**Summary of the included MATLAB scripts**

Please note: The scripts have been written with MATLAB version r2017b and require the Curve Fitting Toolbox, the Bioinformatics Toolbox and the Statistics and Machine Learning toolbox.

* **readData.m**

A script used for reading the data from spreadsheet format to MATLAB. The script cycles through all xlsx-files in the current folder and creates a folder named correspondingly to each spreadsheet, where it moves the original data and the mat-file created by the script. The mat-file contains the *allSheets* struct, where all the original intensity data is stored and the *cellData* array containing the *ca\_response* objects corresponding each cell.

* **ca\_response.m**

A class definition file for the *ca\_response* objects. Contains the parameter definitions for the class and the **ca\_analysis** method function.

* + **ca\_analysis**

A method function, which calculates the bleach correction and the parameters used in clustering for each *ca\_response* object and adds them as their object parameters.

* **findcrossings.m**

A function to determine the indices of the intensity data vector, where the intensity crosses a certain threshold (by default 50%). Called by **ca\_analysis** in **ca\_response.m**. Inputs are the data vector and the threshold value; output is a vector with all the indices that satisfy the crossing condition.

* **run\_analysis.m**

A looping function to loop through all the cells and ROIs in a given *cellData* array. The inputs are the ROI and cell numbers, from which the analysis is started by the user. For example, **run\_analysis(2,13)** starts the analysis from ROI 2 and cell 13. The function calls the function **ca\_analysis** in **ca\_response.m**. The data is saved between each cell to the mat-file in the current folder.

* **sortCurves\_hierarchical.m**

A script for performing the grouping based on the calculated parameters. It extracts the parameters from each *ca\_response* object one ROI at a time in the given *cellData* array and converts them to z-scores (centered to the mean and normalized by standard deviation). The script uses principal component analysis (pca) to find out parameters causing the most of the variance in the data and uses a built-in hierarchical clustering algorithm to find two to four (by default) clusters from the data. The number of the clusters is optimized by minimizing the silhouette values of each cluster. For more information, see MATLAB documentation for *pca, cluster* and *silhouette*. For each ROI, an analysis figure (containing information about the pca and clustering) as well as the grouping plots are saved in the current folder as .fig and .png-files.

* + - **findCenterCurves**

A function in the **sortCurves\_hierarchical.m** script, called by the grouping algorithm. Finds the response curve that corresponds to the current group averages the best. Takes as inputs the grouping index vector from the hierarchical clustering, the parameters of the current group used in the clustering and the number of the group. Output is a vector of indices for each curve, sorted in ascending order of “proximity” to the group averages.

**Basic workflow**

1. Data extraction

Use e.g. ImageJ to segment the image stacks as individual cells and copy the data to a spreadsheet (the Multi Measure plugin works for this). The time vector should be in the first column and the individual cell intensity data vectors in the following columns. You can put multiple ROIs from the same image stack to different sheets in the xlsx-file (they should be named accordingly and should not contain spaces or start with a number, as **readData.m** reads the sheet names). The xlsx-file name syntax is *<cellLine>\_<surface>\_<timepoint>\_<ATPstimulus>.xlsx*, for example

*11-013\_combo\_d63\_ATP1.xslx*. This order is important, as the info is read from the filename.

1. Reading the data to MATLAB

Copy the xlsx-files to be analyzed in the same folder (should not contain any other xlsx-files) and run **readData.m**. This creates a folder structure that has a folder for each spreadsheet file, each containing the original spreadsheet, and an m-file containing *allSheets* struct (the original data in MATLAB form) and the *cellData* array of the *ca\_response* objects for each individual cell.

1. Bleach correction and response analysis

Navigate into each folder created by the **readData**-script and run the **run\_analysis.m** function. Give the function as inputs the ROI and cell numbers, respectively, where the analysis should start. For example, **run\_analysis(1,1)** starts from the beginning of the ROI 1.

* + - **Bleach correction**

The fit area of the baseline used for bleach correction is plotted in red, the original curve in blue and the corrected curve in green. If the fit needs to be changed, type ‘**n**’ to the command line, after which new fit boundaries can be selected with the mouse (only x-coordinate of the crosshair is read). If the fit is ok, enter ‘**y**’ to the command line. You can also enter ‘**s**’ to skip bleach correction or ‘**d**’ to discard the whole curve from the analysis. It is convenient to use the response start point as the fit end point if possible, as the fit end point is the default value used as the response start point in the analysis phase.

* + - **Curve analysis**

After entering ‘**y**’ or ‘**s**’ in the bleach correction, the script calculates the values for the response start (the fit end point by default), 50% rise, maximum intensity and 50% decay points. If these need to be changed, type ‘**n**’ to the command line and choose new points with the mouse in the above-mentioned order. Choose new value with **left mouse button** (only x-coordinate of the crosshair is read) and keep the value the same with **right mouse button** (or any other input). The 50% decay point is plotted in yellow if the intensity stays above 50% after the maximum. This means that the *decay50* and *duration50* parameters are marked as NaN, as they cannot be calculated. This can also happen when the intensity rise is very rapid, and consecutive data points are not close to the calculated 50% intensity (plotted as the blue line), leaving the 50% intensity points far from each other in intensity. The tolerance for this can be adjusted in the script (lines 228 and 330 in file **ca\_response.m**). It is common that the response start and maximum intensity points need adjusting, because spikes in the data can throw off the algorithm from the apparent trend of the response curve. When the values are acceptable, enter ‘**y**’ to the command line and proceed to the next cell.

1. Grouping of the response curves

Once all the cells in a *cellData* array are analyzed, run the **sortCurves\_hierachical.m** script (while in the same folder as the mat-file containing the *cellData*). This creates an analytics plot and a grouping plot of each ROI, both of which are saved to the current folder. The curves calculated by **findCenterCurves** function are plotted in black. The analytics plot contains

* + - a pareto chart of the best principal components from pca,
    - three different scatter plots of the data in the coordinate space given by the pca with the clusters plotted in different colors,
    - a silhouette plot used in optimizing the number of the groups,
    - a heatmap describing the effect of the original parameters to the pca components,
    - a plot comparing the original components of each group and
    - dendrogram describing the linkages of the hierarchical clustering.