

Worm Lab Analysis R Script Guide

Winnie Huang (huiyan.huang@wustl.edu) and Stephanie Morrison (s.morrison@wustl.edu)

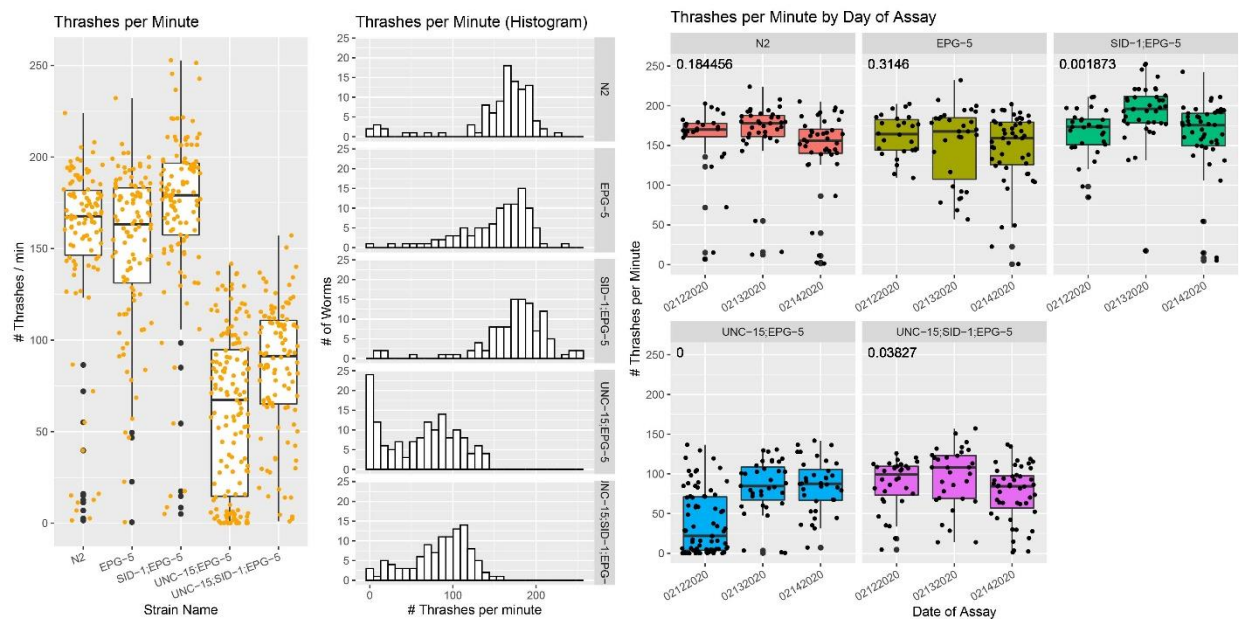
Last Updated April 30th, 2020

Index

- About
- Installation
- Data Formatting
- Running Program
- Troubleshooting

About

This is an R script that takes WormLab analysis .csv files and returns graphs and summary statistics split by strain. Example output:



Installing R and the Script

- 1) <http://ftp.ussg.iu.edu/CRAN/>
 - a. This is the R *language*, which allows your computer to understand R scripts.
 - i. If it helps, this lets your computer understand the “.R” filetype, just like how your computer knows what “.docx” means.
 - b. Install this by selecting the correct OS at the top of the screen (Linux/Mac/Windows)

