Report NO4LSP

Chiodo Martina - 343310
Vigè Sophie - 339268

INTRODUZIONE

DIRE QUALCOSA SU COME SI CALCOLA ROC

The experimental rate of convergence can be approximated by

$$q \approx \frac{\log\left(\frac{\|\hat{e}^{(k+1)}\|}{\|\hat{e}^{(k)}\|}\right)}{\log\left(\frac{\|\hat{e}^{(k)}\|}{\|\hat{e}^{(k-1)}\|}\right)} \quad \text{for } k \text{ large enough} \tag{1}$$

where $\hat{e}^{(k)} = x^{(k)} - x^{(k-1)}$ approximates the error at the k-th iteration.

Specificare gli STOPPING CRITERION per gli algoritmi

PROBLEMA 64

PROBLEMA 75

PROBLEMA 76

Model

The function described in this problem is the following

$$F(\mathbf{x}) = \frac{1}{2}\sum_{k=1}^{n} f_k^2(x)$$
$$f_k(\mathbf{x}) = x_k - \frac{x_{k+1}^2}{10}, \quad 1 \le k < n$$
$$f_n(\mathbf{x}) = x_n - \frac{x_1^2}{10}$$

where n denotes the dimensionality of the input vector $\mathbf{x}$.
The starting point for the minimization is the vector $\mathbf{x}_0 = [2, 2, \dots, 2]$.

To be able to say something more about the behaviour of the problem is useful to look at the gradient of the function $F(\mathbf{x})$ and at its Hessian matrix.

$$\nabla F(\mathbf{x}) = \begin{bmatrix} \frac{\partial F}{\partial x_1}(\mathbf{x}) \\ \vdots \\ \frac{\partial F}{\partial x_k}(\mathbf{x}) \\ \vdots \\ \frac{\partial F}{\partial x_n}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} \frac{\partial}{\partial x_1}\frac{1}{2}\left[f_n^2 + f_1^2\right](\mathbf{x}) \\ \vdots \\ \frac{\partial}{\partial x_k}\frac{1}{2}\left[f_{k-1}^2 + f_k^2\right](\mathbf{x}) \\ \vdots \\ \frac{\partial}{\partial x_n}\frac{1}{2}\left[f_{n-1}^2 + f_n^2\right](\mathbf{x}) \end{bmatrix} = \begin{bmatrix} -\frac{x_1}{5}\left(x_n - \frac{x_1^2}{10}\right) + \left(x_1 - \frac{x_1^2}{10}\right) \\ \vdots \\ -\frac{x_k}{5}\left(x_{k-1} - \frac{x_k^2}{10}\right) + \left(x_k - \frac{x_{k+1}^2}{10}\right) \\ \vdots \\ -\frac{x_n}{5}\left(x_{n-1} - \frac{x_n^2}{10}\right) + \left(x_n - \frac{x_1^2}{10}\right) \end{bmatrix}$$

Due to the particular structure of the function, the Hessian matrix as a sparse structure, with only 3 diagonals different from zero. The non-zero elements are the following:

$$\frac{\partial^2 F}{\partial x_k^2}(\mathbf{x}) = -\frac{1}{5}x_{k-1} - \frac{3}{50}x_k^2 + 1, \quad 1 < k \le n \qquad \frac{\partial^2 F}{\partial x_1^2}(\mathbf{x}) = -\frac{1}{5}x_n - \frac{3}{50}x_1^2 + 1,$$
$$\frac{\partial^2 F}{\partial x_k \partial x_{k+1}}(\mathbf{x}) = -\frac{1}{5}x_{k+1}, \quad 1 \le k < n \qquad \frac{\partial^2 F}{\partial x_n \partial x_1}(\mathbf{x}) = -\frac{1}{5}x_1$$
$$\frac{\partial^2 F}{\partial x_k \partial x_{k-1}}(\mathbf{x}) = -\frac{1}{5}x_k, \quad 1 < k \le n \qquad \frac{\partial^2 F}{\partial x_1 \partial x_n}(\mathbf{x}) = -\frac{1}{5}x_n$$

We can now easily notice that the gradient of the function is null when all the components of the vector $\mathbf{x}$ are equal to 0, in this case the Hessian matrix is positive definite, so the point $\mathbf{x} = \mathbf{0}$ is a minimum of the function $F(\mathbf{x})$. Because of the definition of the function, 0 is the lowest value the function can assume, so the minimum found is global.

