CellScanner

Let's predict what is
in your medium

Help File

Table of Contents

Introduction:

CellScanner is a user-friendly tool that analyses the content of flow cytometry data and predicts samples' composition based on known reference information. On the one hand, based on supervised machine learning methods, the program uses the input data to create a model able to predict the composition of a coculture (in vitro or silico). It can identify the most representative parameters of each species from monoculture information called reference files. The likely species in the community must be known to train the model. The larger the community's complexity to predict, the more likely the prediction accuracy will reduce since the species profile is more likely to be similar, particularly for bacterial species. The prediction's quality is directly dependent on the reference data quality. Two main functions, visible on the main window, allow the user to assess a (1) in-silico community prediction to help the user interpret the result of a (2) in-vitro community prediction.

On the other hand, the implemented unsupervised clustering allows the user to predict the composition by proposing a label to the predicted clusters compared with reference data. The reference data are mandatory for this function but only used to identify the cluster. Then the reference data given by the user can be more flexible and less representative of the species in the community.

CellScanner is usable by individuals with no informatics background and only from flow cytometry data. In addition, the GUI (graphical user interface) limits entries to a minimum and prevents a maximum of typing or selection errors.

CellScanner contains a database that adapts to the user's needs and is accessible from the different windows of the GUI. For example, the database already contains the class 'Species' composed of different records (i.e., bacterial species) and reference files (monoculture files link to a specific record). Similarly, the user can create new classes and records in the database via the GUI. Furthermore, a class does not have to contain species only. It can be a group of cells, a subspecies, or anything defined by an FC file as a monoculture. The database also saves the parameter chosen by the user to apply to the following uses.

I- Installation:

Windows

Download the installer from the https://github.com/Clem-Jos/CellScanner web page and run it. Choose the location of the program. The directory created in the chosen location contains:

- The executable file to run the program
- The database bd.DB
- The 'References' directory for storage of the reference monoculture information.

During the import of reference data, files are copied to the References directory; this directory should not be modified by the user other than from the database management function via the GUI.

- The 'Results' directory for storage of CellScanner's output data.
- The result for all analysis and prediction running are saved in the Result directory.
- The Help.pdf user manual of CellScanner

Linux and Windows (cmd)

Package installing:

- Install Python 3.7 if not already installed on the computer

- Check for pip3 or pip in the cmd; install it if not available

On the cmd, enter the following commands:

```
pip install pyqt5==5.14

pip install sklearn

pip install fcsparser

pip install matplotlib.pyplot
```

Download or clone the directory containing the scripts, database file, and Result directory References directory.

To launch CellScanner from the terminal:

```
cd ~path/to/CellScanner

python3 CellScanner.py
```

or

```
python CellScanner.py
```

Mac

Package installing:

- Install Python 3.7 if not already installed on the computer (you can use homebrew for that)
- Check for pip3 or pip in the cmd; install it if not available

On the terminal, enter the following commands:

```
pip3 install matplotlib==3.1.1

pip3 install PyQt5==5.15.2

pip3 install scikit-learn==0.23.2

pip3 install fcsparser==0.2.0
```

Download or clone the directory containing the scripts, database file, and Result directory References directory.

To launch CellScanner from the terminal:

```
cd ~path/to/CellScanner

python3 CellScanner.py
```

or

```
python CellScanner.py
```

II- The Program

The python-based program uses pyQt5 and scikit-learn packages for GUI and machine learning functions, respectively. The latter contains supervised classification functions (e.g. neural networks, random forests, and logistic regression) and unsupervised clustering algorithms (e.g. agglomerative clustering) used to predict FC data composition.

Both classification and clustering are composed of two main functions. The first functions cluster (***Clustering analysis***) or predict (***Tool analysis***) an in-silico community built in the tool from monoculture files. The main goal of this function is to assess the quality of prediction of both methods before applying them to unknown community composition data. The second function, ***Prediction*** and ***Clustering***, predict the composition of in-vitro communities without quality assessment due to the unknown composition of the communities.

Prediction and Tool analysis overview

Classification functions (***Tool Analysis*** and ***Prediction***) build a predictive model called a classifier trained to recognise certain records based on reference data. Then this newly built classifier is applied to an in-silico or in-vitro community file to perform a prediction (Figure 1).

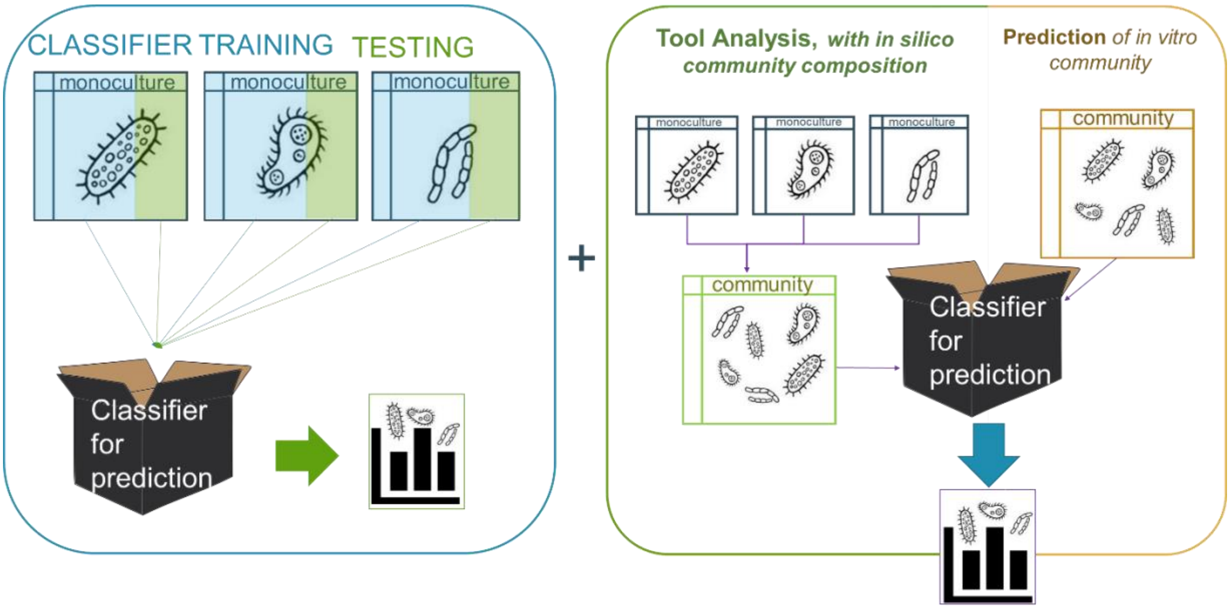

CLASSIFIER TRAINING
TESTING
monoculture
monoculture
monoculture
Classifier for prediction
+
Tool Analysis, with in silico community composition
Prediction of in vitro community
monoculture
monoculture
monoculture
community
community
Classifier for prediction

Figure 1: CellScanner function Prediction and Tool analysis, one run explained. The tool trains a classifier with monoculture to represent a record test and test it on another part of the same monoculture. Once created, the classifier can predict an in silico community built in the tool to assess prediction quality or an in vitro community of unknown composition.

The user gives one or several reference files per record with a specific label to identify them. For each record, a certain number of events (1000 per default) will be used to train the model (7/8) and to test the model (1/8). The user can modify the number of events selected per record and the ratio testing part/training part on the parameters. No overlapping exists at this stage between the training and the testing part. If the number of cells is lower than needed, the program will run with less information.

Once the classifier is created, it can perform a prediction on a community file. The ***Tool analysis*** function merges monoculture files provided by the user with their labels to create an in-silico community. The user can also specify the number of events to select per record in 'Tool analysis: number of cells per record'. Finally, the tool labels all events in the in-silico community and compares them to the initial labels. This comparison generates prediction accuracy and other statistical information that inform the prediction feasibility and the expected prediction quality with the specified community.

For the prediction, the classifier predicts the composition of in-vitro community files. The program does not merge data but can perform several predictions consecutively using the same classifier. All events are initially labelled unknown; then, new labels are given by the classifier. From this function, no statistical information is produced at the prediction step other than the number of events predicted by species.

The creation of the classifier, together with a prediction of an in-vitro or in-silico community, is called a run. To improve the prediction quality, the user can perform several runs consecutively. By default, ten runs are completed. The classifiers are built with the same machine learning methods and reference files, but the training and test set sections are random and different for each. The overlapping of cells in training sets within ten runs is low enough to be meaningless. The classifiers being slightly different, the prediction will variate.

After the runs and according to the parameter selected by the user, either a quality assessment of the prediction (number of events per record, accuracy, precision) is returned to the user, or a majority vote between the run predictions creates a unique average prediction. All other statistical values are calculated from this unique average prediction. The user can set the unknown threshold if the average prediction parameter is selected. The threshold value represents a percentage of classifiers that need to vote the same to label an event as not unknown. If the percentage of classifiers that agree on the label is below the threshold, the event will be labelled 'unknown'. This parameter limits the wrong assignation of a label to an event (False positive FP).

All result files are saved in a directory named after the starting date and time of the calculation. This directory is located in the Result directory of CellScanner.

Clustering and Clustering Analysis overview

Clustering functions were created to predict the composition of an in silico or an in vitro community without optimal reference data. Nevertheless, the function inputs some reference information to associate a cluster to a record by comparing their position in 3D plots.

Input data are first normalised with a common logarithm, for which all negative values are removed from the calculation.

The functions used agglomerative clustering from Scikit-learn and required the number of clusters as input. First, the program clusters from 1 to n+1 groups, n being the number of species expected in the community. Then it finds the best cluster combination with the maximum number of clusters having the maximum distance between them. Once the clusters are identified, CellScanner assigns a label by comparing the distance between their centroids to the references' centroids.

The ***Clustering analysis*** function creates an in-silico community from merged monoculture reference files. The user can provide several reference files that are mixed and specify the number of events per record to select for the community. The function indicates the user's prediction accuracy for this community and shows if the cluster is easily distinguishable for the clustering algorithm.

The ***Clustering*** function takes in-vitro community files as input. Users can provide several in-vitro communities that will be analysed consecutively. No prediction accuracy is given at this stage since the composition is unknown.

Import

The program first imports all files in .csv or .fcs format, used for reference and prediction. For each file, the program checks if the channels name matches the expected channel names described beforehand via the flow cytometer parameter window. If one file is incompatible, a warning window will pop up, and the calculation will stop. Next, the program removes the 'Time' and 'Object number' columns from the imported file, if present, since they are irrelevant to the analysis.

Gating

Once imported, each file is gated according to the option selected.

- Line gating:

If the line gating is selected and the name of the cytometer contains 'accuri' or 'Accuri', the program will remove from each file all events respecting at least one of these equations:

$$FL3A == 0 \; or \; FL1A == 0$$

$$FL3A > 0{,}0241 \times FL1A^{1.0996}$$

$$FSCA > 100000 \; \& \; SSCA > 10000$$

If the line gating is selected and the name contains 'cytoflex' or 'Cytoflex', the program will remove from each file all events respecting at least one of these equations:

$$FL3A == 0 \; or \; FL1A == 0$$

$$log10(FL3A) > 1{,}5 \times log10(FL1A) - 2{,}8$$

$$log10(FL2A) > 2{,}5 \times log10(FL1A) - 9$$

If the name of the cytometer doesn’t contain the previous cited word, no gating is performed.