- i. For Macs: There are a bunch of versions available – you must pick the version that *matches your version of Mac OS*. Click the apple in your taskbar and select “About” to see which version you are on. If you are still getting OS updates, you’ll probably be downloading one of the “Latest release” links
 1. As of right now, Catalina has a different link than all other versions.
 2. Read the notes on this page carefully to decide what version you should be downloading. Keep this page open while running the script for the first time – you may need to download additional applications (listed in the text on this download page) if things don’t run smoothly the first time.
 - c. R comes with a confusing user interface, so we’re going to be using RStudio to interact with R code instead:
 - 2) <https://rstudio.com/products/rstudio/download/>
 - a. This is RStudio, a program that’s very useful for writing and running R Scripts
 - i. If it helps, think of this as the program that reads and writes “.R” files, just like how Word reads and writes “.docx” files.
 - b. Download “RStudio Desktop Free” – we don’t want the paid version or the server version
 - 3) Download the R *script* at <https://wustl.box.com/s/ib1l31yhnwz598ad9x1d5hlvzsxrldcu>
 - a. Put the script in a folder you can find again
 - b. This is the “.R” file that contains the worm lab analysis program. It’s correct to call it either a script or a program. You will use RStudio to run the script, which your computer can run because you also installed the R language.

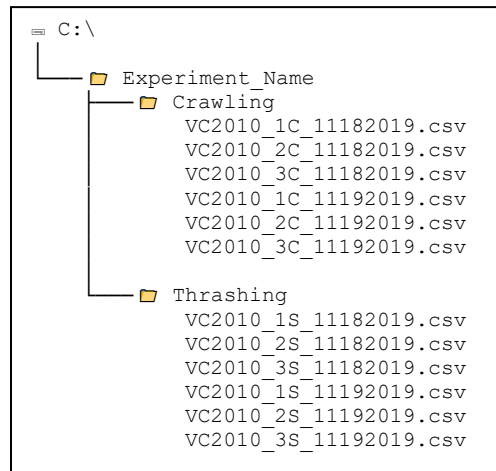
Data Formatting

Your data needs to fit some conditions before we can run it on the R Script

- Your thrashing and crawling .csv files must be in separate folders
- The .csv files must be in the format STRAIN_TRIAL_DATE.csv (e.g. “N2_10001_02122020.csv”)
 - STRAIN should be the exact same sequence of characters across the all files – spelling errors will lead to “extra” strains being created and thus inaccurate data
 - “TRIAL” really just means any extraneous information the investigator wants to keep track of, e.g. replicate, time of day, thrashing vs. crawling
 - DATE can be in any format, though YYYYMMDD is our lab standard
 - The script uses underscores to get information from the filenames – underscores in the wrong place or in incorrect numbers will lead to confusing and/or inaccurate output
 - If you have more than two underscores, know that everything after the first three “clauses” will be ignored. That means that if your file is in the structure

