Mathematical Programming: Modelling and Applications

Sonia Cafieri

LIX, École Polytechnique

cafieri@lix.polytechnique.fr

September 2009

Outline

AMPL

- **A** **M**athematical **P**rogramming **L**anguage
- AMPL language is very close to the mathematical form
- There are AMPL constructions for sets, parameters, variables, objective, constraints: all basic ingredients of optimization problems
- There are ways to write arithmetic expressions: sums over sets, ...
- Many solvers can work with AMPL

AMPL basics

- Declare variables:

```
var x1, x2;
```

- We can also specify bounds on variables:

```
var x1 >=0;
var x2 >=0;
```

- Objective function:
  suppose we want to solve a minimization problem;
  give the name `fun` to the function:

```
minimize fun: 3*x1 - 2*x2;
```

- Constraints:
  impose a very simple linear constraint;
  give the name `constr` to the constraint:

```
subject to constr: x1 + x2 = 4;
```

AMPL basics

We defined a very simple Linear Programming problem.

Solve it using CPLEX.

- Choose the solver:

```
option solver cplex;
```

- Solve the problem:

```
solve;
```

- View the solution:

```
display x1, x2;
```

AMPL basics

Try to put all together. Use AMPL in interactive mode:

- open a terminal
- type `ampl`, obtain:
```
ILOG AMPL 10.100, licensed to "ecolepolytechnique-palaiseau".
AMPL Version 20060626 (Linux 2.6.9-5.ELsmp)
ampl:
```
- define variables, obj. function, constraints and solve the problem with CPLEX:
```
ampl: var x1 >=0; var x2 >=0;
ampl: minimize fun: 3*x1 - 2*x2;
ampl: subject to constr: x1 + x2 = 4;
ampl: option solver cplex;
ampl: solve;
ILOG CPLEX 10.100, licensed to "ecolepolytechnique-palaiseau", options
CPLEX 10.1.0: optimal solution; objective -8
0 dual simplex iterations (0 in phase I)
ampl: display x1, x2;
x1 = 0
x2 = 4

ampl:
```
- type `quit` to exit AMPL.

AMPL basics

Summarizing ...

- each variable is named in a `var` statement;
- the objective function is defined in a statement that begins with `minimize` (or `maximize`) and a name;
- each constraint is defined in a statement that begins with `subject to` and a name;
- multiplication requires an explicit `*` operator;
- the relation $\geq$ is written `>=`;
- different solvers can be used (if available on your computer!);
- `display` shows the optimal values of variables.

A linear programming model

The approach employed so far is very simple and useful for understanding the fundamental concepts. But now suppose

- there are much more variables and constraints;
- the problem data are subject to frequent changes;

$$\Downarrow$$

use a compact description of the general form of the problem:
write a model.

Example: LP in standard form

$$\begin{aligned} \min_x \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

where $x \in \mathbb{R}^n$, $c \in \mathbb{R}^n$, $A = (a_{ij}) \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$.

A linear programming model

The approach employed so far is very simple and useful for understanding the fundamental concepts. But now suppose

- there are much more variables and constraints;
- the problem data are subject to frequent changes;

$$\Downarrow$$

use a compact description of the general form of the problem:
write a model.

Example: LP in standard form

$$\begin{aligned} \min_x \quad & c^T x \\ \text{s.t.} \quad & Ax = b \\ & x \geq 0 \end{aligned}$$

where $x \in \mathbb{R}^n$, $c \in \mathbb{R}^n$, $A = (a_{ij}) \in \mathbb{R}^{m \times n}$, $b \in \mathbb{R}^m$.

A linear programming model

Basic components of a model:

- **sets**
- **parameters**
- **variables**, whose values must be computed by the solver
- **objective**, to be minimized or maximized
- **constraints**, that the solution must satisfy

In our example:

- $n = 2$ variables, $m = 1$ constraint
- variables: x_1, x_2
- parameters: $c = (3, -2)$, $a = (1, 1)$, $b = 4$

A linear programming model in AMPL

AMPL model file

```
# parameters
param n >= 1; # number of variables
param m >= 0; # number of constraints

param c{1..n};
param a{1..m,1..n};
param b{1..m};

# variables
var x{1..n} >=0;

# obj function
minimize fun : sum{j in 1..n} c[j]*x[j];

# constraint
subject to constr {i in 1..m}: sum{j in 1..n} a[i,j]*x[j] = b[i];
```

A linear programming model in AMPL

AMPL model file

Some comment:

- sets, parameters and variables must be declared before they are used, but can appear in any order;
- statements end with semicolons and can be split across lines;
- upper and lower cases letters are different (case-sensitive);
- subscripts are denoted by brackets: $c_j x_j$ –> `c[j]*x[j]`;
- some key words are used: `sum, in`, ...

A linear programming model in AMPL

AMPL data file

```
param n := 2;

param c :=
1  3
2 -2
;

param a :=
1  1
2  1
;

param b :=
4
;
```

AMPL model and data files

- the model describes an infinite number of optimization problems of the same type
- it becomes a specific problem, or instance of the model, when data values are provided
- each collection of data values define a different instance
- the same model can be used with different data
- only data file are to be changed to obtain different instances, the model has to be written only once.

