Root Finding via Bisection

A Quick Review

Bisection Method: Overview

- Bisection method: one of 3 main root finding algorithms you'll see
- Premise: the root lies somewhere in a user-specified initial interval. The interval is narrowed down until it "hugs" the root.
- Called a *bracketing method* because the user must supply the algorithm with a bracket which contains the root
  - Contrast: Newton-Raphson method is an *open method* because it only requires an initial estimate (not a bracket) of the root
- Assumption: $f(x)$ is real and continuous

Aside: $f(x) = 0$ form

- Whenever we mention a function $f(x)$ in root finding, we imply a function in "$f(x) = 0$" form
  - Roots vs. intercepts – there's a difference! (06a video)

- Example: If we want to compute the roots of $x^3 = 27$:

$$f(x) = x^3 - 27$$

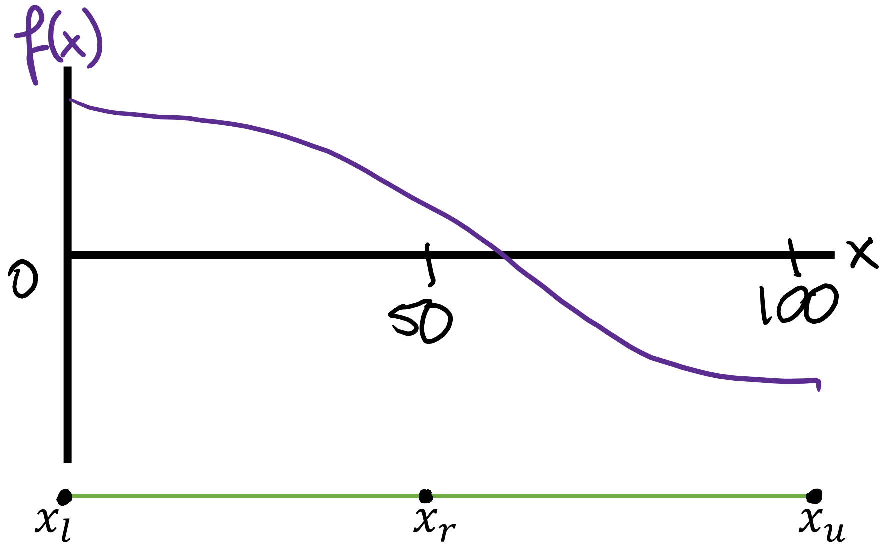
$$\cancel{f(x) = x^3}$$

Brief Algorithm Overview

"such that"

- Specify an initial interval: $[x_l \;\; x_u] \ni sign(f(x_l)) \neq sign(f(x_u))$
- Assume the root x_r lies at the midpoint of the interval
- Divide the initial interval into two (equal) subintervals: $[x_l \;\; x_r]$ and $[x_r \;\; x_u]$
- Evaluate $f(x_l)$ and $f(x_r)$ and determine which of the two subintervals contains a sign change
- Assign new x_l, x_u accordingly and iterate

1st iteration:



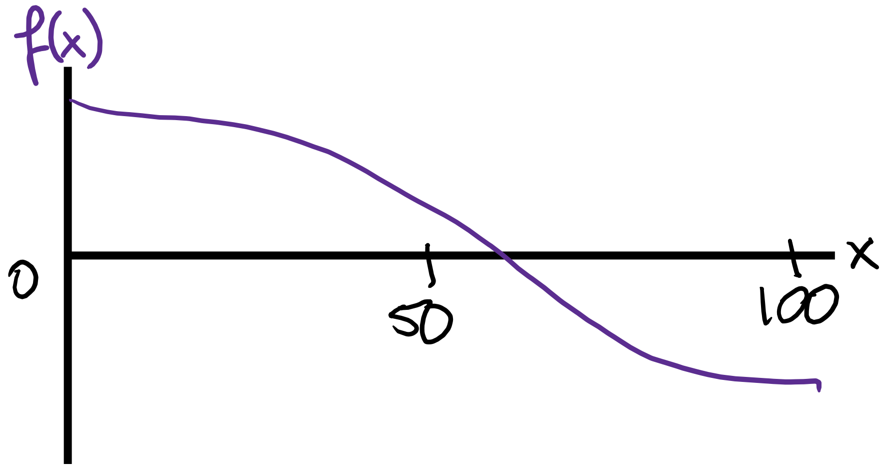
$$x_l = 0, x_u = 100$$
$$x_r = \frac{100 + 0}{2} = 50$$

Subintervals: [0 50] and [50 100]

$f(x_l) > 0, f(x_r) > 0$→ no sign change in [0 50]

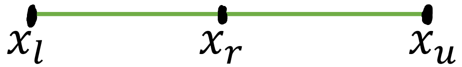
→ Discard [0 50] (no sign change) and keep [50 100] (sign change).

New $x_l = 50$, new $x_u = 100$.



1st iteration:

2nd iteration:



$$x_l = 50, x_u = 100$$

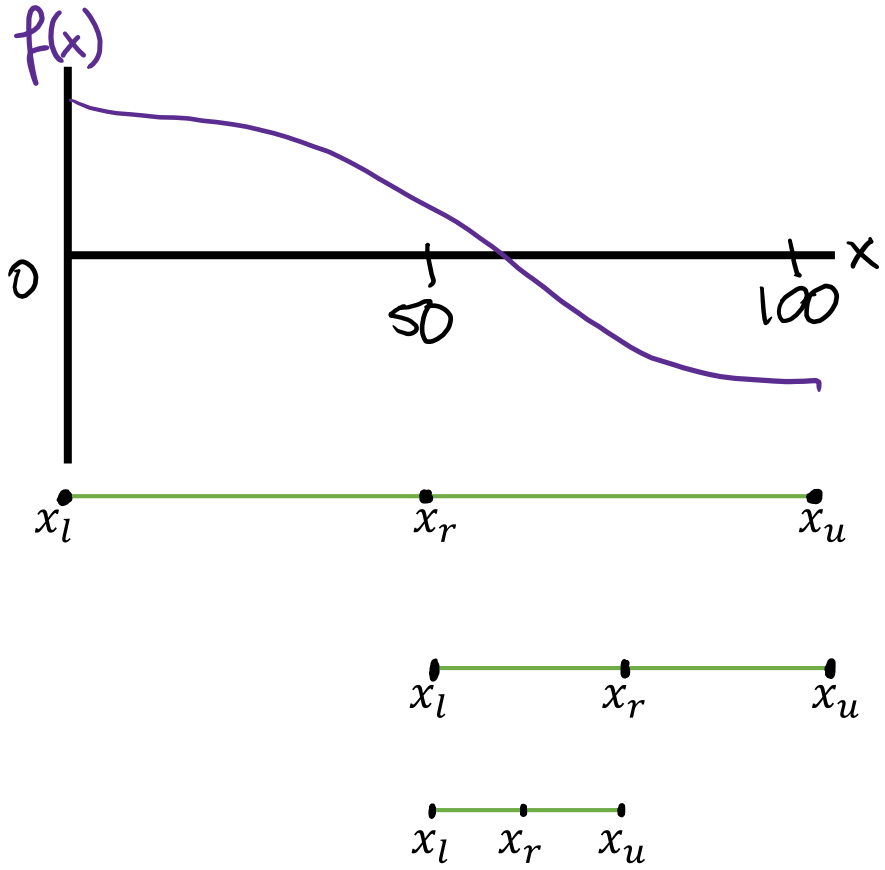
$$x_r = \frac{100 + 50}{2} = 75$$

Subintervals: [50 75] and [75 100]

$f(x_l) > 0, f(x_r) < 0$ → sign change in [50 75]

→ Discard [75 100] (no sign change) and keep [50 75] (sign change).

New $x_l = 50$, new $x_u = 75$.



1st iteration:

2nd iteration:

3rd iteration:

$$x_l = 50, x_u = 75$$

$$x_r = \frac{50 + 75}{2} = 62.5$$

Subintervals: [50 62.5] and [62.5 75]

$f(x_l) > 0, f(x_r) < 0$ → sign change in [50 62.5]

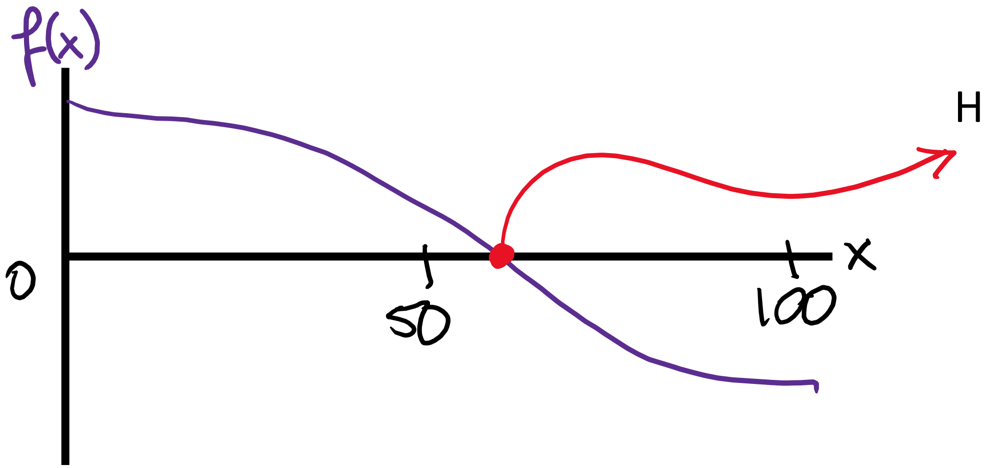
→ Discard [62.5 75] (no sign change) and keep [50 62.5] (sign change).

New $x_l = 50$, new $x_u = 62.5$.

Stopping Criterion

- *Stopping criterion*: condition which terminates the search if met
- Common (but not universal!) stopping criterion: $e_s = 0.0001$

$$|f(x_r)| \leq e_s$$



Here, x_r is close enough such that $|f(x_r)| \leq 0.0001$.

Bisection Pros/Cons

Pros

- Guaranteed to converge
- Logical, easy-to-follow algorithm

Cons

- Slow
- Computationally expensive

Food For Thought 

- From Slide 4: In the initial interval,

$$[x_l \;\; x_u] \ni sign\big(f(x_l)\big) \neq sign\big(f(x_u)\big)$$

Why?