Finding the Roots
or
Solving Nonlinear Equations

Solving Nonlinear Equations and Sets of Linear Equations

- Equations can be solved analytically
  - ✓Solution of $y=mx+b$
  - ✓Solution of $ax^2+bx+c=0$
- Large systems of linear equations
- Higher order equations
  - ✓Specific energy equation $h^3 - Eh^2 + \frac{q^2}{2g} = 0$
- Nonlinear algebraic equations x=g(x)
  - ✓Manning's equation

$$h = Qn\left[\left(\frac{wh}{w+2h}\right)^{2/3} S^{1/2} w\right]^{-1}$$

- Preliminaries
- Fixed-Point Iteration
- Bisection
- Newton’s Method
- The Secant Method
- Hybrid Methods
- Roots of Polynomials

Roots of f(x)=0

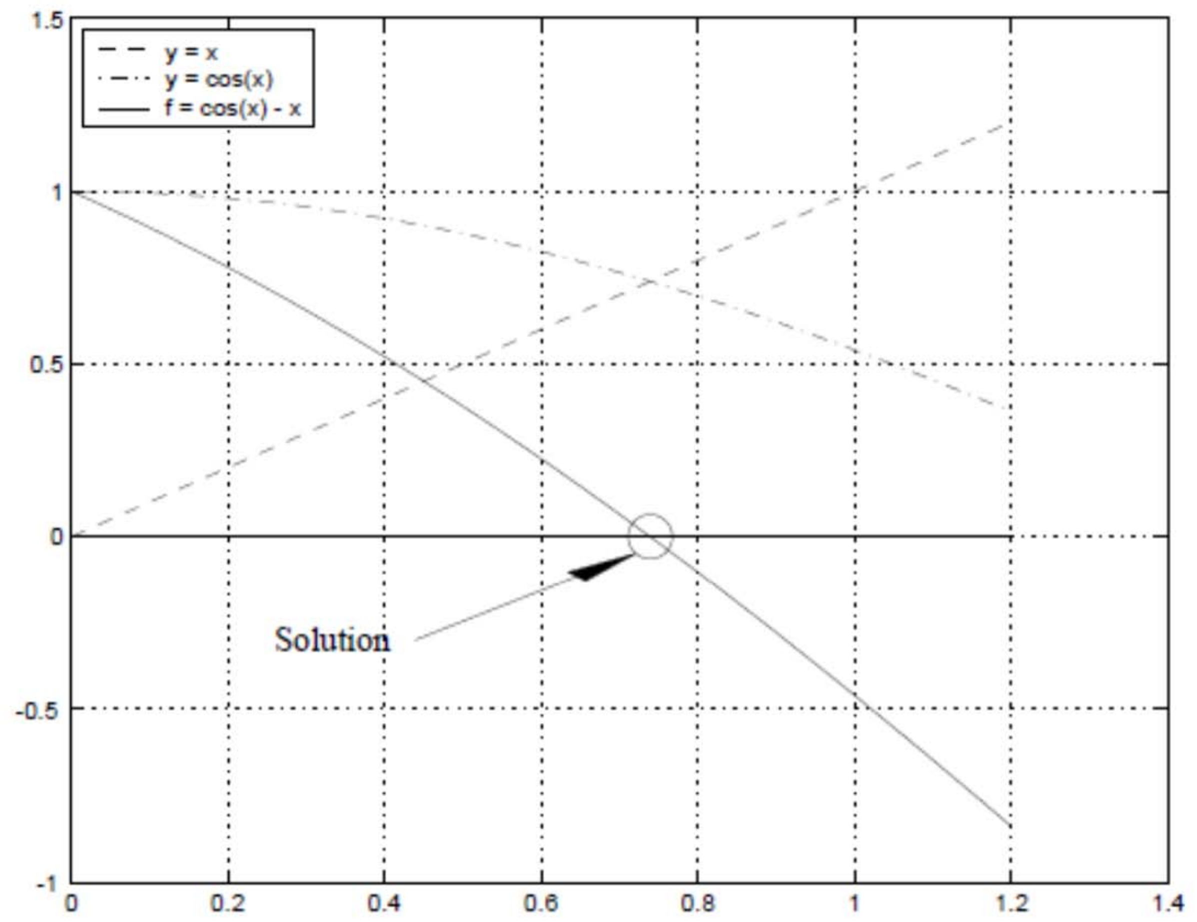
Any function of one variable can be put in the form $f(x) = 0$.

Example:
To find the x that satisfies

$$\cos(x) = x,$$

find the zero crossing of

$$f(x) = \cos(x) - x = 0$$



General Considerations

- Is this a special function that will be evaluated often?

  If so, some analytical study and possibly some algebraic manipulation of the function will be a good idea. Devise a custom iterative scheme.
- How much precision is needed?

  Engineering calculations often require only a few significant figures.
- How fast and robust must the method be?

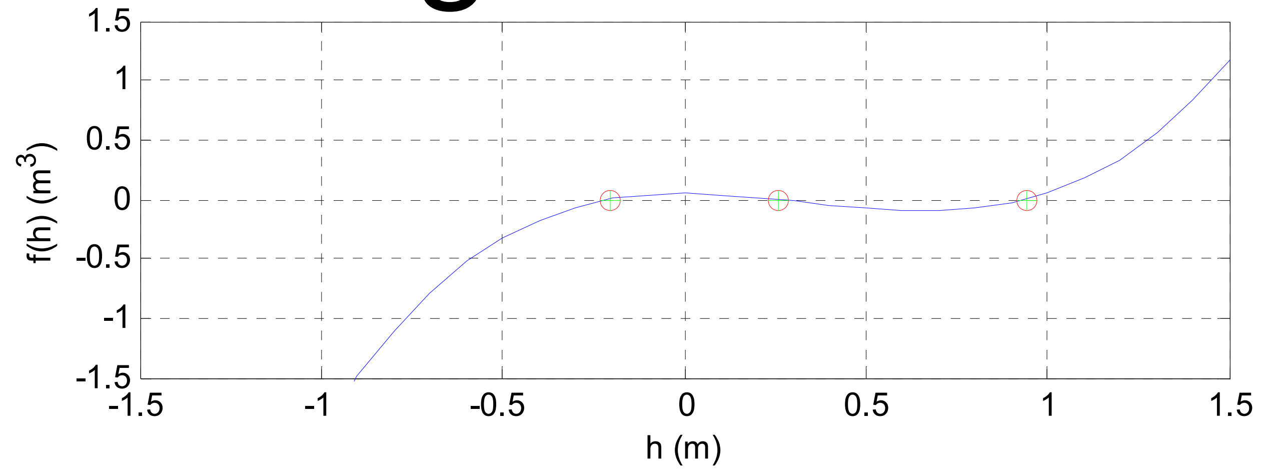
  A robust procedure is relatively immune to the initial guess, and it converges quickly without being overly sensitive to the behavior of f(x) in the expected parameter space.
- Is the function a polynomial?

  There are special procedure for finding the roots of polynomials.
- Does the function have singularities?

  Some root finding procedures will converge to a singularity, as well as converge to a root.

There is no single root-finding method that is best for all situations.

The Basic Root-Finding Procedure



The basic strategy is

1. Plot the function.

   The plot provides an initial guess, and an indication of potential problems.

2. Select an initial guess.

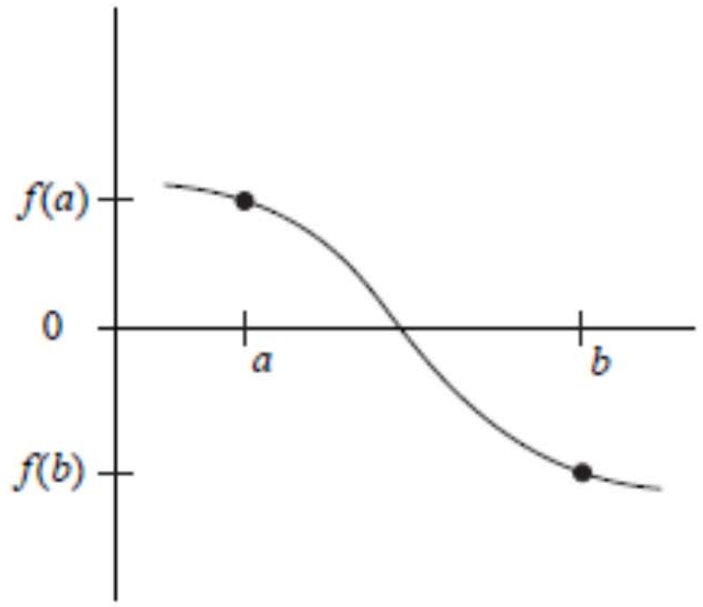
3. Iteratively refine the initial guess with a root-finding algorithm.
   - The result is a sequence of values that, if all goes well, get progressively closer to the root.
   - If x_k is the estimate of the root, ξ, on the *k*-th iteration, then the iterations converge if $\lim_{k\to\infty} x_k = \xi$.

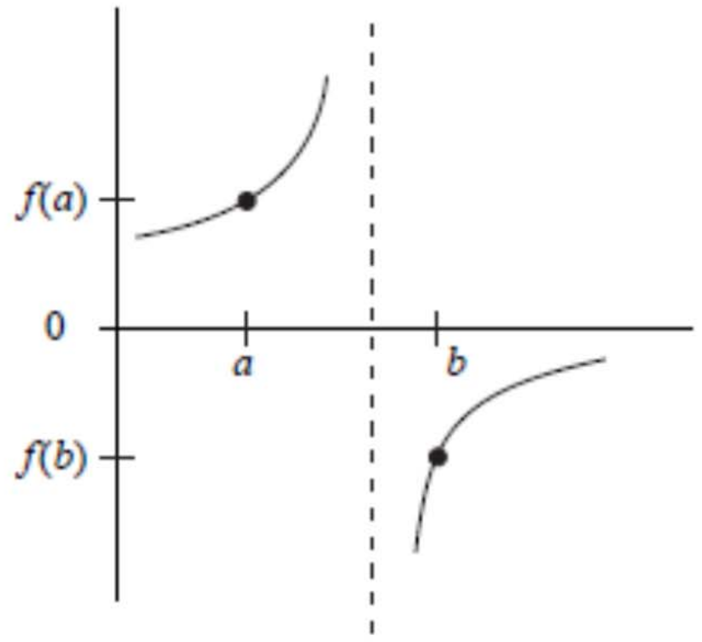
The Basic Root-Finding Procedure

3. Iteratively refine the initial guess with a root-finding algorithm.
   - …
   - For practical reasons, it is desirable that the root finder converges in as few steps as possible.
   - Full automation of the root finding algorithm also requires a way to determine when to stop the iteration.
4. After a root has been found with an automatic root-finding procedure, it is a good idea to substitute the root back into the function to see whether $f(x) \approx 0$.

Bracketing

- Bracketing is a procedure to perform a coarse level search for roots over a large interval on the x-axis.
- The result is the identification of a set of subintervals that are likely to contain roots.
- Bracketing answers the question, “Where might the roots be?”
- A root is bracketed on the interval [a, b] if f(a) and f(b) have opposite sign.
- A sign change occurs for singularities as well as roots.
- Bracketing is used to make initial guesses at the roots, not to accurately estimate the values of the roots.





Algorithm 6.1 Bracket Roots

```
given: $f(x)$, $x_{\min}$, $x_{\max}$, $n$

$dx = (x_{\max} - x_{\min})/n$
$x_{\text{left}} = x_{\min}$
i $= 0$
while $i < n$
   $i \leftarrow i + 1$
   $x_{\text{right}} = x_{\text{left}} + dx$
   if $f(x)$ changes sign in $[x_{\text{left}}, x_{\text{right}}]$
      save $[x_{\text{left}}, x_{\text{right}}]$ for further root-finding
   end
   $x_{\text{left}} = x_{\text{right}}$
end
```

Bracketing Algorithms

- A simple test for sign change: $f(a) \times f(b) < 0$?
  If so, save the bracket; otherwise, refine the interval.
- This test is susceptible to underflow. Try the MATLAB lines below

```
>>format long e
>>fa=1e-120; fb=-1e-300;
>>fa*fb
```

Bracketing Algorithms

- This test is susceptible to underflow. Try the MATLAB lines below

```
>>format long e
>>fa=1e-120; fb=-1e-300;
>>fa*fb
```

- Solution:

```
>>sign(fa)~=sign(fb)
```

Root-Finding Algorithms

We now proceed to develop the following root-finding algorithms:

- Fixed point iteration
- Bisection
- Newton’s method
- Secant method

These algorithms are applied after initial guesses at the root(s) are identified with bracketing (or guesswork).

Fixed-Point Iteration

- Fixed point iteration is a simple method. It only works when the iteration function is convergent.
- As a simple root-finding procedure, fixed-point iteration is often used in hand calculations.

Given f(x) = 0, rewrite it as $x_{new} = g(x_{old})$. a value of x that satisfies x=g(x) will be a root of f(x).

Algorithm:

```
initialize: x0 = . . .
for k = 1, 2, . . .
$x_k = g(x_k-1)$
if converged, stop
end
```

- Starting from an initial guess, x_0, the iteration formula is simply: $x_1 = g(x_0), x_2 = g(x_1), \ldots, x_{n+1} = g(x_n)$,
- You would hope that the sequence $x_0, x_1, x_2, \ldots$ converges to the solution x.
- The fixed-point algorithm gets its name from its behavior at convergence: recycling the output as input does not change the output. The root, ξ, is a fixed point of g(x).

Iterative Equations

- There is generally more than one way in which a function can be written in the form $x = g(x)$.
- A problem one can run into with iterative solutions is finding a form of $g(x)$ that will converge to any particular root.

$$ax^3 + bx^2 + cx + d = 0$$

can be rewritten as

$$x = -\left(ax^3 + bx^2 + d\right) / c$$

or

$$x = \sqrt{-\left(ax^3 + cx + d\right) / b}$$

Classroom Exercise

- $f(x)=x^2-2x-3=0$
- True solutions are $x=-1$ and $x=3$
- $x=g_1(x)=\mathrm{sqrt}(2x+3)$
- $x=g_2(x)=3/(x-2)$
- $x=g_3(x)=(x^2-3)/2$
- Start from $x_0=4$, solve up to x_3.

Convergence of Fixed-Point Iteration

- Starting from an initial guess, x_0, the iteration formula is simply: $x_1 = g(x_0)$, $x_2 = g(x_1)$, …, $x_{n+1} = g(x_n)$,
- The function g is evaluated with successive values of x until $x_{n+1} - x_n$ is as small as desired.
- You would hope that the sequence x_0, x_1, x_2, … converges to the solution x.

Analysis of fixed-point iteration reveals that the iterations are convergent on the interval $a \leq x \leq b$ if the interval contains a root and if

$$\left|g'(x)\right| < 1, \quad \text{for all } x \in \{a \leq x \leq b\}$$

If $-1<g'(x)<0$, the iterations oscillate while converging. If $0<g'(x)<1$, the iterations converge monotonically.

Bisection

- The logic:

  Given a bracketed root, repeatedly halve the interval while continue to bracket the root.
- Since the bracket is always reduced by a factor of two, bisection is sometimes called interval halving.
- Given the initial bracket [a,b], an improved approximation to the root is the midpoint of the interval $x_m=1/2(a+b)$.
- An alternative formula for the midpoint that is less susceptible to roundoff error is

  $x_m=a+1/2(b-a)$.