Nealder Mead Method

We now report a table containing some general results obtained by running the Nealder Mead method on the function $F(\mathbf{x})$.

	avg fbest	avg num of iters	avg time of exec (sec)	n failure	avg roc
10	1.0000e-04	2.1864e+02	4.5116e+00	0.0000e+00	NaN
25	0.0000e+00	1.6804e+03	3.1524e+01	0.0000e+00	NaN
50	2.9039e+01	1.4007e+04	2.6917e+02	1.0000e+01	NaN

Figura 1: Resultats obtained by running the symplex method on the function $F(\mathbf{x})$.

First thing we can notice is that for smaller dimensionalities the symplex method is able to find the minimum in a reasonable amount of time, but when the dimensionality becomes higher the method starts failing. From the plot in figure (2), we can see that for most points belonging to $\mathbb{R}^{50}$, the method keeps iterating until the maximum number of iterations is reached without satisfying the stopping criterion. This behaviour can probably be explained by the fact that when the dimensionality increases the starting point is more far from the minimum due to its definition, so the method needs to perform more iterations to reach the minimum.

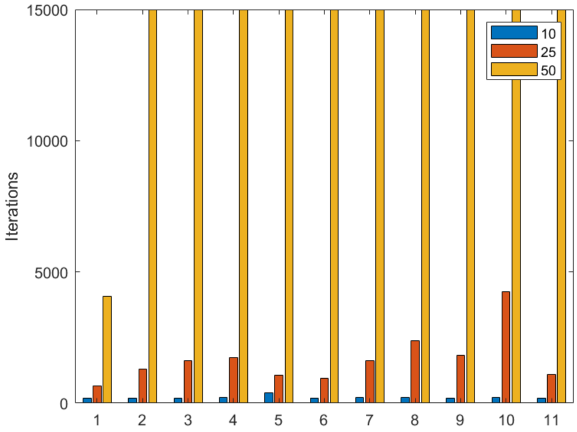
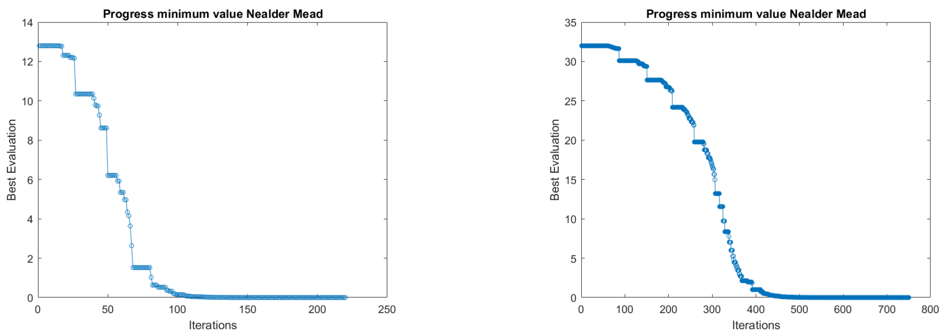


Figura 2: Number of iterations needed by the Nealder Mead method to find the minimum of the function $F(\mathbf{x})$ for each starting point.

From the previous table, we can notice that the experimental rate of convergnce is always `Nan`: this is due to the fact that in the last iterations the value of $\mathbf{x}^{(k)}$ does not change much and thus it yields a division by zero in the formula (1). This can be seen in the following plots, showing that, in the last iterations, the approximated value of the minimum is almost stationary.



(a) dimension 10

(b) dimension 25

Figura 3: Plots of the progresses of the Nealder Mead method for different dimensionalities.

Modified Newton Method - Exact Derivatives

We now report a table containing some general results obtained by running the Modified Newton method on the function $F(\mathbf{x})$. We obviously expect the method to perform better than the symplex method because of the exact derivatives used in the computation of the descent direction.

	avg fbest	avg gradf_norm	avg num of iters	avg time of exec (sec)	n failure	avg roc
1000	2.9818e-10	1.1915e-05	5.4545e+00	2.8048e-02	0.0000e+00	1.7721e+00
10000	2.9521e-16	2.3690e-08	4.9091e+00	2.5717e-02	0.0000e+00	1.9344e+00
100000	3.2292e-15	7.6604e-08	5.0000e+00	2.4656e-01	0.0000e+00	1.9326e+00

Figura 4: Resultats obtained by running the Modified Newton Method on the function $F(\mathbf{x})$ using the exact derivatives.

