COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI ICADE CIHS

Good Optimization Modeling Practices with GAMS

All You Wanted to Know About Practical Optimization but Were Afraid to Ask

Andres Ramos

https://www.iit.comillas.edu/aramos/

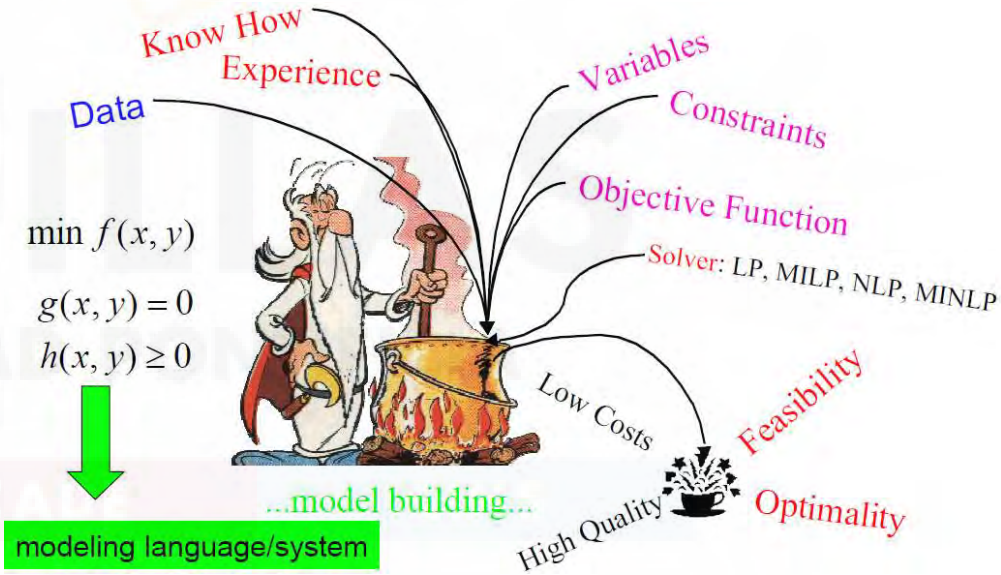
Andres.Ramos@comillas.edu

comillas.edu

“The disciples who received my instructions, and could themselves comprehend them, were seventy-seven individuals. They were all scholars of extraordinary ability.” Confucius

Do not confuse the ingredients of the recipe

- Mathematical formulation
  - LP, MIP, QCP, MCP
- Language
  - GAMS
- Solver
  - CPLEX, GUROBI, PATH
- Solver algorithm
  - Primal simplex, dual simplex, interior point
- Input/output interfaces
  - Text file, CSV, Excel, Matlab, Access
- Operating system
  - Windows, Linux, MacOS
- User-developed algorithm
  - Benders decomposition, Lagrangian relaxation, GA
- Stochastic extensions
  - EMP



Source: http://www.gams.com/presentations/present_modlang.pdf

Questions to deal with

- What you don't know how to do it in GAMS?
- What is the most advanced features you know in GAMS?
- What is the most important advantage/disadvantage of GAMS for you?
- What you would like to do and has not been able to?

Few and practical tips & tricks

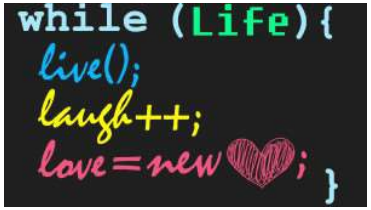
- It's not a systematic approach to teach basic/advanced GAMS features, just selected features I have used in several models

I have gambas I have chopitos
I have croquetas I have jamón
I have morcillas I have ensalá
I have una hueva mu bien aliña.

- It is optimization for shepherds (i.e., practitioners)

Contents

1. Programming Style



2. GAMS Code

```
* bounds on variables

vProduct.up (sc,n,g) $pScenProb(sc) = pMaxProd    (g   ) ;
vConsump.up (sc,n,g) $pScenProb(sc) = pMaxCons    (g   ) ;
vProduct1.up(sc,n,t) $pScenProb(sc) = pMaxProd    (t   ) - pMinProd(t) ;
vIG.up       (sc,n   ) $pScenProb(sc) = pInterGen (n,sc) ;
vENS.up      (sc,n   ) $pScenProb(sc) = pDemand    (n   ) ;
vReserve.up  (sc,n,g) $pScenProb(sc) = pMaxReserve(g   ) ;
vReserve.lo  (sc,n,g) $pScenProb(sc) = pMinReserve(g   ) ;

vCommitt.up (   n,g) = 1 ;
vStartup.up (   n,g) = 1 ;
vShutdown.up(   n,g) = 1 ;

* solve stochastic daily unit commitment model

solve SDUC using MIP minimizing vTotalVCost ;
```

3. Look and Feel

4. Mathematical Formulation

$$\sum_{i=t-TU_g^x+1}^{t} \sum_{y \in \mathcal{M}_g^{F,x}} v_{gi}^{yx} \leq u_{gt}^x \quad \forall g, x, t \in [TU_g^x, T] \qquad (4)$$

$$\sum_{i=t-TD_g^x+1}^{t} \sum_{x \in \mathcal{M}_g^{F,y}} v_{gi}^{xy} \leq 1 - u_{gt}^x \quad \forall g, x, t \in [TD_g^x, T] \,. \qquad (5)$$

5. Advanced Algorithms

1. Programming Style
2. GAMS Code
3. Look and Feel
4. Mathematical Formulation
5. Advanced Algorithms

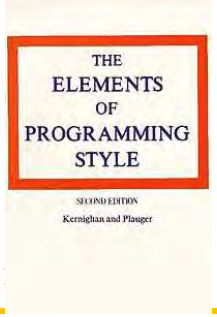
1

Programming Style

Programming

- Discipline whose control is basic in many engineering projects
  - Science: thinking, discipline, rigorousness and experimentation
  - Art: beauty and elegance

- A good design is fundamental
- Before writing any code the optimization problem must be written algebraically
- Learning by reading
- Coding by gradual refinement, incremental implementation
- Use a mockup for development and verification of the model
- Be careful with the details ("God is in the detail")



B.W. Kernighan and P.J. Plauger, ***The Elements of Programming Style***, McGraw Hill, New York, 1978 http://en.wikipedia.org/wiki/The_Elements_of_Programming_Style

Which is the most Spanish beautiful landscape?

Aigüestortes
National Park

Beach of
Menorca

Mediterranean
oak wood

Guadalquivir
marshland

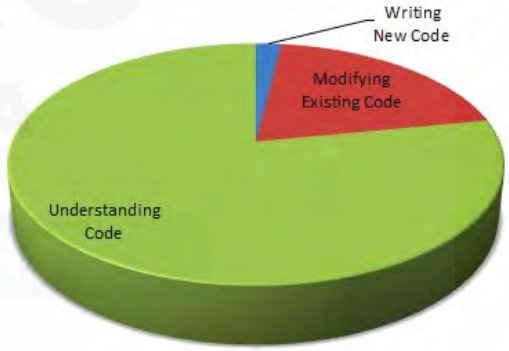
General recommendations

- Act according to the Pareto principle
  - 20 % takes to create the first prototype
  - 80 % of code development is devoted to maintenance and refinement
- MAINTAINABILITY and reusability are crucial
- Code is developed to be read by humans, not by machines. Write code for understanding the model, not for obscuring it.
- Say what you mean, simply and directly.
- Don't stop with your first draft. Refine it.

20%
Write new code

80%
Modify existing code: bugfixes, optimizations, etc.

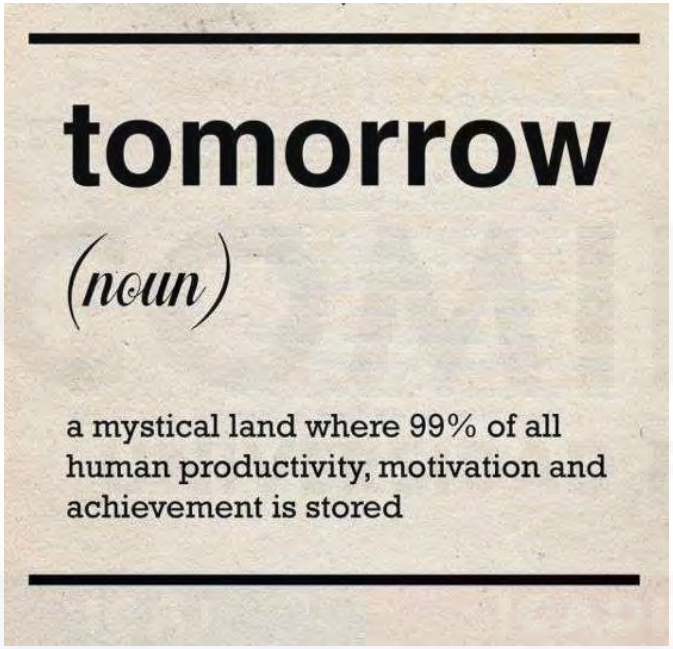
http://www.jetbrains.com/



Good Optimiza http://blog.codinghorror.com/when-understanding-means-rewriting/

Procrastination

- Don’t procrastinate when coding



Documentation. Comments

- It is a crucial task in code development
  - In particular, GAMS was born to explicitly include documentation into the code.
- Code must be self-documented
- Illustrative comments and well localized
- Make sure comments and code agree.
- Don’t just echo the code with comments - make every comment count.
- Don’t comment bad code or tricks - rewrite it.
- Don’t patch bad code - rewrite it.
- Don’t over-comment.

Code style

- Any project manager ought to define the style before starting up a multiple participant project (or maybe just for his/her own help)
- Systematic and consistent use of uppercase and lowercase letters
  - Use lowercase letters instead of uppercase. We are more used to read lowercase letters.
- Clean code, take care of the aesthetics when coding
  - Aesthetics is as important as the content. Code must be read at a glance.
- Format the code to help the reader understand it.
  - Indent to show the logical structure of a program.
  - Keep coherence in the coding rules (indent in repetitive sentences)
  - Align code to show patterns.
  - Make the reading easier (parallelism among consecutive similar sentences, indent)
- Use meaningful and long names for identifiers. Same use of identifiers in different parts of the code.

Efficiency vs. Clarity

- Make it clear and right before you make it faster
- Keep it simple to make it faster
- Don’t sacrifice clarity for small gains in efficiency

1. Programming Style
2. GAMS Code
3. Look and Feel
4. Mathematical Formulation
5. Advanced Algorithms

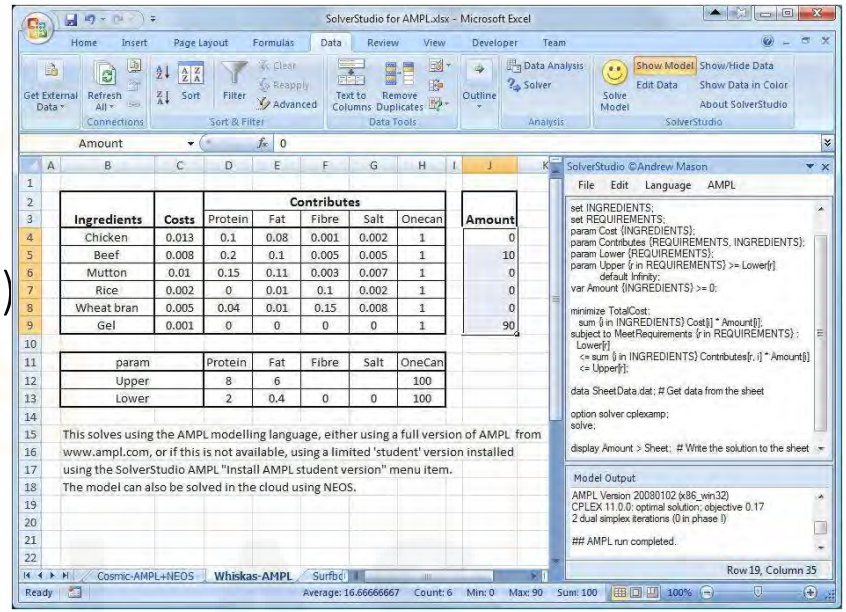
2

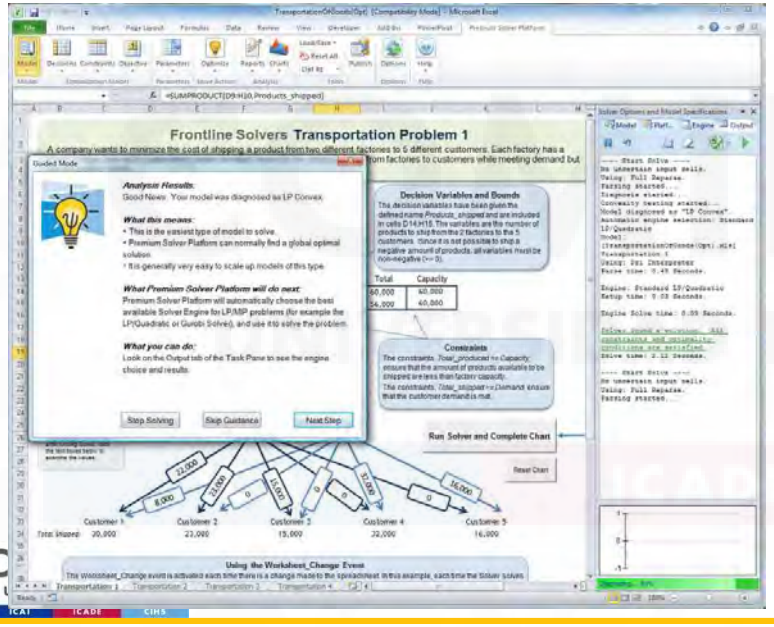
GAMS Code

Search, compare and if you find something better use it

Optimization under Microsoft Excel

SolverStudio (https://solverstudio.org/)

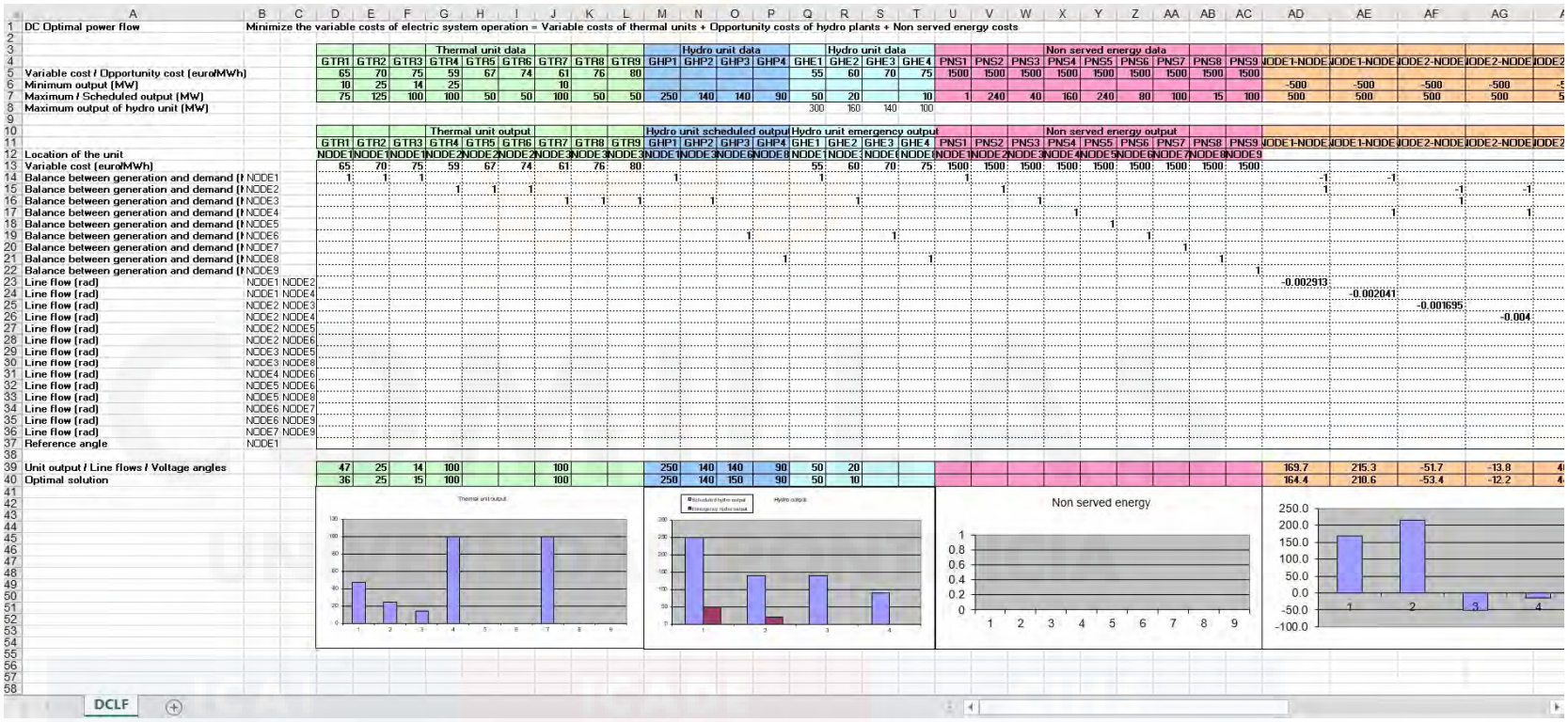




OpenSolver (https://opensolver.org/)

Solver from FrontlineSolvers (https://www.solver.com/premium-solver-platform)

DCLF in Microsoft Excel



Interfaces, languages, solvers

Interface (graphical)
Microsoft Excel
Microsoft Access
SQL
Matlab

Mathematical Language	Algebraic Language
	GAMS
	AMPL
	AIMMS
Python	Pyomo
Julia	JuMP
MatLab	
Solver Studio	

Solver
IBM CPLEX
Gurobi
XPRESS
GLPK
CBC
PATH

Learning by reading first, and then by doing

- GAMS Model Libraries

  (https://www.gams.com/modlibs/)

- Decision Support Models in the Electric Power Industry

  (https://www.iit.comillas.edu/aramos/Ramos_CV.htm#ModelosAyudaDecision)