- Machine gating:

If the machine gating is activated, the user must select a record named 'BLANK', 'Blank' or 'blank' containing only medium background events. CellScanner performs no gating if the selected reference data does not contain one of these records.

Once the blank data is recognised, the program will create six classifiers against blank events for each record. If the blank files only contain background information, the records monoculture contains cellular events and medium background. The classifiers then predict the whole record (non-Blank) reference file(s) used for training. Six predictions per record reference file(s) are produced.

The program creates an average prediction for the six runs with a 70% unknown threshold. (cf Average prediction for the method). Because the background is present in both blank and record reference files, the background event will overlap between the two classes and be labelled either blank or unknown by the tool.

The program removes all blank and unknown predicted events from the record reference files and save these events in a new dataset of blank event. The background file is often small-sized; adding new events in the background dataset will improve the accuracy of the post-gating predictions.

In the case of ***Tool analysis*** or ***Clustering Analysis*** functions, the program also gates the files supplied by the user to create the in-silico community.

The gating uses the new blank dataset and the new gated reference records to train six predictive models per record (excluding blank). An average prediction is calculated for each record, and the unknown and blank values are removed from the new prediction file information.

In the case of ***Prediction*** and ***Clustering*** function, the community files are not gated. The blank dataset with other record reference datasets trains the classifiers used for prediction. The community file's event can be labelled blank if recognised as such.

All files selected for the same record are merged before the gating. So, the figures produced during the gating contain all data from the files of the same record.

Event selection

After the gating, the program treats the prediction files. If the community is unknown, the program will only create a Data Frame for each file provided. In the case of an in-silico community, the program will:

Only for the clustering functions:

- Remove all lines with NAN (not a number)
- Apply the logarithm base ten on each value
- Remove all infinite values

For all functions:

- Merge the data frames from the same record (if several files are provided for the same record)
- Randomly select a specific number of events per record (according to the number of cells per record parameter) or take the whole file if the parameter is set to 0.
- Add a column with the record label of every event.
- Merge all the record information and mix them.
- Save the record label into a table to compare later with the predicted results.

The training and test set selection from references is made at the beginning of each run since the selection must be different each time.

Classifier training, test and number of runs

The user defines the number of runs that default at 10. For each run, a classifier is created, trained and tested on reference data, then applied to a community file (in vitro or in silico) to predict its content.

Selection

The program selects a number of events the user indicates per record (1000 by default) from merging all reference files. Each event is randomly selected with the randint function from the random python package: A number is given between 0 and the size of the reference data frame for a record. Once an event is selected, this event is moved from the initial dataset to a new reference dataset. A new record is randomly selected by choosing a number between 0 and the size of the previous reference table minus the removed event (sampling without replacement).

After selecting the reference data, the program will create a predictive model according to the machine learning method selected by the user. It includes:

- Logistic regression: LogisticRegression function from Scikit-learn package.
- Random forest: RandomForestClassifier function from Scikit-learn package

- Neural network: MLPClassifier function from Scikit-learn package
- Random guessing: this method randomly assigns an event to a record from the list of records used in the calculation.

For each Scikit-learn function, the program splits the reference data into a training and a test set. The user selects the test size that default to 1/7th (0,14) of the reference data. The rest, 6/7th, is used for the training. Finally, a classifier is created, and statistical information assessing its performance on the test set is saved in result files.

The classifier is then used on the data to predict a community (in vitro or silico) and return label prediction from each event on the prediction data.

A classifier is created as often as a prediction is performed, namely the number of runs. The results and statistics can be saved directly, or an average prediction can be calculated from these predictions.

Average prediction and unknown parameters

If the average parameter is selected in the advanced parameter window, the program will proceed as follows:

The program counts the predicted record occurrence for an event from all predictions. This event is then labelled as the record the most occurrent through the n prediction for this specific event. This is described as a majority vote between the classifiers.

Because an event label is predicted as the same record though all prediction is more trustable than an event recognised as three different records, the user can set a threshold to rename 'unknown' the less reliable labels. This unknown threshold sets a minimum percentage of classifiers that must agree on an event to validate the label. For example, for ten runs and a 70% unknown threshold, if eight classifiers agree on labelling an event as record A (= 80% of the classifiers), the event will be labelled A. However, if only six classifiers recognise an event as record B (=60% of the classifiers), the event is labelled as 'unknown'.

Result production

A directory containing the results, named after its creation date and time (Format: YYYYMMDD-HH-MM-SS), is produced during each calculation. It contains the following files:

- The options file is produced at the beginning, and figures, are produced after the gating and after the predictions.
- The training graph matches the first run only. The training confusion matrix is a mean of the n confusion matrices (where n is the number of runs).
- The perdition graph is either the graph from run one if the average prediction is not selected or the graph for the average prediction. The confusion matrix of the ***Tool Analysis*** function can be the mean of the n confusion matrices generated during the n runs or the confusion matrix matching the average prediction.

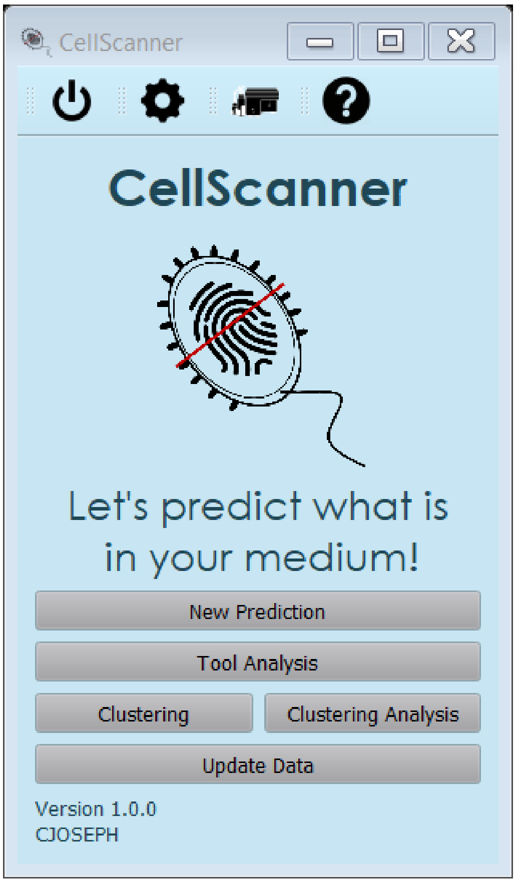
The file format is CSV with ';' separator and '.' As unit separator.

If the average parameter is selected in the advanced parameter window, the program will proceed as follows:

III- Main Window:

The main window (Figure 2) allows the launching of the different functions of the tool.

At the bottom part of the window, five buttons resume the principal functions:



Let's predict what is in your medium!

Version 1.0.0
CJOSEPH

Figure 2: CellScanner main window

- ***New Prediction (Prediction)***: Launch the program function for prediction(s) of in-vitro community data files.
- ***Tool Analysis***: Launch the program function to assess the program's ability to differentiate records by building an in-silico community.
- ***Clustering***: Launch the program function to find clusters in an in-vitro community, and label them according to the reference file.
- ***Clustering Analysis***: Launch the program function to assess the program's ability to find clusters and label them correctly by building an in-silico community.
- ***Update Data***: Give access to the database. This function allows the user to define, modify and complete the reference data information. Setting the reference data can be necessary before running a ***Tool Analysis*** or a ***Prediction***.

At the top of the window, the toolbar contains three buttons:

- ***Exit button***: to quit the program
- ***Parameter button***: Give access to the advanced parameter
- ***Flow* cytometer button**: Give access to the flow cytometer parameter; the user can define which channel to consider for a specific cytometer.
- ***Help button***: open the Help file, online or from the CellScanner directory

IV- Database and accesses:

CellScanner is linked to a database which contains needed information to run predictions on the data from flow cytometry. The database (Figure 3) is built as follows:

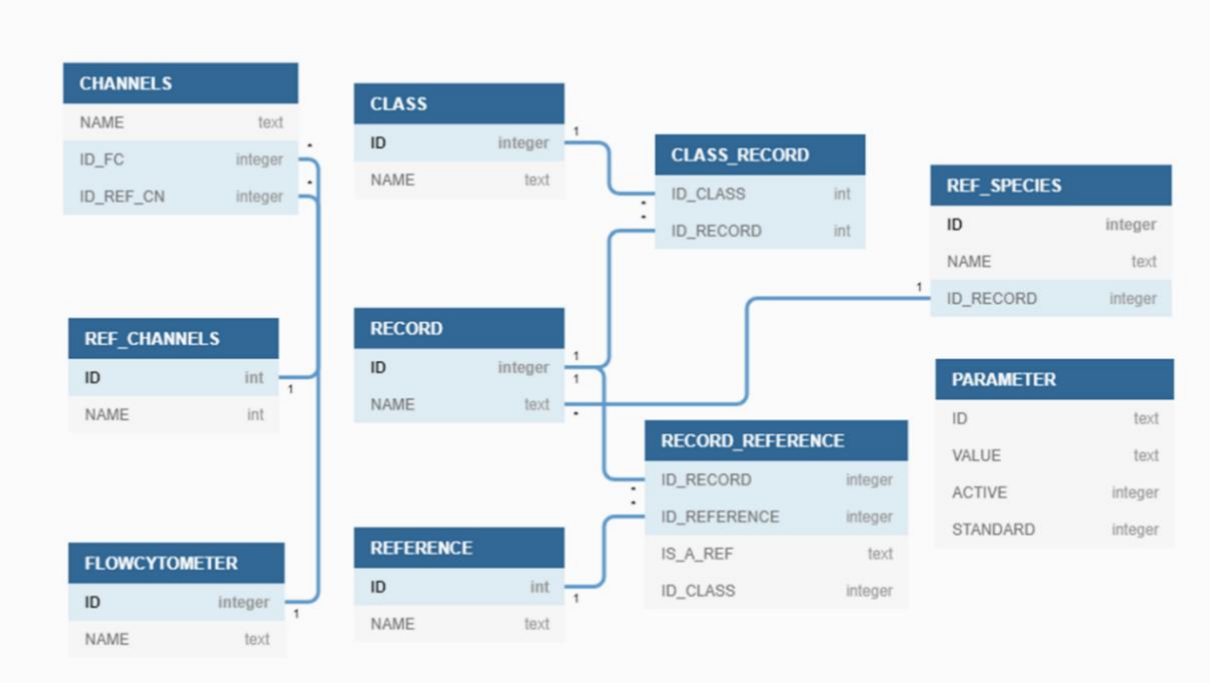


Figure 3: CellScanner database architecture.

The database is divided into three parts. The first one contains the tables 'CHANNELS', 'REF_CHANNELS' and 'FLOWCYTOMETER' and is accessible by the user through the ***Flow cytometer*** button. The 'FLOWCYTOMETER' table contains information about a flow cytometer, including its name and identifier. The REF_CHANNELS table contains the name of the channels that are essential for the calculation, including 'FL1-A', 'FSC-A' of 'SSC-A' needed for the line gating included in the tool. Finally, the 'CHANNELS' table contains a channel's name, the ID of its flow cytometer and reference channel ID if corresponding to one. A flow cytometer is linked to several channels; each channel is linked to one flow cytometer and a maximum of one reference channel.

