Rate of Convergence

Rate of Convergence

- We study different numerical methods to find a root of a equation?
- Because different method converge to the root with different speed.
- Rate of Convergence measures how fast of a sequence converges

Order of a Root

- **Definition (Order of a Root)** Assume that f(x) and its derivatives are defined and continuous on an interval about x = p. We say that f(x) = 0 has a root of order m at x = p if and only if

$$f(p) = 0, \quad f'(p) = 0, \quad f''(p) = 0, \quad f'''(p) = 0, \qquad f^{(m-1)}(p) = 0, \quad f^{(m)}(p) \neq 0$$

.

$$f(x) = (x - p)^m h(x), \quad h(p) \neq 0$$

- A root of order m = 1 is often called a **simple root**,
- and if m > 1 it is called a **multiple root**. A root of order m = 2 is sometimes called a **double root**, and so on.

Rate of Convergence (cont'd)

Definition: Let the sequence $\{r_n\}$ converge to r. Denote the difference between r_n and r by e_n; i.e. $e_n = r_n - r$. If there exists a positive number $p \geq 1$ and a constant $c \neq 0$ such that

$$\lim_{n\to\infty}\frac{|r_{n+1}-r|}{|r_n-r|^p}=\lim_{n\to\infty}\frac{|e_{n+1}|}{|e_n|^p}=c$$

then **p** is called the **order of convergence** of the sequence. The constant c is called the asymptotic error constant.

- If p is large, then the sequence $\{r_n\}$ converges rapidly to r.
- If $p = 1$ and $c<1$, then the convergence is said to be **linear**, and c is called the rate of convergence;
- If $p = 2$, then it is **quadratic**.

Example of convergence

$$\{\frac{11}{2},\frac{21}{4},\frac{41}{8},\ldots,5+\frac{1}{2^n},\ldots\} \text{ and } \{\frac{11}{2},\frac{41}{8},\frac{641}{128},\ldots,5+\frac{1}{2^{2^n-1}},\ldots\}$$

Both sequences converge to 5. However it seems that the second sequence converges faster to 5 than the first one.

$$e_n = r_n - r = \frac{1}{2^n}, \text{ and}$$

$$\lim_{n\to\infty}\frac{|e_{n+1}|}{|e_n|} = \lim_{n\to\infty}\frac{2^n}{2^{n+1}} = \frac{1}{2}$$

First sequence converges linearly ($p = 1$) to 5.

$$e_n = r_n - r = \frac{1}{2^{2^n-1}}, \text{ and}$$

$$\lim_{n\to\infty}\frac{|e_{n+1}|}{|e_n|} = \lim_{n\to\infty}\frac{1/2^{2^{n+1}-1}}{1/(2^{2^n-1})^2} = \frac{1}{2}$$

Second sequence converges quadratically ($p = 2$) to 5.

Rate of Convergence for the Bracket Methods

- The rate of convergence of
  - False position, $p = 1$, linear convergence
  - Netwon's method, $p = 2$, quadratic convergence
  - Secant method, $p = 1.618$.
  - Fixed point iteration, $p = 1$, linear convergence

- The rate value of rate of convergence is just a theoretical index of convergence in general.

Convergence Rate for Newton-Raphson Iteration

- Assume that Newton-Raphson iteration produces a sequence $\{p_k\}_{k=0}^{\infty}$ that converges to the root p of the function **f(x).**
- If p is a simple root, then convergence is quadratic and $|E_{k+1}| \approx \frac{|f''(p)|}{2|f'(p)|}(|E_k|)^2$

for k sufficiently large.

Convergence Rate for **Fixed-Pt. Iteration**

If g is cont. diff. on an open interval of fixed point α with $\max_{x\in[a,b]} |g'(x)| = \gamma < 1$

