Foundations of Data Science lectures
Continuous Optimization: Computational Exercises

Dr Lennon Ó Náraigh

8th December 2023

In practice, solving the 1D sub-problem $\alpha_k = \arg\min_{\alpha>0} f(\boldsymbol{x}_k + \alpha\boldsymbol{p}_k)$ is overkill. Instead, a simpler procedure suffices. One example of such is back-tracking line search. The algorithm is given here:

Algorithm 1 Backtracking Line Search

Choose an initial guess for α_k, call it α. Fix $\rho \in (0,1)$ and $c \in (0,1)$.
while $f(\boldsymbol{x}_k + \alpha\boldsymbol{p}_k) > f(\boldsymbol{x}_k) + c\alpha\boldsymbol{p}_k \cdot \nabla f(\boldsymbol{x}_k)$ **do**
  $\alpha \leftarrow \rho\alpha$.
end while

The first question concerns the implementation of this algorithm.

1. Program the steepest-decent and Newton algorithms using the backtracking line search algorithm. Use them to minimize the Rosenbrock function:

$$f(x,y) = 100(y - x^2)^2 + (1-x)^2. \tag{1}$$

   Set the initial step length $\alpha_0 = 1$ and print the step length used by each method at each iteration. First try the initial point $\boldsymbol{x}_0 = (1.2, 1.2)^T$ and then try the more difficult starting point $\boldsymbol{x}_0 = (-1.2, 1)^T$.

In this second question, you are asked to implement the ordinary SD method and to investigate the circumstances in which it can fail. The theoretical lecture notes can be used to try to understand this failure.

2. Consider the optimization problem,

$$\min f(\boldsymbol{x}), \qquad f(\boldsymbol{x}) = \langle \boldsymbol{a}, \boldsymbol{x}\rangle + \tfrac{1}{2}\langle \boldsymbol{x}, B\boldsymbol{x}\rangle,$$

   where now B is a specific 10×10 matrix and $\boldsymbol{a}$ is a specific 10×1 column vector. The numerical values of these arrays can be found in the spreadsheet `OP_10x10.csv`:

   - The spreadsheet contains a 10×1 array which corresponds to the vector $\boldsymbol{a}$;
   - The spreadsheet contains a 10×10 array B_0.

The array B is obtained from B_0 by the following sequence of steps:

(i) Symmetrize B_0:
$$B_0 \to (B_0 + B_0^T)/2;$$

(ii) Scale B_0:
$$B_0 \to B_0/\max(|B_0|)$$

(iii) Generate a positive-definite matrix:
$$B_0 \to (B_0^T)B_0.$$

The end result of this sequence of operations is the matrix B.

Hence,

(a) Find the minimizer $\boldsymbol{x}_*$ numerically, using the steepest-descent and Newton algorithms.

(b) Why is the convergence so poor in the case of the steepest-descent algorithm?