Introduction to Scientific Computing

(Lecture 7: Nonlinear system of equations)

Bojana Rosić

Institute of Scientific Computing

December 13, 2016

Problem

Our goal is to solve the nonlinear system of equations

$$\mathbf{F}(\mathbf{x}) = \mathbf{0}$$

for unknown $\mathbf{x}$ by iterative procedures. The exact solution will be denoted by $\mathbf{x}_*$.

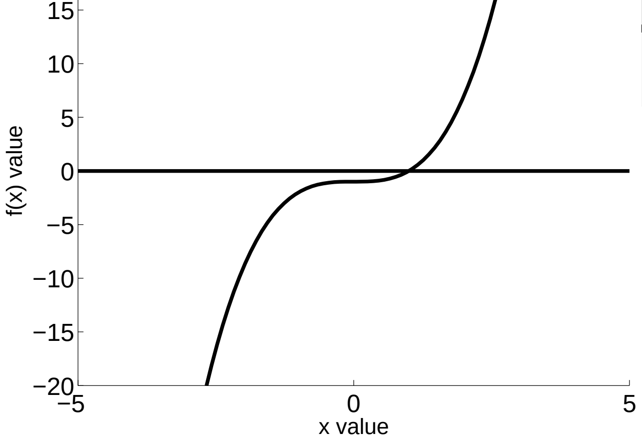
Newton method

One dimensional problem

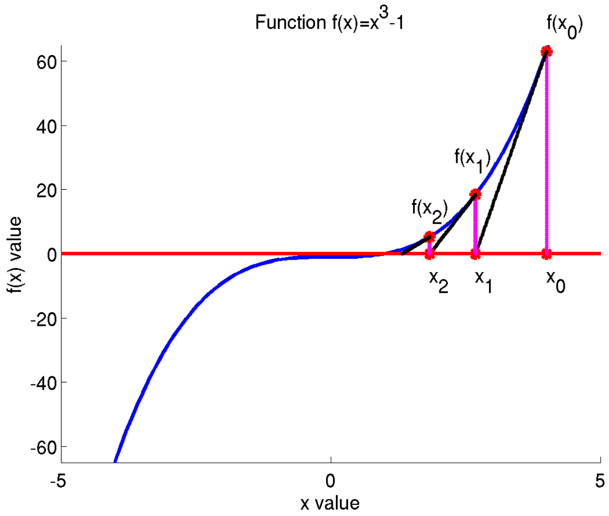
Find the root of equation

$$f(x) = 0$$

What this r



Newton-Raphson method



Geomterical interpretation

Let us observe the line which connects two points $(x_0, f(x_0))$ and (x, y). The equation of line passing through these two points reads

$$y = f(x_0) + (x - x_0)f'(x_0)$$

in which $f'(x_0)$ is the slope (tangent line) at point x_0. We would like to find the point where this line crosses x axis (hopefully this will be our root). This mathces with the condition

$$y = 0, \quad x = x_1$$

Hence,

$$0 = f(x_0) + (x_1 - x_0)f'(x_0) \Rightarrow x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

Geomterical interpretation

Note that

$$f'(x_0) = \mathrm{tg}\ \alpha = \frac{f(x_0)}{x_1 - x_0}$$

i.e.

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}.$$

When we iterate this expression, one obtains

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}$$

which further will be denoted by

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}$$

Mathematical interpretation

Let the solution at iteration k be denoted by $x^{(k)}$, then one may expand the function $f(x)$ in the neighbourhood of $x^{(k)}$ by

$$f(x) = f(x^{(k)}) + f'(x^{(k)})(x - x^{(k)}) + \mathrm{h.o.t}$$

Because $f(x) = 0$, one has

$$0 = f(x^{(k)}) + f'(x^{(k)})(x - x^{(k)}) + \mathrm{h.o.t}$$

i.e.

$$x = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}$$

Mathematical interpretation

The last relation can be iterated in the following manner

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}$$

until *convergence*. From this it follows that the initial point $x^{(0)}$ can have huge influence on the quality of solution. Namely, if $x^{(0)}$ is too far from the solution the method can even diverge. Why? The reason is the accuracy of the Taylor expansion.

Example

Solve the following equations

1. $x^3 - 1 = 0$
2. $(x - 5)^2 = 0$
3. $x^3 - 2x + 2 = 0$

Example: Newton method

The Newton method reads

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}$$

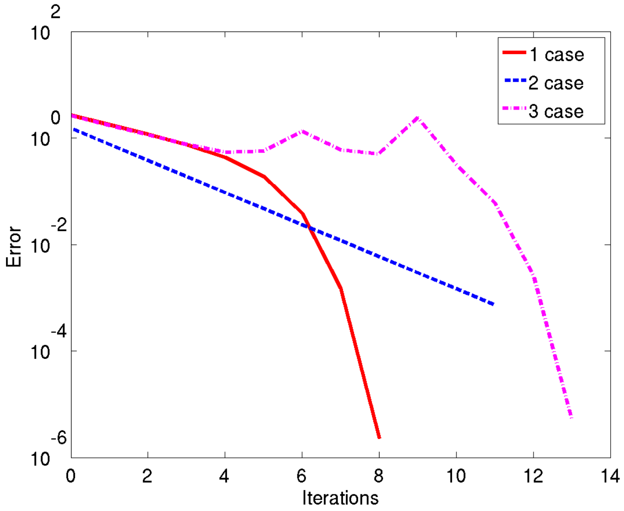
i.e. in our particular case

1. $f(x) = x^3 - 1 = 0 \Rightarrow x^{(k+1)} = x^{(k)} - \frac{(x^{(k)})^3 - 1}{3(x^{(k)})^2}$
2. $f(x) = (x-5)^2 = 0 \Rightarrow x^{(k+1)} = x^{(k)} - \frac{(x^{(k)}-5)^2}{2(x^{(k)}-5)}$
3. $f(x) = x^3 - 2x + 2 = 0 \Rightarrow x^{(k+1)} = x^{(k)} - \frac{(x^{(k)})^3 - 2x^{(k)} + 2}{3(x^{(k)})^2 - 2}$

