Ricerca Operativa - Laboratorio

Lezione 1 - Introduzione ad AMPL

Docente: Luigi De Giovanni

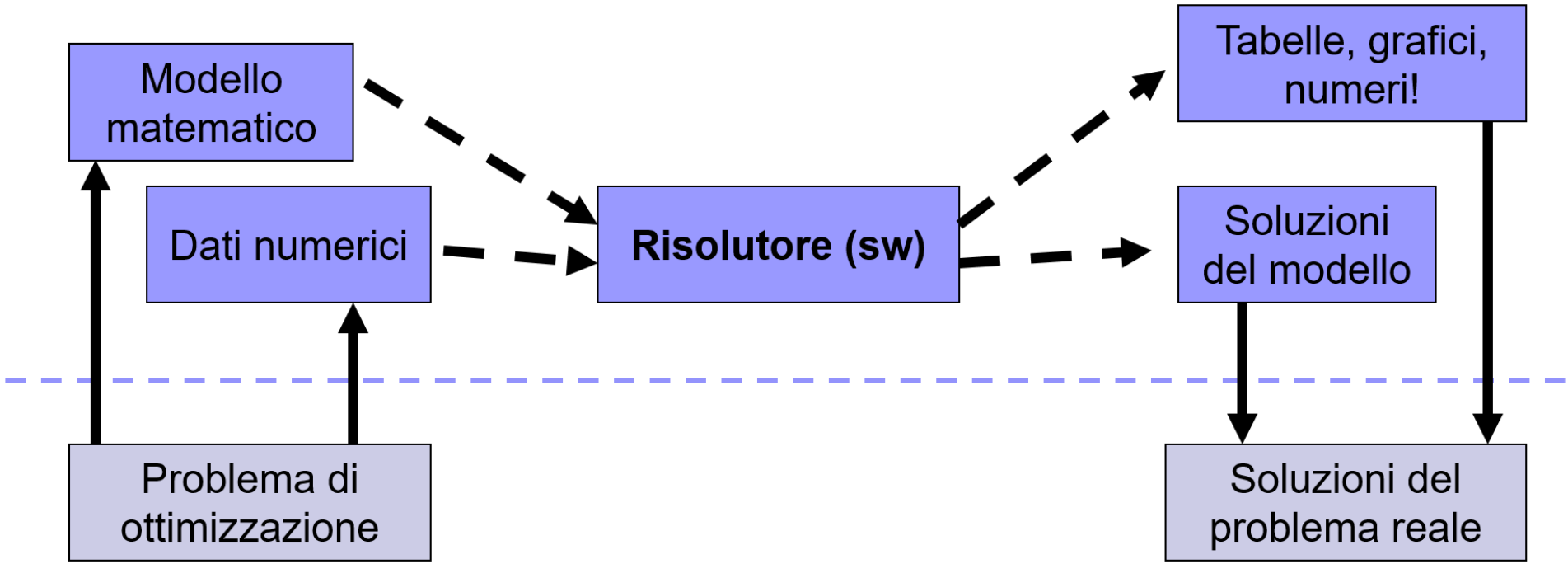
Dipartimento di Matematica “Tullio Levi-Civita”
Università degli Studi di Padova

luigi@math.unipd.it
https://www.math.unipd.it/∼luigi/

Corso di Laurea Magistrale in Matematica
Università degli Studi di Padova
a.a. 2019–2020

