Escaping Excel Hell with Python & Pandas

MinneBar 11 - April 23, 2016

Chris Moffitt

What this presentation is not about….

http://deadline.com/2016/01/oriental-dreamworks-kung-fu-panda-3-producer-melissa-cobb-head-of-studio-1201691805/

What will we cover?

Session Overview

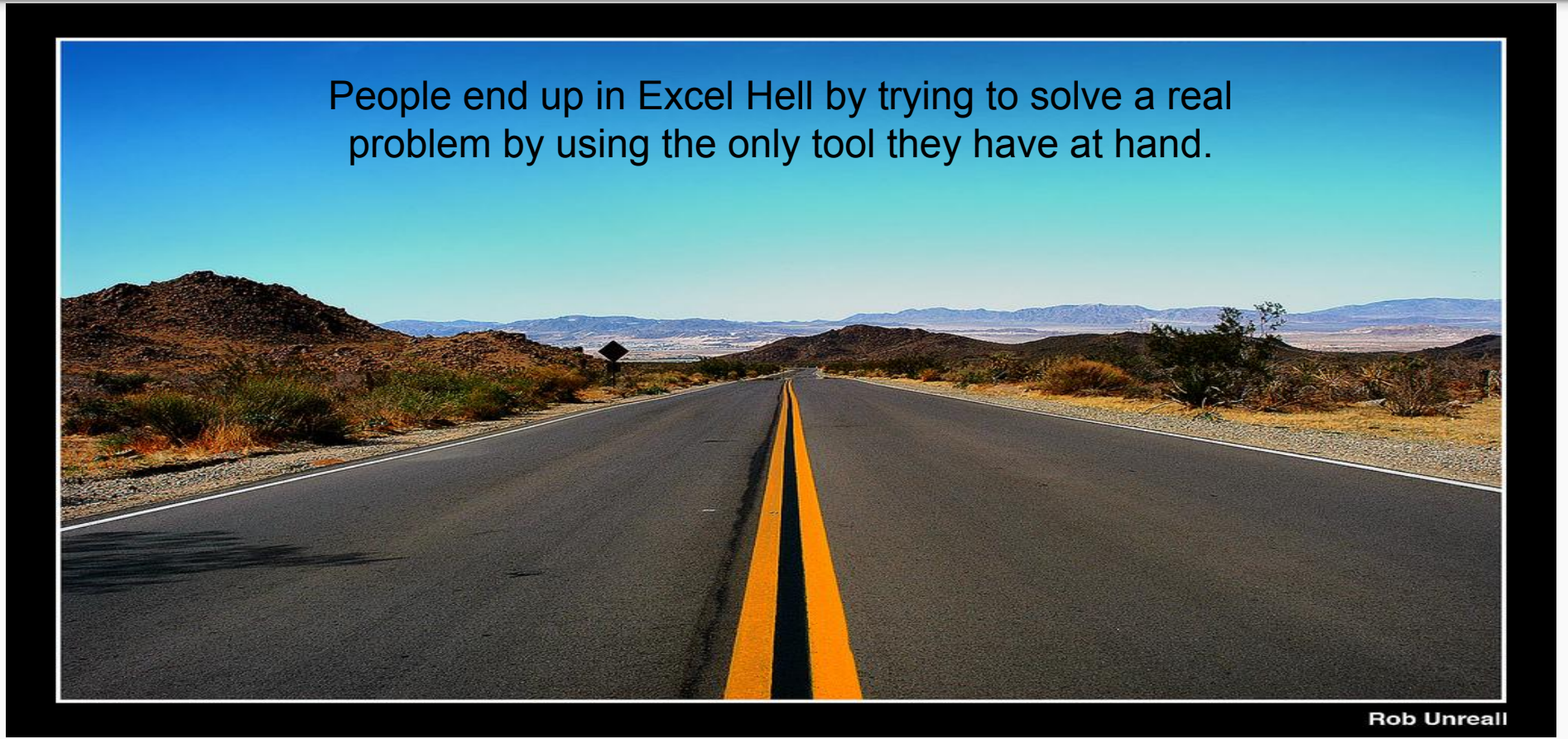
- My background
- What is Excel Hell?
- What are python and pandas?
- Short demo (time permitting)
- What are the benefits (and drawbacks) of using python?
- Tips to get started on the journey

Excel Hell

“A place of torment or misery caused by trying to use Excel as your primary data manipulation tool.”

- Paraphrased from dictionary.com

The Road to Excel Hell is paved with good intentions...

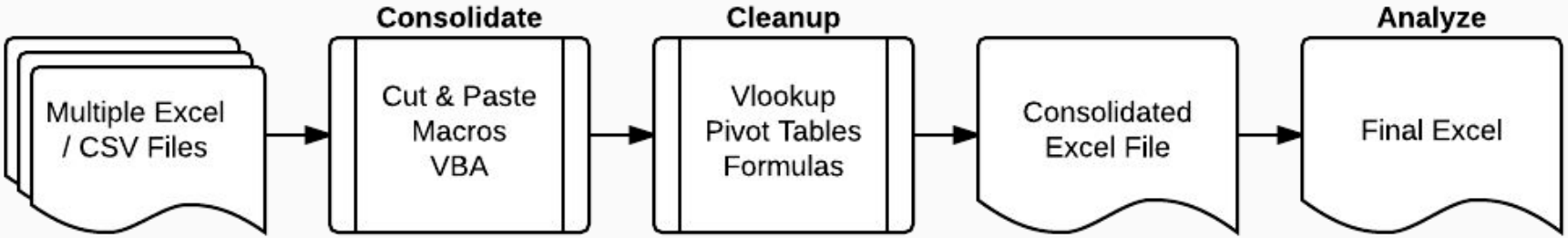


https://www.flickr.com/photos/robunreall/3661141416

Data Wrangling = Gateway to Excel Hell

Data munging or **data wrangling** is loosely the process of manually converting or mapping **data** from one "raw" form into another format that allows for more convenient consumption of the **data** with the help of semi-automated tools.

