USRA REPORT

Modeling Continuous Optimization Algorithms

Over the summer, under the supervision of Professor Philip Loewen, and with great help from PhD student Mclean Edwards, I investigated recent iterative algorithms for solving continuous optimization problems and helped to create Python implementations for the optimization community.

During the first month of my work, I learned to use Python to solve simple problems, such as integrations and differentiation, and to draw certain plots which would be useful in the future project, such as function plots and contour maps.

Then, I took a deep look at Newton's method on solving optimization problems, particularly on quadratic functions. However, the quadratic convergence of Newton's method of minimization was guaranteed only for initial points sufficiently near a critical point satisfying certain hypotheses. For starting point far from a minimum, or in the absence of those hypotheses, Newton's Method might stall or even diverge. In order to overcome this obstacle, we explored various approaches. There are many ways to avoid this problem. One of the most popular was the Broyden-Fletcher-Goldfarb-Shanno (BFGS) method. The BFGS method has an important advantage over Newton's method: it does not require explicit coding of the Hessian matrix containing all second derivatives of the function to be minimized; the Hessian is essential in Newton's method.

After studying these methods, I designed a program to display the minimizing path generated by iterations of the BFGS method. Since part of our goal was to provide an interactive visual representation for a library of algorithms that are currently under development, I added interactive features to my program, allowing users to be able to get a different path starting from any point they clicked in the plotting window of my program. Towards the end of the summer, we decided to modify my program so we could compare different algorithms for approximate minimization. We ran our self-made implementation of the BFGS method against some other methods from Scipy, a Python package, on three functions of two real variables, namely, a power function, a convex function involving trigonometry, a one non-convex function involving trigonometry. It turned out that our self-made BFGS could not give us the correct minimum points with different initial points and functions all the time. In fact, only one method consistently succeeding in doing so – the Powell method as implemented in Scipy.

Overall, BFGS method is a sufficient way of solving unconstrained nonlinear optimization problems. However, if the objective function is complicated enough, BFGS may not work as well. It is known that the BFGS method does a great job at solving unconstrained nonlinear optimization problems of functions with high dimensions. However, our tests show that BFGS may not be the best approach for solving this type of problem with functions with low dimensions. Also, since Powell method from Scipy performed amazingly well on our test functions, it is worth investigating thoroughly how it works.

Yu (Scott) Zhou
Sept 11, 2008