2.29 Numerical Fluid Mechanics
Fall 2011 – Lecture 3

REVIEW Lectures 1-2

- Approximation and round-off errors
  - Absolute and relative errors: $E_a = \hat{x}_a - \hat{x}\ , \quad \varepsilon_a = \dfrac{\hat{x}_a - \hat{x}}{\hat{x}_a}$
    - Iterative schemes and stop criterion: $|\varepsilon_a| = \left|\dfrac{\hat{x}_n - \hat{x}_{n-1}}{\hat{x}_n}\right| \le \varepsilon_s$
    - For n digits: $\varepsilon_s = \dfrac{1}{2}10^{-n}$
  - Number representations
    - Floating-Point representation: $x = m\, b^e \qquad b^{-1} \le m < b^0$
  - Quantizing errors and their consequences
  - Arithmetic operations (e.g. radd.m)
  - Errors of arithmetic/numerical operations
    - Large number of addition/subtractions, large & small numbers, inner products, etc
  - Recursion algorithms (Heron, Horner's scheme, Bessel functions, etc):
    - Order of computations matter
    - Round-off error growth and (in)-stability

2.29 Numerical Fluid Mechanics
Fall 2011 – Lecture 3: Outline

- Truncation Errors, Taylor Series and Error Analysis
  - Taylor series: $f(x_{i+1}) = f(x_i) + \Delta x\, f'(x_i) + \frac{\Delta x^2}{2!} f''(x_i) + \frac{\Delta x^3}{3!} f'''(x_i) + \ldots + \frac{\Delta x^n}{n!} f^n(x_i) + R_n$

    $$R_n = \frac{\Delta x^{n+1}}{n+1!} f^{(n+1)}(\xi)$$
  - Use of Taylor Series to derive finite difference schemes (first-order Euler scheme and forward, backward and centered differences)
  - General error propagation formulas and error estimation, with examples

    Consider $y = f(x_1, x_2, x_3, \ldots, x_n)$. If ε_i's are magnitudes of errors on x_i's, what is the error on y?
    - The Differential Formula: $\varepsilon_y \le \sum_{i=1}^{n} \left| \frac{\partial f(x_1,\ldots,x_n)}{\partial x_i} \right| \varepsilon_i$
    - The Standard Error (statistical formula): $E(\Delta_s y) \cong \sqrt{\sum_{i=1}^{n} \left(\frac{\partial f}{\partial x_i} \right)^2 \varepsilon_i^{\,2}}$
  - Error cancellation (e.g. subtraction of errors of the same sign)
  - Condition number: $K_p = \frac{\bar{x}\, f'(\bar{x})}{f(\bar{x})}$
    - Well-conditioned problems vs. well-conditioned algorithms
    - Numerical stability

Reference: Chapra and Canale, Chaps 3, 4 and 5

Numerical Fluid Mechanics: Lectures 3 and 4 Outline

Reference: Chapra and Canale, Chaps 3, 4 and 5

- Condition Numbers
- Roots of nonlinear equations
  - Bracketing Methods
    - Example: Heron's formula
    - Bisection
    - False Position
  - "Open" Methods
    - Open-point Iteration (General method or Picard Iteration)
      - Examples
      - Convergence Criteria
      - Order of Convergence
    - Newton-Raphson
      - Convergence speed and examples
    - Secant Method
      - Examples
      - Convergence and efficiency
    - Extension of Newton-Raphson to systems of nonlinear equations
  - Roots of Polynomial (all real/complex roots)
    - Open methods (applications of the above for complex numbers)
    - Special Methods (e.g. Muller's and Bairstow's methods)

Truncation Errors, Taylor Series and Error Analysis

Taylor Series:

- Provides a mean to predict a function at one point in terms of its values and derivatives at another point (in the form of a polynomial)
- Hence, any smooth functions can be approximated by a polynomial
- Taylor Series (Mean for integrals theorems):

$$f(x_{i+1}) = f(x_i) + \Delta x\, f'(x_i) + \frac{\Delta x^2}{2!} f''(x_i) + \frac{\Delta x^3}{3!} f'''(x_i) + \ldots + \frac{\Delta x^n}{n!} f^n(x_i) + R_n$$

$$R_n = \frac{\Delta x^{n+1}}{n+1!} f^{(n+1)}(\xi)$$

- = constant + line + parabola + etc

Derivation of General "Differential" Error Propagation Formula

Univariate Case

Recall: $$f(x_{i+1}) = f(x_i) + \Delta x\, f'(x_i) + \frac{\Delta x^2}{2!} f''(x_i) + \frac{\Delta x^3}{3!} f'''(x_i) + \ldots + \frac{\Delta x^n}{n!} f^n(x_i) + R_n$$

$$R_n = \frac{\Delta x^{n+1}}{n+1!} f^{(n+1)}(\xi) = O(\Delta x^{n+1})$$

Hence, $$\Delta y \equiv \Delta f \;=\; f(x_{i+1}) - f(x_i)$$

$$= \Delta x\, f'(x_i) + \frac{\Delta x^2}{2!} f''(x_i) + \frac{\Delta x^3}{3!} f'''(x_i) + \ldots + \frac{\Delta x^n}{n!} f^n(x_i) + R_n$$

$$\text{For } \Delta x \ll 1, \quad \Delta f = \Delta x\, f'(x_i) + O(\Delta x^2) \approx \Delta x\, f'(x_i)$$

Thus, for an error on x equal to Δx such that $|\Delta x| \cong \varepsilon \ll 1$, we have an error on y equal to :

$$\varepsilon_y = |\Delta y| = |\Delta f| \approx |\Delta x\, f'(x_i)| \;=\; |\Delta x|\; |f'(x_i)| = \varepsilon |f'(x_i)|$$