-wikipedia

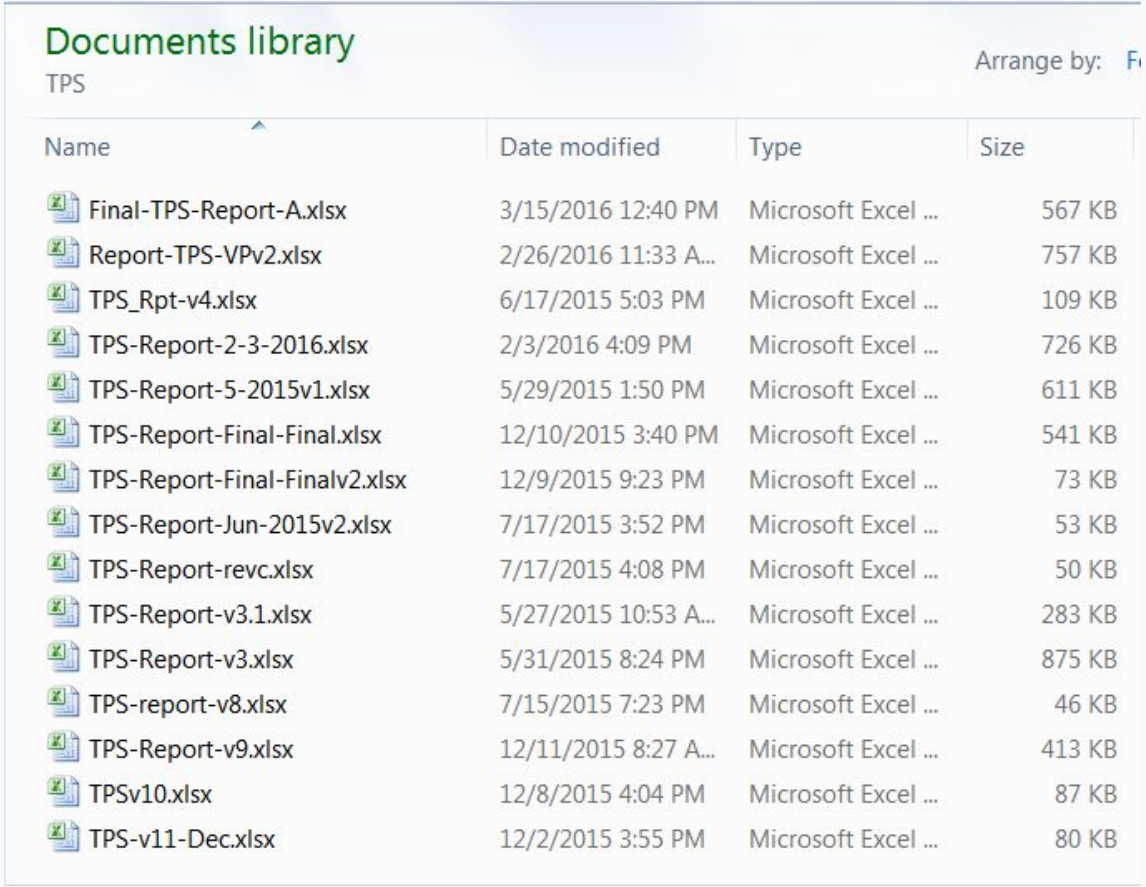


What does Excel Hell look like?

No Version Control

- How do you know which file contains the latest code?
- What changed between versions?
- Why did you make changes?

Look Familiar?



FOMC
Fear of Making Changes

- No single “flow” to the sheet
- How are formulas and VBA tied together?
- What underlying assumptions does the worksheet expect?

Looks like this

TIPSLadder.XLS

	B	C	D	J	K	L	M	N	O	P	Q	R
1	$1,000	Face Value of One Bond			Mar-30-12	Date of WSJ Quotes					0	<- Purchase s
2	$30,000	Desired Annual Real Amount			226.60094	Ref CPI on WSJ Quote Date						
3	$839,046	Calculated Total Cost of Ladder										
4				WSJ Pricing on 03/30/12				# of Years				
5	Matures	Coupon	Term in Years	Ask Price	CPI Adjusted Principal	Yield to Maturity	Years to Cover	To-tal	Be-fore	Af-ter	# Bonds Needed	Cost of Bonds
6	Feb-15-42	0.750%	30.0	96.18750	1,002.83	0.900%	2042	1	-	-	30.0	28,938
7	Feb-15-41	2.125%	30.0	132.34375	1,034.75	0.858%	2041	1	-	-	28.0	38,344
8	Feb-15-40	2.125%	30.0	131.90625	1,048.40	0.839%	2037-2040	4	3	-	107.0	147,971
9	Apr-15-32	3.375%	30.5	152.59375	1,276.63	0.588%	2032-2036	5	-	4	103.0	200,650
10	Apr-15-29	3.875%	30.0	154.37500	1,378.41	0.532%	2029-2031	3	-	2	48.0	102,140
11	Jan-15-29	2.500%	20.0	132.21875	1,055.43	0.497%		-	-	-	-	-
12	Apr-15-28	3.625%	30.0	147.81250	1,401.02	0.515%	2028	1	-	-	14.0	28,992
13	Jan-15-28	1.750%	20.0	119.53125	1,081.65	0.465%		-	-	-	-	-
14	Jan-15-27	2.375%	20.0	128.56250	1,123.65	0.385%	2027	1	-	-	17.0	24,558
15	Jan-15-26	2.000%	20.0	122.81250	1,141.70	0.308%	2026	1	-	-	16.0	22,434
16	Jan-15-25	2.375%	20.5	127.37500	1,202.15	0.204%	2025	1	-	-	15.0	22,969
36	Jul-15-14	2.000%	10.0	109.03125	1,202.15	-1.844%		-	-	-	-	-
37	Apr-15-14	1.250%	5.0	106.12500	1,070.73	-1.693%	2014	1	-	-	15.0	17,045
38	Jan-15-14	2.000%	10.0	106.90625	1,226.37	-1.785%		-	-	-	-	-
39	Jul-15-13	1.875%	10.0	105.28125	1,233.78	-2.143%		-	-	-	-	-
40	Apr-15-13	0.625%	5.0	102.65625	1,072.08	-1.899%	2013	1	-	-	15.0	16,508
41												
42	Weighted Average Real YTM, Total Cost, and Real (Apr-01-20					0.251%		30	3	8	575.0	839,046

