WellInverter User Manual

Yannick Martin

April 27, 2017

Contents

Licence

This software is distributed under the Creative Commons (CC) Attribution-NonCommercial licence. This software includes the Highcharts JS library which is also distributed under the same licence.

The Python package "wellFARE" allowing the computation of gene expression is distributed under the LGPL licence.

Academic use

WellInverter can be used for free for academic and educational uses.

Commercial use

Please contact authors for any commercial use of WellInverter.

Introduction

1 Purpose and scope

The use of fluorescent reporter gene technologies has become widespread in biology and has created a demand for bioinformatics tools to analyze the large amounts of data produced. Currently, few such tools are available to the life sciences community. WellInverter is a web application based on the linear inversion methods described by Zulkower and al .[3] for reconstructing growth rate, promoter activity, and protein concentration from primary fluorescence and absorbance data. This application is deployed on the cloud platform of the French Institute of Bioinformatics (IFB).

2 Compatibility

Operating System

WellInverter is compatible with any operating system that can run a recent browser. In particular, the application can run under Linux, Windows, or MacOS. See section 1.1 for instructions on how to install WellInverter.

Browser Compatibility

WellInverter is compatible with recent web browsers supporting ECMAScript 6 specifications, as shown in Table 1).

Chrome	Firefox	Edge	Safari	Opera	Internet Explorer
≥ 49	≥ 45	≥ 14	≥ 10	≥ 37	

Table 1: Browser compatibility table

Chapter 1

Getting started with WellInverter

1 Install

WellInverter is a web application hosted on the cloud of the IFB (French Institut of Bioinformatics), so there is no need to install the application on your own computer.

A personal IFB cloud account has to be created following this link:

IFB cloud registration page

`https://cloud.france-bioinformatique.fr/accounts/register/`

This account is free for users belonging to the academic community.

If you would like to test the application without registering on the IFB cloud, a demonstration version of WellInverter is available here :

WellInverter demo version

`https://wellinverter.inrialpes.fr`

This demonstration version allows you to obtain an overview of the functionalities of WellInverter, but it is feature limited. in particular, in is not possible to upload your own experimental data to test the application.

Note

WellInverter is also available as a standalone application that can run on a local computer. If you do not have access to the IFB cloud, please ask the authors to obtain this version of the program.

2 Starting a WellInverter virtual machine

First create a virtual disk in order to save your data in the cloud. You only have to create the disk once and attach the disk each time you start a virtual machine.

To create a virtual disk follow these steps:

1 - Click on New vDisk	
2 - Choose a size (in GB) and name the disk	
3 - Click on Create	
4 - Wait (10 s)	
5 - Check if the disk is created	

Each time you want to use WellInverter, a virtual machine acting as the server has to be launched with this disk attached.

To start a machine follow these steps:

1 - Click on New Instance	
2 - Choose the WellInverter appliance	
3 - Name the appliance	
4 - Choose a type of machine (for data treatment, larger machine like c2.large are prefered)	
5 - Attach the data disk	
6 - Click on Run	
8 - Check if the appliance is created	
9 - If the instance of the appliance has not started, its status is shown in blue, Click on Show Instances to refresh	
7 - Wait (30 s - 1 min)	
10 - When the instance is ready, the status indicator switches to green. Click on the http link to access WellInverter	

Important note

After any use of WellInverter, the virtual machine has to be closed to free cloud resources for other users.

3 Authentication

The WellInverter virtual machine server has two authentication methods.

- The first is to login using a default username and password. The default username is ***wellinverter*** and the default password is ***ifbcloud***.
- The recommended method is to use the Cyclone Federation (eduGAIN) authentication, which will redirect you to your academic identity provider. WellInverter will use your academic email as username.

4 A quick overview of WellInverter functionalities

WellInverter proposes a general approach based on regularized linear inversion to solve a range of estimation problems in the analysis of fluorescent and luminescent reporter gene data, notably the inference of growth rate, promoter activity, and protein concentration profiles. This approach is able to reliably reconstruct time-course profiles from weak and noisy signals at low population volumes. Moreover, it captures critical features of those profiles, notably rapid changes in gene expression during growth transitions. In addition, WellInverter also allows you to easily preprocess the data by filtering outliers and correcting background.