The second part contains the tables 'CLASS', 'RECORD', 'REFERENCE', 'CLASS_RECORD', 'RECORD_REFERENCE', and 'REF_SPECIES', which describe the reference data information and are accessible through the ***Update Data*** function. A class is composed of several records that possess several references. A record can be in different classes; a reference can also describe several records. The three tables are associated with each other via the tables CLASS_RECORD and RECORD_REFERENCE. The table REF_SPECIES contains a list of bacterial species that CellScanner uses to perform auto-completion of the record name.

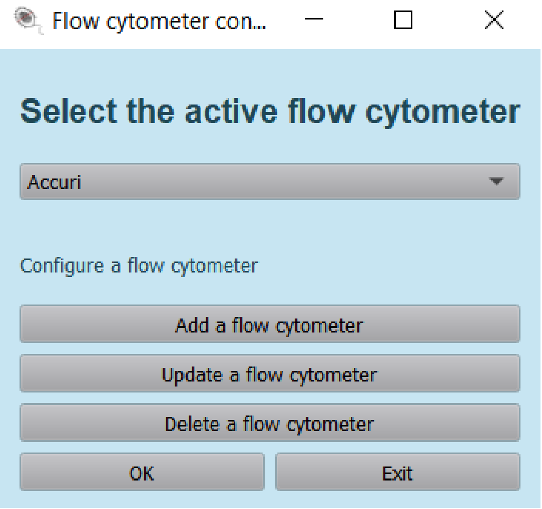
The last table, PARAMETER, contains all the advanced parameter information, their default values and the last values used by the user. These are modifiable via the ***Parameter*** button.

Flow cytometer button:

According to flow cytometers used by the user, the channel names can change. Therefore, the program needs to know which channels are present and which one should be taken into account for the analysis.

The user can decide which channels to use when creating a flow cytometer. Reducing the number of channels selected reduces the computing time. Furthermore, some channel needs to be identified to perform the line gating (cf program/gating).

The flow cytometer window (Figure 4) allows the user to manage flow cytometer information. On the top, the user can select the active cytometer by choosing it in the list and clicking the ok button. The user can create, update or delete a flow cytometer by using the buttons below the list.



Flow cytometer con...

Select the active flow cytometer

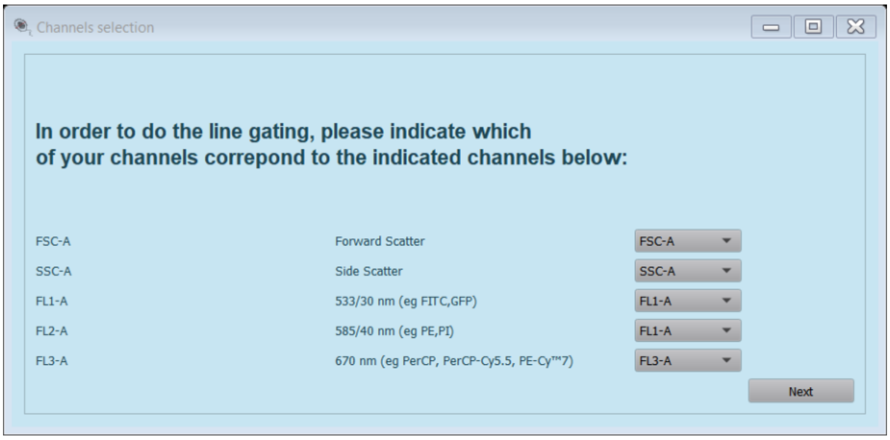
Accuri

Configure a flow cytometer

Figure 4: CellScanner window for FC parameters, obtains from main window FC icon.

- Add a flow cytometer

When creating a new flow cytometer, the user must select a name that does not already exist on the list, or an error message will pop up. If not existing, a new element is created in the FLOWCYTOMETER table of the database. After validating the cytometer name, the user must select a .csv of the .fcs file produced by the target flow cytometer. The program will extract the column names from it. Next, the user must link channels from his file to reference channels based on their description (Figure 5). The program will use these reference channels to perform the line gating.



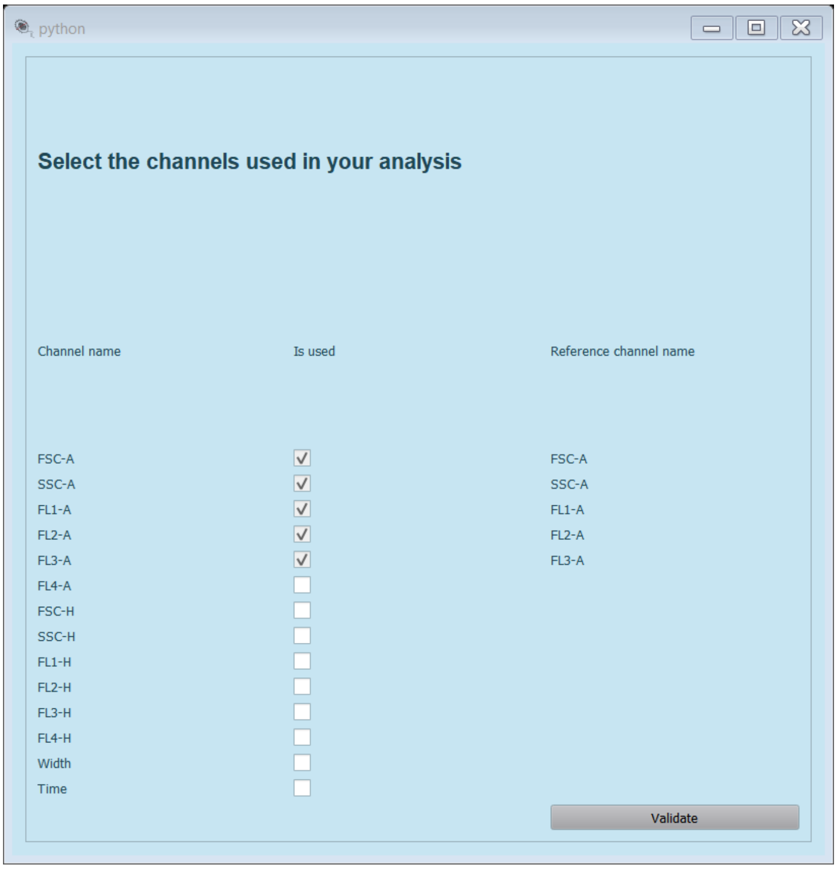
Channels selection

In order to do the line gating, please indicate which
of your channels correpond to the indicated channels below:

FSC-A	Forward Scatter	FSC-A
SSC-A	Side Scatter	SSC-A
FL1-A	533/30 nm (eg FITC,GFP)	FL1-A
FL2-A	585/40 nm (eg PE,PI)	FL1-A
FL3-A	670 nm (eg PerCP, PerCP-Cy5.5, PE-Cy™7)	FL3-A

Figure 5: Channels selection from FC parameter window, accessible from the 'Add a flow cytometer' and 'Update a flow cytometer' button. User can select the channel from their FC matching the CellScanner cytometer names.

Once the user indicates the reference channels (Figure 6), they must select the channels valuable for the predictions. This information is saved in the CHANNELS table in the database.



Select the channels used in your analysis

Channel name	Is used	Reference channel name
FSC-A	☑	FSC-A
SSC-A	☑	SSC-A
FL1-A	☑	FL1-A
FL2-A	☑	FL2-A
FL3-A	☑	FL3-A
FL4-A	□	
FSC-H	□	
SSC-H	□	
FL1-H	□	
FL2-H	□	
FL3-H	□	
FL4-H	□	
Width	□	
Time	□	

Figure 6: CellScanner window for selection of the channel to take into account for the future calculation. The previously described channels are already selected.

The active cytometer must match the reference data files and the selected files for the calculation. If it is not the case, the program will alert the user and cancel the calculation.

CAUTION: Line gating was developed for *Accuri* and *Cytoflex* flow cytometers. Please include one of these in your cytometer name if you want to perform one or the other line gating.

- Update a flow cytometer

The user can modify the selection of the channels linked to a FC name. The user repeats the steps of the FC creation.

- Delete a flow cytometer

The user can delete one or several FC from a list.

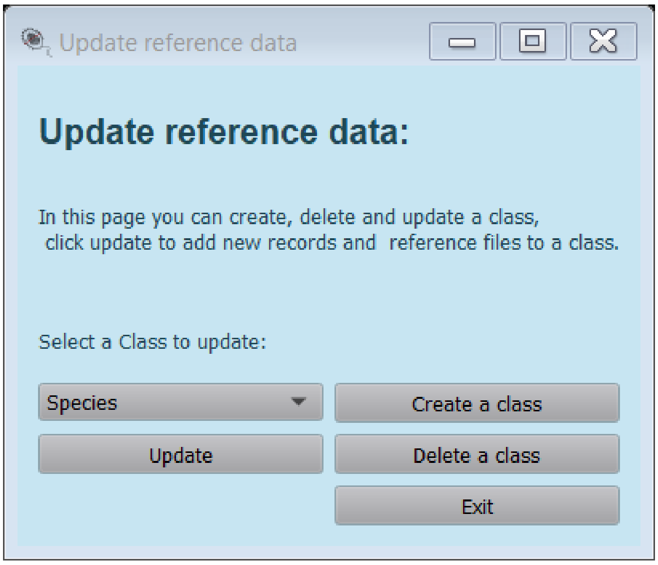
At the end of all modification the user should select their active FC and validate it. If not done an error message will appear later.

Update data function

To perform a prediction, the user needs to indicate reference information to complete the supervised step of machine learning. To facilitate the management of this reference data, the user can save this data and arrange it as he wants by creating objects from the three different types: a class, a record and a reference.

A class is defined by several records (e.g. 'Species'). A record is one of the potential labels used to identify an event (e.g. 'Escherichia coli'). A reference is a link to a .csv or a .fcs file containing flow cytometry data about a specific record. The database links a class from two to many unique records. Each record is associated with one to several unique references. The same record name can be used in different classes. In this case, the records' references will depend on the class.

The user can set several classes according to their needs by using the **Update Data** function (Figure 7).



Update reference data

Update reference data:

In this page you can create, delete and update a class,
click update to add new records and reference files to a class.

Select a Class to update:

Figure 7: Update data window, give access to all record and reference file from the main window.

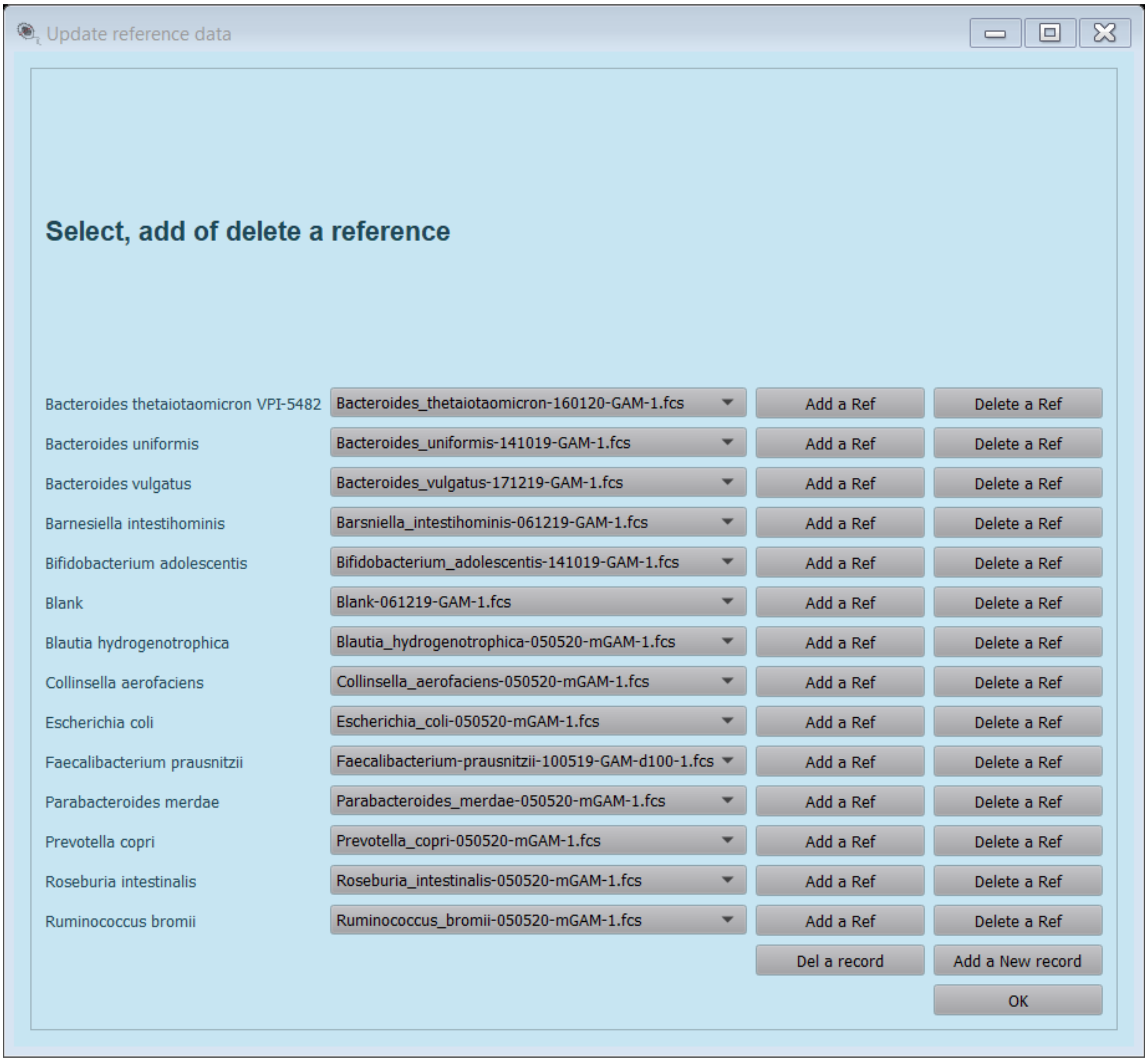
From the main Update reference data Window, the user can either:

- ***Create a class***: After clicking the button, the user enters the name of the new class, and after validation, the new class is created and saved in the CLASS table of the database.

- ***Delete a class***: After clicking the button, the user selects the class/es from the list they want to delete. After validation the class/es is/are deleted.

- ***Update*** a class: After selecting the target class on the list (grey button on the top left) and clicking the button update, a new window appears where the user can modify the record and the reference for the selected class.

The Class species is already in the database and contains some data. By updating it the user can visualise the content of the class (Figure 8).



Select, add of delete a reference

Bacteroides thetaiotaomicron VPI-5482	Bacteroides_thetaiotaomicron-160120-GAM-1.fcs
Bacteroides uniformis	Bacteroides_uniformis-141019-GAM-1.fcs
Bacteroides vulgatus	Bacteroides_vulgatus-171219-GAM-1.fcs
Barnesiella intestihominis	Barsniella_intestihominis-061219-GAM-1.fcs
Bifidobacterium adolescentis	Bifidobacterium_adolescentis-141019-GAM-1.fcs
Blank	Blank-061219-GAM-1.fcs
Blautia hydrogenotrophica	Blautia_hydrogenotrophica-050520-mGAM-1.fcs
Collinsella aerofaciens	Collinsella_aerofaciens-050520-mGAM-1.fcs
Escherichia coli	Escherichia_coli-050520-mGAM-1.fcs
Faecalibacterium prausnitzii	Faecalibacterium-prausnitzii-100519-GAM-d100-1.fcs
Parabacteroides merdae	Parabacteroides_merdae-050520-mGAM-1.fcs
Prevotella copri	Prevotella_copri-050520-mGAM-1.fcs
Roseburia intestinalis	Roseburia_intestinalis-050520-mGAM-1.fcs
Ruminococcus bromii	Ruminococcus_bromii-050520-mGAM-1.fcs

Figure 8: Species class visualisation from the Update function window. Record names are on the left. On their right are lists of their references (at least one has to be activated in the list) followed by the function buttons to add and delete reference file(s).

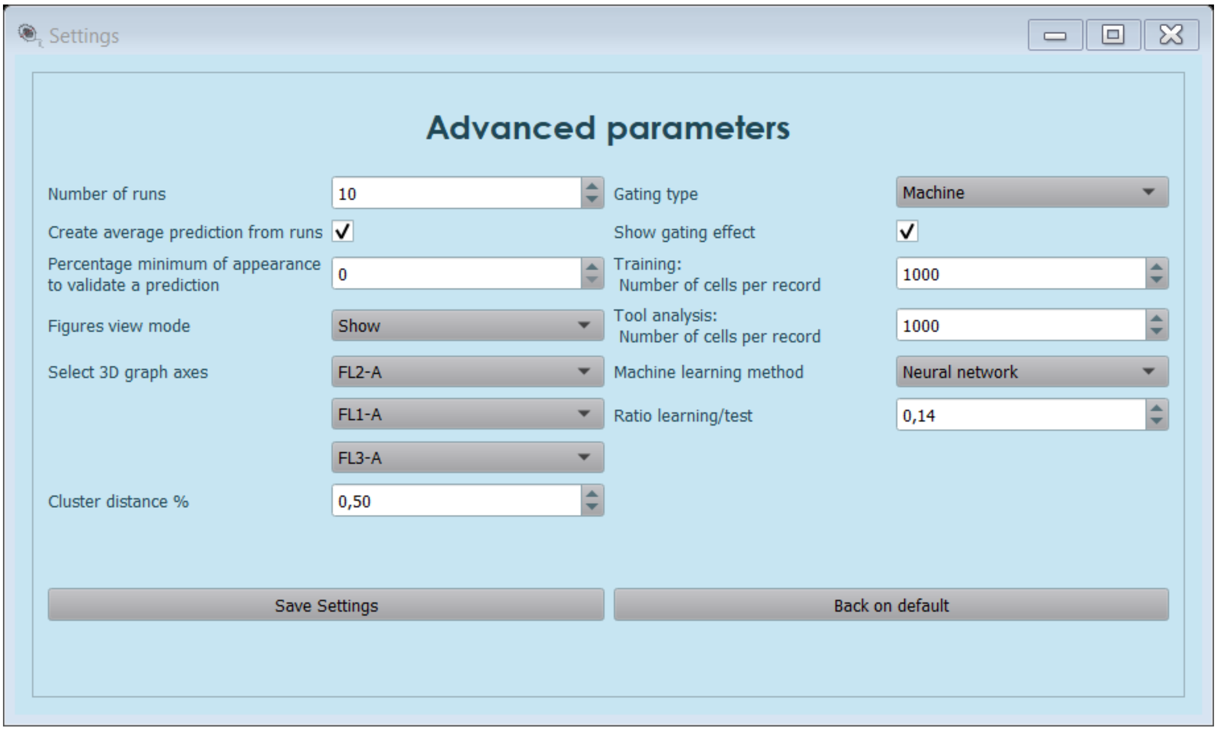
The user can add (‘Add a New record’) or delete (‘Del a record’) a record. Records are added to the database, and the class and the record are associated in the table CLASS_RECORD. For the class Species, an auto-completion system facilitates the naming of a species record; the pre-registered species names are stored in the REF_SPECIES table. Then for each species, the user can add one or several monoculture files as a reference by using the 'Add a Ref' button in front of the specific record. References information is stored in the table REFERENCE and is associated with a record in the table RECORD_REFERENCE. For this, users select reference files through their directories. These files are copied into the reference directory of the program, which the user should not modify manually. The name of the reference file will help the user recognise their experiment and should be informative.

Once added to the database, the user activates references for all functions by selecting the file(s) name in the popup list next to the record name. Several files per species can be used as references. If no reference is selected for a record, the record will not be available in the prediction or assessment functions. Once the database is completed, the user can use the main functions.

For each record, the user can delete one or several reference files with the dedicated button ‘Delete a Ref’ by selecting them in a list.

Settings window:

The settings windows (Figure 9) give access to the database by selecting the advanced parameters needed for the main functions. All the parameter values are saved in the database on the PARAMETER table.



Advanced parameters

Number of runs: 10
Create average prediction from runs: ☑
Percentage minimum of appearance to validate a prediction: 0
Figures view mode: Show
Select 3D graph axes: FL2-A, FL1-A, FL3-A
Cluster distance %: 0,50
Gating type: Machine
Show gating effect: ☑
Training: Number of cells per record: 1000
Tool analysis: Number of cells per record: 1000
Machine learning method: Neural network
Ratio learning/test: 0,14

Figure 9: Setting window from CellScanner main window, accessible with the parameter icon.

The user can define the following parameters:

- **Number of runs**: number of predictive models and predictions done for the ***Prediction*** and ***Tool Analys*** is functions.
- **Create average prediction from runs**: The program creates one prediction from the different runs based on majority vote for the ***Prediction*** and ***Tool Analysis*** functions. (cf: program)
- **Percentage minimum of appearance to validate a prediction:** if **Create an average prediction** is selected, it represents the percentage minimum of classifiers that need to agree to label an event. If under or equal the minimum value, the event is labelled unknown. Only apply for the ***Prediction*** and ***Tool Analysis*** functions. (cf: program)
- **Figures view mode**: The user can decide if the figure pops up on the screen, which allows the user to move the 3D graph to the best position with the 'show' option. The 'save', and save figures automatically on the result directory, or the user can select 'none', and no graph is produced during the experiment.
- **Select 3D graph axes**: allows the user to select the 3 axes on the 3D graphs
- **Cluster distance %:** percentage of distance maximum to address a cluster to a species.
- **Gating type**: select the gating method between None, line or machine.
    - None: no gating is carried out. We advise adding Blank as a species during the prediction.

- Line: the program will use rules described in the script on the *gatingFunction* function on the *machineLearningPackage* package, remove cells responding at the following conditions:
    - accuri:

$$FL3A \leq 0 \ or \ FL1A \leq 0$$

$$FL3A > 0{,}0241 \times FL1A^{1.0996}$$

$$FSCA > 100000 \,\&\, SSCA > 10000$$

$$\log(FSCA) > \log(FSCH) + 0{,}5$$

$$log(FSCA) > log(FSCH) - 0{,}5$$

    - cytoflex:

$$FL3A \leq 0 \ or \ FL1A \leq 0$$

$$\log(FL3A) > 1{,}5 \times \log(FL1A) - 2{,}8$$

$$\log(FL2A) > 2{,}5 * \log(FL1A) - 9$$

$$log(FSCA) > log(FSCH) + 0{,}6$$

$$\log(FSCA) > \log(FSCH) - 0{,}6$$

If the selected cytometer is different from Accuri or Cytoflex (it does not contain either name), no gating is performed.

- Machine: The program uses reference of species called Blank (or containing ['Blk', 'blk', 'Blank', 'blank']) To identify blank into every other reference file (and monoculture for ***Tool Analysis***). If this method is used, the user needs to select Blank as a species (cf methods).

- **Showing gating effect**: This parameter shows or saves 2D or 3D graphs on the effect of gating on data with the line gating. In the same way, a 3D graph is produced with machine learning gating.
- **Training**: **number of cells per record**: The number of cells randomly selected per record after merging the different references from the same record. If the file contains a low number of cells, the program will select the maximum number.
- **Tool analysis: number of cells per record**: The number of cells randomly selected per record after merging the different input files from the same record. If the file contains a low number of cells, the program will select the maximum number.
- **Machine learning method**: The prediction and the machine gating can be run with different Scikit-learn machine learning methods.
    - Neural network: cf program

- Random forest: cf program
- Logistic regression: cf program
- Random guessing: this function randomly attributes a record to a cell. It allows the user to differentiate a real assignation from a random one.

- **Ratio Learning/Test**:

From the number of cells selected for the machine learning training, the tool selects a part to train the model and the rest to test the model. This ratio is, by default, set at 6/7 for the training and 1/7 (0,14) for the test (not done for clustering).

The *Back on the default* button allows the user to reset the default settings. If the settings are modified, the changes are saved using the save settings button. The user needs to press the button to save the settings. The settings are kept in the database until the user changes them or goes back to default values.

V- Tool Analysis:

The **Tool Analysis** function has been developed to assess the prediction accuracy with a set of known information for each record. The program will predict the composition of an in-silico community created with monoculture files. The user should assess the tool's capability to differentiate specific species before using the **Prediction** function.

The user will go through the following steps:

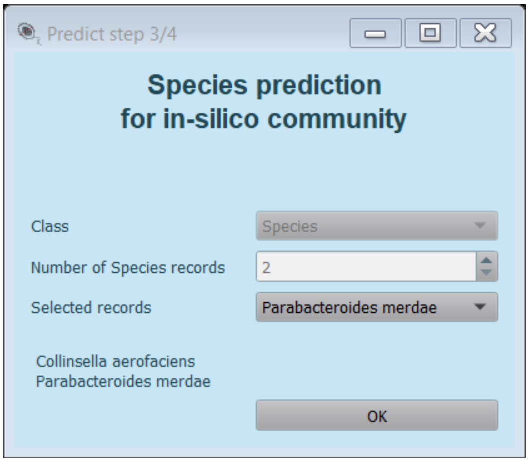
1. Class selection from the list

- Class selection

- Selection of the maximum number of records expected.

- Selection of the record expected from the list (Figure 10). If the list is incomplete, it may be due to missing or inactive references. The reference information is extracted from the database; the user can change the reference file via the **Update** function on the main window.

- Selection of one, several or no monoculture files for each species (at least one file in total) to use to create the in silico community (Figure 11).


Predict step 3/4

Species prediction
for in-silico community

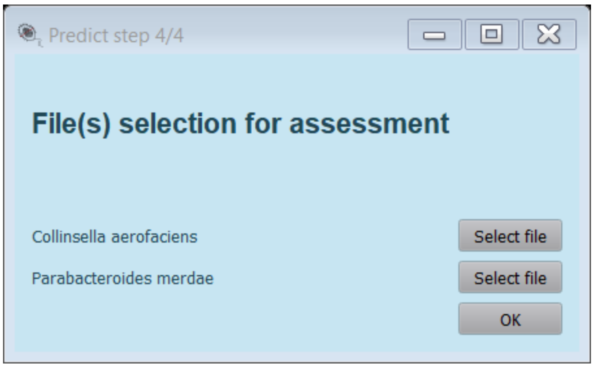
Class: Species

Number of Species records: 2

Selected records: Parabacteroides merdae

Collinsella aerofaciens
Parabacteroides merdae

Figure 10: Record selection from the selected class 'Species'. A similar window exists for the four CellScanner functions.


Predict step 4/4

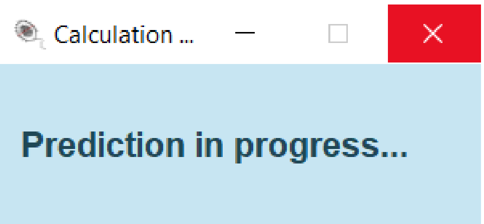
File(s) selection for assessment

Collinsella aerofaciens

Parabacteroides merdae

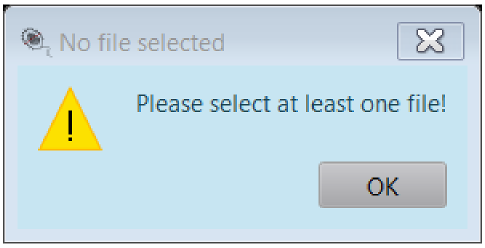
Figure 11: File selection window to create an in silico community. Several file can be selected per records. A multiselection can only be done on one selection

- Validate with the OK button to start the calculation (Figure 12). At least one file has to be selected, or an error message will appear (Figure 13).


Calculation ...

Prediction in progress...

Figure 122: In progress window, the window stays until the ed of the calculation.

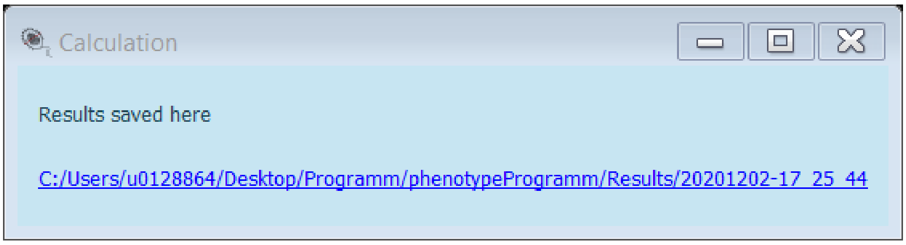

No file selected

Please select at least one file!

Figure 133: Pop-up error message window for a insufisant number of file.

The running time can vary depending on your computer and the parameters of the prediction. It can take 1 minute to 20 min.

Once the calculation is done, a window appears (Figure 14) with a clickable link to the saved results (not clickable on Mac).



Results saved here

C:/Users/u0128864/Desktop/Programm/phenotypeProgramm/Results/20201202-17_25_44

Figure 14: CellScanner result window. The window appears at the end of the calculation and show the path to the result directory. For Windows computer, the path is a clickable link.

If the figure view mode option 'show' was selected in advance parameters, 2D graphs, 3D graphs, and confusion matrix will pop up on the screen during the calculation. The user can manipulate and save them; the calculation continues once the figures' windows are all closed.

Results are saved for the gating (if selected), training/test step (***Tool analysis*** prediction only) and the prediction step. The results contain .csv files, .txt files and .png figures. Result files are detailed in the Result chapter.

If the results show that the program can distinguish the records in the in silico-community, the user can use the Prediction function.

VI- New prediction:

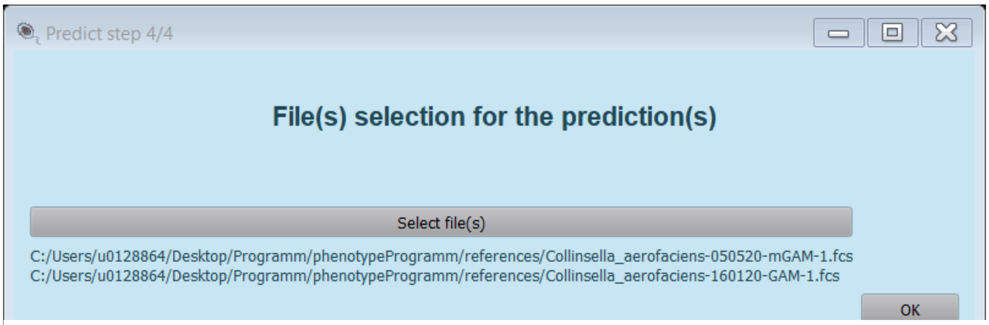
The prediction function is similar to the ***Tool Analysis*** unless the content of the files to predict is unknown. The GUI steps are the same unless for the file selection steps.

- Class selection

- Selection of the maximum number of records expected.

- Selection of the record expected from the list. If the list is incomplete, it may be due to missing or inactive references. The reference information is extracted from the database; the user can change the reference file via the **Update** function on the main window.

- Selection of one or several community files for prediction (Figure 15). CellScanner produces prediction results for each selected file.



C:/Users/u0128864/Desktop/Programm/phenotypeProgramm/references/Collinsella_aerofaciens-050520-mGAM-1.fcs
C:/Users/u0128864/Desktop/Programm/phenotypeProgramm/references/Collinsella_aerofaciens-160120-GAM-1.fcs

Figure 15: File selection window for in vitro community files. Several files can be selected at once, one predition will be produced for each of them.

- The validation of the selection launches the prediction. If no file is selected, an error message will appear. The running time depends on the number of records in the prediction, the number and size of the files selected, and the chosen advanced parameters.

Once the calculation is performed, a window appears with a clickable link to the saved results (not clickable on Mac). Mac users can either directly navigate to the results folder or paste the link into a browser window to access the results.

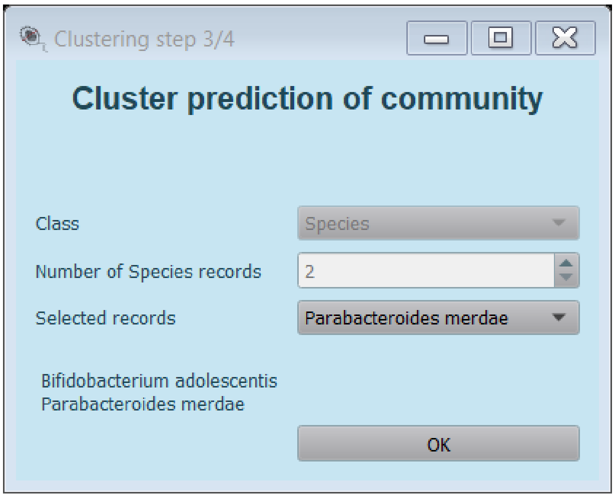
In the created directory, the user can find .png, .csv, and .txt files containing results of the test done on the classifier, with the reference data and results about the prediction for the file(s) selected.

VII- Clustering & clustering analysis:

CellScanner includes an unsupervised classification function in case users do not have high-quality reference data. The Agglomerative clustering function from Scikit-learn is used to identify clusters, and CellScanner suggests a species annotation. The ***Clustering Analysis*** function assesses the method by creating an in silico community from monocultures, and the ***Clustering*** function performs a prediction.

These functions are useful for species forming distinct clusters. The parameters used with the clustering functions are different. The number of runs, training set size, and Ratio learning/ test is not applicable for clustering functions. (c.f. advanced parameter)

The clustering functions use reference files to label the clusters by comparing their position to the record reference data (Figure 16). The launching process of ***Clustering analysis*** and ***Clustering*** is similar to ***Tool analysis*** and ***Prediction*** functions, respectively (Refer to these chapters for interface visuals).



Clustering step 3/4

Cluster prediction of community

Class: Species

Number of Species records: 2

Selected records: Parabacteroides merdae

Bifidobacterium adolescentis
Parabacteroides merdae

Figure 16: Record selection window for clustering function. This indicated the reference to use, to try to label the clusters, and the number of clusters maximum in the community.

After calculation, a window with a link pops up on the screen to access the result directory. This directory contains 3D graphs, confusion matrix, .csv and .txt files containing the ***Clustering*** or the ***Clustering Analysis*** results. Results are described in the Result chapter.

VIII- Results Files

This chapter indexes all the files produced by the program for the different functions and steps. The file produced depends on the parameter chosen.

All the .png figures are saved in the result directory if the option 'save' is selected on the Figure view mode in the advanced parameters. If the parameter is set to 'show', the figures will appear on the screen during the calculation. It is the responsibility of the user to save their figures. If it is set None, no figures are created.

From the different steps, several files are shared between the different functions. Nevertheless, the ***Clustering*** functions do not produce statistics on a test or reference data since no training is done. The following files are created:

New prediction

- Gating files (one per reference file and one per prediction file) for line
- cmData.csv
- Learning_Stat.csv
- Option.txt
- Predictions.csv
- Reference.csv
- References_cm.png
- Training_graph_reference.csv
- Graph_tool_analysis_file.png (one for each predicted file)

Tool analysis

Gating file (one per reference file and one per prediction file) for line

- assessment.csv
- cmData.csv
- Learning_Stat.csv
- option.txt
- Pred_stat.csv
- Predictions.csv
- Reference.csv
- References_cm.png
- Tool_analysis_species_cm.png
- Training_graph_reference repeat 1.csv
- graph_tool_analysis_*n*_species.png

Clustering prediction

- Gating file (same)
- Predictions.csv
- Option.csv
- Clustering graph (one per file)

Clustering analysis

- Gating file
- assessment.csv
- cmData.csv
- option.txt
- Pred_Stat.csv
- Tool_analysis_species _cm.png

Gating figures:

CellScanner produces gating figures only if the advanced parameter 'show gating' is active and if the type of gating is not 'None'. The result obtained will depend on the type of gating selected.

If line gating is chosen, the program will produce a 2D graph to show the gating done on each reference and each prediction file (Figure 17). The red events are removed from the calculation. To perform the line gating, the flow cytometer must contain 'accuracy' or 'cytoflex'.

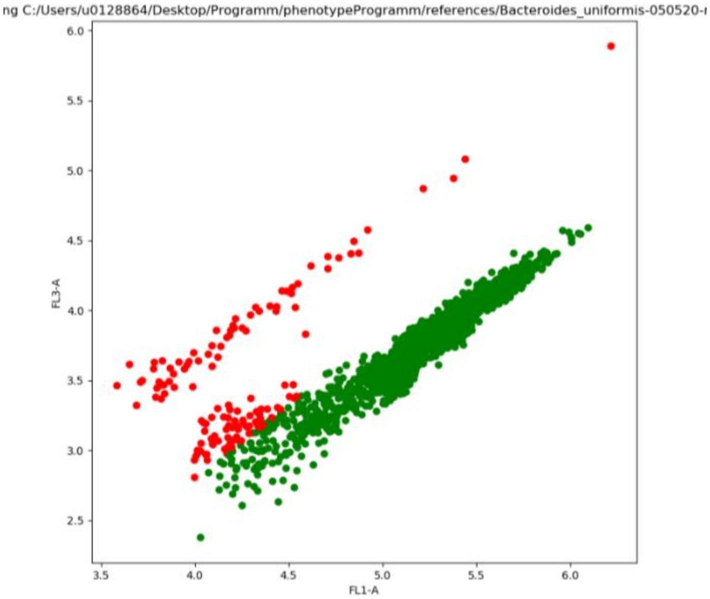


Figure 17: Line gating plot example of Bacteroides unformis with an Accury FC. The red event are removed from the calculation; the greens are kept as events representative of B. uniformis species.

Suppose the machine gating is chosen, and 'blank', 'Blank', 'blk', or 'Blk' is one record during the prediction. In that case, one 3D graph will be produced for each record's reference files and one for each monoculture used for ***Tool Analysis*** (Figure 18) (all reference files for the same record are merged). Blank and unknown values are removed from the calculation.

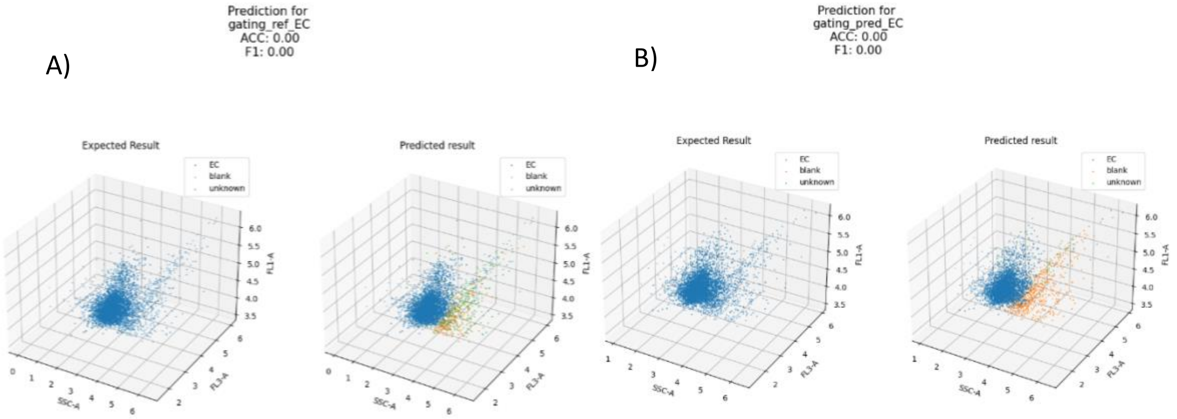


Figure 18: Machine gating plot example for Escherichia coli. A) The gating is performed on reference data. On the left, the graph shows the original label of the events coming from the Escherichia coli monoculture (EC). After training the models with blank and EC monoculture, this same EC (blue) monoculture is predicted with blank (orange) and unknown (green) events. The Blue events are kept to represent EC, and the green and orange events are added to the blank event for future predictions. B) The gating is performed on data used for prediction. Events labelled as blank or EC are more distinct with less unknown since the models are now trained with gated monoculture and more blank events.

The gated files from the reference record are called: graph_tool_analysis_ref_*recor*d.*png*

The gated files from in silico community record are called: graph_tool_analysis_pred_*recor*d.*png*

Prediction figures:

3D graphs

The program produces two types of 3d graphs during the calculations (Figure 19):

Two windows allow the user to compare the expected (Expected result) and predicted results (Predicted result). If the composition of the file is not known, the expected result will only appear as unknown. The user can select the axes on the advanced parameter window before calculating. If the plot pops up in a window, the angle of view can be modified by the user and saved. This graph is produced for the test, prediction, and machine gating of CellScanner.

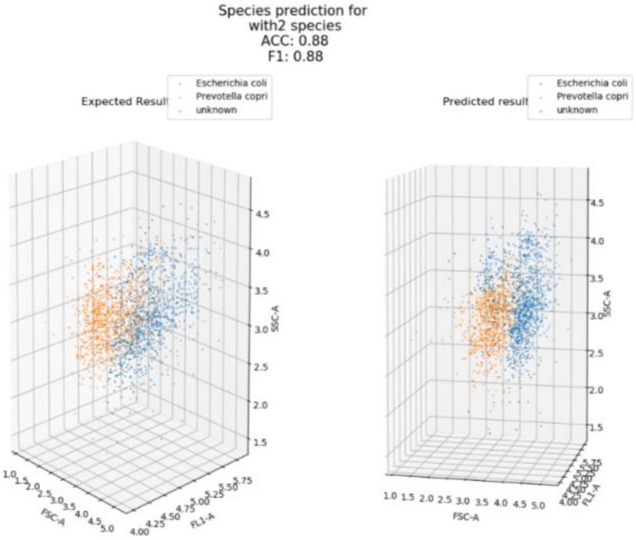


Figure 19 : Example of a 3D graph obtained with Tool Analysis function for an in silico community of Escherichia coli and Prevotella copri. On the left is the graph with the expected result; events are labelled according to their monoculture file origin. The graph on the right shows the event labelled after prediction. 3D graphs are produced in several steps. If the composition of the file is unknown, all events from the left graph would be labelled as unknown.

For the ***Clustering*** function, the plot contains three 3D graphs (Figure 20). One for the reference information, one for the clustering result, and one where the cluster is labelled according to the distance from reference data. The user can then validate the cluster labelling or change it according to their thought.

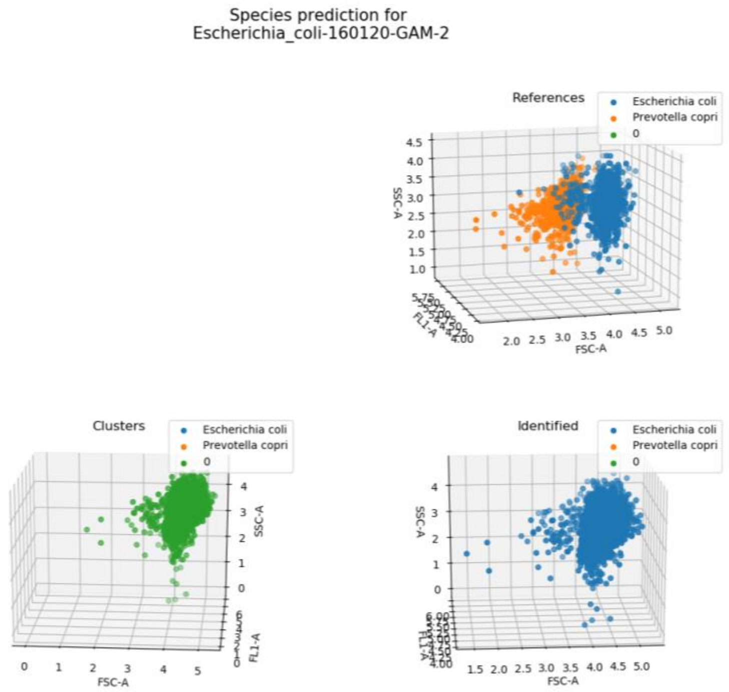


Figure 20: Example of 3D graph for a clustering in-vitro community prediction of Escherichia coli and Prevotella copri. On the top right, the graph shows the event for the two species of the community based on reference data. On the bottom left, the graph represents the cluster found by CellScaner. on the bottom right, the graph shows a label assignation to the found cluster after comparing the cluster to the reference data. Here only Escherichia coli is expected in the community, so the prediction is correct.