Ladder | WSJ | Instructions | Background | DevLog

Ready | Sum= 839,046 | SCRL | CAPS | NUM

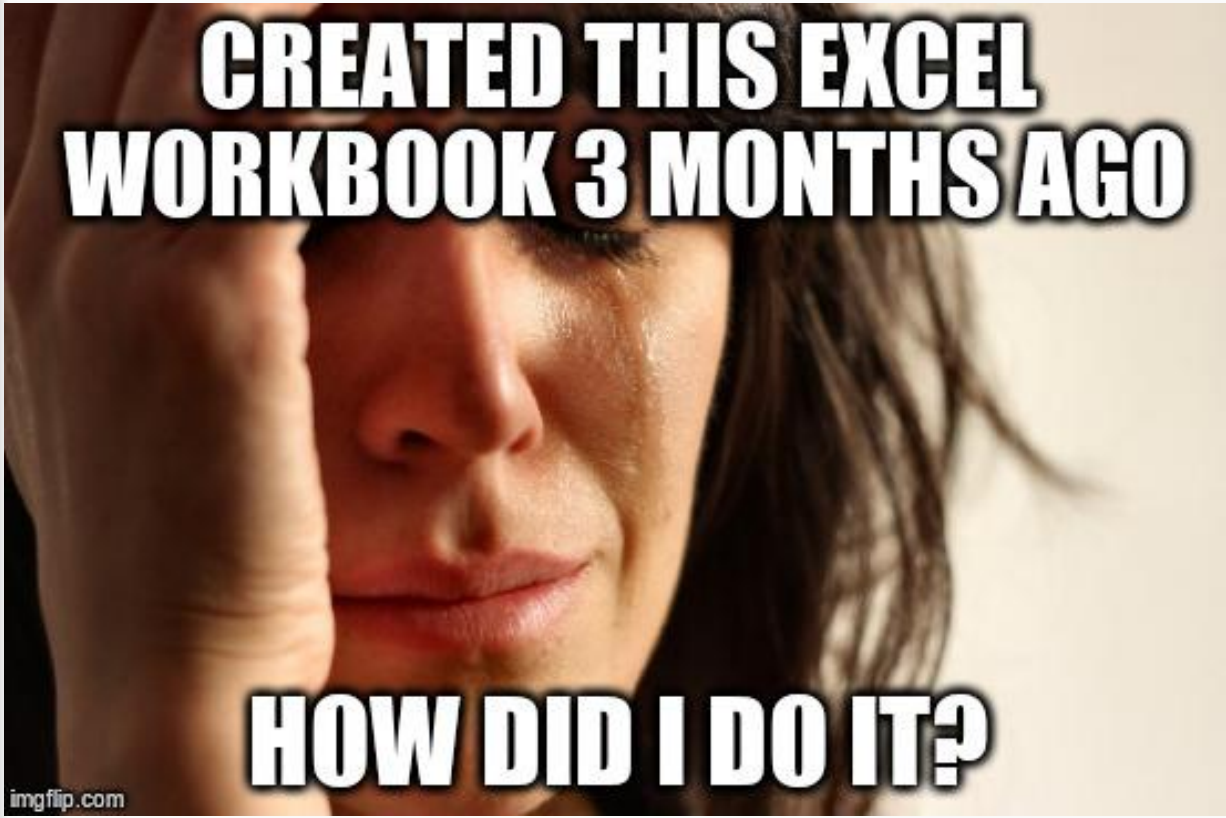
Feels like this

“Big Data” = Big Problems

- Excel Max = 1M rows by 16K columns
- Even if you can import that much data, can you do anything with it?
- Even much smaller data sets are difficult to manipulate at scale

Difficult to document

- How can you replicate your workbook?
- What manual steps did you (or someone else take)?
- Where did the data come from?
- Ultimately the only option is a Word document with screenshots and detailed descriptions.

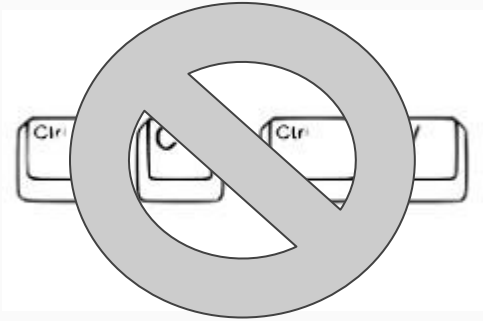


Is the workbook giving you the “right” results?

