mzAccess web service API reference

Version 1.5, 2017-01-11

Contact: Yaroslav.Iyutvinskiy@ki.se

1 Table of Contents

2 Overview

This document describes the web service protocol used by the mzAccess software to transfer data between the mzAccess server and mzAccess clients. The web service is built using the SOAP, and compatible with SOAP protocol v1.1 and 1.2.

The mzAccess service was developed for accessing data files produced by mass spectrometers coupled to chromatography methods, such as liquid chromatography-mass spectrometry (LC-MS) data. In this document we will sometimes refer to the instrument data as LC-MS data, but strictly speaking any data consisting of a sequence of spectra over time can be used.

In mzAccess, a data file is considered as a set of *signals* (intensity measurements) over the mass/charge (MZ) and retention time (RT) two-dimensional space, referred to as *MZ-RT space* (Figure 1). Therefore, any request for some subset of data from a data file can be specified by coordinates describing an *area* in MZ-RT space, possibly processed in some way (for example, averaged over the retention time dimension). All data retrieving functions provided by the mzAccess web service require such MZ-RT coordinates. A summary of the types of data provided by the service is shown schematically in Figure 1. More detailed information about design of mzAccess could be retrieved from original article at Analytical Chemistry or PMC Full text.

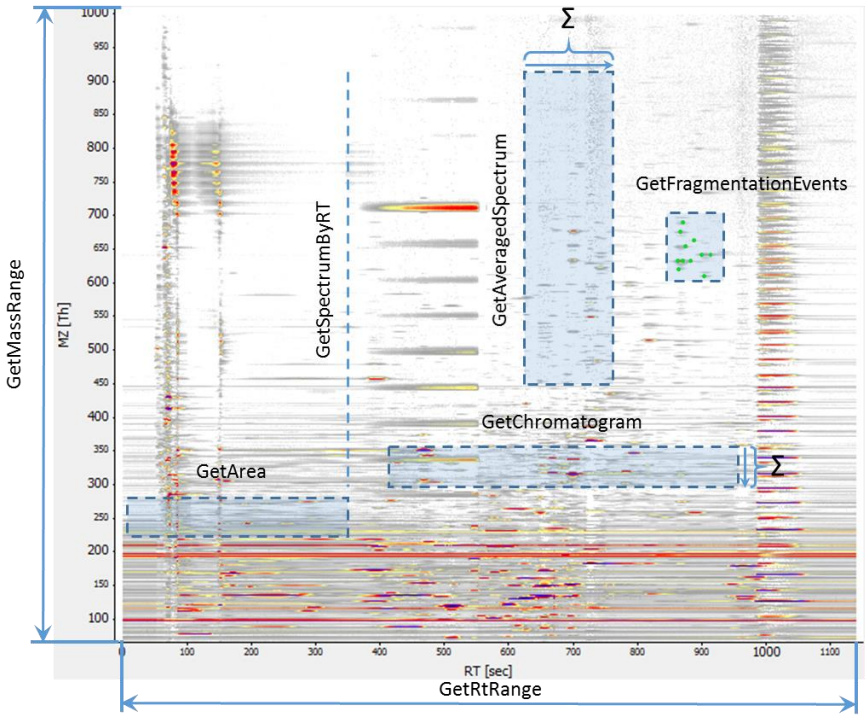

Figure 1. Example 2D Map of MZ-RT space with color-coded intensity. Data types provided by functions as it related to RT/MZ space.

3 Types of data provided

This section describes the various types of data provided by mzAccess.

3.1 Chromatogram

A *chromatogram* (or extracted ion chromatogram, XIC) is a projection of the signals in a certain MZ-RT area, usually over a narrow MZ range, onto the RT dimension. The projection is usually a sum of signals (denoted Σ in Figure 1), but the precise definition depends on the instrument used. A chromatogram consists of a list of (RT, Intensity) pairs, each representing the intensity value at one scan. Scans with zero intensity may be omitted, so the RT values of a chromatogram are not guaranteed to be evenly

spaced. Moreover, for performance reasons, mzAccess always returns chromatogram data as a "flattened" array

```
[ RT1, Intensity1, RT2, Intensity2, ..., RTN, IntensityN ]
```

This avoids excessive overhead in the SOAP protocol associated with representing nested arrays.