StarGen Lite (**Short Term Stochastic Daily Unit Commitment Model**) demo GAMS version
StarGen Lite (**Medium Term Stochastic Hydrothermal Coordination Model**) demo GAMS version
StarGen Lite (**Long Term Generation Expansion Planning Model**) demo GAMS version
StarGen Lite (**Probabilistic Production Cost Model**) demo Excel version

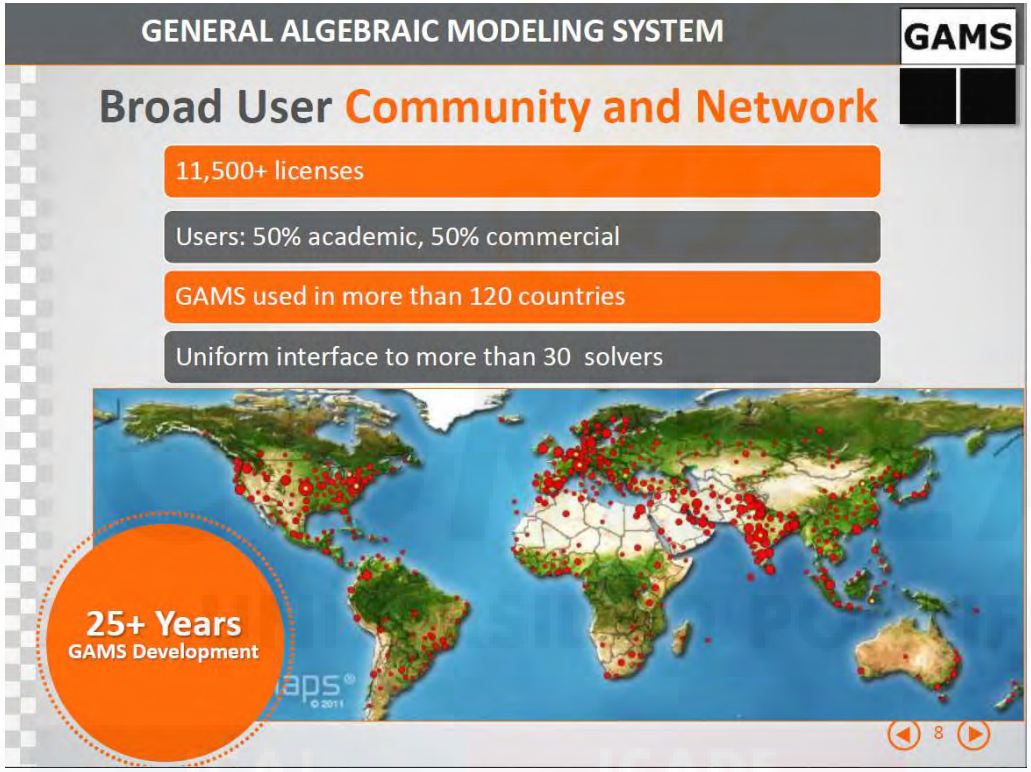
StarNet Lite (**Long Term Transmission Expansion Planning Model**) demo GAMS version

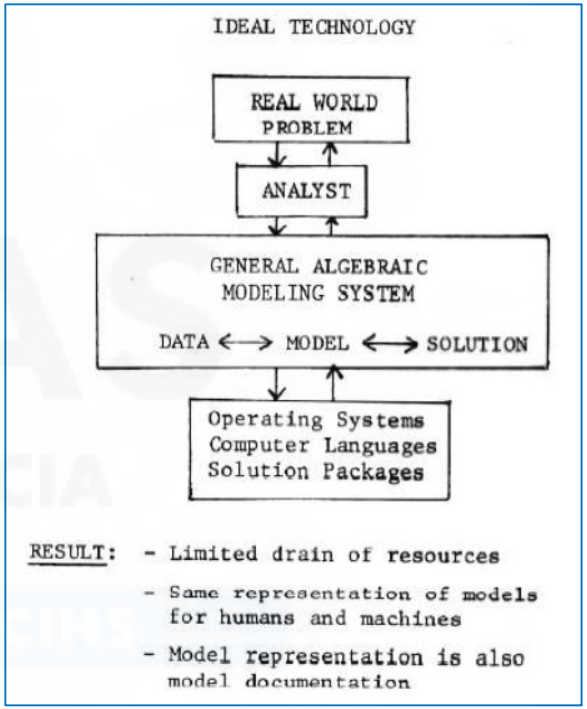
StarMkt Lite (**Cournot Market Equilibrium Model**) demo GAMS version
StarMkt Lite (**Bushnell Market Equilibrium Model**) demo GAMS version
StarMkt Lite (**Long Term Market Generation Expansion Planning Model**) demo GAMS version

StarGen Lite (**Short Term Stochastic Daily Unit Commitment Model**) demo Julia-JuMP version

StarGen Lite (**Short Term Stochastic Daily Unit Commitment Model**) demo Python-Pyomo version

GAMS (General Algebraic Modeling System)





GAMS birth: 1976 World Bank slide

comillas.edu

COMILLAS
UNIVERSIDAD PONTIFICIA
ICAI ICADE CIHS

Developing in GAMS

- Development environment GAMSIDE, GAMSStudio

aaa.gpr

- Documentation
  - GAMS Documentation Center https://www.gams.com/latest/docs/
  - GAMS Support Wiki https://support.gams.com/
  - Bruce McCarl's GAMS Newsletter https://www.gams.com/community/newsletters-mailing-list/
- Users guide Help > GAMS Users Guide
- Solvers guide Help > Expanded GAMS Guide

- Model: FileName.gms
- Results: FileName.lst
- Process log: FileName.log

My first minimalist optimization model

```
positive variables x1, x2
         variable  z

equations of, e1, e2, e3 ;

of .. 3*x1 + 5*x2 =e=  z ;
e1 ..   x1        =l=  4 ;
e2 .. 2*x2        =l= 12 ;
e3 .. 3*x1 + 2*x2 =l= 18 ;

model minimalist / all /
solve minimalist maximizing z using LP
```

$$\max_{x_1,x_2} z = 3x_1 + 5x_2$$

$$\begin{aligned} x_1 &\leq 4 \\ 2x_2 &\leq 12 \\ 3x_1 + 2x_2 &\leq 18 \\ x_1, x_2 &\geq 0 \end{aligned}$$

Blocks in a GAMS model

- Mandatory
  - `variables`
  - `equations`
  - `model`
  - `solve`

- Optional
  - `sets: (alias)`
    - `alias (i,j)` *i* and *j* can be used indistinctly
    - Checking of domain indexes
  - data: `scalars, parameters, table`

Transportation model

There are i can factories and j consumption markets. Each factory has a maximum capacity of a_i cases and each market demands a quantity of b_j cases (it is assumed that the total production capacity is greater than the total market demand for the problem to be feasible). The transportation cost between each factory i and each market j for each case is c_{ij}. The demand must be satisfied at minimum cost.
The decision variables of the problem will be cases transported between each factory i and each market j, x_{ij}.

My first transportation model (classical organization)

```
sets
   I origins      / VIGO, ALGECIRAS /
   J destinations / MADRID, BARCELONA, VALENCIA /

parameters
   pA(i) origin capacity
       / VIGO      350
         ALGECIRAS 700 /

   pB(j) destination demand
       / MADRID    400
         BARCELONA 450
         VALENCIA  150 /

table pC(i,j) per unit transportation cost
           MADRID BARCELONA VALENCIA
VIGO         0.06      0.12     0.09
ALGECIRAS    0.05      0.15     0.11

variables
   vX(i,j) units transported
   vCost   transportation cost

positive variable vX

equations
   eCost        transportation cost
   eCapacity(i) maximum capacity of each origin
   eDemand  (j) demand supply at destination ;

eCost        .. sum[(i,j), pC(i,j) * vX(i,j)] =e= vCost ;
eCapacity(i) .. sum[   j ,           vX(i,j)] =l= pA(i) ;
eDemand  (j) .. sum[ i   ,           vX(i,j)] =g= pB(j) ;

model mTransport / all /
solve mTransport using LP minimizing vCost
```

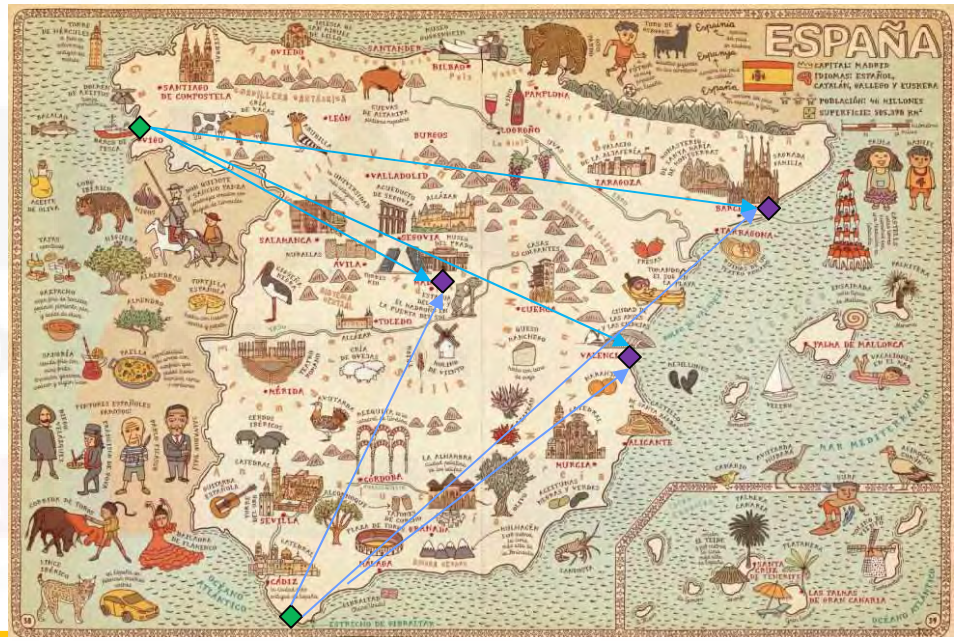
$$\min_{x_{ij}} \sum_{ij} c_{ij} x_{ij}$$

$$\sum_{j} x_{ij} \leq a_i \quad \forall i$$

$$\sum_{i} x_{ij} \geq b_j \quad \forall j$$

$$x_{ij} \geq 0$$

A. Mizielinska y D. Mizielinski *Atlas del mundo: Un insólito viaje por las mil curiosidades y maravillas del mundo* Ed. Maeva 2015



General structure of GAMS sentences

- Commenting
  - Lines with * in the first column
  - `$OnText $OffText` to comment many lines
- No distinction between uppercase and lowercase letters
- Parenthesis (), square bracket [] or braces {} can be used indistinctly to distinguish levels
- Language reserved words appear in bold
- Sentences end with a ;
  - Can be suppressed when the following word is a reserved one

Basic input/output in text format

- Data input from a text file

```
$include FileName.txt
display IdentifierName (shows its content or value)
```

- Data output to a text file

```
file InternalName / ExternalName.txt /
put      InternalName
   put   IdentifierName
putclose InternalName
```

- Specific options to control the output format
  - Put Writing Facility

Functions and operators

- +, -, *, /, ** or power(x,n)
- abs, arctan, sin, cos, ceil, floor, exp, log, log10, max, min, mod, round, sign, sqr, sqrt, trunc, normal, uniform
- gyear, gmonth, gday, ghour, gminute, gsecond, gdow, gleap, jdate, jnow, jstart, jtime
- lt <, gt >, eq =, ne <>, le <=, ge >=
- not, and, or, xor
- diag(set_element,set_element)={1,0}
- sameas(set_element,set_element)={T,F}
- ord, card ordinal and cardinal of a set, SetName.pos ordinal of a set
- sum, prod, smax, smin
- inf , eps, pi are valid as data

Model temporal license

```
abort $[jstart > jdate(2019,5,15)] 'License for this model has expired and it cannot be used any more. Contact the developers'
```

$ Operator in assignments, summations, constraints

- Sets a condition

```
$(value > 0)        $(number1 <> number2)
```

- On the left of an assignment, it does the assignment ONLY if the condition is satisfied

```
if (condition,
    DO THE ASSIGNMENT
) ;
```

- On the right of an assignment, it does the assignment ALWAYS and if the condition is not satisfied it assigns value 0

```
if (condition,
    DO THE ASSIGNMENT
else
    ASSIGNS VALUE 0
) ;
```

Dynamic sets

- Efficiency is strongly related to the use of dynamic sets
- Subsets of static sets whose content may change by assignments

```
sets m          months / m01 * m12 /
     mp(m) even months
display m ;
mp(m) $[mod(ord(m),2) = 0] = yes ;
display mp ;
mp('m03') = yes ;
display mp ;
mp(m) $[ord(m) = 4] = no ;
display mp ;
```

- Fundamental elements in developing GAMS models
- Must be used systematically to avoid the formulation of superfluous equations, variables or assignments

According to legend **Roland**'s Breach was cut by Count Roland with his sword **Durendal** in an attempt to destroy that sword, after being defeated during the Battle of **Roncesvalles** in 778.

Use and abuse of dynamic sets

```
sets
   w           weeks                    / w01  * w52  /
   h           hours                    / h001 * h168 /
   nd          nodes                    / node01 * node99 /
   ln(nd,nd)   lines
   spring(  w) spring weeks             / w13  * w25  /
   days5 (  h) first 5 days of the week / h001 * h120 /
   sprday(w,h) spring working days

alias (nd,ni,nf)

parameters
   pDemand(w,h,nd   ) demand in each node
   pFlow  (w,h,nd,nd) flow   in each line ;

sprday(w,h) $[spring(w)*days5(h)] = yes ;

* these sentences are equivalent

pDemand(w,h,nd      ) $[spring(w)*days5(h)] = uniform(-0.5,0.5) ;
pFlow  (w,h,ni,nf   ) $[spring(w)*days5(h)] = uniform(-1.0,1.0) ;

pDemand(w,h,nd      ) $sprday(w,h)          = uniform(-0.5,0.5) ;
pFlow  (w,h,ni,nf   ) $sprday(w,h)          = uniform(-1.0,1.0) ;

pDemand(sprday,nd   )                       = uniform(-0.5,0.5) ;
pFlow  (sprday,ni,nf)                       = uniform(-1.0,1.0) ;
```

Index shifting. Lag and lead

- `t=J,F,M,A,MA,J,JU,AU,S,O,N,D`

  `vReserve(t) + pInflow(t) - vOutflow(t) =e= vReserve(t+1)`

- Vector values out of the domain are 0

  `vReserve('D') + pInflow('D') - vOutflow('D') =e= 0`

- Circular sequence of an index (++, --)

  `t=J,F,M,A,MA,J,JU,AU,S,O,N,D`

  `vReserve(t) + pInflow(t) - vOutflow(t) =e= vReserve(t++1)`

  `vReserve('D') + pInflow('D') - vOutflow('D') =e= vReserve('J')`

- Inverted order sequence of `PP` index even though `t` is traversed in increasing order

  `PP(t+[card(t)-2*ord(t)+1])`

Operations with sets

- Intersection

```
D(a) = B(a) * C(a)
```

- Union

```
D(a) = B(a) + C(a)
```

- Complementary

```
D(a) = NOT C(a)
```

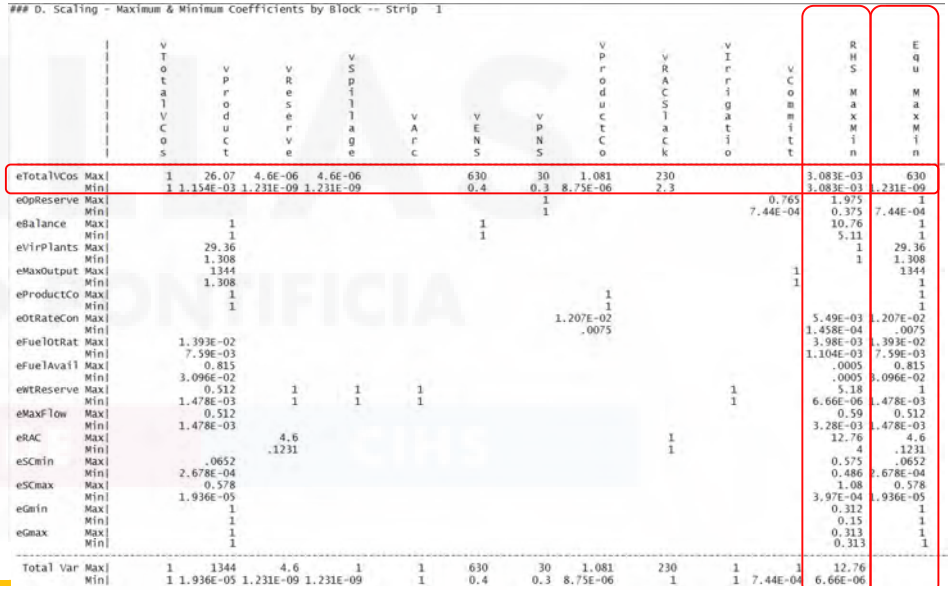
- Difference

```
D(a) = B(a) - C(a)
```

Analytical computation of constraints and variables

- Important to know the estimated size of the optimization problem and dependence with respect to basic elements
- It can be used for detecting formulation errors
- Use LimRow/LimCol
- Suitable to know the constraint matrix structure (GAMSChk)

```
option LP=GAMSChk
```



