Research on Optimization

As a project scientist, my main research contributions have been concerned with the development of methods for solving large-scale optimization problems of the form

$$\min_{x \in \mathbb{R}^n} \quad f(x)$$
subject to \quad $${\ell \leq c(x) \leq u},$$

where $$f : \mathbb{R}^n \rightarrow \mathbb{R}$$ and $$c(x) : \mathbb{R}^n \rightarrow \mathbb{R}^m$$ are twice-continuously differentiable functions. Generally speaking, my research focuses on problems involving hundreds of thousands of variables. The main goals of my research are:

- to implement optimization methods in robust software packages for the scientific and engineering communities;
- to develop methods with global and local convergence properties that are sensible from a practical standpoint; and
- to apply optimization methods to problems in engineering and applied sciences.

A significant portion of my research is dedicated to implementation, verification, and testing of software based on the methods. My recent research has focused on several topics, including the development of sequential quadratic programming methods that utilize exact second derivative information; the application of nonlinear optimization methods to other areas including power systems engineering and aerospace engineering; the development and implementation of methods for unconstrained and bound-constrained optimization problems; and the maintenance and upgrade of our existing optimization software packages.

After receiving my doctoral degree, I continued my work on the solution of large-scale quadratic programming (QP) problems that optimize a general quadratic objective function subject to linear constraints on the variables. Problems have the form

$$\min_{x \in \mathbb{R}^n} \quad q(x) = c^T x + \frac{1}{2} x^T H x$$
subject to \quad $$Ax = b, \quad \ell \leq x \leq u,$$

where $$q(x)$$ is the quadratic objective function with the Hessian matrix $$H$$ not necessarily positive definite, $$A$$ is the $$m \times n$$ constraint matrix, and constant vectors $$b$$ in $$\mathbb{R}^m$$ and $$c, \ell$$ and $$u$$ in $$\mathbb{R}^n$$. As the Hessian matrix $$H$$ is not necessarily positive definite, the quadratic objective of the problem need not be convex and the resulting QP is an NP-hard problem. In the nonconvex case, convergence will be to a point satisfying the second-order necessary conditions for optimality, which may or may not be a local minimizer.

The method considered defines a primal–dual search pair associated with the solution of an equality-constrained subproblem involving a “working set” of linearly independent constraints. The working set is specified by an active-set strategy that controls the inertia (i.e., the number of positive, negative and zero eigenvalues) of the associated KKT matrix. It is established that this inertia-controlling strategy guarantees that each set of KKT equations is well-defined and nonsingular, allowing the use of third-party linear solvers in the solution of the KKT equations. This work led to the creation of the QP software package SQIC software package (see [1]).

In subsequent years, my work expanded naturally to consider sequential quadratic programming (SQP) methods for nonlinear optimization. SQP methods solve problem (1) by finding an approximate solution to a sequence of QP subproblems in which a quadratic model of the objective function is minimized subject to the linearized constraints.
Typical methods will define a convex QP subproblem with a positive semidefinite quasi-Newton approximate Hessian because of difficulties that can arise from a nonconvex objective function. If the problem (1) is not convex, the Hessian of the Lagrangian may be indefinite, even in the neighborhood of a solution. This situation creates a number of difficulties in the formulation and analysis of a conventional SQP method. (i) In the nonconvex case, the QP subproblem may be unbounded below with many local solutions. In addition, nonconvex QP is NP-hard—even for the calculation of a local minimizer. The complexity of the QP subproblem has been a major impediment to the formulation of second-derivative SQP methods. (ii) If the Hessian of the Lagrangian is not positive definite, then computed search directions may not be a descent direction for the merit function. This implies that an alternative direction must be found or the line search must allow the merit function to increase on some iterations.

With the development of SQIC, my colleagues and I began developing a new SQP algorithm that defines the QP subproblems using the exact second derivatives of the problem (see [2], [3]). A convexification scheme to define a local convex approximation of a nonconvex problem was developed to provide the basis of an SQP method that first uses a quasi-Newton method to identify an estimate of the working set at a solution. This estimate is used to initiate a sequence of QP subproblem defined by the Hessian of the Lagrangian. These methods became the basis of the second-derivative solvers in the dense SQP package DNOPT of Gill, Saunders and Wong and the forthcoming large-scale package SNOPT9.

More recently, I have focused on research on the solution of unconstrained and bound-constrained optimization problems, which play an important role in machine learning and data science. My collaborators and I formulated and analyzed a framework for a general class of projected-search methods with favorable global convergence properties. As the objective function is not differentiable along the piecewise-linear path generated by a projected-search method, it is not possible to use a line search based on satisfying the Wolfe conditions, which involve the derivatives at two points on the search path. We then developed a modified Wolfe line-search method suitable for piecewise differentiable functions. This quasi-Wolfe line search makes it possible to compute a step length along the computed search direction. Standard existence and convergence results associated with a conventional Wolfe line search are extended to the quasi-Wolfe case (see [4]). A limited-memory reduced-Hessian method utilizing the aforementioned quasi-Wolfe line search was implemented as Fortran software package LRHB (see [5]). In this method, computed search directions are projected into the gradient subspace defined by the free variables at the current iteration to maintain feasibility of the bounds of the problem. Numerical results from the software implementation are presented, together with comparisons against other well-known codes for bound-constrained optimization.

Grid Optimization Competition

In 2019, I was invited to join a team participating in the year-long Grid Optimization (GO) Competition Challenge 1, developed by the Department of Energy’s Advanced Research and Projects Agency - Energy (ARPA-E) [6]. The competition is part of a larger mission of ARPA-E to modernize and innovate software to achieve a modern power grid. The competition would also incentivize entrepreneurial efforts that align with ARPA-E’s mission to innovate grid software and empower widespread adoption of emerging technologies with the goal of saving billions of dollars in an energy sector with revenues reaching close to $400B per year. To encourage participation in the competition, ARPA-E offered total prizes of up to $4 million, with participants encouraged to enter as “proposal entrants” who were supported by DOE grants of up to $250,000 or “open entrants”.

Our team “GO-SNIP” comprised of Frank E. Curtis from Lehigh University, Daniel Molzahn from Georgia Tech, and Andreas Wächter and Ermin Wei from Northwestern University. For the competition, teams were required to develop and submit software to solve an optimal power flow problem that minimized operating costs while satisfying both engineering constraints (e.g., lineflow, voltage magnitude, and power generation limits) and the power flow equations that model the physics of the transmission network. The competition was structured as a year-long project beginning in the fall of 2018 with three “trial event” checkpoints before the final competition submission deadline in the winter of 2019. The trial events involved increasingly difficult datasets released before and after each event for testing and evaluation. For evaluation and scoring purposes, the competition had two scoring methods,
one focusing on the final objective value determined by the algorithm and another focusing on the robustness of the algorithm relative to the other team submissions. The final objective value of the optimal power flow problem reflects both the generation costs associated with power production and any penalty terms that model the soft constraints, thus ensuring feasibility of each team’s outputs despite constraint violations. Algorithms were expected to run within strict time limits of 5 minutes for the “real-time” division and 45 minutes for the “offline” division.

As a member of the team, I worked on the coding, development, and testing of the grid software, utilizing nonlinear optimization tools known within the optimization community but not generally employed by power systems researchers. Furthermore, theoretical and empirical work was performed on various optimization techniques to create a more efficient algorithm for the competition. The optimal power flow problem is a difficult one to solve due to its size and mathematical characteristics. In order to speed up solution times, methods for identifying active constraints and degeneracy within the constraints were investigated to bridge the jump from interior-point methods to active-set-based sequential quadratic programming methods. Heuristics were developed to speed up solution times.

The results were announced by the Secretary of Energy in February of 2020 with our team finishing in second place overall out of 27 participating teams [7]. Our team “GO-SNIP” was awarded a total of $400,000 in prize money for our performance within four divisions of the competition. As part of success, the team was invited to the Outreach Event held by ARPA-E with participants from the competition as well as industry leaders and partners in the energy sector.

