Untitled

September 4, 2020

Day 5 working lunch exercise

You have 2 hours to complete this excercise. You may work together but each student should turn in their own copy of the exercise. We will grade on effort so please do your best to answer all the questions and work together!

Create a Python notebook to run and document your analysis. You can start with this notebook, or from another existing notebook. When finished, export the notebook and submit it to Blackboard.

1. The half-life of P-32 radioactive isotope is 14.32 days. If you receive a vial of P32 containing 1000 atoms, how many atoms do you have left after 7 days? Plot the radioactive decay function and calculate number of remaining atoms at 7 days.

[45]:

```
from scipy.integrate import solve_ivp
import math
import matplotlib.pyplot as plt
```

[36]:

```
k = math.log(2)/14.32
k
```

[36]:

```
0.04840413272066657
```

[42]:

```
def decay(t, C):
    k=0.04840413272066657
    rate = -k * C
    return rate
```

[53]:

```
C_0=[1000] #initial Condition
tspan = [0, 100] #Time span
output_times = np.linspace(0,100,101)
C_solver = solve_ivp(decay, tspan, C_0, dense_output=True,t_eval=output_times)
C_solver
```

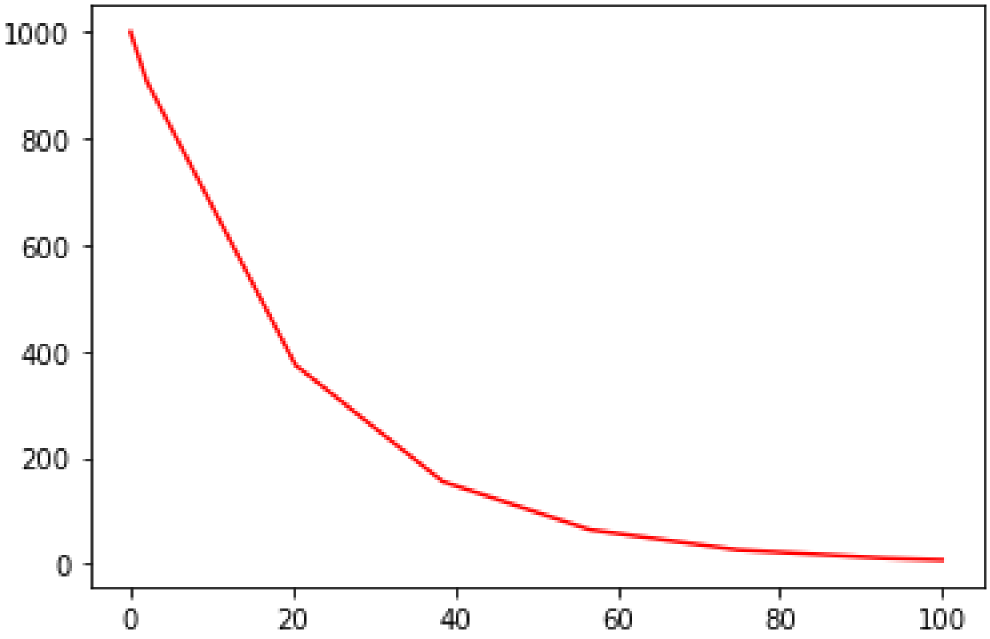
[53]:

```
  message: 'The solver successfully reached the end of the integration
interval.'
     nfev: 50
     njev: 0
      nlu: 0
      sol: <scipy.integrate._ivp.common.OdeSolution object at 0x7fd9a6f35400>
```

```
   status: 0
  success: True
        t: array([  0.,   1.,   2.,   3.,   4.,   5.,   6.,   7.,   8.,   9.,
10.,
        11.,  12.,  13.,  14.,  15.,  16.,  17.,  18.,  19.,  20.,  21.,
        22.,  23.,  24.,  25.,  26.,  27.,  28.,  29.,  30.,  31.,  32.,
        33.,  34.,  35.,  36.,  37.,  38.,  39.,  40.,  41.,  42.,  43.,
        44.,  45.,  46.,  47.,  48.,  49.,  50.,  51.,  52.,  53.,  54.,
        55.,  56.,  57.,  58.,  59.,  60.,  61.,  62.,  63.,  64.,  65.,
        66.,  67.,  68.,  69.,  70.,  71.,  72.,  73.,  74.,  75.,  76.,
        77.,  78.,  79.,  80.,  81.,  82.,  83.,  84.,  85.,  86.,  87.,
        88.,  89.,  90.,  91.,  92.,  93.,  94.,  95.,  96.,  97.,  98.,
        99., 100.])
 t_events: None
        y: array([[1000.        ,  952.7486705 ,  907.73003281,  864.81881284,
         823.90460597,  784.90639786,  747.74489129,  712.34250036,
         678.62335054,  646.51327862,  615.93983275,  586.83227242,
         559.12156844,  532.740403  ,  507.62316961,  483.70597313,
         460.92662975,  439.22466702,  418.54132382,  398.81955039,
         380.00400829,  362.04681364,  344.9229458 ,  328.59980564,
         313.04433114,  298.2241822 ,  284.10774064,  270.6641102 ,
         257.86311653,  245.6753072 ,  234.07195172,  223.0250415 ,
         212.50728987,  202.49213208,  192.9537253 ,  183.86694862,
         175.20740306,  166.95141153,  159.07601888,  151.56027523,
         144.3926048 ,  137.55979418,  131.04806082,  124.84392179,
         118.93419383,  113.30599331,  107.94673624,  102.84413829,
          97.98621478,   93.36128064,   88.95795049,   84.76513857,
          80.77205876,   76.96822461,   73.34344928,   69.8878456 ,
          66.59182605,   63.4463507 ,   60.44616905,   57.58601519,
          54.86014357,   52.26293315,   49.78888736,   47.43263416,
          45.18892599,   43.05263979,   41.01877698,   39.08246348,
          37.23894973,   35.48361064,   33.81194563,   32.21957859,
          30.70225794,   29.25585656,   27.87637187,   26.55995673,
          25.30417566,   24.10694873,   22.96588081,   21.87862852,
          20.84290018,   19.85645585,   18.9171073 ,   18.02271802,
          17.17120322,   16.36052985,   15.58871657,   14.85383374,
          14.15400349,   13.48739962,   12.85224768,   12.24682494,
          11.6694604 ,   11.11853579,   10.59316222,   10.09261006,
           9.61571588,    9.16136403,    8.72848667,    8.31606372,
           7.92312291]])
 y_events: None
```

```
[47]: plt.plot(C_solver.t, C_solver.y[0], 'r-')
```

```
[47]: [<matplotlib.lines.Line2D at 0x7fd9a80c2b38>]
```



```
[59]: C_solver.t[7]
      C_solver.y[0][7]

[59]: 712.3425003602387

[48]: 1000/(2**(7/14.3))

[48]: 712.2667309101179
```

