Electronic Supplementary Material (ESI) for Analytical Methods.

Electronic Supplementary Information for:

Analysis of dissolved organic matter fluorescence using self-organizing maps: mini-review and tutorial

C.W. Cuss[1#], C. Guéguen[2,*]

[1] Environmental and Life Sciences graduate program, Trent University, ON, Canada; email: chadcuss@trentu.ca

[2] Chemistry Department, Trent University, ON, Canada; email: celinegueguen@trentu.ca

[#]Present Address: University of Alberta, Department of Renewable Resources, Edmonton, AB, Canada. Email:cuss@ualberta.ca

[*]Corresponding author; Email celinegueguen@trentu.ca

Contents

Theoretical overview of self-organizing maps

Self-organization of SOM proceeds by iteratively adjusting the map vectors $\boldsymbol{m_i}$ to make them more similar to the sample vectors $\boldsymbol{x_j}$. There are two major steps in each iteration. First, an $\boldsymbol{x_j}$ is randomly chosen from the sample set and a distance metric is used to find the most similar $\boldsymbol{m_i}$, which is subsequently redefined as the closest map vector, $\boldsymbol{m_c}$. In other words, $\boldsymbol{m_c}$ is defined according to the relationship $\|x - m_c\| = min\{\|x - m_i\|\}$. Herein, $\|\cdot\|$ is the Euclidean distance, though other distance metrics can also be used. Next, $\boldsymbol{m_c}$ and proximate $\boldsymbol{m_i}$ (i.e. the map units in the neighbourhood of $\boldsymbol{m_c}$ defined by $\boldsymbol{N_c}$) are made more similar to $\boldsymbol{x_j}$ by an amount that is proportional to the difference between $\boldsymbol{m_c}$ and $\boldsymbol{x_j}$, and to the learning rate $\boldsymbol{\alpha}$. Hence, on iteration $t+1$, $m_i(t+1) = m_i(t) + \alpha(t) \cdot N_c(t) \cdot [x(t) - m_i(t)]$. Here, $\boldsymbol{\alpha}$ and $\boldsymbol{N_c}$ decrease as the number of iterations increases, so that changes to the map become progressively smaller and more localized. These two steps above are repeated until the map converges to a stable configuration. Hence, the distribution of variables on the map and the corresponding relationships between samples asymptotically approaches those of the data set.

Fluor_SOmap package tutorial

This tutorial serves as a guide for researchers wishing to implement Kohonen's self-organizing maps (SOmap) on fluorescence data using Matlab©. The following instructions and commands facilitate the analysis of fluorescence excitation-emission matrix (EEM) data in four modes: variation between the loadings of components described using parallel factor analysis (PARAFAC Fmax values), between percentages of total fluorescence, between full EEMs, and between individual excitation-emission (Ex/Em) wavelength pairs. Using Fmax values or percentages of total fluorescence requires that EEMs are first processed using the DOMfluor toolbox[1] or the drEEM toolbox[2]; however, the algorithms in the Fluor_SOmap package do not rely upon the commands in these toolboxes. EEMs may also be uploaded directly without applying these pre-processing steps (see below), but users will have to preprocess the EEMs manually prior to using Fluor_SOmap (e.g. remove scatter).

Commands are built upon the SOM toolbox version 2.0 [3], available through the following link:

http://www.cis.hut.fi/projects/somtoolbox/download/

The Fluor_SOmap package may be downloaded through the following link:

http://people.trentu.ca/~celinegueguen/Fluor_SOmap.html

The SOM and Fluor_SOmap toolboxes must be installed prior to using the tutorial.

Commands that should be typed or copied into the command window during the tutorial are shown in **bold blue font**. The SOM toolbox and Fluor_SOmap package should be set as working paths in Matlab as follows.

