Root Finding Methods: Simple fixed point method and Newton-Raphson method

Instructor: Ashley Gannon

Fall 2020

Finishing up the Bisection method

Pseudocode example

```
//Input:  left bracket a, right bracket b
//Output:  void, print the root to the screen

//check that there is a root in your initial domain
if f(a)f(b) >= 0
   tell user to try a new domain and exit

//declare/define variables for the bisect method:
   //tolerance, error, c_old, and c_new

loop while error>tolerance
find the mindpoint c_new
//check if there is a root in the lower interval
   if f(a)f(c) < 0
      set b = c
   else
      set a = c
   //Alternatively, you could check the upper interval first.
   compute the error abs((c_new-c_old)/c_new)
   set c_old = c_new for the next iteration
print the root value to the screen
```

Let's code it!

Take the next 15 minutes and try to write the Bisection method, `bisect.cpp`. If you finish writing it early, apply it to the function $f(m)$ from last lecture on the bounds $[140, 150]$.

Post your code to the discussion board **Bisection method code**. Do NOT post screen shots of error messages or your code. Copy and paste the text from your code into the discussion board.

Using our Bisection method code, we found that the root is 142.7 kg with an error of less than 0.00001%.

How confident would you be using that **initial graphical approximation** from the last lecture: the maximum cutoff weight for this bungee jump is 145 kg/ 320lbs?

Using our Bisection method code, we found that the root is 142.7 kg with an error of less than 0.00001%.

How confident would you be using that **initial graphical approximation** from the last lecture: Telling your boss the maximum cutoff weight for this bungee jump is 145 kg/ 320lbs?

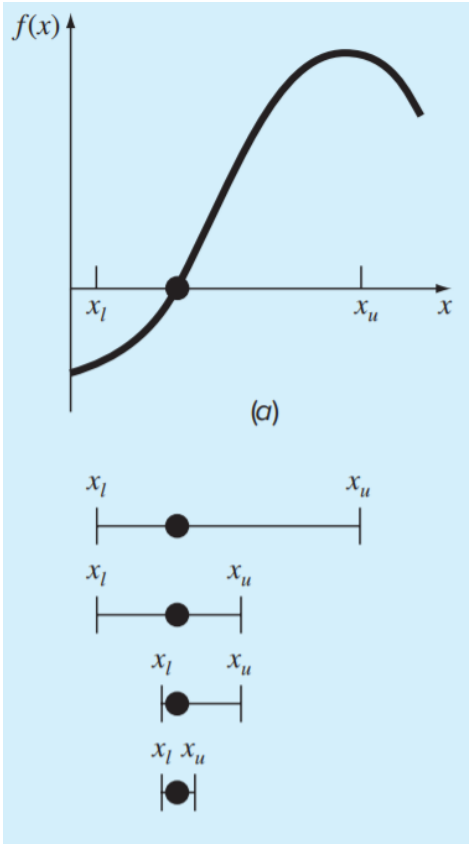
How confident would you be using the result from the bisection method?

Concept review

Bracketing methods

For the *bracketing methods* we discussed last class, we saw that

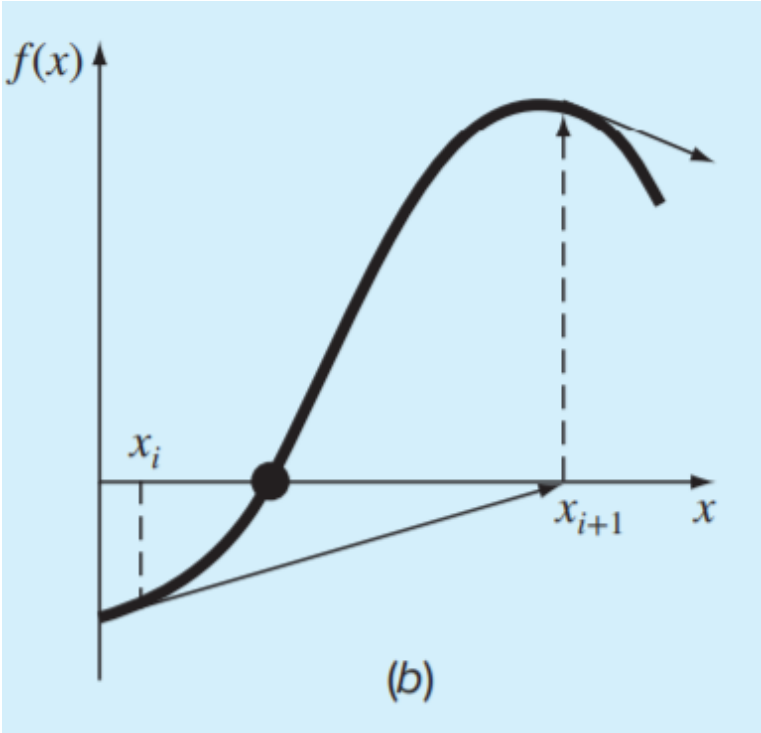
- the root is located within an interval prescribed by a lower and an upper bound.
- Repeated application of these methods always results in closer estimates of the true value of the root.
- Such methods are said to be convergent because they move closer to the true solution as the computation progresses.