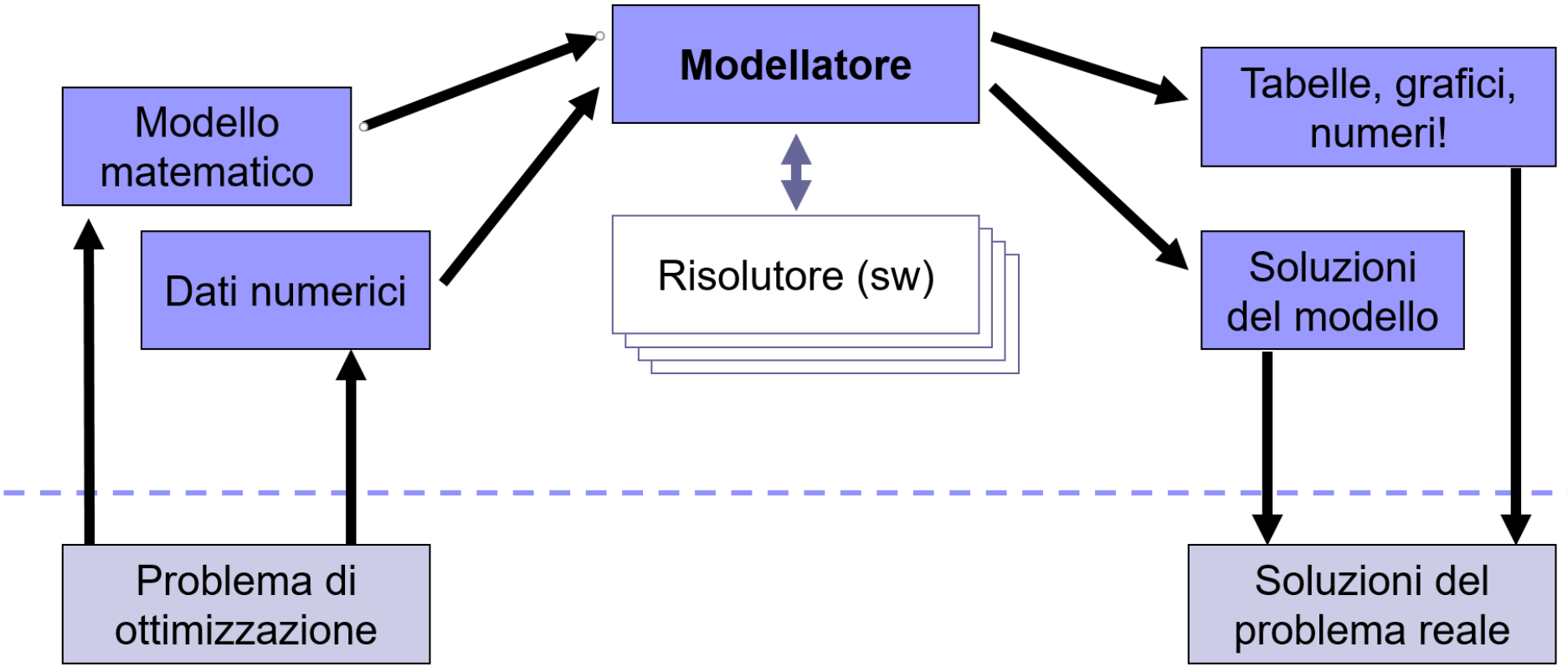
Strumenti per la soluzione di modelli di programmazione matematica (i)

Un **solver** (o risolutore) è un software che riceve in input un modello che descrive un problema di ottimizzazione e produce in output la soluzione ottima del modello (e informazioni ad essa collegate).



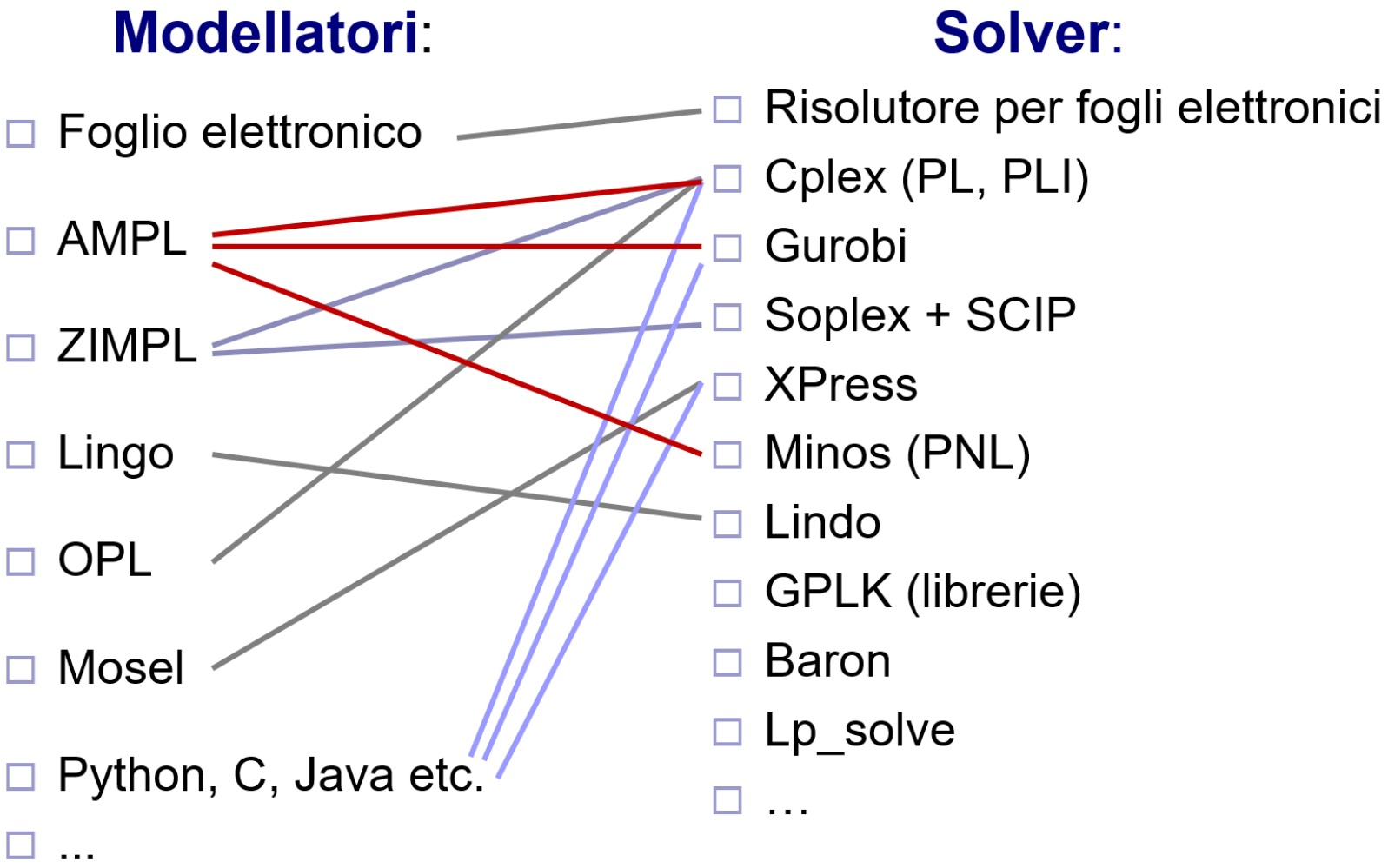
Strumenti per la soluzione di modelli di programmazione matematica (ii)

