Chapter 10
Formatting and Coverting IPython Notebooks

As you might have come to realize through-out the course of this notebook, formatting and sharing the IPython Notebook can be quite simple. If you need to convert your notebook to PDF then your options range from simple to complicated. Since my projects needed to be in PDF, this chapter will focus on sharing my experiences with converting an IPython Notebook to PDF.

10.1 NBConvert to LaTeX to PDF

This is hands down the simplest option.

This is what IPython says about NBConvert: "Currently, nbconvert is provided as a command line tool, run as a script using IPython. A direct export capability from within the IPython Notebook web app is planned."

Like many features I have discussed in this notebook, nbconvert might become obsolete in the future. You can read more about the converting formats and options of nbconvert at http://ipython.org/ipython-doc/1/interactive/nbconvert.html.

10.1.1 IPython Notebook, to LaTeX, to PDF

The best advice I can give you is to format the notebook with LaTeX (for image imbeding, symbols, equations, lists, etc.) or don't do any additional formatting (such as HTML). NBConvert has a built in option that converts to the notebook to LaTeX, then to PDF in one command line execution. The PDF will be a LaTeX version of your notebook if you did not use HTML to format cells (such as (p), (b), (br), etc.) and did not imbed images with HTML (such as (img src=)).

Below is a snippet of Chapter 9 that has been converted to LaTex then PDF. This will give you an idea of the PDF's default formatting/appearance.

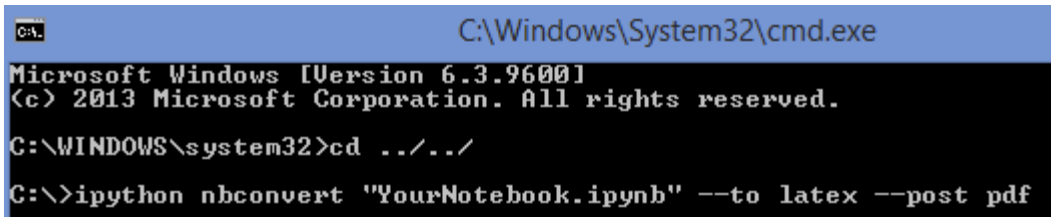
```
In [3]: #function for converting x and y to an angular position
        def calc_theta(y,x):
            return ((180/math.pi)*math.atan2((y-1152.57606607263),(x-394.773399557239)))+90

        #creates a new column in the DataFrame and populates it with theta
        PendFrame['theta']= PendFrame.apply(lambda row: calc_theta(row['y(pixels)'],row['x(pixels)'])
                                            , axis=1)

In [4]: PendFrame.head()

Out[4]:    time(sec)  x(pixels)  y(pixels)      theta
        0    0.00000    347.397     11.801  -2.378128
        1    0.15452    722.760     60.761  16.720526
        2    0.26375    748.770     66.665  18.055472
        3    0.32808    598.070     28.983  10.255820
        4    0.40682    384.457     11.128  -0.517825
```

To perform this conversion, start with opening your command prompt. Next use cd (refer to Section 3.2) and change to the directory your notebook is in. Then execute the code from below with the name of your notebook in place of "YourNotebook".



Now there is a PDF of your notebook in the same directory as the .ipynb file.

10.1.2 LaTeX in the IPython Notebook

If you are unfamiliar with LaTeX, there are tools that will help you generate the equations and symbols you need until you gain more comfort with the language.

The main one I recommend is the Equation Editor from NumberEmpire: http://bit.ly/1v9MPgd

The two ways I use LaTeX code in the notebook is either in a sentence (i.e. when talking about theta I will include the LaTeX code for theta) or as an equation.

To include LaTeX symbols in your writing use a single dollar sign to open and close the code (think of these are parentheses). For example, would be enclosed in dollar signs as \(\theta\).

To include LaTeX equations, use two dollar signs to open and two dollar signs to close. This will give the equation its own space. Here is an example of the code for the pendulum time series from Chapter 9. \[\{(t_i, \theta_i):i=0,...,n\}\]

10.2 NBConvert to HTML

Deep into this project I learned about the previous LaTeX options. I'm not sure how to achieve the same control over the layout in LaTeX as I did formatting this notebook in HTML, but I might have tried. Since I had most of the work done in HTML, I decided it would be more efficient to find a way to generate a formatted PDF (i.e. chapters start on fresh pages, images don't split on page breaks, etc) than to switch large portions of the notebook to LaTeX.

HTML is a very simple formatting option that is supported by the "markdown" option for the notebook cells. It makes sense that notebooks created and shared in a web browers would utilize this code. If you have a notebook that contains any HTML code whatsoever you will need to switch it to LaTeX, use the methods described in this section, or see if a better way has been developed since I wrote this.

If you would like to learn more about HTML, CodeAcademy.com is a great resource for learning new languages.

If you need the HTML codes for various symbols, I like this site: http://bit.ly/1nl7tsl

To perform this conversion from .ipynb to .html, start with opening your command prompt. Next use cd (refer to Section 3.2) and change to the directory your notebook is in. Then execute the code from below with the name of your notebook in place of "YourNotebook".

```
C:\Windows\System32\cmd.exe
Microsoft Windows [Version 6.3.9600]
(c) 2013 Microsoft Corporation. All rights reserved.

C:\WINDOWS\system32>cd ../../

C:\>ipython nbconvert "YourNotebook.ipynb" --to html
```

YourNotebook.html will now be in the same directory as YourNotebook.ipynb.

10.3 HTML to PDF Options

First, I will tell you how I completed this PDF, then I will tell you about alternative options that were suggested. Since I did not try the alternative methods, it will be up to you to investigate them if you would like a better/different way.

10.3.1 HTML to PDF converters and BR

There are many free HTML to PDF converters online. I had trouble finding one that did not split images at the page break. If you manage to find a converter that formats page breaks well, then you're done. If not, you can add the HTML tag BR as many times are you need to in order to force cells further apart from each other. This is an extremely unsophisticated approach. I figured since converters are strictly turning the HTML file into a PDF with no regard for formatting, then they must follow that strict conversion with arbitrary spacing as well. It worked, and does work. In the absence of a "correct" way to complete a task, we must improvise.

```
10 Formatting and Coverting IPython Notebooks
<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br>
```

This method is unavoidably, in my experience, trial and error. Add and remove BRs, convert to HTML, and then convert to PDF and see how the spacing looks. Repeat, repeat, repeat until the desired formatting is achieved. It is as tedious as it sounds, but doesn't take nearly as long as you might think.

10.3.2 Microsoft Word

Since the conversion to HTML is so simple, Dr. Doi came up with the idea of opening the HTML in Microsoft Word. Since my Word is from 2007, I was unable to complete this task. If you have a newer version of Word and have experience with formatting in word, give it a shot. You should be able to open the HTML file, edit it, and export it as a PDF.

10.3.3 Dexy

Dexy was recommended by Dr. Granger and is the library he uses to make PDF documents of his IPython Notebooks. If my BR scheme did not work as well as it did, I would have learned and use Dexy to complete this PDF. Information on Dexy can be found at: http://dexy.it/