unswDataImportArray&Weather

The culmination of our previous work comes here where we try to combine the Array data import and the Weather import for 2018. We have some issues to overcome especially now that we're beginning to handle a bit more data than my computer can easily throw around, but that's not to say it can't be done!

Array

To start we're going to repeat the same set of code from unswDataImportArray with some slight changes. Despite the length of repeating this code, it won't be omitted for reproducability purposes.

In [1]:

```
import pandas as pd
import glob
import os

headers = "TimeStamp;ExlSolIrr;IntSolIrr;SMA-h-On;TmpAmb C;TmpMdul C;WindVel km/h;A.Ms.
Amp;A.Ms.Vol;A.Ms.Watt;A1.Ms.Amp"

headers = dict(enumerate(headers.split(';')))
headers = {k: headers[k] for k in (0, 4, 6, 9)} # choosing which headers we want
for item in headers:
    print(item, headers[item])

path = r"C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWData\2018-Arra
y"
all_files = glob.iglob(os.path.join(path, "*csv"))
df1 = pd.concat((pd.read_csv(f, delimiter=";", header=None, skiprows=6, usecols=headers
).assign(filename = os.path.basename(f)) for f in all_files))

df1 = df1.rename(columns = headers)
df1.index = df1['filename'].str.split('.', expand = True)[0] + " " + df1['TimeStamp']
df1 = df1.drop(columns = ['filename'])
df1.index = pd.to_datetime(df1.index)
print(df1)
df1.info()
```

```
0 TimeStamp
4 TmpAmb C
6 WindVel km/h
9 A.Ms.Watt
                     TimeStamp  TmpAmb C  WindVel km/h  A.Ms.Watt
2018-01-01 00:00:00     00:00     22.33         11.86        NaN
2018-01-01 00:05:00     00:05     22.33          4.38        NaN
2018-01-01 00:10:00     00:10     22.31          5.08        NaN
2018-01-01 00:15:00     00:15     22.33          6.46        NaN
2018-01-01 00:20:00     00:20     22.27          4.34        NaN
...                       ...       ...           ...        ...
2018-12-17 09:10:00     09:10     29.47          0.00    3069.00
2018-12-17 09:15:00     09:15     29.29          0.00    3173.80
2018-12-17 09:20:00     09:20     28.93          0.00    3282.80
2018-12-17 09:25:00     09:25     29.81          0.00    3382.80
2018-12-17 09:30:00     09:30     29.25          0.00    3443.75

[87540 rows x 4 columns]
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 87540 entries, 2018-01-01 00:00:00 to 2018-12-17 09:30:00
Data columns (total 4 columns):
TimeStamp       87540 non-null object
TmpAmb C        76721 non-null float64
WindVel km/h    76721 non-null float64
A.Ms.Watt       49920 non-null float64
dtypes: float64(3), object(1)
memory usage: 3.3+ MB
```

Weather

So now that we have our Array data imported, we have to merge the weather data to the Array table without breaking anything. In the process of doing this, I did break everything, many times. There are some critical changes to the code, being first that instead of creating a new weather table for everything, we are simply attempting to put it straight into the TEBT-Array dataframe in order to save memory.

The key lines to doing this is the "firstMerge" on df1.merge, and all subsequent df1.update(df2). These functions were quite difficult to find as merge, combine_first, and update all do small variations on the same idea. When attempting to use only df1.merge I resulted with a 1.4GB+ dataframe and about 10 minutes of computation time, only to realise instead of having 7 comlumns as a result that I had over 2000. Merge forces new columns to be made from the right_merge into the left_merge, so every merge was adding 3-4 columns. df1.update overcomes this by replacing the NaN's in already existing columns from columns of the same name in df2, overcoming the issue.

We also see that this time, many files didn't make the cut for the dataframe, and failed in the import, but half of these are 0KB files with no header, and others are corrupted in another way that I didn't investiage, so there is little issue.

In [2]:

```
headers = "Timestamp,TZ,01Tpvtg_in (oC),02Tpvtg_out (oC),03Ttankg_in (oC),04Ttankg_out
 (oC),05Ttankg (oC),07Tpvtug_in (oC),08Tpvtug_out (oC),09Ttankug_in (oC),10Ttankug_out
 (oC),11Ttankug (oC),06Flowg,12Flowug,(IR02)T (oC),(SPN1)G_ht (W/m2),(SPN1)G_hd (W/m2),
(SR12)G_tilt (W/m2),(IR02)U/S (W/m2)"
headers = dict(enumerate(headers.split(',')))
headers = {k: headers[k] for k in (0, 15, 16)}
for item in headers:
    print(item, headers[item])
path = r"C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWData\2018-Weat
her"
all_files = glob.iglob(os.path.join(path, "*csv"))

firstMerge = True

for f in all_files:
    try:
        df2 = pd.read_csv(f, header=0, usecols=headers)
        df2 = df2.rename(columns = {'(SPN1)G_ht (W/m2)':'GHI', '(SPN1)G_hd (W/m2)':'DH
I'})
        df2['Timestamp'] = pd.to_datetime(df2['Timestamp'].map(lambda x: '.'.join(str(x
).split('.')[:-1])))
        df2.index = df2['Timestamp']
        if firstMerge:
            df1 = df1.merge(df2, how='left', left_index=True, right_index=True, validat
e="one_to_many")
            firstMerge = False
        #df1.combine_first(df2)
        df1.update(df2)
        #break #remove to attempt all files
    except:
        print("failed: " + f)

df1.info()
print(df1)
```

```
0 Timestamp
15 (SPN1)G_ht (W/m2)
16 (SPN1)G_hd (W/m2)
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\000_20180124T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\000_20180307T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\000_20180308T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\000_20180404T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\000_20180502T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\000_20180503T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\000_20180515T000000.CSV

C:\Users\Clairvoyant Cabbage\Documents\PythonProject\venv\lib\site-package
s\IPython\core\interactiveshell.py:3058: DtypeWarning: Columns (15,16) hav
e mixed types. Specify dtype option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

