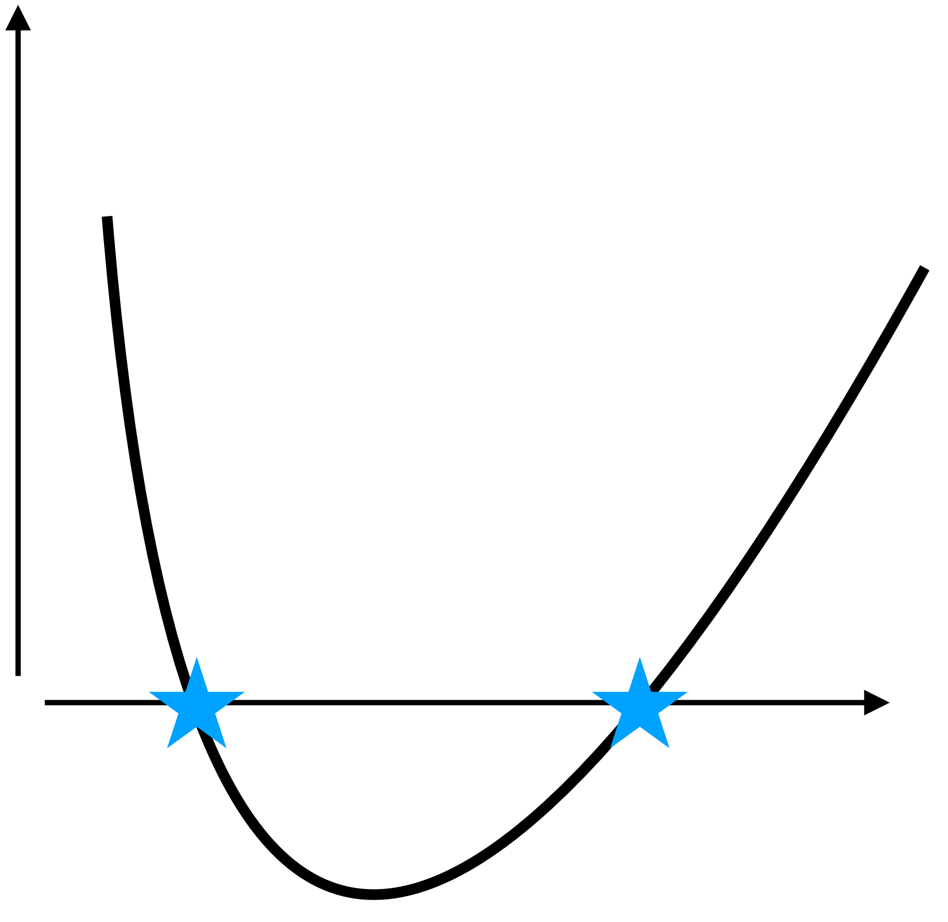
Root finding

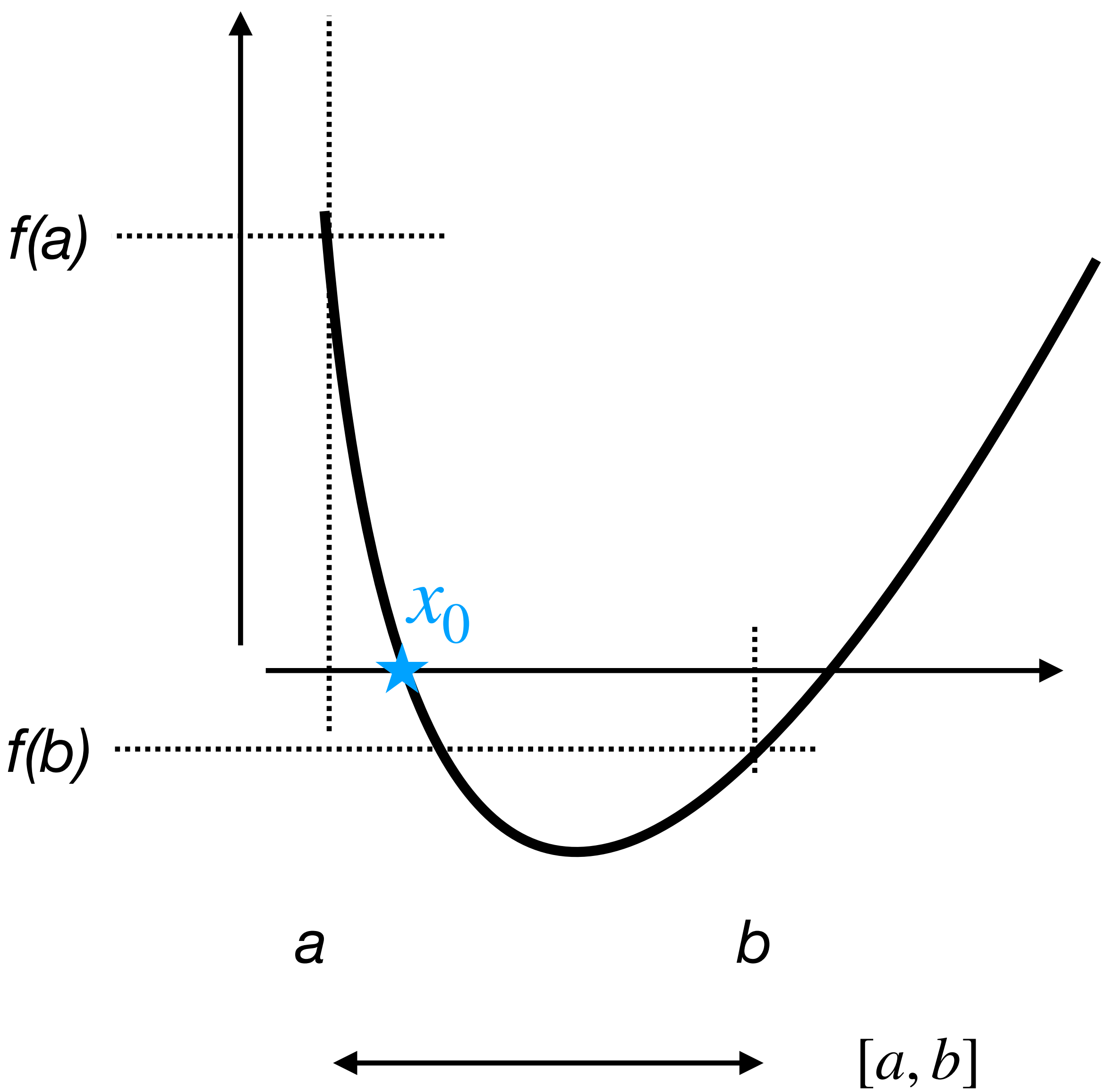


Trial and error

1. guess x_1 (trial)
2. Is $f(x_1) = 0$? (error)
3. improve x_1

until error < eps
(or iterations > max)

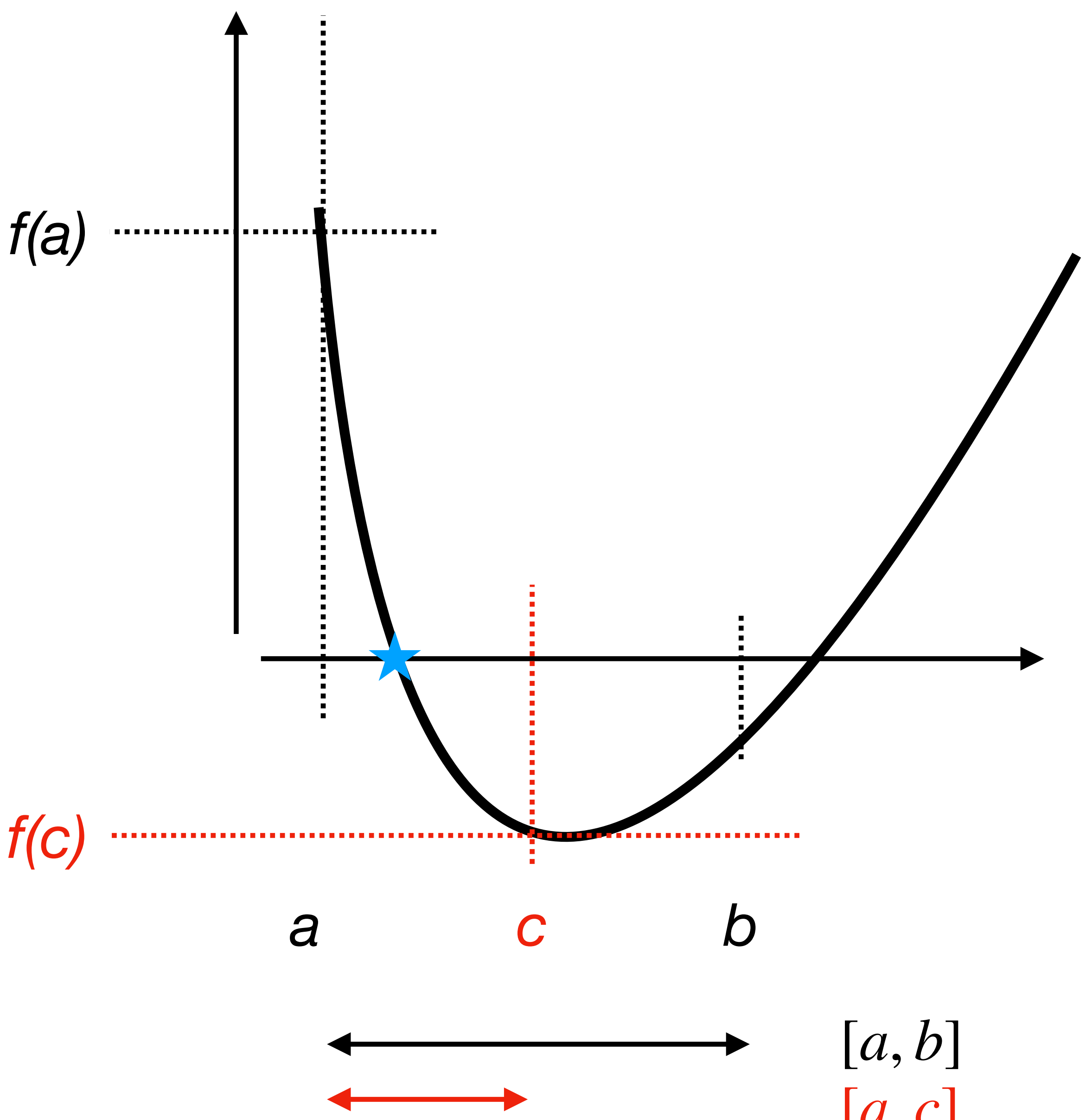
Bisection



$a < x_0 < b$

$f(a) > 0$ **and** $f(b) < 0$

Bisection



$f(a) > 0$ **and** $f(c) < 0$

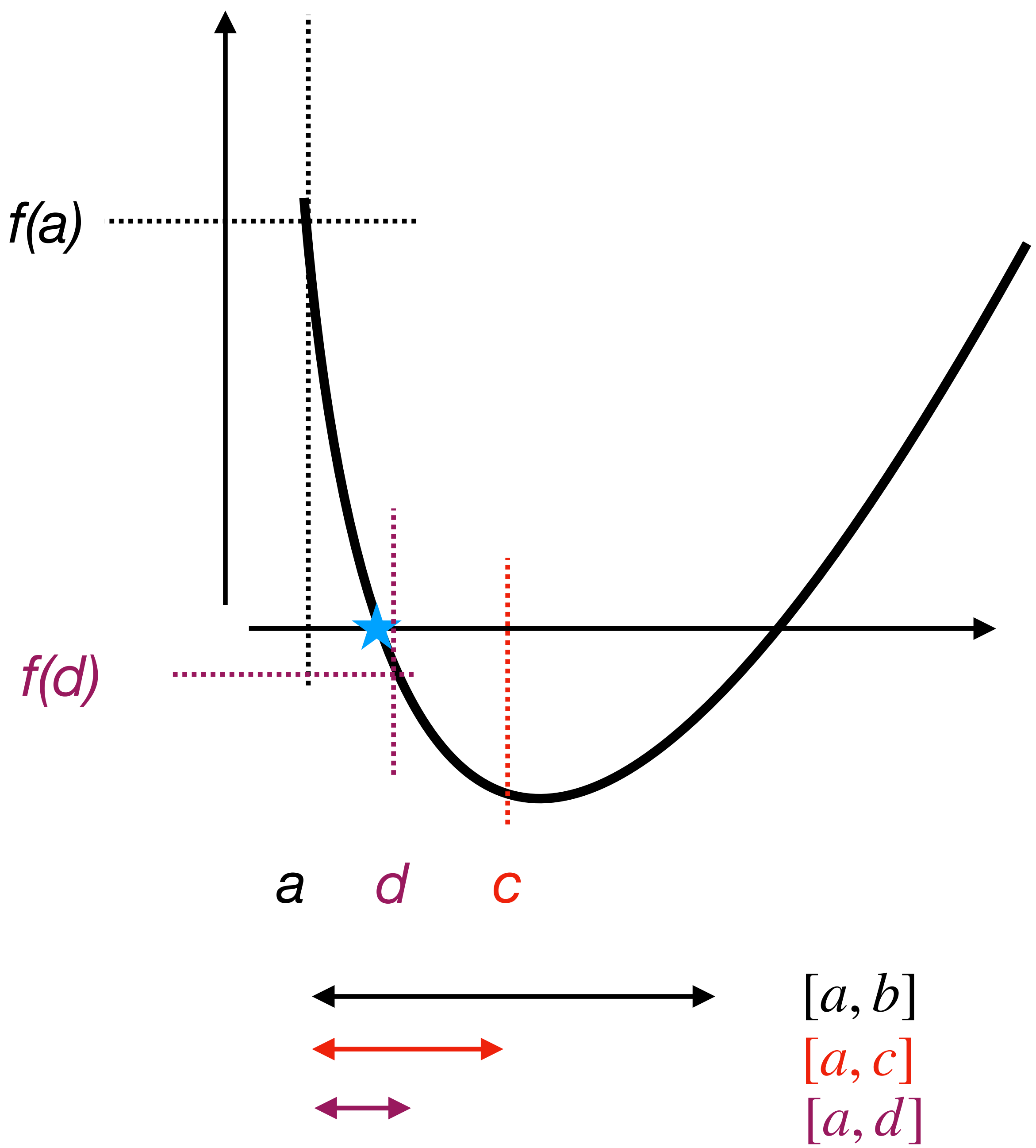
$a < x_0 < c$

Note:

$f(c) < 0$ **and** $f(b) < 0$

so root x_0 is *not* in [c, b]

Bisection



$f(a) > 0$ **and** $f(d) < 0$

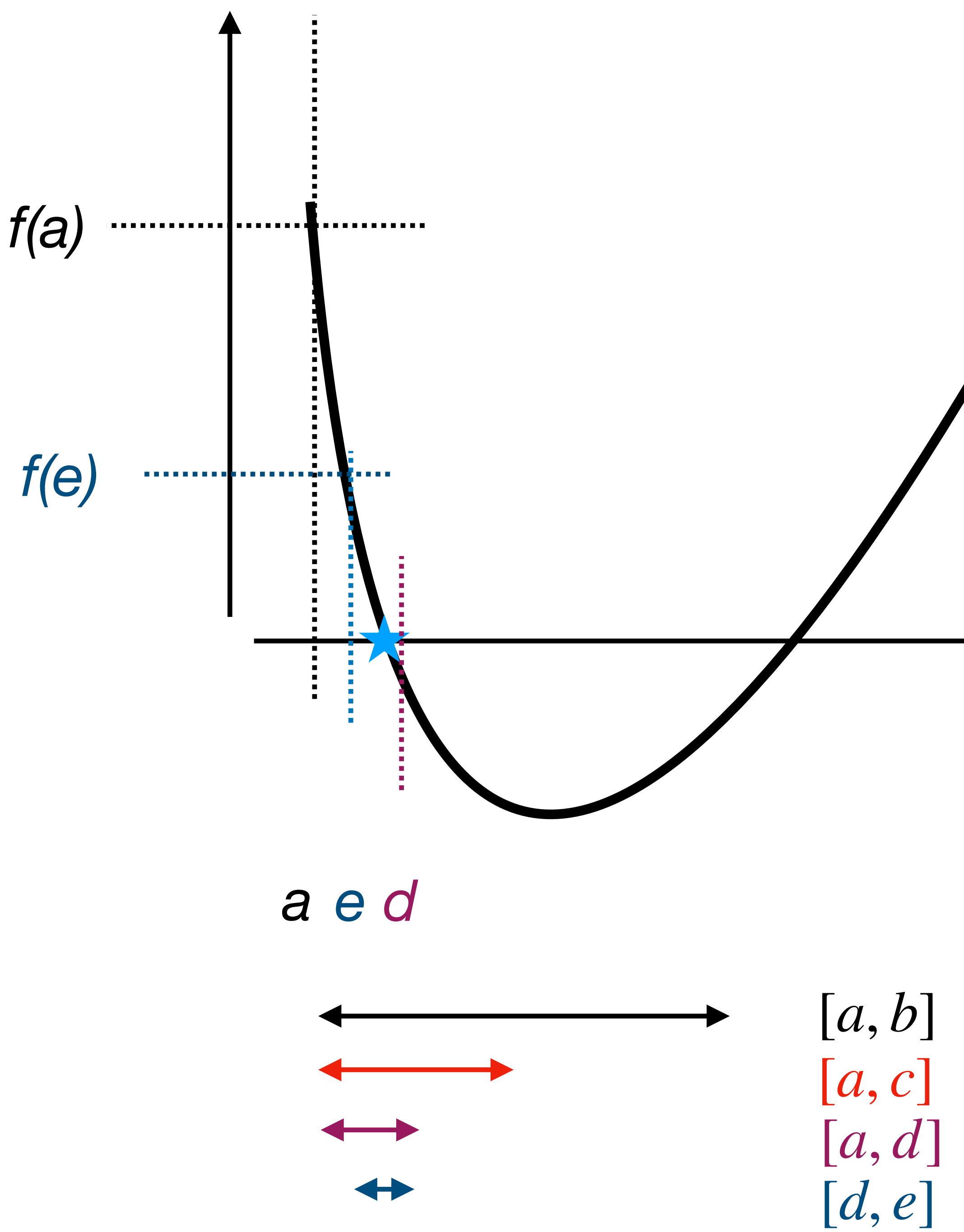
$a < x_0 < d$

Note:

$f(c) < 0$ **and** $f(d) < 0$

so root x_0 is *not* in [d, c]

Bisection



$f(e) > 0$ **and** $f(d) < 0$

$e < x_0 < d$

Note:

$f(a) > 0$ **and** $f(e) > 0$

so root x_0 is *not* in [a, e]

Bisection algorithm

1. bisect