Multivariate case

$$y = f(x_1, x_2, x_3, \ldots, x_n) \qquad \text{For } |\Delta x_i| \ll 1, \quad \varepsilon_y \le \sum_{i=1}^{n} \left| \frac{\partial f(x_1, \ldots, x_n)}{\partial x_i} \right| \varepsilon_i$$

Derivation done in class on the board

General Error Propagation Formula (The Differential Formula)

$$y = f(x_1, x_2, \ldots x_n)$$

Absolute Errors

$$\epsilon_1, \epsilon_2, \ldots \epsilon_n$$

$$\epsilon_y \, ?$$

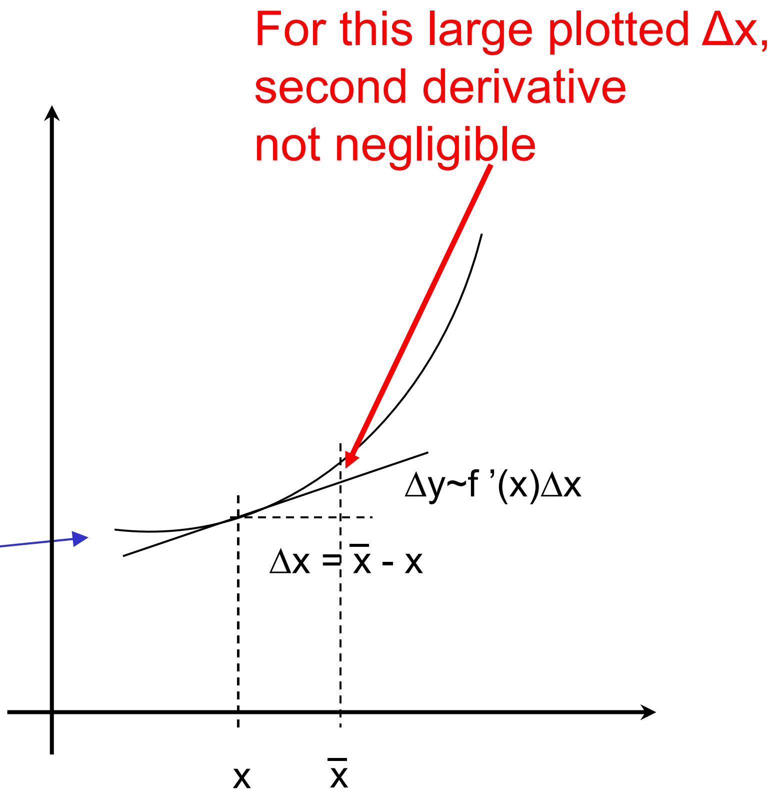
Function of one variable

$$y = f(x) \quad \bar{y} = f(\bar{x})$$

General Error Propagation Formula

$$\Delta y \simeq \sum_{i=1}^{n} \frac{\partial f(x_1, \ldots, x_n)}{\partial x_i} \Delta x_i$$

$$\boxed{\epsilon_y \leq \sum_{i=1}^{n} \left| \frac{\partial f(x_1, \ldots, x_n)}{\partial x_i} \right| |\epsilon_i|}$$



Error Propagation Example with Differential Approach: Multiplications

Multiplication

$$y = x_1 x_2$$

$$\Rightarrow \quad \log y = \log x_1 + \log x_2$$

$$\Rightarrow \quad \frac{1}{y}\frac{\partial y}{\partial x_i} = \frac{1}{x_i}$$

$$\Rightarrow \quad \frac{\partial y}{\partial x_i} = \frac{y}{x_i}$$

=>

Error Propagation Formula

$$\left|\frac{\Delta y}{y}\right| \le \sum_{i=1}^{2} \left|\frac{\Delta x_i}{x_i}\right|$$

$$\varepsilon^r{}_y \le \sum_{i=1}^{2} \varepsilon^r{}_i$$

Relative Errors Add for Multiplication

Another example, more general case: $y = x_1^{m_1} x_2^{m_2} \cdots x_n^{m_n}$

$$\boxed{\varepsilon^r{}_y \le \sum_{i=1}^{n} |m_i|\, \varepsilon^r{}_i}$$

Statistical Error Propagation Estimate: **Error Expectations and the "Standard Error"**

Example: Addition

$$y = x_1 + x_2 + \cdots + x_n$$

a. Chopping

$$\Delta x_i = \bar{x}_i - x_i \leq 0$$

Error Expectation
(on average, half the interval is chopped)

$$\begin{aligned} E(\Delta x_i) &= -b^{-t}/2 \\ E(\Delta y) &= -nb^{-t}/2 \end{aligned}$$

b. Rounding

$$\begin{aligned} E(\Delta x_i) &= 0 \\ E(\Delta y) &= nE(\Delta x_i) = 0 \end{aligned}$$

Statistical Error Propagation Estimate: **Error Expectations and the "Standard Error"**

$$y = y(x_1, x_2, x_3, \ldots, x_n)$$

Standard Error
(assuming Gaussian errors on x_i ,
compute error standard deviation on y)

$$\boxed{E(\Delta_s y) \simeq \sqrt{\sum_{i=1}^{n} \left(\frac{\partial y}{\partial x_i}\right)^2 \epsilon_i^2}}$$

Example: Addition

$$y = x_1 + x_2 + \cdots + x_n$$

$$E(\Delta_s y) \simeq \sqrt{\sum_{i=1}^{n} \epsilon_i^2} = \sqrt{n}\epsilon$$

Standard Error is a more accurate measure of expected errors:
It is likely that the error will be of that magnitude.

