Foundations of Operations Research
Introduction to AMPL and Modellisation

Pierre Hosteins
hosteins@di.unito.it

Politecnico di Milano

October 17th 2013

- Use of scripts in AMPL
- Problems of Linear Programming: practical examples

Memory Refreshing

Let us sum up the useful commands of Mathprog:

- **var**: declare a variable (continuous, integer or binary).
- **set**: defines a set of elements (products, machines, workers, time slices etc...).
- **param**: defines a parameter (costs, maximum or minimum amounts etc...).
- **subject to**: defines a (family of) constraint(s).
- **sum**: sum oprator used in formal notation.
- **minimize/maximize**: defines whether the objective must minimised or maximised.
- **expand**: expand explicitly an expression so you can visualise and check your model.
- **printf**: allows to print results and messages.

Scripts in AMPL

There are different types of files that can be loaded in AMPL:

- **Model file (extension .mod)**: contains the model written in a formal manner.
- **Data file (extension .dat)**: contains the model data.
- **Script file (extension .run)**: contains the commands that AMPL should launch automatically.

A **script** file exempts us to rewrite all the commands to load all files, set the solver, solve the problem and print the results each time we change the data in the data file, for example.

Scripts in AMPL

Let us illustrate on a simple model (`matrix.mod`):

```
set I;
set J;
param a{I,J};
param b{I};
param c{J};
var x{J} >= 0;

minimize Cost:  sum{j in J} c[j]*x[j];

subject to Matrix_Constraint{i in I}:
sum{j in J} a[i,j]*x[j] <= b[i];
```

Scripts in AMPL

The data file is (`matrix.dat`):

```
set I := 1, 2;
set J := 1 2 3;

param a:
     1    2    3     :=
 1   -1    2    3.1
 2   3.3   -7   -1    ;

param b :=
 1   -3
 2   9.2;

param c :=
 1   2
 2   4
 3   3;
```

Scripts in AMPL

Script file (`matrix.run`):

```
model matrix.mod;
data matrix.dat;
option solver cplex;

printf "\nExpand the math model with input data:\n";
expand Cost;
expand MatrixConstraint;
printf "\nInvoke the linear programming solver:\n";
solve;
printf "\nDisplay decision vector:\n";
for{i in I: x[i] > 0} {
printf "x[%s] = %f\n", i, x[i];
}
```

Scripts in AMPL

- The **expand** command can be helpful in your search for errors, by showing you how AMPL instantiated your symbolic model.
- You can integrate directly specific bounds for each variable in its definition:

```
var x{j in J} >= lowerbound[j in J], <= upperbound[j in
J];
var x{j in J} >= 0, <= upperbound[j in J];
```

Exercise: the Diet Problem revisited

We want to make the diet problem a little more concrete: in general one has at disposition different types of meal, e.g. risotto with cheese, or pasta with tomato sauce, roasted chicken with fries etc...

Consider a **set of meals** we can prepare, that use a **set of food types** that we can buy, each one having a **specific cost**, and a **set of nutriments** that need to be assimilated by the organism. Each type of meal needs a **given amount** of each type of food and each type of food brings a **given amount** of each type of nutriment: the body needs to assimilate a **minimum** quantity of each nutriment and should not assimilate more than a **maximum** quantity either.

Find the composition of the diet (each meal can be used only once) that minimises the food cost while maintaining sufficient values for each type of nutriment: define the decision values, the set and parameters describing the data and write down the formal formulation of the model.

Exercise: the Diet Problem revisited

M is the set of meals, F the set of food and N the set of nutrients. c_i is the cost per unit of food $i \in F$, q_{ij} the quantity of nutrient $j \in F$ that gives food $i \in F$ and r_{ik} the quantity of food $i \in F$ needed to prepare meal $k \in M$. m_j and M_j are the minimum and maximum quantities of nutrient $j \in N$ that should be absorbed.
Variables $x_i \geq 0$ represent the quantity of food i that we buy while we define a **binary** variable $s_k \in \{0, 1\}$ for $k \in M$ that is equal to 1 if we select meal k in the diet, 0 otherwise.

