Tips and advice when creating a python software for lab members to use in academia

PyData Global 2021

Jeremy Selva

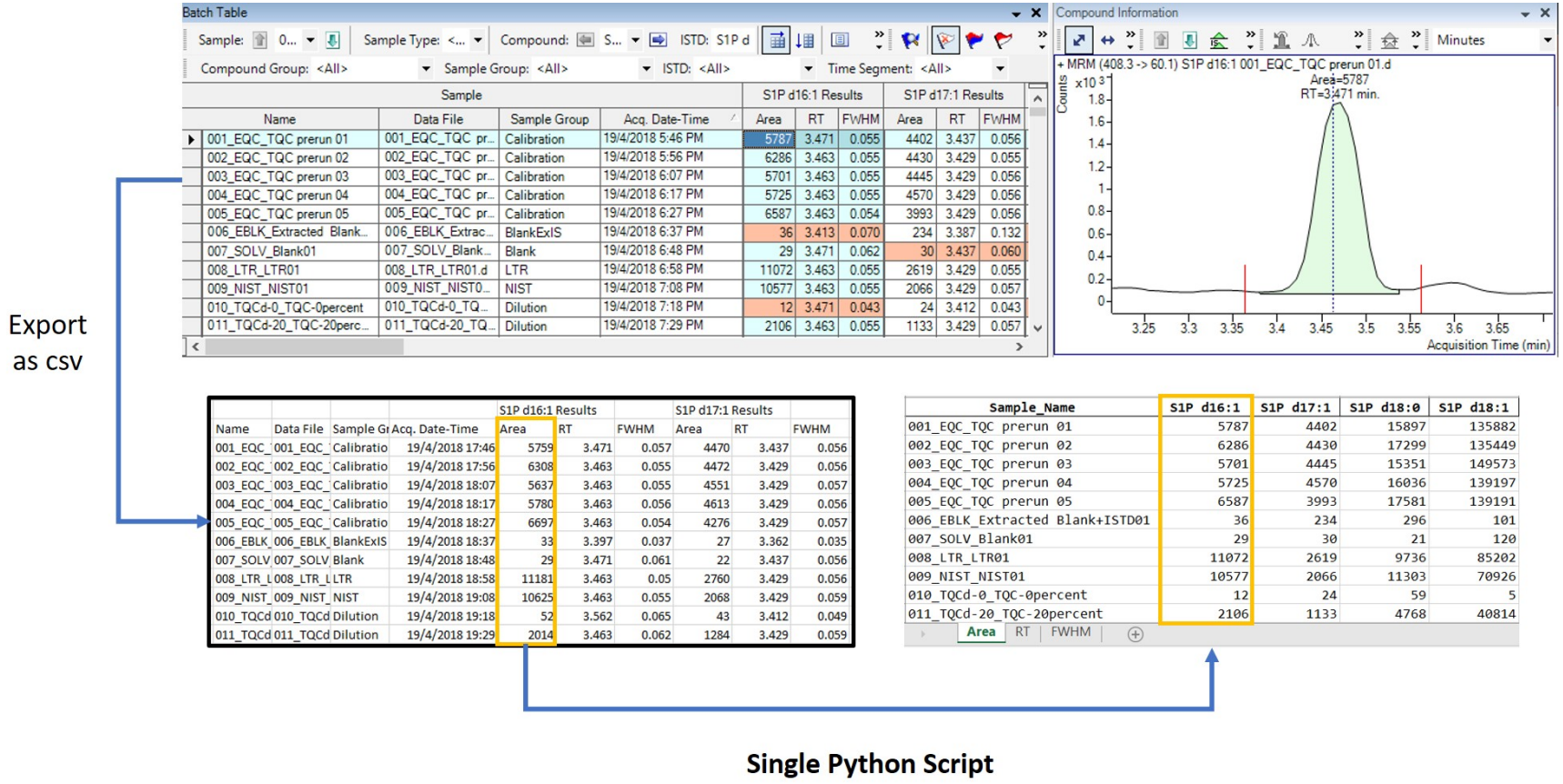
Images by Amonrat Rungreangfangsai

Xaringan Slide Template by Sharla Gelfand

https://jauntyjjs.github.io/PyDataGlobal2021Talk

Introduction

It started with a Python script used to organise Mass Spectrometry data for my project.



Single Python Script

https://jauntyjjs.github.io/PyDataGlobal2021Talk

Introduction

Many asked me to troubleshoot R/Python/Bash scripts, created by past collaborators, that were meant to do similar tasks as mine.

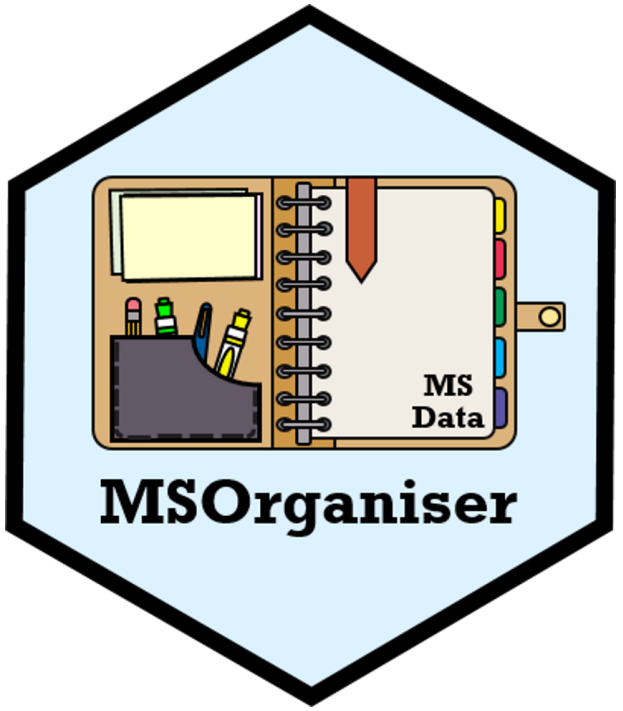
- ***"This script does not work on my computer anymore. I don't know why."*** 😳

Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

Introduction

I combined all these scripts into one Windows executable software in Python called MSOrganiser.