http://dilbert.com/strip/2016-01-07

Additional Challenges

- Ctrl+c Ctrl+v is not a scalable ETL solution

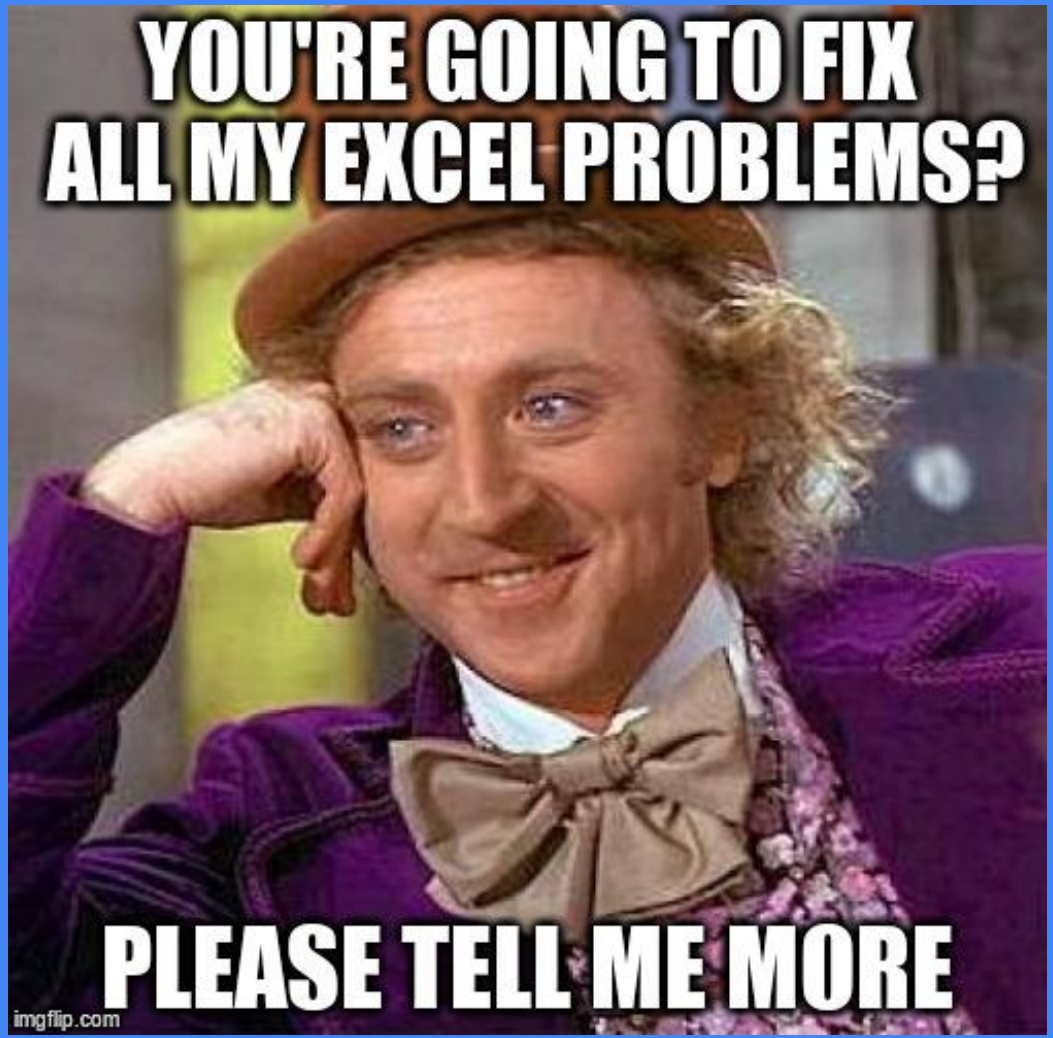


- Difficult to bring in other data (web, SQL, json, etc)

- Challenging to debug complex formulas

```
=IF(ISBLANK($D$3),"",IF(ISERROR(VALUE($D$3)),SUMIF(Qty!$B:$B,$D$3,Qty!$H:$H),SUMIF(Qty!$A:$A,$D$3,Qty!$H:$H)))
```

https://www.flickr.com/photos/cwest666/1458716901


YOU'RE GOING TO FIX
ALL MY EXCEL PROBLEMS?
PLEASE TELL ME MORE
imgflip.com

Introduction to the Technology

- Free, Open Source dynamic language
- Used by many large organizations
- Rich ecosystem of libraries
- Strong usage within scientific computing & data science communities
- Friendly user community
- Runs on all major OS'
- Original development started in 1989
- Good balance between ease and power

General purpose language

- Free, Open Source library for analysis and manipulation of structured data
- Started in 2008
- Designed to be fast for many data calculations
- Scaleable to very large data sets
- Easy to transfer data. Works well with Excel
- Supports complex visualization of data
- Originally developed for financial users but also compares favorably to R for stats

Specifically designed library

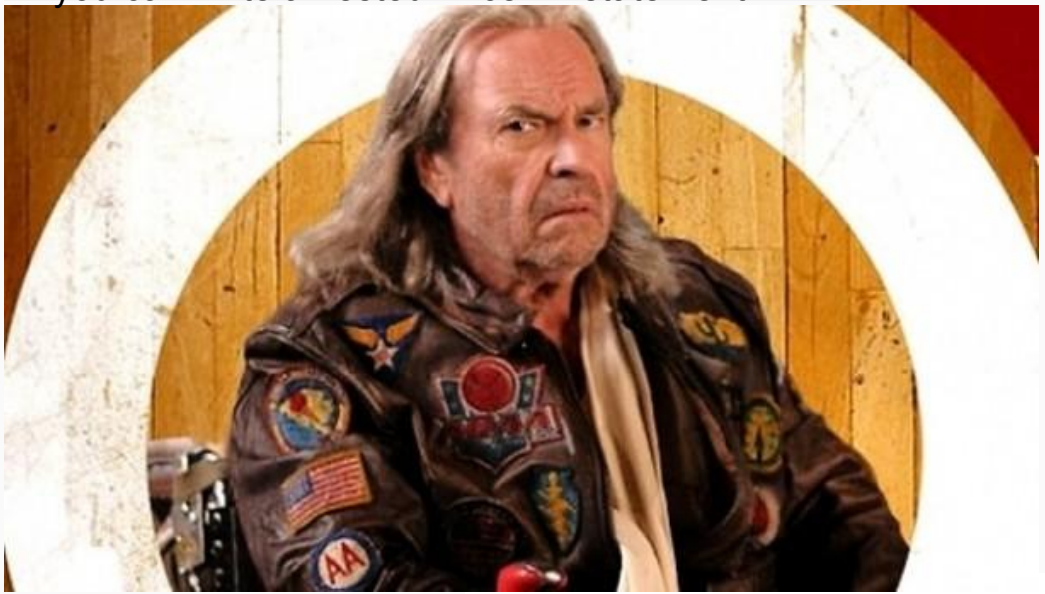
Sounds great but this will not work for me…

1. Excel is not going away

2. In any sufficiently large group of Excel users, there will be an “Alpha User”

3. The “Alpha User” can be trained to start building python-based solutions
   a. Start small
   b. Save time
   c. Increase accuracy
   d. Rinse and repeat

