RawQuant

This guide describes the install and use of RawQuant.

Installation

We are starting with a fresh system here. We recommend uninstalling MSFileReader and Python if they are already installed, as the order of installation is important for proper functioning.

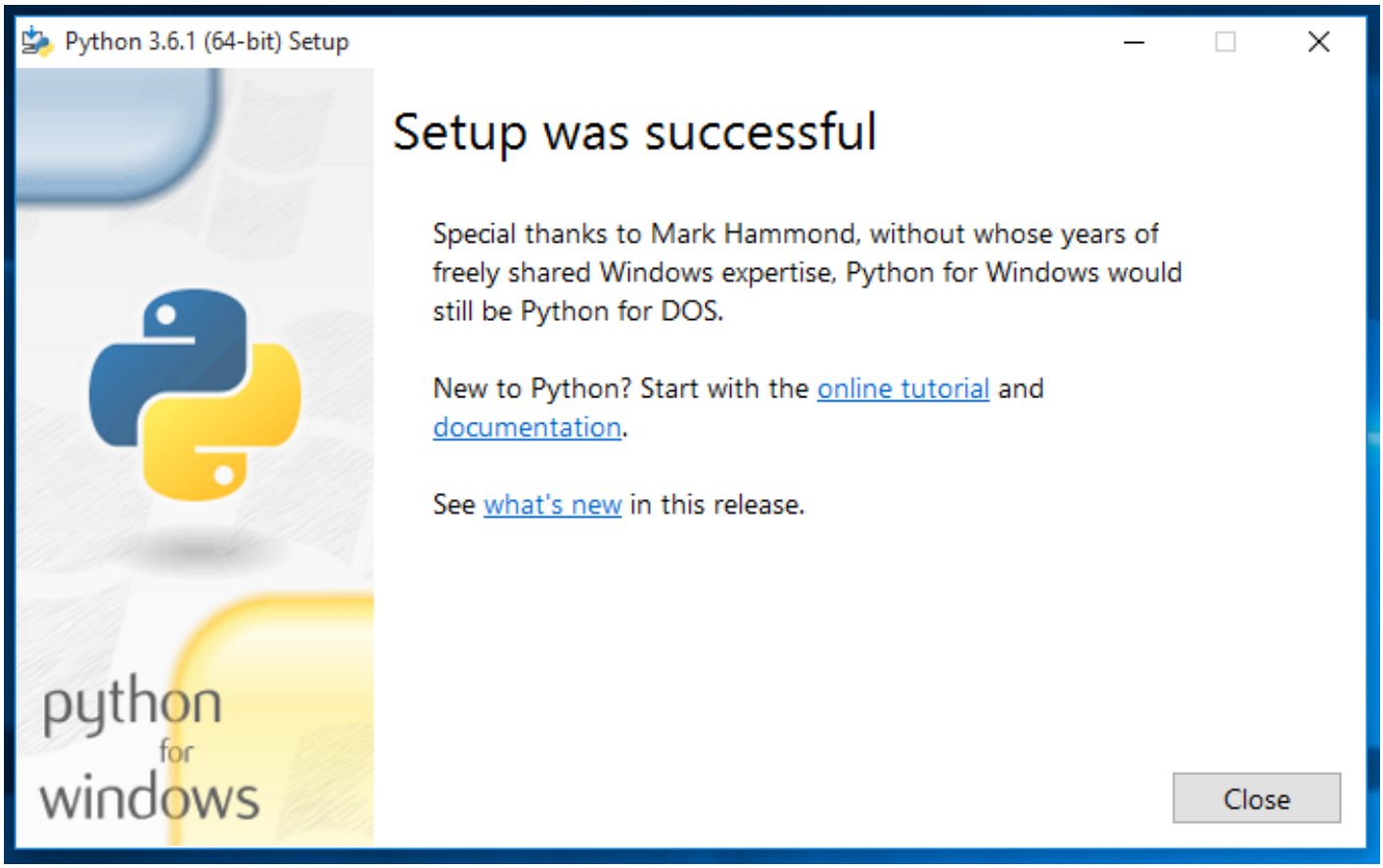
RawQuant does not have any specific system requirements, although the script may run slowly on older machines. The computer we are using in this walkthrough has the specs:

> - Microsoft Windows 10 Enterprise 2015 LTSB
> - 10.0.10240 Build 10240
> - Optiplex 7040
> - x64-based PC
> - Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz, 3408 Mhz, 4 Core(s), 8 Logical Processor(s)
> - Installed Physical Memory (RAM) 8.00 GB

Let's walk through the setup:

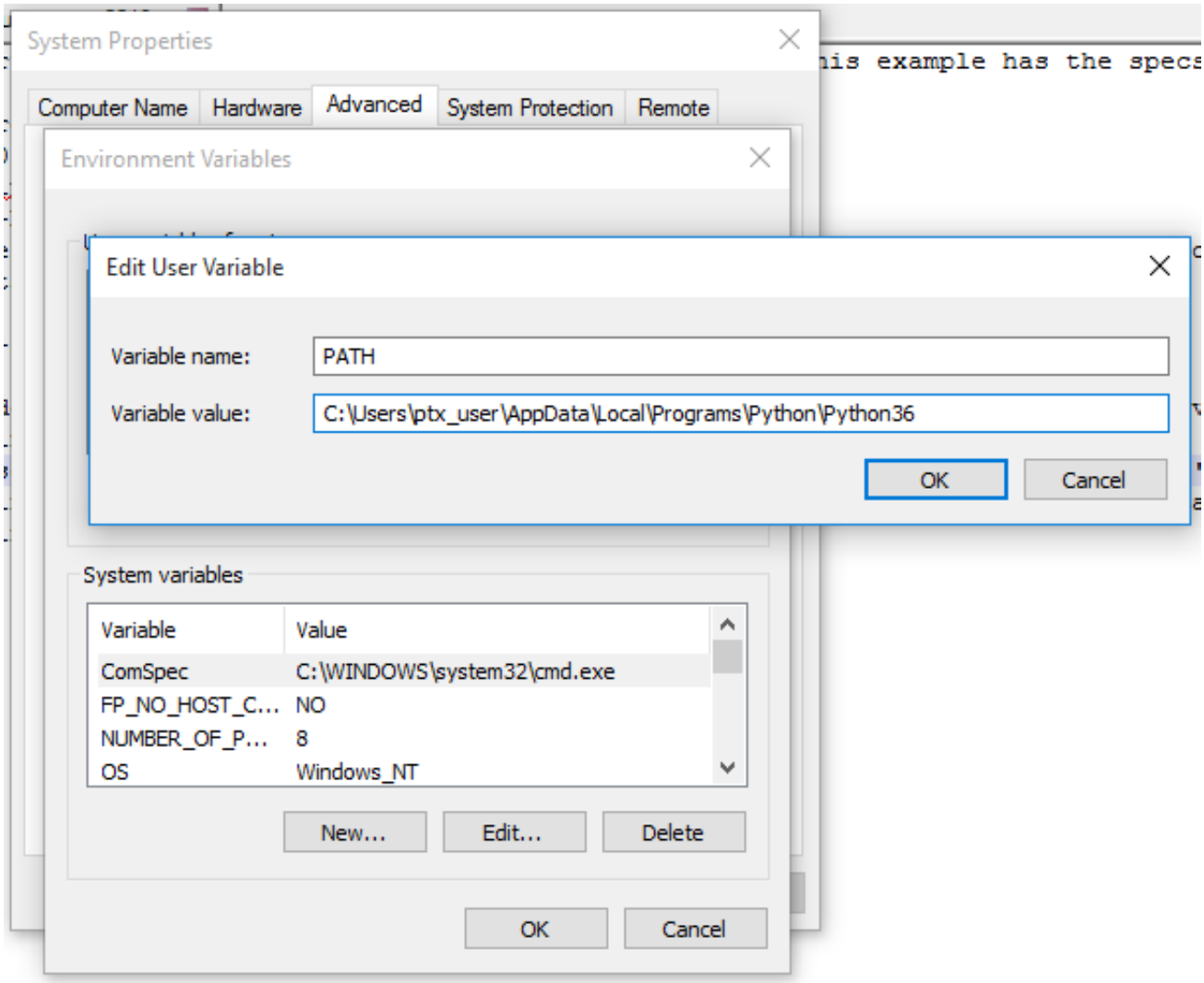
1. Download Python.
    - The version here is important. We have currently only tested version 3.6.1 (64-bit).
    - The install file can be downloaded from Python directly (https://www.python.org/downloads/release/python-361/).

