

Chap 1: Solving Nonlinear Eq.

- **Interval Halving (Bisection)**
- **Linear Interpolation Methods**
- **Newton's Method**
- **Muller's Method**
- **Fixed-Point Iteration**
- **Order of convergence**
- **Multiple Roots**
- **Nonlinear systems**

Introduction -1

- **Problem: Solve $f(\mathbf{x})=0$**
 - **A system of nonlinear equations**

$$f_1(x_1, x_2, \dots, x_n) = 0$$

$$\vdots$$

$$f_n(x_1, x_2, \dots, x_n) = 0$$

- **Nonlinear function can be**
 - **Polynomials**

$$f(x, y) = \sum_{i+j=n} a_{ij} x^i y^j$$

- **Functions involve transcendental functions**
 - **sin, cos, exponentials**

Introduction -2

- **Closed form solutions**
 - **Only available for polynomials of degree less than 5.**
 - Quadratic formula for degree 2
 - Complicated for degrees 3 and 4
 - **For nonlinear polynomial system**
 - Algebraic methods such as Grobner basis can be applied to eliminate variables and reduce the system to a triangular form

Introduction -3

- **Numerical solutions**
 - **Via iteration procedure**
 - **Starting point**
 - **Compute iterates**
 - **Check for termination**
 - **Need to consider**
 - **Convergence**
 - **Rate of convergence**
 - **Stability**
 - **Early error magnified or not?**

Bisection method ⁻¹

- **An ancient but effective method for solving $f(x)=0$**
- **Starting with $[x_1, x_2]$: an interval that bracket a root, with $f(x_1)*f(x_2) < 0$**
 - **f changes signs at x_1 and x_2**
 - **If f is continuous, there must be at least one root between x_1 and x_2**
- **Divides the interval in half, finds in which half the root must lie, and repeat**

Bisection method ⁻²

- **Pseudo code**

Repeat

$$x_3 = (x_1 + x_2) / 2$$

$$\text{if } f(x_3)f(x_1) < 0$$

$$x_2 = x_3$$

$$\text{else } x_1 = x_3$$

Until $(|x_1 - x_2| / 2 < \text{tolerance})$

- **The final value of x_3 approximates the root, with error no more than $|x_1 - x_2| / 2$**
- **The method may produce a false root if $f(x)$ is discontinuous on $[x_1, x_2]$**

Bisection Method ⁻³

- **Advantages**

- It is guaranteed to work if $f(x)$ is continuous in the initial interval, and if the interval actually brackets a root
- The number of iterations to achieve a specified accuracy is known in advance, since the interval $[a, b]$ is halved each time, so

$$\text{error after } n \text{ iterations} < \left| \frac{b-a}{2^n} \right|$$

$$\left| \frac{b-a}{2^n} \right| < \text{Tolerance} \Rightarrow \text{a bound on } n$$

- **Disadvantages**

- **Slow to converge**

- **Why? No information about $f(x)$ is used**

Bisection Method -4

- An example, $a=0$, $b=1$, tolerance= $1E-4$

$$f(x) = 3x + \sin(x) - e^x = 0$$

Table 1.1 The bisection method for $f(x) = 3x + \sin(x) - e^x = 0$, starting from $x_1 = 0$, $x_2 = 1$, using a tolerance value of $1E-4$

Iteration	X_1	X_2	X_3	$F(X_3)$	Maximum error	Actual error
1	0.00000	1.00000	0.50000	0.33070	0.50000	0.13958
2	0.00000	0.50000	0.25000	-0.28662	0.25000	-0.11042
3	0.25000	0.50000	0.37500	0.03628	0.12500	0.01458
4	0.25000	0.37500	0.31250	-0.12190	0.06250	-0.04792
5	0.31250	0.37500	0.34375	-0.04196	0.03125	-0.01667
6	0.34375	0.37500	0.35938	-0.00262	0.01563	-0.00105
7	0.35938	0.37500	0.36719	0.01689	0.00781	0.00677
8	0.35938	0.36719	0.36328	0.00715	0.00391	0.00286
9	0.35938	0.36328	0.36133	0.00227	0.00195	0.00091
10	0.35938	0.36133	0.36035	-0.00018	0.00098	-0.00007
11	0.36035	0.36133	0.36084	0.00105	0.00049	0.00042
12	0.36035	0.36084	0.36060	0.00044	0.00024	0.00017
13	0.36035	0.36060	0.36047	0.00013	0.00012	0.00005

Bisection Method ⁻⁵

- **Some comments**
 - **Recommended: used for finding an approximate value for the root, and the value is refined by more efficient methods**
 - **Most other root-finding methods require a starting value near to a root – lacking this, they may fail completely**
 - **Good practice:**
 - **Graph the function first**
 - **Search for interval that with sign change at ends**
 - **Apply Bisection to get an initial starting value**
 - **Apply better methods**
 - **Not applicable to the case of multiple roots**
 - **Find the root by working on with $f'(x)$, which will be zero at a multiple root.**

Linear Interpolation Methods

- Approximate the function by a straight line
 - Interpolated line
 - The secant method
 - Two x -values nearest to the root
 - False position method
 - Similar to the Bisection method
 - Two points need to bracket the root
 - Tangent line
 - Newton method

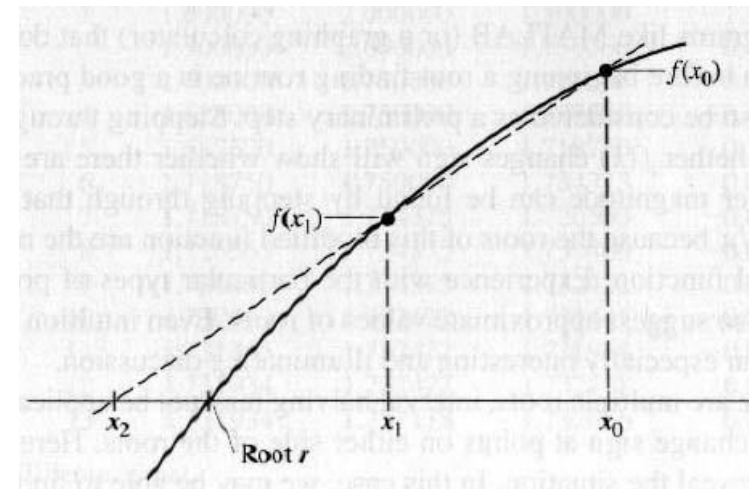


Figure 1.1

Secant method ⁻¹

- **Secant method**
 - **Find two points on the curve near to the root**
 - Draw a graph or apply a few iteration of bisection.
 - Two point may both be on one side of the root, or on opposite sides
 - **Find the line through these two points and find the point it intersects the x-axis**
 - **Repeat the process until the intersection point is close enough to the root.**

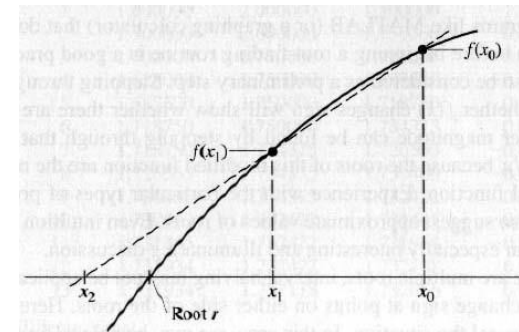


Figure 1.1

Secant method -2

From the similar triangles, we have

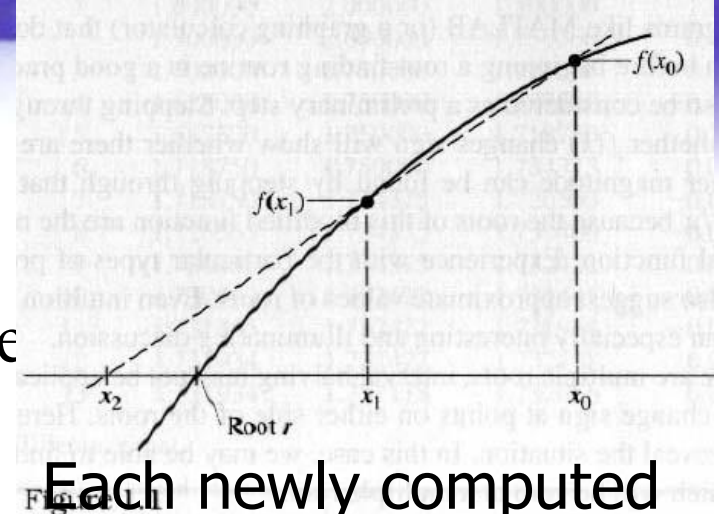
$$\frac{(x_1 - x_2)}{f(x_1)} = \frac{(x_0 - x_1)}{f(x_0) - f(x_1)}$$

Solve for x_2 :

$$x_2 = x_1 - f(x_1) \frac{(x_0 - x_1)}{f(x_0) - f(x_1)}$$

Repeat the iteration, we have

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



Each newly computed value should be nearer to the root.

After second iteration:

Always using the last two computed values.

After first iteration:

There aren't two last computed values.

Swap x_0 and x_1 if

Necessary, such that x_1 is closer to the root.

The Secant Method -3

- **Start with x_0 and x_1 near the root**

if $|f(x_0)| < |f(x_1)|$ then swap x_0 and x_1

Repeat

$$x_2 = x_1 - \frac{f(x_1)(x_0 - x_1)}{f(x_0) - f(x_1)}$$

$$x_0 = x_1$$

$$x_1 = x_2$$

Until $|f(x_2)| < \text{tolerance value}$

Note :

If $f(x)$ is not continuous, the method may fail.

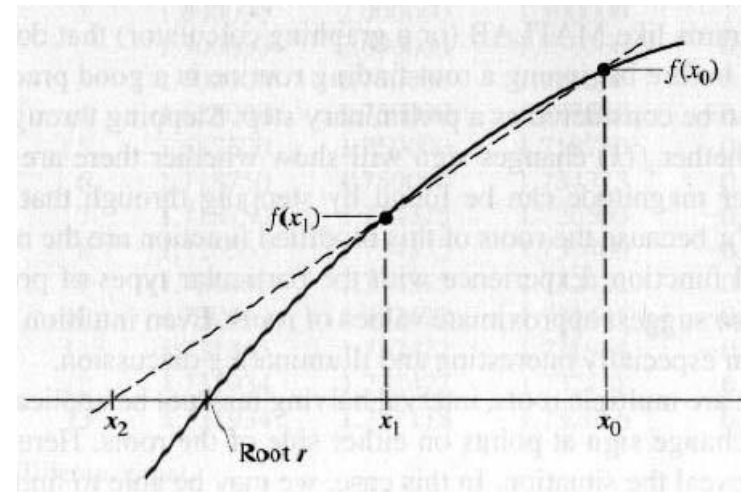


Figure 1.1

The Secant Method -4

An Example

Table 1.2 Secant method on $f(x) = 3x + \sin(x) - e^x$

Iteration	x_0	x_1	x_2	$f(x_2)$
1	1	0	0.4709896	0.2651588
2	0	0.4709896	0.3722771	2.953367E-02
3	0.4709896	0.3722771	0.3599043	-1.294787E-03
4	0.3722771	0.3599043	0.3604239	5.552969E-06
5	0.3599043	0.3604239	0.3604217	3.554221E-08

At $x = .3604217$, tolerance of .0000001 met!

- **Fewer iterations are required compared to bisection: 5 iterations**

The Secant Method ⁻⁵

Problems

- If the function is far from linear near the root, the successive iterates can fly off to points far from the root

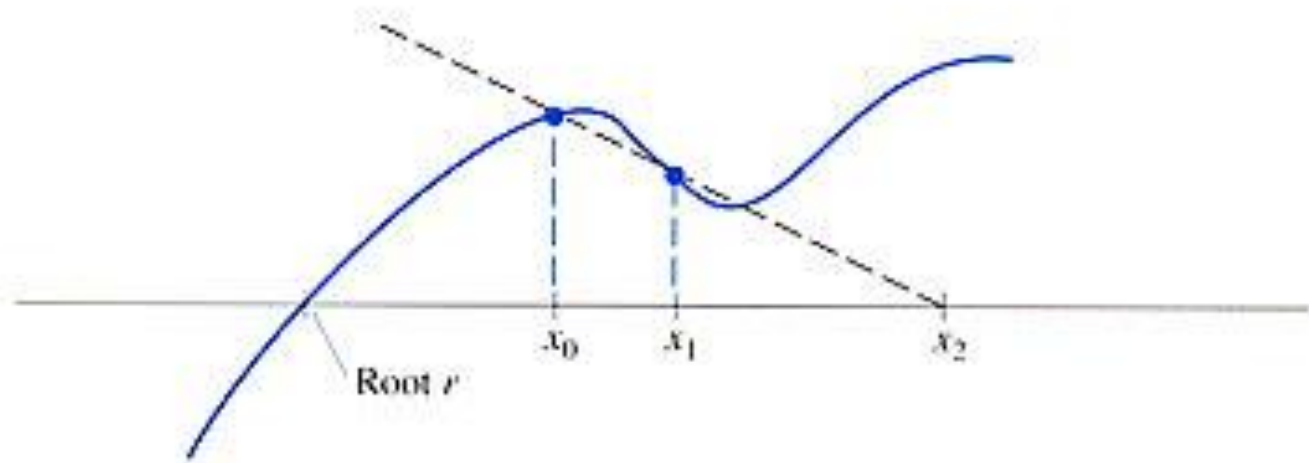


Figure 1.2
A pathological case for the secant method

False Position Method -1

- **Avoid problems of secant method**
 - **Ensure that the root is bracketed between two starting values and remain between the successive pairs.**
 - **Similar to bisection method**
 - **x_0 and x_1 bracket a root**

Differences:

- **Next iterate is taken at the intersection of a line between the pair of x -values and the x -axis rather than the midpoint**
- **Gives faster convergence than does bisection, but at expense of a more complicated algorithm**

False Position Method -2

x_0 and x_1 bracket a root

Repeat

$$x_2 = x_1 - f(x_1) \frac{(x_0 - x_1)}{f(x_0) - f(x_1)}$$

if $f(x_2)$ opposite sign to $f(x_0)$

$$x_1 = x_2$$

$$\text{else } x_0 = x_2$$

Until $|f(x_2)| < \text{tolerance value}$

Note :

if $f(x)$ is not continuous, the method may fail.

A Comparison

- **Speed of convergence**
 - **Secant (best), false position, then bisection**

Table 1.3 Comparison of methods, $f(x) = 3x + \sin(x) - e^x = 0$, $x_0 = 0$, $x_1 = 1$

Iteration	Interval halving		False position		Secant method	
	x	$f(x)$	x	$f(x)$	x	$f(x)$
1	0.5	0.330704	0.470990	0.265160	0.470990	0.265160
2	0.25	-0.286621	0.372277	0.029533	0.372277	0.029533
3	0.375	0.036281	0.361598	2.94×10^{-3}	0.359904	-1.29×10^{-3}
4	0.3125	-0.121899	0.360538	2.90×10^{-4}	0.360424	5.55×10^{-6}
5	0.34375	-0.041956	0.360433	2.93×10^{-5}	0.360422	3.55×10^{-7}
Error after 5 iterations	0.01667		-1.17×10^{-5}		$< -1 \times 10^{-7}$	

(Exact value of root is 0.360421703.)

Newton's Method ⁻¹

- Takes a single initial x_0 (not too far from a root), and the intersection of the tangent line and x-axis as the next
- Is the most widely used method
 - More rapidly convergent than bisection, secant and false position.

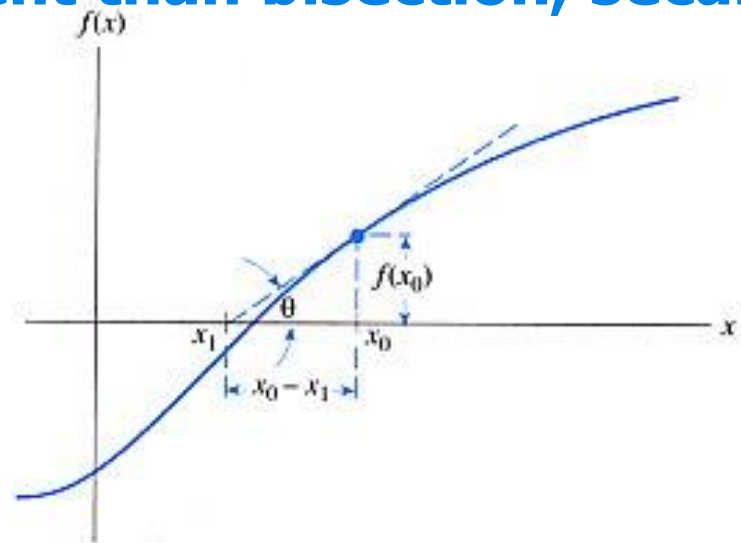


Figure 1.3

Newton's Method -2

- Iteration

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

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

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

\vdots

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

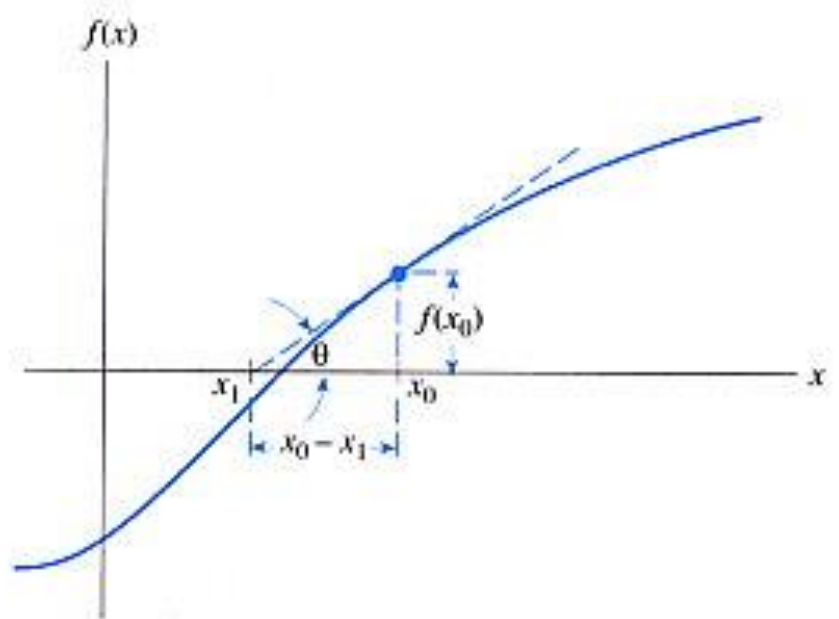


Figure 1.3

Newton's Method -3

**Given a x_0
reasonably
close to the
root.**

Compute $f(x_0)$ and $f'(x_0)$
if $(f(x_0) \neq 0)$ and $(f'(x_0) \neq 0)$

Repeat

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

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

Until $(|x_0 - x_1| < \text{tolerance1})$ or
 $(|f(x_0)| < \text{tolerance2})$

Note :

The method may converge to a root different from the expected one or diverge if the starting value is not close enough to the root.

Newton's Method ⁻⁴

- **Rate of convergence**
 - **Quadratically convergent**
 - Error of each step approaches a constant K times the square of the error of the previous step
 - The net result of this is that the number of decimal places of accuracy nearly doubles at each iteration
 - Fewer steps than previous methods
 - But, at the cost of two function evaluations at each step
 - » Previous methods need only one at each step (after the first step)
 - Faster than any of the methods discussed so far

Newton's Method -5 - Example

$$f(x) = 3x + \sin x - e^x = 0$$

Starting with $x_0 = 0.0$

$$x_1 = 0.33333$$

$$x_2 = 0.36017$$

$$x_3 = 0.3604217$$

$$\text{True } x = 0.360421703$$

Note :

After 3 iterations , the solution is correct to 7 digits.

The error after an iteration is about 1/3 of the square of the previous error.

Newton's Method -6

- **In some cases**
 - **May converge to a different root, diverge, or oscillating**
 - **Converge**
 - **Wandering: jump around, and then jump to a point near to the root, and then converge rapidly**
 - **In endless loop, i.e., cycling**
 - **Starting point near to an inflection point or a turning point (local minimum/maximum)**
 - **Overshooting**
 - **Fly off to infinity when ever reach the minimum or maximum of the curve**
 - **X-intercept x_1 is far from both x_0 and the desired root, may converge to different root**

Newton's Method -7

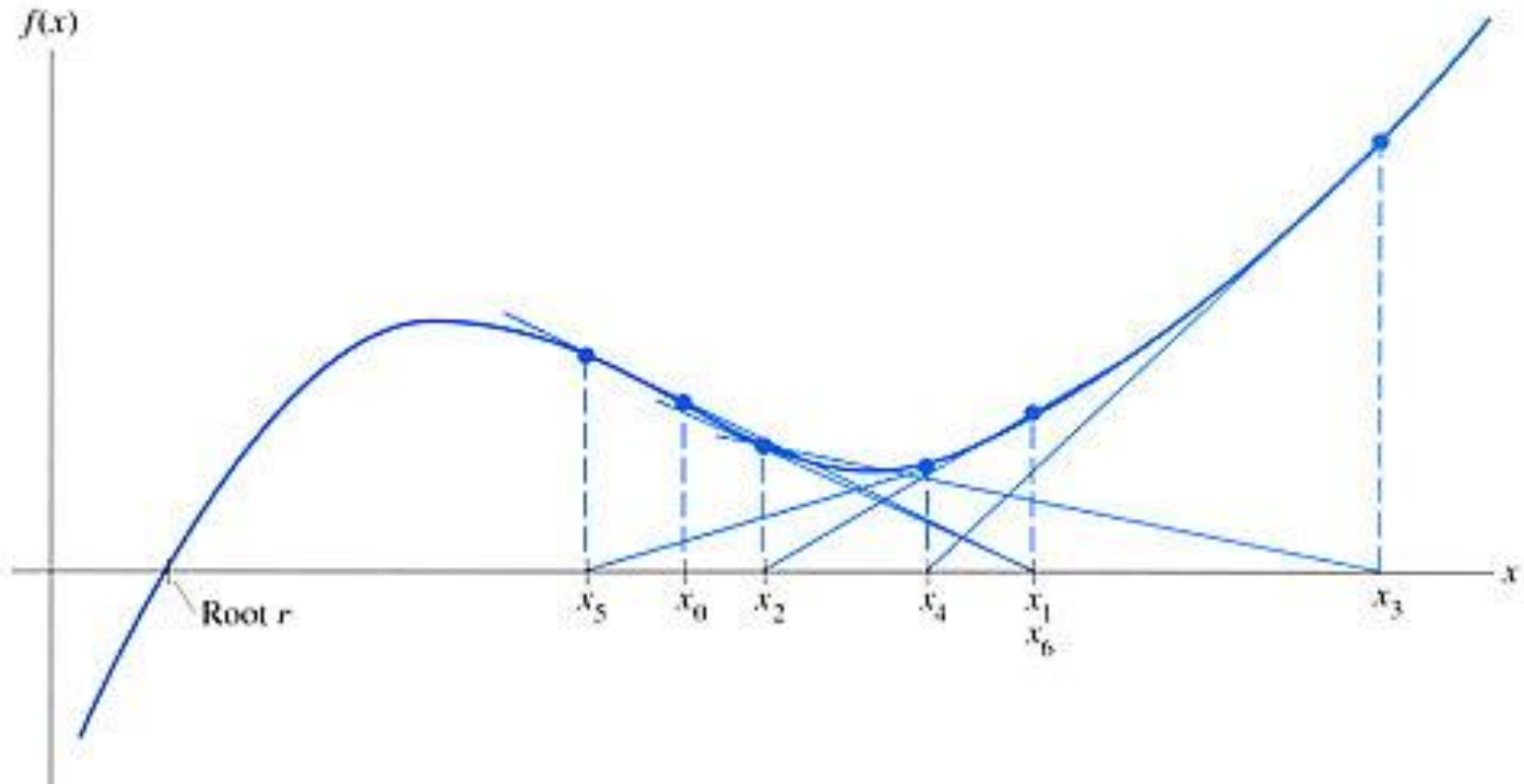


Figure 1.4

Starting point near to a turning point

Newton's Method ⁻⁸ vs. interpolated methods

Interpolated methods :

$$\begin{aligned}x_{n+1} &= x_n - f(x_n) \frac{(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})} \\&= x_n - \frac{f(x_n)}{\frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}}\end{aligned}$$

Difference quotient approximates the derivative!! So closely resemble to Newton method!!

Newton's Method -9

Complex roots

- **Newton's method works with complex roots if a complex starting value is given.**

$$f(x) = x^3 + 2x^2 - x + 5$$

$f(x) = 0$ has a real root at $x = -3$

It has two complex roots because x - axis is not crossed again.

Start with $x_0 = 1 + i$, we have

$$x_1 = 0.486238 + 1.04587i, x_2 = 0.448139 + 1.23665i$$

$$x_3 = 0.462720 + 1.22242i, x_4 = 0.462925 + 1.22253i$$

$$x_5 = 0.462925 + 1.22253i$$

Since agree to 6 significant t digits, we have an estimate good to at least 6 significant t digits.

Start with $x_0 = 1 - i$, we have the conjugate

$$0.462925 - 1.22253i$$

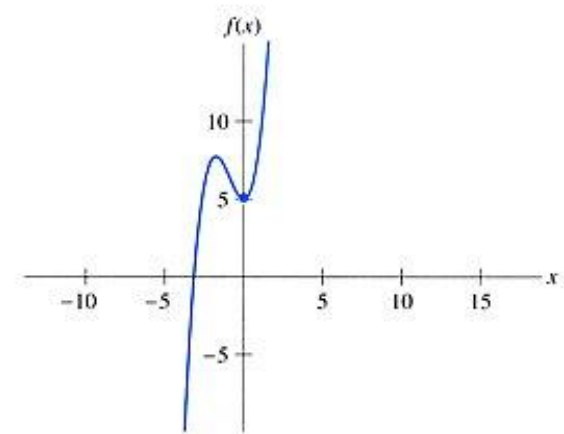


Figure 1.5
Plot of $f(x) = x^3 + 2x^2 - x + 5$

Newton's Method ⁻¹⁰ for polynomials

- **Polynomials have nice behavior:**
 - They are everywhere continuous
 - They are smooth
 - Their derivatives are also continuous and smooth.
 - They are readily evaluated.
 - Number of roots can be predicted
 - Evaluation requires only $+$, $-$, $*$.

Newton's Method ⁻¹¹ for polynomials

- For previous methods, expect Newton method, there is nothing new (gain from the properties of polynomials)
- Polynomial evaluation of $f(x)$ and $f'(x)$ can be done by the use of **synthetic division**, which is based on the well known **remainder theorem**

Newton's Method ⁻¹² for polynomials

- **Evaluating a polynomial by nested form**

- can be done by (or equivalent to) **synthetic division**
- Evaluate $f(x)$ at $x=2$

$$\begin{aligned} f(x) &= 2x^3 + x^2 - 3x - 3 \\ &= ((2x + 1)x - 3)x - 3 \end{aligned}$$

\Rightarrow Process of synthetic division

Synthetic division for $x = 2$:

$$\begin{array}{r|rrrr} x=2 & 2 & 1 & -3 & -3 \\ & & 4 & 10 & 14 \\ \hline & 2 & 5 & 7 & 11 \end{array} \leftarrow \text{Remainder} = f(2)$$

Note : Nested form evaluation :

$$\begin{aligned} &((2x + 1)x - 3)x - 3 \\ \text{with } x = 2 : &((2 \times 2 + 1) \times 2 - 3) \times 2 - 3 \rightarrow 11 \end{aligned}$$

Re min der Theorem:

$$\frac{2x^3 + x^2 - 3x - 3}{x - 2} = 2x^2 + 5x + 7 + \frac{11}{x - 2}$$

That is,

$$2x^3 + x^2 - 3x - 3 = (x - 2)(2x^2 + 5x + 7) + 11$$

Newton's Method ⁻¹³ for polynomials

- Example on synthetic division

Use the root associated with the divisor.
 $x + 3 = 0$
 $x = -3$

$(2x^3 - 5x^2 - x + 3) \div (x + 3)$
MathBits.com

list the coefficients only

-3	2	-5	-1	3
multiply	↓			
		$+(-6)$	$+33$	$+(-96)$
	2	-11	32	-93
				remainder

Solution: $2x^2 - 11x + 32 + \frac{-93}{x+3}$

Newton's Method ⁻¹⁴ for polynomials

- If the reduced (quotient) polynomial is divided by $x-2$ again, the remainder is the value of the derivative at $x=2$

$$f(x) = (x-2)(2x^2 + 5x + 7) + 11$$

$$f'(x) = (2x^2 + 5x + 7) + (x-2)(4x + 5)$$

$f'(2)$ is the value of $2x^2 + 5x + 7$ at $x = 2$

$$\begin{array}{r} x=2 \quad 2 \quad 5 \quad 7 \\ \quad \quad \quad 4 \quad 18 \\ \hline \quad \quad 2 \quad 9 \quad 25 \end{array} \leftarrow f'(2)$$

$$f'(2) = 25$$

$$x_1 = 2 - \frac{11}{25} = 1.56$$

Newton's Method ⁻¹⁵ for polynomials

Synthetic division algorithm :

A way of obtaining $Q_{n-1}(x)$ and R .

$$P_n(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

Reminder Theorem :

$$\begin{aligned} P_n(x) &= (x - a)Q_{n-1}(x) + R \\ &= (x - a)(b_{n-1}x^{n-1} + \cdots + b_1x + b_0) + R \end{aligned}$$

Multiplying out and equating coefficients of like terms in x , we have

$$\left. \begin{aligned} a_n &= b_{n-1} \\ a_{n-1} &= b_{n-2} - ab_{n-1} \\ a_{n-2} &= b_{n-3} - ab_{n-2} \\ &\vdots \\ a_1 &= b_0 - ab_1 \\ a_0 &= R - ab_0 \end{aligned} \right\} \Rightarrow \left\{ \begin{aligned} b_{n-1} &= a_n \\ b_{n-2} &= a_{n-1} + ab_{n-1} \\ b_{n-3} &= a_{n-2} + ab_{n-2} \\ &\vdots \\ b_0 &= a_1 + ab_1 \\ R &= a_0 + ab_0 \end{aligned} \right.$$

b_i and R are in the form of synthetic division

Newton's Method -16 for polynomials

- If the **quotient polynomial** is divided by $x-a$ again, the remainder is the value of the derivative at $x=a$.
WHY?

$$P_n(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0$$

Dividing $P_n(x)$ by $x-a$, we have

$$\frac{P_n(x)}{x-a} = Q_{n-1}(x) + \frac{R}{x-a}$$

That is,

$$P_n(x) = (x-a)Q_{n-1}(x) + R$$

$$\text{So } P_n(a) = R$$

Differentiate $P_n(x)$, we get

$$P_n'(x) = (x-a)Q_{n-1}'(x) + (1)Q_{n-1}(x)$$

Letting $x = a$, we have

$$P_n'(a) = Q_{n-1}(a) = \text{remainder on dividing } Q_{n-1}(x) \text{ by } (x-a).$$

Comparisons

Newton vs. secant

For simple root (i.e., $f'(r) \neq 0$)

- **Rate of convergence**
 - **Newton method: quadratically, order: 2**
 - **Secant method: order: 1.62**
 - **Faster than linear convergence**
 - **Both converge linearly to multiple roots**
- **Robustness**
 - **Newton method: converge, wandering, overshooting, and cycling**
 - **Secant method: converge, wandering, overshooting. But more robust than Newton**

Muller's method -1

- **A quadratic polynomial approximation is made to fit 3 points near a root.**

A quadratic polynomial $p_2(v) = av^2 + bv + c$

to fit $[x_1, f(x_1)], [x_0, f(x_0)], [x_2, f(x_2)]$

- **Using the quadratic rule to obtain the proper zero**

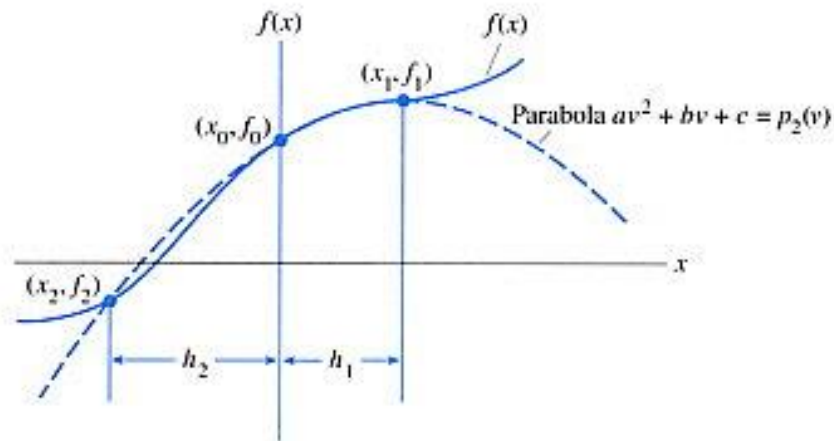


Figure 1.7

Muller's method -2

Simplify the development by transforming axes such that axes pass through the middle point.

So let $v = x - x_0$, and we try to fit the 3 points with $y = p_2(v) = av^2 + bv + c$.

Let $h_1 = x_1 - x_0$, $h_2 = x_0 - x_2$,

Evaluate the coefficients by evaluating $p(v)$ at the three points :

$$v = 0: a(0)^2 + b(0) + c = f_0 \Rightarrow c = f_0$$

$$v = h_1: a(h_1)^2 + b(h_1) + c = f_1$$

$$v = -h_2: a(h_2)^2 + b(-h_2) + c = f_2$$

Let $\gamma = h_2/h_1$, solving two linear equations for a and b :

$$a = \frac{\gamma f_1 - f_0(1 + \gamma) + f_2}{\gamma h_1^2(1 + \gamma)}$$

$$b = \frac{f_1 - f_0 - ah_1^2}{h_1}$$

Muller's method -3

- Solve for the root of $p_2(v) = av^2 + bv + c = 0$ by the quadratic formula, **choosing the root nearest to the middle point x_0 by making the absolute value of denominator as large as possible in the following form:**

$$\text{root} = x_0 - v = x_0 - \frac{2c}{b \pm \sqrt{b^2 - 4ac}} \left(= x_0 + \frac{-b \pm \sqrt{b^2 - 4ac}}{2a} \right)$$

– WHY? See next slide

- More stable compared to standard one
- Choose the sign to give the largest absolute value of denominator: $b > 0$: +, $b < 0$: --, $b = 0$: either

Muller's method -4

$$p_2(v) = av^2 + bv + c$$

$$= a(x - x_0)^2 + b(x - x_0) + c = 0$$

$$v = x - x_0 = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

$$x = x_0 + \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

with \pm chosen to minimize the modulus of the numerator (x is closest to x_0):

$b > 0 \rightarrow$ take $+$, $b < 0 \rightarrow$ take $-$.

But minimizing the numerator may cancel significant digits when b^2 is much larger than $4ac$ (since $\sqrt{b^2 - 4ac} \sim b$).

Example :

$$p(v) = v^2 + 62.10v + 1 = 0$$

Using 4-digit rounding arithmetic $b > 0$, so take $+$,

$$\sqrt{b^2 - 4ac} = 62.06 \approx b = 62.10$$

$$\text{and } x_1 = -0.01610723,$$

with relative error : 2.4×10^{-1}

cancellation error \rightarrow Large error!

Other root :

$$x_2 = -62.08390,$$

with relative error : 3.2×10^{-4}

Muller's method -5

Equivalent form :

$$\left(\frac{-b + \sqrt{b^2 - 4ac}}{2a} \right) \left(\frac{-b - \sqrt{b^2 - 4ac}}{-b - \sqrt{b^2 - 4ac}} \right) \\ = \frac{b^2 - (b^2 - 4ac)}{2a(-b - \sqrt{b^2 - 4ac})} = \frac{-2c}{b + \sqrt{b^2 - 4ac}}$$

$$\left(\frac{-b - \sqrt{b^2 - 4ac}}{2a} \right) \left(\frac{-b + \sqrt{b^2 - 4ac}}{-b + \sqrt{b^2 - 4ac}} \right) \\ = \frac{-2c}{b - \sqrt{b^2 - 4ac}}$$

To guard against this, root is calculated in the equivalent form :

$$x = x_0 - \frac{2c}{b \pm \sqrt{b^2 - 4ac}}$$

with \pm chosen to maximize the modulus of the denominator :

$b > 0 \rightarrow$ take +

$b < 0 \rightarrow$ take -

There will be no case of subtracting two nearly equal numbers.

Muller's method -6

Using the equivalent form :

$$x = x_0 - \frac{2c}{b \pm \sqrt{b^2 - 4ac}}$$

$b > 0$, take +

$$x_1 = -0.01610,$$

relative error : 6.2×10^{-4}

No cancellation error → Smaller error!

Other root :

$$x_2 = -50.00,$$

relative error : 1.9×10^{-1}

Muller's method -7

- Start from three initial point x_1, x_0, x_2
- Find a quadratic form passing through three point and find the roots
- Choose next 3 points that are **most closely spaced**
 - The root is to the right of x_0 : x_0, x_1, root
 - The root is to the left of x_0 : x_0, x_2, root

Muller's method -8

Procedure :

Given points $x_2 < x_0 < x_1$,

- Evaluate the functions f_2, f_0, f_1 .

- Find the coefficients of $p_2(v)$.

