An Overview Of Software For Convex Optimization

Brian Borchers
Department of Mathematics
New Mexico Tech
Socorro, NM 87801
borchers@nmt.edu

In fact, the great watershed in optimization isn't between linearity and nonlinearity, but convexity and nonconvexity.

R. Tyrrell Rockafellar (1993)

A Paradigm Shift

- From the 1960's through the early 1990's, many people divided optimization in linear programming (convex, non-smooth, problems solved by the simplex method) and nonlinear programming (smooth, typically convex, problems solved by gradient based local search methods.)
- The development of interior point methods for linear programming (and later for second order cone and semidefinite programming) in the 1980's and 1990's led to a reassessment of this division of the subject.
- In the new paradigm, the division is between convex optimization problems that can be solved efficiently and other optimization problems that can't be solved efficiently.

A Hierarchy Of Convex Optimization Problems

$$LP \subset CQP \subset SOCP \subset SDP \subset CP.$$

- Polynomial time interior point methods for LP have been generalized to solve problems in this hierarchy up to the level of SDP.
- Many other convex optimization problems can be formulated as structured convex optimization problems that fit into this classification scheme.
- Many nonconvex optimization problems have convex relaxations that can be fit into this scheme. The relaxations can be used to compute bounds on the nonconvex optimization problem.

Two Approaches To Solving Convex Problems

- One approach is to put the problem into the form of an LP, CQP, SOCP, or SDP, and then use a solver for that class of problems.
- An alternative is to develop a special purpose solver for your original problem.
- The first approach allows you to make use of existing solver software and is often the quickest way to get something working. On the other hand, producing a special purpose solver for your problem typically takes more work but might result in a code that can solve the problem more efficiently.
- For the rest of this talk I'll focus on the first approach, since software in the second approach tends to be extremely problem specific.

Interfacing To A Solver

- The solvers discussed in this talk are typically called as subroutines from a main program written in Fortran, C, MATLAB, or Python. The subroutine interfaces are highly specific to a particular solver.
- In addition to subroutine interfaces, most solvers can read problems and write out solutions that have been written in standard file formats. These include the MPS format for linear programming problems and the SDPA sparse file format for semidefinite programming problems. The Optimization Services project of COIN-OR has developed a new XML file format for optimization problems that includes SDP and SOCP constraints. These file formats are useful for exchanging problems with other researchers and for testing a problem with multiple solvers.

Modeling Systems For Convex Optimization

- The process of reformulating your particular optimization problem into a standard form LP, CQP, SOCP, or SDP can be somewhat tricky. There are lots of techniques to learn, and it's easy to make mistakes in doing the reformulation by hand.
- Modeling symptoms make it possible to formulate your optimization problem in a straight forward way. The modeling system takes care of reformulating the problem into standard form and interfacing to a solver, as well as converting the solution back into the original problem formulation.
- Many of the modeling systems described here are setup to work with multiple solvers. It's not uncommon to discover that one solver works better than the others on your particular problem.

Modeling Systems For Convex Optimization

- Modeling systems for linear programming, nonlinear programming, and integer linear and nonlinear programming have been around for a long time.
- Unfortunately, the main stream commercial optimization modeling systems have not moved quickly to support convex optimization modeling.
- GAMS now has limited support for second order constraints, but they don't offer any way to express semidefinite programming constraints.
- AIMMS has support for robust optimization using MOSEK as a second order cone programming solver.

Modeling Systems For Convex Optimization

Several new free/open source modeling systems have come out with the ability to handle convex optimization problems. These include:

- CVX. Built on top of MATLAB, this package can interface to either the SDPT3 or SeDuMi solvers. CVX implements a "disciplined convex programming" philosophy- the language is designed so that you can only express convex optimization problems.
- CVXMOD. Written in Python, this modeling system works with the CVXOPT solver.
- YALMIP. Written in MATLAB, this modeling system supports convex optimization as well as integer programming and nonlinear programming. It works with many solvers, including CPLEX, GUROBI, MOSEK, SDPT3, SEDUMI, CSDP, SDPA, PENNON, ...

Modeling Systems For Convex Optimization

The term "Linear Matrix Inequalities" is often used by electrical engineers to discuss semidefinite programming, particularly in the context of control problems.

- LMI lab (Robust Control Toolbox). Commercial. This package is an add-on to MATLAB that includes a modeling language and a solver for linear matrix inequalities.
- LMIlab translator. Free/Open Source. Translates LMI lab problems into SDP form for solution by another solver such as SDPT3 or SeDuMi.
- xLMI. Free/Open Source. Another MATLAB package that implements a language similar to the LMI toolbox, but using SeDuMi as the solver.

Modeling Systems For Robust Convex Optimization

- AIMMS. Commercial. Allows for robust optimization on linear programming and mixed integer linear programming problems. Works with MOSEK or CPLEX as an SOCP solver.
- ROME. Free/Open Source. Supports distributionally robust optimization as well as more conventional uncertainty sets.
- YALMIP. Free/Open Source. Allows for robust optimization involving uncertainty in LP, SOCP, and SDP constraints.

Modeling Systems For Polynomial Optimization

- GLOPTIPOLY. Free/Open Source. Written in MATLAB.
- SOSTOOLS. Free/Open Source. Written in MATLAB. Requires Symbolic Computation Toolbox.
- SparsePOP. Free/Open Source. Works with SDPA or SeDuMi solvers.

Example: Minimum Rank Matrix Completion

Suppose that we're given some of the elements of an m by n matrix X, along with the information that the matrix is of low rank. Is it possible to recover the matrix?

We could try to solve the optimization problem

$$\begin{array}{ll} \min & \operatorname{rank}(X) \\ & X_{i,j} = M_{i,j} \;\; (i,j) \in \Omega. \end{array}$$

Unfortunately, this is a nonconvex optimization problem that can be extremely hard to solve exactly.

The Nuclear Norm

The nuclear norm of an m by n matrix X is the sum of the singular values of X.

$$\|X\|_* = \sum_{i=1}^{\min(m,n)} \sigma_i(X)$$

It's relatively easy to show that $\|X\|_*$ is a proper matrix norm, and that it is a convex function of X. Note that in the special case where X is symmetric and positive semidefinite, the singular values of X are the eigenvalues of X, and the nuclear norm of X is just the sum of the eigenvalues of X or equivalently, the nuclear norm is the trace of X.

The Nuclear Norm Heuristic

A heuristic for the minimum rank matrix completion problem that is analogous to minimizing the 1-norm to find a sparse solution to a linear system of equations is to minimize the nuclear norm of X.

$$\begin{array}{ll} \min & \|X\|_* \\ & X_{i,j} = M_{i,j} \;\; (i,j) \in \Omega. \end{array}$$

The Connection To Sparse Solutions

Notice that since rank(X) is the number of nonzero singular values of X, the rank minimization problem is

$$\begin{array}{lll} \min & \|\sigma(X)\|_0 & \\ & X_{i,j} & = \quad M_{i,j} \quad (i,j) \in \Omega. \end{array}$$

Since $\|X\|_* = \|\sigma(X)\|_1$, the nuclear norm heuristic is minimizing

$$\begin{array}{lll} \min & \|\sigma(X)\|_1 & \\ & X_{i,j} & = \quad M_{i,j} \quad (i,j) \in \Omega. \end{array}$$

A Useful Lemma

Lemma: (Fazel, 2002) Given a matrix X, $\text{rank}(X) \leq r$ if and only if there exist symmetric and positive semidefinite matrices Y and Z such that

$$\text{rank}(Y) + \text{rank}(Z) \leq 2r$$

and

$$W = \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} \succeq \mathbf{0}.$$

By this lemma, minimizing the rank of the symmetric and positive definite matrix W is equivalent to minimizing the rank of X.