Un **modellatore** fornisce un'interfaccia tra il modello del problema e il risolutore.



Strumenti per la soluzione di modelli di programmazione matematica (iii)

Alcune possibili configurazioni



AMPL

A Mathematical Programming Language (AMPL)

- generatore algebrico di modelli: la sintassi riproduce il linguaggio algebrico
- interfaccia tra il modello algebrico e il solutore numerico

AMPL

A Mathematical Programming Language (AMPL)

- generatore algebrico di modelli: la sintassi riproduce il linguaggio algebrico
- interfaccia tra il modello algebrico e il solutore numerico

Testo di riferimento:

- Robert Fourer, David M. Gay, Brian W. Kernighan. *AMPL A Modeling Language For Mathematical Programming*, Second Edition, Duxbury Thomson, 2003
  disponibile sul sito di AMPL al link
  `https://ampl.com/resources/the-ampl-book`
- Una sintesi è disponibile sulla pagina del corso

Installazione AMPL (i)

Licenze AMPL disponibili:

- "commerciale" (licenze a pagamento "business" o "academic")
- "learning" gratuita (tempo limitato, soggetta a autorizzazione)
- "free demo" (fino a 500 vincoli e 500 [300 NL] variabili)

Installazione AMPL (i)

Licenze AMPL disponibili:

- "commerciale" (licenze a pagamento "business" o "academic")
- "learning" gratuita (tempo limitato, soggetta a autorizzazione)
- "free demo" (fino a 500 vincoli e 500 [300 NL] variabili)

Versioni AMPL "free demo"

- da linea di comando oppure con interfaccia essenziale (amplide)
- per diverse piattaforme (windows, linux, macOS)

Installazione AMPL (i)

Licenze AMPL disponibili:

- "commerciale" (licenze a pagamento "business" o "academic")
- "learning" gratuita (tempo limitato, soggetta a autorizzazione)
- "free demo" (fino a 500 vincoli e 500 [300 NL] variabili)

Versioni AMPL "free demo"

- da linea di comando oppure con interfaccia essenziale (amplide)
- per diverse piattaforme (windows, linux, macOS)

Installazione AMPL con Integrated Development Environment (AMPLIDE)

- (e.g. per linux) scaricare da `https://ampl.com/try-ampl/download-a-free-demo/` il file `amplide.linux32.tgz` o `amplide.linux64.tgz`
- estrarre l'archivio in una directory

Installazione AMPL (ii)

AMPL IDE download for Linux

To install: Download one of the distribution archives amplide.linux32.tgz for a 32-bit version or amplide.linux64.tgz for a 64-bit version. Then extract the contents of this archive by typing the command

```
tar xzf amplide.linux32.tgz
```

or

```
tar xzf amplide.linux64.tgz
```

When the extraction is complete you will see a directory named `amplide.linux32` or `amplide.linux64`. This will be your *AMPL folder;* optionally you may rename it and you may move it to any convenient location on your computer.

To run: Inside your AMPL directory, you will find a directory named `amplide`. Use the `cd` command to make `amplide` your current directory, and then start AMPL IDE with the command `./amplide`. A small "AMPL IDE" window will appear while the program is being loaded, and then the full IDE application window will open. To get started using AMPL IDE, choose **Help Contents** from the **Help** menu at the top of the application window.

AMPL Command Line download for Linux

To install: Download one of the distribution archives, ampl.linux32.tgz for a 32-bit version or ampl.linux64.tgz for a 64-bit version. Then extract the contents of this archive by typing the command

```
tar xzf ampl.linux32.tgz
```

or

```
tar xzf ampl.linux64.tgz
```

When the extraction is complete you will see a directory named `ampl.linux32` or `ampl.linux64`. This will be your *AMPL folder;* optionally you may rename it and you may move it to any convenient location on your computer.

