New and Forthcoming Developments in the AMPL Modeling Language & System

Robert Fourer, David M. Gay

AMPL Optimization
www.ampl.com — 773-336-AMPL

INFORMS Conference on Business Analytics and Operations Research

Huntington Beach, California — 15-17 April 2012
Track 22, *Software Tutorials*

Outline

Motivation

- The optimization modeling cycle
- Optimization modeling languages
- Introductory example

The AMPL company

AMPL's users

- Commercial, government, research & teaching
- Two Edelman competition cases

Future directions

- More powerful interfaces
- More natural modeling
  - Logical conditions
  - Quadratic constraints

The Optimization Modeling Cycle

Steps

- Communicate with problem owner
- Build model
- Prepare data
- Generate optimization problem
- Submit problem to solver
  - CPLEX, Gurobi, KNITRO, CONOPT, MINOS, . . .
- Report & analyze results
- ***Repeat!***

Goals

- Do this quickly and reliably
- Get results before client loses interest
- Deploy for application

What Makes This Hard?

"We do not feel that the linear programming user's most pressing need over the next few years is for a new optimizer that runs twice as fast on a machine that costs half as much (although this will probably happen). Cost of optimization is just not the dominant barrier to LP model implementation.

"The process required to manage the data, formulate and build the model, report on and analyze the results costs far more, and is much more of a barrier to effective use of LP, than the cost/performance of the optimizer."

Krabek, Sjoquist, Sommer,
"The APEX Systems: Past and Future."
***SIGMAP Bulletin* 29 (April 1980) 3-23.**

Optimization Modeling Languages

Two forms of an optimization problem

- Modeler's form
  - Mathematical description, easy for people to work with
- Algorithm's form
  - Explicit data structure, easy for solvers to compute with

Idea of a modeling language

- A computer-readable modeler's form
  - You write optimization problems in a modeling language
  - Computers translate to algorithm's form for solution

Advantages of a modeling language

- Faster modeling cycles
- More reliable modeling and maintenance

Algebraic Modeling Languages

Formulation concept

- Define data in terms of sets & parameters
  - Analogous to database keys & records
- Define decision variables
- Minimize or maximize a function of decision variables
- Subject to equations or inequalities
  that constrain the values of the variables

Advantages

- Familiar
- Powerful
- Implemented

The AMPL Modeling Language

Features

- Algebraic modeling language
- Variety of data sources
- Connections to all solver features
- Interactive and scripted control

Advantages

- Powerful, general expressions
- Natural, easy-to-learn design
- Efficient processing scales well with problem size

Introductory Example

Multicommodity transportation . . .

- Products available at factories
- Products needed at stores
- Plan shipments at lowest cost

. . . with practical restrictions

- Cost has fixed and variables parts
- Shipments cannot be too small
- Factories cannot serve too many stores

Multicommodity Transportation

Given

O Set of origins (factories)

D Set of destinations (stores)

P Set of products

and

a_{ip} Amount available, for each $i \in O$ and $p \in P$

b_{jp} Amount required, for each $j \in D$ and $p \in P$

l_{ij} Limit on total shipments, for each $i \in O$ and $j \in D$

c_{ijp} Shipping cost per unit, for each $i \in O$, $j \in D$, $p \in P$

d_{ij} Fixed cost for shipping any amount from $i \in O$ to $j \in D$

s Minimum total size of any shipment

n Maximum number of destinations served by any origin

Mathematical Formulation

Determine

X_{ijp} Amount of each $p \in P$ to be shipped from $i \in O$ to $j \in D$

Y_{ij} 1 if any product is shipped from $i \in O$ to $j \in D$
0 otherwise

to minimize

$$\sum_{i\in O} \sum_{j\in D} \sum_{p\in P} c_{ijp} X_{ijp} + \sum_{i\in O} \sum_{j\in D} d_{ij} Y_{ij}$$

Total variable cost plus total fixed cost

Multicommodity Transportation

Mathematical Formulation

Subject to

$\sum_{j \in D} X_{ijp} \leq a_{ip}$ for all $i \in O,\, p \in P$

Total shipments of product p out of origin i
must not exceed availability

$\sum_{i \in O} X_{ijp} = b_{jp}$ for all $j \in D,\, p \in P$

Total shipments of product p into destination j
must satisfy requirements

Mathematical Formulation

Subject to

$\sum_{p \in P} X_{ijp} \leq l_{ij} Y_{ij}$ for all $i \in O,\, j \in D$

When there are shipments from origin i to destination j, the total may not exceed the limit, and Y_{ij} must be 1

$\sum_{p \in P} X_{ijp} \geq s Y_{ij}$ for all $i \in O,\, j \in D$

When there are shipments from origin i to destination j, the total amount of shipments must be at least s

$\sum_{j \in D} Y_{ij} \leq n$ for all $i \in O$

Number of destinations served by origin i must be as most n

AMPL Formulation

Symbolic data

```
set ORIG;   # origins
set DEST;   # destinations
set PROD;   # products

param supply {ORIG,PROD} >= 0;  # availabilities at origins
param demand {DEST,PROD} >= 0;  # requirements at destinations
param limit {ORIG,DEST} >= 0;   # capacities of links

param vcost {ORIG,DEST,PROD} >= 0; # variable shipment cost
param fcost {ORIG,DEST} > 0;       # fixed usage cost

param minload >= 0;             # minimum shipment size
param maxserve integer > 0;     # maximum destinations served
```

Multicommodity Transportation

AMPL Formulation

Symbolic model: variables and objective

```
var Trans {ORIG,DEST,PROD} >= 0;   # actual units to be shipped
var Use {ORIG, DEST} binary;       # 1 if link used, 0 otherwise

minimize Total_Cost:
   sum {i in ORIG, j in DEST, p in PROD} vcost[i,j,p] * Trans[i,j,p]
 + sum {i in ORIG, j in DEST} fcost[i,j] * Use[i,j];
```

$$\sum_{i\in O}\sum_{j\in D}\sum_{p\in P} c_{ijp}\, X_{ijp} + \sum_{i\in O}\sum_{j\in D} d_{ij}\, Y_{ij}$$

AMPL Formulation

Symbolic model: constraint

```
subject to Supply {i in ORIG, p in PROD}:
   sum {j in DEST} Trans[i,j,p] <= supply[i,p];
```

$$\sum_{j \in D} X_{ijp} \leq a_{ip}, \text{ for all } i \in O, p \in P$$

Multicommodity Transportation

AMPL Formulation

Symbolic model: constraints

```
subject to Supply {i in ORIG, p in PROD}:
   sum {j in DEST} Trans[i,j,p] <= supply[i,p];

subject to Demand {j in DEST, p in PROD}:
   sum {i in ORIG} Trans[i,j,p] = demand[j,p];

subject to Multi {i in ORIG, j in DEST}:
   sum {p in PROD} Trans[i,j,p] <= limit[i,j] * Use[i,j];

subject to Min_Ship {i in ORIG, j in DEST}:
   sum {p in PROD} Trans[i,j,p] >= minload * Use[i,j];

subject to Max_Serve {i in ORIG}:
   sum {j in DEST} Use[i,j] <= maxserve;
```

Multicommodity Transportation

AMPL Formulation

Explicit data independent of symbolic model

```
set ORIG := GARY CLEV PITT ;
set DEST := FRA DET LAN WIN STL FRE LAF ;
set PROD := bands coils plate ;

param supply (tr):  GARY   CLEV   PITT :=
            bands    400    700    800
            coils    800   1600   1800
            plate    200    300    300 ;

param demand (tr):
           FRA   DET   LAN   WIN   STL   FRE   LAF :=
   bands   300   300   100    75   650   225   250
   coils   500   750   400   250   950   850   500
   plate   100   100     0    50   200   100   250 ;

param limit default 625 ;

param minload := 375 ;
param maxserve := 5 ;
```

AMPL Formulation

Explicit data (continued)

```
param vcost :=

 [*,*,bands]:  FRA  DET  LAN  WIN  STL  FRE  LAF :=
        GARY    30   10    8   10   11   71    6
        CLEV    22    7   10    7   21   82   13
        PITT    19   11   12   10   25   83   15

 [*,*,coils]:  FRA  DET  LAN  WIN  STL  FRE  LAF :=
        GARY    39   14   11   14   16   82    8
        CLEV    27    9   12    9   26   95   17
        PITT    24   14   17   13   28   99   20

 [*,*,plate]:  FRA  DET  LAN  WIN  STL  FRE  LAF :=
        GARY    41   15   12   16   17   86    8
        CLEV    29    9   13    9   28   99   18
        PITT    26   14   17   13   31  104   20 ;

param fcost:   FRA  DET  LAN  WIN  STL  FRE  LAF :=
        GARY  3000 1200 1200 1200 2500 3500 2500
        CLEV  2000 1000 1500 1200 2500 3000 2200
        PITT  2000 1200 1500 1500 2500 3500 2200 ;
```

AMPL Solution

Model + data = problem instance to be solved

```
ampl: model multmipG.mod;
ampl: data multmipG.dat;

ampl: option solver gurobi;

ampl: solve;

Gurobi 4.6.0: optimal solution; objective 235625
404 simplex iterations
45 branch-and-cut nodes

ampl: display Use;

Use [*,*]

:    DET FRA FRE LAF LAN STL WIN    :=
CLEV   1   1   1   0   1   1   0
GARY   0   0   0   1   0   1   1
PITT   1   1   1   1   0   1   0
;
```

Multicommodity Transportation

AMPL Solution

Solver choice independent of model and data

```
ampl: model multmipG.mod;
ampl: data multmipG.dat;

ampl: option solver cplex;

ampl: solve;

CPLEX 12.4.0.0: optimal integer solution; objective 235625
394 MIP simplex iterations
41 branch-and-bound nodes

ampl: display Use;

Use [*,*]

:    DET FRA FRE LAF LAN STL WIN    :=
CLEV   1   1   1   0   1   1   0
GARY   0   0   0   1   0   1   1
PITT   1   1   1   1   0   1   0
;
```

AMPL Solution

Examine results

```
ampl: display {i in ORIG, j in DEST}
ampl?    sum {p in PROD} Trans[i,j,p] / limit[i,j];

:       DET     FRA     FRE     LAF     LAN     STL     WIN     :=
CLEV    1       0.6     0.88    0       0.8     0.88    0
GARY    0       0       0       0.64    0       1       0.6
PITT    0.84    0.84    1       0.96    0       1       0
;

ampl: display Max_Serve.body;

CLEV   5
GARY   3
PITT   5
;

ampl: display TotalCost,
ampl?    sum {i in ORIG, j in DEST} fcost[i,j] * Use[i,j];

TotalCost = 235625
sum {i in ORIG, j in DEST} fcost[i,j]*Use[i,j] = 27600
```

Multicommodity Transportation

AMPL “Sparse” Network

Indexed over sets of pairs and triples

```
set ORIG;   # origins
set DEST;   # destinations
set PROD;   # products

set SHIP within {ORIG,DEST,PROD};

            # (i,j,p) in SHIP ==> can ship p from i to j

set LINK = setof {(i,j,p) in SHIP} (i,j);

            # (i,j) in LINK ==> can ship some products from i to j

...........

var Trans {SHIP} >= 0;   # actual units to be shipped

var Use {LINK} binary;   # 1 if link used, 0 otherwise

minimize Total_Cost:

   sum {(i,j,p) in SHIP} vcost[i,j,p] * Trans[i,j,p]

 + sum {(i,j) in LINK} fcost[i,j] * Use[i,j];
```

AMPL Scripting

Script to test sensitivity to serve limit

```
model multmipG.mod;
data multmipG.dat;

option solver gurobi;

for {m in 7..1 by -1} {
   let maxserve := m;
   solve;
   if solve_result = 'infeasible' then break;
   display maxserve, Max_Serve.body;
}
```

AMPL Scripting

Run showing sensitivity to serve limit

```
ampl: include multmipServ.run;

Gurobi 4.6.0: optimal solution; objective 233150
maxserve = 7
CLEV 5   GARY 3   PITT 6

Gurobi 4.6.0: optimal solution; objective 233150
maxserve = 6
CLEV 5   GARY 3   PITT 6

Gurobi 4.6.0: optimal solution; objective 235625
maxserve = 5
CLEV 5   GARY 3   PITT 5

Gurobi 4.6.0: infeasible
```

Multicommodity Transportation

AMPL Scripting

Script to generate n best solutions

```
param nSols default 0;
param maxSols;

model multmipG.mod;
data multmipG.dat;

set USED {1..nSols} within {ORIG,DEST};

subject to exclude {k in 1..nSols}:

   sum {(i,j) in USED[k]} (1-Use[i,j]) +
   sum {(i,j) in {ORIG,DEST} diff USED[k]} Use[i,j] >= 1;

option solver gurobi;

repeat {
   solve;
   display Use;
   let nSols := nSols + 1;
   let USED[nSols] := {i in ORIG, j in DEST: Use[i,j] > .5};
} until nSols = maxSols;
```

AMPL Scripting

Run showing 3 best solutions

```
ampl: include multmipBest.run;

Gurobi 4.6.0: optimal solution; objective 235625

:     DET FRA FRE LAF LAN STL WIN    :=
CLEV   1   1   1   0   1   1   0
GARY   0   0   0   1   0   1   1
PITT   1   1   1   1   0   1   0 ;

Gurobi 4.6.0: optimal solution; objective 237125

:     DET FRA FRE LAF LAN STL WIN    :=
CLEV   1   1   1   1   0   1   0
GARY   0   0   0   1   0   1   1
PITT   1   1   1   0   1   1   0 ;

Gurobi 4.6.0: optimal solution; objective 238225

:     DET FRA FRE LAF LAN STL WIN    :=

CLEV   1   0   1   0   1   1   1
GARY   0   1   0   1   0   1   0
PITT   1   1   1   1   0   1   0 ;
```

The AMPL Company

AMPL history

The AMPL team

Recent developments

- ❖ Business
- ❖ Academic

AMPL Company

AMPL History

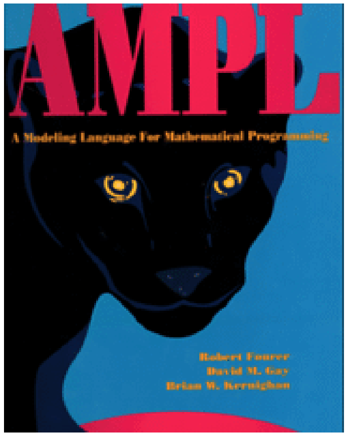
Origins

- AMPL developed at AT&T Bell Laboratories (1986)
  - Robert Fourer, David M. Gay, Brian W. Kernighan
- AMPL sold through distributors (1993)
- AMPL Optimization company formed (2002)

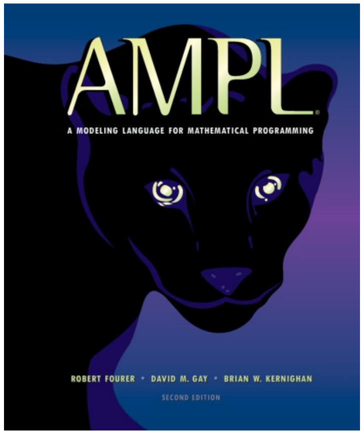
Writings

- "A Modeling Language for Mathematical Programming." *Management Science* **36** (1990) 519–554.
- The AMPL book

1993



2003



The AMPL Team

Robert Fourer

- Managing partner

David M. Gay

- Managing partner

William M. Wells

- Director of business development (joined 2010)

Victor Zverovich

- Optimization software development and support

Business Developments

AMPL intellectual property

- Full rights acquired from Alcatel-Lucent USA
  - corporate parent of Bell Laboratories
- More flexible licensing terms available

CPLEX & Gurobi for AMPL

- CPLEX sales transferred from IBM to AMPL Optimization
- Full lineup of licensing arrangements available

AMPL distributors

- New for Japan: *October Sky Co., Ltd.* →
- Others continue active
  - Gurobi, Ziena/Artelys
  - MOSEK, TOMLAB
  - OptiRisk

Academic Developments

Highly discounted prices for academic use

- AMPL
- Nonlinear solvers: KNITRO, MINOS, SNOPT, CONOPT

Free MIP solvers to academic users

- Gurobi & CPLEX
- 1-year licenses

Free AMPL & solvers for courses

- One-page application (`www.ampl.com/courses.html`)
- Single file for distribution to students
- Streamlined installation — no license file
- Expires when the course is over

AMPL's Users

Business

- Customer relationships
- Customer areas
- Project examples

Government

Academic

Business Customer Relationships

Internal projects

- We supply software & answer a few questions
- Company's employees build the models

Training and consulting

- Available on request

Business Customer Areas

Transportation

- Air, rail, truck

Production

- Planning
  - steel
  - automotive
- Supply chain
  - consumer products

Finance

- Investment banking
- Insurance

Natural resources

- Electric power
- Gas distribution
- Mining

Information technology

- Telecommunications
- Internet services

Consulting practices

- Management
- Industrial engineering

Business Customer Examples

Two award-winning projects

- ZARA (clothing retailing)
- Norske Skog (paper manufacturer)

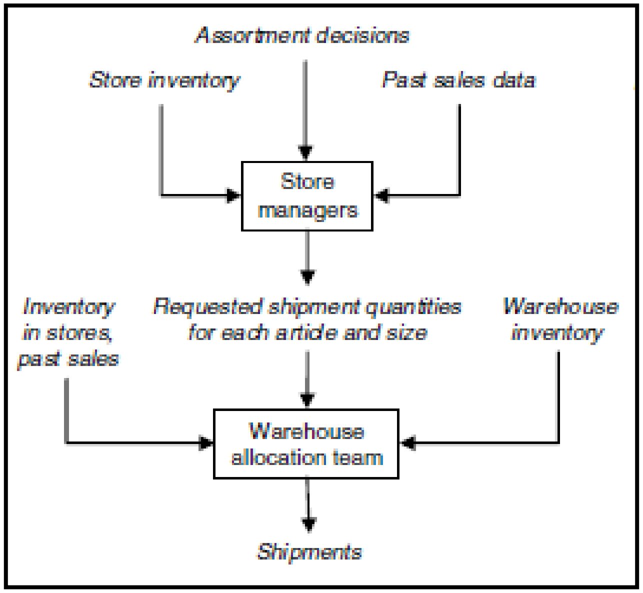
. . . finalists for Edelman Award
for practice of Operations Research