D. Scaling - Maximum & Minimum Coefficients by Block -- Strip 1

		vTotalVCos	vProduct	vReserve	vSpillage	vArc	vENS	vPNS	vProductCo	vRACSlack	vIrrigatio	vCommit	RHS Max	RHS Min	Equ Max	Equ Min
eTotalVCos	Max	1	26.07	4.6E-06	4.6E-06		630	30	1.081	230			3.083E-03		630	
	Min	1	1.154E-03	1.231E-09	1.231E-09		0.4	0.3	8.75E-06	2.3			3.083E-03			1.231E-09
eOpReserve	Max							1				0.765	1.975		1	
	Min							1				7.44E-04	0.375		7.44E-04	
eBalance	Max		1				1						10.76		1	
	Min		1				1						5.11		1	
eVirPlants	Max		29.36										1		29.36	
	Min		1.308										1		1.308	
eMaxOutput	Max		1344									1			1344	
	Min		1.308									1			1	
eProductCo	Max		1						1						1	
	Min		1						1						1	
eOtRateCon	Max								1.207E-02				5.49E-03		1.207E-02	
	Min								.0075				1.458E-04		.0075	
eFuelOtRat	Max		1.393E-02										3.98E-03		1.393E-02	
	Min		7.59E-03										1.104E-03		7.59E-03	
eFuelAvail	Max		0.815										.0005		0.815	
	Min		3.096E-02										.0005		3.096E-02	
ewtReserve	Max		0.512	1	1	1					1		5.18		1	
	Min		1.478E-03	1	1	1					1		6.66E-06		1.478E-03	
eMaxFlow	Max		0.512										0.59		0.512	
	Min		1.478E-03										3.28E-03		1.478E-03	
eRAC	Max			4.6						1			12.76		4.6	
	Min			.1231						1			4		.1231	
eSCmin	Max		.0652										0.575		.0652	
	Min		2.678E-04										0.486		2.678E-04	
eSCmax	Max		0.578										1.08		0.578	
	Min		1.936E-05										3.97E-04		1.936E-05	
eGmin	Max		1										0.312		1	
	Min		1										0.15		1	
eGmax	Max		1										0.313		1	
	Min		1										0.313		1	
Total Var	Max	1	1344	4.6	1	1	630	30	1.081	230	1	1	12.76			
	Min	1	1.936E-05	1.231E-09	1.231E-09	1	0.4	0.3	8.75E-06	1	1	7.44E-04	6.66E-06			

Source: MPES

Number of equations and variables for MHE

		Ecuaciones
Función objetivo	1	1
Balance demanda	$PP'S$	2000
Producción térmica subperiodo	$P(P'-1)ST$	240000
Producción hidráulica subperiodo	$P(P'-1)SH$	100000
Bombeo hidráulica subperiodo	$P(P'-1)SH'$	20000
Horas por escenario	PST	120000
Horas mínimas y máximas por escenario	$2ST$	12000
Horas mínimas y máximas todos escenarios	$2T$	240
Balance embalses	PSE	50000
Gasto del embalse por central	$PSEH$	250000
Gasto total del embalse	PSE	50000
Bombeo desde el embalse	PSE'	10000
Desvío gasto embalse	PSE	50000
Reserva final embalse	SE	2500
Reserva mínima y máxima embalse	$2PSE$	100000
TOTAL		1006741

			Variables
Producción térmica	$pt^s_{pp't}$	$PP'ST$	240000
Producción hidráulica	$ph^s_{pp'h}$	$PP'SH$	100000
Producción del bombeo	$pb^s_{pp'h}$	$PP'SH'$	20000
Vertido del embalse	v^s_{pe}	PSE	50000
Reserva artificial del embalse	ra^s_{pe}	PSE	50000
Reserva del embalse	r^s_{pe}	PSE	50000
Defecto y exceso de entre reservas consecutivas	dr^s_{pe}, er^s_{pe}	$2PSE$	100000
Defecto y exceso de reserva	dr^s_e, er^s_e	$2SE$	5000
Defecto de reserva mínima y exceso de reserva máxima	drm^s_{pe}, erm^s_{pe}	$2PSE$	100000
Horas de funcionamiento por periodo	hr^s_{pt}	PST	120000
Defecto y exceso de horas de funcionamiento por periodo	dhr^s_{pt}, ehr^s_{pt}	$2PST$	240000
Defecto y exceso de horas de funcionamiento	dhn^s_t, ehx^s_t	$2ST$	12000
Gasto del embalse por central	g^s_{peh}	$PSEH$	250000
Gasto del embalse	g^s_{pe}	PSE	50000
Consumo del bombeo	b^s_{pe}	PSE'	10000
TOTAL			1397000

How big is a big optimization problem

- Memory requirements for creating the model (GAMS)
  - 1 GB for every 1 million rows
- Memory requirements for solving the model (solver)
  - Depends on the difficulty in solving the model
    - Integrality gap for MIP problems

Avoid creation of superfluous constraints and variables

- Or how to achieve a compact formulation (small size of the constraint matrix or small density)
- Some redundant constraints can introduce a tighter model, see later
- However, introduce logical conditions (with a $ in GAMS) in the creation of equations or the use of variables to avoid superfluous ones
- Reduction rules: mathematical reasoning or common sense based on the problem context
  - Flows by non existing connections in a network
- Solvers can detect some of these superfluous equations/variables but it is more efficient to avoid their creation (pre-processing)
- Profile, ProfileTol

Debugging an optimization model

- Grammar error
  - Read the error and click in the red line of the log file

- Infeasibility detection
  - Soft (elastic) constraints
    - Introduce a deficit or surplus variable in each equation and penalize them in the objective function. Be careful with penalty parameter
  - Detect the smallest core of infeasible constraints by the LP solver (option *Irreducible Infeasible Subsets iis* in solvers)
    - Once known, they have to be deleted or modified
    - FeasOpt in GUROBI

LP Performance issues and their suggested resolution

LP performance issue	Suggested resolution
Numerical instability	• Calculate and input model data in double precision • Eliminate nearly-redundant rows and/or columns of *A a priori* • Avoid mixtures of large and small numbers: (i) Be suspicious of κ between 10^{10} and 10^{14}; (ii) Avoid data leading to κ greater than 10^{14} • Use alternate scaling (in the model formulation or optimizer settings) • Increase the Markowitz threshold • Employ the numerical emphasis parameter (if available)
Lack of objective function improvement under degeneracy	• Try all other algorithms (and variants) • Perturb data either *a priori* or using algorithmic settings
Primal degeneracy	• Use either dual simplex or interior point on primal problem
Dual degeneracy	• Employ either primal simplex or interior point on primal problem
Both primal and dual degeneracy	• Execute interior point on primal or dual problem
Excessive time per iteration	• Try all other algorithms (and variants) • Use algorithmic settings to conserve memory or purchase more externally • Try less expensive pricing settings if using simplex algorithms
Excessive simplex algorithm iterations	• Try Steepest edge or Devex variable selection
Multiple bound shift removals or significant infeasibilities after removing shifts	• Reduce feasibility and optimality tolerances
Barrier algorithm iterations with little or no progress	• Increase barrier convergence tolerance in order to initiate crossover earlier
Too much time in crossover	• Reduce barrier convergence tolerance in order to provide a better starting point for crossover

E. Klotz, A.M. Newman *Practical guidelines for solving difficult linear programs* Surveys in Operations Research and Management Science 18 (1-2), 1-17, Oct 2013 10.1016/j.sorms.2012.12.001

Efficiency in GAMS code usage (loop)

```
set i / 1*2000 /
alias (i,ii)
parameter X(i,i)

loop ((i,ii),
   X(i,ii) = 4 ;
) ;
```

75.3 s

```
set i / 1*2000 /
alias (i,ii)
parameter X(i,i) ;


X(i,ii) = 4 ;
```

0.3 s

Observer effect

- Changes that the act of observation will make on a phenomenon being observed


```
option Profile=10, ProfileTol=0.01

set i / 1*2000 /
alias (i,ii)
parameter X(i,i)

loop ((i,ii),
   X(i,ii) = 4 ;
) ;
```

74.2 s

```
set i / 1*2000 /
alias (i,ii)
parameter X(i,i)

loop ((i,ii),
   X(i,ii) = 4 ;
) ;
```

72.9 s

Efficiency in GAMS code usage (index order)

```
option Profile=10, ProfileTol=0.01

set i / 1*200 /
    j / 1*200 /
    k / 1*200 /
parameter X(k,j,i), Y(i,j,k) ;

Y(i,j,k) = 2 ;

X(k,j,i) = Y(i,j,k)
```

4.5 s

```
option Profile=10, ProfileTol=0.01

set i / 1*200 /
    j / 1*200 /
    k / 1*200 /
parameter X(i,j,k), Y(i,j,k) ;

Y(i,j,k) = 2 ;

X(i,j,k) = Y(i,j,k)
```

1.3 s

These constructs also exist in GAMS

```
loop (set,
) ;
```

```
while (condition,
) ;
```

```
repeat
until condition;
```

```
if (condition,
else
) ;
```

```
for (i=beginning to/downto end by increment,
) ;
```

```
Break
Continue
```

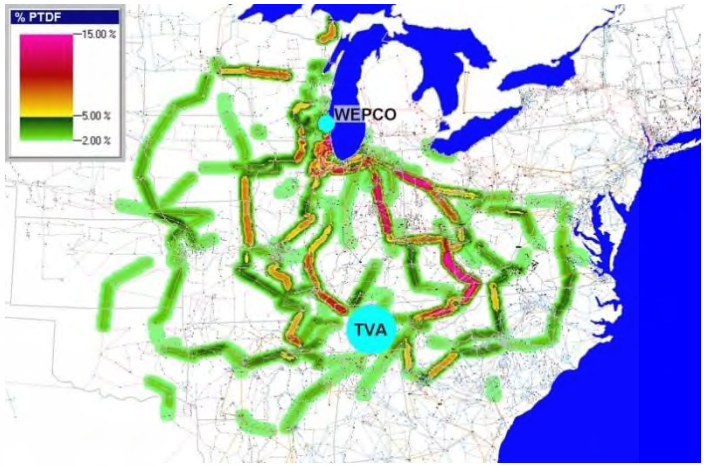
Jump out of the cycle

Detection of isolated subnetworks

```
$Phantom null

sets
   nd                nodes                       / node01 * node19 /
   ndref (nd)        current   reference node    / node01 /
   refnd (nd)        subset of reference nodes   / node01 /
   nc    (nd)        current   connected nodes    / null   /
   nod   (nd)        subset of connected nodes   / null   /
   ln    (nd,nd)     lines
   subnet(nd,nd,nd) subnetworks

parameters
   pAux1 auxiliary / 0 /
   pAux2 auxiliary / 1 /

alias (nd,n1,n2,ni,nf)

file out / out.gms / put out ;

* create a naïve network, a chain
ln(ni,nf) $[ni.pos = nf.pos-1] = yes ;

* break these links
ln('node10','node11') = no ;
ln('node15','node16') = no ;

* detection of isolated subnetworks

* for every subnetwork => max number of iterations
loop (n1 $[sum(nod(nd), 1) < card(nd)],

*  define the reference node for 2nd+ iterations
   ndref(nd) $[n1.pos > 1 and not refnd(nd) and nd.pos = smin(n2 $[not nod(n2)], n2.pos)] = yes ;

*  empty the set of connected nodes
   nc(nd   ) =  no ;
*  connect the reference node
   nc(ndref) = yes ;

   pAux2 = 1 ;

*  for every node => max number of iterations
   loop (n2 $pAux2,
*     count the number of connected nodes
      pAux1  = sum[nc,                          1]       ;
*     add nodes to the set of already connected nodes
      nc(nf) $ sum[nc $[ln(nc,nf) or ln(nf,nc)], 1] = yes ;
*     count the new added nodes
      pAux2  $[sum[nc,                          1] - pAux1 = 0] = 0 ;

      if (pAux2 = 0,
*        subnetwork of the connected lines to a reference node
         subnet(ln(nc,nf),ndref) = yes ;
         subnet(ln(nf,nc),ndref) = yes ;
*        subnetwork of the connected nodes
         nod   (   nc          ) = yes ;
*        subnetwork of the reference nodes
         refnd (          ndref) = yes ;
      ) ;

   ) ;
   display subnet, nod, refnd ;

*  disconnect the reference node
   ndref(nd) = no ;

) ;
```

Inverting a matrix, e.g., PTDF

```
set i / i1*i3 /

table a(i,i) matrix to invert
   i1 i2 i3
i1  1
i2     3
i3        5

parameter ainv(i,i) inverted matrix

execute_unload '       GDXForInverse.gdx' i a
execute        'invert GDXForInverse.gdx  i a GDXFromInverse.gdx  ainv'
execute_load   '                          GDXFromInverse.gdx' ainv
execute        'del    GDXForInverse.gdx      GDXFromInverse.gdx'

display a, ainv
```




```
* computation of the susceptance matrix of the corridors
pYBUS(c2) = - sum[la(c2,cc), 1/pLineX(la)] ;
pYBUS(nf,ni) $c2(ni,nf) = pYBUS(ni,nf)  ;
pYBUS(nf,nf) =  - sum[ni, pYBUS(ni,nf)] ;
pYBUS(ni,ndref(nd)) = 0 ;
pYBUS(ndref(nd),nf) = 0 ;
pYBUS(ni,nd) $[not nc(ni)] = 0 ;
pYBUS(nd,nf) $[not nc(nf)] = 0 ;

* obtaining the inverse of pYBUS and saving it into pYBUSInv
execute_unload '       GDXForInverse.gdx' noref pYBUS
execute        'invert GDXForInverse.gdx  noref pYBUS GDXFromInverse.gdx  pYBUSInv'
execute_load   '                                      GDXFromInverse.gdx' pYBUSInv
execute        'del    GDXForInverse.gdx              GDXFromInverse.gdx' ;

* computation of the PTDF matrix
pPTDF(la(ni,nf,cc),ngd) = [pYBUSInv(ni,ngd) - pYBUSInv(nf,ngd)]/pLineX(la) + eps ;
```

Deployment ready

- Split formulation from data.
- Protect formulation confidentiality
- Secure Work Files
  - Control the access to symbol names
  - Link the model to a specific license

- Set declaration. Initialization.
- Variables
- Equations
- Model

- Include and manipulate input data and parameters.

- Bounds and initialization of variables
- Solve the optimization problem
- Output of the results

Prepared to be deployed

Transportation model (ready for deployment)

Formulation.gms

```
sets
   I origins
   J destinations

parameters
   pA(i  ) origin capacity
   pB(j  ) destination demand
   pC(i,j) per unit transportation cost

variables
   vX(i,j) units transported
   vCost   transportation cost

positive variable vX

equations
   eCost        transportation cost
   eCapacity(i) maximum capacity of each origin
   eDemand  (j) demand supply at destination ;

eCost ..        sum[(i,j), pC(i,j) * vX(i,j)] =e= vCost ;
eCapacity(i) .. sum[j,              vX(i,j)] =l= pA(i) ;
eDemand  (j) .. sum[i,              vX(i,j)] =g= pB(j) ;

model mTransport / all /
```

Remaining.gms

```
$include Data.gms
solve mTransport using LP minimizing vCost
```

Data.gms

```
sets
   I origins      / VIGO, ALGECIRAS /
   J destinations / MADRID, BARCELONA, VALENCIA /

parameters
   pA(i) origin capacity
       / VIGO      350
         ALGECIRAS 700 /

   pB(j) destination demand
       / MADRID    400
         BARCELONA 450
         VALENCIA  150 /

table pC(i,j) per unit transportation cost
           MADRID BARCELONA VALENCIA
VIGO         0.06      0.12     0.09
ALGECIRAS    0.05      0.15     0.11
```

Generate the runtime model

gams Formulation.gms **Save**=model

→ Send model.g00 →

Execute runtime model + data

gams Remaining.gms **Restart**=model

Model log

- Open console from GAMSIde for logging messages from the model
  - Code specific for Windows, UNIX/Linux/macOS

```
$set console
$if '%system.filesys%' == 'MSNT' $set console con
$if '%system.filesys%' == 'UNIX' $set console /dev/tty
$if '%console%.' == '.' abort 'console not recognized'

file console / '%console%' /

sets
   day day      / day01*day10 /
   sc  scenario / sc01* sc02 /

put console
loop ((day,sc),
   putclose 'Day ' day.tl:0 ' Scenario ' sc.tl:0 ' Elapsed Time ' [(jnow-jstart)*86400]:6:3 ' s' sleep(1)
) ;
```

```
$ifthen.MSNT '%system.filesys%' == 'MSNT'
   execute 'del pp.txt' ;
$endif.MSNT
$ifthen.UNIX '%system.filesys%' == 'UNIX'
   execute 'rm  pp.txt' ;
$endif.UNIX
```

GAMS Code Conventions

- Must be defined in blocks. For example, a set and all its subsets should constitute one block in the sets section.
- Names are intended to be meaningful. Follow conventions
  - Items with the same name represent the same concept in different models
  - Units should be used in all definitions
  - Parameters are named pParameterName (e.g., pTotalDemand)
  - Variables are named vVariableName (e.g., vThermalOutput)
  - Equations are named eEquationName (e.g., eLoadBalance)
  - Use short set names (one or two letters) for easier reading
  - Alias duplicate the final letter (e.g., p, pp)
- In the case of variables, the blocks should be defined by meaning and not by variable type (Free (default), Positive, Negative, Binary, Integer, SOS1, SOS2, SemiCont, SemiInt)
- Objective function must be a free variable
- Equations are laid out as clearly as possible, using brackets for readability

Stochastic Daily Unit Commitment of Thermal and Hydro Units (SDUC) (i) (https://iit.comillas.edu/aramos/StarGenLite_SDUC.zip)

```
$Title StarGen Lite Stochastic Daily Unit Commitment of Thermal and Hydro Units (SDUC)

$OnText

Developed by

   Andrés Ramos
   Instituto de Investigacion Tecnólogica
   Escuela Técnica Superior de Ingenieria - ICAI
   UNIVERSIDAD PONTIFICIA COMILLAS
   Alberto Aguilera 23
   28015 Madrid, Spain
   Andres.Ramos@comillas.edu
   https://www.iit.comillas.edu/aramos/Ramos_CV.htm

   January 22, 2019

$OffText

$OnEmpty OnMulti OffListing

* options to skip or not the Excel input/output
* if you want to skip it put these values to 1
* in such a case input files have to be already in the directory created by any other means
* output file will be the tmp.gdx that can be exported to Excel manually
$ifthen.OptSkipExcelInput       %gams.user2% == ""
$  setglobal OptSkipExcelInput  0
$else.OptSkipExcelInput
$  setglobal OptSkipExcelInput  %gams.user2%
$endif.OptSkipExcelInput

$ifthen.OptSkipExcelOutput      %gams.user3% == ""
$  setglobal OptSkipExcelOutput 0
$else.OptSkipExcelOutput
$  setglobal OptSkipExcelOutput %gams.user3%
$endif.OptSkipExcelOutput

* solve the optimization problems until optimality
option OptcR = 0
```