This time, the method always converges to the minimum point in very few iterations, even for higher dimensionalities. We can also appreciate the fact that the approximated rate of convergence is close to 2, as expected for a Newton method.

Modified Newton Method - Approximated Derivatives

Approximating the derivatives of the function $F(\mathbf{x})$ using finite differences is more challenging than it appears due to potential numerical cancellation issues, which can occur when subtracting two nearly equal quantities. Additionally, we aim to derive a formula that minimizes computational cost.

Let's begin by approximating the first-order derivatives of the function $F(\mathbf{x})$ using the centered finite difference formula with step h_k. The subscript k is specified because the following formulas are valid both with a constant increment, $h_k = k$ for all $h = 1, \ldots, n$, and with a specific increment $h_k = h|\hat{x}_k|$ $k = 1, \ldots, n$, where $\hat{\mathbf{x}}$ is the point at which we approximate the derivatives.

$$\frac{\partial F}{\partial x_k}(\mathbf{x}) \approx \frac{F(\mathbf{x}+h_k\vec{e}_k) - F(\mathbf{x}-h_k\vec{e}_k)}{2h_k} = \frac{\sum_{i=1}^n f_i(\mathbf{x}+h_k\vec{e}_k)^2 - \sum_{i=1}^n f_i(\mathbf{x}-h_k\vec{e}_k)^2}{4h_k}$$

We can observe that each term f_i^2 only depends on x_i and x_{i+1}, so $f_i(\mathbf{x}+h_k\vec{e}_k)^2 - f_i(\mathbf{x}-h_k\vec{e}_k)^2 = 0$ for all $i \neq k-1, k$ (or $i \neq 1, n$ if we are considering $k = 1$). This allows to simplify the formula, even in order to decrease the computational cost, as follows

$$\frac{\partial F}{\partial x_k}(\mathbf{x}) \approx \frac{f_{k-1}(\mathbf{x}+h_k\vec{e}_k)^2 - f_{k-1}(\mathbf{x}-h_k\vec{e}_k)^2 + f_k(\mathbf{x}+h_k\vec{e}_k)^2 - f_k(\mathbf{x}-h_k\vec{e}_k)^2}{4h_k} \qquad 1 < k \leq n$$

$$\frac{\partial F}{\partial x_k}(\mathbf{x}) \approx \frac{f_n(\mathbf{x}+h_k\vec{e}_k)^2 - f_n(\mathbf{x}-h_k\vec{e}_k)^2 + f_k(\mathbf{x}+h_k\vec{e}_k)^2 - f_k(\mathbf{x}-h_k\vec{e}_k)^2}{4h_k} \qquad k = 1$$

In order to avoid numerical cancellation, the numerator has been expanded obtaining the following formula

$$\frac{\partial F}{\partial x_1}(\mathbf{x}) \approx \frac{4h_k x_1 - 2/5 h_k x_2^2 - 4/5 h_k x_n x_1 + 8/100 h_k x_1 (x_1^2 + h_k^2)}{4h_k}$$
$$\frac{\partial F}{\partial x_k}(\mathbf{x}) \approx \frac{4h_k x_k - 2/5 h_k x_{k+1}^2 - 4/5 h_k x_{k-1} x_k + 8/100 h_k x_k (x_k^2 + h_k^2)}{4h_k}$$
$$\frac{\partial F}{\partial x_n}(\mathbf{x}) \approx \frac{4h_k x_n - 2/5 h_k x_1^2 - 4/5 h_k x_{n-1} x_n + 8/100 h_k x_n (x_n^2 + h_k^2)}{4h_k}$$

We can now proceed to approximate the second order derivatives of the function $F(\mathbf{x})$ using the centered finite difference formula; this time we need to use two different increments h_i and h_j based on the two components with respect to which we are differentiating. The general formula is the following

$$\frac{\partial^2 F}{\partial x_i \partial x_j}(\mathbf{x}) = \frac{F(\mathbf{x}+h_i\vec{e}_i+h_j\vec{e}_j) - F(\mathbf{x}+h_i\vec{e}_i) - F(\mathbf{x}-h_j\vec{e}_j) + F(\mathbf{x})}{h_i h_j}$$

CONCLUSIONI