Math 1210-23

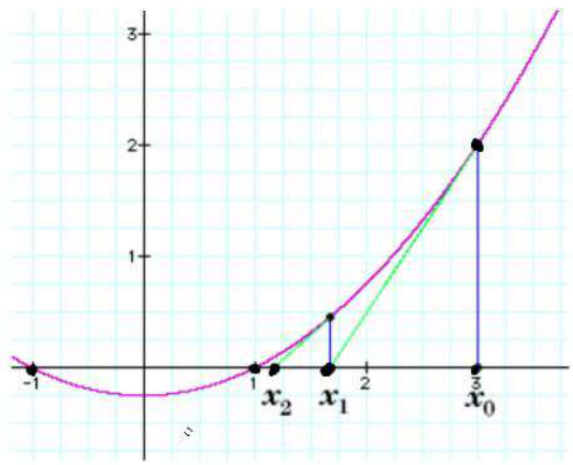
Notes of 3/13/24

Figure 1. Isaac Newton, 1642-1726.

- from `https://breakthrough.neliti.com/isaac-newton/`

3.7 Newton's Method

Newton's Method



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

Figure 2. Newton's Method.

- From https://i.ytimg.com/vi/X03hKJpz8cw/maxresdefault.jpg

- Suppose we want to compute the square root of 2.
- Idea: Apply Newton's Method to

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

```
    |\^/|     Maple 2016 (X86 64 LINUX)
._|\|   |/|_. Copyright (c) Maplesoft, a division of Waterloo Maple Inc. 2016
 \  MAPLE  /  All rights reserved. Maple is a trademark of
 <____ ____>  Waterloo Maple Inc.
      |       Type ? for help.
> restart:
> Digits:=50:
> f:=x^2-2:
> g:=x-f/diff(f,x):
> xn:=1:
> lprint(sqrt(2.0)):
1.4142135623730950488016887242096980785696718753769
>
> for i from 1 to 8 do
>    xn:=evalf(subs(x=xn,g)):
>    lprint(i,xn):
>    end do:
1, 1.5000000000000000000000000000000000000000000000000
2, 1.4166666666666666666666666666666666666666666666667
3, 1.4142156862745098039215686274509803921568627450981
4, 1.4142135623746899106262955788901349101165596221157
5, 1.4142135623730950488016896235025302436149819257762
6, 1.4142135623730950488016887242096980785696718753772
7, 1.4142135623730950488016887242096980785696718753770
8, 1.4142135623730950488016887242096980785696718753770
>
> quit
```

- Same for $f(x) = x^3 - 2$, computing the cube root of 2.

```
    |\^/|     Maple 2016 (X86 64 LINUX)
._|\|   |/|_. Copyright (c) Maplesoft, a division of Waterloo Maple Inc. 2016
 \  MAPLE  /  All rights reserved. Maple is a trademark of
 <____ ____>  Waterloo Maple Inc.
      |       Type ? for help.
> restart:
> Digits:=50:
> f:=x^3-2:
> g:=x-f/diff(f,x):
> xn:=1:
> lprint(evalf(2.0**(1/3))):
1.2599210498948731647672106072782283505702514647015
>
> for i from 1 to 8 do
>    xn:=evalf(subs(x=xn,g)):
>    lprint(i,xn):
>    end do:
1, 1.3333333333333333333333333333333333333333333333333
2, 1.2638888888888888888888888888888888888888888888889
3, 1.2599334934499769664604829439994275159110323945489
4, 1.2599210500177697372930109798984325365370979586 26
5, 1.2599210498948731647791983238845005280761096256985
6, 1.2599210498948731647672106072782283505703655237129
7, 1.2599210498948731647672106072782283505702514647015
8, 1.2599210498948731647672106072782283505702514647015
>
> quit
```

- Here is a different view of Newton's Method. It's given

$$x_{n+1} = g(x_n) \qquad \text{where} \qquad g(x) = x - \frac{f(x)}{f'(x)}.$$

- Clearly

$$f(x) = 0 \qquad \Longleftrightarrow \qquad x = g(x).$$

- x is a **fixed point** of g if $x = g(x)$. Finding x such that $x = g(x)$ is a **Fixed Point Problem.**
- Defining a sequence by

$$x_{n+1} = g(x_n)$$

  is a **fixed point iteration.**
- Newton's Method is a special case of a fixed point iteration.
- (Many) others are possible, for example

$$g(x) = x + f(x) \qquad \text{or} \qquad g(x) = x - f(x).$$

- For example, consider the iteration

$$x_{n+1} = \cos(x_n)$$

- This approaches the solution of

$$x = \cos(x)$$

• It helps to draw a picture.

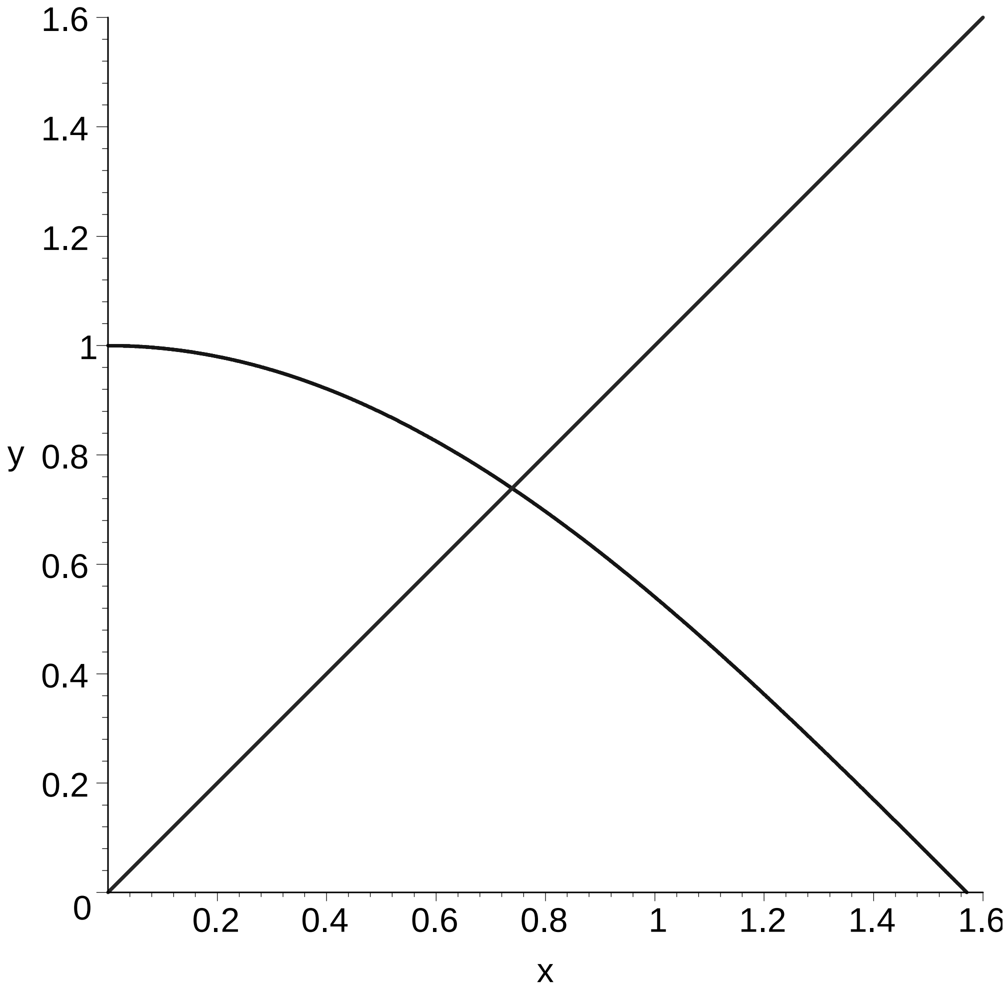


Figure 3. $x_{n+1} = \cos x_n$.

```
    |\^/|     Maple 2021 (X86 64 LINUX)
._|\|   |/|_. Copyright (c) Maplesoft,
a division of Waterloo Maple Inc. 2021
 \  MAPLE  /  All rights reserved. Maple
is a trademark of
 <____ ____>  Waterloo Maple Inc.
      |       Type ? for help.
> restart:
> Digits:=50:
> g:=cos(x);
                                    g :=
cos(x)

> xn:=0:

> for i from 1 to 30 do
>    xn:=evalf(subs(x=xn,g)):
>    lprint(i,xn):
>    end do:
1, 1.
2, .54030230586813971740093660744297660373231042061792
3, .85755321584639341574410627276198979110585842459743
4, .65428979049777914997096647132780849662661549547259
5, .79348035874256559182605423099028400238078491128859
6, .70136877362275652447197052705297773034348004644047
7, .76395968290065422165939741231707253190890369301388
8, .72210242502670773862054697481127480465243355822166
9, .75041776176376046666064455087582779319456260782390
10, .73140404242250985829242687695248252098187449925100
11, .74423735490055686343348005215113012803357849193783
12, .73560474043634733069856973640539275748368371319265
13, .74142508661010920014401902726112299693932404105423
14, .73750689051324281020585960517545015064787542265520
15, .74014733556787575950257467915848696554017582137336
16, .73836920412232321317989257287317426487021175327528
17, .73956720221225608329244451326078900071058843406030
18, .73876031987421132882353185995052513338943228116958
```

```
19, .73930389239690582275122271477664496403887415668468
20, .73893775671534449357804549437232833782398722383022
21, .73918439977149363862420145390534834597576061994902
22, .73901826242741223448907740038962003491532345726144
23, .73913017652967102752054687865439852800610893659115
24, .73905479074691744239130251794798968220690732407241
25, .73910557192653618696278449613712945367504226145999
26, .73907136529894497634106416328487106490885806192561
27, .73909440737909117522169790411143729461367045868569
28, .73907888599499210746774094016752602682504598292696
29, .73908934140339267636385470773874723788487321002156
30, .73908229852240231098247585809094416385928464728212

> quit
memory used=3.1MB, alloc=8.3MB, time=0.04
```

- The following five drawings illustrate different possible behaviors of fixed point iterations.

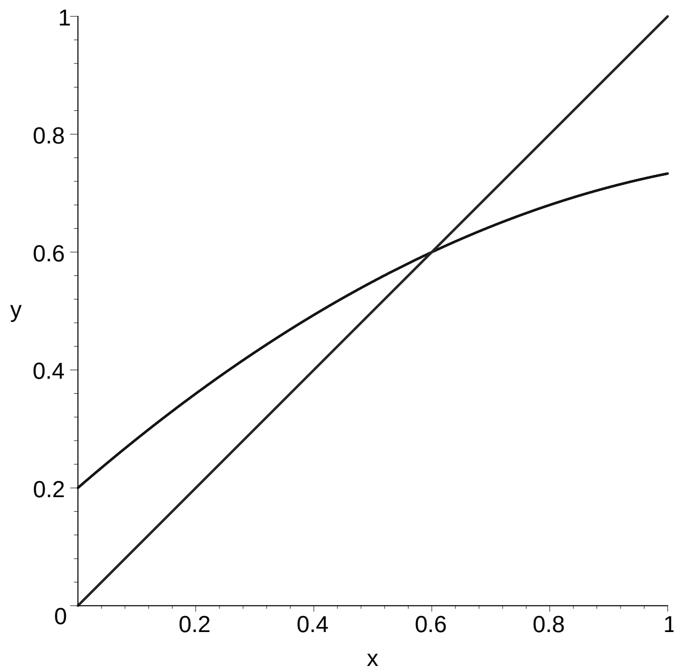


Figure 4. Fixed Point Iteration 1.

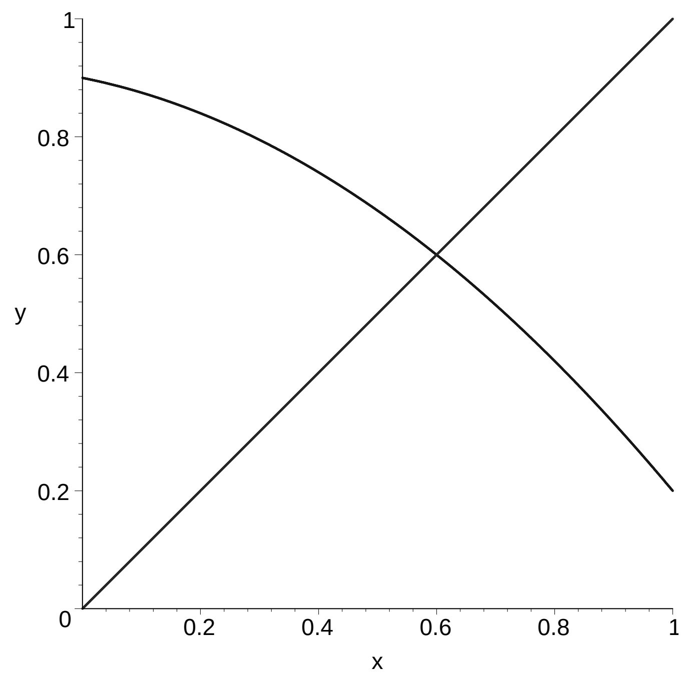


Figure 5. Fixed Point Iteration 2.

