Newton's Method

Announcements

Announcements

- If you want 1 unit (pass/no pass) of credit for in CS 98-52, the CCN is 28867

Announcements

- If you want 1 unit (pass/no pass) of credit for in CS 98-52, the CCN is 28867
  - Only for people who really want extra work that's beyond the scope of normal CS 61A

Announcements

- If you want 1 unit (pass/no pass) of credit for in CS 98-52, the CCN is 28867
  - Only for people who really want extra work that's beyond the scope of normal CS 61A
- Anyone is welcome to attend the extra lectures, whether or not they enroll

Announcements

- If you want 1 unit (pass/no pass) of credit for in CS 98-52, the CCN is 28867
  - Only for people who really want extra work that's beyond the scope of normal CS 61A
- Anyone is welcome to attend the extra lectures, whether or not they enroll
- All info and materials will be posted to cs61a.org/extra.html

Lambda Expressions

(Demo)

Lambda Expressions

Lambda Expressions

```
>>> x = 10
```

Lambda Expressions

```
>>> x = 10

>>> square = x * x
```

Lambda Expressions

```
>>> x = 10

>>> square = x * x
```

An expression: this one evaluates to a number

Lambda Expressions

```
>>> x = 10
```

An expression: this one evaluates to a number

```
>>> square = x * x
```

```
>>> square = lambda x: x * x
```

Lambda Expressions

```
>>> x = 10
```

An expression: this one evaluates to a number

```
>>> square = x * x
```

Also an expression: evaluates to a function

```
>>> square = lambda x: x * x
```

Lambda Expressions

```
>>> x = 10

>>> square = x * x

>>> square = lambda x: x * x
```

An expression: this one evaluates to a number

Also an expression: evaluates to a function

A function

Lambda Expressions

```
>>> x = 10
```

An expression: this one evaluates to a number

```
>>> square = x * x
```

Also an expression: evaluates to a function

```
>>> square = lambda x: x * x
```

A function

with formal parameter x

Lambda Expressions

```
>>> x = 10
```

An expression: this one evaluates to a number

```
>>> square = x * x
```

Also an expression: evaluates to a function

```
>>> square = lambda x: x * x
```

A function

with formal parameter x

that returns the value of "x * x"

Lambda Expressions

```
>>> x = 10
```

An expression: this one evaluates to a number

```
>>> square = x * x
```

Also an expression: evaluates to a function

```
>>> square = lambda x: x * x
```

Important: No "return" keyword!

A function

with formal parameter x

that returns the value of "x * x"

Lambda Expressions

```
>>> x = 10
```

An expression: this one evaluates to a number

```
>>> square = x * x
```

Also an expression: evaluates to a function

```
>>> square = lambda x: x * x
```

Important: No "return" keyword!

A function

with formal parameter x

that returns the value of "x * x"

Must be a single expression

Lambda Expressions

```
>>> x = 10
```

An expression: this one evaluates to a number

```
>>> square = x * x
```

Also an expression: evaluates to a function

```
>>> square = lambda x: x * x
```

Important: No "return" keyword!

A function

with formal parameter x

that returns the value of "x * x"

Must be a single expression

```
>>> square(4)
16
```

Newton's Method

Newton's Method Background

`Quickly finds accurate approximations to zeroes of differentiable functions!`

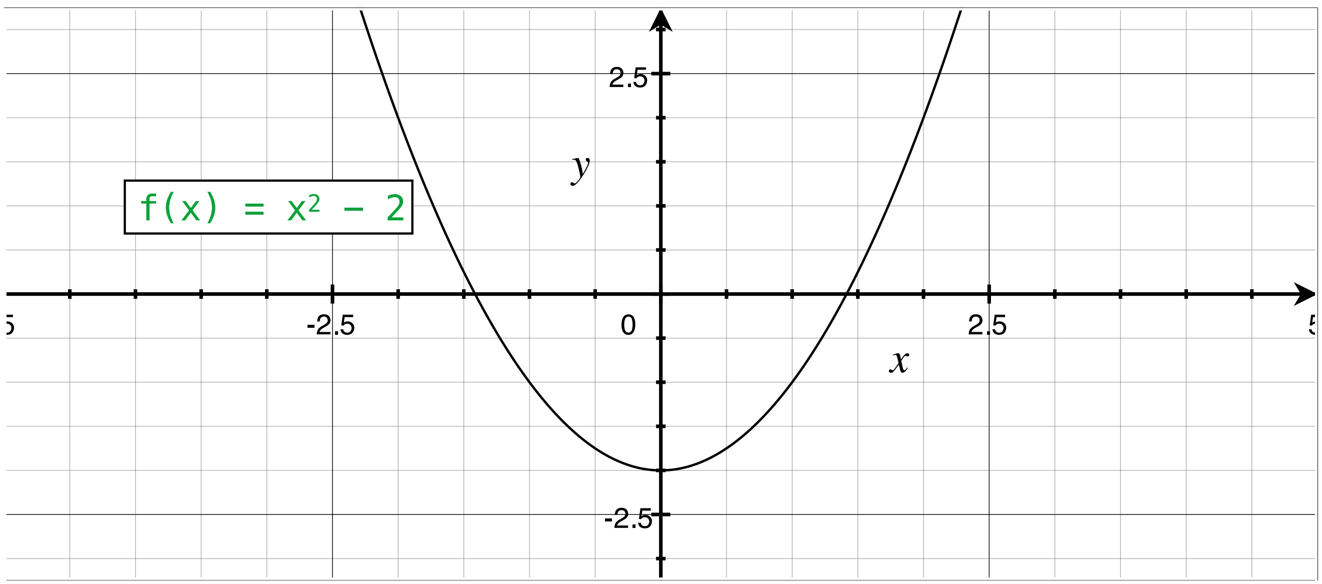
Newton's Method Background

Quickly finds accurate approximations to zeroes of differentiable functions!

$\boxed{f(x) = x^2 - 2}$

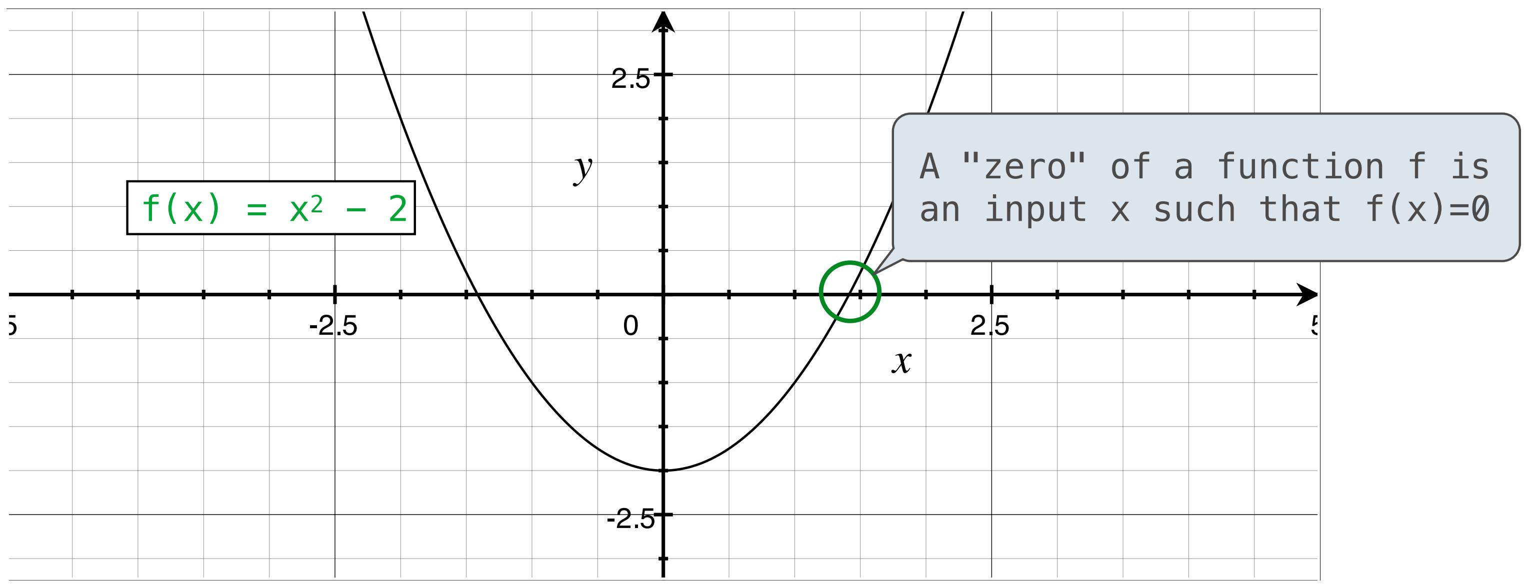
Newton's Method Background

`Quickly finds accurate approximations to zeroes of differentiable functions!`



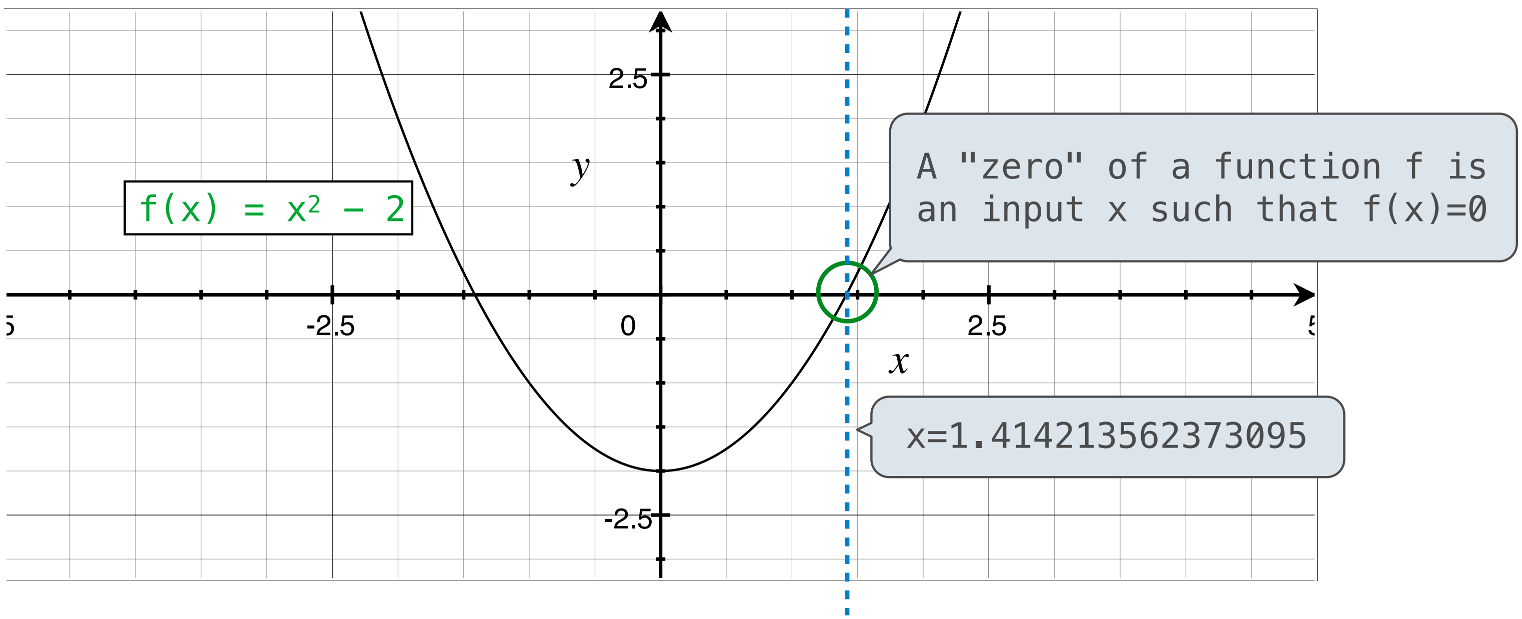
Newton's Method Background

Quickly finds accurate approximations to zeroes of differentiable functions!



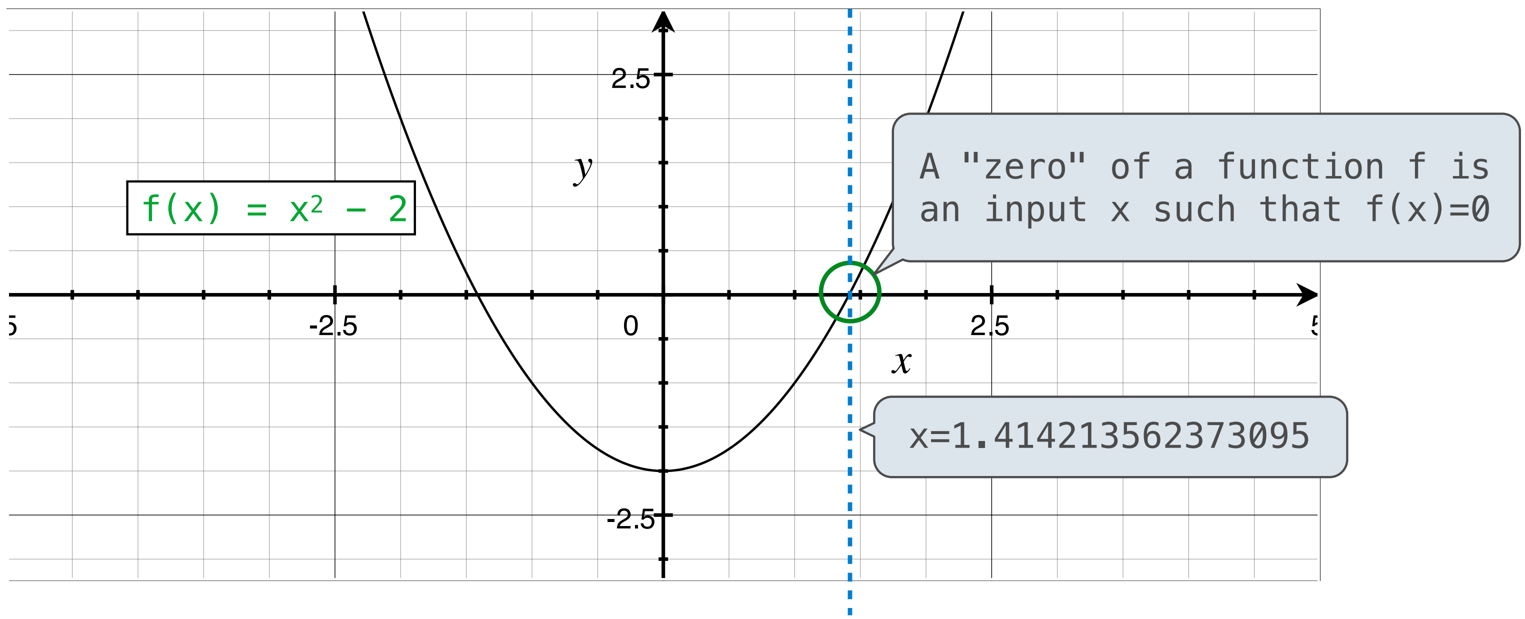
Newton's Method Background

Quickly finds accurate approximations to zeroes of differentiable functions!



Newton's Method Background

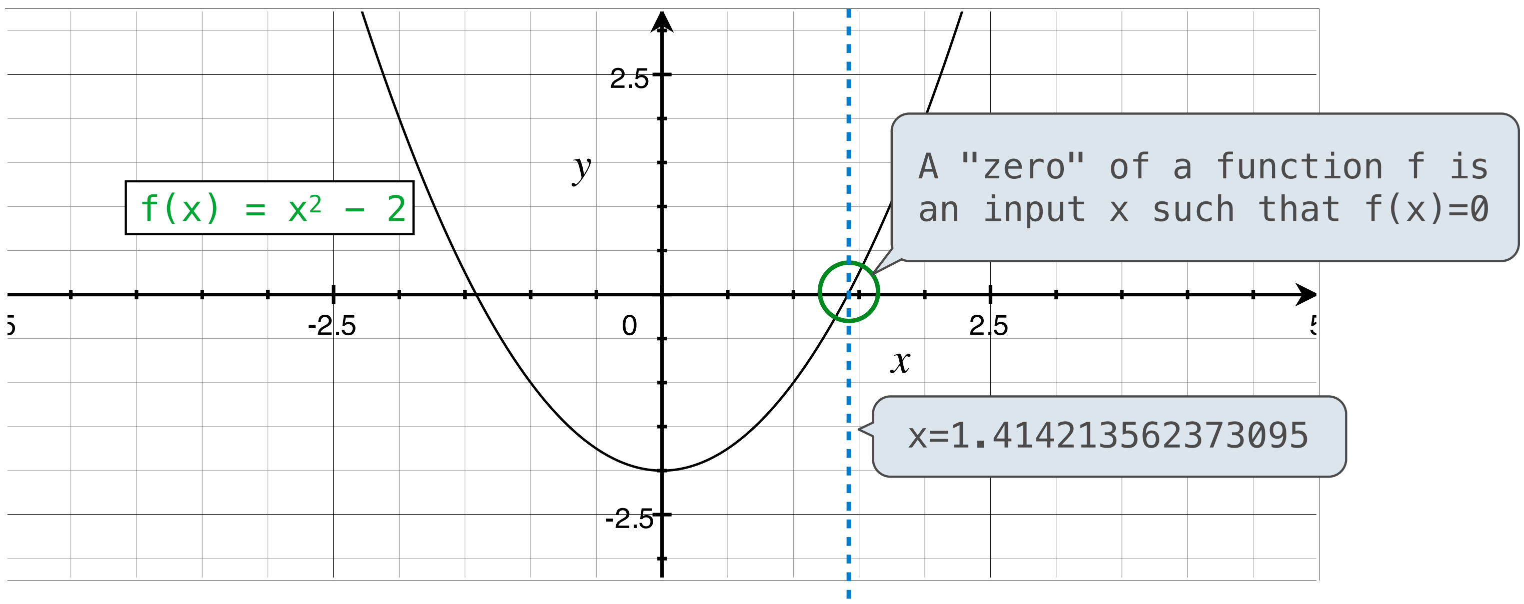
Quickly finds accurate approximations to zeroes of differentiable functions!



Application: a method for computing square roots, cube roots, etc.

Newton's Method Background

Quickly finds accurate approximations to zeroes of differentiable functions!



Application: a method for computing square roots, cube roots, etc.

The positive zero of $f(x) = x^2 - a$ is $\sqrt{a}$. (We're solving the equation $x^2 = a$.)

Newton's Method

Given a function f and initial guess x,

Newton's Method

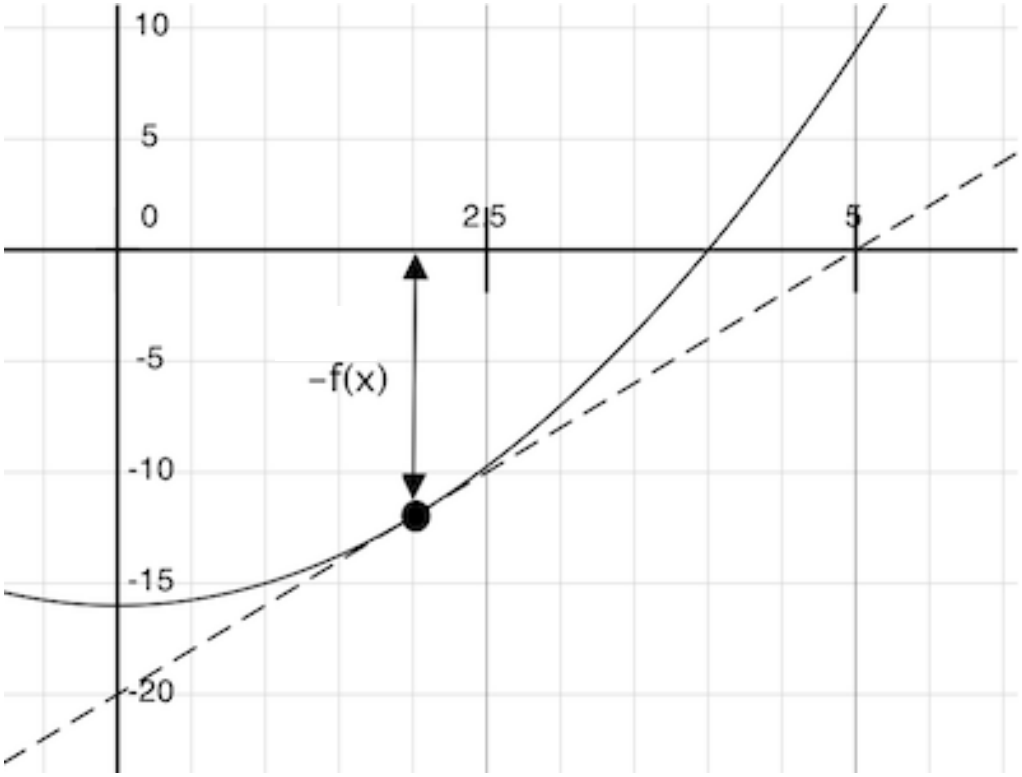
Given a function f and initial guess x,

Repeatedly improve x:

Newton's Method

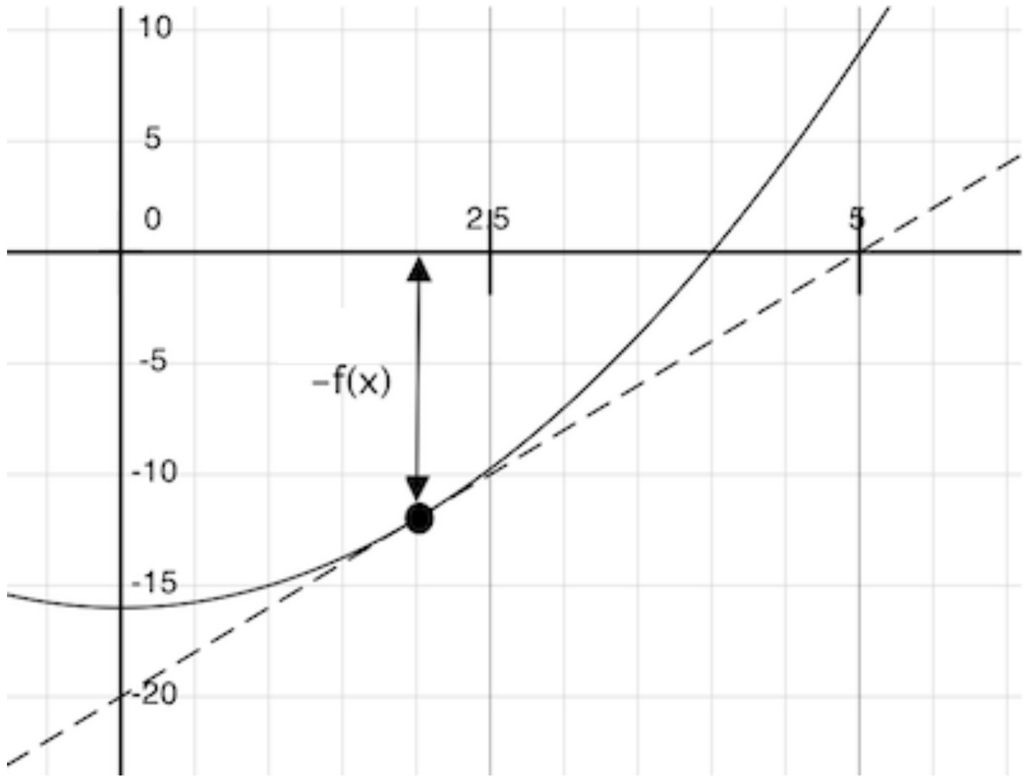
Given a function f and initial guess x,

Repeatedly improve x:



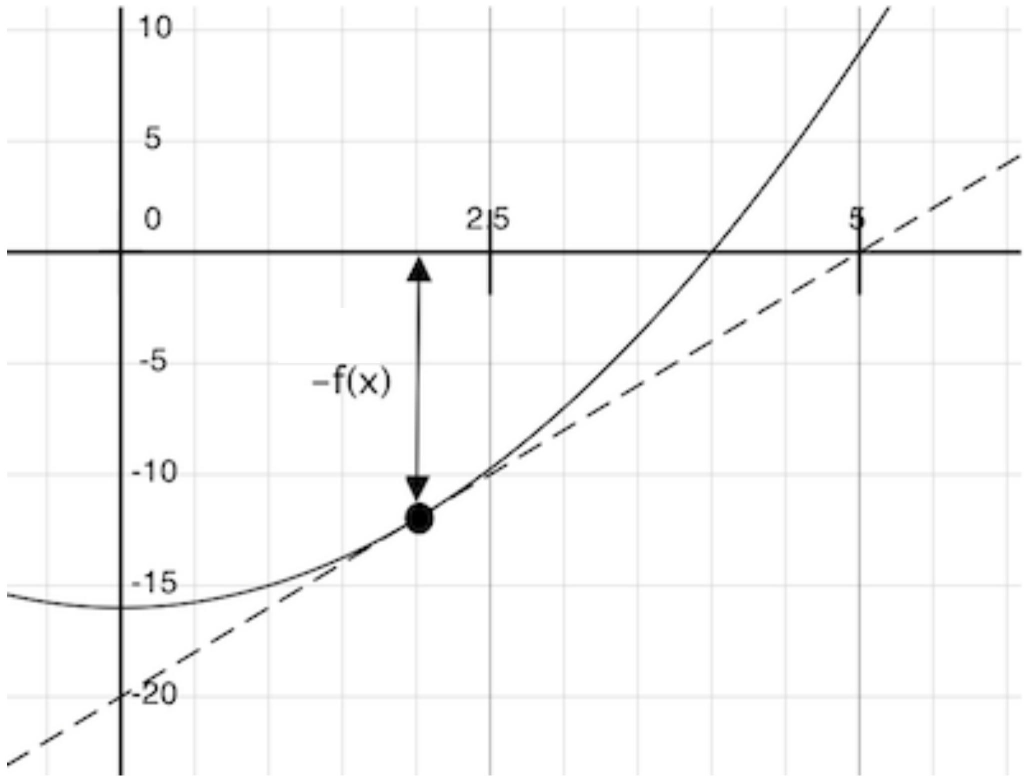
Newton's Method

```
Given a function f and initial guess x,


Repeatedly improve x:

     Compute the value of f
     at the guess: f(x)
```



Newton's Method

```
Given a function f and initial guess x,

Repeatedly improve x:

      Compute the value of f
      at the guess: f(x)

      Compute the derivative
      of f at the guess: f'(x)
```



Newton's Method

Given a function f and initial guess x,

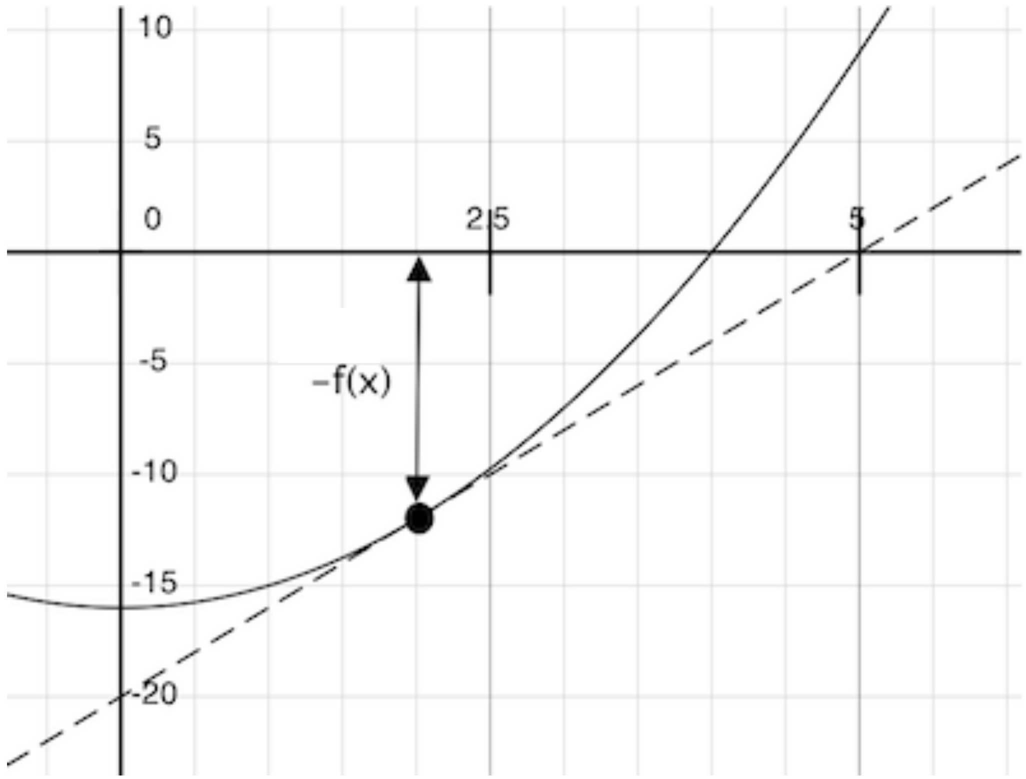
Repeatedly improve x:

Compute the value of f at the guess: f(x)

Compute the derivative of f at the guess: f'(x)

Update guess x to be:

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



Newton's Method

Given a function f and initial guess x,

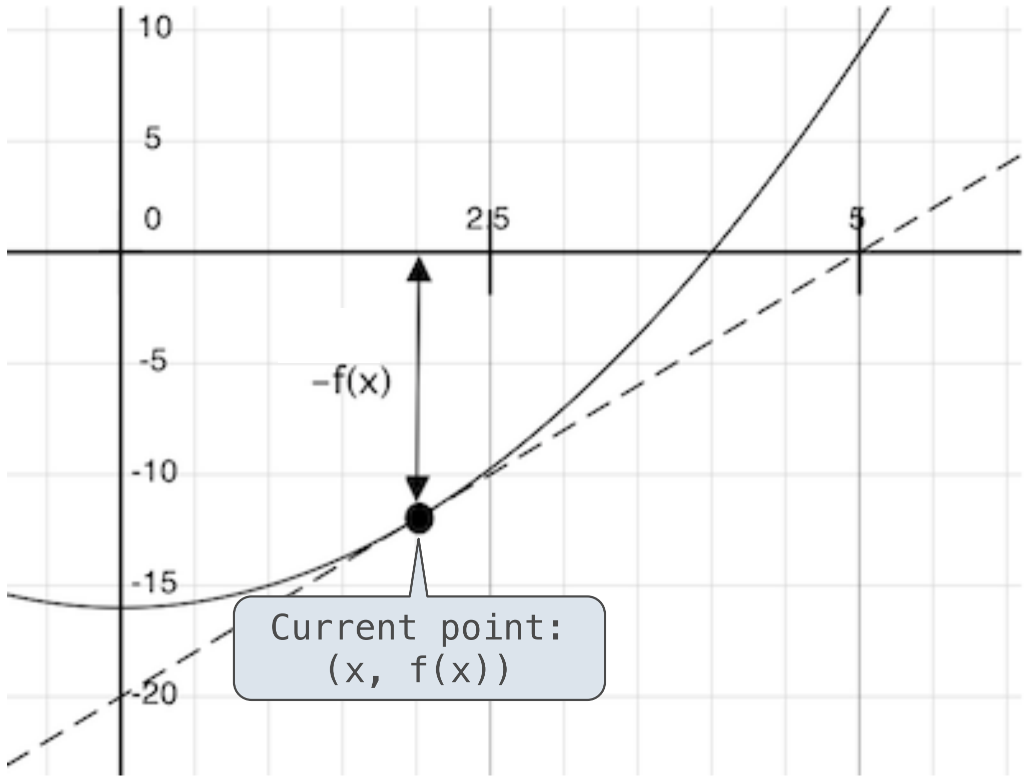
Repeatedly improve x:

Compute the value of f at the guess: f(x)

Compute the derivative of f at the guess: f'(x)

Update guess x to be:

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



Newton's Method

Given a function f and initial guess x,

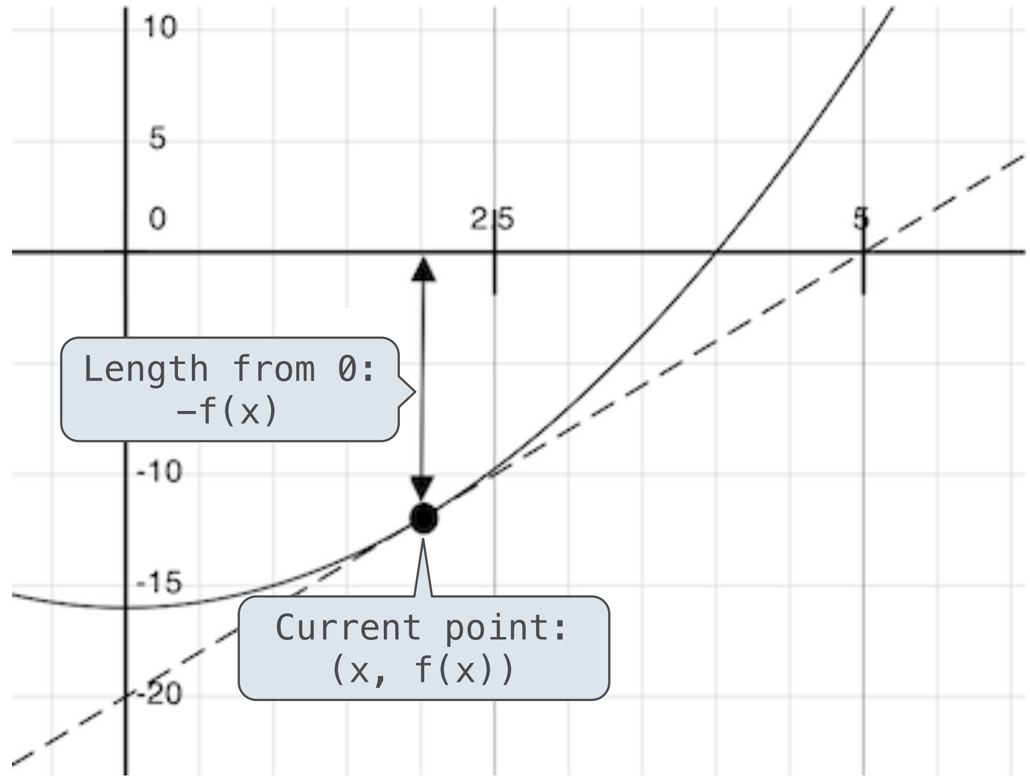
Repeatedly improve x:

Compute the value of f at the guess: f(x)

Compute the derivative of f at the guess: f'(x)

Update guess x to be:

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



Newton's Method

Given a function f and initial guess x,

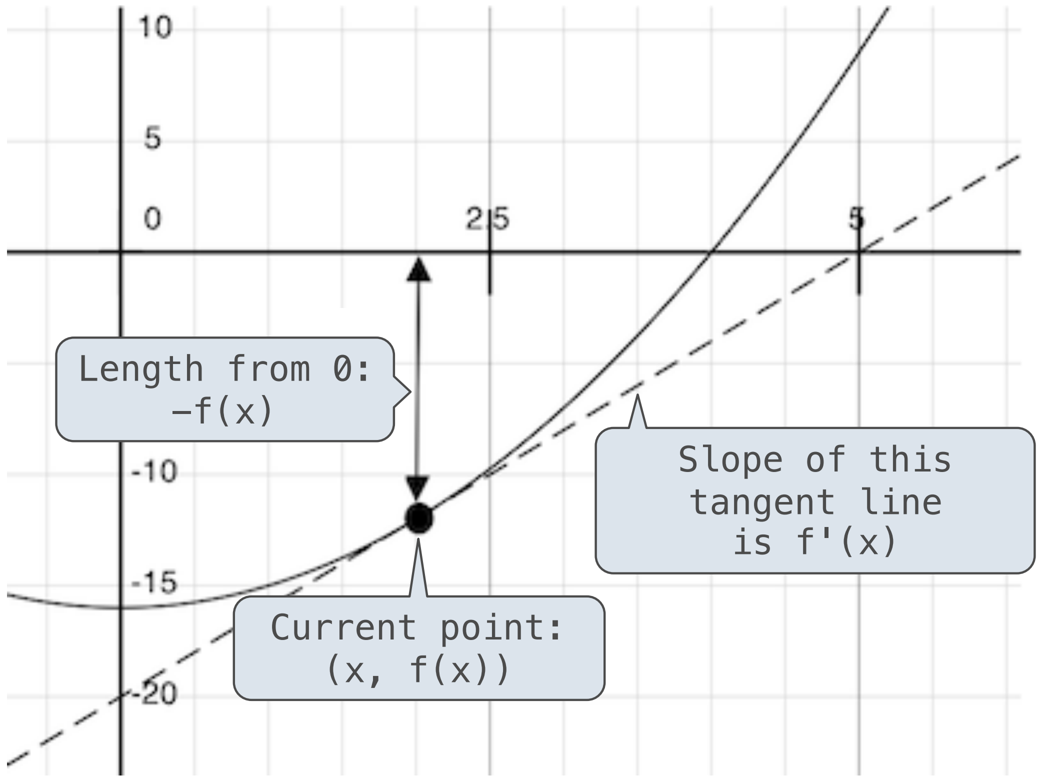
Repeatedly improve x:

Compute the value of f at the guess: f(x)

Compute the derivative of f at the guess: f'(x)

Update guess x to be:

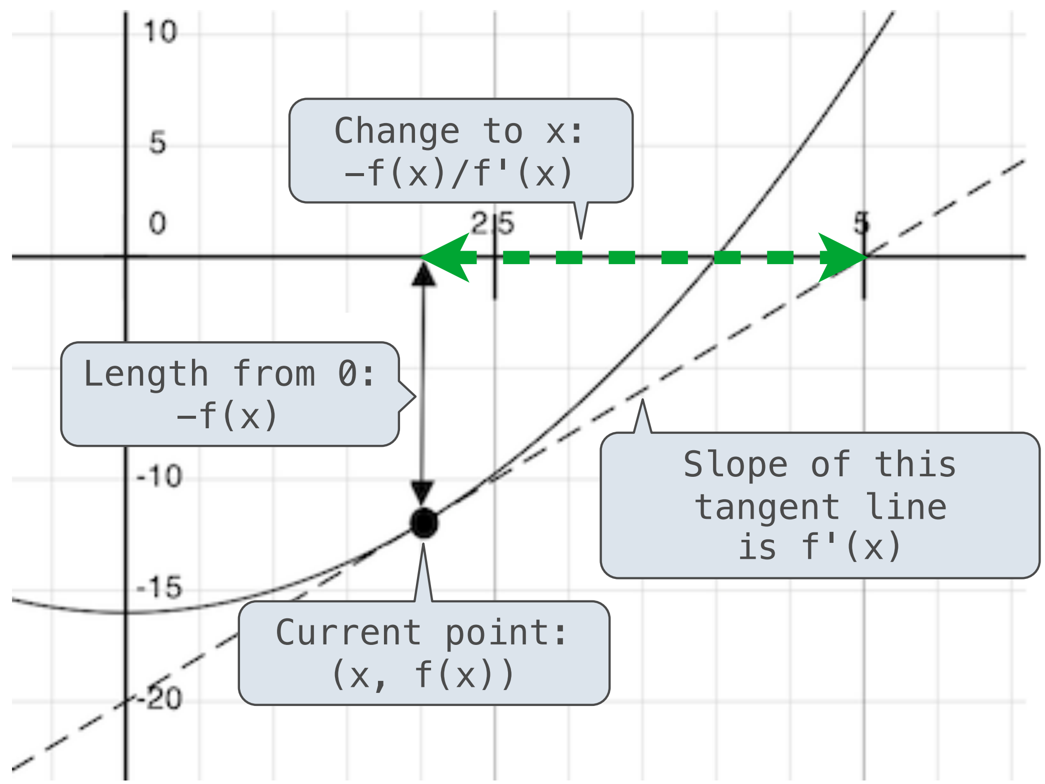
$$x - \frac{f(x)}{f'(x)}$$



Newton's Method

```
Given a function f and initial guess x,

Repeatedly improve x:

    Compute the value of f
    at the guess: f(x)

    Compute the derivative
    of f at the guess: f'(x)

    Update guess x to be:
```

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



Newton's Method

Given a function f and initial guess x,

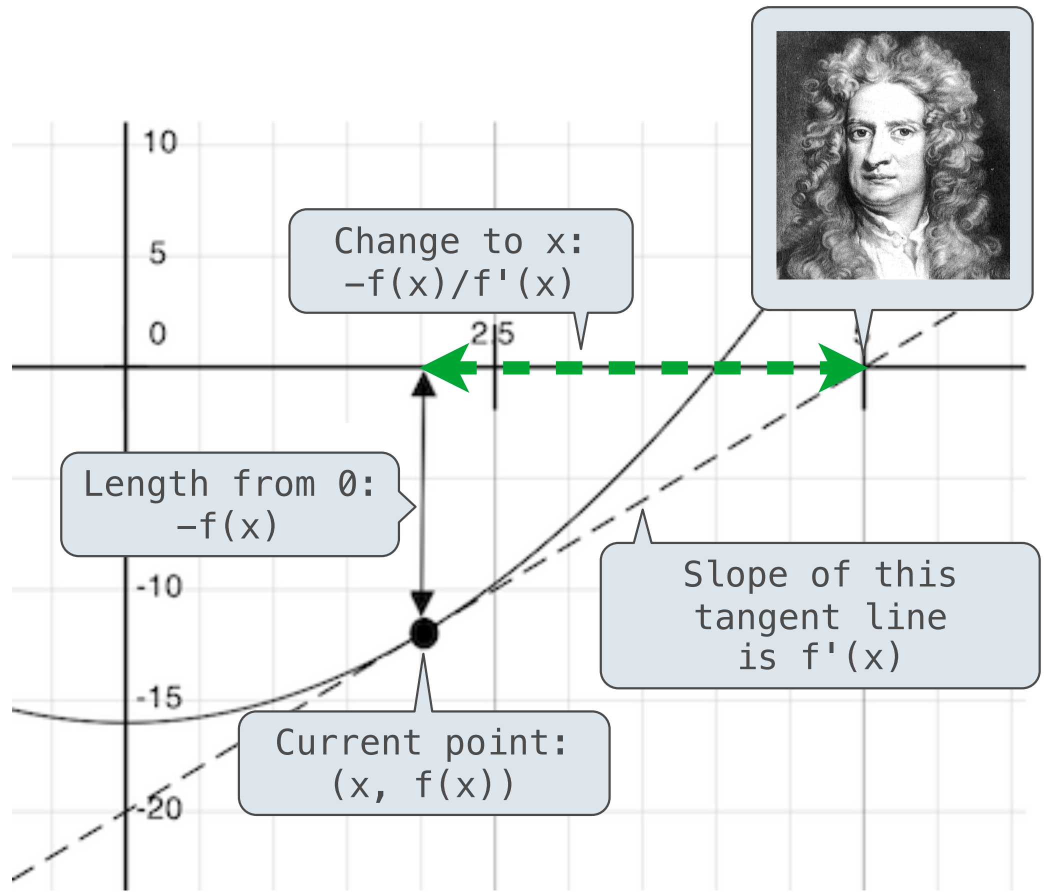
Repeatedly improve x:

Compute the value of f at the guess: f(x)

Compute the derivative of f at the guess: f'(x)

Update guess x to be:

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



Newton's Method

Given a function f and initial guess x,

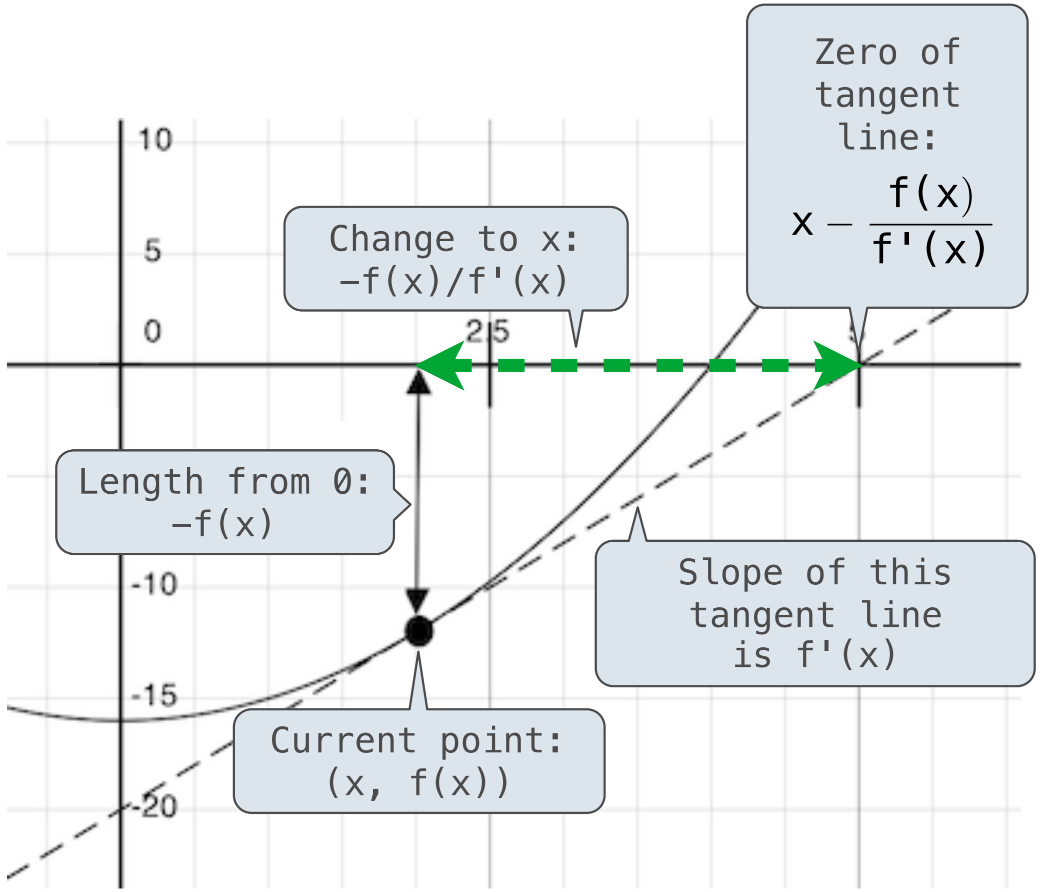
Repeatedly improve x:

Compute the value of f at the guess: f(x)

Compute the derivative of f at the guess: f'(x)

Update guess x to be:

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



Newton's Method

Given a function f and initial guess x,

Repeatedly improve x:

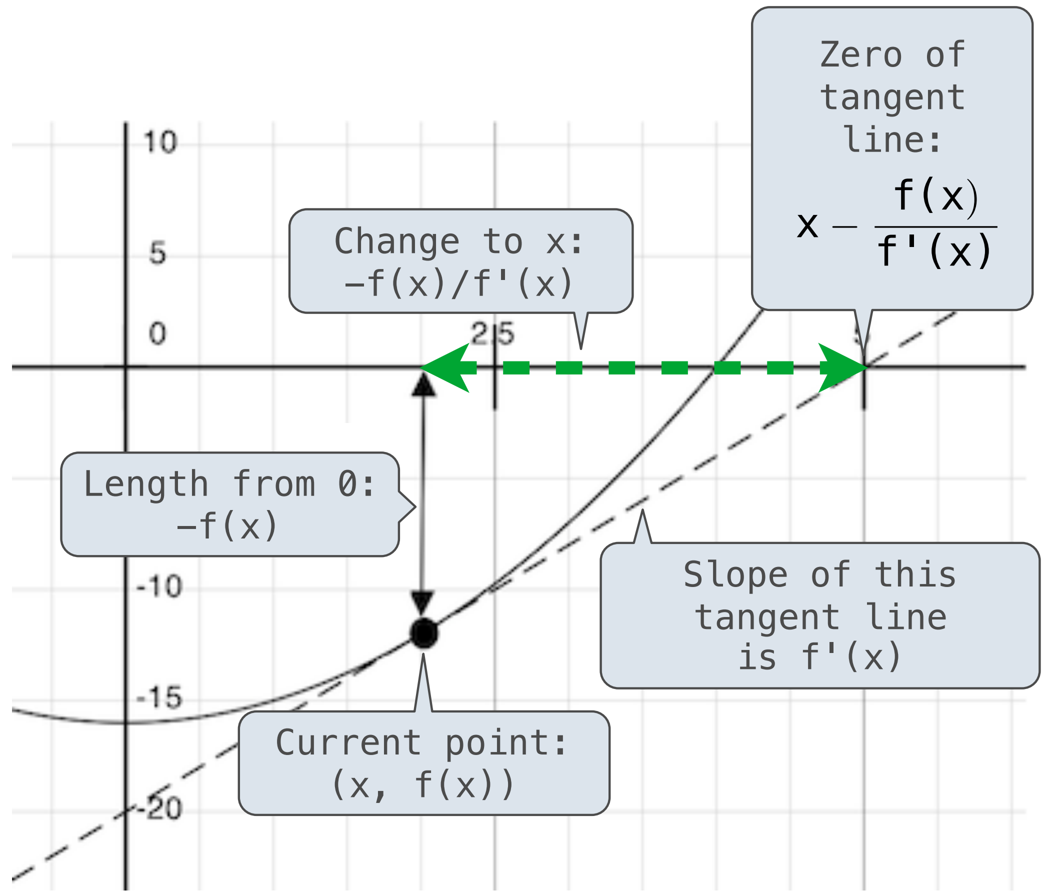
Compute the value of f at the guess: f(x)

Compute the derivative of f at the guess: f'(x)

Update guess x to be:

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

Finish when f(x) = 0 (or close enough)



Newton's Method

Given a function f and initial guess x,

Repeatedly improve x:

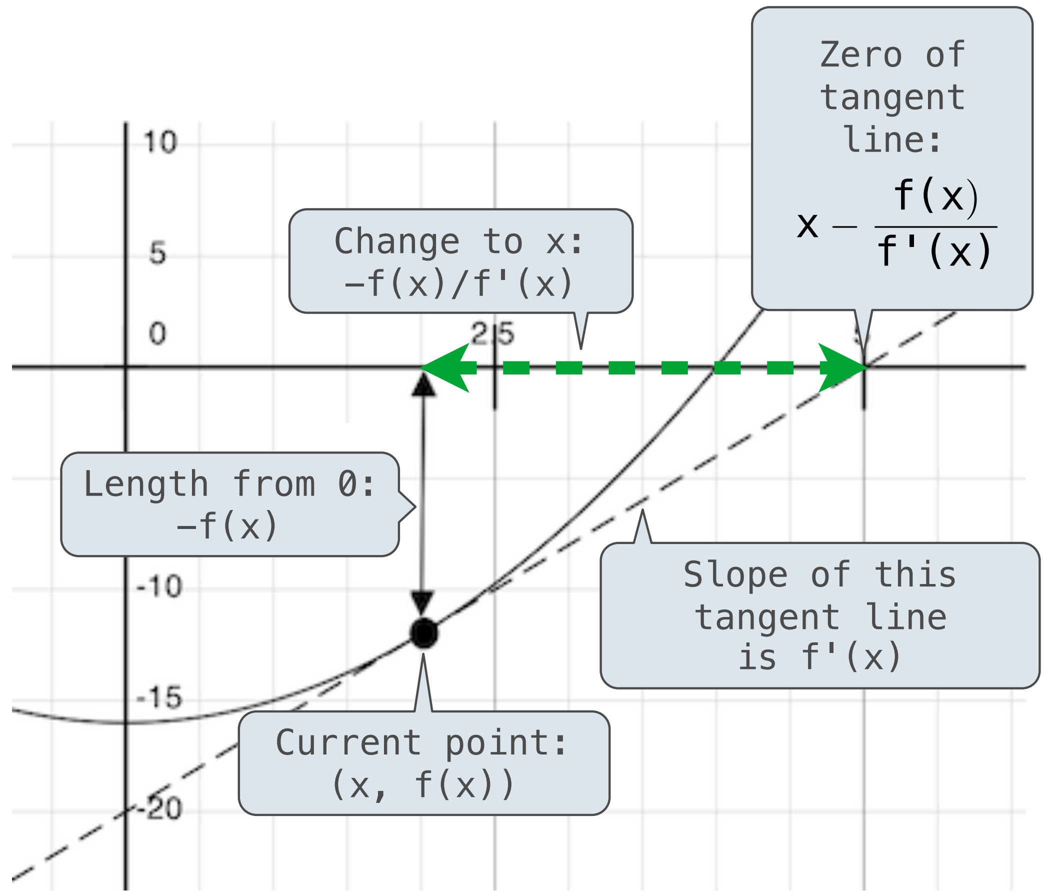
Compute the value of f at the guess: f(x)

Compute the derivative of f at the guess: f'(x)

Update guess x to be:

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

Finish when f(x) = 0 (or close enough)



http://en.wikipedia.org/wiki/File:NewtonIteration_Ani.gif

Using Newton's Method

Using Newton's Method

How to find the square root of 2?

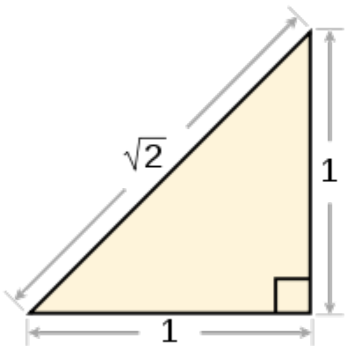
Using Newton's Method

How to find the square root of 2?

```
>>> f  = lambda x: x*x - 2
>>> df = lambda x: 2*x
>>> find_zero(f, df)
1.4142135623730951
```

Using Newton's Method

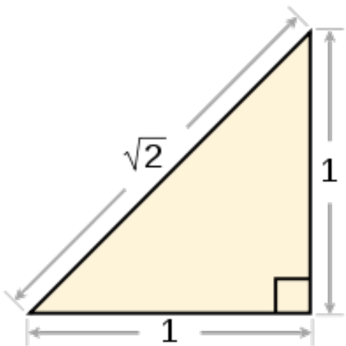
How to find the square root of 2?




```
>>> f  = lambda x: x*x - 2
>>> df = lambda x: 2*x
>>> find_zero(f, df)
1.4142135623730951
```

Using Newton's Method

How to find the square root of 2?



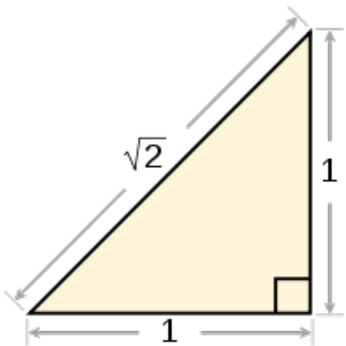

```
>>> f  = lambda x: x*x - 2
>>> df = lambda x: 2*x
>>> find_zero(f, df)
1.4142135623730951
```

$f(x) = x^2 - 2$

$f'(x) = 2x$

Using Newton's Method

How to find the square root of 2?




```
>>> f  = lambda x: x*x - 2
>>> df = lambda x: 2*x
>>> find_zero(f, df)
1.4142135623730951
```

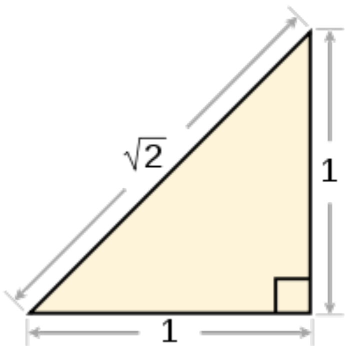
$f(x) = x^2 - 2$

$f'(x) = 2x$

Applies Newton's method

Using Newton's Method

How to find the square root of 2?




```
>>> f  = lambda x: x*x - 2
>>> df = lambda x: 2*x
>>> find_zero(f, df)
1.4142135623730951
```

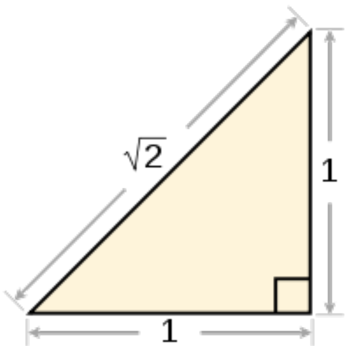
$f(x) = x^2 - 2$
$f'(x) = 2x$

Applies Newton's method

How to find the cube root of 729?

Using Newton's Method

How to find the square root of 2?



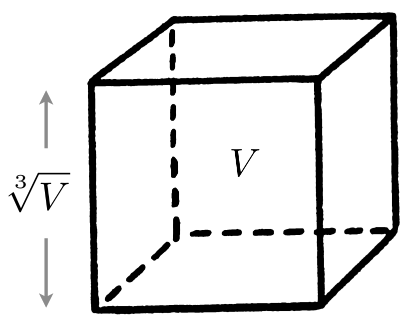

```
>>> f  = lambda x: x*x - 2
>>> df = lambda x: 2*x
>>> find_zero(f, df)
1.4142135623730951
```

$f(x) = x^2 - 2$

$f'(x) = 2x$

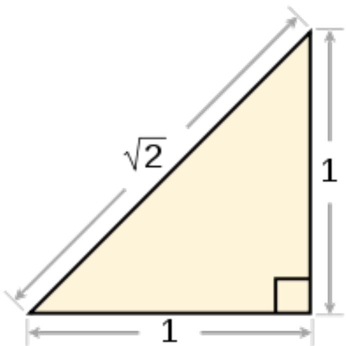
Applies Newton's method

How to find the cube root of 729?



Using Newton's Method

How to find the square root of 2?



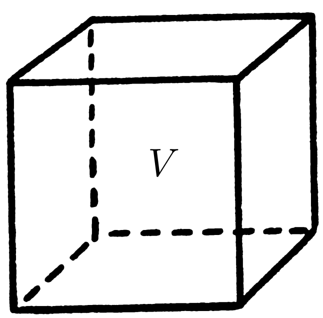

```
>>> f  = lambda x: x*x - 2
>>> df = lambda x: 2*x
>>> find_zero(f, df)
1.4142135623730951
```

$f(x) = x^2 - 2$

$f'(x) = 2x$

Applies Newton's method

How to find the cube root of 729?

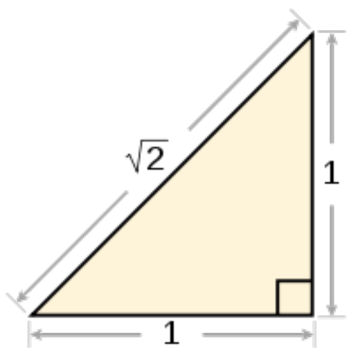


$\sqrt[3]{V}$

```
>>> g  = lambda x: x*x*x - 729
>>> dg = lambda x: 3*x*x
>>> find_zero(g, dg)
9.0
```

Using Newton's Method

How to find the square root of 2?



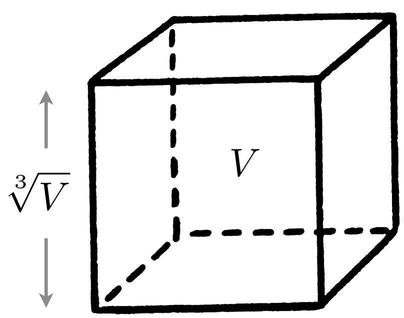

```
>>> f  = lambda x: x*x - 2
>>> df = lambda x: 2*x
>>> find_zero(f, df)
1.4142135623730951
```

$f(x) = x^2 - 2$

$f'(x) = 2x$

Applies Newton's method

How to find the cube root of 729?




```
>>> g  = lambda x: x*x*x - 729
>>> dg = lambda x: 3*x*x
>>> find_zero(g, dg)
9.0
```

$g(x) = x^3 - 729$

$g'(x) = 3x^2$

Iterative Improvement

Special Case: Square Roots

Special Case: Square Roots

How to compute square_root(a)

Idea: Iteratively refine a guess x about the square root of a

Special Case: Square Roots

How to compute square_root(a)

Idea: Iteratively refine a guess x about the square root of a

Update:

Special Case: Square Roots

`How to compute square_root(a)`

Idea: `Iteratively refine a guess x about the square root of a`

Update:

$$x = \frac{x + \frac{a}{x}}{2}$$

Special Case: Square Roots

How to compute square_root(a)

Idea: Iteratively refine a guess x about the square root of a

Update: $$x = \frac{x + \frac{a}{x}}{2}$$

Babylonian Method

Special Case: Square Roots

How to compute square_root(a)

Idea: Iteratively refine a guess x about the square root of a

Update: $$x = \frac{x + \frac{a}{x}}{2}$$ (Demo)

Babylonian Method

Special Case: Square Roots

How to compute square_root(a)

Idea: Iteratively refine a guess x about the square root of a

Update: $$x = \frac{x + \frac{a}{x}}{2}$$

Babylonian Method

(Demo)

Implementation questions:

Special Case: Square Roots

How to compute square_root(a)

Idea: Iteratively refine a guess x about the square root of a

Update: $$x = \frac{x + \frac{a}{x}}{2}$$

Babylonian Method

(Demo)

Implementation questions:

What guess should start the computation?

Special Case: Square Roots

How to compute square_root(a)

Idea: Iteratively refine a guess x about the square root of a

Update: $$x = \frac{x + \frac{a}{x}}{2}$$ (Demo)

Babylonian Method

Implementation questions:

What guess should start the computation?

How do we know when we are finished?

Special Case: Cube Roots

Special Case: Cube Roots

How to compute cube_root(a)

Idea: Iteratively refine a guess x about the cube root of a

Special Case: Cube Roots

How to compute cube_root(a)

Idea: Iteratively refine a guess x about the cube root of a

Update:

Special Case: Cube Roots

How to compute cube_root(a)

Idea: Iteratively refine a guess x about the cube root of a

Update:

$$x = \frac{2 \cdot x + \frac{a}{x^2}}{3}$$

Special Case: Cube Roots

How to compute cube_root(a)

Idea: Iteratively refine a guess x about the cube root of a

Update:

$$x = \frac{2 \cdot x + \frac{a}{x^2}}{3}$$

(Demo)

Special Case: Cube Roots

How to compute cube_root(a)

Idea: Iteratively refine a guess x about the cube root of a

Update: $$x = \frac{2 \cdot x + \frac{a}{x^2}}{3}$$ (Demo)

Implementation questions:

Special Case: Cube Roots

How to compute cube_root(a)

Idea: Iteratively refine a guess x about the cube root of a

Update: $$x = \frac{2 \cdot x + \frac{a}{x^2}}{3}$$ (Demo)

Implementation questions:

What guess should start the computation?

Special Case: Cube Roots

How to compute cube_root(a)

Idea: Iteratively refine a guess x about the cube root of a

Update: $$x = \frac{2 \cdot x + \frac{a}{x^2}}{3}$$ (Demo)

Implementation questions:

What guess should start the computation?

How do we know when we are finished?

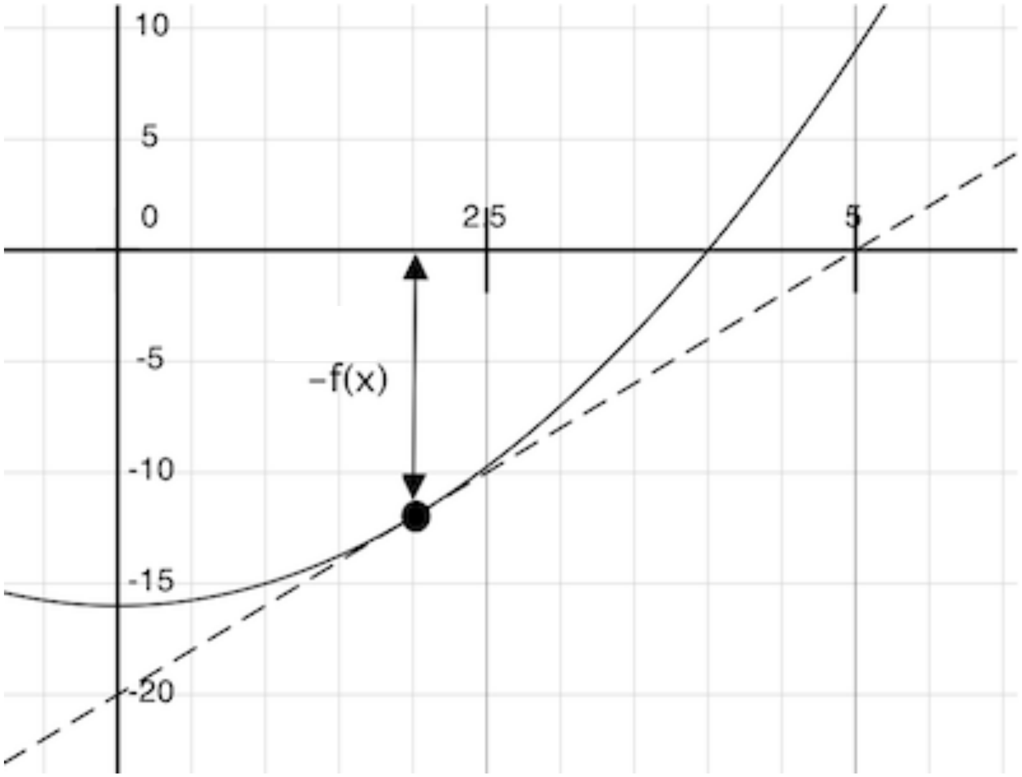
Implementing Newton's Method

(Demo)

Extensions

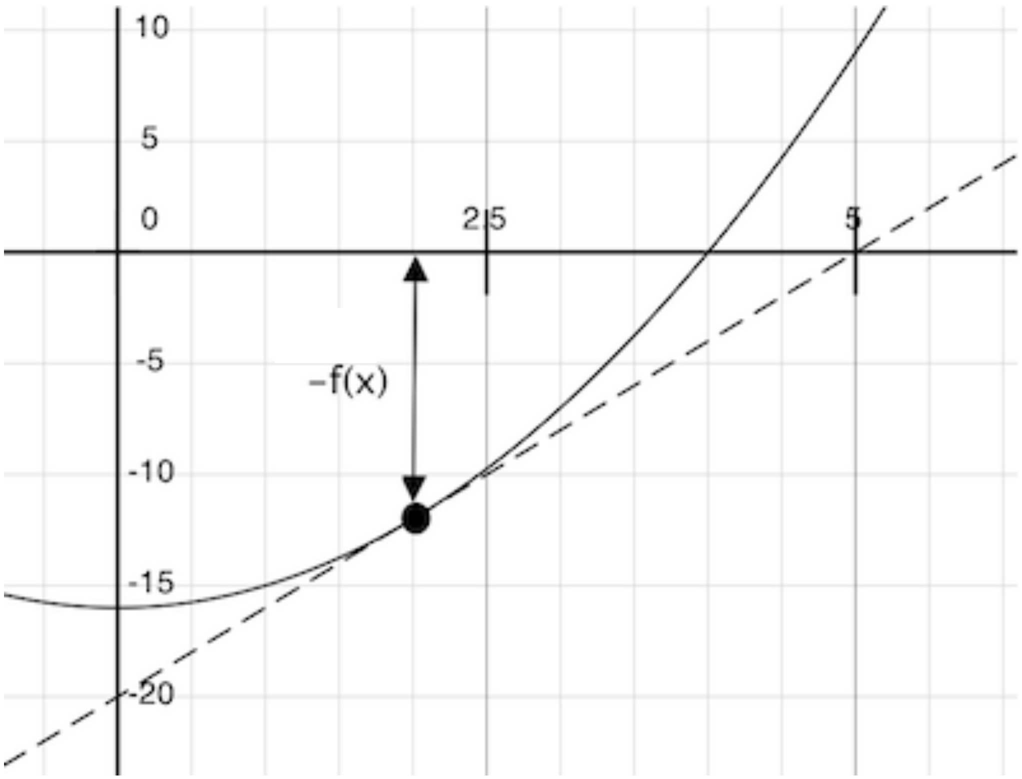
Approximate Differentiation

Approximate Differentiation



Approximate Differentiation

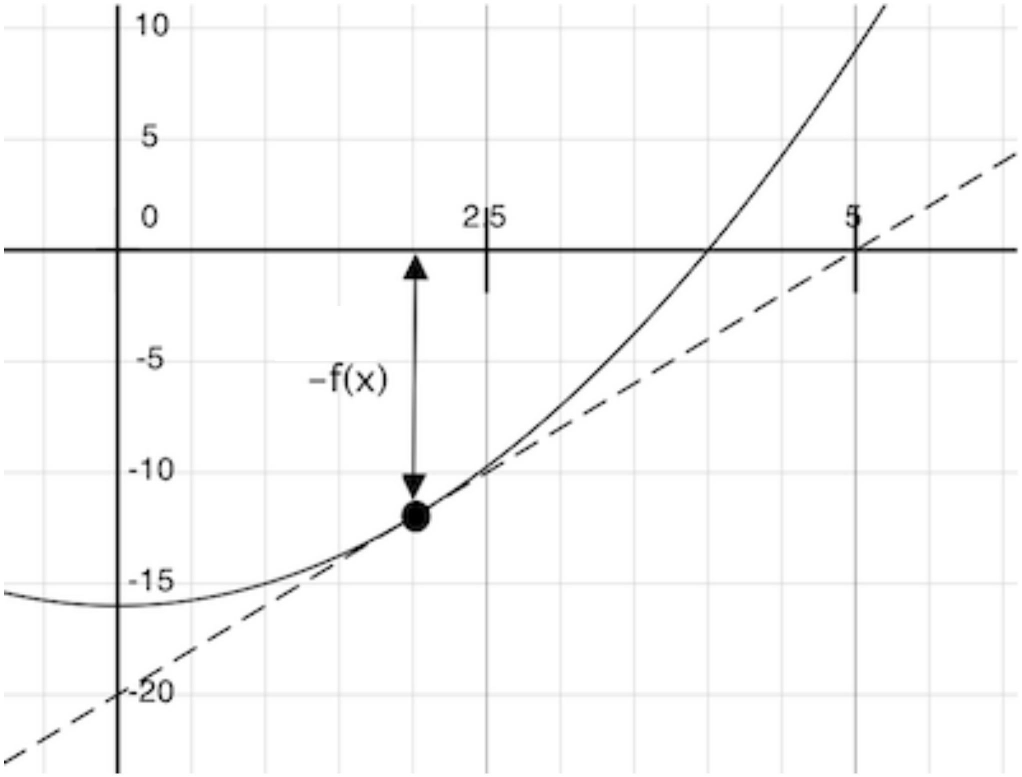
```
Differentiation can be performed
symbolically or numerically
```



