Roots of Equations

Larry Caretto
Mechanical Engineering 309

Numerical Analysis of
Engineering Systems

February 10, 2014

Outline
- Review last lecture
- Continue numerical solution of algebraic equations
- Desired accuracy in iterations
- Methods for roots of equations
  - Bisection
  - Secant method
  - Newton’s Method
  - False position (regula falsi)
  - Successive substitution

Review Computing
- Representation of numbers to any base
- Use of binary storage in computers
- For integer numbers the bit size of the storage limits the range (VBA integer from $-32,768$ to $32,767$ and long from $-2,147,483,648$ to $2,147,483,647$
- For real numbers storage size affects significant figures and magnitude
  - Single $1.4 \times 10^{-45}$ to $3.4 \times 10^{38}$ (~7 sig figs)
  - Double $4.9 \times 10^{-324}$ to $1.8 \times 10^{308}$ (~15 sig figs)

Review Solution of Equations
- Want solution to equations like $ax = \sin(x)$ for a given value of $a$
- It is one equation in one unknown, but we cannot get an explicit solution of the form $x = \text{calculated result}$
  - It is also possible that there will be more than one solution to the equation
  - Root is another word for solution
- Can look at solution graphically (like a graphing calculator)

Review Finding Roots
- Can write all equations as $f(x) = 0$
  - Simply move all terms to left-hand side
- Use trial and error algorithms
  - Two classes of methods
    - One initial guess
    - Two initial guesses that “bracket” root (i.e. root lies between the initial guesses)
  - Methods that bracket root are usually slower but surer
  - May need separate iteration process to get initial guesses bracketing root
**Review Bracketing a Root**

- Basic idea: when a root, f(x) = 0, is bracketed the values of f(x) on opposite sides of the root have opposite signs.
- If f(x)f(x + Δx) < 0 there is a root, f(x) = 0, between x and x + Δx.

**Review Process and Notation**

- Iteration process has one or more initial guesses, called x₀, x₁, etc. of root.
  - Usually have one or two initial guesses.
- Iteration procedure that uses previous estimates, x_k, x_{k-1}, etc. to find a new value x_{k+1}.
  - In most cases two previous iterations are used, but some methods use one.
- Iterations continue until x value is found that gives desired accuracy.

**Review Process and Notation II**

- Some algorithms maintain two current estimates that bracket a root.
- The notation for these estimates is x⁺ and x⁻ where f(x⁺) > 0 and f(x⁻) < 0.
- In these algorithms the iteration computes a value of x_{new} from x⁺ and x⁻.
- The value of x_{new} for iteration k is used as the iteration value x_{k+1} in error calculations on next slide.

**What is Desired Accuracy**

- Three measures:
  - Small value of f: |f| < ε₁.
  - Small value of change in x between iterations: |x_{k+1} - x_k| < ε₂.
  - Small value of relative change in x between iterations: |x_{k+1} - x_k| < ε₃|x_{k+1}|.
- Can use combinations of the three approaches as "or" tests.
- Third approach, relative change in x, is often most useful since the goal is to find an accurate value of x.

**Desired Accuracy in Code**

- Typically use relative error in x.
  - Converged is Boolean variable (true/false).
  - May also use condition directly in if statement.
- Converged = Abs(x_{k+1} - x_k) ≤ desiredRelativeError * Abs(x_{k+1}).
- Can also use dual test:
  - Converged = Abs(f(x_{k+1})) ≤ desiredErrorInF And _
  - Abs(x_{k+1} - x_k) ≤ desiredRelativeError * Abs(x_{k+1}).

