Project 4

Non-linear systems of equations / Newton-Raphson method

Group 2 - Team 12425

Manager : alamhamdi001

Secretary : felosrouti

Developers : jfsornay, ljouhault, aliard

Summary: The goal of this project was to program algorithms dedicated to research the nonlinear equations systems roots. The method promoted here was the Newton-Raphson algorithm and the goal was to evaluate the assets and liabilities of such a solution. This shall be done by testing the method in different settings. In this report, three main parts are raised: The first one consists of implementing Newton-Raphson's algorithm. The second one is about Bairstow's method for polynomial factorization and the last one deals with electrostatic equilibrium problems. *(Aymane Lamhamdi)*

1 Newton-Raphson method (Louis Jouhault)

The goal of this project was to program algorithms dedicated to the research of roots of non-linear equations systems using the Newton-Raphson method. The first step was to create a function able to apply the Newton-Raphson method in any dimension. The algorithm takes a function $f : \mathbb{R}^n \to \mathbb{R}^m$ and its Jacobian matrix J (matrix of all partial derivatives of f) and a starting position U as a vector. Then a new vector V is calculated using the equation 1 and added to the starting vector. By doing this multiple times, using the result as the new starting point, this algorithm converges to the targeted roots.

$$f(U) + H(U).V = 0 \tag{1}$$

However, this method has a small issue. If the function is almost horizontal, its derivatives will be close to 0 and V would have a really high absolute value and be farther from the solution. To avoid that, a solution called "backtracking" was used. Instead of using $U + V$ as the starting point for the next iteration, the program used $U + \alpha.V$ where α is an arbitrary coefficient. Most of the time, $\alpha = 1$, but if $f(U + \alpha.V)$ is further from 0 than $f(U)$, α is divided by 2 again and again until $f(U + \alpha.V)$ gets close enough to 0.

This is the theoretical method but in order to be able to use it in a real program, additional parameters had to be added. First, a parameter called *epsilon* to indicate the desired accuracy ;

since this is a real algorithm, it's really unlikely that it would reach the exact solution. Then, a parameter called N indicating the maximum number of iterations the program should do ; if the epsilon precision can't be reached or really late, this ensure the program will not run for too long. Finally, a last parameter called a_min representing the minimum value that α can take ; in some rare cases α will never be small enough for the solution to stay close to 0, so we have to limit the number of divisions, otherwise the program could run for a really long time.

2 Bairstow method for polynomial factorization (Fatima Ezahrae El Osrouti, Jean-François Sornay)

Introduction - *Jean-François Sornay*

The Newton-Raphson method was used in section 1 in order to solve non-linear equation system. This method allows to approximate equation roots, and can be applied to one-dimensional functions as well as multi-dimensional ones. However, Newton-Raphson's method is unable to find the complex roots of a non-linear equation system. The Bairstow method studied in this section solve this problem.

2.1 Principle of Bairstow's method - *Jean-François Sornay*

The Bairstow method studied allows to find both the real and complex roots of a polynomial. It is based on the idea of synthetic division of the given polynomial by a quadratic function incidentally avoiding complex arithmetic.

The reasoning behind this method is as follow :

Given a polynomial as in equation 2, the Bairstow method consists of dividing this polynomial by a quadratic function as in equation 3. Then, the equation 4 can be considered as $f_n(x)$.

This process is then iterated until the polynomial becomes quadratic or linear, and all the roots have been determined. Thus, if $x^2 + Bx + C$ is an exact factor of $f_n(x)$ then the remainder $R(x)$ is zero and the real (or complex) roots of this quadratic equation are also roots of $f_n(x)$.

$$f_n(x) = \sum_{i=0}^{n} a_i x^i = a_0 + a_1 x + a_2 x^2 + \cdots + a_n x^n \tag{2}$$

$$x^2 + Bx + C \tag{3}$$

$$f_n(x) = (x^2 + Bx + C)Q(x) + R(x) \tag{4}$$

with $Q(x)$ the quotient of $f_n(x)$ by $(x^2 + Bx + C)$

and $R(x)$ the linear function representing the remainder of this division.