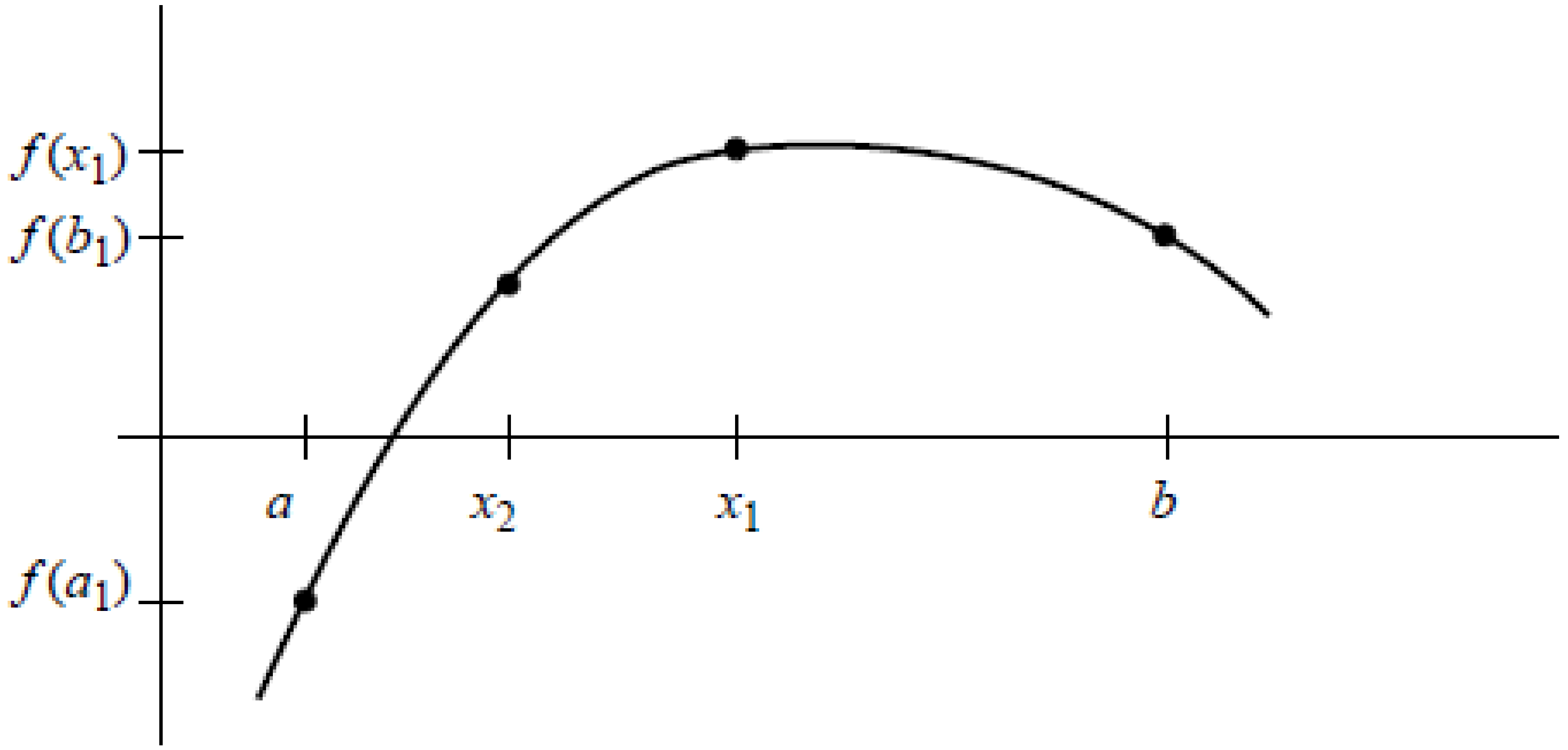
Algorithm

```
initialize: $a = \ldots, b = \ldots$
for $k = 1, 2, \ldots$
    $x_m = a + (b - a)/2$
    if $\text{sign}\,(f(x_m)) = \text{sign}\,(f(x_a))$
        $a = x_m$
    else
        $b = x_m$
    end
    if converged, stop
end
```

- Starting from x=a and x=b
- If f(x=a)*f(x=b)<0, then [a,b] bracket the root
- The interval around the root is halved successively until the value of f(x) is close to 0
- With each halving, the root is kept between the bracketing values by paring the new value with the previous value that gives f(x) of the opposite sign.
- If f(x) is close enough to 0 or the bracket values are close enough with tolerance, stop.

Exercise

Use the figure below to explain how the bisection method works.



In comparison to the fixed point method, what are the pros and cons of the bisection method?

Bisection

- It is robust and, given initial values that bracket the root, the root will be found.
- In other words, bisection always converges if the original interval contained the root.
- It has the disadvantage of being slower to converge than many of the other methods.
- It is often used to find an approximate root to use as an initial value in some of the other more efficient techniques, such as Newton's method.

Solve with bisection:

$$x - x^{1/3} - 2 = 0$$

k	a	b	x_{mid}	$f(x_{mid})$
0	3	4		
1	3	4	3.5	-0.01829449
2	3.5	4	3.75	0.19638375
3	3.5	3.75	3.625	0.08884159
4	3.5	3.625	3.5625	0.03522131
5	3.5	3.5625	3.53125	0.00845016
6	3.5	3.53125	3.515625	-0.00492550
7	3.51625	3.53125	3.5234375	0.00176150
8	3.51625	3.5234375	3.51953125	-0.00158221
9	3.51953125	3.5234375	3.52148438	0.00008959
10	3.51953125	3.52148438	3.52050781	-0.00074632

Analysis of Bisection

Let δ_n be the size of the bracketing interval at the n^{th} stage of bisection. Then

$$\delta_0 = b - a = \text{initial bracketing interval}$$

$$\delta_1 = \frac{1}{2}\delta_0$$

$$\delta_2 = \frac{1}{2}\delta_1 = \frac{1}{4}\delta_0$$

$$\vdots$$

$$\delta_n = \left(\frac{1}{2}\right)^n \delta_0$$

$$\implies \qquad \frac{\delta_n}{\delta_0} = \left(\frac{1}{2}\right)^n = 2^{-n}$$

or $$n = \log_2\left(\frac{\delta_n}{\delta_0}\right)$$

$$\frac{\delta_n}{\delta_0} = \left(\frac{1}{2}\right)^n = 2^{-n} \qquad \text{or} \qquad n = \log_2\left(\frac{\delta_n}{\delta_0}\right)$$

n	$\frac{\delta_n}{\delta_0}$	function evaluations
5	3.1×10^{-2}	7
10	9.8×10^{-4}	12
20	9.5×10^{-7}	22
30	9.3×10^{-10}	32
40	9.1×10^{-13}	42
50	8.9×10^{-16}	52

Convergence Criteria

- An automatic root-finding procedure needs to monitor progress toward the root and stop when current guess is close enough to the desired root.
  - It is extremely unlikely that a numerical procedure will find the precise value of x that makes f(x) exactly zero in floating-point arithmetic.
  - The algorithm must decide how close to the root the iteration should be before stopping the search.
- Convergence checking can consider whether two successive approximations to the root are close enough to be considered equal.
- Convergence checking can examine whether f(x) is sufficiently close to zero at the current guess.

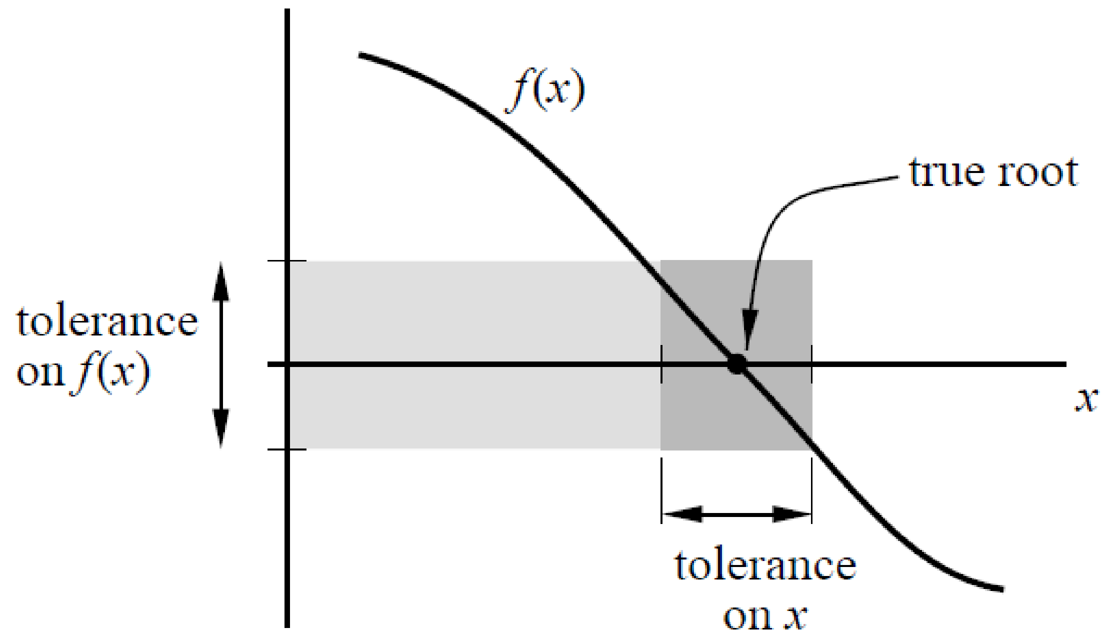
Convergence Criteria

- Convergence checking will avoid searching to unnecessary accuracy.
  - As a practical consideration, it may not be necessary to know the root to a large number of significant figures.
  - Because the computer has finite resolution on the real number line, there is some limit below which refinements in the root cannot be made.
  - Even if the true root is used to evaluate f(x), the result may not be zero, due to roundoff error.

Convergence Criteria

- An automatic root-finding procedure needs to monitor progress toward the root and stop when current guess is close enough to the desired root.
- Convergence checking will avoid searching to unnecessary accuracy.
- Two criteria can be applied in testing for convergence:
  - Check whether successive approximations are close enough to be considered the same: $|x_k - x_{k-1}| < \delta x$
  - Check whether f(x) is close enough zero: $|f(x_k)| < \delta f$
  - You need to specify the two tolerance.

Convergence Criteria on x



$$x_k = \text{current guess at the root}$$

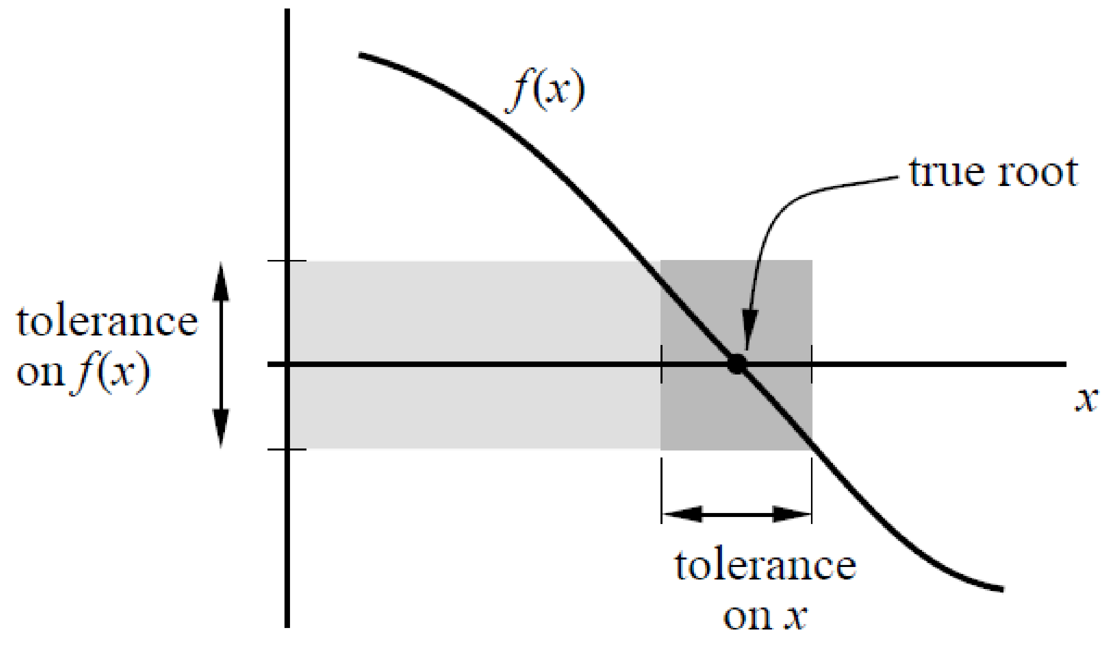
$$x_{k-1} = \text{previous guess at the root}$$

Absolute tolerance: $\left| x_k - x_{k-1} \right| < \delta_x$

Relative tolerance: $\left| \dfrac{x_k - x_{k-1}}{b - a} \right| < \hat{\delta}_x$

- You may choose the extreme value of δ_x=eps, the machine precision.
- The absolute tolerance is workable but highly restricted.
- When $|x_k|>>1$, the convergence test using the absolute tolerance will never be satisfied, because the roundoff error in the subtraction of$|x_k-x_{k-1}|$ will always be larger than the specified tolerance.
- The absolute error is useful when the limit of the iterative sequence is known to be near zero.

Convergence Criteria on x



$$x_k = \text{current guess at the root}$$

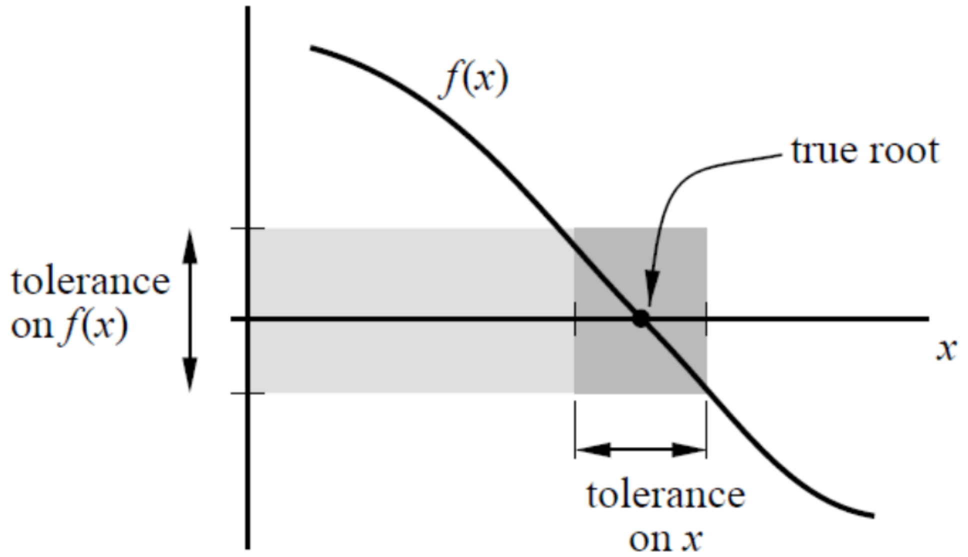
$$x_{k-1} = \text{previous guess at the root}$$

Absolute tolerance: $\left| x_k - x_{k-1} \right| < \delta_x$

Relative tolerance: $\left| \dfrac{x_k - x_{k-1}}{b - a} \right| < \hat{\delta}_x$

- The relative tolerance is a measure of the relative reduction in the original bracket interval.
- The relative convergence criterion is always preferred, since it does not change if the problem is suitably scaled.
- In computer calculation, it is usually a good idea to report both relative and absolute errors.

Convergence Criteria on f(x)



- Machine precision is the lower bound for both criteria. Using convergence criteria less than the machine precision may result in infinite loop.
- One way to specify the tolerances is to use multiples of ε_m.
- Extreme tight convergence criteria would be $\delta x = 5\varepsilon_m$ and $\delta f = 5\varepsilon_m$.

