

Instruction for *peakonly*

Up-to-date version of this document can be found via link https://bit.ly/peakonly_manual

Table of Contents

1. Description.....	2
2. Installing and running the application.....	3
2.1. Downloading, installing and running the application on Windows 7/10.....	3
2.2. [Optional] Optimizing ROI detection component (Windows).....	6
2.3. Downloading, installing and running the application on Linux and MacOS.....	7
2.4. [Optional] Optimizing ROI detection component (Linux, MacOS).	8
3. Getting familiar with the processing pipeline. Analyzing built-in examples of LC-MS data.	9
3.1. Loading *.mzML files.....	9
3.2. Visualizing data from *.mzML files.	10
3.2.1. Plotting chromatograms.	10
3.2.2. Chromatogram panel interface. Adjusting the view.	11
3.3. Detecting features in example data.	15
3.4. Exploring the detected features.	17
3.5. Exporting features.	18
4. Processing of your own LC-MS data.....	19
4.1. [In development] Retention time (RT) correction.....	20
5. [In development] Adjusting output results by re-training of Convolutional Neural Networks with your own LC-MS data.....	20

1. Description

Peakonly is a novel approach written in *Python* for peaks (aka features) detection in raw LC-MS data. The main idea underlying the approach is the training of convolutional neural networks to classify ROIs (regions of interest) into two classes (noise, peaks) and to determine boundaries for every peak to integrate its area. Current approach was developed for the high-resolution LC-MS data for the purposes of metabolomics, but can be applied with several adaptations in other fields that utilize data from high-resolution GC- or LC-MS techniques.

- **Article:** Deep learning for the precise peak detection in high-resolution LC-MS data, *Analytical Chemistry*, 2020, 92, 1, 588-592. <http://dx.doi.org/10.1021/acs.analchem.9b04811>
- **Source code:** <https://github.com/Arseha/peakonly/>

Supported formats:

- .mzML

Operating System Compatibility

peakonly has been tested successfully with:

- Ubuntu 16.04
- macOS Catalina
- Windows 10
- Windows 7

For Windows7/10 commands should be entered through Windows PowerShell. Detailed instruction is available. Be sure that your *python* version is at least 3.6.

Contacts:

Arsenty Melnikov – melnikov.arsenty@gmail.com

Vadim Yanshole – vadim.yanshole@tomo.nsc.ru

International Tomography Center
Siberian Branch of Russian Academy of Sciences
630090,
Institutskaya 3a,
Novosibirsk, Russia

2. Installing and running the application.

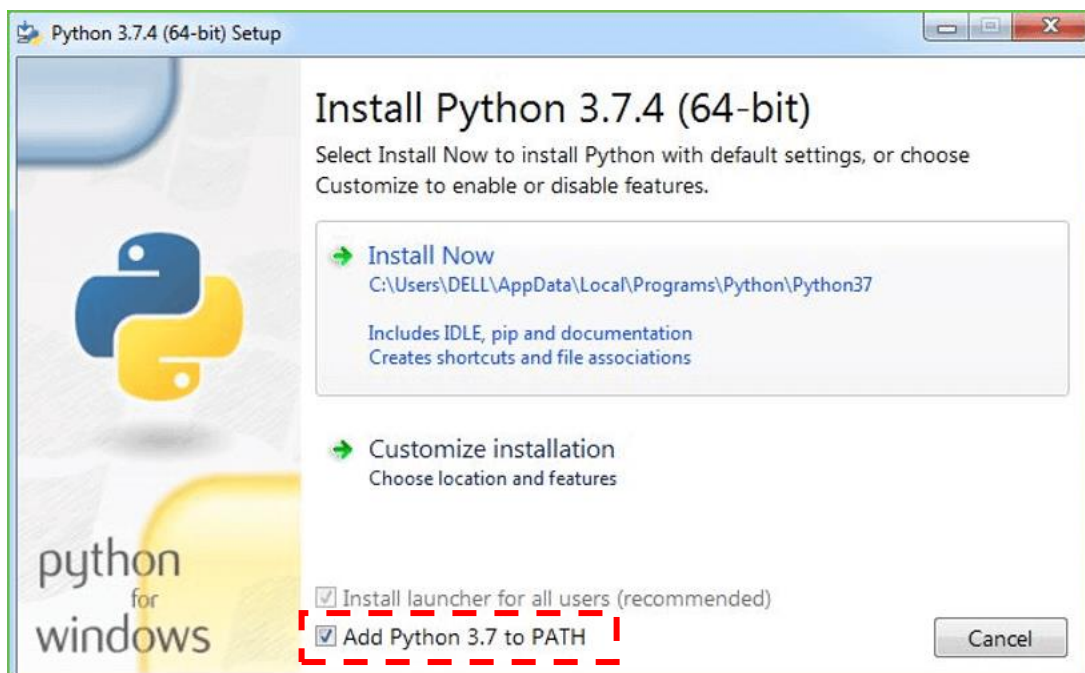
2.1. Downloading, installing and running the application on Windows 7/10.

- 1) Download *Python*, version 3.7.5 is recommended. <https://www.python.org/downloads/release/python-375/>

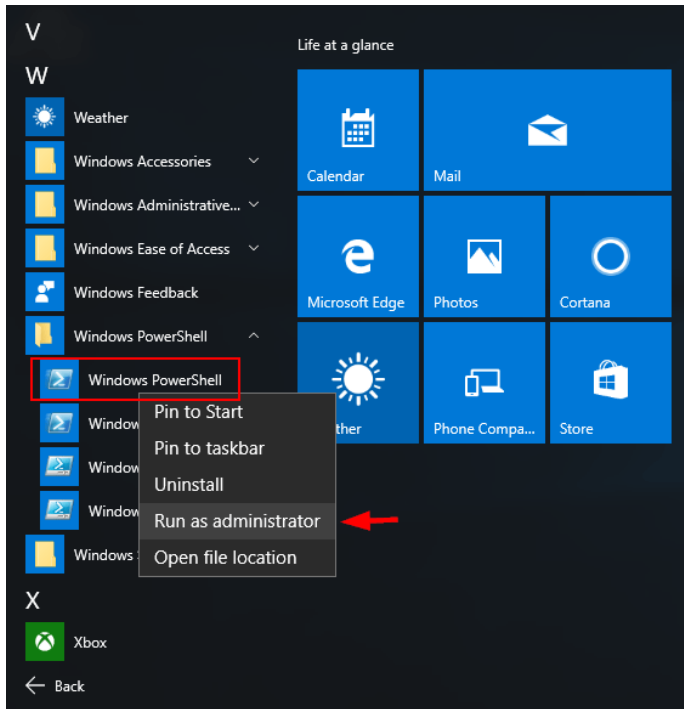
Files

Version	Operating System	Description	MD5 Sum	File Size	GPG
Gzipped source tarball	Source release		1cd071f78ff6d9c7524c95303a3057aa	23126230	SIG
XZ compressed source tarball	Source release		08ed8030b1183107c48f2092e79a87e2	17236432	SIG
macOS 64-bit/32-bit installer	Mac OS X	(Deprecated) for Mac OS X 10.6 and later	cd503606638c8e6948a591a9229446e4	35020778	SIG
macOS 64-bit installer	Mac OS X	for macOS 10.9 and later	20d9540e88c6aaba1d2bc1ad5d069359	28198752	SIG
Windows help file	Windows		608cfa250f8baa11a69bbfcb842c0e0	8141193	SIG
Windows x86-64 embeddable zip file	Windows	for AMD64/EM64T/x64	436b0f803d2a0b393590030b1cd59853	7500597	SIG
Windows x86-64 executable installer	Windows	for AMD64/EM64T/x64	697f7a884e80ccaa9dff3a77e979b0f8	26777448	SIG
Windows x86-64 web-based installer	Windows	for AMD64/EM64T/x64	b8b6e5ce8c27c20bfd28f1366ddf8a2f	1363032	SIG
Windows x86 embeddable zip file	Windows		726877d1a1f5a7dc68f6a4fa48964cd1	6745126	SIG
Windows x86 executable installer	Windows		cfe9a828af6111d5951b74093d70ee89	25766192	SIG
Windows x86 web-based installer	Windows		ea946f4b76ce63d366d6ed0e32c11370	1324872	SIG

- 2) Install *Python*. Make sure to check the box under **Add Python 3.x to PATH**.



- 3) Open *Windows PowerShell* (it can be found using windows search). Run *Windows PowerShell* as **administrator** by the mouse right-click (as depicted). Running without administrator could raise “Permission Error” (or similar errors) during the execution of the next steps.



- 4) Download all the necessary libraries described in the *requirements.txt* via *Windows PowerShell* using *Python pip* module:

a) `pip3 install bintrees, matplotlib, numpy, pandas, pymzML, PyQt5, scipy, tqdm`

This line can be copied in a usual way and pasted into *Windows PowerShell* by the mouse right button.

```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

PS C:\Users\User> pip3 install bintrees, matplotlib, numpy, pandas, pymzML, PyQt5, scipy, tqdm
```

- b) Install *PyTorch* framework:

`pip3 install torch==1.3.1+cpu torchvision==0.4.2+cpu -f https://download.pytorch.org/whl/torch_stable.html`

This line can be copied in a usual way and pasted into *Windows PowerShell* by the mouse right button.

```
Windows PowerShell
PS C:\Users\User> pip3 install torch==1.3.1+cpu torchvision==0.4.2+cpu -f https://download.pytorch.org/whl/torch_stable.html
```

if the installation of *PyTorch* was unsuccessful try to follow the instructions from the *PyTorch* website (<https://pytorch.org/get-started/locally/>).

5) Download the latest release of *peakonly* (<https://github.com/arseha/peakonly/releases>). Save it in convenient folder (*C:\yourpath*).

6) Change path in the *Windows PowerShell* to the downloaded repository folder:

```
cd C:\yourpath\peakonly\
```

Path can be copied from the path line of *Windows Explorer* (Win7) or *File Explorer* (Win10) and entered into *Windows PowerShell* by the mouse right button



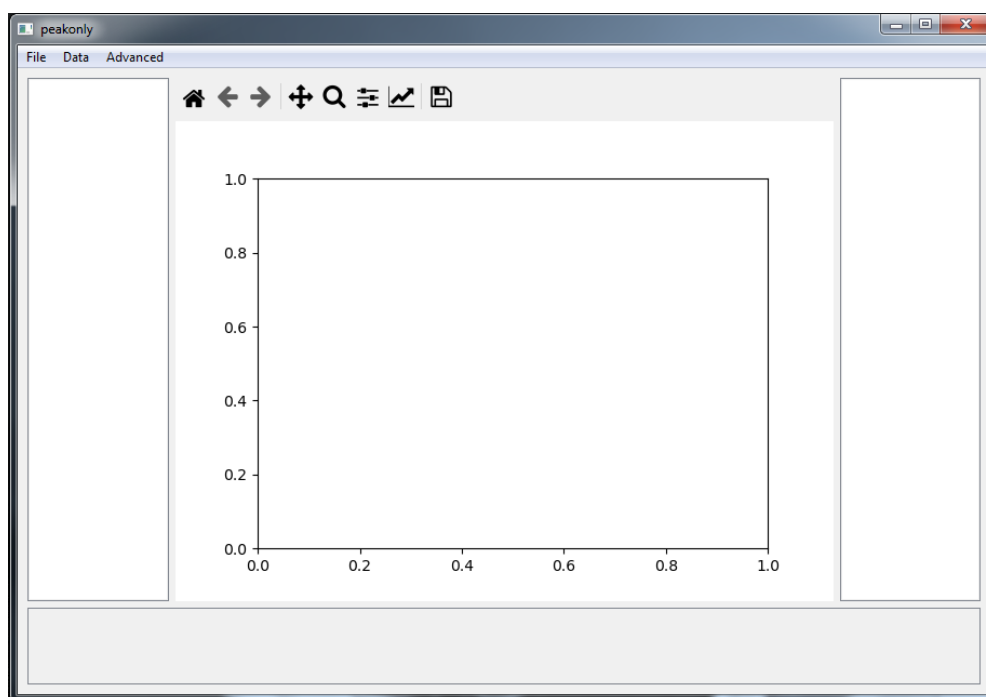
7) Execute **peakonly.py** via *Windows PowerShell*:

```
python peakonly.py
```

This line can be copied in a usual way and pasted into *Windows PowerShell* by the mouse right button.



a *peakonly* window will appear:



Windows PowerShell remains opened. Do not close *Windows PowerShell*, because this will close *peakonly* as well.

To close program choose **File** → **Exit** or simply click X button at the top right corner.

2.2.[Optional] Optimizing ROI detection component (Windows).

If you feel that ROI detection takes considerable amount of time (~minutes), you can optimize the ROI detection component by compiling this component of *peakonly*.

- 1) Install Cython library (*Windows PowerShell*):

```
pip3 install Cython
```

This line can be copied in a usual way and pasted into *Windows PowerShell* by the mouse right button.

- 2) Download Visual C++ Build tools:

<https://visualstudio.microsoft.com/downloads/#build-tools-for-visual-studio-2019>

Build Tools for Visual Studio 2019

These Build Tools allow you to build Visual Studio projects from a command-line interface. Supported projects include: ASP.NET, Azure, C++ desktop, ClickOnce, containers, .NET Core, .NET Desktop, Node.js, Office and SharePoint, Python, TypeScript, Unit Tests, UWP, WCF, and Xamarin.

Download ↓

- 3) Run downloaded *vs_BuildTools***.exe*
- 4) Select **Visual C++ build tools** checkbox (be sure that the **VC++ toolset** and **Windows 10 SDK** are also selected as components)
- 5) Install
- 6) Go to *peakonly* folder in *Windows PowerShell*
- 7) Execute

```
python setup.py build_ext --inplace
```

this will compile several modules of *peakonly*, making ROI detection much faster. In our experience the processing of some specific files without compilation takes ~6 minutes, and after compilation – ~10 seconds.

2.3. Downloading, installing and running the application on Linux and MacOS.

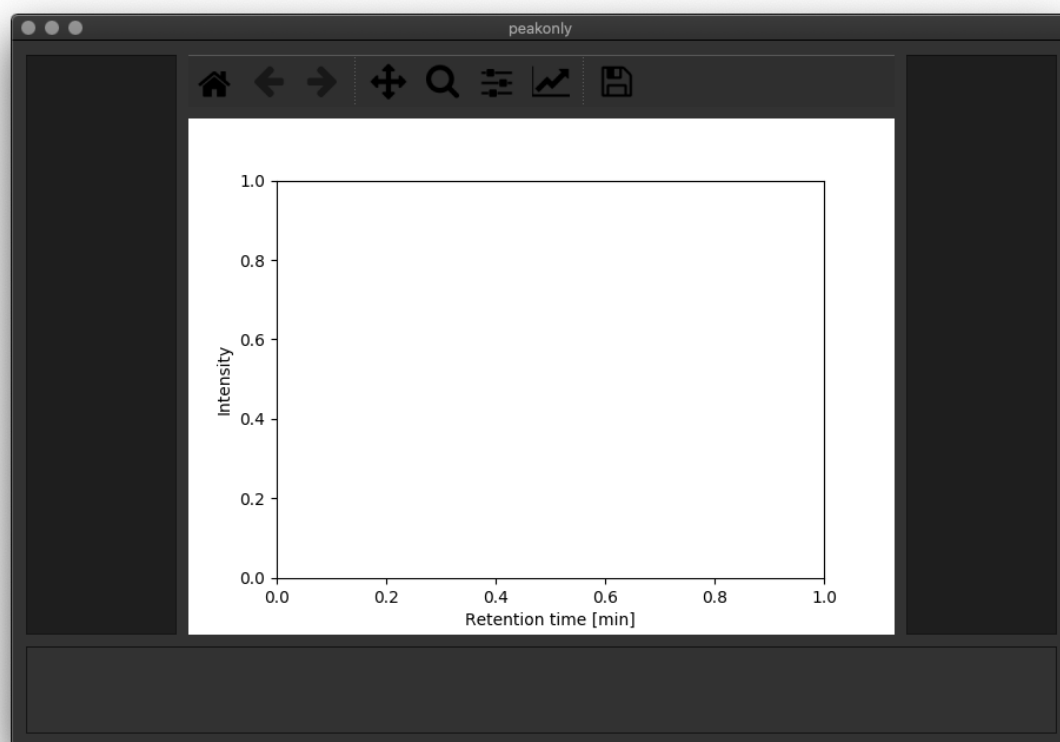
- 1) Make sure that your *python* version ≥ 3.6
- 2) Download the latest release of *peakonly* (<https://github.com/arseha/peakonly/releases>). Save it in convenient folder.
- 3) Install requirements in the following automated way (or you can simply open requirements.txt file and download listed libraries in any other convenient way):

```
pip3 install -r requirements.txt
```

- 4) Run peakonly:

```
python3 peakonly.py
```

- 5) *Peakonly* window should appear:



A view of the window may vary depending on your OS.

2.4.[Optional] Optimizing ROI detection component (Linux, MacOS).

If you feel that ROI detection takes considerable amount of time (~minutes), you can optimize the ROI detection component by compiling this component of *peakonly*.

- 1) Install Cython:

```
pip3 install Cython
```

- 2) Make sure that a C++ compiler is presented in your system
(<https://cython.readthedocs.io/en/latest/src/quickstart/install.html>).
- 3) Run compilation in the following way:

```
python3 setup.py build_ext --inplace
```

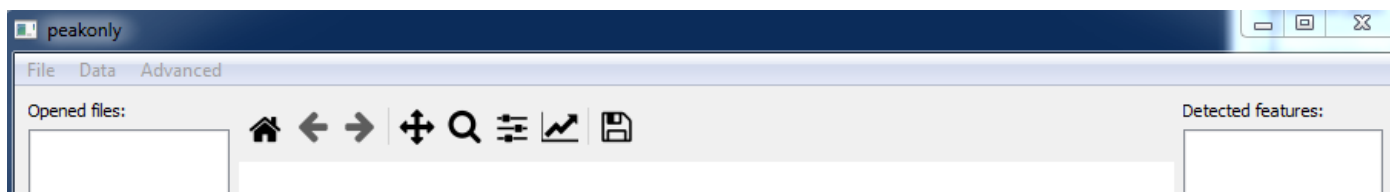
this will compile several modules of *peakonly*, making ROI detection much faster. In our experience the processing of some specific files without compilation takes ~6 minutes, and after compilation – ~10 seconds.

3. Getting familiar with the processing pipeline. Analyzing built-in examples of LC-MS data.

3.1. Loading *.mzML files.

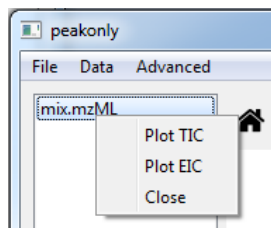
Download an example file **Data** → **Download** → **Download *.mzML example**. This will download **mix.mzML (16.9Mb)** file into **..\peakonly\data** folder. The file contains high-resolution LC-MS data obtained from HPLC-ESI-Q-TOF instrument. Detailed description of the data is provided in the paper by *Melnikov et al., Anal. Chem., 2020, 92, 1, 588-592*.

From the peakonly window choose **File** → **Open** → **Open *.mzML**, choose **mix.mzML** file in **..\peakonly\data** folder, and click open. The loaded file will be shown at the left panel of the application.



To load several *.mzML files (batch) make multiple selection in the window in a usual way. All loaded files will be shown at the left panel of the application.

To close loaded file(s) highlight them and right-click the mouse and choose **Close**.