To run: In a command window, use `cd` to go to your AMPL directory, and type `./ampl` at the system prompt. Then you will see an ampl: prompt and can proceed to type AMPL commands. By default, AMPL `model` and `data` statements will refer to files that you have saved in your AMPL folder; you can instead read from the

Installazione AMPL (iii)

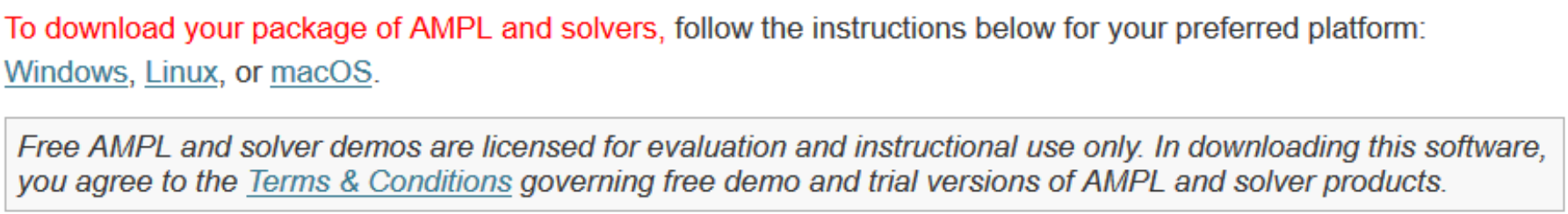
- L'archivio estratto contiene tutti i file necessari, tra cui gli eseguibili di diversi solver: per esempio CPLEX, GUROBI, MINOS, LP_SOLVE, ecc. (versioni gratuite limitate)
- I passi dell'installazione sotto Windows sono analoghi
- Il file eseguibile della IDE di AMPL si trova nella cartella *amplide*, dalla quale eseguire `./amplide` in Linux (`amplide.exe` in Windows)

Installazione AMPL (iii)

- L'archivio estratto contiene tutti i file necessari, tra cui gli eseguibili di diversi solver: per esempio CPLEX, GUROBI, MINOS, LP_SOLVE, ecc. (versioni gratuite limitate)
- I passi dell'installazione sotto Windows sono analoghi
- Il file eseguibile della IDE di AMPL si trova nella cartella *amplide*, dalla quale eseguire `./amplide` in Linux (`amplide.exe` in Windows)
- Esiste una versione di AMPL a linea di comando: rende disponibile un prompt di comandi in una finestra di sistema (eseguire `./ampl` o `ampl.exe`, bisogna usare un editor esterno dei file di testo, si consiglia l'uso di `sw.exe` per richiamare i comandi nel prompt)

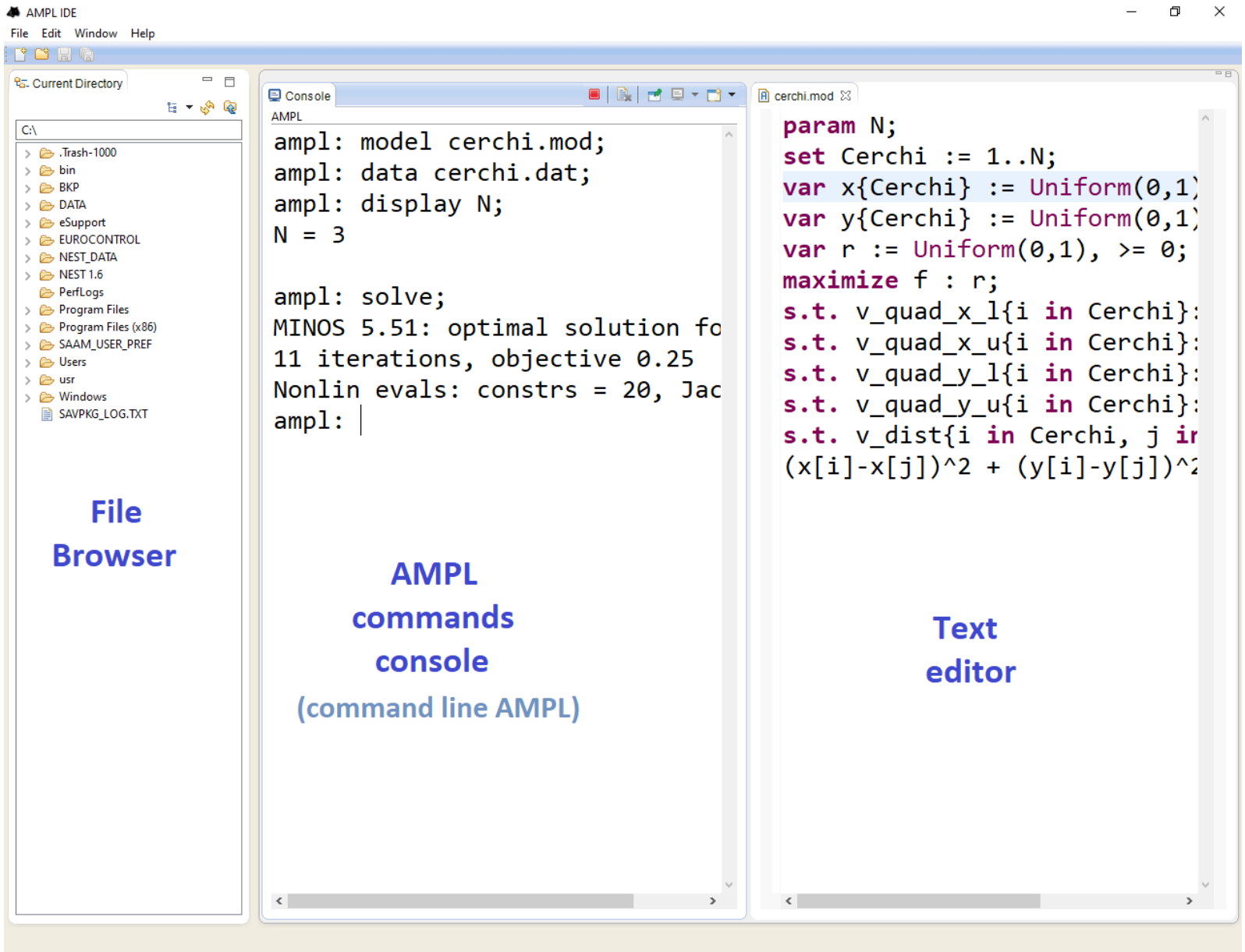
Installazione AMPL (iii)

