

Stanford University, Dept of Management Science and Engineering  
MS&E 318 (CME 338) Large-Scale Numerical Optimization

Instructor: Michael Saunders Spring 2008

Notes 1: **Overview**

## Course Description

The main algorithms and software for constrained optimization, emphasizing the sparse-matrix methods needed for their implementation. Iterative methods for linear equations and least squares. The simplex method. Basis factorization and updates. Interior methods. The reduced-gradient method, augmented Lagrangian methods, and SQP methods.

3 units, Grading basis ABCD/NP, about 5 homeworks, 1 project, no final.

Prerequisites: Basic numerical linear algebra, including LU, QR, and SVD factorizations, and an interest in MATLAB, sparse-matrix methods, and algorithms for constrained optimization.

## Syllabus

1. Overview (problem types, NEOS, MATLAB, TOMLAB, headlines)
2. Iterative methods for symmetric  $Ax = b$  (symmetric Lanczos process, CG, SYMMLQ, MINRES, MINRES-QLP)
3. Iterative methods for unsymmetric  $Ax = b$  and least squares (Golub-Kahan process, CGLS, LSQR, Craig, Arnoldi process, GMRES)
4. The primal simplex method (phase 1 in practice, basis factorization, updating, crash, scaling, degeneracy)
5. Basis updates (PF update, Bartels-Golub, Forrest-Tomlin, Block-LU)
6. A sparse Basis Factorization Package (LUSOL: the engine for MINOS, SQOPT, SNOPT, MILES, PATH, lp\_solve)
7. Primal-dual interior methods for LP (CPLEX, OSL, LOQO, HOPDM, MOSEK) and convex nonlinear objectives (PDCO), Basis Pursuit, BP Denoising (Lasso, LARS, Homotopy, BPdual, SPGL1)
8. The reduced-gradient method (MINOS part 1)
9. BCL methods (Augmented Lagrangians, LANCELOT)
10. LCL methods (MINOS part 2, Knossos)
11. SQP methods (NPSOL, SQOPT, SNOPT)

## 1 Optimization Problems

We study optimization problems involving linear and nonlinear constraints:

NP	$\begin{aligned} & \underset{x \in \mathbb{R}^n}{\text{minimize}} && \phi(x) \\ & \text{subject to} && \ell \leq \begin{pmatrix} x \\ Ax \\ c(x) \end{pmatrix} \leq u, \end{aligned}$
----	---

where  $\phi(x)$  is a linear or nonlinear objective function,  $A$  is a sparse matrix,  $c(x)$  is a vector of nonlinear constraint functions  $c_i(x)$ , and  $\ell$  and  $u$  are vectors of lower and upper bounds. We assume the nonlinear functions  $\phi(x)$  and  $c_i(x)$  are *smooth*: they are continuous and have continuous first derivatives (gradients). Sometimes gradients are not available (or too expensive) and we use finite difference approximations. Sometimes we need second derivatives.

We study algorithms that find a *local optimum* for problem NP. Some examples follow. If there are many local optima, the starting point is usually important.

**LP** Linear Programming  $\min c^T x$  subject to  $\ell \leq \begin{pmatrix} x \\ Ax \end{pmatrix} \leq u$   
 MINOS, SNOPT, SQOPT  
 LSSOL, QPOPT, NPSOL (dense)  
 CPLEX, LOQO, HOPDM, MOSEK, OSL, XPRESS  
 CLP, lp\_solve (open source solvers [4, 11])

**QP** Quadratic Programming  $\min c^T x + \frac{1}{2}x^T H x$  subject to  $\ell \leq \begin{pmatrix} x \\ Ax \end{pmatrix} \leq u$   
 MINOS, SQOPT, SNOPT  
 LSSOL ( $H = B^T B$ , least squares), QPOPT ( $H$  indefinite)  
 CLP, CPLEX, LANCELOT, LOQO, OSL

