What is SPIKE ?

SPIKE a collaborative development for a FT-spectroscopy processing program.

This is the version 0.99.15 - January 2020

SPIKE is a program that allows the processing, the display and the analysis of data-sets obtained from various Fourier-Transform spectroscopies. The name stands for **S**pectrometry **P**rocessing **I**nnovative **KE**rnel.

It allows the processing of **1D** and **2D** FT spectroscopies, implementing Real, Complex and HyperComplex n-dimensionnal Fourier Transform, as well as many other functionalities.

It is written in python (tested in python 3.7 and in python 2.7 up to version 0.99.10) and can be used as a set of tools, using for instance `jupyter notebook` as an interactive front-end.

To our knowledge, it is the first program freely available allowing the processing, display and analysis of 2D-FT-ICR (Fourier Transform Ion Cyclotron Resonance), as well as **Orbitrap** time domain data. processing.

It is still in very active development. Many features are missing, and many other while present, are not fully fixed. However, considering the amount of efforts already present in this code, we decided to make it available. We believe that even in this partial development stage, this program might prove useful for certain usages.

Citing SPIKE

If you happen to use SPIKE successfully please cite it, and refer to this site, as well as the following possible references :

- first publication of the program itself - *rejected from Anal. Chem. Reviewer 1 said "too much NMR", Reviewer 2 said "too much MS", !!*
    1. Chiron L., Coutouly M-A., Starck J-P., Rolando C., Delsuc M-A. SPIKE a Processing Software dedicated to Fourier Spectroscopies https://arxiv.org/abs/1608.06777 (2016)
- first version of the python set-up on which the current SPIKE is partially based
    2. Tramesel, D., Catherinot, V. & Delsuc, M.-A. Modeling of NMR processing, toward efficient unattended processing of NMR experiments. *J Magn Reson* **188**, 56-67 (2007).
- first version of the 2D FT-ICR-MS processing
    3. van Agthoven, M. A., Chiron, L., Coutouly, M.-A., Delsuc, M.-A. & Rolando, C. Two-Dimensional ECD FT-ICR Mass Spectrometry of Peptides and Glycopeptides. *Anal Chem* **84**, 5589-95 (2012).
- presentation of the automation possibilities in NMR

4. Margueritte, L., Markov, P., Chiron, L., Starck, J.-P., Vonthron Sénécheau, C., Bourjot, M., & Delsuc, M.-A. (2018). Automatic differential analysis of NMR experiments in complex samples. Magn. Reson. Chem., 80(5), 1387. http://doi.org/10.1002/mrc.4683

ref 1) is a general purpose reference, the other ones are more specific.

SPIKE features

- FT analysis of 1D data-sets
  - apodisation, phasing, modulus, ...
- Analysis of 2D data-sets
  - phase or amplitude modulation
  - complex or hyper-complex algebra
- Robust processing
  - no limit in data-set size
  - parallel processing of the heaviest processing
    - on multi-core desktop using standard python ressources
    - on large clusters, using **MPI** library
- High-Level features
  - noise reduction (filtering, Linear-Prediction, Cadzow, urQRd, sane, ...)
  - automatic or manual baseline correction
  - NUS data processing
  - 1D and 2D Peak-Picker
- Plugin architecture
  - allow easy extension of the core program
  - reduces cross dependences
- Complete spectral display using matplotlib or bokeh transparently
  - zoom, available in several units (depending on the spectroscopy : seconds, Hz, ppm, m/z, etc...)
  - store to png or pdf
- interaction with the Jupyter Notebook environment

Handles the following Spectroscopies

- **NMR**
  - 1D and 2D are fully supported
  - DOSY
  - no nD yet
  - relaxation data in progress
- **FT-ICR**
  - 1D and 2DFT are fully supported
  - LC-MS in progress
- **Orbitrap**
  - 1D only (!)

- *other spectroscopies are being considered*

Files can be imported from

- NMR:
  - Bruker Topspin
  - NMRNoteBook
  - NPK - *Gifa*
  - SpinIt
- FT-ICR:
  - Bruker Apex
  - Bruker Solarix
- Orbitrap:
  - Thermofisher raw data
- Other
  - csv and txt files
  - any data in memory in a `Numpy` buffer.

Usage

As a processing library

SPIKE is primary meant for being used as a library, code can as simple as :

```
from spike.File import Solarix

dd = Solarix.Import_1D('FTICR-Files/ESI_pos_Ubiquitin_000006.d')  # Import create a basic SI

dd.hamming().zf(2).rfft().modulus()    # we have a simple piped processing scheme
  # here doing apodisation - zerofilling (doubling the size) - FT and modulus.
  # calibration is imported from Bruker - advanced calibration is available

dd.unit = "m/z"
dd.display(zoom=(500,2000))     # display the spectrum for m/z ranging from 500 to 2000

dd.pp(threshold=1E7)          # peak-pick the spectrum in this range
dd.centroid()                 # compute centroids

dd.display(zoom=(856.5, 858.5))    # and zoom on the isotopic peak
dd.display_peaks(zoom=(856.5, 858.5), peak_label=True)
```

interactive mode

SPIKE allows to process datasets interactively from an jupyter (IPython) prompt,
and is perfectly working in `jupyter notebook` or even `jupyter lab`

- Look at the examples files (`eg_*.py` and `*.ipynb`) for examples and some documentation. (* not fully up to data *)
- display is performed using the `Matplotlib` library.
- large 2D-FT-ICR are handled in batch using the `processing.py` batch program, controlled by parameter file called `*.mscf`
- The batch mode supports multiprocessing, both with MPI and natively on multi-core machines (still in-progress)
- large 2D-FT-ICR are stored in a hierarchical format, easyly displayed with an interactive program.
- data-sets are handled in the HDF5 standard file-format, which allows virtually unlimited file size (*tested up to 500 Gb*).

running stand-alone programs

processing.py and visu2D.py are two stand alone programs, written on the top of SPIKE. - processing.py allowing the efficient processing of FT-ICR 2D datasets, with no limit on the size of the final file Produces multi-resolution files

syntax :

```
python -m spike.processing param_file.mscf
```

How do I get SPIKE ?

SPIKE is written in pure Python, and relies on several external libraries. It is compatible and fully tested with python 3.7 (many parts are still compatible with python 2.7 but this is not tested)

dependencies

However it relies on mathematical libraries which should be installed independently.

- matplotlib
- numpy
- scipy
- tables
- pandas

some plugins or extension require additional libraries (`ipympl`, `MPI`, `bokeh`, `mayavi`, ...)

installation

To get it, you can simply - rely on a scientific distribution such as Anaconda or Enthough - install the above python distributions yourself (tricky)

Then you can install it using pip:

```
pip install spike-py
```

Or, if you want to play with the code, and get the bleeding edge version.

- `hg clone` get the devel branch and keep it up-to-date
- (will be move to git soon)

```
python setup.py install
```

or, if you do not want to instal it permanently

```
python setup.py develop
```

using pip

```
pip install spike_py
```

History

SPIKE is originated from the *Gifa* program, developed by M-A Delsuc and others in `FORTRAN 77` since the late eighties. *Gifa* has known several mutations, and finally ended as a partial rewrite called **NPK**. The NPK program is based on some of the original `FORTRAN` code, wrapped in Java and Python, which allows to control all the program possibilities from the Python level. NPK is purely a computing kernel, with no graphical possibilities, and has been used as a kernel embedded in the commercial program NMRNoteBook, commercialized by NMRTEC.

However, NPK was showing many weaknesses, mostly due to the 32bits organization, and a poor file format. So, when a strong scientific environment became available in Python, a rewrite in pure Python was undertaken. To this initial project, called NPK-V2, many new functionalities were added, and mostly the capability to work in other spectroscopies than NMR.

At some point in 2014, we chose to fork NPK-V2 to SPIKE, and make it public.