- L'archivio estratto contiene tutti i file necessari, tra cui gli eseguibili di diversi solver: per esempio CPLEX, GUROBI, MINOS, LP_SOLVE, ecc. (versioni gratuite limitate)
- I passi dell'installazione sotto Windows sono analoghi
- Il file eseguibile della IDE di AMPL si trova nella cartella *amplide*, dalla quale eseguire `./amplide` in Linux (`amplide.exe` in Windows)
- Esiste una versione di AMPL a linea di comando: rende disponibile un prompt di comandi in una finestra di sistema (eseguire `./ampl` o `ampl.exe`, bisogna usare un editor esterno dei file di testo, si consiglia l'uso di `sw.exe` per richiamare i comandi nel prompt)
- Leggere i termini della licenza ("evaluation and instructional use only")


To download your package of AMPL and solvers, follow the instructions below for your preferred platform: Windows, Linux, or macOS.

Free AMPL and solver demos are licensed for evaluation and instructional use only. In downloading this software, you agree to the Terms & Conditions governing free demo and trial versions of AMPL and solver products.

AMPL IDE


AMPL IDE
File Edit Window Help
Current Directory
C:\
.Trash-1000
bin
BKP
DATA
eSupport
EUROCONTROL
NEST_DATA
NEST 1.6
PerfLogs
Program Files
Program Files (x86)
SAAM_USER_PREF
Users
usr
Windows
SAVPKG_LOG.TXT
File Browser
Console
AMPL
ampl: model cerchi.mod;
ampl: data cerchi.dat;
ampl: display N;
N = 3
ampl: solve;
MINOS 5.51: optimal solution fo
11 iterations, objective 0.25
Nonlin evals: constrs = 20, Jac
ampl:
AMPL commands console
(command line AMPL)
cerchi.mod
param N;
set Cerchi := 1..N;
var x{Cerchi} := Uniform(0,1)
var y{Cerchi} := Uniform(0,1)
var r := Uniform(0,1), >= 0;
maximize f : r;
s.t. v_quad_x_l{i in Cerchi}:
s.t. v_quad_x_u{i in Cerchi}:
s.t. v_quad_y_l{i in Cerchi}:
s.t. v_quad_y_u{i in Cerchi}:
s.t. v_dist{i in Cerchi, j in
(x[i]-x[j])^2 + (y[i]-y[j])^2
Text editor

Principali comandi

- ogni comando termina con “;”

Principali comandi

- ogni comando termina con “;”
- si possono inserire commenti preceduti dal simbolo # (viene ignorato il testo fino alla fine della riga)

Principali comandi

- ogni comando termina con “;”
- si possono inserire commenti preceduti dal simbolo `#` (viene ignorato il testo fino alla fine della riga)
- per uscire dall’ambiente AMPL è sufficiente il comando `quit;`

Principali comandi

- ogni comando termina con “;”
- si possono inserire commenti preceduti dal simbolo `#` (viene ignorato il testo fino alla fine della riga)
- per uscire dall’ambiente AMPL è sufficiente il comando `quit;`