**BC** Bound Constraints  $\min \phi(x)$  subject to  $\ell \leq x \leq u$   
 MINOS, SNOPT  
 LANCELOT, L-BFGS-B

**LC** Linear Constraints  $\min \phi(x)$  subject to  $\ell \leq \begin{pmatrix} x \\ Ax \end{pmatrix} \leq u$   
 MINOS, SNOPT, NPSOL

**NC** Nonlinear Constraints  $\min \phi(x)$  subject to  $\ell \leq \begin{pmatrix} x \\ Ax \\ c(x) \end{pmatrix} \leq u$   
 MINOS, SNOPT, NPSOL  
 CONOPT, LANCELOT  
 FILTER, KNITRO, LOQO (second derivatives)  
 IPOPT (open source solver [10])

Algorithms for finding local optima are used to construct algorithms for more complex optimization problems: *stochastic, nonsmooth, global, mixed integer*. An excellent example for **MINLP** is BARON [3].

## 2 AMPL, GAMS, NEOS

A fuller picture emerges from the list of problem types and solvers handled by the AMPL [2] and GAMS [8] modeling systems and the NEOS server [13]. NEOS is a free service provided by Argonne National Laboratory, Illinois. It allows us to submit optimization problems in various formats (MPS, AMPL, GAMS) to be solved remotely on geographically distributed solver stations. NEOS currently has 80 solvers, and processes 10000 jobs per week from thousands of users around the world. Over half a million solver jobs were submitted during the year 1 March 2007 to 29 Feb 2008. Top was 140000 jobs for our SOL/UCSD solver SNOPT.

**BCO** Bound Constrained Optimization  
BLMVM, L-BFGS-B, TRON

**CP** Complementarity Problems  
MILES, PATH

**GO** Global Optimization  
ACRS, GLCF, GLOBMIN, LGO, MLOCPSOA

**LNO** Linear Network Optimization  
NETFLO, RELAX4

**LP** Linear Programming  
BDMLP, BPMPD, CLP, CPLEX, FortMP, MOSEK, OOQP, OSL,  
PCx, XA, XPRESS

**MILP** Mixed Integer Linear Programming  
BONSAIG, CPLEX, FortMP, GLPK, MOSEK, OSL, XA, XPRESS

**MINCO** Mixed Integer Nonlinearly Constrained Optimization  
BARON, DICOPT, MINLP, SBB

**NCO** Nonlinearly Constrained Optimization  
CONOPT, DONLP2, FILTER, KNITRO, IPOPT, LANCELOT,  
LOQO, MINOS, MOSEK, PATHNLP, PENNON, SNOPT

**NDO** Nondifferentiable Optimization  
ACCPM, APPS, BT, DFO, NDA

**QP** Quadratic Programming  
CPLEX, GALAHAD (QPA, QPB), LOQO, OOQP, OSL

**SDP and SOCP** Semidefinite and Second Order Cone Programming  
CSDP, CIRCUT, DSDP, MOSEK, PENNON,  
SDP-LR, SDPA, SDPT3, SeDuMi

**SLP** Stochastic Linear Programming  
CPA, DECIS, MSLIP, OSLSE

**SIO** Semi-Infinite Optimization  
NSIPS

**UCO** Unconstrained Optimization  
CGplus, NMTR, VMLM

### 3 Interactive Optimization Systems

Several systems provide a *graphical user interface* (GUI) or *integrated development environment* (IDE) for mathematical optimization.

**MATLAB** [12] has an Optimization Toolbox with a selection of dense and sparse solvers (none of the above!):

linprog, quadprog, bintprog, lsqlin, lsqnonneg, lsqcurvefit, lsqnonlin, fminbnd, fmincon, fminsearch, fminunc, fseminf, fgoalattain, fminimax

**TOMLAB** [16] provides a complete optimization environment for MATLAB users. There are many problem types, many solvers (including CONOPT, CPLEX, DIDO, KNITRO, LGO, MINLP, MINOS, PATH, PENSDP, PENBMI, SNOPT, SOCS, Xpress), a unified input format, automatic differentiation of M-files with ADMAT and MAD, and a GUI for selecting parameters and plotting output.