However, the general error propagation formula is an upper bound

Error Propagation: Error Cancellation (occurs when errors of the same sign are substracted)

Example: Consider this function of one variable

$$y = f(x) = \sqrt{x^2+1} - x + 200; \quad x = 100 \pm 4$$

$$\frac{\partial z_1}{\partial x} = \frac{x}{\sqrt{x^2+1}}, \quad \frac{\partial z_2}{\partial x} = -1$$

Define:

$$z_1 = \sqrt{x^2+1} \quad \epsilon_1 = 4$$

$$z_2 = 200 - x \quad \epsilon_2 = 4$$

$$y = z_1 + z_2$$

$$\Delta y \simeq \frac{dz_1}{dx}\Delta x + \frac{dz_2}{dx}\Delta x$$

For 2 terms (z_1 and z_2):

Gen. error (max): $E(\Delta y) \;=\; 8$

Stand. Error : $E(\Delta_s y) \;=\; 4\sqrt{2} = 5.6$

Direct error (single y term):

$$\boxed{= \left(\frac{x}{\sqrt{x^2+1}} - 1\right)\Delta x \simeq_{x \gg 1} \frac{-1}{2x^2}\Delta x}$$

Error cancellation

$$x = 100 \pm 4 \Rightarrow |\Delta y| \leq 2\ 10^{-4} \underbrace{\leq 0.5\, 10^{-3}}$$

$$y = 200.005 \pm 0.5\, 10^{-3}$$

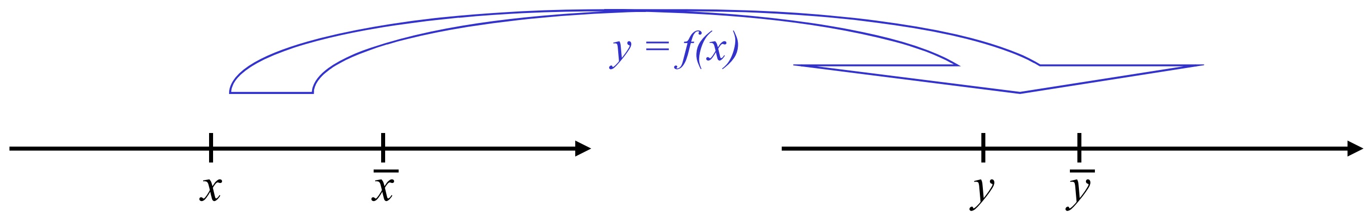
error if 4 digits

The Condition Number

- The *condition* of a mathematical problem relates to its sensitivity to changes in its input values
- A computation is *numerically unstable* if the uncertainty of the input values are magnified by the numerical method
- Considering x and $f(x)$, the *condition number* is the ratio of the relative error in $f(x)$ to that in x.
- Using first-order Taylor series $f(x) = f(\overline{x}) + f'(\overline{x})(x - \overline{x})$
- Relative error in $f(x)$: $\dfrac{f(x) - f(\overline{x})}{f(x)} \cong \dfrac{f'(\overline{x})(x - \overline{x})}{f(\overline{x})}$
- Relative error in x: $\dfrac{(x - \overline{x})}{\overline{x}}$
- Condition Nb = Ratio of relative errors: $\boxed{K_p = \dfrac{\overline{x}\, f'(\overline{x})}{f(\overline{x})}}$

Error Propagation

Condition Number: another derivation and an example



$$\bar{x} = x\,(1+\alpha)$$

$$\bar{y} = y\,(1+\beta)$$
$$= f(\bar{x}) = f(x)\,(1+\beta)$$

Problem Condition Number

$$K_P = \frac{|\beta|}{|\alpha|}$$
$$= \left|\frac{f(\bar{x}) - f(x)}{f(x)}\right| \Big/ \left|\frac{\bar{x}-x}{x}\right|$$
$$= \left|\frac{f(\bar{x}) - f(x)}{\bar{x}-x}\right| \times \left|\frac{x}{f(x)}\right|$$

With first-order Taylor series $\Rightarrow$

$$\simeq \left|x\frac{f'(x)}{f(x)}\right|$$

$$\boxed{K_P \gg 1 \quad \text{Problem ill-conditioned}}$$

Error cancellation example

$$y = f(x) = \sqrt{x^2+1} - x + 200; \quad x = 100 \pm 4$$

$$K_P = \left|100\,\frac{-0.5\ 10^{-4}}{200.005}\right| = 0.2\ 10^{-4}$$

Well-conditioned problem

Error Propagation
Problem vs. Algorithm Condition Numbers

a. Problem Condition Number

$$
\begin{aligned}
y &= \sqrt{x^2+1} - x \\
x &= 100 \;\Rightarrow y = 0.5 \cdot 10^{-2} \\
y' &= \frac{x}{\sqrt{x^2+1}} - 1 \simeq -\frac{1}{2\,x^2} = -10^{-4} \\
K_P &= \left| 100 \frac{-1 \cdot 10^{-4}}{0.5 \cdot 10^{-4}} \right| = 2.0
\end{aligned}
$$

K_A is the algorithm condition number, which may be much larger than K_P due to digitized number representation.

