

# Strumenti software per la PL e PLI

ver 2.2.0

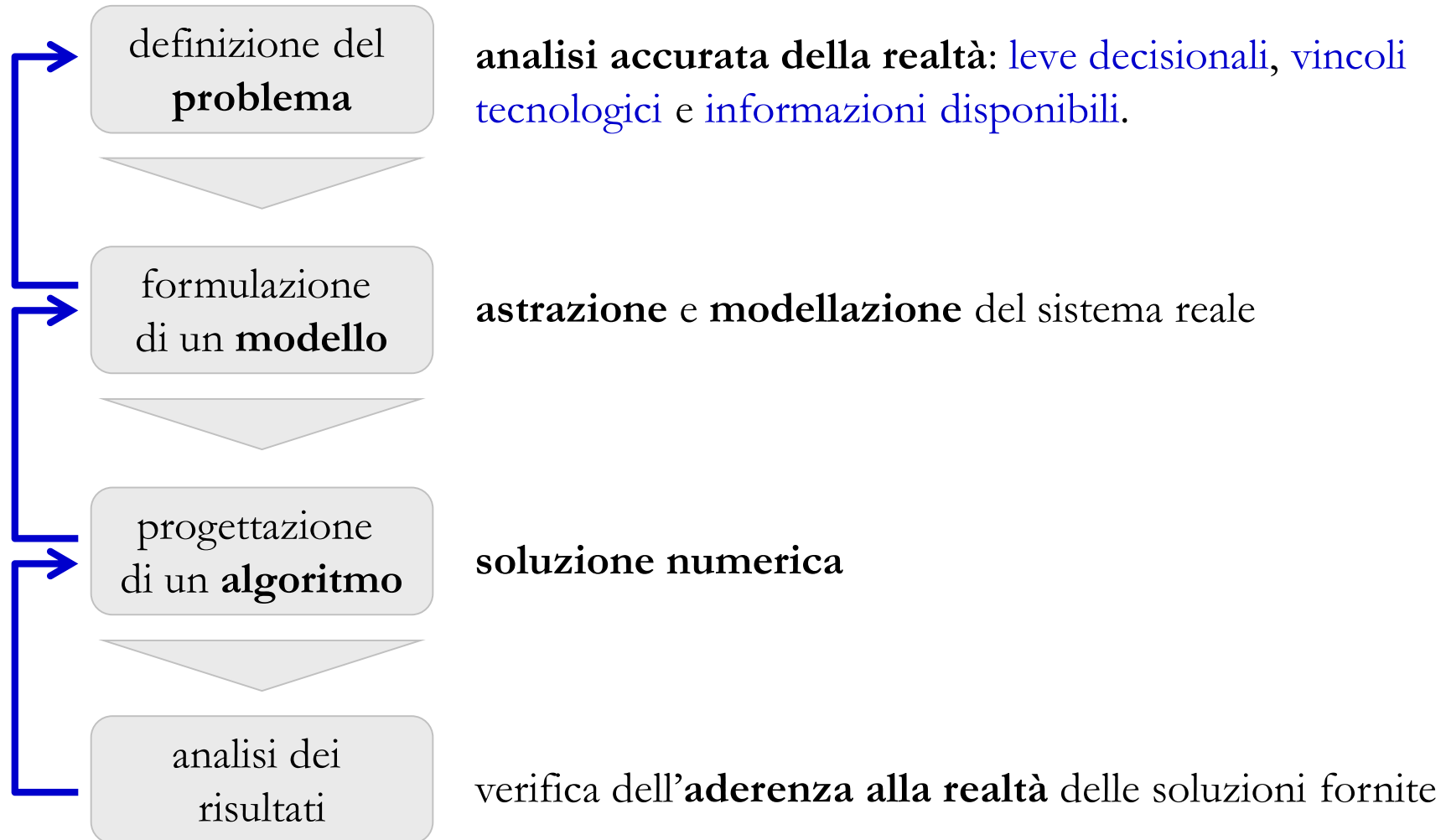
Fabrizio Marinelli

[fabrizio.marinelli@staff.univpm.it](mailto:fabrizio.marinelli@staff.univpm.it)

tel. 071 - 2204823



# Studio di ottimizzazione: approccio classico



# Software di ottimizzazione



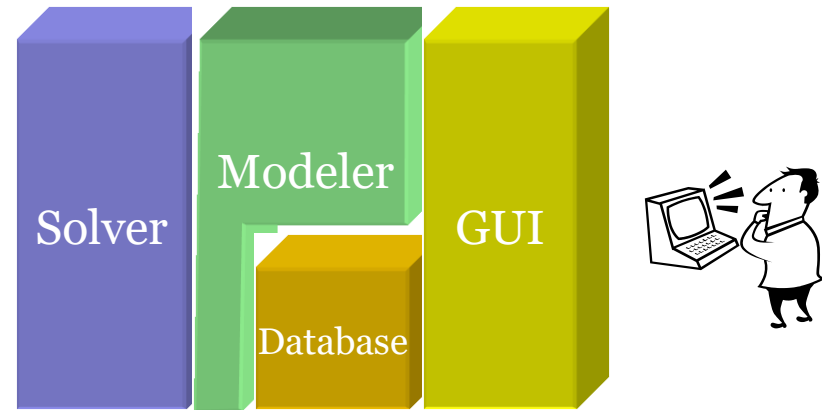
Optimization software

178.000.000 risultati

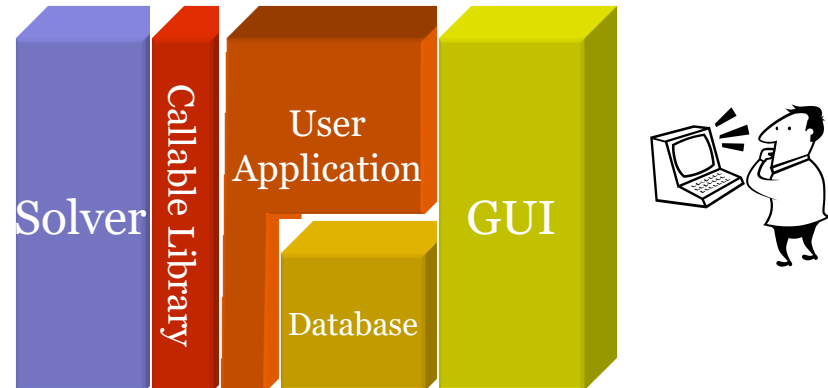
- **Focus:** strumenti software per la soluzione esatta e/o euristica di problemi di *ottimizzazione numerica* (PL/PLI, OC, problemi non lineari, ...).



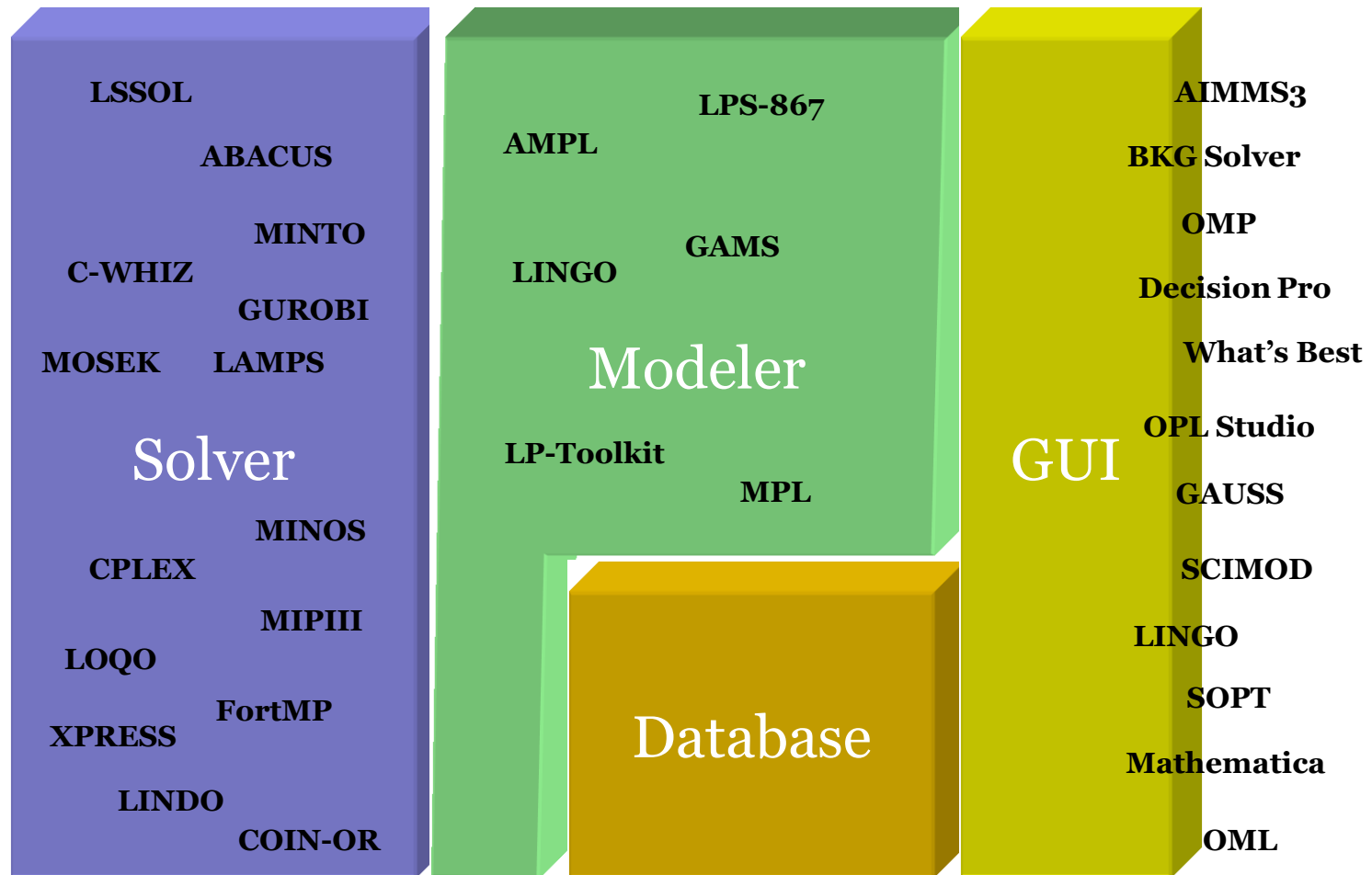
- ▶ **Ambienti integrati *general purpose***  
strumenti generali per risolvere modelli di PL e PLI.  
Richiedono esperienza in modellazione matematica.



