An Elegant Form of the Newton-Raphson Method for the Solution of a set of *N* Non-Linear Equations

Eduardo Fontana
Dep. de Eletrônica e Sistemas
Universidade Federal de Pernambuco
UFPE – CTG - Bloco A, Sala 421, Recife – PE 50740-550 Brasil
E-mail: fontana@ufpe.br

Published: February 10, 2013

Abstract: An elegant matrix form is presented for the generalized Newton-Raphson Method for *N* non-linear equations, to enable simple implementation of the method in numerical computation.

To solve a set of *N* non-linear equations, a direct extension of the Newton-Raphson[1] method is used. For a single variable, the method is based on the simple first order expansion of a function *u* around a point $x = x_g$,

$$u(x_g + \delta x) \approx u(x_g) + u'\left(x_g\right)\delta x, \tag{1}$$

with δx representing a small increment of x_g and

$$u'\left(x_g\right) \equiv \left.\frac{du}{dx}\right|_{x=x_g}. \tag{2}$$

If one wants to determine a root of *u* and if x_g is an initial guess for this root, to a first approximation, the first correction δx can be obtained by setting the RHS of (1) to zero, thus yielding

$$\delta x \approx -\left[u'(x_g)\right]^{-1} u(x_g). \tag{3}$$

As the value of x_g approaches a given root of *u*, the RHS of (3) becomes progressively smaller, so does the correction δx The Newton-Raphson method comprises applying (3) iteratively until a root is found within a given uncertainty ε.

A form very similar to (3) can be obtained for the case of *N* non-linear equations having *N* independent variables. In this case one employs the first order approximation to a function of *N* variables, i.e.,

$$u\left(\tilde{X} + \delta\tilde{X}\right) \approx u\left(\tilde{X}\right) + \left(\tilde{\nabla}^T u\right)\delta\tilde{X}, \tag{4}$$

with

$$\tilde{X} = \begin{pmatrix} x_1 & x_2 & \dots & x_N \end{pmatrix}^T \tag{5}$$

representing a column matrix[1] having the *N* variables as elements and

[1] In the present formulation, matrices are represented by capital letters with an appended tilde symbol.

$$\tilde{\nabla} \equiv \begin{pmatrix} \dfrac{\partial}{\partial x_1} & \dfrac{\partial}{\partial x_2} & \cdots & \dfrac{\partial}{\partial x_N} \end{pmatrix}^T , \tag{6}$$

the column matrix form of the gradient operator. Superscript *T* in (5) and (6) denotes the transpose operation on a matrix.

With the first-order Taylor expansion given in (4), one can easily extend the result given in (3) by writing the set of *N* non-linear equations in the form

$$\tilde{F} = \tilde{0} , \tag{7}$$

with $\tilde{F}$ representing a column matrix with each element corresponding to a non-linear function of the system, i.e.,

$$\tilde{F}(\tilde{X}) = \begin{pmatrix} f_1(\tilde{X}) & f_2(\tilde{X}) & \dots & f_N(\tilde{X}) \end{pmatrix}^T . \tag{8}$$

and with the column matrix $\tilde{0}$ in (7) having *N* null elements.

At the *k*-th iteration one defines the following quantities

$$\tilde{X}(k) \equiv \begin{pmatrix} x_1(k) & x_2(k) & \dots & x_N(k) \end{pmatrix}^T , \tag{9}$$

with $x_i(k)$ representing the value of the *i*-th element of $\tilde{X}$ at the *k*-th iteration,

$$\tilde{\nabla} u(k) \equiv \tilde{\nabla} u \Big|_{\tilde{X} = \tilde{X}(k)} , \tag{10}$$

$$\tilde{F}(k) \equiv \tilde{F}\left[\tilde{X}(k) \right] . \tag{11}$$

The generalized derivative of $\tilde{F}$ can be defined in the form of an $N \times N$ matrix as

$$\tilde{F}' \equiv \begin{pmatrix} \tilde{\nabla} f_1 & \tilde{\nabla} f_2 & \dots & \tilde{\nabla} f_N \end{pmatrix}^T , \tag{12}$$

or, by use of the Kronecker product [2], in the compact form

$$\tilde{F}' = \tilde{\nabla}^T \otimes \tilde{F} , \tag{13}$$

with a corresponding value at the *k*-th iteration

$$\tilde{F}'(k) \equiv \begin{pmatrix} \tilde{\nabla} f_1(k) & \tilde{\nabla} f_2(k) & \ldots & \tilde{\nabla} f_N(k) \end{pmatrix}^T, \tag{14}$$