```
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\000_20180605T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\001_20180309T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\001_20180405T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\001_20180504T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\002_20180310T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\002_20180406T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\003_20180407T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\004_20180408T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\004_20180512T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\005_20180123T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\005_20180409T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\005_20180513T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\006_20180410T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\007_20180411T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\008_20180412T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\009_20180413T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\010_20180414T000000.CSV
failed: C:\Users\Clairvoyant Cabbage\Documents\PythonProject\Thesis\UNSWDa
ta\2018-Weather\235_20190126T000000.CSV
<class 'pandas.core.frame.DataFrame'>
DatetimeIndex: 87540 entries, 2018-01-01 00:00:00 to 2018-12-17 09:30:00
Data columns (total 7 columns):
TimeStamp       87540 non-null object
TmpAmb C        76721 non-null float64
WindVel km/h    76721 non-null float64
A.Ms.Watt       49920 non-null float64
Timestamp       75368 non-null datetime64[ns]
GHI             75367 non-null float64
DHI             75367 non-null object
dtypes: datetime64[ns](1), float64(4), object(2)
memory usage: 7.8+ MB
                     TimeStamp  TmpAmb C  WindVel km/h  A.Ms.Watt  \
2018-01-01 00:00:00     00:00     22.33         11.86        NaN
2018-01-01 00:05:00     00:05     22.33          4.38        NaN
2018-01-01 00:10:00     00:10     22.31          5.08        NaN
2018-01-01 00:15:00     00:15     22.33          6.46        NaN
2018-01-01 00:20:00     00:20     22.27          4.34        NaN
...                       ...       ...           ...        ...
2018-12-17 09:10:00     09:10     29.47          0.00    3069.00
2018-12-17 09:15:00     09:15     29.29          0.00    3173.80
2018-12-17 09:20:00     09:20     28.93          0.00    3282.80
2018-12-17 09:25:00     09:25     29.81          0.00    3382.80
2018-12-17 09:30:00     09:30     29.25          0.00    3443.75
```

```
                            Timestamp        GHI      DHI
2018-01-01 00:00:00 2018-01-01 00:00:00 -22.640380  -28.733
2018-01-01 00:05:00 2018-01-01 00:05:00 -22.004570 -28.3443
2018-01-01 00:10:00 2018-01-01 00:10:00 -21.189972 -26.4709
2018-01-01 00:15:00 2018-01-01 00:15:00 -20.946122 -26.4408
2018-01-01 00:20:00 2018-01-01 00:20:00 -19.850174 -24.6612
...                                 ...        ...      ...
2018-12-17 09:10:00                 NaT        NaN      NaN
2018-12-17 09:15:00                 NaT        NaN      NaN
2018-12-17 09:20:00                 NaT        NaN      NaN
2018-12-17 09:25:00                 NaT        NaN      NaN
2018-12-17 09:30:00                 NaT        NaN      NaN

[87540 rows x 7 columns]
```

Using the data

After getting a combined Array-Weather dataframe, I then attempted to use it for any of the processing methodologies available. I opted to try STL as RDTools-YOY requires more than one year of data (we only have 2018 imported) and weather for all years (only weather data for 2017-18 available as current on unsw servers). Doubtful that STL would simply work straight up we try to execute it, but naturally it throws us an error.

Viewing the Power from the dataframe, we can see that somewhere in the import we have zeroes. The files for this period definitely exist, however I am still identifying the cause and solution. Otherwise, the Power looks completely normal, very much like our test set of Desert Knowledge unit 12 which we used to RDTools test.

In [3]:

```
import seaborn as sns
import matplotlib.pyplot as plt
from pandas.plotting import register_matplotlib_converters
register_matplotlib_converters()
#sns.set_style('darkgrid')
%matplotlib inline

plt.rc('figure',figsize=(16,12))
plt.rc('font',size=13)

freq = pd.infer_freq(df1.index[:10])
df1 = df1.resample(freq).median()

# plot the AC power time series
fig, ax = plt.subplots(figsize=(4,3))
ax.plot(df1.index, df1['A.Ms.Watt'], 'o', alpha = 0.01)
ax.set_ylim(0,7000)
fig.autofmt_xdate()
ax.set_ylabel('AC Power (W)');

try:
    from statsmodels.tsa.seasonal import STL
    stl = STL(df1['A.Ms.Watt'], seasonal=13)
    res = stl.fit()
    fig = res.plot()
    print("done")
except AssertionError as error:
    print(error)
    print("STL calculation failed")
```

```
---------------------------------------------------------------------------
-
ValueError                                Traceback (most recent call las
t)
<ipython-input-3-0b0c0c47621e> in <module>
     21 try:
     22     from statsmodels.tsa.seasonal import STL
---> 23     stl = STL(df1['A.Ms.Watt'], seasonal=13)
     24     res = stl.fit()
     25     fig = res.plot()

statsmodels\tsa\_stl.pyx in statsmodels.tsa._stl.STL.__init__()

~\Documents\PythonProject\venv\lib\site-packages\statsmodels\tsa\tsatools.
py in freq_to_period(freq)
    813     else:  # pragma : no cover
    814         raise ValueError("freq {} not understood. Please report if
you "
--> 815                          "think this is in error.".format(freq))
    816
    817

ValueError: freq T not understood. Please report if you think this is in e
rror.
```