AMPL's Users

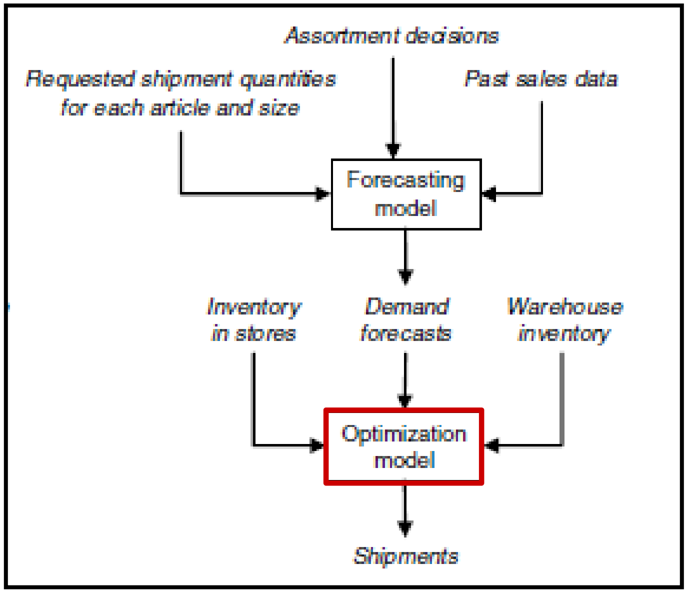
ZARA

Optimization of worldwide shipments

- Legacy process



New Process



- Piecewise-linear AMPL model with integer variables
- Run once for each product each week
- Decides how much of each size to ship to each store
- Increases sales 3-4%

ZARA's Formulation

Given

$S = S^{+} \cup S^{-}$

Set of sizes partitioned into major & regular sizes

J Set of stores

and

W_s Inventory of size s available at the warehouse

I_{sj} Inventory of size s available in store j

P_j Selling price in store j

K Aggressiveness factor (value of inventory remaining in the warehouse after the current shipments)

λ_{sj} Demand rate for size s in store j

N_{sj} Approximation set for size s in the inventory-to-sales function approximation for store j

ZARA's Formulation

Determine

x_{sj} (integer) shipment quantity of each size $s \in S$
to each store $j \in J$ for the current replenishment period

z_j (≥ 0) approximate expected sales across all sizes
in each store $j \in J$ for the current period

to maximize

$$\sum_{j \in J} P_j z_j + K \sum_{s \in S} (W_s - \sum_{j \in J} x_{sj})$$

Total sales plus value of items remaining in warehouse

subject to

$$\sum_{j \in J} x_{sj} \leq W_s \quad \text{for all } s \in S$$

Total shipments of size s
must not exceed amount available in warehouse

ZARA's Formulation

and subject to

$z_j \leq (\sum_{s \in S^+} \lambda_{sj}) y_j + \sum_{s \in S^-} \lambda_{sj} v_{sj}$ for all $j \in J$

$y_j \leq a_i \lambda_{sj} \left(I_{sj} + x_{sj} - i \right) + b_i \lambda_{sj}$ for all $j \in J,\ s \in S^+,\ i \in N_{sj}$

$v_{sj} \leq a_i \lambda_{sj} \left(I_{sj} + x_{sj} - i \right) + b_i \lambda_{sj}$ for all $j \in J,\ s \in S^-,\ i \in N_{sj}$

$v_{sj} \leq y_j$ for all $j \in J,\ s \in S^-$

Relationship between sales
and store inventory after shipments

AMPL's Users

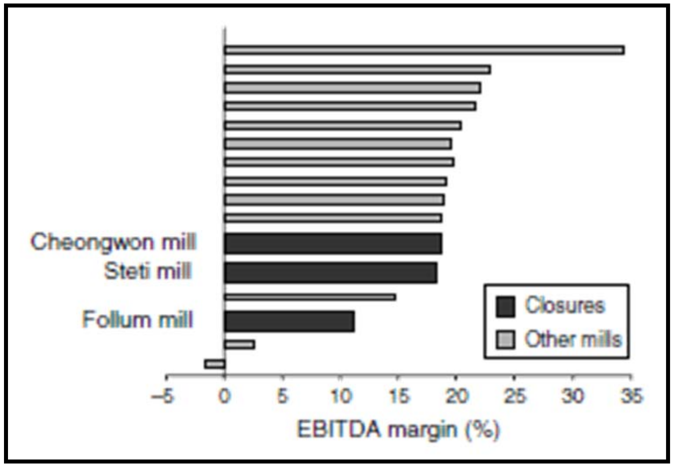
Norske Skog

Optimization of production and distribution

- Australasia
- Europe
  - 640 binary variables
  - 524,000 continuous variables
  - 33,000 constraints

Optimization of shutdown decisions worldwide

- Multiple scenarios
- Numerous sensitivity analyses
- ***Key role of AMPL models***
  - Implemented in a few weeks
  - Modified to analyze alternatives
  - Run interactively at meetings



Norske Skog's Formulation

Given sets

- N Number of mills
- M_n Set of machines at mill N
- M Number of paper machines
- J Number of products
- L Number of raw material sources
- R Number of raw materials
- K Number of customers
- P Number of recipes

