Elementary maths for GMT

Calculus

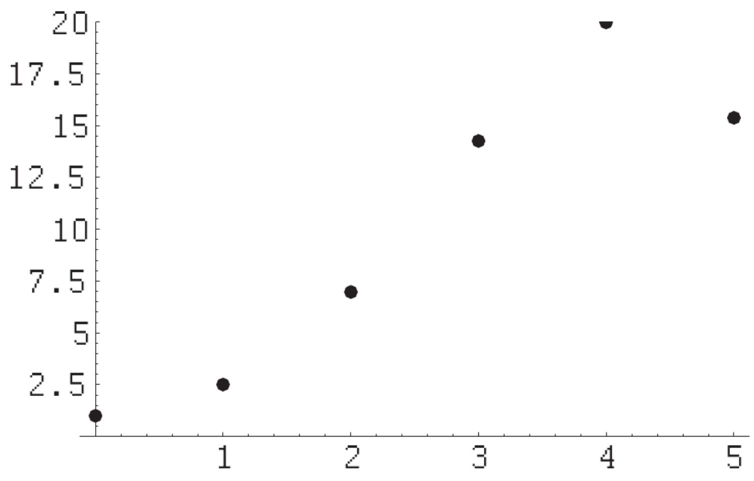
Part 3.2: Numerical methods

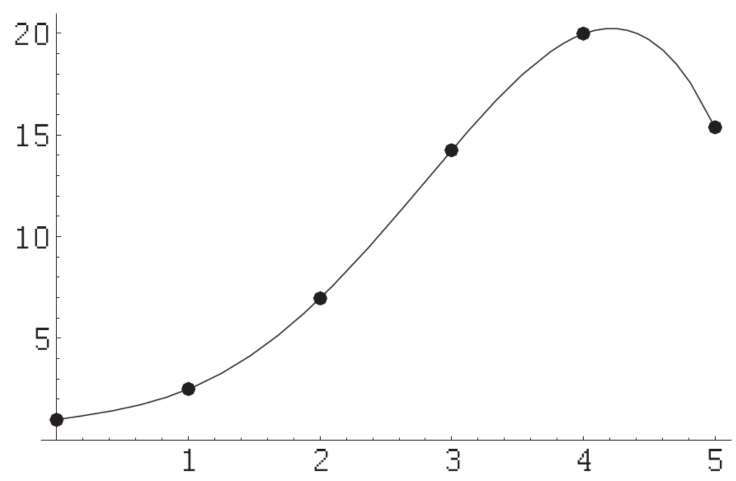
Outline

- Interpolation
- Finding roots
- Finding roots more easily

Interpolation

- Sometimes you have only a discrete number of points of a function *f*
  - For example, a table of values is given or a series of measurements has been conducted
- The task is to find values, or at least an approximation, in between the known points. This is often done by finding a polynomial of the smallest possible degree that passes through the given points.





Interpolation

- Find a polynomial

$$p_k(x) = c_0 + c_1 x + c_2 x^2 + \cdots + c_k x^k$$

of the smallest possible degree *k* that passes through given points $(x_0, y_0), (x_1, y_1), \cdots (x_n, y_n)$ (called support points)

Linear system of equations

- Filling all the support points results in a set of *(n+1)* linear equations with *k* unknowns

$$\begin{cases} c_0 + c_1 x_0 + c_2 {x_0}^2 + \cdots + c_k {x_0}^k = y_0 \\ c_0 + c_1 x_1 + c_2 {x_1}^2 + \cdots + c_k {x_1}^k = y_1 \\ \vdots \\ c_0 + c_1 x_n + c_2 {x_n}^2 + \cdots + c_k {x_n}^k = y_n \end{cases}$$

- This is a lot of work to solve
- Therefore we will show a different method

Lagrange interpolation

- Suppose we have *(n+1)* support points $(x_0, y_0), (x_1, y_1), \cdots (x_n, y_n)$ then the *Lagrange interpolation p* is

$$p(x) = p_0(x) + p_1(x) + \cdots + p_n(x)$$

with

$$p_j(x) = y_j \prod_{\substack{k=0 \\ k \neq j}}^{n} \frac{x - x_k}{x_j - x_k}$$

p_j written out:

$$p_j(x) = y_j \cdot \frac{x - x_0}{x_j - x_0} \cdot \frac{x - x_1}{x_j - x_1} \cdots \frac{x - x_{j-1}}{x_j - x_{j-1}} \cdot \frac{x - x_{j+1}}{x_j - x_{j+1}} \cdots \frac{x - x_n}{x_j - x_n}$$

Example

- Find the Lagrange interpolation through 3 support points

$$p_j(x) = y_j \prod_{\substack{k=0 \\ k \neq j}}^{2} \frac{x - x_k}{x_j - x_k}$$

$$p_0(x) = y_0 \cdot \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2}$$

$$p_1(x) = y_1 \cdot \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2}$$

$$p_2(x) = y_2 \cdot \frac{x - x_0}{x_2 - x_0} \cdot \frac{x - x_1}{x_2 - x_1}$$

Example

- Find the Lagrange interpolation through the support points $(0,0), (-2,4), (3,6)$

$$p_0(x) = y_0 \cdot \frac{x - x_1}{x_0 - x_1} \cdot \frac{x - x_2}{x_0 - x_2} = 0 \cdot \frac{(x+2)}{2} \cdot \frac{(x-3)}{-3} = 0$$