Developing for SPIKE

SPIKE is an open-source program, this means that external contributions are welcomed. If you believe your improvement is useful for other people, please submit a `pull request`. **Note** that pull request should be associated to the `devel` branch. This branch is devoted to new features not fully tested yet and still susceptible of changes, while the `default` branch is meant for stable code.

plugins

If you consider adding some new feature, it is probably a good idea to implement it as a plugin. The code contains already quite a few plugins, some are quite sophisticated - see `Peaks.py` for instance which implements a 1D and 2D peak picker, as well as a centroid evaluation and a full listing capability.

You can check also `fastclean.py` for a very simple plugin, or `wavelet.py` for a plugin relying on an external library which has to be installed.

Some Good Practice

- Spike contains many tools, most of the basic function for data interaction are found in the `NPKData.py` master file; utilities are also scattered in the `util` module. Use then, life will be easier for the users.
- Please write tests, even for the plugins ! We use standard python `unittest`, so nothing fancy. All the tests are run automatically every night (code is `Tests.py`), so it will detect rapidly all potential problem.
- push your pull requests to the `devel` branch - `default` is for the stable releases.

Organisation of the Code

The main program is `NPKData.py`, which defines NPKData object on which everything is built.

Spectroscopies are defined in the `NMR.py`, `FTICR.py` and `Orbitrap.py` code, which sub class NPKData.

Many programs contain routines tests (in an object unittest) that also serve as an example of use. The code goes through extensive tests daily, using the `unittest` Python library. However, many tests rely on a set of tests data-sets which is more than 1Go large, and not distributed here.

Main programs :

a small description of the files:

- NPKData.py the main library, allows all processing for any kind of experiments (1D, 2D and 3D) to be used as a library, in a stand-alone program or in IPython interactive session
- NMR.py The NPKData library adapted to NMR processing
- FTICR.py an extension of NPKData for processing FT-ICR datasets (1D and 2D)
- Orbitrap.py an extension of NPKData for processing Orbitrap datasets (1D)
- processing.py a stand alone program, written on the top of FTICR.py, allowing the efficient processing of FT-ICR 2D datasets, with no limit on the size of the final file Produces multi-resolution files syntax :

  ```
  python -m spike.processing param_file.mscf
  ```

Directories

- *Algo*
  contains algorithms to process data-sets (MaxEnt, Laplace, etc. . .) not everything active !
- *Display*
  a small utility to choose either for regular Matplotlib display of fake no-effect display (for tests)
- *File*
  Importers for various file format for spectrometry, as well as the HDF5 SPIKE native format.
- *plugins*
  Tools automatically plugged in NPK kernel : display utilities, urQRd algorithm and various other tools.
- *Miscellaneous*
  "en vrac"
- *util*
  set of low-level tools used all over in the code
- *v1*
  a library implementing a partial compatibility with the NPKV_V1 program
- *SPIKE_usage_eg*
  example of Python programs using the various libraries available
- *example of configuration files*
  - process_eg.mscf
  - test.mscf
- and *various utilities*
  - NPKConfigParser.py
    reads .mscf files
  - NPKError.py
    generates error msg
  - QC.py
    Quality Check
  - Tests.py
    runs all tests
  - dev_setup.py
    rolls a new version
  - version.py
    defines version number
  - *_init_*.py
    defines library

- rcpylint

- To_Do_list.txt
- QC.txt

- Release.txt

Authors and Licence

Current Active authors for SPIKE are:

- Marc-André Delsuc `madelsuc -at- unistra.fr`
- Laura Duciel `laura.duciel -at- casc4de.eu`

Previous authors:

- Christian Rolando `christian.rolando -at- univ-lille1.fr`
- Lionel Chiron `Lionel.Chiron -at- casc4de.eu`
- Petar Markov `petar.markov -at- igbmc.fr`
- Marie-Aude Coutouly . `Marie-Aude.COUTOULY - at- datastorm.fr`