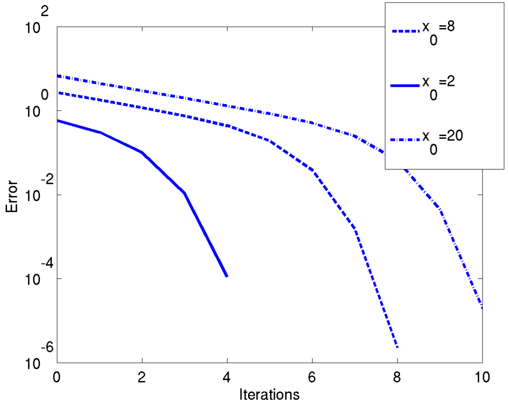
Example: solution

No. of iter.	1. case	2. case	3. case
1	8	8	8
2	5.3385	6.5000	5.3789
3	3.5707	5.7500	3.6470
4	2.4066	5.3750	2.5068
5	1.6620	5.1875	1.7509
6	1.2287	5.0938	1.2137
7	1.0399	5.0469	0.6514
8	1.0015	5.0234	1.9905
9		5.0117	1.3932
10		5.0059	0.8915
11			-1.5163
12			-1.8320
13			-1.7719

Newton method

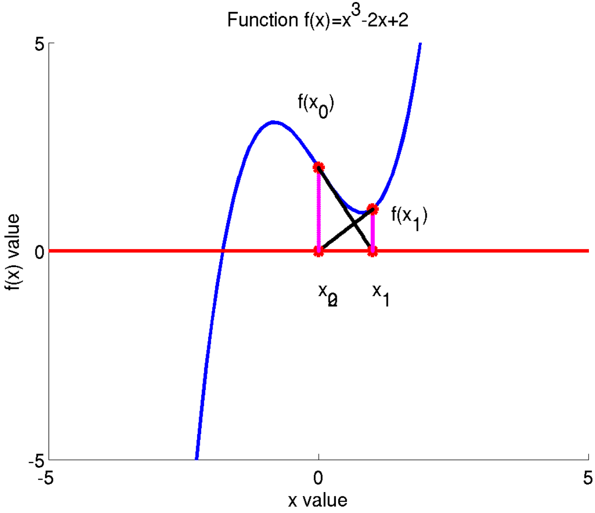


Sensitivity on initial point



Convergence as well as number of iterations depend on the initial guess!

Do we always converge and hit global minimum?



Convergence of the method

To check convergence, one may observe the mapping

$$x^{(k+1)} = \Phi(x^{(k)}) = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}$$

and check its contractivity. For this we may check

$$q = \sup_{x \in [x_* - \delta, x_* + \delta]} |\Phi'|$$

where x_* is the solution (fixed point) and δ is the neighborhood of x_*.

Is the method convergent?

Since

$$\Phi' = 1 - \frac{(f')^2 - ff^{''}}{(f')^2} = \frac{ff^{''}}{(f')^2},$$

and $f'(x_*) \neq 0$ (unles x_* is multiple root), as well as $f(x_*) = 0$ (solution), then

$$\Phi'(x_*) = 0.$$

By the evident continuity of Φ', given any $K > 0$ and $K < 1$

$$\Phi'(x) < K$$

if $x \in [x_* - \delta, x_* + \delta]$ and δ is sufficiently small.

Order of convergence

To compute the order one may evaluate the error

$$\begin{aligned}
d^{k+1} &:= x^{k+1} - x_* = \Phi(x_k) - \Phi(x_*) \\
&= \Phi(x_* + d_k) - \Phi(x_*) \\
&= \Phi(x_*) + \underbrace{\Phi'(x_*)}_{=0} d_k + \frac{1}{2}\Phi''(x_*)d_k^2 + \mathcal{O}(d_k^3) - \Phi(x_*) \\
&= \frac{1}{2}\Phi''(x_*)d_k^2 + \mathcal{O}(d_k^3).
\end{aligned}$$

It follows

$$|x^{(k+1)} - x_*| \leq C|x^{(k)} - x_*|^2.$$

Hence, the convergence is quadratic.

