REVIEW Lecture 3

• Truncation Errors, Taylor Series and Error Analysis
  – Taylor series:
    \[ f(x_{i+1}) = f(x_i) + \Delta x f'(x_i) + \frac{\Delta x^2}{2!} f''(x_i) + \frac{\Delta x^3}{3!} f'''(x_i) + \ldots + \frac{\Delta x^n}{n!} f''(x_i) + R_n \]
    \[ R_n = \frac{\Delta x^{n+1}}{n+1!} f^{(n+1)}(\xi) \]
  – 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 \( \varepsilon_i \)'s are magnitudes of errors on \( x_i \)'s, what is the error on \( y \)?
    • The Differential Formula:
      \[ \varepsilon_y \leq \sum_{i=1}^{n} \left| \frac{\partial f(x_1, \ldots, x_n)}{\partial x_i} \right| \varepsilon_i \]
    • The Standard Error (statistical formula):
      \[ E(\Delta y) \leq \sqrt{\sum_{i=1}^{n} \left( \frac{\partial f}{\partial x_i} \right)^2 \varepsilon_i^2} \]
    – 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
• Roots of nonlinear equations

  – Bracketing Methods:
    • Systematically reduce width of bracket, track error for convergence:
      \[ |\varepsilon_a| = \left| \frac{x_R^n - x_l^n}{x_R^0} \right| \leq \varepsilon_s \]
    • Bisection: Successive division of bracket in half
      – determine next interval based on sign of:
        \[ f(x_{\text{mid-point}}^n) \]
      – Number of Iterations:
        \[ n = \log_2 \left( \frac{\Delta x^0}{E_{a,d}} \right) \]

  • False-Position (Regula Falsi): As Bisection, excepted that next \( x_r \) is the “linearized zero”, i.e. approximate function with straight line using its values at end points, and find its zero:
    \[ x_r = x_U - \frac{f(x_U)(x_L - x_U)}{f(x_L) - f(x_U)} \]

  – “Open” Methods:
    • Systematic “Trial and Error” schemes, don’t require a bracket
    • Computationally efficient, don’t always converge
    • Fixed Point Iteration (General Method or Picard Iteration):
      \[ x_{n+1} = g(x_n) \quad \text{or} \quad x_{n+1} = x_n - h(x_n) f(x_n) \]
Numerical Fluid Mechanics: Lecture 4 Outline

- 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)

Reference: Chapra and Canale, Chaps 3, 4 and 5
Open Methods (Fixed Point Iteration)

Convergence Theorem

**Hypothesis:**
\( g(x) \) satisfies the following Lipschitz condition:

There exist a \( k \) such that if

\[
  x \in I
\]

then

\[
  |g(x) - g(x^e)| = |g(x) - x^e| \leq k|x - x^e|
\]

Then, one obtains the following Convergence Criterion:

\[
x_{n-1} \in I \Rightarrow |x_n - x^e| = |g(x_{n-1}) - x^e| \leq k|x_{n-1} - x^e|
\]

Applying this inequality successively to \( x_{n-1}, x_{n-2}, \text{ etc.} \):

\[
  |x_n - x^e| \leq k^n|x_0 - x^e|
\]

**Convergence**

\( x_0 \in I, \ k < 1 \)
If the derivative of $g(x)$ exists, then the Mean-value Theorem gives:

$$\exists \xi \in [x, x^e] \mid g(x) - g(x^e) = g'(\xi)(x - x^e)$$

$$\begin{cases} 
  x < \xi < x^e \\
  x^e < \xi < x 
\end{cases}$$

Hence, a Sufficient Condition for Convergence

If $|g'(x)|_{x \in I} \leq k < 1 \Rightarrow |g(x) - x^e| \leq k|x - x^e|$
Open Methods (Fixed Point Iteration)

Example: Cube root

\[ x^3 - 2 = 0 \Rightarrow x^e = 2^{1/3} \]

Rewrite

\[ g(x) = x + C(x^3 - 2) \]

\[ g'(x) = 3Cx^2 + 1 \]

Convergence, for example in the \(0 < x < 2\) interval?

\[ |g'(x)| < 1 \iff -2 < 3Cx^2 < 0 \]

