Lecture 5

Solving Nonlinear Equations (root-finding): Bracketing, Bisection & Newton's Method

Owen L. Lewis

Department of Mathematics and Statistics
University of New Mexico

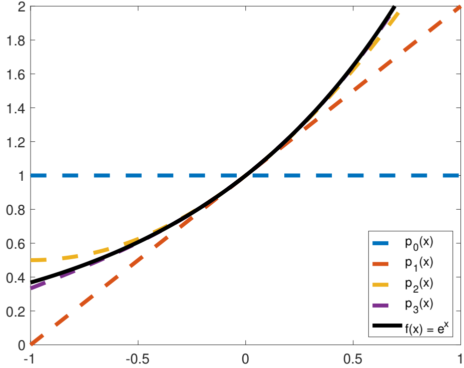
Sept. 3, 2024

Outline

- Finish Taylor Series
- **NOT** Differentiation (incorrect information in lecture 4).
- Begin Root Finding
  - Bracketing
  - Bisection
  - Newton's Method (Maybe)

Taylor Idea

Better & better approximation



Notes

To the board

Taylor Example

Example (e^x)

For the function $f(x) = e^x$, we know $f^{(k)}(x) = e^x$ for all k.
If we let $c = 0$, then $e^0 = 1$ and

$$T(x) = f(c) + (x-c)f'(c) + \frac{(x-c)^2}{2!}f''(c) + \cdots + \frac{(x-c)^n}{n!}f^{(n)}(c) + \ldots$$

becomes

$$\begin{aligned} T(x) &= 1 + (x-0)\cdot 1 + \frac{(x-0)^2}{2}\cdot 1 + \ldots \\ &= 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots \end{aligned}$$

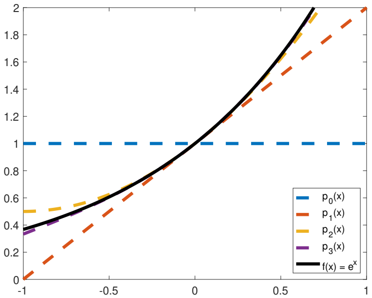
Taylor is Great!

Question: Is $T(x)$ the same as $f(x)$?

Yes!

For “nice” functions $f(x)$

$$f(x) = T(x) = f(c) + (x-c)f'(c) + \frac{(x-c)^2}{2!}f''(c) + \cdots + \frac{(x-c)^n}{n!}f^{(n)}(c) + \ldots$$



Taylor Approximation

- So

$$e^2 = 1 + 2 + \frac{2^2}{2!} + \frac{2^3}{3!} + \dots$$

- But we can't evaluate an infinite series, so we truncate...

Taylor Series Polynomial Approximation

The Taylor Polynomial of degree n for the function $f(x)$ about the point c is

$$p_n(x) = \sum_{k=0}^{n} \frac{(x-c)^k}{k!} f^{(k)}(c)$$

Example (e^x)

In the case of the exponential

$$e^x \approx p_n(x) = 1 + x + \frac{x^2}{2!} + \cdots + \frac{x^n}{n!}$$

Taylor Approximation

"Evaluate" e^2:

- Using 0^{th} order Taylor series: $e^x \approx 1$ does not give a good fit.
- Using 1^{st} order Taylor series: $e^x \approx 1 + x$ gives a better fit.
- Using 2^{nd} order Taylor series: $e^x \approx 1 + x + x^2/2$ gives a a really good fit.

```
x=2;
pn=0;
for j=0:15
  pn = pn + (x^j)/factorial(j);
  err = exp(2)-pn
  pause
end
```

Taylor Approximation Recap

Infinite Taylor Series Expansion (exact)

$$f(x) = f(c) + (x-c)f'(c) + \frac{(x-c)^2}{2!}f''(c) + \cdots + \frac{(x-c)^n}{n!}f^{(n)}(c) + \ldots$$

Finite Taylor Series Approximation

$$f(x) \approx f(c) + (x-c)f'(c) + \frac{(x-c)^2}{2!}f''(c) + \cdots + \frac{(x-c)^n}{n!}f^{(n)}(x),$$

Finite Taylor Series Expansion (exact)

$$f(x) = f(c) + (x-c)f'(c) + \cdots + \frac{(x-c)^n}{n!}f^{(n)}(x) + \frac{(x-c)^{n+1}}{(n+1)!}f^{(n+1)}(\xi),$$

but we don't know ξ (it has to be somewhere between x and c) and it could be different for every x.

Taylor Approximation Error

- How accurate is the Taylor series polynomial approximation?
- The $n+1$ terms of the approximation are simply the first $n+1$ terms of the *exact* expansion:

$$e^x = \underbrace{1 + x + \frac{x^2}{2!}}_{p_2 \text{ approximation to } e^x} + \underbrace{\frac{x^3}{3!} + \dots}_{\text{truncation error}} \tag{1}$$

- So the function $f(x)$ can be written as the Taylor Series approximation plus an error (truncation) term:

$$f(x) = p_n(x) + E_n(x)$$

where

$$E_n(x) = \sum_{k=n+1}^{\infty} \frac{(x-c)^k}{k!} f^{(k)}(c) \quad \text{or} \quad \frac{(x-c)^{n+1}}{(n+1)!} f^{(n+1)}(\xi)$$

Note: Taylor's theorem guarantees some ξ exists, but doesn't tell us what it is.

Truncation Error

Example ($sin(x)$)

The Taylor series expansion of $\sin(x)$ (at the point $c = 0$) is

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \dots$$

If $x \ll 1$, then the remaining terms are small. If we neglect these terms

$$\sin(x) = \underbrace{x - \frac{x^3}{3!} + \frac{x^5}{5!}}_{\text{approximation to sin}} \underbrace{- \frac{x^7}{7!} + \frac{x^9}{9!} - \dots}_{\text{truncation error}}$$

Taylor Series: Computations

- How do we evaluate $f(x) = \frac{1}{1-x}$ computationally?
- Taylor Series Expansion:

$$f(x) = f(c) + (x-c)f'(c) + \frac{(x-c)^2}{2!}f''(c) + \cdots + \frac{(x-c)^n}{n!}f^{(n)}(\xi),$$

- Thus with c = 0

$$\frac{1}{1-x} = 1 + x + x^2 + x^3 + \ldots$$

- Second order approximation:

$$\frac{1}{1-x} \approx 1 + x + x^2$$

Taylor Errors

- How many terms do I need to make sure my error is less than 2×10^{-8} for $x = 1/2$?

$$\frac{1}{1-x} = 1 + x + x^2 + \cdots + x^n + \sum_{k=n+1}^{\infty} x^k$$

- so the error at $x = 1/2$ is

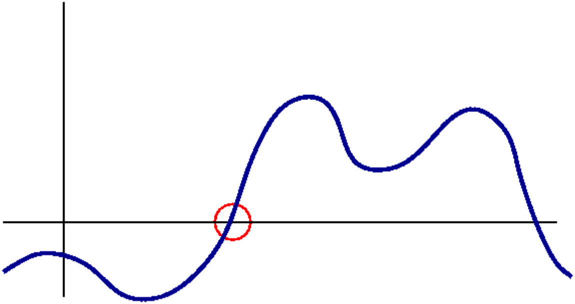
$$e_{x=1/2} = \sum_{k=n+1}^{\infty} \left(\frac{1}{2}\right)^k = \frac{(1/2)^{n+1}}{1-1/2}$$
$$= 2 \cdot (1/2)^{n+1} < 2 \times 10^{-8}$$

- then we need

$$n + 1 > \frac{-8}{\log_{10}(1/2)} \approx 26.6 \quad \text{or}$$
$$n > 26$$

Root Finding

Given a function $f(x)$, find x so that $f(x) = 0$



The point x is the “root” or the “zero” of f.

Root Finding: Solving nonlinear equations

Goals:

- Find roots to equations
- Compare usability of different methods
- Compare convergence properties of different methods

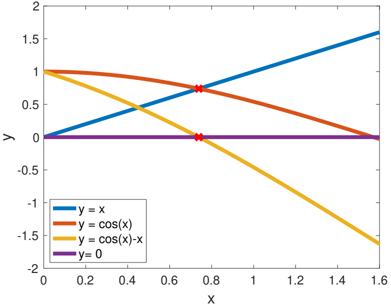
1. Bracketing methods
2. Bisection Method
3. Newton's Method
4. Secant Method

Roots of $f(x)$

- Any single valued equation can be written as $f(x) = 0$
- That is, root finding is a way to solve nonlinear equations

Example

- Find x so that $\cos(x) = x$
- That is, find where $f(x) = \cos(x) - x = 0$



We can actually go the other way too...... (fixed point iteration).

Which Method? Analyze your Application

- Is the function complicated to evaluate?
  - lots of expresions?
  - singularities?
  - simplify? polynomial?
- How accuracte does our root need to be?
- How fast/robust should our method be?

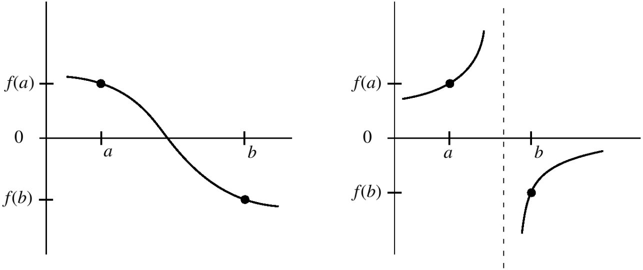
Using these answers, you can pick the right method...

Basic Root Finding Strategy

1. Plot the function
   - Get an initial guess
   - Identify problematic parts
2. Start with the initial guess and iterate

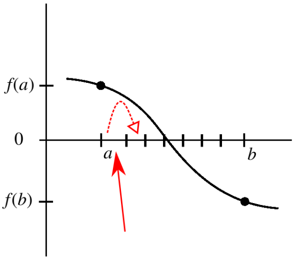
Bracket Basics

- A root x is *bracketed* on $[a, b]$ if $a \leqslant x \leqslant b$.
- If the *bracket* $[a, b]$ is small enough, we expect $f(a)$ and $f(b)$ have opposite sign.
- Changing signs does not guarantee bracketed, however: singularity



- Bracketing helps get an initial guess

Bracket Basics



1. Check to see if bracketed on a sub-interval.
2. Proceed to the next interval.

Bracket Algorithm

- Subdivide interval into *n* chunks
- Check for a sign flip on each interval

Listing 1: Bracket Algorithm

```
given: f(x), x_min, x_max, n

dx = (x_max - x_min)/n
x_left = x_min
i = 0

while i < n
  i = i + 1
  x_right = x_left + dx
  if f(x) changes sign in [x_left, x_right]
    save [x_left, x_right] as an interval with a root
    return
  else
    x_left = x_right
  end
```

Testing Sign

$f(a) \times f(b) < 0$

Should we use?

```
fa = myfunc(a);
fb = myfunc(b);

if(fa*fb < 0)
  (save bracket)
end
```

Better Sign Test

!

Nope. Underflow could lead to a spurious zero result

`sign()`

Use matlab's sign

```
fa = myfunc(a);
fb = myfunc(b);

if(sign(fa) != sign(fb))
  (save bracket)
end
```

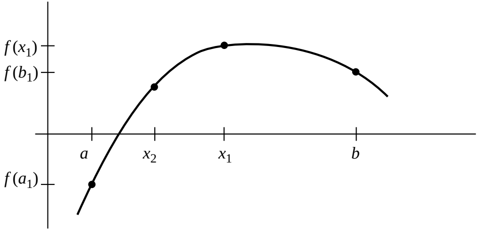
Moving forward...

Bracketing is fine. But we need to find the actual root:

- Bisection
- Newton's Method
- Secant Method
- Fixed Point Iteration

Bisection

Given a bracketed root, halve the interval while continuing to bracket the root



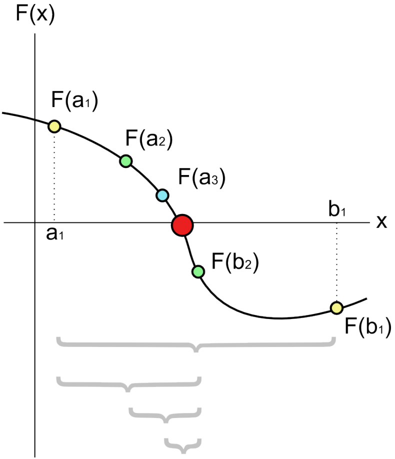
Bisection (2)

For the bracket interval $[a, b]$ the midpoint is

$$x_m = \frac{1}{2}(a + b)$$

Idea:

1. split bracket in half
2. select the bracket that has the root
3. goto step 1



Bisection Algorithm

Listing 2: Bisection

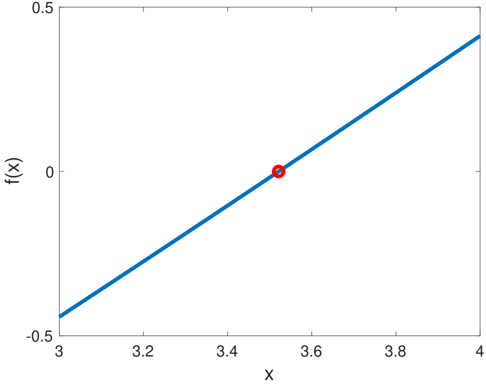
```
initialize:  $a = $ user_param, $b = $ user_param
assume: $\mathrm{sign}(f(a)) \neq \mathrm{sign}(f(b))$

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

Bisection Example

Lets find the root of

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

on the interval $[3, 4]$



Bisection Example

Solve with bisection

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

Initial bracket: $f(a) = f(3) \approx -0.4422$ and $f(b) = f(4) \approx 0.4126$

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

Notes

Analysis of Bisection

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

$$
\begin{aligned}
\delta_0 &= b - a = \text{initial bracketing interval} \\
\delta_1 &= \frac{1}{2}\delta_0 \\
\delta_2 &= \frac{1}{2}\delta_1 = \frac{1}{4}\delta_0 \\
&\vdots \\
\delta_n &= \left(\frac{1}{2}\right)^n \delta_0
\end{aligned}
$$

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

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

Analysis of Bisection

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

The ratio $\frac{\delta_n}{\delta_0}$ measures a relative reduction of your error (assuming you guess that the root is somewhere inside the bracketed interval)

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

Bisection: Error in Root

- If we pick the midpoint $c_n = (a_n + b_n)/2$ as the root then,

$$|x_* - c_n| \leqslant (b_n - a_n)/2 = \delta_n/2$$

  where x_* is the true root.
- Recall that

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

- The error in the root after n steps is

$$\begin{aligned} |x_* - c_n| \leqslant (b_n - a_n)/2 &= \left(\frac{1}{2}\right)^{n+1} \delta_0 \\ &= \left(\frac{1}{2}\right)^{n+1} (b - a) \end{aligned}$$

Bisection: Example

Question: How many steps of bisection are needed in order to compute the root of f so that the error is less than 10^{-8} if $a = -2$ and $b = 3$?

Solution: Find n such that,

$$\left(\frac{1}{2}\right)^{n+1}(b-a) \leqslant 10^{-8}$$

$$\implies \left(\frac{1}{2}\right)^{n+1}(5) \leqslant \times 10^{-8} \qquad \text{multiply by } 2/5\text{, and take natural log}$$

$$\implies \log 2^{-n} \leqslant \log((2/5) \times 10^{-8}) \qquad \text{divide through by} (-\log 2)$$

$$\implies n \geqslant \frac{1}{\log 2}(-\log(2/5) + 8 \log 10) \qquad \text{just evaluate expressions}$$

$$\implies n \geqslant 27.89$$

So, want $n \geqslant 28$ steps.

Is this “good enough”?

This will get us an approx x_n and it is within 10^{-8} of the true answer x^*.

Is it good enough?

Is $f(x_n)$ close to zero?

Convergence Criteria

An automatic root-finding procedure needs to monitor progress towards the root and then stop when the current guess is close enough to the desired root.

This will avoid unneccessary work. Two convergence checks are:

- Check the closeness of successive approximations, for tolerance δ_x

$$|x_n - x_{n-1}| < \delta_x$$

- Check how close $f(x)$ is to zero at the current guess, for tolerance δ_f

$$|f(x_n)| < \delta_f$$

- Which one you use depends on the problem being solved

Different Criteria

- How close we are to the "true" answer:
  Tolerance on input, forward error (book), "error".

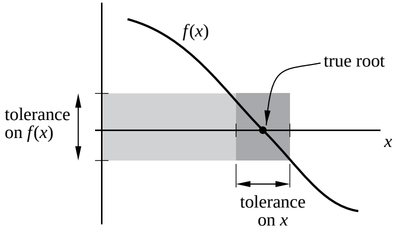
$$|x_n - x^*|$$

- How close we are to solving the problem:
  Tolerance on output, backward error (book), "residual"

$$|f(x_n) - 0|$$

- They are related, but *NOT* the same.

Convergence Criteria on x

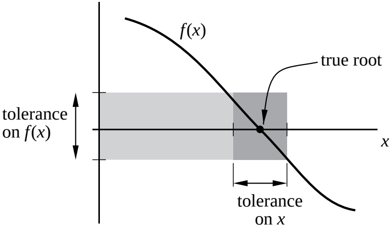


$$x_n = \text{current guess at the root (midpoint of current bracket)}$$
$$x_{n-1} = \text{previous guess at the root (midpoint of previous bracket)}$$

Absolute tolerance: $\left|x_n - x_{n-1}\right| < \delta_x$

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

Convergence Criteria on $f(x)$



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

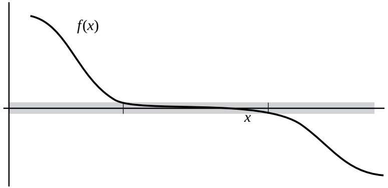
Relative tolerance:

$$|f(x_n)| < \frac{\hat{\delta}_f}{\max\left\{ |f(a_0)|, |f(b_0)| \right\}}$$

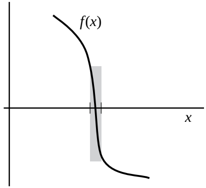
where a_0 and b_0 are the original brackets

Convergence Criteria Compared

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 (safer)



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 (safer)



Relationship Between Criteria

- How are the criteria on x and $f(x)$ related? Consider the ratio of the two criteria
$$\frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}$$
- The limit of this as x_{n-1} and x_n converge to the exact answer x^* is just $f'(x^*)$.
- We can thus expect (this is not yet a proof) that

$$|f(x_n) - f(x_{n-1})| \approx |f'(x^*)||x_n - x_{n-1}|$$

as x_{n-1} and x_n approach the solution x^*.