Chapter 2

Data preprocessing

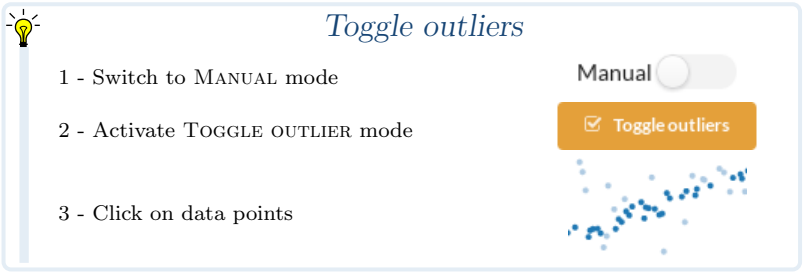
1 Outlier filtering

The data from growth experiments carried out using a microplate reader may contain outliers due to instrument failure or to specific experimental conditions (e.g., the use of glass beads to improve aeration of a growing bacterial culture). WellInverter allows you to filter outliers from the data, either automatically or manually. Manual filtering is used to remove individual data points or to select data points around an user-drawn trend curve. Automatic filtering uses a robust smoothing algorithm to identify the global trend in the data and remove outlying data points.

1.1 Manual filtering

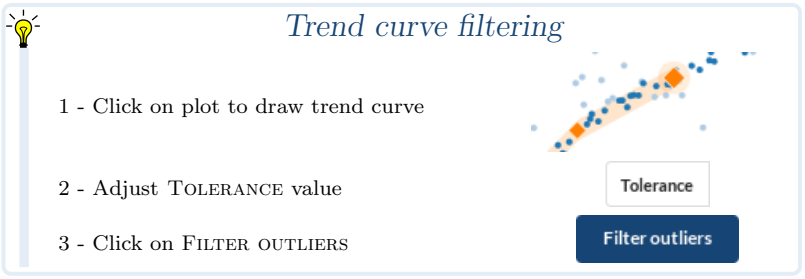
1.1.1 Removing individual data points

To label individual data points as outliers, first activate the Toggle outliers button (it becomes orange) in the top-right corner. Then simply click on data points to remove them from the data set (the point will change color, from dark blue to light blue). To reintegrate the outlier into the data set, click on the data point again (the color will change back to dark blue).



1.1.2 Selecting data along a trend curve

The second alternative is to manually draw a curve capturing the global trend in the data. After checking that the Toggle outliers button is deactivated (dark blue), click on the plot to add points to an orange curve representing the trend in the data. All data points diverging by more than the indicated distance (Tolerance field) from the trend curve are classified as outliers and removed from the data set when clicking on the Filter outliers button in the top-right panel. Point defining the trend curve can be discarded by clicking on the Reset trend curve button.

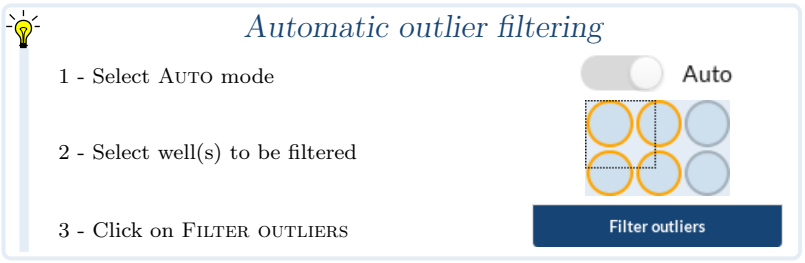


1.2 Automatic filtering

WellInverter is able to automatically filter outliers from several wells. The outlier detection algorithm uses a Savitsky-Golay smoothing algorithm and takes four parameters:

- ***Smoothing window***: Savitsky-Golay smoothing window size.
- ***Standard deviation cut-off***: All points that are further than $cut-off \times \sigma$ from the smoothed curve will be considered as outliers (σ being the standard deviation of the points computed with respect to the smoothed curve).
- ***Lower (resp. upper) penalty coefficient***: Lower (resp. upper) penalty coefficient prevents points from being removed by multiplying the cut-off value in one direction only. If the lower (resp. upper) penalty coefficient equals 3, and the cut-off equals 2, then the cut-off for points below (resp. above) the smoothed curve will be 2 and the cut-off for points above (resp. bellow) the smoothed curve will be 6.
- ***Number of iterations***: Number of times the filtering algorithm is repeated (i.e, steps 1 to 3).

These parameters can be modified in the Parameters tab and can be specified separately for each well in the Outliers filtering tab.