Previous example: error

No. of iter.	1. case	2. case	3. case
0	2.6615	1.5000	2.6211
1	1.7678	0.7500	1.7320
2	1.1641	0.3750	1.1401
3	0.7447	0.1875	0.7560
4	0.4333	0.0938	0.5371
5	0.1887	0.0469	0.5623
6	0.0384	0.0234	1.3391
7	0.0015	0.0117	0.5974
8	2.2828e-06	0.0059	0.5017
9		0.0029	2.4078
10		0.0015	0.3157
11		0.0007	0.0601
12		0	0.0026

Previous example: error

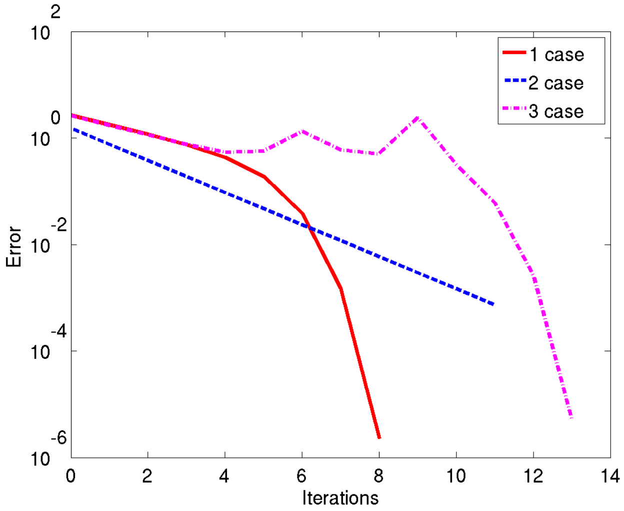
Check now the error between 6th and 7th iteration

$$\epsilon^{(6)} = \|x^{(6)} - x^{(5)}\|$$

$$\epsilon^{(7)} = \|x^{(7)} - x^{(6)}\|$$

1. case:$\epsilon^{(6)} = 0.0384 \Rightarrow (\epsilon^{(6)})^2 = 0.0015 = \epsilon^{(7)}$ (quadratic convergence)
2. case: $\epsilon^{(6)} = 0.0234 \Rightarrow (\epsilon^{(6)})^2 = 5.4756e - 04 < 0.0117 = \epsilon^{(7)}$ (linear convergence)
3. case: $\epsilon^{(6)} = 1.3391 > \epsilon^{(7)}, \epsilon^{(6)} > \epsilon^{(5)}$ (local divergence)

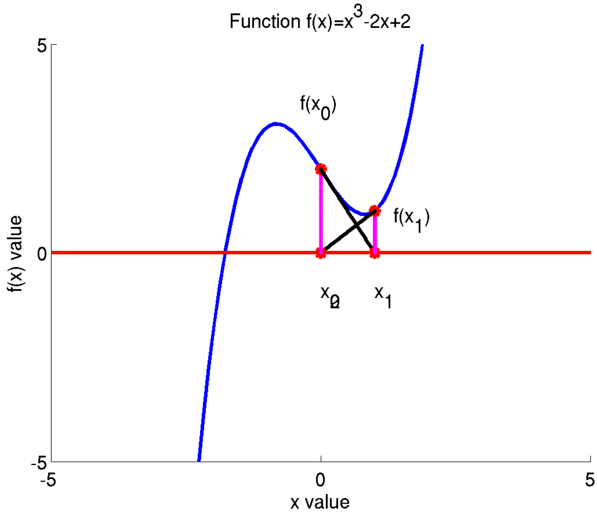
Newton method



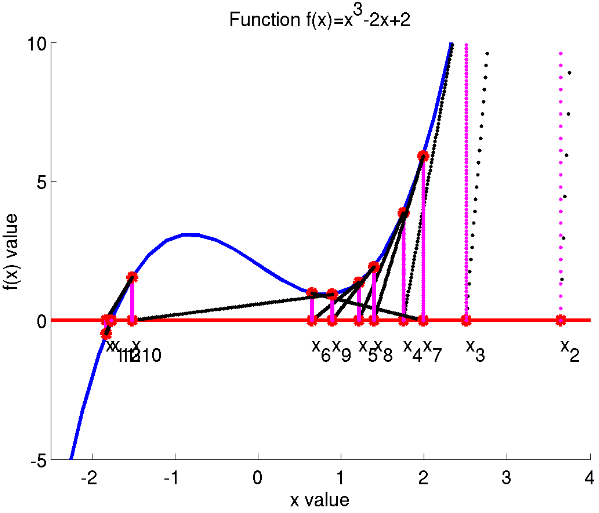
Example: Why this?

1. case has quadratic convergence because the root is not double and the initial point is not near any change from local minima to local maxima
2. case has linear convergence because the root is double
3. case has local divergence because the function changes its convexity (local minima/maxima) in a point between the initial point and the root of the function

Why local divergence?



Why local divergence?



Convergence is not always quadratic!

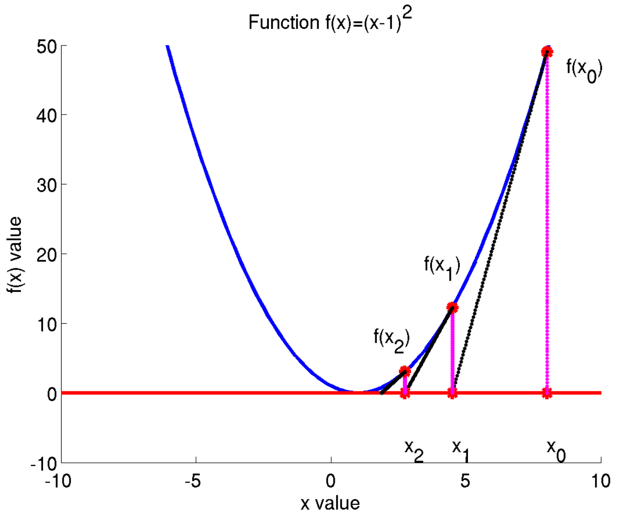
In case of multiple roots the method is only linearly convergent. Let x_* be a root of multiplicity 2, i. e. $f(x_*) = 0$ and $f'(x_*) = 0$. Then the Newton's method converges only linear for $x_0 \in (x_* - \delta, x_* + \delta)$, since

$$\Phi'(x_*) = \lim_{x \to x_*} \frac{f(x)f''(x)}{(f'(x))^2} = \frac{1}{2}.$$

Example:

$$f(x) = (x-1)^2$$
$$f'(x) = 2(x-1)$$

Is the method convergent?



