CS635: Tools and Environments for Optimization. Lecture 1

Michael Ferris

Computer Sciences Department
University of Wisconsin-Madison

January 22, 2014

Today's Outline

- About this class.
- About me.
- About you.
- Modeling:
  - What is it?
  - Why should we care about this stuff?[1]
- GAMS

[1] Besides, of course, the fact that you want to get a good grade

Class Overview

- Meeting Times: Monday, Wednesday, Friday 8:40-9:40AM, 2540 Eng Hall.
- Office Hours: (*Please* try to use them).
  - Monday 11:30AM-12:30PM
  - Wednesday 10AM-11:00AM
- My office is 4381 Computer Sciences
- We will make extensive use of the online discussion forum Piazza: `www.piazza.com`. Homework hints and tips will be posted there.
- Course mailing list: isye635-1-s14@lists.wisc.edu

Course HomePage

- `https://learnuw.wisc.edu/`
- This will be used extensively in this class.
- Assignments will be turned-in via the web page.
- Webpage (including schedule) will be updated quite frequently.

Teaching Assistant

- Okan Akalin
- **Office**: 1306 Computer Sciences
- email: `okan@cs.wisc.edu`
- **Office Hours**: Tuesdays/Thursdays 4:00-5:00PM

Prerequisites

- `Comp Sci302`. You don't necessarily need to know Java, but you do need to be comfortable with a computer programming language. If you are not, then this course is not really for you.
- `Math 340 or equiv`. You need to know a little bit of Calculus, be familiar with Matrix notation, and have the mathematical sophistication necessary to not be intimidated by symbols like $\sum$, $\forall$ and $\in$.
- Knowledge of optimization modeling at the level of `IE 323`, or `IE/CS 525` will be quite useful. If you don't know what linear programming is, get ready to get up to speed quickly.

Course Details

- Learning is better if you participate.
  - I will call on you during class. (Gasp!)
  - A portion of your grade depends on your participation.
- I really want the class to be interactive.
  - We will spend time working through models

Coursework Details

- Assignments
  - Expect to be relatively time-consuming
  - Assigned almost every week
- Course Project
  - More info later...
- I am giving a final exam in this course. This will not weight "too heavily" on the grade. But they will force everyone to *learn for themselves*, and not rely on others to do the homework and projects.

Course Project

- The fundamental goal of the course is to equip you with the experience and background necessary to model and implement a real, large-scale optimization problem.
- We will do a class project in which you will define, model, solve, analyze, and explain the solution(s) to a decision problem of your own choosing.
- Typically (and ideally), students come to class having an idea of the project they would like to do.
- We will try and give some additional ideas.

NEOS and Case Studies

- In this semester, we also will allow students the opportunity (and likely extra credit) to students who contribute case studies to NEOS
  - `http://www.neos-guide.org/Case-Studies`

NEOS Projects

- 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. Gets extra credit!

Typical Project Expectations

- Write one page outline of project and get it approved by me. (Around end of Feb). Respond to questions about project
- Implement an optimization model, describe/output results in a form reasonable for the application. Describe, modify, and implement improvements to the original format if necessary.
- Turn in completed project report (end of course)
  - Short - 4 pages
  - Make clear what you have done, and detail your contribution to the "question at hand".
- A potential oral defense of project.

Grading

- 60% Regular Homework Sets
- 15% Final Exam
- 25% Course Project

Course Texts

- None Required.
- GAMS User's Guide is on course web page

On Reserve

- H.P. Williams, *Model Building in Mathematical Programming*
- Wayne L. Winston, Munirpallam Venkataramanan, *Introduction to Mathematical Programming : Operations Research*
- Ron Rardin, *Optimization in Operations Research*

Turning In Assignments

- We will use `Learn@UW` course dropbox to turn in all assignments.
  - You can turn in your homework from anywhere
  - You have to type your homework.
  - Most (all?) homework will be GAMS programming assignments, so this shouldn't be a problem

Travelin' Man

- I will be out of town delivering research seminars, at conferences, or at research project meetings on the following dates:

  Feb 17, Feb 19, Feb 21

Mathematical Topics

1. Linear Programming
2. Network Programming
3. Integer Programming
4. Stochastic Programming
5. Nonlinear Programming

- Will build models in GAMS: General Algebraic Modeling System

Course Objectives

1. The ability to write down an algebraic formulation of an optimization model that captures the main decision elements of practical problems.
2. The ability to categorize optimization models, and understand the implications of modeling on algorithm performance
3. To become familiar with the operation of state-of-the-art optimization software, including parameters that may significantly affect software performance
4. Advanced knowledge of the GAMS Modeling System for building and solving practical optimization problems
5. Have Fun (and work hard!)