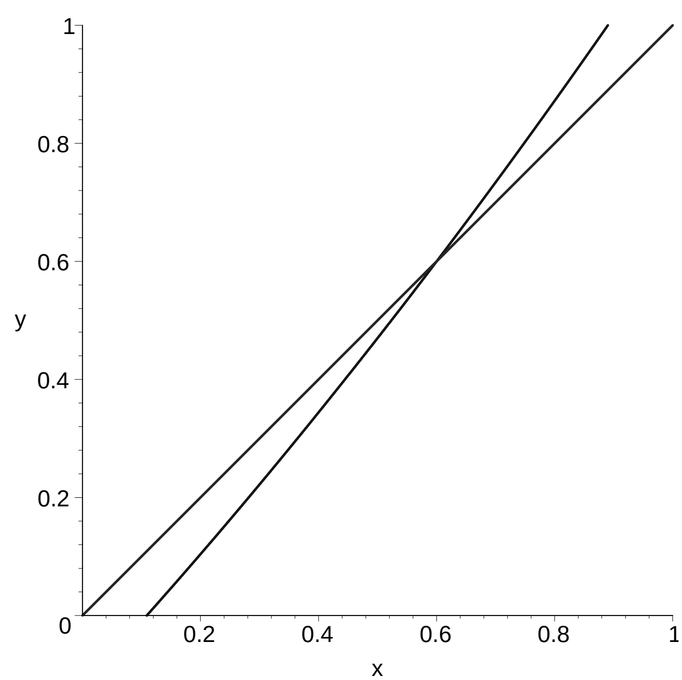


Figure 6. Fixed Point Iteration 3.

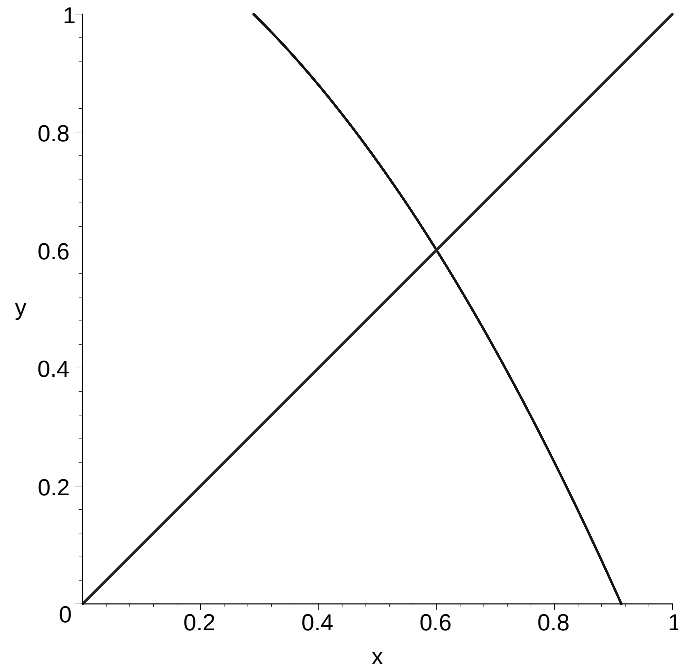


Figure 7. Fixed Point Iteration 4.

- This is an instance of what in mathematics is called **chaos**. If you are interested you can take Math 5470, Chaos and Nonlinear Systems, for more info.

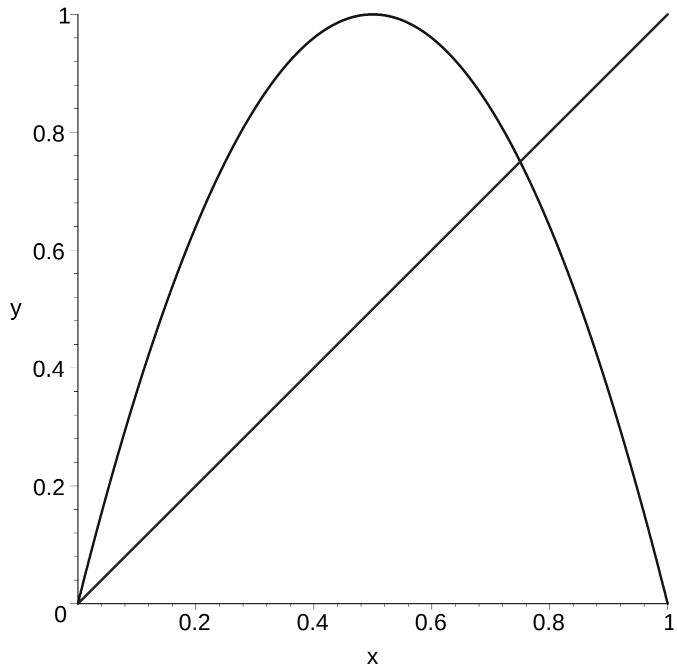


Figure 8. $g(x) = 4x(1-x)$.

- What is going on?

- Why does Newton's Method work so well?

- How do you start?
- It depends on the problem!
- You need to have a good idea of where to look for the solution of your problem.
- For example: Find the roots of

$$f(x) = 10(x-1)(x-2)(x-3) - 1$$

- Commercial: Study Numerical Analysis
- Math 5600, 5610, 5620, 6610, 6620
- Prerequisite: solid Calculus (1210-1220-2210), linear algebra (matrices and vectors) (Math 2270), and some programming experience (the more the better).