Foundations of Applied Math

HW #5 Geometric Similarity, Coding and Euler's Method Due Friday, Oct. 16 by 7 am

**Reminder** You need to turn in a .zipped folder that contains your .tex file, your image files, your python files, your Excel file(s), and the tex file must compile. Rename the .tex file: HW5_YourLastName.tex and call the folder which you will compress: HW4_YourLastName

1. Recall that on Thursday Oct. 8, we ran this line in an example:

```
x=list(map(lambda x: x**3,x))
```

Recall also that in the HW #4 solutions that are posted and that I went over on Friday, Oct. 9, I did this line in the first problem:

```
Pnext=lambda P,Q: P-.1*(Q-500)
```

a) Why does the first example have a map command and the second example does not? Hint. What sort of input does each example take? The first command was to make a list which had a clear connection between the input indeces and the output indeces. The second command, however, was a recursive commmand which depends on the previous values of P and Q, not on the values of the indeces of the array.

b) Run the first example on a list of elements $2, 4, 6, 8$. Paste your python code here and also your output here.

```
# -*- coding: utf-8 -*-
"""
Created on Tue Oct 13 09:32:41 2020

@author: lees19
"""


def main():
    x = list(range(2, 9, 2))
    x = list(map(lambda x: x**3, x))
    print(x)

if(__name__ == "__main__"):
    main()

```

```
In [5]: runfile('C:/Users/lees19/Desktop/homework/Foundations/homeworks/HW5_Lee/number1b.py', wdir='C:/Users/lees19/Desktop/homework/Foundations/homeworks/HW5_Lee')
[8, 64, 216, 512]
```

Python file is also included in number1b.py.

c) Run the second example on $P = 100, Q = 200$. Paste your python code here and also your output here.

```python
# -*- coding: utf-8 -*-
"""
Created on Tue Oct 13 09:32:41 2020

@author: lees19
"""
import numpy as np
import matplotlib.pyplot as plt

def main():
    first = 0
    last = 100
    n = np.linspace(first, last, last+1)
    P = np.zeros(len(n))
    Q = np.zeros(len(n))

    Pnext = lambda P, Q: P-.1*(Q-500)
    Qnext = lambda P, Q: Q+.2*(P-100)

    P[0] = 100
    Q[0] = 200

    for j in range(1, len(n)):
        P[j] = Pnext(P[j-1], Q[j-1])
        Q[j] = Qnext(P[j-1], Q[j-1])

    print(P)
    print(Q)

if(__name__ == "__main__"):
    main()
```

```
In [9]: runfile('C:/Users/lees19/Desktop/homework/Foundations/homeworks/HW5_Lee/number1b.py', wdi
homework/Foundations/homeworks/HW5_Lee')
List for the price:
[ 100.          130.          160.          189.4         217.6
  244.012       268.072       289.25176     307.07008     321.1033648
  330.995248    336.4650639   337.31497485  333.43558451  324.80989468
  311.51549316  293.72489375  271.70398447  245.80857732  216.47909047
  184.23343208  149.65819189  113.39828305   76.14521037   38.62417203
    1.58022948  -34.2361965   -68.08422708  -99.24753373 -127.04915583
 -150.86582726 -170.14151557 -184.39988734 -193.2554288  -196.42297251
 -193.72540764 -185.09938333 -170.59885086 -150.39633072 -124.78183357
  -94.1594098   -59.04134936  -20.04010073   22.14197489   66.72485253
  112.86489067  159.67043176  206.21867503  251.57350967  294.80397081
  335.00296175  371.30587328  402.90872558  429.08546041  449.20402072
  462.74087183  469.29364253  468.59159578  460.50367619  445.04392468
  422.37409965  392.80339612  356.78521061  314.91095716  267.90099951
  216.59282271  161.92662593  104.92857269   46.69198693  -11.64317029
  -68.91216724 -123.94830079 -175.60619099 -222.78511517 -264.45191554
 -299.6630136  -327.58507335 -347.51387283 -358.89097085 -361.3177914
 -354.56679254 -338.58943785 -313.52074731 -279.68026802 -237.56937377
 -187.86487417 -131.40898709  -69.19580253   -2.35443822   67.87084213
  140.14321125  213.05816353  285.17025158  355.02117636  421.16869611
  482.21579233  536.83951463  583.81892108  622.06153725  650.62777498
  668.75278198]
List for the quantity:
[ 200.          200.          206.          218.          235.88
  259.4         288.2024      321.8168      359.667152    401.081168
  445.30184096  491.50089056  538.79390334  586.25689831  632.94401521
  677.90599415  720.20909278  758.95407153  793.29486843  822.45658389
  845.75240198  862.5990884   872.53072678  875.21038339  870.43942546
  858.16425987  838.48030576  811.63306646  778.01622105  738.1667143
  692.75688313  642.58371768  588.55541457  531.6754371   473.02435134
  413.73975684  354.99467531  297.97479864  243.85502847  193.77576233
  148.81939561  109.98751365   78.17924378   54.17122364   38.59961861
   31.94458912   34.51756725   46.45165361   67.69538861   98.01009055
  136.97088471  183.97147706  238.23265171  298.81439683  364.63148891
  434.47229306  507.02046742  580.87919593  654.59751508  726.69825032
  795.70703526  860.18185519  918.74253441  970.09957654 1013.08176797
 1046.66196787 1069.98053241 1082.3658576  1083.35157214 1072.68996952
 1050.36133546 1016.57890202  971.78924186  916.66800366  852.11098063
  779.22059752  699.2879948   613.77098013  524.26820556  432.49001139
  340.22645311  249.3130946   161.59520703   78.89105757    2.95500397
  -64.55887079 -122.13184562 -168.41364304 -202.25280355 -222.72369119
 -229.14952277 -221.12088052 -198.50924781 -161.47519749 -110.47096222
  -46.237223     30.20593547  117.57383839  214.33762261  318.74993006
  428.87548505]
```