If you can write a nested Excel IF statement



You can write python code

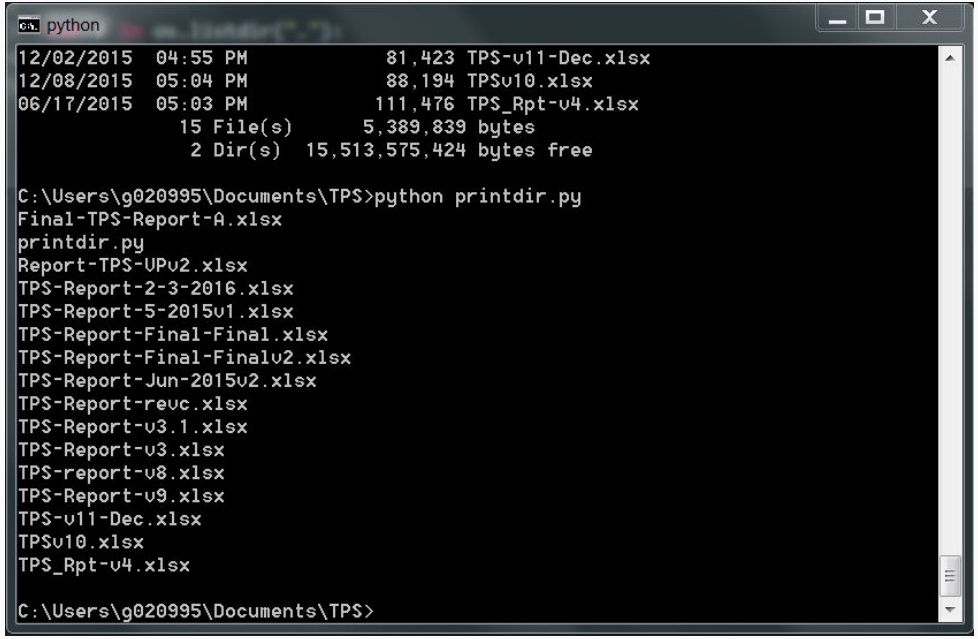
Python 101 - Simple Development Flow

Create script in editor

printdir.py

```
1  import os
2
3  for f in os.listdir("."):
4      print(f)
5
```

Execute on the command line



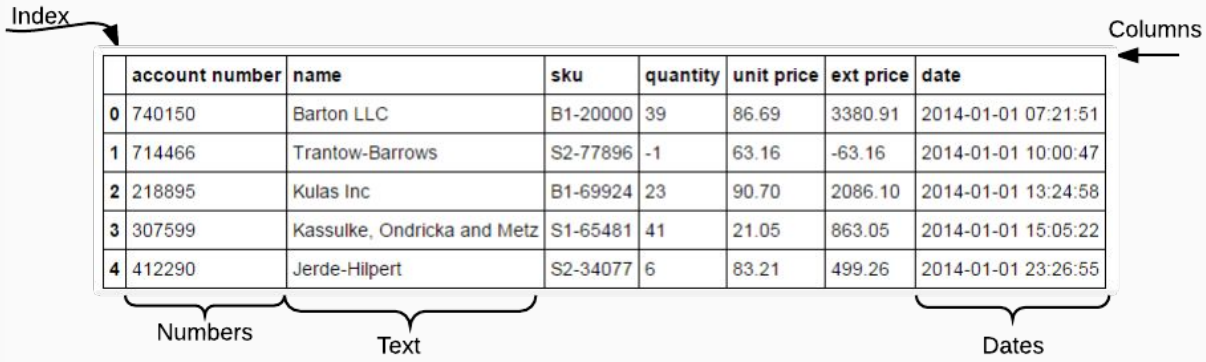
Assuming python installed on system
Will give pointers at the end of presentation

Pandas DataFrame

- Most common data structure in Pandas
- 2 dimensional table - like a spreadsheet

- Goal: Get your data into a DataFrame and manipulate using pandas + python

Pandas DataFrame Fundamentals



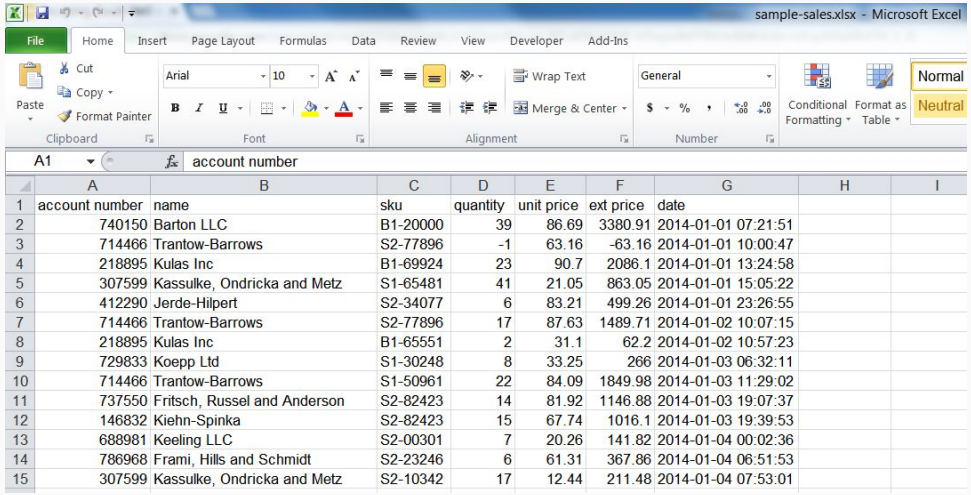
	account number	name	sku	quantity	unit price	ext price	date
0	740150	Barton LLC	B1-20000	39	86.69	3380.91	2014-01-01 07:21:51
1	714466	Trantow-Barrows	S2-77896	-1	63.16	-63.16	2014-01-01 10:00:47
2	218895	Kulas Inc	B1-69924	23	90.70	2086.10	2014-01-01 13:24:58
3	307599	Kassulke, Ondricka and Metz	S1-65481	41	21.05	863.05	2014-01-01 15:05:22
4	412290	Jerde-Hilpert	S2-34077	6	83.21	499.26	2014-01-01 23:26:55

Once your data is in a DataFrame, you can:

- Combine with other DataFrames
- Add additional columns
- Perform mathematical operations
- Clean up the data
- Group and summarize data
- Work with time series
- Plot
- Just about anything you can do in Excel...