Great Expectations

I am expected to...

- Teach
- Answer your questions
- Be at my office hours
- Give you feedback on how you are doing in a timely fashion

You are expected to...

- Learn
- Attend lectures and participate
- Do the problem sets
- Not be rude, if possible.
  - Sleeping, Talking, Showing up late
  - Cell Phones, Texting, Tweeting
  - Leaving in the middle of lecture

Decision Problems

- In this course you will learn to model decision problems

Decision Problems

1. How many clerks do I need at my grocery store?
2. How much capacity should I add to my telecommunications network?
3. How should I design my new airplane wing?
4. Is my new airbag design safer than the previous design?
5. What types of aircraft should fly each flight in a schedule?
6. What stocks should I buy?

- Warning! Not all of these questions are best answered by *optimization* models

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!**

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.

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:

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

Objective:

- $\max z$

Modeling Optimization Models

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.: $\sum_{i \in \text{Madison}} z_i$

Modeling Optimization Models

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

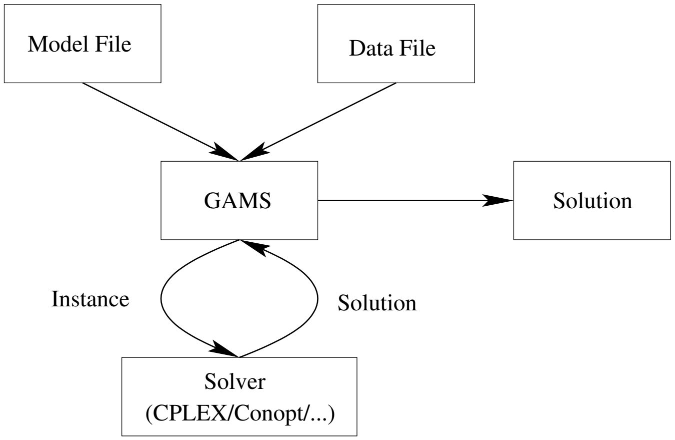
- These categorizations have a significant impact on the tractability of an instance
- You should be able to categorize problem instances

Modeling Optimization Models

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
- We will use the algebraic modeling language GAMS
  - With its Integrated Development Environment (IDE)
  - `http://www.gams.com`

The Modeling Language Interface



Class is not the real world

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

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.

Other Great Optimization Tools/Resources

- NEOS: `http://neos-server.org`
  - Online (web-accessible) Solvers for many classes of optimization problems.
- XPRESS-MP/Mosel: `http://www.fico.com/en/Products/DMTools/Pages/FICO-Xpress-Optimization-Suite.aspx`
  - Commercial Algebraic modeling language and solving engine. Free student-sized versions, and commercial grade versions available on CAE machines.
- COIN-OR: `http://www.coin-or.org/`
  - Open Source software for Operations Research
- Gurobi: `www.gurobi.com`. Free for academic use
- IBM/ILOG CPLEX: `http://www-01.ibm.com/support/docview.wss?uid=swg21419058`
  - Now also free for academic use

A Course Objective

Learn All These Problem Classes (from NEOS)

- Bound Constrained Optimization
- Complementarity Problems
- Global Optimization
- Linear Network Programming
- Linear Programming
- Mixed Integer Linear Programming
- Mixed Integer Nonlinearly Constrained Optimization
- Nonlinearly Constrained Optimization
- Nondifferentiable Optimization
- Semidefinite Programming
- Semi-infinite Optimization
- Stochastic Linear Programming
- Second Order Conic Programming
- Unconstrained Optimization

GAMS Model Types

- GAMS has a different categorization of instances:
  - 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
- You also will hopefully know which solvers are available to solve each of these problem classes.

Installing Gams—An Interactive(?) Demo

1. Obtain GAMS. Three options
   1. Use the CS lab machines that have Gams already installed.
   2. Go to `http://download.gams-software.com/`, Download the appropriate executable for Win32 (or Win64) if your laptop is 64bit. (If you don't know, try winmsd.exe on Windows-XP). GAMS has a GUI that only works on the Windows platform.
   3. 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. Use license file on course web page. (Content/Course Materials/)
4. Thank You GAMS!, $15K ($50K) worth of software

Default Solvers

- Update default solvers: File/Options/Solvers/System Defaults
- Take the Guided Tour. (Help/GAMSIDE Help Topics, Guided Tour)
- GAMS Model Library (File/Model Library/Open GAMS Model Library)

Assignment #0

- Get and install GAMS in your working environment.
- Please email me with questions or problems that you might have
- Next time, we set off with GAMS modeling!