Setting up Fluor_SOmap and checking for correct installation:

1. Create a folder on your computer and name it *Fluor_SOmap*. Extract the content of the files "somtoolbox2_Mar_17_2005.zip" from the website above, and "Fluor_SOmap.zip" into the folder.
2. Start Matlab and "Set Path..." to the Fluor_SOmap folder.
3. To check that Fluor_SOmap was correctly installed, type **Fluor_SOmap** into the Matlab command window and hit Enter. The messages "Fluor_SOmap Toolbox correctly installed." and "SOM Toolbox correctly installed." confirm correct installation.

1) Loading and preparing data

The procedures for data analysis and visualization discussed herein are synchronized with section three of the main text and associated figures to allow confirmation of correct execution. For analyzing either PARAFAC output or raw EEMs, sample names should be placed in the first column of a separate Excel spreadsheet (entitled 'Sample_ids.xls' in this tutorial). For the Fmax data only, the component names (e.g. "Component 1") should be placed in the first column of another separate spreadsheet (entitled 'Component_ids.xls' in this tutorial). These files should be placed in the Fluor_SOmap folder.

To load the sample names from the Excel spreadsheet entitled 'Sample_ids.xls', type:

[num, Sample_names]=xlsread('Sample_ids.xls');

To load the component names from the Excel spreadsheet entitled 'Component_ids.xls', use:

[num, Component_names]=xlsread('Component_ids.xls');

NOTE: In the above and all subsequent commands that load data from an Excel (1997-2003) spreadsheet, you may load and analyze your data in place of the tutorial by replacing the name of the Excel spreadsheet from the tutorial with the name of the spreadsheet containing your data, formatted as outlined in the text and placed in the Fluor_SOmap folder.

a) Loading and preparing data as Fmax values and fluorescence composition

To apply SOM to component loadings produced via PARAFAC analysis (i.e. 'Fmax' values), the values should be placed in a Microsoft Excel (1997-2003) spreadsheet with each sample in a row and variables in columns, and placed in the Fluor_SOmap folder. Only the numerical 'Fmax' values for the PARAFAC components should be present in the worksheet (i.e. remove sample names, component names, etc.). Fmax values will be used for applying SOM to the raw loadings, and to the percentage contribution of each component to total fluorescence.

To load the Fmax values for the tutorial, enter:

```
Fmax_data=xlsread('SOM_fmax_data.xls');
```

To convert the Fmax values to percentage of total fluorescence to analyze fluorescence composition, enter:

```
Fmax_to_percent;
```

To create the Fmax structure 'SOM_Fmax_struct' from the Fmax PARAFAC data for use in the SOM algorithm, and then scale it to unit variance, enter:

```
SOM_Fmax_struct = som_data_struct(Fmax_data, 'labels', Sample_names, 'comp_names', Component_names);
```

SOM_Fmax_struct = som_normalize(SOM_Fmax_struct, 'var');

To likewise create the **fluorescence composition** structure 'SOM_percent_struct' from the Fmax PARAFAC data for use in the SOM, enter:

SOM_percent_struct = som_data_struct(SOM_percent_data, 'labels', Sample_names, 'comp_names', Component_names);

SOM_percent_struct = som_normalize(SOM_percent_struct, 'var');

b) Loading and preparing data as full EEMs and Ex/Em pairs

For applying SOM to raw EEMs, data should be loaded into the Matlab workspace as a variable with the scatter and outliers removed (e.g. 'Test3' from DOMfluor step 11 or 'Xin' from drEEM step 4.4). For the purpose of this tutorial, the full EEM data has been placed in the 'Test3' variable. To load the tutorial version of 'Test3' into the workspace so that it is accessible to Matlab commands, type:

load 'Fluor_SOmap starting workspace'

You may also upload raw EEMs individually from Excel spreadsheets using the **'Read_in_EEMs'** command. Execute **help Read_in_EEMs** in the command window for complete details.