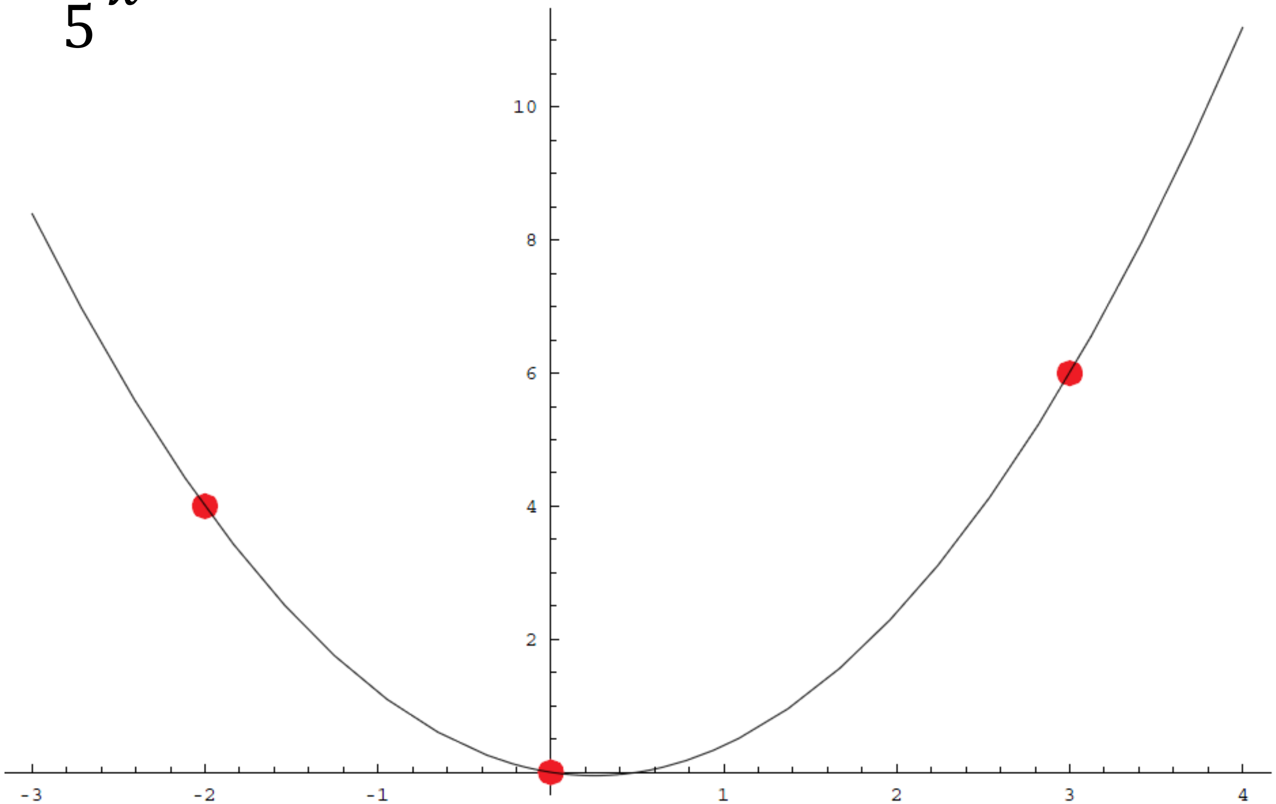
$$p_1(x) = y_1 \cdot \frac{x - x_0}{x_1 - x_0} \cdot \frac{x - x_2}{x_1 - x_2} = 4 \cdot \frac{(x-0)}{-2} \cdot \frac{(x-3)}{-5} = \frac{2}{5}(x^2 - 3x)$$

$$p_2(x) = y_2 \cdot \frac{x - x_0}{x_2 - x_0} \cdot \frac{x - x_1}{x_2 - x_1} = 6 \cdot \frac{(x-0)}{3} \cdot \frac{(x+2)}{5} = \frac{2}{5}(x^2 + 2x)$$

$$\begin{aligned} p(x) &= p_0(x) + p_1(x) + p_2(x) \\ &= \frac{2}{5}(x^2 - 3x) + \frac{2}{5}(x^2 + 2x) = \frac{4}{5}x^2 - \frac{2}{5}x \end{aligned}$$

Example

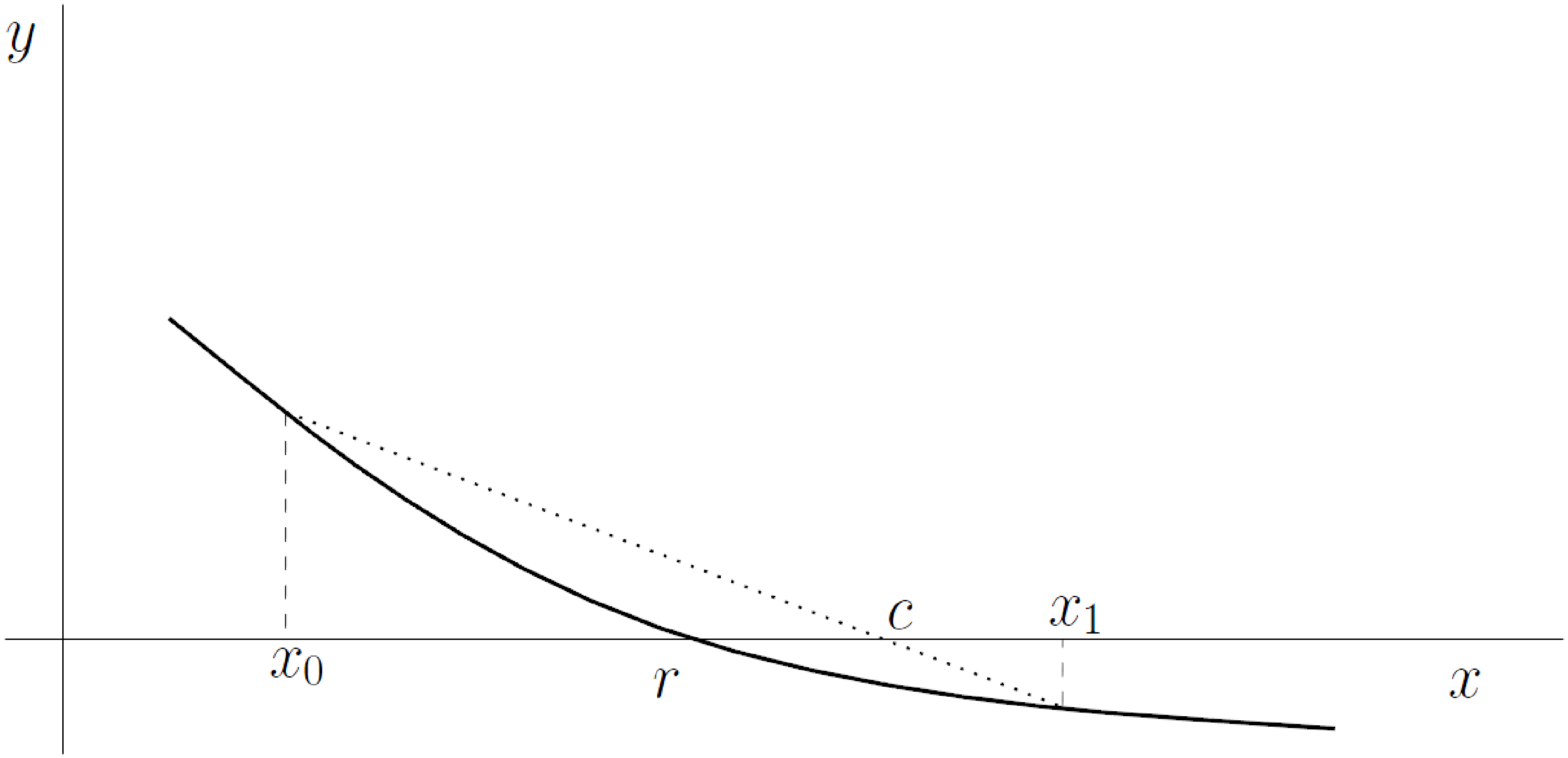
- Find the Lagrange interpolation through the support points $(0,0), (-2,4), (3,6)$
  $$p(x) = \frac{4}{5}x^2 - \frac{2}{5}x$$



