

AMPL: A Modeling Language for Mathematical Programming



AMPL is a comprehensive and powerful algebraic modeling language for linear and nonlinear optimization problems, with discrete or continuous variables.

Concise language using common notation and familiar concepts for modeling and solution analysis

Ideal for rapid prototyping and efficient use in production

Seamlessly connects to many solvers

Best-in-class model presolver and automatic differentiator

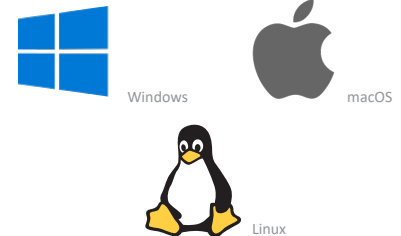
Develop faster than ever with AMPL... and focus on your real business problem!

AMPL's flexibility and convenience make it ideal for rapid prototyping and model development, while its speed and control options make it an efficient choice for repeated production runs.

Key features

- Broad support for sets and set operators:
 - tuples, ordered/circular sets, indexing sets
 - union, intersections, Cartesian products
- General and natural syntax for many problem classes
- Automatic data check using input data constraints
- Advanced nonlinear solver support: initial primal and dual values, automatic differentiation and elimination of «defined» variables, user-defined functions
- Automatic data check using input data constraints
- Easy connection to data sources such as text files, Excel, or SQL databases
- Extended scientific function library

OPERATING SYSTEMS



AMPL Book available online

www.artelys.com/en/composants-numeriques/ampl

Try AMPL for free

www.artelys.com/en/composants-numeriques/ampl

Artelys distributes and supports AMPL worldwide.

Developed by AMPL Optimization LLC.



Classic big-M formulation



*Logic constraint formulation
with no big-M constants*



*Complementarity formulation
with no binary variables*

Problem classes handled by AMPL

- Standard mathematical programming problems (LP, QP, MIP, NLP, MINLP)

```
var x{I} >= 0 <= 1;    var y binary;
minimize Cost: sum{(i,j) in I cross I} c[i,j]*x[i]*x[j] + f*y;
subject to BigM: sum{i in I} a[i]*x[i] >= b - M*y;
```

- Problems with logic constraints

```
var x{I} >= 0 <= 1;    var y binary;
minimize Cost: sum{(i,j) in I cross I} c[i,j]*x[i]*x[j] + f*y;
subject to Logic: y = 0 ==> sum{i in I} a[i]*x[i] >= b;
```

- Problems with complementarity constraints (MPEC, MPCC, MCP)

```
var x{I} >= 0 <= 1;    var y >= 0 <= 1;    var slack;
minimize Cost: sum{(i,j) in I cross I} c[i,j]*x[i]*x[j] + f*y;
subject to Linear: sum{i in I} a[i]*x[i] + slack >= b;
subject to Compl: 0 <= slack complements (1-y) >= 0;
```

- Problems with piecewise linear functions

```
var x{I} >= 0 <= 1;    var y binary;
minimize Cost: f*y +
  sum{(i,j) in I cross I} << 0.5; 0.5*c[i,j], 1.5*c[i,j] >> x[i]*x[j];
subject to BigM: sum{i in I} a[i]*x[i] >= b - M*y;
```

- Network problems expressed using nodes and arcs concepts

- LP problems expressed in matrix form

Artelys USA

150 N. Michigan, Suite 800
Chicago, IL 60601 - USA
+1 (312) 588-3376

Artelys France

81 rue Saint-Lazare
75009 Paris - France
+33 (0)1 44 77 89 00

Artelys Canada

3 Place Ville-Marie, Suite 400
H3B 2E3 Montréal (QC) - Canada
+1 438 239 7736