EEM data will be used for applying SOM to full EEMs, and to the individual Ex/Em pairs. To prepare the tutorial EEM data, type:

EEM_data=Test3;

NOTE: To load your own data that has been pre-processed for PARAFAC instead of the tutorial data, load the Matlab workspace that contains the pre-processed data and replace

'Test3' above with the name of the Matlab variable that contains your pre-processed data. Following the PARAFAC tutorials (see references 1 and 2), this variable will already be named 'Test3' or 'Xin', respectively, so that you may wish to use the same names described above.

EEMs must be further processed by converting them into vectors and removing negatives and non-numbers (i.e. 'NaNs') that may have been introduced by the PARAFAC algorithm. Ex/Em pairs that have a value of zero for all samples must also be removed. These functions can be achieved using the 'EEM_SOMprep' command, which is executed on 'EEM_data' by entering:

EEM_SOMprep;

The message "EEMs successfully prepared for inclusion in SOM data structure." should confirmed the creation of the variable 'SOM_EEMinput', which consists of the EEMs transformed into vectors in preparation for SOM construction. The variable 'ExEm_pairs' is also generated for labelling the variables.

It may also be useful to standardize each EEM on the interval [0, 1] by dividing by the maximum fluorescence to eliminate the influence of concentration differences. While such standardization may amplify the noise level in EEMs with low fluorescence, SOM are noise tolerant[4,5]. If concentration differences are an important consideration in subsequent analyses, then the EEMs should not be standardized. To standardize the EEMs, enter:

Stdize_EEMs;

To create the data structure 'SOM_EEM_struct' from the EEM fluorescence data for use in the SOM, and scale it to unit variance, execute:

SOM_EEM_struct = som_data_struct(SOM_EEMinput, 'labels', Sample_names, 'comp_names', ExEm_pairs);

SOM_EEM_struct = som_normalize(SOM_EEM_struct, 'var');

To reformat the Ex/Em pair labels for clustering output, create the data structure 'SOM_ExEm_struct' from the EEM fluorescence data, and normalize it to unit variance, enter:

Reformat_ExEm;

SOM_ExEm_struct = som_data_struct((SOM_EEMinput)', 'labels', ExEm_pairs2, 'comp_names', Sample_names);

SOM_ExEm_struct = som_normalize(SOM_ExEm_struct, 'var');

The fluorescence data has now been prepared in four modes that may each be used to explore patterns of variance using SOM:

1. Fmax values from PARAFAC (**SOM_Fmax_struct**);
2. Fmax values transformed into fluorescence composition (**SOM_percent_struct**);
3. whole EEMs (**SOM_EEM_struct**); and
4. Ex/Em wavelength pairs (**SOM_ExEm_struct**).

2) Creating multiple SOM and selecting the optimal map

We will find a good approximation to the optimal fit by first setting map dimensions relative to the variation explained by the first two principal components. We will also fit a

number of maps from randomized starting conditions and select the best map based on two criteria: mean quantization error (***mqe***) and topographical error (***tge***). Readers interested in manually adjusting other parameters such as map size, number of iterations, and topography should investigate the 'som_make' function in the SOM toolbox.

Using the '**multi_fit_SOM**' function below, all maps will be stored in the data structure 'SOM_map', based on user-defined slot *i*. Slots SOM_map(*i*) to SOM_map(*i*+*k*) will be used to store the results from linear initialization, where *k* is the number of linearly-initialized maps that are chosen. Slots SOM_map(*i*+*k*+*1*) to SOM_map(*i*+*k*+*l*) will be used to store the results from random initialization, where *l* is the number of randomly-initialized maps that are chosen. *Importantly, the map size for the randomly initialized SOM is based on the map dimensions from the linearly-initialized SOM that was created using the same data structure. Therefore, for each data set and structure a linearly-initialized SOM must be generated prior to the creation of a randomly-initialized SOM. You will be prompted to enter the slot number of the linearly-initialized map corresponding to the appropriate data type prior to the generation of a randomly-initialized map when they are not being created in the same run.*