Finding roots

- Suppose you want to find the roots (zero crossings) of a function *f(x)*
- Given an interval $[x_0,x_1]$ such that on one side of the interval the function is above the x-axis and on the other side of the interval below the x-axis, there is at least one root in the interval
- An approximation of this root can be found by performing a linear interpolation between the two sides of the interval, $(x_0, f(x_0))$ and $(x_1, f(x_1))$, and calculating the intersection of this line with the x-axis

Finding roots



Finding roots: Regula falsi

- Using Lagrange interpolation we find that the interpolation polynomial (a line in this case) is:

$$p(x) = f(x_0) \cdot \frac{x - x_1}{x_0 - x_1} + f(x_1) \cdot \frac{x - x_0}{x_1 - x_0}$$

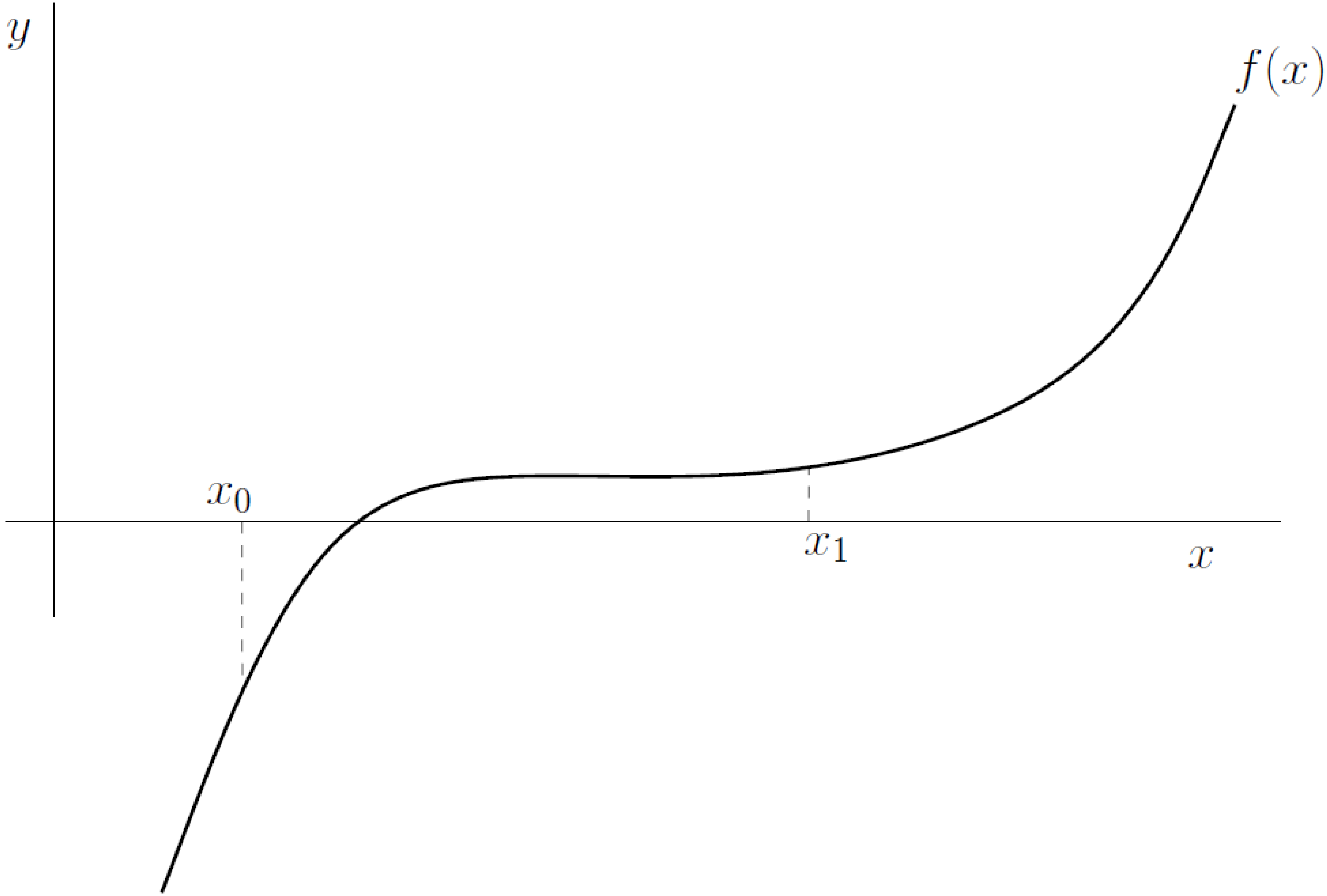
- The intersection point of this line and the x-axis is *(c,0)*, thus:

$$p(c) = f(x_0) \cdot \frac{c - x_1}{x_0 - x_1} + f(x_1) \cdot \frac{c - x_0}{x_1 - x_0} = 0$$