Absolute tolerance: $\left|f(x_k)\right| < \delta_f$

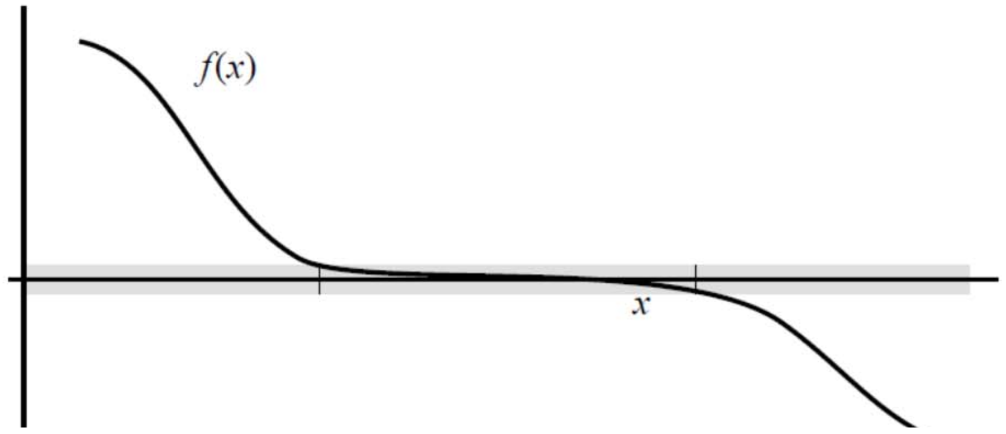
Relative tolerance:

$$|f(x_k)| < \hat{\delta}_f \max\Big\{|f(a_0)|, |f(b_0)|\Big\}$$

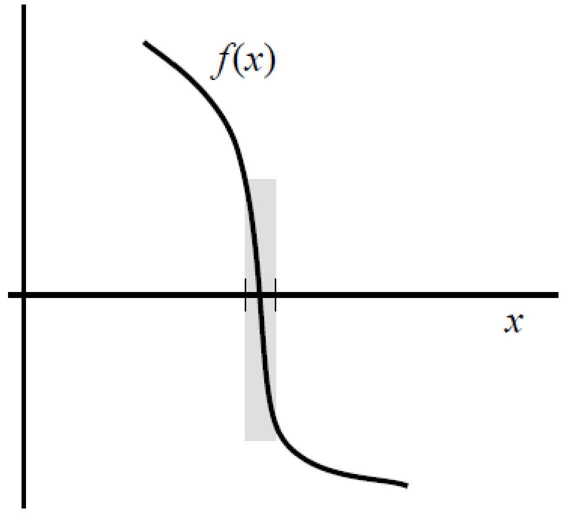
where a_0 and b_0 are the original brackets

Convergence Criteria on f(x)

If $f'(x)$ is small near the root, it is easy to satisfy tolerance on $f(x)$ for a large range of Δx. The tolerance on Δx is more conservative

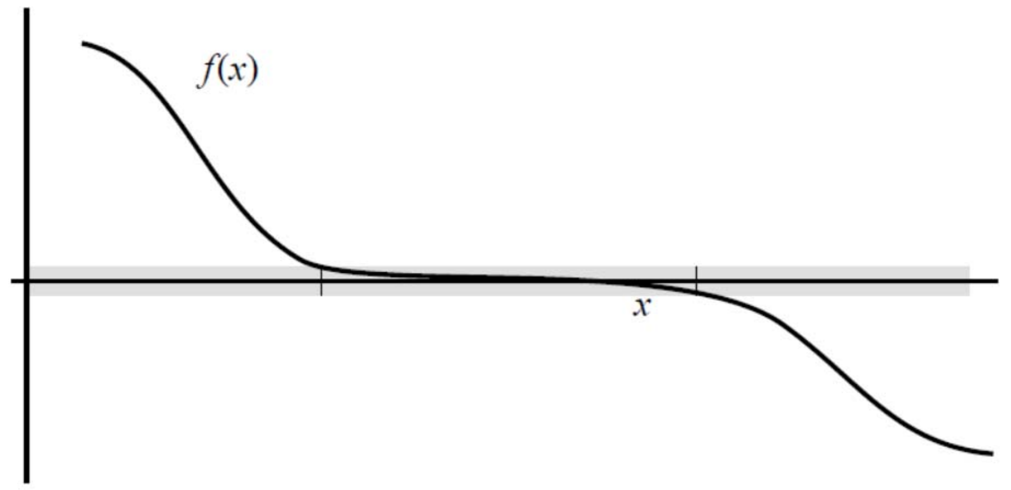


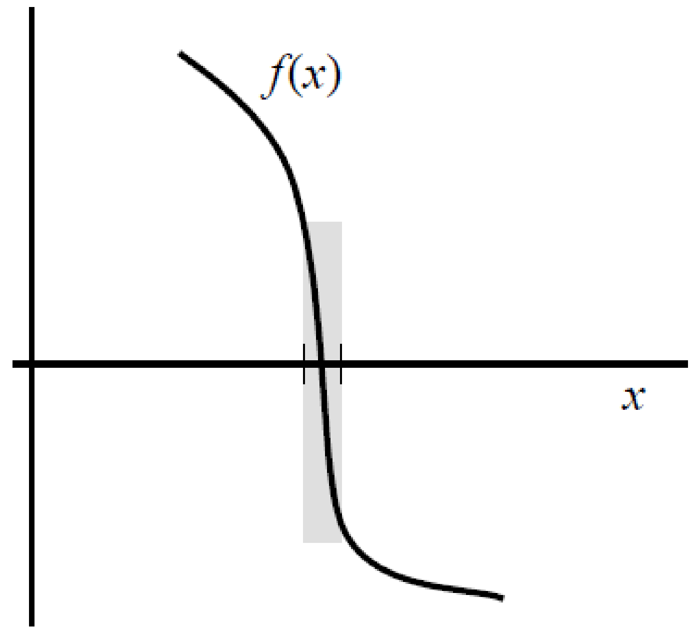
If $f'(x)$ is large near the root, it is possible to satisfy the tolerance on Δx when $|f(x)|$ is still large. The tolerance on $f(x)$ is more conservative



Exercise

How do you measure computational time of an algorithm?





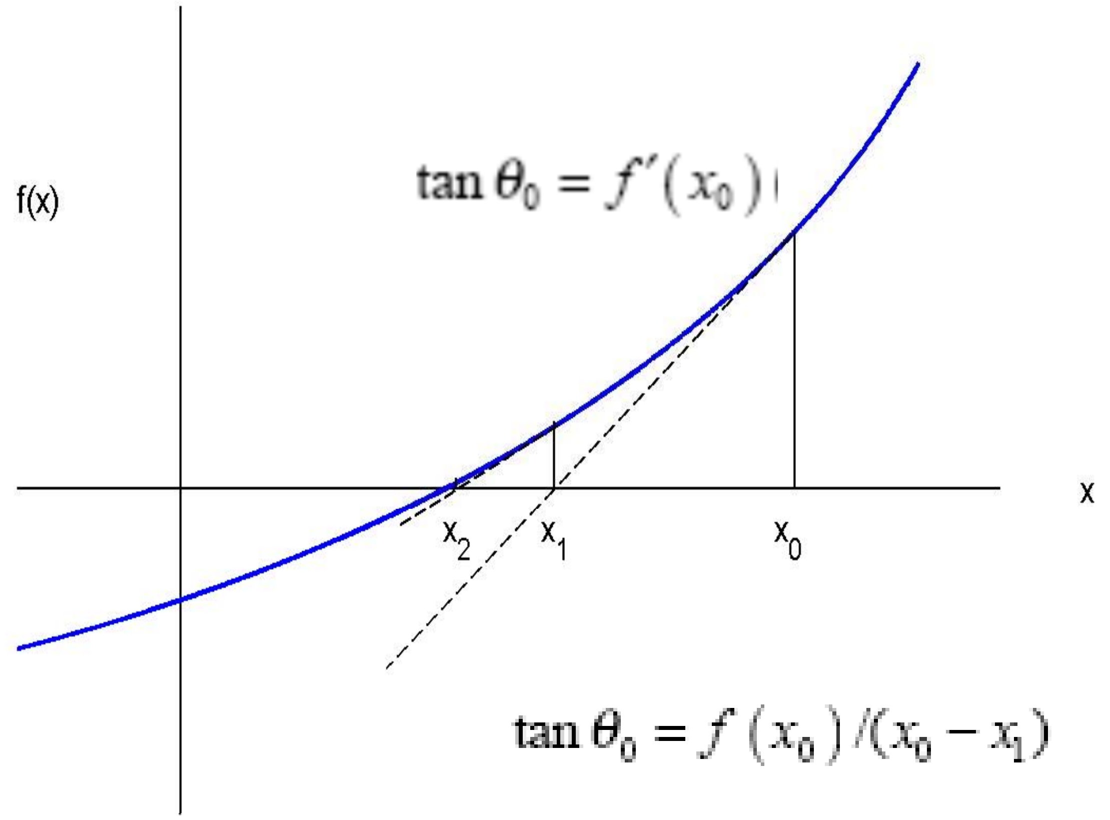
The figures above demonstrate two situations:

- The function value f(x) is insensitive to x, but the root is sensitive.
- The function value f(x) is sensitive to the x, but the root is insensitive.

Under the two circumstances, what is the conservative way to choose the convergence criteria, tolerance on f(x) or Δx?

Newton's Method

- Newton's method, sometimes called the Newton-Raphson method, uses information about the function, f(x), and its first derivative.
- Based on the idea that any function can be approximated by a straight line over some small interval – a fact that is taken advantage of over and over again in numerical solution techniques.
- Begin with an initial estimate x_0 that is close to the real root.



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

Repeating the process at $x = x_1$ gives $x_2 = x_1 - \dfrac{f(x_1)}{f'(x_1)}$

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \qquad \text{for} \quad n=0,1,2\ldots$$

Expand $f(x)$ in Taylor Series around x_k

$$f(x_k + \Delta x) = f(x_k) + \Delta x \left.\frac{df}{dx}\right|_{x_k} + \frac{(\Delta x)^2}{2} \left.\frac{d^2 f}{dx^2}\right|_{x_k} + \dots$$

Substitute $\Delta x = x_{k+1} - x_k$

and neglect 2^{nd} order terms to get

A linear function

$$f(x_{k+1}) \approx f(x_k) + (x_{k+1} - x_k)\, f'(x_k)$$

where

$$f'(x_k) = \left.\frac{df}{dx}\right|_{x_k}$$

Solve the linear function

Goal is to find x such that $f(x) = 0$.

Set $f(x_{k+1}) = 0$ and solve for x_{k+1}

$$0 = f(x_k) + (x_{k+1} - x_k)\, f'(x_k)$$

or, solving for x_{k+1}

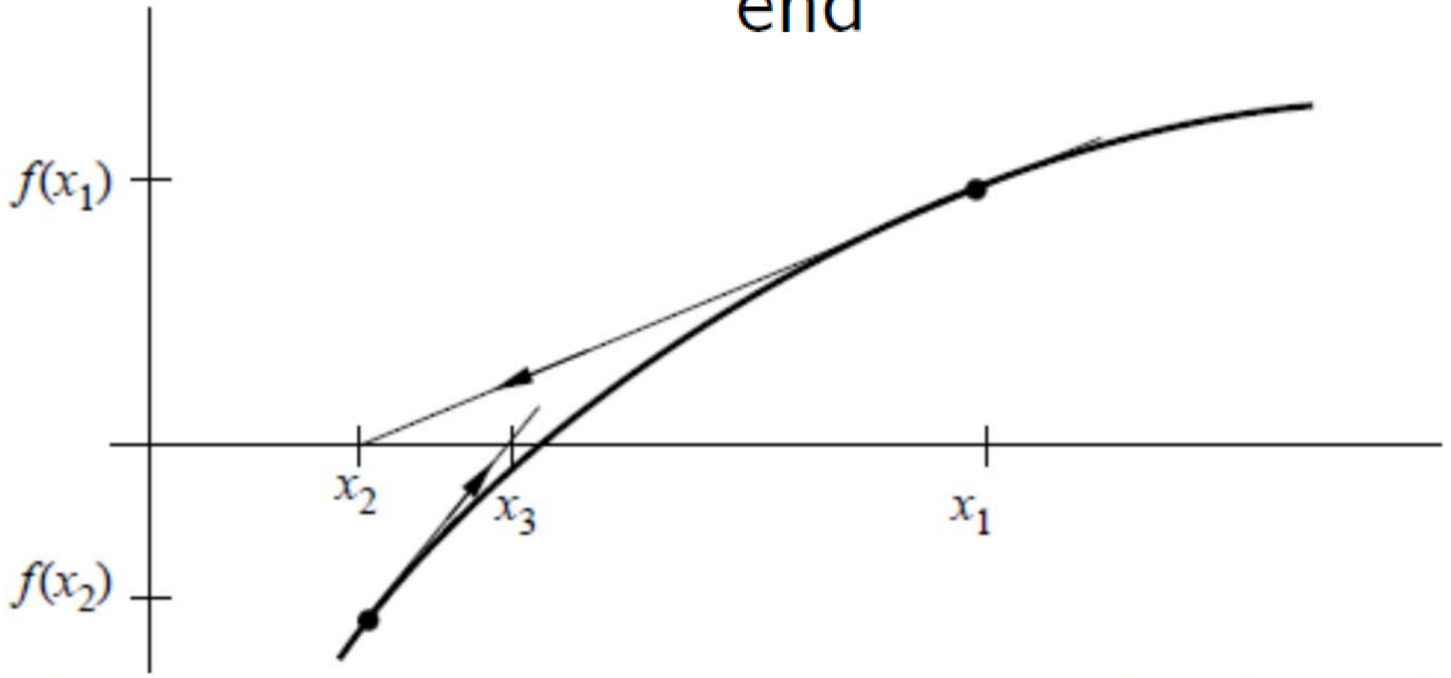
$$\boxed{x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}}$$

Exercise

Explain the figure for the Newton's method

Algorithm:

```
initialize: $x_1 = \ldots$
for $k = 2, 3, \ldots$
    $x_k = x_{k-1} - f(x_{k-1})/f'(x_{k-1})$
    if converged, stop
end
```



What quantities do you need for implementing the Newton's method?

For a current guess x_k, use $f(x_k)$ and the slope $f'(x_k)$ to predict where $f(x)$ crosses the x axis.

Exercise

Solve

$$f(x) = x - x^{1/3} - 2 = 0$$

using the Newton’s method by hand.
Use x_0=3 and calculate to x_4.

Solve:

$$x - x^{1/3} - 2 = 0$$

First derivative is

$$f'(x) = 1 - \frac{1}{3}x^{-2/3}$$

The iteration formula is

$$\boxed{x_{k+1} = x_k - \frac{x_k - x_k^{1/3} - 2}{1 - \frac{1}{3}x_k^{-2/3}}}$$

Relation between Newton's method and fix-point method?

k	x_k	$f'(x_k)$	$f(x)$
0	3	0.83975005	-0.44224957
1	3.52664429	0.85612976	0.00450679
2	3.52138015	0.85598641	3.771×10^{-7}
3	3.52137971	0.85598640	2.664×10^{-15}
4	3.52137971	0.85598640	0.0

Exercise

- Construct g_3 function using the Newton's method.
- Write a MATLAB code to solve this problem.

$$f(x) = x - x^{1/3} - 2 = 0$$

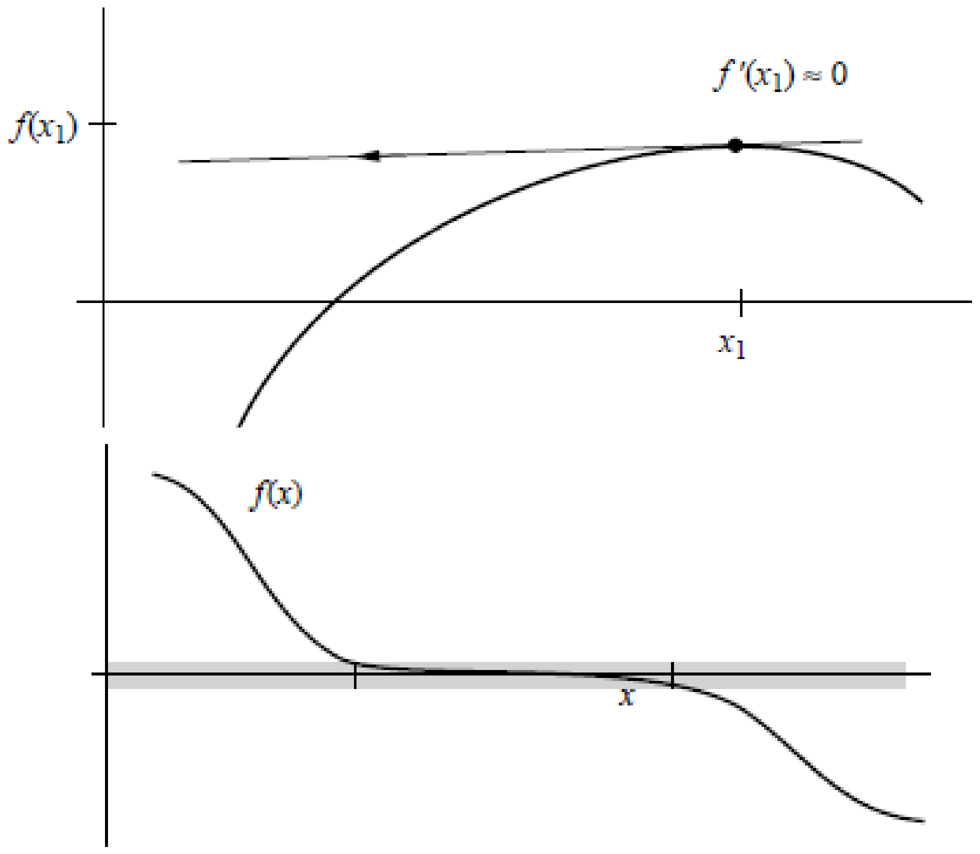
$$g_1(x) = x^{1/3} + 2$$

$$g_2(x) = (x-2)^3$$

$$g_3 = \frac{6 + 2x^{1/3}}{3 - x^{-2/3}}$$

k	$g_1(x_{k-1})$	$g_2(x_{k-1})$	$g_3(x_{k-1})$
0	3	3	3
1	3.4422495703	1	3.5266442931
2	3.5098974493	-1	3.5213801474
3	3.5197243050	-27	3.5213797068
4	3.5211412691	-24389	3.5213797068
5	3.5213453678	-1.451×10^{13}	3.5213797068
6	3.5213747615	-3.055×10^{39}	3.5213797068
7	3.5213789946	-2.852×10^{118}	3.5213797068
8	3.5213796042	∞	3.5213797068
9	3.5213796920	∞	3.5213797068

Divergence of Newton's Method



Since

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

the new guess, x_{k+1}, will be far from the old guess whenever $f'(x_k) \approx 0$

- If the first derivative of f(x) vanishes, then the procedure fails.
- Even though a good initial guess to the root is available, Newton's method may diverge if the first derivative of the function approaches zero near the current estimate of the root.

Newton’s and Secant Methods

- Limitation:

  Newton's method generally works well for cases in which the derivative is known in advance, such as polynomials or other functions with straightforward derivatives.
- Secant method

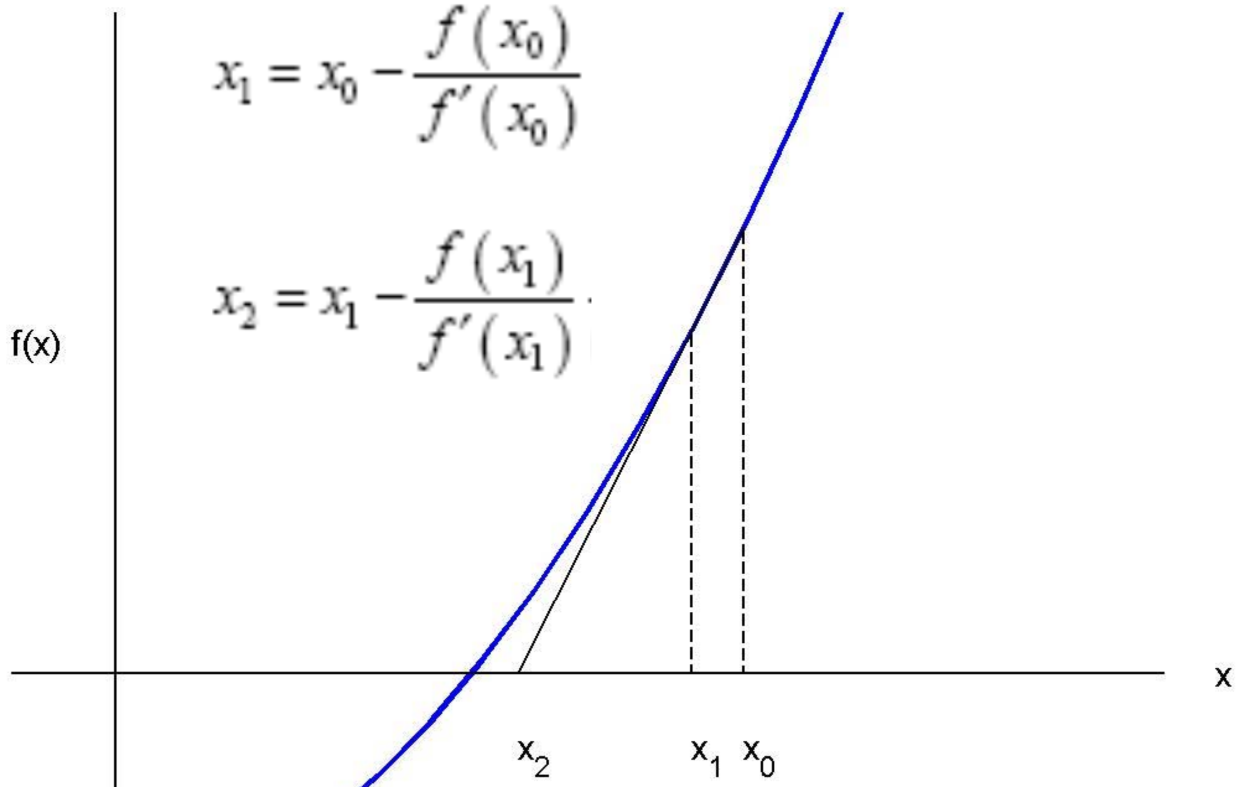
  Use finite difference to approximate the derivative at x_0.
- The secant method uses the last two guesses at the root to estimate the slope.

Secant Method

- Analytical expression of f'(x) is unknown
- Starting from x_1
- Approximate derivative by finite difference

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

for n=1, 2, 3…



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

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

$$f'(x) = \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

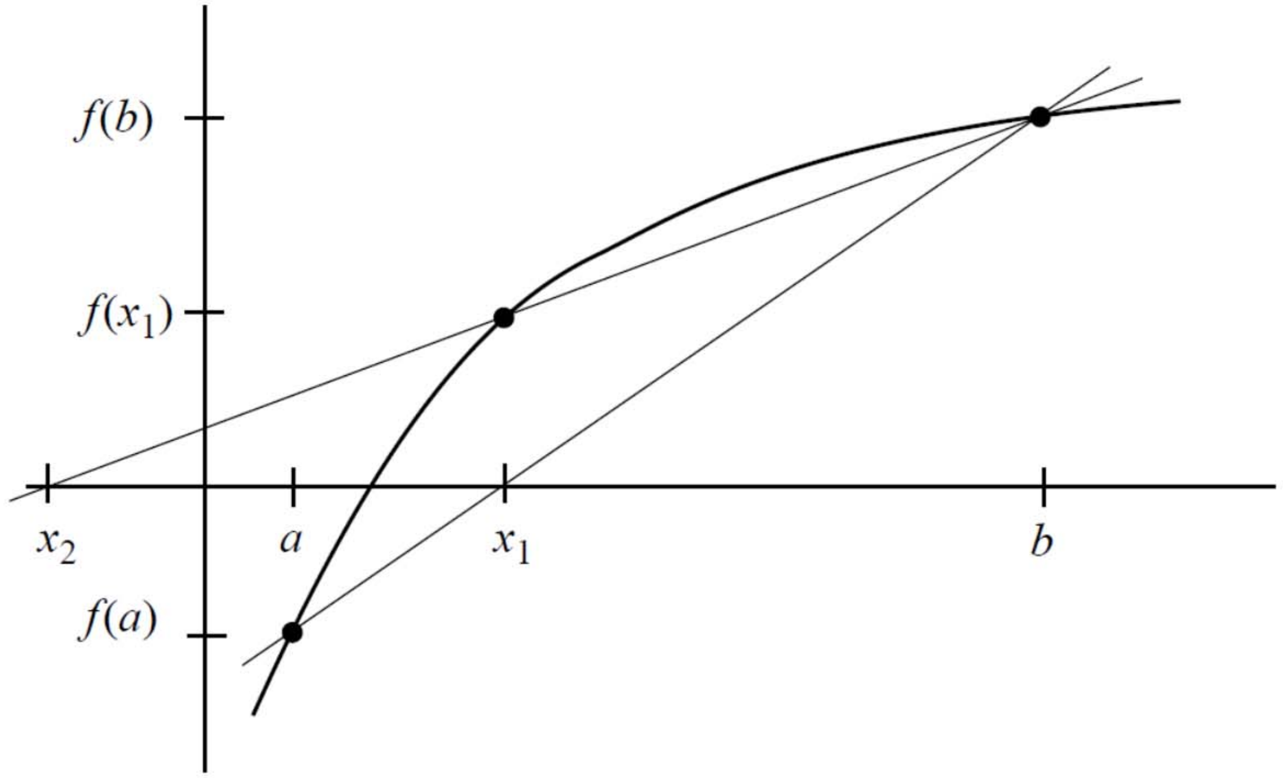
$$\frac{f(x_1) - f(x_0)}{x_1 - x_0} = \frac{\Delta f}{\Delta x}$$

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

Exercise

Use the figure to explain the algorithm.

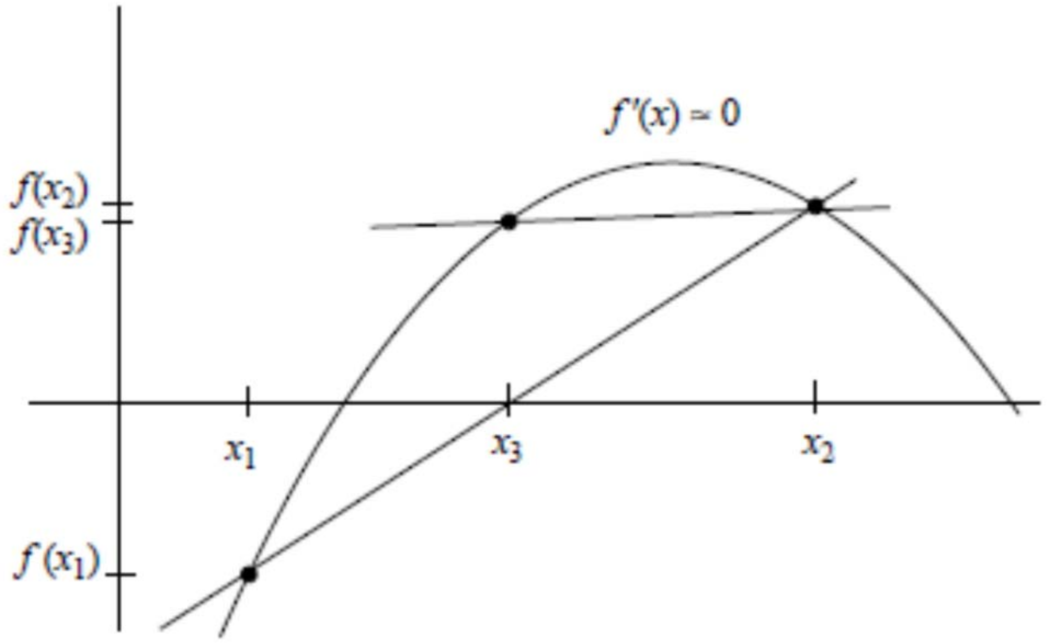
```
initialize: $x_1 = \ldots$, $x_2 = \ldots$
for $k = 2, 3 \ldots$
    $x_{k+1} = x_k$
            $- f(x_k)(x_k - x_{k-1})/(f(x_k) - f(x_{k-1}))$
    if converged, stop
end
```



Given two guesses x_{k-1} and x_k, the next guess at the root is where the line through $f(x_{k-1})$ and $f(x_k)$ crosses the x axis.

Secant Method

- Secant method still needs a good initial guess; otherwise, it may diverge.
- It may also diverge if the numerical estimation of f' is close to zero.



$$x_{k+1} = x_k - f(x_k)\left[\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}\right]$$

The new guess, x_{k+1}, will be far from the old guess whenever $f(x_k) \approx f(x_{k-1})$ and $|f(x)|$ is not small.

Two versions of this formula are (equivalent in exact math)

$$x_{k+1} = x_k - f(x_k)\left[\frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}\right] \qquad (\star)$$

and

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

Equation $(\star)$ is better since it is of the form $x_{k+1} = x_k + \Delta$. Even if Δ is inaccurate the change in the estimate of the root will be small at convergence because $f(x_k)$ will also be small.

Equation $(\star\star)$ is susceptible to catastrophic cancellation:

- $f(x_k) \rightarrow f(x_{k-1})$ as convergence approaches, so cancellation error in denominator can be large.
- $|f(x)| \rightarrow 0$ as convergence approaches, so underflow is possible

Classroom Exercise

- Find the root using the Newton's method for the initial guess of x_1=4 and x_2=3.

$$f(x) = x - x^{1/3} - 2 = 0$$

k	x_{k-1}	x_k	$f(x_k)$
0	4	3	-0.44224957
1	3	3.51734262	-0.00345547
2	3.51734262	3.52141665	0.00003163
3	3.52141665	3.52137970	-2.034×10^{-9}
4	3.52137959	3.52137971	-1.332×10^{-15}
5	3.52137971	3.52137971	0.0

Take-Home Exercise

- Let $f(x)=x^2-6$ and $x_0=1$, use Newton's method to find x_2.
- Let $f(x)=x^2-6$. With $x_0=3$ and $x_1=2$, use Secant method to find x_2.

Summary of Basic Root-Finding Methods

- Plot f(x) before searching for roots
- Bracketing finds coarse interval containing roots and singularities
- Bisection is robust, but converges slowly
- Newton’s Method
  - Requires f(x) and f′(x).
  - Iterates are not confined to initial bracket.
  - Converges rapidly.
  - Diverges if $f'(x) \approx 0$ is encountered.
- Secant Method
  - Uses f(x) values to approximate f′(x).
  - Iterates are not confined to initial bracket.
  - Converges almost as rapidly as Newton’s method.
  - Diverges if $f'(x) \approx 0$ is encountered.

Hybrid Methods

- The root-finding methods discussed so far involve straightforward iterative algorithms that are either robust (e.g., bisection) or converge rapidly (e.g., Newton’s method or the secant method).
- At the expense of increased programming logic, it is possible to combine these methods.
- A hybrid method might combine bisection with a more rapidly converging technique.
- At each iteration, a preliminary step of the faster method is taken. If the resulting estimate of the root is within the original bracket, then it is kept. Otherwise, a bisection step is taken.
- Such a hybrid algorithm converges no more slowly than bisection, and, with luck, it converges much more quickly.
- The advantage of a hybrid technique is that a rather coarse initial guess can be used at the root without worrying that the method will diverge.

The fzero Function

fzero chooses next root as

- Result of reverse quadratic interpolation (RQI) if that result is inside the current bracket.
- Result of secant step if RQI fails, and if the result of secant method is in inside the current bracket.
- Result of bisection step if both RQI and secant method fail to produce guesses inside the current bracket.

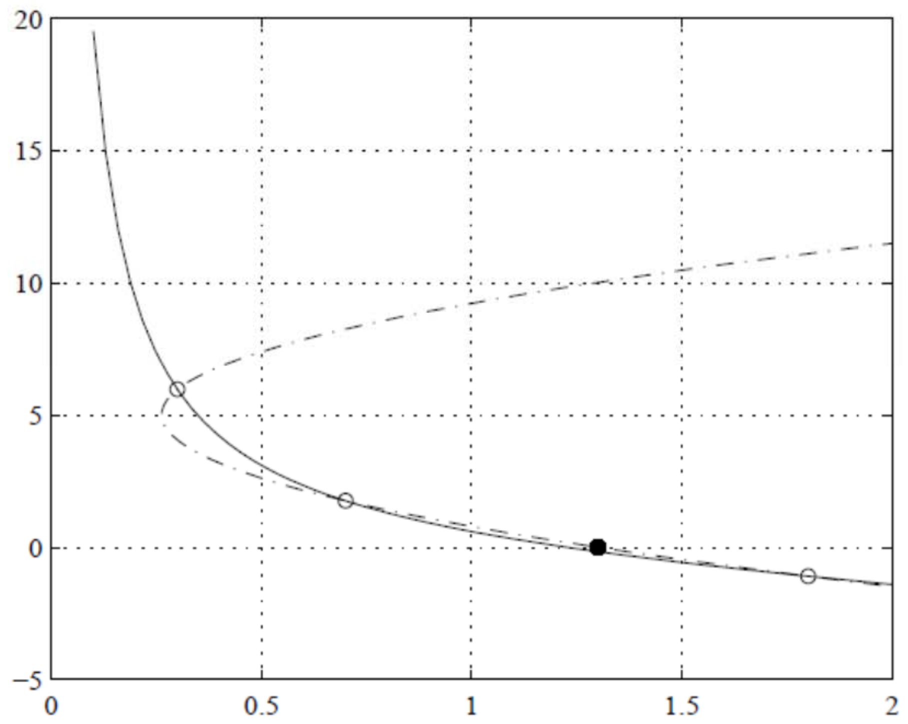
The secant method uses two previous points to get the next one, so why not use three?

Suppose we have three values, a, b, and c, and corresponding function values, $f(a)$, $f(b)$, and $f(c)$. We could interpolate these values by a parabola, a quadratic function of x, and take the next iterate to be the point where the parabola intersects the x-axis. The difficulty is that the parabola might not intersect the x-axis; a quadratic function does not necessarily have real roots. This could be regarded as an advantage. An algorithm known as Muller's method uses the complex roots of the quadratic to produce approximations to complex zeros of $f(x)$. But, for now, we want to avoid complex arithmetic.