Model name

Authorship and version

Allow declaration of empty sets and multiple declaration. Suppress listing

Obtain the optimal solution

Stochastic Daily Unit Commitment of Thermal and Hydro Units (SDUC) (ii)

```
* definitions

sets
   n       hour
   n1(n)   first hour of the day
   sc      scenario

   g       generating unit
   t (g)   thermal    unit
   h (g)   hydro      plant

alias (n,nn)

parameters
   pDemand         (n) hourly load                    [GW]
   pOperReserve    (n) hourly operating reserve       [GW]
   pOperReserveUp  (n) hourly operating reserve up    [GW]
   pOperReserveDw  (n) hourly operating reserve down  [GW]
   pIntermGen   (n,sc) stochastic IG generation       [GW]
   pScenProb      (sc) probability of scenarios       [p.u.]
   pCommitt  (   g,n) commitment of the unit          [0-1]
   pProduct  (sc,g,n) output     of the unit          [GW]
   pIG       (sc,  n) output     of IG generation     [GW]
   pSRMC     (sc,  n) short run marginal cost         [ € per MWh]

   pMaxProd        (g) maximum output                 [GW]
   pMinProd        (g) minimum output                 [GW]
   pMaxCons        (g) maximum consumption            [GW]
   pIniOut         (g) initial output > min load      [GW]
   pIniUC          (g) initial commitment             [0-1]
   pRampUp         (g) ramp up                        [GW per   h]
   pRampDw         (g) ramp down                      [GW per   h]
   pMinTU          (g) minimum time down              [h]
   pMinTD          (g) minimum time up                [h]
   pSlopeVarCost   (g) slope     variable cost        [M€ per GWh]
   pInterVarCost   (g) intercept variable cost        [M€ per   h]
   pEmissionCost   (g) emission           cost        [M€ per GWh]
   pStartupCost    (g) startup            cost        [M€]
   pShutdownCost   (g) shutdown           cost        [M€]
   pMaxReserve     (g) maximum reserve                [GWh]
   pMinReserve     (g) minimum reserve                [GWh]
   pIniReserve     (g) initial reserve                [GWh]
   pEffic          (g) pumping efficiency             [p.u.]
   pInflows      (g,n) inflows                        [GWh]
   pENSCost            energy not served  cost        [M€ per GWh]
   pCO2Cost            CO2 emission       cost        [ € per tCO2]
```

Set definition

Parameter definition

Stochastic Daily Unit Commitment of Thermal and Hydro Units (SDUC) (iii)

```
variables
   vTotalVCost          total system variable cost      [M€]

binary   variables
   vCommitt  (   n,g) commitment of the unit           [0-1]
   vStartup  (   n,g) startup    of the unit           [0-1]
   vShutdown (   n,g) shutdown   of the unit           [0-1]

positive variables
   vProduct  (sc,n,g) output of the unit               [GW]
   vProduct1 (sc,n,g) output of the unit > min load    [GW]
   vConsump  (sc,n,g) consumption of the unit          [GW]
   vENS      (sc,n  ) energy not served                [GW]
   vIG       (sc,n  ) intermittent generation          [GW]
   vWtReserve(sc,n,g) water reserve at end of period [GWh]
   vSpillage (sc,n,g) spillage                       [GWh]

equations
   eTotalVCost          total system variable cost      [M€]
   eBalance  (sc,n  ) load generation balance          [GW]
   eOpReserve(   n  ) operating reserve                [GW]
   eReserveUp(sc,n  ) operating reserve upwards        [GW]
   eReserveDw(sc,n  ) operating reserve downwards      [GW]
   eMaxOutput(sc,n,g) max output of a committed unit   [GW]
   eMinOutput(sc,n,g) min output of a committed unit   [GW]
   eTotOutput(sc,n,g) tot output of a committed unit   [GW]
   eRampUp   (sc,n,g) bound on ramp up                 [GW]
   eRampDw   (sc,n,g) bound on ramp down               [GW]
   eUCStrShut(   n,g) relation among commitment startup and shutdown
   eMinTUp   (   n,g) minimum up   time (    committed)
   eMinTDw   (   n,g) minimum down time (not committed)
   eWtReserve(sc,n,g) water reserve                  [GWh] ;
```

Variables

Equation definition

Stochastic Daily Unit Commitment of Thermal and Hydro Units (SDUC) (iv)

```
* mathematical formulation

eTotalVCost        .. vTotalVCost =e= sum[(sc,n  ), pENSCost        *vENS     (sc,n  )*pScenProb(sc)] +
                                      sum[(sc,n,t), pSlopeVarCost(t)*vProduct (sc,n,t)*pScenProb(sc)] +
                                      sum[(sc,n,t), pEmissionCost(t)*vProduct (sc,n,t)*pScenProb(sc)] +
                                      sum[(   n,t), pInterVarCost(t)*vCommitt (   n,t)] +
                                      sum[(   n,t), pStartupCost (t)*vStartup (   n,t)] +
                                      sum[(   n,t), pShutdownCost(t)*vShutdown(   n,t)] ;

eBalance  (sc,n  ) $ pScenProb(sc) .. sum[t, vProduct(sc,n,t)] + sum[h, vProduct(sc,n,h)] - sum[h, vConsump(sc,n,h)] + vIG(sc,n) + vENS(sc,n)
=e= pDemand(n) ;

eOpReserve(   n  )                 .. sum[t, pMaxProd(t) * vCommitt(n,t)] + sum[h, pMaxProd(h)] =g=   pOperReserve  (n) + pDemand(n) ;
eReserveUp(sc,n  ) $ pScenProb(sc) .. sum[t, pMaxProd(t) * vCommitt(n,t) - vProduct(sc,n,t)]    =g=   pOperReserveUp(n) ;
eReserveDw(sc,n  ) $ pScenProb(sc) .. sum[t, pMinProd(t) * vCommitt(n,t) - vProduct(sc,n,t)]    =l= - pOperReserveDw(n) ;

eMaxOutput(sc,n,t) $[pScenProb(sc) and pMaxProd(t)] .. vProduct(sc,n,t) / pMaxProd(t) =l= vCommitt(n,t) ;
eMinOutput(sc,n,t) $[pScenProb(sc) and pMinProd(t)] .. vProduct(sc,n,t) / pMinProd(t) =g= vCommitt(n,t) ;

eTotOutput(sc,n,t) $ pScenProb(sc)                  .. vProduct(sc,n,t) =e= pMinProd(t)*vCommitt(n,t) + vProduct1(sc,n,t) ;

eRampUp   (sc,n,t) $ pScenProb(sc) .. vProduct1(sc,n,t) - vProduct1(sc,n-1,t) - max[pIniOut(t)-pMinProd(t),0] $n1(n) =l=   pRampUp(t) ;
eRampDw   (sc,n,t) $ pScenProb(sc) .. vProduct1(sc,n,t) - vProduct1(sc,n-1,t) - max[pIniOut(t)-pMinProd(t),0] $n1(n) =g= - pRampDw(t) ;

eUCStrShut(   n,t)                 .. vCommitt(n,t) - vCommitt(n-1,t) - pIniUC(t) $n1(n) =e= vStartup(n,t) - vShutdown(n,t) ;

eMinTUp   (   n,t)                 .. sum[nn $(ord(nn) >= ord(n)+1-pMinTU(t) and ord(nn) <= ord(n)), vStartup (nn,t)] =l=     vCommitt(n,t) ;
eMinTDw   (   n,t)                 .. sum[nn $(ord(nn) >= ord(n)+1-pMinTD(t) and ord(nn) <= ord(n)), vShutdown(nn,t)] =l= 1 - vCommitt(n,t) ;

eWtReserve(sc,n,h) $ pScenProb(sc) .. vWtReserve(sc,n-1,h) + pIniReserve(h) $n1(n) + pInflows(h,n) - vSpillage(sc,n,h) - vProduct(sc,n,h) +
vConsump(sc,n,h)*pEffic(h) =e= vWtReserve(sc,n,h) ;

model mSDUC / all / ;
mSDUC.solprint = 1 ; mSDUC.holdfixed = 1 ; mSDUC.optfile = 1 ;
```

Reduced solution output

Model includes all the equations

Eliminate fixed variables

Mathematical formulation of equations

Stochastic Daily Unit Commitment of Thermal and Hydro Units (SDUC) (v)

```
* read input data from Excel and include into the model

file TMP / tmp_%gams.user1%.txt /
$OnEcho  > tmp_%gams.user1%.txt
   r1=    indices
   o1=tmp_indices.txt
   r2=    param
   o2=tmp_param.txt
   r3=    demand
   o3=tmp_demand.txt
   r4=    oprres
   o4=tmp_oprres.txt
   r5=    oprresup
   o5=tmp_oprresup.txt
   r6=    oprresdw
   o6=tmp_oprresdw.txt
   r7=    IGgen
   o7=tmp_IGgen.txt
   r8=    thermalgen
   o8=tmp_thermalgen.txt
   r9=    hydrogen
   o9=tmp_hydrogen.txt
   r10=   inflows
   o10=tmp_inflows.txt
$OffEcho
* Mac OS X and Linux users must comment the following call and copy and paste the named ranges of the Excel interface into the txt files
$ifthen.OptSkipExcelInput '%OptSkipExcelInput%' == '0'
$call xls2gms m i="%gams.user1%.xlsm" @"tmp_%gams.user1%.txt"
$else.OptSkipExcelInput
$  log Excel input skipped
$endif.OptSkipExcelInput

sets
$include  tmp_indices.txt
;
$include  tmp_param.txt
parameter pDemand(n)        hourly load             [MW] /
$include  tmp_demand.txt
                                                         /
parameter pOperReserve(n)   hourly operating reserve [MW] /
$include  tmp_oprres.txt
                                                         /
parameter pOperReserveUp(n) hourly operating reserve [MW] /
$include  tmp_oprresup.txt
                                                         /
parameter pOperReserveDw(n) hourly operating reserve [MW] /
$include  tmp_oprresdw.txt
                                                         /
table     pIntermGen(n,sc)  stochastic IG generation [MW]
$include  tmp_IGgen.txt
table     pThermalGen(g,*)
$include  tmp_thermalgen.txt
table     pHydroGen  (g,*)
$include  tmp_hydrogen.txt
table     pInflows   (g,n)
$include  tmp_inflows.txt
;

* Mac OS X and Linux users must comment the following execute
execute 'del tmp_"%gams.user1%".txt tmp_indices.txt tmp_param.txt tmp_demand.txt tmp_oprres.txt tmp_oprresup.txt tmp_oprresdw.txt tmp_IGgen.txt tmp_thermalgen.txt tmp_hydrogen.txt tmp_inflows.txt' ;
```

Read input from Excel named ranges and write into text files

Input from text files into GAMS

Delete read text files

Stochastic Daily Unit Commitment of Thermal and Hydro Units (SDUC) (vi)

```
* determine the first hour of the day

n1(n) $[ord(n) = 1] = yes ;

* assignment of thermal units, storage hydro and pumped storage hydro plants

t (g) $[pThermalGen(g,'MaxProd') and pThermalGen(g,'FuelCost')] = yes ;
h (g) $[pHydroGen  (g,'MaxProd')                               ] = yes ;

* scaling of parameters to GW and M€

pDemand       (n   )               = pDemand       (n   ) * 1e-3 ;
pOperReserve  (n   )               = pOperReserve  (n   ) * 1e-3 ;
pOperReserveUp(n   )               = pOperReserveUp(n   ) * 1e-3 ;
pOperReserveDw(n   )               = pOperReserveDw(n   ) * 1e-3 ;
pIntermGen    (n,sc) $pScenProb(sc) = pIntermGen    (n,sc) * 1e-3 ;

pENSCost         = pENSCost                     * 1e-3 ;
pMaxProd     (t) = pThermalGen(t,'MaxProd'     ) * 1e-3 ;
pMinProd     (t) = pThermalGen(t,'MinProd'     ) * 1e-3 ;
pIniOut      (t) = pThermalGen(t,'IniProd'     ) * 1e-3 ;
pRampUp      (t) = pThermalGen(t,'RampUp'      ) * 1e-3 ;
pRampDw      (t) = pThermalGen(t,'RampDw'      ) * 1e-3 ;
pMinTU       (t) = pThermalGen(t,'MinTU'       )        ;
pMinTD       (t) = pThermalGen(t,'MinTD'       )        ;
pSlopeVarCost(t) = pThermalGen(t,'OMVarCost'   ) * 1e-3 +
                   pThermalGen(t,'SlopeVarCost') * 1e-3 * pThermalGen(t,'FuelCost') ;
pEmissionCost(t) = pThermalGen(t,'EmissionRate') * 1e-3 * pCO2Cost ;
pInterVarCost(t) = pThermalGen(t,'InterVarCost') * 1e-6 * pThermalGen(t,'FuelCost') ;
pStartupCost (t) = pThermalGen(t,'StartupCost' ) * 1e-6 * pThermalGen(t,'FuelCost') ;
pShutdownCost(t) = pThermalGen(t,'ShutdownCost') * 1e-6 * pThermalGen(t,'FuelCost') ;

pMaxProd     (h) = pHydroGen  (h,'MaxProd'     ) * 1e-3 ;
pMinProd     (h) = pHydroGen  (h,'MinProd'     ) * 1e-3 ;
pMaxCons     (h) = pHydroGen  (h,'MaxCons'     ) * 1e-3 ;
pEffic       (h) = pHydroGen  (h,'Efficiency'  )        ;
pMaxReserve  (h) = pHydroGen  (h,'MaxReserve'  ) * 1e-3 ;
pMinReserve  (h) = pHydroGen  (h,'MinReserve'  ) * 1e-3 ;
pIniReserve  (h) = pHydroGen  (h,'IniReserve'  ) * 1e-3 ;

* if the initial output of the unit is above its minimum load then the unit is committed, otherwise it is not committed
pIniUC       (g) = 1 $[pIniOut(g) >= pMinProd(g)] ;

* if the efficiency of a hydro plant is 0, it is changed to 1
pEffic    (h) $[pEffic    (h) = 0] =   1 ;

* if the minimum up or down times are 0, they are changed to 1
pMinTU    (t) $[pMinTU    (t) = 0] =   1 ;
pMinTD    (t) $[pMinTD    (t) = 0] =   1 ;
```

First hour of the day

Scaling of parameters

Initial committed units

Stochastic Daily Unit Commitment of Thermal and Hydro Units (SDUC) (vii)

```
* bounds on variables

vProduct.up  (sc,n,g) $pScenProb(sc) = pMaxProd   (g   ) ;
vConsump.up  (sc,n,g) $pScenProb(sc) = pMaxCons   (g   ) ;
vProduct1.up (sc,n,t) $pScenProb(sc) = pMaxProd   (t   ) - pMinProd(t) ;
vIG.up       (sc,n  ) $pScenProb(sc) = pIntermGen (n,sc) ;
vENS.up      (sc,n  ) $pScenProb(sc) = pDemand    (n   ) ;
vWtReserve.up(sc,n,g) $pScenProb(sc) = pMaxReserve(g   ) ;
vWtReserve.lo(sc,n,g) $pScenProb(sc) = pMinReserve(g   ) ;

vCommitt.up (   n,g) = 1 ;
vStartup.up (   n,g) = 1 ;
vShutdown.up(   n,g) = 1 ;

* solve stochastic daily unit commitment model

solve mSDUC using MIP minimizing vTotalVCost ;

* scaling of the results

pCommitt(   t,n)               = vCommitt.l(   n,t)                + eps ;
pProduct(sc,g,n) $pScenProb(sc) = vProduct.l(sc,n,g)*1e3           + eps ;
pIG     (sc,  n) $pScenProb(sc) = vIG.l     (sc,n  )*1e3           + eps ;
pSRMC   (sc,  n) $pScenProb(sc) = eBalance.m(sc,n  )*1e3/pScenProb(sc) + eps ;

* data output to xls file

put TMP putclose 'par=pCommitt rdim=1 rng=UC!a1' / 'par=pProduct rdim=2 rng=Output!a1' / 'par=pIG    rdim=1 rng=IG!a1' / 'par=pSRMC rdim=1
rng=SRMC!a1' /
                 'text="Unit"          rng=UC!a1' / 'text="Scen"        rng=Output!a1' / 'text="Scen"      rng=IG!a1' / 'text="Scen"
rng=SRMC!a1' /
                                                    'text="Unit"        rng=Output!b1'
execute_unload   'tmp_%gams.user1%.gdx' pProduct pCommitt pIG pSRMC
*$ifthen.OptSkipExcelOutput '%OptSkipExcelOutput%' == '0'
execute          'gdxxrw tmp_"%gams.user1%".gdx SQ=n EpsOut=0 O=tmp_"%gams.user1%".xlsx @tmp_"%gams.user1%".txt'
execute          'del    tmp_"%gams.user1%".gdx                                                                    '
*$else.OptSkipExcelOutput
*$  log Excel output skipped
*$endif.OptSkipExcelOutput
execute          'del                                                                    tmp_"%gams.user1%".txt'

$OnListing
```

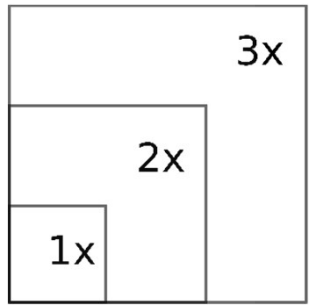
Bounds on variables

Solve the optimization problem

Scaling the results

Write output to Excel

Scaling



- Solvers are powerful but not magic
- Input data and output results must be in commonly used units
- But internally variables, equations, parameters must be around 1 (i.e., from 0.01 to 100). Ratio of largest to smallest matrix coefficient should be < 10^5
- Scaling can be done:
  - Manually (e.g., from MW to GW, from euros to Meuros). Modelers can typically do better because they know the problem
  - Automatically by the language (ModelName.ScaleOpt=1)
  - By the solver (`ScaInd 1` in CPLEX, `ScaleFlag 2` in GUROBI)
- Specially useful in large-scale LP problems or NLP problems and/or when willing to use the dual variables
- The condition number is a measure for the sensitivity of the solution of a system of linear equations to errors in the data.
  - It is the ratio between the largest and smallest eigenvalues
- Condition numbers <10^6 are good enough. Numerical problems arise for condition numbers >10^8 (*ill-conditioned*)
  - `Quality 1` in CPLEX
  - `Kappa 1` in GUROBI