Approximate Differentiation

Differentiation can be performed symbolically or numerically

$f(x) = x^2 - 16$

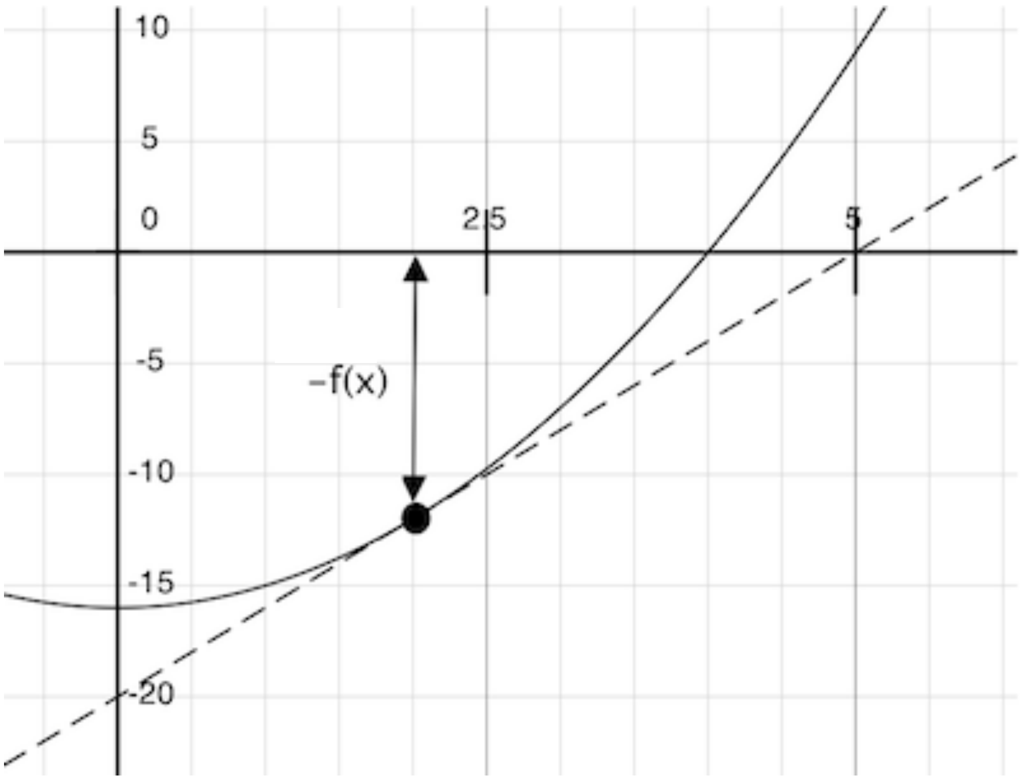


Approximate Differentiation

Differentiation can be performed symbolically or numerically

$f(x) = x^2 - 16$

$f'(x) = 2x$



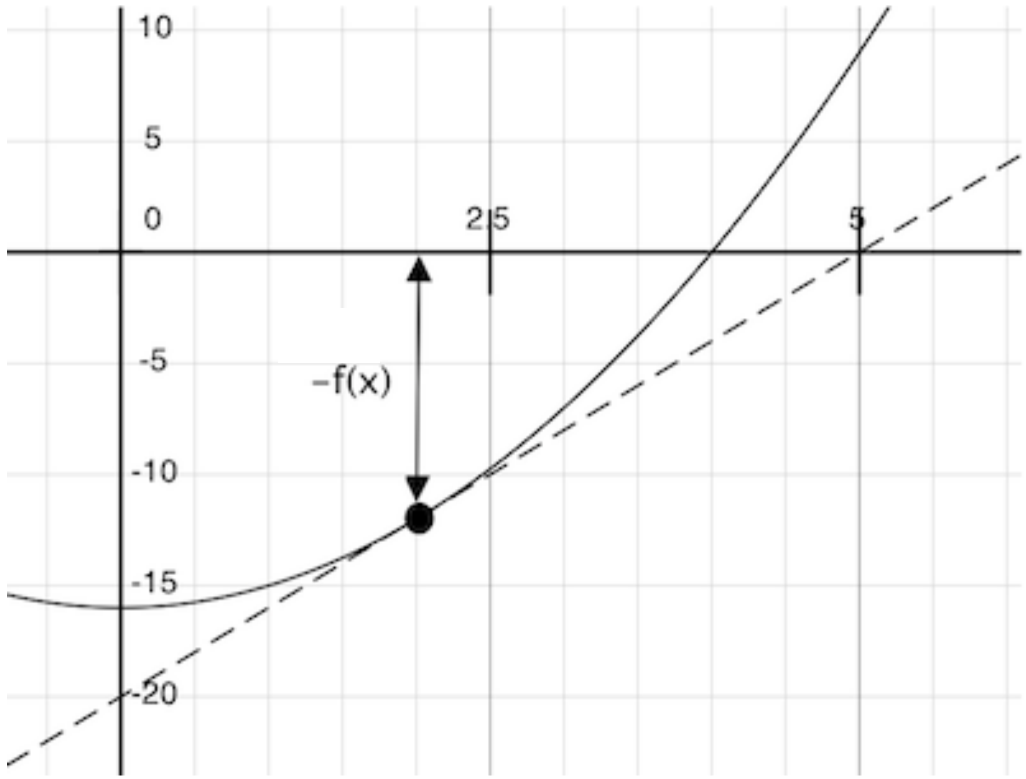
Approximate Differentiation

```
Differentiation can be performed
symbolically or numerically
```

$f(x) = x^2 - 16$

$f'(x) = 2x$

$f'(2) = 4$



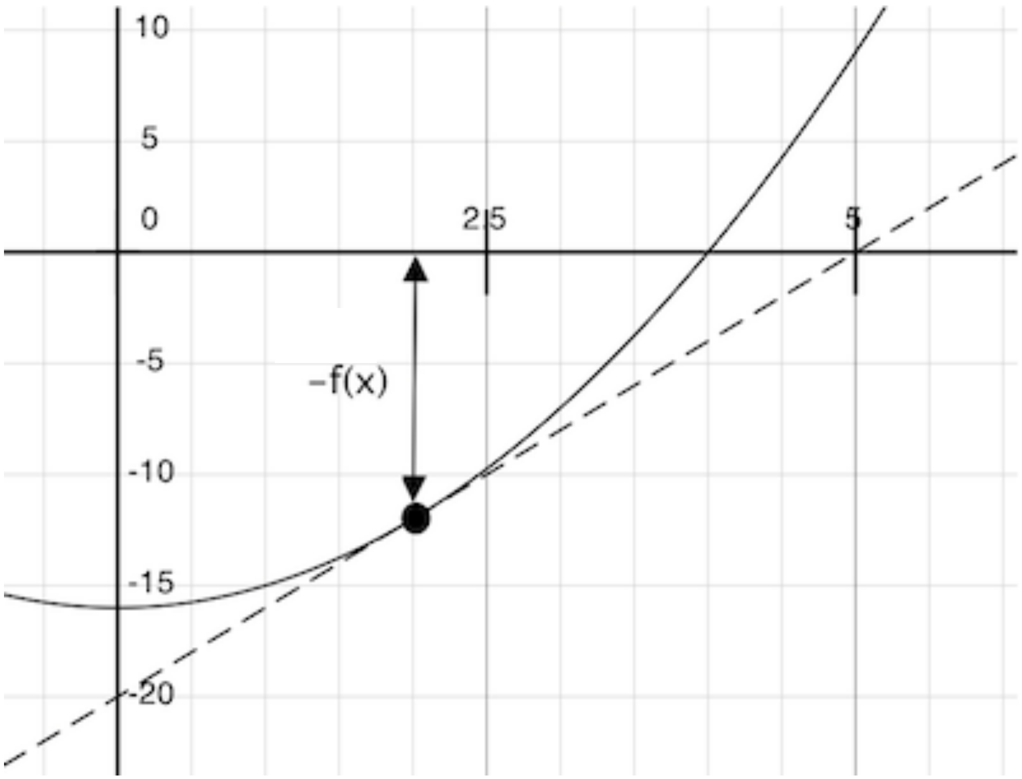
Approximate Differentiation

```
Differentiation can be performed
symbolically or numerically
```

$f(x) = \boxed{x^2} - 16$

$f'(x) = \boxed{2x}$

$f'(2) = 4$



Approximate Differentiation

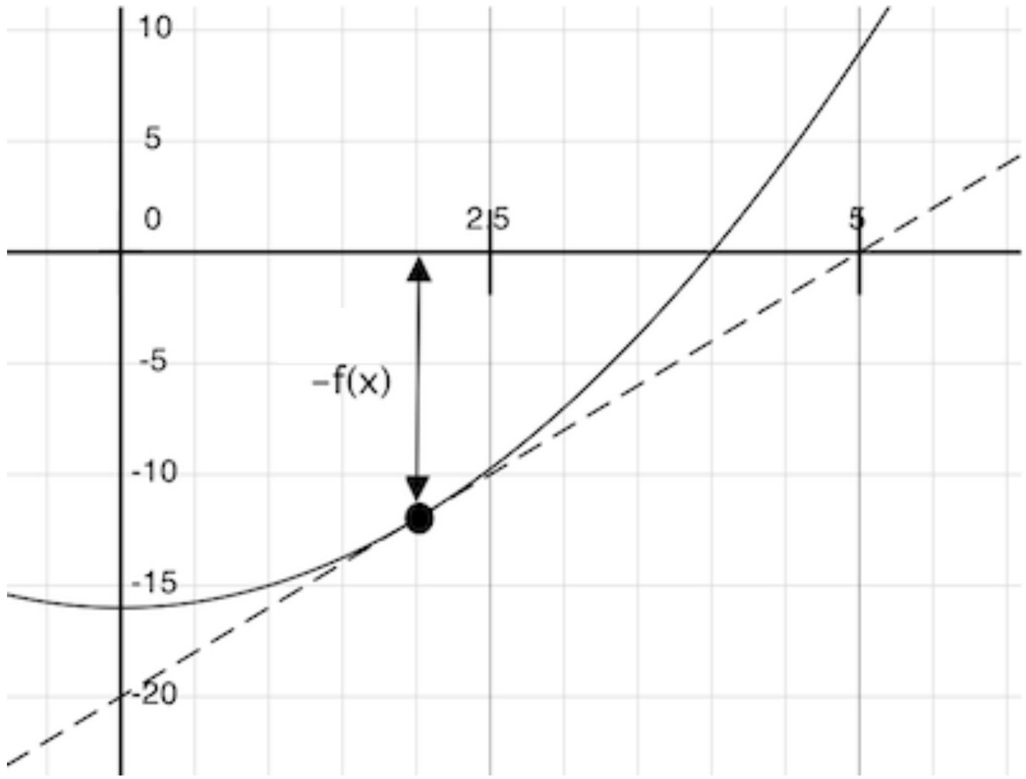
```
Differentiation can be performed
symbolically or numerically
```