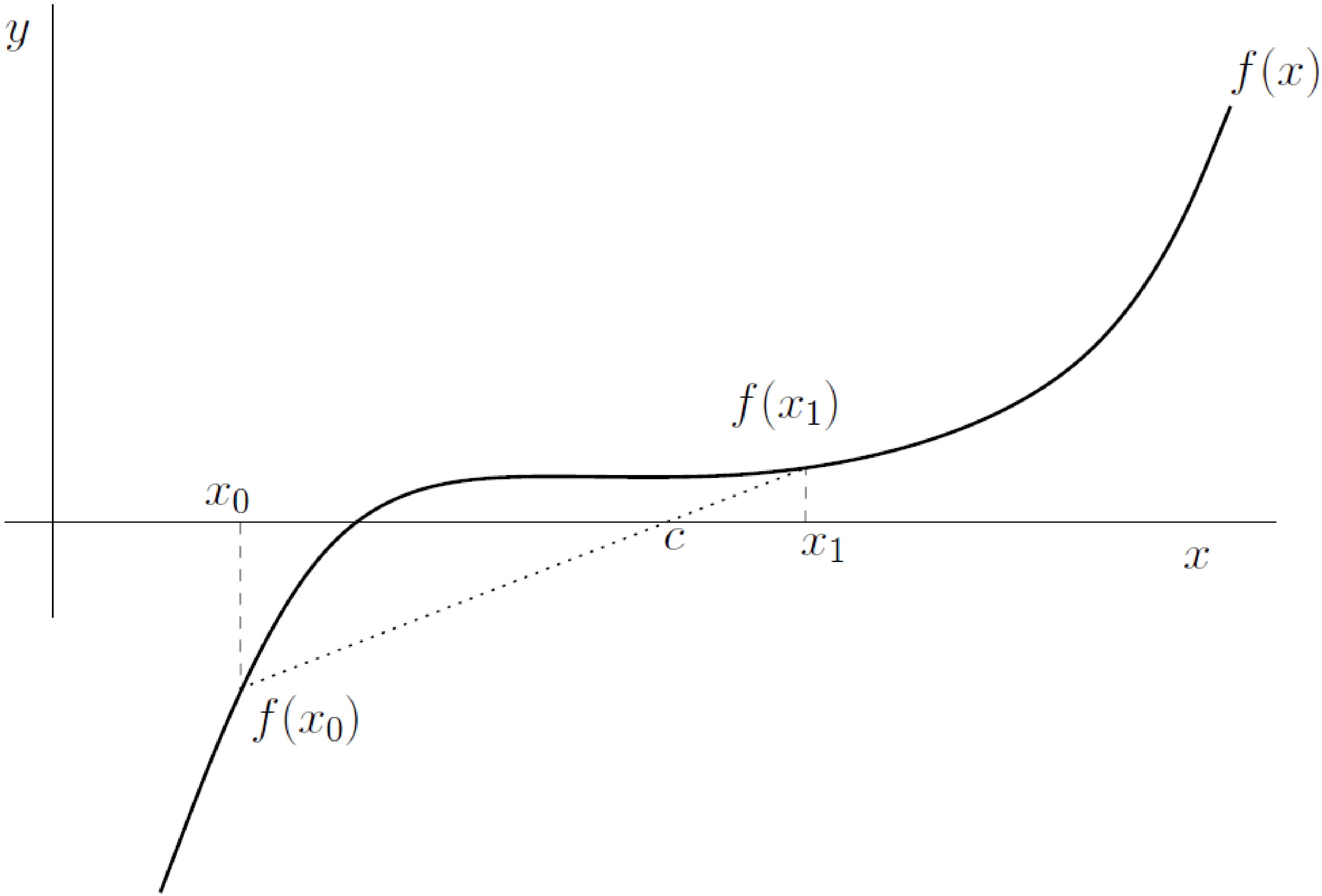
$$c = \frac{x_0 f(x_1) - x_1 f(x_0)}{f(x_1) - f(x_0)}$$