Confusion matrix

The confusion matrix is a table presenting the expected results against the predicted results (Figure 21). The result is normalised; each line represents 100 % of the record population. The user can find on the diagonal the agreement between the expected and the predicted label (i.e. true positive values). Also, the user can find the false positives (identified per column, outside of the diagonal) and the false negatives (identified per row outside of the diagonal). A confusion matrix is produced when testing a classifier and predicting an in-silico community with the ***Tool analysis*** or the ***Clustering Analysis*** function. If the diagonal is dark blue, the program can distinguish the species from each other. If events from a record are often wrongly assigned to another, they are supposedly more alike (e.g. 5 % of the time, *Escherichia coli* is identified as *Bifidobacterium adolescentis*)

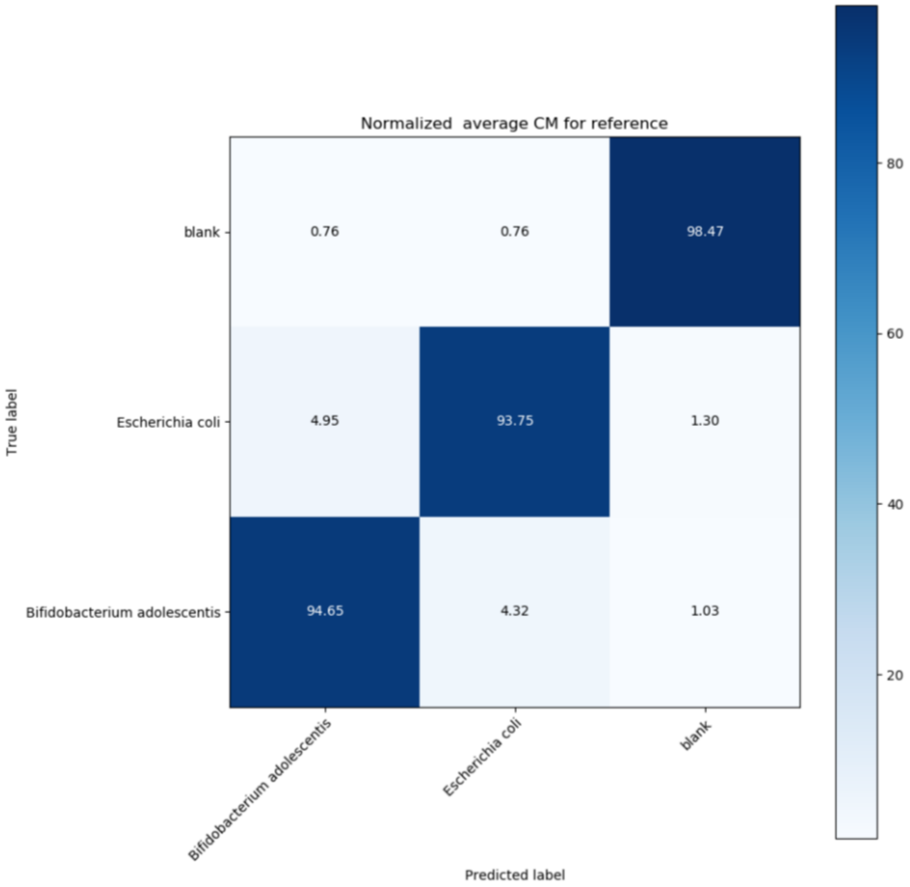


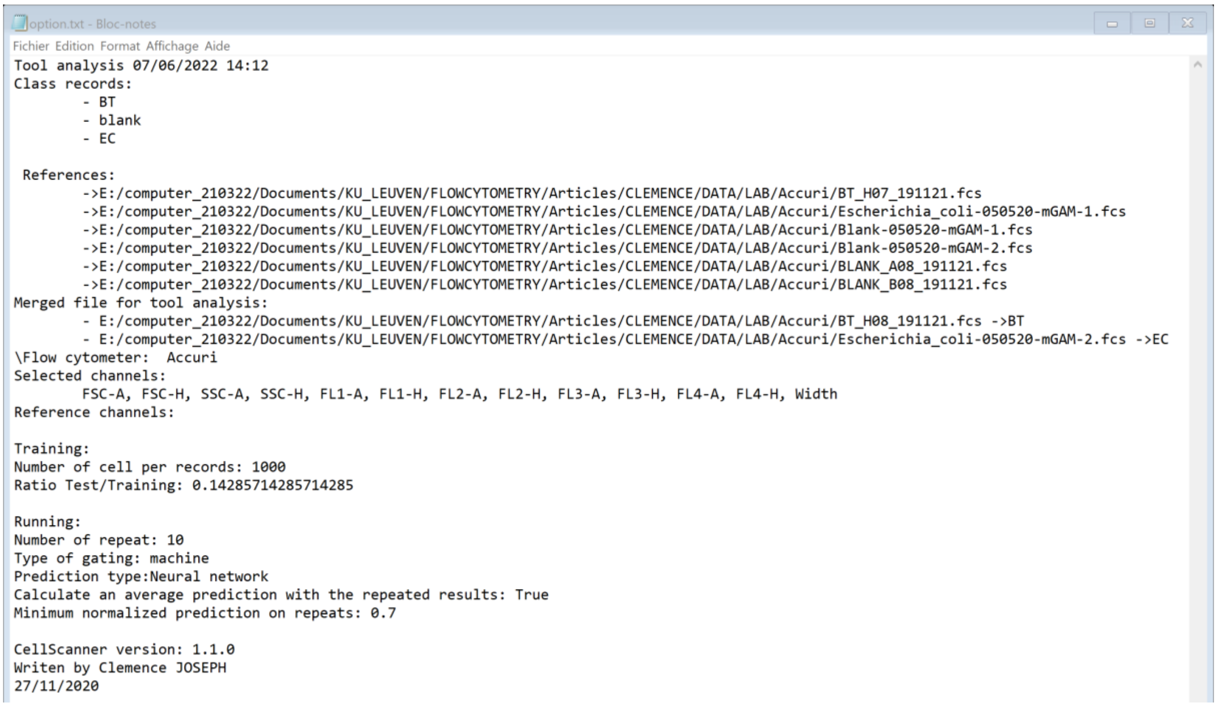
Figure 21: Example of confusion matrix resulting from the in silico community prediction of Escherichia coli and Bifidobacterium adolescentis. On the left are the correct labels; each line represents 100% of the record. On the bottom is shown the predicted label. The correct labelling is shown on the diagonal square. Outside of this are the wrongly assigned events (false positives and false negatives)

Option file

The options file is produced at the beginning of each calculation (Figure 22). It describes all the parameters and data selected for the calculation. With this file, the user can access parameters used in the past and reproduce experiments. It contains:

- a description of the function used,
- the date and time of the calculation,
- Data about files used for the prediction and as a reference,
- Advanced parameter selected (number of runs, number of cells per species…)

- The cytometer name and channels selected,
- The tool version



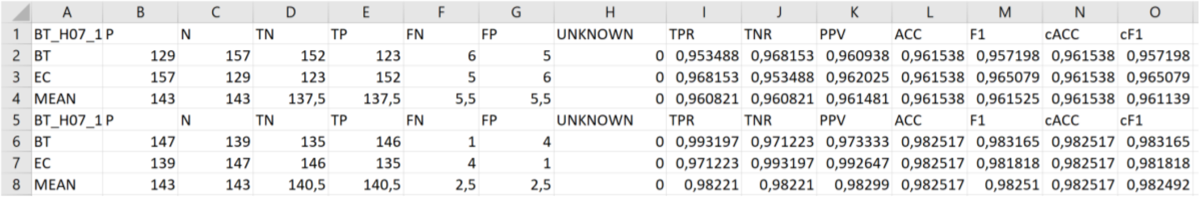
```
Tool analysis 07/06/2022 14:12
Class records:
        - BT
        - blank
        - EC

 References:
        ->E:/computer_210322/Documents/KU_LEUVEN/FLOWCYTOMETRY/Articles/CLEMENCE/DATA/LAB/Accuri/BT_H07_191121.fcs
        ->E:/computer_210322/Documents/KU_LEUVEN/FLOWCYTOMETRY/Articles/CLEMENCE/DATA/LAB/Accuri/Escherichia_coli-050520-mGAM-1.fcs
        ->E:/computer_210322/Documents/KU_LEUVEN/FLOWCYTOMETRY/Articles/CLEMENCE/DATA/LAB/Accuri/Blank-050520-mGAM-1.fcs
        ->E:/computer_210322/Documents/KU_LEUVEN/FLOWCYTOMETRY/Articles/CLEMENCE/DATA/LAB/Accuri/Blank-050520-mGAM-2.fcs
        ->E:/computer_210322/Documents/KU_LEUVEN/FLOWCYTOMETRY/Articles/CLEMENCE/DATA/LAB/Accuri/BLANK_A08_191121.fcs
        ->E:/computer_210322/Documents/KU_LEUVEN/FLOWCYTOMETRY/Articles/CLEMENCE/DATA/LAB/Accuri/BLANK_B08_191121.fcs
Merged file for tool analysis:
        - E:/computer_210322/Documents/KU_LEUVEN/FLOWCYTOMETRY/Articles/CLEMENCE/DATA/LAB/Accuri/BT_H08_191121.fcs ->BT
        - E:/computer_210322/Documents/KU_LEUVEN/FLOWCYTOMETRY/Articles/CLEMENCE/DATA/LAB/Accuri/Escherichia_coli-050520-mGAM-2.fcs ->EC
\Flow cytometer:  Accuri
Selected channels:
        FSC-A, FSC-H, SSC-A, SSC-H, FL1-A, FL1-H, FL2-A, FL2-H, FL3-A, FL3-H, FL4-A, FL4-H, Width
Reference channels:

Training:
Number of cell per records: 1000
Ratio Test/Training: 0.14285714285714285

Running:
Number of repeat: 10
Type of gating: machine
Prediction type:Neural network
Calculate an average prediction with the repeated results: True
Minimum normalized prediction on repeats: 0.7

CellScanner version: 1.1.0
Writen by Clemence JOSEPH
27/11/2020
```

Figure 22: Example of an option file for a Tool analysis prediction with Bacteroides thetaiotaomicron and Escherichia coli. It summarises the parameter and the files used for the calculation. Data files:

Learning_stat.csv

This file contains statistical information about the training step (Figure 23). The program creates a table per run and adds it to the same file. Two tables are visible in the following figure.

In the first column appears the name of the files used and the names of the records. The MEAN indicates the mean value for the community.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	BT_H07_1	P	N	TN	TP	FN	FP	UNKNOWN	TPR	TNR	PPV	ACC	F1	cACC	cF1
2	BT	129	157	152	123	6	5	0	0,953488	0,968153	0,960938	0,961538	0,957198	0,961538	0,957198
3	EC	157	129	123	152	5	6	0	0,968153	0,953488	0,962025	0,961538	0,965079	0,961538	0,965079
4	MEAN	143	143	137,5	137,5	5,5	5,5	0	0,960821	0,960821	0,961481	0,961538	0,961525	0,961538	0,961139
5	BT_H07_1	P	N	TN	TP	FN	FP	UNKNOWN	TPR	TNR	PPV	ACC	F1	cACC	cF1
6	BT	147	139	135	146	1	4	0	0,993197	0,971223	0,973333	0,982517	0,983165	0,982517	0,983165
7	EC	139	147	146	135	4	1	0	0,971223	0,993197	0,992647	0,982517	0,981818	0,982517	0,981818
8	MEAN	143	143	140,5	140,5	2,5	2,5	0	0,98221	0,98221	0,98299	0,982517	0,98251	0,982517	0,982492

Figure 23: Example of Learning_stat.csv file for a Tool Analysis training step with Bacteroides thetaiotaomicron (BT) and Escherichia coli (EC). Only two out of ten runs result are shown. This file summarises the statistical test result for each classifier. All values are explained in the main text.