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

2. pick half with sign change

3. |f(x)| < eps ?

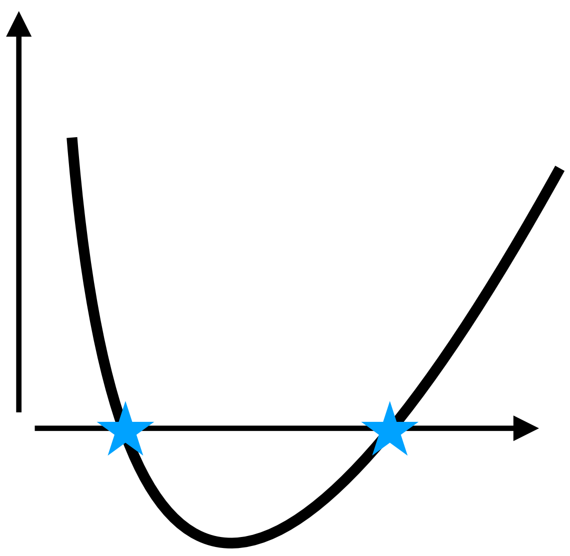
if $f(a)f(x) < 0$

$$x_0 \in [a, x]$$
$$b \leftarrow x$$

else

$$x_0 \in [x, b]$$
$$a \leftarrow x$$

Root finding with trial and error



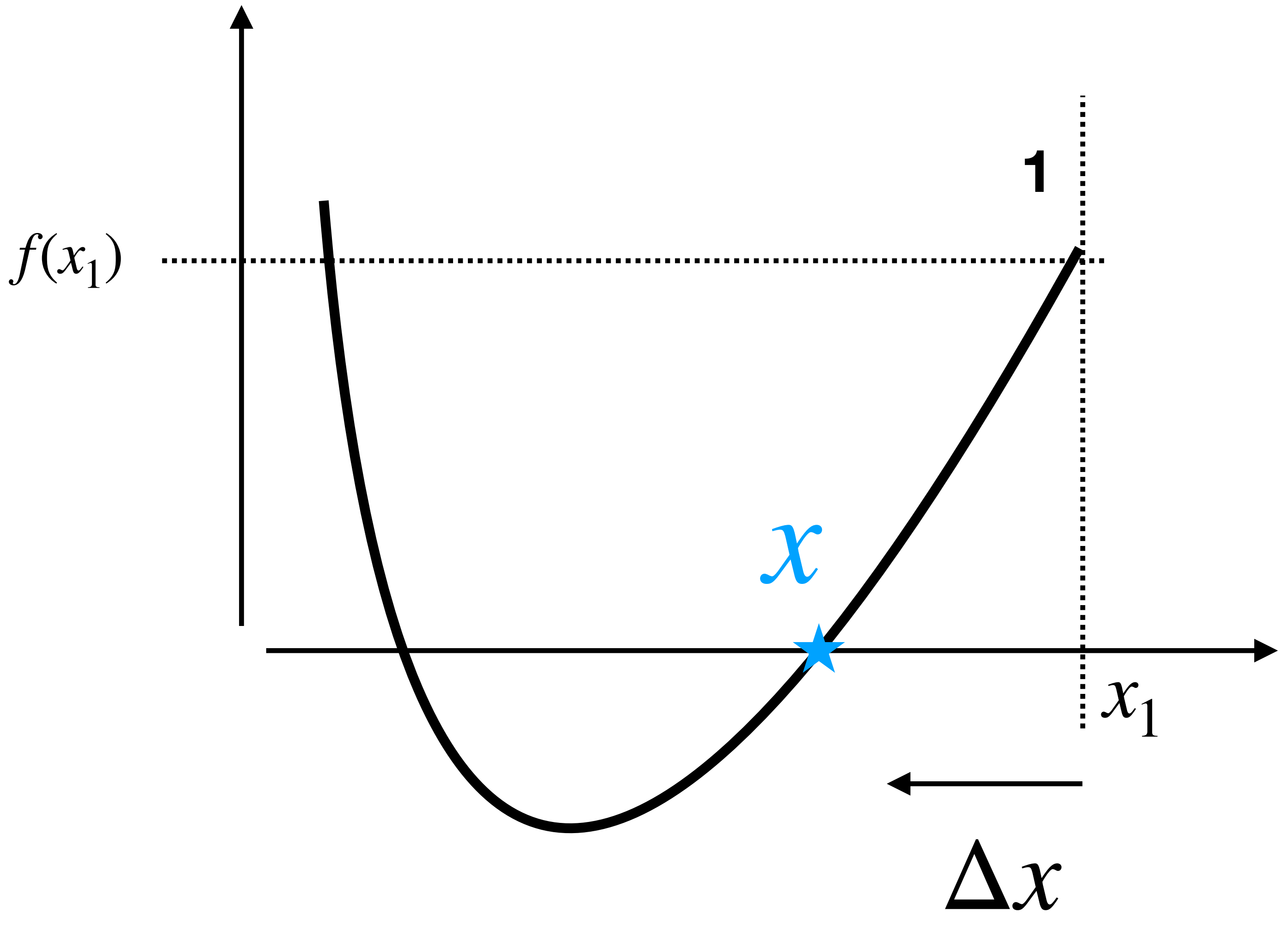
1. guess x_1 (trial)