Models with numerical issues can lead to undesirable results: slow performance, wrong answers or inconsistent behavior. Source: GUROBI

Rule of thumb for selecting an LP optimization algorithm

- Simplex (or dual simplex) method can be the best choice for moderate size (up to 100000 x 100000)
- Interior point method is usually the most efficient for huge and difficult problems
  - It is the most *numerically sensitive* algorithm. Numerical issues can cause crossover to stall
  - It can be *threaded* quite efficiently (compared to simplex)
- Difference in solution time can reach one order of magnitude

Options

Options	
LimRow	Number of rows to show
LimCol	Number of columns to show
SolPrint	Solution output
SolveOpt	Replace
Decimals	Number of decimals in displaying values
IterLim	Maximum number of solver iterations
ResLim	Maximum solution time
Profile	Time profiling
ProfileTol	Profile threshold
Seed	Initialize seed for random numbers

$ Directives

$ Directives	
$OnEmpty	Allow introduction of empty sets
$OnMulti	Allow redeclaration of sets
$OffListing	Suppress listing of the code

Variable attributes

Attributes	
VarName.lo	lower bound
VarName.up	upper bound
VarName.fx	fixes the variable to a constant
VarName.Range	range of the variable
VarName.l	initial value before and optimal value after
VarName.m	marginal value (reduced cost)
VarName.Scale	numerical scale factor
VarName.Prior	branching priority in a MIP model (∞ →not discrete)
VarName.SlackUp	slack from upper bound
VarName.SlackLo	slack from lower bound
VarName.Infeas	infeasibility out of bounds

Equation attributes

Attributes	
EquationName.lo	lower bound
EquationName.up	upper bound
EquationName.l	initial value before and optimal value after
EquationName.m	marginal value (dual variable or shadow price)
EquationName.Scale	numerical scaling factor

Model

```
model ModelName1 / Equation1 Equation3 Equation5 Equation7 /
model ModelName2 / all /
model ModelName3 / ModelName1 – Equation5 + Equation8 /
```

Model attributes

Attributes	
ModelName.ResLim	Resource limit
ModelName.SolveOpt	Replace/merge/clear in consecutive solves
ModelName.SolSlack	Show slack variables
ModelName.SolvePrint	0, 1, 2 (to remove the detailed solution from the .lst file)
ModelName.TryLinear	Try linear model first
ModelName.ModelStat	Model status
ModelName.SolveStat	Solve status
ModelName.NumEqu	Number of equations
ModelName.NumVar	Number of variables
ModelName.NumDVar	Number of discrete variables
ModelName.NumNz	Number of non zeros

Attributes	
ModelName.IterUsd	Number of iterations
ModelName.ResUsd	Resource used
ModelName.BRatio	Basis ratio controls the use of previous basis
ModelName.HoldFixed	Fix and eliminate variables
ModelName.IterLim	Iteration limit
ModelName.NodLim	Node limit
ModelName.OptCA	Absolute optimality tolerance
ModelName.OptCR	Relative optimality tolerance
ModelName.OptFile	Use of an option file
ModelName.PriorOpt	Use of priority

GAMS Call Options

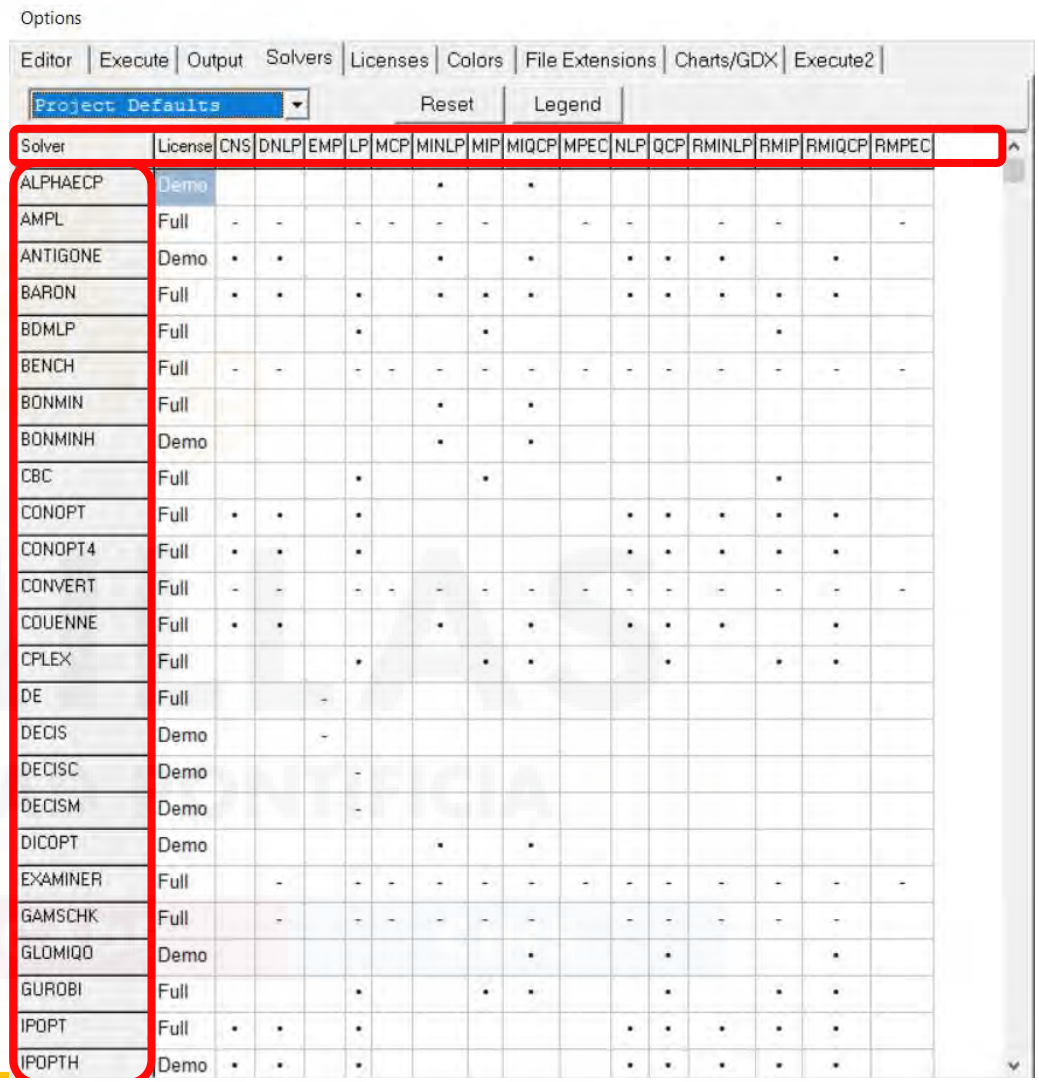
GAMS Options	
Suppress	Suppress echo of the code listing
PW	Page width
PS	Page size
RF	Shows all the symbols
Charset	Allows international characters
U1..U10	User parameter

For example, InterfaceName, SolverSelection, SkipExcelInput, SkipExcelOutput,

```
u1="Excel_Interface_Name" u2=0 u3=0 u4=1 --NumberCores=4
```

Solvers

- All the solvers are hooked to GAMS



Solver	License	CNS	DNLP	EMP	LP	MCP	MINLP	MIP	MIQCP	MPEC	NLP	QCP	RMINLP	RMIP	RMIQCP	RMPEC
ALPHAECP	Demo						•		•							
AMPL	Full	-	-		-	-	-	-		-	-		-	-		-
ANTIGONE	Demo	•	•				•		•		•	•	•		•	
BARON	Full	•	•		•		•	•	•		•	•	•	•	•	
BDMLP	Full				•			•						•		
BENCH	Full	-	-		-	-	-	-	-	-	-	-	-	-	-	-
BONMIN	Full						•		•							
BONMINH	Demo						•		•							
CBC	Full				•			•						•		
CONOPT	Full	•	•		•						•	•	•	•	•	
CONOPT4	Full	•	•		•						•	•	•	•	•	
CONVERT	Full	-	-		-	-	-	-	-	-	-	-	-	-	-	-
COUENNE	Full	•	•				•		•		•	•	•		•	
CPLEX	Full				•			•	•			•		•	•	
DE	Full			-												
DECIS	Demo			-												
DECISC	Demo				-											
DECISM	Demo				-											
DICOPT	Demo						•		•							
EXAMINER	Full		-		-	-	-	-	-	-	-	-	-	-	-	-
GAMSCHK	Full		-		-	-	-	-	-		-	-	-	-	-	
GLOMIQO	Demo								•			•			•	
GUROBI	Full				•			•	•			•		•	•	
IPOPT	Full	•	•		•						•	•	•	•	•	
IPOPTH	Demo	•	•		•						•	•	•	•	•	

Solvers

Solver	License	CNS	DNLP	EMP	LP	MCP	MINLP	MIP	MIQCP	MPEC	NLP	QCP	RMINLP	RMIP	RMIQCP	RMPEC
PYOMO	Full	-	-		-	-	-	-		-	-		-	-		-
SBB	Full						•		•							
SCIP	Full	•	•				•	•	•		•	•	•		•	
SELKIE	Full			•												
SNOPT	Demo	•	•		•						•	•	•	•	•	
SOLVEENGINE	Demo				•			•						•		
SOPLEX	Full				•									•		
XA	Demo				•			•						•		
XPRESS	Demo				•			•	•			•		•	•	

Solver	License	CNS	DNLP	EMP	LP	MCP	MINLP	MIP	MIQCP	MPEC	NLP	QCP	RMINLP	RMIP	RMIQCP	RMPEC
JAMS	Full			•												
KESTREL	Full	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
KNITRO	Demo	•	•		•		•		•	•	•	•	•	•	•	•
LGO	Demo		•		•						•	•	•	•	•	
LINDO	Demo		•	•	•		•	•	•		•	•	•	•	•	
LINDOGLOBAL	Demo		•		•		•	•	•		•	•	•	•	•	
LINGO	Full		-		-		-	-			-		-	-		
LOCALSOLVER	Demo	•	•				•	•	•		•	•	•		•	
LOCALSOLVER70	Demo	•	•				•	•	•		•	•	•		•	
LOGMIP	Full			•												
LS	Full				-									-		
MILES	Full					•										
MINOS	Demo	•	•		•						•	•	•	•	•	
MOSEK	Demo		•		•			•	•		•	•	•	•	•	
MPECDUMP	Full	-	-		-	-	-	-		-	-		-	-		-
MPSGE	Demo															
MSNLP	Demo		•								•	•	•		•	
NLPEC	Full					•				•						•
ODHCPLEX	Full							•	•							
OSICPLEX	Full				•			•						•		
OSIGUROBI	Full				•			•						•		
OSIMOSEK	Full				•			•						•		
OSIXPRESS	Full				•			•						•		
PATH	Full	•				•										
PATHNLP	Full		•		•						•	•	•	•	•	

Boosting performance

- Threads
- GUSS (Gather-Update-Solve-Scatter)
  - Use of **sensitivity analysis** for solving many similar problems
- Grid and Multi-Threading Solve Facility
  - Use of **multiple cores** of a computer
- You can **launch several GAMS process simultaneously** being careful with conflicting filenames

Scenario analysis of the transportation problem solved with GUSS

```
sets
   I  origins
   J  destinations
   SC scenarios

parameters
   pA    (i     ) origin capacity
   pB    (j     ) destination demand
   pC    (i,j   ) per unit transportation cost
   pBS   (sc,  j) stochastic destination demand
   pX    (sc,i,j) stochastic units transported
   pCost (sc    ) stochastic transportation cost
   pPrice(sc,  j) stochastic spot price

variables
   vX(i,j) units transported
   vCost   transportation cost

positive variable vX

equations
   eCost        transportation cost
   eCapacity(i) maximum capacity of each origin
   eDemand  (j) demand supply at destination ;

eCost ..        sum[(i,j), pC(i,j) * vX(i,j)] =e= vCost ;
eCapacity(i) .. sum[   j ,           vX(i,j)] =l= pA(i) ;
eDemand  (j) .. sum[ i,              vX(i,j)] =g= pB(j) ;

model mTransport / all /
```

```
set scen_dem stochastic demand scenario dictionary /
   sc      . scenario . ''

*  update   the LHS with values of the RHS
   pB      . param    . pBS

*  store in the RHS with values of the LHS
   vX      . level    . pX
   vCost   . level    . pCost
   eDemand . marginal . pPrice                    /
************************************************************
sets
   I origins      / VIGO, ALGECIRAS /
   J destinations / MADRID, BARCELONA, VALENCIA /
   SC scenarios   / sc000*sc999 /

parameters
   pA(i) origin capacity
       / VIGO      350
         ALGECIRAS 700 /

   pBS(sc,j) stochastic destination demand
      / sc000 . MADRID    400
        sc000 . BARCELONA 450
        sc000 . VALENCIA  150 / ;

* lazy input, feeding data for all the scenarios with random demand
pBS(sc,j) = pBS('sc000',j) * [1+uniform(-0.05,0.05)] ;

table pC(i,j) per unit transportation cost
           MADRID BARCELONA VALENCIA
VIGO        0.06     0.12     0.09
ALGECIRAS   0.05     0.15     0.11 ;
************************************************************

* initialization of the destination demand
pB(j) = pBS('sc000',j) ;

solve mTransport using LP minimizing vCost scenario scen_dem
```

Scenario analysis of the transportation problem solved with Grid computing and GUSS (i)

```
sets
   I  origins
   J  destinations
   SC scenarios

parameters
   pA    (i     ) origin capacity
   pB    (j     ) destination demand
   pC    (i,j   ) per unit transportation cost
   pBS   (sc,  j) stochastic destination demand
   pX    (sc,i,j) stochastic units transported
   pCost (sc    ) stochastic transportation cost
   pPrice(sc,  j) stochastic spot price

variables
   vX(i,j) units transported
   vCost   transportation cost

positive variable vX

equations
   eCost        transportation cost
   eCapacity(i) maximum capacity of each origin
   eDemand  (j) demand supply at destination ;

eCost ..        sum[(i,j), pC(i,j) * vX(i,j)] =e= vCost ;
eCapacity(i) .. sum[   j ,           vX(i,j)] =l= pA(i) ;
eDemand  (j) .. sum[ i,              vX(i,j)] =g= pB(j) ;

model mTransport / all /
```

```
sets
   gs(sc)   scenarios per GUSS run
   sh       solution headers / System.GUSSModelAttributes /
   scen_dem stochastic demand scenario dictionary /
      sc        . scenario . ''
      scen_optn . opt      . st_report_o

*     update   the LHS with values of the RHS
      pB        . param    . pBS

*     store in the RHS with values of the LHS
      vX        . level    . pX
      vCost     . level    . pCost
      eDemand   . marginal . pPrice              /
```

```
************************************************************
sets
   I origins      / VIGO, ALGECIRAS /
   J destinations / MADRID, BARCELONA, VALENCIA /
   SC scenarios   / sc000*sc999 /

parameters
   pA(i) origin capacity
      / VIGO      350
        ALGECIRAS 700 /

   pBS(sc,j) stochastic destination demand
      / sc000 . MADRID    400
        sc000 . BARCELONA 450
        sc000 . VALENCIA  150 / ;

* lazy input, feeding data for all the scenarios with random demand
pBS(sc,j) = pBS('sc000',j) * [1+uniform(-0.05,0.05)] ;

table pC(i,j) per unit transportation cost
           MADRID BARCELONA VALENCIA
VIGO        0.06     0.12     0.09
ALGECIRAS   0.05     0.15     0.11 ;
```

Scenario analysis of the transportation problem solved with Grid computing and GUSS (ii)

```
sets
*  using four cores and assignment of scenarios to cores
   core            grid jobs to run  / core001*core004       /
   coresc(core,sc) cores to scenario / core001.(sc000*sc249)
                                       core002.(sc250*sc499)
                                       core003.(sc500*sc749)
                                       core004.(sc750*sc999) /

parameter
   scen_optn          scenario options / OptFile 2, LogOption 1, SkipBaseCase 1,
                                        UpdateType 1, RestartType 1, NoMatchLimit 999 /
   st_report_o(sc,sh) status report
   pGridHandle(core)  grid handles ;

* initialization of the destination demand
pB(j) = pBS('sc000',j) ;

mTransport.SolveLink = %SolveLink.AsyncGrid% ;
loop (core,
   gs(sc) = coresc(core,sc)
   if (sum[gs(sc), 1] > 0,
      solve mTransport using LP minimizing vCost scenario scen_dem ;
      pGridHandle(core) = mTransport.Handle ;
   ) ;
) ;
repeat
   loop (core $HandleCollect(pGridHandle(core)),
      display $HandleDelete (pGridHandle(core)) 'Trouble deleting handles' ;
      pGridHandle(core) = 0 ;
   ) ;
until card(pGridHandle) = 0 or TimeElapsed > 1000 ;
mTransport.SolveLink = %SolveLink.LoadLibrary% ;

display st_report_o
```