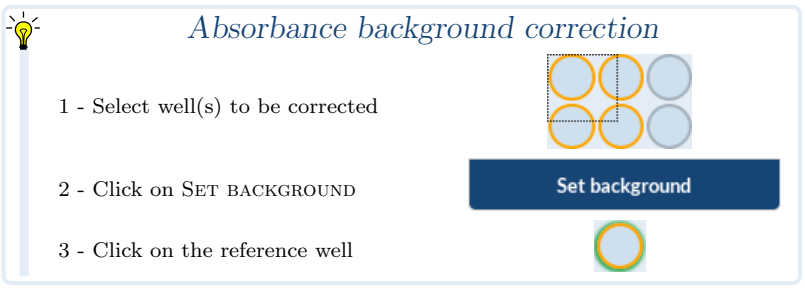
2 Background correction

WellInverter allows you to correct the background for both absorbance and fluorescence measurements. In the first case, background correction is a simple subtraction of the absorbance of a reference well (usually the growth medium without bacteria), whereas in the second case background correction subtracts the autofluorescence signal of a reference culture (often wild-type bacteria).

2.1 Absorbance background correction

Absorbance background can be corrected using a reference background well, containing only the growth medium. The correction is made by subtracting the absorbance value of the reference well from the absorbance

value of the well(s) to be corrected using a time interpolation of the reference well data. In case there is no reference well for the absorbance background, a fixed value can be subtracted. This value can be set in the PARAMETERS tab and is directly subtracted from the absorbance readings in every wells, except for wells having a defined reference well and the reference wells themselves.



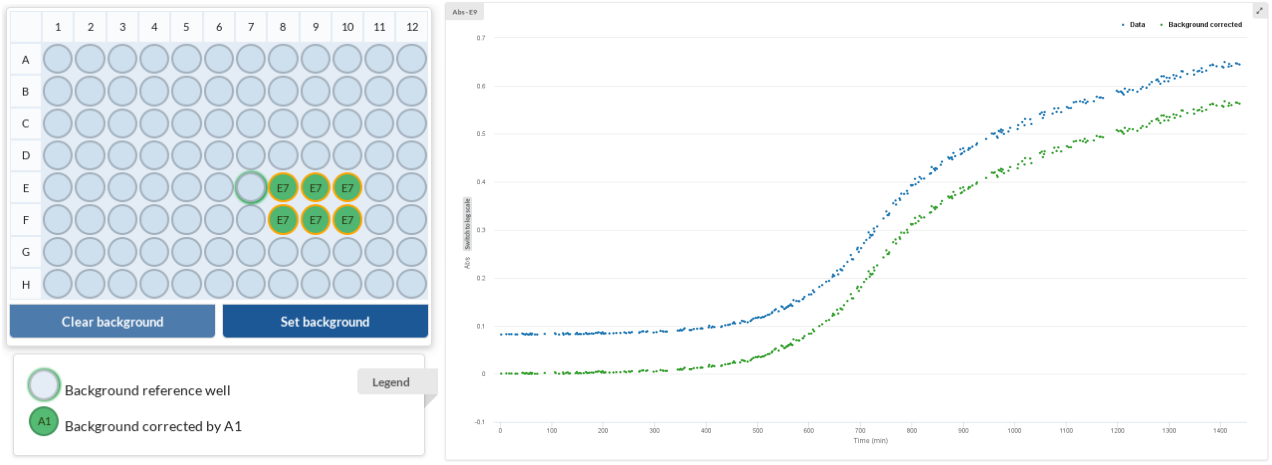
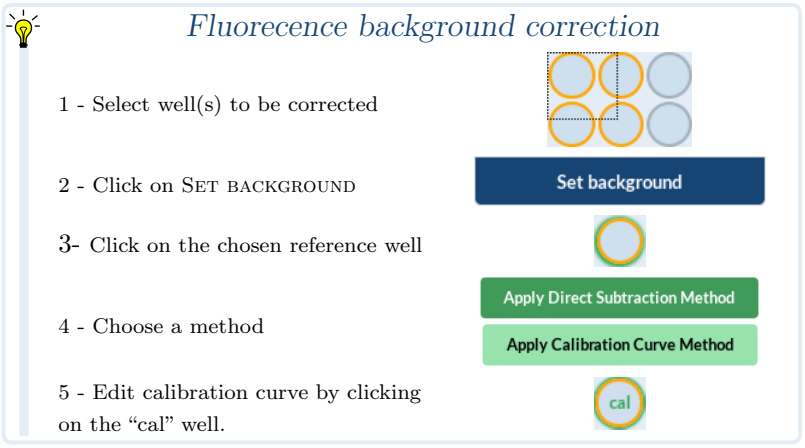


Figure 2.1: Absorbance background correction.
Left: In this example E7 is the reference well (containing only growth medium) for wells labeled in green. Right: Preview of the content of the E9 well (in blue the original data, in green the background-corrected data).