- Using this approximation of the root we can narrow the search, recursively

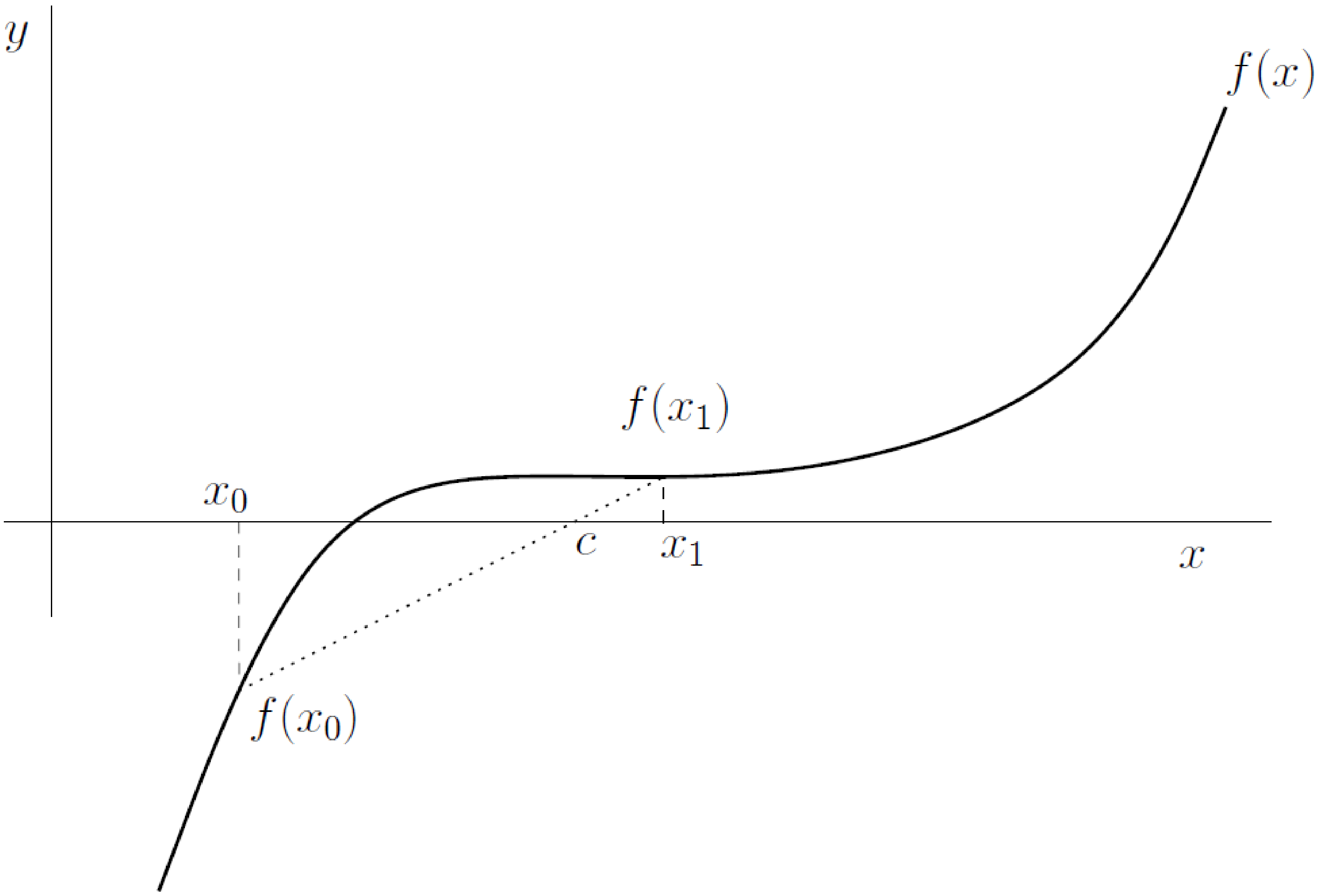
Finding roots: Regula falsi



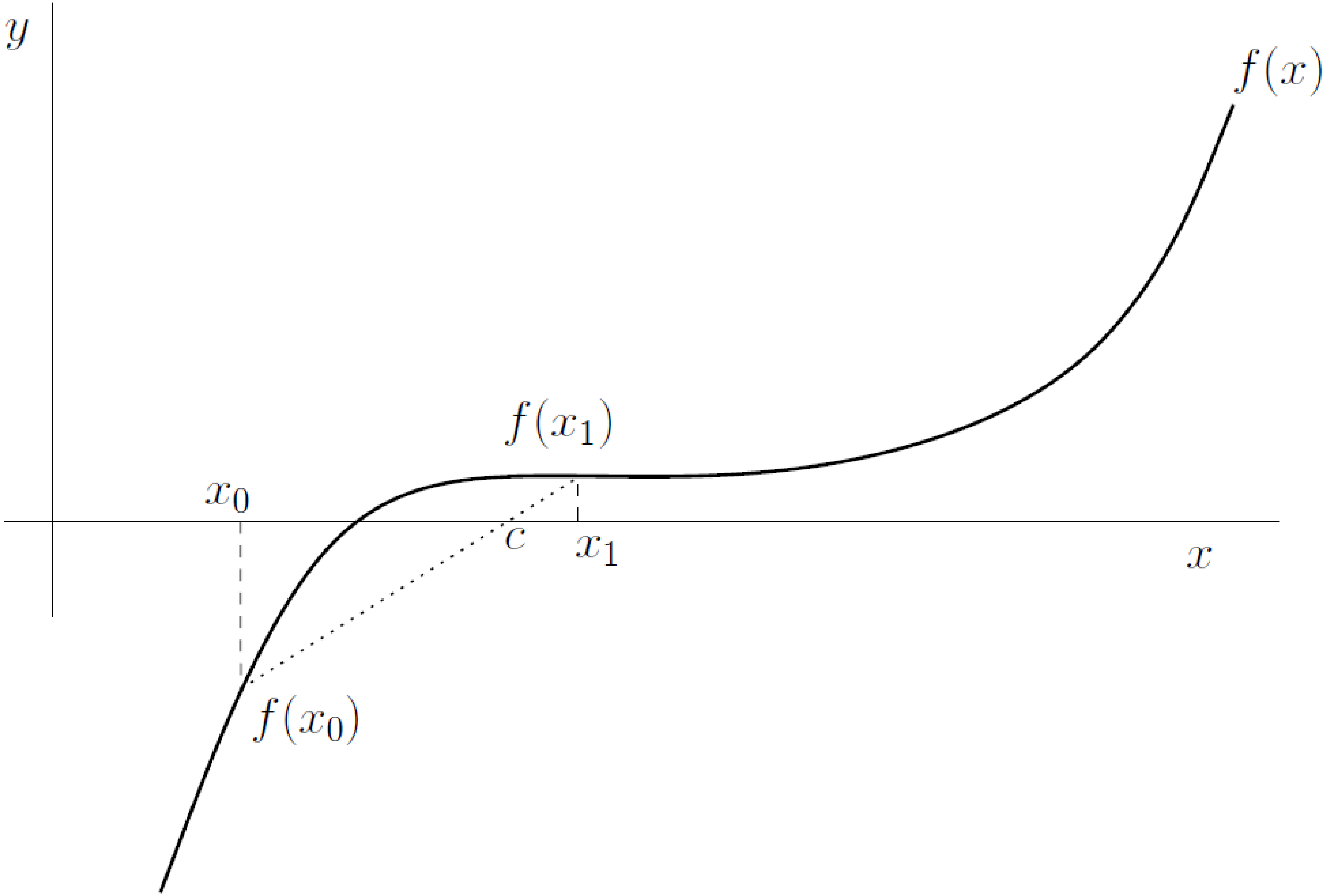
Finding roots: Regula falsi



Finding roots: Regula falsi



Finding roots: Regula falsi



Finding roots: Regula falsi

- If $f(c) = 0$, the root is found (at ±ε)
- If $f(c)f(x_0) < 0$, the root is to the left of *c*, thus iterate the process with x_0 and $x_1 = c$
- If $f(c)f(x_0) > 0$, the root is to the right of *c*, thus iterate the process with $x_0 = c$ and x_1

Finding roots: Picard iteration

- The regula falsi iteration needs two initial points bracketing a root. *Picard iteration* uses only one point.
- We need to change the equation

$$f(x) = 0 \tag{1}$$

into

$$g(x) = \mathrm{x} \tag{2}$$

with g chosen so that the solutions to equation 1 are equal to the solutions to equation 2

- By taking $x_{n+1} = g(x_n)$, and beginning with x_0 as an initial guess we approximate the root
  - If you want to understand how this works, make a drawing and realize that an iteration basically mirrors the function in the line *y* = *x*