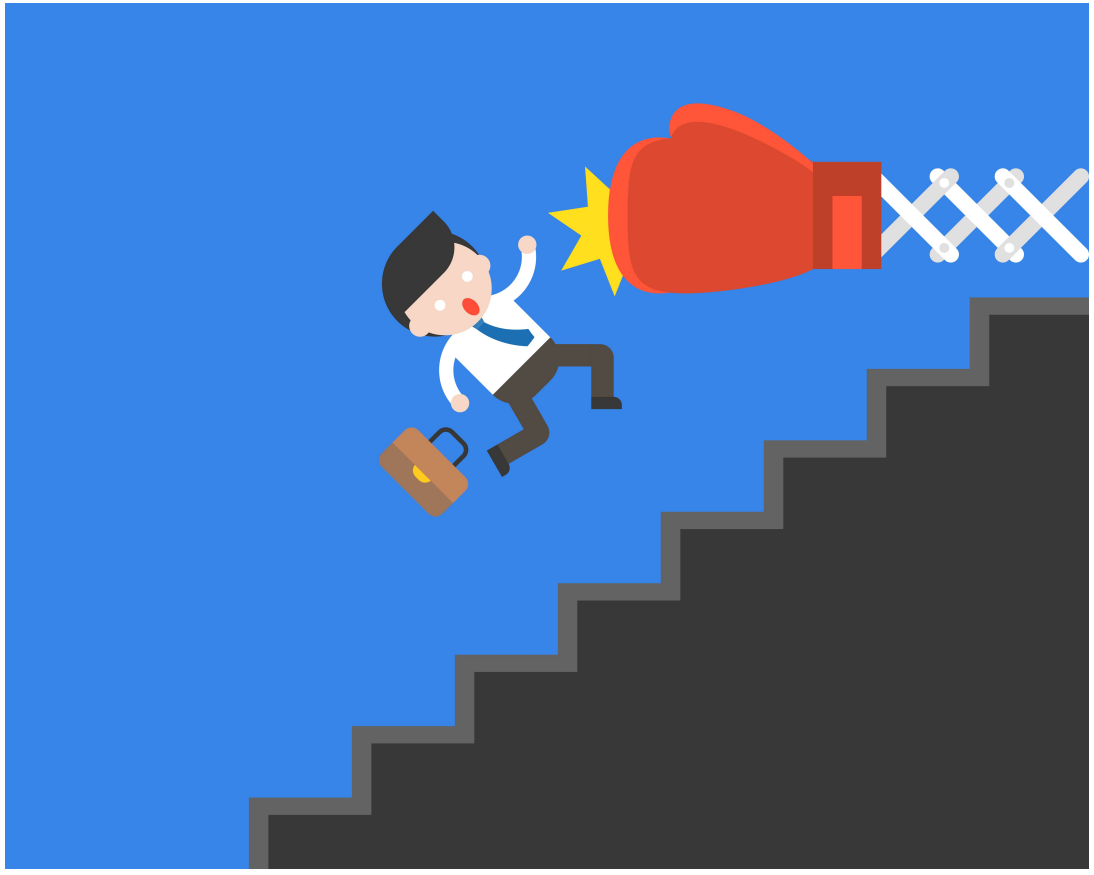
https://github.com/SLINGhub/MSOrganiser

Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

Introduction

Looking back, I realise that things have not been easy...



Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

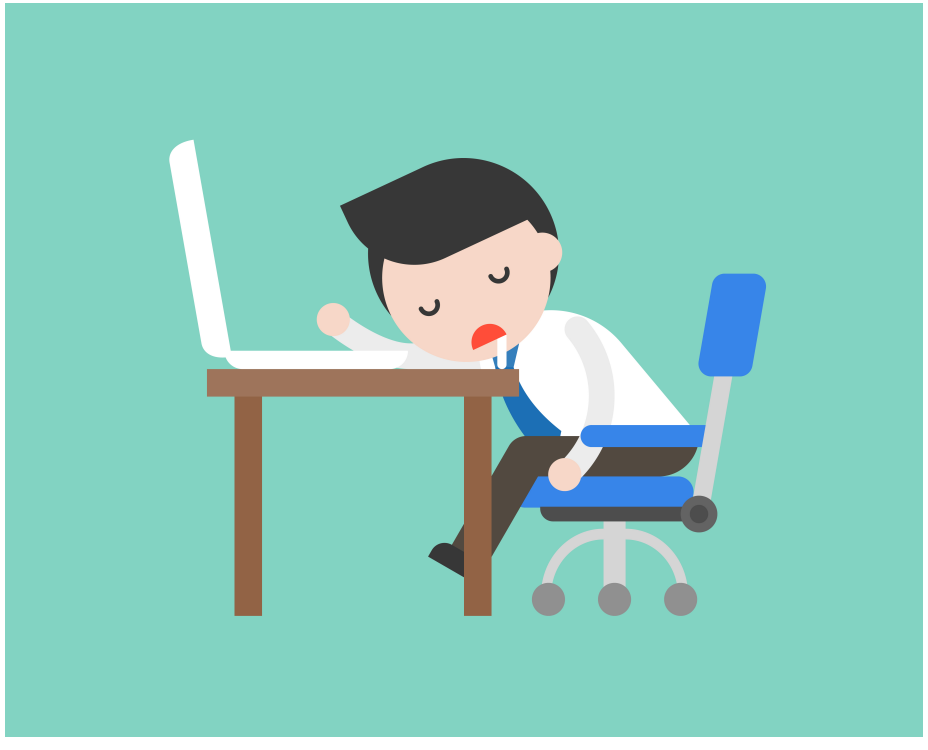
Introduction

In academia, most of us are not trained in coding or software development. It is hard for us to write software for our own research paper, it is even more challenging to create software for others to use.

Daniel Lemire's blog

On the quality of academic software

- Most academic researchers write software for themselves. As Cook put it: "People who have only written software for their own use have no idea how much work goes into writing software for others." Cooking your own food is a lot easier than being a chef in a restaurant. The difference between the two is at least an order of magnitude, if not two.
  There is very little appreciation for this fact in academia. The default is to write throw-away code: you write the code, you use it and you forget about it. Issues like maintenance and documentation are discussed at length in some classes, but it is rarely put in practice within academia.



Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

Introduction

In this talk, I decide to share some things that have helped me tremendously.

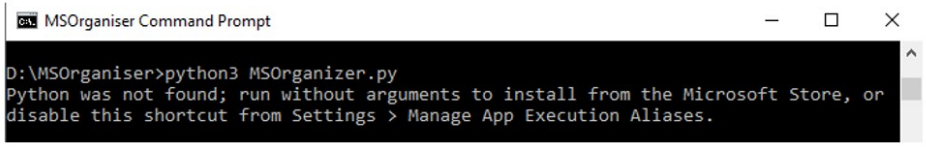
Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

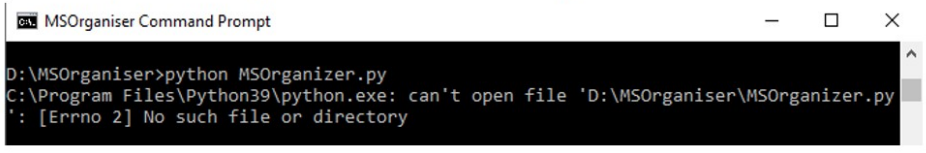
1: Convert scripts to a web service or executable software

Users make less error when clicking buttons compared to typing in command lines

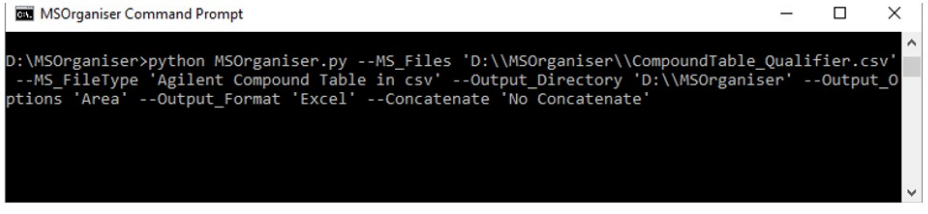
Attempt 1 … Need to install Python and libraries



Attempt 50 … Figure out how the script works

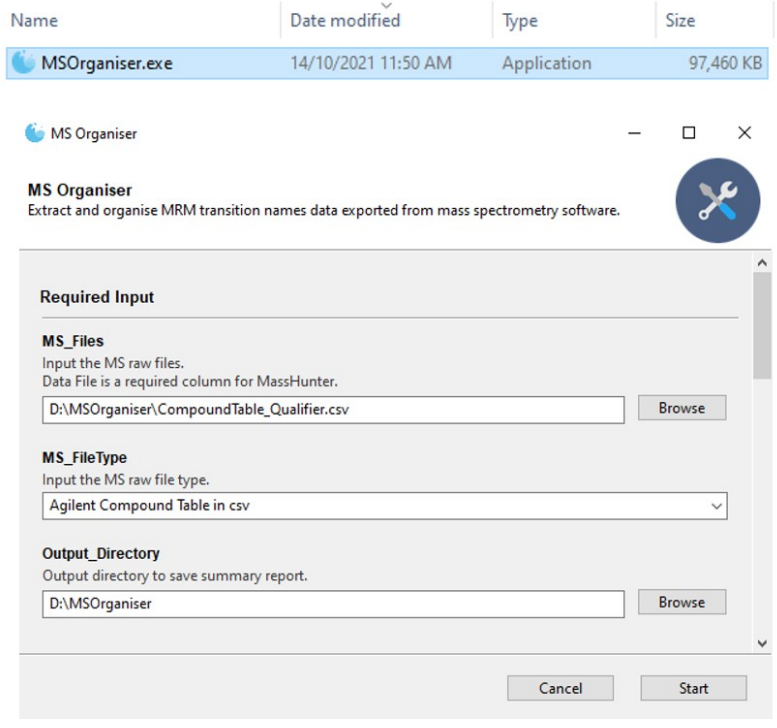


Attempt 100 … Finally



VS

Attempt 1 … Double-click, fill in parameters, click Start

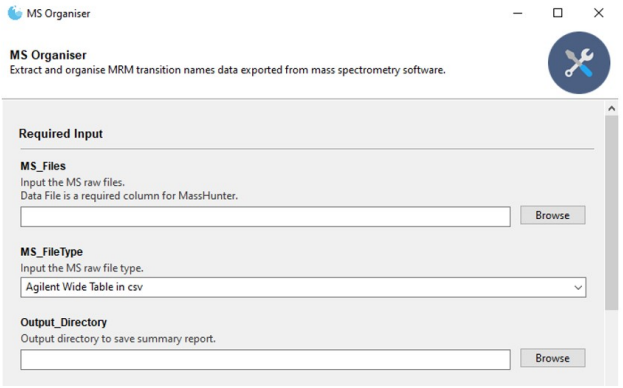


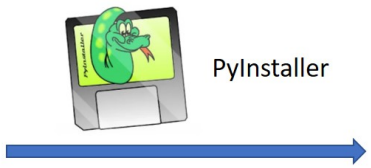
https://jauntyjjs.github.io/PyDataGlobal2021Talk

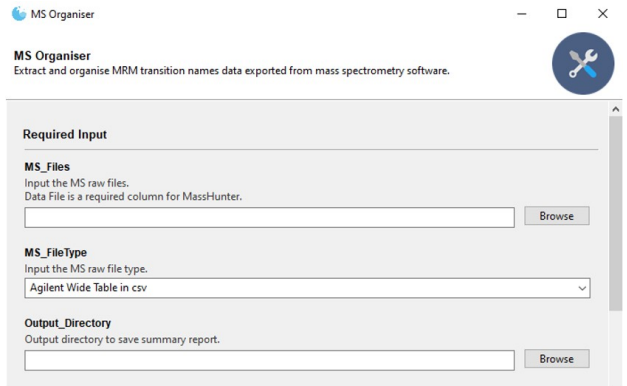
Advice 1 Tips

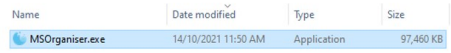
- Chriskiehl's Gooey to create a simple GUI interface quickly.
- Pyinstaller to convert the scripts to a stand alone executable program.

```
#Required Arguments
#nargs="+" is needed to turn multiple input into a list.
required_args.add_argument('--MS_Files',
                           required=True,
                           nargs='+',
                           help="Input the MS raw files.\nData File is a required column for M
                           widget='MultiFileChooser',
                           default=stored_args.get('MS_Files'))
required_args.add_argument('--MS_FileType',
                           required=True,
                           choices=['Agilent Wide Table in csv',
                                    'Agilent Compound Table in csv',
                                    'Multiquant Long Table in txt'],
                           help='Input the MS raw file type.', default=MS_FileType)
required_args.add_argument('--Output_Directory',
                           required=True,
                           action='store',
                           help="Output directory to save summary report.",
                           widget="DirChooser", default=stored_args.get('Output_Directory'))
```









https://jauntyjjs.github.io/PyDataGlobal2021Talk

Advice 1 Tips

Jack McKew's blog helps me make use of the two tools to get what I need.

https://jauntyjjs.github.io/PyDataGlobal2021Talk

I forgot about new users

Simple program

3 page
user manual

Several feature
request later

Complex program

30 page
user manual

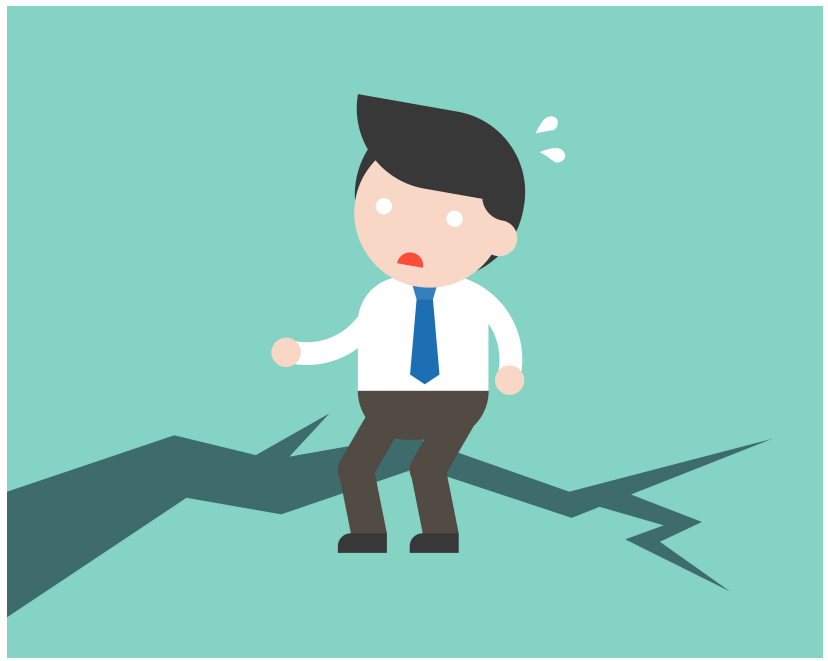
Baby Dragon by drawinghowtos.com

Cartoon Hydra by how-to-draw-cartoons-online.com

I forgot about new users

New users typical reaction to complex program

- Scared 😟 and Confused 😳
- *"Maybe this tool is not for me"* 😂



Cartoon Hydra by how-to-draw-cartoons-online.com

Images by Amonrat Rungreangfangsai

2: Give a software overview using cheatsheet

MSOrganiser Summary

Input Raw data

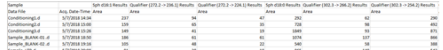
Table of valid MRM data formats

MRM transition names data form	Compulsory column names		Restricted column names
	Without Qualifiers	With Qualifiers	
Agilent's WideTable form	Sample • Data File Compound Results • Area	Qualifier Results • Area	Sample • Quantification Message Columns from • ISTD Compound Methods • ISTD Compound Results
Agilent's CompoundTable form	Sample • Data File Compound Method • Name Compound Results • Area	Qualifier Method • Transition Qualifier Results • Area	

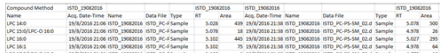
Example

Agilent's Wide Table form

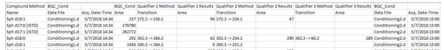
Agilent's Wide Table form with Qualifier



