IEMS 450-1 – Mathematical Programming I

Fall 2009

TTh 11:00-12:20

Dr. Halit Üster, uster@northwestern.edu, C140, 491-7008

Course Description and Objectives:

This is a first graduate level course in mathematical programming (or optimization), an approach to modeling complex systems that helps people to improve the performance of complex systems of many kinds everything from airline crew scheduling to trash disposal to supply-chain design to cream cheese blending. It is a required core course for first-year students in the MS/PhD program in Industrial Engineering and Management Sciences, and it is also taken by students in a variety of other engineering programs.

The goal of this course is to teach you how to use optimization technology effectively, and to introduce you to aspects of the subject that are fundamental to research. Among the skills to be emphasized are the insight to know when optimization is an appropriate tool, the expertise to build and solve appropriate optimization models, and the technical knowledge to interpret and use the solutions as well as to develop new optimization techniques and properties. Topics to be covered include linear, network, discrete, and nonlinear optimization. (A more detailed account of nonlinear optimization will offered in IEMS 450-2, to be offered by Prof. Sanjay Mehrotra in the winter quarter.)

Homework assignments address theoretical concepts, computational algorithms, and issues of practical implementation.

Prerequisites: A course covering linear algebra, especially matrices and linear equation systems.

Texts:

AMPL: A Modeling Language for Mathematical Programming, 2nd edition, by R. Fourer, D.M. Gay and B.W. Kernighan, ISBN 0534509835, Brooks/Cole/Cengage Learning, 2003; available from the Norris Center bookstore and online sources. You can get by with the first edition, which is usually available at a low price.

Optimization Models and Optimization Methods by Robert Fourer, available under Course Documents at the Blackboard (CMS) website for this course.

Lecture notes on linear equation systems are also available at the same location for downloading by those who want to review the subject.

Office Hours: My official office hours are 1:00-2:00 TTh. However, you can stop by anytime when I am in the office. Alternatively, you can make an appointment by phone or e-mail. I encourage each one of you to ask any questions you might have on the material during the lecture or right after the class. It is very much to your benefit to get these points cleared as soon as they occur. You are also strongly encouraged to come and see me in the office whenever you have problems.

Grading and Exam Dates: There will be one mid-term exam (100 points), one final exam (100 points), and homework assignments (total 100 points). Exam dates are as follows:

Mid-term exam	Tuesday, October 27 (tentative)
Final exam	Friday, December 11, 12:00 – 2:00

Computers and Software:

You will make extensive use of AMPL, a software package specialized for solving large-scale optimization problems. Separate handouts on using AMPL explain how to start using AMPL on the PCs in the IE/MS Computer Lab (Tech C135) and how to download a free student version of AMPL to your own computer. Please see "Using AMPL Studio" or "Using Command-Line AMPL" posted under Course Documents at the Blackboard (CMS) website.

Later in the course we may also use GIDEN, a specialized network optimization package. Further instructions will be posted as needed.

Homework Policy:

To plagiarize is to take and use another person's ideas or writings or inventions as one's own. Plagiarism is a serious offense, and we must be vigilant against it. Our homework policy consists of two parts, each of which is aimed at avoiding plagiarism and establishing a workable, collegial atmosphere.

1. Always write up your homework solutions alone that means without another person or another person's paper. In the case of a computer assignment, do not have another person's computer files with you when you are creating your own.

2. If you got any help with a particular problem from someone, then cite that person in your write-up. For example, I thank Julie Swann for helping me solve this problem. If you worked together with one or more other people, then each of you should cite the other(s).

Late homework submissions will not be accepted.