**AIMMS** [1] includes solvers for LP, MILP, MINLP, NLP, QCP, QP. Its modeling language has historical connections to GAMS.

**GAMS IDE** [8] provides an IDE for GAMS PC installations.

**Frontline Systems** [7] provides Excel spreadsheet and Visual Basic access to optimizers for many problem classes: LP, QP, MILP, NLP, GO, NDO.

**ILOG OPL Studio** [9] is a further IDE for LP, MILP, and constraint-programming applications, based on a C-like modeling language.

**COMSOL Optimization Lab** [6] provides the LP and NLP capabilities of SNOPT to users of COMSOL Multiphysics and COMSOL Script [5]. Most problem classes are handled by SNOPT. A Nelder-Mead simplex algorithm is included for unconstrained optimization without derivatives.

### 4 Sparse Linear Systems

Underlying almost all of the optimization algorithms is the need to solve a sequence of linear systems  $Ax = b$  (where “ $x$ ” is likely to be a *search direction*,  $\Delta x$ ).

We will study the following software for linear systems. Most are implemented in Fortran 77. MA57 includes an F90 interface. UMFPACK is written in C. Some of the iterative solvers are available in F77, F90, and MATLAB from SOL [28]. Note that PETSc [15] provides many direct and iterative solvers for truly large problems.

*Direct methods* factorize  $A$  into a product of triangular matrices, which are sparse and well-defined even if  $A$  is singular or ill-conditioned.

**LUSOL** [22] Square or rectangular  $Ax = b$ ,  $A = LU$ , plus updating

**MA48** [21] Square or rectangular  $Ax = b$ ,  $A = LU$

**MA57** [20] Symmetric  $Ax = b$ ,  $A = LDL^T$  or  $LBL^T$  (may be indefinite)

**UMFPACK** [17, 18] Square  $Ax = b$  (used for MATLAB’s `[L,U,P,Q] = lu(A)`)

**SuperLU** [19, 23] Square  $Ax = b$  (parallel: shared or distributed memory)

**PARDISO** [14] Symmetric or unsymmetric  $Ax = b$  (parallel: shared memory)

*Iterative methods* regard  $A$  as a “black box” for computing matrix-vector products  $Ax$  and/or  $A^T y$  for given  $x$  and  $y$ .

**CG, PCG** [12, 15] Symmetric positive-definite  $Ax = b$

**SYMMLQ** [24, 28, 15] Symmetric nonsingular  $Ax = b$  (may be indefinite)

**MINRES** [24, 28, 15] Symmetric  $Ax = b$  (may be indefinite and/or singular)

**GMRES** [27, 15] Unsymmetric  $Ax = b$

**CGLS, LSQR** [25, 26, 28, 15]  $Ax = b$ ,  $\min \|Ax - b\|_2^2$ ,  $\min \left\| \begin{pmatrix} A \\ \gamma I \end{pmatrix} x - \begin{pmatrix} b \\ 0 \end{pmatrix} \right\|_2^2$