Solution:

- Higher precision
- Rewrite algorithm

b. Algorithm (4 Significant Digits) Condition Number

$$
\begin{aligned}
\bar{y} &= \sqrt{0.1 \cdot 10^5 + 1} - 0.1 \cdot 10^3 \\
&= \sqrt{(0.1000 + 0.0000\boxed{1}) \cdot 10^5} - 0.1 \cdot 10^3 = 0
\end{aligned}
$$

(base 10)

$$
|\beta| = \left| \frac{\bar{y} - y}{y} \right| = \frac{0.5 \cdot 10^{-2}}{0.5 \cdot 10^{-2}} = 1
$$

$$
|\alpha| = \left| \frac{\bar{x} - x}{x} \right| \le \frac{1}{2} 10^{1-t} \simeq \frac{1}{2} 10^{-3}
$$

Algorithm Condition Number

$$
\boxed{K_A = \frac{|\beta|}{|\alpha|} \simeq 2000}
$$

Well-conditioned Algorithm

$$
y = \frac{1}{\sqrt{x^2+1} + x}
$$

$$
\bar{y} = \frac{1}{0.1 \cdot 10^3 + 0.1 \cdot 10^3} = 0.5 \cdot 10^{-}2
$$

$$
|\beta| \simeq 0 \Rightarrow \boxed{K_A \simeq 0 \ll 1}
$$

Roots of Nonlinear Equations

- Finding roots of equations ($\mathbf{x} \mid \boldsymbol{f}(\mathbf{x}) = \mathbf{0}$) ubiquitous in engineering problems, including numerical fluid dynamics applications
  - In general: finding roots of functions (implicit in unknown parameters/roots)
  - Fluid Application Examples
    - Implicit integration schemes
    - Stationary points in fluid equations, stability characterizations, etc
    - Fluid dynamical system properties
    - Fluid engineering system design
- Most realistic root finding problems require numerical methods
- Two types of root problems/methods:
  1. Determine the real roots of algebraic/transcendental equations (these methods usually need some sort of prior knowledge on a root location)
  2. Determine all real and complex roots of polynomials (all roots at once)

Roots of Nonlinear Equations

- Bracketing Methods
  - Bracket root, systematically reduce width of bracket, track error for convergence
  - Methods: Bisection and False-Position (Regula Falsi)
- “Open” Methods
  - Systematic “Trial and Error” schemes, don’t require a bracket (can start from a single value)
  - Computationally efficient, but don’t always converge
  - Methods: Open-point Iteration (General method or Picard Iteration), Newton-Raphson, Secant Method
  - Extension of Newton-Raphson to systems of nonlinear equations
- Roots of Polynomials (all real/complex roots)
  - Open methods (applications of the above for complex numbers)
  - Special Methods (e.g. Muller’s and Bairstow’s methods)

Bracketing Methods: Graphical Approach

A function typically changes sign at the vicinity of the root ⇒ “bracket” the root

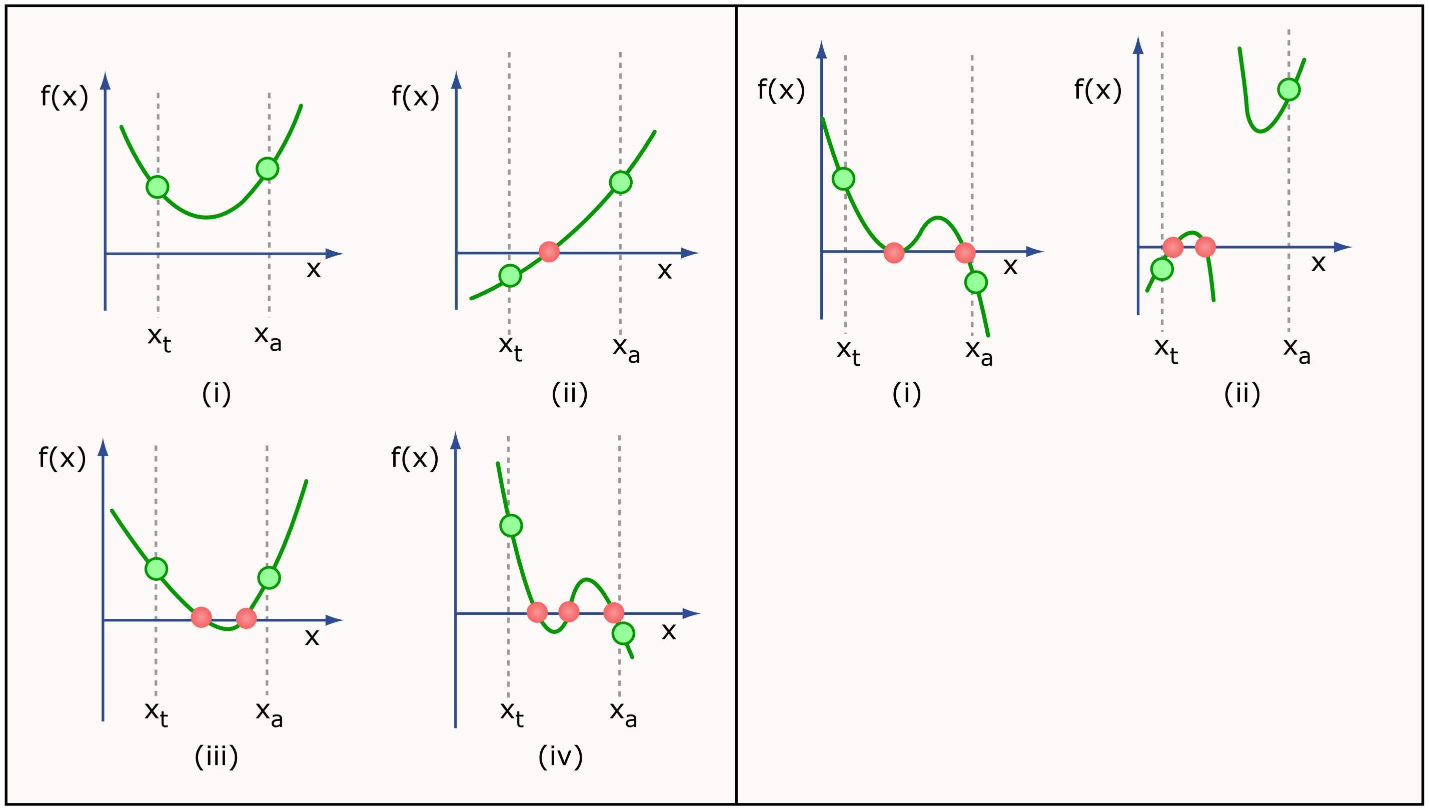


Image by MIT OpenCourseWare.