**Termination Condition**

- What if you never get desired accuracy in iterations?
- Need to limit number of iterations.
- If maximum iterations are exceeded return an error message.
- Message may contain latest estimate and its relative error.
- Will show coding examples later.
### Review Secant Method

- Starts with two initial guesses, $x_0$ and $x_1$, which need not bracket root
- Each iteration uses values of two successive guesses, $x_k$ and $x_{k-1}$ to find new guess, $x_{k+1}$
- Approximate behavior of $f(x)$ near guess as straight line gives algorithm

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

### Review Secant Example

- Solve $f(x) = 0.05x - \sin(x) = 0$
- Pick $x_0 = 2$ and $x_1 = 2.5$
- $f(x_0) = f(2) = 0.05(2) - \sin(2) = -0.809$
- $f(x_1) = f(2.5) = 0.05(2.5) - \sin(2.5) = -0.473$

| $x_k$ | $f(x_k)$ | $|x_k - x_{k-1}|$ |
|------|----------|--------------------|
| 0    | 2        | -8.09E-01          |
| 1    | 2.5      | -4.73E-01          |
| 2    | 3.20493820516827 | 2.24E-01          |
| 3    | 2.97884928100731  | 7.0E-01          |
| 4    | 2.99134973893018  | 1.1E-02          |
| 5    | 2.99145653304953  | 1.4E-07          |
| 6    | 2.99145643398817  | 7.9E-03          |
| 7    | 2.99145643340058  | 0.00E+00         |
| 8    | 2.99145643340058  | 0.00E+00         |

### Review Secant Example II

- Note: All significant figures of calculator or spreadsheet used in calculations
- Rounding shown here to save space
- Apply general equation to second iteration

$$x_3 = 2.5 - (0.473)(2.5 - 2.5) = 3.20$$

$$f(x_3) = 0.05(3.20) - \sin(3.20) = 0.224$$

### Review Secant Example III

- Find root of $f(x) = 3xe^{-x^2} - 1 = 0$
- Start with initial guesses $x_0 = 0$ and $x_1 = 1$
- Repeat the following steps to get a new guess, $x_{k+1}$, from the guesses $x_k$ and $x_{k-1}$
- Compute $x_{k+1}$ from the equation

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

- Compute $f(x_{k+1})$
- Repeat the process $|x_{k+1} - x_k| < 0.003 |x_{k+1}|$

### Secant Method Exercise

- Find root of $f(x) = 3xe^{-x^2} - 1 = 0$
- $x_0 = 0$ and $x_1 = 1$
- $f(x_0) = 0.05(2.5) - \sin(2.5) = -0.473$
- $f(x_1) = 0.05(2.5) - \sin(2.5) = -0.473$
- $x_{k+1} = x_k - \frac{f(x_k) x_k - x_{k-1}}{f(x_k) - f(x_{k-1})}$
- $x_{k+1} = 2.5 - \frac{(-0.473)(2.5 - 2.5)}{-0.473 - (-0.473)} = 3.20$
- $f(x_{k+1}) = 0.05(3.20) - \sin(3.20) = 0.224$
Precedence Warning

- What is $-x^2$?
- Depends on the rules of precedence
- In VBA this is the same as $-(x^2)$
  - Exponentiation has higher precedence than unary minus
- On Excel worksheet this is the same as $(-x)^2$
  - Unary minus has higher precedence than exponentiation

Solution to Secant Exercise

\[ f(x_0) = 3x_0 e^{-x_0^2} - 1 = 3(0)e^{-0^2} - 1 = -1 \]
\[ f(x_1) = 3x_1 e^{-x_1^2} - 1 = 3(1)e^{-1^2} - 1 = 0.1036 \]
\[ x_{k+1} = x_k - \frac{x_k - x_{k-1}}{f(x_k) - f(x_{k-1})} \]
\[ x_2 = x_1 - (x_1 - x_0) \frac{x_1 - x_0}{f(x_1) - f(x_0)} \]
\[ x_2 = 1 - 0.1036 \frac{0.1036 - 1}{0.1036 - (-1)} = 0.9061 \]
\[ |x_2 - x_1| = |0.9061 - 1| = 0.939 > 0.003 \]
\[ |x_2| = 0.0027 \]
Not converged!