Norske Skog's Formulation

Given capital parameters

l_n	fixed cost of mill n running for one period (excluding machine fixed costs)
f_m	fixed cost of machine m running for one period
θ_m	proportion of fixed running costs saved from a temporary shutdown on machine m
q_m	minimum time that machine m must be shut before savings accrue
ϕ_m	amortized cost of a permanent closure of machine m

Norske Skog's Formulation

and operating parameters

g_{mjk} variable freight cost for shipping product j from machine m to customer k

a_{mjp} capacity of machine m making product j using recipe p

c_{mjp} variable cost incurred by producing one tonne of product j using recipe p on machine m

h_{mjrp} tonnes of raw material r required to make one tonne of product j using recipe p on machine m

π_{mrl} procurement, transportation, and process cost of raw material r from source l for machine m

W_{rl} supply of raw material r at source l

d_{jk} demand for product j by customer k

s_{jk} sales price for product j by customer k

Norske Skog's Formulation

Make capital decisions

δ_n 1 if mill n closes, 0 otherwise

μ_m 1 if machine m shuts down permanently, 0 otherwise

u_m time that machine m has been shut down

ξ_m 1 if machine m has been shut down long enough to accrue savings, 0 otherwise

v_m time that qualifies for savings on machine m

Norske Skog's Formulation

and operating decisions

x_{mjp} tonnes of product j made on machine m using recipe p

y_{mjk} tonnes of product j made on machine m and delivered to customer k

w_{mrl} tonnes of raw material r from source l used by machine m

σ_{mp} 1 if recipe p is used on machine m, 0 otherwise

Norske Skog's Formulation

Maximize

$$\sum_{m=1}^{M}\sum_{j=1}^{J}\left(\sum_{k=1}^{K}(s_{jk} - g_{mjk})y_{mjk} - \sum_{p=1}^{P} c_{mjp}x_{mjp}\right)$$

$$- \sum_{m=1}^{M}\sum_{l=1}^{L}\sum_{r=1}^{R} \pi_{mrl}w_{mrl}$$

$$+ \sum_{m=1}^{M} \theta_m f_m v_m$$

$$- \sum_{n=1}^{N}(l_n(1 - \delta_n) + \lambda_n\delta_n)$$

$$- \sum_{m=1}^{M}(f_m(1 - \mu_m) + \phi_m\mu_m)$$

Income from sales,
minus raw material, production and distribution costs,
plus savings from shutdowns,
minus fixed operating and shutdown costs

Norske Skog's Formulation

Subject to

$\sum_{j=1}^{J} \sum_{p=1}^{P} \frac{x_{mjp}}{a_{mjp}} = 1 - u_m$ for $m = 1, \ldots, M$

Capacity used equals capacity available

$\sum_{k=1}^{K} y_{mjk} = \sum_{p=1}^{P} x_{mjp}$ for $j = 1, \ldots, J,\ m = 1, \ldots, M$

Amounts produced equal amounts shipped

$\sum_{m=1}^{M} y_{mjk} \leq d_{jk}$ for $j = 1, \ldots, J,\ k = 1, \ldots, K$

Amounts produced do not exceed demand

$\sum_{j=1}^{J} \sum_{p=1}^{P} h_{mjrp} x_{mjp} = \sum_{l=1}^{L} w_{mrl}$ for $m = 1, \ldots, M,\ r = 1, \ldots, R$

Raw material used equals raw material purchased

$\sum_{m=1}^{M} w_{mrl} \leq W_{rl}$ for $l = 1, \ldots, L,\ r = 1, \ldots, R$

Raw material purchased does not exceed amount available

Norske Skog's Formulation

and subject to

$\sum_{p=1}^{P} \sigma_{mp} = 1 - \mu_m$ for $m = 1, \ldots, M$

$x_{mjp} \leq a_{mjp}\sigma_{mjp}$ for $j = 1, \ldots, J,\ m = 1, \ldots, M,\ p = 1, \ldots, P$

$\delta_n \leq \mu_m$ for $m \in M_n,\ n = 1, \ldots, N$

$v_m \leq \xi_m$ for $m = 1, \ldots, M$

$v_m \leq 1 - \mu_m$ for $m = 1, \ldots, M$

$v_m \leq u_m - q_m\xi_m$ for $m = 1, \ldots, M$

Definitions of zero-one variables

Government Customers

Financial agencies

- United States
- Canada
- Sweden

U.S. departments

- Census Bureau
- Army Corps of Engineers

U.S. research centers

- Argonne National Laboratory
- Sandia National Laboratories
- Lawrence Berkeley Laboratory

Academic Customers

Research

- Over 250 university installations worldwide
- Nearly 1000 citations in scientific papers
  - engineering, science, economics, management

Teaching

- Linear & nonlinear optimization
  - Graph optimization
  - Stochastic programming
- Operations Research
- Specialized courses
  - Supply chain modeling
  - Electric power system planning
  - Transportation logistics
  - Communication network design & algorithms