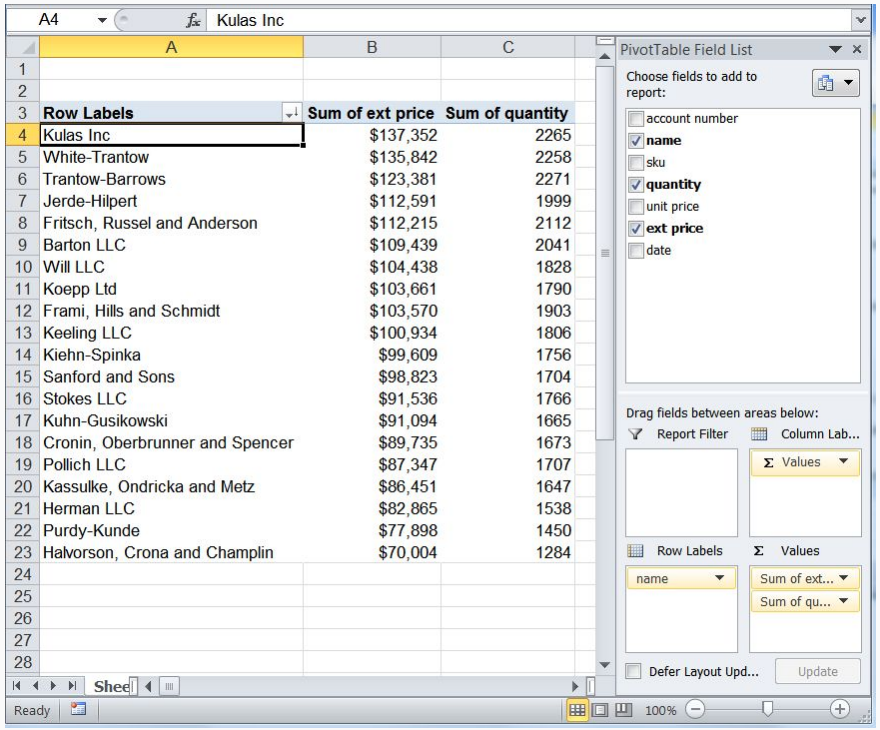
Example Data Manipulation: Excel vs. Pandas

Determine Total Sales By Customer



	A	B	C	D	E	F	G
1	account number	name	sku	quantity	unit price	ext price	date
2	740150	Barton LLC	B1-20000	39	86.69	3380.91	2014-01-01 07:21:51
3	714466	Trantow-Barrows	S2-77896	-1	63.16	-63.16	2014-01-01 10:00:47
4	218895	Kulas Inc	B1-69924	23	90.7	2086.1	2014-01-01 13:24:58
5	307599	Kassulke, Ondricka and Metz	S1-65481	41	21.05	863.05	2014-01-01 15:05:22
6	412290	Jerde-Hilpert	S2-34077	6	83.21	499.26	2014-01-01 23:26:55
7	714466	Trantow-Barrows	S2-77896	17	87.63	1489.71	2014-01-02 10:07:15
8	218895	Kulas Inc	B1-65551	2	31.1	62.2	2014-01-02 10:57:23
9	729833	Koepp Ltd	S1-30248	8	33.25	266	2014-01-03 06:32:11
10	714466	Trantow-Barrows	S1-50961	22	84.09	1849.98	2014-01-03 11:29:02
11	737550	Fritsch, Russel and Anderson	S2-82423	14	81.92	1146.88	2014-01-03 19:07:37
12	146832	Kiehn-Spinka	S2-82423	15	67.74	1016.1	2014-01-03 19:39:53
13	688981	Keeling LLC	S2-00301	7	20.26	141.82	2014-01-04 00:02:36
14	786968	Frami, Hills and Schmidt	S2-23246	6	61.31	367.86	2014-01-04 06:51:53
15	307599	Kassulke, Ondricka and Metz	S2-10342	17	12.44	211.48	2014-01-04 07:53:01

- Mock sales transaction data for customers
- Summarize sales by customer
- Generate a simple pivot table to analyze



	A	B	C
3	Row Labels	Sum of ext price	Sum of quantity
4	Kulas Inc	$137,352	2265
5	White-Trantow	$135,842	2258
6	Trantow-Barrows	$123,381	2271
7	Jerde-Hilpert	$112,591	1999
8	Fritsch, Russel and Anderson	$112,215	2112
9	Barton LLC	$109,439	2041
10	Will LLC	$104,438	1828
11	Koepp Ltd	$103,661	1790
12	Frami, Hills and Schmidt	$103,570	1903
13	Keeling LLC	$100,934	1806
14	Kiehn-Spinka	$99,609	1756
15	Sanford and Sons	$98,823	1704
16	Stokes LLC	$91,536	1766
17	Kuhn-Gusikowski	$91,094	1665
18	Cronin, Oberbrunner and Spencer	$89,735	1673
19	Pollich LLC	$87,347	1707
20	Kassulke, Ondricka and Metz	$86,451	1647
21	Herman LLC	$82,865	1538
22	Purdy-Kunde	$77,898	1450
23	Halvorson, Crona and Champlin	$70,004	1284