2.2 Fluorescence background correction

The aim of fluorescence background correction is to remove the contribution of autofluorescence from the measured fluorescence signal. The autofluorescence is the fluorescence emitted by a reference culture, usually wild-type bacteria. Two methods are available for correcting the fluorescence background:

- The direct method consists in the simple subtraction of the fluorescence values in the reference well from the fluorescent values in the well of interest. This approach is valid if the growth kinetics of the cultures in the reference well and the well of interest are the same.
- The calibration curve method is useful when the growth kinetics of the reference well and the well of interest are different. A calibration curve is constructed from the data in the reference well by relating the autofluorescence values to the corresponding absorbance values. The fluorescence values in the well of interest are corrected by subtracting the corresponding autofluorescence values from the calibration curve (i.e, the autofluorescence values observed at the same absorbance value).



2.2.1 Direct method

The direct method is a simple subtraction of two curves (like for absorbance background correction). The fluorescence values in the reference well are subtracted from the fluorescence values in the well of interest.

Important note

The direct subtraction approach is valid if the growth kinetics of the cultures in the reference well and the well to be corrected are the same.

2.2.2 Calibration curve method

The calibration curve method is a more robust approach than direct subtraction. The fluorescence background signal to be subtracted is determined by means of a calibration curve, relating the autofluorescence to the absorbance in a reference well. Figure 2.2 gives an explanation of the calibration curve approach. This approach is valid even if growth kinetics in the reference well is different from the growth kinetics in the well of interest.

Parameters for calibration curve. To construct the calibration curve, two parameters are defined: a smoothing window and an extrapolation distance. It is possible to change these parameters by selecting the reference well. The MIN and MAX vertical dashed line can be moved so that points outside $[MIN, MAX]$ interval are ignored in the construction of the calibration curve. If the measured absorbance value lies outside the calibration curve absorbance limits plus the extrapolation distance, the autofluorescence values are set to the closest value at the lower or upper bound.