- ▶ **Software verticalizzati**  
progettati per applicazioni specifiche.  
L'operatore definisce i problemi e interpreta i risultati rimanendo nel proprio dominio applicativo.



# Software *general purpose*: cosa offre il mercato



- Ci sono più di 200 pacchetti software

# Alcuni riferimenti in rete

- ▶ **NEOS Guide to Optimization Software**

<http://neos-guide.org/>

- ▶ **Software, Benchmark, Books, Tutorials**

<http://plato.la.asu.edu/guide.html>

- ▶ **Software for Optimization: A Buyer's Guide**

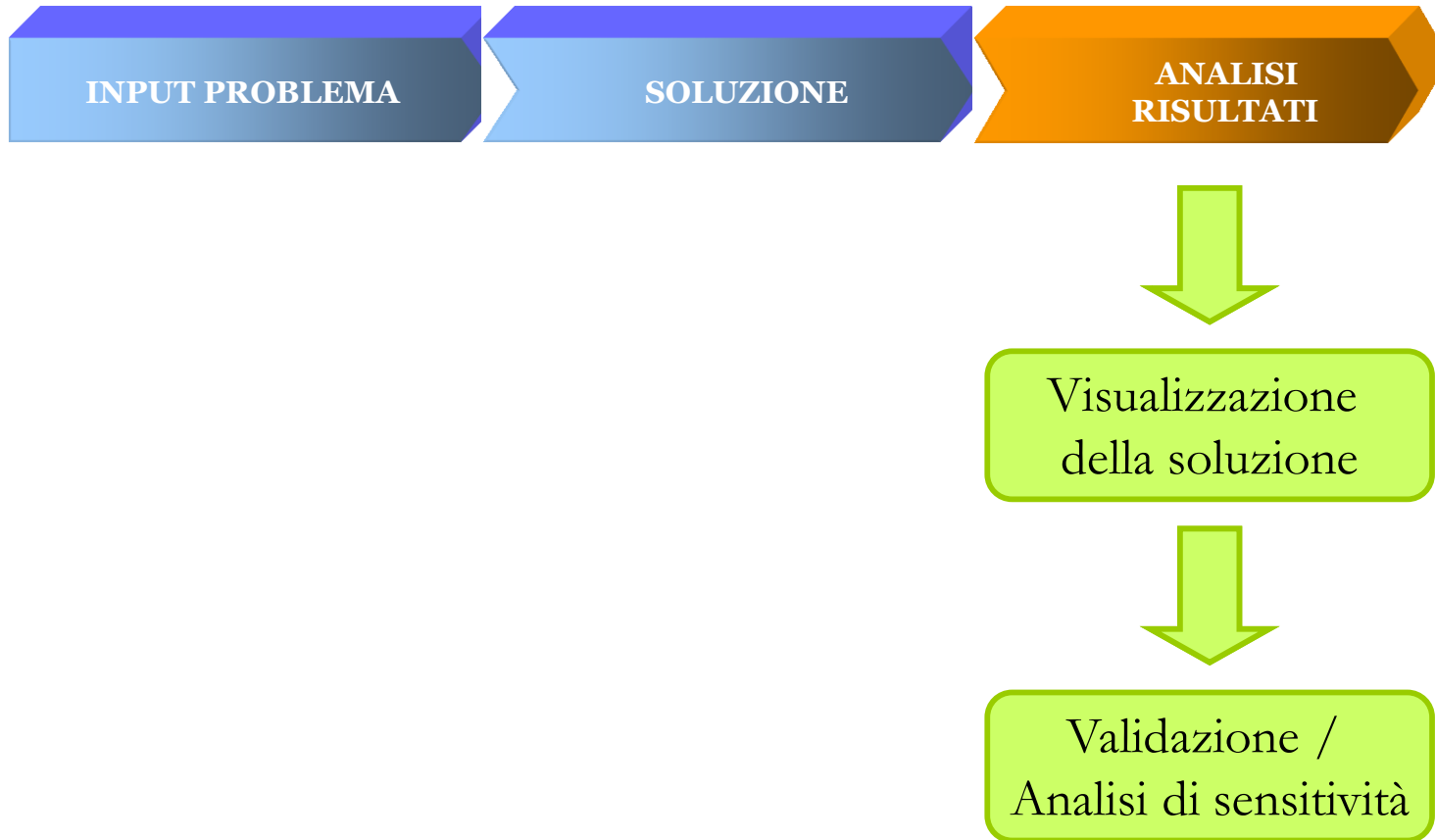
<http://www.ampl.com/solvers.html>

- ▶ **2017 linear programming software survey**

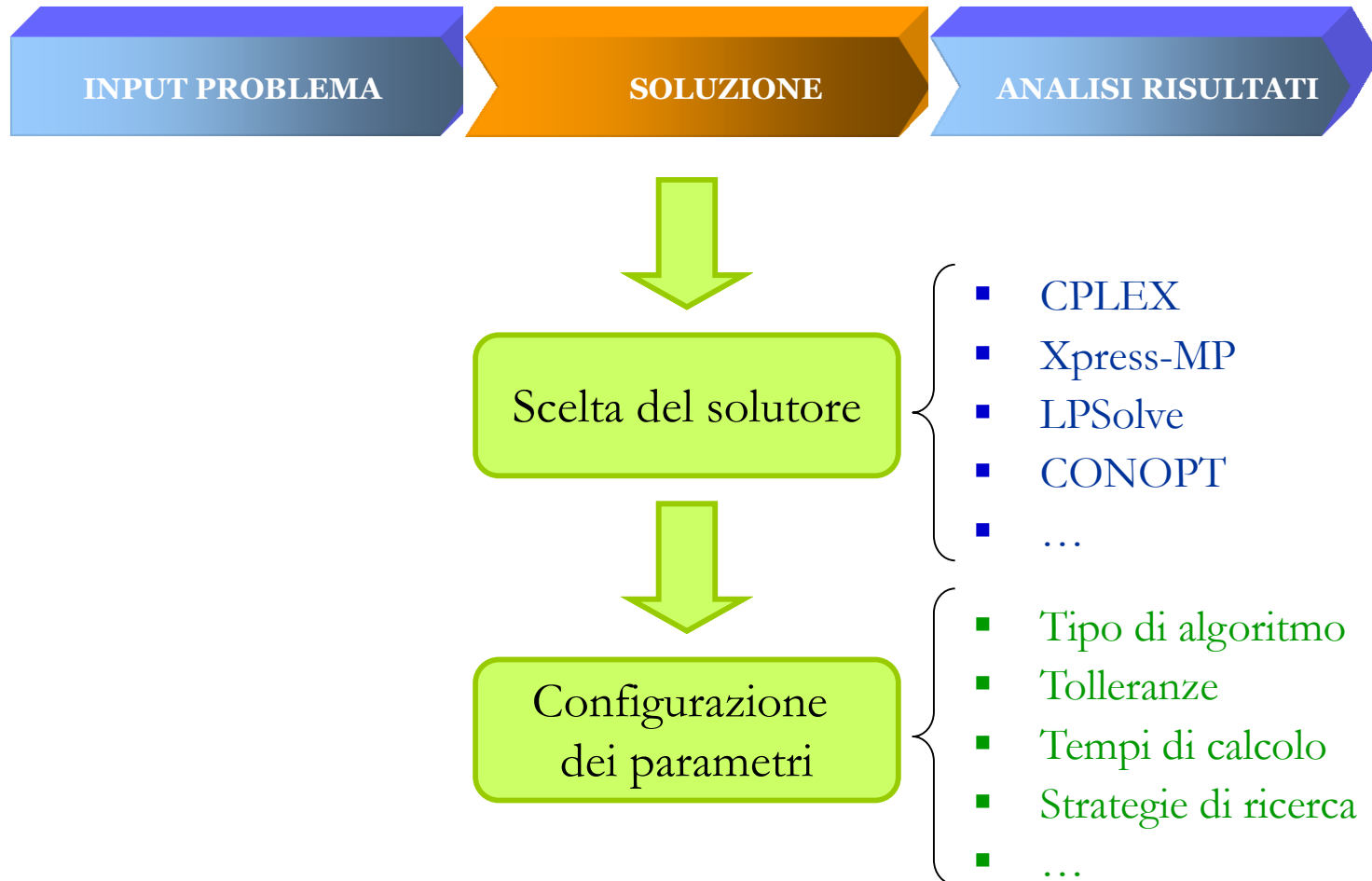
<https://www.informs.org/ORMS-Today/Public-Articles/June-Volume-44-Number-3/Linear-Programming-Software-Survey>