Solution to Secant Exercise II

\[ x_1 = 1 \quad f(x_1) = 0.1036 \]
\[ f(x_2) = 3x_2 e^{-x_2^2} - 1 = 3(0.9061)e^{-0.9061^2} - 1 = 0.1960 \]
\[ x_3 = x_2 - \frac{x_2 - x_1}{f(x_2) - f(x_1)} \]
\[ x_3 = 0.9061 - 0.1960 \frac{0.9061 - 1}{0.9061 - 0.1036} = 1.105 \]
\[ |x_3 - x_2| = |1.105 - 0.906| = 0.199 > 0.003 \]
Not converged!

Solution to Secant Exercise III

\[ x_2 = 0.9061 \quad f(x_2) = 0.1960 \]
\[ f(x_3) = 3x_3 e^{-x_3^2} - 1 = 3(1.105)e^{-1.105^2} - 1 = -0.0228 \]
\[ x_4 = x_3 - \frac{x_3 - x_2}{f(x_3) - f(x_2)} \]
\[ x_4 = 1.105 - (-0.0228) \frac{1.105 - 0.9061}{(-0.0228) - 0.1960} = 1.085 \]
\[ |x_4 - x_3| = |1.085 - 1.105| = 0.021 > 0.003 \]
Not converged!

Solution to Secant Exercise IV

\[ x_3 = 1.105 \quad f(x_3) = -0.0228 \]
\[ f(x_4) = 3x_4 e^{-x_4^2} - 1 = 3(1.085)e^{-1.085^2} - 1 = 3.449x10^{-3} \]
\[ x_5 = x_4 - \frac{x_4 - x_3}{f(x_4) - f(x_3)} \]
\[ x_5 = 1.085 - (3.449x10^{-3}) \frac{1.085 - 1.105}{3.449x10^{-3} - (-0.0228)} = 1.087 \]
\[ |x_5 - x_4| = |1.087 - 1.085| = 0.002 < 0.003 \]
Converged!

Solution to Secant Exercise V

<table>
<thead>
<tr>
<th>k</th>
<th>( x_k )</th>
<th>( f(x_k) )</th>
<th>( x_k - x_{k-1} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>( x_0 )</td>
<td>( x_1 )</td>
<td>( x_2 )</td>
</tr>
<tr>
<td>1</td>
<td>0.00000</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>2</td>
<td>0.9060939428197</td>
<td>0.196013</td>
<td>0.093910</td>
</tr>
<tr>
<td>3</td>
<td>1.1053561023637</td>
<td>0.022770</td>
<td>0.093910</td>
</tr>
<tr>
<td>4</td>
<td>1.0846185436620</td>
<td>0.003449</td>
<td>0.0033</td>
</tr>
<tr>
<td>5</td>
<td>1.087346719195</td>
<td>2.99E-05</td>
<td>0.002728</td>
</tr>
<tr>
<td>6</td>
<td>1.0873705979212</td>
<td>2.39E-05</td>
<td>0.002728</td>
</tr>
<tr>
<td>7</td>
<td>1.0873705643998</td>
<td>5.08E-13</td>
<td>0.002728</td>
</tr>
<tr>
<td>8</td>
<td>1.0873705644002</td>
<td>4.05E-13</td>
<td>0.002728</td>
</tr>
<tr>
<td>9</td>
<td>1.0873705644002</td>
<td>0.00000</td>
<td>0.002728</td>
</tr>
</tbody>
</table>
Basic idea: when a root, $f(x) = 0$, is bracketed, the values of $f(x)$ on opposite sides of the root have opposite signs.

- For bracketed root, define $x_j$ as value of $x$ such that $f(x_j) > 0$ and $x_j$ for $f(x_j) < 0$.

Bisection Method

- Start with two guesses $x_l$ and $x_r$ that bracket a root with $f(x_l) > 0$ and $f(x_r) < 0$.
- Repeat the following steps for the two current guesses $x_l$ and $x_r$:
  - Compute the midpoint $x_{\text{new}} = (x_l + x_r)/2$.
  - If $f(x_{\text{new}}) > 0$ replace $x_l$ by $x_{\text{new}}$.
  - Else replace $x_r$ by $x_{\text{new}}$.
- Continue this process until a value of $x_{\text{new}}$ gives desired accuracy.