Instead of a quadratic in x, we can interpolate the three points with a quadratic function in y. That's a "sideways" parabola, $P(y)$, determined by the interpolation conditions

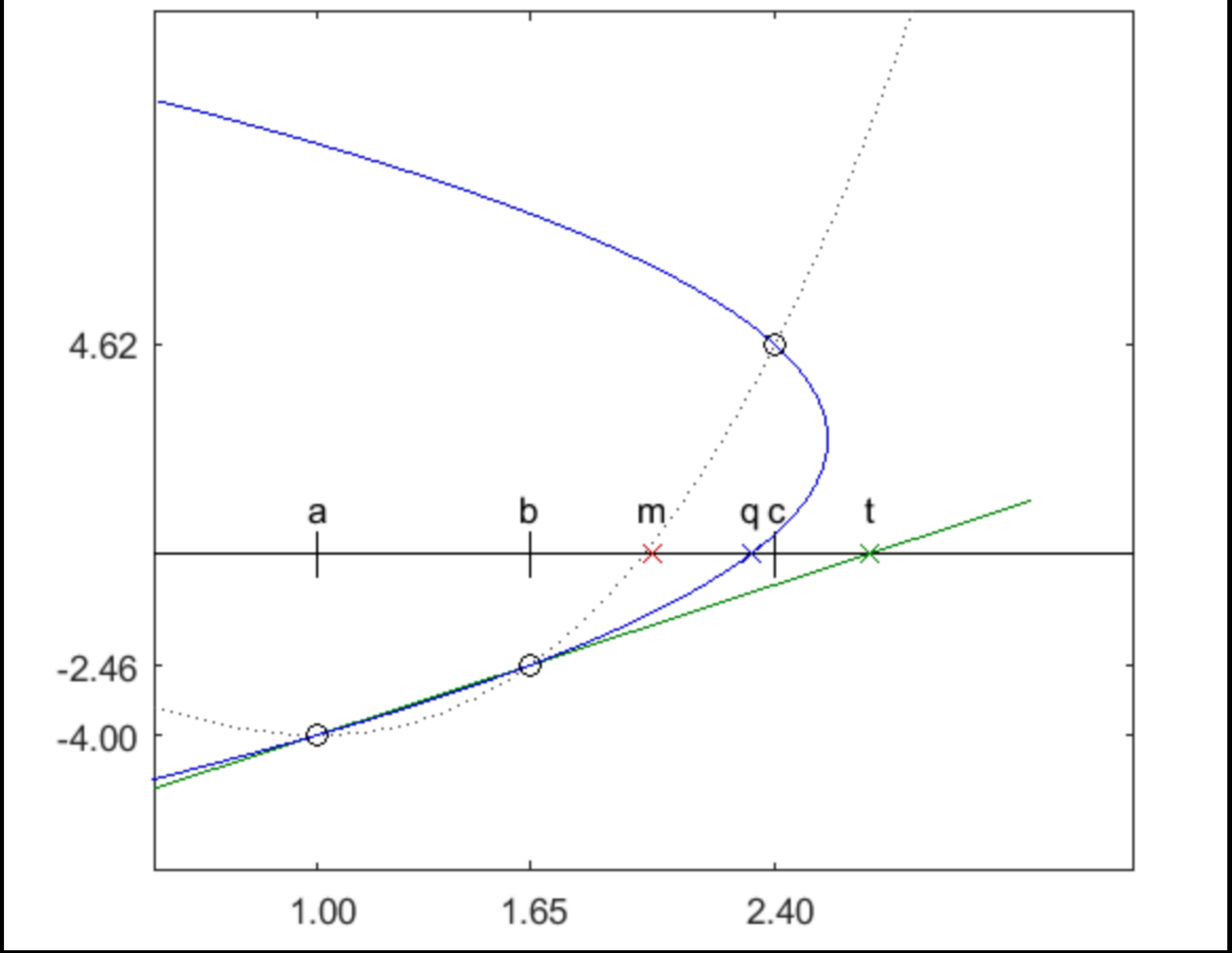
$$a = P(f(a)),\ b = P(f(b)),\ c = P(f(c)).$$

This parabola always intersects the x-axis, which is $y = 0$. So $x = P(0)$ is the next iterate.



Find the point where the sideways parabola passing through three pairs of $(x, f(x))$ values crosses the x axis.

Brent's zeroin



The fzero Function

- fzero is a hybrid method that combines bisection, secant and reverse quadratic interpolation
- Syntax:

  r = fzero('fun',x0)

  r = fzero('fun',x0,options)

  r = fzero('fun',x0,options,arg1,arg2,...)
- x0 can be a scalar or a two element vector
- If x0 is a scalar, fzero tries to create its own bracket.
- If x0 is a two element vector, fzero uses the vector as a bracket.

The fzero Function

- Optional parameters to control fzero are specified with the optimset function.

Examples:

- Tell fzero to display the results of each step:

```
>> options = optimset('Display','iter');
>> x = fzero('myFun',x0,options)
```

- Tell fzero to use a relative tolerance of 5×10^{-9}:

```
>> options = optimset('TolX',5e-9);
>> x = fzero('myFun',x0,options)
```

- Tell fzero to suppress all printed output, and use a relative tolerance of 5×10^{-4}:

```
>> options = optimset('Display','off','TolX',5e-4);
>> x = fzero('myFun',x0,options)
```

The fzero Function

Allowable options:

Option type	Value	Effect
'Display'	'iter'	Show results of each iteration
	'final'	Show root and original bracket
	'off'	Suppress all print out
'TolX'	tol	Iterate until $\lvert\Delta x\rvert < \max\left[\texttt{tol}, \texttt{tol} * \texttt{a}, \texttt{tol} * \texttt{b}\right]$ where $\Delta x = (b-a)/2$, and $[a, b]$ is the current bracket.

The default values of 'Display' and 'TolX' are equivalent to

```
options = optimset('Display','iter','TolX',eps)
```

Classroom Exercise

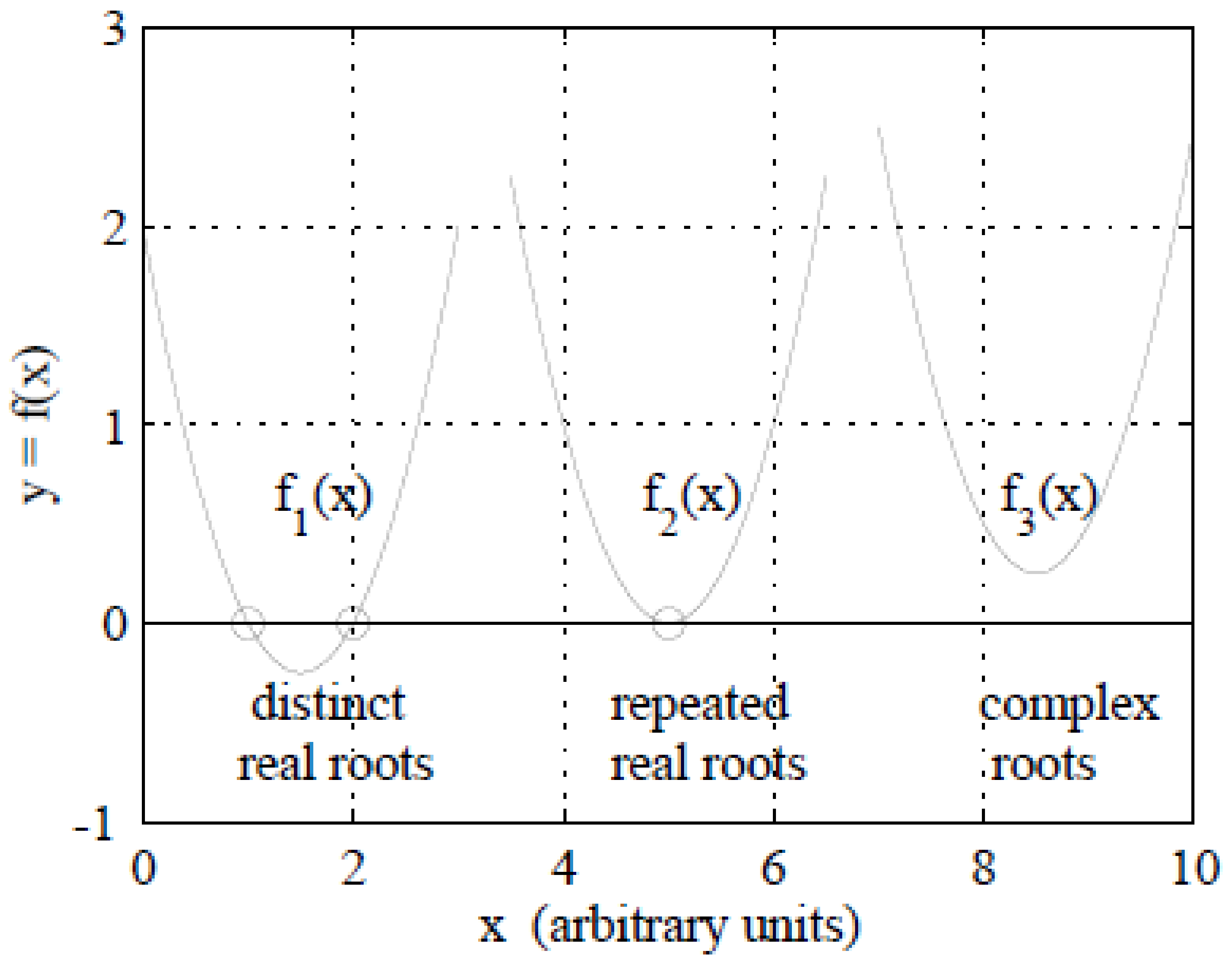
Use fzero to solve

$$f(x) = x - x^{1/3} - 2 = 0$$

(1) Use $0 \leq x \leq 5$ as x_0
(2) Use $x_0 = 4.5$

Roots of Polynomials

- The routines discussed up to this points can be applied to polynomials, because a polynomial in x can always be put in the form f(x)=0.
- These methods have difficulty due to
  - Repeated roots
  - Complex roots
  - Sensitivity of roots to small perturbations in the polynomial coefficients (conditioning).



$f_1(x)=x^2-3x+2$: has two distinct roots

$f_2(x)=x^2-10x+25$: has repeated real roots

$f_3(x)=x^2-17x+72.5$: has complex roots

Algorithms for Finding Polynomial Roots

- To find the roots of a polynomial, it is better to use specialized numerical algorithms that are designed to deal with the complications:
- Bairstow's method
- Muller's method
- Laguerre's method
- Jenkin's–Traub method
- Companion matrix method

The root Function

The built-in **roots** function uses the companion matrix method

- No initial guess
- Returns *all* roots of the polynomial
- Solves eigenvalue problem for companion matrix

Write polynomial in the form

$$c_1 x^n + c_2 x^{n-1} + \ldots + c_n x + c_{n+1} = 0$$

Then, for a *third* order polynomial

```
>> c = [c1 c2 c3 c4];
>> r = roots(c)
```

Exercise

Use the roots function to find the roots of

$$f_1(x) = x^2 - 3x + 2$$

$$f_2(x) = x^2 - 10x + 25$$

$$f_3(x) = x^2 - 17x + 72.5$$

Convergence Rates of Iterative Methods

- Unlike linear equations, most nonlinear equations cannot be solved in a finite number of steps.
- We must usually resort to an iterative method that produces increasingly accurate approximation to the solution, and we terminate the iteration when the result is sufficiently accurate.
- The total cost of solving the problem depends on both the cost per iteration and the number of iterations required for convergence, and there is often a trade-off between the two factors.

Rate of Convergence

- http://en.wikipedia.org/wiki/Rate_of_convergence
- In numerical analysis, the speed at which a convergent sequence approaches its limit is called the **rate of convergence**.
- Although strictly speaking, a limit does not give information about any finite first part of the sequence, this concept is of practical importance if we deal with a sequence of successive approximations for an iterative method, as then typically fewer iterations are needed to yield a useful approximation if the rate of convergence is higher.
- This may even make the difference between needing ten or a million iterations.

Rate of Convergence

- Suppose that the sequence $\{x_k\}$ converges to the number L.
- We say that this sequence **converges linearly** to L, if there exists a number μ ∈ (0, 1) such that

$$\lim_{k\to\infty} \frac{|x_{k+1} - L|}{|x_k - L|} = \mu.$$

- We say that the sequence **converges with order *q*** for $q > 1$ to L if

$$\lim_{k\to\infty} \frac{|x_{k+1} - L|}{|x_k - L|^q} = \mu \text{ with } \mu > 0.$$

Convergence Rate

- We denote the error at iteration *k* by e_k, and it is usually given by $e_k = x_k - x^*$, where x_k is the approximate solution at iteration *k*, and x* is the true solution.
- Some methods for one-dimensional problems do not actually produce a specific approximate solution x_k, but merely an interval known to contain the solution, with the length of the interval decreasing as iterations proceed. <u>For such methods, we take e_k to be the length of the interval at iteration *k*.</u>
- In either case, a method is said to converge with rate *r* if $\lim_{k\to\infty} \frac{\|e_{k+1}\|}{\|e_k\|^r} = C$ for some finite nonzero C.
- Some particular case of interest are these:
  - If r=1 and C<1, the convergence rate is linear.
  - If 1<r<2, the convergence rate is superlinear.
  - If r=2 the convergence rate is quadratic.

Convergence Rate

- One way to interpret the distinction between linear and superlinear convergence is that, asymptotically, a linear convergent sequence gains a constant number of digits of accuracy per iteration, whereas a superlinearly convergent sequence gains an increasing number of digits of accuracy with each iteration.
- Specially, a linearly convergent sequence gains $-\log_{\beta}(C)$ digits per iteration, but a superlinearly convergent sequence has *r* times as many digits of accuracy after each iteration as it has the previous iteration.
- In particular, a quadratically convergent method doubles the number of digits of accuracy with each iteration.

Exercise

- Why do we say that the bisection method has linear convergence rate?
- What is the convergence rate of the fix-point method?

Convergence Rate of Bisection Method

- Suppose that the sequence $\{x_k\}$ converges to the number L.
- Determine the convergence rate by examining

$$\lim_{k\to\infty} \frac{|x_{k+1} - L|}{|x_k - L|} = \mu. \text{ or } \lim_{k\to\infty} \frac{|x_{k+1} - L|}{|x_k - L|^q} = \mu \text{ with } \mu > 0.$$

- But in the bisection method, we are not really interested in the solution x_k but the interval when halving the original interval.
- Then, what is the convergence rate of the bisection method?

Convergence Rate of Bisection Method

- The error after n iteration is smaller than |(b-a)/2n|.
- Since the bracket size is reduced by a constant factor at each step, the convergence rate (i.e., the reduction in the error) is said to be linear.

Let δ_n be the size of the bracketing interval at the n^{th} stage of bisection. Then

$$\delta_0 = b - a = \text{initial bracketing interval}$$

$$\delta_1 = \frac{1}{2}\delta_0$$

$$\delta_2 = \frac{1}{2}\delta_1 = \frac{1}{4}\delta_0$$

$$\vdots$$

$$\delta_n = \left(\frac{1}{2}\right)^n \delta_0$$

$$\Longrightarrow \qquad \frac{\delta_n}{\delta_0} = \left(\frac{1}{2}\right)^n = 2^{-n}$$

or

$$n = \log_2\left(\frac{\delta_n}{\delta_0}\right)$$

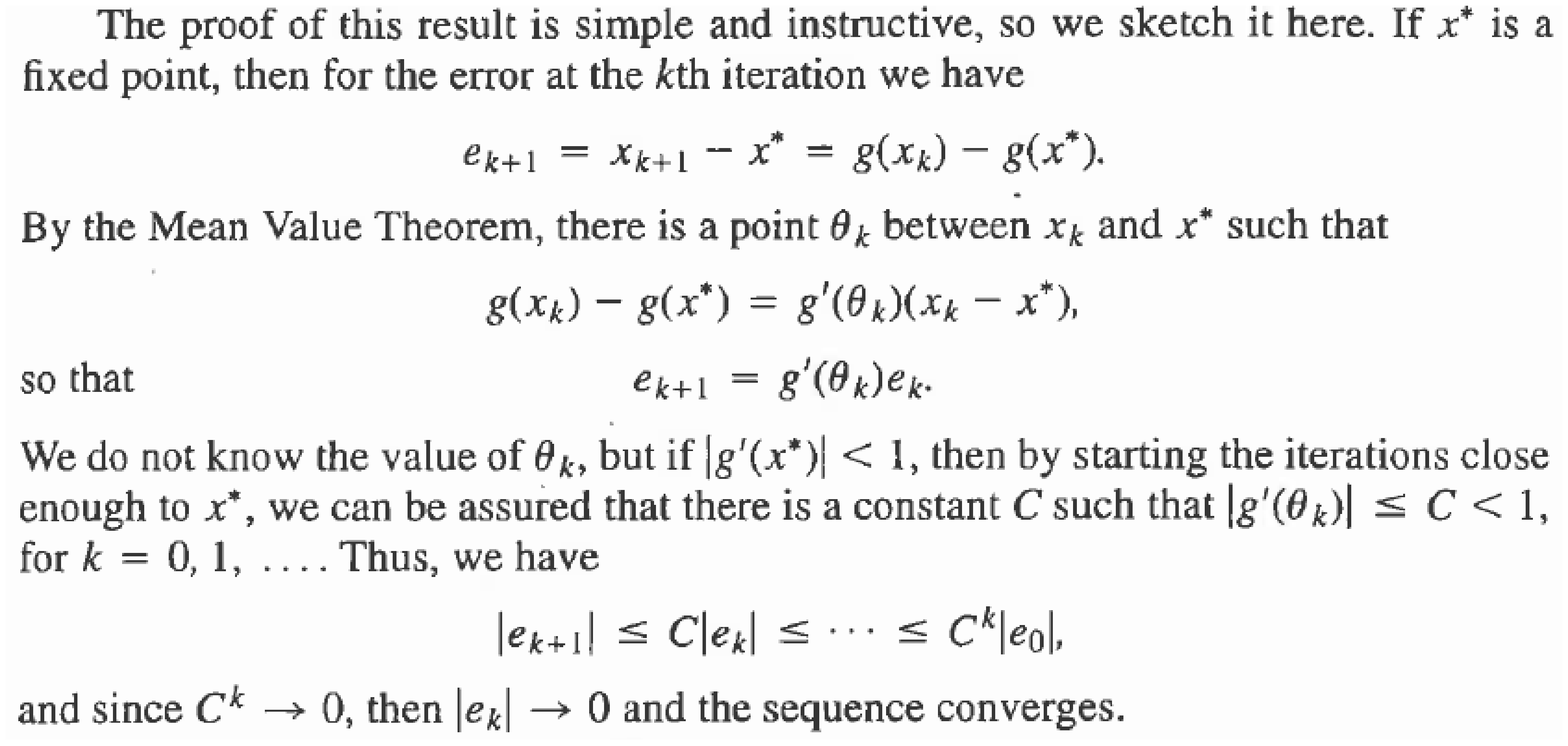
Bisection Method

If we want, we can have the sequence of x_k, which will be conveniently the sequence of x_{mid}.

$$x - x^{1/3} - 2 = 0$$

k	a	b	x_{mid}	$f(x_{mid})$
0	3	4		
1	3	4	3.5	-0.01829449
2	3.5	4	3.75	0.19638375
3	3.5	3.75	3.625	0.08884159
4	3.5	3.625	3.5625	0.03522131
5	3.5	3.5625	3.53125	0.00845016
6	3.5	3.53125	3.515625	-0.00492550
7	3.51625	3.53125	3.5234375	0.00176150
8	3.51625	3.5234375	3.51953125	-0.00158221
9	3.51953125	3.5234375	3.52148438	0.00008959
10	3.51953125	3.52148438	3.52050781	-0.00074632

Convergence Rate of Fix-Point Method



The proof of this result is simple and instructive, so we sketch it here. If x^* is a fixed point, then for the error at the kth iteration we have

$$e_{k+1} = x_{k+1} - x^* = g(x_k) - g(x^*).$$

By the Mean Value Theorem, there is a point θ_k between x_k and x^* such that

$$g(x_k) - g(x^*) = g'(\theta_k)(x_k - x^*),$$

so that

$$e_{k+1} = g'(\theta_k)e_k.$$

We do not know the value of θ_k, but if $|g'(x^*)| < 1$, then by starting the iterations close enough to x^*, we can be assured that there is a constant C such that $|g'(\theta_k)| \le C < 1$, for $k = 0, 1, \ldots$. Thus, we have

$$|e_{k+1}| \le C|e_k| \le \cdots \le C^k|e_0|,$$

and since $C^k \to 0$, then $|e_k| \to 0$ and the sequence converges.

1. Rate of Convergence

Definition 1. *If a sequence $x_1, x_2, \ldots, x_n$ converges to a value r and if there exist real numbers $\lambda > 0$ and $\alpha \geq 1$ such that*

$$\lim_{n\to\infty} \frac{|x_{n+1} - r|}{|x_n - r|^\alpha} = \lambda \tag{1}$$

then we say that α is the ***rate of convergence*** *of the sequence.*

When $\alpha = 1$ we say the sequence converges *linearly* and when $\alpha = 2$ we say the sequence converges *quadratically*. If $1 < \alpha < 2$ then the sequence exhibits *superlinear* convergence.

2. Fixed-Point Iterations

Many root-finding methods are *fixed-point* iterations. These iterations have this name because the desired root r is a **fixed-point** of a function $g(x)$, i.e., $g(r) \to r$. To be useful for finding roots, a fixed-point iteration should have the property that for x in some neighborhood of r, $g(x)$ is closer to r than x is, leading to the iteration

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

Newton's method is an example of a fixed-point iteration since

$$x_{n+1} = g(x_n), \quad g(x) = x - \frac{f(x)}{f'(x)} \tag{2}$$

Theorem 1. *Let r be a fixed-point of the iteration $x_{n+1} = g(x_n)$ and suppose that $g'(r) \neq 0$. Then the iteration will have a **linear** rate of convergence.*

Proof. Using Taylor's Theorem for an expansion about fixed-point r we find

$$g(x) = g(r) + g'(r)(x-r) + \frac{g''(\xi)}{2}(x-r)^2 \tag{3}$$

where ξ is some value between x and r. Evaluating at x_n and noting that $x_{n+1} = g(x_n)$ and $g(r) = r$ we obtain

$$x_{n+1} = r + g'(r)(x_n - r) + \frac{g''(\xi)}{2}(x_n - r)^2.$$

Subtracting r from both sides and dividing by $x_n - r$ gives

$$\frac{x_{n+1} - r}{x_n - r} = g'(r) + \frac{g''(\xi)}{2}(x_n - r)$$

which, as $n \to \infty$, yields

$$\lim_{n \to \infty} \frac{|x_{n+1} - r|}{|x_n - r|} = |g'(r)|. \tag{4}$$

Comparing this with Equation (1) we see that $\alpha = 1$ and $\lambda = |g'(r)|$, indicating that the method converges linearly. $\square$

$$g_1(x) = x^{1/3} + 2$$

$$g_2(x) = (x - 2)^3$$

$$g_3(x) = \frac{6 + 2x^{1/3}}{3 - x^{2/3}}$$

k	$g_1(x_{k-1})$	$g_2(x_{k-1})$	$g_3(x_{k-1})$
0	3	3	3
1	3.4422495703	1	3.5266442931
2	3.5098974493	-1	3.5213801474
3	3.5197243050	-27	3.5213797068
4	3.5211412691	-24389	3.5213797068
5	3.5213453678	-1.451×10^{13}	3.5213797068
6	3.5213747615	-3.055×10^{39}	3.5213797068
7	3.5213789946	-2.852×10^{118}	3.5213797068
8	3.5213796042	∞	3.5213797068
9	3.5213796920	∞	3.5213797068

Proof of quadratic convergence for Newton's iterative method [edit]

According to Taylor's theorem, any function $f(x)$ which has a continuous second derivative can be represented by an expansion about a point that is close to a root of $f(x)$. Suppose this root is α. Then the expansion of $f(\alpha)$ about x_n is:

$$f(\alpha) = f(x_n) + f'(x_n)(\alpha - x_n) + R_1 \tag{1}$$

where the Lagrange form of the Taylor series expansion remainder is

$$R_1 = \frac{1}{2!} f''(\xi_n)(\alpha - x_n)^2 \,,$$

where ξ_n is in between x_n and α.

Since α is the root, (1) becomes:

$$0 = f(\alpha) = f(x_n) + f'(x_n)(\alpha - x_n) + \tfrac{1}{2} f''(\xi_n)(\alpha - x_n)^2 \tag{2}$$

Dividing equation (2) by $f'(x_n)$ and rearranging gives

$$\frac{f(x_n)}{f'(x_n)} + (\alpha - x_n) = \frac{-f''(\xi_n)}{2f'(x_n)}(\alpha - x_n)^2 \tag{3}$$

Remembering that x_{n+1} is defined by

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \,, \tag{4}$$

one finds that

$$\underbrace{\alpha - x_{n+1}}_{\varepsilon_{n+1}} = \frac{-f''(\xi_n)}{2f'(x_n)} (\underbrace{\alpha - x_n}_{\varepsilon_n})^2 \,.$$

That is,

$$\varepsilon_{n+1} = \frac{-f''(\xi_n)}{2f'(x_n)} \cdot {\varepsilon_n}^2 \, .$$

Taking absolute value of both sides gives

$$|\varepsilon_{n+1}| = \frac{|f''(\xi_n)|}{2\,|f'(x_n)|} \cdot {\varepsilon_n}^2 \, .$$

Equation (6) shows that the rate of convergence is quadratic if the following conditions are satisfied:

1. $f'(x) \neq 0$; for all $x \in I$, where I is the interval $[\alpha - r, \alpha + r]$ for some $r \geq |\alpha - x_0|$;
2. $f''(x)$ is continuous, for all $x \in I$;
3. x_0 *sufficiently* close to the root α.

- ds

k	x_k	$f'(x_k)$	$f(x)$
0	3	0.83975005	-0.44224957
1	3.52664429	0.85612976	0.00450679
2	3.52138015	0.85598641	3.771×10^{-7}
3	3.52137971	0.85598640	2.664×10^{-15}
4	3.52137971	0.85598640	0.0

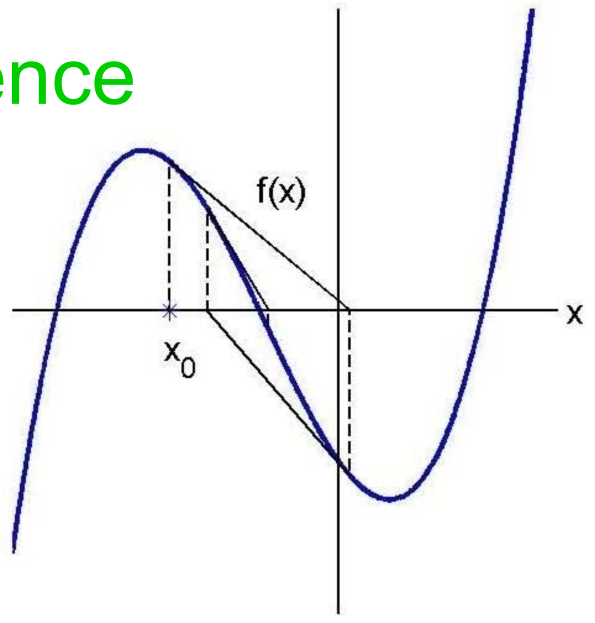
Convergence of Newton's Method

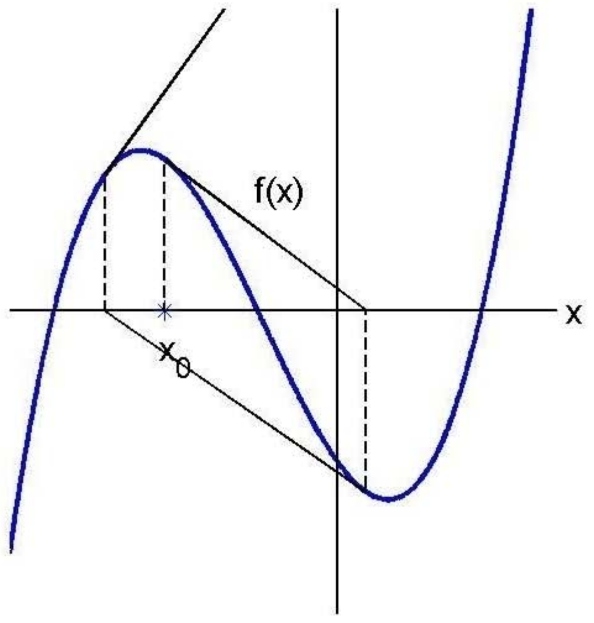
The quadratic convergence of Newton's method comes with restrictions.

- Newton's method is not guaranteed to converge unless $|x_0-\xi|\leq\rho$, where, in general, both ξ and ρ are not known.
- In other words, Newton's method converges if the initial guess is close enough to the root, but neither the root nor a measure of "close enough" can be prescribed ahead of time.
- If $f(x)$ has multiple roots (or root of multiplicity), $f'(\xi)=0$, the convergence of Newton's method is only linear, not quadratic.

Initial Value x_0

Convergence





Divergence

- The closer the initial guess x_0 is to the root, the faster and more certain the convergence.
- A bad choice of initial value may result in divergence (roots cannot be found)

System of Nonlinear Equations

- We now consider nonlinear equations in more than one dimension.
- The multidimensional case is much more difficult than the scalar case for a variety of reasons:
  - A much wider range of behavior is possible, so that a theoretical analysis of the existence and number of solution is much more complex.
  - It is not possible, in general, to guarantee convergence to the correction solution or to bracket the solution to produce an absolutely safe method.
  - Computational overhead increases rapidly with the dimension of the problem.
- Many one-dimensional methods do not generalized directly to n dimensions.
- The most popular and powerful method that does generalize is Newton’s method.

Open Channel Hydraulics

- Apply the principles of fluid mechanics to surface-water flow in channels (e.g., rivers, streams, and canals).
- Why does the flow in rivers vary from deep and tranquil to shallow and torrential?
- What controls the depth of water in a river?
- How are water depth and discharge in a stream related?

Open Channel Flow

Difference between open channel flow and pipe flow

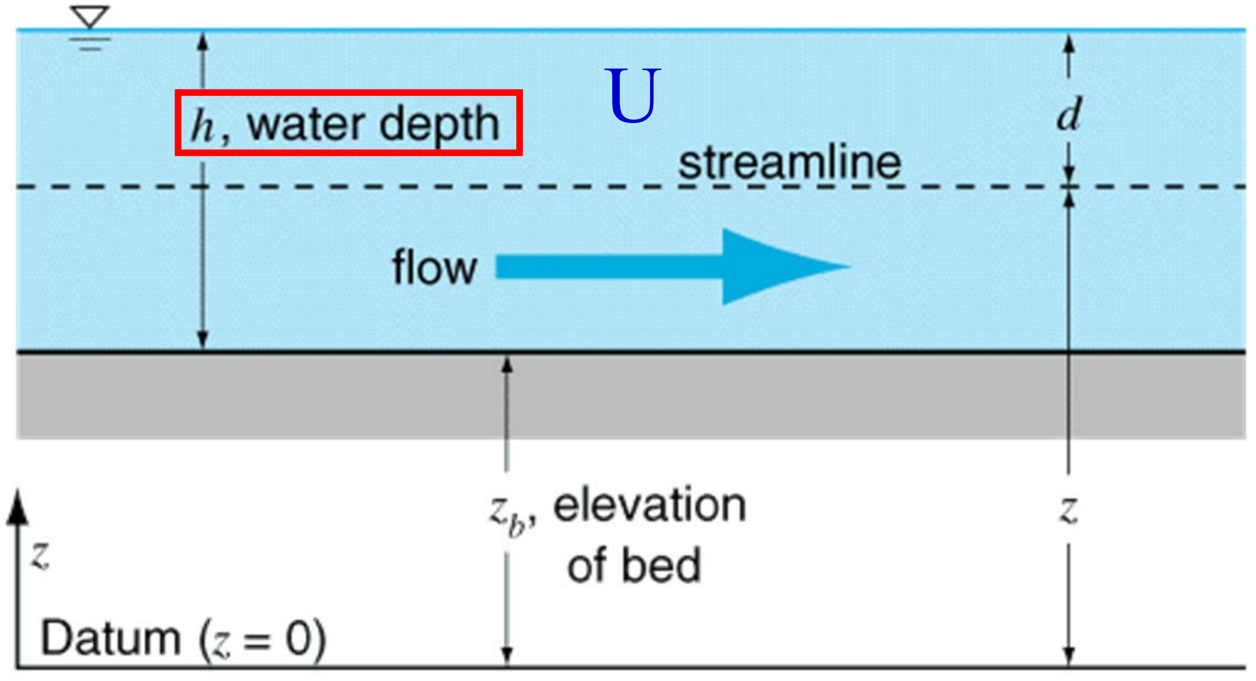
- The channel flow is only partially enclosed by a solid boundary.
- The upper "boundary" of an open channel flow, between the water and the atmosphere, is called a **free surface**.

Start from simple case:

flow to be steady and frictionless so that the Bernoulli equation can be used.

