Stanford University, Management Science & Engineering (and ICME)

MS&E 318 (CME 338) Large-Scale Numerical Optimization

Instructor: Michael Saunders Spring 2018

Notes 3: **Suggested projects**

The class project can be one of your own choice, or perhaps one of the suggestions below. It's always good to write code that can be used by others (perhaps forever!). Ideally we will be able to add your work to the list of freely available software here: `http://stanford.edu/group/SOL/software/`.

At some point the projects may seem open-ended. Don't worry. Grades will be awarded based on best effort within a reasonable time. Perhaps you will be motivated to continue adding features during the summer.

1 MINRES with local reorthogonalization

This is intended for users of GMRES on symmetric systems for which the preconditioner is so good that not many iterations are expected. Like LSMR, it will still be efficient on problems of arbitrary size if the input parameter `localSize` is no bigger than 10 or 20 (say).

GMRES is intended for square unsymmetric systems, but it is often used on symmetric saddle-point systems because it stores all Arnoldi vectors and will generally take fewer iterations than MINRES. With local reorthogonalization we can make MINRES as effective as GMRES while retaining symmetry. It feels "better".

A preliminary version of MINRES with local reorthogonalization has been implemented in `http://stanford.edu/group/SOL/software/minres/minresReorthog.zip` by ICME student Danielle Maddix. This could be used as a starting point. The project is to make sure that the code works with a positive-definite preconditioner `M`.

2 DQGMRES

For $n \times n$ systems $Ax = b$, the most effective method is often GMRES (Saad and Schultz [5]) or GMRES(m) for some restart frequency $m \le n$ that limits the number of n-vectors required but may interfere with convergence. GMRES uses the Arnoldi process for orthogonal transformation of A to upper Hessenberg form. A lesser known method DQGMRES(m) by Saad and Wu [6] limits the storage required by truncating each iteration of the Arnoldi process to use at most m of the most recently generated n-vectors. It is sure to converge for any $m > 1$ and seems to deserve further attention.

3 USYMQR

Saunders, Simon, and Yip [7] proposed two methods for solving square systems $Ax = b$ based on orthogonal tridiagonalization of A. Twenty years later, Reichel and Ye [4] discussed the same tridiagonalization for rectangular systems. We have a Fortran 90 implementation of USYMLQ for compatible systems $Ax = b$. It is of interest to implement USYMQR for both $Ax = b$ and least-squares problems $\min \|Ax - b\|_2$.

4 PDCO search directions

Currently, `options.Method` = 1, 2, 3, 4 tells PDCO to use certain methods to compute the dual search direction Δy and use it to obtain the primal direction Δx. Implement some or all of `options.Method` = 11, 12, 13 to solve for Δx before Δy. The choice of `Method` may depend on whether A is over-determined or under-determined.

5 BPprimal with general bounds

`BPprimal.m` is a MATLAB code that solves the regularized BPDN problem

$$\begin{array}{ll}\underset{x,\,y}{\text{minimize}} & \|x\|_1 + \tfrac{1}{2}\delta\|x\|_2^2 + \tfrac{1}{2}\lambda\|y\|_2^2 \\ \text{subject to} & Ax + \lambda y = b.\end{array}$$

The active-set method for `BPprimal.m` is the reduced-gradient method with basic variables y throughout ($B = \lambda I$). The superbasic variables are nonzero elements of x associated with their support S from A. The project task is to include general bounds:

$$\begin{array}{ll}\underset{x,\,y}{\text{minimize}} & \|x\|_1 + \tfrac{1}{2}\delta\|x\|_2^2 + \tfrac{1}{2}\lambda\|y\|_2^2 \\ \text{subject to} & Ax + \lambda y = b, \quad \ell \le x \le u.\end{array}$$

Download `http://stanford.edu/group/SOL/software/asp/matlab/asp-v1.0.zip` to obtain the MATLAB package ASP. `BPprimal.m` uses several auxiliary functions from another solver `BPdual.m` within ASP.