Bisection Evaluation

- Slow to reach accurate solution.
- Does not make use of function values.

- First iteration had very low value of $f(x_{\text{new}})$, indicating that this guess was close to root, but only the sign of this root was used.
- There are times when the iteration keeps same value $x_j$ or $x_r$ the same for several iterations.
- But, this method will always converge; it will only blow up if there is a discontinuity.
Newton’s Method

- Based on taking two terms in a Taylor series \( f(x) = f(x_k) + (df/dx)_{x=x_k} (x - x_k) \)
- Solve this series approximation for \( x_{k+1} \) that sets \( f(x) = 0 \)
- \( 0 = f(x_k) + (df/dx)_{x=x_k} (x_{k+1} - x_k) \)
- \( x_{k+1} = x_k - \frac{f(x_k)}{(df/dx)_{x=x_k}} \)
- Newton’s method requires only one initial guess, but it requires the evaluation and calculation of the derivative

Newton’s Method II

- Make one initial guess, \( x_0 \)
- Repeat the following step to get a new guess, \( x_{k+1} \), from the old guess, \( x_k \)
- Compute \( x_{k+1} \) from the following equation

\[
x_{k+1} = x_k - \frac{f(x_k)}{(df/dx)_{x=x_k}}
\]
- Continue iteration until the value of \( x_{k+1} \) and/or \( f(x_{k+1}) \) is sufficiently accurate

Evaluation of Newton’s Method

- Sometimes used in computer code for problems that have to be solved often
- In these cases initial derivative calculation is small part of total work
- Otherwise use of simpler methods (like secant method) are faster because there is no need to evaluate derivative there
- Secant method is sometimes described as a Newton-like method where the derivative is estimated numerically rather than being calculated exactly as in Newton’s method
**False Position (Regula Falsi)**

- This method uses an linear expression for the new guess like the secant method
  \[ x_{k+1} = x_k - \frac{f(x_k) (x_k - x_{k-1})}{f(x_k) - f(x_{k-1})} \]
- Unlike the secant method, which uses the two most recent iterations, false position always uses the two most recent guesses which bracket the root

**Algorithm Detail**

- Repeat the following steps for the two current guesses \( x_1 \) and \( x_2 \):
  - Use a linear interpolation (like the secant method) to get \( x_{new} \) and \( f_{new} = f(x_{new}) \):
    \[ x_{new} = x_2 - \frac{f(x_2) (x_2 - x_1)}{f(x_2) - f(x_1)} \]
    \[ f_{new} = f(x_{new}) \]
  - If \( f_{new} > 0 \) replace \( x_1 \) by \( x_{new} \)
  - If \( f_{new} < 0 \) replace \( x_2 \) by \( x_{new} \)
  - Continue until iterations give desired accuracy for \( x \) or \( f \)

**False Position Algorithm**

- Start with two guesses \( x_1 \) and \( x_2 \) that bracket a root with \( f(x_1) > 0 \) and \( f(x_2) < 0 \)
  - Notation similar to bisection method
  - Use modified version of secant interpolation algorithm that works with the two guesses that bracket the root to get new guess, \( x_{new} \) and \( f_{new} = f(x_{new}) \):
    \[ x_{new} = x_2 - f(x_2) \frac{x_2 - x_1}{f(x_2) - f(x_1)} \]
    \[ f_{new} = f(x_{new}) \]

**False Position Example**

- Example: apply this algorithm to solve \( f(x) = 0.05x - \sin(x) = 0 \)
  - Initial values \( x_1 = 2 \), \( x_2 = 4 \) from bisection
  - \( f(2) = 0.05(2) - \sin(2) = -0.8093 \), so \( x_2 = 2 \)
  - \( f(4) = 0.05(4) - \sin(4) = 0.9568 \), so \( x_1 = 4 \)
  - Apply interpolation formula for first iteration:
    \[ x_{new} = x_2 - f(x_2) \frac{x_2 - x_1}{f(x_2) - f(x_1)} \]
    \[ f_{new} = f(x_{new}) = 0.05(2.916) - \sin(2.916) = -0.007139 \]