- Build pivot table and sort values

Equivalent Pandas Example

simple_report.py

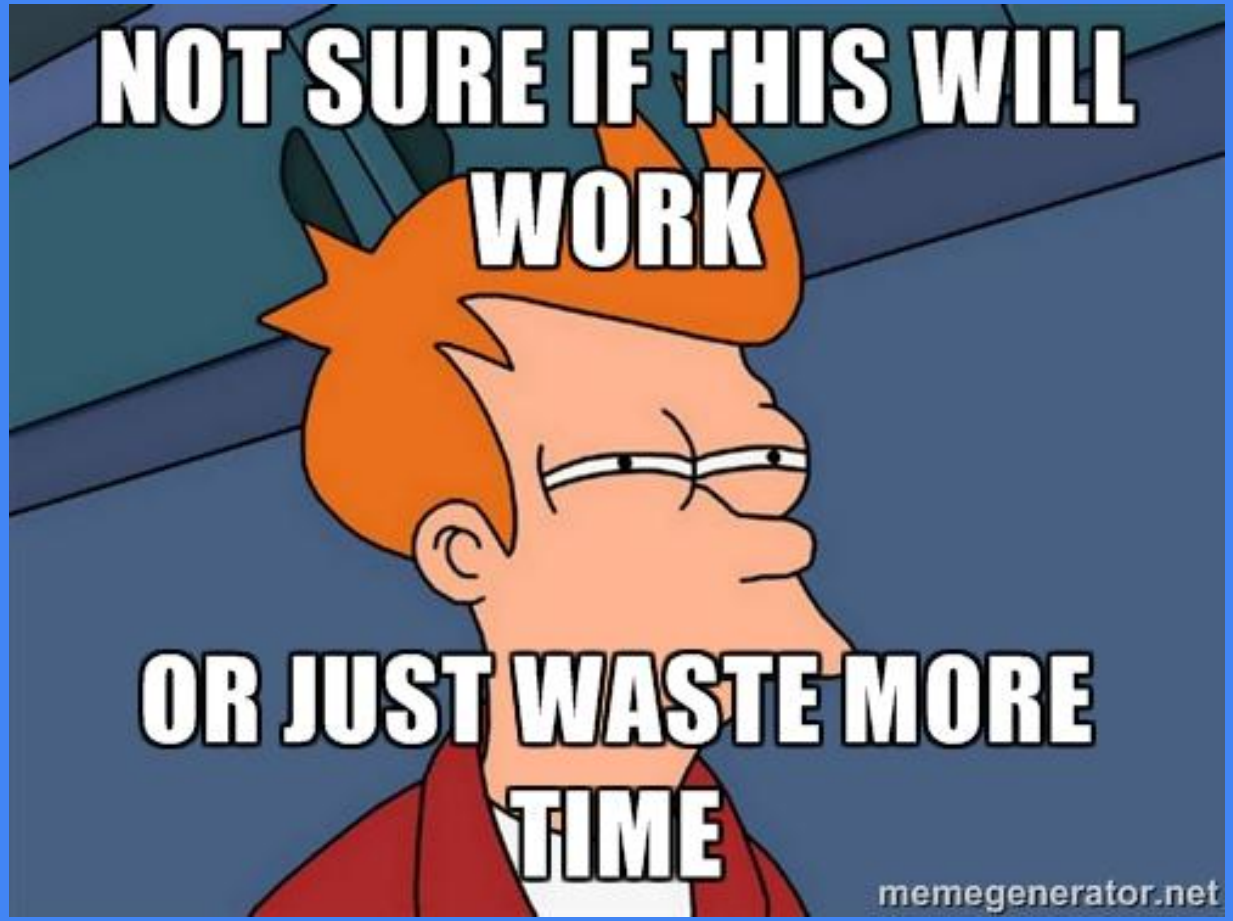
```python
import pandas as pd

# Read in the input file from finance
df = pd.read_excel("sample-sales.xlsx")

# Summarize sales data by customer
summary = (df.groupby("name")["ext price", "quantity"].sum().round(0)
             .sort_values(["ext price", "quantity"], ascending=[False, False]))

# Save to output file
summary.to_excel("report.xlsx")
```

- Create simple_report.py file
- Execute at the command line

report.xlsx

	A	B	C
1	name	ext price	quantity
2	Kulas Inc	137352	2265
3	White-Trantow	135842	2258
4	Trantow-Barrows	123381	2271
5	Jerde-Hilpert	112591	1999
6	Fritsch, Russel and Anderson	112215	2112
7	Barton LLC	109438	2041
8	Will LLC	104438	1828
9	Koepp Ltd	103661	1790
10	Frami, Hills and Schmidt	103570	1903
11	Keeling LLC	100934	1806
12	Kiehn-Spinka	99609	1756
13	Sanford and Sons	98823	1704
14	Stokes LLC	91536	1766
15	Kuhn-Gusikowski	91094	1665
16	Cronin, Oberbrunner and Spencer	89735	1673
17	Pollich LLC	87347	1707
18	Kassulke, Ondricka and Metz	86451	1647
19	Herman LLC	82865	1538
20	Purdy-Kunde	77898	1450
21	Halvorson, Crona and Champlin	70004	1284
22			

Pandas Demo


NOT SURE IF THIS WILL WORK
OR JUST WASTE MORE TIME
memegenerator.net

Python “program” is just text… And this is good

- Use version control
  - Safety net for changes
  - Bread crumbs to understand history
  - Back everything up
- Use comments
  - State business reasons
  - Links to Stack Overflow solutions
  - “Notes to your future self”
- Follows a process flow
  - Some hope of figuring it out in the future
  - Less chance of Excel-like spaghetti logic

chris1610 / pbpython

Code | Issues 0 | Pull requests 0 | Wiki | Pulse | Graphs

History for pbpython / code / advanced_excel.py

Commits on Dec 7, 2015

Determine table size based on the size of the incoming dataframe
chris1610 committed on Dec 7, 2015

Advanced excel code example
chris1610 committed on Dec 7, 2015

30 code/advanced_excel.py View

```
@@ -4,10 +4,13 @@
 """
 from __future__ import print_function
 import pandas as pd
+from xlsxwriter.utility import xl_rowcol_to_cell


-def format_excel(writer):
+def format_excel(writer, df_size):
     """ Add Excel specific formatting to the workbook
+    df_size is a tuple representing the size of the dataframe - typically called
+    by df.shape -> (20,3)
     """
     # Get the workbook and the summary sheet so we can add the formatting
     workbook = writer.book
@@ -16,15 +19,20 @@ def format_excel(writer):
     money_fmt = workbook.add_format({'num_format': 42, 'align': 'center'})
     worksheet.set_column('A:A', 20)
     worksheet.set_column('B:C', 15, money_fmt)
-    worksheet.add_table('A1:C22', {'columns': [{'header': 'account',
-                                                'total_string': 'Total'},
-                                               {'header': 'Total Sales',
-                                                'total_function': 'sum'},
-                                               {'header': 'Average Sales',
-                                                'total_function': 'average'}],
-                                   'autofilter': False,
-                                   'total_row': True,
-                                   'style': 'Table Style Medium 20'})
+    # Add 1 to row so we can include a total
+    # subtract 1 from the column to handle because we don't care about index
+    table_end = xl_rowcol_to_cell(df_size[0] + 1, df_size[1] - 1)
+    # This assumes we start in the left hand corner
+    table_range = 'A1:{}'.format(table_end)
```

