the set “iter” and the arrays “f1_res” and “f2_res”.

```
parameter z_res(iter) "z results to store";
parameter f1_res(iter) "f1 results to store";
parameter f2_res(iter) "f2 results to store";
parameter x_res(i,j,k,iter) "x results to store";

loop (iter,
    w1=w1_vec(iter);
    w2=1 - w1_vec(iter);
    w2_vec(iter)=w2;

    option lp=CPLEX;
    Solve Model1 using lp minimizing z;
    z_res(iter)=z.l;
    f1_res(iter)=f1.l;
    f2_res(iter)=f2.l;
    x_res(i,j,k,iter)=x.l(i,j,k);
    );

display z_res;
display f1_res;
display f2_res;
display w1_vec;
display w2_vec;
display x_res;

file GAMSresults /C:\DIRECTORY\results.dat/;
put GAMSresults;
loop(iter,
         put iter.tl, @5, f1_res(iter), @18, f2_res(iter) /

         );
```

Multiobjective Optimization

- Example 3:

```
clc, clear all, close all

[iter, f1, f2] = textread('results.dat', '%s %f %f', 20);

figure
plot(f1,f2,'-o')
title('Pareto Front');
xlabel('f1');
ylabel('f2');
```



References

- https://en.wikipedia.org/wiki/General_Algebraic_Modeling_System
- http://www.gams.com/
- http://www.gams.com/help/index.jsp
- http://www.gamsworld.org/performance/xpresslib/

Universidad de los Andes

Thanks!

Universidad de los Andes

Questions?