- Compute the roots :

$$x_r = x_0 - \frac{2c}{b \pm \sqrt{b^2 - 4ac}}$$

Choose the root closest to x_0

by making the denominator as large as possible, and label it x_r

If $x_r > x_0$

then rearrange to x_0, x_1, x_r

else rearrange to x_0, x_2, x_r

Until $|f(x_r)| < \text{Ftol}$

Muller's method -9

- **Converge rate: similar to that for Newton method (actually order 1.85)**
- **No derivative evaluation and only one function evaluation per iteration (after we have obtained the next point)**
- **Will find a complex root if complex starting value is given**
- **May fail under some conditions**
 - **What will make the denominator of root's expression zero or nearly zero**

Muller's method -10

EXAMPLE 1.2 Find a root between 0 and 1 of the same transcendental function as before: $f(x) = 3x + \sin(x) - e^x$. Let

$$x_0 = 0.5, \quad f(x_0) = 0.330704 \quad h_1 = 0.5,$$

$$x_1 = 1.0, \quad f(x_1) = 1.123189 \quad h_2 = 0.5,$$

$$x_2 = 0.0, \quad f(x_2) = -1 \quad \gamma = 1.0.$$

Then

$$a = \frac{(1.0)(1.123189) - 0.330704(2.0) + (-1)}{1.0(0.5)^2(2.0)} = -1.07644,$$

$$b = \frac{1.123189 - 0.330704 - (-1.07644)(0.5)^2}{0.5} = 2.12319,$$

$$c = 0.330704,$$

and

$$\begin{aligned} \text{root} &= 0.5 - \frac{2(0.330704)}{2.12319 + \sqrt{(2.12319)^2 - 4(-1.07644)(0.330704)}} \\ &= 0.354914. \end{aligned}$$

Muller's method -10

For the next iteration, we have

$$\begin{array}{lll} x_0 = 0.354914, & f(x_0) = -0.0138066 & h_1 = 0.145086, \\ x_1 = 0.5, & f(x_1) = 0.330704 & h_2 = 0.354914, \\ x_2 = 0, & f(x_2) = -1 & \gamma = 2.44623. \end{array}$$

Then

$$a = \frac{(2.44623)(0.330704) - (-0.0138066)(3.44623) + (-1)}{2.44623(0.145086)^2(3.44623)} = -0.808314,$$

$$b = \frac{0.330704 - (-0.0138066) - (-0.808314)(0.145086)^2}{0.145086} = 2.49180,$$

$$c = -0.0138066,$$

$$\begin{aligned} \text{root} &= 0.354914 - \frac{2(-0.0138066)}{2.49180 + \sqrt{(2.49180)^2 - 4(-0.808314)(-0.0138066)}} \\ &= 0.360465. \end{aligned}$$

After a third iteration, we get 0.3604217 as the value for the root, which is identical to that from Newton's method after three iterations. ▲

Fixed-Point Iteration ⁻¹

- **A useful way for root finding**
- **Basis for some important theory**
- **Fixed point** $f(x) = 0 \Rightarrow x = g(x)$ (in several forms)
 r : a fixed point of g if $r = g(r)$
- **Fixed-point iteration** $x_{n+1} = g(x_n), \quad n = 0, 1, 2, 3, \dots$
- **Under *suitable conditions*, the fixed-point iteration converges to the fixed point r , a root of $f(x)=0$**
- **Different rearranges will converge at different rate, or converge to different root, or diverge**

Fixed-Point Iteration -2

$$f(x) = x^2 - 2x - 3 = 0 \quad \text{roots : } -1, 3$$

1. $x = g_1(x) = \sqrt{2x+3}$

Start with $x_0 = 4$:

$$x_1 = 3.31662, \quad x_2 = 3.10375$$

$$x_3 = 3.03439, \quad x_4 = 3.01144$$

$$x_5 = 3.00381 \Rightarrow \text{converges to } x = 3$$

2. $x = g_2(x) = \frac{3}{x-2}$

$$x_1 = 1.5, \quad x_2 = -6, \quad x_3 = -0.375,$$

$$x_4 = -1.263158, \quad x_5 = -0.919355$$

$$x_6 = -1.02762, \quad x_7 = -0.990876$$

$$x_8 = -1.00305 \Rightarrow \text{converges to } x = -1$$

3. $x = g_3(x) = \frac{x^2 - 3}{2}$

$$x_0 = 4,$$

$$x_1 = 6.5$$

$$x_2 = 19.625,$$

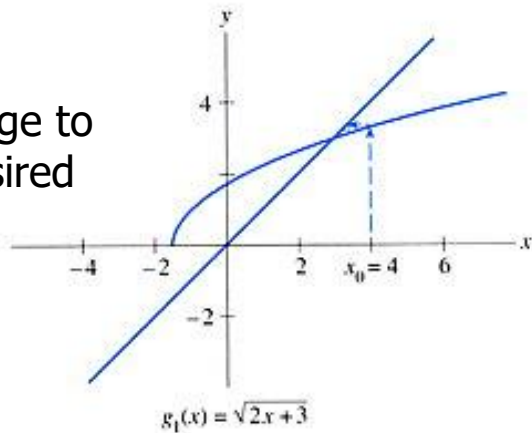
$$x_3 = 191.070 \Rightarrow \text{diverges}$$

Different arrangements have different convergence behavior. Why?

Look into this problem by using the graph of two intersecting plots of $y = x$ and $y = g(x)$.

Fixed-Point Iteration -3

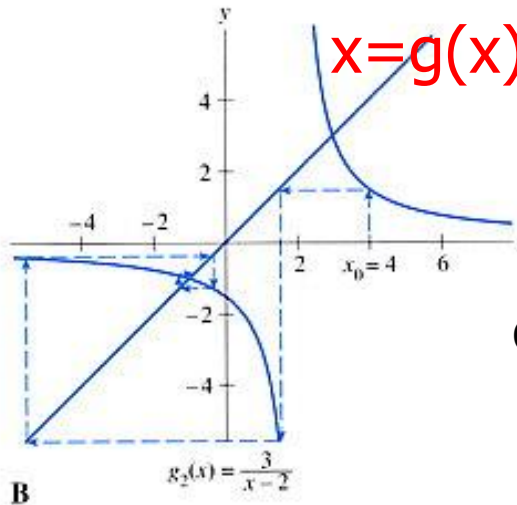
Converge to the desired root



A

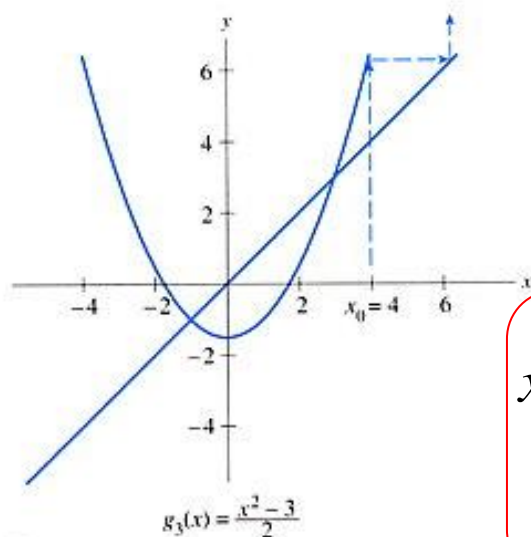
$$x=g(x) \Leftrightarrow y=x \text{ and } y=g(x)$$

Converge to another root



B

Diverge!



C

$$x_k \xrightarrow{g(x_k)} x_{k+1}$$

$$x_k \xrightarrow[\text{to the graph of } g]{\text{move up or down}} P_k \xrightarrow[\text{to the graph of } y=x]{\text{move right or left}} Q_{k+1} \xrightarrow[\text{to the } x\text{-axis}]{\text{move up or down}} x_{k+1}$$

$$g(x_k) \quad y = g(x_k) \quad x_{k+1} = y = g(x_k)$$

Figure 1.8

Fixed-Point Iteration -4

Rate of Convergence

Assume $\{x_n\}_{n=0}^{\infty}$ converges to r .

Let $e_n = x_n - r$.

If $|e_n|$ approaches to $c|e_{n-1}|^k$ as n becomes infinite, we say that the sequence converges to r with order of convergence k , or formally :

The sequence $\{x_n\}_{n=0}^{\infty}$ is called linearly converge to r if $e_n \rightarrow 0$ in such a way that

$$\lim_{n \rightarrow \infty} \frac{e_n}{e_{n-1}} = C_L, \text{ where } 0 < |C_L| < 1$$

The sequence $\{x_n\}_{n=0}^{\infty}$ is called superlinearly converge to r if $e_n \rightarrow 0$ in such a way that

$$\lim_{n \rightarrow \infty} \frac{e_n}{e_{n-1}} = 0$$

The sequence $\{x_n\}_{n=0}^{\infty}$ is called quadratically convergeto r if $e_n \rightarrow 0$ in such a way that

$$\lim_{n \rightarrow \infty} \frac{e_n}{e_{n-1}^2} = C_Q, \text{ where } C_Q \neq 0$$

Fixed-Point Iteration -5

$x_{n+1} = g(x_n)$ and r is the root (fixed point)

- **Error**

e_{n+1} and e_n

$$x_{n+1} - r = g(x_n) - r = \frac{g(x_n) - g(r)}{x_n - r} (x_n - r)$$

If $g(x)$ and $g'(x)$ are continuous on the interval from r to x_n , the mean value theorem implies that

$$x_{n+1} - r = g'(\tau_n)(x_n - r),$$

where τ_n lies between x_n and r , and hence

$$|e_{n+1}| = |g'(\tau_n)| |e_n|$$

- **Condition for convergence**

Suppose $|g'(x)| < K < 1$ for all x in the interval of size h around the root.

If x_0 is chosen in this interval, fixed point iteration will converge because

$$|e_{n+1}| < K |e_n| < K^2 |e_{n-1}| < \cdots < K^{n+1} |e_0|.$$

Note that all succeeding iterates will lie in this interval and will converge to r .

Fixed-Point Iteration -6

Convergence Rate

- Order of convergence: **linear**

$$\begin{aligned}|e_{n+1}| &= |x_{n+1} - r| \\ &= |g'(\tau_n)(x_n - r)| \\ &= |g'(\tau_n)| |e_n|,\end{aligned}$$

where τ_n lies between x_n and r .

So if $|g'(x)| < K < 1$ for all x in the interval,

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|} = \lim_{n \rightarrow \infty} |g'(\tau_n)|, \text{ where } \tau_n \text{ lies between } x_n \text{ and } r.$$

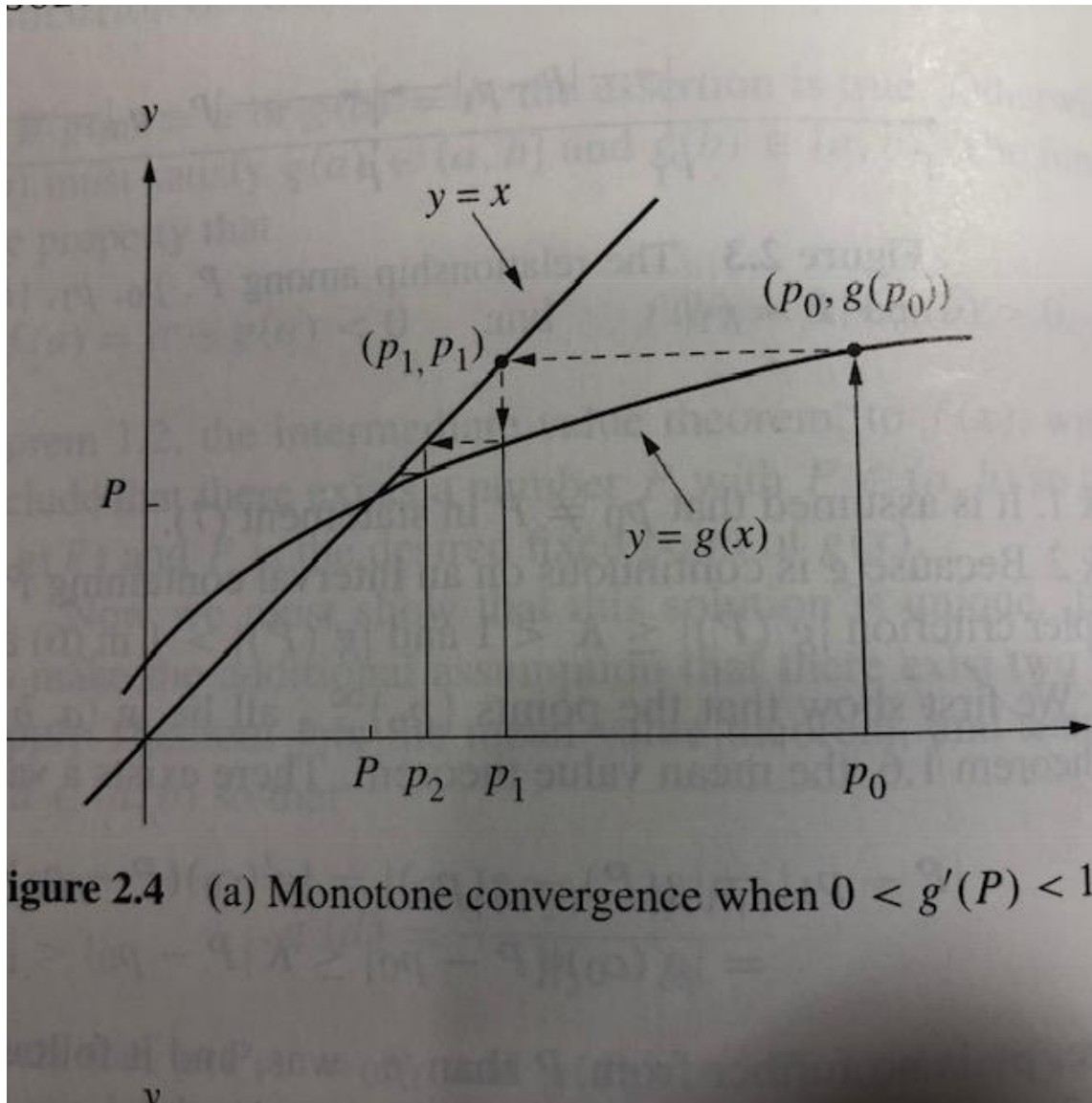
$$= |g'(r)| < 1 \text{ (since } x_n \text{ converges to } r, \tau_n \text{ converges to } r \text{)}$$

Converge linearly!