2. Is $f(x_1) = 0$? (error)

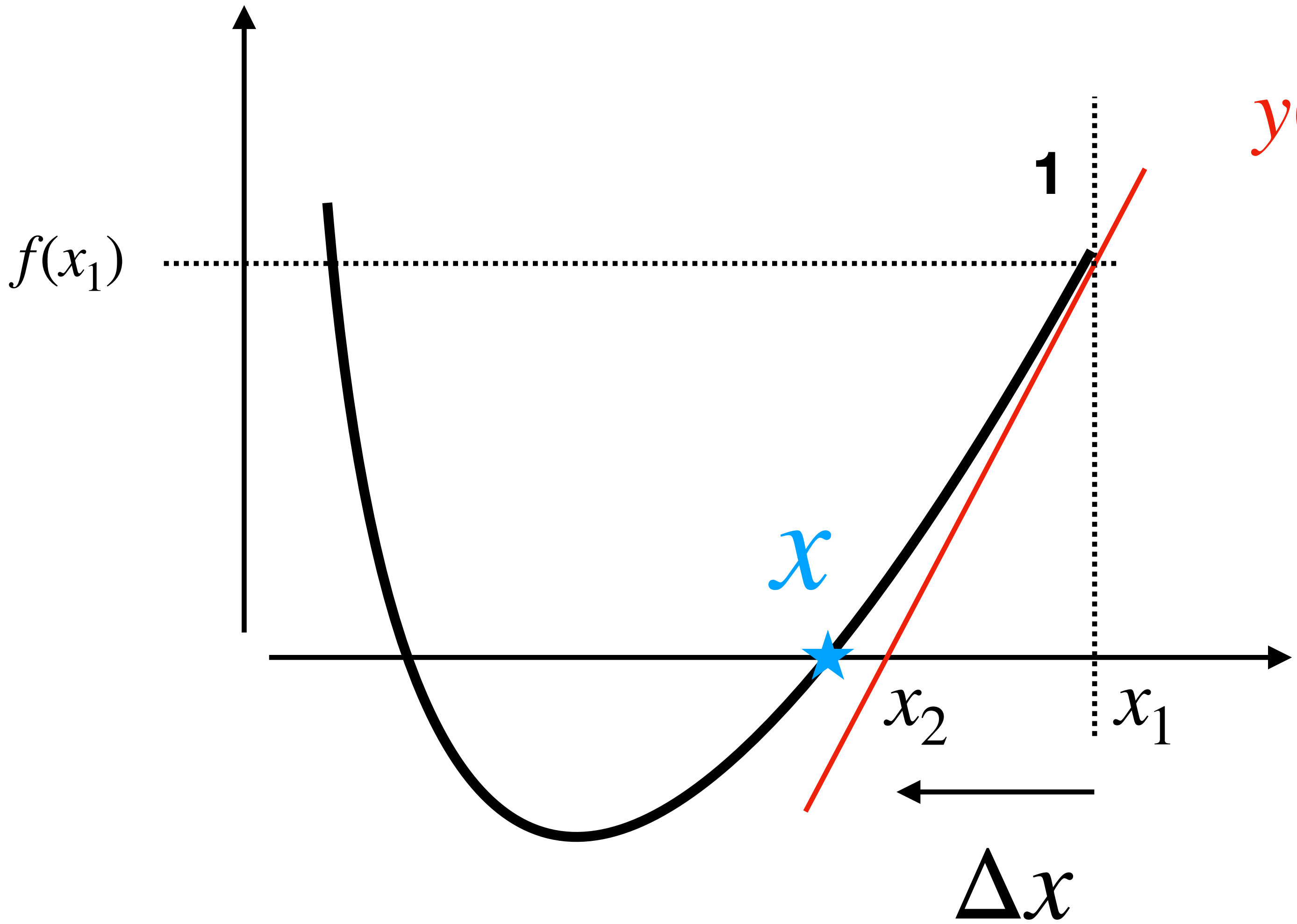
3. improve x_1

until error < eps
(or iterations > max)

