Why use a modeling language: a view from optimization

Michael C. Ferris & Jeff Linderoth

Computer Sciences Department and WID
University of Wisconsin-Madison

July 31, 2012

Models

"A model should be as simple as possible and yet no simpler"
–Albert Einstein

Models

"A model should be as simple as possible and yet no simpler"
–Albert Einstein

Why Model

1. To get an answer!
2. From building a model, we can gain insight.
3. We can "experiment" with a model.

Types of models

- Physical
  - Airline wing design
- Abstract
  - Statistical: Time Series, Regression, etc...
  - Simulation
  - Economic
  - **Optimization!**

Components of an Optimization Model

1. Decision variables
   - Variables representing the unknown quantities

Variables:

- x: Pounds of barley to purchase
- y: Pounds of hops to purchase
- z: Gallons of beer made

Components of an Optimization Model

1. Decision variables
   - Variables representing the unknown quantities
2. Constraints
   - Requirements that all solutions must satisfy, (expressed algebraically)

Variables:

- x: Pounds of barley to purchase
- y: Pounds of hops to purchase
- z: Gallons of beer made

Constraints:

- $y \leq 2$
- $x \leq 8y$
- $z = 0.4x + 0.9y$

Components of an Optimization Model

1. Decision variables
   - Variables representing the unknown quantities
2. Constraints
   - Requirements that all solutions must satisfy, (expressed algebraically)
3. Objective
   - A quantity that you would like to make as small or as large as possible.

Variables:

- x: Pounds of barley to purchase
- y: Pounds of hops to purchase
- z: Gallons of beer made

Constraints:

- $y \leq 2$
- $x \leq 8y$
- $z = 0.4x + 0.9y$

Objective:

- $\max z$

Solving

- Actually involves gathering and processing data: Turning your model into an instance.
  - Model : A structure containing (algebraic) relationships between entities.
  - Instance : A combination of data and model that can be solved. (i.e – it has "numbers")
  - Spreadsheet models "blur" this distinction, that's why I don't like them very much.

Solving

- Actually involves gathering and processing data: Turning your model into an instance.
  - Model : A structure containing (algebraic) relationships between entities.
  - Instance : A combination of data and model that can be solved. (i.e – it has "numbers")
  - Spreadsheet models "blur" this distinction, that's why I don't like them very much.
- Algebraic modeling languages are better!
  - Have hooks to solvers
  - Many have hooks to spreadsheets and databases

Data

```
set i /a, b, c, d/;
alias(i,j,k);
set orig(i) /a/;
set dest(i) /d/;
parameter demand /4000/;

set arcs(i,i) /a.(b*c),(b*c).d/;

table data(i,i,*)
            fixed\_time congestion\_factor
     a.b    45          0
     a.c    0           0.01
     b.d    0           0.01
     c.d    45          0
     c.b    eps         eps;

parameter fixed\_time(i,i) Fixed travel time on a link;
parameter congestion\_factor(i,i) Flow-dependent effects;

fixed\_time(arcs) = data(arcs,'fixed\_time');
congestion\_factor(arcs) = data(arcs,'congestion\_factor');
```