# Software *general purpose*: funzionalità

- Processo di soluzione articolato in 3 fasi



► Processo di soluzione articolato in 3 fasi





► Processo di soluzione articolato in 3 fasi



**Modalità 1:** il modello non è separato dai dati dell'istanza (stanno insieme in un unico file):

$$\begin{aligned} z &= \max 25x_A + 28x_B \\ 0.4x_A + 0.24x_B &\leq 312 \\ 0.4x_A + 0.45x_B &\leq 360 \\ 0.1x_A + 0.31x_B &\leq 160 \\ 0.1x_A &\leq 70 \\ x_A, x_B &\geq 0 \end{aligned}$$

► Processo di soluzione articolato in 3 fasi



**Modalità 2:** modello e dati (cioè istanza) del problema sono concettualmente e fisicamente separati (sono in file diversi)

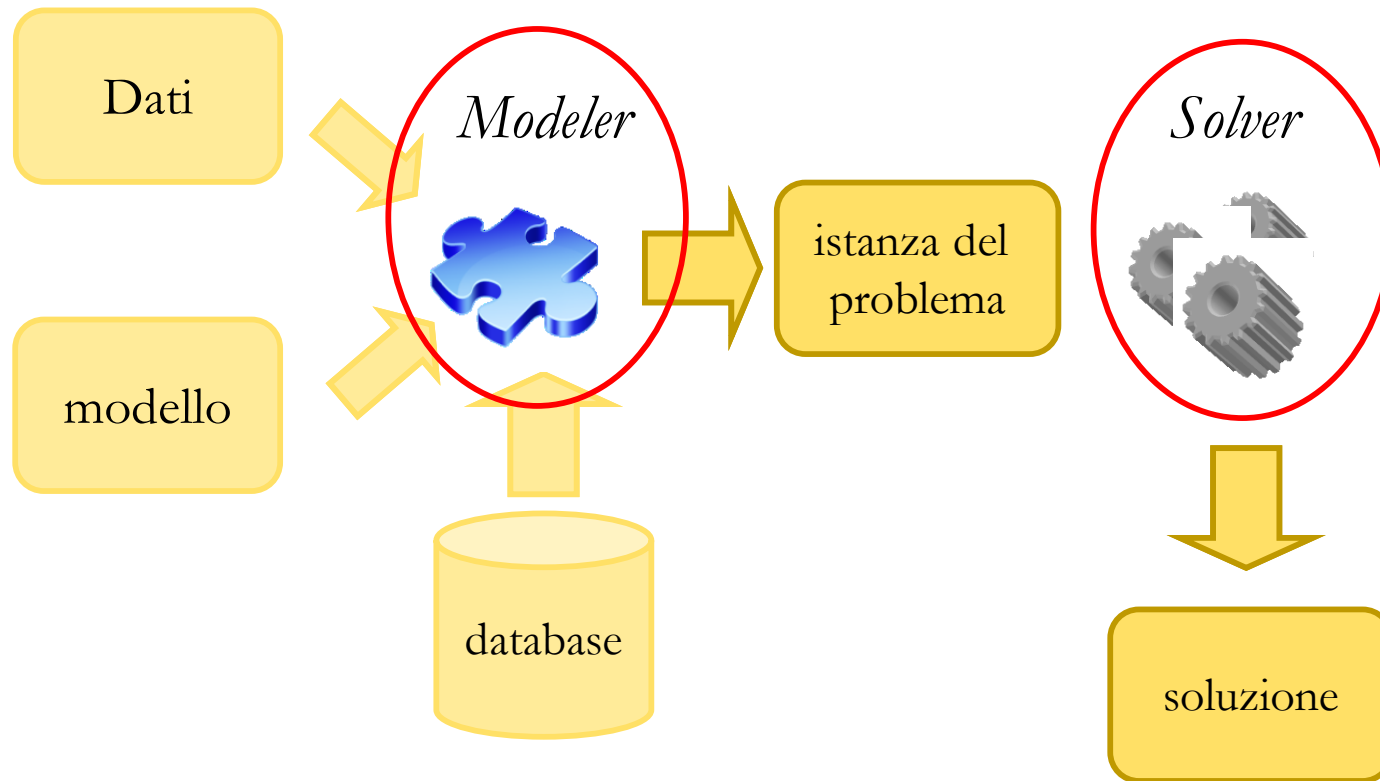
$$\begin{aligned} \max \quad & \sum_{i=1}^n c_i x_i \\ & \sum_{i=1}^n a_{ji} x_i \leq b_j \quad j = 1, \dots, m \\ & x_i \geq 0 \quad i = 1, \dots, n \end{aligned}$$



$$\begin{aligned} n &= 2, m = 4 \\ \mathbf{c} &= [25, 28] \\ \mathbf{b} &= [312, 360, 160, 70] \\ \mathbf{A} &= \begin{bmatrix} 0.4 & 0.24 \\ 0.4 & 0.45 \\ 0.1 & 0.31 \\ 0.1 & 0 \end{bmatrix} \end{aligned}$$

# Modellatore e Solutore

sono software *stand-alone*  
utilizzabili in modo combinato.



# Alcune configurazioni possibili

- ▶ I **solver** e i **modeler** sono software *stand-alone* utilizzabili in modo combinato. In genere
  - un modeler permette la selezione di più solver;
  - un solver processa modelli in diversi formati (ottenuti con diversi modeler)

## ▶ [Esempio]

AMPL

- ▶ Può utilizzare più di 35 solver tra cui CPLEX, LPSolve, MOSEK, Xpress-MP, SNOPT, MINTO

<http://www.ampl.com/>

LPSolve

- ▶ Accetta in input file in formato MathProg, CPLEX, LINDO, Xpress-MP,...

<http://lpsolve.sourceforge.net/5.5/>

# Linguaggi di modellazione algebrica (AML)

- ▶ I linguaggi di modellazione algebrica sono:
  - prevalentemente **dichiarativi** (e non *procedurali*);
  - ad **alto livello**: dati e comandi sono molto astratti e ciò permette di scrivere “programmi” molto concisi e comprensibili.  
Tali programmi non richiedono l’uso di sofisticate strutture dati e/o di particolari conoscenze informatiche;
  - formalmente **strutturati** e quindi interpretabili direttamente da un software;

► I linguaggi di modellazione algebrica realizzano:

- la *separazione fisica e concettuale* tra **dati** e **modello**. Ciò introduce una grande flessibilità:
  - si possono cambiare i dati senza cambiare modello, ovvero si possono risolvere immediatamente diverse istanze dello stesso problema;
  - si possono importare i dati da diverse sorgenti (file di testo, spreadsheet, database, ...)
  - si possono sfruttare le funzionalità di analisi proprie delle basi di dati;
- la *separazione* tra attività di **modellazione** e **soluzione**: lo stesso modello può essere risolto da diversi *solver* scelti sulla base della disponibilità o dell'adeguatezza.

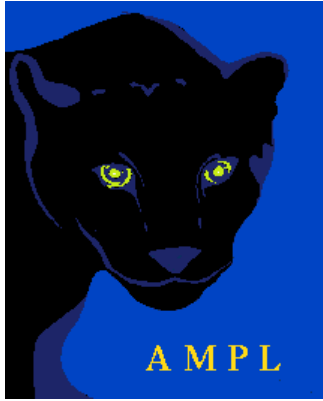
- ▶ *GAMS (General Algebraic Modeling System)*  
<http://www.gams.com>
- ▶ *MPL (Mathematical Programming Language)*  
<http://www.maximal-usa.com/mpl/>
- ▶ *AMPL (Another Mathematical Programming Language)*  
<http://www.ampl.com>
- ▶ **LINGO**  
<http://www.lindo.com>

Questi linguaggi AML hanno tutti la stessa struttura generale ma ognuno ha la propria sintassi