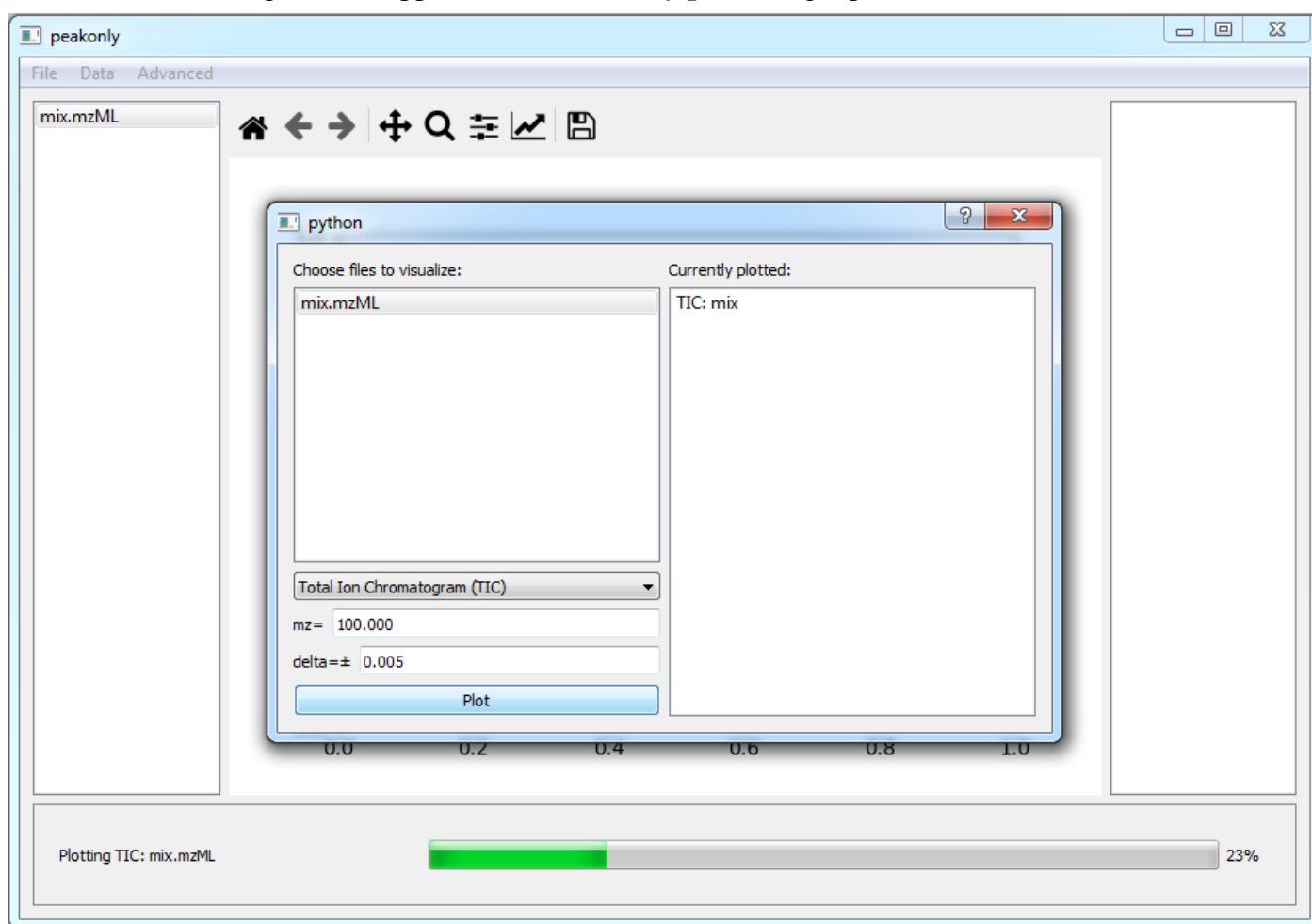


3.2. Visualizing data from *.mzML files.

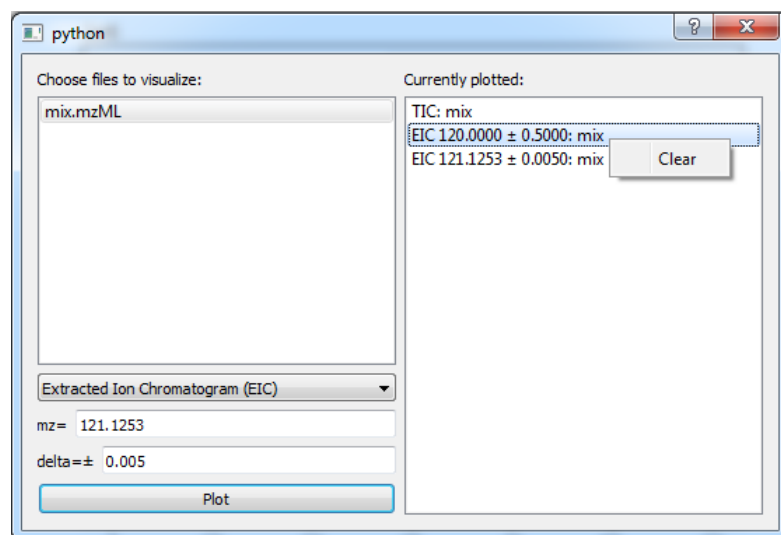
3.2.1. Plotting chromatograms.

Peakonly has two options to visualize loaded data by plotting **Total Ion Chromatogram (TIC)** and plotting **Extracted Ion chromatogram (EIC)**. To plot chromatograms choose **Data** → **Visualization**. A new window will pop up.

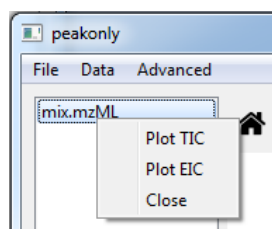
- To plot **TICs** it is necessary to highlight file(s), choose **Total Ion Chromatogram (TIC)** option from the drop-down menu and click **Plot** button. The progress bar will appear at the bottom of the main window and plotted chromatogram will appear in **Currently plotted** right panel.
- To plot **EICs** it is necessary to highlight file(s), choose **Extracted Ion Chromatogram (EIC)** option from the drop-down menu, choose desired **m/z** and **delta ±m/z** (in Daltons) and click **Plot** button. Plotted chromatogram will appear in the **Currently plotted** right panel.



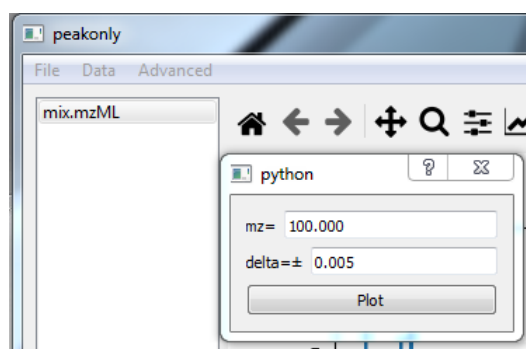
- To remove chromatograms highlight them in the **Currently plotted** right panel, right-click the mouse and choose **Clear**.



Alternatively, TICs and EICs can be plotted from the main window.



- To plot **TIC** highlight desired files in the left **Opened files** panel, right-click the mouse and choose **Plot TIC**.
- To plot **EIC** highlight desired files in the left **Opened files** panel, right-click the mouse and choose **Plot EIC**. A new window will pop up, where one choose desired **m/z** and **delta** $\pm m/z$ (in Daltons) and click **Plot** button.

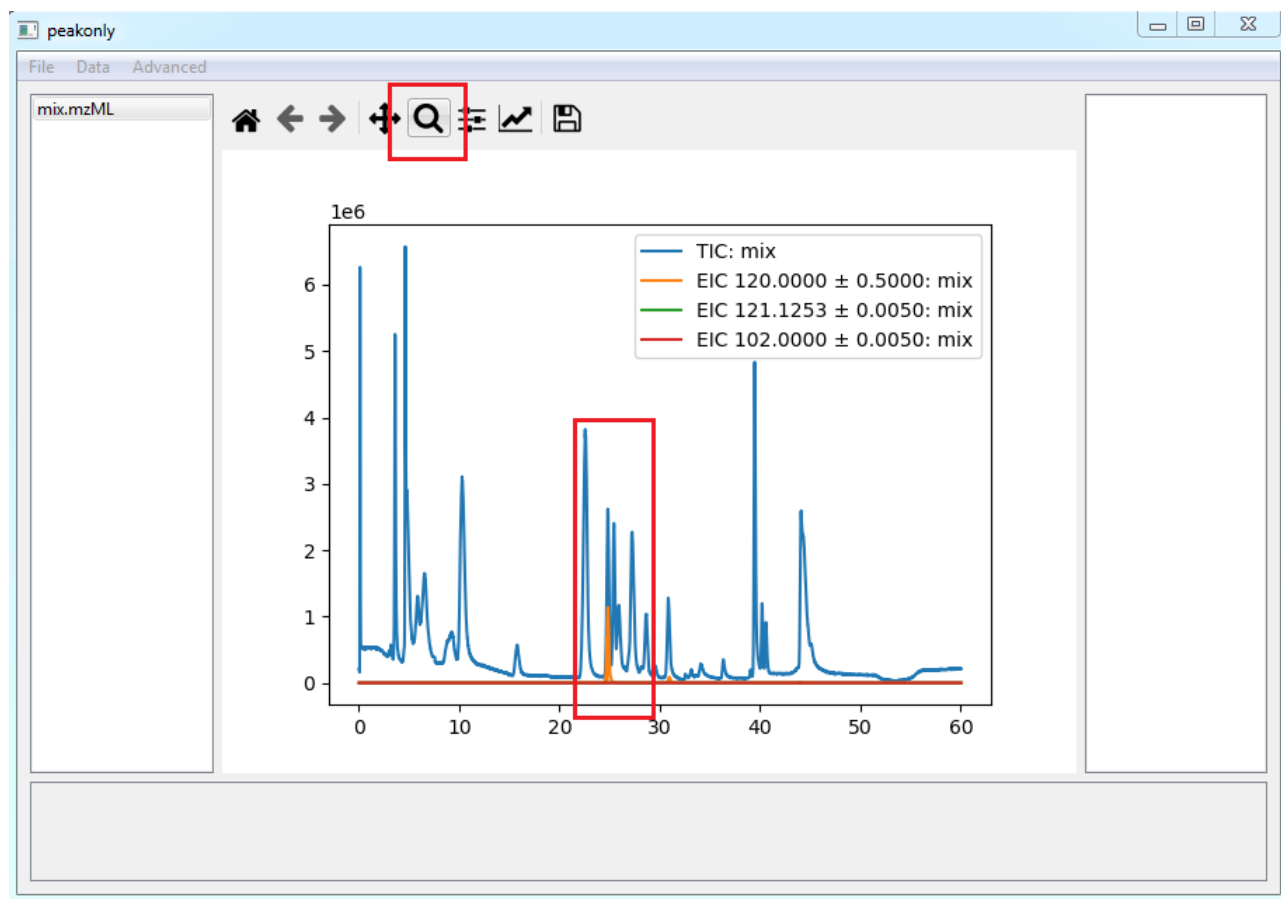


These plotted chromatograms can further as well be managed from the **Data** → **Visualization** menu.

3.2.2. Chromatogram panel interface. Adjusting the view.


The visualization component of peakonly is implemented on *matplotlib* module for python. A more detailed description of its usage can be found: https://matplotlib.org/3.1.1/users/navigation_toolbar.html

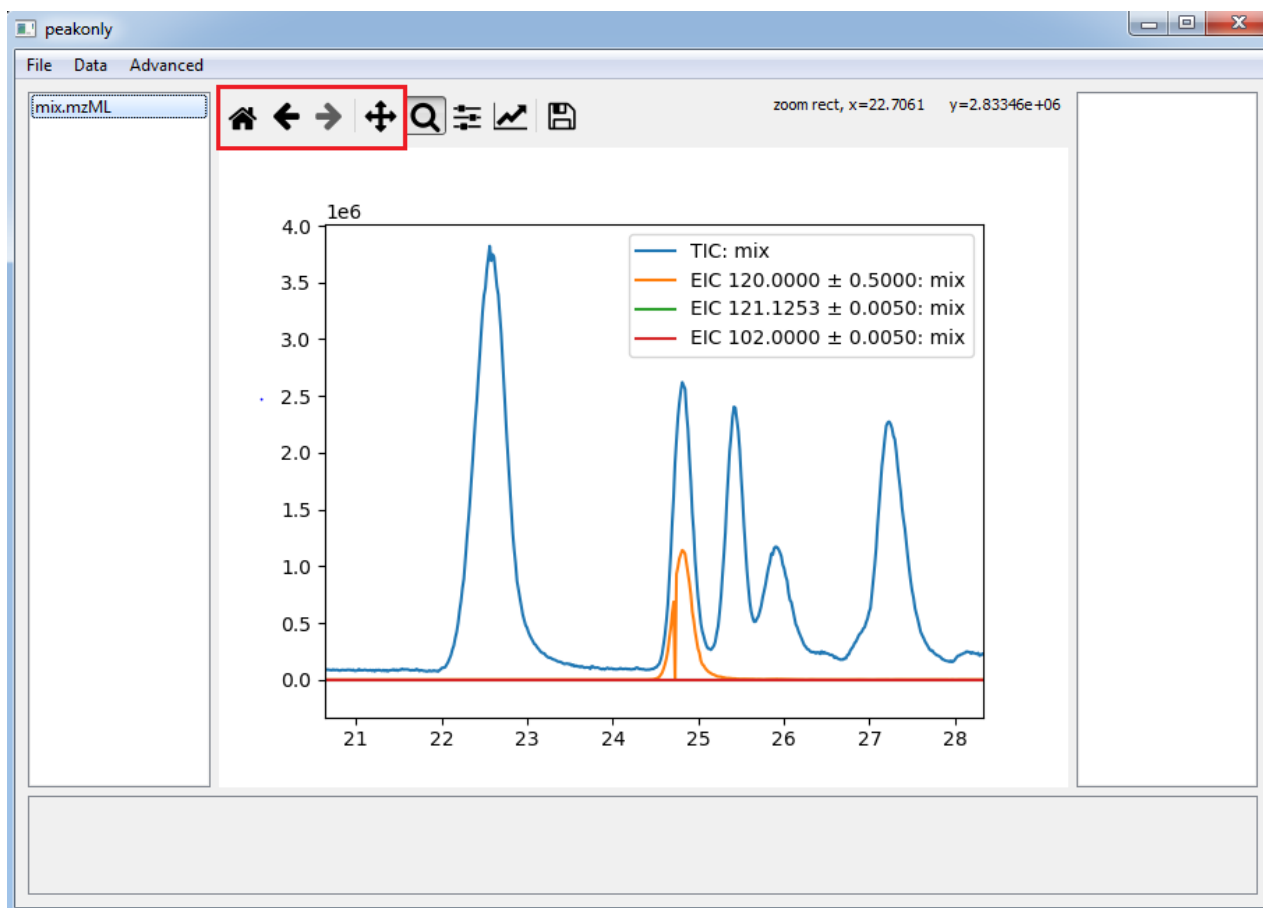
To zoom in the part of the plotted chromatograms in the center panel choose **Zoom to rectangle** button (magnifier icon) and choose desired area by holding mouse left button.



To unzoom hold mouse right button and choose area.

In addition, view can be changed with the following tools:

- Choose **Reset original view** (house icon)
- Choose **Back to previous view** (left arrow icon)
- Choose **Forward to next view** (right arrow icon)
- To move plot, choose **Pan/Zoom**  icon, hold mouse left button and move mouse. To change scale, choose **Pan/Zoom** icon, hold mouse right button and move mouse.