or equivalently

$$\tilde{F}'(k) \equiv \tilde{\nabla}^T \otimes \tilde{F}(k). \tag{15}$$

Assuming that an initial guess $\tilde{X}(0)$ is set to the matrix $\tilde{X}$, by use of (4) and (14), it can be shown, after a few algebraic manipulations, and by extension of the result given in (3), that the new matrix $\tilde{X}(1)$, at the first iteration, can be written approximately as

$$\tilde{X}(1) \approx \tilde{X}(0) - \tilde{F}'(0)^{-1}\tilde{F}(0). \tag{16}$$

or, according to (13),

$$\tilde{X}(1) \approx \tilde{X}(0) - \left[\tilde{\nabla}^T \otimes \tilde{F}(0)\right]^{-1}\tilde{F}(0). \tag{17}$$

This procedure is repeated iteratively and at the (*k*+1)-th iteration, one obtains

$$\tilde{X}(k+1) \approx \tilde{X}(k) - \tilde{F}'(k)^{-1}\tilde{F}(k). \tag{18}$$

Alternatively, (18) can be written as

$$\tilde{X}(k+1) \approx \tilde{X}(k) - \left[\tilde{\nabla}^T \otimes \tilde{F}(k)\right]^{-1}\tilde{F}(k). \tag{19}$$

The iterative procedure is stopped within a certain maximum uncertainty of the elements of $\tilde{X}$. For example, defining the absolute difference matrix at the (*k*+1)-th iteration as

$$\tilde{\Delta}(k+1) \equiv \begin{pmatrix} \Delta_1(k+1) & \Delta_2(k+1) & \ldots & \Delta_N(k+1) \end{pmatrix}^T, \tag{20}$$

with

$$\Delta_i(k+1) \equiv \left|x_i(k+1) - x_i(k)\right|, \tag{21}$$

one can establish, for instance, that the iterative procedure should be interrupted when the maximum variation of any element of $\tilde{X}$ is ε, i.e.,

$$\max\left[\tilde{\Delta}\left(k+1\right)\right] < \varepsilon, \tag{22}$$

with $\max[.]$ representing the largest element of the matrix $\tilde{\Delta}\left(k+1\right)$.

In implementing computational algorithms to solve a set of *N* non-linear equations, basic definitions (5) and (8) are used to define functions and variables, and (18) or (19) is used for the iterative calculation, with the interruption criterion (22). Table I shows a pseudocode for the computational implementation of the procedure. As can be noticed from the program structure, by use of the matrix formulation presented in this paper one can organize the algorithm using a rather simple and modular scheme.

Table I – Pseudocode to solve a set of *N* non-linear equations.

1. Initialization:
   - Define matrix $F(X)$ all at once. In this instance, *X* is defined along the definition of *F*.
   - Calculate $F'(X)$. To improve speed, derivatives can be calculated numerically.
   - Set initial guess Y.
   - Set a small positive value to the uncertainty parameter ε.
   - $err \leftarrow 100$ (Set a high value to the error parameter)

2. Calculation:

```
Root ← | while $err > \varepsilon$
       |     $Y\text{old} \leftarrow Y$
       |     $Y \leftarrow Y - \left[F'(Y)\right]^{-1} F(Y)$
       |     $err \leftarrow \max\left(\overrightarrow{\left|Y - Y\text{old}\right|}\right)$
       | $Y$
```

3. Remarks:
   - Mathcad programming is used as a model.
   - Parameters X, $F(X)$, $F'(X)$, $Yold$, Y and $Root$ are matrices.
   - Parameters err and ε are scalars.
   - The vectorize operation $\overrightarrow{\left|Y - Y\text{old}\right|}$ produces a matrix in which each element is the absolute value of the difference between corresponding elements of Y and $Y\text{old}$.
   - The function max( ) calculates the maximum element of the matrix $\overrightarrow{\left|Y - Y\text{old}\right|}$.
   - The last value of Y stores the solution of the set of *N* non-linear equations.

References

[1] G. B. Arfken and H. J. Weber, "Mathematical Methods for Physicists," 5th. ed., San Diego, CA, Academic Press, pp. 1085-1086 (2001).

[2] Wikipedia contributors, "Kronecker product," Wikipedia, The Free Encyclopedia, Accessed February 6, 2013.