NONLINEAR SYSTEMS OF EQUATIONS: NEWTON-RAPHSON METHOD

The Newton-Raphson method is the method of choice for solving nonlinear systems of equations. Many engineering software packages (especially finite element analysis software) that solve nonlinear systems of equations use the Newton-Raphson method. The derivation of the method for nonlinear systems is very similar to the one-dimensional version in the root finding section which is explain before. Assume a nonlinear system of equations of the form:

$$\begin{aligned} f_1(x_1, x_2, \cdots, x_n) &= 0 \\ f_2(x_1, x_2, \cdots, x_n) &= 0 \\ &\vdots \\ f_n(x_1, x_2, \cdots, x_n) &= 0 \end{aligned}$$

If the components of one iteration $x^{(i)} \in \mathbb{R}^n$ are known as: $x_1^{(i)}, x_2^{(i)}, \cdots, x_n^{(i)}$, then, the Taylor expansion of the first equation around these components is given by:

$$\begin{aligned} f_1\left(x_1^{(i+1)}, x_2^{(i+1)}, \cdots, x_n^{(i+1)}\right) \approx & f_1\left(x_1^{(i)}, x_2^{(i)}, \cdots, x_n^{(i)}\right) + \\ & \left.\frac{\partial f_1}{\partial x_1}\right|_{x^{(i)}} \left(x_1^{(i+1)} - x_1^{(i)}\right) + \left.\frac{\partial f_1}{\partial x_2}\right|_{x^{(i)}} \left(x_2^{(i+1)} - x_2^{(i)}\right) + \cdots + \\ & \left.\frac{\partial f_1}{\partial x_n}\right|_{x^{(i)}} \left(x_n^{(i+1)} - x_n^{(i)}\right) \end{aligned}$$

Applying the Taylor expansion in the same manner for $f_2, \cdots, f_n$, we obtained the following system of linear equations with the unknowns being the components of the vector $x^{(i+1)}$:

$$\begin{pmatrix} f_1\left(x^{(i+1)}\right) \\ f_2\left(x^{(i+1)}\right) \\ \vdots \\ f_n\left(x^{(i+1)}\right) \end{pmatrix} = \begin{pmatrix} f_1\left(x^{(i)}\right) \\ f_2\left(x^{(i)}\right) \\ \vdots \\ f_n\left(x^{(i)}\right) \end{pmatrix} + \begin{pmatrix} \left.\frac{\partial f_1}{\partial x_1}\right|_{x^{(i)}} & \left.\frac{\partial f_1}{\partial x_2}\right|_{x^{(i)}} & \cdots & \left.\frac{\partial f_1}{\partial x_n}\right|_{x^{(i)}} \\ \left.\frac{\partial f_2}{\partial x_1}\right|_{x^{(i)}} & \left.\frac{\partial f_2}{\partial x_2}\right|_{x^{(i)}} & \cdots & \left.\frac{\partial f_2}{\partial x_n}\right|_{x^{(i)}} \\ \vdots & \vdots & \vdots & \vdots \\ \left.\frac{\partial f_n}{\partial x_1}\right|_{x^{(i)}} & \left.\frac{\partial f_n}{\partial x_2}\right|_{x^{(i)}} & \cdots & \left.\frac{\partial f_n}{\partial x_n}\right|_{x^{(i)}} \end{pmatrix} \begin{pmatrix} \left(x_1^{(i+1)} - x_1^{(i)}\right) \\ \left(x_2^{(i+1)} - x_2^{(i)}\right) \\ \vdots \\ \left(x_n^{(i+1)} - x_n^{(i)}\right) \end{pmatrix}$$

By setting the left hand side to zero (which is the desired value for the functions $f_1, f_2, \cdots, f_n$, then, the system can be written as:

$$\begin{pmatrix} \frac{\partial f_1}{\partial x_1}\Big|_{x^{(i)}} & \frac{\partial f_1}{\partial x_2}\Big|_{x^{(i)}} & \cdots & \frac{\partial f_1}{\partial x_n}\Big|_{x^{(i)}} \\ \frac{\partial f_2}{\partial x_1}\Big|_{x^{(i)}} & \frac{\partial f_2}{\partial x_2}\Big|_{x^{(i)}} & \cdots & \frac{\partial f_2}{\partial x_n}\Big|_{x^{(i)}} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_n}{\partial x_1}\Big|_{x^{(i)}} & \frac{\partial f_n}{\partial x_2}\Big|_{x^{(i)}} & \cdots & \frac{\partial f_n}{\partial x_n}\Big|_{x^{(i)}} \end{pmatrix} \begin{pmatrix} \left(x_1^{(i+1)} - x_1^{(i)}\right) \\ \left(x_2^{(i+1)} - x_2^{(i)}\right) \\ \vdots \\ \left(x_n^{(i+1)} - x_n^{(i)}\right) \end{pmatrix} = - \begin{pmatrix} f_1\left(x^{(i)}\right) \\ f_2\left(x^{(i)}\right) \\ \vdots \\ f_n\left(x^{(i)}\right) \end{pmatrix}$$

Setting $K_{ab} = \frac{\partial f_a}{\partial x_b}\Big|_{x^{(i)}}$, the above equation can be written in matrix form as follows:

$$K\Delta x = -f$$

where K is an $n \times n$ matrix, $-f$ is a vector of n components and Δx is an n-dimensional vector with the components $\left(x_1^{(i+1)} - x_1^{(i)}\right), \left(x_2^{(i+1)} - x_2^{(i)}\right), \cdots, \left(x_n^{(i+1)} - x_n^{(i)}\right)$. If K is invertible, then, the above system can be solved as follows:

$$\Delta x = -K^{-1} f \Rightarrow x^{(i+1)} = x^{(i)} + \Delta x$$

it's useful to know the general formula for the inverse of a 2×2 matrix:

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix}^{-1} = \frac{1}{ad - bc} \begin{bmatrix} d & -b \\ -c & a \end{bmatrix}$$

provided $ad - bc \neq 0$ (if $ad - bc = 0$, the matrix is singular)

there are similar, but much more complicated, formulas for the inverse of larger square matrices, but the formulas are rarely used

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} * \begin{bmatrix} e & f \\ g & h \end{bmatrix} = \begin{bmatrix} ae + bg & af + bh \\ ce + dg & cf + dh \end{bmatrix}$$

$$\begin{bmatrix} a & b \\ c & d \end{bmatrix} * \begin{bmatrix} k \\ m \end{bmatrix} = \begin{bmatrix} ak + bm \\ ck + dm \end{bmatrix}$$

EXAMPLE

Use the Newton-Raphson method with $\varepsilon_s = 0.0001$ to find the solution to the following nonlinear system of equations:

$$x_1^2 + x_1 x_2 = 10 \qquad x_2 + 3x_1 x_2^2 = 57$$

SOLUTION

In addition to requiring an initial guess, the Newton-Raphson method requires evaluating the derivatives of the functions $f_1 = x_1^2 + x_1 x_2 - 10$ and $f_2 = x_2 + 3x_1 x_2^2 - 57$. If $K_{ab} = \frac{\partial f_a}{\partial x_b}$, then it has the following form:

$$K = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{pmatrix} = \begin{pmatrix} 2x_1 + x_2 & x_1 \\ 3x_2^2 & 1 + 6x_1 x_2 \end{pmatrix}$$

Assuming an initial guess of $x_1^{(0)} = 1.5$ and $x_2^{(0)} = 3.5$, then the vector f and the matrix K have components:

$$f = \begin{pmatrix} x_1^2 + x_1 x_2 - 10 \\ x_2 + 3x_1 x_2^2 - 57 \end{pmatrix} = \begin{pmatrix} -2.5 \\ 1.625 \end{pmatrix} \qquad K = \begin{pmatrix} 6.5 & 1.5 \\ 36.75 & 32.5 \end{pmatrix}$$

The components of the vector Δx can be computed as follows:

$$\Delta x = -K^{-1} f = \begin{pmatrix} 0.53603 \\ -0.65612 \end{pmatrix}$$

Therefore, the new estimates for $x_1^{(1)}$ and $x_2^{(1)}$ are:

$$x^{(1)} = x^{(0)} + \Delta x = \begin{pmatrix} 2.036029 \\ 2.843875 \end{pmatrix}$$

The approximate relative error is given by:

$$\varepsilon_r = \frac{\sqrt{(0.53603)^2 + (-0.65612)^2}}{\sqrt{(2.036029)^2 + (2.843875)^2}} = 0.2422$$