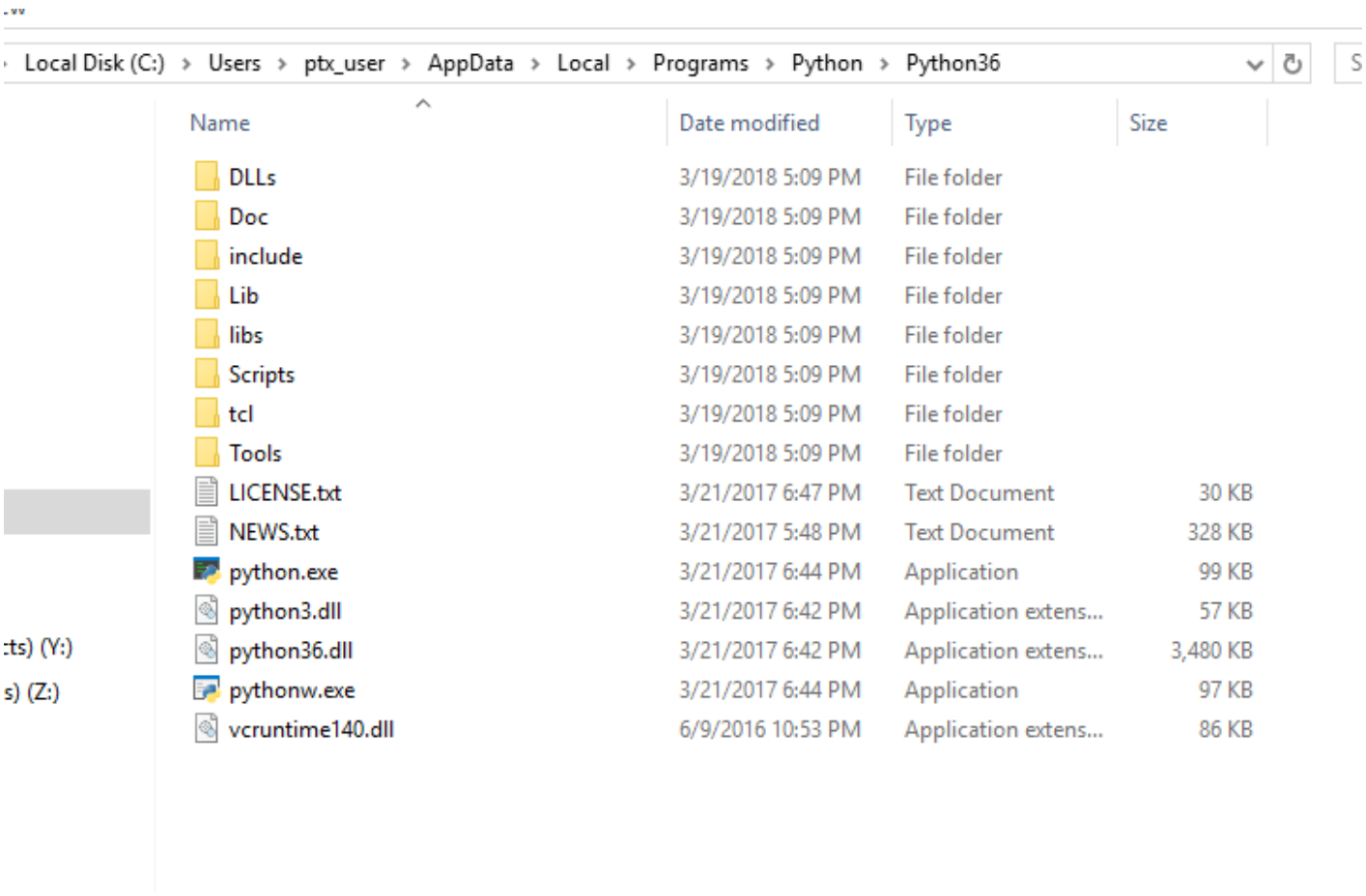
2. Install Python. Use the default "Install Now" option.



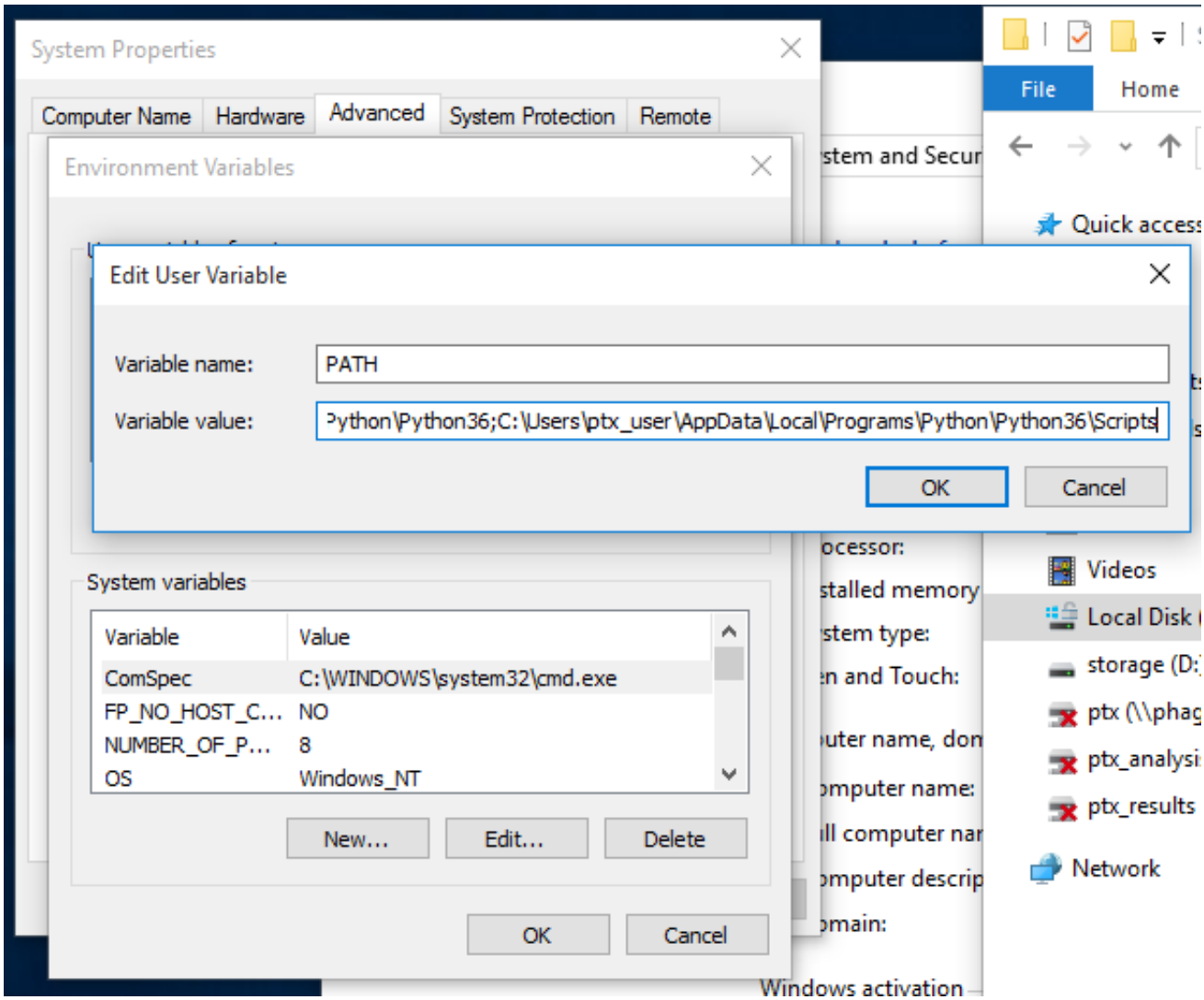
3. Setup Python in the environment variables.