Example

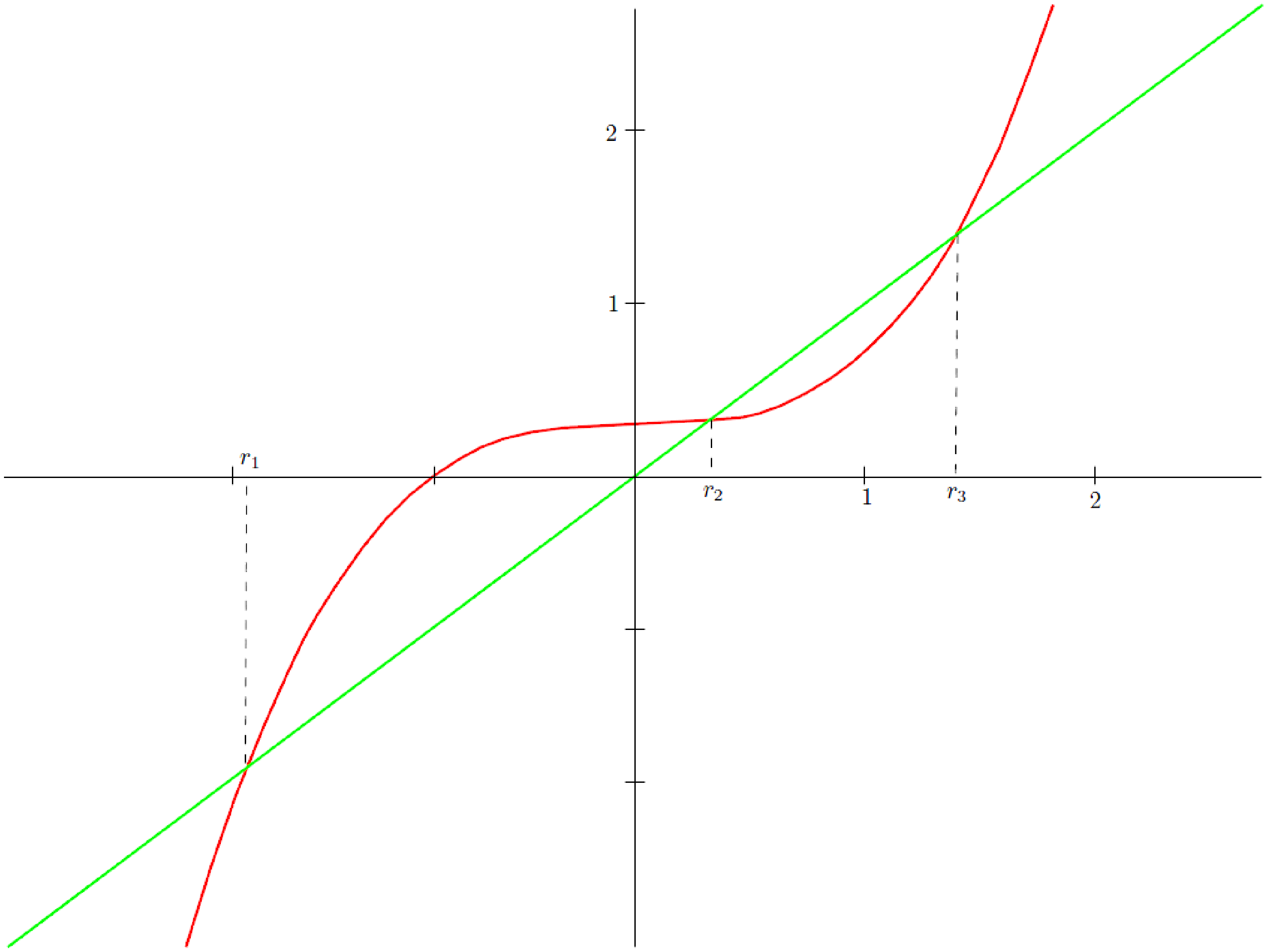
- Let assume the function $f(x) = x^3 - 3x + 1$
- Set *f* to zero, then isolate *x* to find a possible *g*
  $g(x) = \frac{1}{3}(x^3 + 1)$
- Thus, $x_{n+1} = g(x_n) = \frac{1}{3}({x_n}^3 + 1)$
- Therefore taking $x_0 = 1$ results in the following iterations

$$x_1 = g(x_0) = \frac{2}{3} \approx 0.6667$$

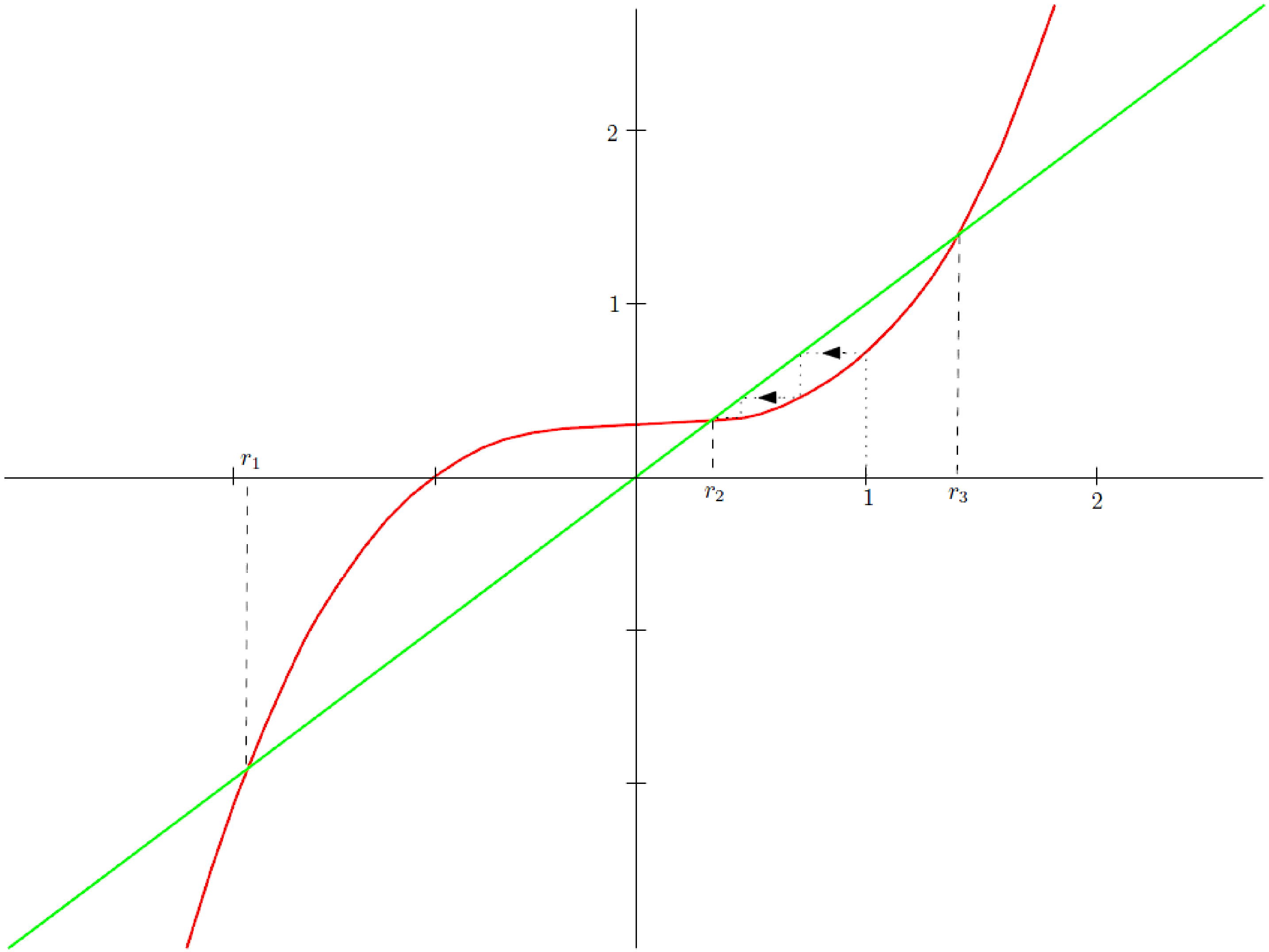
$$x_2 = g(x_1) = \frac{35}{81} \approx 0.4321$$

