Particle-Analysis.R

How to use the script, common error messages and general setup

Lindsay Scheidemann

October 26, 2023

This is the documentation for the R-Script *Particle-Analysis.R*. It is meant as a manual as well as a first reference for problems encountered. In the first part the most common problems are discussed, and an extensive explanation and introduction on how to run the script is given. The second part offers an overview on the set-up of the script and explanations on the ongoing calculations and methods used within.

Contents

Chapter 1

Manual - How to obtain results using the script?

For working with the script only, the lines 15-50 are the ones, that require actions. Based on my experience, I only ever open documentations, when things refuse to work. Therefore I will be putting the most practical part first, also to save unnecessary scrolling, and follow with the strict manual afterwards.

1.1 Error Messages . . . and now?

This section includes the most likely reasons for problems and how to fix them. The most common problems, that may occur are:

- **the helper-script:** the version of the accompanying helper-script *TEP-functions.R* is incomplete/old, not in the same folder as the main script *Particle-Analysis.R* or missing altogether. If so, please check, whether the script is there and in the right folder. Additionally the main script should contain a list of functions necessary for running it. Please check, whether all of these are in the helper-script and organize the new version of *TEP- functions.R*, if necessary.
- **Wrong working directory:** the working directory should be where you stored the scripts in the beginning, and it usually ends there...this is especially important in case some trouble occurred (file couldn't be found, etc), which forced the script to stop during runtime, and you now try to re-source the script, type *setwd(scriptloc)* into your console first and it will easily run again. In my experience this is one of the most frequent causes, of error messages.

- **using the wrong input table format (xls/xlsx):** If any other input format is used column names may vary in their exact typing, if you can't use these formats (xls is preferred), you may change to csv, tsv or other, but you will have to change column names in the script (Sorry for this, but most people do prefer Excel for looking at and working with data and for some reason it seems to loose less spelling while the file is read).

- **using wrong or incomplete input tables:** You should have received a blank xls draft layout for your table together with the script. If that isn't the case, please make sure your input includes the following columns (with specified names): "filter_ID", "volume[ml]", "treatment", "images[nr]", "filename". The order of the columns does not matter but spelling does. More on this in the section on the input table.

- **using the wrong input file format for the imageJ output:** The script can handle csv, tsv and "fake-xls" (you named the file *name.xls*, when imageJ asked you, but have not yet opened and saved it with Excel). If this is the case try re-saving it in one of these formats.

- **the histvec-vector doesn't start with 0:** by adjusting the histvec-vector it is possible to include/exclude large particles from the calculation of the slope and carbon-content, as well as adjusting the step size. However this vector needs to start with 0. The script can not handle this, since I didn't consider it necessary to exclude small particles. These are usually the most statistically reliable size range and the most abundant ones.

- **missing libraries:** the script contains a list of required libraries. Please check and install the missing package if required.

- **wrong or incomplete paths (directory):** Please don't work with relative working directories. You may, but then you have to trace all changes of working directory during the script to make sure it works. Also as usually, check the spelling.

- **typos:** check the spelling of directories, filenames, etc. Usually the error message should give you a good idea on where to look, by telling you what it could not find. Remember to execute *setwd(scriptloc)*, after you corrected the typo, before running it again.

1.2 When and for what can the script be used?

This script can take the imageJ output of analysis for several filters. It cannot do the work of setting thresholds in ImageJ and running the analysis with the correct filter-pore size and scaling given for you. As it is based on the output gained by the macro from Kathrin, it may be useful to use one of these for aid with the ImageJ-analysis. It can count the particles of different size classes, per filter, it can tell you the size of the smallest and largest particle found on the filter, it will calculate the number of particles per liter and the area of particles per liter based on the particles counted and sized by ImageJ, and, for TEP, it can calculate the carbon-content. It can already handle empty blanks (which don't have an imageJ output file, but still need to be added to the input table, because the script can't predict filters).

There are some assumptions on which the Script relies. Please check them and modify the script where necessary, if any of these aren't met:

- You have used 25 mm diameter filters for the filtration. In case you haven't, the columns per liter will be wrong, since the surface of the filter is wrong. So in the event, that you used another filter size, adjust these formula for you or transform the script so that filtersize is another input-variable, if you expect this to be useful.

- The pictures where taken with one of the current AG-Engel Microscopes for TEP and CSP analysis (October 2023). This turns crucial, when it comes to the magnification, a) because the script is only familiar with the magnifications that can be used with this microscope right now and b) because the area covered per image can be wrong otherwise, which will result in the per liter columns being wrong. If you used another microscope, or magnifications have changed, add to the *obsarea* function in the accompanying function-script.

- You don't use relative working directories. The script is designed, so that you don't need to have all files and the scripts in one folder. You may have the scripts separate or you prefer to keep input and output-files in different places. That is totally fine with the script, but that also means, that it jumps between directories to find everything it needs and to store files in the end. If you really prefer to use relative directories, check where the script comes from when it goes where and keep scriptloc global.

- You have already run the ImageJ-analysis for the respective filters and saved the output in a suitable format (csv, tsv or ”fake-xls” (not yet opened and re-saved in Excel))
- You have made sure the input-table is correct. Apart from the information you have to give to the script in the beginning, all further information are taken from the input table. If this table contains a mistake, this will result in wrong values for the filter where the mistake was.

1.3 Setting up the script

1.3.1 Summary: files, programs and libraries on your computer - Requirements for the R-Script

Files: Script (Particle-Analyis.R), function-script (TEP-Functions.R), ImageJ-output-files (all in the same folder; csv, tsv or fake-xls), Input-table (xls/xlsx).

Programs: R, at least 3.4.4 (as that was the first version I started the script in, I am not sure about older R versions but they most likely will work)

Libraries: The package readxl is required, apart from that the script works with base R

As it is based on the output gained by the macro from Kathrin, it may be useful to use one of these for the ImageJ-Analysis. Your output files from image-analysis definitely need the column “Area”. If you don’t use the macro, please make sure, that this column exists.

1.3.2 The input table

Initially, the input table was meant to allow for a more intuitive and a less error-prone way to enter the necessary information. Over time this allowed the script to become more generic and turned out to be a more useful tool in general work with the filters, as the table is the first place to collect the information. As such the table can no longer be disentangled from the script or replaced. Different file formats may however be used, depending on how the format is read in (are the column names read correctly?), you may have to adjust all the column names, where input$[columnname] is called. However this problem should not occur with Excel-Files.

Table 1.1: *Example from an input table*

filter_ID	volume[ml]	treatment	images[nr]	filename	threshold	particles
D0-Ag1-A	50	D00-Ag-1	34	D0-Ag1-A-TEP.csv	30	344
D0-Ag1-B	50	D00-Ag-1	34	D0-Ag1-B-TEP.csv	30	225

Table 1.1 shows the first two lines of an input table. The columns "filter_ID", "volume[ml]", "images[nr]", and "filename" need to be filled for the analysis otherwise the script won't run. "filter_ID" should contain any identifier or name you give to the specific filter. It may contain systematic numbers like "12-A", or just "12", more complex names like "station_5_depth_3_B", or anything else (name them "Hans" or "Dieter", if you like), as this is kept as a character string within the script. As in general with programs, please be careful with (=avoid) " " (blank spaces) in the name, they should not cause trouble, but you never know and the filter_ID is used as a name during handling of the histogram data. No two filters are allowed to have the same filter_ID. "volume[ml]" should contain the volume filtered onto the specific filter in ml; "images[nr]" should contain the amount of images, not that you have taken from the filter, but that you have included into the analysis (=stack) in imageJ. Neither "volume[ml]" nor "images[nr]" are allowed to contain anything else than numbers. The last really important column "filename" should be filled with the complete name of the file, including the fileending. The script will not work, if you put the filename into quotation-marks, because this automatically happens, when the input table is read. In the case of empty filters (usually blanks) there will not be an output file by imageJ. In this case you just leave the "filename"-column free for this specific filter. If this causes trouble, usually with a different input format, type "NA" (without the quotation marks).

The remaining columns "treatment", "threshold" and "particles" are of a more practical origin. If you want to have the data plotted, you must fill the "treatment" column, this also turns useful, if you continue to process the results in R, since this information is copied into the results-table. The treatment may have any format (still be careful with special characters, formatting may get lost) and may include blank spaces without any risk of causing trouble. In the above example (see 1.1) the text in the column combines different information (time and Mesocosm), that can be separated and backtranslated by a different script later on which allows the user to pass different information with this one piece of character string for later use in R. "threshold" refers

to the threshold set in imageJ, this information is not used by the script at all, but can be useful in case files get lost, or you have to return to your imageJ analysis for some reason. Similarly the ”particles” column contains the number of particles counted by imageJ during analysis. This information may get useful in case you have to redo imageJ analysis, because of lost files, or if there is confusion about the file. The script will use that number, but not from the input table. Instead it will read this number directly from the imageJ-output file specified.

In general you may add any column, that is useful to you, or of interest, or maybe just crosses your mind. You may also freely rearrange the columns to support your workflow or thinking patterns. Trouble will only occur, if a) the necessary columns are empty (except ”filename” or filled with the wrong format) or b) the spelling of the necessary columns or ”treatment” changes, without being changed in the script. b) also applies to errors or formatting consequences when the input-table cannot be read correctly. You don't have to create a new Excel-file for each input you have, you may also use different sheets within one file. Please don't mix different analysis within on Excel sheet.

Suboptimal input that will cause trouble and has to be avoided includes, apart from above mentioned, empty lines and repeating filter IDs. Empty lines will, depending on format, either include empty lines, which will cause trouble, or they will cause the file-reading process to stop with the empty line, so the later parts will be lost. Also it is advisable to start with the top line of the Excel sheet (as the column names) but this can be adjusted within the read_excel command. Repeating filter IDs cause the script to stop, because it cannot actively figure out where to place its results in the table. As usually two samples don't have the same name this should not occur, but it can happen as a consequence of restructuring, so please delete lines, that carry the same filter_ID as others. Further the first line may never contain an empty filter, so if your first sample is an empty blank, please move it to another position in the input-table.

1.3.3 The variables you have to declare and their meaning

Following you find a list of all the variables from the input-part of the script, and one line of code, where you are expected to make sure your information are correct and what is expected there. These are kept in the order in which they occur and the line of the script is added. Please keep in mind, that the line in the script may change a bit over time, if you add or delete lines, so you might have to check the adjacent lines as

well to find the variables.

- scriptloc (line 17): Here you need to give the directory where you have stored both scripts. This information needs to be global.
- magnification (line 27): Here you need to give the total magnification used to take the images. So far this can only take the values 100, 200, 400 and 1000. 200 is usually used.
- histvec (line 28): as the name implies this one is a vector. It has to start with zero (otherwise the script will stop). It contains the borders of the bins for the histogram of the observed Particles. This one should be given some consideration, since larger than maximum particles will not be included into the TEP-carboncontent, or any other size specific analyses that may be added at later points.
- method_sizedis (line 29): This one decides on the adjustments done to your dataset to allow for the linear regression of the log/log transformed data. The method "AG_Engel_Standard" is recommended and will set itself in case the method given doesn't exist or there is a typo; this process comes with a printed information. "AG_Engel_Standard" means that all counts smaller or equal to ten will be excluded due to bad counting stats. Method "Excel-Mastersheet" cuts the data at a maximum ESD of 33 μm and converts all zero counts to one before log/log transformation. The method "Mari&Kiorboe(1996)", refers to the method used in the respective paper. Here data are cut below 3 μm and again after 40μm. This is because they traced the particles manually, so small particles are underrepresented. Large particles were excluded due to bad counting stats (see the paper for more information on the method or size distribution in general). The last method "zerotail_omit" cuts the data after the third zero in a row to occur in the data. This may contain useful information, but tends to underestimate the slope.
- Particletype (line 35): This is most of the time just an information for yourself. It only gets interesting, when Particletype is "TEP" because then the script will calculate the carboncontent. This does of course not make sense, if you have CSP or other Particles. In this case you may add extra analyses that you are aware of.

- filterdir (line 37): similar to scriptloc this contains a path. The respective path should lead to the folder, where you saved all your imageJ-files listed in the input table. Alternatively you have to add the full path into the filenames, in case these vary between filters. In this case please give the path to the first filter's directory here.

- tabledir (line 38): This path should describe where you have saved your input-table.

- $input <- read_$excel("input_test.xlsx", sheet="Vorlage", col_names=TRUE) (line 40): this might look a bit scary at first, but you only have to adjust the spaces that are marked in blue. It refers to your input-table. The read_excel command will automatically guess, whether your table is in xls or xlsx. The first blue space should include the filename (including the ending). Further you will have to adjust the name of the Excel sheet to the one that your input-table has (second blue part) col_names=TRUE will tell R, that your table has column names, which will therefore be taken over. Please don't change this one, because the script recognizes its information by column name and Filter_ID.

- Plotresults (line 44): This one is meant as a question. Do you wish to see some very basic boxplots about your data? If so answer with TRUE. This will plot your data by treatment, so it will turn confusing, if the treatment column contains many different levels. Also this is not possible, if the treatment column was left empty. If you don't want to see plots, or you want to make others, that better fit your data, answer with FALSE.

- saveresults (line 46): same as above. Would you like to have your results in an Excel-file? Yes=TRUE, no=FALSE. Even if you save it as an Excel, you can as well save it manually as .rds or whatever you want. The Excel is not an Excel-file to start with, it is tab-separated, as this by experience goes well with importing into Excel.

- decimal_seperator (line 47): As a consequence of the fact, that the Excel is not by nature an Excel, but a tsv, you may make it even less complicated to read the file into your Excel by choosing the seperator that works with your default. "," will cause numbers to look like 3,14; "." will cause numbers to look like 3.14, other seperators would be possible, but usually don't make much sense (and

you may find yourself having an awfully hard time trying to explain the numbers to Excel).

- filename (line 48): If you want to save the output, what do you want the file to be called? Despite the above, it usually pays to directly choose the ending xls, as this will try to open in Excel by default on most computers. You have to give the file with an ending.
- savedir (line 49): If you want to save the output, where should the file end up? You have to give the path to the desired directory here, which also means you do not have to include it into the filename.

After having given all these information and providing the input-table, the script should now be running on your computer. Lean back and observe (hopefully). If error messages occur the first part (Error Messages...and now?) covers the most frequent troubles that may occur. Otherwise a print may tell you where the trouble is, if not I am afraid you may have to search.

1.4 Information and communication by the script

So the script is running (great), hopefully it should also communicate to set the user at ease and inform you about problems, observations, or its state of affairs.

1.4.1 The prints-what does the script tell me and when?

Figure 1.1 shows what the general output looks like. After starting the script (here with the source command done from the console window), it will tell you which filter (Filter_ID) it is working on and a number that increases by one with each filter. That is the normal minimum output, depending on settings and filters, there won't be more. In image 1.1 you can also see the most frequent non-standard output (after filter 16). Filter D7-Blank-B was not associated with a filename in the input-table. This usually only happens for empty-filters. R prints the question whenever handling an empty filter, so you, as a user, can check with the filter-ID whether this is what you actually want or whether the filename just escaped the input-table. Further, if you have chosen to save the results (saveresults=TRUE) the script will print a short note, when it has finished saving the file. If you have chosen plots, a plot window will open, or they occur in the plot-window of Rstudio as soon as the script has finished. All

other prints only occur, when the script has found a problem. They are supposed to help you solve the problem by being more specific and possibly less cryptic than the error message occuring next.

1.4.2 The output graphs

When you have the output the script finally ran without error. So you can now worry about the output. If you have graphs these are simple, unformatted box-plots by treatment. The graphs come with labels that explain, what you can see in each of the three graphs. They are not coloured or anything. The only purpose of these graphs it to get a first idea. Figure 1.2 shows an example plot. In the upper left the concentration in mm^2/l is depicted, grouped by the respective treatment. The top right plot shows the same graph for particle abundance in millions of particles per liter. The calculated slopes of the log-log-transformed size-distribution is shown in the lower left graph, if a suficient percentage of the filters allowed for calculation of the slope. The lower right graph is a variation of the top-left one. The difference is, that in this graph the single data-points are shown which allows the user to get a better idea of the observed distribution within the different treatments. If there are to many groups, the graphs may become confusing. Also as can be seen in Figure 1.1 to long treatment names are sometimes automatically omitted if the space along the axis doesn't suffice to

```
> source("Particle-Analysis.R")
1         D0-Ag1-A
2         D0-Ag1-B
3         D0-Ag2-A
4         D0-Ag2-B
5         D0-Ag3-A
6         D0-Ag3-B
7         D0-AgNP1-A
8         D0-AgNP1-B
9         D0-AgNP2-A
10        D0-AgNP2-B
11        D0-AgNP3-A
12        D0-AgNP3-B
13        D0-Blank-A
14        D0-Blank-B
15        D0-C1-A
16        D7-Blank-B
[1] "No particles on filter?"
17        D7-C1-A
```

Figure 1.1: **Normal output of an Error-free run with one empty blank**

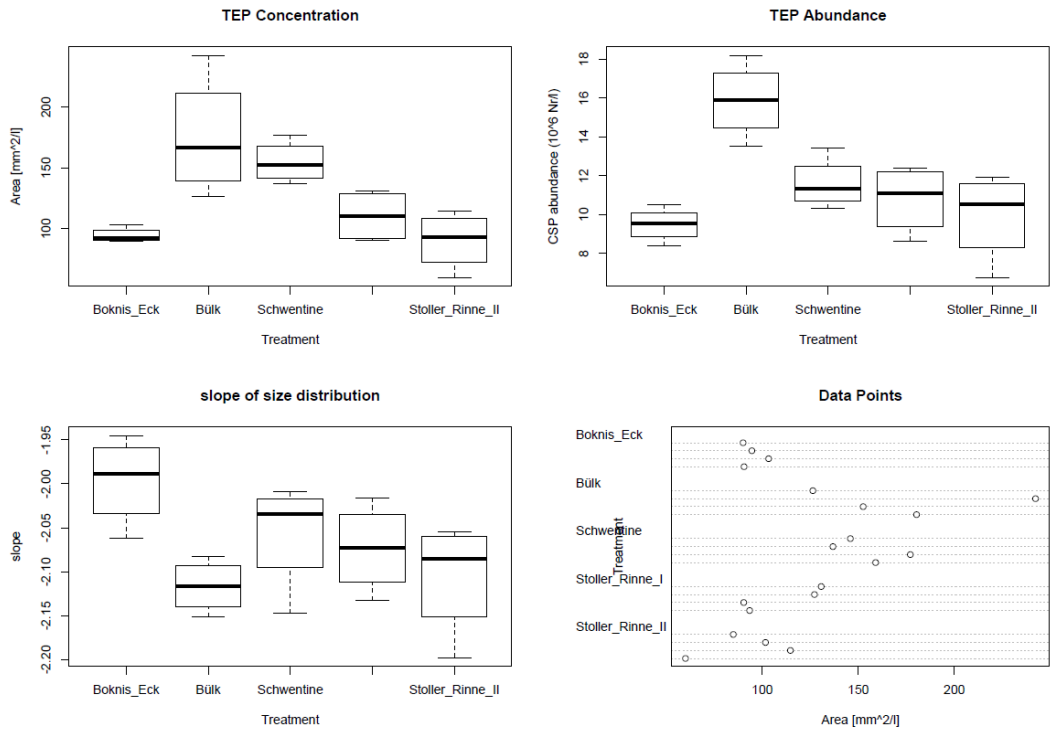

Figure 1.2: **An example of the standard output graphs showing different stations from the Kiel Bight from October 2018**

put them there, as happens with the label for "Stoller_Rinne_I". In this case it might be helpful to know, that R sorts the treatments alphabetically for plotting, since no instruction on how to sort them is given in the script. The output-graphs are not saved automatically: if you like them for further reference, manual saving is always necessary.

1.4.3 The output table

The results-table and the Excel are the same, just some column names differ, since they need to be more practical in R, but it can pay off to have them more plain in Excel. One is saved on your computer, the other one only exists within the R-environment (if you don't save it otherwise). It has several columns:

- Filter_ID: this one is taken over from your input table
- Number of Particles (observed): The number of Particles imageJ counted within your stack
- filtered Volume [ml]: information is taken from the input-table

- Number of Pictures taken [per filter]: information is also taken from the input-table
- Number of particles $[*10^6/Liter]$: here the particles were calculated to the unit millions per Liter, so this value is corrected for the filtered volume and the observed area. The value was divided by a million to get more nice values to work with.
- Area of Particles (observed): in this column you find the sum of all the areas of the particles that have been counted by imageJ in the stack. It is in μm^2.
- Area of Particles $[cm^2/Liter]$: this contains the value of the previous column corrected for the filtered volume and the area observed. Note that the unit has changed to cm^2 to have nicer numbers.
- Treatment: This one is taken from the input-table
- slope: In this column you will find the slope of a linear regression of the log/log-transformed size distribution
- intercept: this one includes the intercept of the linear regression of the log/log-transformed size distribution
- Number of count data points included into the regression for size distribution (in R: nreg_points): the number of data points, that remained after the data was cut according to the chosen method. Please note, that both intercept and slope are automatically set to NA, whenever this falls below three, since linear regression does not make sense with less than three data points.
- p-value (linear regression): This column contains the p-level referring to the linear regression on the log/log transformed data (slope and intercept).
- adjusted r^2 (linear regression) *and* r^2 (linear regression): These columns contain the respective r^2 values for the linear regression on the log/log transformed data (slope and intercept).
- minESD: The ESD of the smallest particle counted by imageJ within the stack
- maxESD: The ESD of the largest particle counted by imageJ on the stack (if some of these are higher, than the borders set in histvec, you might like to consider rerunning with different borders in some cases).

- The following columns are named with the classmids and contain the amount of particles counted and sized by imageJ within the stack, that are within the respective size class. The amount of size classes, as well as their size can be adjusted in by changing the histvec-settings. These values are not corrected for filtered volume or observed area. They contain the raw counts.
- In case of TEP you have carboncontent and carboncontent per Liter: These columns follow the standards of the above. They contain the carboncontent of the TEP particles (that were within the range of histvec) calculated according to Mari (1999).

Chapter 2

General Setup - How are the results obtained?

This part is to allow a simpler access to the script and is meant to facilitate amendments and changes that may become necessary, as well as the backgrounds of calculations performed and the effects of specific variables. In case you make changes or amendments to the script, which are meant to stay, please include them here, so future users can understand what you have done more easily.

2.1 The in-script interface

This part is supposed to contain all the variables that may need to be adjusted for use and those information, that you may want to find easily. The first block with comments is to offer an overview on past changes as well as the information on what is needed to run the script. After those comments you find the more stable variables (e.g. method_sizedis, hist_vec and magnification, which usually don't need to be adjusted, but you should still check whether they are correct and you may want to adjust them easily). Lines 32 to 50 do require to be checked and adjusted every time you run a different analysis. After this block, the script sets itself up and then goes over to analysis (line 61), where it calculates and obtains the results.

2.1.1 Variables to look out for (lines 25-30)

The **magnification** greatly influences the result because it prescribes which area of the filter you have seen. In most cases pictures are taken at 200x magnification, which means you don't have to adjust it. However, if for some reason you have used a different magnification, please remember to correct accordingly. **hist_vec** is more of interest in most analysis. If you have high particle counts over a smaller range,

you may optimize your size distribution decrease the step of size-classes, or if your particles tend to be small, you can also use it to have smaller output tables and save calculation times (if you have a lot of filters). Further, when having large particles, many of them might be excluded automatically, so you would have to adjust here. In this case you may also decrease step size to obtain higher counts in some larger size classes. Please keep in mind, that this effects the results you obtain for the size distribution and carbon-content for TEP, so you want to choose this carefully. The last variable, you may consider here, is **method_sizedis** which determines how the size distribution is calculated. The size distribution is based on Mari and Kiørboe (1996). The default is AG_Engel_Standard because this method is very robust. However, in case you want to consider different approaches, the next part briefly explains how the calculations are performed for all methods.

method_sizedis - Your choice

1. **Excel-Mastersheet:** First all Particles with an ESD larger than 33 μm are excluded. After that all zero counts are set to one and the data are log/log transformed. Last the linear regression takes place. Please note: in contrast to the other methods the upper borders of the size classes are used for this one. The mid-value of the size-class is chosen for all other methods.

2. **Mari& Kiorboe(1996):** The included size classes are in the range of $3\mu m < x < 40\mu m$. Before the log/log transformation, a constant value of 0.001 is added to the counts obtained for each class. After the log/log transformation, values, that were obtained by points that were zero initially, are set back to zero and the regression takes place.

3. **zerotail_omit:** The data are being cut after the third size class in a row, in which zero particles were counted. Again, 0.001 is added to the counts of all size-classes and log/log transformation takes place. Here as well zero counts are transferred back to zero after the transformation, and regression takes place.

4. **AG_Engel_Standard:** In this method any size-class with less than 10 counts is rejected because of bad counting statistics. The remaining data points are log/log transformed. If less than three size classes are left, no regression is performed and NA reported back. This method is the default.

2.2 The Analysis part

2.2.1 Internal preparations and reading the filters

Lines 51–71 are some preparations for the script run based on the input. First, the size of the respective images is requested (from the information stored in the function magniffactor). Next the filter-vector for the loop is created and formatting for the treatment and filename column is ensured. Lines 60-70 define the results-dataframe, all columns that are not in the masterhist dataframe, have to be defined here, before they can be used elsewhere. This dataframe is filled during the script run, by adding the information in the line with the respective filter_ID. After these preparations are done, the script jumps to the directory, where ImageJ results are stored and loops over the filters given in the filter_ID-column of the input table.

In the next part, the script loops over the filters given in the input-table. First it checks whether there is a filename attributed to the filter. If that is not the case, the script sets all variables that need to be taken from the file or calculated from histogram data zero or NA respectively. If there is a filename for the filter, the script will recognize the format based on the ending and read the file (lines 110–125).

2.2.2 Analysis of non-empty filters

In the part from lines 126–134 it takes the output from the analysis, calculating ESD from area based on the formula

$$ESD = \sqrt{\frac{Area}{\pi}} \times 2$$

and calculate ESV based on ESD by

$$ESV = \frac{4}{3} \times \pi \times (\frac{ESD}{2})^3.$$

Further it denotes the total number of particles and the complete area of particles observed in the image analysis. Additionally, the ESD of the smallest and largest particle are saved. In the following lines (136–163) the histogram is created and the size distribution calculated for non-empty filters. This also includes the coresponding p-level and r^2. For the next part (lines 165–172) the numbers per liter are calculated (for all filters) based on the formula:

$$\frac{\frac{\pi}{4} * diameter^2_{of filter in \mu m} * value_{observed}}{Pictures_{included in the analysis for the respective filter} * Area_{of each picture in \mu m^2} * Volume_{of sample filtered in l}}.$$

If TEP-Particles are analysed (Particletype="TEP"), the Carboncontent of TEP-Particles is calculated (lines 180–190) according to Mari (1999), using the formula:

$$\sum_{k=1}^{n} 0.25 * 10^{-6} * r_k^{2.55} * counts_k,$$

where k the respective size class, r the representative radius (middle of the size class) in μm^2 and counts the amount of particles counted for that size class. For this step the histogram data are used.

2.3 The end of the script

After all the calculations have been done, the script needs to do some more steps before finishing. These steps are generally made for the convenience of the user and therefore not necessary for the analysis. Straight after finishing the calculations (lines 191–200) the result table and the histogram data are combined, to allow for just one output table to hold all the information. In the next step (lines 205–211) the output table is saved if this was requested by the user. Furthermore the script jumps back to its initial working directory, in order to restore its initial conditions and be ready to be run with the next set of filters. Last (lines 215–230) the script plots the results in a simplified way, in case this was requested.

Chapter 3

Appendix - code details

3.1 List of functions and what they do

- indan (short for individual analysis): input arguments: TEP-data (an ImageJ output-file that has been read into R), return: TEP-data (with extra columns ESD and ESV), calculations/actions performed: calculate ESD from area based on the formula $\sqrt{\frac{Area}{\pi}} \times 2$ and calculate ESV based on ESD by $\frac{4}{3} \times \pi \times (\frac{ESD}{2})^3$
- obsarea (short for observed area): Input arguments: microscope (given by user at the beginning of the script), magnification (given by user at the beginning of the script), return: area of pictures in μm^2, calculations/actions performed: translate combination of microscope and magnification into picture size (by given values)
- size_distribution: input arguments: histdata (output from hist(data)), return: slope and intercept of the log/log transformed size distribution, calculations/actions performed: log/log transformation of the histogram data, linear regression of the transformed data
- addto_masterhist: input arguments: filter_ID, hists (output from hist(data)), masterhist (table where previous histogram counts are combined), isEmpty (information whether the respective filter is empty), return: masterhist (to which the Extra-filter is added), calculations/actions performed: extract the values from the hist-output and adding it to the masterhist table
- calc_Ccontent: input arguments: masterhist (table where all the histogram counts are combined), return: masterhist (to which the row carboncontent is

now added), calculations/actions performed: take the counts and mids for each size class and filter, and calculate the carboncontent according to Mari (1999) for each filter, add this information to masterhist

- save_xls: input arguments: results (table where all the results are combined), filename (to-be), decimal_separator (”.” or ”,”), return: none, calculations/actions performed: results are written into a tab-separated file with the specified filename, the decimal separator specified will be used.

Bibliography

Xavier Mari. Carbon content and C:N ratio of transparent exopolymeric particles (TEP) produced by bubbling exudates of diatoms. *Marine Ecology Progress Series*, 183:59–71, 1999. doi: 10.3354/meps183059.

Xavier Mari and Thomas Kiørboe. Abundance, size distribution and bacterial colonization of transparent exopolymeric particles (TEP) during spring in the Kattegat. *Journal of Plankton Research*, 18(6):969–986, 1996. doi: 10.1093/plankt/18.6.969.