Typically, a chromatogram over a narrow MZ range represents the elution profile of one or more compounds of the same molecular mass. The area under curve of elution profile is used for estimation of absolute or relative abundance of a compound. An example of chromatogram data is shown at Figure 2. Dots represent actual data points where each point is for signal intensity in spectra taken at particular RT.

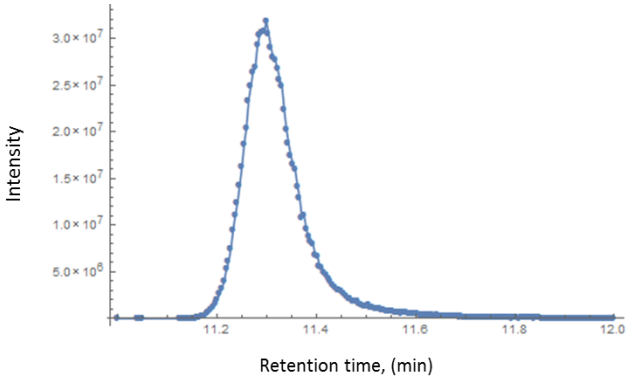

Figure 2. Sample of chromatogram data.

3.2 Spectrum

A *spectrum* is a projection of the signals in a certain MZ-RT area over a RT range onto the MZ dimension. While a chromatogram describes the elution profile over time of compounds at a specific MZ, a spectrum provides MZ information for all compounds eluted at a specific RT range. A spectrum consists of a list of (MZ, Intensity) pairs, each representing the intensity value at one mass/charge point. Similar to chromatograms, for performance reasons mzAccess always returns spectrum data as a "flattened" array

```
[ MZ1, Intensity1, MZ2, Intensity2, ..., MZN, IntensityN ]
```

The MZ values are generally not regularly spaced.

LC-MS data is produced by capturing a sequence of spectra over time, and each spectrum captured is assigned a unique *scan number*, a unique integer that identifies the spectrum. In mzAccess, spectra can be retrieved directly by their scan number (see GetSpectrumByScanNumber), and this method can to access both MS^1 spectra and fragmentation spectra (MS/MS or higher order fragmentation events). In addition, individual MS^1 spectra can be accessed by specifying an approximate retention time (see GetSpectrumByRT). Spectra can also be obtained by averaging across a given RT range (see GetAverageSpectrum).

3.2.1 Centroid and Profile modes

Spectra are available in *centroid mode* or *profile mode*, provided this information exists in the underlying instrument data file. Centroid mode spectra are result of peak detection procedures implemented by vendor software libraries that usually generate a single (MZ, Intensity) pair for each MZ peak; this is the most commonly used mode. Profile mode provides unprocessed spectra as recorded by the mass spectrometer, which can be used to reveal sources of MZ estimation errors or MZ peak detection mistakes. For example, in Figure 3, the peak at MZ = 279.0 at left panel of figure should experience positive MZ shift as it is located on the shoulder of bigger peak at 279.1 Da.