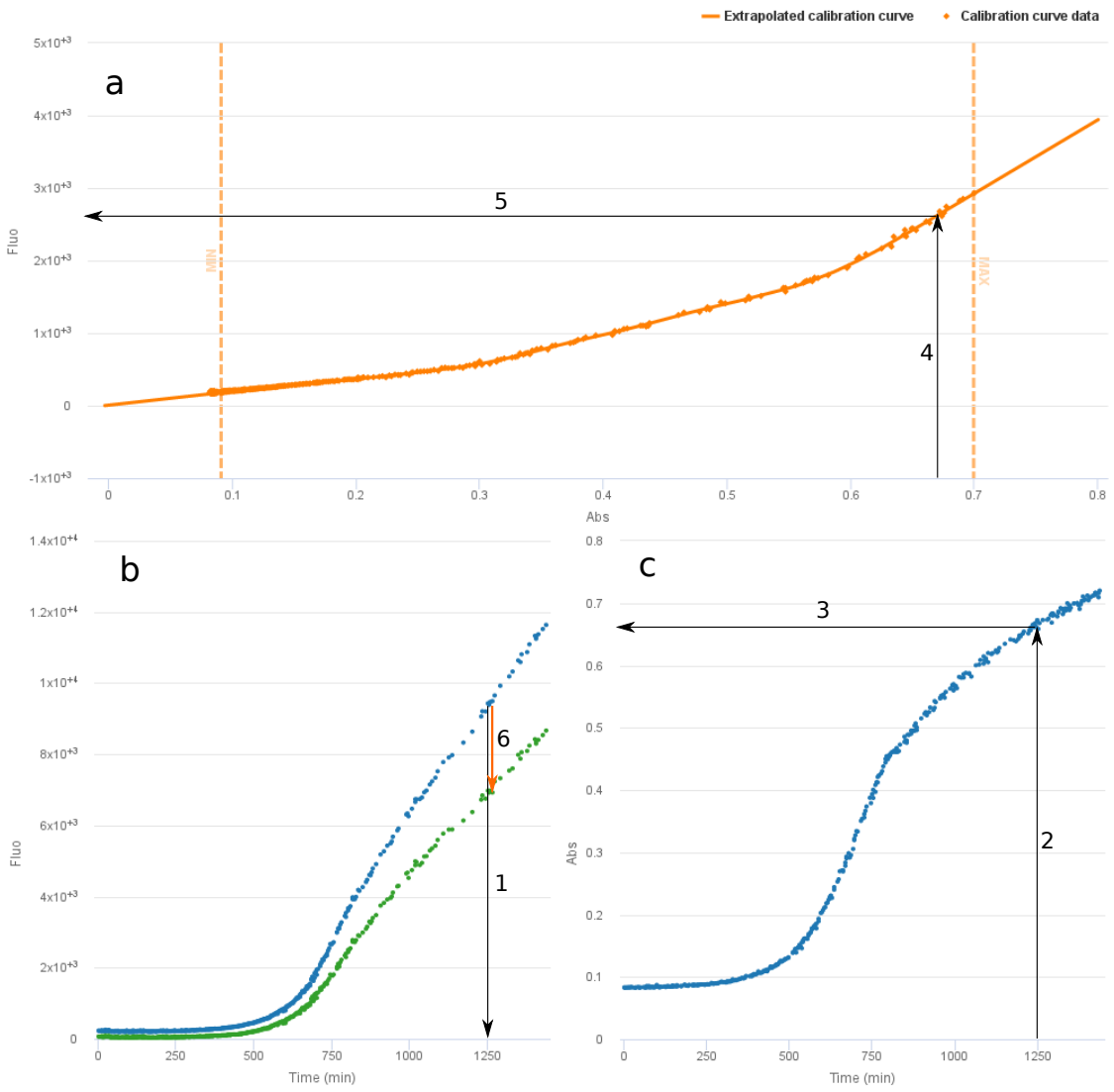


Figure 2.2: Calibration curve approach for fluorescence background correction.
a - Calibration curve of the reference well for fluorescence background correction.
b - Fluorescence of the well of interest (in green the background-corrected fluorescence values).
c - Absorbance of the well of interest.
A given fluorescence data point (1) corresponds to an absorbance data point in the same well (2-3). This absorbance value is associated with an autofluorescence value in the calibration curve (4-5). The autofluorescence value is subtracted from the fluorescence signal (6) to yield the corrected fluorescence.

Chapter 3

Reconstruction of growth rate, promoter activity, and protein concentration

1 Method

The reconstruction of quantities of biological interest, notably the growth rate of a culture, the activity of the promoter of the reporter gene, and the concentration of protein products, is based on the one-step model described in Zulkower and al. [3]. Please refer to this paper for a detailed presentation of the method. This user manual focuses on explaining the influence of parameters in the reconstruction of the quantities of interests.

in the PARAMETERS tab, it is possible to change the model parameters and the inference method parameters. Model parameters are parameters describing the system, while inference method parameters change the way in which the reconstruction algorithm is applied.

Important note

Do not forget to set the appropriate parameters of the reconstruction algorithm in the PARAMETERS tab.

1.1 Model parameters

For fluorescent measurements, the reporter and protein degradation rates can be set. For absorbance measurements, it is possible to force a positive solution for the growth rate.

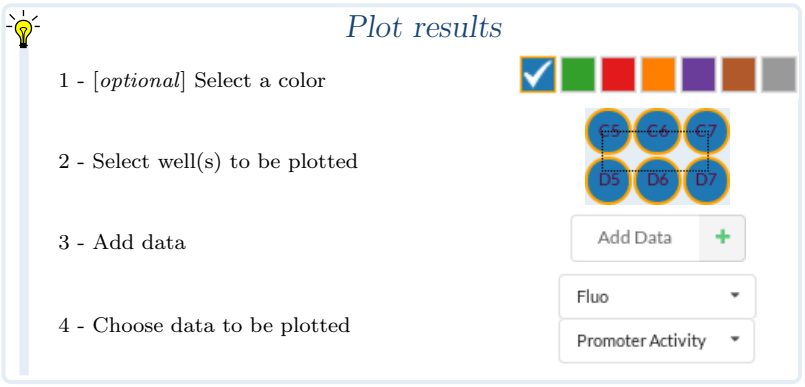
1.2 Inference method parameters

- ***Regularization parameter*** *(min - max - N)*: The regularization parameter is chosen among a set of N values equally spaced (in log scale) in the interval $\left[10^{min}, 10^{max}\right]$. This choice defines the use of Generalized Cross-Validation.
- ***Epsilon_L***: Negligible factor for the derivation matrix of the Generalized Cross-Validation method. When the value of this parameter is not carefully chosen, it introduces an unwanted penalty on the initial values of the quantity to estimate.
- ***Control time points***: Number of control points as a percentage of the number of time points in the data curve.

2 Plotting results

To plot a result, use the PLOTS tab. Select one or several wells, which thus take the color checked in the upper color bar (you can change this color before selecting the wells). Once you have selected the wells you want to plot, add data by clicking on the ADD DATA button. Select the measure (absorbance, fluorescence) and choose the quantities of interest:

- **Primary**: Raw data contained in the file produced by the microplate reader.
- **Outlier-filtered**: Data from which outliers (as defined in the OUTLIER tab) have been removed.
- **Background-corrected**: Data from which the background has been subtracted (as defined in the BACKGROUND CORRECTION tab).
- **Growth rate** [*Absorbance only*]: Growth rate of the culture computed from the absorbance data.
- **Reporter concentration** [*Fluorescence only*]: Reporter concentration computed from the fluorescence and absorbance data.
- **Promoter activity** [*Fluorescence only*]: Promoter activity computed from the fluorescence and absorbance data.
- **Protein concentration** [*Fluorescence only*]: Protein concentration computed from the fluorescence and absorbance data.



When the result has been plotted, it is possible to

- Switch from linear scale to log scale by clicking on the corresponding axis title button.
- Zoom in by selecting an area on the plot (zoom out by clicking Reset zoom).
- Get information on data by clicking on a point.
- Export the plot as an image by clicking on the top-right menu icon.
- Go full-screen by clicking on the top right icon.

By default, if two or more wells have the same color, the mean and the standard deviation of the quantity computed over the wells are plotted. It is possible to change this default using the plot options.

2.1 Plot options

For each color defined in the microplate, it is possible to

- Hide it on the plot chart (toggle the eye symbol).

- Plot all the wells as individual curves (plot all).
- [*two or more wells*] Plot the mean value of these wells (plot mean).
- [*two or more wells*] Plot the standard deviation (plot SD).
- [*two or more wells*] Plot the standard error of the mean (plot SEM).

When two or more colors are selected, the mean, standard deviation, and standard error of the mean of all colored wells can be plotted by clicking on the last row.

If the plot takes a very long time to refresh, because of the large number of selected data points, activate the Quick Plot option which limits the number of points per curve to 200. This option uses the Largest-Triangle-Three-Buckets algorithm [2], preserving the visual characteristics of the original curve.

2.2 Grouping wells

For convenience, it is possible to group wells by clicking on the Add Group button under the Well groups tab. When hovering over the microplate, a tooltip will appear if the well belongs to a group. When clicking on this well, all wells in the group are colored.

To by-pass group selection, hold the Shift key while hovering over the microplate.

3 Exporting the results of the analysis

The export of the analysis results is possible by downloading a JSON file that can be read by most common data analysis tools (Python, MATLAB, R ...). The Export tab works exactly like the plot tab, except that instead of plotting curves, it generates a JSON file with the following structure:

- Measure 1 (ex. Fluo)
  - Data 1 (ex. Primary)
    - Selected well 1 (ex. E1)
      - time : [array]
      - signal : [array]
    - Group of well 1 (ex. E3, E4)
      - Mean
        - time : [array]
        - signal : [array]
      - SD
        - time : [array]
        - signal : [array]
    - ...
  - Data 2 (ex. Growth Rate)
    - ...
  - ...
- Measure 2 (ex. Abs)
  - Data 1 (ex. Background-corrected)
    - ...
  - Data 2 (ex. Promoter Activity)
    - ...
  - ...

The checkbox Export all data can also be selected. This will lead to the computation and export of all the following quantities for the selected wells: Primary, Outlier-filtered, Background-corrected, Growth rate, Reporter concentration, Promoter activity, Protein concentration. Depending on the number of selected wells, the number of data points, and the inference parameters, it may take some time to compute these quantities for the whole dataset.

Appendices

Adding new grammars to parse experimental data

WellInverter uses a grammar-based parser to analyze the data contained in the files produced by the microplate reader. The parser in based on Antlr4 [1].

To create a new parser, several steps must be followed. First provide a name for your grammar (here "MyGrammar").

1 - Registering a new plugin

Available plugins are stored in pythonserver/plugins/parsing. To create a new plugin using Antlr for parsing, copy "ParserAntlr.py.template" and name the new file "MyGrammarAntlr.py"

Then edit the contents of the file to enter the name and the description of your parser and its data file extensions:

```
GRAMMAR_NAME = "MyGrammar"
PARSER_TIMEOUT = 120

def pluginfo():
	return {"name":"MY GRAMMAR",
			"description":"Here is a short description of MyGrammar parser.",
			"filetypes":".xls,.xlsx"}
```

You can set a different timeout if your parsing task may take more than 120 seconds.

2 - Creating a grammar

This is the most time-consuming step of defining a new parser. You have to create a grammar file that can be interpreted by Antlr and transformed into a Java parser.

Create a file named "MyGrammar.g" in the following directory:

pythonserver/plugins/parsing/antlr4Parser/grammars/MyGrammar/

The definition of a grammar file is further explained on the Antlr4 website.

`http://www.antlr.org`

You may want to have a look at other grammar files used by WellInverter when defining your own grammar file. In the grammar file, Java code must be inserted in order to process the input and export it into the JSON WellInverter format. The Java class WIExperiment will help in doing this:

```
@header {
	import wiantlrparser.WIExperiment;
}

@members {
	WIExperiment wiExperiment = new WIExperiment();
}
```

Here is an API overview of the WIExperiment class:

```
/** setTimeFormat
* Sets the time format used inside the experiment file.
* timeFormat can be "microseconds", "milliseconds", "seconds", "minutes", "hours", "days"
* or a date format as defined by SimpleDateFormat (e.g. "hh:mm:ss").
*/
wiExperiment.setTimeFormat(String timeFormat);

/** toMinutes
* Converts a string time to minutes using timeFormat.
*/
wiExperiment.toMinutes(String value)

/** setStartTime
* Sets the starting time of the experiment.
* If subtractToValues is true, this time will be subtracted to all further values in the dataset :
* If startTime is a string, it is converted using toMinutes before being subtracted.
*/
setStartTime(String startTime)
setStartTime(String startTime, Boolean subtractToValues)
setStartTime(Double startTime, Boolean subtractToValues)

/** setPlateWidth
* Useful when the names of the wells are integers.
* Will allow to convert well index to well name.
*/
wiExperiment.setPlateWidth(Integer width)

/** wellIndexToName
* Convert a well index to a well name
* setPlateWidth must have been called before.
* First n = plateWidth wells will be named A1 - An
* Then next n wells B1 - Bn ...
*/
wiExperiment.wellIndexToName(Integer index)

/** addMeasure
* Adds a measure with its name (e.g. "Fluo", "Abs" ...) */
wiExperiment.addMeasure(String measureName,Integer measureType)

/** setCurrentMeasure
* Sets the current measure name.
* Call is necessary before adding values to the dataset.
* The currentMeasure must be one of the measure defined when calling addMeasure.
*/
wiExperiment.setCurrentMeasure(String currentMeasure)

/** setCurrentWell
* Sets the current well name.
* If the currentWell looks like an index, like "1" "2" ...,
* it will convert it to a "A1" styled name calling wellIndexToName.
* Call is necessary before adding values to the dataset.
*/
wiExperiment.setCurrentWell(String currentWell) -> can be int, in this case wellIndexToName is
    called before setting the current well.

/** addValue
```

```
* Adds a couple (time,value), i.e. a datapoint, to the dataset.
* setCurrentWell and setCurrentMeasure must have been called before.
* If time is a String, it is converted to time using
*/
wiExperiment.addValue(String time, String value)
wiExperiment.addValue(Double time, String value)

/** addValue, addTime
* If you can't store your couple (time,value) you can add time and value separately
* addValue and addTime must be called the same number of time for each couple
    (currentMeasure,currentWell).
* Datapoints for each well will be (1st time, 1st value), (2nd time, 2nd value) ...
* If time is a String, it is converted to time using
*/
wiExperiment.addValue(String value)
wiExperiment.addTime(String time)
wiExperiment.addTime(Double time)

/** addAction
* Adds a general parameter in the information tab of WellInverter.
*/
wiExperiment.addAction(String actionName,String actionValue)

/** addProgram
* Adds a parameter linked to a measure in the information tab of WellInverter.
* The measure can be defines as :
* - string (measure name)
* - int (measure index, in the same order as calls to addMeasure)
* - by using addProgramToNextMeasure, that will add the parameter to the next
* measure added by calling addMeasure.
*/
addProgram(String measureName, String programName, String programValue)
addProgram(Integer measureIndex, String programName, String programValue)
addProgramToNextMeasure(String programName, String programValue)

/** writeJson
* At the end of the parsing, WIExperiment should write the JSON file.
* Do not output any text in the standard output as WellInverter application
* uses this output to read the saved data filename.
*/
wiExperiment.writeJson();
```