2. You are tracking the growth of two bacterial strains over time. You notice that the two strains grow slightly differently, but what affect does strain have on the doubling time? To answer this, you will need to:

a) Tidy the following dataset and plot growth of the two samples
b) Fit a curve of exponential growth to the dataset
c) Use this curve to estimate growth rate and compare between the two.

```
[138]: import pandas as pd
       import numpy as np
       from scipy.optimize import curve_fit
       df = pd.read_csv("TECANgrowth.csv",header=None)

[139]: df.head()

[139]:      0       1        2        3        4        5        6        7    \
       0    t  0.0000  10.0000  20.0000  30.0000  40.0000  50.0000  60.0000
```

```
1  WT_0  0.2010   0.2165   0.2298   0.2429   0.2606   0.2809   0.3113
2  kd_0  0.1922   0.1970   0.2044   0.2122   0.2185   0.2297   0.2463

         8        9    …        131        132        133        134  \
0  70.0000  80.0000    …  1300.0000  1310.0000  1320.0000  1330.0000
1   0.3467   0.3684    …     0.5284     0.5270     0.5273     0.5290
2   0.2661   0.2945    …     0.4359     0.4368     0.4374     0.4375

         135        136        137        138        139        140
0  1340.0000  1350.0000  1360.0000  1370.0000  1380.0000  1390.0000
1     0.5267     0.5300     0.5287     0.5295     0.5328     0.5276
2     0.4362     0.4411     0.4380     0.4385     0.4374     0.4383

[3 rows x 141 columns]
```

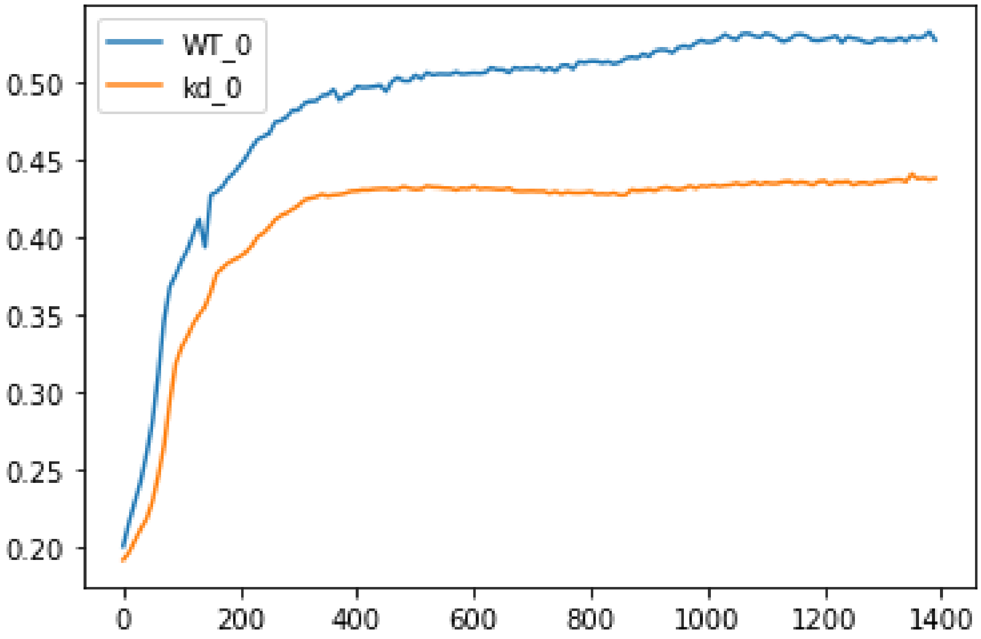
[140]:
```
df = df.T
new_header = df.iloc[0] #grab the first row for the header
df = df[1:] #take the data less the header row
df.columns = new_header #set the header row as the df header
df = df.astype(float)
```

