Python merge_csv protocol

Daniëlla Zinn, daniellazinn@hotmail.nl

This script merges CSV files created by the ImageJ set_environment.ijm script. Separate files will be gathered as multiple sheets inside one Excel file. Data is filtered and contrast for each point is calculated, by subtracting data for different counters.

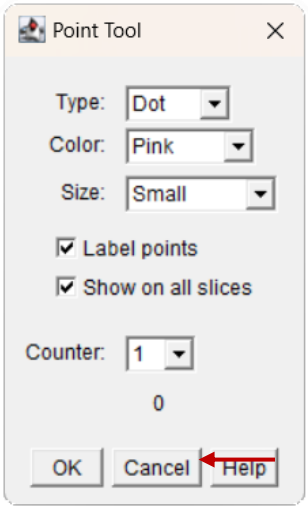
1. First, it is important to have CSV sheets with columns that show the mean, counter, and count for example for every brain slice. This can be achieved by running set_environment.ijm, as described in the ImageJ set_environment protocol, but could also be done by hand. Make sure all files are in the same folder.
2. **Download** merge_csv.py.
3. **Download** something to visualise code, such as **Visual Studio Code**, which will be used in this protocol.
4. **Personalize** the script.
    a. Change the content of the "**measurements**" variable into your own path. This should be the folder in which all CSV files exist, for example:

```
14    measurements = r"C:\Users\Danie\Documents\Internship\Counting\Measurements\*"
```

    Since the backward slash "\" is a special character in Python strings, an "r" should be placed before the string. If the path includes forward slashes "/", the "r" can be removed. Important to add an asterisk "*" at the end of the path, indicating all files from the folder.
    b. Change the content of the "**merge_data**" variable into your own path. This should be the path to an empty Excel file that will be used to write all data to (so the code should end with ".xlsx":

```
16    merge_data = r"C:\Users\Danie\Documents\Internship\Counting\Merge_data.xlsx"
```

    c. Check which **counters** you want to include into the Excel sheet. "Counter" refers to the number in the ImageJ Point Tool, indicated with the red arrow on the image below:



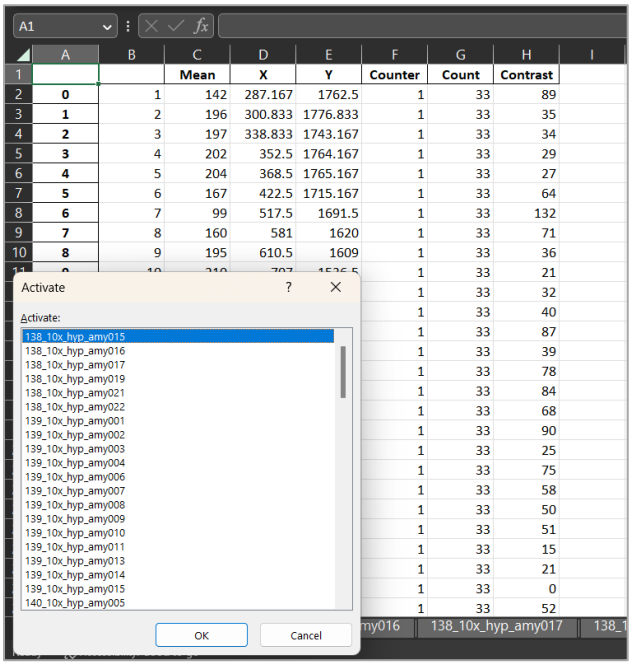
    i. The code assumes that 1 and 2 are the counters of interest and that 3 and 4 were used for measuring colour contrast. This would be done by placing points in the background and places of interest, for example:

    ii. If counters 3 and 4 were not used for this purpose and all used counters are of interest, **delete lines 35-49** in the script of merge_csv.py.
    iii. In this example the RGB Decimal Code data from 4 would be subtracted from 2 to calculate the difference (always subtract light from dark).
    iv. If the contrast data is wanted, check whether the code applies to your situation. Most important to check are the red underlined parts of code.

```
# initialise an empty pandas Series called contrast_one, which will be used to calculate and store the contrast value for counter 1.
contrast_one = pd.Series([], dtype=float)
# initialise an empty pandas Series called contrast_two, which will be used to calculate and store the contrast value for counter 2.
contrast_two = pd.Series([], dtype=float)

# if counter 1 and 3 values are present, subtract counter 1 values from the mean of the counter 3 values
if 1 in data["Counter"].values and 3 in data["Counter"].values:
    mean_three = round(data[data["Counter"] == 3]["Mean"].mean())
    contrast_one = mean_three - data[data["Counter"] == 1]["Mean"]

# if counter 2 and 4 values are present, subtract counter 2 values from the mean of the counter 4 values
if 2 in data["Counter"].values and 4 in data["Counter"].values:
    mean_four = round(data[data["Counter"] == 4]["Mean"].mean())
    contrast_two = mean_four - data[data["Counter"] == 2]["Mean"]

# add a column named "Contrast", consisting of the values in contrast_one and contrast_two
data["Contrast"] = pd.concat([contrast_one, contrast_two])
data = data[data["Counter"] <= 2]  # filter the data by selecting only the rows where the values in the "Counter" column are less
# than or equal to 2, the filtered DataFrame is then assigned back to the data variable, effectively removing rows where the
# "Counter" value is greater than 2 from the DataFrame
```

5. **Run** the script after saving the changes (Ctrl+S).
    a. In Visual Studio Code this can be done by clicking the play button in the right corner: 
    However, it is likely that an error message will pop up:
    `ModuleNotFoundError: No module named 'openpyxl'`
    If this message appears, type “python merge_csv.py” (without quotation marks) inside the terminal where this message appears, right after the file location of this script, for example:
    `PS C:\Users\Danie\Documents\Internship\Cellcount_and_Analysis_Fiji-ImageJ> python merge_csv.py`

6. **Check** the Excel file, it should look something like this (the window pops up after right-clicking on the arrow in the left lower corner ):



7. **Troubleshooting**:
    a. If the Excel file does not contain any data after running the code, try typing something in the file, saving it, deleting the content, and saving it again.