Reach: A segment of a stream or river channel

In a very short **reach**, frictional losses and elevation change can be disregarded.



Convenient reference elevation. Sea level is often used.

Bernoulli's Equation

$$\frac{u^2}{2g} + \frac{p}{\rho g} + z = H$$

$$\frac{u^2}{2g} + \frac{p}{\rho g} + (z_b + h - d) = H$$

Horizontal flow → no fluid acceleration in the vertical direction + steady state flow → total acceleration=0 → *p*=*ρgd* → *p*/*ρg*=*d*

$$\frac{u^2}{2g} + d + (z_b + h - d) = H \longrightarrow \boxed{\frac{U^2}{2g} + h} + z_b = H \qquad E + z_b = H$$

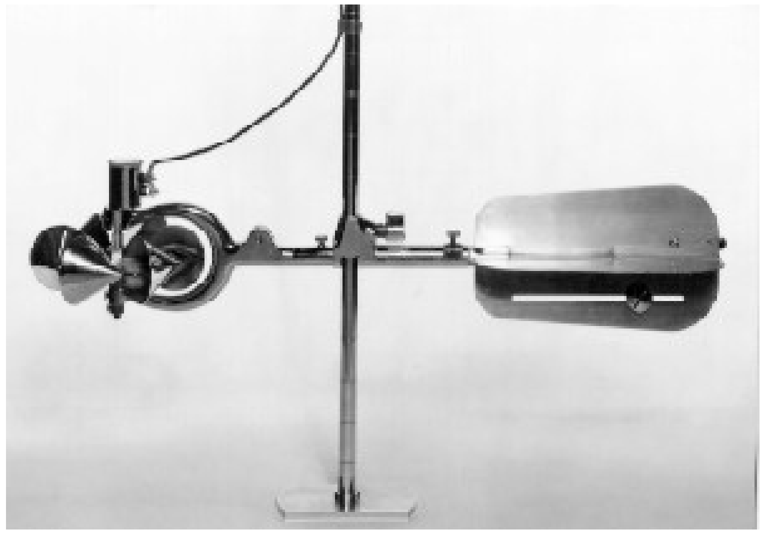
$$E = \frac{U^2}{2g} + h$$

Specific energy (*E* [L]): the energy per unit weight of the flowing water *relative to the stream bottom*.

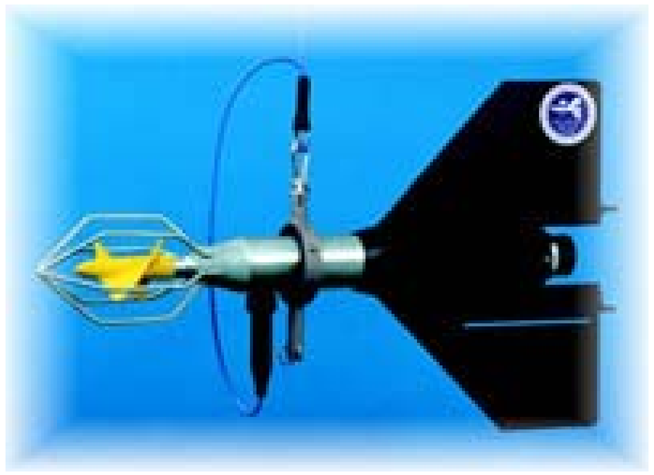
Measuring Flow in Natural Channels

- Current meter to measure river velocity and

Price current meter, 1882


Nowadays

Propeller

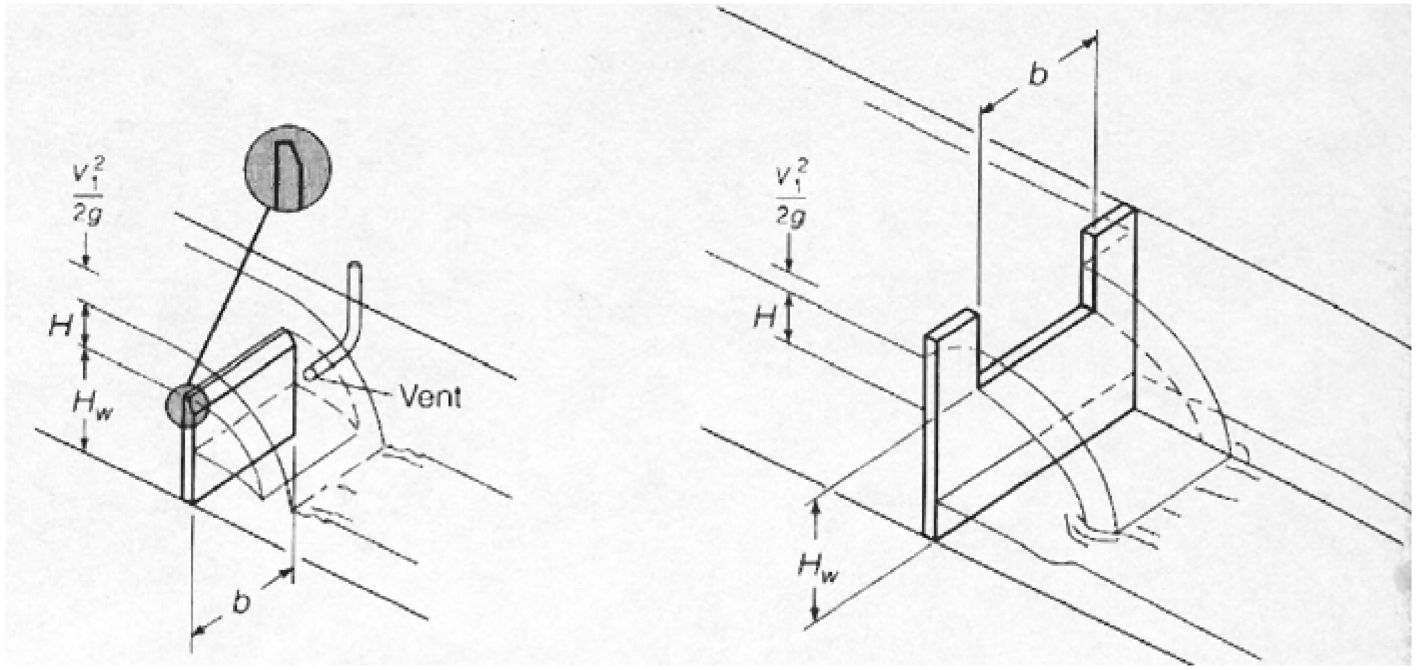

Current meter or missile?

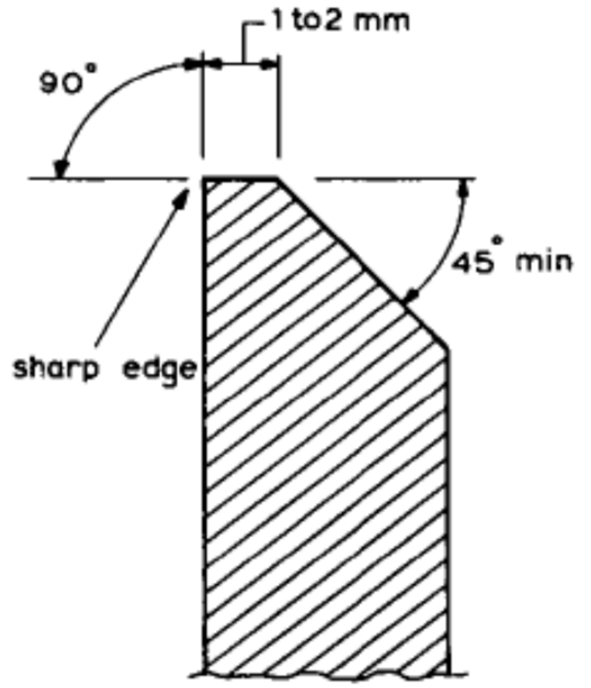
The current meter consists of a cup- or propeller-type rotor. The number of revolutions of the rotor in a given period (n) is directly proportional to the velocity (v) of the river: $v=a+bn$.

Rectangular Sharp-crested Weir

Suppressed: rectangular section spans the full width of the channel

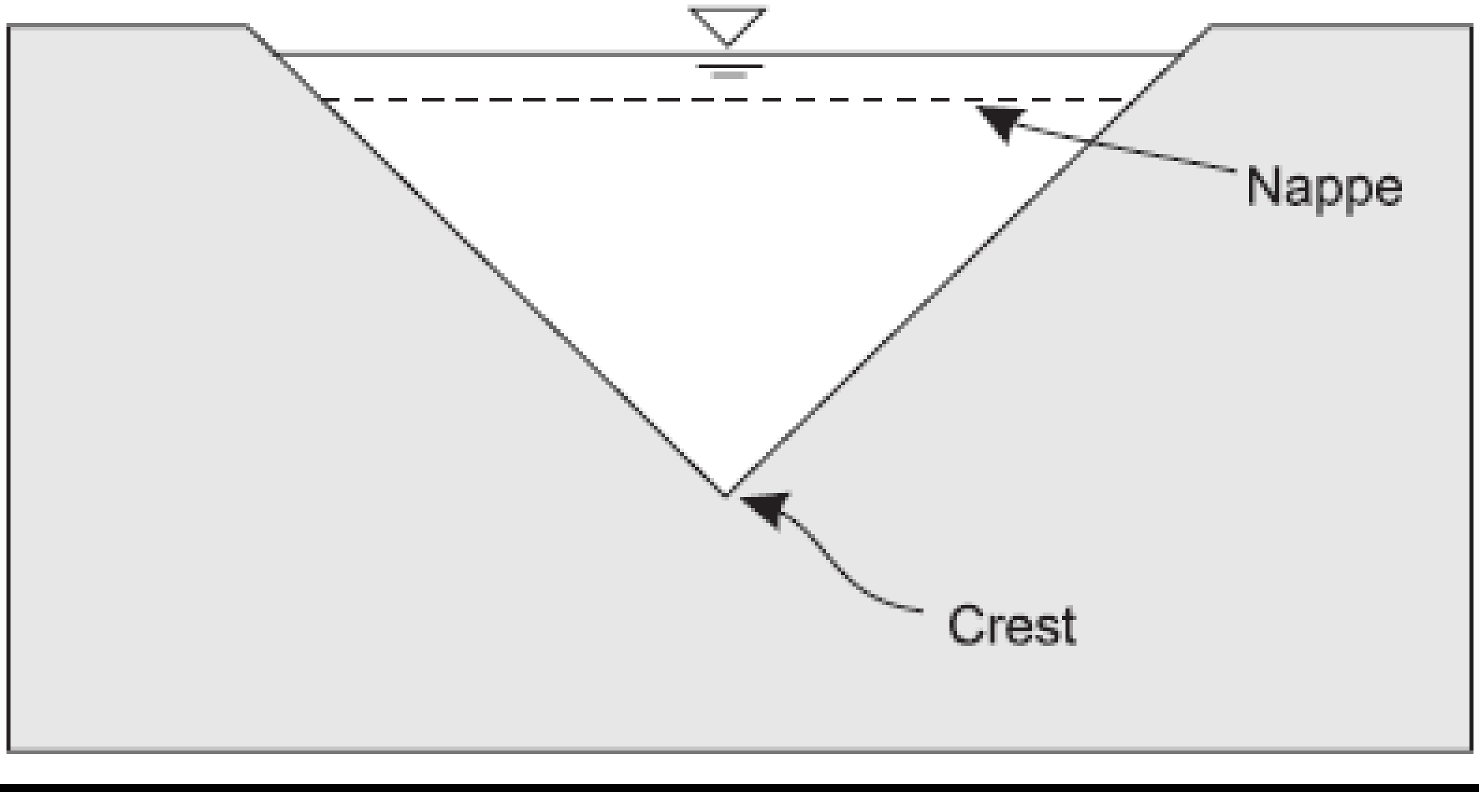
Weir with end contraction: weir length less than the width of the channel





Notched Weirs

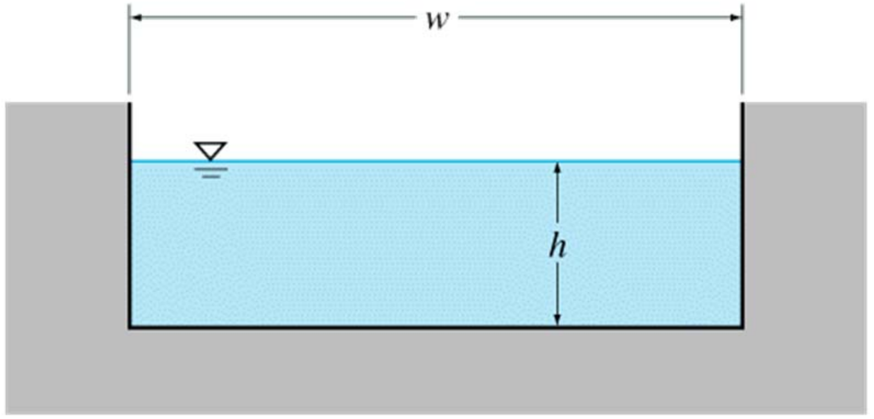
V-notch



$$E + z_b = H$$

If the elevation of the channel bottom remains constant, then because *H* is constant, *E* must also be constant.

$$E = \frac{U^2}{2g} + h$$



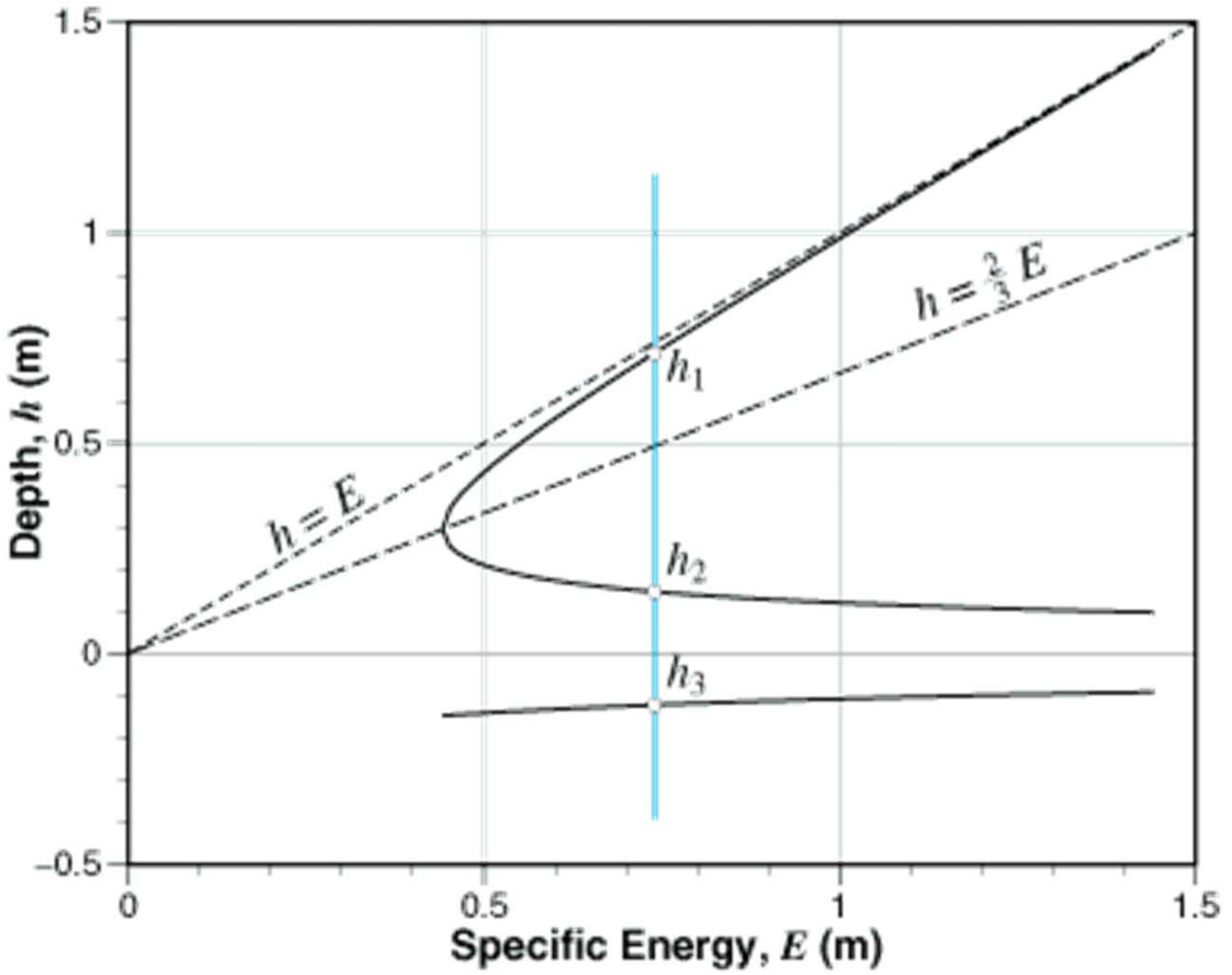
Channel with a rectangular cross section

$$Q = Uwh$$

Specific discharge: $q_w = \frac{Q}{w} = Uh = \text{constant}$

$$E = \frac{q_w^2}{2gh^2} + h \qquad U = q_w/h$$

If *E* and q_w are known, then *h* can be estimated.