Newton-Raphson



Newton-Raphson

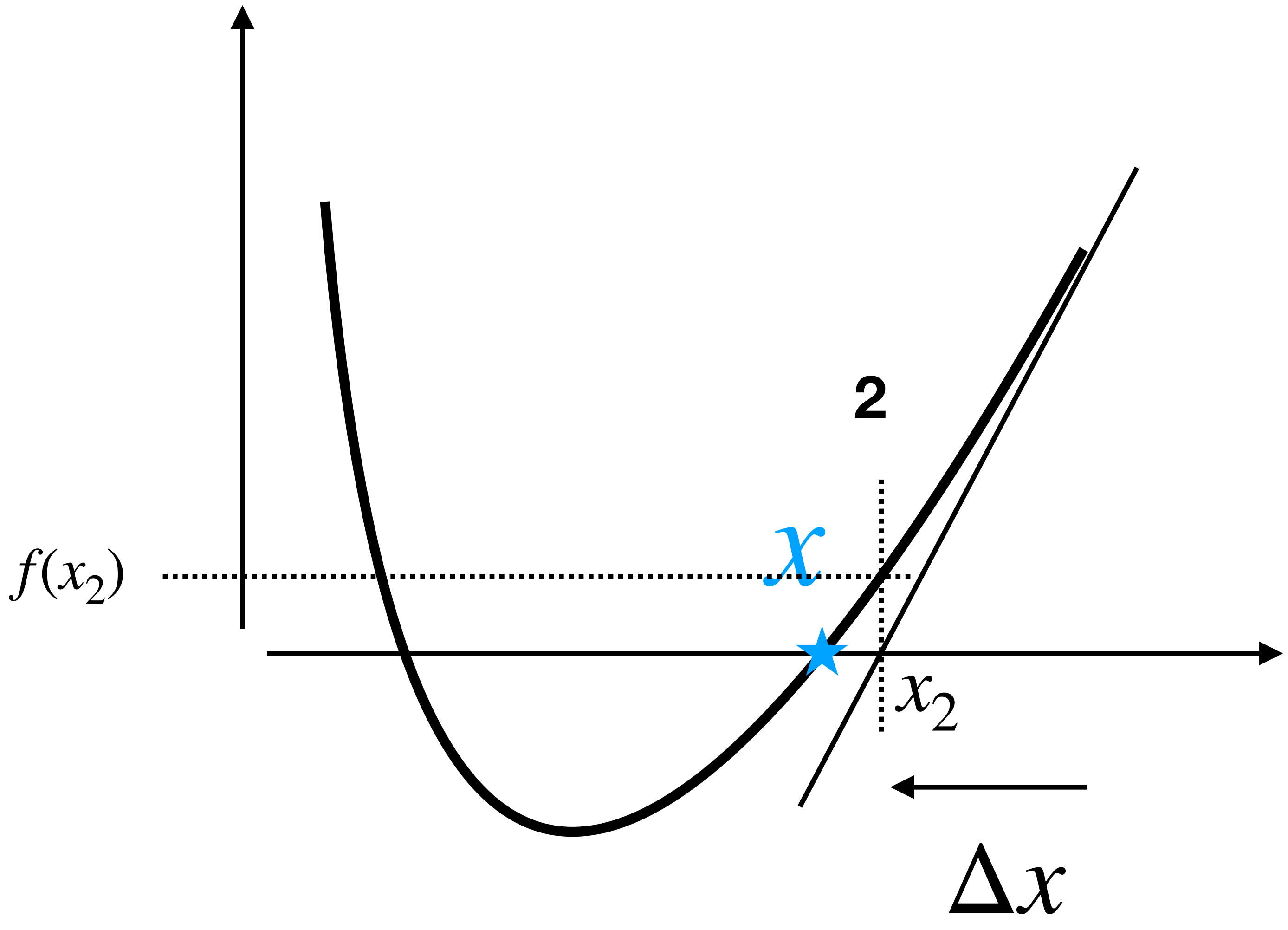


$y(x) = mx + b$

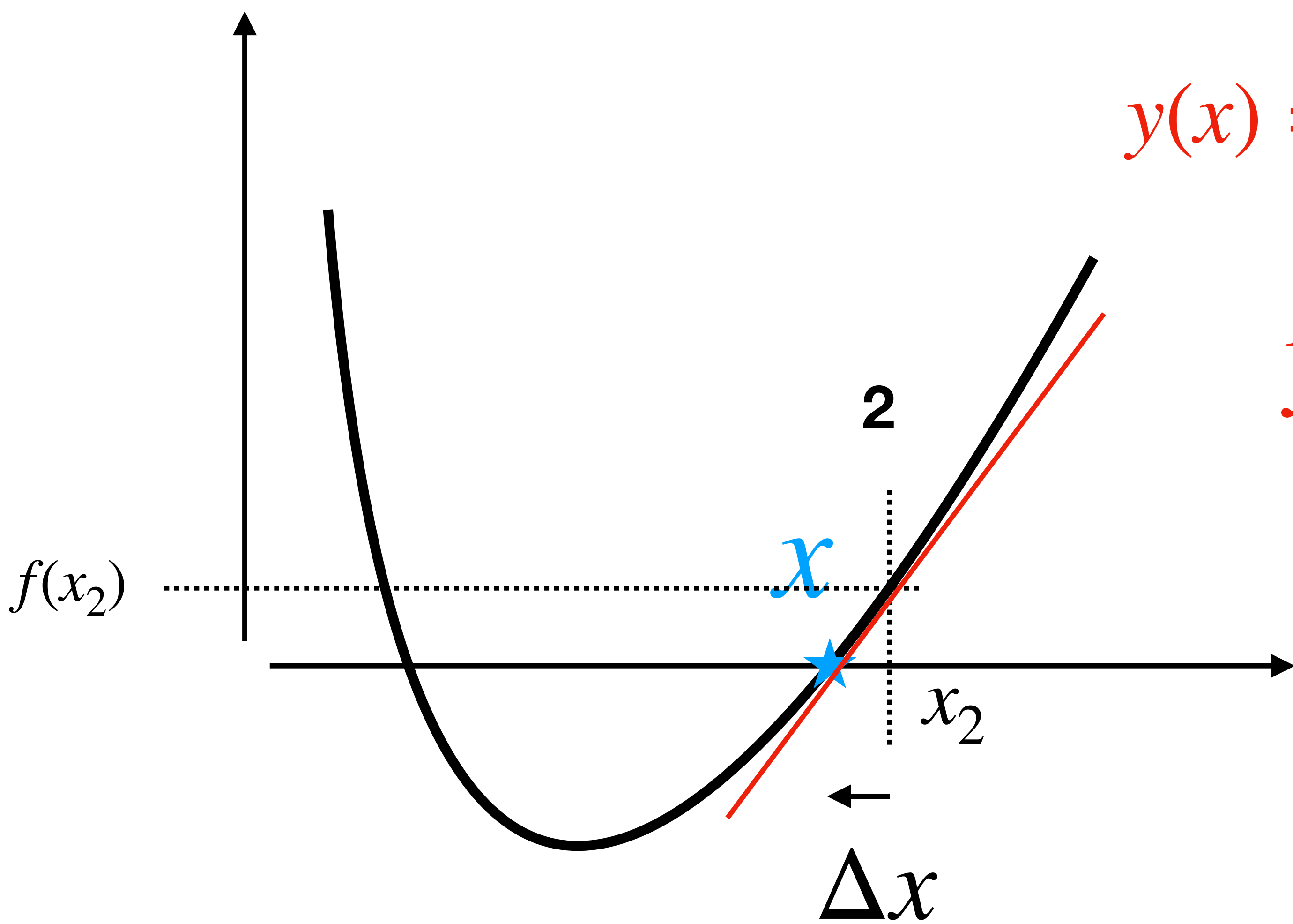
$$y(x) = f'(x_1)x + x_2$$

$$x_2 = x_1 + \Delta x$$

Newton-Raphson



Newton-Raphson

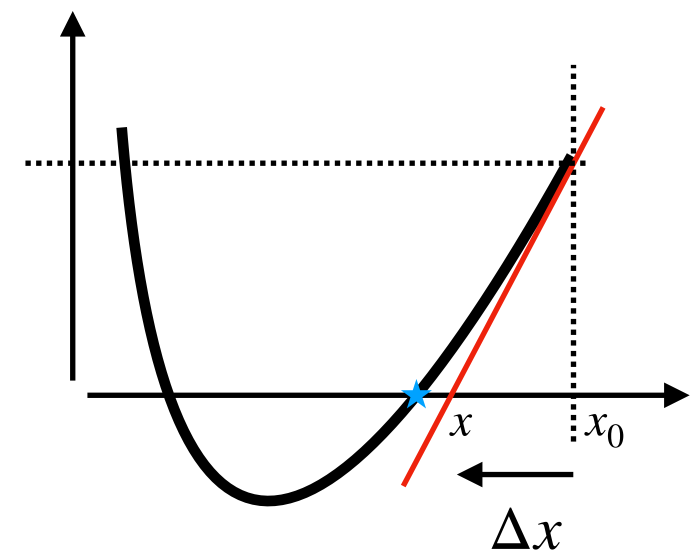


$$y(x) = mx + b$$

$$y(x) = f'(x_2)x + x_3$$

$$x_3 = x_2 + \Delta x$$

NR algorithm



$y(\xi) = f'(x_0)\xi + x$

$x = x_0 + \Delta x$

x_0	**initial guess for root**