2.2 Implementation of Bairstow's method

This part aims to describe the different steps we took in order to implement the Bairstow method.

2.2.1 First step : nullifying $R(x)$ - *Jean-François Sornay*

In order to nullify $R(x)$, the quadratic function represented by the equation 3 has to be infinitely close to an exact factor of $f_n(x)$. Therefore, it is equivalent to determine a function whose zero is computed in dimension 2 associating (B, C) to ($r(B,C)$, $s(B,C)$), where r and s are the coefficients of $R(x)$ such as $R(x) = rx + s$ while B and C are the coefficients of the quadratic equation. So, this function calculate the division's remainder of $f_n(x)$ by the quadratic function and it is defined from $\mathbb{R}^2$ to $\mathbb{R}^2$.

In the Bairstow method, Newton-Raphson's approach is used to adjust the coefficients B and C until the rest is close enough to 0. However, it requires a function calculating the Jacobian matrix of B and C.

2.2.2 Second step : Finding the Partial derivatives of $R(x)$ - *Fatima Ezahrae El Osrouti*

After writing the function that allows us to get a null remainder of a polynomial $P(x)$, the Jacobian matrix associated to this function is the next thing we need in order to apply the Raphson-Newton method, and thus partial derivatives of the function f described in the previous subsection.
The idea consists of manipulating the given formula of the polynomial $P(x) = (x + Bx + C)Q(x) + Rx + S$ by deriving it with respect to C on one hand, and with respect to B on the other hand. Since P(x) is independent of B and C, hence we obtain the two equations bellow:

$0 = (\mathrm{x}^2 + \mathrm{Bx} + \mathrm{C})\frac{\partial Q}{\partial C} + \mathrm{Q(x)} + \frac{\partial R}{\partial C}\mathrm{x} + \frac{\partial S}{\partial C}$

$0 = (\mathrm{x}^2 + \mathrm{Bx} + \mathrm{C})\frac{\partial Q}{\partial B} + \mathrm{Q(x)} + \frac{\partial R}{\partial B}\mathrm{x} + \frac{\partial S}{\partial B}$

Thus if if we perform a second synthetic division of $P(x)$, also known as the Euclidean division of $Q(x)$ by the same quadratic polynomial x^2+Bx+C, resulting a remainder R_1x+S_1, we get :

$\frac{\partial R}{\partial C}$ = -R_1 and $\frac{\partial S}{\partial C}$ = -S_1

To get the other two partial derivatives we use the roots of the quadratic polynomial that are already known, finally we get :

$\frac{\partial R}{\partial B}$ = BR_1 - S_1 and $\frac{\partial S}{\partial B}$ = CR_1.

Now that we succeeded at calculating all the partial derivatives of the function mentioned in the previous subsection, our jacobian matrix can be easily constructed and we can move to the next step!

2.2.3 Third step : Application of Bairstow's method - *Fatima Ezahrae El Osrouti*

The third step is the final step to find the roots of the polynomial $P(x)$.

All in all, to apply the Raphson-Newton method we need a function f, the jaconbian matrix associated and other constants that we "randomly" fix (such as the departing vector U0, the precision epsilon(which should be infinitesimal, here it is fixed at 0.001) and the integer N_{max}(which is fixed at 500) that represent the number of rounds we will go through in the Newton-Raphson algorithm).

The function f takes in parameters B and C, and returns the remainder function implemented in the first subsection .

The function J takes in parameters a 2 dimensional vector and computes the Jacobian matrix of the function f .

The idea here is to find the couple (B,C) that can nullify the function f using the Raphson-Newton method.Once this couple is found, it is easy to extract 2 roots of the polynomial $P(x)$ since in this case we have : $P(x) = Q(x) * (x + Bx + C)$. Thus these two roots are the roots of the quadratic $x + Bx + C$ which can be easily extracted using the delta method.

We repeat the same steps described above for the polynomial $Q(x)$ if its degree is strictly superior to 2, until we find all the roots of the polynomial P(x).