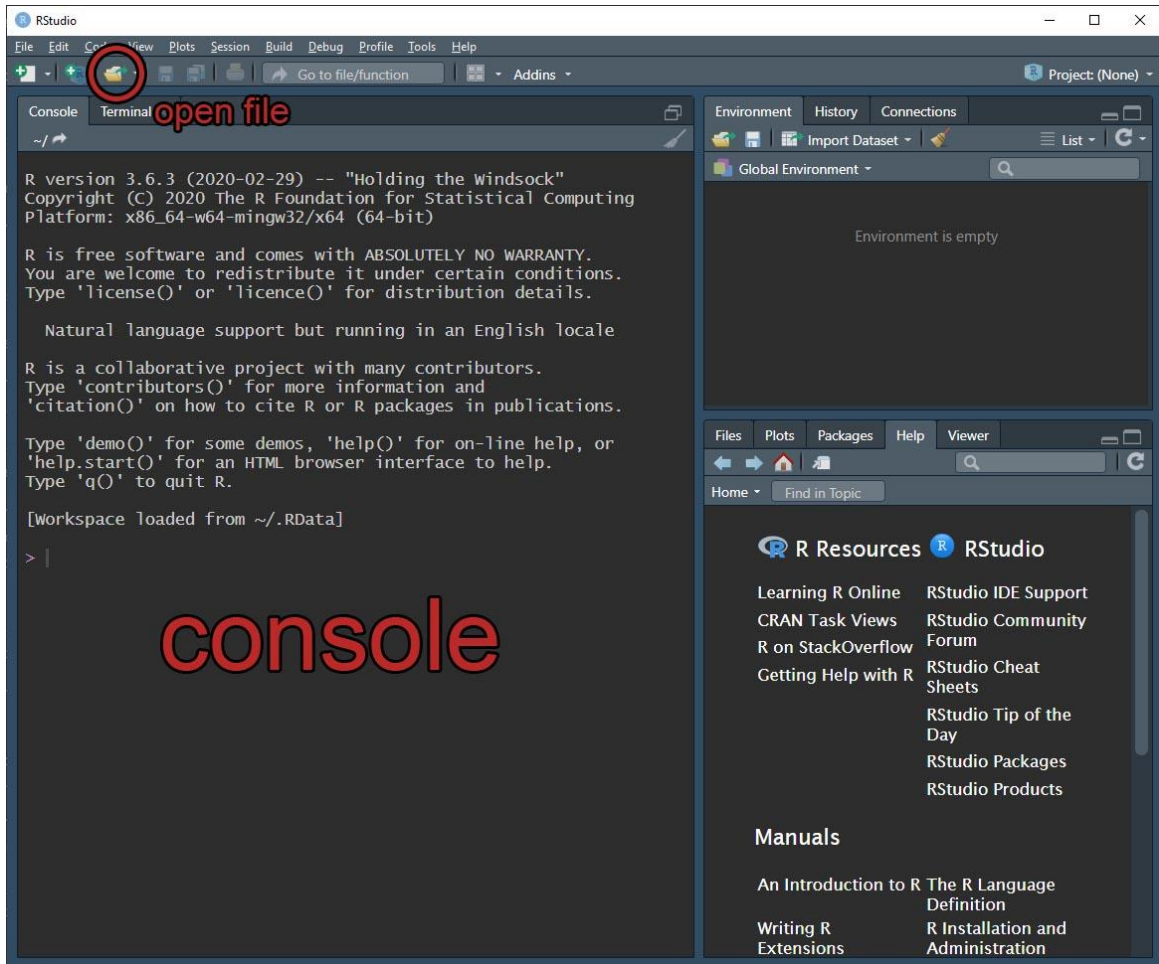
STRAIN_TRIAL_DATE_Track Summary.csv, for example, the program will still work correctly even if it warns you that the files look non-standard.

- It can be a pain to manually re-name a large number of files – google methods to batch rename files, or ask Steph/Winnie for help renaming them with a script
- Here is an example of a correct file structure:



Running the Script on your Files

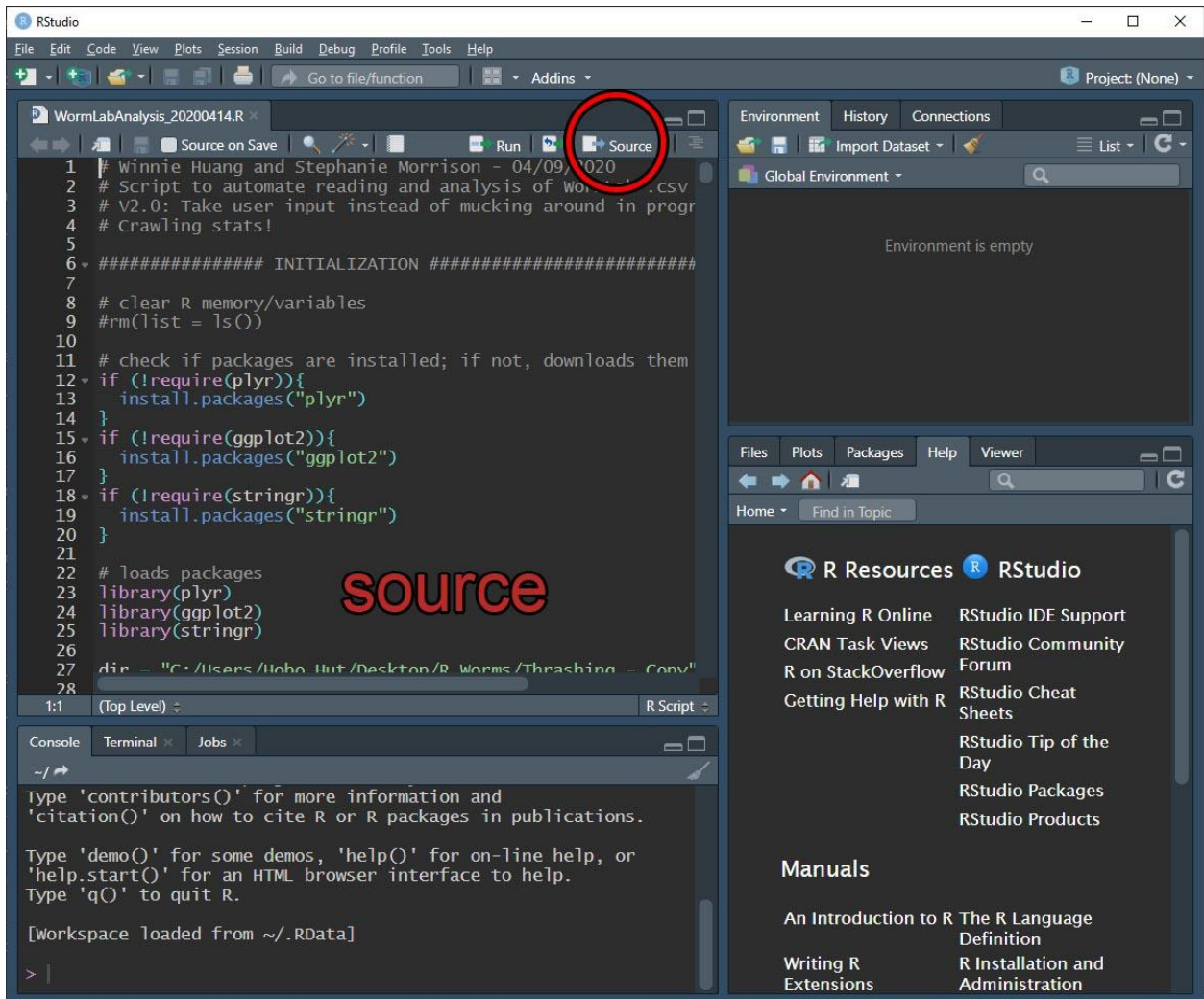
- 1: Open RStudio



You will get a window that looks something like this (but probably in a white/grey theme). The window on the left is the “console”. This window lets you enter R functions without the use of a script.