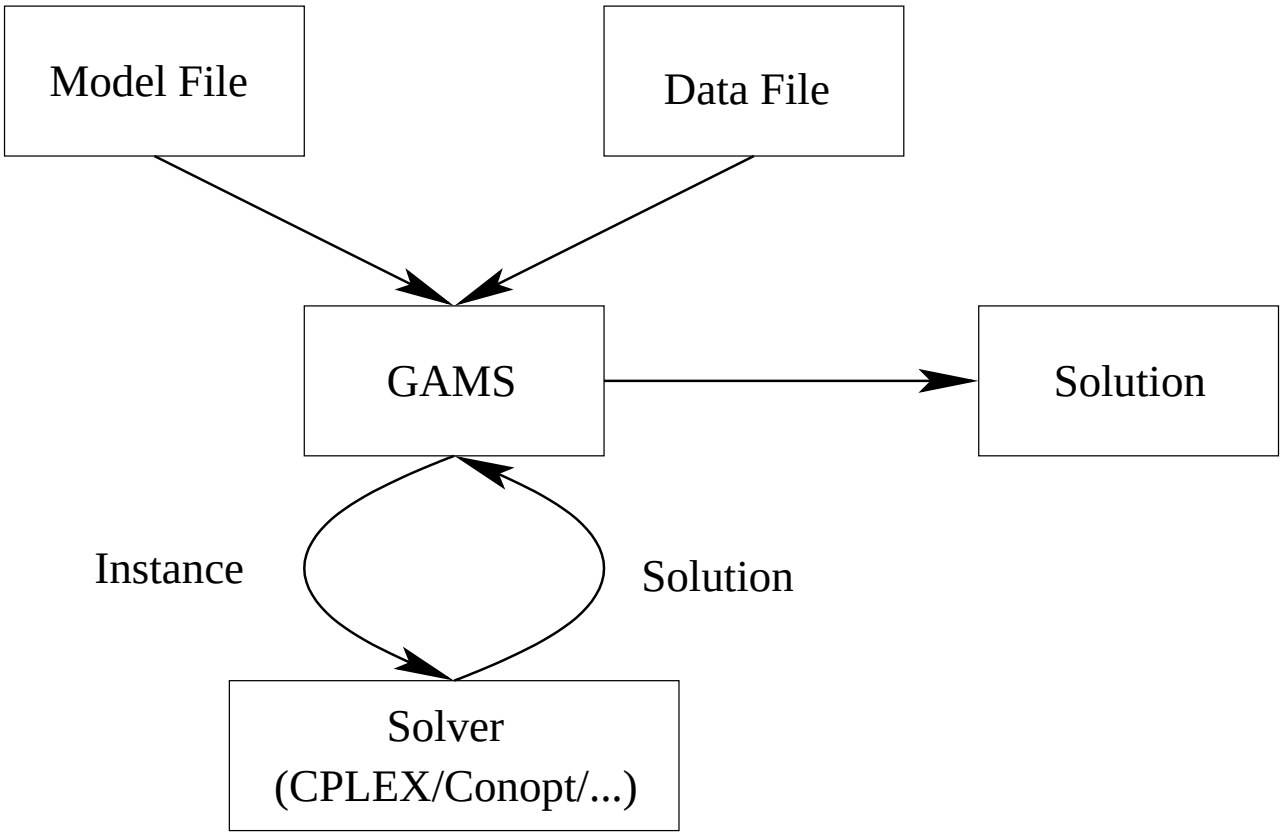
Model

```
positive variables Flow(i,j);
variables T(j);

equations Short(i,j), Balance(k);

Balance(k)..
  sum(arcs(i,k), Flow[i,k]) + demand$orig(k)
  =e= sum(arcs(k,j), Flow[k,j]) + demand$dest(k);

Short(i,j)$arcs(i,j)..
  fixed_time[i,j] + congestion_factor[i,j] * Flow[i,j]
  + T(j) =g= T(i);

model braess /Short.Flow,Balance.T/;
```

Solution

```
solve braess using mcp;

arcs('c','b') = yes;
fixed_time(arcs) = data(arcs,'fixed_time');
congestion_factor(arcs) = data(arcs,'congestion_factor');

solve braess using mcp;

fixed_time('c','b') = 2;
congestion_factor('c','b') = 0;

solve braess using mcp;
```

The Modeling Language Interface



This talk is not the real world

- We will often combine Model and Data file into one file, but this is not best practice

Common Fallacies

1. How can anyone possibly get and be confident in the data that makes up the model?

Common Fallacies

1. How can anyone possibly get and be confident in the data that makes up the model?
   - We're not. The analyst must understand the reality of the process to deduce whether the model solution makes sense

Common Fallacies

1. How can anyone possibly get and be confident in the data that makes up the model?
   - We're not. The analyst must understand the reality of the process to deduce whether the model solution makes sense
2. It's "too abstract"

Common Fallacies

1. How can anyone possibly get and be confident in the data that makes up the model?
   - We're not. The analyst must understand the reality of the process to deduce whether the model solution makes sense
2. It's "too abstract"
   - It's not. The analyst must be able to explain *why* the solution approach is proper

Common Fallacies

1. How can anyone possibly get and be confident in the data that makes up the model?
   - We're not. The analyst must understand the reality of the process to deduce whether the model solution makes sense
2. It's "too abstract"
   - It's not. The analyst must be able to explain *why* the solution approach is proper
3. If we got an answer on the computer, it must be right!

Common Fallacies

1. How can anyone possibly get and be confident in the data that makes up the model?
   - We're not. The analyst must understand the reality of the process to deduce whether the model solution makes sense
2. It's "too abstract"
   - It's not. The analyst must be able to explain *why* the solution approach is proper
3. If we got an answer on the computer, it must be right!
   - It's not. All models are wrong. But some are useful. (George Box)

Common Fallacies

1. How can anyone possibly get and be confident in the data that makes up the model?
   - We're not. The analyst must understand the reality of the process to deduce whether the model solution makes sense
2. It's "too abstract"
   - It's not. The analyst must be able to explain *why* the solution approach is proper
3. If we got an answer on the computer, it must be right!
   - It's not. All models are wrong. But some are useful. (George Box)

Common Fallacies

1. How can anyone possibly get and be confident in the data that makes up the model?
   - We're not. The analyst must understand the reality of the process to deduce whether the model solution makes sense
2. It's "too abstract"
   - It's not. The analyst must be able to explain *why* the solution approach is proper
3. If we got an answer on the computer, it must be right!
   - It's not. All models are wrong. But some are useful. (George Box)

The upshot!