It contains:

P: Expected positive value

N: Expected negative value

TN: True negative

TP: True positive

FN: False negative

FP: False positive

UNKNOWN: number of events predicted unknown for the record

TPR: True Positive Rate (Sensitivity)

$$TPR = \frac{TP}{P}$$

TNR: True Negative Rate (Specificity)

$$TNR = \frac{TN}{N}$$

PPV: Positive Predicted Value (Precision)

$$PPV = \frac{TP}{TP + FP}$$

ACC: Accuracy

$$ACC = \frac{TP + TN}{P + N}$$

F1: F1 score

$$F1 = \frac{2TP}{2TP + FP + FN}$$

cACC: Corrected accuracy

$$cACC = \frac{TP + TN}{P + N - UNKNOWN}$$

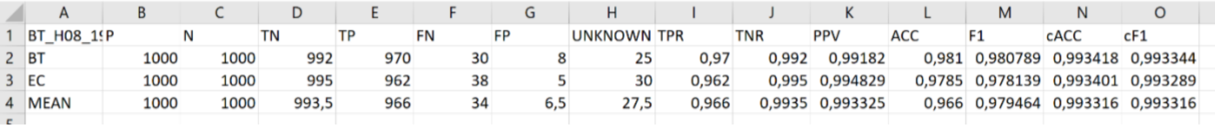
cF1: Corrected F1 score

$$cF1 = \frac{2TP}{2TP + FP + FN - UNKNOWN}$$

The unknown events are only present for an average prediction if the option is selected. On this file, values are always null, so F1 and cF1 are the same such as ACC and cACC.

Pred_stat.csv

This file contains the same statistic information but for the ***Tool Analysis*** function about the in-silico community prediction (Figure 24). It has one table if the average prediction parameter is activated or as much as the number of runs if not. Here known value can be different from zero, and the corrected accuracy and F1 score are modified according to the previous equation.

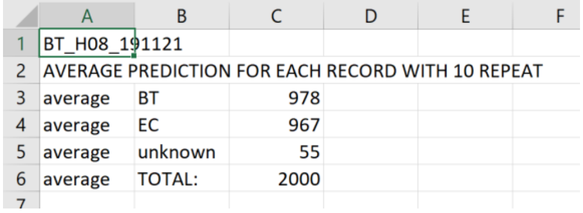


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	BT_H08_1!	P	N	TN	TP	FN	FP	UNKNOWN	TPR	TNR	PPV	ACC	F1	cACC	cF1
2	BT	1000	1000	992	970	30	8	25	0,97	0,992	0,99182	0,981	0,980789	0,993418	0,993344
3	EC	1000	1000	995	962	38	5	30	0,962	0,995	0,994829	0,9785	0,978139	0,993401	0,993289
4	MEAN	1000	1000	993,5	966	34	6,5	27,5	0,966	0,9935	0,993325	0,966	0,979464	0,993316	0,993316

Figure 24: Example Pred_stat.csv file for a Tool Analysis training step with Bacteroides thetaiotaomicron (BT) and Escherichia coli (EC). This file summarises the statistical in silico prediction result for each classifier or the unique average prediction (the case here). All values are explained in the main text.

Prediction.csv

The prediction file contains the number of cells predicted per record for all files selected for the prediction or the mixed community done for the ***Tool Analysis*** prediction (Figure 25).

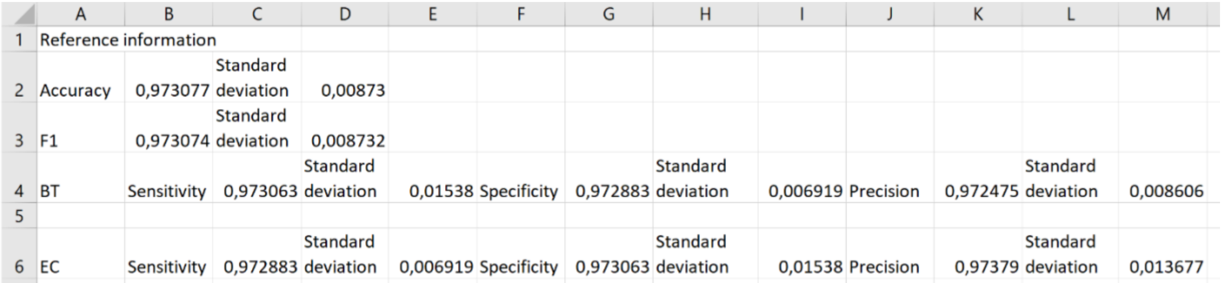


	A	B	C	D	E	F
1	BT_H08_191121					
2	AVERAGE PREDICTION FOR EACH RECORD WITH 10 REPEAT					
3	average	BT	978			
4	average	EC	967			
5	average	unknown	55			
6	average	TOTAL:	2000			

Figure 25: Example of Prediction.csv file for a Tool Analysis prediction step with Bacteroides thetaiotaomicron (BT) and Escherichia coli (EC). The file summarises the prediction by reporting the number of events per record. This result is shown for every classifier prediction or only for the average prediction, and this is for every file if the prediction concern in vitro communities.

Reference.csv

The reference file contains a summary of the learning stat and the average from the different runs (Figure 26).



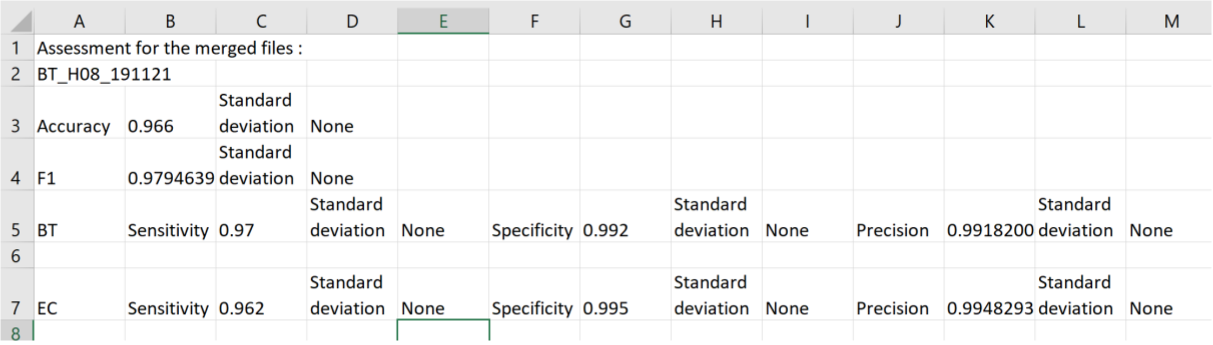
	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Reference information												
2	Accuracy	0,973077	Standard deviation	0,00873									
3	F1	0,973074	Standard deviation	0,008732									
4	BT	Sensitivity	0,973063	Standard deviation	0,01538	Specificity	0,972883	Standard deviation	0,006919	Precision	0,972475	Standard deviation	0,008606
5													
6	EC	Sensitivity	0,972883	Standard deviation	0,006919	Specificity	0,973063	Standard deviation	0,01538	Precision	0,97379	Standard deviation	0,013677

Figure 26: Example of Reference.csv file for a Tool Analysis training step with Bacteroides thetaiotaomicron (BT) and Escherichia coli (EC). This file summarises the statistical values of the ten classifier tests. All values are explained in the main text.

It contains the mean accuracy, F1 score for the community (mean value per run) and the sensitivity and precision for each species. For each statistical value, a standard deviation is calculated.

Assessement.csv

The reference file contains a summary of the pred_stat.csv file, doing the average from the different runs. This file is only created for the ***Tool Analysis*** function (Figure 27).



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	Assessment for the merged files :												
2	BT_H08_191121												
3	Accuracy	0.966	Standard deviation	None									
4	F1	0.9794639	Standard deviation	None									
5	BT	Sensitivity	0.97	Standard deviation	None	Specificity	0.992	Standard deviation	None	Precision	0.9918200	Standard deviation	None
6													
7	EC	Sensitivity	0.962	Standard deviation	None	Specificity	0.995	Standard deviation	None	Precision	0.9948293	Standard deviation	None
8													

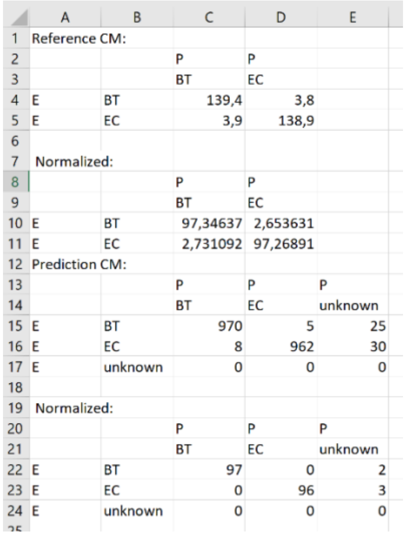
Figure 27: Example of Assessement.csv file or a Tool Analysis prediction step with Bacteroides thetaiotaomicron (BT) and Escherichia coli (EC). This file summarises the statistical values of the ten classifier predictions or the average prediction, for which no standard deviation is available since a unique value is produced. All values are explained in the main text.

If the average parameter is activated, the values from the Pred_stat.csv file are repeated. Only one prediction is created from the several runs; no mean or standard deviation can be calculated.

cmData.csv

The cmData file contains the confusion matrix values shown in the figures. It includes the confusion matrix normalised and none normalised for the test step (Reference CM) and the prediction step of the ***Tool Analysis*** function (Prediction CM).

The prediction confusion matrix does not appear for an unknown community prediction. The confusion matrix from references (test) is not on the clustering output since no training is done within these functions.



	A	B	C	D	E
1	Reference CM:				
2			P	P	
3			BT	EC	
4	E	BT	139,4	3,8	
5	E	EC	3,9	138,9	
6					
7	Normalized:				
8			P	P	
9			BT	EC	
10	E	BT	97,34637	2,653631	
11	E	EC	2,731092	97,26891	
12	Prediction CM:				
13			P	P	P
14			BT	EC	unknown
15	E	BT	970	5	25
16	E	EC	8	962	30
17	E	unknown	0	0	0
18					
19	Normalized:				
20			P	P	P
21			BT	EC	unknown
22	E	BT	97	0	2
23	E	EC	0	96	3
24	E	unknown	0	0	0

Figure 28

P: predicted

E: expected

Reference CM:

IX- Function description (developer version only)

All functions from the following files are described inside.

LEXICON:

Reference file: flow cytometric file where the composition is known and used to train the model. In the case of species, it has to be a known monoculture. One reference file must contain information about one record from the selected class.

Class: Name of a group of records used to create an analysis. By default, the program contains the class Species.

Record: element from a class which has to be identified during analysis. A record from a class species is often a bacterial species (e.g. *Escherichia coli*). == Species

Species: units of data that need to be classified together. It can be a bacterial species, a group of a species, a group of cells, or a community.