These graphs show general ways in which roots may occur on a bounded interval. In (i) and (ii), if the function is positive at both boundaries, there will be either an even number of roots or no roots; in the other two graphs, if the function has opposite signs at the boundary points, there will be an odd number of roots within the interval.

These graphs show some exceptions to the general cases illustrated on the left. In (i), a multiple root occurs where the function is tangential to the x axis. There are an even number of intersections for the interval. In (ii) is depicted a discontinuous function with an even number of roots and end points of opposite signs. Finding the roots in such cases requires special strategies.

Roots of Nonlinear Equations: Bracketing Methods

$$f(x) = 0$$

Example – Square root

$$x^2 - a = 0 \Rightarrow x = \sqrt{a}$$

Heron's Principle

$$x > 0$$

$$x^2 - a = 0 \Leftrightarrow x = \frac{a}{x}$$

Guess root

If $x_0 > \sqrt{a} \quad \Leftrightarrow \quad \frac{a}{x_0} < \sqrt{a}$

If $x_0 < \sqrt{a} \quad \Leftrightarrow \quad \frac{a}{x_0} > \sqrt{a}$

Mean is better guess

$$x_1 = (x_0 + \frac{a}{x_0}) / 2$$

Iteration Formula

$$x_k = (x_{k-1} + \frac{a}{x_{k-1}}) / 2$$

heron.m

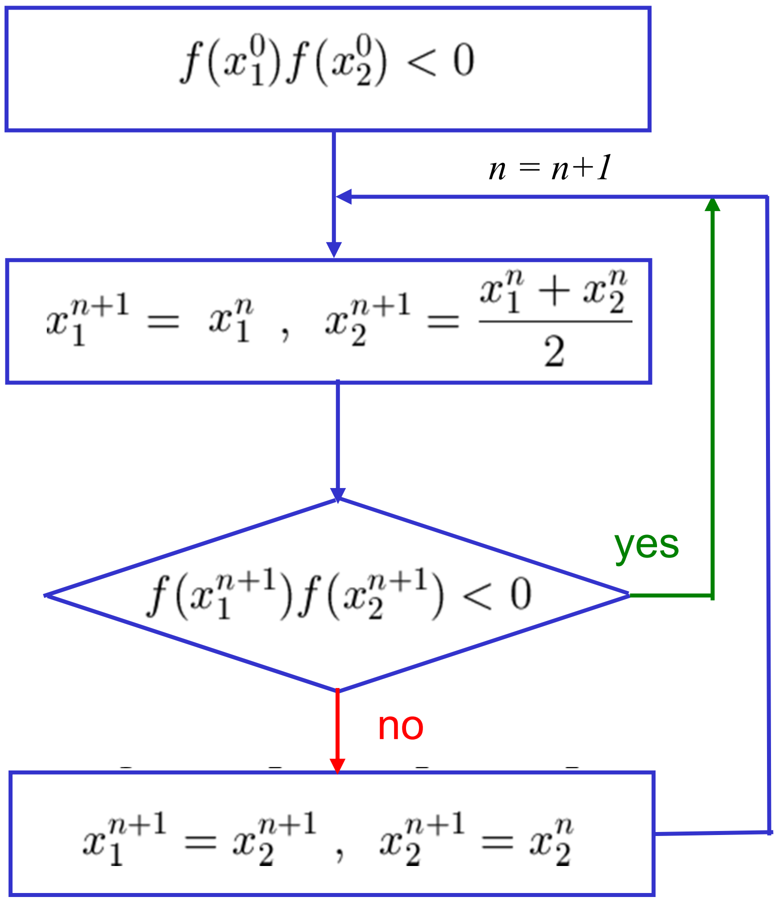
```
a=2;
n=6;
g=2;
% Number of Digits
dig=5;
     sq(1)=g;
     for i=2:n
      sq(i)= 0.5*radd(sq(i-1),a/sq(i-1),dig);
     end
     '      i       value   '
     [ [1:n]' sq']
     hold off
     plot([0 n],[sqrt(a) sqrt(a)],'b')
     hold on
     plot(sq,'r')
     plot(a./sq,'r-.')
     plot((sq-sqrt(a))/sqrt(a),'g')
     legend('sqrt','xn','s/xn','Relative Err')
     grid on
```

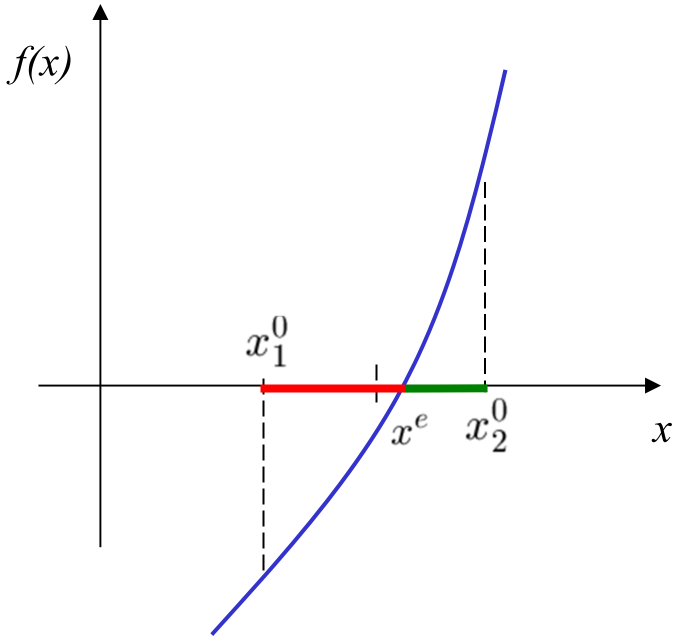
```
       i        value


    1.0000    2.0000
    2.0000    1.5000
    3.0000    1.4167
    4.0000    1.4143
    5.0000    1.4143
    6.0000    1.4143
```