x	**updated guess**

$x = x_0 + \Delta x$ **correction?**

$$f(x = x_0 + \Delta x) \approx f(x_0) + \Delta x \frac{df}{dx}\bigg|_{x_0}$$

$$f(x_0) + f'(x_0)\Delta x = 0$$

$$\Delta x = -\frac{f(x_0)}{f'(x_0)}$$

while $|f(x) > \epsilon|$

$$\Delta x = -\frac{f(x)}{f'(x)}$$

$$x \leftarrow x + \Delta x$$

Newton-Raphson

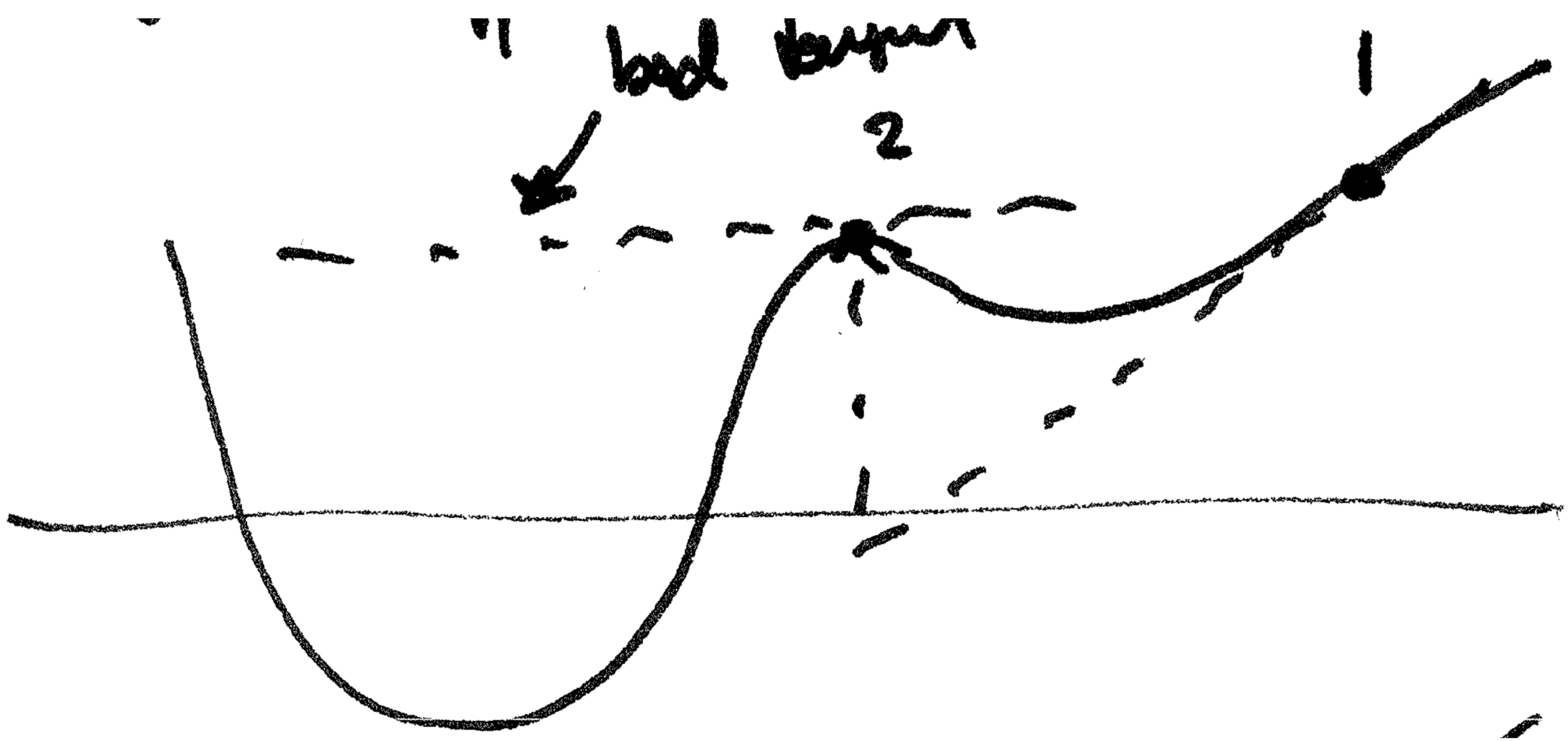
Advantages

- converges very quickly (quadratical convergence!!)
- fast
- works best with analytical derivative (but can use numerical ones)

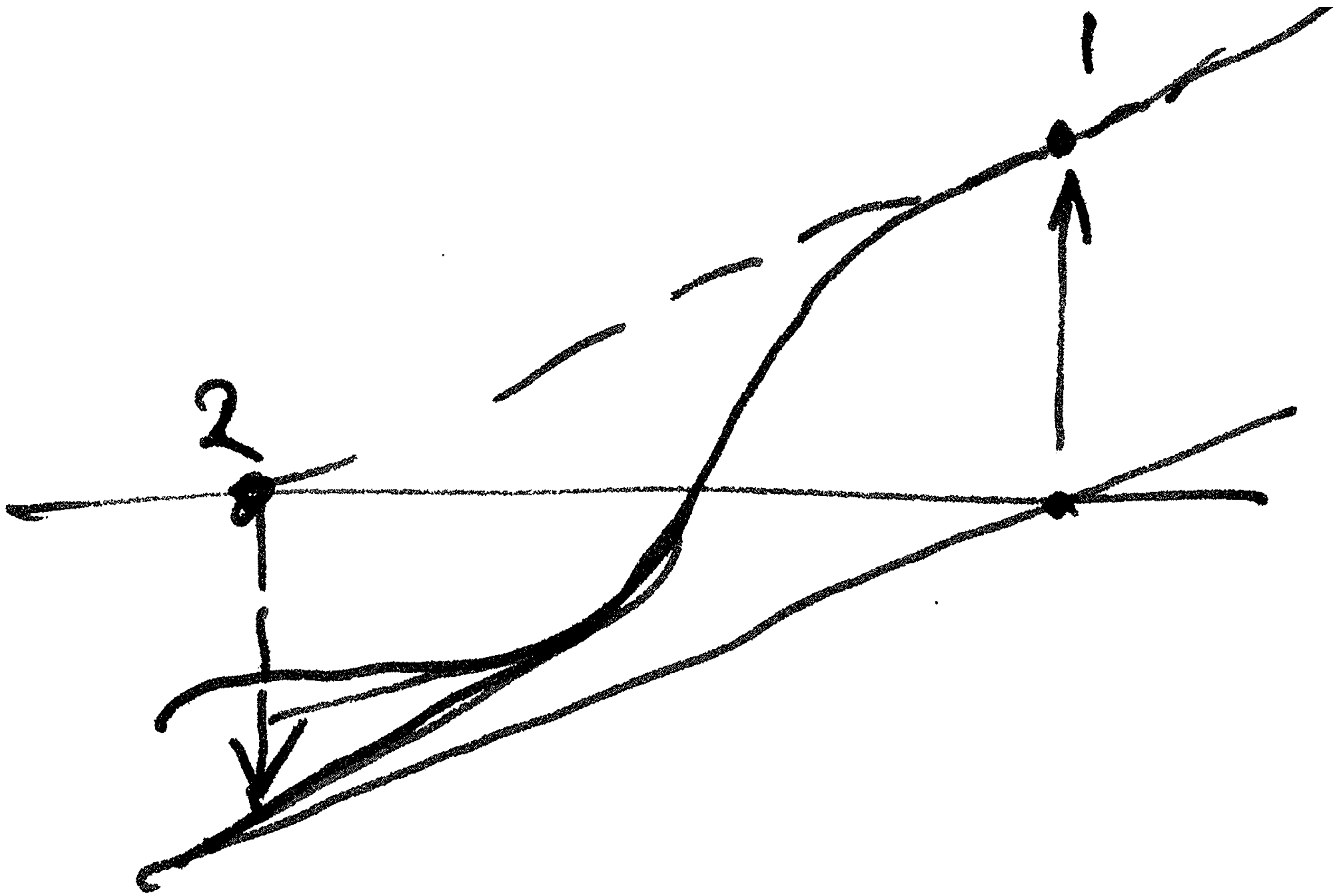
Disadvantages

- guess must be close to root
- can fail/loop in certain situations:

NR – FAILS!



NR – FAILS!



Improvements

- start with bisection to get close to root, then home in with Newton-Raphson

- implement *backtracking* : if new guess increases error then go back and try smaller guess

$$x \leftarrow x + \Delta x/2$$

Newton-Raphson