For \(0 < x < 2\) \(\Rightarrow -1/6 < C < 0\)

\[ C = -\frac{1}{6} \Rightarrow x_{n+1} = g(x_n) = x_n - \frac{1}{6}(x_n^3 - 2) \]

Converges more rapidly for small \(|g'(x)|\)

\[ g'(1.26) = 3C \cdot 1.26^2 + 1 = 0 \iff C = -0.21 \]

Ps: this means starting in smaller interval than \(0 < x < 2\) (smaller \(x\)’s)

\[
\begin{align*}
n &= 10; \\
g &= 1.0; \\
C &= -0.21; \\
sq(1) &= g; \\
\text{for } i &= 2:n \\
sq(i) &= sq(i-1) + C*(sq(i-1)^3 - a); \\
\end{align*}
\]

```
cube.m
for i=2:n
    sq(i) = sq(i-1) + C*(sq(i-1)^3 - a);
end
hold off
f=plot([0 n],[a^(1./3.) a^(1/3.)],'b')
set(f,'LineWidth',2);
hold on
f=plot(sq,'r')
set(f,'LineWidth',2);
f=plot((sq-a^(1./3.))/(a^(1./3.)),'g')
set(f,'LineWidth',2);
legend('Exact','Iteration','Error');
f=title(['a = ' num2str(a) ', C = ' num2str(C)])
set(f,'FontSize',16);
going
```

a = 2, C = -0.21

![Graph showing convergence](image)
Open Methods (Fixed Point Iteration)

Converging, but how close: What is the error of the estimate?

Consider the Absolute error:

\[ |x_{n-1} - x^e| \leq |x_{n-1} - x_n| + |x_n - x^e| \]
\[ = |x_{n-1} - x_n| + |g(x_{n-1}) - g(x^e)| \]
\[ = |x_{n-1} - x_n| + |g'(\xi)||x_{n-1} - x^e| \]
\[ \leq |x_{n-1} - x_n| + k|x_{n-1} - x^e| \]

\[ \Rightarrow \]
\[ |x_{n-1} - x^e| \leq \frac{1}{1-k} |x_{n-1} - x_n| \]

Hence, at iteration n:

\[ |x_n - x^e| \leq k|x_{n-1} - x^e| \leq \frac{k}{1-k} |x_{n-1} - x_n| \]

**Fixed-Point Iteration Summary**

\[ x_{n+1} = g(x_n) \]

Absolute error:

\[ |x_n - x^e| \leq \frac{k}{1-k} |x_{n-1} - x_n| \]

Convergence condition:

\[ |g'(x)| \leq k < 1 \text{, } x \in I \]

Note: Total compounded error due to round-off is bounded by

\[ \varepsilon_{r-o} / (1-k) \]
Order of Convergence for an Iterative Method

• The speed of convergence for an iterative method is often characterized by the so-called Order of Convergence

• Consider a series $x_0, x_1, \ldots$ and the error $e_n = x_n - x^e$. If there exist a number $p$ and a constant $C \neq 0$ such that

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

then $p$ is defined as the Order of Convergence or the Convergence exponent and $C$ as the asymptotic constant

- $p=1$ linear convergence,
- $p=2$ quadratic convergence,
- $p=3$ cubic convergence, etc

• Note: Error estimates can be utilized to accelerate the scheme (Aitken’s extrapolation, of order $2p-1$, if the fixed-point iteration is of order $p$)

• Fixed-Point: often linear convergence, $e_{n+1} = g'(\xi) e_n$
“Open” Iterative Methods: Newton-Raphson

- So far, the iterative schemes to solve $f(x) = 0$ can all be written as

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

- Newton-Raphson: one of the most widely used scheme

- Extend the tangent from current guess $x_n$ to find point where x axis is crossed:

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

$$f(x_{n+1}) = f(x_n) + f'(x_n) (x_{n+1} - x_n) = 0 \implies$$
Newton-Raphson Method:
Its derivation based on the local derivative and the rate of convergence

Non-linear Equation
\[ f(x) = 0 \Leftrightarrow x = g(x) \]