## References

- [1] AIMMS modeling environment. <http://www.aimms.com/aimms>.
- [2] AMPL modeling system. <http://www.ampl.com>.
- [3] BARON global optimization system. <http://archimedes.scs.uiuc.edu/baron/baron.html>.
- [4] CLP open source LP, QP, and MILP solver. <http://www.coin-or.org/faqs.html>.
- [5] COMSOL AB. <http://www.comsol.com>.
- [6] COMSOL Optimization Lab. <http://www.comsol.com>.
- [7] Frontline Systems, Inc. spreadsheet modeling system. <http://www.solver.com>.
- [8] GAMS modeling system. <http://www.gams.com>.
- [9] ILOG OPL Studio modeling environment. <http://www.ilog.com/products/oplstudio>.
- [10] IPOPT open source NLP solver. <https://projects.coin-or.org/Ipopt>.
- [11] lp\_solve open source LP and MILP solver. [http://groups.yahoo.com/group/lp\\_solve/](http://groups.yahoo.com/group/lp_solve/).
- [12] MATLAB matrix laboratory. <http://www.mathworks.com>.
- [13] NEOS server for optimization. <http://www-neos.mcs.anl.gov>.
- [14] PARDISO parallel sparse solver. <http://www.computational.unibas.ch/cs/scicomp/software/pardiso>.
- [15] PETSc toolkit for scientific computation. <http://www.mcs.anl.gov/petsc>.
- [16] TOMLAB optimization environment for MATLAB. <http://tomopt.com>.
- [17] UMFPAK solver for sparse  $Ax = b$ . <http://www.cise.ufl.edu/research/sparse/umfpack>.
- [18] T. A. Davis. *Direct Methods for Sparse Linear Systems*. Fundamentals of Algorithms. SIAM, Philadelphia, 2006.
- [19] J. W. Demmel, S. C. Eisenstat, J. R. Gilbert, X. S. Li, and J. W. H. Liu. A supernodal approach to sparse partial pivoting. *SIAM J. Matrix Anal. Appl.*, 20(3):720–755, 1999.
- [20] I. S. Duff. MA57: a Fortran code for the solution of sparse symmetric definite and indefinite systems. *ACM Trans. Math. Software*, 30(2):118–144, 2004.
- [21] I. S. Duff and J. K. Reid. The design of MA48: a code for the direct solution of sparse unsymmetric linear systems of equations. *ACM Trans. Math. Software*, 22(2):187–226, 1996.
- [22] P. E. Gill, W. Murray, M. A. Saunders, and M. H. Wright. Maintaining LU factors of a general sparse matrix. *Linear Algebra and its Applications*, 88/89:239–270, 1987.
- [23] X. S. Li and J. W. Demmel. SuperLU-DIST: A scalable distributed-memory sparse direct solver for unsymmetric linear systems. *ACM Trans. Math. Software*, 29(2):110–140, 2003.
- [24] C. C. Paige and M. A. Saunders. Solution of sparse indefinite systems of linear equations. *SIAM J. Numer. Anal.*, 12:617–629, 1975.

- [25] C. C. Paige and M. A. Saunders. LSQR: An algorithm for sparse linear equations and sparse least squares. *ACM Trans. Math. Software*, 8(1):43–71, 1982.
- [26] C. C. Paige and M. A. Saunders. Algorithm 583; LSQR: Sparse linear equations and least-squares problems. *ACM Trans. Math. Software*, 8(2):195–209, 1982.
- [27] Y. Saad. *Iterative Methods for Sparse Linear Systems*. SIAM, Philadelphia, second edition, 2003.
- [28] SOL downloadable software. <http://www.stanford.edu/group/SOL/software.html>.

## Exercises

Due Wed April 9

1. Review the GAMS website: <http://www.gams.com>.
  - (a) Try to find all the solvers that can solve LP problems. (This may include nonlinear solvers.)
  - (b) Find the solvers that can handle general NLP problems (with nonlinear objective and/or nonlinear constraints).
  - (c) Is there a solver that can solve LP and QP problems but not general NLPs?
2. Review the AMPL website: <http://www.ampl.com>.
  - (a) Find the solvers that can solve MILP problems (linear programs with some integer variables).
  - (b) Find the solvers that solve general NLP problems using an *interior method*.
3. Review the NEOS Server for Optimization: <http://www-neos.mcs.anl.gov>. In particular, study the FAQ page.
  - (a) For an LP model, is it easy or hard to determine if a feasible solution exists (compared to finding an optimal solution)?
  - (b) In the context of LP and NLP, what does “Programming” really mean?