Roots of Nonlinear Equations: Bracketing Methods
Bisection

Incremental search method defined as:
"Keep dividing in two the interval within which at least one root lies"

Algorithm

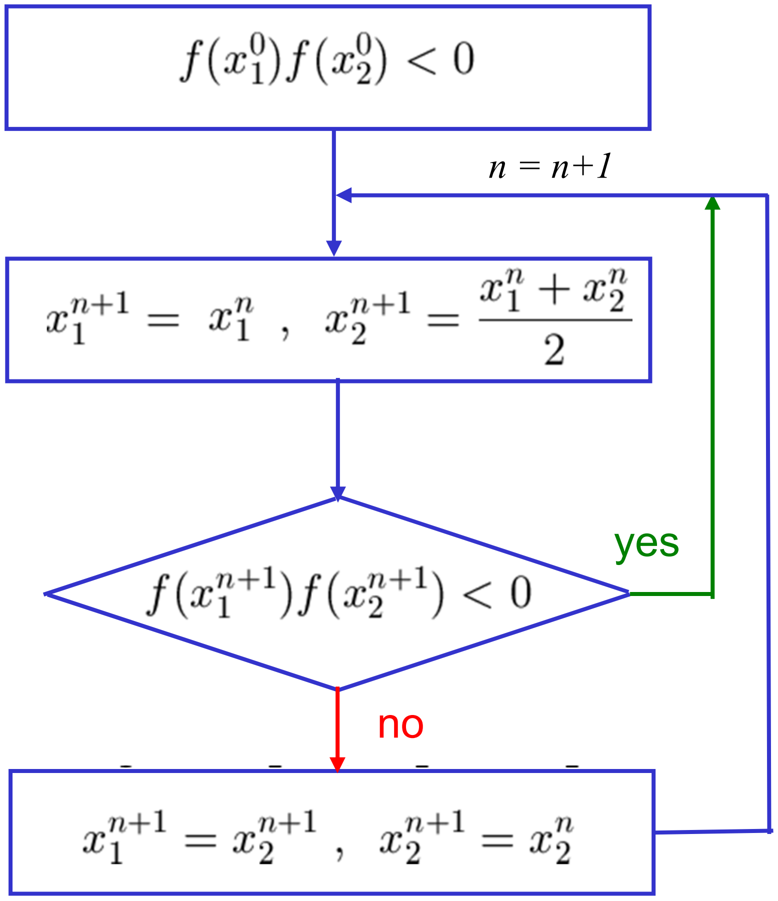

$f(x_1^0)f(x_2^0) < 0$
$n = n+1$
$x_1^{n+1} = x_1^n \; , \; x_2^{n+1} = \frac{x_1^n + x_2^n}{2}$
$f(x_1^{n+1})f(x_2^{n+1}) < 0$
yes
no
$x_1^{n+1} = x_2^{n+1} \; , \; x_2^{n+1} = x_2^n$



Roots of Nonlinear Equations: Bracketing Methods

Bisection

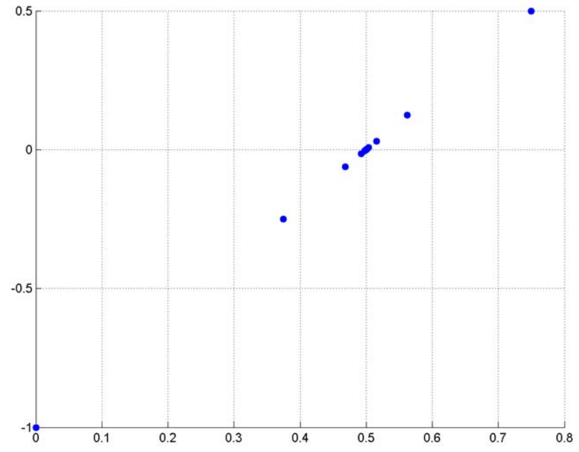
Algorithm


$f(x_1^0)f(x_2^0) < 0$
$n = n+1$
$x_1^{n+1} = x_1^n \;,\; x_2^{n+1} = \frac{x_1^n + x_2^n}{2}$
$f(x_1^{n+1})f(x_2^{n+1}) < 0$
yes
no
$x_1^{n+1} = x_2^{n+1} \;,\; x_2^{n+1} = x_2^n$

bisect.m

```
% Root finding by bi-section
f=inline(' a*x -1','x','a');
a=2
figure(1); clf; hold on
x=[0 1.5]; eps=1e-3;
err=max(abs(x(1)-x(2)),abs(f(x(1),a)-f(x(2),a)));
while (err>eps & f(x(1),a)*f(x(2),a) <= 0)
    xo=x; x=[xo(1) 0.5*(xo(1)+xo(2))];
    if ( f(x(1),a)*f(x(2),a) > 0 )
        x=[0.5*(xo(1)+xo(2)) xo(2)]
    end
    x
    err=max(abs(x(1)-x(2)),abs(f(x(1),a)-f(x(2),a)));
    b=plot(x,f(x,a),'.b'); set(b,'MarkerSize',20);
    grid on;
end
```

Stop criterion above based on Δx~0 and Δf(x)~0



Roots of Nonlinear Equations: Bracketing Methods
Bisection

- Stop criteria: when relative estimated error or relative added value is negligible:

$$\left|\varepsilon_a\right| = \left|\frac{\hat{x}_r^{\,n} - \hat{x}_r^{\,n-1}}{\hat{x}_r^{\,n}}\right| \leq \varepsilon_s$$