**False Position Algorithm II**

- Apply algorithm rules:
  - If \( f_{new} > 0 \) replace \( x_1 \) by \( x_{new} \)
  - If \( f_{new} < 0 \) replace \( x_2 \) by \( x_{new} \)
  - If \( f_{new} < 0 \) so \( x_{new} = 2.916 \) replaces \( x_2 = 2 \)
  - Retain previous value of \( x_1 = 4 \)
  - Continue iteration with these values:
    \[ x_{new} = x_1 - f(x_1) \frac{x_1 - x_2}{f(x_1) - f(x_2)} \]
    \[ x_{new} = 4 - 0.9568 \frac{4 - 2.916}{0.9568 - (-0.8093)} = 2.998 \]
    \[ f_{new} = f(2.998) = -0.006346 \]

**False Position Iterations**

- False position takes more iterations than secant, but is less prone to error
  - Note that last four iterations give new value of \( x_2 \) leaving same value for \( x_1 \)
### Successive Substitution

- Easiest algorithm, but very slow and may not converge
- Write equation in form \( x = g(x) \)
  - Example: to solve \( e^x = \sin(x) \) we can write \( x = \ln(\sin(x)) \) or \( x = \sin^{-1}(e^x) \)
- Algorithm: Start with initial guess \( x_0 \)
- Compute new guess, \( x_{k+1} = g(x_k) \)
- Repeat until two values of \( x \) are sufficiently close

### Successive Solution Example

- \( 0.05x = \sin(x) \) is \( x = g(x) = 20\sin(x) \)
- Take initial guess, \( x_0 = 3 \)
  - \( x_1 = 20\sin(x_0) = 20\sin(3) = 1.51 \times 10^{-1} \)
  - \( x_2 = 20\sin(x_1) = 20\sin(1.51 \times 10^{-1}) = 7.53 \times 10^{-3} \)
  - \( x_3 = 20\sin(x_2) = 20\sin(7.53 \times 10^{-3}) = 3.76 \times 10^{-4} \)
  - \( x_4 = 20\sin(x_3) = 20\sin(3.76 \times 10^{-4}) = 1.88 \times 10^{-5} \)
  - \( x_5 = 20\sin(x_4) = 20\sin(1.88 \times 10^{-5}) = 9.41 \times 10^{-7} \)
  - \( x_6 = 20\sin(x_5) = 20\sin(9.41 \times 10^{-7}) = 4.71 \times 10^{-8} \)
  - \( x_7 = 20\sin(x_6) = 20\sin(4.71 \times 10^{-8}) = 2.35 \times 10^{-9} \)

What is \( x_8 \)?

### First Quiz Results

- Number of students: 20
- Maximum possible score: 25
- Mean: 19.5
- Median: 22
- Standard deviation: 6.52

Grade distribution:

<table>
<thead>
<tr>
<th>Grade</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>11</td>
</tr>
<tr>
<td>1</td>
<td>12</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>16</td>
</tr>
<tr>
<td>4</td>
<td>11</td>
</tr>
<tr>
<td>5</td>
<td>22</td>
</tr>
<tr>
<td>6</td>
<td>22</td>
</tr>
<tr>
<td>7</td>
<td>22</td>
</tr>
<tr>
<td>8</td>
<td>23</td>
</tr>
<tr>
<td>9</td>
<td>25</td>
</tr>
<tr>
<td>10</td>
<td>25</td>
</tr>
<tr>
<td>11</td>
<td>25</td>
</tr>
</tbody>
</table>

### Comments on Quiz

- Cannot use statements like following with Elseif: \( \text{if } x<0 \text{ Then } y = 3 \)
- Do not use unnecessary $\$
- Can simplify Elseif clauses using information from previous conditions
- In VBA use log for natural logarithm
- Watch parens in \( 1/n+1 \) vs. \( 1/(n+1) \)