Principali comandi

- ogni comando termina con “;”
- si possono inserire commenti preceduti dal simbolo `#` (viene ignorato il testo fino alla fine della riga)
- per uscire dall’ambiente AMPL è sufficiente il comando `quit;`

- dichiarazione delle variabili con la parola chiave `var` seguita da un’etichetta

Principali comandi

- ogni comando termina con “;”
- si possono inserire commenti preceduti dal simbolo `#` (viene ignorato il testo fino alla fine della riga)
- per uscire dall’ambiente AMPL è sufficiente il comando `quit;`

- dichiarazione delle variabili con la parola chiave `var` seguita da un’etichetta
- dichiarazione della funzione obiettivo con la parola chiava `minimize` (oppure `maximize`) seguita da un’etichetta, da “:”, e da un’espressione algebrica

Principali comandi

- ogni comando termina con “;”
- si possono inserire commenti preceduti dal simbolo `#` (viene ignorato il testo fino alla fine della riga)
- per uscire dall’ambiente AMPL è sufficiente il comando `quit;`

- dichiarazione delle variabili con la parola chiave `var` seguita da un’etichetta
- dichiarazione della funzione obiettivo con la parola chiava `minimize` (oppure `maximize`) seguita da un’etichetta, da “:”, e da un’espressione algebrica
- dichiarazione dei vincoli con la parola chiave `subject to` (oppure `s.t.`), seguita da un’etichetta, da “:”, e dalla relativa espressione algebrica

Risolutori

Alcuni risolutori presenti nella versione AMPL student:

- CPLEX: problemi di programmazione lineare e problemi di programmazione quadratica convessi (continui e interi)
- GUROBI: problemi di programmazione lineare e problemi di programmazione quadratica convessi (continui e interi)
- LPSOLVE: problemi di programmazione lineare (continui e interi)
- MINOS: problemi di programmazione non lineare (continui)

Risolutori

Alcuni risolutori presenti nella versione AMPL student:

- CPLEX: problemi di programmazione lineare e problemi di programmazione quadratica convessi (continui e interi)
- GUROBI: problemi di programmazione lineare e problemi di programmazione quadratica convessi (continui e interi)
- LPSOLVE: problemi di programmazione lineare (continui e interi)
- MINOS: problemi di programmazione non lineare (continui)

Comandi relativi alla scelta e all'utilizzo del solutore:

- per scegliere quale risolutore utilizzare, si usa l'espressione `option solver` seguita dal nome del risolutore (il solutore di default è MINOS)
- per risolvere il problema, si usa l'espressione `solve`

Esempio (i)

$$\max \; x_1 + x_2$$
$$x_1 + x_2 \leq 1$$
$$x_1 - x_2 \leq 2$$
$$x_1, \, x_2 \geq 0$$

Esempio (i)

$$\max\ x_1 + x_2$$
$$x_1 + x_2 \leq 1$$
$$x_1 - x_2 \leq 2$$
$$x_1,\ x_2 \geq 0$$

Tramite prompt dei comandi:

```
ampl: var x1;
ampl: var x2;
ampl: maximize f: x1 + x2;
ampl: subject to v1: x1 + x2 <= 1;
ampl: s.t. v2: x1 - x2 <= 2;
ampl: s.t. v3: x1 >= 0;
ampl: s.t. v4: x2 >= 0;
ampl: option solver cplex;
ampl: solve;
```

Esempio (ii)

Output:

```
CPLEX 12.8.0.0: optimal solution; objective 1;
0 dual simplex iterations (0 in phase I)
```

Per visualizzare il valore delle variabili all'ottimo:

```
ampl: display x1;
ampl: display x2;
```

Per visualizzare il valore della funzione obiettivo all'ottimo:

```
ampl: display f;
```

File di modello `.mod` (i)

Utilizzando la linea di comando, il modello viene automaticamente cancellato uscendo da AMPL. È conveniente scrivere il modello in un file di testo, che deve avere estensione `.mod`

File di modello `.mod` (i)

Utilizzando la linea di comando, il modello viene automaticamente cancellato uscendo da AMPL. È conveniente scrivere il modello in un file di testo, che deve avere estensione `.mod`

esempio.mod

```
var x1;
var x2;

maximize f: x1 + x2;

subject to v1: x1 + x2 <= 1;
s.t. v2: x1 - x2 <= 2;
s.t. v3: x1 >= 0;
s.t. v4: x2 >= 0;
```

File di modello `.mod` (ii)

Assumendo di aver salvato il file del modello `esempio.mod` in ~/MODELLI [C:\MODELLI], per leggere il file è sufficiente scrivere dal prompt di AMPL il comando `model`

```
ampl: model ~/MODELLI/esempio.mod;
[ampl: model C:\MODELLI\esempio.mod;]
```

Per evitare di dover indicare il percorso completo, si può utilizzare

```
ampl: cd ~/MODELLI;
[ampl: cd C:\MODELLI;]
```

Si può utilizzare il "File Browser" dell'IDE per selezionare la cartella di lavoro (in alternativa a `cd`)

File di script `.run` (i)

Si possono creare file di testo con i comandi da eseguire (script). Tali file devono avere estensione `.run`.