$$x_3 = g(x_2) = \frac{574316}{1594232} \approx 0.36$$

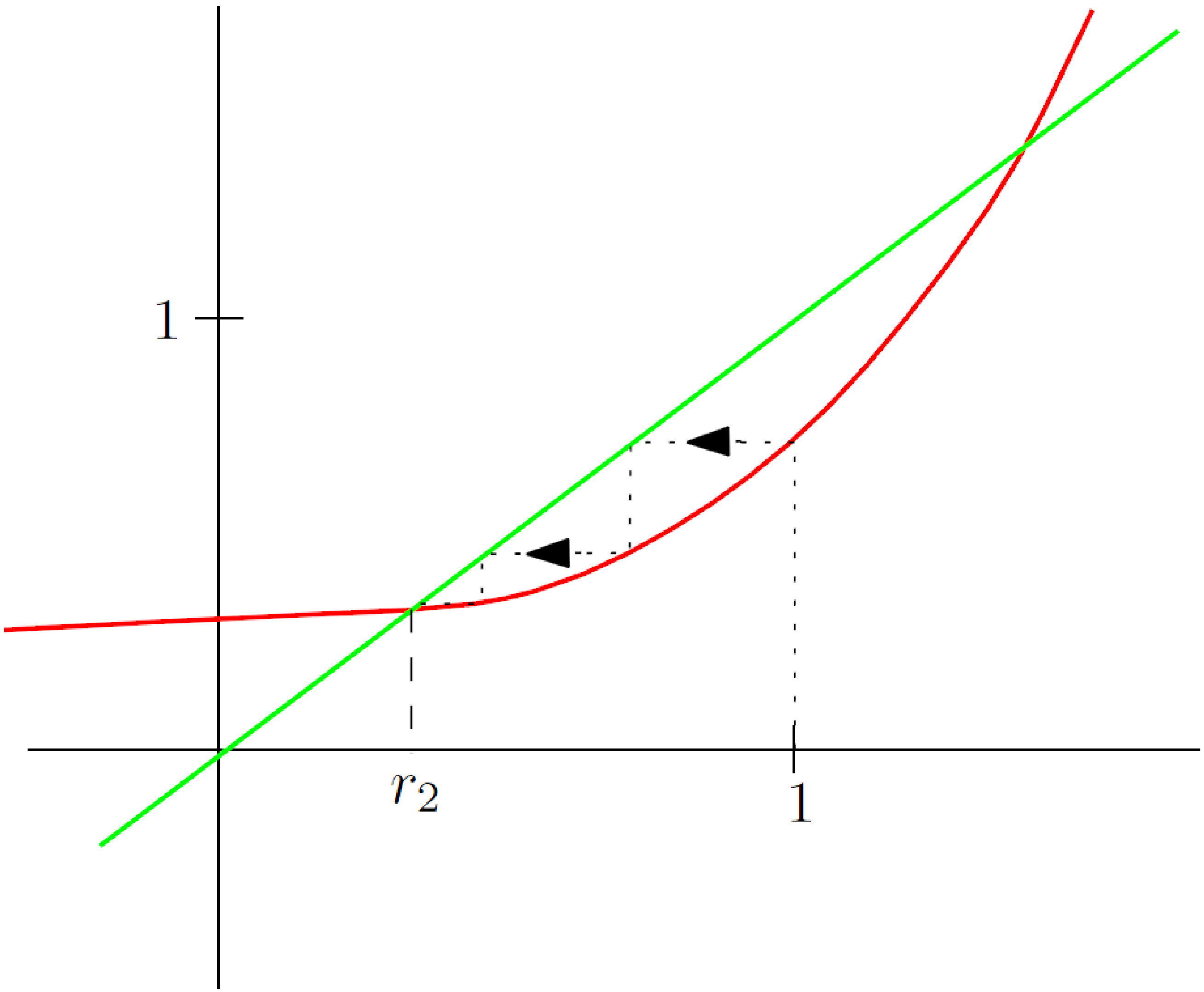
Example



Example



Example

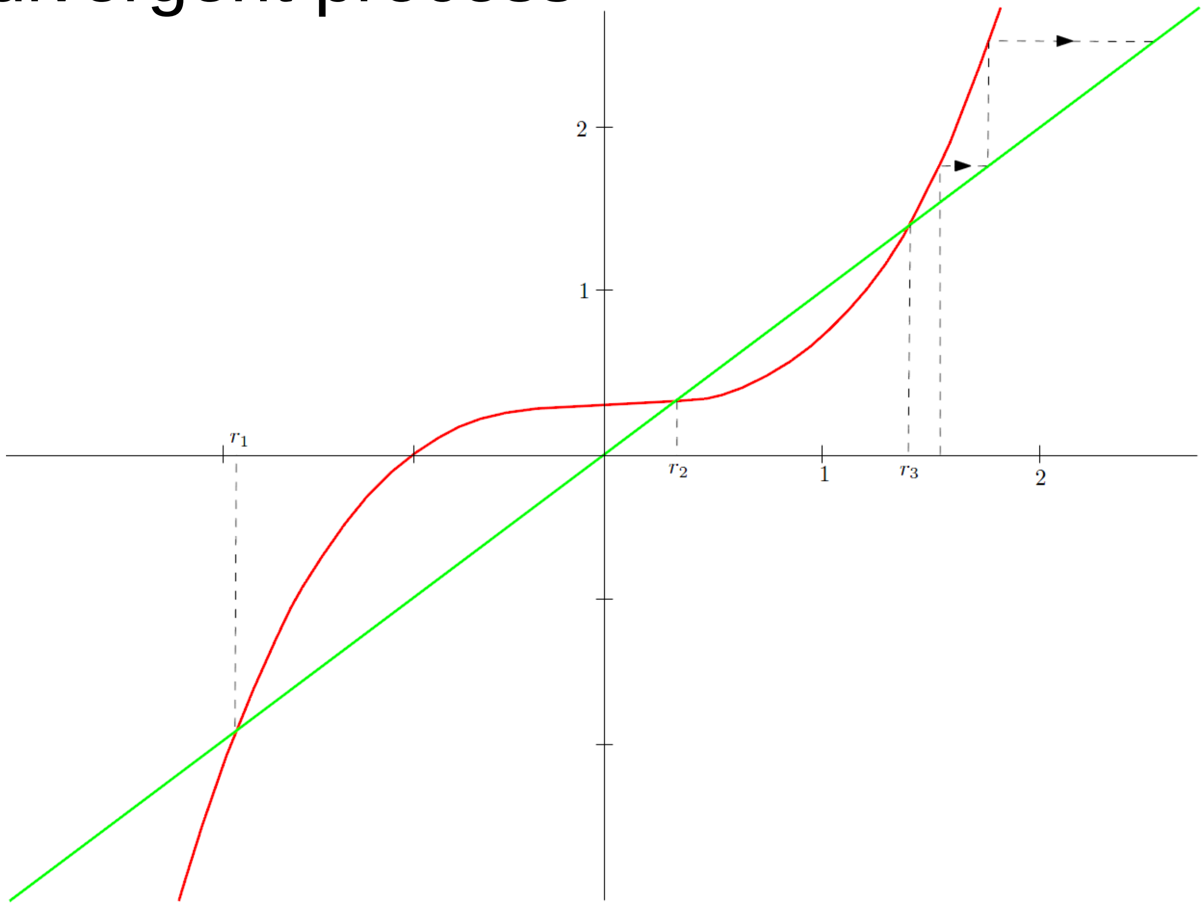


Finding roots: Picard iteration

- Drawback
  - There is no guarantee that the process converges, and finds a specific root
  - Taking a different initial guess can result in finding a different root

Example