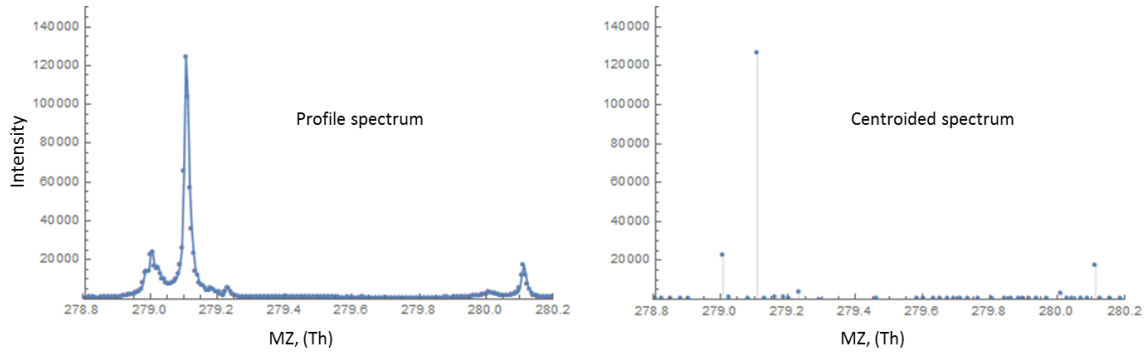

Figure 3. Profile and centroided spectra modes for the same MZ-RT area. Points represent actual MZ/Intensity values provided by service.

3.3 MZ-RT Area

mzAccess can also retrieve all signals acquired in a certain MZ-RT area as list of (MZ, RT, Intensity) triples (see GetArea). As with chromatograms and spectra, mzAccess returns these triples as a flattened array for performance reasons, in the following order:

```
[ MZ1, RT1, Intensity1, MZ2, RT2, Intensity2, ..., MZN, RTN, IntensityN ]
```

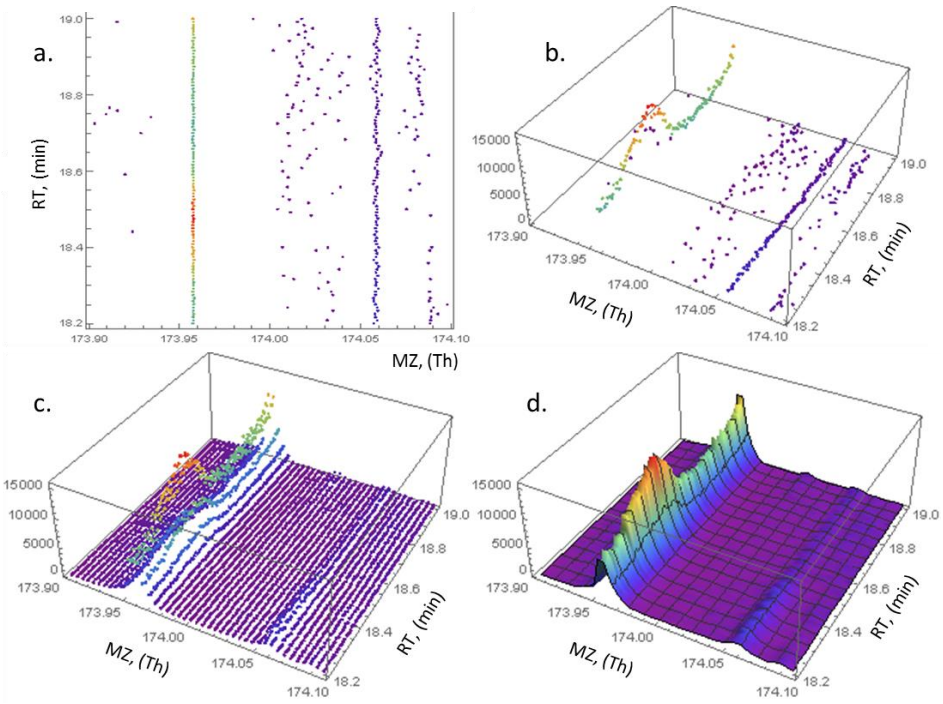

Figure 4. Four different views of MZ-RT area data: a. MZ-RT 2d space with color-coded intensity points of detected peaks (centroided mode); b. points of detected peaks in MZ-RT-Intensity 3d space; c. points of measured intensities (profile mode) in MZ-RT-Intensity 3d space; d. measured intensities fitted to surface in MZ-RT-Intensity 3d space

MZ-RT areas can be used for a deeper analysis of LC-MS data, for example, tracing MZ errors over time, studying dependence between MZ error and intensity. MZ-RT areas can also be requested in profile mode (if supported by the underlying instrument data file), which can be useful to investigate how signals vary across the RT dimension, for example suppression of weak signals by a more intense peak nearby in the MZ dimensions. When analyzing or displaying MZ-RT area profile data, it have to be taken into account that measured MZ values are fluctuating over spectra, and, therefore, does not provide regular grid in MZ dimension.