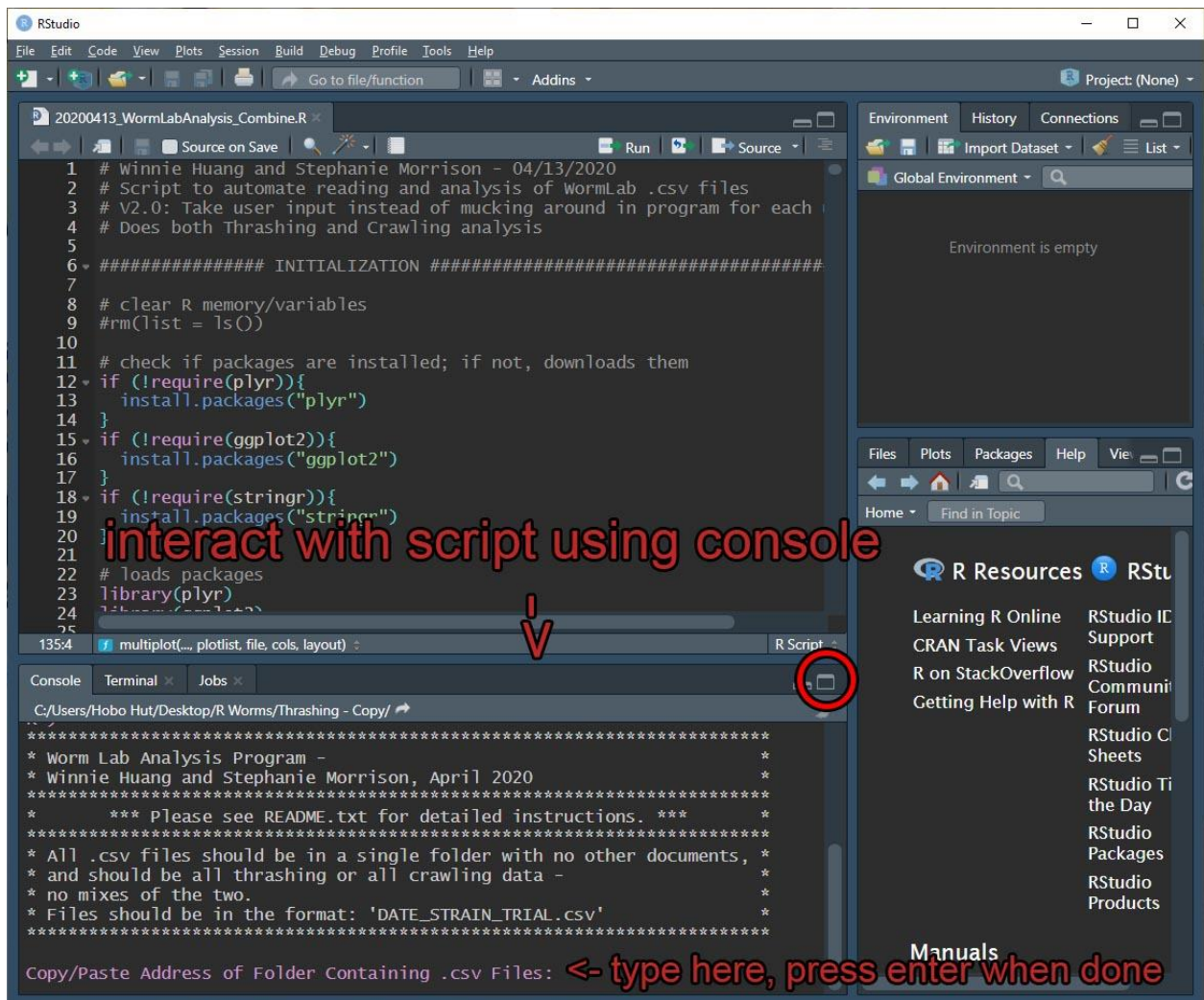
We will use the console to interact with the script when it’s running.

2: Open the WormLab script by clicking the open button on top and navigating to where you stored the file. This is the file you downloaded from WUSTL Box; it should look something like “20200429_WormLabAnalysis.R”



Upon opening the script, the “source” window will appear. This is where you can write your own R scripts. **Be careful that you don’t accidentally write in this window**, otherwise the code will stop running as intended.

3: Run the script by clicking the source button.



The script will start running on the console window. You can maximize the console window by clicking the maximize button (circled) if you want to see things more clearly. This will also make accidentally writing in the “source” window much less likely. Just know that *you will need to re-open the source panel every time you want to re-run the script (by clicking “Source”)*. The divider on the right of the console can also be dragged to the right to make the console wider.