File di script `.run` (i)

Si possono creare file di testo con i comandi da eseguire (script). Tali file devono avere estensione `.run`.

esempio.run

```
reset;
model C:\MODELLI\esempio.mod;
option solver cplex;
solve;
display x1;
display x2;
display f;
```

N.B. Il comando `reset` ha la funzione di eliminare da AMPL i dati relativi ad un modello caricato precedentemente. È conveniente introdurlo all'inizio di ogni file `.run`.

File di script `.run` (ii)

Per lanciare un file `.run`:

- nella console di AMPL si utilizza il comando `include esempio.run`
- fuori dall'ambiente AMPL o dall'IDE, è sufficiente aprire un terminale del prompt dei comandi di sistema, (spostarsi nella cartella contenente l'eseguibile `ampl`, se questa cartella non è inclusa nel path) e dare il comando `./ampl [<percorso/>]esempio.run`

Esempio modello PL (i)

Esempio 1

Un dietologo deve preparare una dieta, scegliendo tra cibi a base di

- verdura: apporto di 5 mg/kg di proteine, 6 mg/kg di ferro e 5 mg/kg di vitamine, al costo di 4 euro/kg;
- carne: apporto di 15 mg/kg di proteine, 10 mg/kg di ferro e 3 mg/kg di vitamine, al costo di 10 euro/kg;
- frutta: apporto di 4 mg/kg di proteine, 5 mg/kg di ferro e 12 mg/kg di vitamine, al costo di 7 euro/kg.

Determinare la dieta di costo minimo che assicuri un apporto giornaliero di proteine, ferro e vitamine di almeno 20 mg, 30 mg e 30 mg, rispettivamente.

Esempio modello PL (ii)

$$\begin{aligned}
&\min\ 4x_1 + 10x_2 + 7x_3 \\
&5x_1 + 15x_2 + 4x_3 \geq 20 \\
&6x_1 + 10x_2 + 5x_3 \geq 30 \\
&5x_1 + 3x_2 + 12x_3 \geq 30 \\
&x_1 \geq 0,\ x_2 \geq 0,\ x_3 \geq 0
\end{aligned}$$

Esempio modello PL (iii)

dieta.mod

```
var x1;        # kg di verdura
var x2;        # kg di carne
var x3;        # kg di frutta

minimize f: 4*x1 + 10*x2 + 7*x3;

s.t. v_proteine: 5*x1 + 15*x2 + 4*x3 >= 20;
s.t. v_ferro: 6*x1 + 10*x2 + 5*x3 >= 30;
s.t. v_vitamine: 5*x1 + 3*x2 + 12*x3 >= 30;
s.t. v_nonneg1: x1 >= 0;
s.t. v_nonneg2: x2 >= 0;
s.t. v_nonneg3: x3 >= 0;
```

Esempio modello PL (iv)

dieta.run

```
reset;

model dieta.mod;

option solver cplex;
solve;

display f;
display x1;
display x2;
display x3;
```

Dichiarazione variabili

La non-negatività delle variabili può essere espressa direttamente nella dichiarazione, che permette di specificare un limite inferiore (lower bound lb) e un limite superiore (upper bound ub)

```
var nameVar [>= <lb>] [, <= <lb>];
```

Nell'esempio, usando

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

si possono eliminare i vincoli `v_nonneg...`

Esempio modello PL: estensioni

Modificare opportunamente i file relativi al problema per determinare:

- la variazione del costo se si vogliono assumere al più di 3 kg di verdura e almeno 1 kg di frutta

Esempio modello PL: estensioni

Modificare opportunamente i file relativi al problema per determinare:

- la variazione del costo se si vogliono assumere al più di 3 kg di verdura e almeno 1 kg di frutta
- l'impatto dell'inserimento nella dieta di almeno 500 grammi (o più, se conviene) di alimenti a base di pesce che apportano 10 mg/kg di proteine, 15 mg/kg di ferro e 2 mg/kg di vitamine, al costo di 9 euro/kg

Esercizio: portfolio optimization

Un cliente affida ad un'agenzia finanziaria 100 000 euro da impiegare in fondi di investimento. I fondi attualmente offerti dal mercato sono di cinque tipi, come riassunto in tabella:

Nome	Tipo	Moody's rating	Durata (anni)	Rendita alla maturazione
A	privato	Aa	9	4,5%
B	pubblico	A	15	5,4%
C	stato	Aaa	4	5,1%
D	stato	Baa	3	4,4%
E	privato	Ba	2	6,1%

Si sa che i fondi pubblici e dello stato sono tassati del 30% alla fine del periodo. Il cliente chiede di riservare almeno il 40% del capitale a fondi pubblici e dello stato e vuole che la durata media dell'investimento non superi i 5 anni. Inoltre, trasformando il Moody's rating in una scala numerica (Aaa = 1, Aa = 2, A = 3, Baa = 4 e Ba = 5), il valore medio del rischio dell'investimento non deve superare 1,5. Si vuole massimizzare la rendita finale dell'investimento.

