

For comparison purposes, note that Newton's method with initial guess $x_0 = -3$ converges in 6 iterations to the same solution -2.074304 of $f(x) = 0$:

	Secant Method	Newton's Method
x_1	-2.503129020	-2.5
x_3	-2.309651708	-2.211666639
x_3	-2.159035029	-2.094956590
x_4	-2.095347041	-2.074874887
x_5	-2.076550029	-2.074304856
x_6	-2.074368712	-2.074304403
x_7	-2.074304603	
x_8	-2.074304403	

8.3 Numerical methods for N -dimensional nonlinear problems

For N dimensional problems there are no practical analogs of the bisection method. However, Newton's method can easily be extended to N dimensional problems. An approximate Newton's method similar to the secant method and based on finite difference-type approximations also exists.

8.3.1 The N -dimensional Newton's Method

Let $F : \mathbb{R}^N \rightarrow \mathbb{R}^N$ given by $F(x) = (F_i(x))_{i=1:N}$, where $F_i : \mathbb{R}^N \rightarrow \mathbb{R}$. Assume that all the partial order derivatives of the functions $F_i(x)$, $i = 1 : N$, are continuous. We want to solve the nonlinear N -dimensional problem

$$F(x) = 0.$$

Recall from (1.38) that the gradient $DF(x)$ of $F(x)$ is an $N \times N$ matrix:

$$DF(x) = \begin{pmatrix} \frac{\partial F_1}{\partial x_1}(x) & \frac{\partial F_1}{\partial x_2}(x) & \cdots & \frac{\partial F_1}{\partial x_N}(x) \\ \frac{\partial F_2}{\partial x_1}(x) & \frac{\partial F_2}{\partial x_2}(x) & \cdots & \frac{\partial F_2}{\partial x_N}(x) \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial F_N}{\partial x_1}(x) & \frac{\partial F_N}{\partial x_2}(x) & \cdots & \frac{\partial F_N}{\partial x_N}(x) \end{pmatrix}.$$

The recursion for Newton's method follows from the linear Taylor expansion of $F(x)$, i.e.,

$$F(x) \approx F(x_k) + DF(x_k)(x - x_k); \quad (8.41)$$

cf. (5.34) for $a = x_k$. Here, $DF(x_k)(x - x_k)$ is a matrix–vector multiplication. Let $x = x_{k+1}$ in (8.41). Approximating $F(x_{k+1})$ by 0 (which happens in the limit, if convergence to a solution for $F(x) = 0$ is achieved), we find that

$$0 \approx F(x_k) + DF(x_k)(x_{k+1} - x_k). \quad (8.42)$$

Changing (8.42) into an equality and solving for x_{k+1} , we obtain the following recursion for Newton's method for N dimensional problems:

$$x_{k+1} = x_k - (DF(x_k))^{-1}F(x_k), \quad \forall k \geq 0. \quad (8.43)$$

At each step, the vector $(DF(x_k))^{-1}F(x_k)$ must be computed. In practice, the inverse matrix $(DF(x_k))^{-1}$ is never explicitly computed, since this would be very expensive computationally. Instead, we note that computing the vector $v_k = (DF(x_k))^{-1}F(x_k)$ is equivalent to solving the linear system

$$DF(x_k)v_k = F(x_k).$$

This is done using numerical linear algebra methods, e.g., by computing the LU decomposition factors of the matrix $DF(x_k)$ and then doing a forward and a backward substitution.

It is not our goal here to discuss such methods; see [27] for details on numerical linear algebra methods. We subsequently assume that a routine for solving linear systems called `solve_linear_system` exists such that, given a nonsingular square matrix A and a vector b , the vector

$$x = \text{solve_linear_system}(A, b)$$

is the unique solution of the linear system $Ax = b$.

The vector $v_k = (DF(x_k))^{-1}F(x_k)$ can then be computed as

$$v_k = \text{solve_linear_system}(DF(x_k), F(x_k)),$$

and recursion (8.43) can be written as

$$x_{k+1} = x_k - \text{solve_linear_system}(DF(x_k), F(x_k)), \quad \forall k \geq 0.$$

The N -dimensional Newton's method iteration is stopped and convergence to a solution to the problem $F(x) = 0$ is declared when the following two conditions are satisfied:

$$\|F(x_{new})\| \leq \text{tol_approx} \quad \text{and} \quad \|x_{new} - x_{old}\| \leq \text{tol_consec}, \quad (8.44)$$

where x_{new} is the most recent value generated by Newton's method and x_{old} is the value previously computed by the algorithm. Here, $\| \cdot \|$ represents

the Euclidean norm⁶. Possible choices for the tolerance factors are `tol_consec` = 10^{-6} and `tol_approx` = 10^{-9} ; see the pseudocode from Table 8.2 for more details.