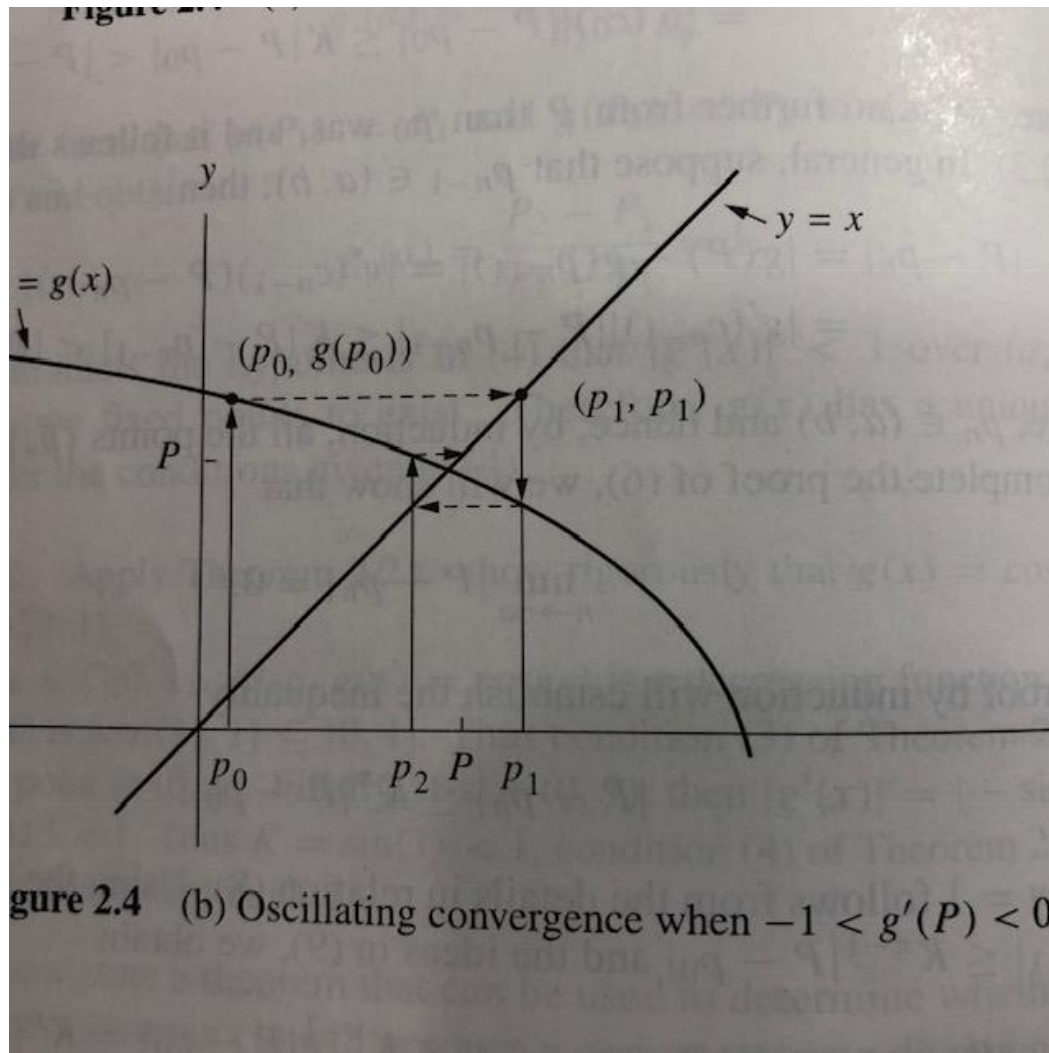
Fixed-Point Iteration -7

- **Converge $|g'(p)| < 1$**
 - **Monotone converging**
 - $0 < g'(p) < 1$
 - **Oscillating converging**
 - $-1 < g'(p) < 0$
- **Diverge $|g'(p)| > 1$**
 - **Monotone diverging**
 - $1 < g'(p)$
 - **Oscillating diverging**
 - $g'(p) < -1$

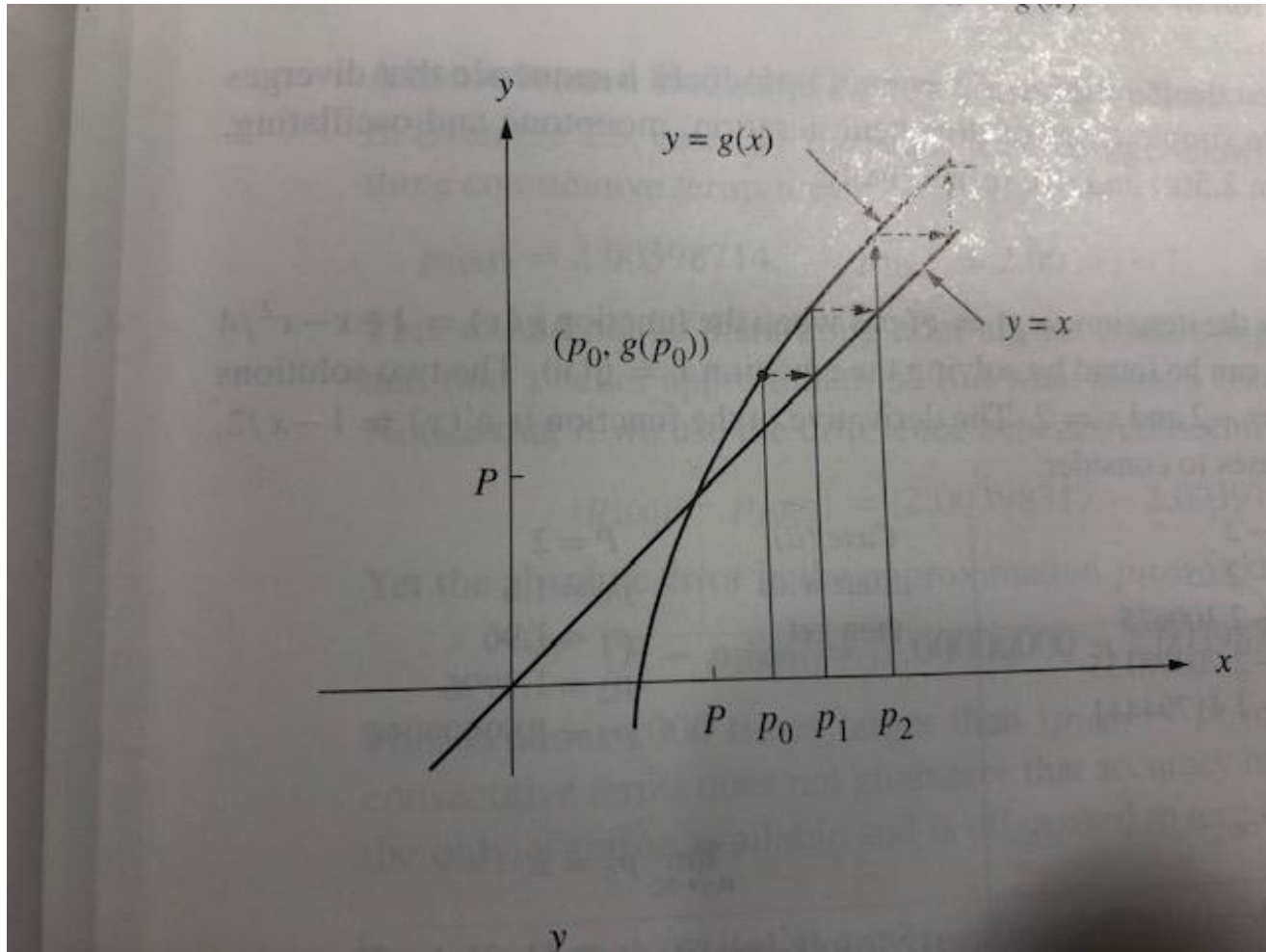
Fixed-Point Iteration -9



Fixed-Point Iteration -8

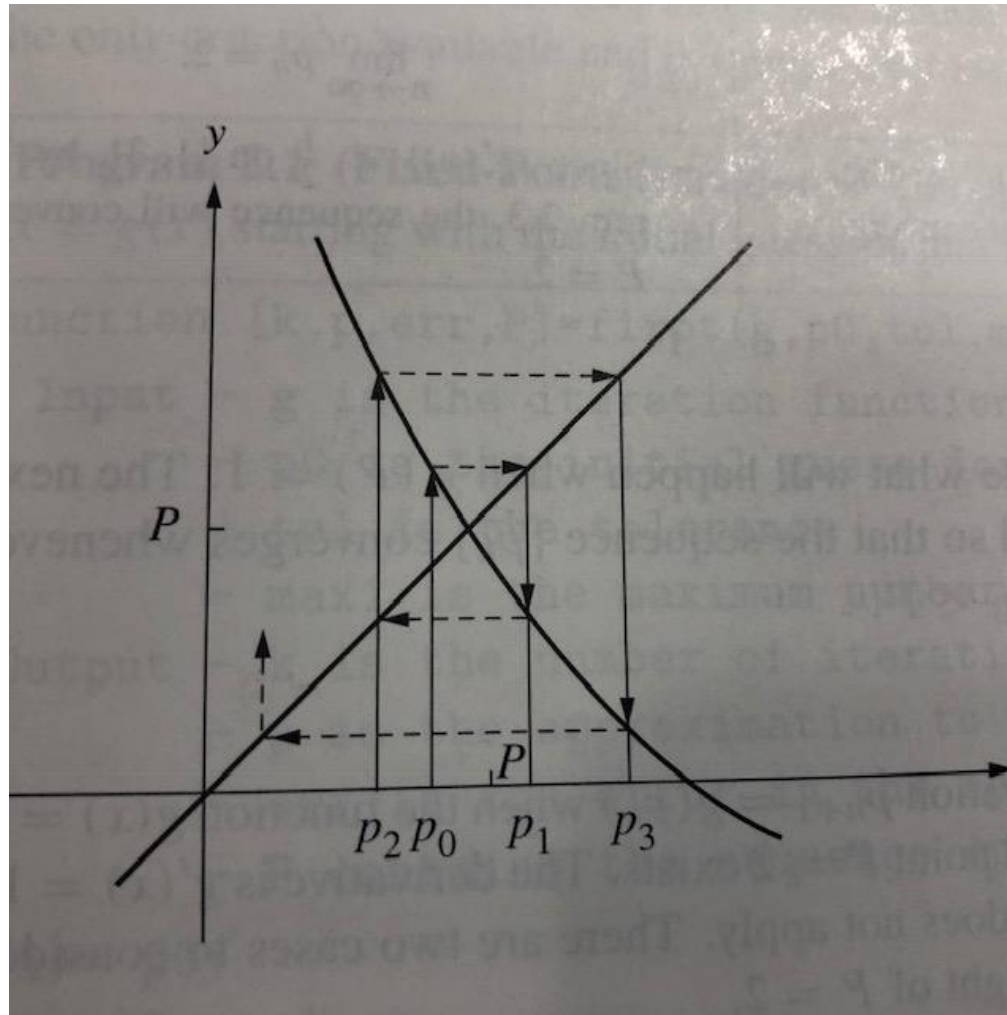


Fixed-Point Iteration -10



$1 < g'(x)$: Monotone divergent

Fixed-Point Iteration ⁻¹¹



$g'(x) < -1$: Oscillating divergent

Convergence Condition for Newton Method

Rewrite the Newton method as a fixed-point iteration :

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

$$g'(x) = \frac{f(x)f''(x)}{[f'(x)]^2}$$

Based on the condition for convergence of fixed-point iteration :

Newton method converges if, on an interval about the root r ,

$$|g'(x)| = \left| \frac{f(x)f''(x)}{[f'(x)]^2} \right| < 1$$

provided that x_0 is in the interval.

Convergence Order for Newton Method (for simple root)

By Taylor expansion of $g(x)$ at r :

$$g(x) = g(r) + g'(r)(x-r) + \frac{g''(\xi)}{2}(x-r)^2$$

where ξ lies in the interval from x_n to r .

$$\text{Since } f(r) = 0, g'(r) = \frac{f(r)f''(r)}{[f'(r)]^2} = 0,$$

we have

$$g(x_n) = g(r) + \frac{g''(\xi_n)}{2}(x_n - r)^2$$

$$e_{n+1} = x_{n+1} - r = g(x_n) - g(r) = \frac{g''(\xi_n)}{2}e_n^2$$

As the iterates x_n approach to r when n approaches to infinity, so does ξ_n . That is,

$$\begin{aligned} \lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|^2} &= \lim_{n \rightarrow \infty} \left| \frac{g''(\xi_n)}{2} \right| \\ &= \left| \frac{g''(r)}{2} \right| \neq 0 \end{aligned}$$

(for simple root)

So Newton method is quadratically convergent.