The competition has also lead to the writing of an article that presents the details of the algorithm submitted to the competition along with heuristics developed to enhance performance. Developments unique to our team’s research, including contingency ranking and screening techniques, the handling of complementarity constraints (which are generally not handled by methods for nonlinear optimization), and preprocessing of the optimal power flow model information are also detailed.

Our team’s participation in Challenge 2 of the Grid Optimization competition began in September of 2020 and is currently on-going.

**Numerical software**

Throughout my career, I have focused on the implementation and design of robust numerical optimization software and have been involved with several packages including:

- **SQIC** for large-scale quadratic programming;
- **DNOPT** for dense, medium-scale nonlinear programming and its quadratic programming solver **DQOPT**;
- **SNOPT7** for sparse, large-scale nonlinear programming and its quadratic programming solver **SQOPT7**;
- **LRHB** for large-scale bound-constrained and unconstrained problems;
- **PDQP**, a primal–dual active-set method for quadratic programming (see [8]); and
- **SNOPT9**, a new version of **SNOPT7** that utilizes exact second-derivative information.

In discussions and consultations with various research groups, I have also implemented and released several upgrades for the nonlinear optimization software package **SNOPT7**. The most recent and prominent of these was the improvement of the **HOT start** feature that allows users to “pick up” the solution of a nonlinear optimization problem from where they left off. For example, in some applications of aerospace engineering, users may start to solve a problem with **SNOPT7** but terminate before a solution is found. Engineers may then examine the output of **SNOPT7** and adjust parameters, and attempt to continue the solution of their problem. Because problems can be complex and require, in some cases, hours or even days to evaluate the mathematical functions, efficiency is crucial. The improvements to the **HOT start** feature were critical by reducing the number of function evaluations required.
To keep up with new and emerging advances in numerical and scientific computing, I have developed and maintained various interfaces in Matlab, C, C++, Python, and Julia for SNOPT7. The most recent release was for the Julia computing language; while still in its nascent stages, Julia has emerged as a high-quality tool that has adopted many of the best features of Python and Matlab.

Web-based software

In early 2017, I launched the UC San Diego optimization website (https://www.ccom.ucsd.edu/~optimizers). The optimization group recognized the need for a modern website that could provide up-to-date documentation and downloadable libraries for the optimization community. In addition to providing downloads of our optimization software, the website also provides software to licensed users on several platforms and in a variety of formats that are compatible with the Fortran, C, C++, Julia, Matlab, and Python languages.

The website uses Django, a Python-based free and open-source web framework, and a database server to automate the creation and distribution of license files and software to users. Since February of 2017, over 2000 users have used our website to download precompiled libraries of our software.

Work on the NEOS Server

The NEOS Server is a free internet-based service for solving numerical optimization problems hosted by the Wisconsin Institute for Discovery at the University of Wisconsin in Madison. The server provides access to more than 60 state-of-the-art solvers in more than a dozen optimization categories. As the optimization support specialist for the NEOS Server, my responsibilities include assisting in the administration of the server, maintaining and implementing optimization solvers on the server, managing content for the NEOS Guide (https://www.neos-guide.org), implementing additional functionality to the server, and serving as the first contact for users of the server.

Since 2017, I have improved documentation of the services provided by NEOS and added several solvers and new input file formats to the server. One of my first tasks was to upgrade the “driver” system that provides the interface between optimization solvers and the NEOS server. The upgrade made the internal code more manageable with less code duplication. I have also implemented several upgrades to the server to facilitate new features for GAMS solvers (The General Algebraic Modeling System). In 2018, I was invited to participate in the IMA special workshop “COIN fORgery: Developing Open-Source Tools for Operations Research” as a member of the NEOS team.

In 2020, the NEOS team completed a major upgrade of the internal server code with a transition to Python3. The majority of the work was performed by myself and another member of the team. As part of the transition, I also spearheaded the move to the Jinja templating system to ease maintenance of the webpages for the NEOS Server website.

References