- For example, taking as initial guess $x_0 = 1.6$ will give a divergent process



Finding roots: Newton-Raphson

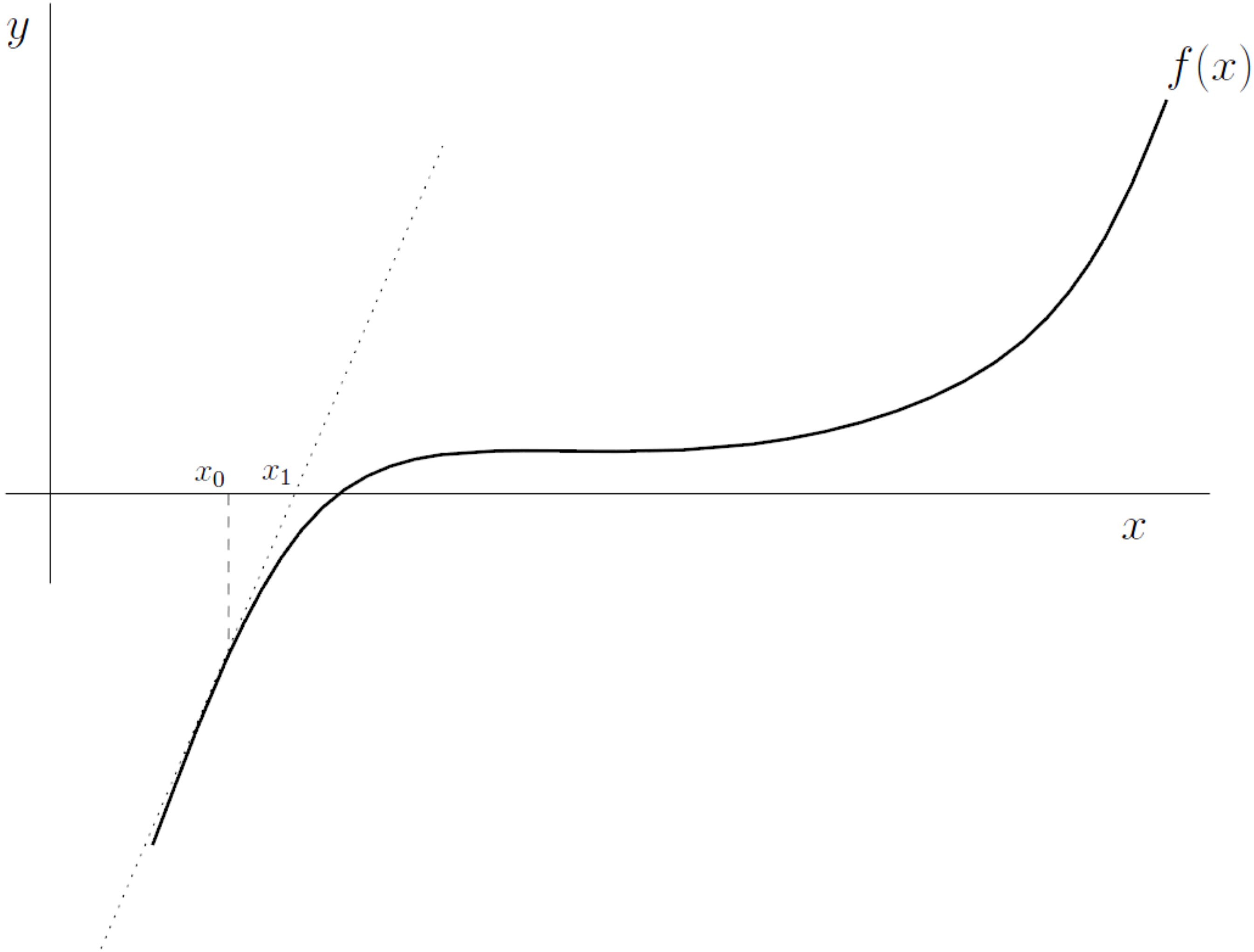
- Similar to Picard iteration but converges faster
- The roots of $f(x) = 0$ are approximated by defining *g* as

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

and iterating toward the solution using

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Example



Example