For the second iteration the vector f and the matrix K have components:

$$f = \begin{pmatrix} -0.06437 \\ -4.75621 \end{pmatrix} \qquad K = \begin{pmatrix} 6.9159 & 2.0360 \\ 24.2629 & 35.7413 \end{pmatrix}$$

The components of the vector Δx can be computed as follows:

$$\Delta x = -K^{-1} f = \begin{pmatrix} -0.03733 \\ 0.15841 \end{pmatrix}$$

Therefore, the new estimates for $x_1^{(2)}$ and $x_2^{(2)}$ are:

$$x^{(2)} = x^{(1)} + \Delta x = \begin{pmatrix} 1.998701 \\ 3.002289 \end{pmatrix}$$

The approximate relative error is given by:

$$\varepsilon_r = \frac{\sqrt{(-0.03733)^2 + (0.15841)^2}}{\sqrt{(1.998701)^2 + (3.002289)^2}} = 0.04512$$

For the third iteration the vector f and the matrix K have components:

$$f = \begin{pmatrix} -0.00452 \\ 0.04957 \end{pmatrix} \qquad K = \begin{pmatrix} 6.9997 & 1.9987 \\ 27.0412 & 37.0041 \end{pmatrix}$$

The components of the vector Δx can be computed as follows:

$$\Delta x = -K^{-1} f = \begin{pmatrix} 0.00130 \\ -0.00229 \end{pmatrix}$$

Therefore, the new estimates for $x_1^{(3)}$ and $x_2^{(3)}$ are:

$$x^{(3)} = x^{(2)} + \Delta x = \begin{pmatrix} 2.00000 \\ 2.999999 \end{pmatrix}$$

The approximate relative error is given by:

$$\varepsilon_r = \frac{\sqrt{(0.00130)^2 + (-0.00229)^2}}{\sqrt{(2.00000)^2 + (2.999999)^2}} = 0.00073$$

Finally, for the fourth iteration the vector f and the matrix K have components:

$$f = \begin{pmatrix} 0.00000 \\ -0.00002 \end{pmatrix} \qquad K = \begin{pmatrix} 7 & 2 \\ 27 & 37 \end{pmatrix}$$

The components of the vector Δx can be computed as follows:

$$\Delta x = -K^{-1} f = \begin{pmatrix} 0.00000 \\ 0.00000 \end{pmatrix}$$

Therefore, the new estimates for $x_1^{(4)}$ and $x_2^{(4)}$ are:

$$x^{(4)} = x^{(3)} + \Delta x = \begin{pmatrix} 2 \\ 3 \end{pmatrix}$$

The approximate relative error is given by:

$$\varepsilon_r = \frac{\sqrt{(0.00000)^2 + (0.00000)^2}}{\sqrt{(2)^2 + (3)^2}} = 0.00000$$

Therefore, convergence is achieved after 4 iterations which is much faster than the 9 iterations in the fixed-point iteration method. The following is the Microsoft Excel table showing the values generated in every iteration:

Iteration		x_in	K matrix		f	Δ x		x_out		er
1	x_1 in	1.5	6.5	1.5	-2.50000	Δ x_1	0.53603	x_1 out	2.036029	0.24224
	x_2 in	3.5	36.75	32.5	1.62500	Δ x_2	-0.65612	x_2 out	2.843875	
2	x_1 in	2.03603	6.9159	2.0360	-0.06437	Δ x_1	-0.03733	x_1 out	1.998701	0.04512
	x_2 in	2.84388	24.2629	35.7413	-4.75621	Δ x_2	0.15841	x_2 out	3.002289	
3	x_1 in	1.99870	6.9997	1.9987	-0.00452	Δ x_1	0.00130	x_1 out	2.000000	0.00073
	x_2 in	3.00229	27.0412	37.0041	0.04957	Δ x_2	-0.00229	x_2 out	2.999999	
4	x_1 in	2.000000	7.0000	2.0000	0.00000	Δ x_1	0.00000	x_1 out	2.000000	0.0000002
	x_2 in	2.999999	27.0000	37.0000	-0.00002	Δ x_2	0.00000	x_2 out	3.000000	

The Newton-Raphson method can be implemented directly in Mathematica using the "FindRoot" built-in function:

VIEW MATHEMATICA CODE ⌄ VIEW PYTHON CODE ⌄

MATLAB code for the Newton-Raphson example
MATLAB file 1 (nrsystem.m)
MATLAB file 2 (f.m)
MATLAB file 3 (K.m)

