Deterministic optimization assignment

Pol Pastells

(Dated: January 10, 2021)

The *Rosenbrock's function* is:

$$f(x, y) = 100(y - x^2)^2 + (1 - x)^2$$

The exercise consisted in solving the problem of minimizing f over $\mathbb{R}^2$ starting at the point (-1.5,-1). The global minimum is found at (1,1), with $f(1, 1) = 0$.

I. CONVEXITY

f is *not* a convex function, given that its Hessian matrix is not positive semidefinite.

The Hessian matrix is:

$$H = \begin{pmatrix} 1200x^2 - 400y + 2 & -400x \\ -400x & 200 \end{pmatrix}$$

and $x^T Hx$ for and arbitrary vector $x = (a, b)$ equals $(1200x^2 - 400y + 2)a^2 - 800xab + 200b^2$. With the second term unbounded from below, and the whole expression not a sum of squares.

We can also see this with an example:

$$f\left(\frac{(-1, 1) + (0, 0)}{2}\right) = 8.5 > 2.5 = \frac{f(-1, 1) + f(0, 0)}{2}$$

II. CONJUGATE GRADIENT METHOD

We've coded the conjugate gradient method, updating the position according to:

$$\vec{x}^{(k+1)} = \vec{x}^{(k)} + \alpha_k \vec{p}^{(k)}$$

$$\vec{p}^{(k+1)} = \vec{\nabla} f^{(k+1)} - \beta_k \vec{p}^{(k)}$$

β_k is computed using the Polak-Ribière formula [1]:

$$\beta_k = \frac{\vec{\nabla} f^{(k)^T} (\vec{\nabla} f^{(k)} - \vec{\nabla} f^{(k-1)})}{\vec{\nabla} f^{(k-1)^T} \vec{\nabla} f^{(k-1)}}$$

Where $\vec{\nabla} f^{(k)} \equiv \vec{\nabla} f(\vec{x}^{(k)})$. And α_k is solved for with a line search using the backtrackr algorithm found in [2].

It takes 26 iterations for the method to converge to the minimum within machine precision.

III. LEVENBERG-MARQUARDT ALGORITHM

The Levenberg-Marquardt algorithm exploits the fact that our function is a sum of squares:

$$F(x) = \frac{1}{2} \sum_{i=1}^{m} (f_i(\vec{x}))^2 = \frac{1}{2} ||\vec{f}(\vec{x})||^2$$

And uses the $\vec{f}(\vec{x})$ and its Jacobian to solve the problem. It is a damped *Gauss-Newton* method. It is supposed to be more robust, but slower than other methods. In our implementation it takes 20 iterations for it to converge to the minimum within machine precision, therefore it is faster than the previous method in this case.

Our implementation follows the pseudocode found in [3].

IV. COMPARISON

Here we show a table comparing the number of iterations necessary for convergence depending on the initial position.

x_0	y_0	CG iterations	LM iterations
-1.5	-1	26	20
200	3	56	23
-1833	-13412	71	7590
-200	200	152	136

Table I: Number of iterations necessary to reach the global minimum for the conjugate gradient method (CG) and the Levenberg-Marquardt algorithm (LM).

We see that the number of iterations, and which method is faster wildly depends on the initial conditions, but at least for the ones we've tried they always arrive to the global minimum.

[1] Elijah Polak and Gerard Ribiere. Note sur la convergence de méthodes de directions conjuguées. *ESAIM: Mathematical Modelling and Numerical Analysis-Modélisation Mathématique et Analyse Numérique*, 3(R1):35–43, 1969. II

[2] Alfio Quarteroni, Riccardo Sacco, and Fausto Saleri. *Numerical mathematics*, volume 37. Springer Science & Business Media, 2010. II

[3] H. B. Nielsen K. Madsen and O. Tingleff. *Methods for Non-Linear Least Squares Problems.* Technical University of Denmark, 2004. III