Depending upon the number of samples and variables in the data structure, the fitting of SOM can take several minutes so that fitting several maps may take several hours. When analyzing data with a large number of variables (e.g. full EEMs), it may therefore be preferable to fit a small number of randomly-initialized maps (< 5) first rather than a large number, and then increase the number depending upon the amount of variation in *mqe* and *tge*. When '**multi_fit_SOM**' is initiated, you will be asked how many maps have already been created (i.e. how many exist in the 'SOM_map' variable) to avoid overwriting existing maps. Once a set of

maps has been generated, the *mqe* and *tge* of all maps stored in the 'SOM_map' variable will be displayed for use in selecting the map with the best fit. Enter:

multi_fit_SOM;

You will be prompted to enter information about existing maps/maps to be generated. Answer the questions as follows:

How many maps have already been created? 0

How many linearly-initialized maps would you like to create? 1

How many randomly-initialized maps would you like to create? 10

How many rough tuning iterations would you like to conduct? 100

How many fine tuning iterations would you like to conduct? 5000

Which data structure would you like to use (enter: SOM_Fmax_struct, SOM_percent_struct, SOM_EEM_struct, or SOM_ExEm_struct)? SOM_Fmax_struct

Matlab will display the current running time and the estimated total running time throughout the generation of each map. This step will take several minutes to complete. The final output table should be similar to Table S1, but the *mqe* and *tge* of the randomly-initialized maps will differ slightly.

Table S1: Fitting parameters for linearly- (**1**) and randomly-(**2-11**) initialized maps of the Fmax data.

Map	*mqe*	*tge*
1	0.6035	0.0741
2	0.5397	0.0864
3	0.6095	0.0123
4	0.6070	0.0123
5	0.5577	0.0370
6	0.5519	0.0988
7	0.5465	0.0123
8	0.5479	0.0123
9	0.5788	0.0247
10	0.5547	0.0617
11	0.5169	0.0123

The map with the lowest values of *mqe* and *tge* is the best choice for subsequent analysis (i.e. Map 11 in Table S1). If one map has the lowest *mqe* while a different map has the lowest *tge*, then the choice of the optimal map depends on whether the similarity of model vectors to sample vectors (lowest *mqe*), or the grouping of the samples according to their similarity to neighbours (lowest *tge*) is more important.

For now, we will proceed with the analysis of the Fmax data and generate SOM for the other data formats when they are needed. When maps are correctly generated using the other data modes, the *mqe* and *tge* of the linearly-initialized maps are shown in Table S2.

Table S2: *mqe* and *tge* for linearly-initialized maps created using different data structures.

Linearly-initialized maps	mqe	tge
Percent (i.e. SOM_percent_struct)	0.9332	0.0267
Full EEM (i.e. SOM_EEM_struct)	40.3342	0.000
Ex/EM pair (i.e. SOM_ExEm_struct)	2.8552	0.0625

3) Numerical and graphical data output

The ability to visualize organized data and the distribution of variables on a two-dimensional surface (i.e. the map) is a major benefit of SOM. Accordingly, there are several ways in which the organized data may be displayed.

a) Unified distance matrix (U-matrix) and BMU map

To display the U-matrix and BMU map for the linearly initialized map of the Fmax data (Figure S1), type:

Show_SOM_map; and answer the questions as follows:

What map would you like to label (i.e. number in the SOM_map data structure)? 1

What size of text would you like to use (8 recommended)? 8

Do you want to plot the BMU map (1 for yes, 0 for no)? 1

To display the first map in the SOM_map variable (i.e. the lineraly-initialized map), type **1.** This step should be repeated for all 11 maps to ensure that the general pattern of organization is preserved throughout (i.e. clusters by sample type with some differences based on molecular weight, and the same sample types isolated and mixed).

Also apply **multi_fit_som** to the percent data (i.e. 'SOM_percent_struct') to create one linearly-initialized map, noting that 11 maps have already been created and stored in the 'SOM_map' data structure. Use **Show_SOM_map** to visualize the resulting map and U-matrix, which reveals that concentration interfered with quality in the map of Fmax values, since the relative position of the samples has changed so that SRM now differs the most from the other samples (Figure 3).

b) **Hit histogram and BMU data output**

To show only the number of samples associated with each map unit (i.e. the 'hit histogram'), enter:

Hit_histogram;

for SOM_map(1), and compare the result with the BMU map from step **a)** (Figures S1, S2). If desired, this command will also export the sample names that were loaded in step 1 to a user-defined Excel file, with each sample name listed in the row that corresponds to its most similar map unit (BMU). Map units are numbered from top to bottom, and left to right. Hence, map unit 1 and row 1 of the Excel file are associated with samples WET-3 to WET-7, unit 8 and row 8

ares associated with STL-3 to STL-5, and unit 40 and row 40 are associated with SSM-3 to SSM-5 (Figure S1). Up to six groups of samples may also be plotted in distinct colours using:

Hit_histogram_groups;

Answer the questions as follows to plot '3' groups, the first with five samples (sample numbers 37, 38, 39, 40, 41 for samples STL-2 to STL-6), the second with six samples, (46 to 51; SSM-2 to -7) and the third group with six (56 to 61; WET-3 to -8).

What is the number in the SOM_map variable corresponding to the map that you would like to label? 1

How many groups of samples do you wish to plot (1-6)? 3

Group 1:

How many samples do you wish to include in this group? 5

Enter sample number corresponding to order in Sample_ids.xls 37

Enter sample number corresponding to order in Sample_ids.xls 38

….

Enter sample number corresponding to order in Sample_ids.xls 41

Group 2:

How many samples do you wish to include in this group? 6

…and so on. Within each group, the size of the coloured portion of each map square is proportional to the number of samples. Compare the result with the BMU map generated in step **a)**.

c) **Codebook vectors**

Codebook vectors are the values of the variables in the vector underlying each map unit. Thus, for the 'SOM_Fmax_struct' data, the codebook vector for each map unit contains the Fmax values for each PARAFAC component. Plot the codebook vector for map unit number one on the

linearly-initialized SOM of Fmax values (i.e. 'SOM_map(1)') using the command and answering questions as follows:

Plot_codebook_vector;

Which map would you like to plot (i.e. number corresponding to map in "SOM_map" data structure?) 1

What is the number of the map unit that you would like to plot? 1

Plot more codebook vectors (enter 0 for no, or map unit number to plot)?

Also plot codebook vectors for the map units in the other corners of the same map (e.g. units **8**, **34**, and **40**) to see that units 8 (STL samples), 34 (high molecular weight samples) and 40 (SSM samples) are respectively dominated by C1, C2, and C3 (Figure S3).

Do you want to export all codebook vectors to file (1 for yes, 0 for no)?

Entering **1** to export the codebook vectors to an Excel file will list the vectors for each map unit in the row corresponding to its map unit number, with the corresponding values of the variables from 'Component_ids.xls' increasing from left to right along the columns.

d) **Component planes**

Component planes show the distribution of the variables across the map corresponding to the samples shown on the BMU map. To display the distribution of C1-C7 on the linearly initialized SOM of percent values, use:

som_show(SOM_map(12), 'comp', [1:7]);

Compare **som_show(SOM_map(13), 'comp', [1955, 3822, 4879, 6729])** to display the distributions of excitation/emission wavelength pairs 240/450, 275/350, 305/400, and 350/400 from the SOM of the raw EEM data. If the commands have been correctly executed up to this

point, the component planes for the percent and raw EEM data generated above should be identical to Figures 5A and 5B .