3 Electrostatic equilibrium (Arthur Liard, Aymane Lamhamdi)

3.1 Preliminary analysis of the objectives - *Arthur Liard*

Once the Newton-Raphson method is implemented, it is possible to be applied to practical examples. This part is focused on finding the equilibrium positions of N electrostatic charges contained in the interval $[-1, 1]$ at $(x_1, x_2, ..., x_N)$ with two fixed charges at positions -1 and 1. The particles are

linked through the total electrostatic energy of the system defined with:

$$E(x_1, x_2, ..., x_N) = \sum_{i=1}^{N} \log|x_i + 1| + \log|x_i - 1| + \frac{1}{2} \sum_{j=1, j\neq i} \log|x_i - x_j|$$

In order to determine the equilibrium positions, the solution of the following equation is required:

$$\nabla E(x_1, x_2, ..., x_N) = [\frac{\partial E(x_1, x_2, ..., x_N)}{\partial x_i}] = 0 \tag{5}$$

It corresponds to the kind of equation the Newton-Raphson method is made for. Therefore the first objective is to calculate first ∇E and then the Jacobian matrix J of ∇E.

∇E calculus Let's first rewrite E with only terms depending on a given x_i:

$$E(x_1, x_2, ..., x_N) = \log|x_i + 1| + \log|x_i - 1| + \frac{1}{2} \sum_{j=1, j\neq i} \log|x_i - x_j| + \sum_{j=1, j\neq i}^{N} \frac{1}{2} \log|x_j - x_i|$$

$$E(x_1, x_2, ..., x_N) = \log|x_i + 1| + \log|x_i - 1| + \sum_{j=1, j\neq i}^{N} \log|x_i - x_j|$$

The partial derivative with respect to x_i is then

$$\nabla E_i = \frac{\partial E(x_1, x_2, ..., x_N)}{\partial x_i} = \frac{1}{x_i + 1} + \frac{1}{x_i - 1} + \sum_{j=1, j\neq i}^{N} \frac{1}{x_i - x_j} \tag{6}$$

The absolute value in logarithm is erased off the equation when differentiated.

Jacobian matrix calculus It is now possible to calculate the Jacobian matrix.

$$J = [\frac{\partial \nabla E}{\partial x_1} \cdots \frac{\partial \nabla E}{\partial x_N}]$$

$$J_{i,j} = \begin{cases} \frac{-1}{(x_i+1)^2} - \frac{1}{(x_i-1)^2} - \sum_{j=1, j\neq i}^{N} \frac{1}{(x_i-x_j)^2} & \text{si} \quad i = j \\ \frac{1}{(x_i-x_j)^2} & \text{si} \quad i \neq j \end{cases} \tag{7}$$

Thus we have every information required to apply the Newton-Raphson method in order to find our equilibrium positions.

3.2 Searching for the equilibrium positions and comparing them to the Legendre polynomial derivative roots - *Aymane Lamhamdi*

The object of this part is to solve the equation (5) using Newton's method and to compare the solutions to the Legendre polynomials derivative roots. Firstly, the equilibrium positions have been found using the Newton-Raphson method implemented in the first section. Those positions have been tested by verifying that:

$$\nabla E(x_1, x_2, ..., x_N) = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}, \text{ where } x_i \text{ is the equilibrium position of the charge } i.$$

Next, the Legendre polynomials derivative roots are calculated according to the number of charges in the system besides those positioned at the extremities -1 and 1. For n charges, the comparison is made between the n equilibrium positions of the charges and the n roots of the derivative of the Legendre polynomials of order $n+1$. Both roots were displayed on the same graph and the resultant figures 1 and 2 exhibit that they are superimposed.

