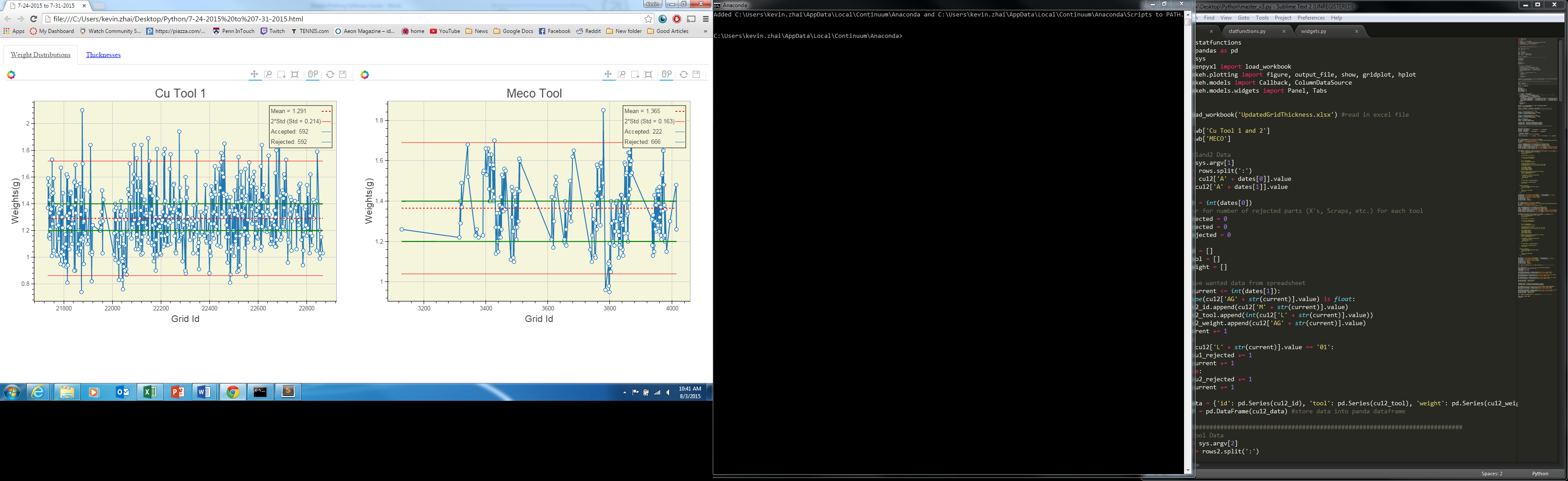
Bokeh Plotting Software Guide – 8/5/15

Kevin Zhai

GTAT Summer 2015 Internship Project



# Main Features:

-Grid Weight Visualization and Statistical Controls for CuTool1 and 2, in addition to Meco Tool

-Meco Thickness visualization and statistical controls

-Dynamic selection for interactively updating the average and standard deviation controls for grid weights

-Ability to plot one day of grid production or a range of days, and inspecting all the data in one place

-Resize, pan, zoom, hover, and save tools for all graphs

# Documentation:

<https://docs.python.org> – Python 2.7

<http://bokeh.pydata.org> – Bokeh – Python Based graphing library

<http://pandas.pydata.org> – Pandas – Python library for handling large data sets

<http://openpyxl.readthedocs.org> – OpenPyxl – Python Library for reading/writing Excel files

# List of Files:

**Master\_v3.py** – main script for plotting Copper Tool 1 weights and MECO tool weights/thicknesses. Does not have handling for Copper Tool 2. Includes dynamic selection, allowing user to select a range of data points and see the updated mean and standard deviation controls

**Master.py** – older version of Master\_v3.py that includes plotting for Copper Tool 1, Copper Tool 2, and Meco tool. Does not include statistical controls for MECO thicknesses or dynamic selection

**Statfunctions.py** – defined helper functions used in other master scripts

# Quickstart Overview:

To begin plotting, the master scripts need to be called in a command line interface (Terminal for Mac, Command Prompt in Windows). The commands differ slightly for each one, but as a basic overview:

|  |  |  |
| --- | --- | --- |
| **Purpose** | **Mac Terminal** | **Windows Command Prompt** |
| Change Directory | cd | cd |
| List Files | ls | dir |
| Copy File | cp | copy |
| Stop Running Process | Ctrl+C | Ctrl+C |
| Enter Python Interpreter | python | python |

In the directory with the master scripts, enter:

**python (name of script) XXXX:XXXX YYYY:YYYY**

to run the desired script. The XXXX:XXXX are the Excel row ranges from the CuTool1and2 worksheet, and the YYYY:YYYY are the Excel row ranges for the MECO tool worksheet. In the current version of the scripts, both of these ranges need to be entered after the script name or an error will be thrown, because the script expects these two arguments to be provided. For example, on 8/2/15, the MECO tool was not run. In order to run the script for 8/2/15 however, the YYYY:YYYY has to be provided still, so the user may choose an arbitrary range.

In many cases, it is very useful to enter the Python interpreter after a script is run, allowing the user to interactively examine each variable of the script. This is especially useful for debugging after an error is thrown. To start the interpreter after running a script, enter:

**python –i (name of script) XXXX:XXXX YYYY:YYYY**

To exit the interpreter, press **Control+D.**

If no errors were thrown and the script is successfully run, an .html file will be produced in the same directory and automatically opened in the default web browser. The name of the file will be date or date range of the produced grids.

# Implementation Details:

## Importing

* Import the necessary libraries and files, including statfunctions, openpyxl, bokeh methods, and pandas.

## Reading the Data

* Use OpenPyxl to load the workbook in a variable in order to read the data. The file must be named UpdatedGridThickness.xlsx and located in the same directory as the script, but it can be changed to a different name.
* Take as the first argument and second arguments to the script the row ranges for Cu Tools and Meco Tool respectively.
* Using the row ranges, read the correct Excel cells for the desired data and store everything in appropriate lists. Handle counters for the rejected grids

## Storing Data into Pandas

* Using the Pandas Library, create a dictionary mapping column names to the Panda series created from the distinct lists of data
* Using the created dictionary, create a Pandas Dataframe for each tool, storing all of the data

## Retrieving Data from Pandas

* Set the multiplier for the standard deviation controls
* Create Lists to be used in Bokeh from the main Panda Dataframes

## Bokeh Plotting

- Handle the string manipulation in order to have the date be the name of the output html file

-Create each plot for each tool, starting with creating the overall figure (graph itself) and then adding the associated glyphs to each figure (lines, circles, etc.)

-Embedded javascript code for dynamic selection interaction

http://bokeh.pydata.org/en/latest/docs/user\_guide/interaction.html#userguide-interaction

-Organize plots into desired format (grid, horizontal, tabs, etc.)

http://bokeh.pydata.org/en/latest/docs/user\_guide/layout.html

# How To:

**Change the multiple of the standard deviation controls (have 1\*std lines, 2.5\*, etc.)** : change cl variable in master script, line 94

**Add more tools** : Go through steps II through V using an existing tool as a framework and adjusting the layout and tabs at the end

**Change initial data source (excel, SQL, Google Docs, etc.)**: Import the correct libraries for handling the desired database and edit the implementation in part II.

**Adding widget interaction:** http://bokeh.pydata.org/en/latest/docs/user\_guide/interaction.html#userguide-interaction

# General Notes

-Data Source Object documentation: http://bokeh.pydata.org/en/latest/docs/reference/models.html#bokeh.models.sources.DataSource

-Select Tools do not work well with Bokeh Gridplot layout

-tab spacing is very important in Python!