Agilent's Compound Table form

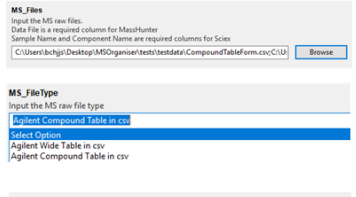


Agilent's Compound Table form with Qualifier

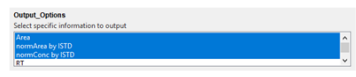


Organise MRM data to tidy form

1. **Load MRM Files and specify MS_FileType and Output Directory**

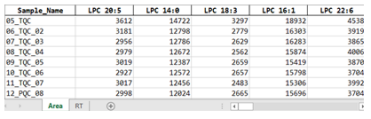


2. **Choose Output options**



3. **Click start to organise data**

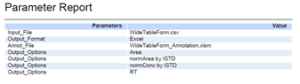
By default, organised data will be output into an Excel file in wide table form



Sample_Name	LPC 20:5	LPC 14:0	LPC 18:3	LPC 16:1	LPC 22:6
05_TQC	3612	14722	3297	18932	4538
06_TQC_02	3181	12798	2779	16303	3919
07_TQC_03	2956	12786	2629	16283	3865
08_TQC_04	2979	12672	2562	15874	4006
09_TQC_05	3019	12387	2659	15419	3870
10_TQC_06	2927	12572	2657	15798	3704
11_TQC_07	3017	12456	2483	15306	3992
12_PQC_08	2998	12024	2665	15696	3704

A report pdf file will also be provided to record the parameters used

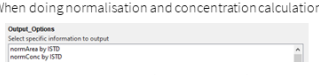
Parameter Report



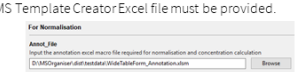
Parameters	Value
Input_File	WideTableForm.csv
Output_Format	Excel
Annot_File	WideTableForm_Annotation.xlsm
Output_Options	Area
Output_Options	normArea by ISTD
Output_Options	normConc by ISTD
Output_Options	RT

Input Annotation File

When doing normalisation and concentration calculation,



MS Template Creator Excel file must be provided.



Refer to the summary sheet or manual to see how it is used.

Here are some input examples.

Transition Name Annotation

Transition_Name	Transition_Name_ISTD
LPC 14:0	LPC 20:0 (IS)
LPC 15:0	LPC 20:0 (IS)

ISTD Annotation

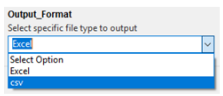
ISTD_Table				
Transition_Name_ISTD	ISTD_Concentration			
	ISTD_Conc_[ng/mL]	ISTD_Note	to	ISTD_Conc_[uM]
LPC 20:0 (IS)	50	0.1046		141.0659481
MHC d18:1/18:0(d7) (IS)	50	0.279		181.1565832

Sample Annotation

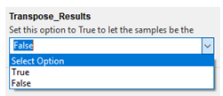
Data_File_Name	Merge_Status	Sample_Name	Sample_Type	Sample_Amount	Sample_Amount_Unit	ISTD_Mixture_Volume_[uL]	Concentration_Unit
WideTableFormRow1.csv Valid		05_TQC	TQC	10	uL	190	nmol/uL
WideTableFormRow1.csv Valid		06_TQC_02	TQC	10	uL	190	nmol/uL
WideTableFormRow1.csv Valid		07_TQC_03	TQC	10	uL	190	nmol/uL
WideTableFormRow1.csv Valid		08_TQC_04	TQC	10	uL	190	nmol/uL

Additional Features

Different output format for the organised data.

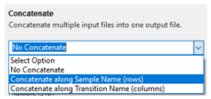


Set Transpose_Results to True to let the sample name be the columns instead of the transition names.



Transition_Name	05_TQC	06_TQC_02
LPC 20:5	3612	3181
LPC 14:0	14722	12798
LPC 18:3	3297	2779
LPC 16:1	18932	16303
LPC 22:6	4538	3919
LPC 15:0	10430	8380
LPC 20:4	28891	25150
LPC 18:2	150750	129316

Select the following concatenation options to combine multiple input files to one result file.



Updated: 2021-10

https://jauntyjjs.github.io/PyDataGlobal2021Talk

Advice 2 Tips

RStudio cheatsheet examples and template and advice

RStudio Cheatsheets

The cheatsheets below make it easy to use some of our favorite packages. From time to time, we will add new cheatsheets. If you'd like us to drop you an email when we do, click the button below.

SUBSCRIBE TO CHEATSHEET UPDATES

CONTRIBUTED CHEATSHEETS

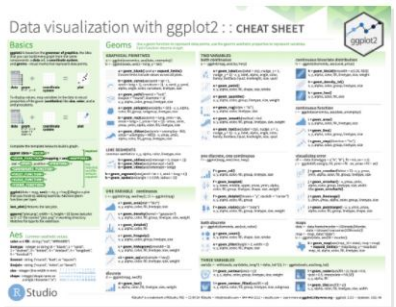
TRANSLATIONS

HOW TO CONTRIBUTE

Data visualization with ggplot2 cheatsheet

The ggplot2 package lets you make beautiful and customizable plots of your data. It implements the grammar of graphics, an easy to use system for building plots. Updated August 2021.

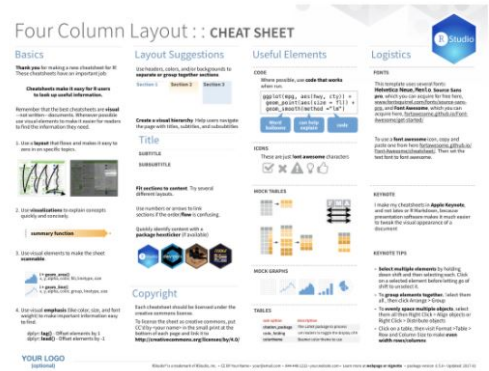
DOWNLOAD



Contribute a New Cheat Sheet

Want to contribute a cheat sheet of your own?

We'd like to help you make and share high quality cheat sheets on R topics. The template below provides a useful starting place. It contains tips for designing a three or four column cheat sheet, as well as reusable elements to build your sheet with. Check out the README for more tips on making good cheat sheets.



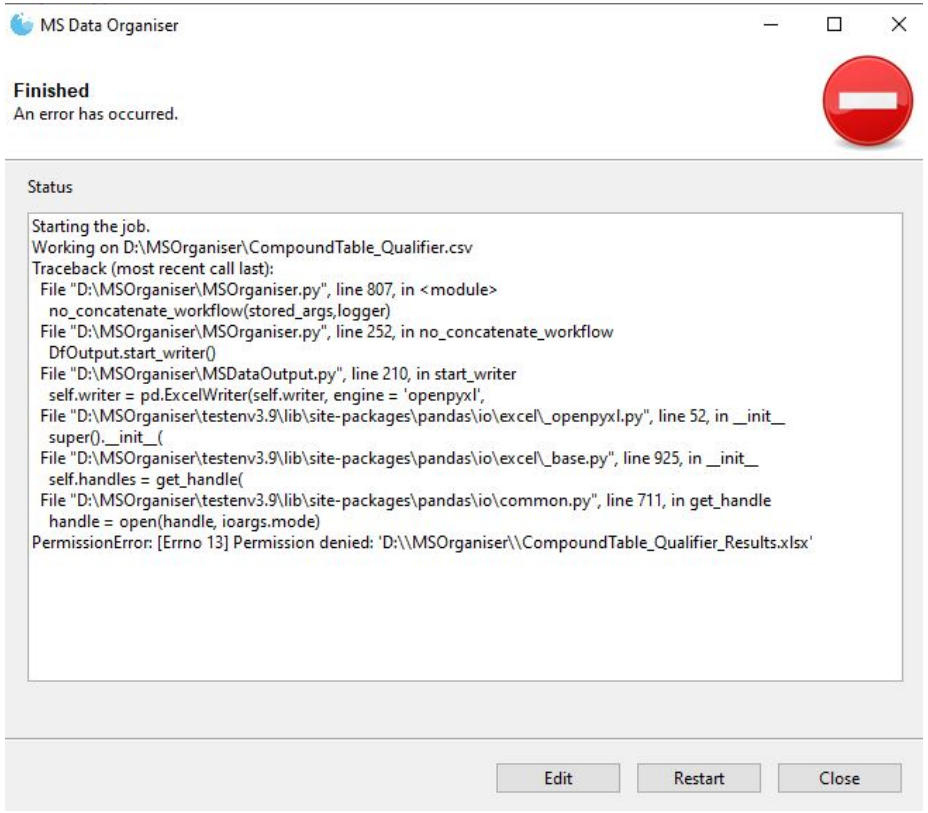
Download Template for Keynote

Download Template for Powerpoint

https://jauntyjjs.github.io/PyDataGlobal2021Talk

3: Provide helpful messages when users make a mistake

Users and programmers make mistakes sometimes. Software error messages are unavoidable.



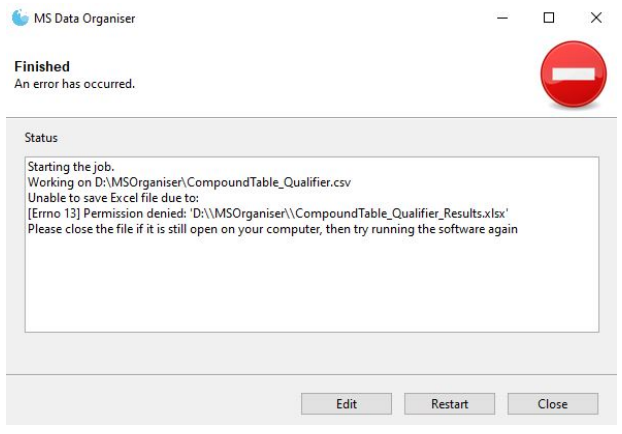
Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

3: Provide helpful messages when users make a mistake

However, if you can make them helpful, it does goes a long way.

```
try:
    self.writer = pd.ExcelWriter(self.writer, engine = 'openpyxl',
                                 #options = options,
                                 engine_kwargs = {'options': {'strings_to_formulas': False,
                                                              'strings_to_urls' : False
                                                              }
                                                  }
                                 )
except IOError as i:
    if self.ingui:
        print('Unable to save Excel file due to:',flush=True)
        print(i,flush=True)
        print('Please close the file if it is still open on your computer, ' +
              'then try running the software again',flush=True)
    if self.logger:
        self.logger.error('Unable to save Excel file due to:')
        self.logger.error(i)
        self.logger.error('Please close the file if it is still open on your computer, ' +
                          'then try running the software again')
    sys.exit(-1)
```



Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

Advice 3 Tips

Saadia Minhas' blog provides some good tips

4. Be Humble — Don't Blame User

A good error message is humble. It conveys the issues gracefully to its user without blaming him for his actions.

The user can perform an incorrect action again and again. But the design's responsibility is to inform him about his mistakes in a good way.

> "A good way to incorporate more human tone to your error messages is to think about explaining it out loud to someone. How does it sound when you speak it in conversation." — Sonia Gregory

User is informed in a humble way about the problem.

The error message seems to blame the user.

Micheal Lynch's tip to show openness to criticism

11. Artfully solicit missing information

Sometimes code review notes leave too much room for interpretation. When you receive a comment like, "This function is confusing," you probably wonder what "confusing" means, exactly. Is the function too long? Is the name unclear? Does it require more documentation?

For a long time, I struggled to clarify ambiguous notes without sounding defensive. My instinct was to ask, "What's confusing about it?" but that comes across as grouchy.

Once, I unintentionally sent a vague note to my teammate, and he responded in a way that I found fantastically disarming:

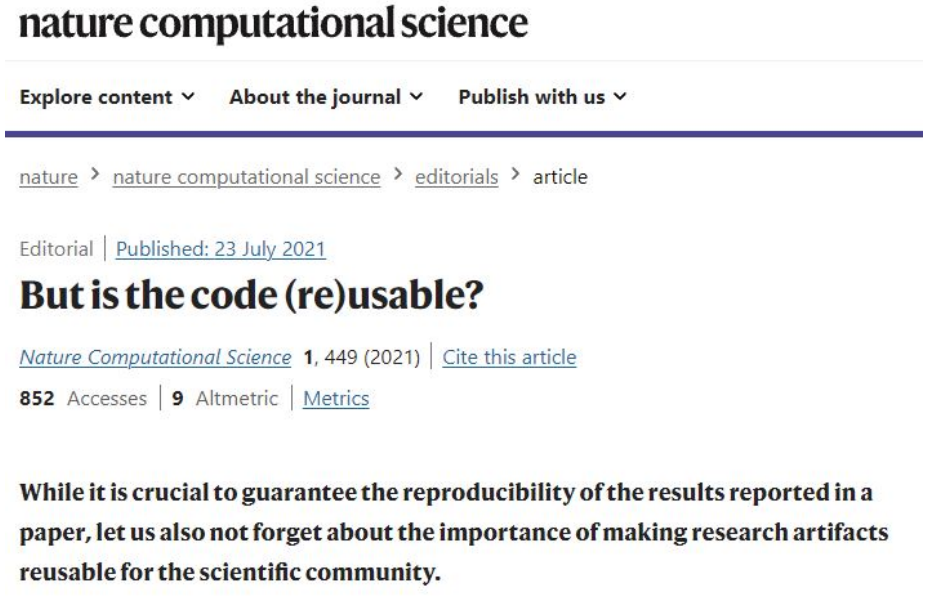
> *What changes would be helpful?*

I love this response because it signals a lack of defensiveness and openness to criticism. Whenever a reviewer gives me unclear feedback, I always respond with some variation of, "What would be helpful?"

Another useful technique is to guess your reviewer's intent and proactively edit your code based on that assumption. For a note like, "this is confusing," give your code a second look. Usually, there's *something* you can do to improve clarity. A revision communicates to your reviewer that you're amenable to change, even if it's not the one they had in mind.

4: Include necessary software reports to show reliability

Academia journals are getting more demanding with regards to software, wanting reusability besides reproducibility.



nature computational science

Explore content ▾ About the journal ▾ Publish with us ▾

nature › nature computational science › editorials › article

Editorial | Published: 23 July 2021

But is the code (re)usable?

Nature Computational Science **1**, 449 (2021) | Cite this article

852 Accesses | **9** Altmetric | Metrics

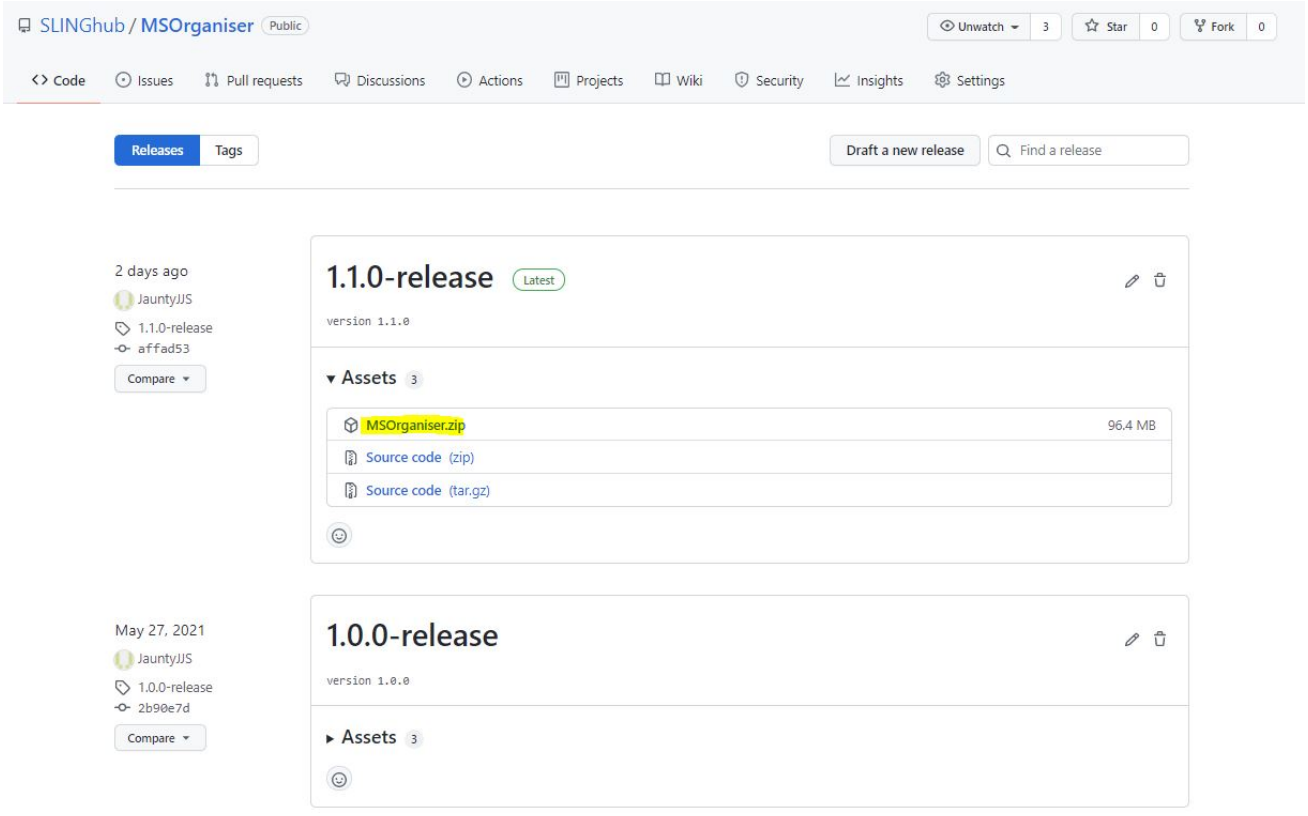
While it is crucial to guarantee the reproducibility of the results reported in a paper, let us also not forget about the importance of making research artifacts reusable for the scientific community.

We editors at *Nature Computational Science* do our best to ensure that the source codes associated with our papers have the aforementioned elements to make them reusable as much as possible. In addition to asking our referees about reusability during the code peer-review process, we also ask authors to fill out a code and software submission checklist that covers reusability requirements. But we also urge you, the author, to start thinking more about reusability when you are working on a project and submitting a paper. For instance, make sure that you have detailed instructions, making the code easier to install whenever possible, and making it easier to run with new data and parameters as well. A useful tip is to ask a colleague from your lab to try out your code to see whether they can successfully follow the instructions and run your code on a potentially different machine.

https://www.nature.com/articles/s43588-021-00109-9

Advice 4 Tips

Tag a label on different software version and encourage users to cite not just the software name but the version number as well.



https://jauntyjjs.github.io/PyDataGlobal2021Talk

Advice 4 Tips

Create a report file/table (pdf or excel) that store the user's input parameters

Parameter Report

Parameters	Value
Input_File	CompoundTable_Qualifier.csv
Input_File_Type	Agilent Compound Table in csv
Output_Options	Area
Output_Format	Excel
Concatenate	No Concatenate
Allow_Multiple_ISTD	False
Transpose_Results	False
Long_Table	False
Long_Table_Annot	False

Chris Moffitt's example using WeasyPrint

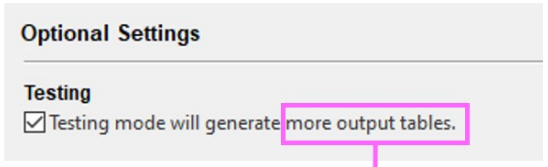
Matt Clarke's example using Gilfoyle

https://jauntyjjs.github.io/PyDataGlobal2021Talk

Advice 4 Tips

Show pre-processing results to explain how the software calculate the final results.

They are helpful when there is a need to troubleshoot for logical errors



`normArea by ISTD = ( Area / ISTD_Area )`

	Sample_Name	LPC 20:5	LPC 14:0	LPC 20:0 (IS)
2	05_TQC	0.247414206	1.008425235	1
3	06_TQC_02	0.249451066	1.003607277	1
4	07_TQC_03	0.234231379	1.013153724	1
5	08_TQC_04	0.23028757	0.979591837	1
6	09_TQC_05	0.237342767	0.973820755	1
7	10_TQC_06	0.237774167	1.021283509	1
8	11_TQC_07	0.248128958	1.024426351	1

Area | ISTD_Area | **normArea_by_ISTD** | Transition_Name_Annot

equals

	Transition_Name	Transition_Name_ISTD
2	LPC 14:0	LPC 20:0 (IS)
3	LPC 20:0 (IS)	LPC 20:0 (IS)
4	LPC 20:5	LPC 20:0 (IS)
5		

Transition_Name_Annot | Area | ISTD_Area | normA

	Sample_Name	LPC 20:5	LPC 14:0	LPC 20:0 (IS)
2	05_TQC	3612	14722	14599
3	06_TQC_02	3181	12798	12752
4	07_TQC_03	2956	12786	12620
5	08_TQC_04	2979	12672	12936
6	09_TQC_05	3019	12387	12720
7	10_TQC_06	2927	12572	12310
8	11_TQC_07	3017	12456	12159

Area | ISTD_Area | Transition_Name_Annot | normArea_by_ISTD

divided by

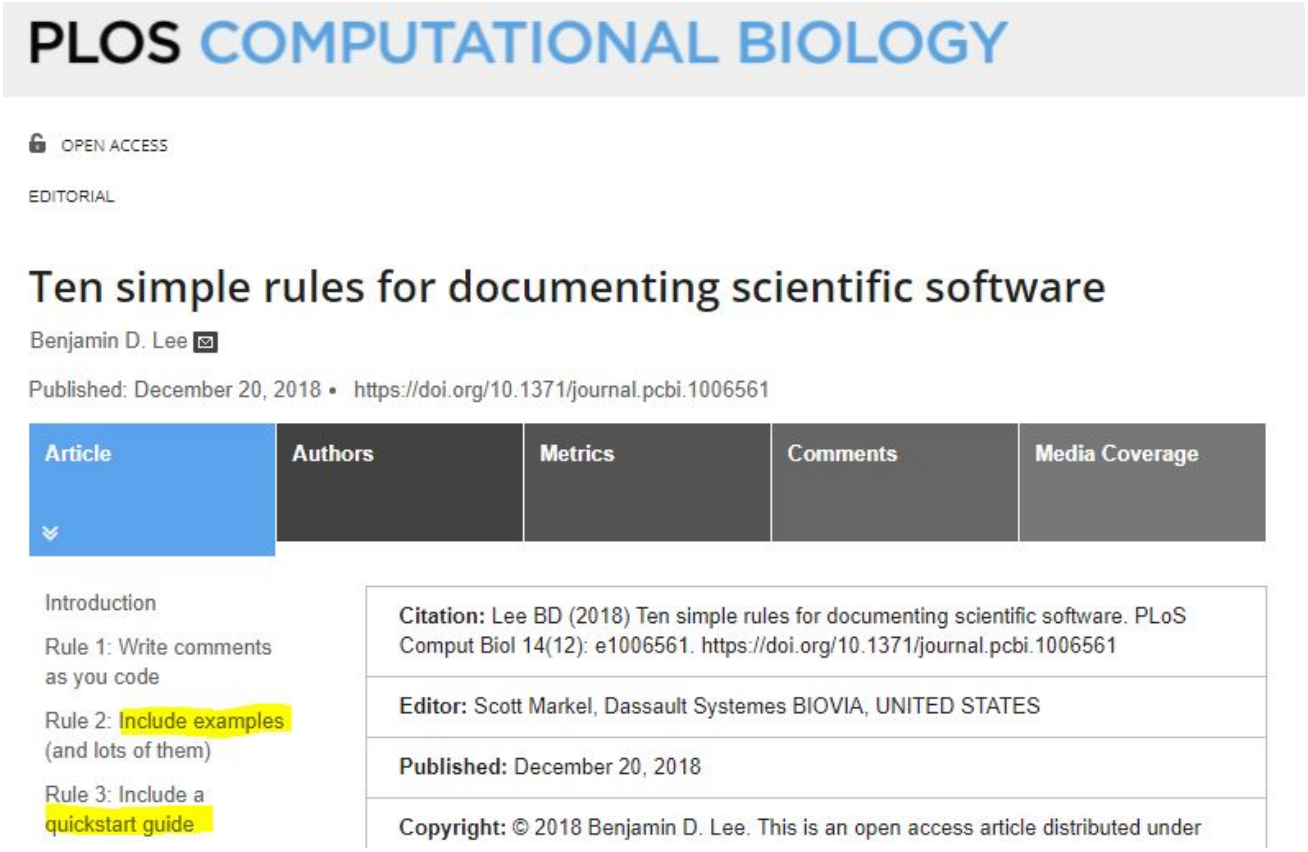
	Sample_Name	LPC 20:5	LPC 14:0	LPC 20:0 (IS)
2	05_TQC	14599	14599	14599
3	06_TQC_02	12752	12752	12752
4	07_TQC_03	12620	12620	12620
5	08_TQC_04	12936	12936	12936
6	09_TQC_05	12720	12720	12720
7	10_TQC_06	12310	12310	12310
8	11_TQC_07	12159	12159	12159

Area | **ISTD_Area** | Transition_Name_Annot | normArea_by_ISTD

https://jauntyjjs.github.io/PyDataGlobal2021Talk

My documentation issue

I started my documentation by trying to follow some advice provided by Lee's article



PLOS COMPUTATIONAL BIOLOGY

OPEN ACCESS

EDITORIAL

Ten simple rules for documenting scientific software

Benjamin D. Lee

Published: December 20, 2018 • https://doi.org/10.1371/journal.pcbi.1006561

Article | Authors | Metrics | Comments | Media Coverage

Introduction

Rule 1: Write comments as you code

Rule 2: Include examples (and lots of them)

Rule 3: Include a quickstart guide

Citation: Lee BD (2018) Ten simple rules for documenting scientific software. PLoS Comput Biol 14(12): e1006561. https://doi.org/10.1371/journal.pcbi.1006561

Editor: Scott Markel, Dassault Systemes BIOVIA, UNITED STATES

Published: December 20, 2018

Copyright: © 2018 Benjamin D. Lee. This is an open access article distributed under

My documentation issue

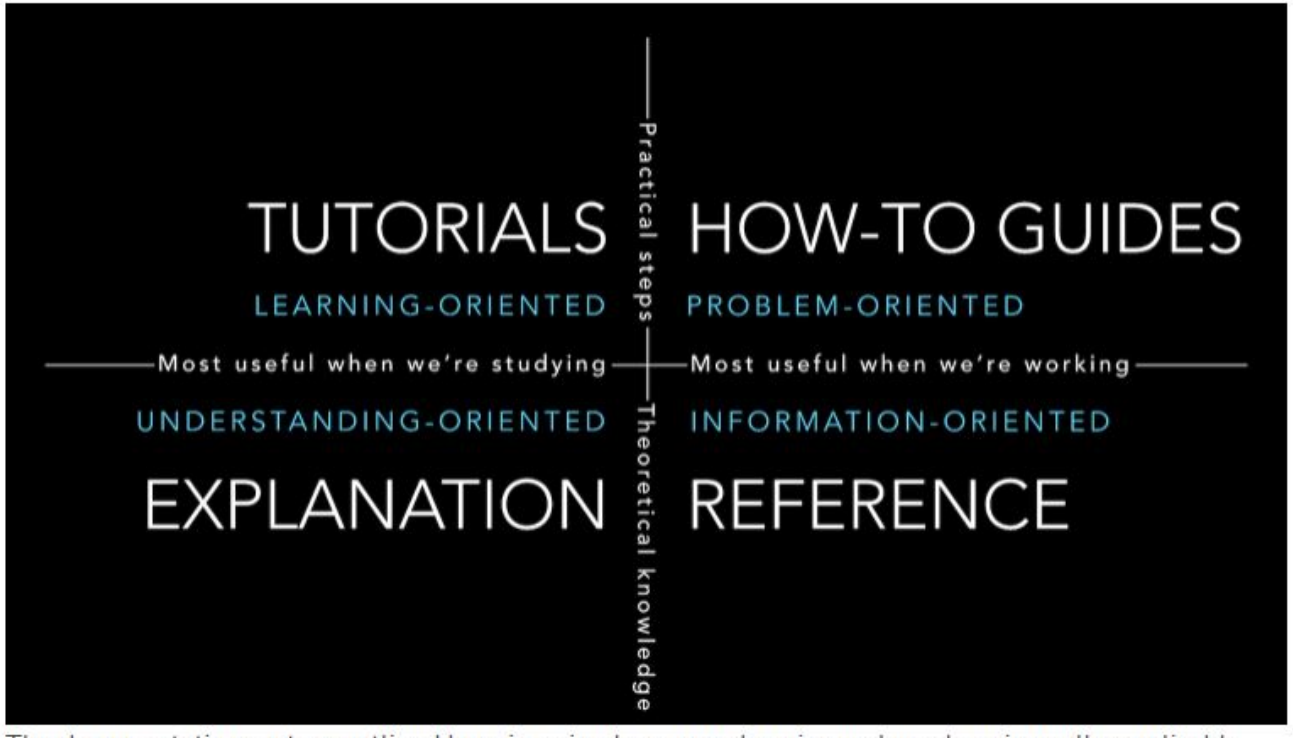
As my documentation starts to get longer and more complex, people find it hard to understand.

Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

5: Organise your documentation into specific structures

Thankfully, I came across DIVIO website that encourages me to split my documentations into manageable categories.