6 Complex-step estimates of gradients

Let $F(x)$ be a smooth real scalar function of the real scalar x with gradient $g(x)$ and Hessian $H(x)$, and let h be a small real step. Taylor series expansions give

$$\begin{aligned} F(x+h) &= F(x) + hg(x) + \frac{h^2}{2!}H(x) + O(h^3), \\ F(x-h) &= F(x) - hg(x) + \frac{h^2}{2!}H(x) + O(h^3). \end{aligned}$$

Thus, $g(x)$ can be estimated by forward differences or central differences as

$$\begin{aligned} g_f(x) &= (F(x+h) - F(x))/h + O(h), \\ g_c(x) &= (F(x+h) - F(x-h))/2h + O(h^2), \end{aligned}$$

where $h \approx \epsilon^{1/2}$ and $\epsilon^{1/3}$ respectively give best accuracy on a machine with floating-point precision ϵ (according to [2], under certain assumptions). In double-precision arithmetic, this means $h \approx 10^{-8}$ and 10^{-5}. The error in the gradient approximations is then $O(10^{-8})$ and $O(10^{-10})$.

If $F(z)$ is an analytic function of complex z and h is a tiny step off the real axis, the Taylor series is

$$F(x+ih) = F(x) + ihg(x) - \frac{h^2}{2!}H(x) - \frac{ih^3}{3!}F'''(x) + O(h^4),$$

and the complex-step estimate of the gradient at real x is

$$g_{cs}(x) = \text{imag}(F(x+ih))/h + O(h^2),$$

where $\epsilon \le h \le \epsilon^{1/2}$ gives best accuracy, as described by Moler [3]. Thus in double precision with $h = \epsilon^{1/2} \approx 10^{-8}$ or smaller down to ϵ, the error in the gradient approximation should be $O(\epsilon) \approx 10^{-16}$ (full precision).

The project can involve a particular function $F(x)$[1] that arises in optimal design of computer experiments [1]. We have a Fortran 90 implementation of the function and have been optimizing it using MINOS in quadruple precision, with gradients estimated by forward and central differencing. For difficult cases, $O(\epsilon^{1/2})$ accuracy in the gradients is not enough (where quad $\epsilon \approx 10^{-34}$!). The function involves the error function `erf(x)`. The complex-step method therefore needs the complex error function `erf(x+ih)`, which Fortran doesn't have! Luckily h is very small. We can construct a series that converges rapidly.

[1]The IMSPE function (Integrated Mean-Square Prediction Error).

References

[1] S. B. Crary. New research directions in computer experiments: ϵ-clustered designs. In *JSM Proceedings, Statistical Computing Section*, pages 5692–5706, Alexandria, VA, 2013. `https://drcrary.files.wordpress.com/2014/06/crary-src2012proceedingsrev20131010.pdf`.

[2] P. E. Gill, W. Murray, and M. H. Wright. *Practical Optimization.* Academic Press Inc. [Harcourt Brace Jovanovich Publishers], London, 1981.

[3] C. B. Moler. Complex Step Differentiation. Cleve's Corner: Cleve Moler on Mathematics and Computing. `https://blogs.mathworks.com/cleve/2013/10/14/complex-step-differentiation/`.

[4] L. Reichel and Q. Ye. A generalized LSQR algorithm. *Numer. Linear Algebra Appl.*, 15:643–660, 2008.

[5] Y. Saad and M. H. Schultz. GMRES: a generalized minimum residual algorithm for solving nonsymmetric linear systems. *SIAM J. Sci. and Statist. Comput.*, 7:856–869, 1986.

[6] Y. Saad and K. Wu. DQGMRES: a direct quasi-minimal residual algorithm based on incomplete orthogonalization. *Numer. Linear Algebra Appl.*, 3(4):329–343, 1996. A copy from the Stanford library is here: `http://stanford.edu/class/msande318/refs/DQGMRES.pdf`.

[7] M. A. Saunders, H. D. Simon, and E. L. Yip. Two conjugate-gradient-type methods for unsymmetric linear equations. *SIAM J. Numer. Anal.*, 25(4):927–940, 1988.