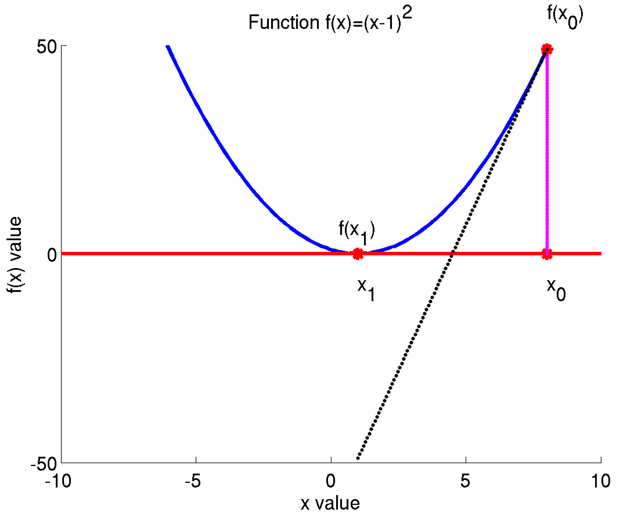
Modified Newton method

Let x_* be a root of multiplicity m, i. e. $f(x_*) = 0$, $f'(x_*) = 0$, and $f^{(m)}(x_*) = 0$. Then the modified Newton scheme reads as

$$x_{k+1} := x_k - m\frac{f(x_k)}{f'(x_k)}.$$

The convergence is then quadratic.

Modified Newton



Stopping criteria

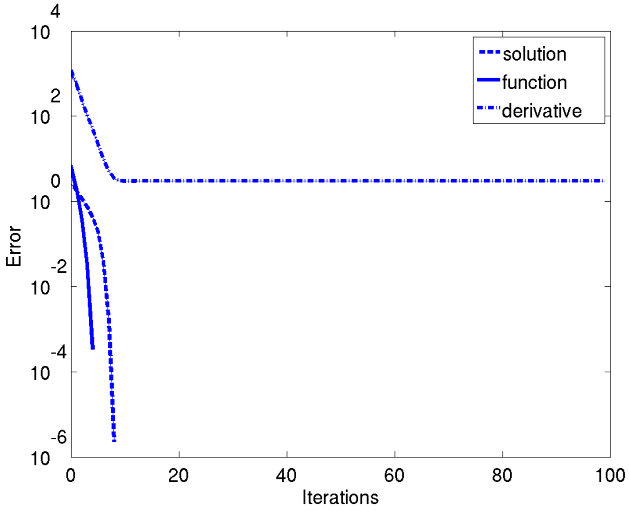
There are several stopping criteria that can be used in the practice

1. $\epsilon = \|x^{(k)} - x^{(k-1)}\| < tol$
2. $\epsilon = \|f(x^{(k)})\| < tol$
3. $\epsilon = \|f'(x^{(k)})\| < tol$ (this one is not easy to satisfy)

Note that other criteria also exist. They can combine some of the previous ones.

Stopping criteria

For the same tolerance $1e-3$



Newton-Raphson method for general system

In general case (not only for scalar function):

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \frac{\mathbf{F}(\mathbf{x}^{(k)})}{\mathbf{F}'(\mathbf{x}^{(k)})}$$

where (k) denotes the iteration number.

Never invert the matrix: inexact Newton method

The formula

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \frac{\mathbf{F}(\mathbf{x}^{(k)})}{\mathbf{F}'(\mathbf{x}^{(k)})}$$

can be rewritten as

$$\mathbf{F}'(\mathbf{x}^{(k)})(\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}) = -\mathbf{F}(\mathbf{x}^{(k)})$$

By denoting $\Delta\mathbf{x}^{(k)} = (\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)})$ and $\mathbf{J}_k = \mathbf{F}'(\mathbf{x}^{(k)})$ one may further write

$$\mathbf{J}_k \Delta\mathbf{x}^{(k)} = -\mathbf{F}(\mathbf{x}^{(k)})$$

To solve this system you may use any of methods we have learned in last two weeks. This means that inside the iteraion (k) one would perform another local iteration to solve the linear system.

Never invert the matrix: inexact Newton method

Hence,

$$\mathbf{J}_k \Delta \mathbf{x}^{(k)} = -\mathbf{F}(\mathbf{x}^{(k)})$$

can be solved by for example Jacobi method by taking

$$\mathbf{D} = \mathrm{diag}\,(\mathbf{J}_k), \quad \mathbf{R} = \mathbf{J}_k - \mathbf{D}$$

and then computing

$$(\Delta \mathbf{x}^{(k)})^{(i)} = (\Delta \mathbf{x}^{(k)})^{(i-1)} + \mathbf{D}^{-1}(-\mathbf{F}(\mathbf{x}^{(k)}) - \mathbf{J}_k(\Delta \mathbf{x}^{(k)})^{(i)})$$

Once $\Delta \mathbf{x}^{(k)}$ has converged one may compute

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \Delta \mathbf{x}^{(k)}$$

Example

$$f_1(x_1, x_2) = x_1^3 + x_2 - 1 = 0$$
$$f_2(x_1, x_2) = x_2^3 - x_1 + 1 = 0$$

The Jacobian:

$$\mathbf{J} := \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} \end{bmatrix}$$

$$J_{11} = \partial f_1 / \partial x_1 = 3x_1^2$$

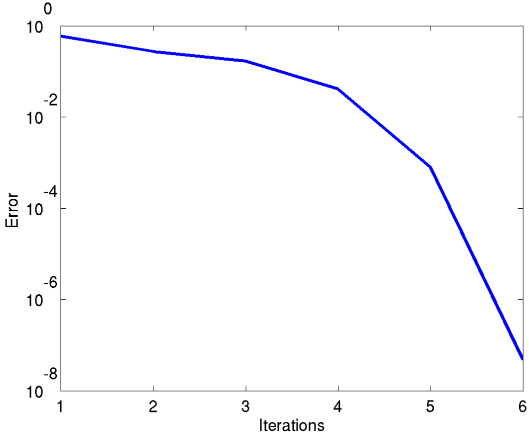
$$J_{12} = \partial f_1 / \partial x_2 = 1$$

$$J_{21} = \partial f_2 / \partial x_1 = -1$$

$$J_{22} = \partial f_2 / \partial x_2 = 3x_2^2$$

Example

x_1	x_2
0.5000	0.5000
1.0800	0.4400
0.9477	0.2033
0.9881	0.0399
0.9999	0.0008
1.0000	0.0000
1.0000	0.0000



Stationary Newton method

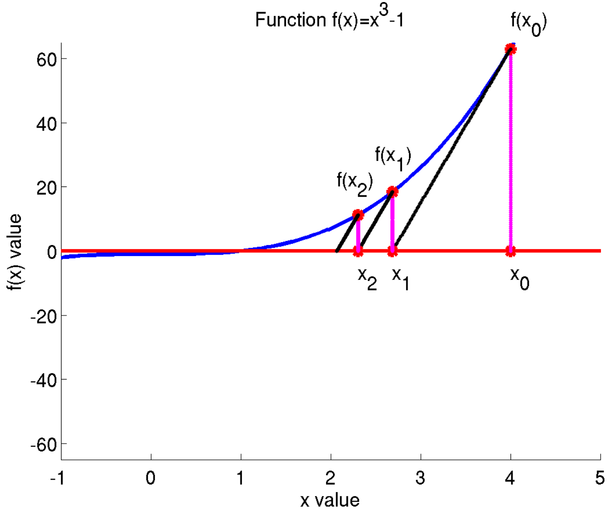
Does not require the knowledge of the Jacobian in each iteration and can be written as

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{B}_0^{-1}\mathbf{F}(\mathbf{x}^{(k)}),$$

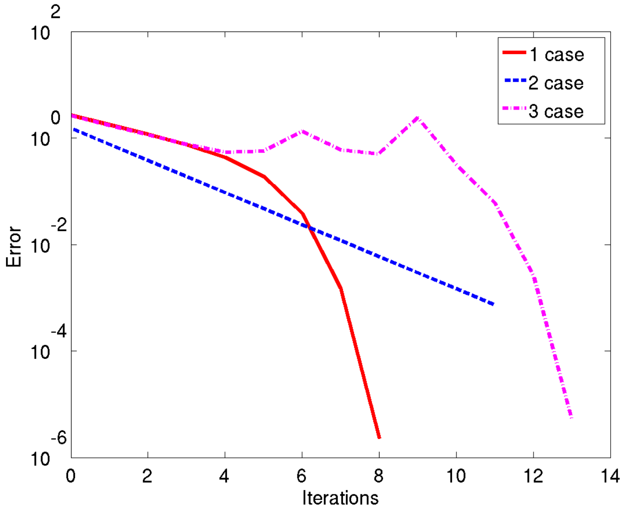
where the matrix $\mathbf{B}_0$ is the exact Jacobian in the first iteration.

This method requires more iterations, but they are cheaper.