Convergence Condition of Bisection Method

- [From Mathews and Fink, p 54]

Theorem

Assume that $f \in C[a, b]$ and that there exists a number $r \in [a, b]$ such that $f(r) = 0$. If $f(a)f(b) < 0$, the sequence $\{x_n\}_{n=0}^{\infty}$ generated by the bisection process, then

$$|x_n - r| \leq \frac{b - a}{2^{n+1}} \quad \text{for } n = 0, 1, 2, \dots$$

Therefore

$$\lim_{n \rightarrow \infty} x_n = r.$$

Convergence Condition of Bisection Method

Pf : Since the root r and midpoint x_n lie in $[a, b]$,

$$|x_n - r| \leq \frac{b_n - a_n}{2}, \text{ for all } n.$$

Observe that $b_1 - a_1 = \frac{b_0 - a_0}{2} \quad (a_0 = a, b_0 = b)$

$$b_2 - a_2 = \frac{b_1 - a_1}{2} = \frac{b_0 - a_0}{2^2}$$

By induction, we have

$$b_n - a_n = \frac{b_0 - a_0}{2^n}$$

So

$$|x_n - r| \leq \frac{b_n - a_n}{2} \leq \frac{b_0 - a_0}{2^{n+1}}$$

Convergence Condition of Bisection Method

Pf : Since the root r and midpoint x_n lie in $[a, b]$,

$$|x_n - r| \leq \frac{b_n - a_n}{2}, \text{ for all } n.$$

Basic step : Observe that $b_1 - a_1 = \frac{b_0 - a_0}{2}$ ($a_0 = a, b_0 = b$)

$$b_2 - a_2 = \frac{b_1 - a_1}{2} = \frac{b_0 - a_0}{2^2}$$

Induction step : Assuming it is true for $n-1$, we have

$$b_n - a_n = \frac{b_{n-1} - a_{n-1}}{2} = \frac{1}{2} \frac{b_0 - a_0}{2^{n-1}} = \frac{b_0 - a_0}{2^n}$$

So

$$|x_n - r| \leq \frac{b_n - a_n}{2} \leq \frac{b_0 - a_0}{2^{n+1}}$$

Proof by induction:

Basic step: prove it is true for **n=1**

Induction step:

Assume it is true for **n=k**

Prove it is true for **n=k+1**

Convergence Order of Bisection Method

- One-half of the current interval is an upper bound to the error, which can serve as the estimate of the error
- So $|e_{n+1}| = 0.5 |e_n|$ Linearly convergent!!

Since

$$e_{n+1} = |x_{n+1} - r| \leq \frac{b_{n+1} - a_{n+1}}{2}, \quad e_{n+1} \approx \frac{b_{n+1} - a_{n+1}}{2}$$

$$e_n = |x_n - r| \leq \frac{b_n - a_n}{2}, \quad e_n \approx \frac{b_n - a_n}{2}$$

$$\lim_{n \rightarrow \infty} \frac{e_{n+1}}{e_n} = \left(\frac{b_{n+1} - a_{n+1}}{2} \right) / \left(\frac{b_n - a_n}{2} \right) = \frac{1}{2}$$

$$(\text{since } b_{n+1} - a_{n+1} = \frac{b_n - a_n}{2})$$

Convergence Order of Secant Method

Consider $x_{n+1} = g(x_n, x_{n-1})$.

Apply Taylor series and derive

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n e_{n-1}|} = \frac{1}{2} \left| \frac{f''(r)}{f'(r)} \right|$$

Further error analysis leads to

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|^\alpha} = K,$$

where

$$\alpha = \frac{1 + \sqrt{5}}{2} = 1.62, \quad K = \frac{1}{2} \left| \frac{f''(r)}{f'(r)} \right|^{1/\alpha}$$

So convergence order is 1.618.

Convergence Order for All Methods: summary

- **Fixed-point iteration: Order: linear**
- **Bisection: Order: linear**
 - One-half of the current interval is an upper bound to the error, which can serve as the estimate of the error
 - So $|e_{n+1}| = 0.5 |e_n|$
- **False Position: Order: linear**
 - $x_{n+1} = g(x_n, x_{n-1})$
 - The root is always bracketed, so x_{n-1} can be thought as a constant
 - It is exactly as the fixed-point iteration, so it is linearly convergent

Convergence order for All Methods : summary

- **Secant method: order: 1.62**

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

- **Similar analysis as the Newton method results in**

$$e_{n+1} = \frac{g''(\xi_1)g''(\xi_2)}{2} e_n e_{n-1}$$

- **Better than linear convergence but poorer than quadratic convergence. It has been shown the order is 1.62.**

- **Muller method: order=1.85**

- **Newton method: order= 2 (quadratic)**

Starting Point Issues

- **Newton method requires the initial point to be close to a root. General cases:**
 - **Converge, or scattering and then converge (wandering)**
 - **Oscillating**
 - **Diverge**
- **Newton method for polynomials (all roots)**
 - **Obtain an approximate for the first root**
 - **Proceed to obtain additional roots from the reduced polynomial, in which synthetic division can be employed to improve an initial estimate.**
 - **The process is repeated until the reduced polynomial is of degree 2**

Multiple Roots ⁻¹

- Multiple roots

- Examples

- $f(x)=(x+1)^3$: triple root at $x=-1$
 - $f(x)=(x-2)^2$: double root at $x=2$

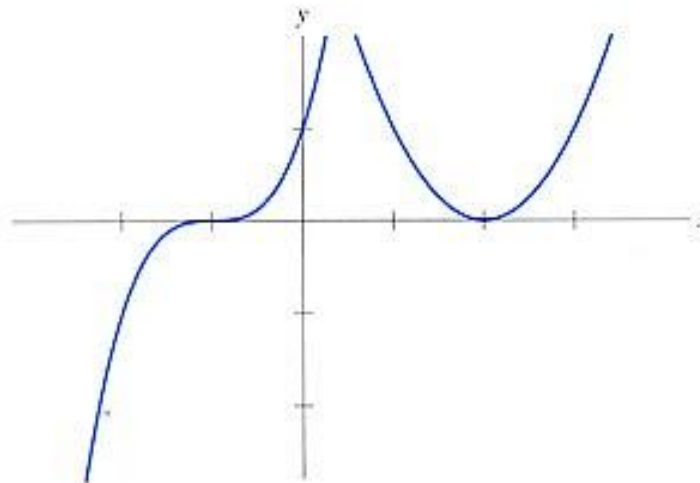


Figure 1.9

Multiple Roots -2

Previous methods do not work well

- Newton/secant method converges only **linearly** to a double/triple root
- Bisection and false position methods fail to get a double root. (because no sign changes)
- Muller's method is fastest, then Newton method, and then Secant method

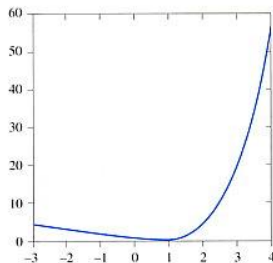


Figure 1.10
Plot of $(x-1)(e^{(x-1)} - 1)$

Table 1.7 Getting a double root, for $f(x) = (x-1)(e^{(x-1)} - 1)$

	Secant method	Newton's method	Muller's method
Estimate after 9 iterations	1.00331	1.00126	1.00058
Start value(s)	1.2, 1.5	2.0	0, 1.2, 1.5

Double root at 1

Multiple Roots -3

- **Problems for Newton method**

- **Slow convergence: Converge linearly at double/triple roots with ratio $(k-1)/k$**

- **Ratio of errors=1/2 for double roots**
- **Ratio of errors=2/3 for triple roots**

- **Another disadvantage: Imprecision**

- **Curve is flat near the root**

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

- **$f'(x)$ will always be 0 at a root**
- **There is a “neighborhood of uncertainty” around the root, where $f(x)$ are very small around the root**
 - » **Computers will find $f(x)$ equal to 0 throughout the neighborhood of the root**
 - » **Program cannot distinguish which value is really the root**
 - » **Using double precision helps to decrease the neighborhood of uncertainty**

Multiple Roots -4

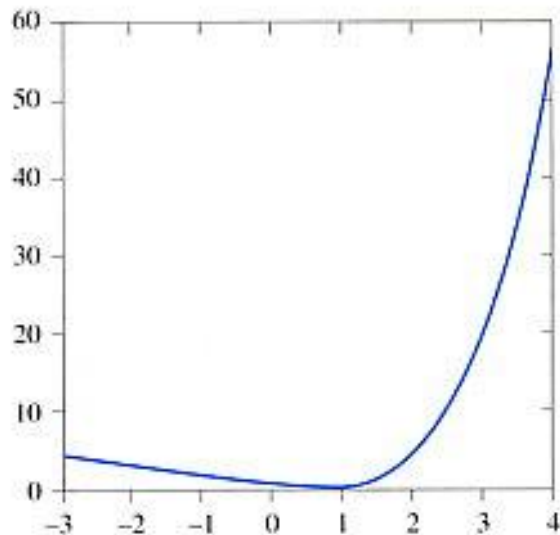


Figure 1.10
Plot of $(x-1)(e^{x-1} - 1)$

Table 1.4 Errors when finding a double root

Iteration	Error	Ratio
1	0.3679	
2	0.1666	0.453
3	0.0798	0.479
4	0.0391	0.490
5	0.0193	0.494
6	0.0096	0.497
7	0.0048	0.500
8	0.0024	0.500

Converge linearly to a double root
with ratio=1/2

Multiple Roots -5

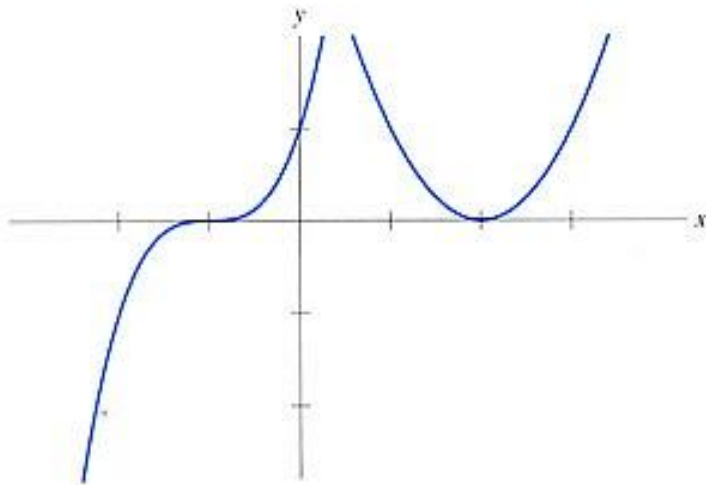


Figure 1.9

Table 1.10 Successive errors with Newton's method, for $f(x) = (x + 1)^3 = 0$

Iteration	Error	Iteration	Error
0	0.5	6	0.0439
1	0.3333	7	0.0293
2	0.2222	8	0.0195
3	0.1482	9	0.0130
4	0.0988	10	0.00867
5	0.0658		

Converge linearly to a triple root
with ratio=2/3

Multiple Roots -6

- **Linear convergence of Newton method to a double root and triple root**
- **For double roots**
 - **Bisection/false position methods fail for double roots**
- **For triple roots**
 - **All methods work, but even slower**

Multiple Roots -7

- **Why linearly converge with error ratio $(k-1)/k$?**

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

Taylor series of $g(x)$ about r :

$$g(x) = g(r) + g'(r)(x-r) + (g''(\xi)/2)(x-r)^2$$

For simple roots, $g'(r) = 0$ leads to quadratic convergence.

For a root r of multiplicity k ,

$$\text{we cannot say } g'(r) = \frac{f(r)f''(r)}{(f'(r))^2} = 0. \text{ Why? } 0/0$$

We can factor out $(x-r)^k$ from $f(x)$ to get

$$f(x) = (x-r)^k Q(x), \text{ where } Q(r) \neq 0,$$

even though $f'(r) = f''(r) = \dots = f^{(k-1)}(r) = 0$.

Multiple Roots -8

Consider $g'(x) = \frac{f(x)f''(x)}{[f'(x)]^2}$ and let $h = x - r$

$$g'(x) = \frac{h^k Q(x) [h^k Q''(x) + 2kh^{k-1} Q'(x) + k(k-1)h^{k-2} Q(x)]}{[h^k Q'(x) + kh^{k-1} Q(x)]^2}$$

$$= \frac{h^{2k-2} Q(x) [h^2 Q''(x) + 2kh Q'(x) + k(k-1) Q(x)]}{h^{2k-2} [h Q'(x) + k Q(x)]^2}$$

When $x = r$, both denominator and numerator are 0, we cannot say $g'(r) = 0$.

