Optimization modeling: recent enhancements and future extensions

Michael C. Ferris

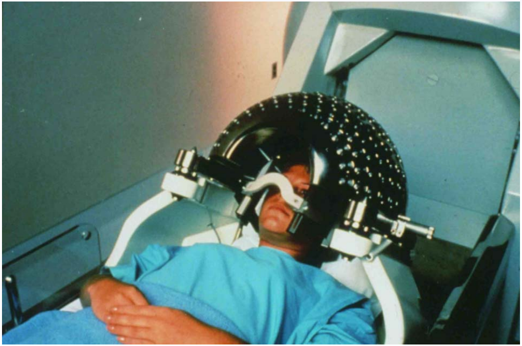
University of Wisconsin-Madison

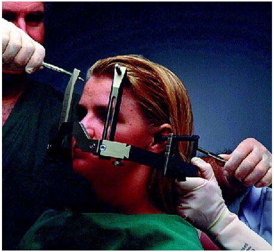
University of Minnesota
April 25, 2007

Modeling languages: an example

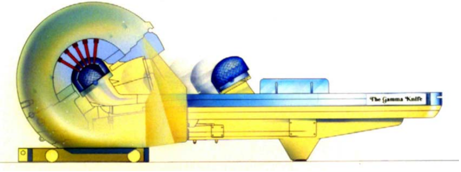
$$\min c^T x \text{ s.t. } a_i^T x \leq b_i, i = 1, 2, \ldots, m$$

```
set i, j; parameter b(i), c(j), A(i,j);

variables obj, x(j);
equations defobj, cons(i);

defobj.. obj =e= sum(j, c(j)*x(j));

cons(i).. sum(j, A(i,j)*x(j)) =l= b(i);

model lpmod /defobj, cons/;
solve lpmod using lp min obj;
```





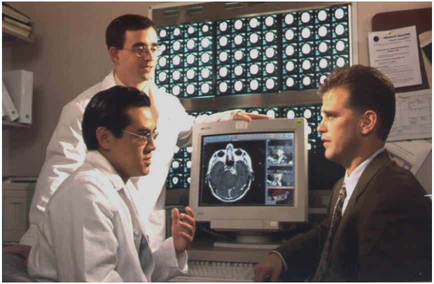
The Gamma Knife

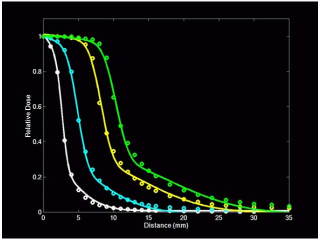


Optimization model

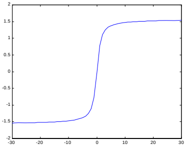
$$
\begin{aligned}
\min_{t_{s,w},x_s} \quad & Under(Target) && \text{(Dose under 1)} \\
\text{s.t.} \quad & Dose(i) = \sum_{s \in S, w \in W} t_{s,w} D_w(x_s, i) && (D_w \text{ nonlinear function}) \\
& 0 \le Under(i) \ \ge 1 \ - Dose(i) && \\
& Dose(Target) / \Big(\sum_{s,w} t_{s,w} \overline{D_w} \Big) \ \ge \ P && \text{(Conformity)} \\
& \sum_{s,w} \arctan(t_{s,w}) \le N \ \pi/2 && (\text{Use} \le N \text{ shots}) \\
& 0 \le Dose(i) \le 1, \ 0 \le t_{s,w} &&
\end{aligned}
$$

Model is very large, nonlinear and requires “quick” solution





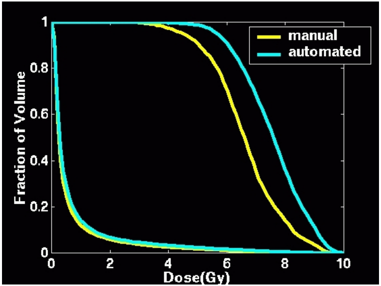
Nonlinear dose distribution



Approximate combinatorics

$$\forall s \in S$$

$$\sum_{w} arctan(t_{s,w}) \leq \frac{\pi}{2}$$



Dose-volume histogram

Solution process

Modeling system allows multiple models to be solved, each generating better approximations to underlying problem

- Rotate data (prone/supine)
- Skeletonization starting point procedure (network LP)
- Solve conformity NLP subproblem (to estimate P)
- Coarse grid shot optimization (reduced # i's)
- Refine grid (add violated locations) and resolve NLP
- Refine smoothing parameter and resolve NLP
- Round and fix locations, solve MIP for exposure times

manual

optimized

Modeling languages: state-of-the-art

- Optimization models improve understanding of underlying systems and facilitate operational improvements
- Key link to applications, prototyping of optimization capability
- Widely used in:
  - engineering - operation/design
  - economics - policy/energy modeling
  - military - operations/planning
  - finance, medical treatment, supply chain management, etc.
- Interface to solutions: facilitates automatic differentiation, separation of data, model and solver
- Modeling languages no longer novel: typically represent another tool for use within a solution process.

Modeling Language Limitations

- Data (collection) remains bottleneck in many applications
  - Tools interface to databases, spreadsheets, Matlab
- Problem format is old/traditional

$$\min_x f(x) \text{ s.t. } g(x) \leq 0, h(x) = 0$$

  - Support for integer, sos, semicontinuous variables.
  - Limited support for logical constructs
- Limited treatment of uncertainties
- Little support for modern (grid or parallel) machines

New types of constraints

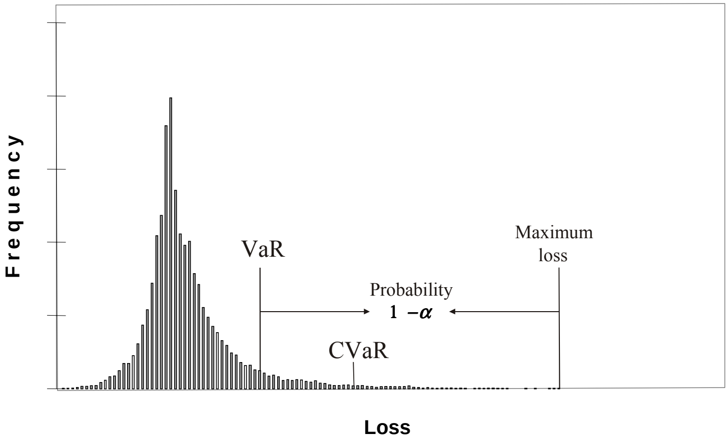
- range constraints $L \leq Ax - b \leq U$
- robust programming (probability constraints, stochastics)

$$f(x, \xi) \leq 0, \forall \xi \in \mathcal{U}$$

- conic programming $a_i^T x - b_i \in K_i$
- soft constraints
- rewards and penalties

Some constraints can be reformulated easily, others not!

CVaR constraints: mean excess dose (radiotherapy)



Move mean of tail to the left!

ENLP: Primal problem

$$\min_{x \in X} f_0(x) + \theta(f_1(x), \ldots, f_m(x))$$

Original problem:

$$\begin{array}{ll} \min\limits_{x_1, x_2, x_3} & \exp(x_1) \\ \text{s.t.} & \log(x_1) = 1 \\ & x_2^2 \leq 2 \\ & x_1/x_2 = \log(x_3), 3x_1 + x_2 \leq 5, x_1 \geq 0, x_2 \geq 0 \end{array}$$

Soft penalization of red constraints:

$$\begin{array}{ll} \min\limits_{x_1, x_2, x_3} & \exp(x_1) + 5 \left\| \log(x_1) - 1 \right\|^2 + 2 \max(x_2^2 - 2, 0) \\ \text{s.t.} & x_1/x_2 = \log(x_3), 3x_1 + x_2 \leq 5, x_1 \geq 0, x_2 \geq 0 \end{array}$$

ENLP: Primal problem

$$\min_{x \in X} f_0(x) + \theta(f_1(x), \ldots, f_m(x))$$

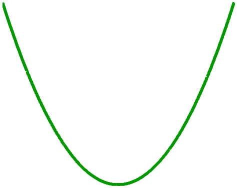
$$X = \left\{x \in \mathbf{R}^3 : 3x_1 + x_2 \leq 5, x_1 \geq 0, x_2 \geq 0\right\}$$

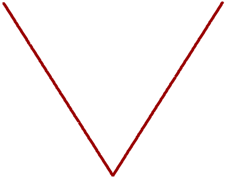
$$f_1(x) = \log(x_1) - 1, f_2(x) = x_2^2 - 2, f_3(x) = x_1/x_2 - \log(x_3)$$

$$\theta_1(u) = 5\left\|u\right\|^2, \theta_2(u) = 2\max(u, 0), \theta_3(u) = \psi_{\{0\}}(u)$$

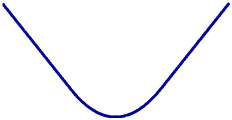
θ nonsmooth due to the max term; θ separable in example.

Examples of different θ





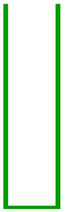
but solution reformulations are very different



$$\theta(u) = \begin{cases} \gamma u - \frac{1}{2}\gamma^2 & \text{if } u \geq \gamma \\ \frac{1}{2}u^2 & \text{if } u \in [-\gamma, \gamma] \\ -\gamma u - \frac{1}{2}\gamma^2 & \text{else} \end{cases}$$

Huber function used in robust statistics.

More general θ functions



In general any piecewise linear penalty function can be used: (different upside/downside costs). Also cone constraints.
General form:

$$\theta(u) = \sup_{y \in Y} \{ y' u - k(y) \}$$

θ can take on ∞ and may be nonsmooth; it is convex.

Specific choices of k and Y

- L_2: $k(y) = \frac{1}{4\lambda} y^2$, $Y = (-\infty, +\infty)$
- L_1: $k(y) = 0$, $Y = [-\rho, \rho]$
- L_∞: $k(y) = 0$, $Y = \Delta$, unit simplex
- Huber: $k(y) = \frac{1}{4\lambda} y^2$, $Y = [-\rho, \rho]$
- Second order cone constraint: $k(y) = 0$, $Y = C^\circ$

Elegant Duality

For these θ (defined by $k(\cdot)$, Y), duality is derived from the Lagrangian:

$$\mathcal{L}(x, y) = f_0(x) + \sum_{i=1}^{m} y_i f_i(x) - k(y)$$
$$x \in X, y \in Y$$

- Dual variables in Y not simply ≥ 0 or free.
- Saddle point theory, under convexity.
- Dual Problem and Complete Theory.
- Special case: ELQP - dual problem is also an ELQP.

Implementation: convert tool

```
$echo nlp2mcp > convert.opt

e1.. obj =e= exp(x1);
e2.. log(x1)-1 =e= 0;
e3.. sqr(x2)-2 =e= 0;
e4.. x1/x2 =e= log(x3);
e5.. 3*x1 + x2 =l= 5;

$onecho > enlpinfo.scr
e2 sqr 5
e3 plus 2
$offecho

solve mod using nlp min obj;
```

Library of different θ functions implemented.

First order conditions

- Every optimizer knows how to reformulate. One way:

$$\begin{array}{rlcl} 0 \in & \nabla_x \mathcal{L}(x, y) & + & N_X(x) \\ 0 \in & -\nabla_y \mathcal{L}(x, y) & + & N_Y(y) \end{array}$$

  $N_X(x)$ is the normal cone to the closed convex set X at x.

- Automatically creates an MCP:

```
model enlp /   gradLx.x,
              -gradLy.y /;
solve enlp using mcp;
```

- Available β-release - this week!
- Extend X and Y beyond simple bound sets.

Large scale issues

Problem			LUSOL	
n	dim	nnz	time	pct LU
20	1600	68171	0.418	77.0%
50	10000	587112	9.166	91.6%
100	40000	2773928	49.308	93.2%

Problem			UMFPACK	
n	dim	nnz	time	pct LU
20	1600	66684	0.218	56.4%
50	10000	658755	2.268	66.3%
100	40000	2778235	11.520	73.2%

Example: Robust Linear Programming

Data in LP not known with certainty:

$$\min c^T x \text{ s.t. } a_i^T x \leq b_i, i = 1, 2, \ldots, m$$

Suppose the vectors a_i are known to be lie in the ellipsoids

$$a_i \in \varepsilon_i := \{\bar{a}_i + P_i u : \|u\|_2 \leq 1\}$$

where $P_i \in \mathbf{R}^{n \times n}$ (and could be singular, or even 0).
Conservative approach: robust linear program

$$\min c^T x \text{ s.t. } a_i^T x \leq b_i, \text{ for all } a_i \in \varepsilon_i, i = 1, 2, \ldots, m$$

Robust Linear Programming as SOCP/ENLP

The constraints can be rewritten as:

$$\begin{aligned} b_i &\geq \sup\left\{a_i^T x : a_i \in \varepsilon_i\right\} \\ &= \bar{a}_i^T x + \sup\left\{u^T P_i^T x : \|u\|_2 \leq 1\right\} = \bar{a}_i^T x + \left\|P_i^T x\right\|_2 \end{aligned}$$

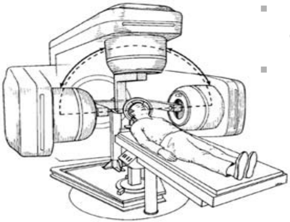
Thus the robust linear program can be written as

$$\min c^T x \text{ s.t. } \bar{a}_i^T x + \left\|P_i^T x\right\|_2 \leq b_i, i = 1, 2, \ldots, m$$

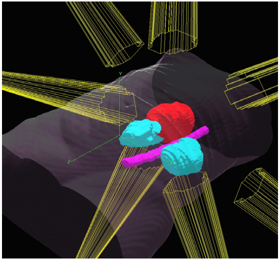
$$\min c^T x + \sum_{i=1}^{m} \psi_C(b_i - \bar{a}_i^T x, P_i^T x)$$

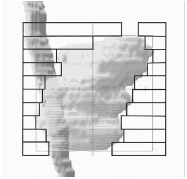
where C represents the second-order cone. Our extension allows automatic reformulation and solution (as SOCP) by Mosek or Conopt.

Radiotherapy Treatment

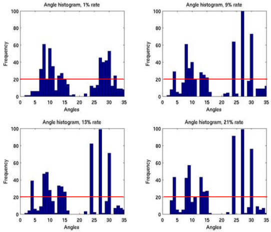


- Fire from multiple angles
- Superposition allows high dose in target, low elsewhere





- Beam shaping via collimator
- Other enhancements
- Sampling allows good angles to be determined quickly and in parallel



How to solve: Gams/Grid

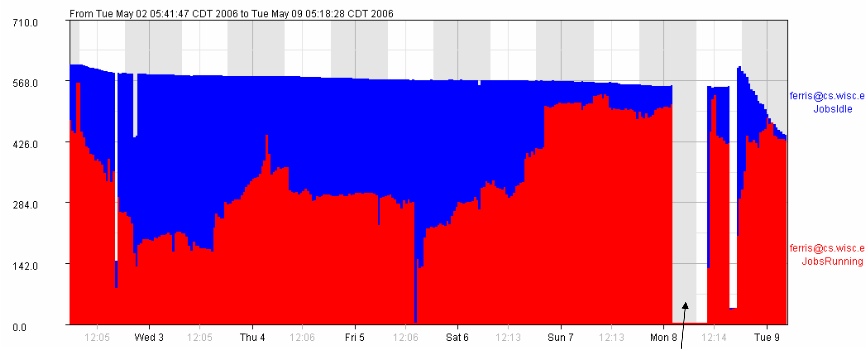
Commercial modeling system - abundance of real life models to solve

- solvelink = 3;
- solve mod using minlp min obj;
- execute_loadhandle mod;
- Multiple jobs spawned for grid solution, can be collected asynchronously
- Computational engine configurable (e.g. Condor, MW-GAMS, background process)

Grid Computer: Condor

- Grid: pool of connected computers managed and available as a common computing resource
- Condor: uses dedicated clusters and cycles from desktop workstations ($>$ 1000 machines available for "ferris")
- Heterogeneous machines: condor_chirp for inter-worker communication
- Machines updated regularly
- Fault tolerance
- Available for download, configurable

Grid resources used



Partitioned into 1000 subproblems, over 300 machines running for multiple days

main submitting machine died, jobs not lost

Optimization Strategy

CPLEX uses a branch and bound/cut procedure for global optimization with clever search strategies. Fully utilize grid by:

- Strategy: Partition using Strong Branching
- Have one machine working on heuristic solutions for original problem
  - CPLEX mipemphasis 1 or 4
- Subproblem emphasis on best-bound
  - CPLEX mipemphasis 3
- Repartition longest running jobs
- Restart from incumbent (cf NLP) after machine failures

Some MIPLIB results

	ROLL3000	A1C1S1	TIMTAB2 (added problem cuts)
# subproblems	986	1089	3320
objective	12890	11503.4	1096557.
Cplex B&B nodes	400,034	1,921,736	17,092,215
CPU time used	50h	3452h	2384h
CPU time wasted	0.5h	248h	361h
Wall time	Over night	Over night	Over night

Scheduling Multistage Batch Plants

Problem Features:

- Solution within 1 day
- Three level decision process (GAMS)
- Split order into batches
- Assign batches to processing units
- Sequence batches over stages

Solution:

- Instance 1: solved sequentially CPLEX
- Instance 2: solved GAMS/CPLEX/Condor
- Instance 3: gap (1176-1185) after 24h

Adaptive SB Method

- Split model using domain expertise at top levels
- 234 jobs, fixes batches and some assignments
- Apply (very) strong branching to generate a collection of subproblems
- Solve each subproblem
- If 2 hour time limit reached, reapply strong branching to subdivide and resolve
- Instance 3 solved (22 hours) - 4 branching levels
- (5 days,22 hrs; nodes = 58,630,425; 7356 jobs)

Conclusions and future extensions

- Practical/usable implementation of Rockafellar's ENLP approach within a modeling system
- System can easily formulate second order cone programs, robust optimization, soft constraints via piecewise linear penalization (with strong supporting theory)
- Easy switch to generate optimizations for grid solution

- Enhance library of (implemented) θ functions
- Exploit structure of θ in solvers
- Extend MCP solvers to VI solvers
- Exploit grid computing infrastructure
  - "Time-constrained" problems (cf "real-time")
  - Re-optimization (model updating)
  - Global optimization
  - Decomposition approaches
- Further application deployment