Esercizio: portfolio optimization (soluzione)

$$
\begin{aligned}
\max \quad & 4,5x_A + 0,7 \cdot 5,4x_B + 0,7 \cdot 5,1x_C + 0,7 \cdot 4,4x_D + 6,1x_E \\
& s.t. \\
& x_A + x_B + x_C + x_D + x_E \leq 100\,000 \\
& 2x_A + 3x_B + x_C + 4x_D + 5x_E \leq 1,5(x_A + x_B + x_C + x_D + x_E) \\
& 9x_A + 15x_B + 4x_C + 3x_D + 2x_E \leq 5(x_A + x_B + x_C + x_D + x_E) \\
& x_B + x_C + x_D \geq 40\,000 \\
& x_i \in \mathbb{R}_+, \forall i \in \{A, B, C, D, E\}
\end{aligned}
$$

Esercizio: agricoltura

Un'azienda agricola produce mais, soia e grano in tre tenute A, B e C. La tenuta A dispone di 600 ettari di terreno e di una riserva d'acqua di $8 \times 10^6 mc$. La tenuta B dispone di 700 ettari di terreno e di $5 \times 10^6 mc$ d'acqua. La tenuta C dispone di 450 ettari di terreno e di $6 \times 10^6 mc$ d'acqua. La resa economica di ogni ettaro di terreno è di 5, 6 e 7 migliaia di euro per le produzioni di mais, soia e grano, rispettivamente. Ogni ettaro di terreno consuma acqua per 20000, 10000 e 10000 mc se coltivato rispettivamente a mais, soia o grano. Le direttive della comunità europea impongono che l'estensione complessiva del terreno coltivato a soia dall'azienda non superi il 40% del totale del suolo coltivato. L'azienda vuole massimizzare la resa economica delle tre tenute.

Esercizio: agricoltura (soluzione)

$$
\begin{aligned}
\max \quad & [5(x_{AM} + x_{BM} + x_{CM}) + 7(x_{AS} + x_{BS} + x_{CS}) \\
& \qquad +6(x_{AG} + x_{BG} + x_{CG})] \\
& s.t. \\
\text{(Disponibilità acqua)} \quad & 20x_{AM} + 10x_{AS} + 10x_{AG} \leq 8000 \\
& 20x_{BM} + 10x_{BS} + 10x_{BG} \leq 5000 \\
& 20x_{CM} + 10x_{CS} + 10x_{CG} \leq 6000 \\
\text{(Disponibilità terreno)} \quad & x_{AM} + x_{AS} + x_{AG} \leq 600 \\
& x_{BM} + x_{BS} + x_{BG} \leq 700 \\
& x_{CM} + x_{CS} + x_{CG} \leq 450 \\
\text{(Massimo 40\% a soia)} \quad & x_{AS} + x_{BS} + x_{CS} \leq 0,4 \sum_{i,j} x_{ij} \\
\text{(Dominio)} \quad & x_{ij} \in \mathbb{R}_+, \forall i \in \{A, B, C\}, j \in \{M, S, G\}
\end{aligned}
$$

Esercizio: raffineria

Una raffineria produce benzina verde e benzina super a partire da due tipi di greggio A e B, usando tre impianti. Il primo impianto può produrre 2 barili di verde e 3 di super a partire da 4 barili di greggio di tipo A e 3 barili di greggio di tipo B. Il secondo impianto può produrre 4 barili di verde e 2 di super a partire da 3 barili di greggio di tipo A e 4 barili di greggio di tipo B. Il terzo impianto può produrre 2 barili di verde e 2 di super a partire da 3 barili di greggio di tipo A e 3 barili di greggio di tipo B. Gli impianti lavorano sempre con le proporzioni specificate. La benzina verde viene venduta a 120 euro al barile, la super a 150 euro al barile. Sono disponibili, per questo mese, 5000 barili di greggio di tipo A e 6000 di tipo B. Determinare la produzione che massimizza il profitto mensile.

Esercizio: raffineria (soluzione)

$$\begin{aligned}
\max \quad & 120(2x_1 + 4x_2 + 2x_3) + 150(3x_1 + 2x_2 + 2x_3) \\
& s.t. \\
\text{(Disponibilità greggio)} \quad & 4x_1 + 3x_2 + 3x_3 \leq 5000 \\
& 3x_1 + 4x_2 + 3x_3 \leq 6000 \\
& \\
& x_i \in \mathbb{R}_+, \forall i = 1..3
\end{aligned}$$