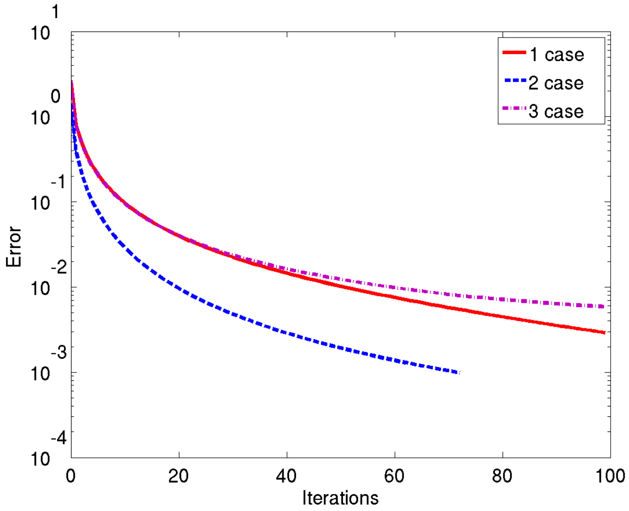
Stationary Newton method



Newton method



Stationary Newton method



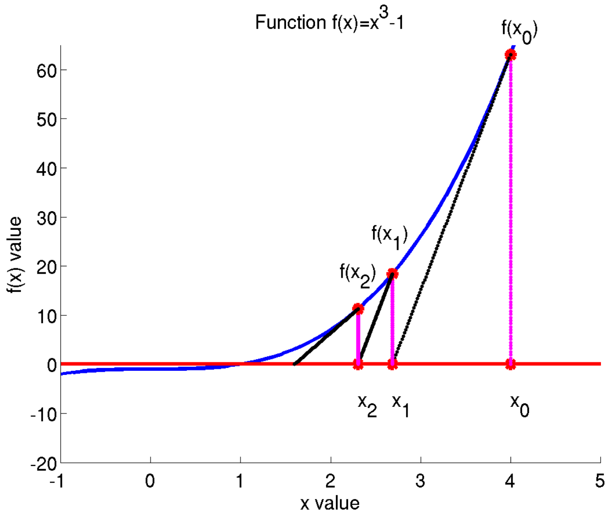
Newton method with restarts

Does not require the knowledge of the Jacobian in each iteration and can be written as

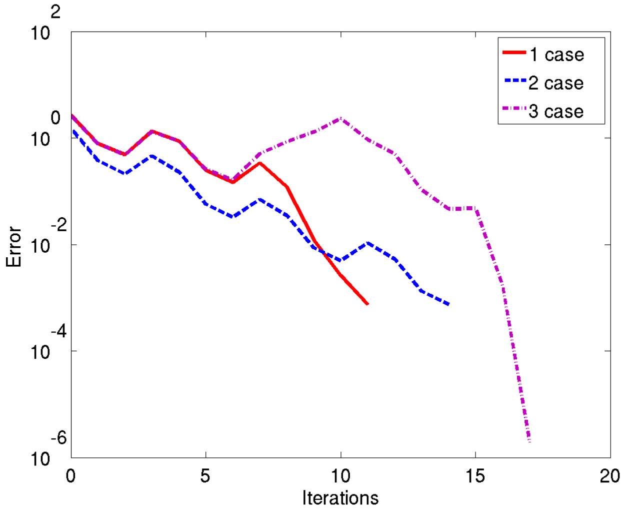
$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - \mathbf{B}_k^{-1}\mathbf{F}(\mathbf{x}^{(k)}), \quad \text{where}$$
$$\mathbf{B}_k = J_k, \quad mod(k,m) = 0, \quad \mathbf{B}_k = \mathbf{B}_{k-1} \quad \text{otherwise}$$

This means that we compute Jacobian after m iterations

Newton method with restarts



Newton method: restart



Secant method

One dimensional problem

Find the root of equation

$$f(x) = 0$$

Secant method

In each iteration of Newton method

$$x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}$$

one has to compute the Jacobian $f'(x^{(k)})$ which is usually very expensive operation. To avoid this, one may use approximation of Jacobian in a form of the difference quotient

$$f'(x^{(k)}) \approx \frac{f(x^{(k)}) - f(x^{(k-1)})}{x^{(k)} - x^{(k-1)}}$$

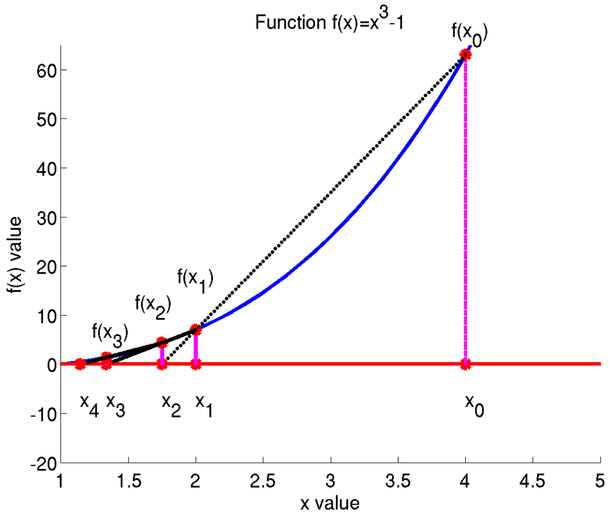
Secant method

Secant method:

$$x^{(k+1)} = x^{(k)} - \frac{x^{(k)} - x^{(k-1)}}{f\big(x^{(k)}\big) - f\big(x^{(k-1)}\big)} f\big(x^{(k)}\big)$$

This method needs two starting values and does not belong to the class of fixed point iterations.

Geometrical representation



Quasi-Newton method

Quasi-Newton method

Quasi–Newton methods are generalisations of the one dimensional secant method to higher space dimensions. They do not require the knowledge of the Jacobian and can be written as

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} - (\mathbf{B}^{(k)})^{-1}\mathbf{F}(\mathbf{x}^{(k)}), \quad \text{where}$$
$$\mathbf{B}^{(k)}\mathbf{s}^{(k)} = \mathbf{B}^{(k)}(\mathbf{x}^{(k)} - \mathbf{x}^{(k-1)}) = \mathbf{F}(\mathbf{x}^{(k)}) - \mathbf{F}(\mathbf{x}^{(k-1)}) = \mathbf{y}(\mathbf{x}^{(k)}).$$

The matrices $\mathbf{B}^{(k)}$ are the secant-approximation to the Jacobian in the k-th step.

- The second equation is called secant condition.
- When the iterative scheme satisfies the secant condition , it is called a Quasi-Newton scheme.

Quasi-Newton method

From the secant condition one may get infinitely many matrices $\mathbf{B}_k$ (n^2 unknowns given n knowns). To make the system well posed one could observe a series of secant conditions

$$\mathbf{B}^{(k)}\mathbf{s}^{(j)} = \mathbf{y}(\mathbf{x}^{(j)}), \quad j = i - n + 1, ..., i$$

This would mean that we have provided $n + 1$ points of $\mathbf{x}$. However, this would be very expensive and on the other side very unstable method. Instead of computing $\mathbf{B}^{(k)}$ from scratch, Broyden reasoned that the previous approximation $\mathbf{B}^{(k-1)}$ can be updated to$\mathbf{B}^{(k)}$ such that

$$m = \mathrm{rank}\ (\mathbf{B}^{(k)} - \mathbf{B}^{(k-1)}) = \mathrm{rank}\ (\Delta\mathbf{B}^{(k)})$$

is small (typically taken to be $m = 1$ or $m = 2$).

The rank of a matrix B is the size of the largest collection of linearly independent columns (rows) of B.

Broyden's method

Broyden's method updates the matrix $\mathbf{B}^{(k-1)}$ by

$$\mathbf{B}^{(k)} = \mathbf{B}^{(k-1)} + \Delta\mathbf{B}^{(k)} = \mathbf{B}^{(k-1)} + \mathbf{u}^{(k)}(\mathbf{s}^{(k)})^T$$

in which $\mathbf{u}^{(k)}(\mathbf{s}^{(k)})^T$ is rank one matrix and $u^{(k)}$ is unknown. From the secant condition

$$\mathbf{B}^{(k)}\mathbf{s}^{(k)} = \mathbf{y}^{(k)}$$

one has

$$\mathbf{B}^{(k-1)}\mathbf{s}^{(k)} + \mathbf{u}^{(k)}(\mathbf{s}^{(k)})^T\mathbf{s}^{(k)} = \mathbf{y}^{(k)}$$

i.e.

$$\mathbf{u}^{(k)} = \frac{\mathbf{y}^{(k)} - \mathbf{B}^{(k-1)}\mathbf{s}^{(k)}}{(\mathbf{s}^{(k)})^T\mathbf{s}^{(k)}} \Rightarrow \mathbf{B}^{(k)} = \mathbf{B}^{(k-1)} + \frac{\mathbf{y}^{(k)} - \mathbf{B}^{(k-1)}\mathbf{s}^{(k)}}{(\mathbf{s}^{(k)})^T\mathbf{s}^{(k)}}(\mathbf{s}^{(k)})^T$$

Broyden's method

Let $A \in \mathbb{R}^{d \times d}$ be a non-singular matrix, and let $\mathbf{u}, \mathbf{v} \in \mathbb{R}^d$. Then the following conclusions hold.

1. The matrix $\mathbf{B}^{(k)} = \mathbf{B}^{(k-1)} + \mathbf{u}\mathbf{v}^T$ is non-singular if and only if σ is nonzero, where
$$\sigma := 1 + \mathbf{v}^\top (\mathbf{B}^{(k-1)})^{-1}\mathbf{u}.$$
2. When σ is nonzero, then the inverse of $\mathbf{B}^{(k)}$ can be computed as
$$\mathbf{B}^{-1} = (\mathbf{B}^{(k-1)} + \mathbf{u} \otimes \mathbf{v})^{-1} = (\mathbf{B}^{(k-1)})^{-1} - \frac{(\mathbf{B}^{(k-1)})^{-1}(\mathbf{u}\mathbf{v}^T)(\mathbf{B}^{(k-1)})^{-1}}{\sigma}.$$

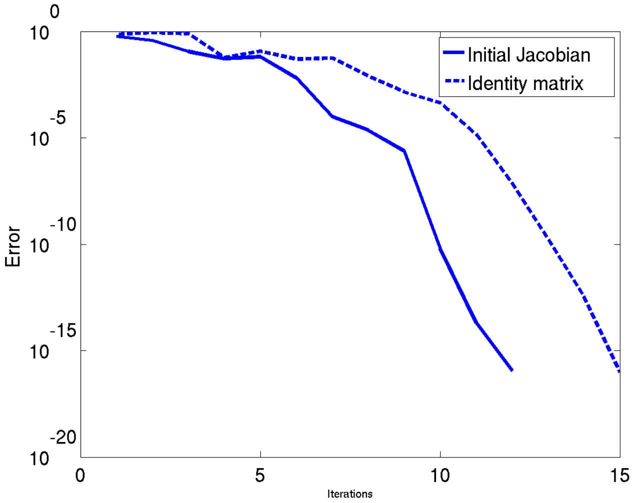
This formula is known as Sherman-Morrison-Woodbury formula.

Broyden's method

To use this method, one requires the initial guess for $\mathbf{B}^{(0)}$. Some possible choices are

- exact Jacobian at iteration 0
- identiity matrix, etc.