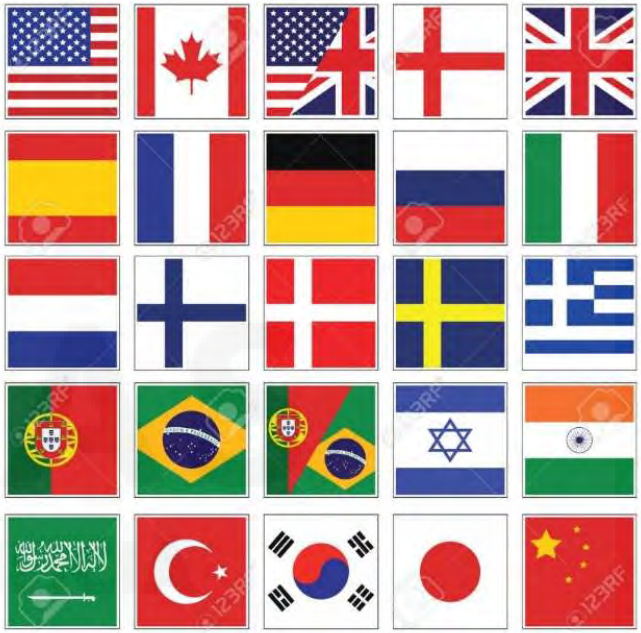
How to write multiple language versions

```
* Language for Excel headings: Spanish 0 English 1 French 2
$set language               1


$ifthen.language %language% == 0
$  set iTitle MiModelo Versión 6.19 --- 11 Julio 2018
$  include ModelEs.gms
$elseif.language %language% == 1
$  set iTitle MyModel Release 6.19 --- July 11, 2018
$  include ModelEn.gms
$elseif.language %language% == 2
$  set iTitle MonModel Version 6.19 --- 11 Juillet 2018
$  include ModelFr.gms
$endif.language
```

```
* File ModelEs.gms
$setglobal Lema Más sabe el diablo por viejo que por diablo
```

```
* File ModelEn.gms
$setglobal Lema The devil knows many things because he is old
```

```
* File ModelFr.gms
$setglobal Lema Le diable sait beaucoup parce qu'il est vieux
```



Introducing flexibility

```
$SetGlobal ppp 100

parameter pDimension / %ppp% /
set u       / unit1*unit%ppp% /

display pDimension, u
```

Releasing memory

a) Define a dummy solve

b) Clear parameters

c) Run dummy model

```
option profile=10

set i / 1 * 10000000 /
parameter pp(i) ;

pp(i) = 33 ;

* dummy optimization problem used for releasing memory
variable vDummy
equation eDummy ; eDummy .. vDummy =e= 0 ;
model    mDummy / eDummy / ;

* only parameters that are no longer used can be cleared
option Clear=pp

* solve a dummy optimization problem to release memory usage
solve mDummy using LP minimizing vDummy
```

GAMS to LaTeX

```
sets
   I origins      / VIGO, ALGECIRAS /
   J destinations / MADRID, BARCELONA, VALENCIA /

parameters
   pA(i) origin capacity
       / VIGO      350
         ALGECIRAS 700 /

   pB(j) destination demand
       / MADRID    400
         BARCELONA 450
         VALENCIA  150 /

table pC(i,j) per unit transportation cost
           MADRID BARCELONA VALENCIA
VIGO         0.06      0.12     0.09
ALGECIRAS    0.05      0.15     0.11

variables
   vX(i,j) units transported
   vCost   transportation cost

positive variable vX

equations
   eCost        transportation cost
   eCapacity(i) maximum capacity of each origin
   eDemand  (j) demand supply at destination ;

eCost        .. sum[(i,j), pC(i,j) * vX(i,j)] =e= vCost ;
eCapacity(i) .. sum[   j ,           vX(i,j)] =l= pA(i) ;
eDemand  (j) .. sum[ i   ,           vX(i,j)] =g= pB(j) ;

model mTransport / all /
solve mTransport using LP minimizing vCost
```

$$\min_x \sum_{ij} c_{ij} x_{ij}$$

$$\sum_j x_{ij} \le a_i \quad \forall i$$

$$\sum_i x_{ij} \ge b_j \quad \forall j$$

$$x_{ij} \ge 0$$

Generate the doc file

gams transport.gms **DocFile**=transport

Write the tex file

model2tex transport

Symbols

Sets

Name	Domains	Description
I	*	origins
J	*	destinations

Parameters

Name	Domains	Description
pA	I	origin capacity
pB	J	destination demand
pC	I, J	per unit transportation cost

Variables

Name	Domains	Description
vX	I, J	units transported
vCost		transportation cost

Equations

Name	Domains	Description
eCost		transportation cost
eCapacity	I	maximum capacity of each origin
eDemand	J	demand supply at destination

Equation Definitions

eCost

$$\sum_{I,J} (\mathrm{pC}_{I,J} \cdot \mathrm{vX}_{I,J}) = \mathrm{vCost}$$

eCapacity$_I$

$$\sum_J (\mathrm{vX}_{I,J}) \le \mathrm{pA}_I \qquad \forall I$$

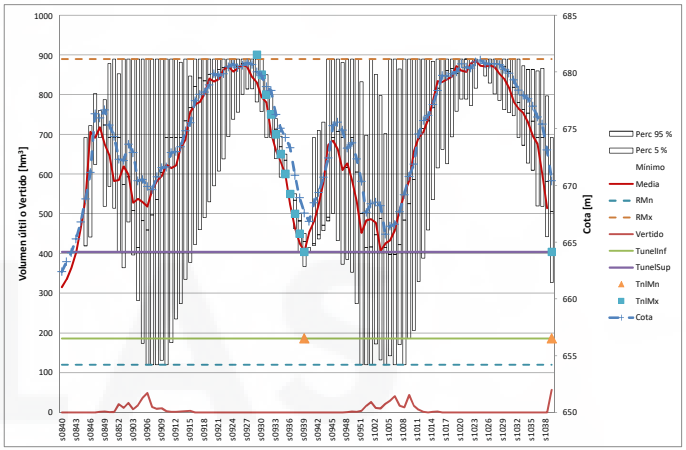
eDemand$_J$

$$\sum (\mathrm{vX}_{I,J}) \ge \mathrm{pB}_J \qquad \forall J$$

$\mathrm{vX}_{I,J} \ge 0 \ \forall I, J$

Each tool has a specific purpose

- GDX (GAMS Data eXchange) utilities to interface with other applications
- Interfaces with other programs
  - Microsoft Excel (gdx2xls, xls2gms)
  - Microsoft Access (gdx2access, mdb2gms)
  - SQL (gdx2sqlite, sql2gms)
  - Matlab (gdxmrw)
  - R (gdxrrw)
- APIs
  - .Net
  - Java
  - Python



- Input/output data and simple graphs (Microsoft Excel, Microsoft Access)
- Advanced graphs (GNUPlot, Matlab)

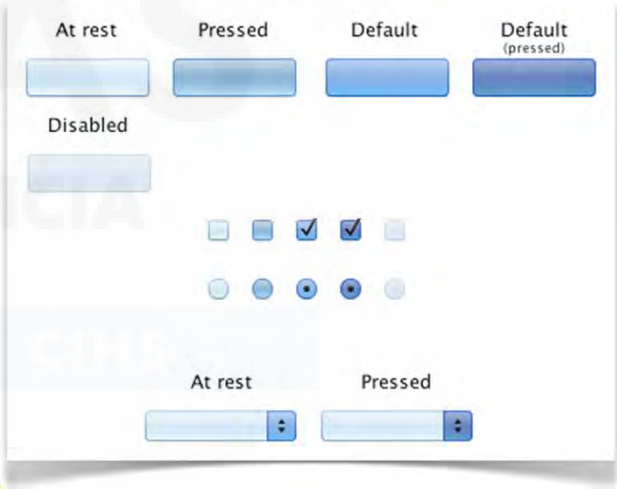
1. Programming Style
2. GAMS Code
3. Look and Feel
4. Mathematical Formulation
5. Advanced Algorithms

3

Look and Feel

Look and feel

In software design, look and feel is used in respect of a graphical user interface and comprises aspects of its design, including elements such as colors, shapes, layout, and typefaces (the “look”), as well as the behavior of dynamic elements such as buttons, boxes, and menus (the “feel”) (http://en.wikipedia.org/wiki/Look_and_feel)



Interface (https://iit.comillas.edu/aramos/StarGenLite_SDUC.zip)

- Microsoft Excel (with macros[1]) is the most popular application in business environment for data manipulation and is used as interface to
  - Introduce the input data,
  - Run the model and
  - Present the output results numerically and graphically

- Menu sheet contains buttons to
  - Run the model
  - Load the results
- Green sheets are used to introduce input data in named ranges
  - Green cells can be modified by the user
    - The other ones it is suggested not to modify them.
- Salmon sheets are used to present the output data
- Blue sheets are used to plot the results

1. *Enable the use of macros when opening it in Excel*

Input / Output

- Input
  - Use simple input formats. Make input easy to prepare and adapted to the user.
  - Check input for validity and plausibility. This can be done in input/output interface (e.g., using conditional formatting in Microsoft Excel) or in GAMS
  - Identify bad input, recover if possible, if not abort the execution giving information to the user
  - Document your data layouts
  - Definitions with physical dimensions
  - References to the data origin
- Output
  - It is difficult to determine the infeasibility of an optimization program. It is better to check bad data.
  - Produce self-explanatory output.

Input Sheets

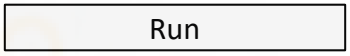
- In each sheet one o several named ranges are included to input the data.
- You can introduce or delete data but the named ranges have to be kept. To avoid errors it is important to insert/delete rows/columns out of the first or last row/column, i.e., in any intermediate row/column.
- The range names can not be changed by the user, are predefined by the GAMS model.
- The named range input in included directly as GAMS input files so it must be grammatically correct.

Output Sheets

- Their names are predefined by the model and can not be changed by the user.

Running the model

- From the Microsoft Excel interface
  - Write the GAMS directory
  - Press the button Run



  - Remember to look for errors in StarGenLite_XXX.lst
  - Press the button Load results

- From the GAMSIde
  - Write u1="StarGenLite_XXX" in the upper right window
  - Run GAMS (F9)
  - Remember to look for errors in StarGenLite_XXX.lst
  - Results are written to Microsoft Excel file tmp_StarGenLite_XXX.xls
  - Press the button Load results form the Microsoft Excel interface

1. Programming Style
2. GAMS Code
3. Look and Feel
4. Mathematical Formulation
5. Advanced Algorithms

4

Mathematical Formulation

Reformulation in MIP problems

- Most MIP problems can be formulated in different ways
- In MIP problems, a good formulation is crucial to solve the model
- How good is a MIP formulation?
  - Integrality gap difference between the objective function of the MIP and LP relaxation solutions
- Given two equivalent MIP formulations, one is *stronger* (*tighter/better*) than the other, if the feasible region of the linear relaxation is strictly contained in the feasible region of the other one. Integrality gap is lower.

E. Klotz, A.M. Newman *Practical guidelines for solving difficult mixed integer linear programs* Surveys in Operations Research and Management Science 18 (1-2), 18-32, Oct 2013 10.1016/j.sorms.2012.12.001

Preprocessing by GUROBI

```
--- StarNetLite_TEPM_Iceland.gms(15000) 986 Mb
---   1,167,736 rows  2,004,441 columns  8,140,314 non-zeroes
---   0 nl-code  0 nl-non-zeroes
---   7 discrete-columns
***   63,652 relaxed-columns WARNING
--- StarNetLite_TEPM_Iceland.gms(15000) 984 Mb
--- Executing GUROBI: elapsed 0:00:20.917
--- StarNetLite_TEPM_Iceland.gms(15000) 984 Mb  3 secs

Gurobi           24.6.1 r55820 Released Jan 18, 2016 WEI x86 64bit/MS
Windows

Gurobi link license.
Gurobi library version 6.5.0
Reading parameter(s) from "C:\Users\aramos\Desktop\Aramos\TEPES\gurobi.opt"
>>  Method   2
>>  IntFeasTol 1e-9
>>  OptimalityTol 1e-9
>>  FeasibilityTol 1e-9
>>  IIS   1
>>  RINS     100
>>  DisplayInterval 1
>>  NumericFocus      1
>>  Kappa   1
>>  MarkowitzTol 0.999
>>  UseBasis 0
>>  CrossOver 0
>>  Names  0
>>  AggFill 0
>>  GomoryPasses 0
>>  Heuristics 0.001
>>  MipFocus 3
>>  *PreDual 0
>>  *PrePasses 3
>>  *PreSolve 2
>>  *PreSparsify 1

Finished reading from "C:\Users\aramos\Desktop\Aramos\TEPES\gurobi.opt"
Starting Gurobi...
Optimize a model with 1167735 rows, 2004440 columns and 8140308 nonzeros
Coefficient statistics:
  Matrix range    [7e-09, 1e+03]
  Objective range [1e+00, 1e+00]
  Bounds range    [4e-07, 5e+00]
  RHS range       [1e-18, 1e+01]
Presolve removed 257457 rows and 497937 columns (presolve time = 2s) ...
Presolve removed 259520 rows and 500000 columns (presolve time = 2s) ...
Presolve removed 470416 rows and 710896 columns (presolve time = 3s) ...
Presolve removed 470616 rows and 711132 columns (presolve time = 4s) ...
Presolve removed 470666 rows and 753485 columns (presolve time = 5s) ...
Presolve removed 470705 rows and 753535 columns (presolve time = 6s) ...
Presolve removed 470705 rows and 753535 columns
Presolve time: 6.23s
Presolved: 697030 rows, 1250905 columns, 5645696 nonzeros
```

40 % reduction in rows,
38 % in columns and
30 % in nonzeros

Preprocessing by GUROBI

- Before and after presolve can help you in detecting improvements in the formulation
- Allows to get the optimization problem after the presolve

```
ModelName          = read(“OriginalProblem.lp")
ModelNamePresolved = ModelName.presolve()
ModelNamePresolved.write(“PresolvedProblem.lp")
```

Warehouse location problem (no limits) (i)

- Choose where to locate warehouses among a set of locations and assign clients to the warehouses minimizing the total cost. No limits means that there is no limit in the number of clients assigned to a warehouse.
- Data

j locations

i clients

c_j localization cost in j

h_{ij} cost of satisfying the demand of client i from j

- Variables

$$y_j = \begin{cases} 1 & \text{warehouse located in } j \\ 0 & \text{otherwise} \end{cases}$$

x_{ij} fraction of demand of client i met from j

Warehouse location problem (no limits) (ii)

Formulation #1

$$\min \sum_j c_j y_j + \sum_{ij} h_{ij} x_{ij}$$
$$\sum_j x_{ij} = 1 \quad \forall i$$
$$x_{ij} \le y_j \quad \forall ij$$
$$y_j \in \{0,1\}, x_{ij} \in [0,1]$$

Number of constraints: $I + IJ$

Formulation #2

$$\min \sum_j c_j y_j + \sum_{ij} h_{ij} x_{ij}$$
$$\sum_j x_{ij} = 1 \quad \forall i$$
$$\sum_i x_{ij} \le M y_j \quad \forall j$$
$$y_j \in \{0,1\}, x_{ij} \in [0,1]$$

Number of constraints: $I + J$

- Both formulations are MIP equivalent. However, formulation #1 is stronger
- Intuitively the fewer constraints the better. That's true in LP. However, in many MIP problems the more constraints the better.

Production problem with fixed and inventory costs (i)

- Data

t time period

c_t fixed cost, p_t variable cost, h_t inventory cost

d_t demand

- Variables

$$y_t = \begin{cases} 1 \text{ to produce} \\ 0 \text{ not produce} \end{cases}$$

x_t amount produced

s_t inventory at the end of the period

- Formulation #1

$$\min \sum_t \left(c_t y_t + p_t x_t + h_t s_t \right)$$

$$s_{t-1} + x_t = d_t + s_t \quad \forall t$$

$$x_t \le M y_t \quad \forall t$$

$$s_0 = s_T = 0$$

$$x_t, s_t \ge 0, y_t \in \{0,1\}$$

Number of constraints: $2T$

Number of variables: $3T$

Production problem with fixed and inventory costs (ii)

- Variables

$$y_t = \begin{cases} 1 \text{ to produce} \\ 0 \text{ not produce} \end{cases}$$

q_{it} quantity produced in period i to meet the demand in period $t \geq i$

- Formulation #2

$$\min \sum_{t=1}^{T} \sum_{i=1}^{t} (p_i + h_i + h_{i+1} + \cdots + h_{t-1}) q_{it} + \sum_{t=1}^{T} c_t y_t$$

$$\sum_{i=1}^{t} q_{it} = d_t \quad \forall t$$

$$q_{it} \leq d_t y_i \quad \forall it$$

$$q_{it} \geq 0, y_t \in \{0,1\}$$

Number of constraints: $T + T^2/2$

Number of variables: $T + T^2/2$

- Formulation #2 is better. However, it has a greater number of constraints and variables.

Some tips for MIP

- It can be interesting increase the number of variables if they can be used in the branching strategy of B&B.
  - For example, “artificial” division of a zone in regions N, S, E and W to branch first in these zonal regions.
- Alternatively, introduce lazy constraints (only in GAMS/GUROBI under GAMS)
- Avoid the use of big M parameters or put tight (lowest upper bound) values for the big M
- GAMS/CPLEX/GUROBI supports the use of an indicator constraint $x \leq My$

$$\begin{aligned} &minFy + Vx \\ &x \leq My \\ &x \geq 0 \\ &y \in \{0,1\} \end{aligned} \quad \rightarrow \quad \begin{aligned} &minFy + Vx \\ &x \leq 0 \\ &x \geq 0 \\ &y \in \{0,1\} \end{aligned}$$

Write in the file cplex.opt

```
indic constraint$y 0
```

Tight and compact unit commitment

- G. Gentile, G. Morales-España and A. Ramos *A Tight MIP Formulation of the Unit Commitment Problem with Start-up and Shut-down Constraints* EURO Journal on Computational Optimization 5 (1), 177–201 March 2017 10.1007/s13675-016-0066-y
- G. Morales-España, C.M. Correa-Posada, A. Ramos *Tight and Compact MIP Formulation of Configuration-Based Combined-Cycle Units* IEEE Transactions on Power Systems 31 (2), 1350-1359, March 2016 10.1109/TPWRS.2015.2425833
- G. Morales-España, J.M. Latorre, and A. Ramos *Tight and Compact MILP Formulation for the Thermal Unit Commitment Problem* IEEE Transactions on Power Systems 28 (4): 4897–4908, Nov 2013 10.1109/TPWRS.2012.2222938
- G. Morales-España, J.M. Latorre, and A. Ramos *Tight and Compact MILP Formulation of Start-Up and Shut-Down Ramping in Unit Commitment* IEEE Transactions on Power Systems 28 (2): 1288-1296, May 2013 10.1109/TPWRS.2012.2222938

Ramp constraints

$$P_{nt} - P_{n-1,t} \leq rup_t$$
$$P_{n-1,t} - P_{nt} \leq rdw_t$$

Classical

$$P_{nt} - P_{n-1,t} \leq rup_t \, UC_{nt}$$
$$P_{n-1,t} - P_{nt} \leq rdw_t (UC_{nt} + SD_{nt})$$

Tighter

- Ramp equations are considered in the periods only when the unit is connected

P_{nt} : output above the minimum load
rup_t: upwards ramp limit for generator t
rdw_t: downwards ramp limit for generator t
UC_{nt}: 1 if generator t is connected in hour n, 0 otherwise
SU_{nt}: 1 if generator t is started in hour n
SD_{nt}: 1 if generator t is shutdown in hour n

Why the constraint is tighter?