Using model and data files

When using model and data files, a solution can be found by typing just a few statements:

```
ampl: model es1.mod;
ampl: data es1.dat;
ampl: option solver cplex;
ampl: solve;
ILOG CPLEX 10.100, licensed to "ecolepolytechnique-palaiseau", options:
CPLEX 10.1.0: optimal solution; objective -8
0 dual simplex iterations (0 in phase I)
ampl: display x;
x [*] :=
1  0
2  4
;
```

The `model` and `data` commands each specify a file to be read, the model file (es1.mod) and the data file (es1.dat).

The diet problem

- A classical optimization problem.
- It was formulated as linear programming (LP) problem by George Stiegler in the 1930s-1940s (before Dantzig introduced the simplex method).
- The problem is to find a *minimal cost diet* that satisfies some nutritional requirements defined by the recommended dietary allowances.
- It was motivated by the desire of defining a diet for american army in order to meet the nutritional requirements while minimizing the cost.
- Stiegler used a heuristic method and he guessed a solution of $39.93 per year (1939 prices).
- In 1947, Jack Laderman solved the problem using the new simplex method: the linear program consisted of 9 equation in 77 unknowns. It took nine clerks using hand-operated desk calculators 120 man days to solve for the optimal solution of $39.69.

  Stigler's guess for the optimal solution was off by only 24 cents per year!

The diet problem: mathematical formulation

The problem is to find a *minimal cost diet* that satisfies some nutritional requirements defined by the recommended dietary allowances.

LP formulation:

$$\begin{aligned} \min_x \quad & c^T x \\ \text{s.t.} \quad & Ax \geq b \\ & x \geq 0 \end{aligned}$$

where

- the variable x is the amounts of foods purchased,
- the vector c contains the costs of the foods,
- the matrix A gives the nutrient contents of the foods,
- the vector b contains the lower bounds of the nutrients.

Exercise: The diet problem

The problem solved in 1940s had 9 nutrients and 77 food items: we now consider a simpler problem.
Let us consider only 3 foods: bread, beef, fruit.

- 1 (unit of) bread costs 1 euro,
  1 (unit of) beef costs 6 euro,
  1 fruit costs 0.6 euro.
- 1 (unit of) bread contains 3 units of vitaminA, 0 units of vitaminB and 1 units of proteins,
  1 (unit of) beef contains 2 units of vitaminA, 1 units of vitaminB and 6 units of proteins,
  1 fruit contains 5 units of vitaminA, 4 units of vitaminB and 0 units of proteins.
- The minimum requirement over one day is of 30 units of vitamin A, 15 units of vitamin B, 25 units of proteins.

Find a diet with minimum cost.

The diet problem: AMPL model

Write an AMPL model. Define:

- sets
- parameters
- decision variables
- objective function
- constraints

Sets:

`set FOODS` foods (`bread, beef, fruit`)

`set NUTRIENTS` nutrients (`vitaminA, vitaminB, proteins`)

Variables:

`var x{FOODS} >= 0` quantity of each food to buy

The diet problem: AMPL model

Write an AMPL model. Define:

- sets
- parameters
- decision variables
- objective function
- constraints

Sets:
`set FOODS` foods (`bread, beef, fruit`)
`set NUTRIENTS` nutrients (`vitaminA, vitaminB, proteins`)

Variables:
`var x{FOODS} >= 0` quantity of each food to buy

The diet problem: AMPL model

Parameters:

`param cost{FOODS}` cost of each food

`param amount{NUTRIENTS, FOODS}` amount of nutrients in each food

`param minimum{NUTRIENTS}` minimum required amount of each nutrient

Objective function:

```
minimize total_cost:  sum{j in FOODS} cost[j]*x[j];
```

Constraints:

```
subject to min_nutr_day{i in NUTRIENTS}:
   sum{j in FOODS} amount[i,j]*x[j] >= minimum[i];
```

The diet problem: AMPL model

```
# sets
set FOODS;
set NUTRIENTS;

# parameters
param cost{FOODS};
param amount{NUTRIENTS, FOODS};
param minimum{NUTRIENTS};

# decision variables
var x{FOODS} >= 0;

# obj function
minimize total_cost: sum{j in FOODS} cost[j]*x[j];

# constraint
subject to min_nutr_day{i in NUTRIENTS}:
            sum{j in FOODS} amount[i,j]*x[j] >= minimum[i];
```

The diet problem: AMPL data

```
set FOODS := bread, beef, fruit;
set NUTRIENTS := vitaminA, vitaminB, proteins;

param cost :=
   bread 1
   beef  6
   fruit 0.6
   ;

param amount:   bread  beef  fruit :=
   vitaminA      3      2      5
   vitaminB      0      1      4
   proteins      1      6      0;

param minimum :=
   vitaminA  30
   vitaminB  15
   proteins  25;
```

The diet problem: AMPL run

```
ampl: model diet.mod;
ampl: data diet.dat;
ampl: option solver cplex;
ampl: solve;
```

Alternatively, write a run file `diet.run`:

```
model diet.mod;
data diet.dat;

option solver cplex;
solve;

display x;
```

and use it:

```
ampl < diet.run;
```

The diet problem: solution

```
ILOG CPLEX 10.100, licensed to "ecolepolytechnique-palaiseau", options: e
CPLEX 10.1.0: optimal solution; objective 26.69565217
3 dual simplex iterations (0 in phase I)
x [*] :=
 beef  3.69565
bread  2.82609
fruit  2.82609
;
```