*AMPL*



# Installazione e configurazione

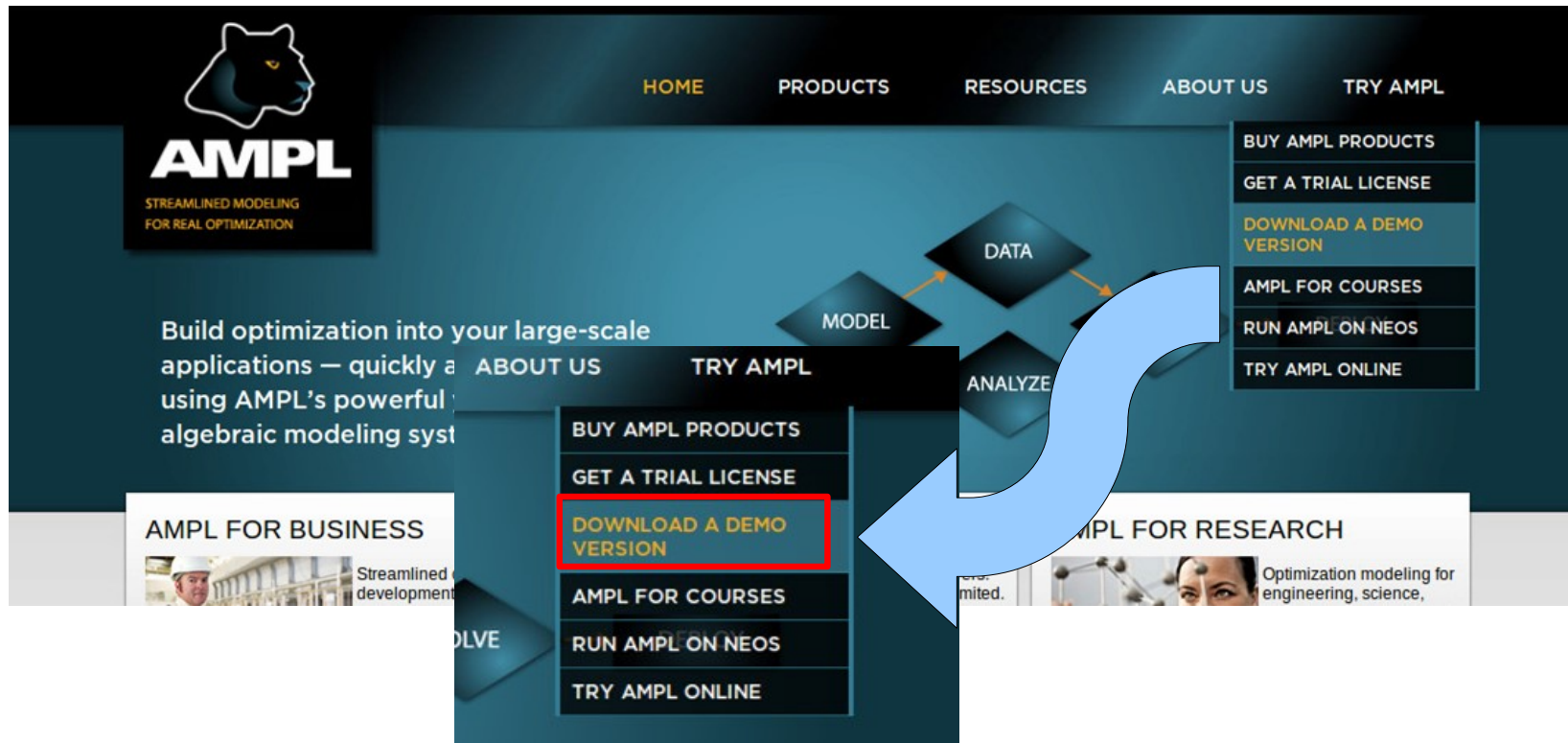


- Il software ([student Edition](http://www.ampl.com/)) è disponibile sul sito <http://www.ampl.com/> al link  
[TRY AMPL\DOWNLOAD A FREE DEMO](#)

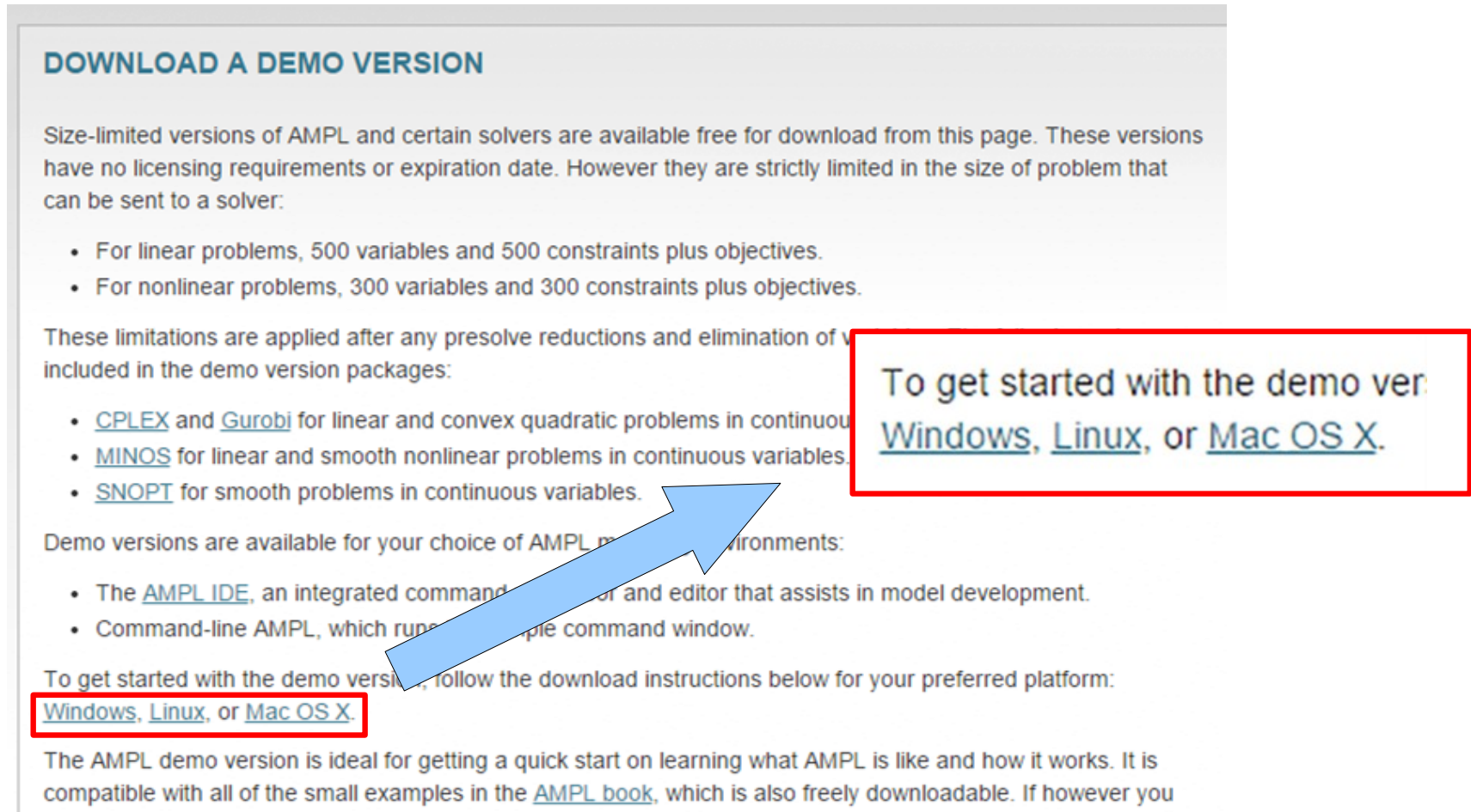
## ► Istruzioni per l'installazione:

1. Estrai l'archivio *amplcm1.zip* in una cartella a propria scelta
2. Apri una finestra terminal e scrivi *ampl*
3. Il solver di default è MINOS; per usare CPLEX usa il comando  
*option solver cplex;*

- Dal sito [www.ampl.com](http://www.ampl.com) andate nella pagina per scaricare una versione demo



- Scaricare la versione AMPL IDE per il vostro sistema operativo



**DOWNLOAD A DEMO VERSION**

Size-limited versions of AMPL and certain solvers are available free for download from this page. These versions have no licensing requirements or expiration date. However they are strictly limited in the size of problem that can be sent to a solver:

- For linear problems, 500 variables and 500 constraints plus objectives.
- For nonlinear problems, 300 variables and 300 constraints plus objectives.

These limitations are applied after any presolve reductions and elimination of variables. The solvers included in the demo version packages:

- [CPLEX](#) and [Gurobi](#) for linear and convex quadratic problems in continuous variables.
- [MINOS](#) for linear and smooth nonlinear problems in continuous variables.
- [SNOPT](#) for smooth problems in continuous variables.

Demo versions are available for your choice of AMPL environments:

- The [AMPL IDE](#), an integrated command window and editor that assists in model development.
- Command-line AMPL, which runs in a separate command window.

To get started with the demo version, follow the download instructions below for your preferred platform:

[Windows, Linux, or Mac OS X.](#)

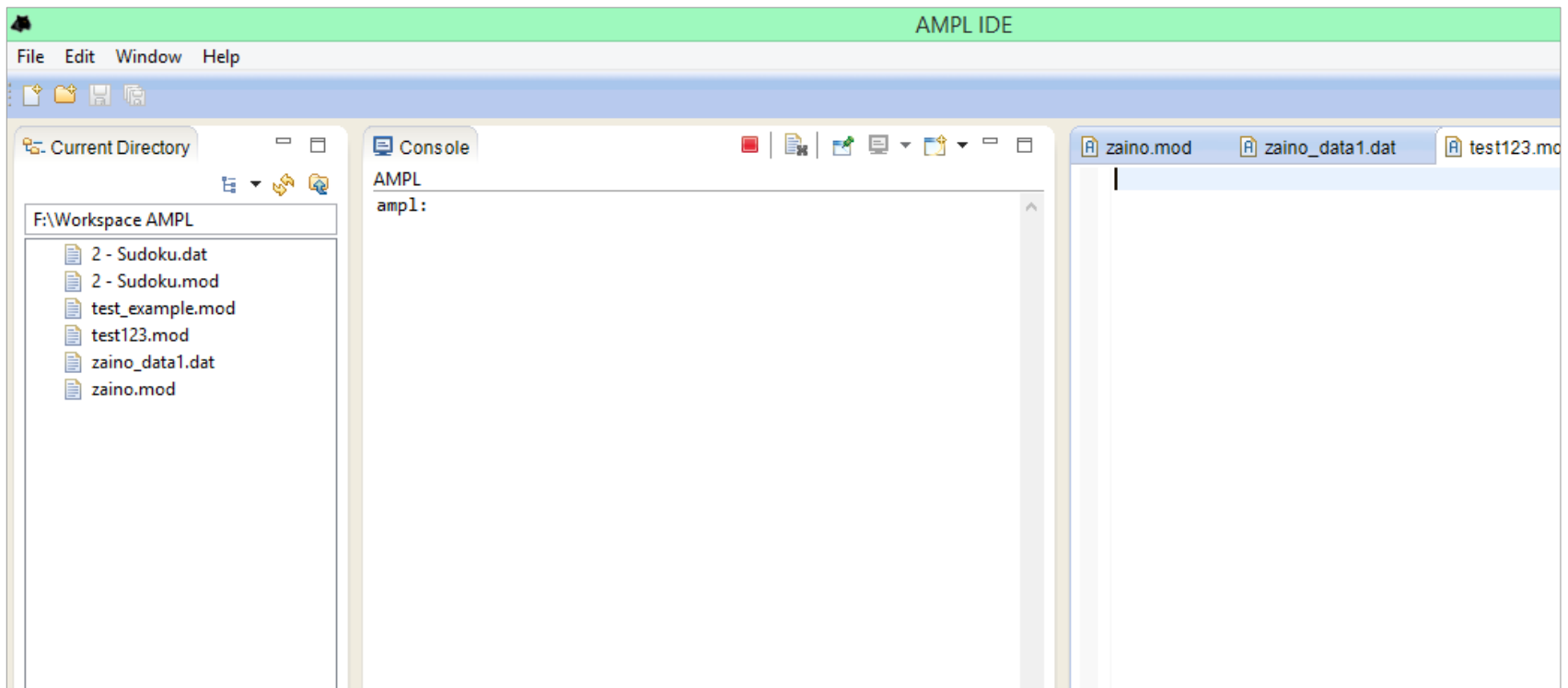
The AMPL demo version is ideal for getting a quick start on learning what AMPL is like and how it works. It is compatible with all of the small examples in the [AMPL book](#), which is also freely downloadable. If however you

- Scaricare e scompattare il file *amplide-demo-mswin.zip* e spostare la cartella appena estratta **amplide-demo** nel percorso che si preferisce.