Matplotlib also allows visualizing subplots. For our opinion, for LC-MS or GC-MS data visualizations, this is not an absolutely necessary option, but it is left by default in *peakonly*. To **Configure subplots** choose its



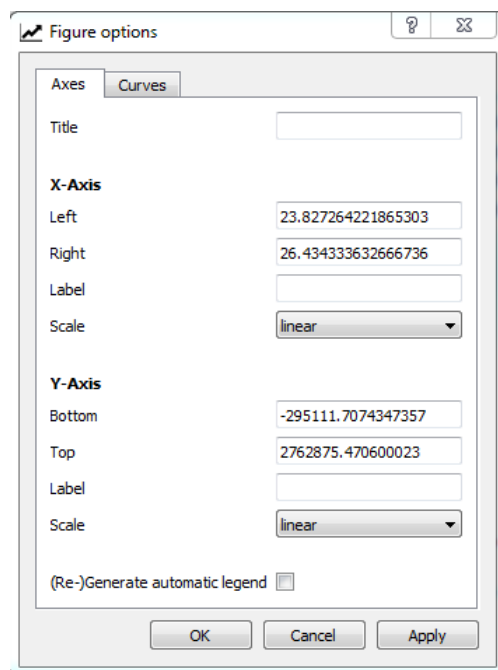
button.



Figure options can be accessed with button. A new window will pop up with **Axes** and **Curves** tabs.

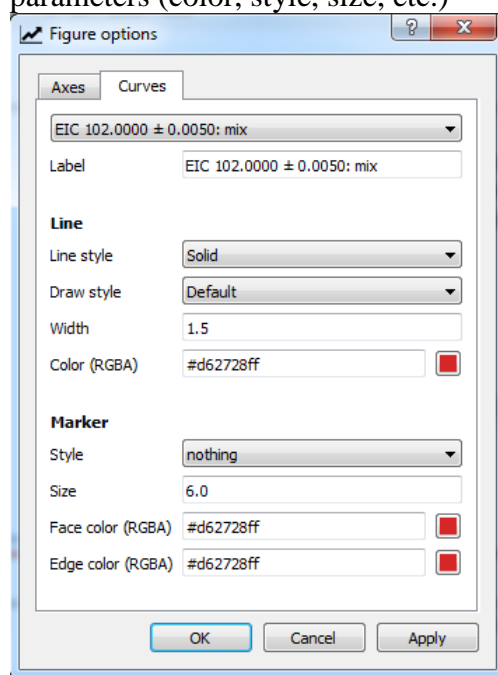
Axes tab allows to:


- Change view more precisely by changing start and end points of the x- and y-axis;
- Change labels of the x- and y-axis
- Change title of the plot
- Choose whether linear, logarithmic or logit scale should be used in x- and y-axis



Curves tab allows to:

- Choose chromatogram
- Change its label
- Change its line and marker parameters (color, style, size, etc.)



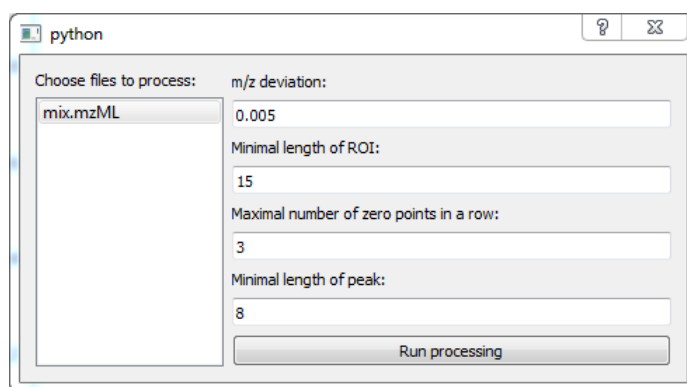
Current view can be saved as bitmap or vector graphics with  button.

3.3. Detecting features in example data.

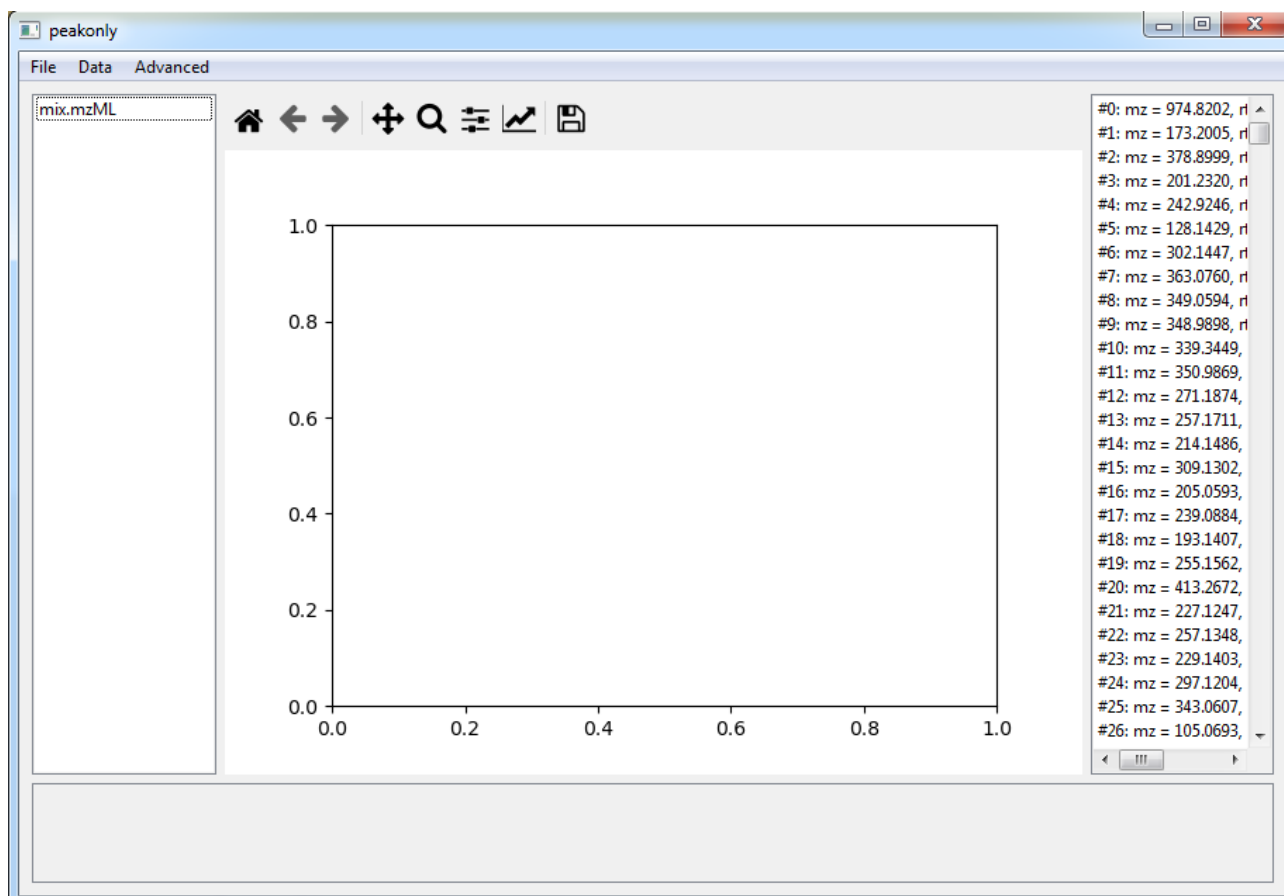
Download trained models of convolutional neural networks (CNN). **Data** → **Download** → **Download trained models**. This will download CNN weights (~5Mb) into `..\peakonly\data\weights` folder.