Convergence Criteria
\[ |g'(x_n)| < k < 1 \Rightarrow |x_n - x^e| \leq k|x_{n-1} - x^e| \]

Fast Convergence
\[ |g'(x^e)| = 0 \]

\[ g(x) = x + h(x)f(x), \quad h(x) \neq 0 \]

\[ g'(x^e) = 1 + h(x^e)f'(x^e) + h'(x^e)f(x^e) \]
\[ = 1 + h(x^e)f'(x^e) \]

\[ g'(x^e) = 0 \Leftrightarrow h(x) = -\frac{1}{f'(x)} \]

Newton-Raphson Iteration
\[ x_{n+1} = g(x_n) = x_n - \frac{f(x_n)}{f'(x_n)} \]
Newton-Raphson Method: Example

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

Example – Square Root

\[ x = \sqrt{a} \Leftrightarrow f(x) = x^2 - a = 0 \]

Newton-Raphson

\[ x_{n+1} = x_n - \frac{x_n^2 - a}{2x_n} = \frac{1}{2} \left( x_n + \frac{a}{x_n} \right) \]

Same as Heron’s formula

```matlab
a=26; n=10; g=1;
sq(1)=g;
for i=2:n
    sq(i)= 0.5*(sq(i-1) + a/sq(i-1));
end
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')
grid on
```

\[ a=26; \]
\[ n=10; \]
\[ g=1; \]
\[ \text{sqr.m} \]
Newton-Raphson Example: Its use for divisions

\[ x = \frac{1}{a} \]

\[ f(x) = ax - 1 = 0 \]

\[ f'(x) = a \]

\[ \frac{f(x)}{f'(x)} = \frac{ax - 1}{a} = x^e(ax - 1) \approx x(ax - 1) \]

which is a good approximation if \( \frac{|x - x^e|}{|x^e|} \ll 1 \)

Hence, Newton-Raphson for divisions:

\[ x_{n+1} = x_n - x_n(ax_n - 1) \]

div.m

```matlab
a=10;
n=10;
g=0.19;
sq(1)=g;
for i=2:n
    sq(i)=sq(i-1) - sq(i-1)*(a*sq(i-1) -1) ;
end
hold off
plot([0 n],[1/a 1/a],'b')
hold on
plot(sq,'r')
plot((sq-1/a)*a,'g')
grid on
legend('Exact','Iteration','Rel Error');
title(['x = 1/' num2str(a)])
```

\( x = 1/10 \)
Newton-Raphson: Order of Convergence

Define:

\[ \epsilon_n = x_n - x^e \]

Taylor Expansion:

\[ g(x_n) = g(x^e) + \epsilon_n g'(x^e) + \frac{1}{2} \epsilon_n^2 g''(x^e) \cdots \]

Since \( g'(x^e) = 0 \), truncating third order terms and higher, leads to a second order expansion:

\[ g(x_n) - g(x^e) \approx \frac{1}{2} \epsilon_n^2 g''(x^e) \]

\[ \Rightarrow \]

\[ \epsilon_{n+1} = x_{n+1} - x^e \approx \frac{1}{2} \epsilon_n^2 g''(x^e) \]

Relative Error:

\[ \frac{\epsilon_{n+1}}{|x^e|} \approx \frac{1}{2} |x^e| g''(x^e) \left( \frac{\epsilon_n}{|x^e|} \right)^2 = A(x^e) \left( \frac{\epsilon_n}{|x^e|} \right)^2 \]

\[ \epsilon_{n+1} \approx \epsilon_n A \]

Note:

\[ g(x) = x - \frac{f}{f'} \quad , \quad g'(x) = \frac{f f''}{f'^2} \quad \text{and} \quad g''(x) = \frac{f''}{f'} + \frac{f f''}{f'^2} + f(...) \]
Newton-Raphson: Issues

a) Inflection points in the vicinity of the root, i.e. $f''(x^e) = 0$

b) Iterations can oscillate around a local minima or maxima

c) Near zero slope encountered

d) Zero slope at the root

Image by MIT OpenCourseWare.