4) k-means clustering of SOM

The **'Cluster_SOM'** function allows the user to choose the range of clusterings to consider, the number of clusterings to conduct, and the final number of clusters. The Davies-Bouldin Index (DBI) is calculated and displayed graphically to aid in choosing the number of clusters. For each number of clusters, only the optimal (i.e. lowest error) solution from all clusterings is retained. Cluster membership is output in the Matlab command window as a vector of numbers that increases with map unit for ease of comparison with other output (e.g. the 'Samples by BMU' Excel output file). Apply clustering to the linearly-initialized SOM of the percent data (stored in 'SOM_map(12)') for up to ten clusters with 100 clusterings each using the following function, and answering the associated questions:

Cluster_SOM;

Which SOM would you like to plot (i.e. number corresponding to map in "SOM_map" variable?) 12

Up to how many clusters would you like to consider? 10

How many clusterings would you like to perform for each number of clusters (> 10 recommended)? 100

ans =

7 clusters are recommended according the minimum DBI

(Between four and eight clusters will likely be recommended by the DBI plot.)

How many clusters would you like to choose?

Choose to view the solutions for several numbers of clusters and compare the cluster borders with the respective U-matrix and BMU map to see that the sample grouping is best described using seven clusters (Figures 3, 6).

Next, we will create a new map using the 'SOM_ExEm_struct' data, label it, and assess the number of clusters. First, execute:

multi_fit_SOM;

and answer the questions (see section 2 for reference) to acknowledge that 13 maps have already been generated and stored (which can also be seen by considering that the 'Value' for the 'SOM_map' variable listed in the Matlab workspace window is '1x13 struct.'). Enter '1' to create a single linearly-initialized map, and '0' to create no randomly-initialized maps. Use:

Show_SOM_map;

to assign each Ex/Em pair to its respective BMU, but do not plot the BMU map because the large number of Ex/Em pairs will make it illegible. Instead, use:

Hit_Histogram;

to view the clustering of Ex/Em pairs on the map. Then, perform k-means clustering with up to ten clusters and 100 clusterings each by applying

Cluster_SOM.

View the solutions containing between five and seven clusters and compare them with the hit histogram and U-matrix to see that the optimal solution contains between five and seven clusters.

Next, we will project the clusters of Ex/Em wavelength pairs onto a contour plot of the EEM, allowing visualization of the correlated fluorescence regions that best describe variation through the data set; the resulting components are similar to PARAFAC components, but without deconvolution of overlapping spectra. Complete the visualization by executing the following command and answering the associated questions:

Visualize_ExEm_clusters;

How many clusters of Ex/Em pairs would you like to consider? 5

Also view the six- and seven-cluster solutions to see how the SOM partitions the EEM to explain greater amounts of variation (Figure 7):

Consider other numbers of clusters (enter 0 for no, or number of clusters to view)? 6

Consider other numbers of clusters (enter 0 for no, or number of clusters to view)? 7

Consider other numbers of clusters (enter 0 for no, or number of clusters to view)? 0

Do you want to export the cluster information to file (1 for yes, 0 for no)?

When the cluster information is exported, two different Excel worksheets are created: the first contains the cluster number associated with each Ex/Em pair, each in a separate column, while the second sheet contains the EEM of cluster identities used to produce the plot. **'Cluster_SOM'** should always be executed immediately prior to executing **'Visualize_ExEm_clusters'**, because the latter uses the clustering values from the 'SOM_map' that were created by **'Cluster_SOM'**.

This completes the basic tutorial for analyzing fluorescence using self-organizing maps. More specialized tools are discussed below in the supplementary commands section.

5) Supplementary commands

a) Codebook EEMs and vectors for the full EEM data

Prior to plotting codebook vectors for full EEM data, it is helpful to reshape the codebook vectors into EEMs. Create a single linearly-initialized SOM for the full EEM data now by applying **multi_fit_SOM** to '**SOM_EEM_struct**', noting that 'SOM_map' already contains 12 maps. The time required to generate this map may be on the order of hours, since there are thousands of variables (Ex/Em pairs) in each sample vector. Once the map has been generated, use **Show_SOM_map** (selecting map 13) to label the map. Next, execute:

Plot_codebook_EEM;

for map '13' and map unit '1' to display the EEM most similar to the 12 samples of high molecular weight (Figures 4A). Next, plot the codebook EEMs for units 11 (samples OSM-3 to OSM-8) and 35 (SSM-2 to SSM-6) of the same map to compare. Enter '0' twice to stop plotting codebook EEMs and decline the offer to export them out to an Excel spreadsheet. Next, apply **Plot_codebook_vector** to the same three map units to assess the fluorescence loadings on each Ex/Em pair. Compare the intensity of the signal spikes at high component values (noise) relative to the rounded peak-like features in the middle components for each map unit (Figure 4B). These alternating views are useful for visualizing the spectral features of samples by plotting the codebook EEMs and vectors of the associated best-matching unit (BMU) on the map.

When the codebook EEMs are output to a user-defined Excel filename, each EEM is placed in the worksheet number that corresponds to its map unit. Emission wavelengths increase along rows, and excitation wavelengths increase along columns, both corresponding to the values in the original 'Test3' or 'Xin' variable from PARAFAC, or the Excel files that contained the Ex and Em ranges if EEMs were entered manually.

b) Comparing fluorescence described by clusters

To compare the levels of fluorescence described by the clusters that were visualized using **'Visualize_ExEm_clusters'** across samples, use:

Compare_clusters;

Apply this algorithm to the 6-cluster solution. The non-standardized fluorescence stored in 'EEM_data' will be used to calculate the fluorescence associated with each cluster in every sample. NaNs and negative values are removed prior to summation, and the area-standardized fluorescence of the clusters is provided in 3 formats: 1) 'raw', 2) standardized on [0,1], and 3)

percentage of total fluorescence (not area-standardized). If the cluster fluorescence is output to Excel for further visualization and analysis, the cluster number increases along the columns in the worksheet, and the row ordering corresponds to the order of sample names provided in 'Sample_ids.xls'. Raw fluorescence is exported to the first worksheet, standardized values are exported to the second sheet, and percentages are exported to the third sheet.

The 'clustfluor' Matlab variable that is created by **'Compare_clusters'** can also be used to visualize differences between samples and changes in samples with increasing molecular weight within Matlab. For example, to visualize the **1** (raw) and **3** (percentage of total) fluorescence for sample **48** of 'Sample_ids.xls' (i.e. SSM-3), type in

subplot(1,2,1);bar(squeeze(clustfluor(1,48,:)));title('Raw');

subplot(1,2,2);bar(squeeze(clustfluor(3,48,:)));title('Percent of total'); (Figure S4)

To display the change in **1** (raw), **2** (standardized), and **3** (percentage) of total fluorescence in cluster **2** for increasing molecular weight fractions of sample SSM (i.e. rows **46** to **54** in 'Sample_ids.xls'), type in:

subplot(1,3,1);bar(squeeze(clustfluor(1,46:54,2)));title('Raw');

subplot(1,3,2);bar(squeeze(clustfluor(2,46:54,2)));title('Standardized');

subplot(1,3,3);bar(squeeze(clustfluor(3,46:54,2)));title('Percent of total'); (Figure S5)

Works cited

1 C. A. Stedmon and R. Bro, *Limnol. Oceanogr.: Methods,* 2008, 6, 572-579.

2 K. R. Murphy, C. A. Stedmon, D. Graeber and R. Bro, *Anal. Methods*, 2013, doi: 10.1039/c3ay41160e.

3 E. Alhoniemi, J. Himberg, J. Parhankangas and J. Vesanto, *SOM Toolbox 2.0*, 2005, Laboratory of Computer and Information Science, Helsinki University of Technology, Finland.

4 T. Kohonen, *Self-organizing maps (3e)*. Springer-Verlag, Berlin, 2001, 502 p.

5 A. Astel, S. Tsakovski, P. Barbieri and V. Simenov, *Wat. Res.,* 2007, 41, 4566-4578.