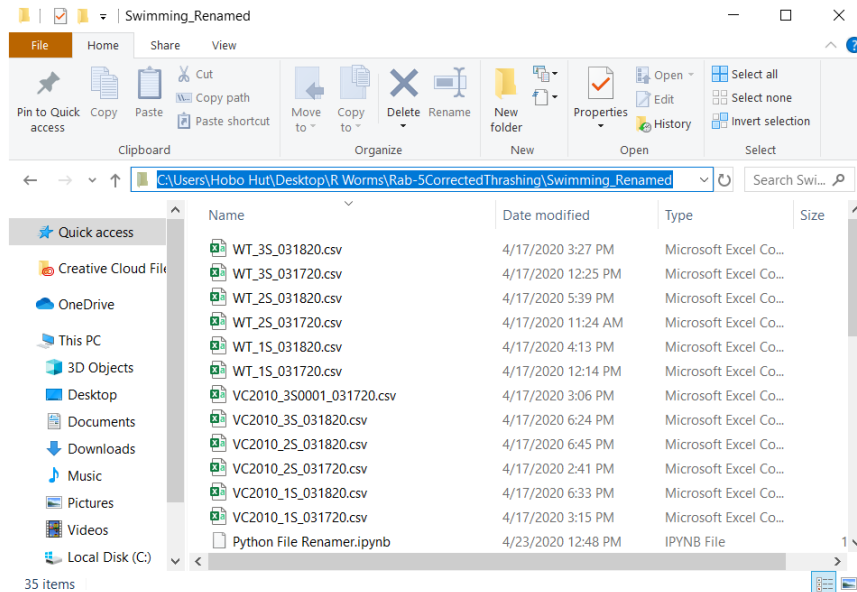
For the first run, the script will install packages from the internet that are needed to run the program. Enter ‘Y’ every time you get prompted for permission to install something. If things are successful, you can then proceed to running the program. You may need to hit “Install” on a yellow banner across the Source panel. If you see errors, you will likely need to download something to ‘help’ R download and install these packages. Check the notes on the download page you got R from, make sure your version is correct, and see if there are links to programs you’ll need to download for R to work properly.

Alternatively, or if the R website doesn’t help you, Google the errors you are getting and look for solutions from people with the same problem as you. If something asks you to enter something into the Terminal, go to the “Terminal” tab on the same window in RStudio the Console is in.

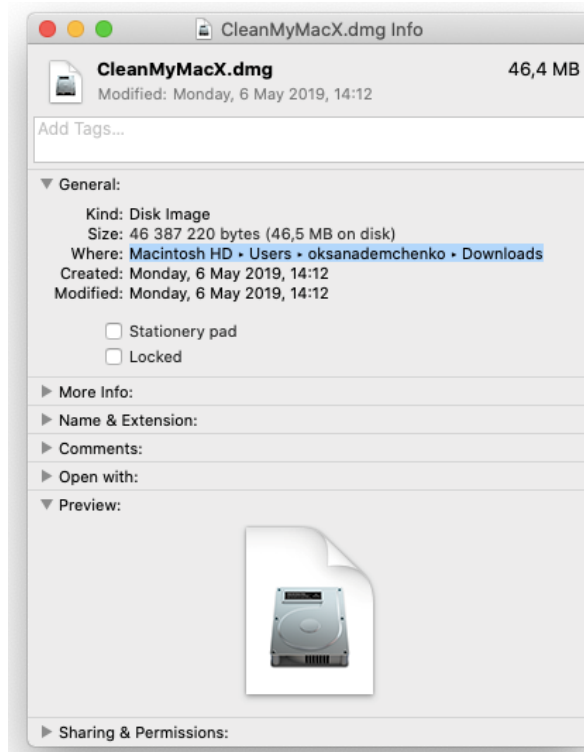
If you ever notice in the middle of running the script that you need to change something, or if you ever make a mistake, just hit “esc” and the script will stop running. You can restart it by hitting “Source” again. The script will “forget” everything each time you restart it, so you get a clean slate each run.