The Nuclear Norm Heuristic

Applying the nuclear norm heuristic to the symmetric matrix in the lemma, we get

$$\begin{array}{lrcl} \min & \frac{1}{2}(\mathrm{trace}(Y) + \mathrm{trace}(Z)) & & \\ & X_{i,j} & = & M_{i,j} \;\; (i,j) \in \Omega \\ & \begin{bmatrix} Y & X \\ X^T & Z \end{bmatrix} & \succeq & \mathbf{0}. \end{array}$$

This is a semidefinite programming problem that can be solved efficiently by interior point methods, at least for reasonably small X matrices. I haven't proved it here, but it can be shown that the optimal value of this SDP is equal to the optimal value of the nuclear norm minimization problem on the original matrix X.

The Nuclear Norm Heuristic

```
function X=completelowrank(M)
[m,n]=size(M);
cvx_begin sdp
  variable X(m,n)
  minimize (norm_nuc(X))
  subject to
    for i=1:m
      for j=1:n
        if (~isnan(M(i,j)))
          X(i,j)==M(i,j)
        end
      end
    end
cvx_end
```

Discussion: Minimum Rank Matrix Completion

- The example illustrates a common pattern- replace a difficult to solve nonconvex optimization problem with a related surrogate/relaxation that is a convex optimization problem.
- The example also shows how a modeling system can vastly simplify the process of converting a convex optimization problem into standard form.
- It's important to note that this is really not the best way to solve a large scale minimum rank matrix completion problem- as the matrix size and number of specified elements increases, the SDP formulation will become impractical to solve with a primal dual interior point method. There are special purpose algorithms for this problem that are much more efficient in practice.

Primal–Dual Barrier Methods For SOCP/SDP

- Primal-Dual barrier methods are most commonly used in general purpose solvers for SOCP and SDP. In each iteration, the primal-dual barrier method computes a Newton's method step for a perturbed version of the KKT conditions.
- The Newton's method step involves the solution of a symmetric and positive definite system of equations that becomes very ill-conditioned as we approach an optimal solution.
- The method can be implemented as either an infeasible interior point method or using a self-dual embedding.
- Although several slightly different search directions have been considered by various researchers, the HKM direction is now used by most of the solvers.

Primal–Dual Barrier Methods For SOCP/SDP

- We'll consider the storage requirements and computational complexity for an SDP with m constraints and matrices of size n by n.
- The primal-dual method requires storage for an m by m symmetric and positive definite matrix that is usually fully dense. It also requires storage for a number of n by n matrices, some of which may be sparse. In practice, the $O(m^2 + n^2)$ storage requirement is often more limiting than the computational complexity of the iterations.
- Each iteration of the algorithm requires $O(mn^3 + m^2n^2)$ time for the computation of the m by m system matrix, $O(m^3)$ time to compute the Cholesky factorization of this matrix, and $O(n^3)$ time for various factorizations and multiplications of n by n matrices.

Primal–Dual Codes For SOCP and SDP

- CPLEX. Commercial. LP+SOCP.
- CSDP. Free/Open Source. LP+SDP. Written in C, with interfaces to MATLAB, R, and Python. Parallel versions available.
- CVXOPT. Free/Open Source. LP+SOCP+SDP. Written in Python.
- MOSEK. Commercial. LP+SOCP.
- SeDuMi. Free/Open Source. LP+SOCP+SDP. Written in MATLAB plus MEX.
- SDPA. Free/Open Source. LP+SDP. Written in C+. Parallel and extended precision versions available.
- SDPT3. Free/Open Source. LP+SOCP+SDP. Written in MATLAB plus MEX.

Other General Purpose Codes For SOCP/SDP

- ConicBundle. LP+SOCP+SDP. Free/Open Source. Uses a bundle method.
- DSDP. LP+SDP. Free/Open Source. Uses a dual interior point method.
- LOQO. LP+SOCP. Commercial. Treats the SOCP as a nonlinear programming problem.
- PENNON. LP+SDP. Commercial. Uses an augmented Lagrangian method.
- SDPLR. LP+SDP. Free/Open Source. Uses a low rank factorization within an augmented Lagrangian method.

Issues To Consider In Picking A Solver

- Accuracy. Do you need solutions that are accurate to seven digits, or are solutions accurate to two or three digits good enough?
- Robustness. Do you need a solver that is extremely reliable in producing solutions to the required accuracy or can you live with failures?
- Interfaces to Modeling Systems. Does the solver work with your favorite modeling system?
- Performance. How fast is the solver? Does it work in parallel on multiprocessor systems or distributed memory clusters?