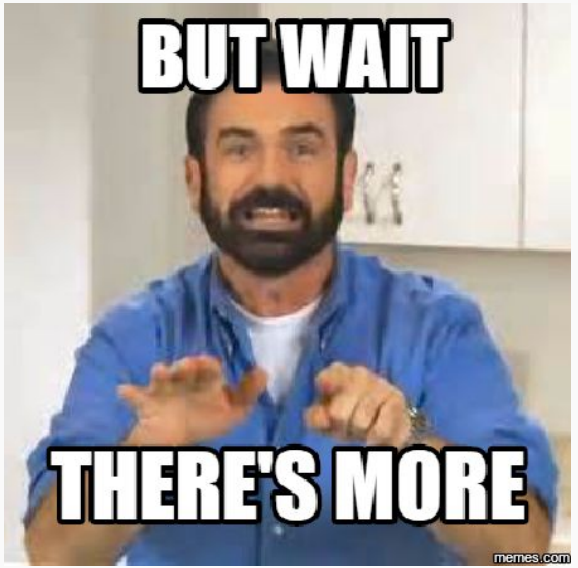
https://twitter.

Pandas - It Slices! It Dices!

- Easy to get data into and out of a DataFrame
  - (Excel, CSV, SQL, json, HDF, HTML, Latex, msgpack)
- Optimized and designed for speed
  - Easy to read/write multi-MB files
  - Portions compiled to C for improved performance
  - Vectorized functions (look mom no loops)!
- Scaleable as your data grows
- Handles missing data well
- Supports complexing merging and joining
- Excellent time-series support

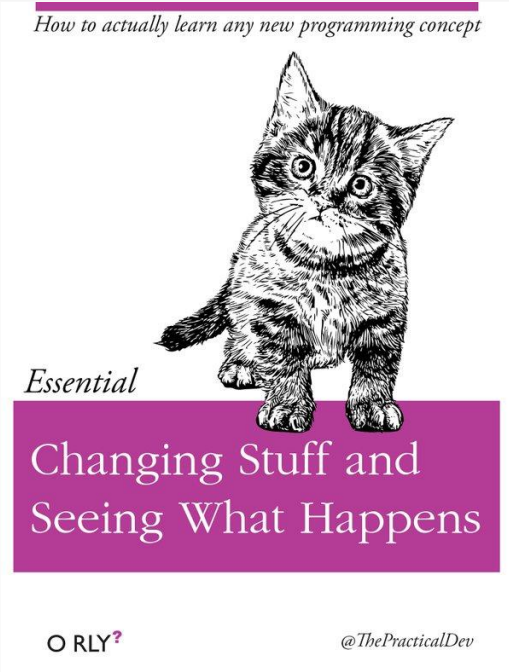
Python

- Generally regarded as easy to learn
- Rich ecosystem
  - (web, db, text, science, system admin and more)
- Runs well on windows
- Mature, widely adopted language but continues to evolve
- “Glue Language” - handy for data manipulation
  - Useful for moving, renaming files, web scraping etc.



There are other benefits too

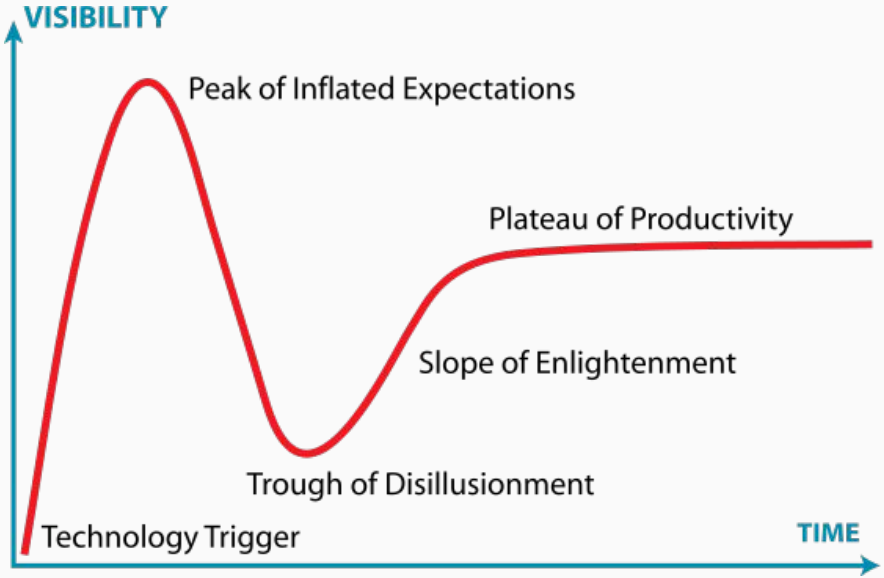
- Learn how to think about data
  - Stack, unstack, melt, tidy data
- Learn as little or as much as you need
  - Could do 10-20 line scripts or build 100+ lines
- Build a library of components
  - Leverage small wins for bigger success
- New skill



https://twitter.com/ThePracticalDev/status/720257210161311744

Despite all the upside, there are challenges

- Need to learn a new language
  - takes time, slower than doing it the old way
- Deploying solution to others
- Formatting of Excel
- Mental gymnastics to learn new paradigm



But it is worth it!

Next Steps & Resources

Python Notes:

- python 3.5+
- Anaconda or miniconda for your environment

Start with a decent text editor

- Sublime, Atom, etc.

Use Version Control!

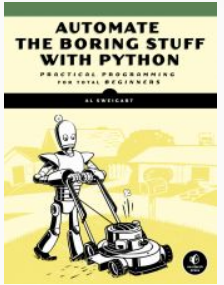
- git, hg or even svn (if you have to)

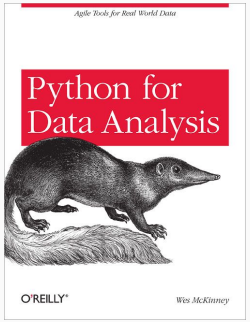
Resources:

Automate The Boring Stuff with Python

Practical Business Python :)

Pandas





Thank you!

Chris Moffitt <chris@moffitts.net>
@chris1610
pbpython.com