- Each successive iteration halves the maximum error, a general formula thus relates the error and the number of iterations:

$$n = \log_2\left(\frac{\Delta x^0}{E_{a,d}}\right)$$

where Δx^0 is an initial error and $E_{a,d}$ the desired error

Another Bracketing Method: The False Position Method (Regula Falsi)

- Bisection is a brute force scheme
  - Somewhat inefficient approach
  - Doesn't account for magnitudes of $f(x_L)$ and $f(x_U)$

- False Position

$$x_r = x_U - \frac{f(x_U)\,(x_L - x_U)}{f(x_L) - f(x_U)}$$

$$f(x_r) = f(x_U) + \frac{f(x_U) - f(x_L)}{x_U - x_L}(x_r - x_U) \cong 0$$

$$\frac{f(x_L)}{(x_r - x_L)} = \frac{f(x_U)}{(x_r - x_U)}$$

$$x_r = x_U - \frac{f(x_U)\,(x_L - x_U)}{f(x_L) - f(x_U)}$$

A graphical depiction of the false position method, Regula Falsi. Shaded in green are the triangles used to derive the method.

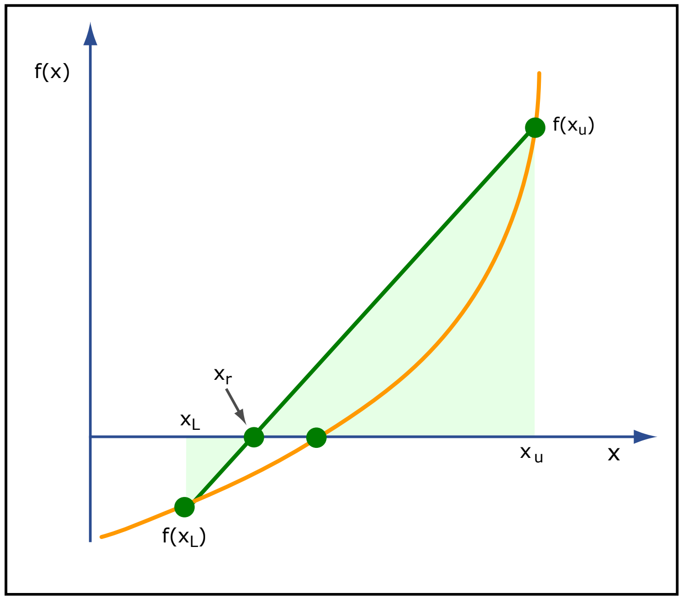


Image by MIT OpenCourseWare.

Comments on Bracketing Methods (Bisection and False Postion)

- Bisection and False Position:
  - Convergent methods, but one needs to minimize function evaluations
- Error of False Position can decrease much faster than that of Bisection, but if the function is far from a straight line, False Position will be slow (one of the end point remains fixed)
- Remedy: "Modified False Position"
  - e.g. if end point remains fixed for a few iterations, reduce end-point function value at end point, e.g. divide interval by 2
- One can not really generalize for Root Finding methods (results are function depend)
- Always substitute estimate of root x_r at the end to check how close estimate is to $f(x_r)=0$

Roots of Nonlinear Equations: **Open Methods**

Fixed Point Iteration (General Method or Picard Iteration)

Non-linear Equation

$$f(x) = 0$$

Goal: Converging series

$$x_0, x_1, \ldots x_n \rightarrow x^e \;, \quad n \rightarrow \infty$$

Rewrite Problem

$$f(x) = 0 \Leftrightarrow g(x^e) = x^e$$

Example

$$g(x) = x + c \cdot f(x)$$

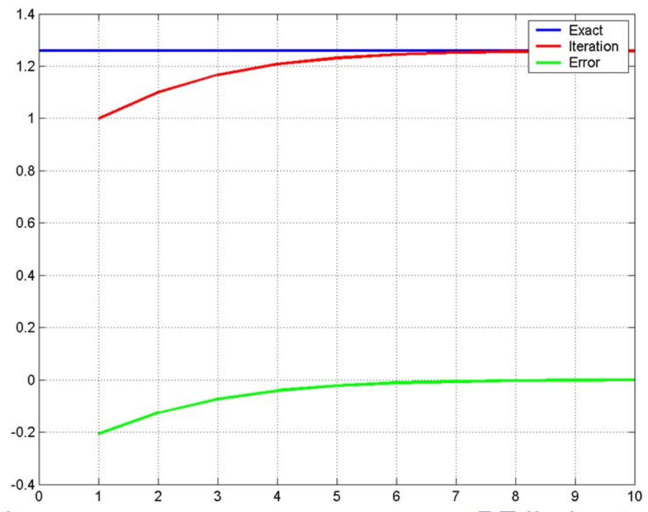
Iteration

$$x_n = g(x_{n-1})$$

Example: Cube root

cube.m

```
% f(x) = x^3 - a = 0
% g(x) = x + C*(x^3 - a)
a=2;
n=10;
g=1.0;
C=-0.1;
     sq(1)=g;
     for i=2:n
      sq(i)= sq(i-1) + C*(sq(i-1)^3 -a);
     end
     hold off
     plot([0 n],[a^(1./3.) a^(1/3.)],'b')
     hold on
     plot(sq,'r')
     plot( (sq-a^(1./3.))/(a^(1./3.)),'g')
     grid on
```