2. Do 2.3) Projects #3 on p. 94. Include your python code in your compressed folder and write your analysis here.

   Just as we did in Project 2, we start with some assumptions:

$$E_{spent} \propto S$$
$$E_{gain} \propto B$$
$$E_{spent} = E_{gain}$$
$$\rho = \frac{m}{V} \rightarrow V\rho = m \Rightarrow V \propto W$$

   Then, since $E_{spent} \propto B$, $B \propto E_{spent}$. From there we see that B is proportional to W:

$$B \propto E_{spent} \propto S \propto V^{\frac{2}{3}} \propto W^{\frac{2}{3}}$$
$$B \propto W^{\frac{2}{3}}$$

   Since $B$ is also proportional to $pV_{heart} \propto pV \propto pW$, we also find that $pW \propto W^{\frac{2}{3}}$ Dividing both sides by W, we find that $p \propto W^{-\frac{1}{3}}$. With this information, we use python to find the proportionality constant using our data set:

```
In [13]: runfile('C:/Users/lees19/Desktop/homework/Foundations/homeworks/HW5_Lee/number2.py', wdir='C:/Users/lees19/Desktop/
homework/Foundations/homeworks/HW5_Lee')
                            OLS Regression Results
==============================================================================
Dep. Variable:                  pulse   R-squared:                       0.842
Model:                            OLS   Adj. R-squared:                  0.827
Method:                 Least Squares   F-statistic:                     53.49
Date:                Thu, 15 Oct 2020   Prob (F-statistic):           2.56e-05
Time:                        15:56:43   Log-Likelihood:                -70.954
No. Observations:                  12   AIC:                             145.9
Df Residuals:                      10   BIC:                             146.9
Df Model:                           1
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------
const            77.2104     34.954      2.209      0.052      -0.671     155.092
weight^(-1/3)  1164.9797    159.289      7.314      0.000     810.063    1519.897
==============================================================================
Omnibus:                        2.770   Durbin-Watson:                   1.411
Prob(Omnibus):                  0.250   Jarque-Bera (JB):                1.160
Skew:                           0.759   Prob(JB):                        0.560
Kurtosis:                       3.126   Cond. No.                         5.73
==============================================================================
```

   without forcing our y-intercept, and having a decent $R^2$ value, we fail to reject the null hypothesis that our y-intercept was zero. Therefore, we assume our data is proportional and force the y-intercept to be zero:

```
In [6]: runfile('C:/Users/lees19/.spyder-py3/untitled0.py', wdir='C:/Users/lees19/.spyder-py3')
                                 OLS Regression Results
=======================================================================================
Dep. Variable:                  pulse   R-squared (uncentered):                   0.884
Model:                            OLS   Adj. R-squared (uncentered):              0.873
Method:                 Least Squares   F-statistic:                              83.67
Date:                Thu, 15 Oct 2020   Prob (F-statistic):                    1.79e-06
Time:                        15:30:14   Log-Likelihood:                         -73.338
No. Observations:                  12   AIC:                                      148.7
Df Residuals:                      11   BIC:                                      149.2
Df Model:                           1
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [0.025      0.975]
---------------------------------------------------------------------------------
weight^(-1/3)  1371.6083    149.951      9.147      0.000    1041.569    1701.647
==============================================================================
Omnibus:                        7.768   Durbin-Watson:                   1.233
Prob(Omnibus):                  0.021   Jarque-Bera (JB):                3.540
Skew:                          -0.971   Prob(JB):                        0.170
Kurtosis:                       4.819   Cond. No.                         1.00
==============================================================================
```

From our python code, we find that our coefficient $k = 1371.6083$. Therefore, we find that $p = 1371.6083W^{-\frac{1}{3}}$

However, using this model, and looking at the percent errors for each of the weights, we find that the percent error is quite large for every single weight:

| | Weight | Percent Error |
|---|---|---|
| 0 | 4 | 30.918047 |
| 1 | 25 | 29.987544 |
| 2 | 200 | 44.156732 |
| 3 | 300 | 31.702976 |
| 4 | 2000 | 46.895304 |
| 5 | 5000 | 33.156548 |
| 6 | 30000 | 48.067714 |
| 7 | 50000 | 46.812602 |
| 8 | 70000 | 53.776275 |
| 9 | 450000 | 52.897674 |
| 10 | 500000 | 56.797044 |
| 11 | 3000000 | 80.187081 |

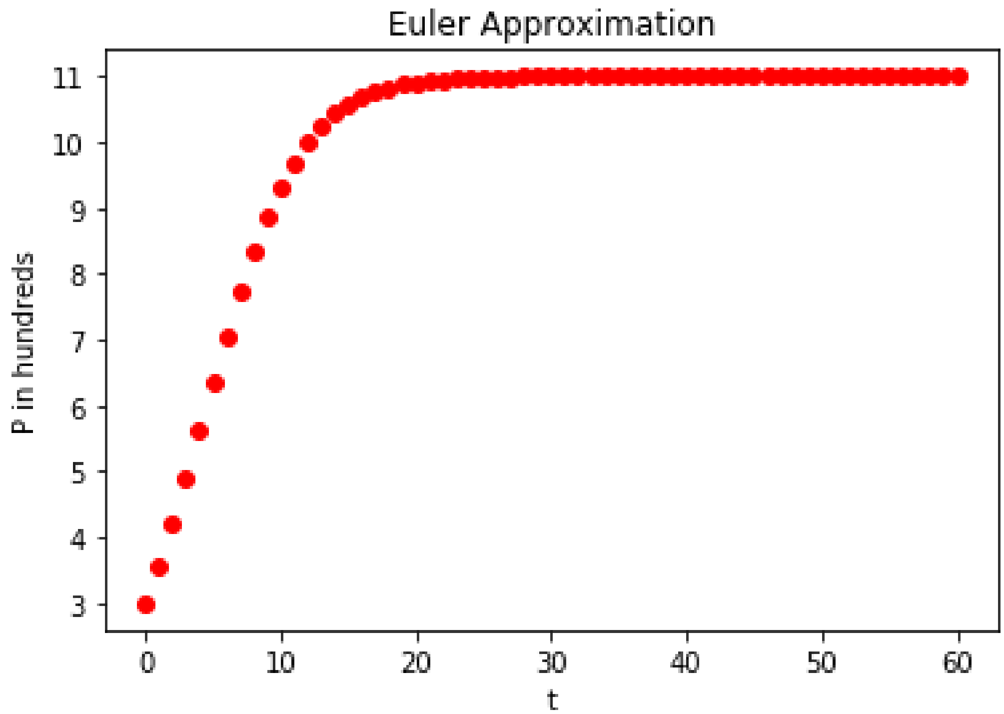
Therefore, we find that though our model is proportional, however the model we found does not fit our data very well.

3. Given
$$\frac{dP}{dt} = 0.24P(11 - P), P(0) = 3$$
where $P$ is measured in 100's, answer the following.

a) Using Python, implement Euler's Method with step size $h = 0.1$ to approximate the solution for $P(2)$ and $P(6)$. Write down the values for each. Also obtain a plot of the solution. Include your python file in your compressed folder and paste (use the includegraphics command) your plot here.

*Solution.*

Using Euler's method and a step size of $h = .1$, we find that the approximation for P(2) and P(6) using python is approximately: $P(2) = 10.902853$ and $P(6) = 11.000000$.

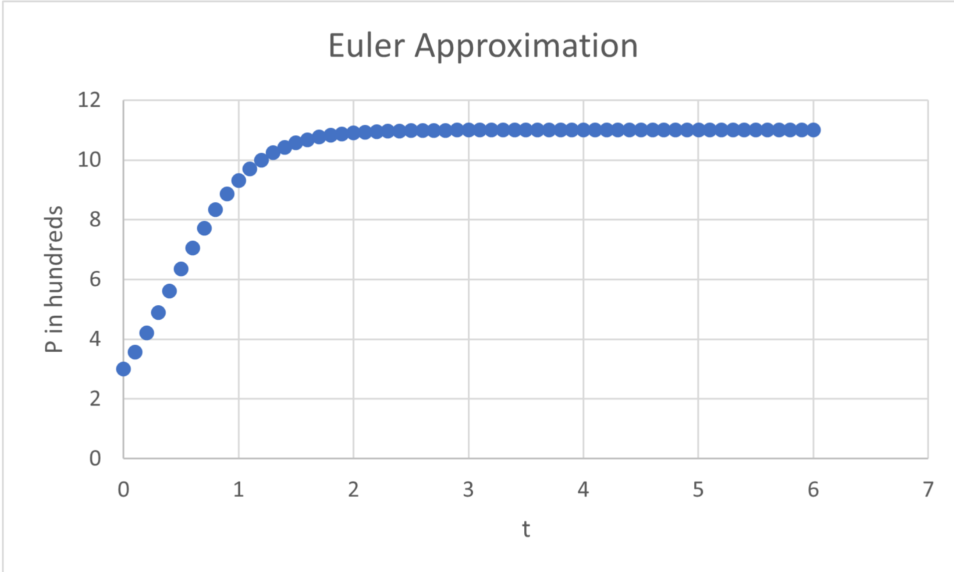



The python file 'number3.py' is included.

b) Do this in Excel. Include your Excel file in your compressed folder. Your work should match part a.

*Solution.*

Using excel, we get the same values for $P(2) = 10.90285$ and $P(6) = 11.00000$ and obtain a very similar looking graph as well:




The excel file 'number3.xlsx' is also included.

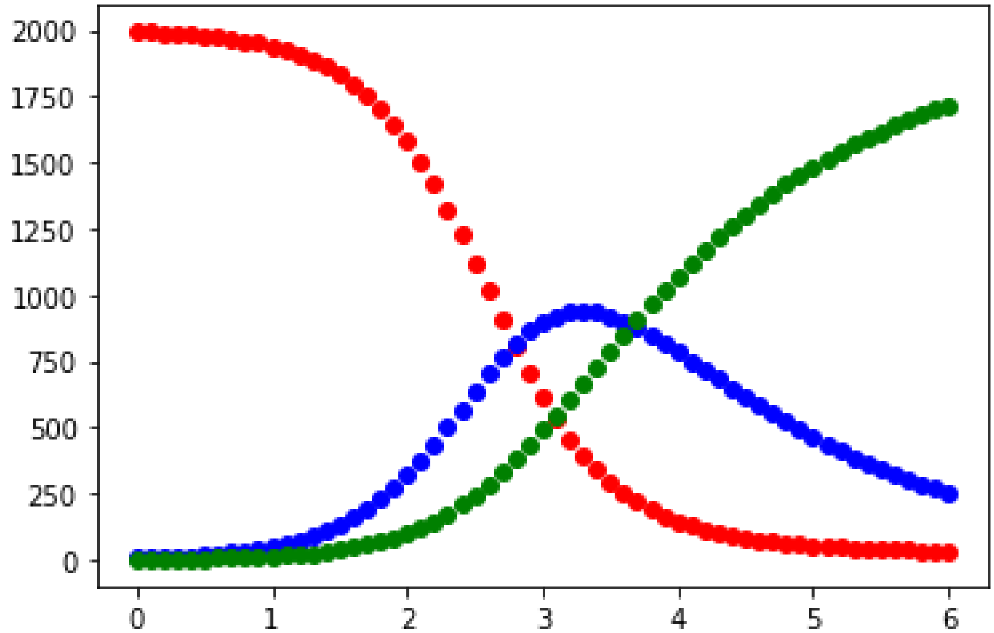
4. Given:

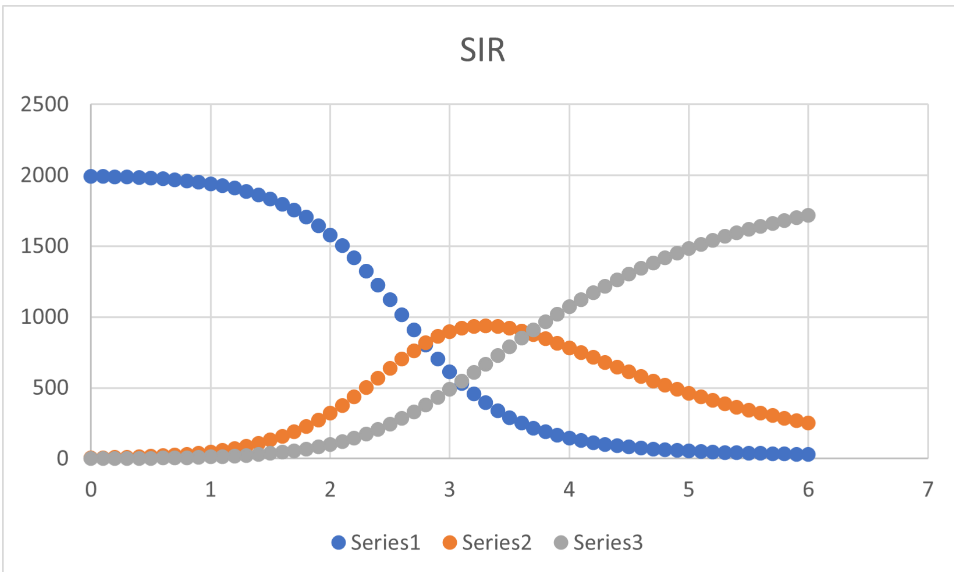
$$\frac{dR}{dt} = 0.65I(t)$$
$$\frac{dI}{dt} = -0.65I(t) + .0015I(t)S(t)$$
$$\frac{dS}{dt} = -.0015I(t)S(t)$$

Suppose that there are 2000 people in the population. Suppose also that initially 6 people are infected.

a) Using Python, implement Euler's Method with step size $h = 0.1$
*Solution.*
Python file 'number4.py' is included.

b) Do this in Excel. Include your Excel file in your compressed folder. Your work should match part a.
*Solution.*
Excel file 'number4.xlsx' is included. Using both excel and python, we see very similar results.