Open methods

The *open methods* described in this lecture and the next lecture

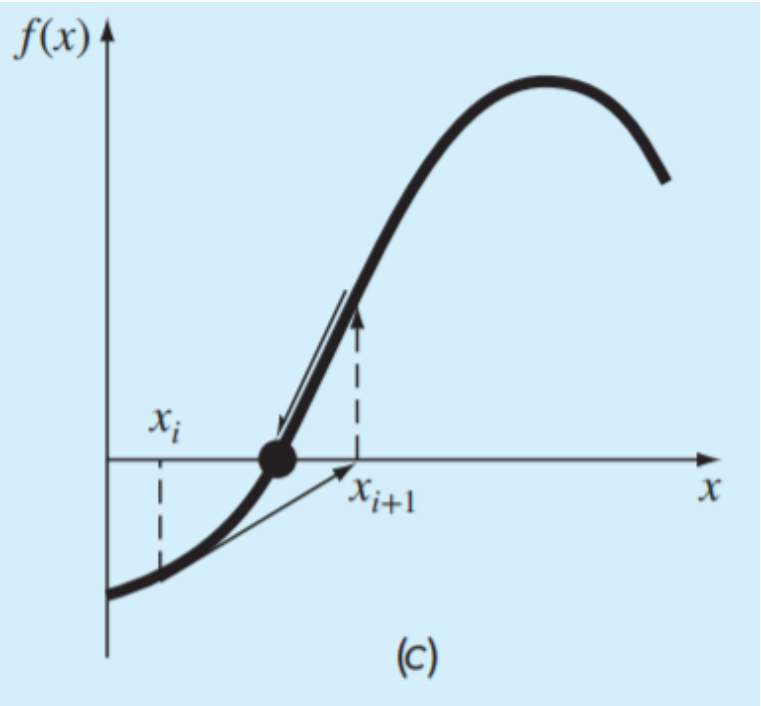
- will sometimes only require a single starting value
- will sometimes require two starting values, but they do not necessarily bracket the root
- will sometimes diverge away from the true solution as the computation progresses



Open methods

The *open methods* described in this lecture and the next lecture

- will sometimes only require a single starting value
- will sometimes require two starting values, but they do not necessarily bracket the root
- will sometimes diverge away from the true solution as the computation progresses
- will converge quicker than the bracketing methods when they converge.



Simple Fixed-Point Iteration

Simple fixed-point iteration method

Open methods employ a formula to predict the root. Such a formula can be developed for *simple fixed-point iteration* (also known as *one-point iteration* or *successive substitution*) by rearranging the function $f(x) = 0$ so that x is on the left-hand side of the equation.

Simple fixed-point iteration method

Open methods employ a formula to predict the root. Such a formula can be developed for *simple fixed-point iteration* (also known as *one-point iteration* or *successive substitution*) by rearranging the function $f(x) = 0$ so that x is on the left-hand side of the equation.

This transformation can be accomplished either by algebraic manipulation or by simply adding x to both sides of the original equation. The utility of this is that it provides a formula to predict a new value of x as a function of an old value of x.

Simple fixed-point iteration method

Open methods employ a formula to predict the root. Such a formula can be developed for *simple fixed-point iteration* (also known as *one-point iteration* or *successive substitution*) by rearranging the function $f(x) = 0$ so that x is on the left-hand side of the equation.

This transformation can be accomplished either by algebraic manipulation or by simply adding x to both sides of the original equation. The utility of this is that it provides a formula to predict a new value of x as a function of an old value of x.

Therefore, given an initial guess at the root x_i, the function can be used to compute a new estimate x_{i+1}.