Roots of Nonlinear Equations:
Bracketing vs. Open Methods in Graphics

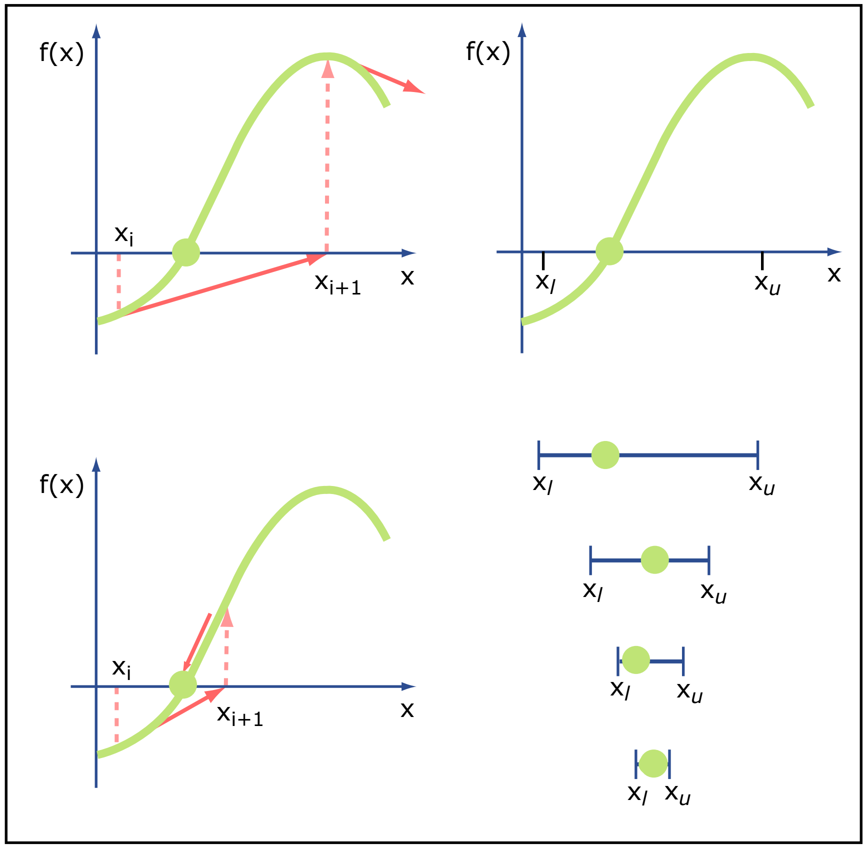


Image by MIT OpenCourseWare.

A graphical depiction of the fundamental differences between the bracketing method (shown on the right) and the open methods (shown on the left). On the right, the root is constrained within the interval; on the left, a formula is used to project iteratively between x_i and x_{i+1} and can diverge or converge rapidly.

Open Methods (Fixed Point Iteration) Stop-criteria

"Unrealistic" stop-criteria: continue while

$$x_{k+1} \neq x_k$$

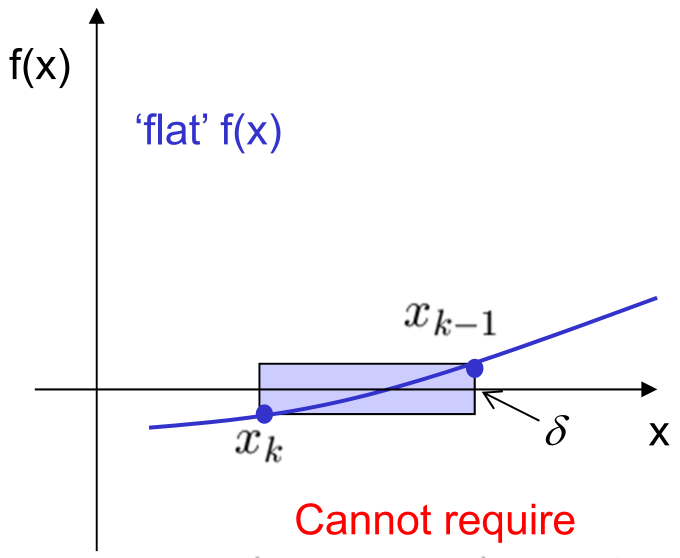
Realistic stop-criteria

$$|x_k - x_{k-1}| \ < \ \delta$$

$$|f(x) - f(x_k)| \ < \ \delta$$

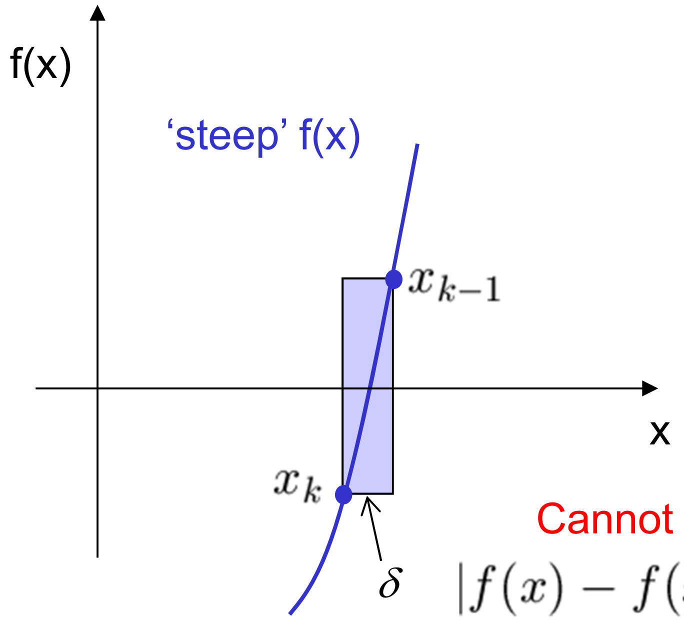
Machine Epsilon

$$\boxed{\text{Use an "or" combination of the two criteria}}$$



Cannot require

$$|x_k - x_{k-1}| \ < \ \delta$$



Cannot require

$$|f(x) - f(x_k)| \ < \ \delta$$