

# TÉCNICAS DE DESCOMPOSICIÓN EN PROGRAMACIÓN MATEMÁTICA

Dr. Víctor M. Albornoz  
Departamento de Industrias.  
Campus Santiago Vitacura, Chile.  
Universidad Técnica Federico Santa María

Instituto de Computación, Facultad de Ingeniería, UdelaR.  
Montevideo, lunes 13 de Mayo de 2024

La Investigación de Operaciones es una disciplina que aplica métodos analíticos avanzados para contribuir a la toma de buenas decisiones.

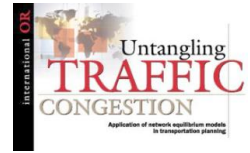
**OPERATIONS RESEARCH: THE SCIENCE OF BETTER**

TIME-STARVED EXECUTIVES ARE MAKING BOLDER DECISIONS WITH LESS RISK AND BETTER OUTCOMES. THEIR SECRET: OPERATIONS RESEARCH.

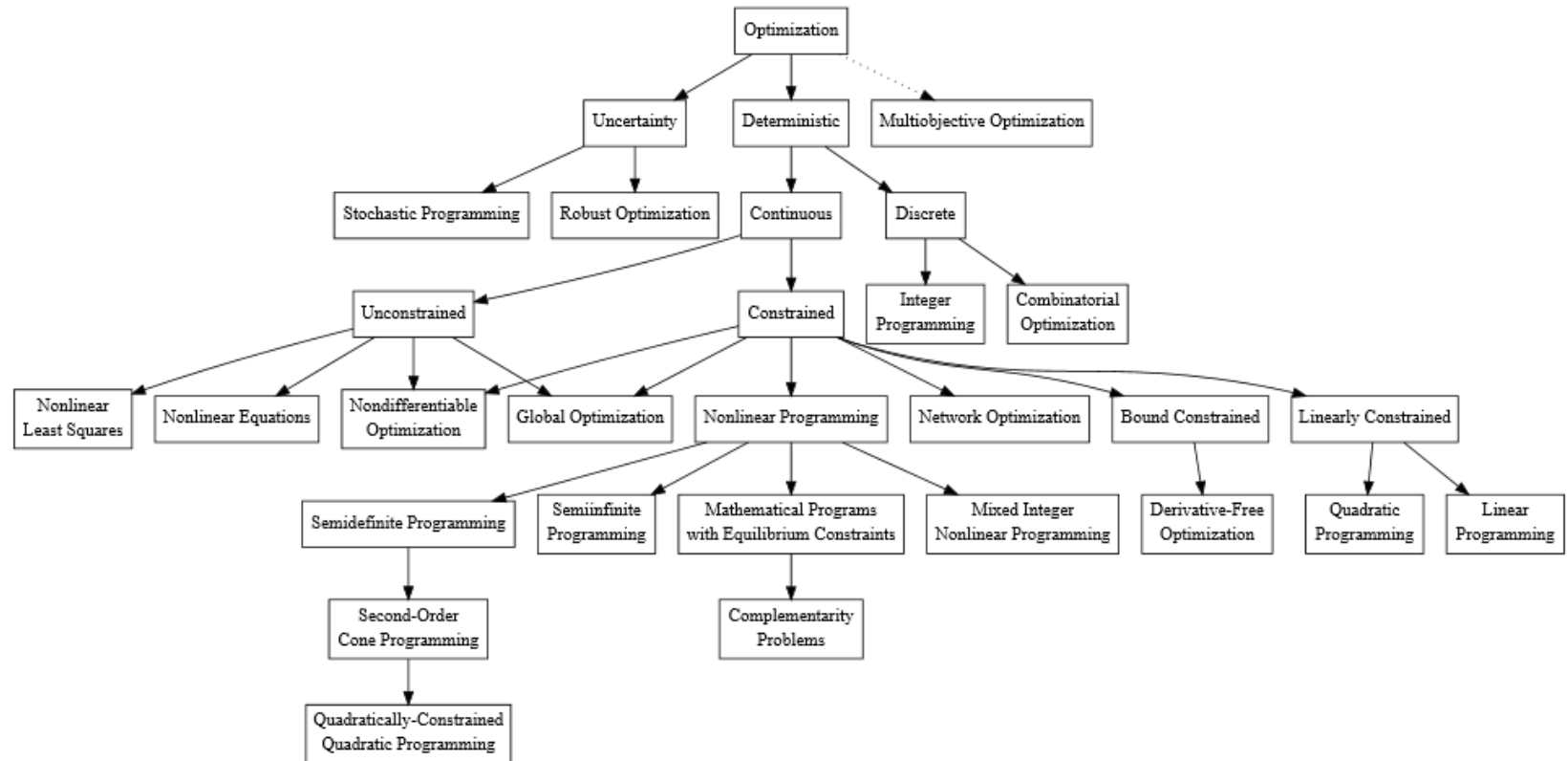
Entre sus ámbitos de aplicación destacan:

- *gestión de inventarios,*
- *planificación y programación de la producción,*
- *gestión de cadenas de suministro,*
- *diseño y operación de sistemas energéticos,*
- *localización óptima de instalaciones,*
- *problemas de distribución y ruteo vehicular,*
- *gestión de recursos naturales.*
- *gestión en sistemas de salud,*
- *etc.*

Ante problemas de naturaleza real, esta disciplina contempla metodologías para generar decisiones admisibles o encontrar la mejor solución a un problema (modelos generativos) o simplemente para evaluar posibles conjuntos de soluciones (modelos evaluativos).



Entre los primeros destacan especialmente los *modelos de optimización*, los cuales son muy variados dada la naturaleza de las decisiones y el problema abordado



Al formular un modelo de optimización uno comúnmente expresa de manera algebraica la función objetivo y las distintas ecuaciones e inecuaciones que definen las restricciones del problema en términos de sus variables de decisión.

**Set**

$L$  = the set of locations

**Parameters**

$x_i$  = the x-coordinate for location  $i$ ,  $\forall i \in L$

$y_i$  = the y-coordinate for the location  $i$ ,  $\forall i \in L$

$d_i$  = projected demand for the next period for location  $i$ ,  $\forall i \in L$

$s_i$  = number of helicopters currently assigned to location  $i$ ,  $\forall i \in L$

$c$  = transportation cost per kilometer

$dist_{ij}$  = Euclidean distance between location  $i$  and location  $j$ ,  $\forall i \in L \forall j \in L$

$$dist_{ij} = \sqrt{(x_j - x_i)^2 + (y_j - y_i)^2}$$

**Variables**

$z_{ij}$  = number of helicopters to be moved from location  $i$  to  $j$ ,  $\forall i \in L \forall j \in L$

**Objective Function**

Minimize  $\sum_{(i,j) \in L \times L} dist_{ij} * c * z_{ij}$

**Constraints**

Flow balance constraint for each location  $i$  in set  $L$

$$\sum_{j \in L} z_{ji} + s_i = d_i + \sum_{j \in L} z_{ij}, \forall i \in L$$



Complementa lo anterior la existencia de *lenguajes de modelado algebraico* que permiten emplear la notación común para la representación de modelos y una sintaxis simple en el desarrollo de algoritmos.

*Ventajas:* resulta fácil leer datos, comunicarse con un solver determinado, examinar y exportar la solución de un problema y programar un algoritmo. *Desventaja:* programas de alto nivel.



La lista de programas de modelado algebraico incluye:

AIMMS

AMPL

GAMS

JuMP

LINGO

MPL

OPL Studio

PLAM

Pyomo



Estos programas deben ser utilizados conjuntamente con un solver de acuerdo a la naturaleza del modelo a resolver, por ejemplo: cplex, gurobi, knitro, minos y xpress.

Sin embargo, en numerosas situaciones estos pueden ser insuficientes ante problemas complejos y de gran tamaño.

## HIPÓTESIS DEL CURSO.

- *Los Métodos de Descomposición proveen una técnica numérica de optimización que ha resultado apropiada ante problemas complejos.*

## OBJETIVO DEL CURSO.

- *Introducir y formar en el empleo de técnicas algorítmicas de descomposición para la resolución de problemas de programación matemática de alta complejidad y gran escala.*

# CONTENIDOS

0. Clase Bienvenida.

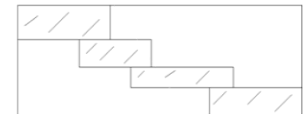
1. Introducción a los Métodos de Decomposición.

2. Formulación y resolución de modelos en AMPL.

3. Método de Benders.



4. Generación de Columnas.



5. Método de Dantzig & Wolfe.



6. Conclusiones, Extensiones y palabras finales.

## BIBLIOGRAFIA.

- Bazaraa et al. (1998).
- Bertsimas y Tsitsiklis (1997).
- Fourer et al. (2003).

## SISTEMA DE EVALUACIÓN.

- Actividades Complementarias (20%).
- Trabajo Final (80%).

## 2.- Formulación y resolución de modelos en AMPL

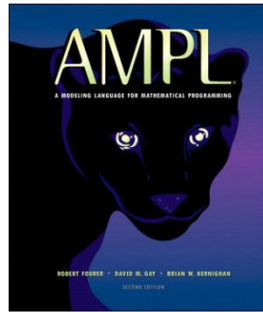
Los *lenguajes de modelado algebraico* existen desde fines de los años 70, creados pensando inicialmente en problemas de programación lineal.

El desarrollo de AMPL se inició en 1985.



Descarga versión estudiantil de AMPL en:  
<http://ampl.com/try-ampl/download-a-free-demo/>

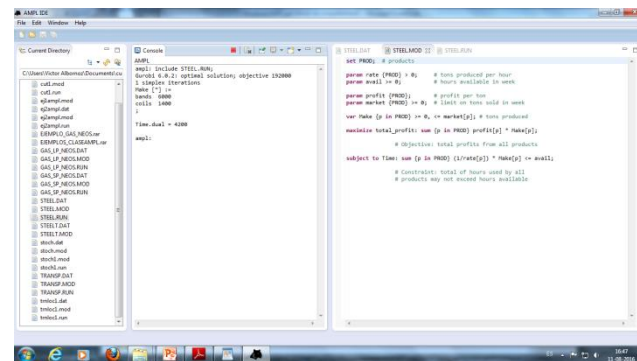
Manual AMPL



Descarga versión académica de AMPL en la  
página del curso en EVA.

Los modelos en AMPL permiten representar las variables de decisión, restricciones y la función objetivo expresados con la ayuda de conjuntos y parámetros, todos en términos algebraicos.

Toda la información contempla diferentes archivos con las extensiones: **.mod** (para el modelo), **.dat** (para los datos del modelo) y **.run** (para la ejecución del modelo).





Ejemplo 0. Una pequeña refinería emplea crudo 1 y crudo 2 para la elaboración de kerosene. El costo de un barril de crudo 1 es de 50 dólares y el de crudo 2 de 90 dólares.

La planta puede procesar hasta un máximo de 100 barriles de crudo por semana y tiene un rendimiento (promedio) que permite obtener 2 barriles de kerosene por cada barril de crudo 1 y 4 barriles de kerosene por cada barril de crudo 2. La próxima semana se tiene a modo de estimación que se podrá vender al menos 360 barriles de kerosene.

Un modelo que permite obtener un plan óptimo de compras de ambas materias primas considera:

Variables de decisión:

$x_1$  = barriles a comprar de crudo 1,

$x_2$  = barriles a comprar de crudo 2.

En tal caso el modelo resultante corresponde a:

$$\text{Min } 50x_1 + 90x_2$$

$$\text{s. a. } x_1 + x_2 \leq 100$$

$$2x_1 + 4x_2 \geq 360$$

$$x_1 \geq 0, x_2 \geq 0.$$

**Solución óptima:**  $x_1=0$ ,  $x_2=90$ . **Valor óptimo:**  $v(P)=8100$ .

En lo concerniente a la representación de un modelo en AMPL, este se declara en un archivo .mod, representando el mismo con los siguientes elementos y comandos:

Conjuntos de índices (set),

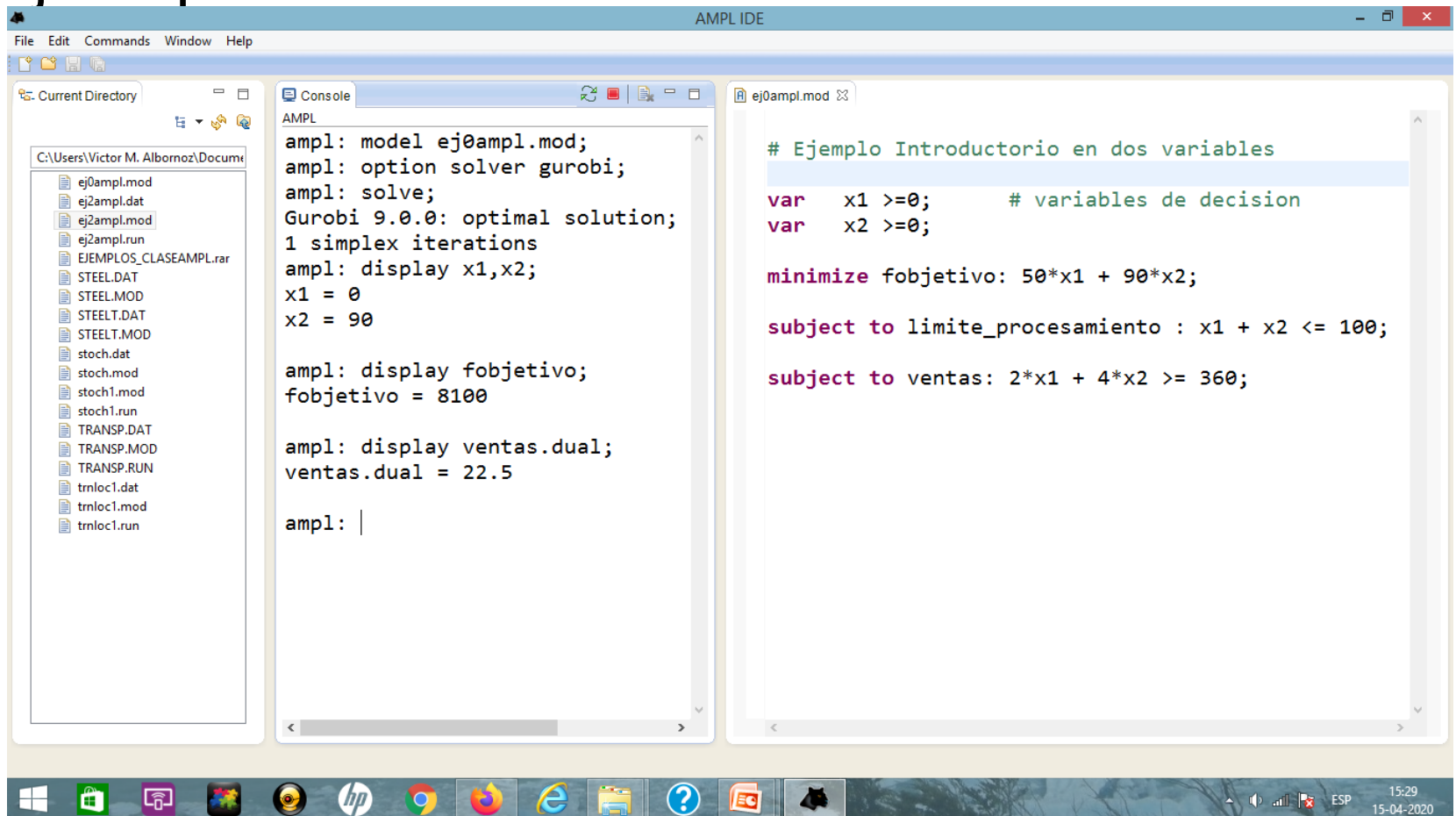
Parámetros (param),

Variables de decisión (var),

Función objetivo (minimize/maximize) y

Restricciones (subject to).

El modelo del ejemplo en AMPL puede representarse y resolverse empleando el archivo ej0ampl.mod.



The screenshot displays the AMPL IDE interface. On the left, the 'Current Directory' pane shows a file explorer view of the directory C:\Users\Victor M. Albornoz\Docume, containing files such as ej0ampl.mod, ej2ampl.dat, and ej2ampl.run. The central 'Console' pane shows the execution output for the AMPL model, including the command 'ampl: solve;', the solver output 'Gurobi 9.0.0: optimal solution; 1 simplex iterations', and the results 'fobjetivo = 8100' and 'ventas.dual = 22.5'. On the right, the 'ej0ampl.mod' file is open, showing the following AMPL code:

```
# Ejemplo Introdutorio en dos variables
var x1 >=0;      # variables de decision
var x2 >=0;

minimize fobjetivo: 50*x1 + 90*x2;

subject to limite_procesamiento : x1 + x2 <= 100;

subject to ventas: 2*x1 + 4*x2 >= 360;
```

Ejemplo 1. Considere el siguiente modelo de producción con múltiples productos:

Parámetros.

$u_p$  = beneficio por tonelada del producto  $p \in P$ .

$r_p$  = ton producidas del producto  $p \in P$  por hora.

$av$  = horas disponibles de producción.

$d_p$  = demanda máxima del producto  $p \in P$ .

Variable de decisión.

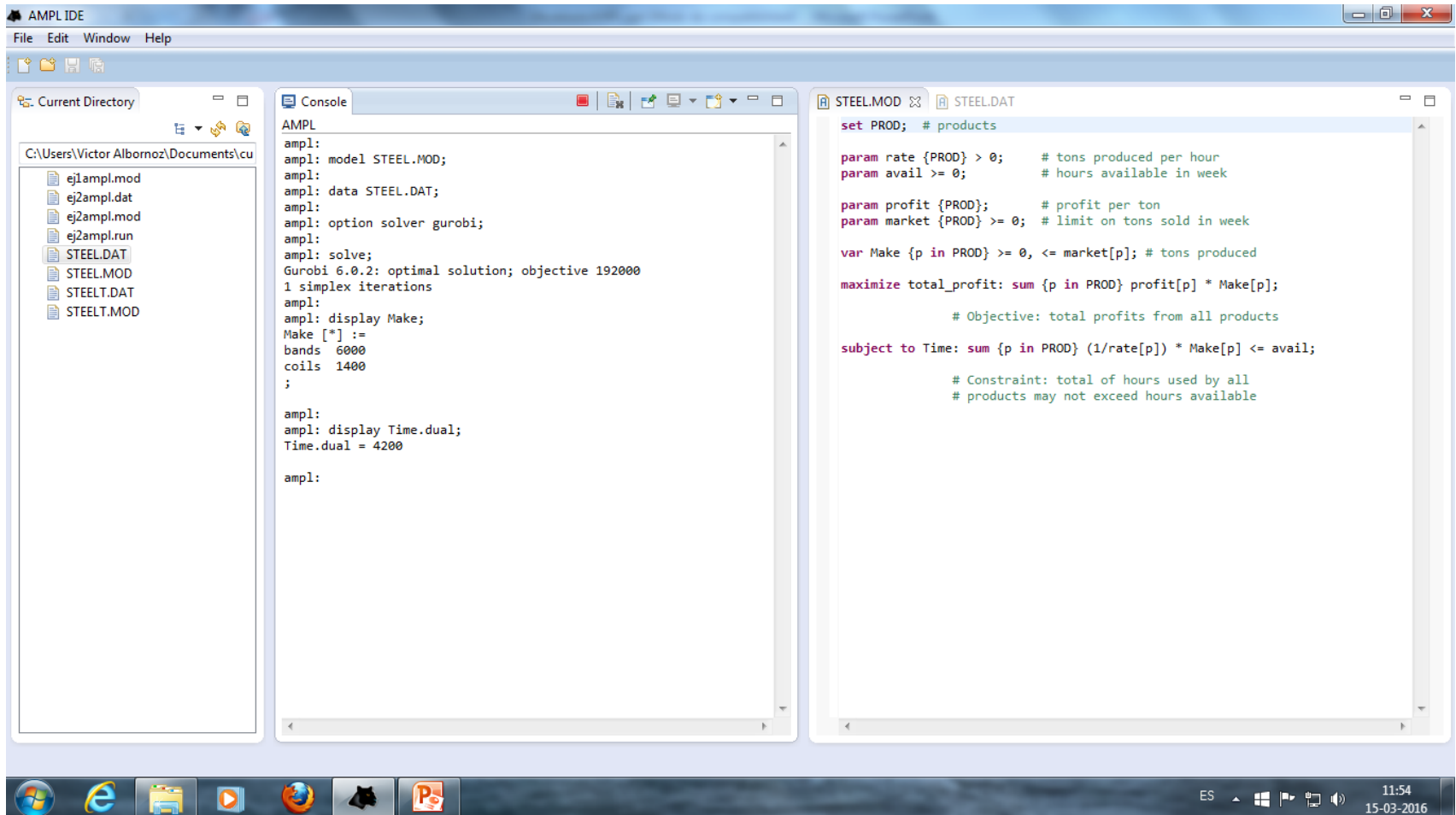
$X_p$  = toneladas elaboradas del producto  $p \in P$

$$\text{Max } \sum_{p \in P} u_p X_p$$

$$\text{s.a. } \sum_{p \in P} (1/r_p) X_p \leq av$$

$$0 \leq X_p \leq d_p \quad p \in P$$

# El modelo de producción en AMPL puede revisarse en los archivos STEEL.MOD y STEEL.DAT:



The screenshot shows the AMPL IDE interface. On the left, the 'Current Directory' pane shows the file structure: `C:\Users\Victor Alborno\Documents\cu` containing `ej1ampl.mod`, `ej2ampl.dat`, `ej2ampl.mod`, `ej2ampl.run`, `STEEL.DAT`, `STEEL.MOD`, `STEEL.DAT`, and `STEELT.MOD`.

The 'Console' pane shows the execution output:

```
AMPL
ampl:
ampl: model STEEL.MOD;
ampl:
ampl: data STEEL.DAT;
ampl:
ampl: option solver gurobi;
ampl:
ampl: solve;
Gurobi 6.0.2: optimal solution; objective 192000
1 simplex iterations
ampl:
ampl: display Make;
Make [*] :=
bands 6000
coils 1400
;

ampl:
ampl: display Time.dual;
Time.dual = 4200

ampl:
```

The 'STEEL.DAT' pane shows the data file content:

```
set PROD; # products

param rate {PROD} > 0; # tons produced per hour
param avail >= 0; # hours available in week

param profit {PROD}; # profit per ton
param market {PROD} >= 0; # limit on tons sold in week

var Make {p in PROD} >= 0, <= market[p]; # tons produced

maximize total_profit: sum {p in PROD} profit[p] * Make[p];

# Objective: total profits from all products

subject to Time: sum {p in PROD} (1/rate[p]) * Make[p] <= avail;

# Constraint: total of hours used by all
# products may not exceed hours available
```

The Windows taskbar at the bottom shows the system tray with the time 11:54 and date 15-03-2016.

Ejemplo 2. Consideramos un problema muy relevante a nivel táctico correspondiente al de *planificación agregada de la producción*.

Este problema consiste en hallar una política óptima de producción de un determinado conjunto de productos para satisfacer demandas fluctuantes en el tiempo, de modo de minimizar costos de producción e inventario, considerando la disponibilidad de diversos recursos escasos.

## Parámetros.

$c_{pt}$  = costo unitario de producción del producto  $p$  en periodo  $t$ .

$h_{pt}$  = costo unitario de inventario del producto  $p$  en periodo  $t$ .

$u_{pt}$  = beneficio por unidad del producto  $p$  en periodo  $t$ .

$d_{pt}$  = demanda máxima de unidades de producto  $p$  en  $t$ .

$r_p$  = unidades producidas del producto  $p$  por hora.

$av_t$  = horas disponibles de producción en periodo  $t$ .

$l_{p0}$  = inventario inicial del producto  $p$ .



Por su parte, las *variables de decisión* del modelo corresponden a:

$X_{pt}$  = unidades elaboradas del producto  $p$  en periodo  $t$

$I_{pt}$  = nivel de inventario del producto  $p$  al término de periodo  $t$

$S_{pt}$  = unidades vendidas del producto  $p$  en periodo  $t$

## *Modelo de Producción multi-producto con múltiples periodos*

$$\text{Max } \sum_p \sum_t (u_{pt} S_{pt} - c_{pt} X_{pt} - h_{pt} I_{pt})$$

s.a.

$$X_{pt} + I_{pt-1} = S_{pt} + I_{pt} \quad p \in P; t=1, \dots, T$$

$$\sum_{p \in P} (1/r_p) X_{pt} \leq av_t \quad t=1, \dots, T$$

$$0 \leq S_{pt} \leq d_{pt} \quad p \in P; t=1, \dots, T$$

$$I_{pt} \geq 0, X_{pt} \geq 0, \quad p \in P; t=1, \dots, T$$

# Si discute a continuación el modelo descrito y su formulación en AMPL asociada a los archivos STEELT.MOD y STEELT.DAT:

The screenshot shows the AMPL IDE interface. On the left, the 'Current Directory' pane shows the file structure for the project. The 'Console' pane displays the AMPL command sequence and the resulting optimal solution. The 'STEELT.MOD' and 'STEELT.DAT' files are open in the editor.

**Current Directory:** C:\Users\Victor Albornoz\Documents\cu

- ej1ampl.mod
- ej2ampl.dat
- ej2ampl.mod
- ej2ampl.run
- STEELT.DAT
- STEELT.MOD
- STEELT.DAT
- STEELT.MOD

**Console:**

```
AMPL
bands 2 8000 8500 0
bands 3 0 0 0
bands 4 2000 2000 0
coils 0 . . 0
coils 1 3857 1337 2520
coils 2 0 2500 20
coils 3 4480 4500 0
coils 4 4200 4200 0
;

ampl: reset data;
ampl: solve;
Error executing "solve" command:
error processing var Make[...]:
no data for set PROD

ampl: reset;
ampl: model STEELT.MOD;
ampl: data STEELT.DAT;
ampl: option solver gurobi;
ampl: solve;
Gurobi 6.0.2: optimal solution; objective 727990
16 simplex iterations
ampl: display Make, Sell, Inv;
.      Make  Sell  Inv      :=
bands 0      .      .      10
bands 1 2590    .      2000   600
bands 2 8000    8500   100
bands 3 6400    6500    0
bands 4 6500    6500    0
coils 0      .      .      0
coils 1 3787    0      3787
coils 2 0      2500   1287
coils 3 0      1287    0
coils 4 1050   1050    0
;

ampl:
```

**STEELT.DAT:**

```
set PROD; # products
param T > 0; # number of weeks

param rate {PROD} > 0; # tons per hour produced
param inv0 {PROD} >= 0; # initial inventory
param avail {1..T} >= 0; # hours available in week
param market {PROD,1..T} >= 0; # limit on tons sold in week

param prodcost {PROD} >= 0; # cost per ton produced
param invcost {PROD} >= 0; # carrying cost/ton of inventory
param revenue {PROD,1..T} >= 0; # revenue per ton sold

var Make {PROD,1..T} >= 0; # tons produced
var Inv {PROD,0..T} >= 0; # tons inventoried
var Sell {p in PROD, t in 1..T} >= 0, <= market[p,t]; # tons sold

maximize total_profit:
sum {p in PROD, t in 1..T} (revenue[p,t]*Sell[p,t] -
prodcost[p]*Make[p,t] - invcost[p]*Inv[p,t]);

# Total revenue less costs in all weeks

subject to time {t in 1..T}:
sum {p in PROD} (1/rate[p]) * Make[p,t] <= avail[t];

# Total of hours used by all products
# may not exceed hours available, in each week

subject to initial {p in PROD}: Inv[p,0] = inv0[p];

# Initial inventory must equal given value

subject to balance {p in PROD, t in 1..T}:
Make[p,t] + Inv[p,t-1] = Sell[p,t] + Inv[p,t];

# Tons produced and taken from inventory
# must equal tons sold and put into inventory
```

**Taskbar:** Writable, Insert, 1:1, 17:54, 14-03-2016