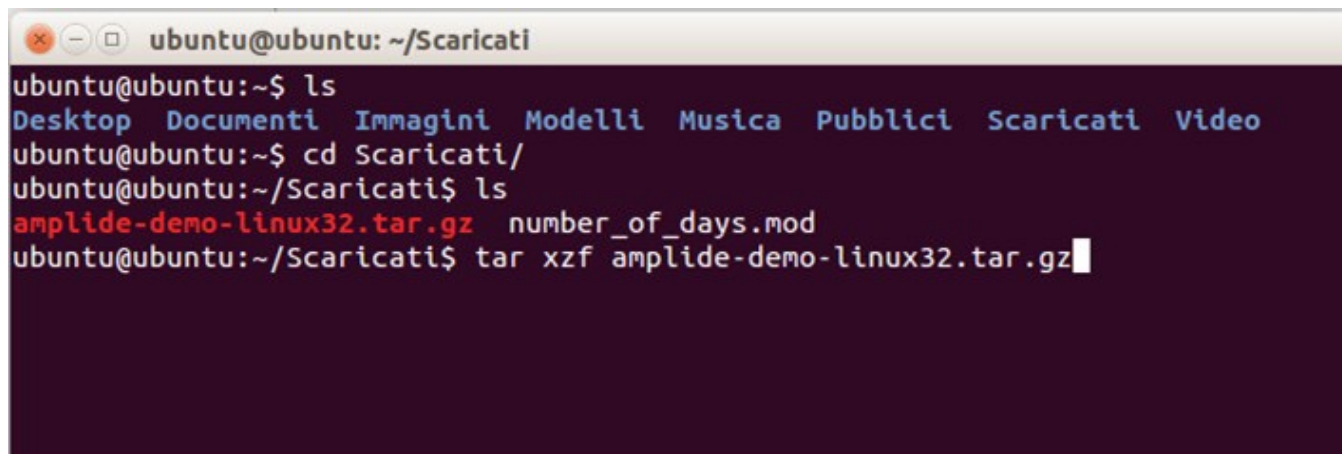
#### AMPL IDE demo version for Windows

**To install:** Download the distribution zipfile, [amplide-demo-mswin.zip](#). Double-click the zipfile icon, or apply an unzip utility, to extract the folder named `amplide` from the zipfile. You may move this folder to any convenient location on your computer.

- Eseguire il file *amplide.exe* per lanciare l'ambiente di sviluppo di AMPL

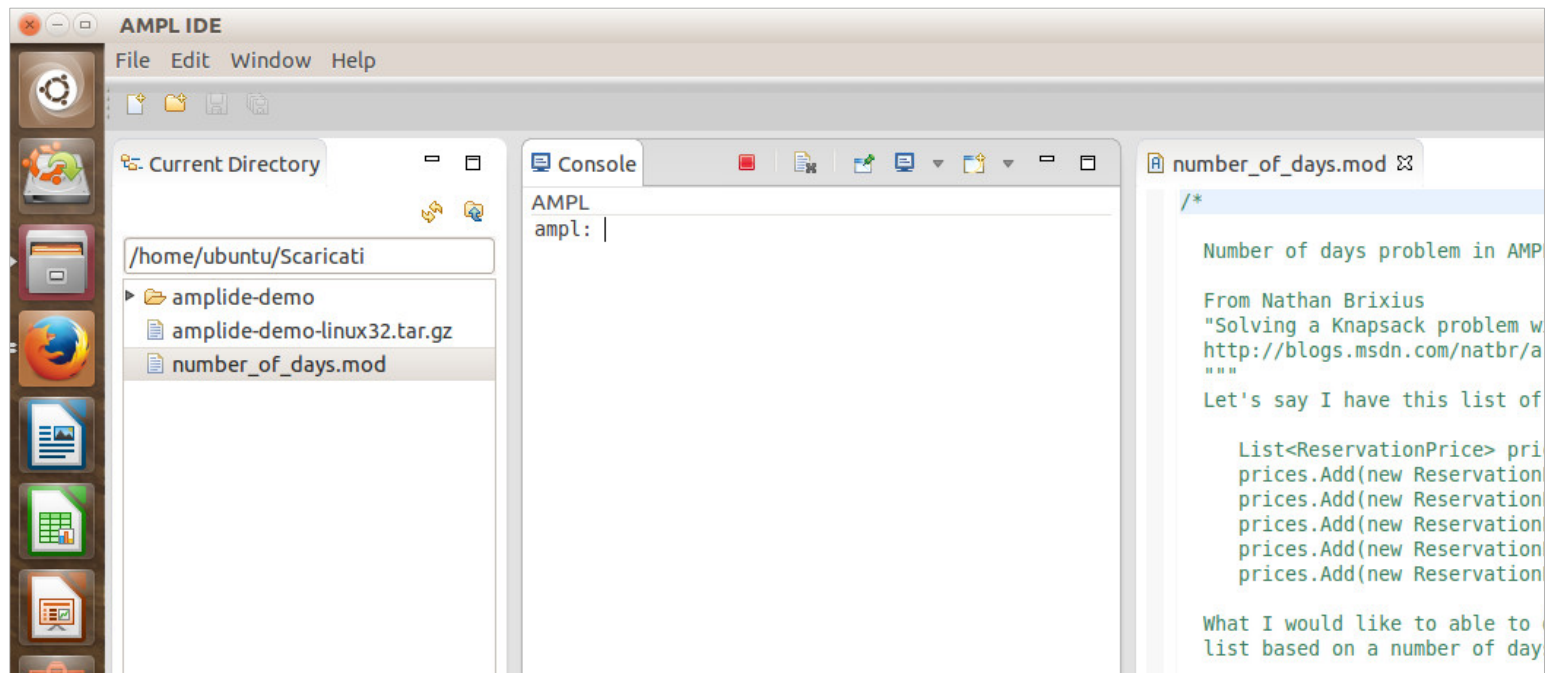


- Scaricare il pacchetto *amplide-demo-linux32.tar.gz* oppure *amplide-demo-linux64.tar.gz* in base al vostro sistema operativo
- Con un shell posizionarsi nello stesso percorso della cartella dove si è scaricato il file ed eseguire il comando:  
**tar xzf** *<nome\_del\_file\_scaricato>*



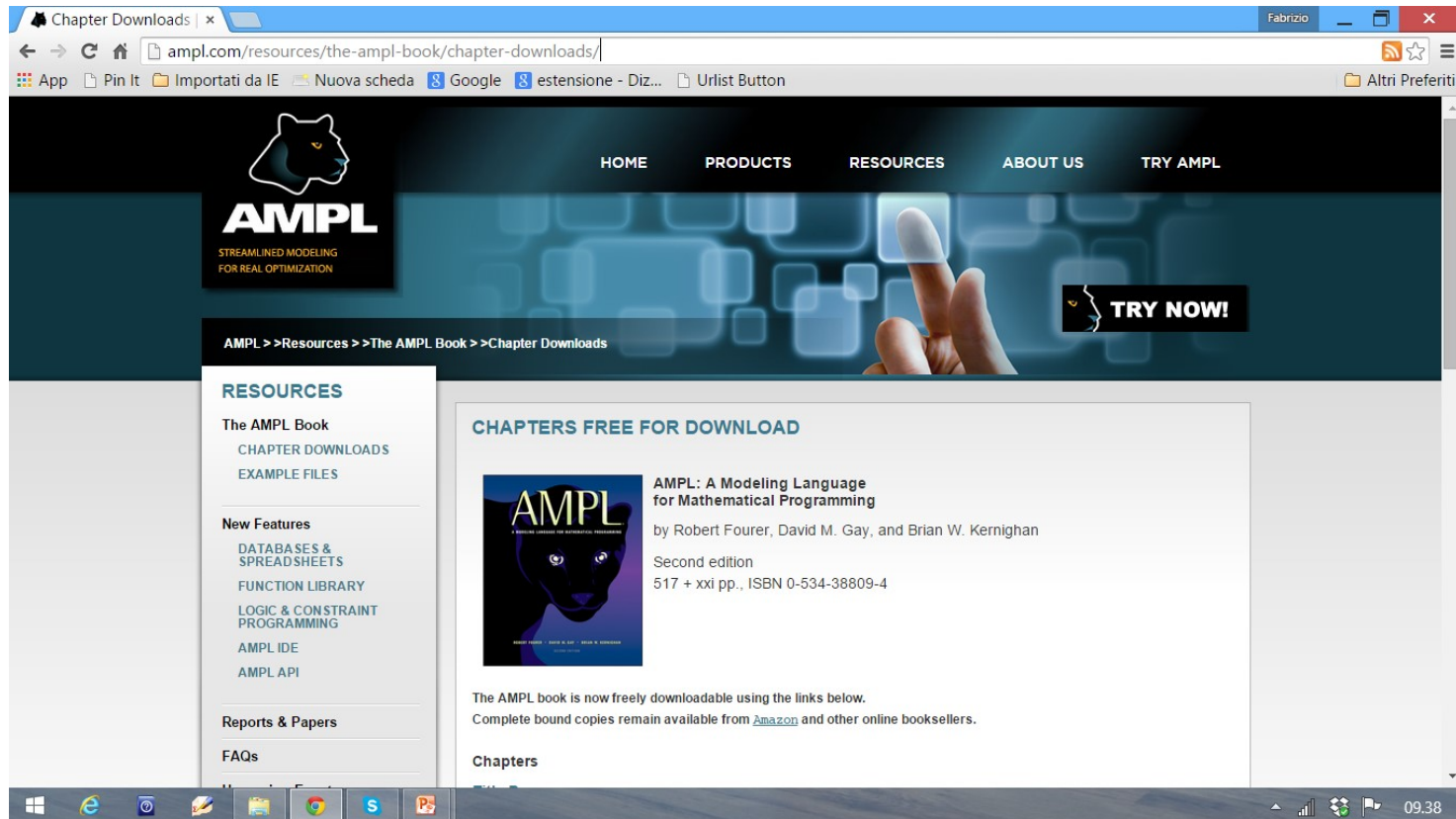
```
ubuntu@ubuntu: ~/Scaricati
ubuntu@ubuntu:~$ ls
Desktop  Documenti  Immagini  Modelli  Musica  Pubblici  Scaricati  Video
ubuntu@ubuntu:~$ cd Scaricati/
ubuntu@ubuntu:~/Scaricati$ ls
amplide-demo-linux32.tar.gz  number_of_days.mod
ubuntu@ubuntu:~/Scaricati$ tar xzf amplide-demo-linux32.tar.gz
```