    - Go to 'Control Panel'.
    - Click 'System and Security'.
    - Click 'System'.
    - Click 'Advanced System Settings'.
    - Click 'Environment Variables'.
    - On PATH, click 'Edit'.
    - Add the directory of the Python install to the PATH variable.


his example has the specs
System Properties
Computer Name
Hardware
Advanced
System Protection
Remote
Environment Variables
Edit User Variable
Variable name:
PATH
Variable value:
C:\Users\ptx_user\AppData\Local\Programs\Python\Python36
OK
Cancel
System variables
Variable
Value
ComSpec
C:\WINDOWS\system32\cmd.exe
FP_NO_HOST_C...
NO
NUMBER_OF_P...
8
OS
Windows_NT
New...
Edit...
Delete
OK
Cancel

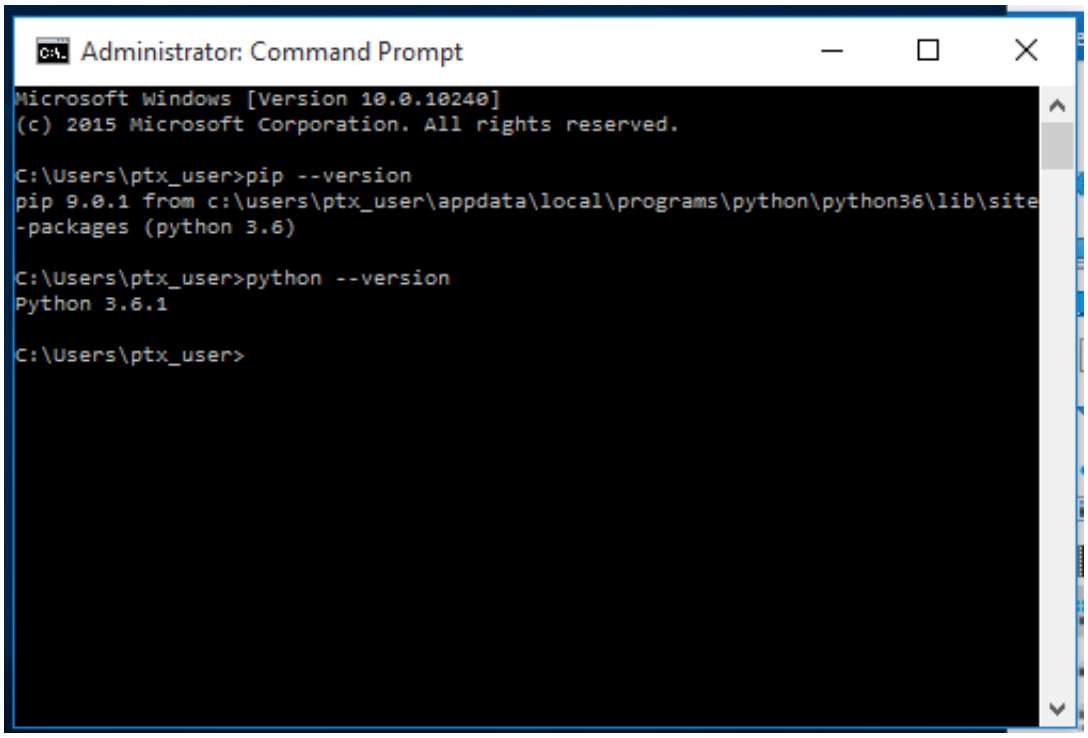


4. Add pip to the environment variables as well using the same process.

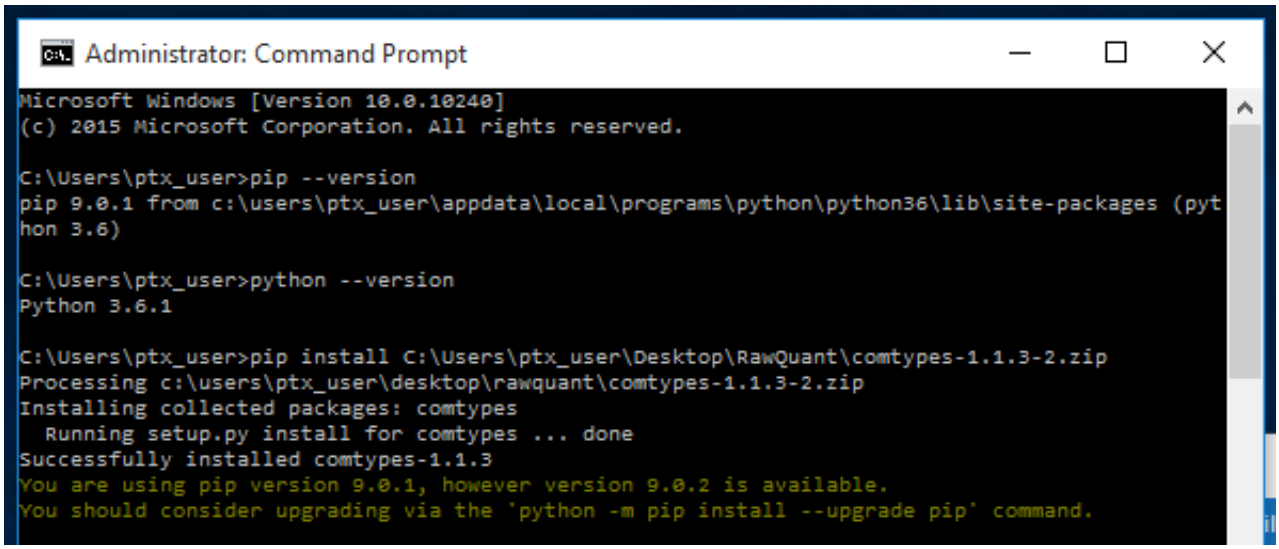


5. Check that Python and pip are installed correctly.

    - Open 'Command Prompt'.
    - Type 'python --version'.
    - Type 'pip --version'.



6. Install the comtypes package.
    - The version here is important (want version 1.1.3).
    - The install file can be downloaded from Python directly (https://pypi.python.org/pypi/comtypes/1.1.3).
    - Open 'Command Prompt' if not already open.
    - Type 'pip install 'path to comtypes.zip file'.
    - Restart your computer.

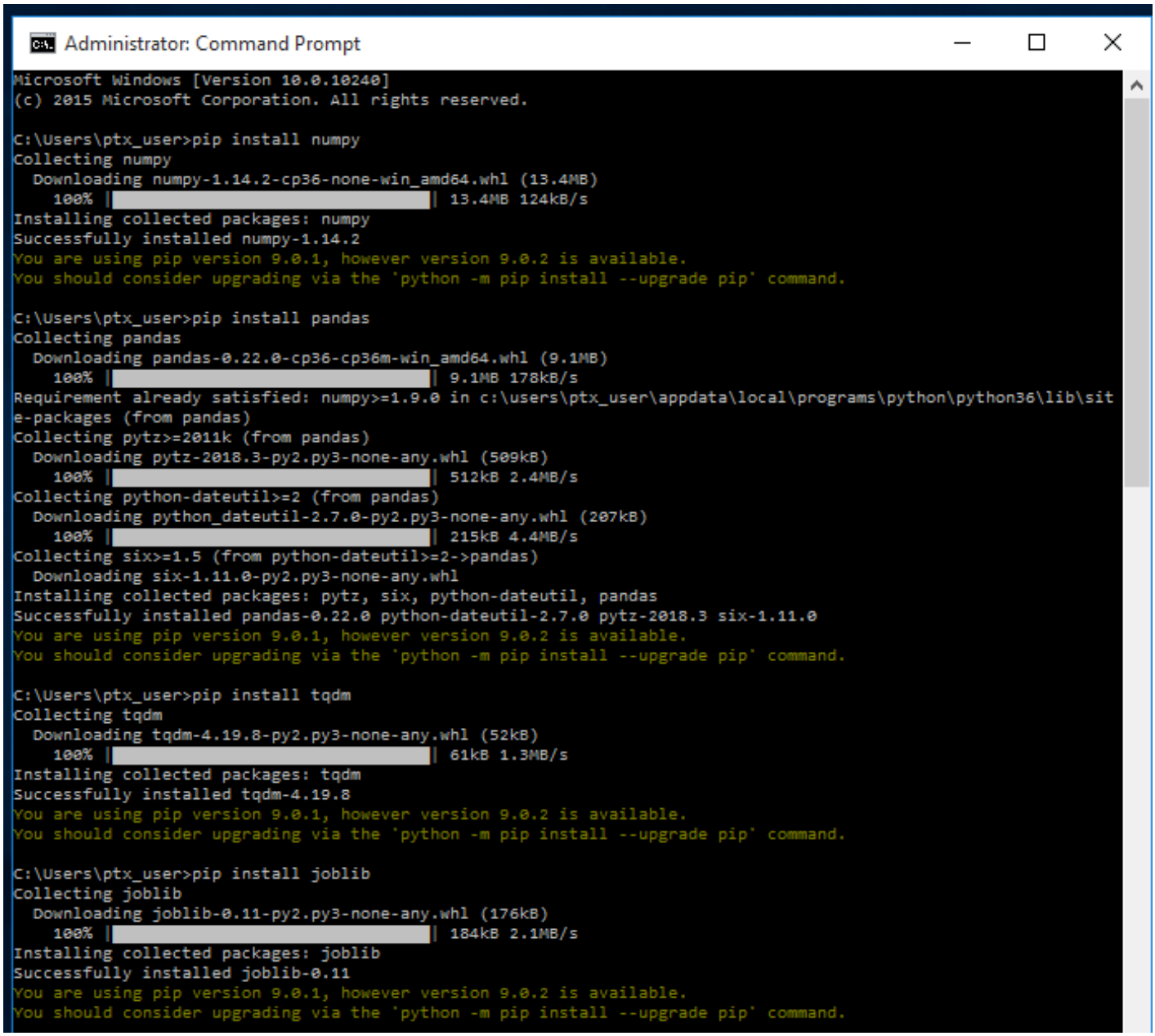


7. Install MSFileReader version 3.0 SP2, 64-bit only (do not install the 32-bit version!).
    - The version here is important. We have currently only tested the mentioned version.

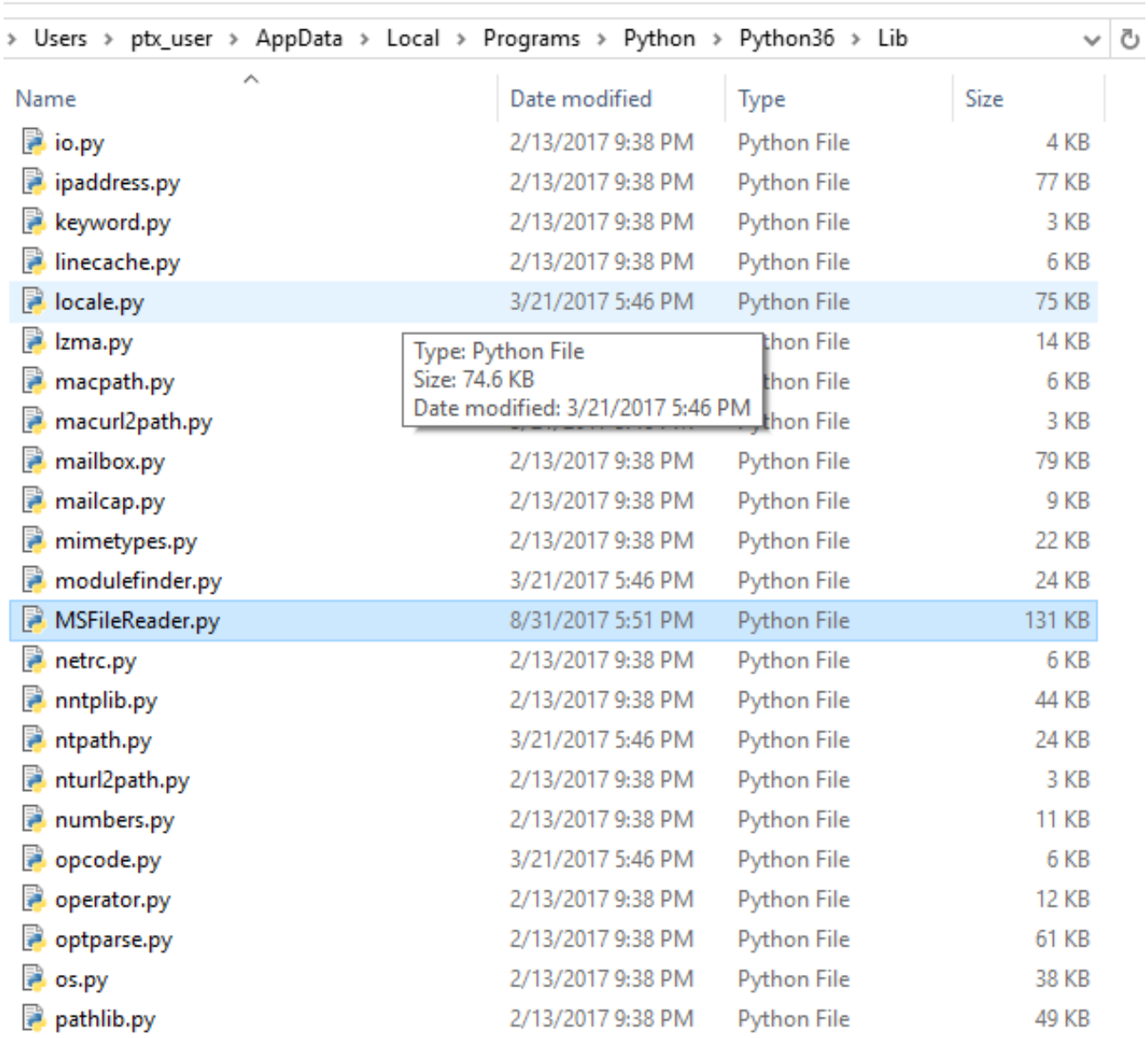
- The install file can be downloaded from Thermo (https://thermo.flexnetoperations.com/control/thmo/login?nextURL=%2Fcontrol%2Fthmo%2Fdownload%3Felement%3D6306677). This website requires registration, which is free.
- Alternatively, the install file can be downloaded from the RawQuant GitHub page (https://github.com/kevinkovalchik/RawQuant).
- After install, restart your computer.

8. Install the other Python dependencies.

    - Open 'Command Prompt'.
    - Install dependencies using the command 'pip install 'package name'.
    - Repeat for packages numpy, pandas, tqdm, joblib, argparse.



9. Copy the MSFileReader.py file from the Python bindings to the Python directory.
    - The MSFileReader.py file can be downloaded from the RawQuant GitHub page (https://github.com/kevinkovalchik/RawQuant).
    - Copy and paste in the appropriate directory. This will be the Lib directory of the Python you mapped to the environment previously.



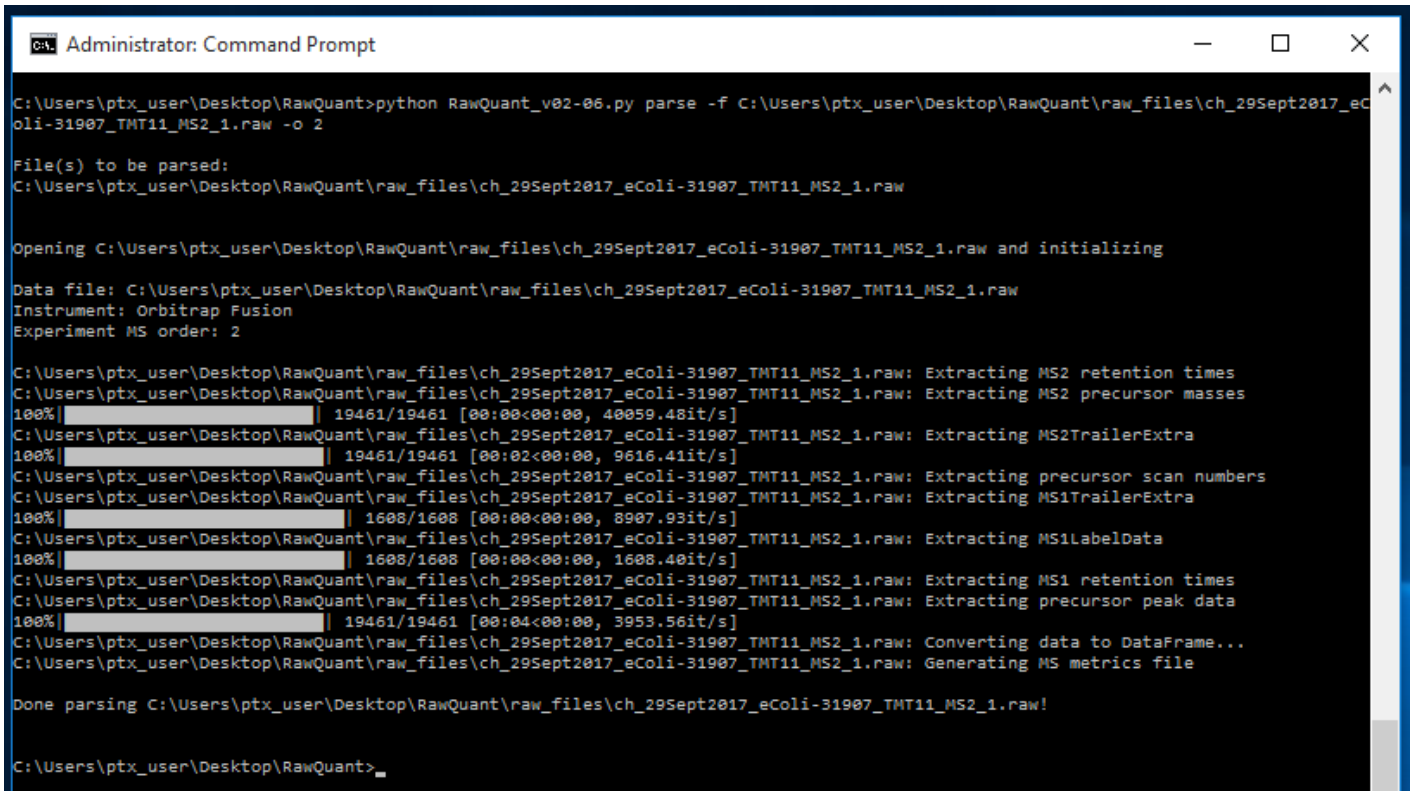
10. Check if RawQuant works.

    - Open 'Command Prompt'.
    - Navigate to the directory where you have stored the RawQuant.py script.
    - Make sure you have admin priveleges. The first time you run RawQuant, the comtypes package needs to build a couple of sub-libraries. After you have run it the first time, you no longer need admin priveleges. In the image below, note the comtypes 'generating' lines. This will only happen the first time you run RawQuant with admin.
    - Enter the command below into the terminal and hit enter.

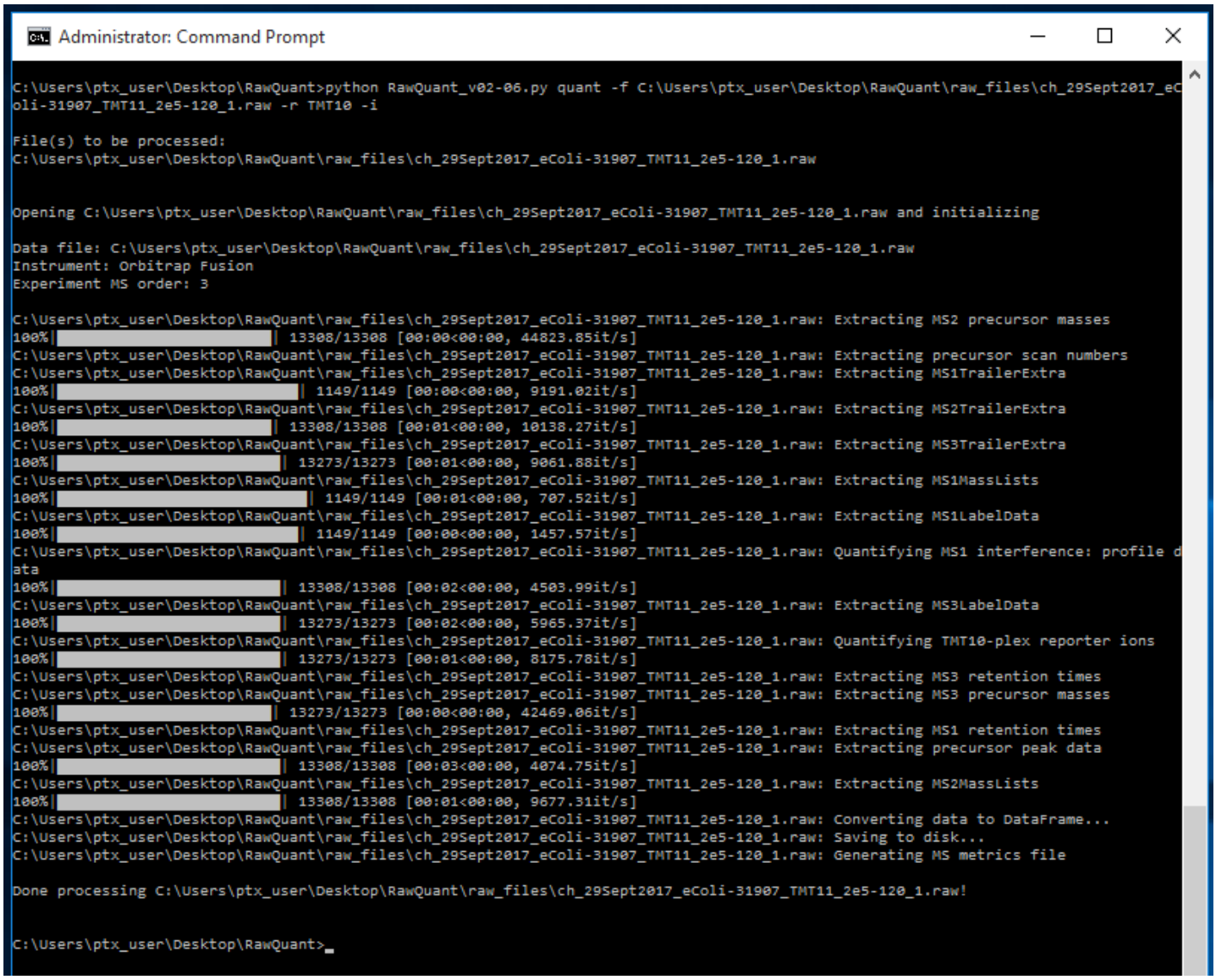
```
python RawQuant.py -h
```

11. Test RawQuant on a real raw file to see if it is working completely.
    - Download a test raw file from the EBI PRIDE repository for RawQuant (https://www.ebi.ac.uk/pride/archive/projects/PXD008787) or use your own raw file.
    - Test the raw file (in this example I tested an MS2 and SPS-MS3 file).

```
python RawQuant.py parse -f <path to your raw file> -o 2
```




```
python RawQuant.py quant -f <path to your raw file> -r TMT10 -i
```



12. If everything has worked properly, the commands should finish without error, and text output files should be generated in the same directory where the processed raw file(s) is located.

Processing Files with RawQuant

This section will walk through some common commands and input scenarios for RawQuant as well as discussing some of the output data.

1. Download a test raw file from the EBI PRIDE repository for RawQuant (https://www.ebi.ac.uk/pride/archive/projects/PXD008787) or use your own raw file.
2. Navigate to the directory where you have stored the RawQuant.py script
3. To invoke the general help for RawQuant use the command:

```
python RawQuant.py -h
```

4. To invoke the help for the specific parse and quant functions, use extensions of the previous command.

```
python RawQuant.py parse -h

python RawQuant.py quant -h
```

5. These help boxes are worth reading and cover a lot of the basic usage and commands with RawQuant.

6. If you are ever confused about input files for things like 'custom reagents', use the examples functionality of RawQuant to generate some files for you to use as a guide.

```
python RawQuant.py examples -h
```

7. Let's perform our own analysis. I am going to use a file acquired on an Orbitrap Fusion downloaded from the above PRIDE repository that was acquired as MS1-Orbitrap HCD-MS2-Orbitrap. The file name is "ch 29Sept2017 eColi-31907 TMT11 MS2 1.raw" if you would like to follow along with the same file.

8. First lets process this file using the 'parse' functionality of RawQuant. There are multiple flags that the parse command can use as input.

    - '-f' this is the input file, or a list of files separated by spaces.
    - '-m' this can be used to specify a text file that contains multiple input files to be processed. One per line.
    - '-o' this specifies the MS orders to be parsed. Can be one number (e.g. -o 2) or a list separated by spaces (e.g. -o 1 2 3). If -o is set to 0, no parsing will be done. This last functionality is potentially desirable if you are looking to only generate an MGF output. If you input a list of values that contains 0 (e.g. 0 1 2), no parsing will be done.
    - '-mgf' this flag will trigger MGF generation.
    - '-spd' this will suppress the progress bar during processing. But this is cool to look at, so why would you do that?

```
python RawQuant.py parse -h
```

9. I am interested in generating parsed output for MS1 and MS2 scans in my raw file, as well as creating an MGF output I can use in a database search.

```
python RawQuant.py parse -f C:\Users\ptx_user\Desktop\RawQuant\ch_29Sept2017_eCol
i-31907_TMT11_MS2_1.raw -o 1 2 -mgf
```

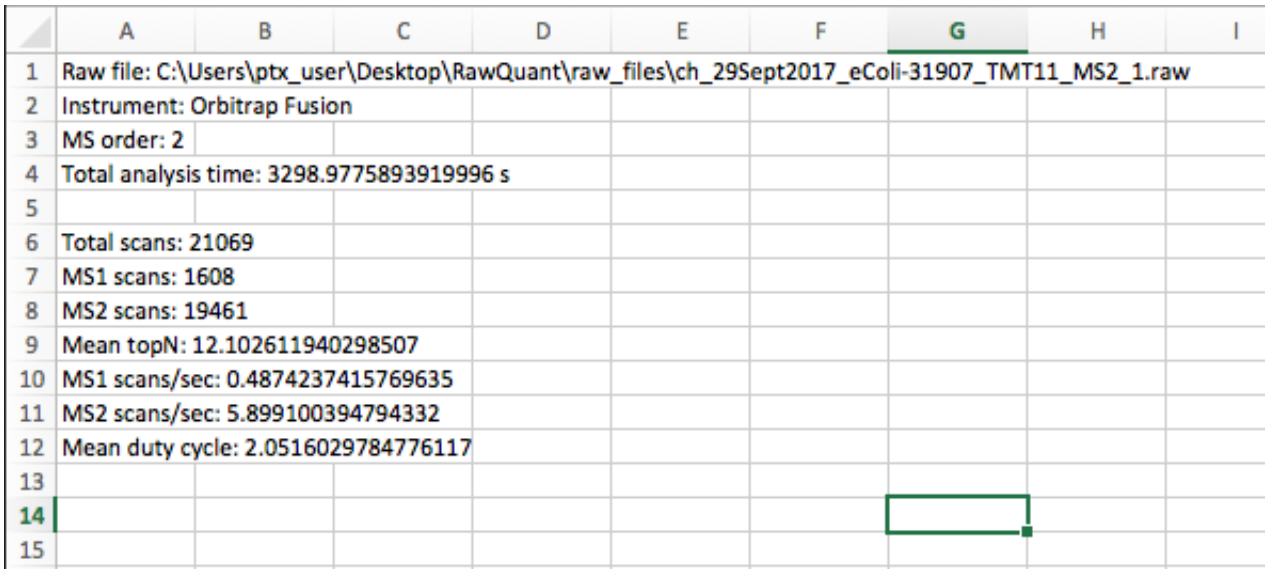
10. Once processing is complete, you should see four files created in the same directory as the raw file.

- a 'metrics.txt' file - this contains some general metrics of the raw file, such as number of MS1 scans, topN rate, duty cycle.
- 'MS1ParseData.txt' and 'MS2ParseData.txt' files - these contain the metrics for every MS1 and MS2 scan found in the raw file.
- 'MGF.mgf' file - this is a standard MGF output. Use with any search engine who will take MGF input!

11. Let's look at the output files a little closer at each of the files. They are tab delimited text files, so they can be opened in most editors and Excel.

12. Let's look at the 'metrics.txt' file first. This contains some useful 'quick-glance' information.

    - The MS order value denotes the level where MS2 data was acquired.
    - Total scans denotes the total number of scans in the entire file.
    - MS1 scans counts the number of MS1 scans.
    - MS2 scans counts the number of MS2 scans.
    - Mean topN is the average number of MS2 events triggered in an MS1 scan across the entire run.
    - MS1 scans/sec counts the number of MS1 scans that occurs every second.
    - MS2 scans/sec counts the number of MS2 events that are triggered every second.
    - Mean duty cycle calculates the time (in seconds) between neighbouring MS1 events.



	A	B	C	D	E	F	G	H	I
1	Raw file: C:\Users\ptx_user\Desktop\RawQuant\raw_files\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw								
2	Instrument: Orbitrap Fusion								
3	MS order: 2								
4	Total analysis time: 3298.9775893919996 s								
5									
6	Total scans: 21069								
7	MS1 scans: 1608								
8	MS2 scans: 19461								
9	Mean topN: 12.102611940298507								
10	MS1 scans/sec: 0.4874237415769635								
11	MS2 scans/sec: 5.899100394794332								
12	Mean duty cycle: 2.0516029784776117								
13									
14									
15									

13. What about the 'MS1ParseData.txt' file.

    - ScanNumber is the scan index for the MS1 event.
    - RetentionTime is the retention time (in minutes) of the MS1 event.
    - MS1IonInjectionTime is the injection time (in ms) for the MS1 event.

	A	B	C
1	ScanNumber	RetentionTime	MS1IonInjectionTime
2	1	5.015732722	50
3	2	5.025169948	50
4	3	5.03378067	50
5	4	5.042388411	50
6	5	5.050980166	50
7	6	5.059625992	50
8	7	5.068215525	50
9	8	5.076823933	50
10	9	5.085415218	50
11	10	5.094060965	50
12	11	5.102648622	50
13	12	5.111254428	50

14. What about the 'MS2ParseData.txt' file. This contains a lot of information.

    - ScanNumber is the scan index for the MS2 event.
    - MS1ScanNumber is the MS1 event that the MS2 scan was triggered from.
    - RetentionTime is the retention time (in minutes) where the MS2 scan was acquired.
    - PickedRetentionTime is the retention time (in minutes) where the MS2 ion was targeted for acquisition (so, this is the retention time of the MS1 scan event that it was triggered from).
    - PeakMaxRetentionTime is the retention time (in minutes) where the maximum intensity for the parent ion selected for MS2 analysis was observed.
    - PrecursorMass is the m/z of the parent ion selected for MS2.
    - PrecursorCharge is the determined charge state of the parent ion selected for MS2.
    - PrecursorPickedIntensity is the intensity of the precursor ion in the MS1 scan closest to where the MS2 scan was carried out (just prior).
    - PrecursorMaxIntensity is the maximum observed intensity across the elution profile for the parent ion selected for MS2.
    - PrecursorArea is the detected area across the elution profile for the parent ion selected for MS2.
    - PrecursorRetentionWidth is the width of the elution profile for the parent ion.
    - MS1IonInjectionTime is the injection time (in ms) of the MS1 scan where the MS2 event was triggered from.
    - MS2IonInjectionTime is the injection time (in ms) of the MS2 scan event itself.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N
1	ScanNumber	MS1ScanNumber	RetentionTime	PickedRetentionTime	PeakMaxRetentionTime	PrecursorMass	PrecursorCharge	PrecursorPickedIntensity	PrecursorMaxIntensity	PrecursorArea	PrecursorRetentionWidth	MS1IonInjectionTime	MS2IonInjectionTime	
2	34	32	5.293206915	5.283369497	9.79381141	468.2688599	2	25122.48047	7056801	6586402.954	4.443821546	50	120	
3	237	235	7.030103957	7.020372532	7.037566735	488.7542419	2	26411.96094	33617.26172	26455.83265	0.113278116	50	120	
4	240	238	7.047782943	7.037566735	7.037566735	492.246521	2	34252.85547	34252.85547	17170.16279	0.113278116	50	120	
5	247	245	7.099278101	7.089553194	7.133650648	393.1958313	2	51641.33594	115019.9297	26928.8931	1.156756263	50	120	
6	249	246	7.107936452	7.098117178	7.106744339	1069.376587	2	26439.25977	40328.30859	2483.47631	0.07029287	48.24	120	
7	251	248	7.116915452	7.106744339	7.106744339	1060.864014	2	25825.69141	25825.69141	1159.519452	0.044097454	50	120	
8	255	253	7.143425576	7.133650648	7.133650648	397.704895	2	26665.55859	26665.55859	3454.509599	0.331429871	50	120	
9	259	257	7.169616995	7.159846064	7.185995858	435.7221375	2	39923.82422	69542.22656	6586.712845	0.287332417	50	120	
10	263	261	7.19582499	7.185995858	7.185995858	396.2137146	2	25213.27148	25213.27148	2171.220385	0.138244594	50	120	
11	277	275	7.307880261	7.298090658	7.342151399	432.2244263	2	28323.36328	115239.6563	9040.96475	0.174531667	50	120	
12	280	278	7.325677897	7.315345508	7.342151399	401.7400208	2	42129.15625	102286.0625	7407.271223	0.386069976	50	120	
13	282	279	7.334340381	7.32448752	7.32448752	417.7348938	2	44769.65625	44769.65625	3771.25194	0.079185755	50	120	
14	285	283	7.35225474	7.342151399	7.377276413	442.2289734	2	39716.11328	119454.2422	10181.13876	0.48706664	50	120	
15	289	287	7.378416963	7.368675395	7.377276413	410.229248	2	31886.85547	35782.24609	2402.496794	0.078831666	50	120	
16	291	288	7.387068927	7.377276413	7.377276413	524.2188721	2	34930.19531	34930.19531	3000.629968	0.10394693	47.208	120	
17	297	295	7.430583327	7.420983065	7.472622325	483.7930908	2	25868.99219	30958.18359	4313.071957	0.300078099	50	120	
18	304	302	7.482224432	7.472622325	7.498751875	395.859375	3	30531.18945	56206.0625	3751.957454	0.130088506	50	120	
19	306	303	7.490951319	7.481034416	7.551071571	554.2745972	2	27262.17773	110067.6953	12722.5249	0.329789823	50	120	

15. Let's move on to processing a quant file. For this example I am going to use a file that was obtained using an SPS-MS3 approach on an Orbitrap Fusion from the above PRIDE repository that was acquired as . The file name is "ch 29Sept2017 eColi-31907 TMT11 2e5-120 1.raw" if you would like to follow along with the same file.

16. First lets process this file using the 'quant' functionality of RawQuant. There are multiple flags that the quant command can use as input.

    - '-f' this is the input file, or a list of files separated by spaces.
    - '-m' this can be used to specify a text file that contains multiple input files to be processed. One per line.
    - '-r' this is the labeling reagents. Built-in options are TMT0, TMT2, TMT6, TMT10, TMT11, iTRAQ4, and iTRAQ8.
    - '-cr' this is if you want to use custom reagents. Should be a csv file containing user-defined labels and masses. To see an example, use the command 'python RawQuant.py examples -r'.
    - '-i' this will trigger quantification of isolation interference.
    - '-o' this specifies the MS order where the quantification values are. Can be one number (e.g. -o 2). This flag is optional. If not input, the highest MS order will be assumed. Possible values are 2 and 3.
    - '-mgf' this flag will trigger MGF generation.
    - '-spd' this will suppress the progress bar during processing.
    - '-c' this will correct for isotopic impurities in the reagents. Input should be a csv file containing an impurity matrix. For an example, try 'python RawQuant.py examples -c'.

```
python RawQuant.py quant -h
```

17. I am interested in generating quant output for MS3 scans in my raw file, as well as creating an MGF output I can use in a database search.

```
python RawQuant.py quant -f C:\Users\ptx_user\Desktop\RawQuant\ch_29Sept2017_eCol
i-31907_TMT11_2e5-120_1.raw -r TMT10 -i -mgf
```

18. Once processing is complete, you should see three files created in the same directory as the raw file.

- a 'metrics.txt' file - this contains some general metrics of the raw file, such as number of MS1 scans, topN rate, duty cycle, just as before.
- 'QuantData.txt' - this contains the quantification and scan data for the MS3 events.
- 'MGF.mgf' file - this is a standard MGF output. Use with any search engine who will take MGF input!

19. Let's look a little closer at the 'QuantData.txt' file. It contains a lot of information.

    - ScanNumber is the scan index for the MS3 event.
    - MS2ScanNumber is the MS2 event that the MS3 scan was triggered from.
    - MS1ScanNumber is the MS1 event that the MS2 scan that led to the MS3 was triggered from.
    - RetentionTime is the retention time (in minutes) where the MS3 scan was acquired.
    - PickedRetentionTime is the retention time (in minutes) where the MS2 ion was targeted for acquisition (so, this is the retention time of the MS1 scan event that it was triggered from).
    - PeakMaxRetentionTime is the retention time (in minutes) where the maximum intensity for the parent ion selected for MS2 analysis was observed.
    - PrecursorMass is the m/z of the parent ion selected for MS2.
    - PrecursorCharge is the determined charge state of the parent ion selected for MS2.
    - PrecursorPickedIntensity is the intensity of the precursor ion in the MS1 scan closest to where the MS2 scan was carried out (just prior).
    - PrecursorMaxIntensity is the maximum observed intensity across the elution profile for the parent ion selected for MS2.
    - PrecursorArea is the detected area across the elution profile for the parent ion selected for MS2. Note, this is not done by curve fitting.
    - PrecursorRetentionWidth is the width of the elution profile for the parent ion.
    - MS1IonInjectionTime is the injection time (in ms) of the MS1 scan where the MS2 event was triggered from.
    - MS2IonInjectionTime is the injection time (in ms) of the MS2 scan event.
    - MS3IonInjectionTime is the injection time (in ms) of the MS3 scan event.
    - MS1Interference in the amount of signal in the isolation window that is not from the selected precursor (represented as a percent).
    - tmt126_mass is the observed mass for the specified TMT reporter ion.
    - tmt126_ppm is the error (in ppm) of the observed mass vs. the expected reporter ion mass.
    - tmt126_intensity is the observed intensity of the reporter ion.
    - tmt126_res is the observed resolution of the reporter ion.
    - tmt126_bl is the observed background signal of the reporter ion.
    - tmt126_noise is the Orbitrap pre-amplifier noise for the reporter ion.
    - SPSMass are the masses of the ions selected from the MS2 scan for SPS-MS3 analysis.
    - SPSIntensity are the intensities of the selected SPS ions.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
1	ScanNumber	MS2ScanNur	MS1ScanNur	RetentionTin	PickedReten	PeakMaxRet	PrecursorMa	PrecursorCh	PrecursorPic	PrecursorMa	PrecursorAre	PrecursorRet	MS1IonInjec	MS2IonInjec	MS3IonInjec	MS1Interfere	tmt126_mas
2	57	56	54	5.48424114	5.4729598	6.89496049	468.26825	2	23616.123	5000482.5	2739899.99	4.18149333	50	30	120	9.15882035	
3	82	81	79	5.68260153	5.67128484	7.09388619	413.265594	3	6689.83496	7812.10547	13591.3589	2.36984979	50	30	120	32.27341	
4	91	90	88	5.74340678	5.73209012	5.73209012	577.125549	2	8374.66895	8374.66895	12535.4235	2.4989167	50	30	120	0	
5	114	113	111	5.92477981	5.91321093	5.91321093	714.777893	3	6326.43555	6326.43555	3606.85647	0.5853997	50	30	120	36.9506253	
6	184	183	181	6.50994822	6.49861063	8.01217823	577.125916	2	5047.13965	116630.578	14114.3382	2.51340242	50	30	120	0	
7	195	194	192	6.58795992	6.57659754	6.57659754	511.471405	2	5666.28467	5666.28467	4417.83409	1.50808394	50	30	120	19.1078254	
8	234	233	231	6.90616034	6.89496049	8.01217823	424.242279	2	16204.3408	45084.0703	22590.1322	2.51340242	50	30	120	16.7219486	
9	246	244	242	6.98689164	6.97247012	6.97247012	988.852722	2	8092.18213	8092.18213	586.615517	0.01925011	50	30	120	27.2920719	
10	247	245	242	6.98727101	6.97247012	6.99172022	409.722961	2	5047.68262	5148.72266	518.280027	0.01925011	50	30	120	35.8528552	
11	250	249	243	6.99457474	6.98104374	6.98104374	979.838684	2	5026.18262	5026.18262	318.587757	0.00857363	50	30	120	0	
12	254	252	248	7.00664528	6.99172022	7.03766308	488.755035	2	12851.1865	15406.7031	1143.52273	0.1218285	50	30	120	0	
13	255	253	248	7.00699357	6.99172022	7.03766308	492.246277	2	7117.10547	17060.4453	874.683432	0.07380338	50	30	120	15.3221504	
14	262	261	259	7.04889443	7.03766308	7.03766308	402.730042	2	5159.43359	5159.43359	298.759062	0	50	30	120	0	
15	266	265	263	7.06667539	7.05484712	7.05484712	404.869446	3	6196.87109	6196.87109	212.048616	0.02624854	50	30	120	22.291366	
16	270	268	264	7.07837795	7.06391162	7.09388619	1062.36926	2	10106.7588	12180.3691	593.036486	0.07505319	50	30	120	0	
17	271	269	264	7.07871331	7.06391162	7.06391162	1053.85754	2	9472.40137	9472.40137	197.655623	0.01775068	50	30	120	0	
18	275	273	267	7.08906557	7.0725978	7.10287224	1069.37793	2	10047.9229	69342.0156	2636.58477	0.06624225	50	30	120	0	
19	276	274	267	7.08943625	7.0725978	7.09388619	1060.86182	2	7602.15576	42485.1719	1586.66197	0.05730251	50	30	120	0	
20	279	278	272	7.09646475	7.08316553	8.01217823	393.195404	2	64878.3125	512108.813	34058.8111	0.94869914	50	30	120	7.46831618	

Using the RawQuant output in your analysis

There are numerous ways you can capitalize on the data output from RawQuant. For some code examples that process RawQuant output and combine them with search results from PeptideShaker, please see the GitHub associated with the RawQuant manuscript (https://github.com/chrishuges/RawQuant_JPR-2018). There are a wide variety of RNotebook code snippets here that can be followed for doing numerous analyses. Potential analysis pipelines of RawQuant data include:

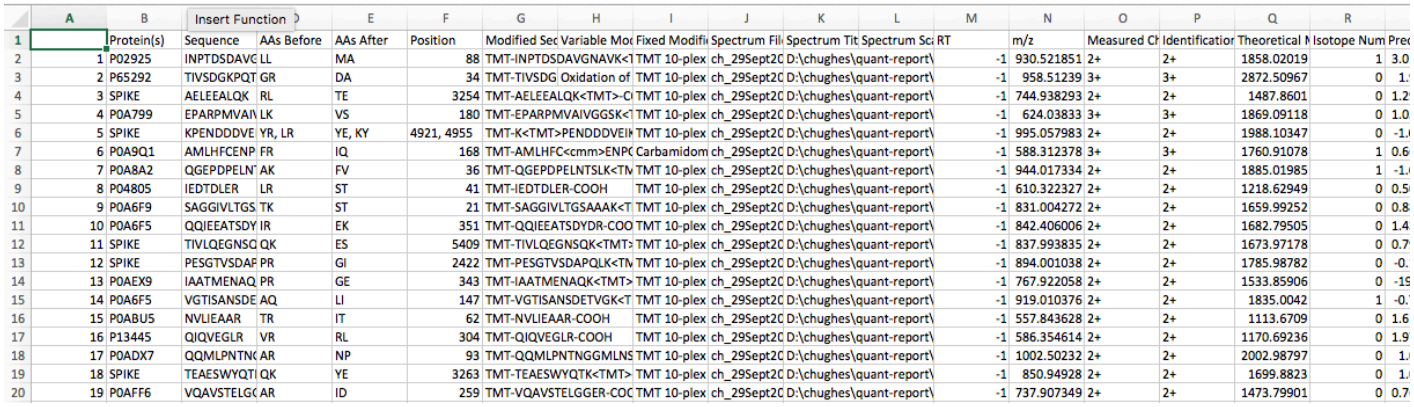
- Parse your data file with RawQuant. Use the generated 'metrics' file to measure the performance of your MS across a wide variety of acquisition parameters, or monitor the performance over time.
- Parse your data file with RawQuant and generate an MGF. Process the MGF using a search tool that accepts MGF input (e.g. Mascot, SearchGUI...and many other search engines). Use the MS2 scan numbers of the peptide matches to combine with the parsed RawQuant data (ScanNumber column in RawQuant). Check the metrics of the peptides that were identified, are you always hitting the max injection time for MS2? Can you afford to scan faster? Or do you need to go slower?
- Quantify your data file with RawQuant and generate an MGF. As before, search your data using any program that will take an MGF input. Now you have many options:
    - Combine the quant and search output using ScanNumbers, just as before. PeptideShaker, for example, outputs these by default in the peptide spectral match output, as does Proteome Discoverer. Process your data in R using some of the code mentioned above.
    - Input your quantification matrix into MSnbase (R) and process in combination with your modifications data.
    - Use the MGF file alone in Isobar (R) to gather quantification and identification data.
    - Use another tool I don't know about to do the same thing! The open source nature of RawQuant here gives you many options. Test and choose what works best for your own sample analysis!

Let's look at an example for combining the RawQuant data with search engines.

1. In the RawQuant manuscript, we used SearchGUI and PeptideShaker a lot. As you saw above, the RawQuant output always contains the columns 'ScanNumber', or 'MS2ScanNumber', 'MS1ScanNumber'

and so on.

2. PeptideShaker can output a variety of reports. The one we are interested in for combination is the 'Default PSM Report'.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	
1		Protein(s)	Sequence	AAs Before	AAs After	Position	Modified Sec	Variable Mo	Fixed Modifi	Spectrum Fil	Spectrum Tit	Spectrum Sc	RT	m/z	Measured Ch	Identificatior	Theoretical M	Isotope Num	Prec
2	1	P02925	INPTDSDAVG	LL	MA	88	TMT-INPTDSDAVGNAVK<T	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	930.521851	2+	2+	1858.02019	1	3.0	
3	2	P65292	TIVSDGKPQT	GR	DA	34	TMT-TIVSDG	Oxidation of	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	958.51239	3+	3+	2872.50967	0	1.	
4	3	SPIKE	AELEEALQK	RL	TE	3254	TMT-AELEEALQK<TMT>-C	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	744.938293	2+	2+	1487.8601	0	1.2	
5	4	P0A799	EPARPMVAIV	LK	VS	180	TMT-EPARPMVAIVGGSK<T	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	624.03833	3+	3+	1869.09118	0	1.0	
6	5	SPIKE	KPENDDDVE	YR, LR	YE, KY	4921, 4955	TMT-K<TMT>PENDDDVEI	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	995.057983	2+	2+	1988.10347	0	-1.	
7	6	P0A9Q1	AMLHFCENP	FR	IQ	168	TMT-AMLHFC<cmm>ENP(	Carbamidom	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	588.312378	3+	3+	1760.91078	1	0.6	
8	7	P0A8A2	QGEPDPELN	AK	FV	36	TMT-QGEPDPELNTSLK<TM	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	944.017334	2+	2+	1885.01985	1	-1.	
9	8	P04805	IEDTDLER	LR	ST	41	TMT-IEDTDLER-COOH	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	610.322327	2+	2+	1218.62949	0	0.5	
10	9	P0A6F9	SAGGIVLTGS	TK	ST	21	TMT-SAGGIVLTGSAAAK<T	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	831.004272	2+	2+	1659.99252	0	0.8	
11	10	P0A6F5	QQIEEATSDY	IR	EK	351	TMT-QQIEEATSDYDR-COO	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	842.406006	2+	2+	1682.79505	0	1.4	
12	11	SPIKE	TIVLQEGNSQ	QK	ES	5409	TMT-TIVLQEGNSQK<TMT>	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	837.993835	2+	2+	1673.97178	0	0.7	
13	12	SPIKE	PESGTVSDAF	PR	GI	2422	TMT-PESGTVSDAPQLK<TM	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	894.001038	2+	2+	1785.98782	0	-0.	
14	13	P0AEX9	IAATMENAQ	PR	GE	343	TMT-IAATMENAQK<TMT>	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	767.922058	2+	2+	1533.85906	0	-1	
15	14	P0A6F5	VGTISANSDE	AQ	LI	147	TMT-VGTISANSDETVGK<T	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	919.010376	2+	2+	1835.0042	1	-0.	
16	15	P0ABU5	NVLIEAAR	TR	IT	62	TMT-NVLIEAAR-COOH	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	557.843628	2+	2+	1113.6709	0	1.6	
17	16	P13445	QIQVEGLR	VR	RL	304	TMT-QIQVEGLR-COOH	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	586.354614	2+	2+	1170.69236	0	1.9	
18	17	P0ADX7	QQMLPNTNG	AR	NP	93	TMT-QQMLPNTNGGMLNS	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	1002.50232	2+	2+	2002.98797	0	1.	
19	18	SPIKE	TEAESWYQTI	QK	YE	3263	TMT-TEAESWYQTK<TMT>	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	850.94928	2+	2+	1699.8823	0	1.	
20	19	P0AFF6	VQAVSTELGG	AR	ID	259	TMT-VQAVSTELGGER-COO	TMT 10-plex	ch_29Sept2(	D:\chughes\quant-report\		-1	737.907349	2+	2+	1473.79901	0	0.7	

3. As you can see, there is a lot of information here. Unfortunately, at the moment, the scan number is a bit messed up because of the MGF format of RawQuant (known bug), but this information is still accessible from the 'Spectrum Title' column.

Spectrum File	Spectrum Title
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8653
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8658
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8666
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8661
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 6401
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8667
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8668
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 6408
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8669
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 6405
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7590
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7591
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7584
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7586
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7585
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7587
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8631
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8636
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7589
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8634
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8635
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8638
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7595
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7593
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7592
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 7597
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8643
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8644
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8645
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8646
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8648
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8619
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 3001
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8629
ch_29Sept2017_eColi-31907_TMT11_MS2_1_MGF.mgf	D:\chughes\quant-report\hughes_tmt-methods2\ch_29Sept2017_eColi-31907_TMT11_MS2_1.raw spectrum 8624

4. Using R, Python, or your language of choice, you can combine the RawQuant output with the PeptideShaker results by combining this spectrum number with the appropriate column from RawQuant.

- For a parsed file where -o 2 was used, this will be the 'ScanNumber' column.
- for a quant file where quant was in MS2, this will be the 'ScanNumber' column.
- for a quant file where the quant was SPS-MS3, this will be the 'MS2ScanNumber' column.

5. For combining with other search engines, you simply need to locate the output file that contains the MS2 scan number of a given peptide spectral match, or peptide match and align this with the same values in the RawQuant tables.

    - In MaxQuant, this data is available in the 'peptides.txt' output file.
    - In the Trans-Proteomic Pipeline, this data can be made available in the PeptideProphet output.
    - In Proteome Discoverer, this data is available in the PSM report.

What about using the RawQuant output for other tools?

1. As mentioned above, the quantification matrices can be directly input into R or Python (or any other tools that accept tab delimited text files.

    - You can use the quantification data as input to a wide variety of R packages that take expression data, such as microarray tools like Limma, or VSN.
    - Perform differential abundance analysis of the peptides using R packages like Limma or PECA.
    - You can also use the data as input for other packages that are more proteomics focused, such as MSnbase.

2. The MGF output can be used as input to a variety of pipelines based in R or Python.

    - Use the MGF output as a basis for quantification and identification using Isobar.
    - Perform database matching with the MGF using packages such as rTandem and MSGFplus.

3. The parsed data can also be directly input into R and Python to perform your own calculations of topN, scan rate, ID rate, etc...

    - Use the R code guides found on https://github.com/chrishuges/RawQuant_JPR-2018 to get ideas and help.
    - Design your own code to track instrument performance over time based on parsed data.

The massive number of tools available to proteomics researchers now means that you can combine RawQuant data with virtually any sort of analysis pipeline. The specifics will depend on the software tools you are using, and the data made available by these packages.

Final notes

RawQuant is in active development. We hope to have a GUI-based version sometime in the near future. If you

have any issues, or have encountered a bug, or there is a feature you would really like to see, please do not hesitate to contact Kevin Kovalchik (kkovalchik@bcgsc.ca) or Christopher Hughes (chughes@bcgsc.ca), and we will do our best to get you using our tool!