Example



10^{0}
10^{-5}
10^{-10}
10^{-15}
10^{-20}
Error
0
5
10
15
Iterations
Initial Jacobian
Identity matrix

Nonlinear system seen as optimisation problem

Optimisation

Note that the process of solving

$$\mathbf{F}(\mathbf{x}) = \mathbf{0}$$

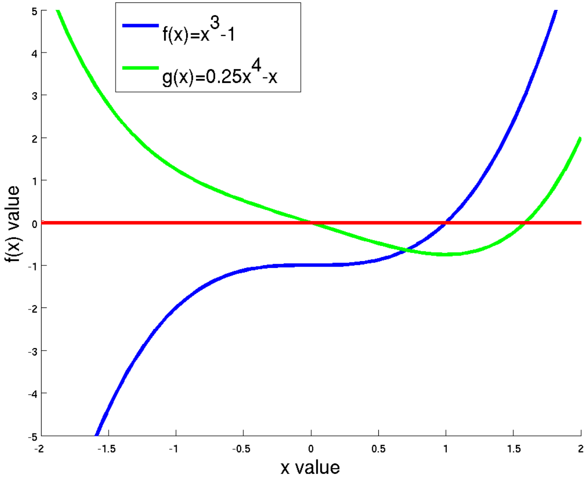
corresponds to the process of minimising the function

$$\mathbf{x} = \min_{\mathbf{x}} \; G(\mathbf{x})$$

such that

$$\mathbf{F}(\mathbf{x}) = G'(\mathbf{x})$$

Graphical interpretation



Optimisation

The minimum can be found by taking

$$G'(\mathbf{x}) = \mathbf{F}(\mathbf{x}) = \mathbf{0}, \quad \text{position of x where is minimum of G(x)}$$

and

$$G^{''}(x) = \mathbf{F}'(\mathbf{x}), \quad \text{being ¿0 one has convex function}$$

is positive definite. To find minimum one has to find the direction in which the function G decreases, performing the step in that direction and then repeating the process. The direction of decreasing (descent direction) satisfies the condition

$$G(\mathbf{x} + \alpha \mathbf{d}^{(k)}) < G(\mathbf{x}), \alpha \in [0, \delta).$$

for $\delta > 0$.

Optimisation

Note that the previous relation can be rewritten as

$$\lim_{\alpha\to 0}(G(\mathbf{x}+\alpha\mathbf{d}^{(k)})-G(\mathbf{x}))/\alpha<0 \longrightarrow G'(\mathbf{x})^\top\mathbf{d}^{(k)}=F(\mathbf{x})^\top\mathbf{d}^{(k)}<0.$$

which leads to a sufficient condition for $\mathbf{d}^{(k)}$ to be a direction of descent

$$F(\mathbf{x}^{(k)})^\top\mathbf{d}^{(k)}<0.$$

The directions taken in Newton's scheme,

$$\mathbf{d}^{(k)}=-F'(\mathbf{x}^{(k)})^{-1}F(\mathbf{x}^{(k)}),$$

are directions of descent if $F'(\mathbf{x}^{(k)})$ is positive definite because then $F(\mathbf{x}^{(k)})^\top\mathbf{d}^{(k)}=-F(\mathbf{x}^{(k)})^\top F'(\mathbf{x}^{(k)})^{-1}F(\mathbf{x}^{(k)})<0$

Line search

Iterative solvers for nonlinear equations change the current guess $\mathbf{x}^{(k)}$ in the k-th iteration by walking into a direction $\mathbf{d}^{(k)}$,

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \mathbf{d}^{(k)}.$$

Their robustness can considerably be enhanced by scaling the step-size taken in the k-th iteration by a factor α_k and then taking the step

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}.$$

Given $\mathbf{d}^{(k)}$, a value for α_k is found iteratively by searching along the line $\mathbf{x}^{(k)} + \alpha \mathbf{d}^{(k)}, \alpha > 0$, hence the name line-searches.

Line search

There is no fail-proof strategy for choosing α_k leading to a convergent scheme in all cases.
Here, we will only mention the so-called Curry-Principle which is based on choosing α_k as a minimiser of the function $\phi(\alpha) = G(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)})$. For every minimiser of ϕ it holds that $\phi' = F(\mathbf{x}^{(k)} + \alpha\mathbf{d}^{(k)})^\top \mathbf{d}^{(k)} = 0$, and hence the following algorithm results:
Algorithm:
Given a direction of descent $\mathbf{d}^{(k)}$, set the next iterate as

$$\mathbf{x}^{(k+1)} = \mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)}$$

where $\alpha_k > 0$ is chosen as the smallest positive value such that

$$F(\mathbf{x}^{(k)} + \alpha_k \mathbf{d}^{(k)})^\top \mathbf{d}^{(k)} = 0.$$

Line search

Note that the Newton methods are locally convergent. *an iterative method is called locally convergent if the successive approximations produced by the method are guaranteed to converge to a solution when the initial approximation is already close enough to the solution*. To make methods globally convergent, the idea of linear search has been introduced.

Literature

- Peter Deuflhard, Newton Methods for Nonlinear Problems
- C.T. Kelley, Iterative Methods for Optimization
- Todd Young and Martin J. Mohlenkamp, Introduction to Numerical Methods and Matlab programming

Wishing you to find the direction of the steepest decent in the following year :)