- Con la shell posizionarsi nella cartella **amplide** ed eseguire l'IDE con il comando `./amplide`





<http://ampl.com/resources/the-ampl-book/chapter-downloads/>





# Esempio: mix produttivo

- **[Problema]** La società *Merlin* produce i concimi *prato starter* (tipo A) e *prato estate* (tipo B) che vende rispettivamente a 25 e 28 €/Kg. Considerando la composizione dei singoli concimi e le disponibilità in magazzino (vedi tabella) qual è il guadagno massimo che si può ottenere producendo i concimi di tipo A e B?

	composizione			
	Azoto	Fosforo	Potassio	Magnesio
tipo A	40%	40%	10%	10%
tipo B	24%	45%	31%	-
disponibilità (Kg)	312	360	160	70

# mix produttivo: modello

$$z = \max 25x_A + 28x_B$$

$$\text{C1:} \quad 0.4x_A + 0.24x_B \leq 312$$

Vincolo sulla disponibilità di azoto

$$\text{C2:} \quad 0.4x_A + 0.45x_B \leq 360$$

Vincolo sulla disponibilità di fosforo

$$\text{C3:} \quad 0.1x_A + 0.31x_B \leq 160$$

Vincolo sulla disponibilità di potassio

$$\text{C4:} \quad 0.1x_A \leq 70$$

Vincolo sulla disponibilità di magnesio

$$\text{C5:} \quad x_A, x_B \geq 0$$

Vincoli di non negatività

# Le istruzioni base di un modello

$$z = \max 25x_A + 28x_B$$

$$C1: \quad 0.4x_A + 0.24x_B \leq 312$$

$$C2: \quad 0.4x_A + 0.45x_B \leq 360$$

$$C3: \quad 0.1x_A + 0.31x_B \leq 160$$

$$C4: \quad 0.1x_A \leq 70$$

$$C5: \quad x_A, x_B \geq 0$$

```
reset;
```

```
# Dichiarazione delle variabili
```

```
var xA >= 0, integer;
```

```
var xB >= 0, integer;
```

```
# Funzione obiettivo
```

```
maximize z: 25*xA + 28*xB;
```

```
# vincoli
```

```
subject to disp_azoto: 0.4*xA + 0.24*xB <= 312;
```

```
subject to disp_fosforo: 0.4*xA + 0.45*xB <= 360;
```

```
subject to disp_potassio: 0.1*xA + 0.31*xB <= 160;
```

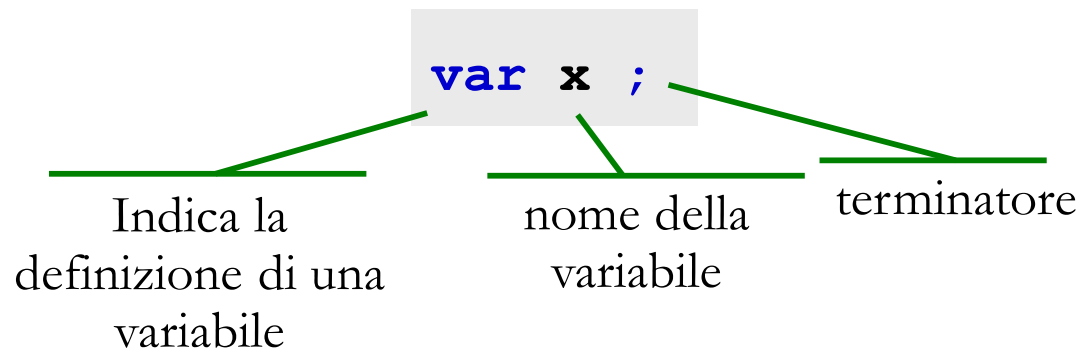
```
subject to disp_magnesio: 0.1*xA <= 70;
```



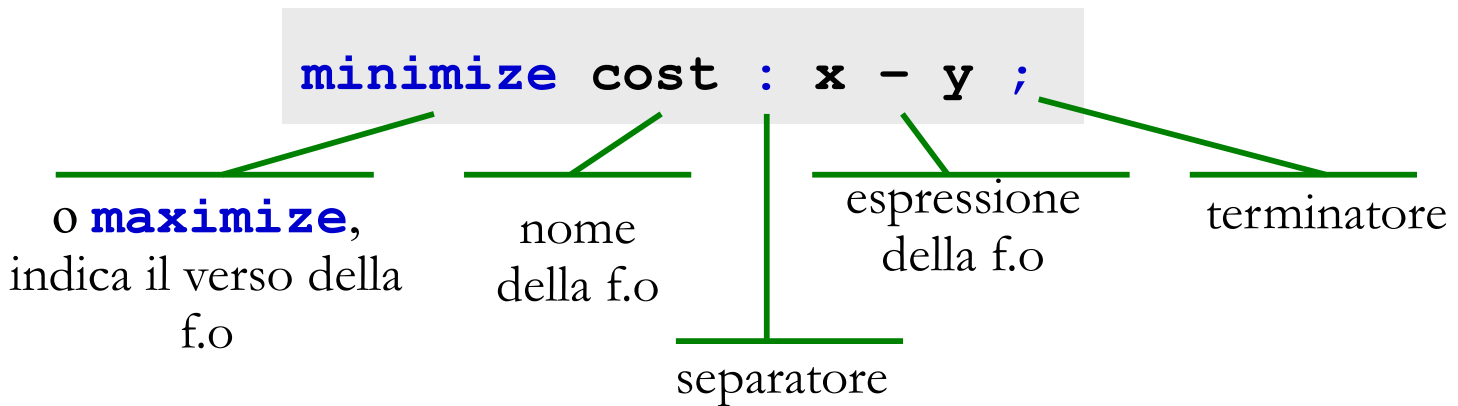
La scrittura di un modello inizia con il comando **model**; e termina con il comando **end**;

- I nomi delle variabili, dei vincoli e della funzione obiettivo devono essere **unici** (AMPL è *case sensitive*)
- **#** indica l'inizio di un commento
- **;** indica la fine di una riga (funzione obiettivo, vincolo,...)

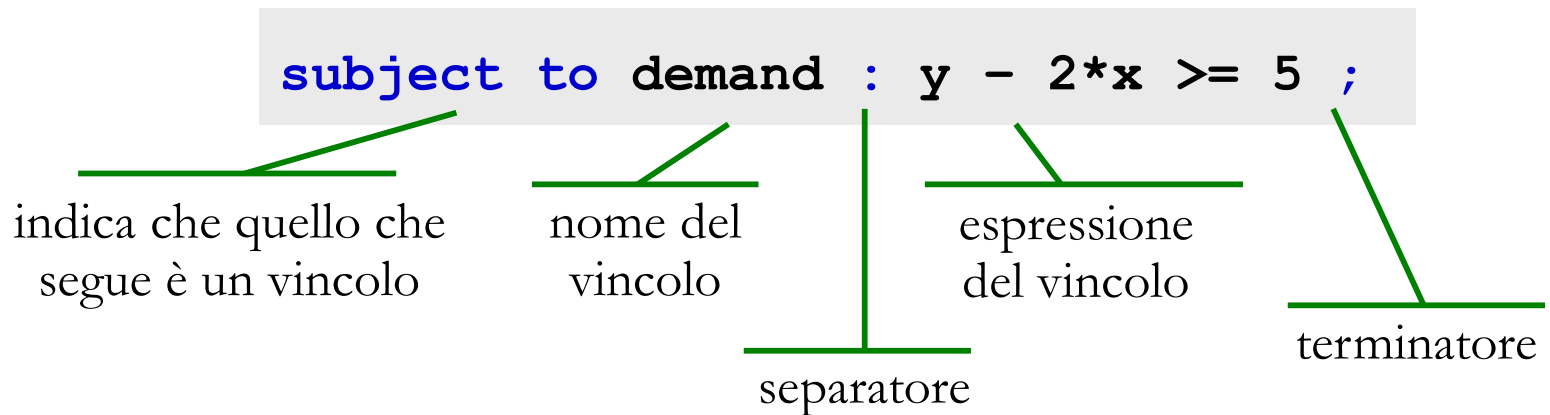
- Definizione di una variabile



- Definizione della funzione obiettivo



- Definizione di un vincolo



# I comandi di sopravvivenza

**Importante!** ogni comando deve terminare con il carattere ;

- `reset;` Cancella il modello e i dati in memoria
- `option solver cplex;` Imposta cplex come solutore
- `solve;` Risolve il modello presente in memoria
- `display var1;` Visualizza il valore della variabile **var1**



# mix produttivo – un modello parametrico

- La scrittura esplicita del modello può essere lunga e tediosa oltre che limitata a una sola istanza. Nella maggior parte dei casi è utile utilizzare insiemi indicizzati che indicano variabili, parametri (e anche vincoli)
- Il modello del precedente esempio può essere reso di dimensione variabile introducendo parametri e variabili indicizzate.