Given that the constraint uses the output above the minimum load, it can only be applied when the unit is committed until the following period the unit was committed.

	n	1	2	3	4	5	6	7	8	9	10
	UC_n	0	0	0	0	1	1	1	1	0	0
	SU_n	0	0	0	0	1	0	0	0	0	0
	SD_n	0	0	0	0	0	0	0	0	1	0
RampUp	UC_n	0	0	0	0	1	1	1	1	0	0
RampDw	$UC_n + SD_n$	0	0	0	0	1	1	1	1	1	0

Reformulation of an NLP problem

$$\min \sum_{i=1}^{n} \sum_{j=i+1}^{n} q_{ij} x_i x_j$$

$$\sum_{j=1}^{n} x_j = 1$$

$$\sum_{j=1}^{n} r_j x_j = r_0$$

$$\min \sum_{i=1}^{n} x_i \sum_{j=i+1}^{n} q_{ij} x_j$$

$$\sum_{j=1}^{n} x_j = 1$$

$$\sum_{j=1}^{n} r_j x_j = r_0$$

$$\min \sum_{i=1}^{n} x_i w_i$$

$$w_i = \sum_{j=i+1}^{n} q_{ij} x_j$$

$$\sum_{j=1}^{n} x_j = 1$$

$$\sum_{j=1}^{n} r_j x_j = r_0$$

- Formulation #2 is better than the #1. The evaluation of the objective function in #1 requires $2n^2/2$ multiplications. In #2 only $n + n^2/2$
- Formulation #3 has essentially the same number of multiplications but they appear in linear constraints. Number of constraints is bigger but all of them are linear. Linear algebra is much more efficient. Formulation #3 is the most efficient

Reformulation of an NLP problem

$$\min \frac{x+y}{\sum_i z_i}$$

$$\begin{aligned} &\min \frac{u}{v} \\ &u = x + y \\ &v = \sum_i z_i \\ &v \geq \varepsilon \end{aligned}$$

- Formulation #1 has a lot of nonlinear variables and it not protected against division by zero
- Formulation #2 has only 2 nonlinear variables, the remaining ones appear in linear equations, and the denominator is lower bounded to avoid division by zero. Model is easier to solve and more robust

Product of two variables $x_1 x_2$

$$x_1 x_2 = y_1^2 - y_2^2$$

Quadratic separable form

$$y_1 = (x_1 + x_2)/2$$
$$y_2 = (x_1 - x_2)/2$$
$$l_1 \leq x_1 \leq u_1$$
$$l_2 \leq x_2 \leq u_2$$
$$\frac{1}{2}(l_1 + l_2) \leq y_1 \leq \frac{1}{2}(u_1 + u_2)$$
$$\frac{1}{2}(l_1 - u_2) \leq y_2 \leq \frac{1}{2}(u_1 - l_2)$$

AIMMS Modeling Guide - Integer Programming Tricks

Solving real problems

- MIP
  - Solve with a sensible relative optimality tolerance
  - Provide an initial solution based on specific knowledge of the model or use the solution from a previous solve
- NLP
  - Introduce sensible bounds on variables AND
  - Provide a good enough starting point AND
  - Scale the problem
- Avoid to introduce symmetry (totally equal decision variables)
  - Symmetry-breaking constraint $x_i \geq x_{i+1}$

CPLEX Performance Tuning for MIP
(http://www-01.ibm.com/support/docview.wss?uid=swg21400023)

- Names no
- NodeFileInd 3
- **NodeSel** 0
- **VarSel** 3
- **StartAlg** 4
- MemoryEmphasis 1
- WorkMem 1000
- **MIPEmphasis 2**
- MIPSearch 2
- SolveFinal 0
- Solution Polishing
- **Solution pool**
- FlowCovers
- FeasOptMode 2
- FeasOpt 1
- **tuning cplex.opt**
- **RINSHeur 100**
- FpHeur 2

Pure branch and bound

- Cuts -1
- HeurFreq -1

Presolve

- PreInd, PrePass

- Solution method of LP problem
  - ✓ First iteration (interior point or simplex method)
  - ✓ Successive iterations (primal or dual simplex)
- Priority for variable selection
  - ✓ Select variables that impact the most in the o.f. (e.g., investment vs. operation variables)
- Initial cutoff or incumbent
  - ✓ Initial valid bound of the o.f. estimated by the user

GUROBI

- Most important parameters
  - Threads, MIPFocus
- Solution Improvement
  - ImproveStartTime, ImproveStartGap
- Termination
  - TimeLimit
  - MIPGap, MIPGapAbs
  - NodeLimit, IterationLimit, SolutionLimit
  - Cutoff
- Speeding Up The Root Relaxation
  - Method

- ❑ Numerical issues
  - ✓ **Presolve, PrePasses, Aggregate, AggFill, PreSparsify, PreDual, PreDepRow**
  - ✓ **NumericFocus**
- ❑ Heuristics
  - ✓ Heuristics, SubMIPNodes, MinRelNodes, PumpPasses, ZeroObjNodes
  - ✓ **RINS 100**
- ❑ Cutting Planes
  - ✓ Cuts, GomoryPasses, FlowCoverCuts, MIRCuts

http://www.gurobi.com/documentation/6.5/refman/mip_models.html
files.gurobi.com/Numerics.pdf

1. Programming Style
2. GAMS Code
3. Look and Feel
4. Mathematical Formulation
5. Advanced Algorithms

5

Advanced Algorithms

Transportation problem solved as MCP

```
sets
   I origins      / VIGO, ALGECIRAS /
   J destinations / MADRID, BARCELONA, VALENCIA /

parameters
   pA(i) origin capacity
       / VIGO      350
         ALGECIRAS 700 /

   pB(j) destination demand
       / MADRID    400
         BARCELONA 450
         VALENCIA  150 /

table pC(i,j) per unit transportation cost
          MADRID BARCELONA VALENCIA
VIGO       0.06      0.12      0.09
ALGECIRAS  0.05      0.15      0.11

variables
   vX(i,j) units transported
   vCost   transportation cost

positive variable vX

equations
   eCost        transportation cost
   eCapacity(i) maximum capacity of each origin
   eDemand  (j) demand supply at destination ;

eCost        .. sum[(i,j), pC(i,j) * vX(i,j)] =e= vCost ;
eCapacity(i) .. sum[   j ,           vX(i,j)] =l= pA(i) ;
eDemand  (j) .. sum[ i   ,           vX(i,j)] =g= pB(j) ;

model mTransport / all /
solve mTransport using LP minimizing vCost
```

$$\min_x \sum_{ij} c_{ij} x_{ij}$$

$$\sum_j x_{ij} \le a_i \quad \forall i$$

$$\sum_i x_{ij} \ge b_j \quad \forall j$$

$$x_{ij} \ge 0$$

$$\mathcal{L} = \sum_{ij} c_{ij} x_{ij} + \alpha_i \left(\sum_j x_{ij} - a_i \right) + \beta_j \left(b_j - \sum_i x_{ij} \right)$$

$$\frac{\partial L}{\partial x_{ij}} \rightarrow$$

$$c_{ij} + \alpha_i \ge \beta_j \quad : x_{ij} \quad \forall ij$$

$$-\sum_j x_{ij} \ge -a_i \quad : \alpha_i \quad \forall i$$

$$\sum_i x_{ij} \ge b_j \quad : \beta_j \quad \forall j$$

$$x_{ij}, \alpha_i, \beta_j \ge 0$$

```
sets
   I origins      / VIGO, ALGECIRAS /
   J destinations / MADRID, BARCELONA, VALENCIA /

parameters
   pA(i) origin capacity
       / VIGO      350
         ALGECIRAS 700 /

   pB(j) destination demand
       / MADRID    400
         BARCELONA 450
         VALENCIA  150 /

table pC(i,j) per unit transportation cost
          MADRID BARCELONA VALENCIA
VIGO       0.06      0.12      0.09
ALGECIRAS  0.05      0.15      0.11

variables
   vX(i,j) units transported
   vA(i  ) Lagrange multiplier of capacity constraint
   vB(  j) Lagrange multiplier of demand   constraint

positive variables vX, vA, vB

equations
   eProfit(i,j) marginal cost >= marginal profit
   eCapacity(i) maximum capacity of each origin
   eDemand  (j) demand supply at destination ;

eProfit(i,j) ..  vA(i) + pC(i,j)  =g=  vB(j) ;
eCapacity(i) .. -sum[j,  vX(i,j)] =g= -pA(i) ;
eDemand  (j) ..  sum[i,  vX(i,j)] =g=  pB(j) ;

model mTransport / eProfit.vX eCapacity.vA eDemand.vB /
solve mTransport using MCP
```

Fixed-Charge Transportation Problem (FCTP)

Flows (second stage)

Investment decisions (first stage)

Capacity of each origin

Demand of each destination

Flow can pass only for installed connections

Complete problem

$$\min_{x_{ij},y_{ij}} \sum_{ij} (f_{ij}y_{ij} + c_{ij}x_{ij})$$

$$\sum_j x_{ij} \le a_i \quad \forall i$$

$$\sum_i x_{ij} \ge b_j \quad \forall j$$

$$x_{ij} \le M_{ij}y_{ij} \quad \forall ij$$

$$x_{ij} \ge 0, y_{ij} \in \{0,1\}$$

- Bd Relaxed Master

$$\min_{y_{ij},\theta} \theta + \sum_{ij} (f_{ij}y_{ij})$$

$$\delta^l\theta - \theta^l \ge \sum_{ij} \pi_{ij}^l M_{ij}(y_{ij}^l - y_{ij}) \quad l = 1, \dots, k$$

$$y_{ij} \in \{0,1\}$$

O.F. of the subproblem at iteration l

Dual variables of linking constraints at iteration l

Master proposal at iteration l

- Bd Subproblem

$$\min_{x_{ij}} \sum_{ij} (c_{ij}x_{ij})$$

$$\sum_j x_{ij} \le a_i \quad \forall i$$

$$\sum_i x_{ij} \ge b_j \quad \forall j$$

$$x_{ij} \le M_{ij}y_{ij}^k \quad \forall ij \quad : \pi_{ij}^k$$

$$x_{ij} \ge 0$$

FCTP solved by Benders decomposition (i)

```
$Title Fixed-charge transportation problem (FCTP) solved by Benders decomposition

* relative optimality tolerance in solving MIP problems
option OptcR = 0

sets
   L          iterations        / l1 * l20 /
   LL(l)      iterations subset
   I          origins           / i1 * i4  /
   J          destinations      / j1 * j3  /

* Begin problem data

parameters
   A(i)       product offer
              / i1 10, i2 30, i3 40, i4 20 /
   B(j)       product demand
              / j1 20, j2 50, j3 30 /

table C(i,j) per unit variable transportation cost
       j1  j2  j3
   i1   1   2   3
   i2   3   2   1
   i3   2   3   4
   i4   4   3   2

table F(i,j) fixed transportation cost
       j1  j2  j3
   i1  10  20  30
   i2  20  30  40
   i3  30  40  50
   i4  40  50  60

* End problem data

abort $(sum[i, A(i)] < sum[j, B(j)]) 'Infeasible problem'

parameters
   BdTol         relative Benders tolerance / 1e-6 /
   Z_Lower       lower bound         / -inf /
   Z_Upper       upper bound         /  inf /
   Y_L  (l,i,j) first stage variables values              in iteration l
   PI_L (l,i,j) dual variables of second stage constraints in iteration l
   Delta(l)      cut type (feasibility 0 optimality 1)    in iteration l
   Z2_L (l)      subproblem objective function value      in iteration l

positive variable
   X(i,j)        arc flow

binary   variable
   Y(i,j)        arc investment decision

variables
   Z1            first  stage objective function
   Z2            second stage objective function
   Theta         recourse function
```

FCTP solved by Benders decomposition (ii)

```
equations
   EQ_Z1          first  stage     objective function
   EQ_Z2          second stage     objective function
   EQ_OBJ         complete problem objective function
   Offer    (i  ) offer  at origin
   Demand   (  j) demand at destination
   FlowLimit(i,j) arc flow limit
   Bd_Cuts    (l) Benders cuts ;

EQ_Z1          .. Z1 =e= sum[(i,j), F(i,j)*Y(i,j)] + Theta ;

EQ_Z2          .. Z2 =e=                          sum[(i,j), C(i,j)*X(i,j)] ;

EQ_OBJ         .. Z1 =e= sum[(i,j), F(i,j)*Y(i,j)] + sum[(i,j), C(i,j)*X(i,j)] ;

Offer    (i  ) .. sum[j, X(i,j)] =l= A(i) ;

Demand   (  j) .. sum[i, X(i,j)] =g= B(j) ;

FlowLimit(i,j) .. X(i,j) =l= min[A(i),B(j)] * Y(i,j) ;

Bd_Cuts(ll)    .. Delta(ll) * Theta =g= Z2_L(ll) -
                  sum[(i,j), PI_L(ll,i,j) * min[A(i),B(j)] * (Y_L(ll,i,j) - Y(i,j))] ;

model Master_Bd     / EQ_Z1  Bd_Cuts               /
model Subproblem_Bd / EQ_Z2  Offer Demand FlowLimit /
model Complete      / EQ_OBJ Offer Demand FlowLimit / ;

X.up(i,j) = min[A(i),B(j)]

* to allow CPLEX correctly detect rays in an infeasible problem
* only simplex method can be used and no preprocessing neither scaling options
* optimality and feasibility tolerances are very small to avoid primal degeneration

file COPT / cplex.opt /
put  COPT putclose 'ScaInd -1' / 'LPMethod 1' / 'PreInd 0' / 'EpOpt 1e-9' / 'EpRHS 1e-9' / ;

Subproblem_Bd.OptFile = 1 ;
```

FCTP solved by Benders decomposition (iii)

```
* parameter initialization

LL      (l) = no ;
Theta.fx    =  0 ;
Delta   (l) =  0 ;
Z2_L    (l) =  0 ;
PI_L(l,i,j) =  0 ;
Y_L (l,i,j) =  0 ;

* Benders algorithm iterations
loop (l $(abs(1-Z_Lower/Z_Upper) > BdTol),

*  solving master problem
   solve Master_Bd using MIP minimizing Z1 ;

*  storing the master solution
   Y_L(l,i,j) = Y.l(i,j) ;

*  fixing first-stage variables and solving subproblem
   Y.fx( i,j) = Y.l(i,j) ;

*  solving subproblem
   solve Subproblem_Bd using RMIP minimizing Z2 ;

*  storing parameters to build a new Benders cut
   if (Subproblem_Bd.ModelStat = 4,
      Delta(l) = 0 ;
      Z2_L (l) = Subproblem_Bd.SumInfes ;
   else
*     updating Lower and upper bound
      Z_Lower =            Z1.l ;
      Z_Upper = min(Z_Upper, Z1.l - Theta.l + Z2.l) ;

      Theta.lo = -inf ;
      Theta.up =  inf ;

      Delta(l) =    1 ;
      Z2_L (l) = Subproblem_Bd.ObjVal ;
   ) ;

   PI_L(l,i,j) = FlowLimit.m(i,j) ;

   Y.lo(  i,j) = 0 ;
   Y.up(  i,j) = 1 ;

*  increase the set of Benders cuts
   LL(l) = yes ;
) ;

solve Complete using MIP minimizing Z1
```

Stochastic FCTP

Flows
(second stage)

Investment decisions
(first stage)

Capacity of each origin

Demand of each destination

Flow can go only for installed connections

Complete problem

$$\min_{x_{ij}^{\omega}, y_{ij}} \sum_{ij} \left(f_{ij} y_{ij} + \sum_{\omega} p_{\omega} c_{ij} x_{ij}^{\omega} \right)$$

$$\sum_{j} x_{ij}^{\omega} \leq a_i \quad \forall i\omega$$

$$\sum_{i} x_{ij}^{\omega} \geq b_j^{\omega} \quad \forall j\omega$$

$$x_{ij}^{\omega} \leq M_{ij}^{\omega} y_{ij} \quad \forall ij\omega$$

$$x_{ij}^{\omega} \geq 0, y_{ij} \in \{0,1\}$$

Deterministic & Stochastic FCTP

```
$Title Deterministic fixed-charge transportation problem (DFCTP)

* relative optimality tolerance in solving MIP problems
option OptcR = 0

sets
   I         origins           / i1 * i4 /
   J         destinations      / j1 * j3 /


parameters
   A(i)      product offer  / i1 20, i2 30, i3 40, i4 20 /
   B(j)      product demand / j1 20, j2 50, j3 30 /




table C(i,j) per unit variable transportation cost
       j1  j2  j3
   i1   1   2   3
   i2   3   2   1
   i3   2   3   4
   i4   4   3   2

table F(i,j) fixed transportation cost
       j1  j2  j3
   i1  10  20  30
   i2  20  30  40
   i3  30  40  50
   i4  40  50  60

abort $(sum[i, A(i)] < sum[j, B(j)]) 'Infeasible problem'

positive variable
   X(i,j)       arc flow
binary   variable
   Y(i,j)       arc investment decision
variables
   Z1           objective function

equations
   EQ_OBJ          complete problem objective function
   Offer    (i  ) offer  at origin
   Demand   (  j) demand at destination
   FlowLimit(i,j) arc flow limit ;

EQ_OBJ          .. Z1 =e= sum[(i,j), F(i,j)*Y(i,j)] + sum[(i,j), C(i,j)*X(i,j)] ;
Offer    (i  ) .. sum[j, X(i,j)] =l= A(i) ;
Demand   (  j) .. sum[i, X(i,j)] =g= B(j) ;
FlowLimit(i,j) .. X(i,j) =l= min[A(i),B(j)] * Y(i,j) ;

model Complete      / EQ_OBJ Offer Demand FlowLimit / ;

X.up(i,j) = min[A(i),B(j)]

solve Complete using MIP minimizing Z1
```

```
$Title Stochastic fixed-charge transportation problem (SFCTP)

* relative optimality tolerance in solving MIP problems
option OptcR = 0

sets
   I         origins           / i1 * i4 /
   J         destinations      / j1 * j3 /
   S         scenarios         / s1 * s3 /

parameters
   A(i)      product offer        / i1 20, i2 30, i3 40, i4 20 /
   P(s)      scenario probability / s1 0.5, s2 0.3, s3 0.2 /

table B(s,j) product demand
       j1  j2  j3
   s1  21  51  31
   s2  32  22  52
   s3  53  33  23

table C(i,j) per unit variable transportation cost
       j1  j2  j3
   i1   1   2   3
   i2   3   2   1
   i3   2   3   4
   i4   4   3   2

table F(i,j) fixed transportation cost
       j1  j2  j3
   i1  10  20  30
   i2  20  30  40
   i3  30  40  50
   i4  40  50  60

loop (s, abort $(sum[i, A(i)] < sum[j, B(s,j)]) 'Infeasible problem' )

positive variable
   X(s,i,j)     arc flow
binary   variable
   Y(i,j  )     arc investment decision
variables
   Z1           objective function

equations
   EQ_OBJ          complete problem objective function
   Offer    (s,i  ) offer  at origin
   Demand   (s,  j) demand at destination
   FlowLimit(s,i,j) arc flow limit ;

EQ_OBJ          .. Z1 =e= sum[(i,j), F(i,j)*Y(i,j)] + sum[(s,i,j), P(s)*C(i,j)*X(s,i,j)] ;
Offer    (s,i  ) .. sum[j, X(s,i,j)]  =l= A(  i) ;
Demand   (s,  j) .. sum[i, X(s,i,j)]  =g= B(s,j) ;
FlowLimit(s,i,j) .. X(s,i,j) =l= min[A(i),B(s,j)] * Y(i,j) ;

model Complete      / EQ_OBJ Offer Demand FlowLimit / ;

X.up(s,i,j) = min[A(i),B(s,j)]

solve Complete using MIP minimizing Z1
```

Stochastic FCTP with EMP

```
$title Deterministic fixed-charge transportation problem (FCTP)

* relative optimality tolerance in solving MIP problems
option OptcR = 0

sets
   I         origins          / i1 * i4 /
   J         destinations     / j1 * j3 /

parameters
   A(i)      product offer
             / i1 20, i2 30, i3 40, i4 20 /
   B(j)      product demand
             / j1 11, j2 44, j3 66 /

table C(i,j) per unit variable transportation cost
       j1  j2  j3
   i1   1   2   3
   i2   3   2   1
   i3   2   3   4
   i4   4   3   2

table F(i,j) fixed transportation cost
       j1  j2  j3
   i1  10  20  30
   i2  20  30  40
   i3  30  40  50
   i4  40  50  60

positive variable
   X(i,j)       arc flow

binary   variable
   Y(i,j)       arc investment decision

variables
   Z1           objective function

equations
   EQ_OBJ         complete problem objective function
   Offer    (i  ) offer  at origin
   Demand   (  j) demand at destination
   FlowLimit(i,j) arc flow limit ;

EQ_OBJ         .. Z1 =e= sum[(i,j), F(i,j)*Y(i,j)] + sum[(i,j), C(i,j)*X(i,j)] ;
Offer    (i  ) .. sum[j, X(i,j)] =l= A(i) ;
Demand   (  j) .. sum[i, X(i,j)] =g= B(j) ;
FlowLimit(i,j) .. X(i,j) =l= 100 * Y(i,j) ;

model Complete      / all / ;

X.up(i,j) = 100
```

```
set         S       scenarios            / s1 * s3 /
parameter P(s)      scenario probability / s1 0.5, s2 0.3, s3 0.2 /
          YS(s,i,j) arc investment decision
          XS(s,i,j) arc flow

table  BS(s,j) product demand
       j1  j2  j3
   s1  21  51  31
   s2  32  22  52
   s3  53  33  23

file     emp / '%emp.info%' / ; emp.pc=2
put      emp
put '* problem %gams.i%' / "jrandvar "
loop (j,
   put B.tn(j) " "
)
loop (s,
   put P(s)
   loop (j,
      put BS(s,j)
   )
)
put / "stage 2 B X Offer Demand FlowLimit"
putclose emp

set dict / s . scenario . ''
           B . randvar  . BS
           X . level    . XS
           Y . level    . YS /

loop (s, abort $(sum[i, A(i)] < sum[j, BS(s,j)]) 'Infeasible problem' ) ;

solve Complete minimizing Z1 using emp scenario dict

display XS, YS
```

Cournot model: formulation

- Objective function: profits

$$B_e = p \cdot q_e - C_e$$

- Inverse demand function

$$p = f\left(\sum q_e\right)$$

- Cournot conjecture: vertical supply

$$\frac{\partial p}{\partial q_e} = \frac{\partial p}{\partial q}\frac{\partial (q_e + q_{e*})}{\partial q_e} = p' \leftrightarrow \frac{\partial q_{e*}}{\partial q_e} = 0$$

- Optimality conditions

$$\frac{\partial B_e}{\partial q_e} = 0 \rightarrow MR_e = p + q_e p' = MC_e\left(q_e\right) \rightarrow q_e = \frac{p - MC_e\left(q_e\right)}{-p'}$$

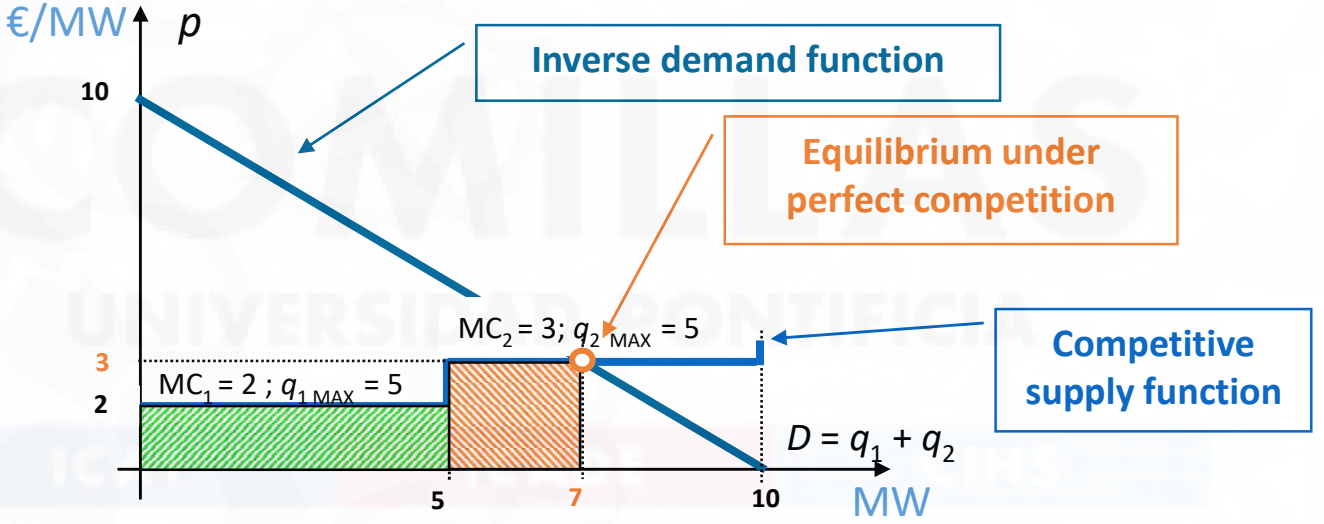
e	Company
e^*	Other companies
B_e	Profits
p	Price
q_e	Output
C_e	Variable costs
MC_e	Marginal cost
MR_e	Marginal revenue
$p\text{-}MC_e$	**Price mark-up**

System of equations

Own output decision will not have an effect on the decisions of the competitors

Cournot model: example (I)

- Perfect competition
  - Company 1: $MC_1 = 2$ €/MW $q_{1\,MAX} = 5$ MW
  - Company 2: $MC_2 = 3$ €/MW $q_{2\,MAX} = 5$ MW
  - Inverse demand function: $p = 10 - (q_1 + q_2)$



Cournot model: example (II)

- Duopoly
  - Company 1: $MC_1 = 2$ €/MW
  - Company 2: $MC_2 = 3$ €/MW
  - IDF: $p = 10 - (q_1 + q_2)$; $p' = -1$

Cournot

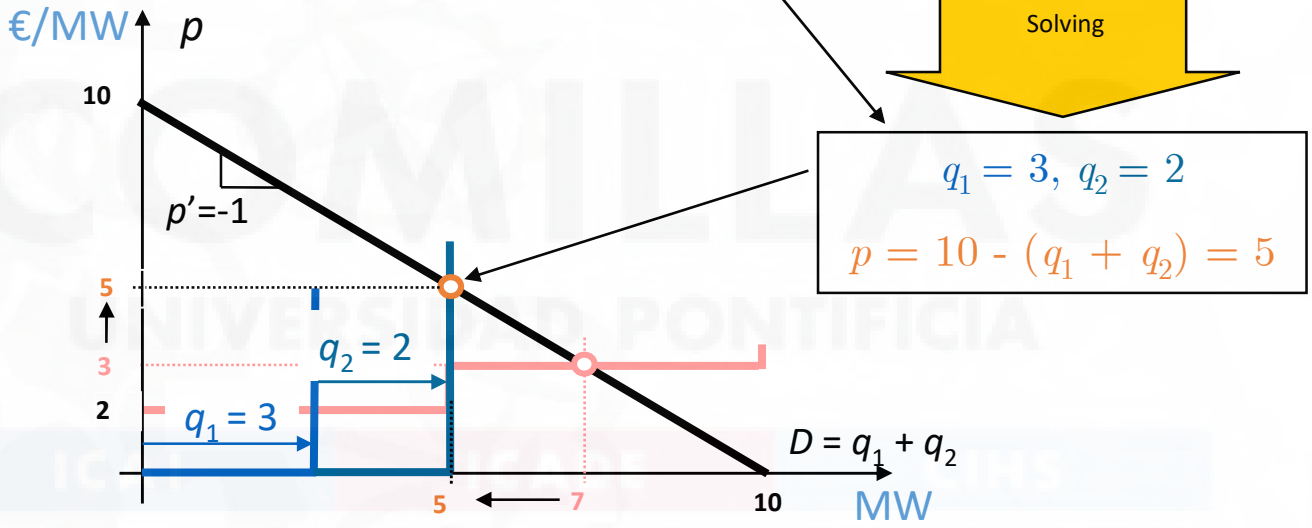
$$\frac{\partial B_1}{\partial q_1} = 0 \rightarrow p + q_1 \cdot (-1) = 2$$

$$\frac{\partial B_2}{\partial q_2} = 0 \rightarrow p + q_2 \cdot (-1) = 3$$

Solving

$$q_1 = 3,\ q_2 = 2$$

$$p = 10 - (q_1 + q_2) = 5$$



Cournot Market Equilibrium

```
$Title Cournot Market Equilibrium Model

$OnText

Developed by

   Andrés Ramos
   Instituto de Investigación Tecnológica
   Escuela Técnica Superior de Ingeniería - ICAI
   UNIVERSIDAD PONTIFICIA COMILLAS
   Alberto Aguilera 23
   28015 Madrid, Spain
   Andres.Ramos@comillas.edu

   August 5, 2016

$OffText

sets
   p periods       / p1      /
   t thermal units / t1 * t2 /

parameters
   pMarginalCost(t) marginal costs                    [€ per MW] / t1 2, t2 3 /
   pMaxThrOutput(t) maximum output                        [MW] / t1 5, t2 5 /

   pPriceInter  (p) price intercept for each period [€ per MW  ] / p1 10      /
   pPriceSlope  (p) price     slope for each period [€ per MW^2] / p1 -1      /

positive variables
   vSMP      (  p) system marginal price [€ per MW]
   vThrOutput(t,p) thermal unit output         [MW]

binary   variables
   vCommitment(t,p) thermal unit commitment    [p.u.]
```

Cournot Market Equilibrium

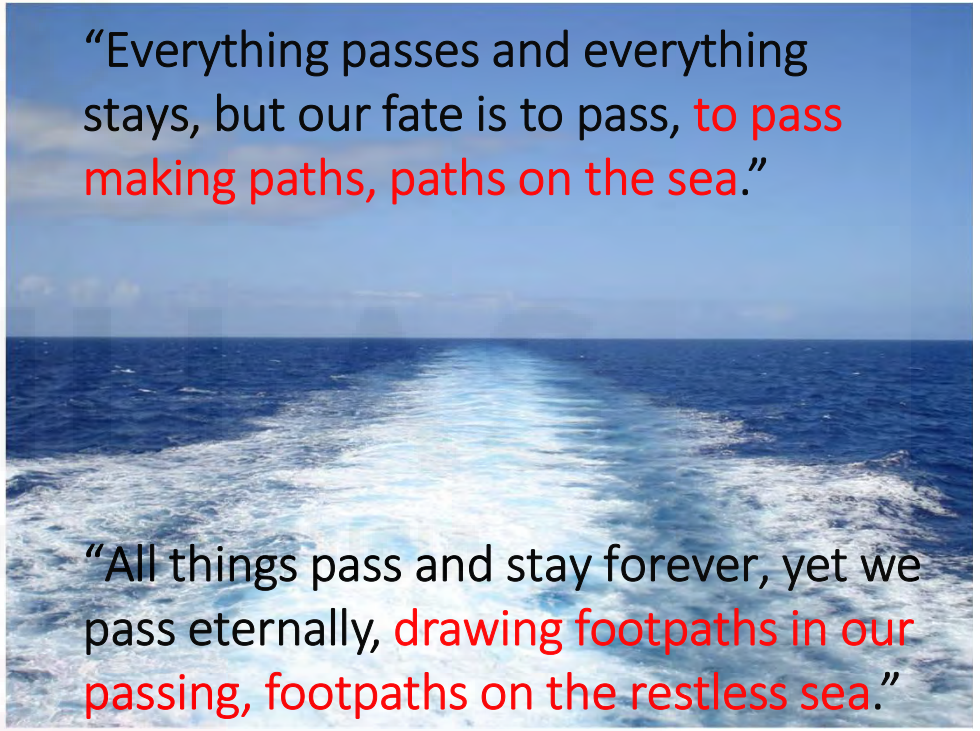
```
equations
   eDemandFunction(  p) price as a function of the load [€ per MW]
   eDerivRevenue  (t,p) derivative of the net revenue   [€ per MW] ;

eDemandFunction(  p) .. vSMP(p) =g= pPriceInter(p)   + pPriceSlope(p)*sum[t, vThrOutput(t,p)] ;

eDerivRevenue  (t,p) .. pMarginalCost(t) =g= vSMP(p) + pPriceSlope(p)*       vThrOutput(t,p)  ;

model mCournot / eDemandFunction.vSMP eDerivRevenue.vThrOutput /

variable
   vTotalCost      total investment and operation cost

equations
   eTotalCost      total investment and operation cost [€]
   eLoadBalance(p) balance between generation and load [€ per MW] ;

eTotalCost      .. vTotalCost =e= - sum[(t,p), pMarginalCost(t)*vThrOutput(t,p)] ;

eLoadBalance(p) .. pMarginalCost('t2') =e= pPriceInter(p) + pPriceSlope(p)*sum[t, vThrOutput(t,p)] ;

model mPerfectCompetition / eTotalCost eLoadBalance /

variable
   vSocialWelfare  social welfare [€]

equations
   eSocialWelfare  social welfare [€] ;

eSocialWelfare .. vSocialWelfare =e= sum[p, (pPriceInter(p) + pPriceSlope(p)*sum[t, vThrOutput(t,p)]/2)*sum[t, vThrOutput(t,p)]]
                                                                                - sum[(t,p), pMarginalCost(t)*vThrOutput(t,p)] ;

model mSocialWelfare / eSocialWelfare / ;

vThrOutput.up(t,p) = pMaxThrOutput(t) ;

* Cournot equilibrium model

solve mCournot using MCP ;

* perfect competition model (cost minimization)

solve mPerfectCompetition using MIP maximizing vTotalCost

* perfect competition model (social welfare maximization)

solve mSocialWelfare using QCP maximizing vSocialWelfare
```

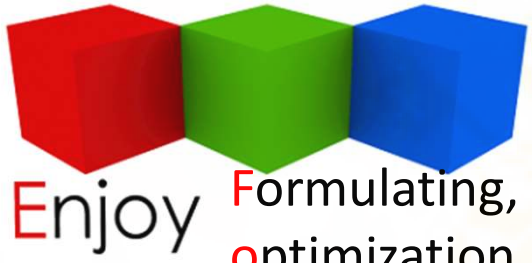
UC model competition

- Formulate mathematically the minimum up and down time of a thermal unit
- Introduce the constraints in the UC model
- Compare the efficiency of the formulation for a large-scale model
  - Tuning the solver parameters
  - Stronger formulation of the constraints

Antonio Machado. Cantares

"Todo pasa y todo queda,
pero lo nuestro es pasar,
pasar haciendo caminos,
caminos sobre el mar."





Formulating, writing and solving optimization models

Thank you for your attention

Prof. Andres Ramos

https://www.iit.comillas.edu/aramos/

Andres.Ramos@comillas.edu