Four cases in which there is poor convergence with the Newton-Raphson method.
Roots of Nonlinear Equations: 
Secant Method

1. In Newton-Raphson we have to evaluate 2 functions \( f(x_n), \ f'(x_n) \)

2. \( f(x_n) \) may not be given in closed, analytical form, i.e. in CFD, it is often a result of a numerical algorithm

Approximate Derivative:
\[
f'(x_n) \approx \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}
\]

Secant Method Iteration:
\[
x_{n+1} = x_n - \frac{f(x_n)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})} = \frac{f(x_n)x_{n-1} - f(x_{n-1})x_n}{f(x_n) - f(x_{n-1})}
\]

Only 1 function call per iteration! : \( f(x_n) \)
Secant Method: Order of convergence

Absolute Error

\[ \epsilon_n = x_n - x^e \]

\[ \epsilon_{n+1} = x_{n+1} - x^e = \frac{f(x^e + \epsilon_n)(x^e + \epsilon_{n-1}) - f(x^e + \epsilon_{n-1})(x^e + \epsilon_n)}{f(x^e + \epsilon_n) - f(x^e + \epsilon_{n-1})} - x^e \]

Using Taylor Series, up to 2\(^{nd}\) order

**Absolute Error**

\[ \epsilon_{n+1} \approx \frac{1}{2} \epsilon_{n-1} \epsilon_n \frac{f''(x^e)}{f'(x^e)} \]

**Relative Error**

\[ \frac{\epsilon_{n+1}}{|x^e|} \approx \frac{\epsilon_{n-1}}{|x_e|} \frac{\epsilon_n}{|x_e|} \frac{f''(x^e)}{2f'(x^e)} x^e \]

Convergence Order/Exponent

By definition:

\[ \epsilon_n = A(x^e) \epsilon_{n-1}^m \Rightarrow \epsilon_{n-1} = \left( \frac{1}{A} \epsilon_n \right)^{1/m} = B(x^e) \epsilon_n^{1/m} \]

Then:

\[ \epsilon_{n+1} = C(x^e) \epsilon_n \epsilon_{n-1} = D(x^e) \epsilon_n^{1/m} \epsilon_n^{1/m} = D(x^e) \epsilon_n^{1+1/m} \]

\[ \Rightarrow 1 + \frac{1}{m} = m \iff m = \frac{1}{2} (1 + \sqrt{5}) \approx 1.62 \]

Error improvement for each function call

**Secant Method**

\[ \epsilon_{n+1}^* \approx \epsilon_n^{1.62} \]

**Newton-Raphson**

\[ \epsilon_{n+1}^* = \epsilon_n^2 \]

2.29 Numerical Fluid Mechanics PFJL Lecture 4, 16
Roots of Nonlinear Equations

Multiple Roots

p-order Root

\[ f(x) = (x - x^e)^p f_1(x) \ , \ f_1(x^e) \neq 0 \]

Newton-Raphson

\[
x_{n+1} = g(x_n) = x_n - \frac{(x_n - x^e)^p f_1(x_n)}{p(x_n - x^e)^{p-1} f_1(x_n) + (x_n - x^e)^p f'(x_n)}
\]

\[
=> x_{n+1} = x_n - \frac{(x_n - x^e)f_1(x_n)}{pf_1(x_n) + (x_n - x^e)f'(x_n)}
\]

Convergence

\[
|x_{n+1} - x^e| \leq k|x_n - x^e| \simeq |g'(x^e)||x_n - x^e|
\]

\[
g'(x^e) = 1 - \frac{1}{p}
\]

Slower convergence the higher the order of the root
Roots of Nonlinear Equations

Bisection

Algorithm

\[ f(x_1^0) f(x_2^0) < 0 \]

\[ n = n + 1 \]

\[ x_1^{n+1} = x_1^n, \quad x_2^{n+1} = \frac{x_1^n + x_2^n}{2} \]

\[ f(x_1^{n+1}) f(x_2^{n+1}) < 0 \]

Yes

\[ x_1^{n+1} = x_2^{n+1}, \quad x_2^{n+1} = x_2^n \]

No

Less efficient than Newton-Raphson and Secant methods, but often used to isolate interval with root and obtain approximate value. Then followed by N-R or Secant method for accurate root.
Useful reference tables for this material: