PyQuan

Mark M. Smits

February 17, 2016

Contents

1 Introduction

PyQuan is software developed to automate the analysis of GC/MS data. The purpose is to automate the most labour intensive parts of the analysis, while keeping control of the whole process. This is done by creating images in the alignment, quantification and normalization stages. My philosophy is that even the most sophisticated automatated quality control cannot beat an expert view on visually presented data.

PyQuan is designed around the freely available AMDIS package `http://chemdata.nist.gov/dokuwiki/doku.php?=chemdata:amdis`. AMDIS is used to identify the peaks, while PyQuan (1) checks the identification based on a sophisticated alignment algorithm and a reference library, (2) quantify the checked peaks and (3) normalize the peaks to either the total ion current, the sum of identified peaks, or (an) internal standard(s). Additionally, it is possible to 'backfill peaks that are missed by AMDIS (false negatives), and include them in the quantification.

Please, when using this software for scientific purposes, refer to:
Smits MM, Carleer R, Colpaert JV (2016) PYQUAN: A rapid workflow around the AMDIS deconvolution software for high throughput analysis of pyrolysis GC/MS data. Journal of Analytical and Applied Pyrolysis. doi: 10.1016/j.jaap.2016.01.006.

Feel free to contact the developer (Mark Smits: lycopoda@gmail.com) for any questions or remarks.

2 Setting up the software

2.1 Software installation

Installl the AMDIS software `http://chemdata.nist.gov/mass-spcs/amdis/downloads/` . This version of PyQuanruns on Python 3. If you don't have already python on your computer, I recommend to install Anaconda (`https://store.continuum.io/cshop/anaconda/`). Anaconda is a python distribution that includes the right packages to run PyQuan, except netCDF4. Install netCDF4 by typing on the command line: conda netcdf4. Miniconda is a low weight alternative, but here you have to install several packages: scipy, numpy, matplotlib and netCDF4.

2.2 Command line window

PyQuan runs from a command line. By creating a 'command prompt window', linked to the script folder, you simplifies the use of the program. In MS Windows systems, right-click in desktop and select *New → Shortcut*. Type *%comspec%* in the location bar, press enter, and select *next*. In the window you change the name of your shortcut (e.g. 'Pyquan'). Now, select the shortcut in the desktop. Press the *right mouse button* and select *Properties* to configure the shortcut. Select *shortcut tab*. Change in *start in option*, type the location of the PyQuan code folder. Next, select the *option tab* and check the boxes: *QuickEdit mode*, *Insert mode* and *Discard Old Duplicates*.

2.3 AMDIS library

Next step is to create or move your AMDIS library in .msl format to the library folder. With AMDIS you can manage your library. From e.g. the NIST database, you import new spectra into the library. NB: each compound should have a code that could be recognized by PyQuan. In AMDIS, edit the compound information, and start the name with your code, followed by ':'. For example:
004a: 1-Propene, 2-methoxy-. The code itself can have any character except '?' and ':'.

2.4 RT library

Now, you create your retention time (RT) library, by running:
python update_library.py <amdis library> <RT library>
This creates a csv file with per row:
Code, RT, Limit, m/z, Name, Source.
To create a RT library, you run the script *update_library.py <amdis library> <RT library>*. If you do not specify the amdis library and the name of your RT library, it will take that information from your *pyquan.ini* file. This script can also be used to sync your RT library with your AMDIS library. It will add new compounds to your RT library. When created, you need manually add the RT information with

e.g. Excel or a simple text editor. Don't worry if Excel removes the first zero's from a number. PyQuan can still link those codes to the right AMDIS code.

The reference *RT* values can be any linear scale relative to the expected real RT values. Kovat's or Lee's retention index are not linear! If you use different GC temperature programs or switch to a different type of column, you should create a reference library for each setup. *Tip:* when changing to a new setup, first run the analyse with the previous library. The normalization against the reference graph show clearly where your previous library deviates from the new conditions. In the *pyquan.ini* file you can indicate which RT library you use for your curent analysis.

Limit denotes the valid RT range for that compound. Slight variation in machine settings or the condition of the GC column can cause deviations in RT. In the actual analyse, after the alignment procedure, the expected RT's are fine-tuned, and the valid RT range is reduced, as variation in RT is much less within one run, than between runs.

In the *m/z* column, you give for each compound the mass-to-charge ratio that you want to use to quantify that compound. You either choose to use one m/z value, or multiple (separate values by '+). Based on the total spectrum, the area of the selected m/z value(s) are calculated to total ion current for each compound.

Automated *m/z* estimation and *lim* will be implemented in a future version of *update_library.py*.

In the *name* column, you find the name for each compound, if speficied in the amdis library. If not present, you can add the name in this column. If your name contains comma's, Excel automatically place "" around the name (although they are not visible in Excel). *If you edit your library in a simple text editor, you have to do it manually.*

Optionally, you can give the likely *source* of the compound if the last column.

2.5 Folder structure

- Code:
  Contains the software code.

- Library:
  Here, your AMDIS library and RT libraries should be stored.

- Init:
  The initialization files can be found here. They are used to control analysis parameters.

- Data:
  GC/MS data files, in CDF format, are stored here.

- Projects:
  Here, you start a new project by storing the AMDIS output file. During the analysis, a folder named after the project is created. In this folder, the output data will be stored. It will contain the following subfolders:

  – Alignment:
    This folder contains images of the alignment process, for visual control. It also contains for each compound the RT, both the actual and converted to the reference scale, and the fit with the mass spectrum in the AMDIS library.

  – AMDIS:
    The AMDIS output file is separated in invidual sample files, and stored here.

  – Data:
    Details of each quantified peak (area, fitten curve, etc.) are stored, per sample, in this folder as a data file.

  – Images:
    Each quantified peak is stored as an image, showing the choosen m/z intensity for quantification and the fitted peak. In the case of overlapping peaks, neighbouring peaks are plotted as well. The image files are stored as <code> <sample > .png. In this way, it is easy to check the peaks for each code sequentely.

  – Normalization:
    Contains the TIC images of each sample, together with the calculated baseline and all quantified peaks. This helps to quickly check for peaks that are not or wrongly identified.

3 Start a new project

The analysis of a series of samples involves 3 steps:

1. Run AMDIS for peak identification.
2. Choose the preferred parameters for analysis.
3. Run the project in PyQuan.

3.1 Run AMDIS for peak identification

Make sure, you have stored your data in .CDF format (ANDI-MS) in the data folder. Open AMDIS. Make sure that AMDIS uses the AMDIS library, stored in the library folder. Create a batch job for all samples in your project.

- Set *analysis type* to ‘simple’
- Select Generate report’, and ‘Report all hits’
- Save batch job as *<projectname> _AMDIS.job*

Run the batch job. This will store an output file in the project folder, named *<project name>_AMDIS.data*.

3.2 Choose the preferred parameters for analysis

Parameters can be changed in the *Pyquan.ini* file, which can be opened in any text editor. Here, you can change the RT library name, AMDIS library name, and parameter values for alignment, normalization and peak fitting. Also the preferred separator for CSV files can be changed (generally ‘,’ or ‘;’. If you want to restore the ini file to the initial settings, you can run:
default.py pyquan

4 Analyse the results

The analysis is started by writing on the command line:
python analyse.py <project name>
This command will first create a *runlist* and the folder structure in the project folder. Automatically follow the calibration step, quantification step and the normalization step. Theses steps can be run separately via the commands:
python calibrate.py <project name> python pyquan.py <project name> python normalize.py <project name>

My recommended workflow is:

- First run the total analysis.
- Then check the peak identification and quantification (see below).
- If needed, run the quantification for certain codes or individual peaks again (see section *Peak correction*).

- Rerun the normalization step and check for missing peaks in the normalization results (see below).
- If needed go back to AMDIS to extract the mass spectrum of peaks missing from the quantification, and compare with e.g. the NIST database or literature. Even if identification fails, the unknown compound can still be analysed. Just add the mass spectrum to the AMDIS library. Update the RT library (see section *RT library.* Rerun AMDIS, and quantify this compound (see section *Peak correction*).
- When the quantification step was satisfactory, run the normalization step again. Check again for missing peaks. If needed repeat the previous step.

4.1 runlist and project folders

When a project is run for the first time, a runlist is created in the project folder. This is a csv-file with two columns: the sample name and a second column that contain an ‘X. The ‘X indicates which sample is included in the analysis. By removing the ‘X in the runlist file you can easily (temporarily) exclude specific samples from the analysis.

The software automatically moves the AMDIS batch output file into the project AMDIS folder. The file is divided in output files for each sample.

4.2 Calibration step

The calibration step aligns each sample against each other and against the RT reference library. The alignment can be visually checked from a series of .PNG images stored in the alignment folder (Figure 1). Compound RT's (both the actual in minutes and normalized to the RT library) and the fit with the mass spectrum (extracted from the AMDIS output files) are stored in a spreadsheet in .CSV format.

4.3 Quantification step

At the quantification stage, each identified peak that fits within the limits of the expected RT of that compound (based on the RT reference library) is quantified. The default approach is to fit an exponential-Gaussian hybrid function over the intensity data of selected m/z value(s). If peaks overlap, a process called deconvolution is used. In that case the neighbouring peak(s) are included in the fit as well, to untangle the different peaks.

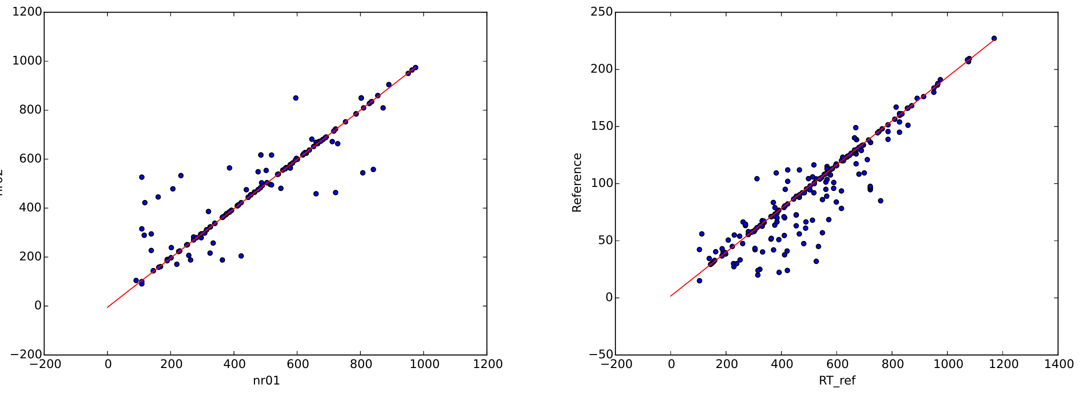


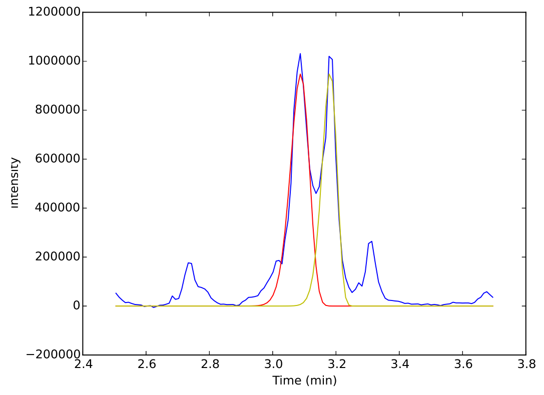
Figure 1: Alignment images. Left: the dots are compounds, identified in both samples by AMDIS. The red line is the best fit through most data points. It represents the aligment factor between both samples. Right: Here, the dots represent compounds both present in the samples and in the RT reference library. With on the X-axis the median of the RT in the aligned samples, and on the Y-axis the RT in the reference library. The red line represents the correction factor with the RT reference library.

If, in the rare case that an exponential-Gaussian hybrid function cannot fit through the data, the actual surface under the intensity peak is measured. If selected in *pyquan.ini* all peaks are quantified in this way (set *change_fit_peak* to 'no'). But with this approach deconvolution is not possible. I would recomment to use the curve fit approach (which is the default).

For each peak that is quantified, an image file is stored in the *image* folder (Fig. 2). The naming of the images (<*code*>_<*sample*>*.png*) allows sorting of the images per compound code, and makes it convenient to quickly scan the quantified peaks code by code.
All quantification data is stored in data files for each sample in the *data* folder.

4.4 Normalization step

Peak areas can be normalized to (i) the sum of the TIC's of each quantified compound (ii) the sum of TIC's over the whole time range minus the baseline (iii) one or more internal standard(s) or any compound you like . The sum of all quantified peaks can be found, superimposed on the TIC, in an overview image (<*sample*



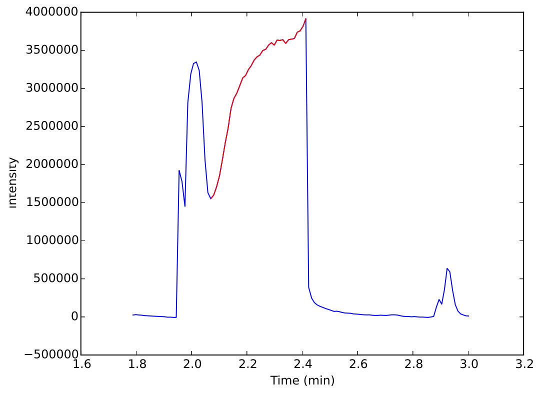


Figure 2: Left: peak quantified with an exponential-Gaussian hybrid function. In red the peak of interest, in yellow adjacent convoluting peaks. Right: a peak where fitting a curve didn't work, area is measured under the actual intensity data.

name>_complete.png) and in 1/10 time slides as well. These images reveal which peaks are not quantified yet (see Fig. 3).

After the normalization step, output files can be found in the project folder. *<project name>_area_norm.csv* contains the normalized area data. If you are interested in the non-normalized area data: they can be found in the project data folder, together with the AMDIS identification fit, RT time and the RT converted to the refence scale.

5 Peak correction

If a peak is not correctly quantified, the fitting parameters or estimated RT can be adjusted in an init file (*check_peak.ini*). The peak can be quantified again with the code:

python check_peak.py

In the ini file, which can be edited with any text editor, you can choose which compound in which sample should be re-quantified. If you fill in 'all' as sample name, the compound is re-quantified in all samples (except the ones that are deselected in the runlist file). You can also choose to quantify the actual surface under the intensity peak, instead of the area under the fitted curve.

Peaks can also be removed. Write on the command line: *python remove.py*

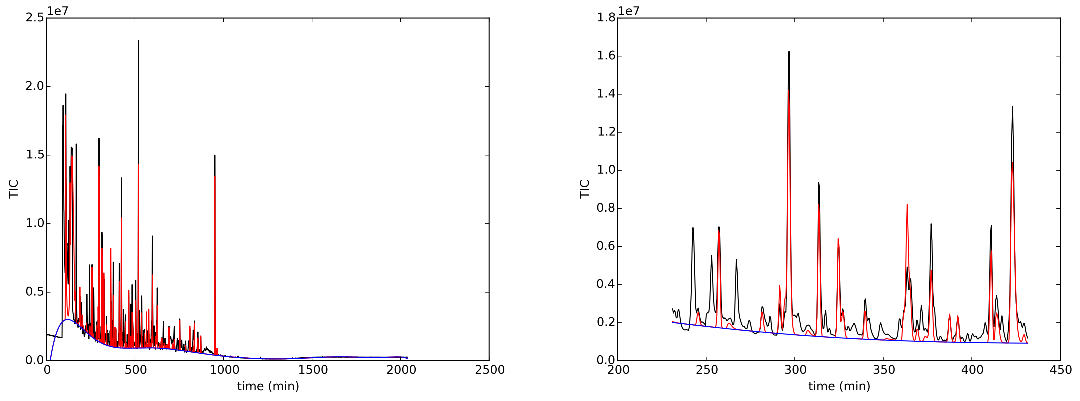


Figure 3: Normalization images. The TIC in black, the sum of quantified peaks in red, and the baseline in blue. Left: complete time range. Right: 1/10 time range.

<*project name*> < *sample name*> <*code*>

If you fill in 'all' instead of a sample name. The compound will be removed from all samples in the runlist.

6 Problems

If you experience any problems, please send an e-mail to lycopoda@gmail.com