Assignment_2_EDA

May 1, 2020

0.1 Assignment 2: EDA with Python

1. Import libraries

- Importing the required libraries to be used for analysis

```
In [1]: import numpy as np
        import matplotlib.pyplot as plt
        import pandas as pd
        import seaborn as sns
```

```
In [2]: %matplotlib inline
        sns.set(color_codes=True)
        plt.style.use("bmh")
        plt.rcParams.update({"figure.figsize" : (6, 4),
                             "axes.facecolor" : "white",
                             "axes.edgecolor":  "black"})
```

2. Load the data to a file

- The data is an output set that is expected to be obtained after running my simulations for thermal hydraulics analyis using porous media approach;
- It uses semicolon as a separator, but can easily be converted to csv by simple macro in vim
  - Convert all semicolons to comma :%s/;/,/g
  - remove the last character from each line :%s/.{1}$//
- Use panda data frame to read the csv file

```
In [3]: data_file = 'https://raw.githubusercontent.com/chirayubatra/che609/master/Nuclear.dat'
        data = pd.read_csv(data_file)
        df = pd.DataFrame(data)
```

```
In [4]: df.shape
```

```
Out[4]: (465, 14)
```

- use df.head(n) to take a look at the data and if it has been imported porperly

```
In [5]: df.head(8)
```

Out[5]:

	time(s)	keff(-)	power(W)	flux0	flux1(m-2s-1)	TFuel_Max	\
0	200.000010	1.00346	10056000.0	4914500.0	5141900.0	1228.0	
1	200.000022	1.00346	10122000.0	4946700.0	5175500.0	1228.0	
2	200.000036	1.00346	10199000.0	4984000.0	5214600.0	1228.0	
3	200.000054	1.00346	10287000.0	5027100.0	5259700.0	1228.0	
4	200.000074	1.00346	10384000.0	5074800.0	5309600.0	1228.0	
5	200.000095	1.00346	10482000.0	5122700.0	5359700.0	1228.0	
6	200.000117	1.00346	10581000.0	5171000.0	5410200.0	1228.0	
7	200.000141	1.00346	10681000.0	5219600.0	5461000.0	1228.0	

	Avg	Min	TCladding_Max	Avg.1	Min.1	TCoolant_Max	Avg.2	Min(K)
0	800.49	600.95	713.6	647.53	600.18	685.92	628.47	600.02
1	800.49	600.95	713.6	647.53	600.18	685.92	628.47	600.02
2	800.49	600.95	713.6	647.53	600.18	685.92	628.47	600.02
3	800.49	600.95	713.6	647.53	600.18	685.92	628.47	600.02
4	800.49	600.95	713.6	647.53	600.18	685.92	628.47	600.02
5	800.49	600.95	713.6	647.53	600.18	685.92	628.47	600.02
6	800.49	600.95	713.6	647.53	600.18	685.92	628.47	600.02
7	800.49	600.95	713.6	647.53	600.18	685.92	628.47	600.02

```
In [6]: df.columns
```

```
Out[6]: Index(['time(s)', 'keff(-)', 'power(W)', 'flux0', 'flux1(m-2s-1)', 'TFuel_Max',
               'Avg', 'Min', 'TCladding_Max', 'Avg.1', 'Min.1', 'TCoolant_Max',
               'Avg.2', 'Min(K)'],
              dtype='object')
```

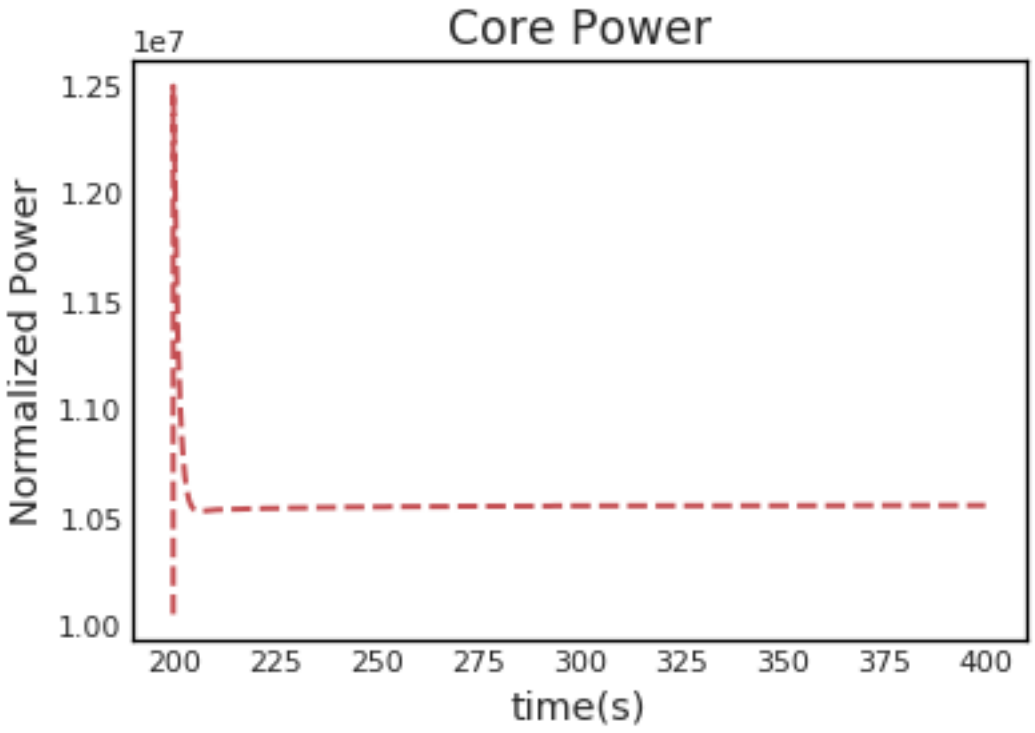
```
In [7]: df.dtypes
```

```
Out[7]: time(s)          float64
        keff(-)          float64
        power(W)         float64
        flux0            float64
        flux1(m-2s-1)    float64
        TFuel_Max        float64
        Avg              float64
        Min              float64
        TCladding_Max    float64
        Avg.1            float64
        Min.1            float64
        TCoolant_Max     float64
        Avg.2            float64
        Min(K)           float64
        dtype: object
```