Let's consider the function

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

If we let $f(x) = 0$ and move x to our left-hand side, we end up with

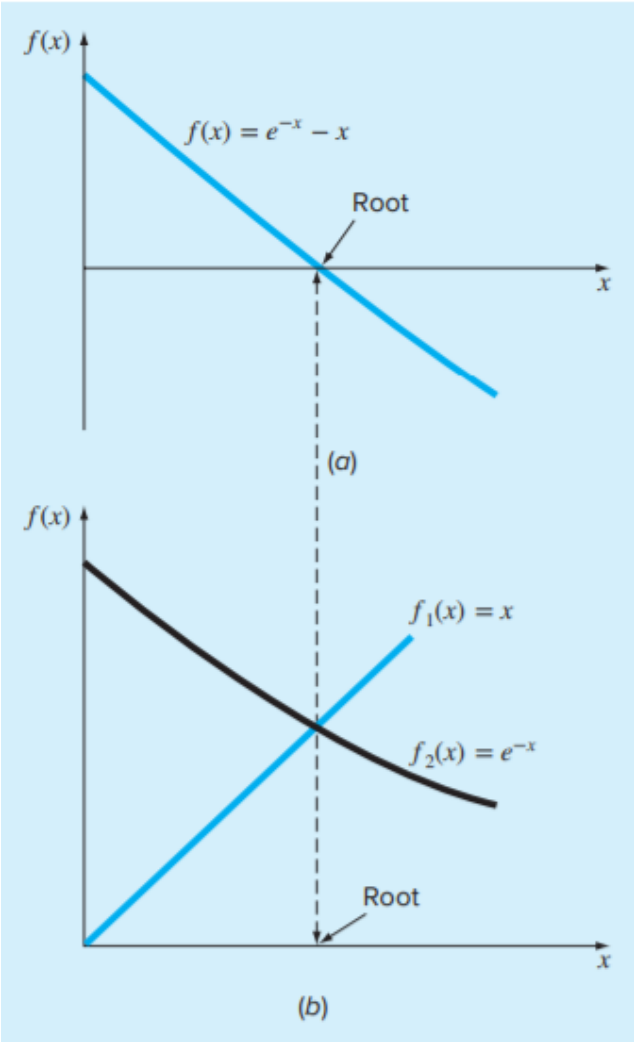
$$x = e^{-x}$$

The x on our left-hand side becomes x_{i+1}, the x on our right-hand side becomes x_i. This formula now predicts a new value of x as a function of the old value of x.

$$x_{i+1} = e^{-x_i}$$

Simple fixed-point iteration method

Thinking about this graphically,



The x values corresponding to the intersections of these functions represent the roots of $f(x) = 0$. Splitting the function into two component parts and plotting them is called the *two curve method*.

Simple fixed-point iteration method

Use simple fixed-point iteration to locate the root of $f(x) = e^{-x} - x$.

$$x_{i+1} = e^{-x_i}$$

i	x_i	x_{i+1}
0	0.000	$x_2 = e^{-x_0} = 1.000$

Simple fixed-point iteration method

Use simple fixed-point iteration to locate the root of $f(x) = e^{-x} - x$.

$$x_{i+1} = e^{-x_i}$$

i	x_i	x_{i+1}
0	0.000	$x_1 = e^{-x_0} = 1.000$
1	1.000	$x_2 = e^{-1.000} = 0.3679$

Simple fixed-point iteration method

Use simple fixed-point iteration to locate the root of $f(x) = e^{-x} - x$.

$$x_{i+1} = e^{-x_i}$$

i	x_i	x_{i+1}
0	0.000	$x_1 = e^{-x_0} = 1.000$
1	1.000	$x_2 = e^{-1.000} = 0.3679$
2	0.3679	$x_3 = e^{-0.3679} = 0.6922$

Simple fixed-point iteration method

Use simple fixed-point iteration to locate the root of $f(x) = e^{-x} - x$.

$$x_{i+1} = e^{-x_i}$$

i	x_i	x_{i+1}
0	0.000	$x_1 = e^{-x_0} = 1.000$
1	1.000	$x_2 = e^{-1.000} = 0.3679$
2	0.3679	$x_3 = e^{-0.3679} = 0.6922$
3	0.6922	$x_4 = e^{-0.6922} = 0.5005$

Simple fixed-point iteration method

Use simple fixed-point iteration to locate the root of $f(x) = e^{-x} - x$.

$$x_{i+1} = e^{-x_i}$$

i	x_i	x_{i+1}
0	0.000	$x_1 = e^{-x_0} = 1.000$
1	1.000	$x_2 = e^{-1.000} = 0.3679$
2	0.3679	$x_3 = e^{-0.3679} = 0.6922$
3	0.6922	$x_4 = e^{-0.6922} = 0.5005$
4	0.5005	$x_5 = e^{-0.5005} = 0.6062$

Simple fixed-point iteration method

Use simple fixed-point iteration to locate the root of $f(x) = e^{-x} - x$.

$$x_{i+1} = e^{-x_i}$$

i	x_i	x_{i+1}
0	0.000	$x_1 = e^{-x_0} = 1.000$
1	1.000	$x_2 = e^{-1.000} = 0.3679$
2	0.3679	$x_3 = e^{-0.3679} = 0.6922$
3	0.6922	$x_4 = e^{-0.6922} = 0.5005$
4	0.5005	$x_5 = e^{-0.5005} = 0.6062$
5	0.6062	$x_6 = e^{-0.6062} = 0.5454$
6	0.5454	$x_7 = e^{-0.5454} = 0.5796$
7	0.5796	$x_8 = e^{-0.5796} = 0.5601$
8	0.5601	$x_9 = e^{-0.5601} = 0.5711$
9	0.5711	$x_{10} = e^{-0.5711} = 0.5649$

Notice each iteration brings the estimate closer to the true value of the root: 0.56714329.

Simple fixed-point iteration method

Use simple fixed-point iteration to locate the root of $f(x) = e^{-x} - x$.

$$x_{i+1} = e^{-x_i}$$

i	x_i	x_{i+1}	$\frac{\epsilon_i}{\epsilon_{i-1}}$
0	0.000	$x_1 = e^{-x_0} = 1.000$	–
1	1.000	$x_2 = e^{-1.000} = 0.3679$	0.763
2	0.3679	$x_3 = e^{-0.3679} = 0.6922$	0.460
3	0.6922	$x_4 = e^{-0.6922} = 0.5005$	0.628
4	0.5005	$x_5 = e^{-0.5005} = 0.6062$	0.533
5	0.6062	$x_6 = e^{-0.6062} = 0.5454$	0.586
6	0.5454	$x_7 = e^{-0.5454} = 0.5796$	0.556
7	0.5796	$x_8 = e^{-0.5796} = 0.5601$	0.573
8	0.5601	$x_9 = e^{-0.5601} = 0.5711$	0.564
9	0.5711	$x_{10} = e^{-0.5711} = 0.5649$	0.569

Notice that the true percent relative error $\frac{\epsilon_i}{\epsilon_{i-1}}$ for each iteration is roughly proportional to the error from the previous iterations.

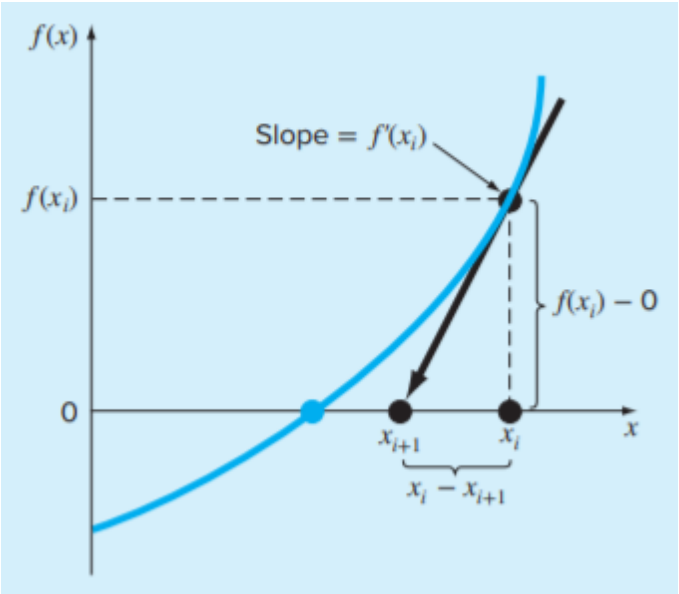
This property is called *linear convergence*, and it is characteristic of fixed-point iteration.

Deriving the Newton-Raphson method

Deriving the Newton-Raphson method

The Newton-Raphson method is one of the most widely used of all root-locating methods.

- If the initial guess at the root is x_i, a tangent can be extended from the point $(x_i, f(x_i))$
- The point where this tangent crosses the x axis usually represents an improved estimate of the root.
- The point where this tangent $(f'(x))$ crosses the x axis usually represents an improved estimate of the root.



Algebraically, the first derivative at x is equivalent to the slope

$$f'(x_i) = \frac{f(x_i) - 0}{x_i - x_{i+1}}$$

Which can be re-arranged to the form

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}$$

which is called the *Newton-Raphson formula*.

Newton Raphson method

Example problem

Let's look at this example again...

Use the Newton-Raphson method to locate the root of $f(x) = e^{-x} - x$.The derivative of $f(x)$ is $f'(x) = -e^{-x} - 1$.

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} = x_i - \frac{e^{-x_i} - x_i}{-e^{-x_i} - 1}$$

i	x_i	x_{i+1}
0	0.000000000	$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = 0.500000000$

Example problem

Let's look at this example again...

Use the Newton-Raphson method to locate the root of $f(x) = e^{-x} - x$.The derivative of $f(x)$ is $f'(x) = -e^{-x} - 1$.

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} = x_i - \frac{e^{-x_i} - x_i}{-e^{-x_i} - 1}$$

i	x_i	x_{i+1}
0	0.000000000	$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = 0.500000000$
1	0.500000000	$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} = 0.566311003$

Example problem

Let's look at this example again...

Use the Newton-Raphson method to locate the root of $f(x) = e^{-x} - x$.The derivative of $f(x)$ is $f'(x) = -e^{-x} - 1$.

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} = x_i - \frac{e^{-x_i} - x_i}{-e^{-x_i} - 1}$$

i	x_i	x_{i+1}
0	0.000000000	$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = 0.500000000$
1	0.500000000	$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} = 0.566311003$
2	0.566311003	$x_3 = x_2 - \frac{f(x_2)}{f'(x_2)} = 0.567143165$

Example problem

Let's look at this example again...

Use the Newton-Raphson method to locate the root of $f(x) = e^{-x} - x$.The derivative of $f(x)$ is $f'(x) = -e^{-x} - 1$.

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} = x_i - \frac{e^{-x_i} - x_i}{-e^{-x_i} - 1}$$

i	x_i	x_{i+1}
0	0.000000000	$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = 0.500000000$
1	0.500000000	$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} = 0.566311003$
2	0.566311003	$x_3 = x_2 - \frac{f(x_2)}{f'(x_2)} = 0.567143165$
3	0.567143165	$x_4 = x_3 - \frac{f(x_3)}{f'(x_3)} = 0.567143290$

Example problem

Let's look at this example again...

Use the Newton-Raphson method to locate the root of $f(x) = e^{-x} - x$.The derivative of $f(x)$ is $f'(x) = -e^{-x} - 1$.

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} = x_i - \frac{e^{-x_i} - x_i}{-e^{-x_i} - 1}$$

i	x_i	x_{i+1}
0	0.000000000	$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)} = 0.500000000$
1	0.500000000	$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)} = 0.566311003$
2	0.566311003	$x_3 = x_2 - \frac{f(x_2)}{f'(x_2)} = 0.567143165$
3	0.567143165	$x_4 = x_3 - \frac{f(x_3)}{f'(x_3)} = 0.567143290$
4	0.567143290	

Notice each iteration brings the estimate closer to the true value of the root: 0.56714329, and at a much faster rate than the simple-fixed point method.

Stopping criteria

We need to think about the stopping criteria for the Newton-Raphson method. Common stopping criteria used are

- checking if the function value is close to 0, $|f(x_i)| < \epsilon$,
- checking if the difference between successive iterations is small, $|x_{i+1} - x_i| < \epsilon$,
- Capping the number of iterations, $i > maxIter$.

The second option is often used instead of or in conjunction with the first option because it is possible for $f(x_i)$ to be close to 0, but x_i is not close to the root x^*. Additionally, Newton's method is not guaranteed to converge. Therefore, you should always include an iteration cap.

Sign-off activity

Write a pseudocode for the Newton-Raphson method based on the example and the stopping criteria outlined in the previous slides.

When you are finished, post your draft in the discussion board **Sign-off activity lecture 11**. We will go over an example at the beginning of next class. You may sign off once you've finished.