$$\text{Minimise:} \quad \sum_{i \in F} c_i \, x_i$$

$$\forall i \in F, \quad \sum_{k \in M} r_{ik} \, s_k = x_i$$

$$\forall j \in N, \quad m_j \leq \sum_{i \in F} q_{ij} \, x_i \leq M_j$$

Exercise: Shoe Production (2)

We want to plan the shoes production for a firm for a single day. 5 models of shoes can be produced, exploiting the handwork of 3 artisans. The time needed by each artisan to manufacture a pair of shoes of each model is indicated in the following table (hours/unit):

Model/Artisan	1	2	3
1	1.5	3.5	4
2	1.8	2	3
3	2	1	2.5
4	2.5	1.5	3.5
5	3.5	3.5	4.2

Exercise: Shoe Production (2)

Each artisan must work no less than 8 hours a day, but no more than 10. The next table indicates, for each model of shoes, the selling price and an estimation of the maximal amount the market can absorb:

Model	Price	Demand
1	56	4
2	86	3
3	45	3
4	42	5
5	65	8

Give an integer linear programming formulation to the problem of maximising the revenue, subject to production and demand constraints. Write an AMPL model file and a data file and solve it by CPLEX.

Exercise: Shoe Production (2)

We call S the set of shoes, A the set of artisans. The price and demand for each $s \in S$ is called p_s and d_s. The time needed by $a \in A$ to manufacture $s \in S$ is t_{sa} and minimum and maximum working time for any $a \in A$ are called T_{min} and T_{max}.
Decision variables are the amount of each shoe model s produced by each artisan a: $x_{sa} \geq 0$ (integer).

$$\text{Maximise} \quad \sum_{s \in S, a \in A} p_s \, x_{sa}$$

$$\forall s \in S, \quad \sum_{a \in A} x_{sa} \leq d_a$$

$$\forall a \in A, \quad \sum_{s \in S} t_{sa} \, x_{sa} \leq T_{max}$$

$$\forall a \in A, \quad \sum_{s \in S} t_{sa} \, x_{sa} \geq T_{min}$$

Exercise: Shoe Production (2)

Model file:

```
set S;
set A;
param t{S,A};
param p{S};
param d{S};
param Tmin;
param Tmax;
var x{S,A} integer, >= 0;

maximize Sells:  sum{s in S, a in A} p[s] * x[s,a];
subject to Demand{s in S}:  sum{a in A} x[s,a] <= d[a];
subject to MaxTime{a in A}:  sum{s in S} t[s,a] * x[s,a] <=
Tmax;
subject to MinTime{a in A}:  sum{s in S} t[s,a] * x[s,a] >=
Tmin;
```

Exercise: Shoe Production (2)

Data file:

```
set S := 1 2 3 4 5;
set A := 1 2 3;
param Tmin := 8;
param Tmax := 10;
param p :=
1 56
2 86
3 45
4 42
5 65
;
```

Exercise: Shoe Production (2)

```
param d :=
1 4
2 3
3 3
4 5
5 8
;
param t :
     1     2     3     :=
 1   1.5   3.5   4
 2   1.8   2     3
 3   2     1     2.5
 4   2.5   1.5   3.5
 5   3.5   3.5   4.2   ;
```

Exercise: Transportation Problem

A company has to ship empty containers stocked in the depots of Verona, Perugia, Roma, Pescara, Taranto, and Lamezia, to the main national harbors of Genova, Venezia, Ancona, Napoli, and Bari. The number of empty containers available in each depot, and the demand of containers in each harbor are:

	Availability
Verona	10
Perugia	12
Roma	20
Pescara	24
Taranto	18
Lamezia	40

	Demand
Genova	20
Venezia	15
Ancona	25
Napoli	33
Bari	21

Exercise: Transportation Problem

The transportation cost is of **1 Euro per km**. The distances from the depots to the harbors, in km, are the following:

	Genova	Venezia	Ancona	Napoli	Bari
Verona	290	115	355	715	810
Perugia	380	340	165	380	610
Roma	505	530	285	220	450
Pescara	655	450	155	240	315
Taranto	1010	840	550	305	95
Lamezia	1072	1097	747	372	333

Give a linear programming formulation for the problem of minimizing the total transportation cost while fulfilling the demand.