3. Plot the power value

```
In [8]: plt.plot(df['time(s)'], df['power(W)'], 'r--',label='power')
        plt.ylabel('Normalized Power')
```

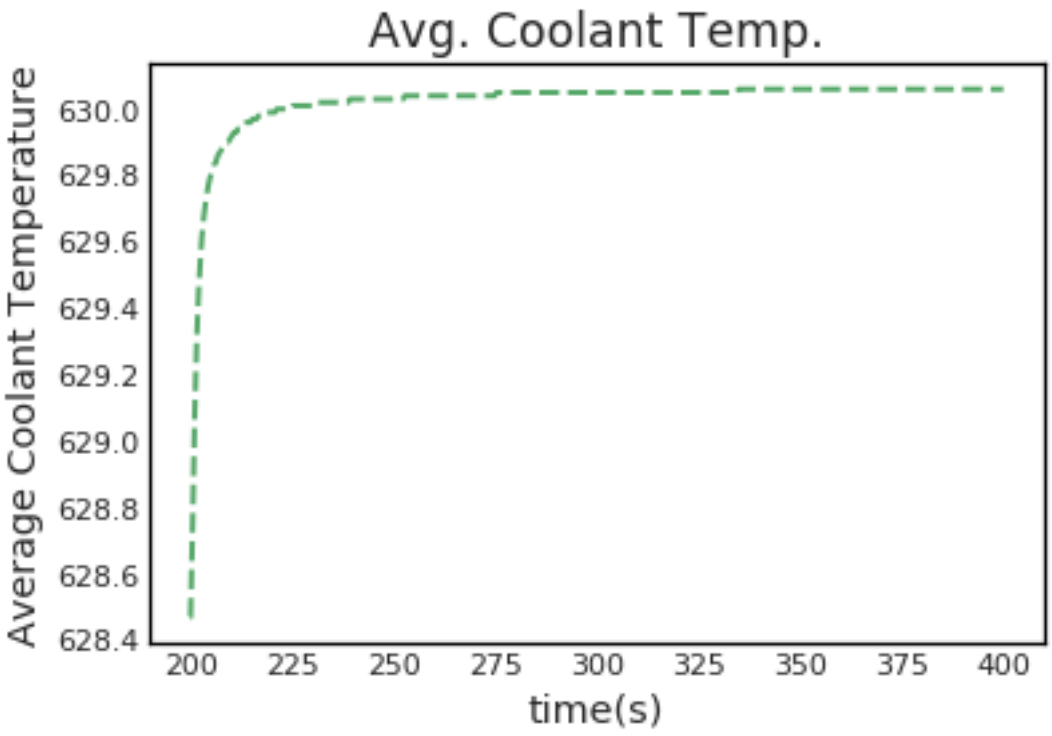
```
plt.xlabel('time(s)')
plt.title('Core Power')
plt.grid(False)
```



The above plot shows normalized power, and it can be observed that the initial peak happened at the start of the transient and then due to SCRAM, power is reduced to decay heat

4. Plot average coolant temperature

```
In [9]: plt.plot(df['time(s)'], df['Avg.2'], 'g--',label='Average Coolant Temperature')
        plt.ylabel('Average Coolant Temperature')
        plt.xlabel('time(s)')
        plt.title('Avg. Coolant Temp.')
        plt.grid(False)
```



There is a slight increase in the coolant temperature due to power increase, and then it stabilizes as the decay heat removal process starts

0.1.1 Summary

This is a simple example that shows how to initialize a jupyter notebook, import the data from existing format in pandas dataframe and then use matplotlib to plot required graphs for analysis

In [ ]: