Development of an automated script for FrED run condition extraction.

Exploration of the automated extraction of FrED operating conditions from run rogs.

Uses production run data - "./Production Runs/." Creates report in - "./"

J. Cuiffi - Penn State New Kensington

In [75]:

```
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
```

In [76]:

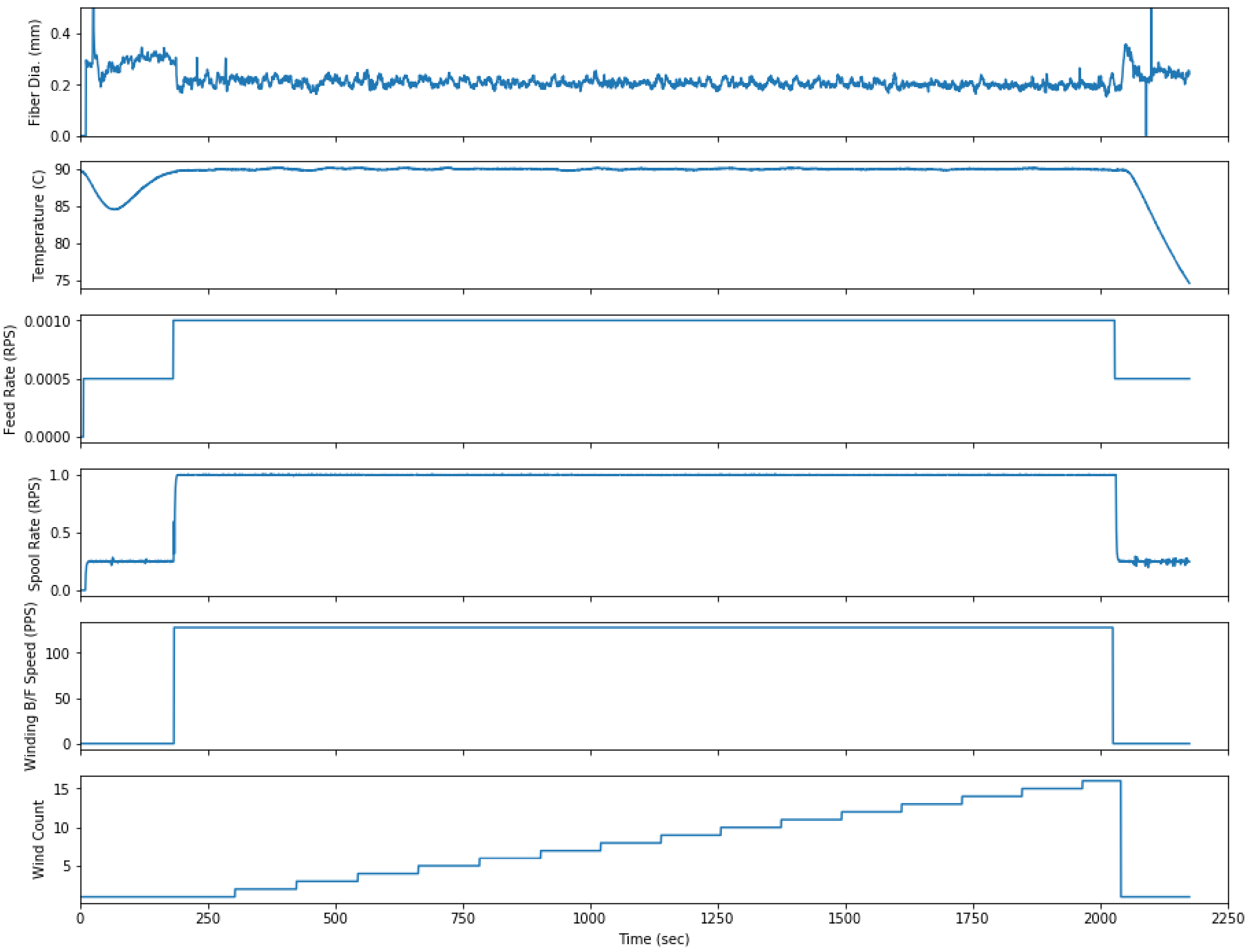
```
# load list of production run logs
# Note: many of the files have multiple run conditions, all should have the standard headers from
#       "fredmanGUI.py" - column headers shown below
path_local = 'C:/Users/cuiff/Dropbox/Python Common Library/python-fred/data/Condition Production Runs/'
file_list = os.listdir(path_local)
print("File List:")
print(file_list)
print("File Column Header:")
print(pd.read_csv(path_local + file_list[0]).columns)
```

```
File List:
['log_Manual Control__2020-04-12_12-13-50.csv', 'log_Manual Control__2020-04-12_17-26-48.csv', 'lo
g_Manual Control__2020-04-12_17-49-04.csv', 'log_Manual Control__2020-04-12_20-11-28.csv', 'log_Ma
nual Control__2020-04-12_20-44-15.csv', 'log_Manual Control__2020-04-12_21-03-31.csv', 'log_Manual
Control__2020-04-12_21-18-14.csv', 'log_Manual Control__2020-05-21_12-33-02.csv', 'log_Manual Cont
rol__2020-05-21_12-56-02.csv', 'log_Manual Control__2020-05-21_13-10-43.csv', 'log_Manual Control_
_2020-05-21_13-42-43.csv', 'log_Manual Control__2020-05-22_09-14-30.csv', 'log_Manual Control__202
0-05-22_10-51-30.csv', 'log_Manual Control__2020-05-22_13-15-23.csv', 'log_Manual Control__2020-05
-22_14-35-17.csv', 'log_Manual Control__2020-05-26_10-47-57.csv']
File Column Header:
Index(['Time (sec)', 'Run Time (sec)', 'Heater Set (C)', 'Heater Duty (0-1)',
       'Filament Feed Rate Set (RPS)', 'Spool Wind Rate Set (RPS)',
       'Spool Duty (0-1)', 'Wind B-F Speed (PPS)',
       'Filament Diameter Set (mm)', 'Heater Actual (C)',
       'Filament Feed Rate Actual (RPS)', 'Spool Wind Rate Actual (RPS)',
       'Wind Direction (R/L)', 'Wind Count (#)',
       'Filament Diameter Actual (mm)', 'Total Fiber Produced (m)',
       'Heater Current (mA)', 'Spool DC Motor Current (mA)',
       'Stepper and 12V Current (mA)', 'Total Power (W)',
       'Total Energy Used (Wh)'],
      dtype='object')
```

In [77]:

```
# example file with one run condition
run_one = pd.read_csv(path_local + 'log_Manual Control__2020-04-12_12-13-50.csv')
# plot data
fig, axs = plt.subplots(6,1,sharex=True)
fig.set_size_inches(15,12)
axs[0].plot(run_one['Run Time (sec)'], run_one['Filament Diameter Actual (mm)'])
axs[0].set_ylim([0,.5])
axs[0].set_ylabel('Fiber Dia. (mm)')
axs[1].plot(run_one['Run Time (sec)'], run_one['Heater Actual (C)'])
axs[1].set_ylabel('Temperature (C)')
axs[2].plot(run_one['Run Time (sec)'], run_one['Filament Feed Rate Actual (RPS)'])
axs[2].set_ylabel('Feed Rate (RPS)')
axs[3].plot(run_one['Run Time (sec)'], run_one['Spool Wind Rate Actual (RPS)'])
axs[3].set_ylabel('Spool Rate (RPS)')
axs[4].plot(run_one['Run Time (sec)'], run_one['Wind B-F Speed (PPS)'])
axs[4].set_ylabel('Winding B/F Speed (PPS)')
axs[5].plot(run_one['Run Time (sec)'], run_one['Wind Count (#)'])
axs[5].set_ylabel('Wind Count')
axs[5].set_xlabel('Time (sec)')
axs[5].set_xlim([0,2250])
print('Single condition example run - log_Manual Control__2020-04-12_12-13-50.csv')
```

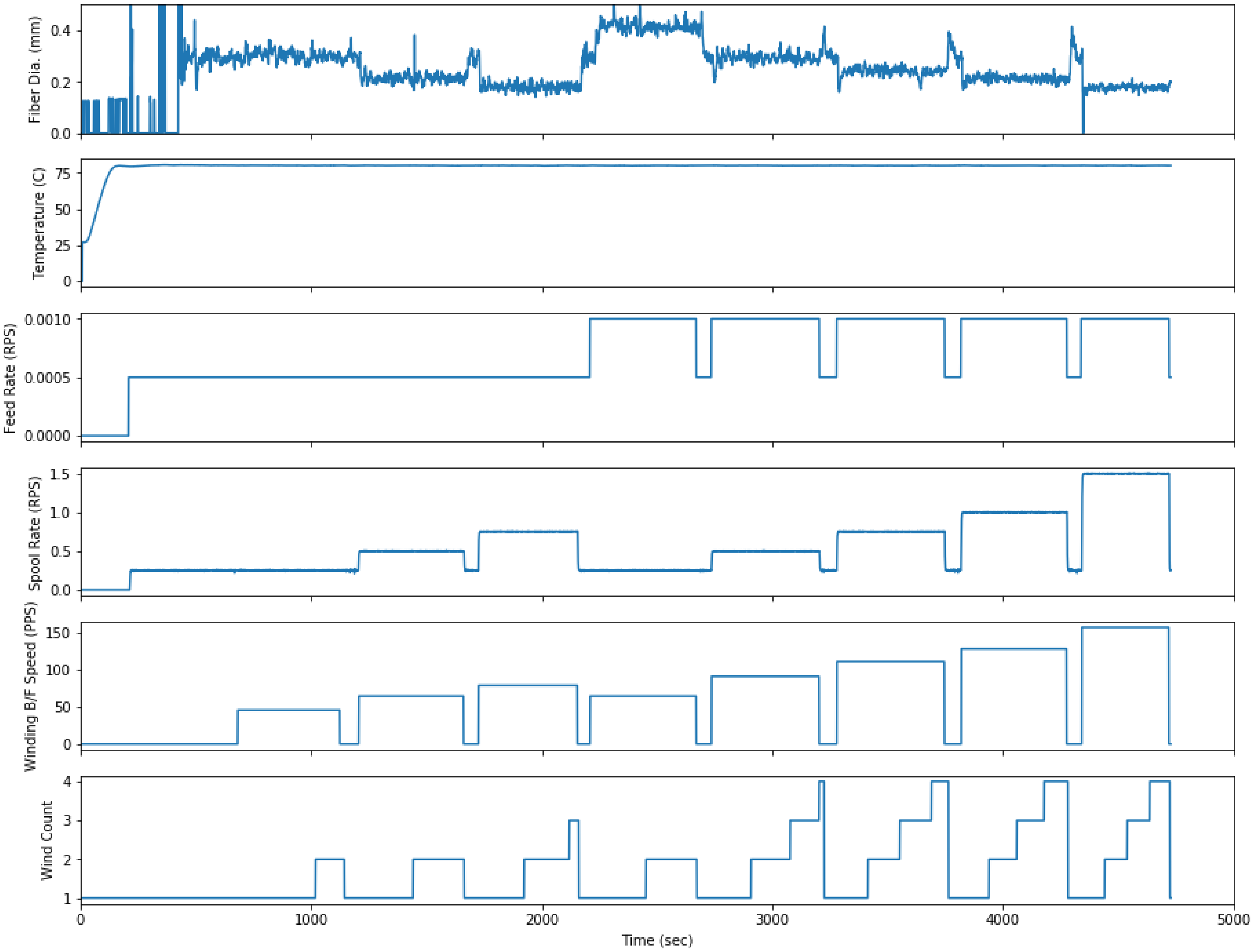
```
Single condition example run - log_Manual Control__2020-04-12_12-13-50.csv
```



In [78]:

```
# example file with multiple run conditions
run_many = pd.read_csv(path_local + 'log_Manual Control__2020-05-22_13-15-23.csv')
# plot data
fig, axs = plt.subplots(6,1,sharex=True)
fig.set_size_inches(15,12)
axs[0].plot(run_many['Run Time (sec)'], run_many['Filament Diameter Actual (mm)'])
axs[0].set_ylim([0,.5])
axs[0].set_ylabel('Fiber Dia. (mm)')
axs[1].plot(run_many['Run Time (sec)'], run_many['Heater Actual (C)'])
axs[1].set_ylabel('Temperature (C)')
axs[2].plot(run_many['Run Time (sec)'], run_many['Filament Feed Rate Actual (RPS)'])
axs[2].set_ylabel('Feed Rate (RPS)')
axs[3].plot(run_many['Run Time (sec)'], run_many['Spool Wind Rate Actual (RPS)'])
axs[3].set_ylabel('Spool Rate (RPS)')
axs[4].plot(run_many['Run Time (sec)'], run_many['Wind B-F Speed (PPS)'])
axs[4].set_ylabel('Winding B/F Speed (PPS)')
axs[5].plot(run_many['Run Time (sec)'], run_many['Wind Count (#)'])
axs[5].set_ylabel('Wind Count')
axs[5].set_xlabel('Time (sec)')
axs[5].set_xlim([0,5000])
print('Multiple condition example run - log_Manual Control__2020-05-22_13-15-23.csv')
```

```
Multiple condition example run - log_Manual Control__2020-05-22_13-15-23.csv
```



Notes on clipping out valid conditions:

Given the methods in starting and stopping the filament run, a Winding Back/Forth setting of greater than zero is a great indicator the spooling/running of fiber. In order to get a valid snapshot of the operating conditions, the following parameters must be considered:

1) Wind B/F rate > 0 - Indicates the running of filament at a set of conditions

2) Use a list of unique spooling rates and feed rates to pull out each set condition

3) The fiber diamter takes time to stabilize (~10-60sec), it is unclear how long it takes for the various conditions, but 60sec is a safe number without eliminating a lot of data

4) As the spool wind count increases, the diameter of the fiber (with no auto control) should slowly decrease. Keeping the wind count <= 5 is safe to ignore this effect.

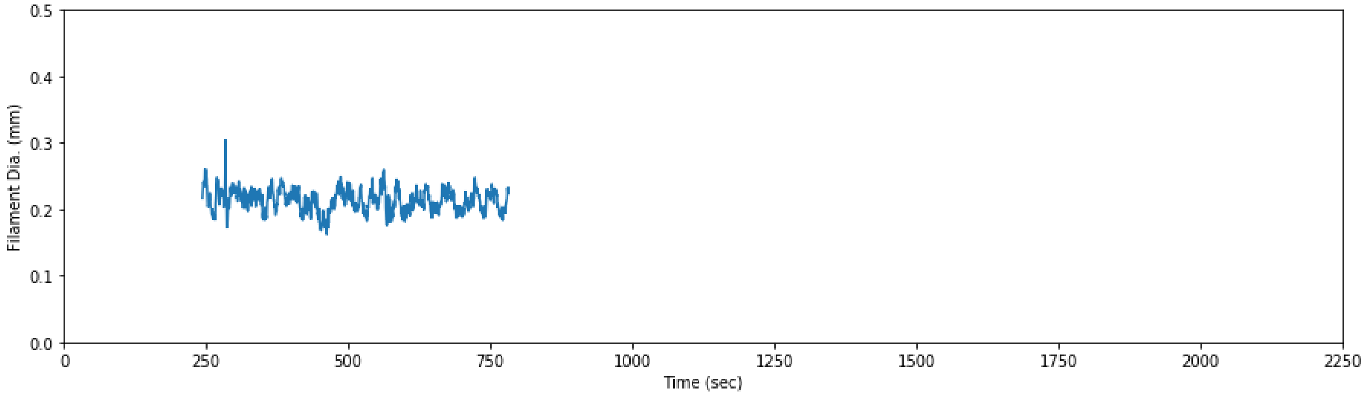
5) Ensure that there is at least 2 minutes of data

In [79]:

```python
# example clipping out data from one run file
# uses boolean dataframe masks to clip data
# Condition: wind b/f > 0
run_clip = run_one[run_one['Wind B-F Speed (PPS)'] > 0.0]
# Condition: time > 60 sec from start
run_clip = run_clip[run_clip['Run Time (sec)'] > (run_clip.iloc[0,1] + 60.0)]
# Condition: wind count <= 5
run_clip = run_clip[run_clip['Wind Count (#)'] <= 5]

# plot
fig, ax1 = plt.subplots()
fig.set_size_inches(15,4)
ax1.set_ylim([0,.5])
ax1.set_xlim([0,2250])
ax1.plot(run_clip['Run Time (sec)'], run_clip['Filament Diameter Actual (mm)'])
ax1.set_ylabel('Filament Dia. (mm)')
ax1.set_xlabel('Time (sec)')
print('Clipped data from single run file.')
print('Fiber Diameter Mean (mm) = {0}'.format(run_clip['Filament Diameter Actual (mm)'].mean()))
print('Fiber Dia. Std. Dev. (mm) = {0}'.format(run_clip['Filament Diameter Actual (mm)'].std()))
```

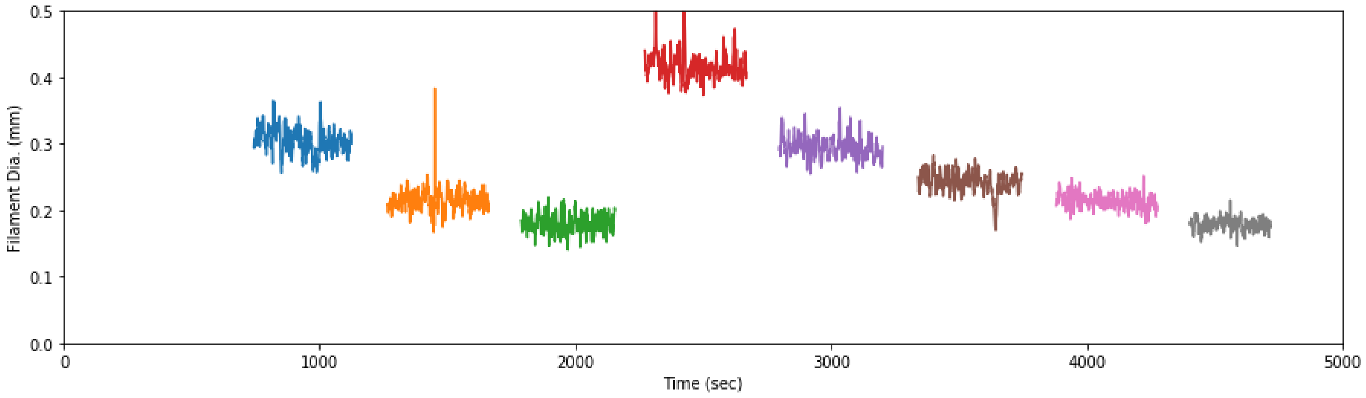
```
Clipped data from single run file.
Fiber Diameter Mean (mm) = 0.2134874651810587
Fiber Dia. Std. Dev. (mm) = 0.016746499406160787
```



In [80]:

```
# example clipping out data from multiple run file
# Condition: wind b/f > 0
run_clips = run_many[run_many['Wind B-F Speed (PPS)'] > 0]
print('Clipped data from a run file with multiple test conditions.')
fig, ax1 = plt.subplots()
fig.set_size_inches(15,4)
ax1.set_ylim([0,.5])
ax1.set_xlim([0,5000])
ax1.set_ylabel('Filament Dia. (mm)')
ax1.set_xlabel('Time (sec)')
# iterate over list of unique feed rates
for frate in run_clips['Filament Feed Rate Actual (RPS)'].unique():
    fclips = run_clips[run_clips['Filament Feed Rate Actual (RPS)'] == frate]
    # iterate over list of unique spool rates for each feed rate
    for srate in fclips['Spool Wind Rate Set (RPS)'].unique():
        clip = fclips[fclips['Spool Wind Rate Set (RPS)'] == srate]
        # Condition: time > 60 sec from start
        clip = clip[clip['Run Time (sec)'] > (clip.iloc[0,1] + 60.0)]
        # Condition: wind count <= 5
        clip = clip[clip['Wind Count (#)'] <= 5]
        print('Feed Rate (RPS) = {0}'.format(frate) + '  Spool Rate = {0}'.format(srate))
        print('Fiber Diameter Mean (mm) = {0}'.format(clip['Filament Diameter Actual (mm)'].mean
()))
        print('Fiber Dia. Std. Dev. (mm) = {0}'.format(clip['Filament Diameter Actual (mm)'].std
()))
        ax1.plot(clip['Run Time (sec)'], clip['Filament Diameter Actual (mm)'])
```

```
Clipped data from a run file with multiple test conditions.
Feed Rate (RPS) = 0.0005  Spool Rate = 0.25
Fiber Diameter Mean (mm) = 0.30355118110236273
Fiber Dia. Std. Dev. (mm) = 0.01821554078892266
Feed Rate (RPS) = 0.0005  Spool Rate = 0.5
Fiber Diameter Mean (mm) = 0.21520253164556932
Fiber Dia. Std. Dev. (mm) = 0.016121823455926027
Feed Rate (RPS) = 0.0005  Spool Rate = 0.75
Fiber Diameter Mean (mm) = 0.18083695652173906
Fiber Dia. Std. Dev. (mm) = 0.013955575428297088
Feed Rate (RPS) = 0.001  Spool Rate = 0.25
Fiber Diameter Mean (mm) = 0.4174246231155781
Fiber Dia. Std. Dev. (mm) = 0.019142264597957183
Feed Rate (RPS) = 0.001  Spool Rate = 0.5
Fiber Diameter Mean (mm) = 0.2954148148148152
Fiber Dia. Std. Dev. (mm) = 0.015893232191185908
Feed Rate (RPS) = 0.001  Spool Rate = 0.75
Fiber Diameter Mean (mm) = 0.24279115479115462
Fiber Dia. Std. Dev. (mm) = 0.016524444361946156
Feed Rate (RPS) = 0.001  Spool Rate = 1.0
Fiber Diameter Mean (mm) = 0.21374559193954654
Fiber Dia. Std. Dev. (mm) = 0.01106720799902452
Feed Rate (RPS) = 0.001  Spool Rate = 1.5
Fiber Diameter Mean (mm) = 0.17885759493670864
Fiber Dia. Std. Dev. (mm) = 0.009392768540956092
```



```
In [81]: # create script to process all of the files
         # Notes: assumes a single operating temperature in a file
         path_local_data = 'C:/Users/cuiff/Dropbox/Python Common Library/python-fred/data/Condition Product
         ion Runs/'
         path_local_report = 'C:/Users/cuiff/Dropbox/Python Common Library/python-fred/data/Reports/'
         # create output dataframe
         cols=['Run File','Feed Rate Ave (RPS)','Spool Wind Rate Set (RPS)','Spool Rate Ave (RPS)', 'Wind B
         F Rate Ave (PPS)',
               'Heater Set (C)', 'Heater Temp Ave (C)', 'Filament Diamter Ave (mm)', 'Filament Std Dev (m
         m)',
               'System Power Ave (W)', 'System Power Std Dev (W)', 'Heater Current Ave (mA)',
               'Heater Current Std Dev (mA)', 'Spool DC Motor Current Ave (mA)', 'Spool DC Motor Current St
         d Dev (mA)',
               'Stepper and 12V Current Ave (mA)', 'Stepper and 12V Current Std Dev (mA)']
         outdf = pd.DataFrame(columns=cols)
         # get list of datafiles
         file_list = os.listdir(path_local_data)
         # iterate through files
         for file in file_list:
             run_data = pd.read_csv(path_local_data + file)
             # Condition: wind b/f > 0
             run_data = run_data[run_data['Wind B-F Speed (PPS)'] > 0]
             # iterate over list of unique feed rates
             for frate in run_data['Filament Feed Rate Actual (RPS)'].unique():
                 fclips = run_data[run_data['Filament Feed Rate Actual (RPS)'] == frate]
                 # iterate over list of unique spool rates for each feed rate
                 for srate in fclips['Spool Wind Rate Set (RPS)'].unique():
                     clip = fclips[fclips['Spool Wind Rate Set (RPS)'] == srate]
                     # Condition: time > 60 sec from start
                     clip = clip[clip['Run Time (sec)'] > (clip.iloc[0,1] + 60.0)]
                     # Condition: wind count <= 5
                     clip = clip[clip['Wind Count (#)'] <= 5]
                     # Ensure that there is at least 2 minutes of data
                     if (clip['Run Time (sec)'].size > 0):
                         if ((clip.iloc[-1,1] - clip.iloc[0,1]) >= 120.0):
                             # Append data to dataframe
                             outdf = outdf.append(pd.Series([file,clip['Filament Feed Rate Actual (RPS)'].m
         ean(),
                                                                 clip['Spool Wind Rate Set (RPS)'].mean(),
                                                                 clip['Spool Wind Rate Actual (RPS)'].mean(),
                                                                 clip['Wind B-F Speed (PPS)'].mean(),
                                                                 clip['Heater Set (C)'].mean(),
                                                                 clip['Heater Actual (C)'].mean(),
                                                                 clip['Filament Diameter Actual (mm)'].mean(),
                                                                 clip['Filament Diameter Actual (mm)'].std(),
                                                                 clip['Total Power (W)'].mean(), clip['Total Power
          (W)'].std(),
                                                                 clip['Heater Current (mA)'].mean(), clip['Heater C
         urrent (mA)'].std(),
                                                                 clip['Spool DC Motor Current (mA)'].mean(),
                                                                 clip['Spool DC Motor Current (mA)'].std(),
                                                                 clip['Stepper and 12V Current (mA)'].mean(),
                                                                 clip['Stepper and 12V Current (mA)'].std(),
                                                                ],
                                                                index=outdf.columns), ignore_index=True)
         # export output dataframe to csv
         outdf.to_csv(path_local_report + 'Run Condition Data Summary.csv',index=False)

In [ ]:
```