[141]:
```
df
```

[141]:
```
0          t    WT_0    kd_0
1        0.0  0.2010  0.1922
2       10.0  0.2165  0.1970
3       20.0  0.2298  0.2044
4       30.0  0.2429  0.2122
5       40.0  0.2606  0.2185
..       …       …       …
136   1350.0  0.5300  0.4411
137   1360.0  0.5287  0.4380
138   1370.0  0.5295  0.4385
139   1380.0  0.5328  0.4374
140   1390.0  0.5276  0.4383

[140 rows x 3 columns]
```

[142]:
```
plt.plot(df.t,df.WT_0, label="WT_0")
plt.plot(df.t,df.kd_0, label="kd_0")
plt.legend()
```

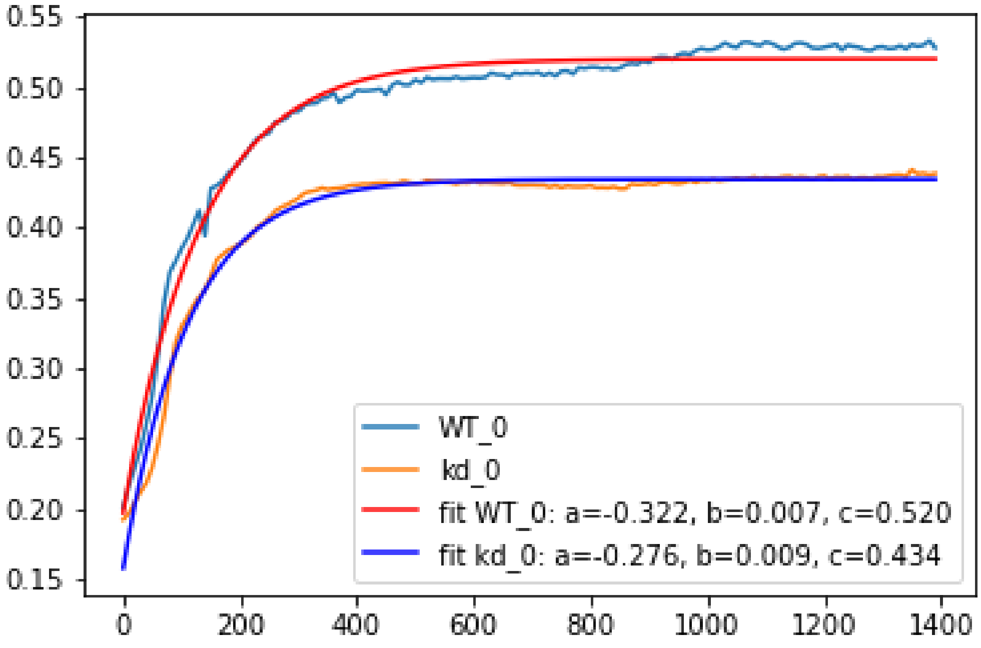
[142]:
```
<matplotlib.legend.Legend at 0x7fd9a3e63da0>
```



Answer:

[146]:
```
def func(x, a, b, c):
    return a * np.exp(-b * x) + c

popt_wt, pcov_wt = curve_fit(func, df.t, df.WT_0, p0=(0, 0.2, 0))
popt_kd, pcov_kd = curve_fit(func, df.t, df.kd_0, p0=(0, 0.2, 0))

plt.plot(df.t,df.WT_0, label="WT_0")
plt.plot(df.t,df.kd_0, label="kd_0")

plt.plot(df.t, func(df.t, *popt_wt), 'r-',label='fit WT_0: a=%5.3f, b=%5.3f,␣
 ↪c=%5.3f' % tuple(popt_wt))

plt.plot(df.t, func(df.t, *popt_kd), 'b-',
         label='fit kd_0: a=%5.3f, b=%5.3f, c=%5.3f' % tuple(popt_kd))
plt.legend()
```

[146]:
```
<matplotlib.legend.Legend at 0x7fd9a47fa4e0>
```



```
[126]: popt_kd
```

```
[126]: array([-2.20641002e-01,  4.79896457e+02,  4.12841007e-01])
```

```
[127]: popt_wt
```

```
[127]: array([-2.88894238e-01,  6.07590688e+02,  4.89894243e-01])
```

```
[133]: df = df.iloc[0:50]
```