First, the program will ask you for the file path to the folder containing your .csv files to analyze. (File path and Address mean the same thing in this context – just where to find the .csv files)

- For Windows users:
 - Select the file path in the File Explorer address bar:



-
- For Mac users:
 - Right click (or ctrl+click) one of your .csv files and select “Get Info”
 -



○

- (Example from <https://macpaw.com/how-to/get-file-path-mac>)
- You will get an information window like above. Highlight the entire path next to "Where:"

4: Copy this address and paste it into the console, then press Enter. The program will not show the full filepath after you paste, but it has received all of it. The text has just wrapped to a second line inside the response area that you can't see.

The program will do a cursory check for filename errors. If it detects any, it will show a warning:

```
*** Warning: Non-standard file names detected. ***
Recommend stopping analysis and re-checking file names.
```

If you see this, hit esc, check through your files again and make sure they are following the STRAIN_TRIAL_DATE.csv format. Each file should have exactly 2 underscores.

After correcting file names, restart the script from the beginning.

Not getting this warning is not a guarantee that your files are error-free, nor is getting the warning a guarantee that your files won't work – it's just a simple check for your benefit.

```
Copy/Paste Address of Folder Containing .csv Files: C:\Programs\strain_date_trial\datasets\thrashing
Select the strain to be displayed in slot 1 of 4
('0' will exit program)

1: t90i1
2: t90i22
3: t90t
4: vc2010

Selection:
```

The program will now ask you to order the strains by how you want them to appear in the graphs. This list of strains is pulled directly from your file names – if it doesn't look like what you expect, you need to double-check your files.

5: Enter the number next to the strain you want first.

The menu will re-appear with the first strain removed

```
Selection: 2
Select the strain to be displayed in slot 2 of 4
('0' will exit program)

1: t90i1
2: t90t
3: vc2010

Selection: |
```

Note how the strains are re-numbered starting from 1 again! Make sure to double-check what number the strain is after one has been removed.

If you accidentally order the strains incorrectly, just hit esc and start over from the beginning.

6: Keep entering until all your strains are assigned a position

The program will now ask you for the “display name” for a given strain. For example, your files may have “epg5” as the strain name, but you actually want the graphs to say “Epg-5”.

7: Type what you want the displayed strain to be called and press enter. Keep naming your strains until you are done.

The program will ask you if you want to save the strain name data for future analysis. If you say yes, two .txt files (both start with the # symbol) with the strain order and display names will be saved into your directory with your .csv files. If you need to re-run the program for whatever reason, this allows it to remember your earlier choices and avoid the ordering and naming steps.

You can move these two files into another analysis folder (e.g. from Thrashing to Crawling) if the strains are the same and you are confident your files are named correctly.

8: Enter Y or N

```
Would you like to save the strain name data for future analysis?  
Y or N: n  
  
What is the name of this experiment?  
(This will be appended to each result document generated, so make it short.)  
Experiment Name:
```

The program then asks you to name the experiment. *This should be just a brief, one-word description or ID* so that you can remember what experiment the data came from. Example: “Rab-5”

It is to your benefit if you keep this short, because this will be added to every result file this program makes and can make absurdly long file names.

9: Name your experiment something short

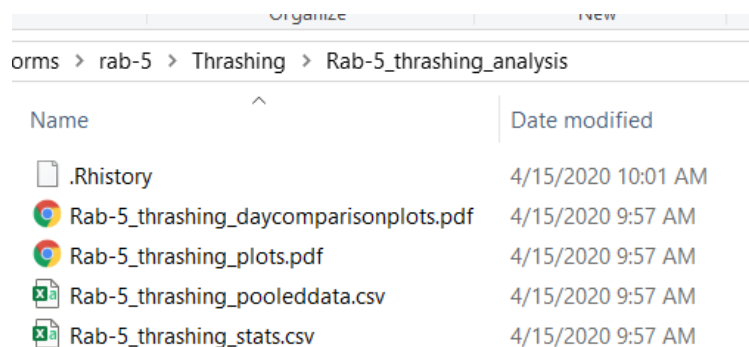
```
Are these files Crawling or Thrashing data?  
  
1: Crawling  
2: Thrashing  
  
Selection: |
```






Now you must select what analyses this program will run on your files. (You may know “Thrashing” as “Swimming,” but we are moving towards using “Thrashing” exclusively.)

10: Enter 1 or 2 to indicate Crawling or Thrashing

The program will now spit out a couple of numbers and end. You will know the program is done because instead of a prompt (like “Selection:” or “Y or N:”), the bottom line just has an arrow (>).

Navigate to the folder you entered into the program. You will see a folder created inside it with your experiment name, analysis done, and the word “analysis.” (Example: Rab-5_thrashing_analysis)



Organize View	
orms > rab-5 > Thrashing > Rab-5_thrashing_analysis	
Name	Date modified
 .Rhistory	4/15/2020 10:01 AM
 Rab-5_thrashing_daycomparisonplots.pdf	4/15/2020 9:57 AM
 Rab-5_thrashing_plots.pdf	4/15/2020 9:57 AM
 Rab-5_thrashing_pooleddata.csv	4/15/2020 9:57 AM
 Rab-5_thrashing_stats.csv	4/15/2020 9:57 AM

Inside this folder will be two .csv files: One with all the raw data gathered into one file, and one with statistics on comparisons between the strains involved in your experiment.

There are also two .pdf files: _plots.pdf is a strain-to-strain comparison. _daycomparison.pdf shows all your strains divided by date so outliers can be easily identified (the ANOVA values on the _stats.csv refer to these comparisons.)

11: When you exit the program, R will ask if you want to save – always say no. You don’t want to save over the script if you accidentally typed inside the source window, and there is no need to save the workspace because you want it to be “fresh” every time you run the program.

Troubleshooting

- The program can be stopped at any point by hitting the esc key. Just re-start by hitting Source again
 - This is where having the .txt files with strain name data comes in handy – entering the strain order and display names takes a lot of time, so loading from these files speeds up

re-runs of the program. Just make sure that your strains were correct when you made these files, or the program may break.

- If the program asks you to order/name strains that do not match your actual strain names, double check your naming scheme. It's likely that you have something other than STRAIN_TRIAL_DATE.csv and the program has gotten confused.
- If you get this error:
 - **Error in setwd(readline(prompt = "Copy/Paste Address of Folder Containing .csv Files: ")) : cannot change working directory**
 - The program can't find the filepath you entered. It's possible something is misspelled or missing in your filepath
 - Make sure that your folder holding your .csv files doesn't end in a space. The program is not able to see the space at the end when you enter the filepath and you will be stuck with this error.
- If you get this error:
 - **Error in t.test.default(x, y) : not enough 'y' observations**
 - All your tests are on the same date. The program assumes you will be entering data for at least two different dates for each strain and will break if this is not the case.
 - This probably happened because you were curious to see the graphs on a particular day and selected only a few .csv files. I recommend looking at the day-to-day comparison plots instead; each strain's graph will have the same y-axis as the others, so you can still do comparisons between strains.
- If you get this error:
 - **Error in rbind(deparse.level, ...) : numbers of columns of arguments do not match**
 - This happens when some of your .csv files have more or fewer rows than other .csv files. Might result from the WormLab video analysis program getting an update and adding new rows – though this should be fixed for now.
 - Check through your .csv files and look for discrepancies between them. Consider re-deriving the .csv files from the video analysis program.
- If you get this error:
 - **Error: unexpected '/' in "/"**
 - This happens when the program is not actually running but you are trying to enter the filepath. Re-run the program by hitting "Source" as explained above. You may need to un-minimize the Source panel, if that's something you did previously.
- If you have some other error:
 - First try googling the exact error and see what comes up. Sometimes errors are common enough that solutions are findable on the Internet without much effort

- If you get stuck, feel free to contact Steph (s.morrison@wustl.edu) and ask for help troubleshooting. Please send all information you can so that your problem can be added to this README for future users. Thank you!