f(x) = $\boxed{x^2}$ − 16

f'(x) = $\boxed{2x}$

f'(2) = 4

$$f'(x) = \lim_{a \to 0} \frac{f(x+a) - f(x)}{a}$$



Approximate Differentiation

Differentiation can be performed symbolically or numerically

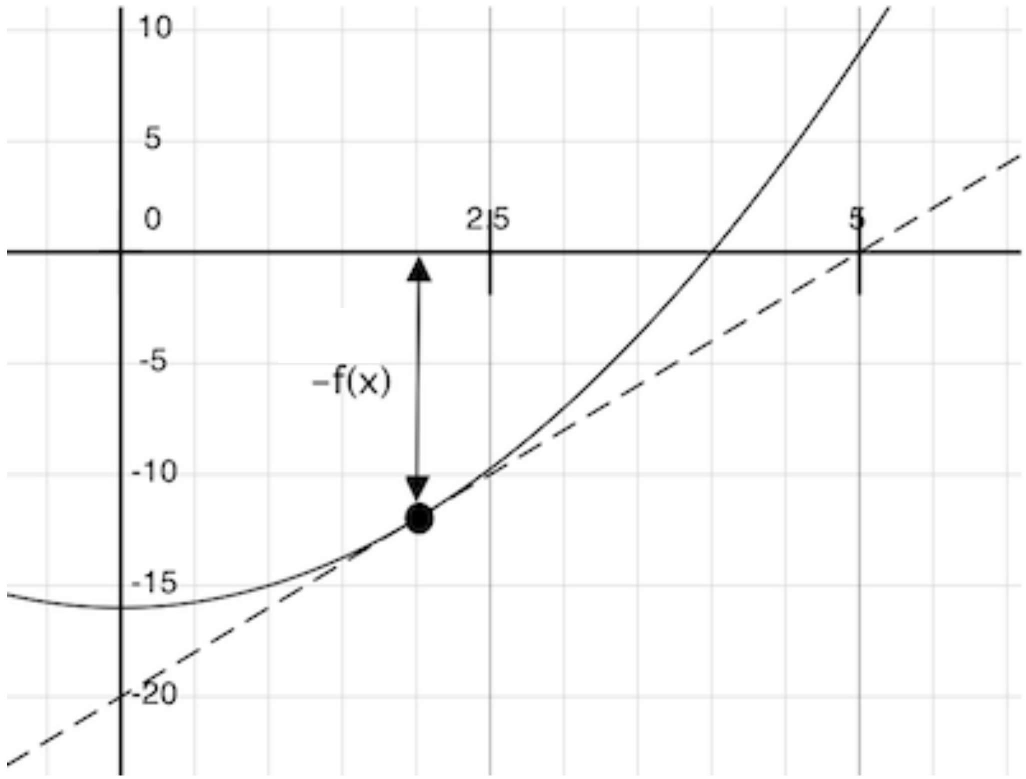
$f(x) = \boxed{x^2} - 16$

$f'(x) = \boxed{2x}$

$f'(2) = 4$

$$f'(x) = \lim_{a \to 0} \frac{f(x+a) - f(x)}{a}$$

$$f'(x) \approx \frac{f(x+a) - f(x)}{a}$$



Approximate Differentiation

Differentiation can be performed symbolically or numerically

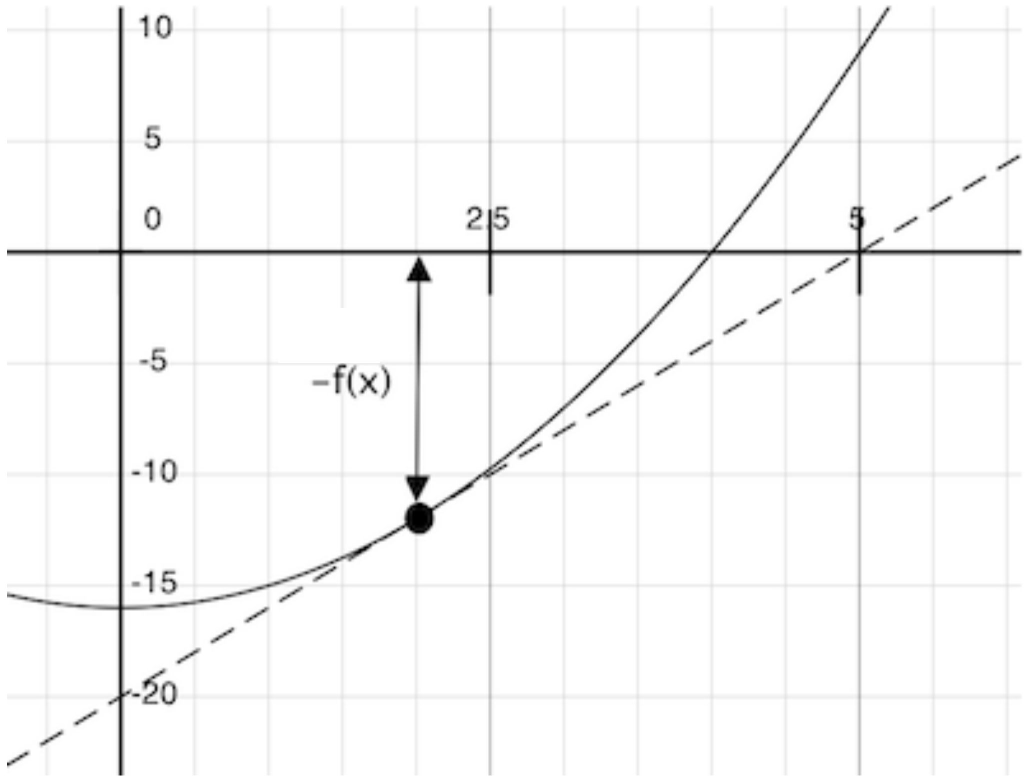
f(x) = $\boxed{x^2}$ – 16

f'(x) = $\boxed{2x}$

f'(2) = 4

$$f'(x) = \lim_{a \to 0} \frac{f(x+a) - f(x)}{a}$$

$$f'(x) \approx \frac{f(x+a) - f(x)}{a}$$ (if a is small)



Approximate Differentiation

Differentiation can be performed symbolically or numerically

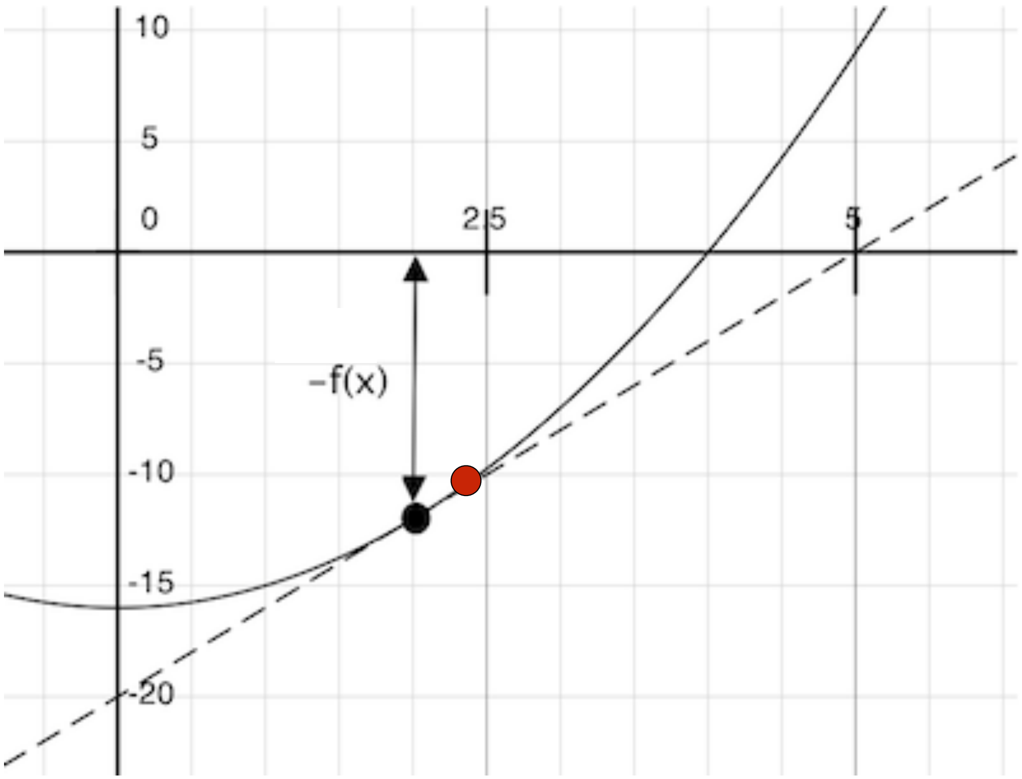
f(x) = $\boxed{x^2}$ – 16

f'(x) = $\boxed{2x}$

f'(2) = 4

$$f'(x) = \lim_{a \to 0} \frac{f(x+a) - f(x)}{a}$$

$$f'(x) \approx \frac{f(x+a) - f(x)}{a}$$ (if a is small)



Approximate Differentiation

Differentiation can be performed symbolically or numerically

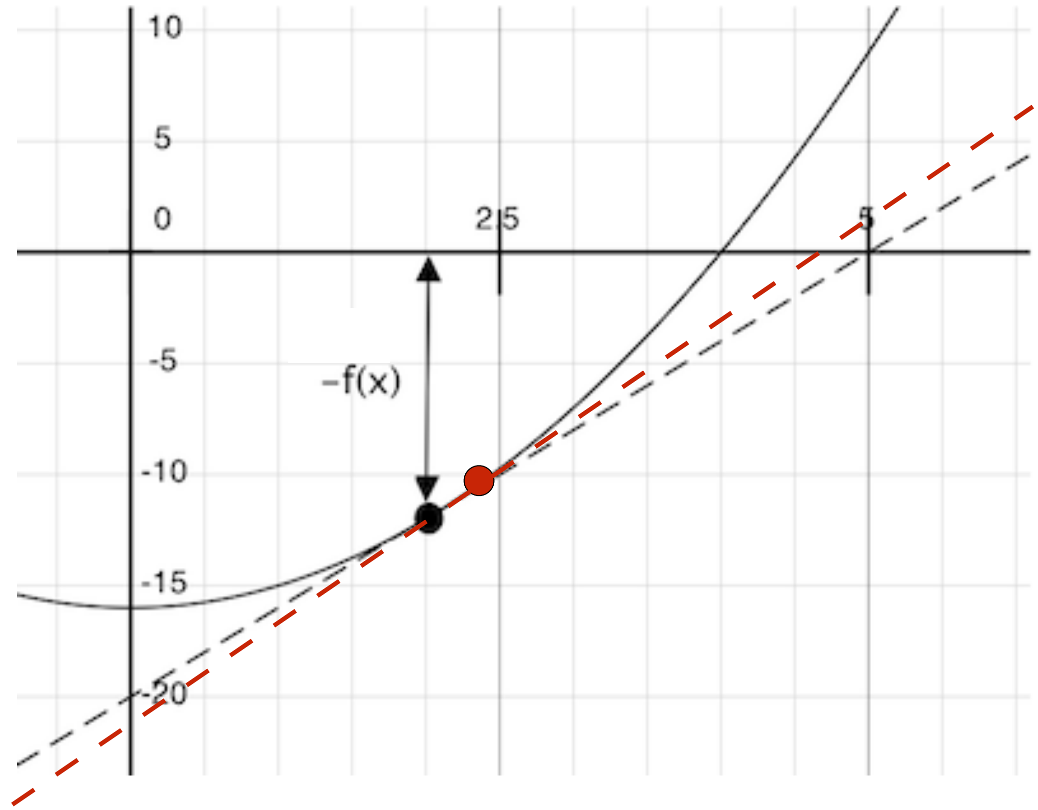
f(x) = $\boxed{x^2}$ – 16

f'(x) = $\boxed{2x}$

f'(2) = 4

$$f'(x) = \lim_{a \to 0} \frac{f(x+a) - f(x)}{a}$$

$$f'(x) \approx \frac{f(x+a) - f(x)}{a} \quad \text{(if } a \text{ is small)}$$



Approximate Differentiation

```
Differentiation can be performed
symbolically or numerically
```

f(x) = x^2 – 16

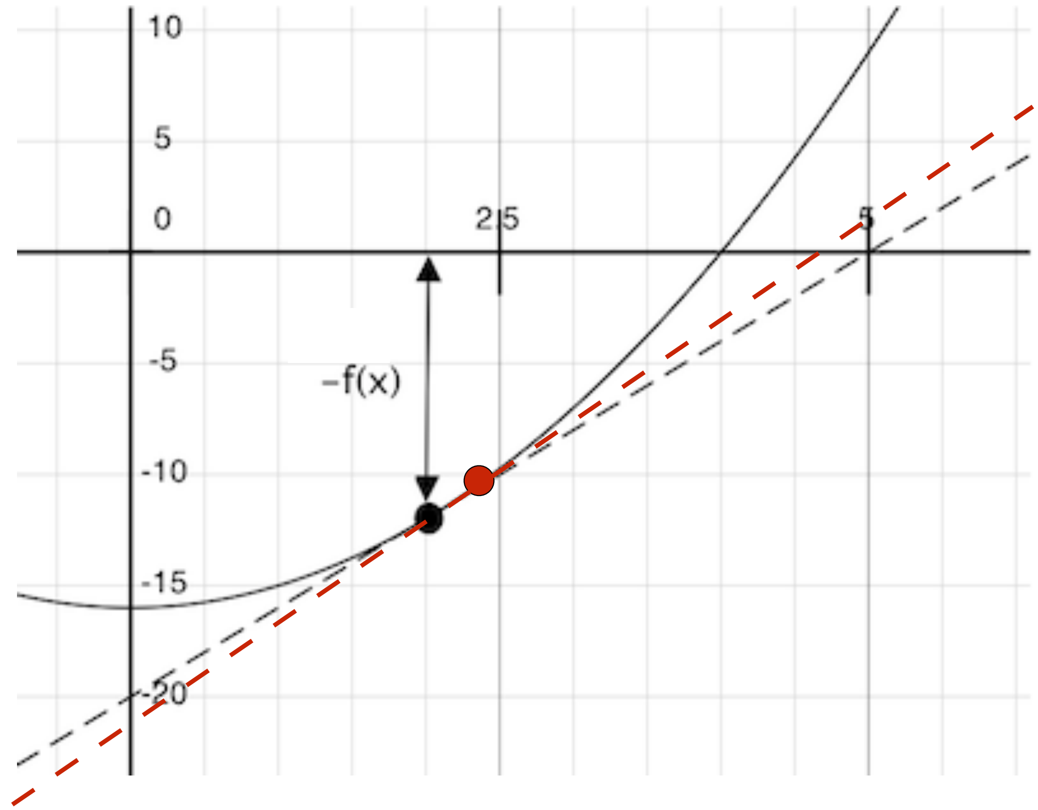
f'(x) = 2x

f'(2) = 4

$$f'(x) = \lim_{a \to 0} \frac{f(x+a) - f(x)}{a}$$

$$f'(x) \approx \frac{f(x+a) - f(x)}{a}$$ (if a is small)

(Demo)



Inverse Function

Inverse Function

The inverse $f^{-1}(y)$ of a differentiable, one-to-one function computes the value x such that $f(x) = y$

Inverse Function

The inverse $f^{-1}(y)$ of a differentiable, one-to-one function computes the value x such that $f(x) = y$

(Demo)