Conclusions

- We've discussed a general approach to solving a convex optimization problem by putting it into LP, SOCP, or SDP standard form and using a general purpose solver. You should also consider the alternative of using a special purpose solver for your particular problem.
- You can convert your problem to standard form by hand, or you can use a modeling system to do the reformulation.
- Primal–dual interior point methods are most widely used in general purpose solvers, but these methods are limited to small to medium sized problems.
- Other algorithms have so far not been as robust as general purpose SOCP/SDP solvers. There's definitely a need for a robust and reasonably accurate first order solver for SOCP/SDP problems.

Useful Resources

Hans Mittelmann at ASU maintains a useful collection of web pages on optimization that includes

- A decision tree for optimization software that can be used to help find available codes for a class of problems.
  http://plato.asu.edu/guide.html
- benchmarks on many classes of optimization problems, including LP, SOCP, and SDP. These benchmarks can sometimes be misleading. It's important to also check the accuracy of the solutions returned by the different solvers by carefully examining the solution logs.
  http://plato.asu.edu/sub/benchm.html

Useful Resources

- AIMMS. A commercial modeling package for LP, MILP, and robust LP and MILP problems. http://www.aimms.com/
- ConicBundle. An open source convex optimization code with special support for SDP and SOCP constraints. www-user.tu-chemnitz.de/~helmberg/ConicBundle/
- CPLEX. A commercial solver for LP, QP, SOCP, and mixed integer programming problems. http://www-01.ibm.com/software/integration/optimization/cplex-optimizer/
- CSDP. An open source SDP solver, written in C, with parallel versions and MATLAB, R, and Python interfaces. https://projects.coin-or.org/Csdp/

Useful Resources

- CVX. An open source modeling system, written in MATLAB, that works with SDPT3 and SeDuMi as solvers. http://cvxr.com/cvx/
- CVXMOD. An open source modeling system, written in Python, that works with the CVXOPT solver. http://cvxmod.net/
- CVXOPT. An open source solver for conic optimization problems, written in Python, that uses the Nesterov-Todd scaling. Also interfaces to MOSEK and DSDP. http://abel.ee.ucla.edu/cvxopt/
- DSDP. An open source dual interior point method for LP+SDP. http://www.mcs.anl.gov/hs/software/DSDP/

Useful Resources

- GAMS. A commercial modeling system for linear, nonlinear, mixed integer linear and nonlinear, and second order cone programming problems. http://www.gams.com/
- Gloptipoly3. An open source modeling system for polynomial optimization. http://homepages.laas.fr/henrion/software/gloptipoly3/
- Optimization Services. An XML standard for encoding problems and solutions. http://www.optimizationservices.org/
- PENNON. A commercial solver that uses an augmented Lagrangian method to solve problems with SDP constraints. http://www.penopt.com/
- ROME. An open source modeling system for robust optimization. http://robustopt.com.

Useful Resources

- SDPA. An open source solver for LP+SDP, written in C++, including parallel and extended precision versions. http://sdpa.indsys.chuo-u.ac.jp/sdpa/
- SDPT3. An open source solver for LP+SOCP+SDP, written in MATLAB. http://www.math.nus.edu.sg/∼mattohkc/sdpt3.html
- SeDuMi. An open source solver for LP+SOCP+SDP, written in MATLAB. http://sedumi.ie.lehigh.edu/
- SOSTOOLS. An open source modeling system for polynomial optimization, written in MATLAB, works with SDPT3 and SeDuMi. http://www.cds.caltech.edu/sostools/
- SparsePOP. An open source modeling system for polynomial optimization which uses SDPA or SeDuMi as a solver. http://www.is.titech.ac.jp/∼kojima/SparsePOP/SparsePOP.html

Useful Resources

- YALMIP. An open source modeling system for convex (and some nonconvex) optimization problems that works with many different solvers. http://users.isy.liu.se/johanl/yalmip/