(Optimization) models should be one tool in the decision making process.

Another View at Model Components

1. Inputs
   - Sets. Used typically for algebraic models.
     - e.g., P: Set of products, I: Set of locations
   - "Numbers". These are called parameters. The parameters may be indexed over sets.
     - e.g., u_p: The maximum amount of product p available

Another View at Model Components

1. Inputs
   - Sets. Used typically for algebraic models.
     - e.g., P: Set of products, I: Set of locations
   - "Numbers". These are called parameters. The parameters may be indexed over sets.
     - e.g., u_p: The maximum amount of product p available
2. Decision Variables
   - "Numbers you are allowed to change". It is the goal of the optimization to find the "best" values of these controls (or decision variables). Decision variables can also be indexed over sets
     - e.g., z_i: Gallons of beer to ship to location i

Another View at Model Components

1. Inputs
   - Sets. Used typically for algebraic models.
     - e.g., P: Set of products, I: Set of locations
   - "Numbers". These are called parameters. The parameters may be indexed over sets.
     - e.g., u_p: The maximum amount of product p available
2. Decision Variables
   - "Numbers you are allowed to change". It is the goal of the optimization to find the "best" values of these controls (or decision variables). Decision variables can also be indexed over sets
     - e.g., z_i: Gallons of beer to ship to location i
3. Outputs
   - These may be optimal values of the decision variables, or a derived value, such as the objective function value
   - e.g.: $\sum_{i \in \text{Madison}} z_i$

Solution Analysis

Warning!

Optimization is extreme. If there is a mistake in your model, optimization will usually find it.

- Verification : Is the model correct?
  - Are physical laws being obeyed.
- Validation : Does the model give an accurate picture of reality?
  - Create instances for which you can expect a certain type of solution, and see if the model returns such a solution.
  - Careful! Maybe your intuition about a solution is wrong!
- Sensitivity Analysis
  - How much are extra resources worth to me?
  - This is "marginal information."
- What-if Analysis : Change the instance and re-run.

Categories of Optimization Models

- Linear vs. Nonlinear?
  - Are the functional relationships between decision variables linear functions or nonlinear functions?
- Convex vs. Nonconvex?
  - Are the functional relationships convex?
- Discrete vs. Continuous?
  - Must the decision variables take only discrete values?
- Deterministic vs. Stochastic?
  - Is uncertainty in the model explicity considered?

Categories of Optimization Models

- Linear vs. Nonlinear?
  - Are the functional relationships between decision variables linear functions or nonlinear functions?
- Convex vs. Nonconvex?
  - Are the functional relationships convex?
- Discrete vs. Continuous?
  - Must the decision variables take only discrete values?
- Deterministic vs. Stochastic?
  - Is uncertainty in the model explicity considered?

The upshot

- These categorizations have a significant impact on the tractability of an instance
- You should be able to categorize problem instances

GAMS Model Types

- LP: Linear Programming
- MIP: Mixed-Integer Programming
- NLP: Non-Linear Programming
- MCP: Mixed Complementarity Problems
- MPEC: Mathematical Programs with Equilibrium Constraints
- CNS: Constrained Nonlinear Systems
- DNLP: Non-Linear Programming with Discontinuous Derivatives
- MINLP: Mixed-Integer Non-Linear Programming
- QCP: Quadratically Constrained Programs
- MIQCP: Mixed Integer Quadratically Constrained Programs

Other Great Optimization Tools/Resources

- COIN-OR: `http://www.coin-or.org/`
- NEOS: `http://www.neos-guide.org/`

NEOS Case Studies

- `http://www.neos-guide.org/NEOS/index.php/Case_Studies`.
- A web-wiki description of a problem, understandable to the general public.
- A model and submission that would solve the problem and provide some form of output/visualization.

Installing Gams—An Interactive(?) Demo

1. Obtain GAMS. Two options
   1. Go to `http://www.gams.com/download/`, Download the appropriate executable for Win32 (or Win64) if your laptop is 64bit.
   2. There are also executables for Mac OS X (make sure to get right one), and Linux
2. Installation instructions at...
   - `http://www.gams.com/docs/gams/win-install.pdf`
   - `http://support.gams-software.com/doku.php?id=installation:how_do_i_install_the_gams_version_for_macintosh`
3. Interfaces to:
   - Matlab and R (via GDX files)