Load **mix.mzML** file with high-resolution LC-MS data (**Chapter 3.1**). Choose **Data** → **Feature detection** option. New window will pop up. Choose file to process by highlighting it (it is already highlighted by default), adjust detection parameters (or leave default values):

- **m/z deviation** in Daltons– maximal allowed deviation of peak maxima points for the detection of Regions of Interest (ROI) in LC-MS data by the *centWave*-based algorithm (Detailed description: *Tautenhahn, et al, BMC Bioinf. 2008, 9, 1–16.*). By default = 0.005 Da.
- **Minimal length of ROI** in points. By default = 15 points. This parameter is needed to exclude very short ROI containing only noise. 1 Hz data collection rate results in 1 point per second, 2 Hz – 2 points per second, etc. If your narrowest LC peak is 20 seconds, and the data collection rate is 1 Hz, it is good to make this parameter = 30
- **Maximal number of zero points in a row**. By default = 3 points. In the *centWave*, the appearance of zero points led to the immediate ROI termination. We added the possibility that ROI can include zero points. The purpose of this modification is the following: noise sometimes can randomly form something looking like a peak; without vicinity points, it is hard to classify whether the region contains peak or noise, even for a human expert. It is recommended to use a default value. However, if the algorithm found lots of small intensity false-positive signals with short ROIs it would be useful to increase this parameter.
- **Minimal length of peak** in points. By default = 8 points. This parameter is needed to exclude very short peaks. The area of such short peak is very unreliable, and should not be used for quantitative/semi-quantitative purposes. If your narrowest LC peak is 20 seconds, and the data collection rate is 1 Hz, it is good to make this parameter = 20



Push **Run processing** button, this will begin the feature detection process. It should take 10-30 seconds to process **mix.mzML** file. The progress bar will appear at the bottom of the main window and the resulting list of the detected features will be shown in the right **Detected features** panel of the main window.



3.4.Exploring the detected features.

Features list in the right panel is sorted by the increase of m/z. It has the following attributes:

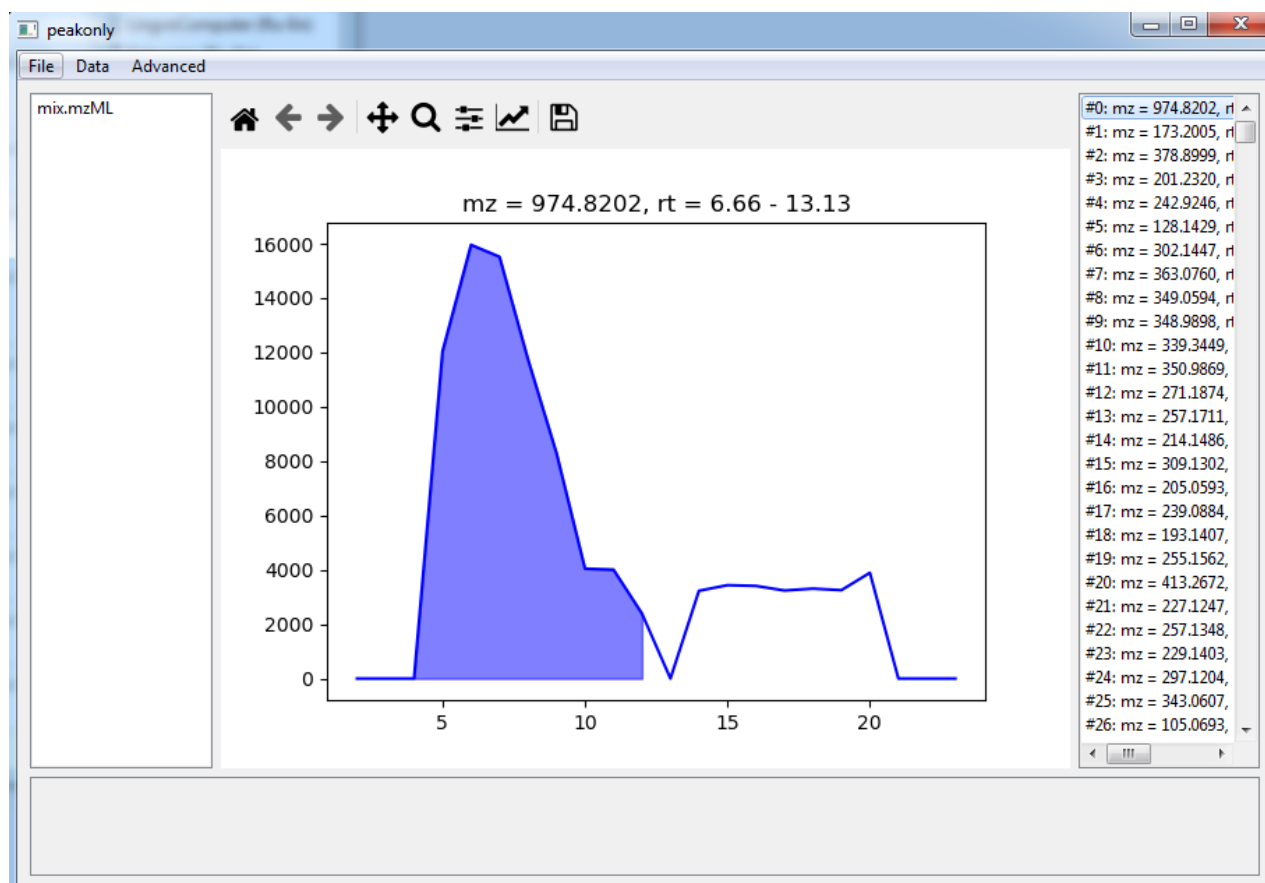
1 **2** **3**
#0: m/z = 974.8202, rt = 6.66 - 13.13
#1: m/z = 173.2005, rt = 8.80 - 19.77
#2: m/z = 378.8999, rt = 6.83 - 21.34
#3: m/z = 201.2320, rt = 8.92 - 23.27

1 – Feature unique order number

2 – Median m/z. Calculated as median from all the points in the LC peak (feature).

3 – Boundaries of the LC peak (feature), its start and end points. In minutes or seconds depending on source *.mzML file.

To plot the detected peak, double-click the corresponding feature from the list. A plot will appear at the center panel. Alternatively, right-click on a feature from the list and choose either “**Plot with RT correction**” or “**Plot without RT correction**”. For a single file, the result will be the same; for multiple files refer to **Chapter 4**.



Plotted graph will depict the ROI with found LC peak (feature). X-axis shows RT in points, Y-axis shows the intensity. Colored area under the curve indicates the boundaries for the detected peak in this ROI. There could be more than one peak found in a ROI; in this case, several features will be created.

Example: If there are two features detected in the ROI, first feature will depict only this particular (first) peak in the graph, while second feature belonging to this ROI (provided in the list in **Detected features** panel) will not contain information about the first peak. The plot view can be adjusted (**Chapter 3.2.2**).