ANOTHER EXPLANATION:

Nonlinear systems

It is possible to use Newton–Raphson method to solve a *system* of nonlinear equations:

$$\begin{aligned} f_1(x_1,\ldots,x_n) &= 0 \\ &\vdots \\ f_n(x_1,\ldots,x_n) &= 0 \end{aligned}$$

Note that the number of unknowns equals the number of equations. We can write this using vector notation as

$$\mathbf{f}(\mathbf{x}) = \mathbf{0},$$

where $\mathbf{x} = (x_1,\ldots,x_n)'$ and $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}),\ldots,f_n(\mathbf{x}))'$. The Newton–Raphson iteration for this system is

$$\mathbf{x}_{n+1} = \mathbf{x}_n - (D\mathbf{f}(\mathbf{x}_n))^{-1}\mathbf{f}(\mathbf{x}_n),$$

where $D\mathbf{f}(\mathbf{x})$ is the Jacobian matrix with (i,j)–th entry given by

$$\frac{\partial f_i(\mathbf{x})}{\partial x_j}.$$

EXAMPLE: Let us solve the system

$$\begin{aligned} xy + x^2 - y^3 - 1 &= 0 \\ x + 2y - xy^2 - 2 &= 0 \end{aligned}$$

Here $f_1(x,y) = xy + x^2 - y^3 - 1$ and $f_2(x,y) = x + 2y - xy^2 - 2$. So the Jacobian matrix is

$$D\mathbf{f}(\mathbf{x}) = \begin{bmatrix} y + 2x & x - 3y^2 \\ 1 - y^2 & 2 - 2xy \end{bmatrix}$$

This has inverse given by

$$(D\mathbf{f}(\mathbf{x}))^{-1} = \frac{1}{(y+2x)(2-2xy) - (x-3y^2)(1-y^2)} \begin{bmatrix} 2 - 2xy & 3y^2 - x \\ y^2 - 1 & y + 2x \end{bmatrix}$$

The following table shows a few sample iterations.

n	x	y
0	0.34	0.5
1	1.0896157	0.6101134
2	0.8689638	0.9595518
3	0.9842601	0.9682277
4	0.9902055	0.9854784
5	0.9951545	0.9927297
6	0.9975793	0.9963689
7	0.9987903	0.9981854
8	0.9993953	0.9990929
9	0.9996977	0.9995465
10	0.9998489	0.9997733
11	0.9999244	0.9998866
12	0.9999622	0.9999433
13	0.9999811	0.9999717
14	0.9999906	0.9999858
15	0.9999953	0.9999929
16	0.9999976	0.9999965
17	0.9999988	0.9999982
18	0.9999994	0.9999991
19	0.9999997	0.9999996
20	0.9999999	0.9999998

Obviously we are converging to the solution $x = 1, y = 1$

EXAMPLE:

Find the roots of the following nonlinear system equations.

$$x_1^2 + x_2^2 = 1, \qquad x_2 = \sin(x_1).$$

Write this as a root-finding problem: $f_1 = f_2 = 0$ *with*

$$f_1(x_1, x_2) = x_1^2 + x_2^2 - 1, \qquad f_2(x_1, x_2) = x_2 - sin(x_1).$$

The Jacobian matrix is

$$J = \nabla \mathbf{f}(\mathbf{x}) = \begin{pmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{pmatrix} = \begin{pmatrix} 2x_1 & 2x_2 \\ -\cos(x_1) & 1 \end{pmatrix}.$$

Use the formula for the inverse of a 2-by-2 matrix:

$$\begin{pmatrix} a & b \\ c & d \end{pmatrix}^{-1} = \frac{1}{ad - bc} \begin{pmatrix} d & -b \\ -c & a \end{pmatrix},$$

to obtain

$$J^{-1} = \frac{1}{2x_1 + 2x_2 \cos(x_1)} \begin{pmatrix} 1 & -2x_2 \\ \cos(x_1) & 2x_1 \end{pmatrix}.$$

The Newton iteration is therefore

$$\begin{pmatrix} x_{1,n+1} \\ x_{2,n+1} \end{pmatrix} = \begin{pmatrix} x_{1,n} \\ x_{2,n} \end{pmatrix} - J^{-1} \begin{pmatrix} x_{1,n}^2 + x_{2,n}^2 - 1 \\ x_{2,n} - \sin x_{1,n} \end{pmatrix}.$$