and looking at the actual values we see the actual numbers are also very similar :

```
S        I       R
1994.0  6.0     0.0
1992.2054       7.4046  0.39000000000000007
1989.992677384274       9.136023615726002       0.871299
1987.2655843699183      11.269275095059493      1.4651405350221902
1983.906327986337       13.896028597461942      2.1976434162010574
1979.7710651260793      17.128049598884495      3.1008852750360836
1974.6846225763918      21.101168924644384      4.2142084989635755
1968.4343995073095      25.979816013624813      5.5857844790654605
1960.763464977196       31.962062502852692      7.27447251995107‌4
1951.3629583140594      39.28503510330394       9.352006582636498
1939.8640539665612      48.230412169087415      11.905533864351256
1925.8299875353405      59.12950180931735       15.040510655341938
1908.7489823754793      72.367089351573 18.883928272947564
1888.0293911518856      88.3828197673143        23.58778908079981
1862.9989869428503      107.66834069147427      29.33267236567524
1832.911085497745       130.7577999916336       36.33111451062107
1796.9609723297508      158.20865616017173      44.830371510077256
1754.316755239066       190.56931060044536      55.11393416048842
1704.168915045958       228.33014560452426      67.50093934951737
1645.801944569886       271.8556566163023       82.34239881381144
1578.688859324671       321.2981241814577       100.01301649387109
1502.6043939495937      376.49821148474024      120.89739456566585
1417.7452139159236      436.8850077719022       145.36977831217396
1324.8364695958883      501.3962265867639       173.76730381734762
1225.196268590902       568.4456728636105       206.35805854548727
1120.7276409975311      635.9653317208468       243.30702728162197
1013.8160521011736      701.5391740553493       284.644773843477
907.1313007259067       762.6238791170185       330.24482015707474
803.3613020116981       816.823325688621        379.8153722996809
```

| s | i | r |
|---|---|---|
| 1994 | 6 | 0 |
| 1992.205 | 7.4046 | 0.39 |
| 1989.993 | 9.136024 | 0.871299 |
| 1987.266 | 11.26928 | 1.465141 |
| 1983.906 | 13.89603 | 2.197643 |
| 1979.771 | 17.12805 | 3.100885 |
| 1974.685 | 21.10117 | 4.214208 |
| 1968.434 | 25.97982 | 5.585784 |
| 1960.763 | 31.96206 | 7.274473 |
| 1951.363 | 39.28504 | 9.352007 |
| 1939.864 | 48.23041 | 11.90553 |
| 1925.83 | 59.1295 | 15.04051 |
| 1908.749 | 72.36709 | 18.88393 |
| 1888.029 | 88.38282 | 23.58779 |
| 1862.999 | 107.6683 | 29.33267 |
| 1832.911 | 130.7578 | 36.33111 |
| 1796.961 | 158.2087 | 44.83037 |
| 1754.317 | 190.5693 | 55.11393 |
| 1704.169 | 228.3301 | 67.50094 |
| 1645.802 | 271.8557 | 82.3424 |
| 1578.689 | 321.2981 | 100.013 |
| 1502.604 | 376.4982 | 120.8974 |
| 1417.745 | 436.885 | 145.3698 |
| 1324.836 | 501.3962 | 173.7673 |
| 1225.196 | 568.4457 | 206.3581 |
| 1120.728 | 635.9653 | 243.307 |
| 1013.816 | 701.5392 | 284.6448 |
| 907.1313 | 762.6239 | 330.2448 |
| 803.3613 | 816.8233 | 379.8154 |

5. Suppose squirrels and chipmunks compete for common resources in someone's backyard where they live all year long.

   Let

   - $C(t)$=the population of chipmunks at time $t$
   - $S(t)$= the population of squirrels at time $t$

   Suppose also that:

   - Chipmunks grow at a rate proportional to their population at any time.
   - Chipmunks decrease at a rate proportional to their interactions with each other. This is called intracompetition and occurs because chipmunks are competing with each other for the same food and resources.
   - Squirrels grow at a rate proportional to their population at any time.
   - Squirrels decrease at a rate proportional to their interactions with each other.
   - Squirrels and chipmunks both compete for the same resources too. So this means that for squirrels and chipmunks they decrease at a rate proportional to the number of interactions they have with each other. Assume that the proportionality constant is the same for both.

Write down a differential equation system that models this.

$$\frac{dC}{dt} = k_1 C - k_2 C^2 - k_5 CS \tag{1}$$

$$\frac{dS}{dt} = k_3 S - k_4 S^2 - k_5 CS \tag{2}$$