Exercise

$$E = \frac{q_w^2}{2gh^2} + h$$

or

$$h^3 - Eh^2 + \frac{q^2}{2g} = 0$$

$$f(h) = h^3 - Eh^2 + \frac{q^2}{2g} = 0$$

Find roots of *f*(*x*)=0.

Solve *h* for *E*=0.75m and q_w = 0.5 m^2 s^{-1}.

The root Function

The built-in **roots** function uses the companion matrix method

- No initial guess
- Returns *all* roots of the polynomial
- Solves eigenvalue problem for companion matrix

Write polynomial in the form

$$c_1 x^n + c_2 x^{n-1} + \ldots + c_n x + c_{n+1} = 0$$

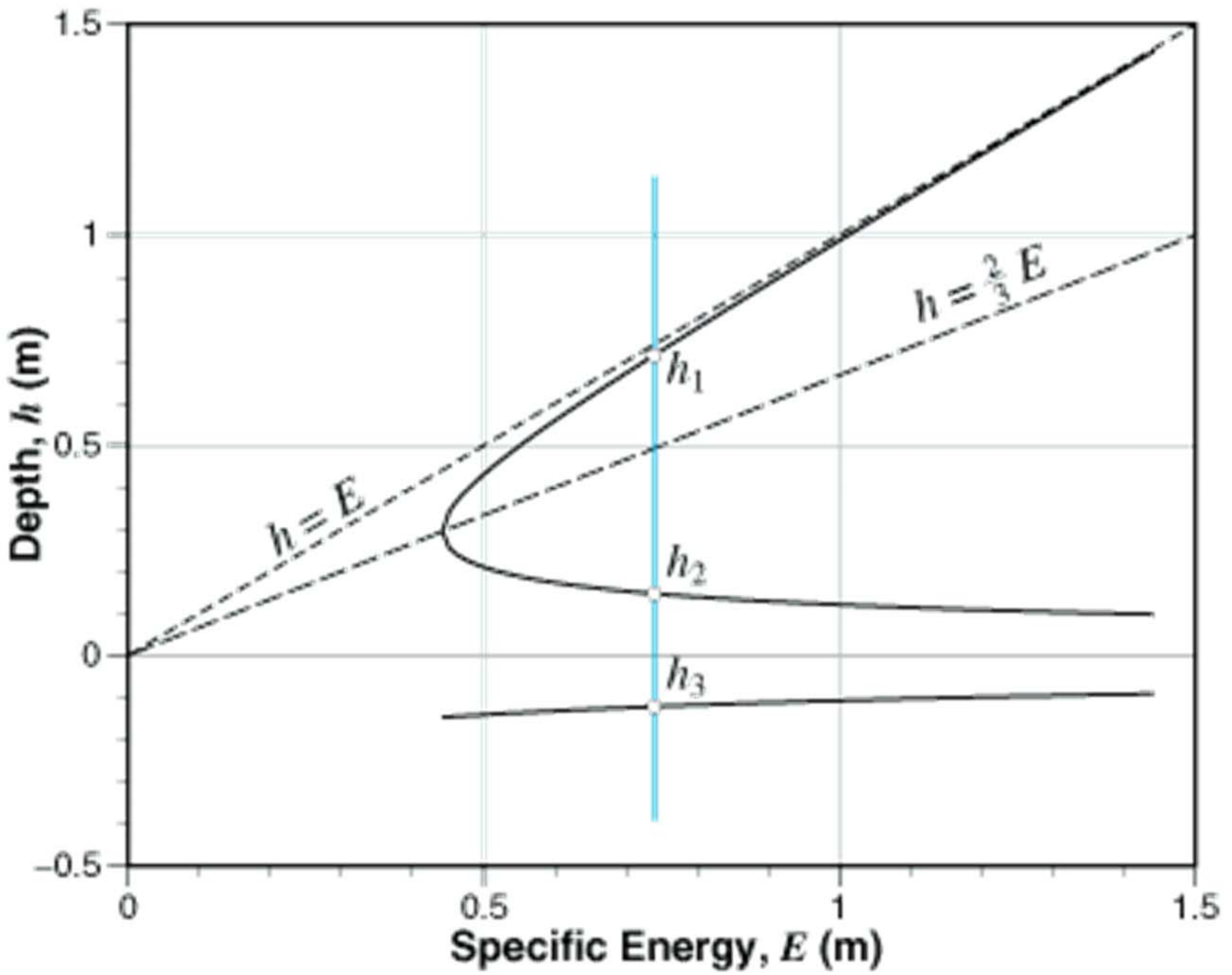
Then, for a *third* order polynomial

```
>> c = [c1 c2 c3 c4];
>> r = roots(c)
```

$$E = \frac{q_w^2}{2gh^2} + h$$

Specific Energy Diagram for a given q_w

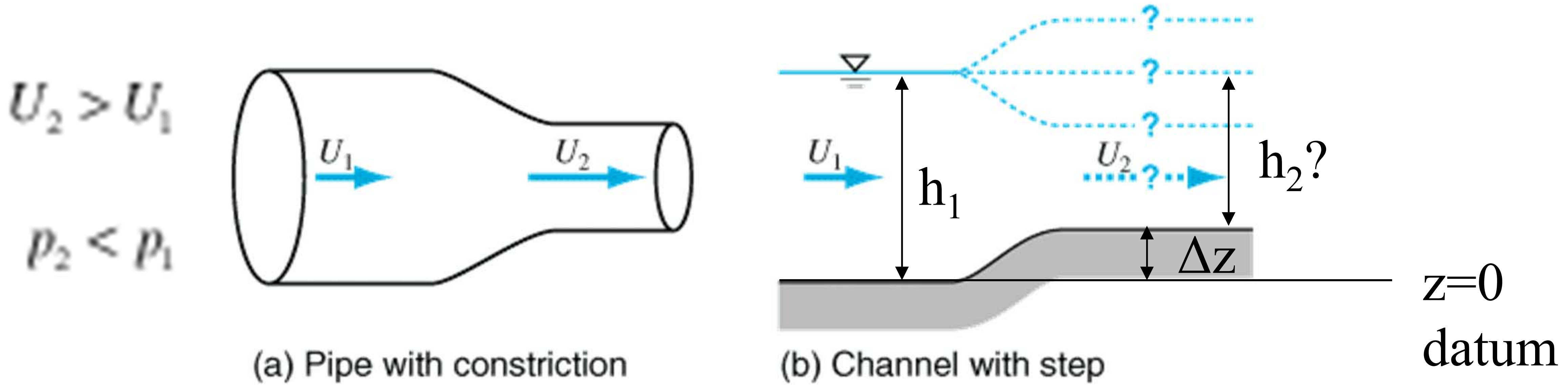
For E=0.75m and q_w = 0.5 m^2 s^{-1}, there are three possible depths for the cubic equation relating E, q_w, and h. The negative h_3 has no physical meaning, and can be discarded. The other two depths are all possible, called alternative depths.



Bernoulli Equation + Continuity Equation

Further work is needed!

Flow Over a Vertical Step



What will the surface do as the flow crosses the step: rise, fall, or remain unchanged?

The answer is gained by considering the relationship among: specific energy, specific discharge, and water depth.

Bernoulli's Equation $$\frac{U_1^2}{2g} + h_1 = \frac{U_2^2}{2g} + h_2 + \Delta z \longrightarrow E_1 = E_2 + \Delta z = H$$

the flow velocity and depth upstream of the step are known $$E_1 = h_1 + \frac{U_1^2}{2g} \qquad q_w = U_1 h_1$$

constant

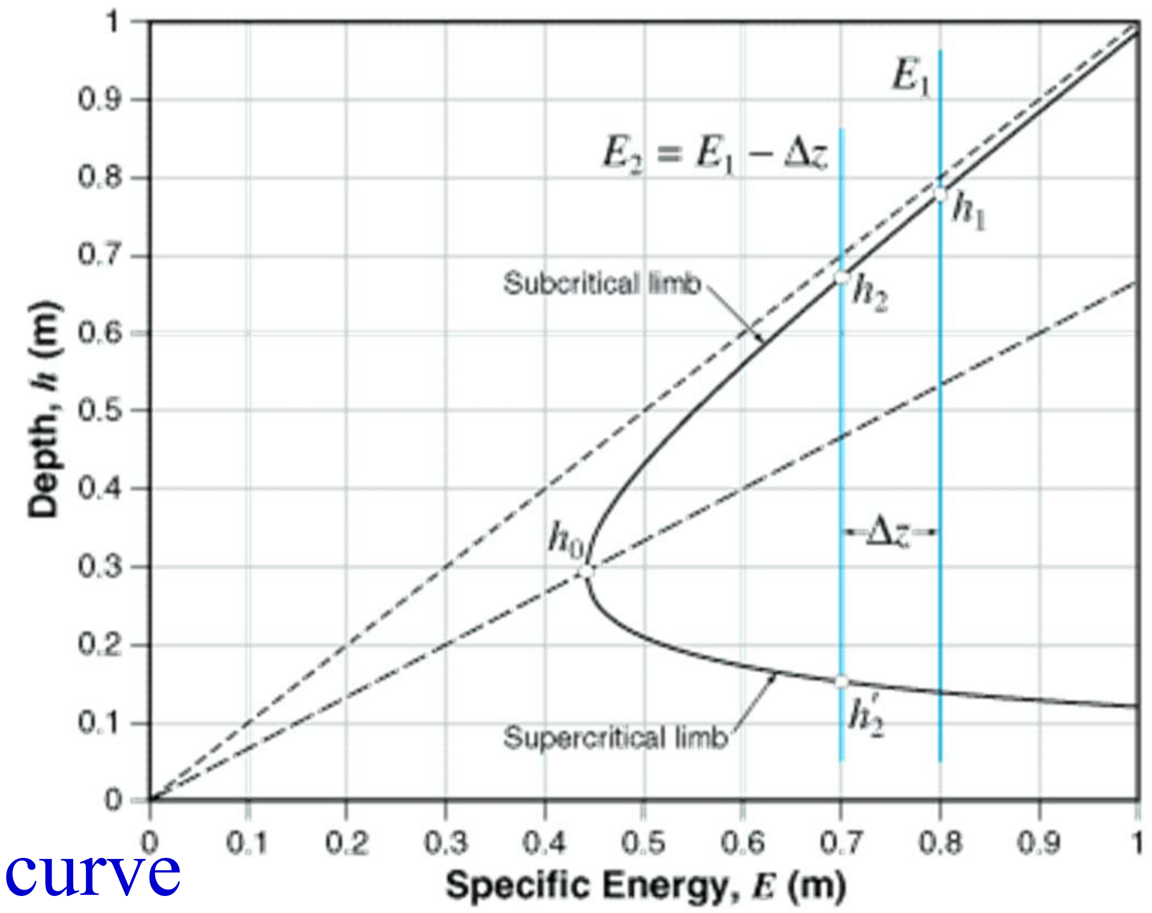
q_w is known and constant, so we can create a specific energy diagram

$$E_1 = E_2 + \Delta z = H$$

- Locate $E_1 = 0.8$ m
- Estimate $\Delta z = 0.1$ m
- Locate $E_2 = E_1 - \Delta z = 0.7$ m
- Find alternative solutions h_2 and h_2' ($h_1 > h_2 > h_2'$)

$h_1 \rightarrow h_2 \rightarrow h_0 \rightarrow h_2'$

Along the same specific energy curve



$h_1 \rightarrow h_2$:

Flow moves up

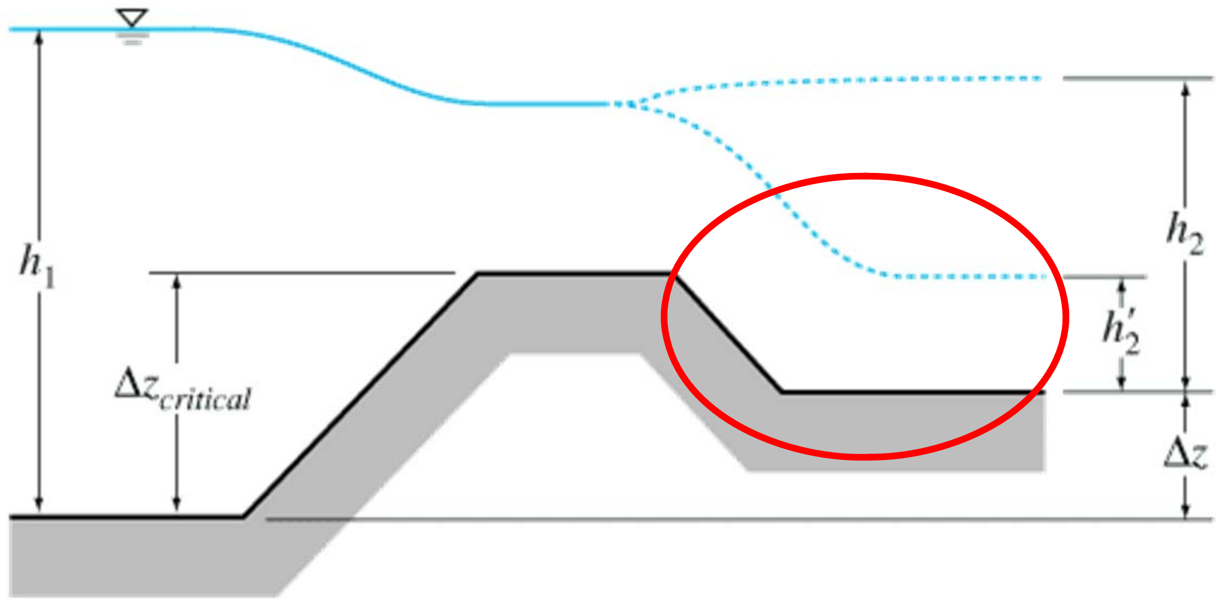
Lose specific energy

$h_2 \rightarrow h_0 \rightarrow h_2'$:

Gain specific energy

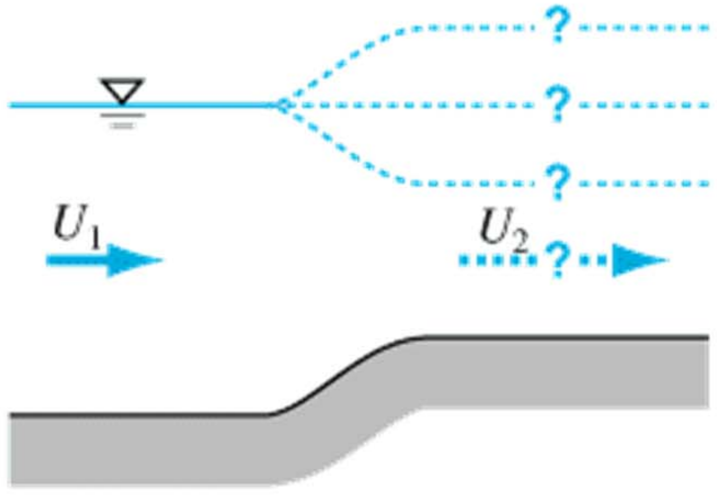
Flow moves down

Such a hump does not exist.



h_2 is the only solution physically meaningful

What is the relationship between h_1 and $h_2 + \Delta z$?



(b) Channel with step

$$\frac{U_1^2}{2g} + h_1 = \frac{U_2^2}{2g} + h_2 + \Delta z$$

$$h_1 - (h_2 + \Delta z) = \frac{U_2^2}{2g} - \frac{U_1^2}{2g}$$

$q_w = U_1 h_1 = U_2 h_2$

$h_1 > h_2$

$U_1 < U_2$

$$\frac{U_2^2}{2g} - \frac{U_1^2}{2g} > 0$$

$$h_1 - (h_2 + \Delta z) > 0$$

$$h_1 > h_2 + \Delta z$$

the water surface *drops* and velocity *increases* as water flows over the step.
This conclusion explains why canoeists are wary of sudden changes in stream elevation, as they often mean that large rocks are below.

What for a decrease step?
What if width changes?