## Parametri

- $n$  = numero prodotti
- $m$  = numero sostanze chimiche

## Parametri indicizzati

- $c_i$  = ricavo unitario in Euro del prodotto  $i$  (con  $i = 1, \dots, n$ )
- $b_j$  = disponibilità in Kg della sostanza  $j$  (con  $j = 1, \dots, m$ )
- $a_{ij}$  = % di sostanza  $j$  nel prodotto  $i$

## Variabili indicizzate

- $x_i$  = produzione in Kg del prodotto  $i$  (con  $i = 1, \dots, n$ )

$$\max \sum_{i=1}^n c_i x_i$$

$$\sum_{i=1}^n a_{ij} x_i \leq b_j \quad j = 1, \dots, m$$

$$x_i \geq 0 \quad i = 1, \dots, n$$

- ▶ Come si esprimono in AMPL vettori e matrici di parametri?
- ▶ Come si esprimono in AMPL sommatorie e insiemi di vincoli?

# AMPL – parametri e indici (modello)

- Parametro

```
param n;
```

- Vettore di parametri indicizzati da 1 a  $n$

```
param c{i in 1..n};
```

- Matrice ( $n \times m$ ) di parametri

```
param a{i in 1..n, j in 1..m};
```

**Nota:** La dichiarazione di variabili indicizzate è analoga

# AMPL – sommatorie e set di vincoli (modello)

- Sommatoria  $\sum_{i=1}^n c_i x_i$

```
sum {i in 1..n} c[i]*x[i];
```

- Set di vincoli  $x_i \geq 0 \quad i = 1, \dots, n$

```
subject to segno{i in 1..n}: x[i] >= 0;
```

# AMPL – parametri e indici (dati)

- Parametro

```
param n := 2;
```

- Vettore di parametri indicizzati da 1 a  $n$

```
param c := ① 25 ② 28;
```

$i = 1, \dots, n$

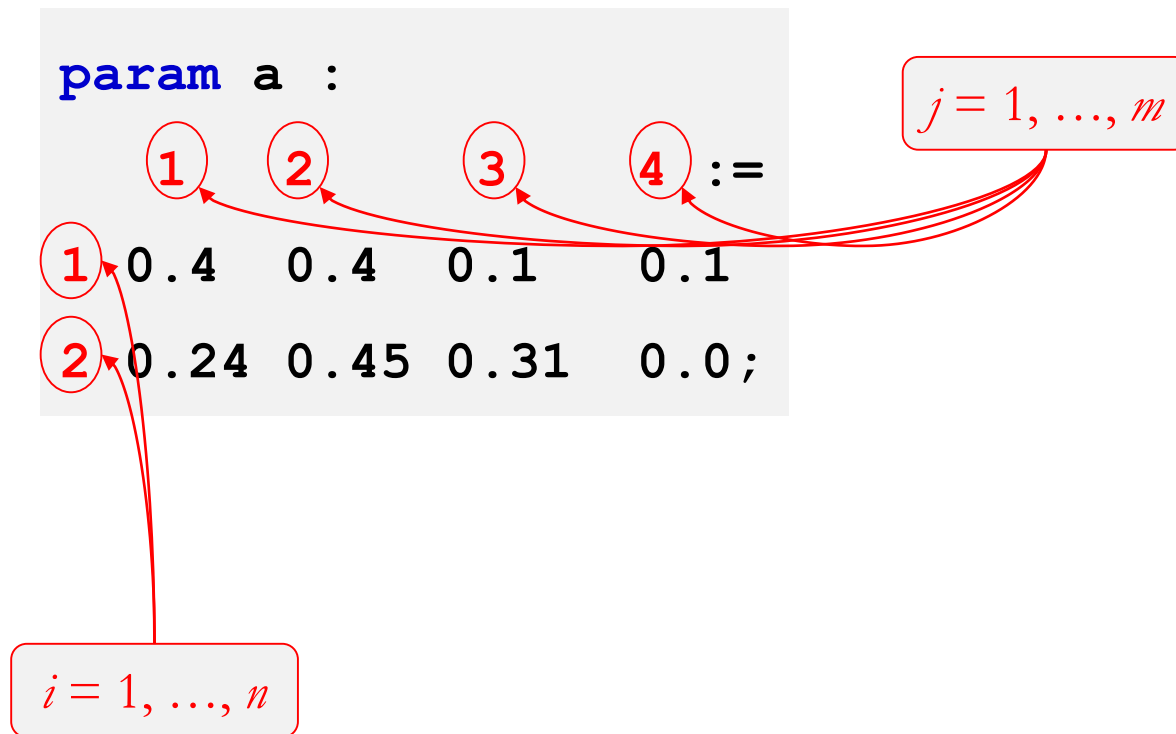
```
param c :=
```

① 25

② 28;

Si può utilizzare **un layout più intuitivo**  
(AMPL ignora l' “a capo” )

- Matrice ( $n \times m$ ) di parametri



## AMPL – insiemi (modello)

- Si possono definire insiemi (**set**) utili per indicizzare insiemi finiti di elementi

```
param n;  
  
param m;  
  
set prodotti := {1..n};  
  
set sostanze := {1..m};  
  
param c{prodotti};  
  
param a{prodotti,sostanze};  
  
sum {i in prodotti} c[i]*x[i];
```



# Le nozioni di sopravvivenza

- ▶ La separazione tra **modello** e **dati** è fondamentale per effettuare numerose prove sullo stesso modello.
- ▶ AMPL prevede tre tipi diversi di file (tutti di testo e modificabili con un semplice editor, per esempio *notepad*)
  - estensione **.mod**: descrizione in forma *parametrica* del **modello**
  - estensione **.dat**: dati del problema (i valori da assegnare ai parametri del modello)
  - estensione **.run**: *script* di comandi, cioè lista di comandi che viene eseguita automaticamente

# I comandi di sopravvivenza

- `model file.mod;`

Legge il modello descritto in file.mod

- `data file.dat;`

Legge i dati presenti in file.dat

- `include file.run;`

Esegue lo script di comandi file.run

# AMPL: script di comandi

- Per eseguire automaticamente una sequenza di comandi si può utilizzare uno script (file **.run**) con la lista dei comandi, ed eseguirlo con il comando **include file.run;**

```
reset;  
  
model mix_prod_set.mod;  
  
data mix_prod_set.dat;  
  
option solver cplexamp;  
  
solve;  
  
display x;
```

# Comandi utili – Insiemi

- ▶ Con il comando **default** si indicano i valori iniziali di un parametro, una variabile e un insieme

File .mod	File .dat
<pre>param t;  set T := {1..t};  set U{T} <b>default {};</b></pre>	<pre>param t := 1000;</pre>



l'insieme U, indicizzato su T, inizialmente è vuoto.

# Comandi utili – Parametri (1)

File .mod	File .dat
set M;	set M := a b c d;
set 0;	set 0 := x y z;
param p{M,0};	param p: a b c d := e 1 5 3 2 f 4 3 1 0 g 3 2 7 8;

# Comandi utili – Parametri (2)

File .mod	File .dat
param n;	param n:= 3;
set M;	set M:= a b c d;
set O;	set O:= e f g h;
set N;	set N:= {1..n};
	param q:
	[*,*,1]:     a   b   c   d :=
	e 30 10 8 10
	f 22 7  10 7
	g 19 11 12 10
	[*,*,2]:     a   b   c   d :=
	e 39 14 11 14
	f 27 9  12 9
	g 24 14 17 13
	[*,*,3]:     a   b   c   d :=
	e 41 15 12 16
	f 29 9  13 9
	g 26 14 17 13;
param q{M,O,N};	

# Comandi utili – Variabili

- ▶ Se non espresso tramite comandi, le variabili sono considerate numeri reali. Al momento della dichiarazione si possono esprimere vincoli sul valore della variabile

---

**File .mod**

---

`var x;`

`var x{N};`

`var x{1..n} >=0 <=1;`

`var x{1..n,M} integer;`

`var x{M,O,N} binary;`

---

# Comandi utili – funzioni matematiche

Valore assoluto di $x$	<code>abs(x)</code>
Parte intera inferiore di $x$	<code>ceil(x)</code>
Parte intera superiore di $x$	<code>floor(x)</code>
Logaritmo naturale $\log_e x$	<code>log(x)</code>
Logaritmo decimale $\log_{10} x$	<code>log10(x)</code>
Esponenziale di $x$	<code>exp(x)</code>
Radice quadrata di $x$	<code>sqrt(x)</code>
Minimo tra 2 o più numeri ( $x, y, z, \dots$ )	<code>min(x, y, z, ...)</code>
Massimo tra 2 o più numeri ( $x, y, z, \dots$ )	<code>max(x, y, z, ...)</code>