Then: $\forall x_0 \in N(\alpha)$

1. The iteration $x_{n+1} = g(x_n)$ converges to a for any initial guess x_0
2. Error estimate

$$|\alpha - x_n| \le \frac{\gamma^n}{1-\gamma} |x_1 - x_0|$$

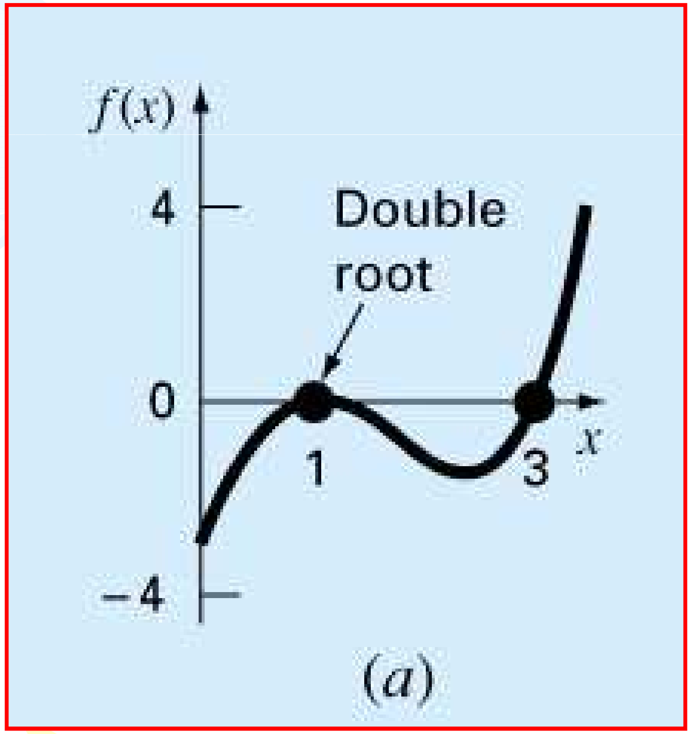
1. rate of convergence

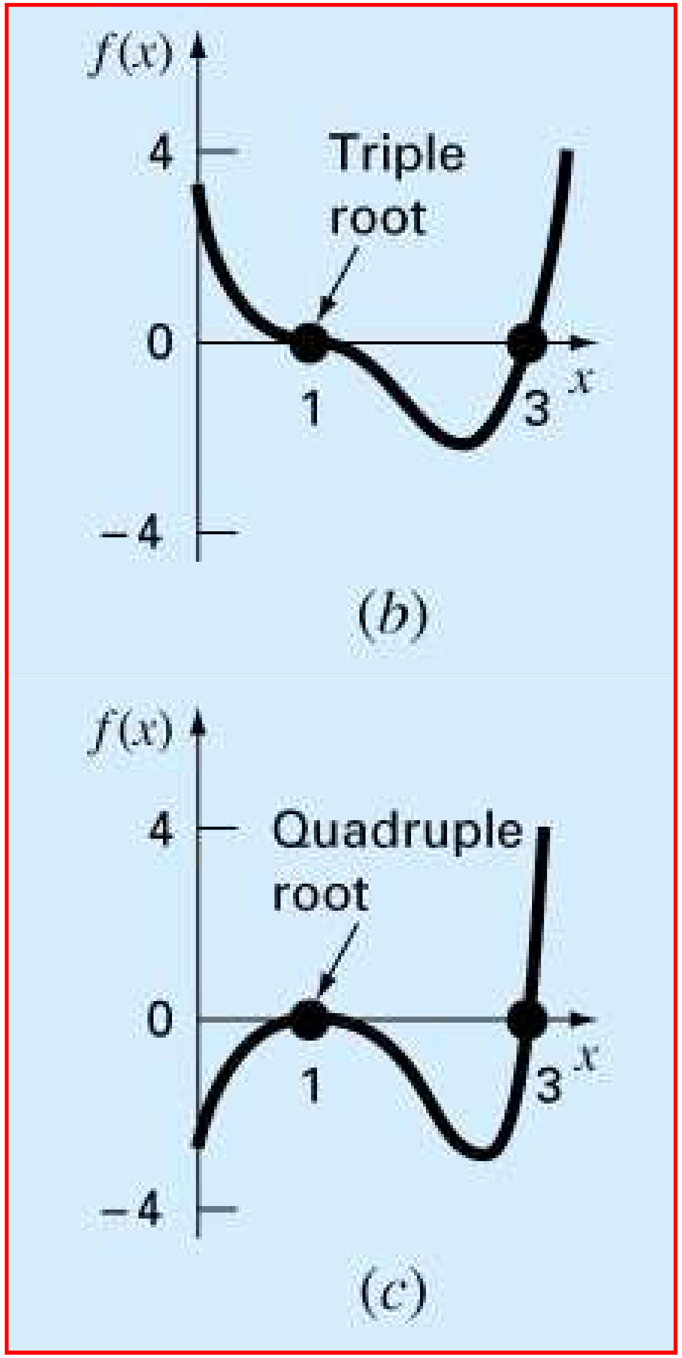
$$\lim_{n\to\infty} \frac{\alpha - x_{n+1}}{\alpha - x_n} = g'(\alpha)$$

Multiple Roots

- **Problems with multiple roots**
- **The function does not change sign at even multiple roots (i.e., m = 2, 4, 6, …)**
- **$f'(x)$ goes to zero - need to put a zero check for $f(x)$ in program**
- **slower convergence (linear instead of quadratic) of Newton-Raphson and secant methods for multiple roots**

Examples of Multiple Roots





Multiple Roots

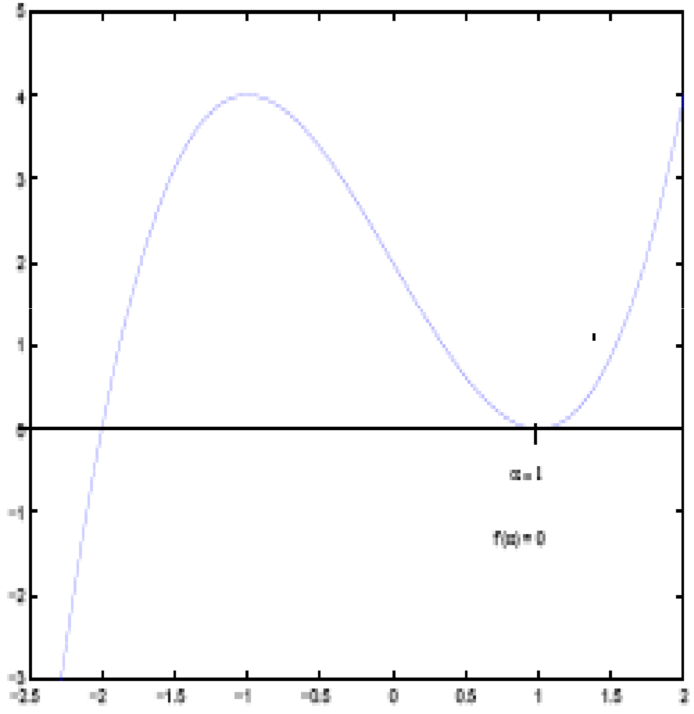
- A function $f(x)$ is said to have a root x^* of multiplicity $p > 1$, if $f(x) = (x - x^*)^p g(x)$ for some function $g(x)$ and $g(x^*) \neq 0$.
- If p is an integer, then we have

$$f(x^*) = f'(x^*) = f''(x^*) = \ldots f^{[p-1]}(x^*) = f^{[p]}(x^*) = 0$$

- For example, $f(x) = (x-1)(x-1)(x-2)(x-3)$ has a double root (*i.e.*, $p = 2$) and $f(x)=(x-1)^2 g(x)$ and $g(x) = (x-2)(x-3)$.
- Newton's method and the secant method converge only *linearly* for multiple roots.

Multiple roots:

Consider $f(x) = x^3 - 3x + 2 = (x-1)^2(x+2)$



Compare convergence for $f = (x-1)(x+2) = 0$ with $f = (x-1)^2(x+2) = 0$.

n	$f = (x-1)(x+2)$	$f = (x-1)^2(x+2)$
1	2.0	2.0
2	1.199	1.5555
3	1.0117647	1.29790666
4	1.00004377	1.15539019
5	1.000	1.07956221
		(slow convergence)

To accelerate convergence to a multiple root adapt Newton-Raphson to,

$$x_{n+1} = x_n - m \cdot \frac{f(x_n)}{f'(x_n)}$$

Example: Double root $m = 2$

$$\begin{aligned} f(x_n) &= \cancel{f(\alpha)} + (x_n - \alpha)\cancel{f'(\alpha)} + \tfrac{1}{2}(x_n - \alpha)^2 f''(\alpha) + \dots \\ f'(x_n) &= \cancel{f'(\alpha)} + (x_n - \alpha) f''(\alpha) + \dots \end{aligned}$$

$$\Rightarrow \frac{f(x_n)}{f'(x_n)} \approx \tfrac{1}{2}(x_n - \alpha) \qquad \Rightarrow \alpha \approx x_n - 2\frac{f(x_n)}{f'(x_n)}.$$

Convergence Rate for Newton-Raphson Iteration

- If p is a multiple root of order m, then convergence is linear for k sufficiently large

$$|E_{k+1}| \approx \frac{m-1}{m} |E_k|$$

Slower convergence the higher the order of the root

For multiple root, convergence rate of Newton's method is only linear, with constant $C = 1 - (1/m)$, where m is multiplicity

k	$f(x) = x^2 - 1$	$f(x) = x^2 - 2x + 1$
0	2.0	2.0
1	1.25	1.5
2	1.025	1.25
3	1.0003	1.125
4	1.00000005	1.0625
5	1.0	1.03125

Modified Newton-Raphson Method

- **When the multiplicity of the root is known**

$$x_{i+1} = x_i - m\frac{f(x_i)}{f'(x_i)}$$

- **Double root : m = 2**
- **Triple root : m = 3**
- **Simple, but need to know the multiplicity m**
- **Maintain quadratic convergence**

Multiple Roots:

- Two modifications to Newton's method can still maintain quadratic convergence.
- If x^* is a multiple root of multiplicity p, then use the following instead of the original:

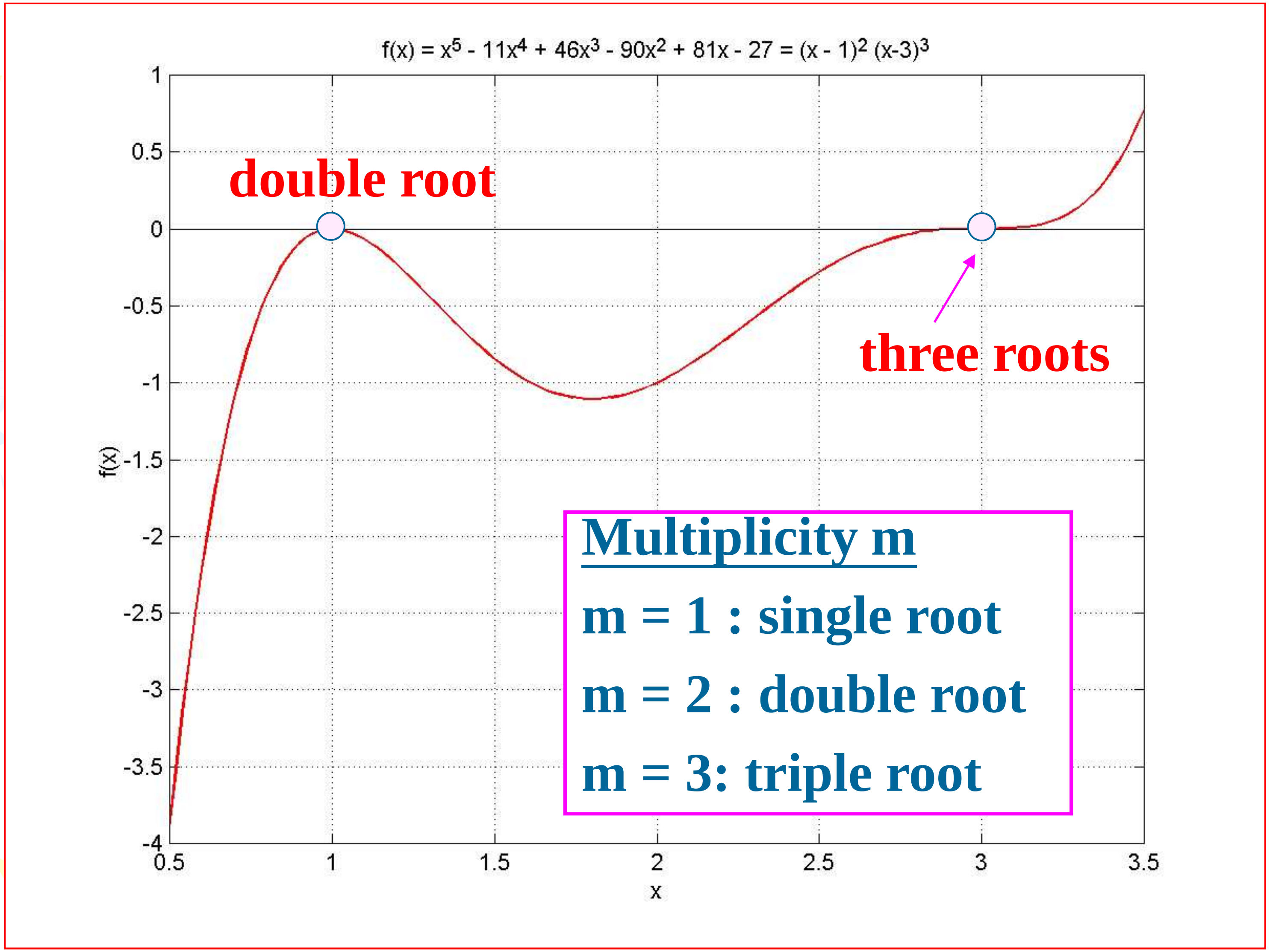
$$x_{k+1} = x_k - p \times \frac{f(x_k)}{f'(x_k)}$$

- Or, let $g(x) = f(x)/f'(x)$ and use the following:

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

Multiple Root with Multiplicity m

$$f(x) = x^5 - 11x^4 + 46x^3 - 90x^2 + 81x - 27$$



Original Newton's method
m = 1

```
» multiple1('multi_func','multi_dfunc');
enter multiplicity of the root = 1
enter initial guess x1 = 1.3
allowable tolerance tol = 1.e-6
maximum number of iterations max = 100
Newton method has converged
```

step	x	y
1	1.30000000000000	-0.44217000000004
2	1.09600000000000	-0.06361262220902 1
3	1.04407272727272	-0.01453442847741 8
4	1.02126549372889	-0.00350359197248 2
5	1.01045853297516	-0.00086139138942 8
6	1.00518770530932	-0.00021362727675 0
7	1.00258369467652	-0.00005319712394 7
8	1.00128933592285	-0.00001327339304 4
9	1.00064404356011	-0.00000331513217 6
10	1.00032186610620	-0.00000082838226 2
11	1.00016089418619	-0.00000020704553 1
12	1.00008043738571	-0.00000005175515 1
13	1.00004021625682	-0.00000001293800 3
14	1.00002010751461	-0.00000000323440 5
15	1.00001005358967	-0.00000000080860 5
16	1.00000502663502	-0.00000000020213 5
17	1.00000251330500	-0.00000000005052 7
18	1.00000125681753	-0.00000000001262 6
19	1.00000062892307	-0.00000000000316 2

Modified Newton's Method
m = 2

```
» multiple1('multi_func','multi_dfunc');
enter multiplicity of the root = 2
enter initial guess x1 = 1.3
allowable tolerance tol = 1.e-6
maximum number of iterations max = 100
Newton method has converged
```

step	x	y
1	1.30000000000000	-0.44217000000004
2	0.89199999999999	-0.10925953065677 9
3	0.99229251101321	-0.00048075868939 2
4	0.99995587111371	-0.00000001557990 0
5	0.99999999853944	-0.00000000000000 7
6	1.00000060664549	-0.00000000000293 5

Double root : m = 2

$$f(x) = x^5 - 11x^4 + 46x^3 - 90x^2 + 81x - 27 = 0$$

Modified Newton's Method with u = f / f′

- A more general modified Newton-Raphson method for multiple roots

$$u = \frac{f(x)}{f'(x)}$$

$$\Rightarrow x_{i+1} = x_i - \frac{u(x_i)}{u'(x_i)}$$

$$f(x) = (x - x^*)^m$$

$$f'(x) = m(x - x^*)^{m-1}$$

$$u(x) = \frac{f(x)}{f'(x)} = \frac{x - x^*}{m}$$

- $u(x)$ contains only single roots even though $f(x)$ may have multiple roots

$$x_{i+1} = x_i - \frac{f(x_i) f'(x_i)}{[f'(x_i)]^2 - f(x_i) f''(x_i)}$$

Modified Newton's method

$$f(x) = x^5 - 11x^4 + 46x^3 - 90x^2 + 81x - 27 = 0$$

```
>> [x,f] = multiple2('multi_func','multi_dfunc','multi_ddfunc');
enter initial guess: xguess = 10
allowable tolerance: es = 1.e-6
maximum number of iterations: maxit = 100
Newton method has converged
```

Triple root at x = 3

step	x	f	df/dx	d2f/dx2
1	10.00000000000000	27783.00000000000000	18081.00000000000000	9380.00000000000000
2	2.42521994134897	-0.385717068699165	1.471933198691602	-1.734685930313219
3	2.80435435817775	-0.024381150764611	0.346832001230098	-3.007964394244482
4	2.98444590681717	-0.000014818785758	0.002843242444783	-0.361760865258020
5	2.99991809093254	-0.000000000002188	0.000000080500286	-0.001965495593481
6	2.99999894615774	-0.000000000000028	0.000000000013529	-0.000025292161013
7	2.99999841112323	0.000000000000000	0.000000000030582	-0.000038132921304

- **Original Newton-Raphson method required 135 iterations**
- **Modified Newton's method converged in only 7 iterations**

Comparison of Methods

Method	Initial guesses	Convergence rate	Stability	
Bisection	2	Slow	Always	
False position	2	Medium	Always	
Fixed-pointed iteration	1	Slow	Possibly divergent	
Newton-Raphson	1	Fast	Possibly divergent	Evaluate f'(x)
Modified Newton-Raphson	1	multiple roots Slow	Possibly divergent	
Secant	2	Medium to fast	Possibly divergent	Initial guesses don't have to bracket root
Modified secant	2	Fast	Possibly divergent	