The documentation system outlined here is a simple, comprehensive and nearly universally-applicable scheme. It is proven in practice across a wide variety of fields and applications.

https://jauntyjjs.github.io/PyDataGlobal2021Talk

5: Organise your documentation into specific structures

Following in its footsteps, this is what I come up with

Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

Advice 5 Tips

Tutorial: Github README

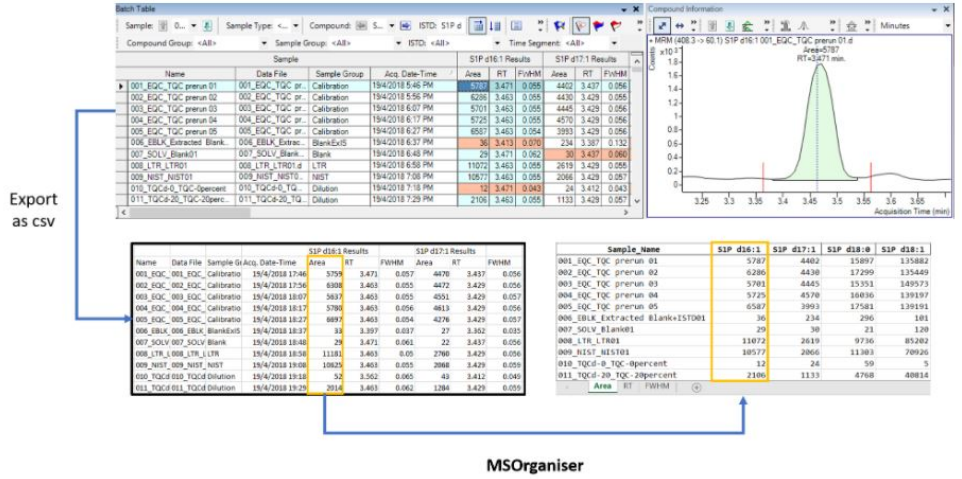
README.md

MSOrganiser

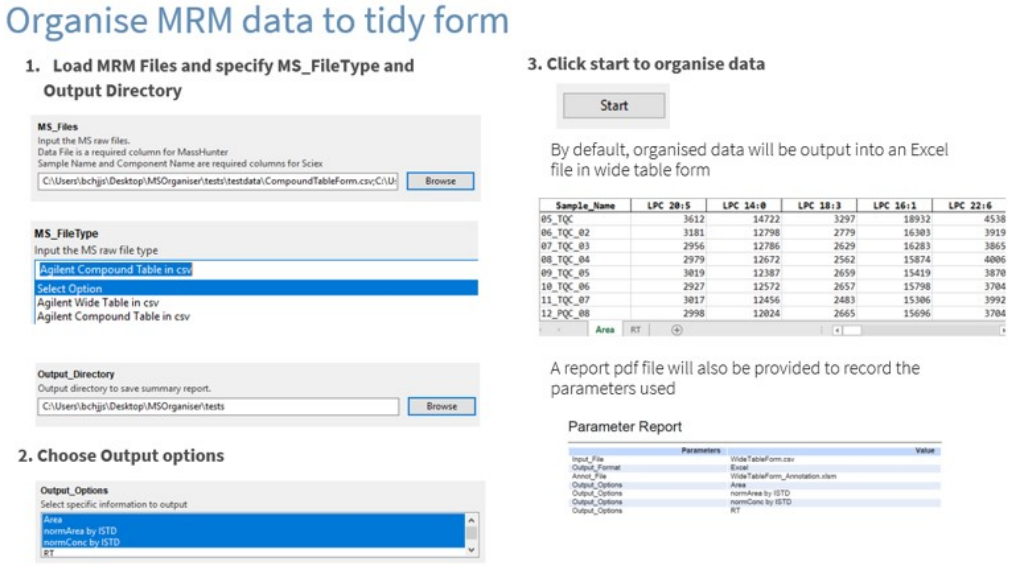
License MIT | Maintainer JauntyJJS | Python 3.9.4

`MSOrganiser` is created to provide users a convenient way to extract and organise MRM transition names data exported from mass spectrometry software into an Excel or csv file in a few button clicks.

With the addition of the `MSTemplate_Creator`, the software is also able to normalize the peak area with respect to the internal standard's peak area and calculate the concentration of the analytes.



How To Guide: Cheatsheet



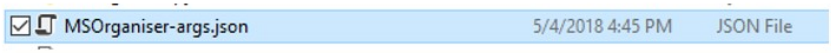
https://jauntyjjs.github.io/PyDataGlobal2021Talk

Advice 5 Tips

Explanation: User Documentation

5.7 Saving your settings with a Json file

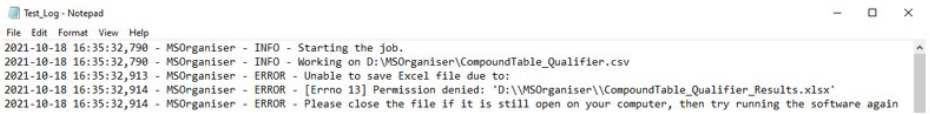
When you open MSOrganiser for a second time, you will realise that MSOrganiser is able to remember your previous settings. This is due to the presence of a json file created when the software starts to run



To clear your settings, just delete the json file.

5.8 Log files

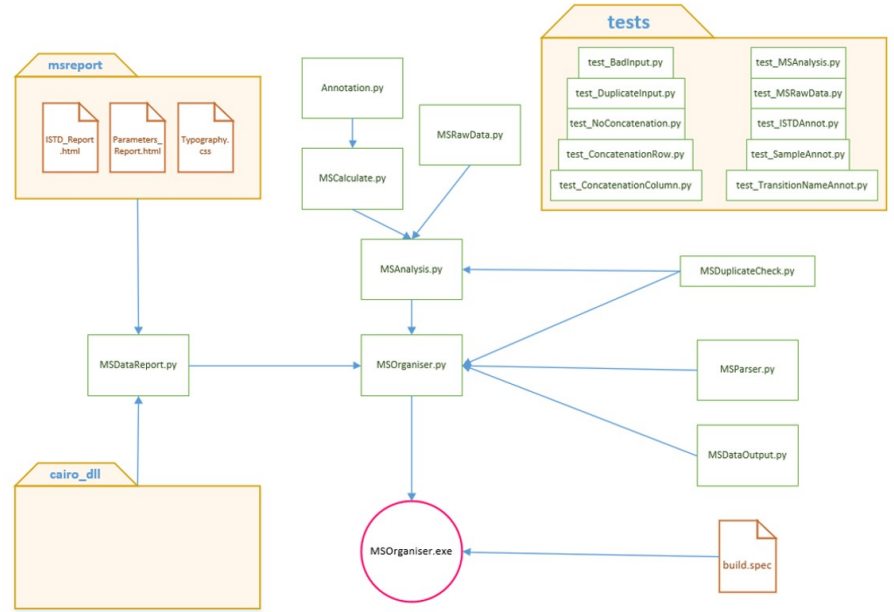
MSOrganiser also keep some log files to keep track of its work status and save any warnings that you have missed.



Currently MS Organiser is able to keep track of log files for three days.

Reference: Developer Documentation

11 Code Structure Outline



https://jauntyjjs.github.io/PyDataGlobal2021Talk

Take Home Advice:
Create a software that gives a lasting impact.

- The main purpose for creating a software is not to make us popular, it is to help others with their problems.
- A problem no matter how small can be as annoying as big ones
- The more annoying the problem the software tries to solve, the more useful it is.

Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

Take Home Advice:
Create a software that gives a lasting impact.

- Don't feel discouraged when you are tasked to create a tool that does small and simple things.
- Instead "do (these) small things with great love." -- *Saint Teresa of Calcutta*
- "If you can impact a few people deeply, they will just shout from the rooftops for you. The breadth of the impact will be a matter of time". -- *Yihui Xie* blog

Impact: Depth or Breadth?

Yihui Xie / 2018-08-29

Images by Amonrat Rungreangfangsai

https://jauntyjjs.github.io/PyDataGlobal2021Talk

All the best...

Review of advice

- Convert scripts to a web service or executable software.
- Give a software overview using cheat sheets.
- Provide helpful messages when users make a mistake.
- Include necessary software reports to show reliability.
- Organise your documentation into specific structures.

Take home advice

- Create a software that gives a lasting impact.