User manual of the Coincidence data analysis software (ANACONDA 2)

Bart Oostenrijk, Benjamin Bolling, Lisa Rämisch

October 22, 2018

Contents

1 Introduction

This document shows how the data analysis package 'ANACONDA 2', or 'ANACONDA' can be installed and used, with the focus on new users. ANACONDA allows the analysis of data recorded in experiments that use single-particle detectors (e.g. ions, electrons, photons ...), for example to study the correlation between these particles. These are often called 'coincidence' spectrocopic treatment methods.

2 Installation

End user installation:

- Download (or 'clone') the file 'ANACONDA.mltbx'
- drag and drop the file into the matlab (command) window, and click install in the pop-up window.
- A gettingstarted (example) file can be found in the '/doc/' directory.

Developer installation:

- copy the 'package' folder (included in the repository) to your local system.
- add this folder to the path (only the folder, not the subfolders), either by the command:
  - addpath(fullfile('path','to','folder', ..., 'package'))
  - or do this by right-clicking on the folder in the file browser in MATLAB, and click 'add to path'.

you are now ready to use all functions within the package.

Use the GUI (under development) by writing 'GUI.main' in your command window.

3 Getting started

The matlab toolbox comes with an interactive script (*/package/doc/GettingStarted.mlx*), or a plain matlab script (*/package/doc/GettingStarted_plain.m*) that shows a few simple commands and minimal working examples of functions and macros.

Load first data After installing the package, one can start importing the data. There are several data formats supported by the package. Depending on which format you want to treat, read the one of the following functions by typing into the MATLAB command window:

```
>> edit IO.COBOLD.import_example ;% importing ASCII delimited
    datafiles from COBOLD PC. If you do not have the appropriate
    ASCII-delimited file format, contact Roentdek.
>> edit IO.DLT2ANA.import_example ;% importing the DLT data format
    from the Labview data acquisition software (by Erik Mansson, Lund
    University)
>> edit IO.EPICEA.import_example ;% Importing ASCII delimited
    datafiles from EPICEA. If you do not have the appropriate
    ASCII-delimited file format, contact the EPICEA (or PLEIADES
    beamline) experts.
```

At the end of importing the data, it is advised to save the data to a '.mat' (MATLAB) format. For example, in the case of the COBOLD import, one could write:

```
data = IO.COBOLD.import_example('file/to/path', 'filename'); %
    import the data
IO.save_exp(data, 'file/to/path', 'filename'); & save it to a binary
    '.mat' file
```

Create first metadata Metadata is literally 'data about the data', and the package needs to know more about this data to start the data treatment. For example, we need to know what this 'raw' data contains. This kind of information can be stored in a separate 'metadata' file. Each datafile must contain such a file (habitually in the same folder as the data, with the same name except the prefix 'md_' and extension '.m'). So, create a metadata called 'md_filename.m' with the file to the datafile that is called 'filename.mat'. This file 'md_filename.m' could look like this:

```
% Metadata file for 'filename'.
%% Defaults
% The default values are loaded from the package:
exp_md = metadata.defaults.exp.CIEL.md_all_defaults();
%% Custom values
% Here, we can customise these default values.
% For instance, the mass-2-charge conversion factor:
exp_md.conv.det1.m2q.factor = 2500;
```

The metadata file is read as a matlab script when it is read, so all MATLAB functionalities are available. The name of the metadata struct has to be 'exp_md'.

4 Metadata

The metadata contains all the information *about* the data, such as conversion parameters, but also filter and plot parameters. The package comes shipped with 'default' parameters, which can be used when the data is first imported by the user. We have seen in the previous section how to start a first metadata file from those defaults. This section explains the different fields in the metadata, and what they can be used for. The metadata can be divided into different categories:

sample , e.g. atomic mass, expected fragment masses, constituent masses.

photon beam , e.g. the photon energy, intensity, duration, etc

spectrometer , e.g. the name, voltages and relevant dimensions of the used spectrometer are listed here.

detectors , e.g. the names and properties of the detectors are stored in here.

correct The parameters needed to execute corrections onto the raw data, before conversion. For example, translation in X and Y to move the centre of detection into the origin of the coordinate system.

calibrate The information needed to perform the calibrations. Note that these are not the actual calibration factors, they are stored in the 'convert' field.

fit The fitting parameters.

convert The conversion factors (sorted in terms of detectors), such as mass to charge conversion.

plot The user-preferred plotstyle.

We will go through these categories and clarify the fields used in them.
TODO

```
exp1_md.sample
exp1_md.photon
exp1_md.spec
exp1_md.corr
exp1_md.calib
exp1_md.fit
exp1_md.conv
exp1_md.plot
```