Final Project: Modified `zero-in` for root-finding

We would like to find a root of the equation

$$f(x) = 0, \quad \text{for} \quad x \in \mathbf{R}$$

given an initial interval $[a, \ b]$ with

$$f(a) \cdot f(b) < 0.$$

with a combination of two methods

- **bisection** method, for its reliability
- **inverse quadratic interpolation** (IQI) method, for its higher order of convergence.

inverse quadratic interpolation (IQI) method

Given three pairs of points $(x_0,\ f_0)$, $(x_1,\ f_1)$, $(x_2,\ f_2)$, IQI defines a quadratic polynomial in f that goes through these points,

$$x(f) = \frac{(f - f_1)(f - f_2)}{(f_0 - f_1)(f_0 - f_2)} x_0 + \frac{(f - f_0)(f - f_2)}{(f_1 - f_0)(f_1 - f_2)} x_1 + \frac{(f - f_0)(f - f_1)}{(f_2 - f_0)(f_2 - f_1)} x_2$$

This leads to an estimate for the root $x_3 \stackrel{def}{=} x(0)$:

$$x_3 = \frac{f_1 f_2}{(f_0 - f_1)(f_0 - f_2)} x_0 + \frac{f_0 f_2}{(f_1 - f_0)(f_1 - f_2)} x_1 + \frac{f_0 f_1}{(f_2 - f_0)(f_2 - f_1)} x_2$$

Modified `zero-in` for root-finding in a sketch

For given initial interval $[a,\ b]$ with

$$f(a) \cdot f(b) < 0.$$

We would like to find a root of the equation $\quad f(x) = 0, \quad$ for $\quad x \in \mathbf{R}$

Modified `zero-in` for root-finding in a sketch

For given initial interval $[a,\ b]$ with

$$f(a) \cdot f(b) < 0.$$

We would like to find a root of the equation $\quad f(x) = 0, \quad$ for $\quad x \in \mathbf{R}$

1. **set** $x_0,\ x_1,\ x_2 = a,\ b,\ c \stackrel{def}{=} \frac{a+b}{2}$
2. **let** $x_3 = \mathtt{IQI}(x_0,\ x_1,\ x_2)$
   - **if** $x_3 \notin [a,\ b]$
     - **do** `bisection` steps on $[a,\ b]$
     - **set** new interval $[a^{\mathbf{new}},\ b^{\mathbf{new}}] \subset [a, b]$ with

       $$f(a^{\mathbf{new}}) \cdot f(b^{\mathbf{new}}) < 0, \qquad \textbf{repeat} \text{ step (1)}$$
   - **else if** $|f(x_3)|$ has not DECREASED by a factor of 2 within 4 consecutive **IQI** iterations,
     - **do** `bisection` steps on $[a,\ b]$
     - **set** new interval $[a^{\mathbf{new}},\ b^{\mathbf{new}}] \subset [a, b]$ with

       $$f(a^{\mathbf{new}}) \cdot f(b^{\mathbf{new}}) < 0, \qquad \textbf{repeat} \text{ step (1)}$$
   - **repeat IQI** in step (2)
3. **stop** when iteration converged

Algorithm Sketch

procedure MODIFIEDZEROIN(f, I, params) ▷ Basic algorithmic structure
- **set** $x_0 = \mathtt{I}.a$, $x_1 = \mathtt{I}.b$, $x_2 = \frac{1}{2}(x_0 + x_1)$
- **while** root not yet found **do**
  - **while** root not yet found **do**
    - **set** $x_3 = \mathtt{IQI}(x_0, x_1, x_2)$
    - **if** $x_3 \notin [a, b]$ **then** ▷ IQI failure
      - **break**
    - **end if**
    - **if** $|f(x_3)|$ NOT DECREASED by a factor of 2 in 4 iterations **then**
      - **break**
    - **end if**
    - **repeat IQI** using safety bracket
  - **end while**
  - **perform bisection**
- **end while**
- **return** root/ **report** failure

end procedure

More details!

1. *function calls*: when the function $f(\cdot)$ is called.
   1.1 There are THREE function calls in IQI formula

$$
\begin{aligned}
f_0 &= f(x_0), \quad f_1 = f(x_1), \quad f_2 = f(x_2) \\
x_3 &= \frac{f_1\, f_2}{(f_0 - f_1)\,(f_0 - f_2)}\, x_0 + \frac{f_0\, f_2}{(f_1 - f_0)\,(f_1 - f_2)}\, x_1 + \frac{f_0\, f_1}{(f_2 - f_0)\,(f_2 - f_1)}\, x_2
\end{aligned}
$$

   1.2 There are EIGHTEEN function calls in IQI formula

$$
x_3 = \frac{f(x_1)\, f(x_2)}{(f(x_0) - f(x_1))\,(f(x_0) - f(x_2))}\, x_0 + \frac{f(x_0)\, f(x_2)}{(f(x_1) - f(x_0))\,(f(x_1) - f(x_2))}\, x_1 + \frac{f(x_0)\, f(x_1)}{(f(x_2) - f(x_0))\,(f(x_2) - f(x_1))}\, x_2
$$

Faster with 1.1, needs further improvement for full extra credit

More details!

1. *function calls*: when the function $f(\cdot)$ is called.
   1.1 There are THREE function calls in IQI formula

$$\begin{aligned} f_0 &= f(x_0), \quad f_1 = f(x_1), \quad f_2 = f(x_2) \\ x_3 &= \frac{f_1\, f_2}{(f_0 - f_1)\,(f_0 - f_2)}\, x_0 + \frac{f_0\, f_2}{(f_1 - f_0)\,(f_1 - f_2)}\, x_1 + \frac{f_0\, f_1}{(f_2 - f_0)\,(f_2 - f_1)}\, x_2 \end{aligned}$$

   1.2 There are EIGHTEEN function calls in IQI formula

$$x_3 = \frac{f(x_1)\, f(x_2)}{(f(x_0) - f(x_1))\,(f(x_0) - f(x_2))}\, x_0 + \frac{f(x_0)\, f(x_2)}{(f(x_1) - f(x_0))\,(f(x_1) - f(x_2))}\, x_1 + \frac{f(x_0)\, f(x_1)}{(f(x_2) - f(x_0))\,(f(x_2) - f(x_1))}\, x_2$$

$\boxed{\text{Faster with 1.1, needs further improvement for full extra credit}}$

2. **IQI** with safety bracket:
   - **sort** $x_0,\ x_1 = a,\ b;\ \ x_2 \in (a,\ b)$ with $f(a) \cdot f(b) < 0$
   - **let** $x_3 = \mathtt{IQI}(x_0,\ x_1,\ x_2) \in (a,\ b)$
   - if $f(x_0) \cdot f(x_2) < 0$, continue **IQI** with $x_0,\ x_2,\ x_3$:
     $x_0^{\mathbf{new}} = \mathbf{min}(x_0,\ x_2,\ x_3)$, $x_1^{\mathbf{new}} = \mathbf{max}(x_0,\ x_2,\ x_3)$.
   - if $f(x_2) \cdot f(x_1) < 0$, continue **IQI** with $x_1,\ x_2,\ x_3$:
     $x_0^{\mathbf{new}} = \mathbf{min}(x_1,\ x_2,\ x_3)$, $x_1^{\mathbf{new}} = \mathbf{max}(x_1,\ x_2,\ x_3)$.

Programming project submission details (I)

You should turn in a .m file `modifiedzeroinxxx.m` which contains a matlab function of the form

```
function [root,info] = modifiedzeroinxxx(func,Int,params)
```

- xxx is your student id
- `func` is a function handle
- `Int` contains the initial interval [`Int`.a, `Int`.b]
- `params` is an object that contains at least two fields `params.root_tol` and `params.func_tol`.

Your algorithm should terminate once the interval containing the root is at most `params.root_tol` in length, or the function value at the current iterate is at most `params.func_tol` in absolute value.

Programming project submission details (II)

On output, `root` is the computed root, and `info` should have at least one field `info.flag`, which is 0 for a successful execution, and 1 otherwise. Your program will be tested on

test functions on pages 8 and 9, on the given intervals.

1. (100 points) Your root finder finds all roots within 25 function calls.
2. (70 points) Your root finder finds all roots.
3. You get an **extra credit** of 5 points for finding roots within 25 function calls on test functions on page 9.
4. (0 points) Your program does not run.

We will choose `params.root_tol` = `params.func_tol` = 10^{-7}. Your program will receive 0 points if it is found to be highly similar to the matlab built-in function `fzero`. While we allow group discussions, you must do all your programming work by yourself.

Submit your .m file on gradescope by 23:59PM, May 9, 2025

Test functions (easy)

- $f(x) = x * exp(-x) - 2 * x + 1$ on interval $[0, 3]$
- $f(x) = x\,\mathbf{cos}(x) - 2\,x^2 + 3\,x - 1$ on interval $[1, 3]$
- $f(x) = x^3 - 7\,x^2 + 14\,x - 6$ on interval $[0, 1]$
- $f(x) = \sqrt{x} - \mathbf{cos}(x)$ on interval $[0, 1]$
- $f(x) = 2\,x\,\mathbf{cos}(2\,x) - (x + 1)^2$ on interval $[-4, -2]$

Test functions (hard)

- $f(x) = x^3 - 32\,x + 128$ on interval $[-8, 0]$
- $f(x) = x^4 - 2x^3 - 4x^2 + 4x + 4$ on interval $[0, 2]$
- $f(x) = -x^3 - \cos(x)$ on interval $[-3, 3]$
- $f(x) = (x - 5)^7 - 10^{-1}$ on interval $[0, 10]$
- $f(x) = (x - 3)^{11}$ on interval $[2.4, 3.4]$