NU Disability Policy

`http://www.northwestern.edu/disability/`

To be eligible for disability-related services; students must have a visibly obvious or documented disability as defined by the Americans with Disabilities Act of 1990 (ADA) and Section 504 of the Rehabilitation Act of 1973. Under the ADA and Section 504, a person has a disability if he/she has a physical or mental impairment that substantially limits one or more major life activities such as walking, standing, seeing, speaking, hearing, sitting, breathing, and/or taking care of oneself. SSD is the designated office at Northwestern University that obtains and files disability related documents, certifies eligibility for services, determines reasonable accommodations, and develops plans for the provision of such accommodations. Students with disabilities are also offered auxiliary services, including assessment, library and lab assistants, notetakers, tutoring, assistive/adaptive technology, academic, psycho/social support, and mentorship.

Certifying Eligibility for Services

When appropriate, SSD requests disability-related documents from the appropriate licensed professional to certify a student as having a disability and to determine reasonable accommodations. Students who suspect that they have a disability, and have not received a formal assessment, may be referred to on-campus (Counseling and Psychological Services, Department of Communication Sciences and Disorders) or offcampus resources for an evaluation. Pending receipt of documentation, SSD reserves the right to deny services or accommodations.

Topics in IE/MS D50-1 — Fall 2009

1. **Formulation of Optimization Models** (linear and integer programs)
   - (a) Input (example: diet)
   - (b) Output
   - (c) Input-Output
   - (d) Network
   - (e) Hybrids (multiperiod, etc.)
   - (f) Extensions (min-max, piecewise-linear)
2. **Solving Linear Programs**
   - (a) Simplex method
     - i. How it starts
     - ii. How it proceeds
     - iii. How it stops
     - iv. Why it works
   - (b) Improvements to the simplex method
     - i. Extensions (phase 1, bounded variables)
     - ii. Making the iterations faster
     - iii. Reducing the number of iterations
     - iv. Avoiding degeneracy
   - (c) Alternatives to the simplex method
     - i. Interior-point methods
     - ii. Specialized methods
3. **Duality**
   - (a) "Taking" the dual
   - (b) Weak duality property
     - i. Statement and proof
     - ii. Use: corollaries and consequences
   - (c) Strong duality property
   - (d) Complementary slackness property
     - i. Statement
     - ii. Use and interpretation
   - (e) Interpretation of dual variables (sensitivity analysis)
4. **Integer Programming**
   - (a) Models
     - i. Fixed-charge network flows
     - ii. Facility location
     - iii. Traveling salesman problem
     - iv. Others
   - (b) Solution techniques
     - i. General-purpose: branch and bound

    ii. Special-purpose: cutting planes, branch and cut
    iii. Heuristics
  (c) Bounds

5. **Network Optimization**
  (a) Minimum spanning tree problem
    i. Model
    ii. Solve – Prim's Algorithm
  (b) Shortest path problem
    i. Model
    ii. Solve – Dijkstra's Algorithm
    iii. Linear programming model
  (c) Maximum flow problem
    i. Model
    ii. Solve – augmenting-path algorithm (Ford-Fulkerson)
    iii. Linear programming model
  (d) Minimum cut problem
    i. Model
    ii. Solve – augmenting-path algorithm (Ford-Fulkerson)

6. **Dynamic Programming**
  (a) Models (inventory/orders, production/inspection)
  (b) Translate to a shortest-path model
  (c) Solve using AMPL

7. **Computational Complexity**
  (a) Hard problems
    i. Touring and routing problems
    ii. Integer programming
    iii. The class of NP-hard problems
    iv. Methods to solve: branch-and-bound, branch-and-cut, heuristics
  (b) Not-so-hard problems
    i. Linear programming problem
    ii. Minimum spanning tree problem
    iii. Shortest path problem
    iv. Maximum flow problem
    v. The class of polynomial algorithms (P)
    vi. Methods to solve: simplex method, Prim's algorithm, Dijkstra's algorithm

8. **Nonlinear Programming**
  (a) Models
    i. Physical nonlinearities
    ii. Economic nonlinearities
    iii. Mathematical nonlinearities
  (b) Solution techniques
    i. Convergence to a solution
    ii. Fast convergence (Newton's method)
    iii. Global convergence (steepest descent)