Future Directions

More powerful interfaces

- Enhanced user interfaces (IDEs)
- Callable interfaces for deployment (APIs)

More natural modeling

- Logical conditions
- Quadratic constraints

Logical Conditions

Minimum-shipment constraints

- From each origin to each destination, *either* ship nothing *or* ship at least `minload` units

Conventional linear mixed-integer formulation

```
var Trans {ORIG,DEST,PROD} >= 0;

var Use {ORIG, DEST} binary;

.......

subject to Multi {i in ORIG, j in DEST}:

   sum {p in PROD} Trans[i,j,p] <= limit[i,j] * Use[i,j];

subject to Min_Ship {i in ORIG, j in DEST}:

   sum {p in PROD} Trans[i,j,p] >= minload * Use[i,j];
```

Zero-One Alternatives

Mixed-integer formulation using implications

```
subject to Multi {i in ORIG, j in DEST}:
   Use[i,j] = 0  ==>  sum {p in PROD} Trans[i,j,p] = 0;

subject to Min_Ship {i in ORIG, j in DEST}:
   Use[i,j] = 1  ==>
      minload <= sum {p in PROD} Trans[i,j,p] <= limit[i,j];
```

```
subject to Multi_Min_Ship {i in ORIG, j in DEST}:
   Use[i,j] = 1  ==>
      minload <= sum {p in PROD} Trans[i,j,p] <= limit[i,j]
   else sum {p in PROD} Trans[i,j,p] = 0;
```

. . . implemented in CPLEX

Non-Zero-One Alternatives

Disjunctive constraint

```
subject to Multi {i in ORIG, j in DEST}:
   sum {p in PROD} Trans[i,j,p] = 0  or
   minload <= sum {p in PROD} Trans[i,j,p] <= limit[i,j];
```

Discontinuous domain

```
var SumTrans {i in ORIG, j in DEST}
   in {0} union interval[minload,limit[i,j]];
.......
subject to Multi {i in ORIG, j in DEST}:
   SumTrans[i,j] = sum {p in PROD} Trans[i,j,p];
```

Quadratic Constraints

Given a traffic network

N Set of nodes representing intersections

e Entrance to network

f Exit from network

$A \subseteq N \cup \{e\} \times N \cup \{f\}$

Set of arcs representing road links

with associated data

b_{ij} Base travel time for each road link $(i,j) \in A$

s_{ij} Traffic sensitivity for each road link $(i,j) \in A$

c_{ij} Capacity for each road link $(i,j) \in A$

T Desired throughput from e to f

Formulation

Determine

x_{ij} Traffic flow through road link $(i,j) \in A$

t_{ij} Actual travel time on road link $(i,j) \in A$

to minimize

$\Sigma_{(i,j)\in A}\, t_{ij} x_{ij} / T$

Average travel time from e to f

Formulation *(cont'd)*

Subject to

$$t_{ij} = b_{ij} + \frac{s_{ij} x_{ij}}{1 - x_{ij}/c_{ij}} \quad \text{for all } (i,j) \in A$$

Travel times increase as flow approaches capacity

$$\Sigma_{(i,j)\in A}\, x_{ij} = \Sigma_{(j,i)\in A}\, x_{ji} \quad \text{for all } i \in N$$

Flow out equals flow in at any intersection

$$\Sigma_{(e,j)\in A}\, x_{ej} = T$$

Flow into the entrance equals the specified throughput

Traffic Network

AMPL Formulation

Symbolic data

```
set INTERS;             # intersections (network nodes)

param EN symbolic;      # entrance
param EX symbolic;      # exit

   check {EN,EX} not within INTERS;

set ROADS within {INTERS union {EN}} cross {INTERS union {EX}};

                            # road links (network arcs)

param base {ROADS} > 0;     # base travel times
param sens {ROADS} > 0;     # traffic sensitivities
param cap {ROADS} > 0;      # capacities

param through > 0;          # throughput
```

Traffic Network

AMPL Formulation *(cont'd)*

Symbolic model

```
var Flow {(i,j) in ROADS} >= 0, <= .9999 * cap[i,j];
var Time {ROADS} >= 0;

minimize Avg_Time:
  (sum {(i,j) in ROADS} Time[i,j] * Flow[i,j]) / through;

subject to Travel_Time {(i,j) in ROADS}:
  Time[i,j] = base[i,j] + (sens[i,j]*Flow[i,j]) / (1-Flow[i,j]/cap[i,j]);

subject to Balance_Node {i in INTERS}:
  sum{(i,j) in ROADS} Flow[i,j] = sum{(j,i) in ROADS} Flow[j,i];

subject to Balance_Enter:
  sum{(EN,j) in ROADS} Flow[EN,j] = through;
```

Traffic Network

AMPL Data

Explicit data independent of symbolic model

```
set INTERS := b c ;

param EN := a ;
param EX := d ;

param: ROADS: base cap sens :=
         a b    4   10   .1
         a c    1   12   .7
         c b    2   20   .9
         b d    1   15   .5
         c d    6   10   .1 ;

param through := 20 ;
```

AMPL Solution *(cont'd)*

Model + data = problem to solve, using CPLEX?

```
ampl: model traffic.mod;
ampl: data traffic.dat;

ampl: option solver cplex;
ampl: solve;

CPLEX 12.3.0.0:
Constraint _scon[1] is not convex quadratic
since it is an equality constraint.
```

Traffic Network

AMPL Solution *(cont'd)*

Look at the model again . . .

```
var Flow {(i,j) in ROADS} >= 0, <= .9999 * cap[i,j];
var Time {ROADS} >= 0;

minimize Avg_Time:
  (sum {(i,j) in ROADS} Time[i,j] * Flow[i,j]) / through;

subject to Travel_Time {(i,j) in ROADS}:
  Time[i,j] = base[i,j] + (sens[i,j]*Flow[i,j]) / (1-Flow[i,j]/cap[i,j]);

subject to Balance_Node {i in INTERS}:
  sum{(i,j) in ROADS} Flow[i,j] = sum{(j,i) in ROADS} Flow[j,i];

subject to Balance_Enter:
  sum{(EN,j) in ROADS} Flow[EN,j] = through;
```

Traffic Network

AMPL Solution *(cont'd)*

Quadratically constrained reformulation

```
var Flow {(i,j) in ROADS} >= 0, <= .9999 * cap[i,j];
var Delay {ROADS} >= 0;

minimize Avg_Time:
  sum {(i,j) in ROADS} (base[i,j]*Flow[i,j] + Delay[i,j]) / through;

subject to Delay_Def {(i,j) in ROADS}:
  sens[i,j] * Flow[i,j]^2 <= (1 - Flow[i,j]/cap[i,j]) * Delay[i,j];

subject to Balance_Node {i in INTERS}:
  sum{(i,j) in ROADS} Flow[i,j] = sum{(j,i) in ROADS} Flow[j,i];

subject to Balance_Enter:
  sum{(EN,j) in ROADS} Flow[EN,j] = through;
```

AMPL Solution *(cont'd)*

Model + data = problem to solve, using CPLEX?

```
ampl: model trafficQUAD.mod;
ampl: data traffic.dat;

ampl: option solver cplex;
ampl: solve;

CPLEX 12.3.0.0:
QP Hessian is not positive semi-definite.
```

AMPL Solution *(cont'd)*

Simple conic quadratic reformulation

```
var Flow {(i,j) in ROADS} >= 0, <= .9999 * cap[i,j];
var Delay {ROADS} >= 0;
var Slack {ROADS} >= 0;

minimize Avg_Time:
  sum {(i,j) in ROADS} (base[i,j]*Flow[i,j] + Delay[i,j]) / through;

subject to Delay_Def {(i,j) in ROADS}:
  sens[i,j] * Flow[i,j]^2 <= Slack[i,j] * Delay[i,j];

subject to Slack_Def {(i,j) in ROADS}:
  Slack[i,j] = 1 - Flow[i,j]/cap[i,j];

subject to Balance_Node {i in INTERS}:
  sum{(i,j) in ROADS} Flow[i,j] = sum{(j,i) in ROADS} Flow[j,i];

subject to Balance_Enter:
  sum{(EN,j) in ROADS} Flow[EN,j] = through;
```

AMPL Solution *(cont'd)*

Model + data = problem to solve, using CPLEX!

```
ampl: model trafficSOC.mod;
ampl: data traffic.dat;

ampl: option solver cplex;
ampl: solve;

CPLEX 12.3.0.0: primal optimal; objective 61.04693968
15 barrier iterations

ampl: display Flow;

Flow :=
a b    9.55175
a c   10.4482
b d   11.0044
c b    1.45264
c d    8.99561
;
```

AMPL Solution *(cont'd)*

Same with integer-valued variables

```
var Flow {(i,j) in ROADS} integer >= 0, <= .9999 * cap[i,j];
```

```
ampl: solve;

CPLEX 12.3.0.0: optimal integer solution within mipgap or absmipgap;
   objective 76.26375017
19 MIP barrier iterations
0 branch-and-bound nodes

ampl: display Flow;

Flow :=
a b    9
a c   11
b d   11
c b    2
c d    9
;
```

AMPL Design for Convex Quadratics

Problem types

- Elliptical: quadratic programs (QPs)
- Conic: second-order cone programs (SOCPs)

Current situation

- Each solver recognizes some elementary forms
- Modeler must convert to these forms

Goal

- Recognize many equivalent forms
- Automatically convert to a canonical form
- Further convert as necessary for each solver