3 - Compiling the grammar

MyGrammar.g should now be compiled to produce Java code that can be called by WellInverter. To do this open a command line and call:

WINDOWS

```
java -cp ".;../../WIAntlrParser.jar;../../lib/antlr-4.6-complete.jar;$CLASSPATH" org.antlr.v4.Tool
    MyGrammar.g
javac -cp ".;../../WIAntlrParser.jar;../../lib/antlr-4.6-complete.jar;$CLASSPATH" *.java
```

UNIX

```
java -cp ".:../../WIAntlrParser.jar:../../lib/antlr-4.6-complete.jar:$CLASSPATH" org.antlr.v4.Tool
    MyGrammar.g
javac -cp ".:../../WIAntlrParser.jar:../../lib/antlr-4.6-complete.jar:$CLASSPATH" *.java
```

4 - Testing the grammar

You now have to restart the WellInverter server and after opening the browser you should see that a new parser appears. You should now be able to upload and parse the data in the desired format!

Local installation of WellInverter

WellInverter can also be deployed locally instead of being accessed through the French Institute of Bioinformatics (IFB) cloud. Please ask the developers to obtain the program files.

> 📎 *WellInverter*
>
> `https://team.inria.fr/ibis/wellinverter/`

Operating system

WellInverter is compatible with any operating system that can run Python 3.5, Java and Javac.

Automated installation scripts are available for Windows and Linux.

Java

WellInverter uses a Java library called Antlr to parse experiment files. Therefore the computer running the WellInverter server needs to have Java and Java JDK installed to compile and run the parser of the data files.

Install Java and Java JDK on your computer. java and javac binaries must be in your environment path.

Python 3.5

WellInverter needs Python 3.5 to run the server and analyze the data (using the wellFARE package). The simplest solution to install a Python distribution is to install anaconda.

> 📎 *Download Python 3.5 distribution*
>
> `https://www.continuum.io/downloads`

If you already have an Anaconda python version, it is recommended to create a virtual environment for WellInverter (to avoid any package version conflict).

```
conda create -n wellinverterenv python=x.x anaconda
source activate wellinverterenv
```

After creating a virtual environment, install the following packages:

```
conda install numpy -y
conda install scipy -y
conda install decorator -y
conda install pyzmq -y
conda install psutil -y
conda install tornado -y
conda install bcrypt -y
conda install docopt -y
conda install xlrd -y
conda install colorama -y
```

Then go to both the wellFARE and pyLoadBalancer directories and, in each of these, run the following command:

```
python3 setup.py install
```

To create a default local account for WellInverter, navigate to WellInverter/pythonserver and run

```
python3 manageusers.py add username -p password
```

To run WellInverter, navigate to WellInverter/pythonserver and run

```
python3 startWellInverter.py
```

WellInverter should now be started and can be accessed by browsing to http://localhost:8000

WellInverter local access

```
http://localhost:8000
```

Bibliography

[1] Terence Parr. *The Definitive ANTLR 4 Reference*. Pragmatic Bookshelf, 2nd edition, 2013.

[2] Sveinn Steinarsson. Downsampling time series for visual representation. Master's thesis, University of Iceland, 1979.

[3] Valentin Zulkower, Michel Page, Delphine Ropers, Johannes Geiselmann, and Hidde De Jong. Robust reconstruction of gene expression profiles from reporter gene data using linear inversion. *Bioinformatics*, 31(12):i71–i79, 2015.