# Comandi utili – funzioni matematiche

Arcoseno( $x$ )	<code>acos(x)</code>
Arcoseno iperbolico( $x$ )	<code>acosh(x)</code>
Arcocoseno( $x$ )	<code>asin(x)</code>
Arcocoseno iperbolico( $x$ )	<code>asinh(x)</code>
Arcotangente iperbolico( $x$ )	<code>atanh(x)</code>
Seno( $x$ )	<code>sin(x)</code>
Coseno( $x$ )	<code>cos(x)</code>
Tangente( $x$ )	<code>tan(x)</code>
Tangente iperbolica( $x$ )	<code>tanh(x)</code>

# Comandi utili – operatori aritmetici

Potenza	$\wedge$ oppure $**$
Numero negativo	$-$
Somma	$+$
Sottrazione	$-$
Moltiplicazione	$*$
Divisione	$/$
Divisione	div
Modulo	mod
Differenza non negativa: $\max(a - b, 0)$	less
Sommatoria	sum
Produttoria	prod
Minimo	min
Massimo	max

# Comandi utili – operatori aritmetico-logici

Maggiore, maggiore o uguale

>, >=

Minore, minore o uguale

<, <=

Uguale

= oppure ==

Diverso

<> oppure !=

Negazione logica

Not oppure !

And logico

and oppure &&

Quantificatore esistenziale logico

exists

Quantificatore universale logico

forall

Or logico

or oppure ||

# Comandi utili – operatori insiemistici

Unione di insiemi	<code>union</code>
Intersezione di insiemi	<code>inter</code>
Differenza tra insiemi	<code>diff</code>
Differenza simmetrica tra insiemi	<code>svmdiff</code>
Prodotto cartesiano tra insiemi	<code>cross</code>
Appartenenza ad un insieme	<code>in</code>
Non appartenenza ad un insieme	<code>not in</code>

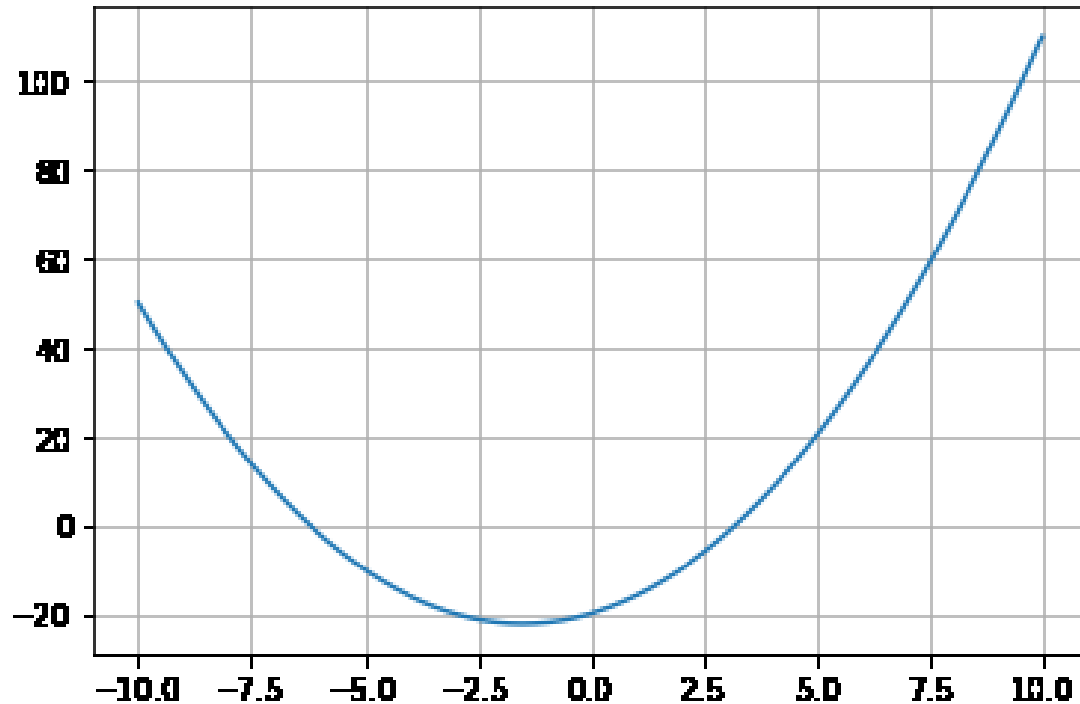
# Esercizio: mix produttivo

- 1. La società Merlin decide di produrre il nuovo fertilizzante prato felice (tipo C) che venderà a 23 €/Kg. La composizione del prato felice è 30% Azoto, 35% Fosforo, 5% Potassio e 30% Magnesio. Qual è il nuovo mix-produttivo ottimo?
- 2. La miscelazione dei 3 fertilizzante richiede rispettivamente 5 min/kg per il tipo A, 7 min/kg per il tipo B e 3 min/kg per il tipo C. Qual è la produzione ottimale se gli impianti sono disponibili per 8 ore?

# Esercizio: minimo funzione

- Si vuole determinare il punto di minimo della funzione

$$y = x^2 + 3x - 20$$



# Esercizio: minimo funzione







- Si vuole determinare la soluzione del seguente sistema lineare

$$\begin{bmatrix} 3 & -3 & 1 \\ -1 & 1 & 2 \\ 2 & 1 & -3 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

# Esercizio: lo schieramento più forte

- Dispongo di 12 giocatori di pallavolo. Considerando le abilità personali in ciascun ruolo (esemplificate dagli score riportati in tabella), qual è la squadra (composta da 6 giocatori) più forte che posso schierare?

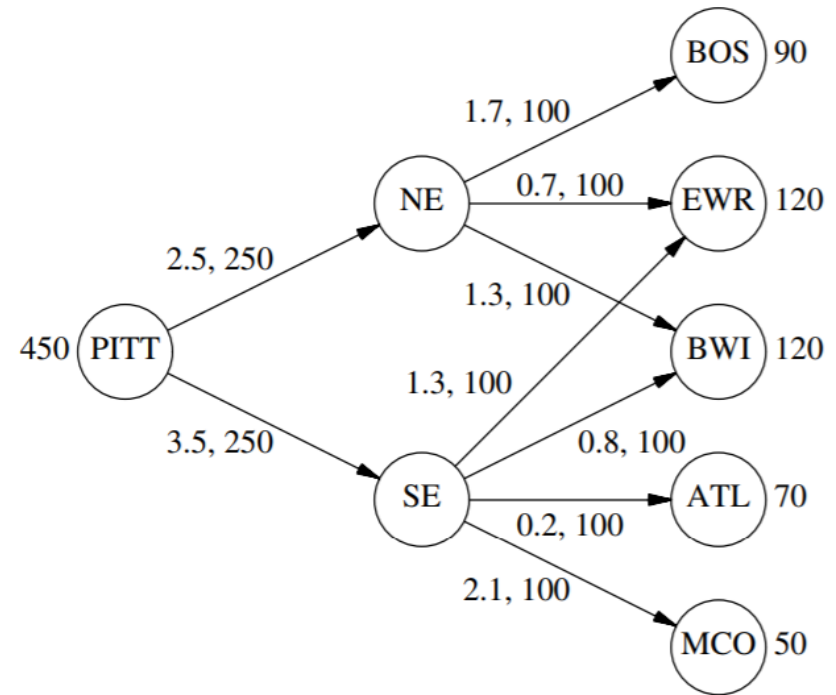


	A	B	C	D	E	F	G	H	M	N	P	Q
 • Palleggiatore	1	2	1	3	1	4	2	3	3	2	2	3
 • Martello 1	2	3	3	2	3	1	2	2	3	2	1	1
 • Centrale 2	3	2	3	3	1	1	2	3	2	2	1	1
 • Opposto	2	1	3	2	3	4	1	3	3	2	3	3
 • Martello 2	1	2	2	2	2	1	2	1	2	3	3	3
 • Centrale 1	4	3	4	3	1	3	1	1	1	3	2	1



# Esercizio: min-cost transshipment problem

- Il grafo orientato riporta una rete distributiva che collega l'impianto produttivo di PITT ai depositi di BOS, EWR, BWI, ATL e MCO, passando attraverso i centri di distribuzione di NE e SE. Gli archi sono etichettati con il costo di spedizione per unità di prodotto e la capacità di trasporto massima. Le 450 unità di prodotto di PITT devono essere trasportate e ripartite tra i depositi al fine di soddisfare la domanda di ciascuno di essi (e.g., 90 da BOS, 120 da EWR). Implementare in AMPL un modello di PL/PLI in grado di individuare la soluzione ottima del problema.



*In AMPL: A Modeling Language for  
Mathematical Programming, pag. 320*

# Esercizi

Per ognuno dei seguenti problemi, definire un modello di PL/PLI, codificarlo in AMPL e determinare una soluzione ottima

- Con l'acqua alla gola
- Un problema di trasporto
- Un problema di trasporto (2)
- Il quadrato magico

Esercizi tratti dall'eserciziario  
«Esercizi di Programmazione Matematica»  
(accessibile dalla pagina moodle del corso)

# Bibliografia e contenuti multimediali

1. AMPL: A Modeling Language for Mathematical Programming, *Robert Fourer, David M. Gay, e Brian W. Kernighan*, 2003
2. AMPL Tutorial, *Tulia Herrera*



- [AMPL – Download & Installation](#)
- [AMPL – Quick Start Guide](#)
- [AMPL – Model and Data Separation](#)
- [AMPL – NEOS server](#)