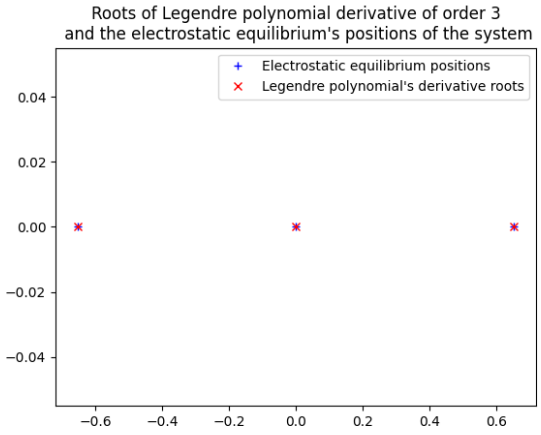

Figure 1: Roots of Legendre polynomial derivative of order 3 and the electrostatic equilibrium positions of the system

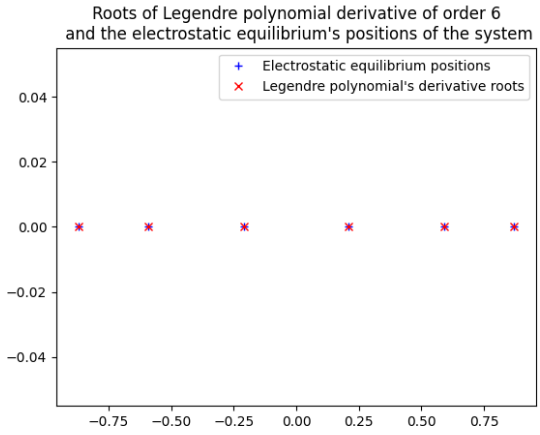

Figure 2: Roots of Legendre polynomial derivative of order 6 and the electrostatic equilibrium positions of the system

3.3 Maximum or minimum ? - *Aymane Lamhamdi*

Electrostatic equilibrium positions calculated in the previous section correspond to either to the maximum or the minimum of the system energy. To determine that and due to the fact that the system has only one equilibrium position, the total electrostatic energy of the equilibrium position was compared to any other position's energy. That revealed that the equilibrium positions calculated correspond to the maximum of the energy.

In a system composed of one charge besides the two charges positioned in the extremities, the figure 3 displays its electrostatic energy depending on the position of the charge and it reveals that in this case the equilibrium position is 0 which corresponds to the maximum of the energy.

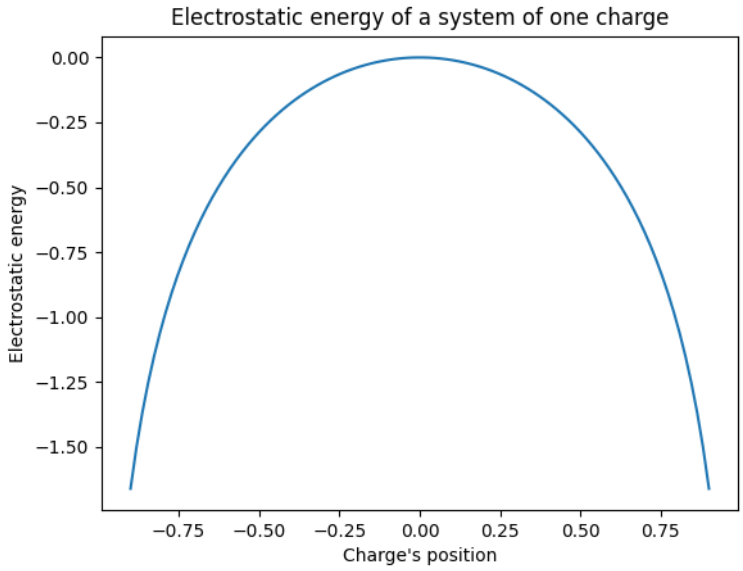

Figure 3: 1charge-system electrostatic energy

4 Conclusion - *Arthur Liard*

Through this project, research of non-linear equation system solving algorithms and practical applications have been explored. The Newton-Raphson method is probably the most common one to find their roots, because it's intuitive and efficient enough. Bairstow's method brings a wider range

of solutions adding the complex ones. They allow to study physics experiments, as shown with the possibles subjects to answer (Electrostatic Equilibirium, Computation of the Lagrangian points, or The wave equation). Therefore they are truly interesting methods to know, although calculating the Jacobian matrix of a function might feel exhausting.