Table 8.4: Pseudocode for the N -dimensional Newton's Method

<p>Input: x_0 = initial guess $F(x)$ = given function <code>tol_approx</code> = largest admissible value of $\ F(x)\$ when solution is found <code>tol_consec</code> = largest admissible distance between two consecutive approximations when solution is found</p> <p>Output: x_{new} = approximate solution for $f(x) = 0$</p> <p>$x_{new} = x_0; x_{old} = x_0 - 1$ while ($\ F(x_{new})\ > \text{tol_approx}$) or ($\ x_{new} - x_{old}\ > \text{tol_consec}$) $x_{old} = x_{new}$ compute $DF(x_{old})$ $x_{new} = x_{old} - \text{solve_linear_system}(DF(x_{old}), F(x_{old}))$ end</p>
--

As was the case for the one-dimensional version, the N -dimensional Newton's method converges quadratically if certain conditions are satisfied.

Theorem 8.5. *Let x^* be a solution of $F(x) = 0$, where $F(x)$ is a function with continuous second order partial derivatives. If $DF(x^*)$ is a nonsingular matrix, and if x_0 is close enough to x^* , then Newton's method converges quadratically, i.e., there exists $M > 0$ and n_M a positive integer such that*

$$\frac{\|x_{k+1} - x^*\|}{\|x_k - x^*\|^2} < M, \quad \forall k \geq n_M.$$

Example: Use Newton's method to solve $F(x) = 0$, for

$$F(x) = \begin{pmatrix} x_1^3 + 2x_1x_2 + x_3^2 - x_2x_3 + 9 \\ 2x_1^2 + 2x_1x_2^2 + x_2^3x_3^2 - x_2^2x_3 - 2 \\ x_1x_2x_3 + x_1^3 - x_3^2 - x_1x_2^2 - 4 \end{pmatrix}$$

⁶If $v = (v_i)_{i=1:N}$ is a vector in \mathbb{R}^N , then

$$\|v\| = \left(\sum_{i=1}^N |v_i|^2 \right)^{1/2}.$$

Answer: Note that

$$DF(x) = \begin{pmatrix} 3x_1^2 + 2x_2 & 2x_1 - x_3 & 2x_3 - x_2 \\ 4x_1 + 2x_2^2 & 4x_1x_2 + 3x_2^2x_3^2 - 2x_2x_3 & 2x_3^3x_3 - x_2^2 \\ x_2x_3 + 3x_1^2 - x_2^2 & x_1x_3 - 2x_1x_2 & x_1x_2 - 2x_3 \end{pmatrix}.$$

We use the algorithm from Table 8.4 with $\text{tol.consec} = 10^{-6}$ and $\text{tol.approx} = 10^{-9}$. For the initial guess

$$x_0 = \begin{pmatrix} 1 \\ 2 \\ 3 \end{pmatrix}, \text{ the solution } x^* = \begin{pmatrix} -1.690550759854953 \\ 1.983107242868416 \\ -0.884558078475291 \end{pmatrix}$$

is found after 9 iterations.

For the initial guess $x_0 = (2 \ 2 \ 2)^t$, the solution $x^* = (-1 \ 3 \ 1)^t$ is found after 40 iterations. \square

8.3.2 The Approximate Newton's Method

In many instances, it is not possible (or efficient) to find a closed formula for the matrix $DF(x)$ which is needed for Newton's method; cf. (8.43). In these cases, finite difference approximations can be used to estimate each entry of $DF(x)$. The resulting method is called the Approximate Newton's Method.

The entry of $DF(x)$ on the position (i, j) , i.e., $\frac{\partial F_i}{\partial x_j}(x)$, is estimated using forward finite differences approximations (6.2) as

$$\frac{\partial F_i}{\partial x_j}(x) \approx \Delta_j F_i(x) = \frac{F_i(x + he_j) - F_i(x)}{h}, \quad (8.45)$$

where h is a small number and e_j is a vector with all entries equal to 0 with the exception of the j -th entry, which is equal to 1, i.e., $e_j(k) = 0$, for all $k \neq j$ and $e_j(j) = 1$. Note that if $x = (x_i)_{i=1:N}$, then

$$x + he_j = \begin{pmatrix} x_1 \\ \vdots \\ x_{j-1} \\ x_j + h \\ x_{j+1} \\ \vdots \\ x_N \end{pmatrix}.$$

Let

$$\Delta F(x) = \begin{pmatrix} \Delta_1 F_1(x) & \Delta_2 F_1(x) & \dots & \Delta_N F_1(x) \\ \Delta_1 F_2(x) & \Delta_2 F_2(x) & \dots & \Delta_N F_2(x) \\ \vdots & \vdots & \vdots & \vdots \\ \Delta_1 F_N(x) & \Delta_2 F_N(x) & \dots & \Delta_N F_N(x) \end{pmatrix}.$$