Actually dividing denominator and numerator by h^{2k-2} , we have

$$\begin{aligned} \lim_{h \rightarrow 0} g'(x) &= g'(r) \\ &= \frac{Q(r) [0 \cdot Q''(r) + 0 \cdot Q'(r) + k(k-1)Q(r)]}{[0 \cdot Q'(r) + kQ(r)]^2} \\ &= \frac{k(k-1)Q(r)^2}{k^2 Q(r)^2} = \frac{k-1}{k} \neq 0 \end{aligned}$$

$$\begin{aligned} g(x_n) &= g(r) + g'(\xi_n)(x_n - r) \\ e_{n+1} &= x_{n+1} - r = g(x_n) - g(r) = g'(\xi_n)e_n \\ \lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|} &= \lim_{n \rightarrow \infty} g'(\xi_n) = g'(r) = \frac{k-1}{k}. \end{aligned}$$

So the convergence is linear with

$$\lim_{n \rightarrow \infty} \frac{|e_{n+1}|}{|e_n|} = \frac{k-1}{k}.$$

Remedies for Multiple Roots with Newton Method ⁻¹

1. For a root of multiplicity k , restore quadratic convergence by modifying the formula to (if k is known)

$$x_{n+1} = x_n - k \frac{f(x_n)}{f'(x_n)} \equiv g_k(x_n)$$

Why?

At $f(r) = 0$, $g_k(r) = r$. Based on

$f(x) = (x - r)^k Q(x)$, where $Q(r) \neq 0$,

we have

$$g'_k(x) =$$

$$\frac{(r - x) \{ k(r - x)QQ'' + Q'[2kQ - (k - 1)(r - x)Q'] \}}{[(r - x)Q' + kQ]^2}$$

We can easily see that $g'_k(r) = 0$, so the modified Newton method converge quadratically at multiple roots (including simple root where $k = 1$).

Remedies for Multiple Roots with Newton Method -2

With modified Newton method :

$$f(x) = (x - 1)(e^{(x-1)} - 1)$$

The third iterate is

$$x_3 = 1.00088 \text{ with } f(x_3) = 0.00000$$

Note : we also find that

$$e_{n+1} = 0.24e_n^2,$$

confirming quadratic convergence.

Remedies for Multiple Roots with Newton Method ⁻³

- But k is unknown in advance!!
- Guess a k by comparing a graph of $f(x)$ with the plots of $(x-r)^k$ using an approximate r and various values of k .
 - The flatness will be similar, but this is not justified
- Divide $f(x)$ by $(x-r)$ and deflate the function, reducing the multiplicity by one.
 - But r is unknown
 - Dividing f by $(x-s)$, where s is an approximate of r , does almost the same thing.

Remedies for Multiple Roots with Newton Method ⁻⁴

2. If k is known

- $k=2$

- Newton method can be applied to $f'(x)=0$ to converge quadratically to a double root of $f(x)=0$. (simple root of $f'(x)=0$)

- $k=3$

- Similarly, the method can be applied to $f''(x)=0$ to get quadratic convergence to a triple root. (simple root of $f''(x)=0$)

$$f(x) = (x-r)^2 Q(x), \text{ where } Q(r) \neq 0,$$

$$f'(x) = 2(x-r)Q(x) + (x-r)^2 Q'(x)$$

$$= (x-r)[2Q(x) + (x-r)Q'(x)]$$

$$\text{where } 2Q(r) + (r-r)Q'(r) = 2Q(r) \neq 0$$

Remedies for Multiple Roots with Newton Method -5

3. If k is known [Acton 1970]

A most tempting scheme:

- When $f(x)$ has a root r of multiplicity k , we have

$$f(x) = (x - r)^k Q(x) \text{ and } Q(r) \neq 0$$

$$f'(x) = k(x - r)^{k-1} Q(x) + (x - r)^k Q'(x)$$

$$= (x - r)^{k-1} [kQ(x) + (x - r)Q'(x)]$$

- If we divide $f(x)$ by $f'(x)$, we effectively deflate $f(x)$ $n-1$ times and we now work with a new function that has only a **single root** at $x=r$.
Why? See next page.

Remedies for Multiple Roots with Newton Method -6

If $f(x)$ has a root of multiplicity k at $x = r$,

$$f(x) = (x - r)^k Q(x), \text{ where } Q(r) \neq 0.$$

Let

$$\begin{aligned} S(x) &= \frac{f(x)}{f'(x)} = \frac{(x - r)^k Q(x)}{k(x - r)^{k-1} Q(x) + (x - r)^k Q'(x)} \\ &= (x - r) \frac{Q(x)}{kQ(x) + (x - r)Q'(x)} \end{aligned}$$

which has a simple root at $x = r$ (since $Q(r) \neq 0$).

When $S(x)$ used in Newton formula, we have

$$\begin{aligned} x_{n+1} &= x_n - \frac{S(x_n)}{S'(x_n)} \\ &= x_n - \frac{f(x_n)f'(x_n)}{[f'(x_n)]^2 - f(x_n)f''(x_n)} \equiv g(x_n) \end{aligned}$$

Accelerating convergence

Aitken acceleration ⁻¹

- Used to speed up convergence of any sequence that is **linearly** convergent.

Definition s :

Given the sequence $\{p_n\}_{n=0}^{\infty}$, define the forward difference

$$\Delta p_n = p_{n+1} - p_n, \text{ for } n \geq 0.$$

Higher powers $\Delta^k p_n$ is defined recursively by

$$\Delta^k p_n = \Delta^{k-1}(\Delta p_n), \text{ for } n \geq 2.$$

Accelerating convergence

Aitken acceleration -2

Theorem :

[From Friedman/Kandel, p90]

Assume that the sequence

$\{p_n\}_{n=0}^{\infty}$ converge linearly to p
and that $p_n - p \neq 0$ for all $n \geq 0$.

If there exists a real number

A with $|A| < 1$

such that

$$\lim_{n \rightarrow \infty} \frac{p_{n+1} - p}{p_n - p} = A,$$

then the sequence $\{q_n\}_{n=0}^{\infty}$

defined by

$$\begin{aligned} q_n &= p_n - \frac{(\Delta p_n)^2}{\Delta^2 p_n} \\ &= p_n - \frac{(p_{n+1} - p_n)^2}{p_{n+2} - 2p_{n+1} + p_n} \end{aligned}$$

converges to p faster than $\{p_n\}_{n=0}^{\infty}$
in the sense that

$$\lim_{n \rightarrow \infty} \left| \frac{q_n - p}{p_n - p} \right| = 0.$$

Accelerating convergence

Aitken acceleration -3

Proof :

Since there exists a real number

A with $|A| < 1$ such that

$$\lim_{n \rightarrow \infty} \frac{p_{n+1} - p}{p_n - p} = A,$$

we have

$$\frac{p_{n+1} - p}{p_n - p} \approx A \approx \frac{p_{n+2} - p}{p_{n+1} - p}$$

when n is large.

This implies

$$(p_{n+1} - p)^2 \approx (p_n - p)(p_{n+2} - p)$$

Expanding the terms and

cancelling p^2 yields

$$\begin{aligned} p &\approx \frac{p_{n+2}p_n - p_{n+1}^2}{p_{n+2} - 2p_{n+1} + p_n} \\ &= p_n - \frac{(p_{n+1} - p_n)^2}{p_{n+2} - 2p_{n+1} + p_n} = q_n. \end{aligned}$$

We need to prove that $\{q_n\}_{n=0}^{\infty}$

converges to p faster than

$\{p_n\}_{n=0}^{\infty}$ in the sense that

$$\lim_{n \rightarrow \infty} \left| \frac{q_n - p}{p_n - p} \right| = 0.$$

Accelerating convergence

Aitken acceleration -4

Proof : (continued)

Let $e'_n = q_n - p$, then

$$\begin{aligned} e'_n &= p_n - p - \frac{(p_{n+1} - p_n)^2}{p_{n+2} - 2p_{n+1} + p_n} \\ &= e_n - \frac{(e_{n+1} - e_n)^2}{e_{n+2} - 2e_{n+1} + e_n} \\ &= \frac{e_n e_{n+2} - e_{n+1}^2}{e_{n+2} - 2e_{n+1} + e_n} \end{aligned}$$

Note that

Based on $\lim_{n \rightarrow \infty} \frac{e_{n+1}}{e_n} = A$,

$$e_{n+1} = (A + \theta_n)e_n$$

$$\begin{aligned} e_{n+2} &= (A + \theta_{n+1})e_{n+1} \\ &= (A + \theta_{n+1})(A + \theta_n)e_n \end{aligned}$$

where $\theta_n, \theta_{n+1} \rightarrow 0$

as $n \rightarrow \infty$.

Accelerating convergence

Aitken acceleration -5

Proof : (continued)

Therefore

$$\begin{aligned}
 e'_n &= \frac{(A + \theta_{n+1})(A + \theta_n)e_n^2 - (A + \theta_n)^2 e_n^2}{(A + \theta_{n+1})(A + \theta_n)e_n - 2(A + \theta_n)e_n + e_n} \\
 &= \frac{(A + \theta_{n+1})(A + \theta_n) - (A + \theta_n)^2}{(A + \theta_{n+1})(A + \theta_n) - 2(A + \theta_n) + 1} e_n \\
 &= \frac{(A + \theta_n)[(A + \theta_{n+1}) - (A + \theta_n)]}{(A^2 - 2A + 1) + A\theta_{n+1} + A\theta_n + \theta_{n+1}\theta_n - 2\theta_n} \\
 &= \frac{(A + \theta_n)}{(A - 1)^2 + A(\theta_{n+1} + \theta_n) + \theta_n(\theta_{n+1} - 2)} (\theta_{n+1} - \theta_n) e_n
 \end{aligned}$$

Hence

$$\frac{e'_n}{e_n} = \frac{(A + \theta_n)}{(A - 1)^2 + A(\theta_{n+1} + \theta_n) + \theta_n(\theta_{n+1} - 2)} (\theta_{n+1} - \theta_n)$$

and

$$\lim_{n \rightarrow \infty} \left| \frac{e'_n}{e_n} \right| = \frac{A}{(A - 1)^2} \lim_{n \rightarrow \infty} (\theta_{n+1} - \theta_n) = 0.$$

Accelerating convergence

Aitken acceleration ⁻⁶

Advantages:

- **No significant additional computing time, because the time for computing q_n is negligible, once p_n , p_{n+1} , p_{n+2} are already given.**
- **Aitken acceleration speeds up the convergence for any sequence that is linearly convergent.**

Rate of convergence of Aitken method

- **Quadratic? NO!! Still linear, but faster than that for original sequence. WHY??**

Accelerating convergence

Aitken acceleration -7

Theorem :

Consider $p_{n+1} = g(p_n)$. If the sequence $\{p_n\}_{n=0}^{\infty}$ converge linearly to p and

$$\lim_{n \rightarrow \infty} \frac{e_{n+1}}{e_n} = A \quad (= g'(p)),$$

for $|A| < 1$, then the Aitken sequence $\{q_n\}_{n=0}^{\infty}$ behaves asymptotically according to

$$\lim_{n \rightarrow \infty} \frac{e'_{n+1}}{e'_n} = A^2 < A$$

Accelerating convergence

Aitken acceleration -7

Proof : Since

$$\begin{aligned} e'_n &= \frac{(A + \theta_{n+1})(A + \theta_n)e_n^2 - (A + \theta_n)^2 e_n^2}{(A + \theta_{n+1})(A + \theta_n)e_n - 2(A + \theta_n)e_n + e_n} \\ &= \frac{(A + \theta_n)}{(A - 1)^2 + A(\theta_{n+1} + \theta_n) + \theta_n(\theta_{n+1} - 2)} (\theta_{n+1} - \theta_n)e_n \end{aligned}$$

Hence

$$\frac{e'_{n+1}}{e'_n} = \frac{(A + \theta_{n+1})[(A - 1)^2 + A(\theta_n + \theta_{n+1}) + \theta_n(\theta_{n+1} - 2)]}{(A + \theta_n)[(A - 1)^2 + A(\theta_{n+1} + \theta_{n+2}) + \theta_{n+1}(\theta_{n+2} - 2)]} \frac{e_{n+1}(\theta_{n+2} - \theta_{n+1})}{e_n(\theta_{n+1} - \theta_n)}$$

Since

$$\lim_{n \rightarrow \infty} \theta_n = 0 \text{ and } \lim_{n \rightarrow \infty} \frac{e_{n+1}}{e_n} = A$$

We have

$$\lim_{n \rightarrow \infty} \frac{e'_{n+1}}{e'_n} = A \lim_{n \rightarrow \infty} \frac{(\theta_{n+2} - \theta_{n+1})}{(\theta_{n+1} - \theta_n)}.$$

We need to show that

$$\lim_{n \rightarrow \infty} \frac{(\theta_{n+2} - \theta_{n+1})}{(\theta_{n+1} - \theta_n)} = A.$$

Accelerating convergence

Aitken acceleration -7

Proof :

Consider

$$g(x) = g(r) + (x-r)g'(r) + \frac{(x-r)^2}{2} g''(\xi),$$

where ξ is between x and r .

$$\begin{aligned} e_{n+1} &= x_{n+1} - r = g(x_n) - g(r) \\ &= (x_n - r)g'(r) + \frac{(x_n - r)^2}{2} g''(\xi_n), \end{aligned}$$

where ξ_n is between x_n and r , so $\lim_{n \rightarrow \infty} \xi_n = r$.