To empty the list with the detected features list choose **File** → **Clear panel with detected features**

3.5.Exporting features.

There are two options to save results:

- 1) export list of features as ***.csv** file,
 - 2) export pictures of ROI with the detected peaks as multiple ***.png** files. (This can take several minutes)
- The ***.png** files will contain plots for features with RT correction (**Chapter 4.1**).

The resulted ***.csv** file will contain information about m/z and RT (*rtmax*, *rtmin*) for every peak as well as calculated areas (*intensity**). *Intensity** will represent the full path to the ***.mzML** file(s). The table from ***.csv** should look the following way:

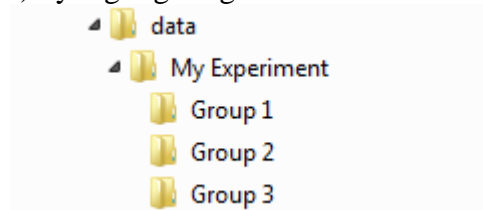
	mz	Rtmax	rtmin	Intensity1	Intensity2	...
0	974.8202068	0.110937302	0.218880159	74008	74008	
1	173.2005336	0.146637654	0.329484568	19576	19576	
2	378.8999329	0.113816667	0.355616667	24448	24448	
3	201.2320027	0.148654505	0.387826126	23224	23224	
4	242.92461	0.102415132	0.411717763	53544	53544	
...						

4. Processing of your own LC-MS data.

You can detect features in your own high-resolution LC-MS (GC-MS) data with the use of convolutional neural networks (CNN) of *peakonly*. Feature detection can be achieved with already trained CNNs. Download trained models of convolutional neural networks (CNN). **Data** → **Download** → **Download trained models**. This will download CNN weights (~5Mb) into `..\peakonly\data\weights` folder.

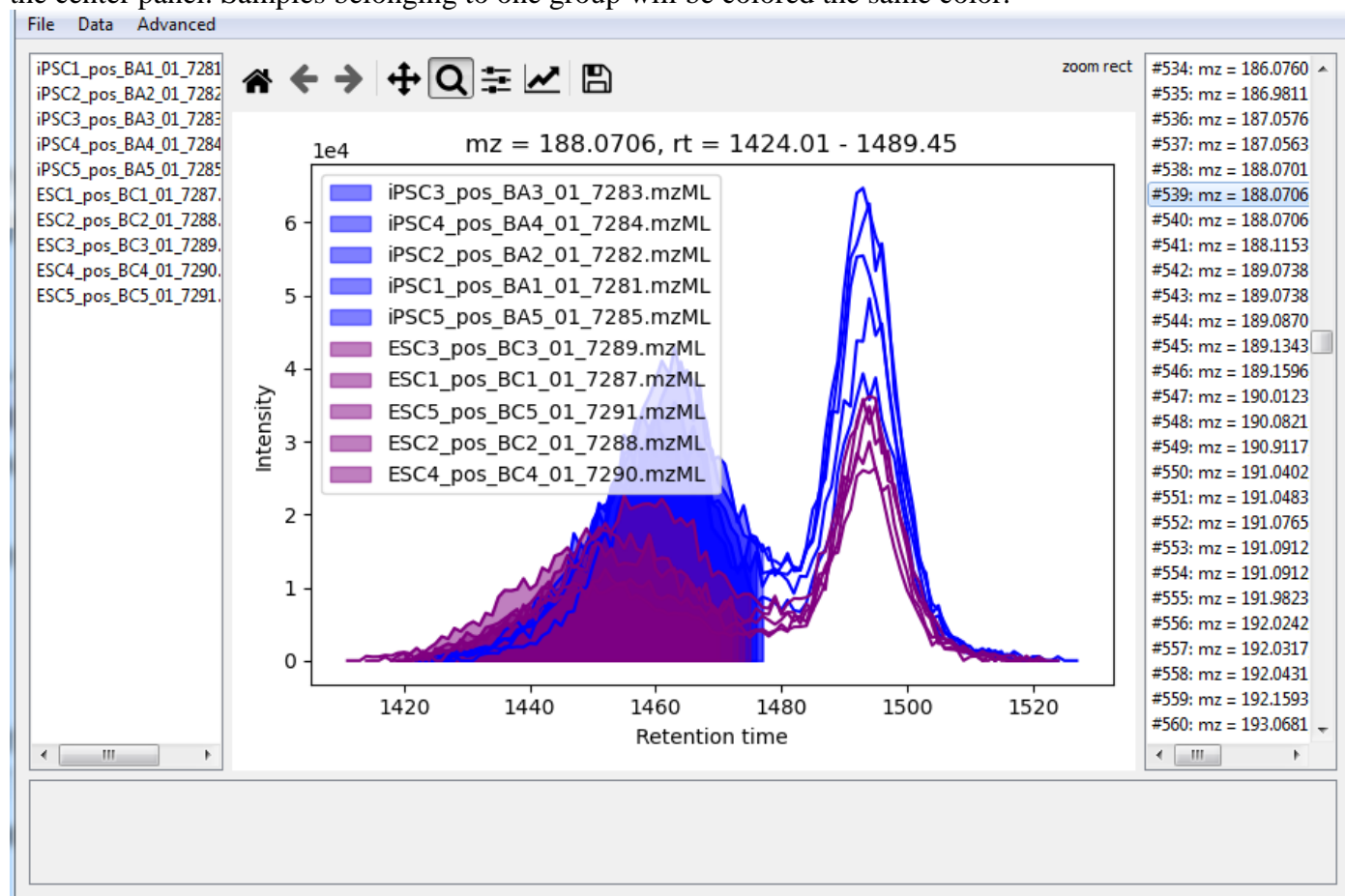
Load (**Chapter 3.1**) one or several (batch) ***.mzML** files.

You can mark groups for the samples in a batch. To mark samples place files belonging to one group into separate folders. To load entire experiment choose **File** → **Open** → **Open folder with *.mzML**. Select desired folder (e.g. **My Experiment**) by highlighting it and click **Choose folder** button.



Choose **Data** → **Feature detection** option (**Chapter 3.3**). New window will pop up. Choose file to process by highlighting it (all files are highlighted by default), adjust detection parameters (**Chapter 3.3**), or leave default values. Push **Run processing** button.

Detected features will appear in the right panel (**Chapter 3.4**). To plot the detected peak, double-click the corresponding feature from the list. A plot with retention time (RT) correction (**Chapter 4.1**) will appear at the center panel. Samples belonging to one group will be colored the same color.



Alternatively, right-click on a feature from the list and choose either **“Plot with RT correction”** or **“Plot without RT correction”**.

Save obtained results as *.csv or *.png files (Chapter 3.5).

4.1.[In development] Retention time (RT) correction

[This chapter and corresponding application components are under development.]

Retention time correction is needed for the better alignment of features between samples in a batch. Chromatography usually results in not exactly the same RT for a feature throughout all samples. Particularly pronounced RT shift is observed in a hydrophilic interaction liquid chromatography (HILIC). Peakonly tries to align peaks across samples.

5. [In development] Adjusting output results by re-training of Convolutional Neural Networks with your own LC-MS data.

[This chapter and corresponding application components are under development.]