Due to the continuity of g'' , we can write

$$g''(\xi_n) = g''(r) + \theta'_n, \lim_{n \rightarrow \infty} \theta'_n = 0.$$

Hence

$$e_{n+1} = e_n A + \frac{e_n^2}{2} (A' + \theta'_n),$$

where $g'(r) = A, g''(r) = A'$

Because $e_{n+1} = (A + \theta_n)e_n$, we obtain

$$e_{n+1} = e_n A + \frac{e_n^2}{2} (A' + \theta'_n) = (A + \theta_n)e_n$$

$$\text{and } \theta_n = \frac{e_n}{2} (A' + \theta'_n).$$

Finally,

$$\begin{aligned} \lim_{n \rightarrow \infty} \frac{\theta_{n+1}}{\theta_n} &= \lim_{n \rightarrow \infty} \frac{\frac{e_{n+1}}{2} (A' + \theta'_{n+1})}{\frac{e_n}{2} (A' + \theta'_n)} \\ &= \lim_{n \rightarrow \infty} \left(\frac{e_{n+1}}{e_n} \frac{A' + \theta'_{n+1}}{A' + \theta'_n} \right) = \lim_{n \rightarrow \infty} \frac{e_{n+1}}{e_n} = A. \end{aligned}$$

Therefore

$$\lim_{n \rightarrow \infty} \frac{\theta_{n+2} - \theta_{n+1}}{\theta_{n+1} - \theta_n} = \lim_{n \rightarrow \infty} \frac{\frac{\theta_{n+2}}{\theta_{n+1}} - 1}{1 - \frac{\theta_n}{\theta_{n+1}}} = \frac{A - 1}{1 - \frac{1}{A}} = A$$

Accelerating convergence

Aitken acceleration -9

Theorem:

Given a tolerance $\varepsilon > 0$, let N, N' denote the number of iterations required for convergence using fixed point iteration and Aitken's iteration, respectively. Then

$$\lim_{\varepsilon \rightarrow \infty} \frac{N'}{N} = \frac{1}{2}$$

Accelerating convergence

Aitken acceleration -8

$$N=11, N'=5$$

$$|e_4/e_3|=0.312 \sim |A|=g'(s)=(3/8) s^2=0.308$$

$$|e'_4/e'_3|=0.096 \sim A^2=0.095$$

Example 3.3.4.

Let $f(x) = 1 - (x^3/8)$, $0 \leq x \leq 1$ (Fig. 3.3.1) and consider the equation $x = f(x)$, given a first approximation $x_0 = 0$ and a tolerance $\epsilon = 10^{-6}$. The exact solution computed to nine significant digits is $s = 0.906795303$. A comparison between the performances of the SIM and ATKN is given in Table 3.3.3.

SIM: Fixed point iteration

Table 3.3.3. $x = 1 - (x^3/8)$, $0 \leq x \leq 1$, $x_0 = 0$, $\epsilon = 10^{-6}$

n	x_n	x'_n	$ \epsilon_n/\epsilon_{n-1} $	$\epsilon'_n/\epsilon'_{n-1}$
0	0.000000000	0.888888889		
1	1.000000000	0.906020558	0.103	0.043
2	0.875000000	0.906717286	0.341	0.101
3	0.916259766	0.906788044	0.298	0.093
4	0.903846331	0.906794608	0.312	0.096
⋮				
12	0.906796083			

Accelerating convergence

Aitken acceleration -10

Example 3.3.1.

Consider $x = e^{-x}$, $1/2 \leq x \leq 2/3$, $x_0 = 0.5$. Table 3.3.1 contains the first eight iterations, using the SIM, compared with eight iterations based on Aitken's scheme.

Table 3.3.1. $x = e^{-x}$ (Aitken's Method)

n	x_n	x'_n
0	0.500000	0.567624
1	0.606531	0.567299
2	0.545239	0.567193
3	0.579703	0.567159
4	0.560065	0.567148
5	0.571172	0.567145
6	0.564863	0.567144
7	0.568438	0.567143
8	0.566409	
\vdots		
22	0.567143	

We see that ATKN yields the first 6 digits of the exact solution after 8 iterations, compared with the 22 that are needed for the SIM (to provide the same accuracy). \square

Example 3.3.2.

Accelerating convergence

Aitken acceleration -11

Exact solution to 8 decimal digits: 0.73908513

ITERATION
Table 3.3.2. $x = \cos x$ (Aitken's Method)

n	x_n	x'_n
0	1.00000000	0.72801036
1	0.54030231	0.73366516
2	0.85755322	0.73690629
3	0.65428979	0.73805042
4	0.79348036	0.73863610
5	0.70136877	0.73887658
6	0.76395968	0.73899224
7	0.72210243	0.73904251
8	0.75041776	0.73906595
9	0.73140404	0.73907638
10	0.74423735	0.73908118
11	0.73560474	
\vdots		
26	0.73909441	

To approximate r within error 10^{-5} :
Aitken method: requires 11 iterations
Fixed-point iteration: requires 25 iterations.

Nonlinear systems

- **Nonlinear system**

$$f_1(x_1, x_2, x_3, \dots, x_n) = 0$$

$$f_2(x_1, x_2, x_3, \dots, x_n) = 0$$

$$\vdots$$

$$f_n(x_1, x_2, x_3, \dots, x_n) = 0$$

- **2x2 system**

$$f(x, y) = 0$$

$$g(x, y) = 0$$

Solution set : Regarded as the intersections
of two algebraic curves.

Nonlinear systems

- **Example**

$$\begin{cases} x^2 + y^2 = 4 \\ e^x + y = 0 \end{cases}$$

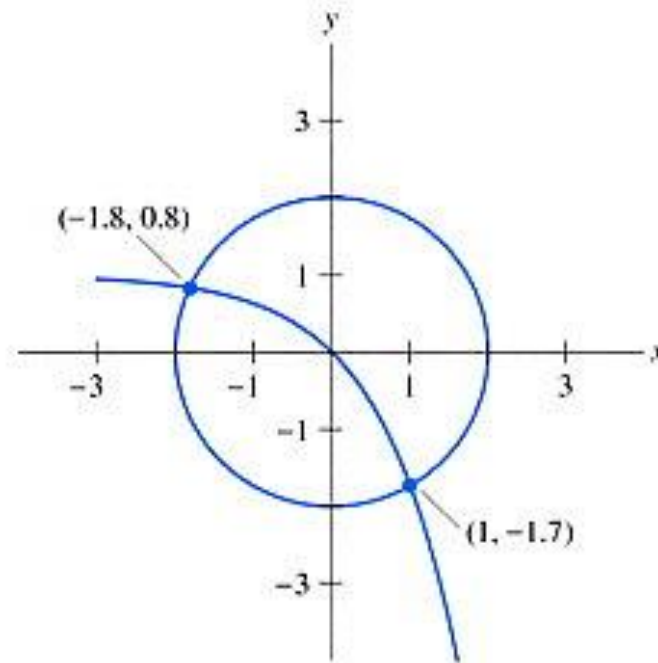


Figure 1.11

Nonlinear Systems Methods

Similar to solving $f(x)=0$

- **Fixed point iteration**
 - Different formulae behave differently
 - Converge linearly
- **Newton method**
 - Derivative vs. Jacobian matrix
 - Converge quadratically

Nonlinear Systems

Fixed Point Iteration

Consider the 2x2 system :

$$\begin{cases} f_1(x, y) = 0 \\ f_2(x, y) = 0 \end{cases}$$

Fixed point equation :

$$\begin{cases} x = g_1(x, y) \\ y = g_2(x, y) \end{cases}$$

If the starting point is sufficiently close to the fixed point (p, q) , and if

$$\left| \frac{\partial g_1}{\partial x}(p, q) \right| + \left| \frac{\partial g_1}{\partial y}(p, q) \right| < 1 \quad \text{and}$$

$$\left| \frac{\partial g_2}{\partial x}(p, q) \right| + \left| \frac{\partial g_2}{\partial y}(p, q) \right| < 1$$

then the iteration converges to the fixed point.

Note :

If the conditions are not met, the iteration might diverge.

Nonlinear Systems

Fixed Point Iteration

$$\begin{cases} x^2 - 2x - y + 0.5 = 0 \\ x^2 + 4y^2 - 4 = 0 \end{cases}$$

Fixed point formulation :

$$\begin{cases} x = \frac{x^2 - y + 0.5}{2} \\ y = \frac{-x^2 - 4y^2 + 8y + 4}{8} \end{cases}$$

Starting with $(0,1)$, it converges to $(-0.2, 1.0)$.

Observe that

$$\left| \frac{\partial g_1}{\partial x}(p, q) \right| + \left| \frac{\partial g_1}{\partial y}(p, q) \right|$$

$$= |x| + |-0.5| < 1, \text{ and}$$

$$\left| \frac{\partial g_2}{\partial x}(p, q) \right| + \left| \frac{\partial g_2}{\partial y}(p, q) \right|$$

$$< \frac{|-x|}{4} + |-y + 1| < 1$$

for $-0.5 < x < 0.5$ and

$0.5 < y < 1.5$.

Starting with $(2,0)$, it diverges away from the solution.

Nonlinear Systems

Newton Method

Newton method for solving $\begin{cases} f(x, y) = 0 \\ g(x, y) = 0 \end{cases}$

Let (r, s) be a root and expand both functions as a Taylor series about (x_i, y_i) , where (x_i, y_i) is a point near the root:

$$\begin{cases} f(r, s) = 0 = f(x_i, y_i) + f_x(x_i, y_i)(r - x_i) \\ \quad + f_y(x_i, y_i)(s - y_i) + \cdots \\ g(r, s) = 0 = g(x_i, y_i) + g_x(x_i, y_i)(r - x_i) \\ \quad + g_y(x_i, y_i)(s - y_i) + \cdots \end{cases}$$

Truncating both series give

$$\begin{cases} 0 = f(x_i, y_i) + f_x(x_i, y_i)(r - x_i) + f_y(x_i, y_i)(s - y_i) \\ 0 = g(x_i, y_i) + g_x(x_i, y_i)(r - x_i) + g_y(x_i, y_i)(s - y_i) \end{cases}$$

Nonlinear systems

Newton method

The system is rewritten as

$$\begin{cases} f_x(x_i, y_i)(r - x_i) + f_y(x_i, y_i)(s - y_i) = -f(x_i, y_i) \\ g_x(x_i, y_i)(r - x_i) + g_y(x_i, y_i)(s - y_i) = -g(x_i, y_i). \end{cases}$$

The iterate is derived by solving

$$\begin{cases} f_x(x_i, y_i)\Delta x_i + f_y(x_i, y_i)\Delta y_i = -f(x_i, y_i) \\ g_x(x_i, y_i)\Delta x_i + g_y(x_i, y_i)\Delta y_i = -g(x_i, y_i) \end{cases}$$

where $x_{i+1} = x_i + \Delta x_i$ and $y_{i+1} = y_i + \Delta y_i$ are improved estimate of the root.

Repeating this iteration until both $f(x_i)$ and $g(y_i)$ are close to zero.

Nonlinear Systems

Newton Method

$$\begin{cases} f_x(x_i, y_i)\Delta x_i + f_y(x_i, y_i)\Delta y_i = -f(x_i, y_i) \\ g_x(x_i, y_i)\Delta x_i + g_y(x_i, y_i)\Delta y_i = -g(x_i, y_i) \end{cases}$$

can be rewritten as

$$J(x_i, y_i) \begin{bmatrix} \Delta x_i \\ \Delta y_i \end{bmatrix} = \begin{bmatrix} -f(x_i, y_i) \\ -g(x_i, y_i) \end{bmatrix}$$

If $J(x_i, y_i)$ is nonsingular, Δx_i and Δy_i can be found by solving the linear system.

New iterate is obtained by

$$\begin{bmatrix} x_{i+1} \\ y_{i+1} \end{bmatrix} = \begin{bmatrix} x_i \\ y_i \end{bmatrix} + \begin{bmatrix} \Delta x_i \\ \Delta y_i \end{bmatrix}.$$

Nonlinear Systems

Newton Method

- The Newton method for solving nonlinear system converges quadratically, but requiring
 - n^2+n function evaluations at each step
 - 6 function evaluations for a 2x2 system
 - 12 function evaluations for a 3x3 system
 - Solving an $n \times n$ linear system