3.4 Fragmentation events

The function GetFragmentationEvents provides information on ions from given RT-MZ area that were selected for fragmentation by the instrument. For each fragmentation event in the area, one instance of the `FragmentationInfo` structure is provided. This data structure is described below in C# format (see also the WDSL service description).

```
public class FragmentationInfo {
    public double RT;
    public double ParentMZ;
    public int ScanNumber;
    public int MSOrder;
    public string Description;
}
```

The fields of this structure are interpreted as follows:

RT: the exact retention time when fragment mass spectra was acquired

ParentMZ: the MZ for the parent ion of interest.

ScanNumber: the scan number identifying the generated fragmentation spectrum.

MSOrder: the fragmentation order. This equals 2 for a fragmentation of an MS^1 ion, 3 for a fragmentation of an MS^2 ion, and so on.

Description: a free text description of the fragmentation spectrum, if available from the instrument file.

Given this information, a fragmentation spectrum can be obtained using the GetSpectrumByScanNumber function. The fragmentation spectrum holds information about chemical structure of selected ions and can be used for identification of compounds. With this information, you can select spectra, which is most relevant for compound. For example, you can select fragment spectrum taken close to the top of elution profile as it shown on the picture.

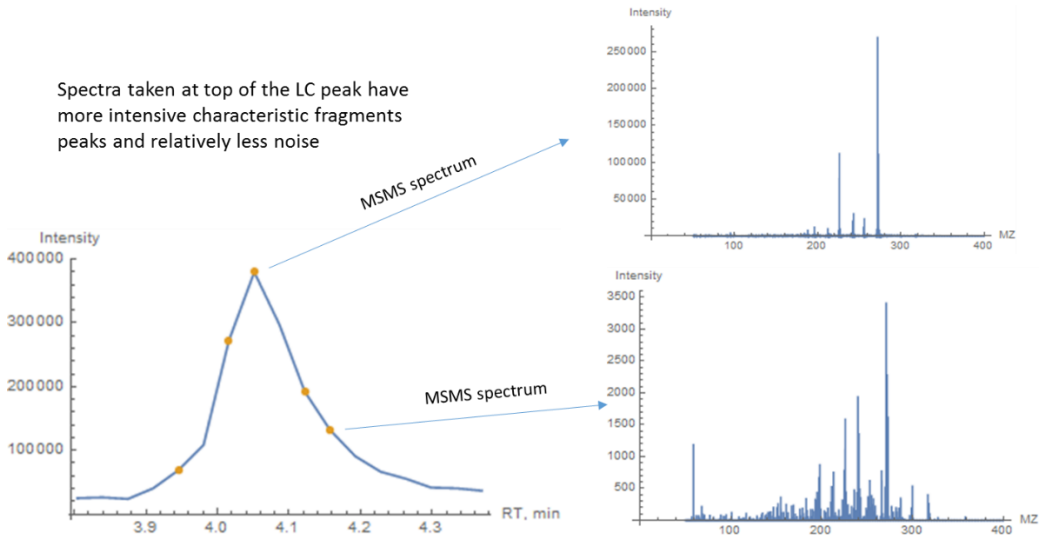

Figure 5. Chromatogram with fragmentation events fitted to the chromatograms curve and examples of fragment spectra taken at different point of chromatogram curve.

4 Web service conventions

This section describes common conventions used when calling the web service.

4.1 Common parameters

These parameters are commonly used throughout the web service, with the following interpretations.

FileName: the name of the data file from which to retrieve data (a text string). Stated without path and extension. Hence, the file name alone (without extension) *must be unique* among all files on an mzAccess server; it is not possible to keep files with the same names in subfolders.

MZLow: the lowest value of m/z to be considered.

MZHigh: the highest value of m/z to be considered.

RTLow: the lowest value of retention time to be considered.

RTHigh: the highest value of retention time to be considered.

ErrorMessage: an out parameter that will contain an error message (text string) after the call if an error occurred. If no error occurs, this parameter is set to null. See section 4.3.

Profile: an optional Boolean logic value. If set to `true`, profile mode data will be returned, provided that this is supported by the underlying data file; otherwise, an error is returned. If set to `false` or not specified, centroid mode data will be returned. See also section 3.2.1.

Cache: an optional Boolean logic value. If set to `true`, or not specified, data will be retrieved using the caching mechanism. If set to false, primary instrument data will be used. If the `Profile` parameter is set to `true`, this parameter is overridden, since profile mode data is only available from primary data files.

ScanNumber: an integer that identifies a spectrum.

4.2 Batch functions

Batch functions allow retrieving data from multiple files and/or MZ and RT ranges in a single call. They are typically much faster than performing multiple calls to the corresponding “basic” function, as they take advantage of optimizations made for this purpose in the caching mechanism, and also reduce network overhead. All batch functions require for each parameter an array instead of a single value; each element of this array corresponds to the single parameter value of the “basic” function, and return an array of arrays, each corresponding to the result of the “simple” function.

For example, instead of calling the “basic” `GetChromatogram` function N times to get chromatograms at a given MZ and RT window from a set of N files, one may call the batch function `GetChromatogramArray`, providing an array `[FileName1, FileName2, ..., FileNameN]` of the relevant file names, and corresponding arrays or N elements for the other parameters (in this case, they will be arrays of repeated values). If instead chromatograms for a range of MZ values are desired (for example, for a set of mass isotopomers of a compound), the mzLow and mzHigh parameters should be arrays filled accordingly.

As batch function considered to provide full set of parameters for every single operation, the length of incoming array parameters must be equal, otherwise exception will be thrown

4.3 Exception Handling

If an exception occurs, the mzAccess server will report on the error by placing a string in the `ErrorMessage` out parameter; otherwise, this parameter will be set to `null`. *The caller should test for an exception* by checking if `ErrorMessage` equals `null`. If an exception has occurred, the results of the call are undefined and should be discarded.

All the exception contains name of the function called and necessary complement information

The following strings can occur in the `ErrorMessage` out parameter:

`"File exception: ..."`

If requested file has not been found at start time of the server, it will not be accessible with service. Error message will contain requested file name.

`"Argument exception: ..."`

If length of arrays in Array calls is not equal.

`"DataUnavailable exception: …"`

Occurs when user ask for data not available in certain files. Some examples are: Profile or MSMS data from cashe, profile data from raw files where it has not been saved, centroid data from mzML if mzML contains requested spectra in profile mode.

`"UnsupportedFeature exception: …"`

Occurs when user ask for process currently not supported, like average spectra of cache or mzML files, or when data files contains unsupported kinds of information like multiple precursor selection.

`"RawFile exception: …"`

Occurs when there is a problem with loading of data files, like inconsistent mzML or broken raw file.

`"Uncategorized exception: ... "`

Occurs in case of any other problems on the server side. Error message will contain exception message, full list of parameters and exception stack information. In addition, this message will be written to server application log.

4.4 Cache mode and performance notes

As described in the main article, cache mode provides fast access to chromatogram and MZ-RT area data in centroid mode. In this mode, the mzAccess server makes use of an optimized file format where only MZ peaks above a certain threshold value are retained. These MZ peaks are defined by vendor's peak detection algorithm. The details may vary between mzAccess server implementations and vendor plug-ins; for more information, see the mzAccess server documentation. Most of the functions in web service can be called in cache mode, by setting the `Cache` parameter to `true`. This is also the default setting.

Cache mode has been designed to be efficient for retrieving chromatograms and MZ-RT areas with a narrow MZ range, and should be used routinely with these data types, but is less efficient when requesting chromatograms or areas over wide MZ range (say, >5 Da on an orbitrap instrument). Cache mode does not provide notable benefits when retrieving spectra.

Using batch calls provides an additional performance gain over sequential calls to "basic" functions, as they eliminate the effect of network latency and make more efficient use of network bandwidth. Moreover, batch calls in cache mode are optimized for extracting the same MZ ranges from multiple files, for example when investigating a single compound across many samples analyzed in some experiment, and should always be used for such repetitive data access.

Finally, the mzAccess web service also supports asynchronous calls. If your client programming environment supports asynchronous calls to web services, you may want to consider this option, particularly for slower network connections.

5 Function reference

This section describes the functions available in the web service in the C# format. Language-independent function information in WDSL format can also be obtained at any time directly from the published web service of an mzAccess server.

5.1 GetChromatogram

```
public double[] GetChromatogram(string FileName, double MZLow, double MZHigh, double
RTLow, double RTHigh, out string ErrorMessage, bool Cache = true);
```

Description: retrieves a chromatogram. The intensity value at each RT in this chromatogram depends on the instrument used, and may not agree with those obtained in corresponding spectra or RT/MZ areas. With `Cache = true`, if there are more than one signal in specified MZ range, the highest intensity signal will be kept.

Returns: a flat array of chromatogram data, as described in section 3.1.

5.2 GetArea

```
public double[] GetArea(string FileName, double MZLow, double MZHigh, double RTLow,
double RTHigh, out string ErrorMessage, bool Cache = true, bool Profile = false);
```

Description: retrieves all intensity data for the specified MZ-RT area.

Returns: a flat array of area data, as described in section 3.3. Empty array , if there is no data in requested area.

5.3 GetSpectrumByScanNumber

```
public double[] GetSpectrumByScanNumber(string FileName, double MZLow, double MZHigh,
int ScanNumber, out string ErrorMessage, bool Cache = false, bool Profile = false);
```

Description: retrieves a spectrum identified by the `ScanNumber` parameter

Returns: Flat array of data in form of `[Mass1; Intensity1; Mass2; Intensity2;…; MassN; IntensityN]` as described in section 3.2. Empty array, if there is no data in requested area.

Remarks: It is only the function, which is able to return MS^2, MS^3 and higher order MS spectra from original raw files. Since cache does not provide performance here by default it set to false.

5.4 GetSpectrumByRT

```
public double[] GetSpectrumByRT(string FileName, double MZLow, double MZHigh, double RT,
out string ErrorMessage, bool Cache = false, bool Profile = false);
```

Description: Gets Mass/Intensity pairs for particular MS^1 spectra identified by Retention Time.

Returns: Last spectrum acquired at the time less or equal to RT parameter. Spectra is in form of flat array of data in form of `[Mass1; Intensity1; Mass2; Intensity2;…; MassN; IntensityN]` as described in section 3.2. Empty array, if there is no data in requested area.

5.5 GetAverageSpectrum

```
public double[] GetAverageSpectrum(string FileName, double MZLow, double MZHigh, double
RTLow, double RTHigh, out string ErrorMessage, bool Profile = false);
```

Description: Gets Mass/Intensity pairs for averaged (summed) spectra for Retention Time range calling underlying function of instrumental API.

Returns: Flat array of data in form `of [Mass1; Intensity1; Mass2; Intensity2;…; MassN; IntensityN]`. Empty array, if there is no data in requested area.

Remarks: Have no option for cached files, since averaging/summing procedure is Instrumental API depended and often relies on profile data. Resulting intensities can differ from what have been found with single spectra or chromatogram function.

5.6 GetFragmentationEvents

```
public FragmentationInfo[] GetFragmentationEvents(string FileName, double MZLow, double MZHigh, double RTLow, double RTHigh, out string ErrorMessage)
```

Description: Describe fragmentation events, which occur in requested LC-MS area.

Returns: Array of `FragmentationInfo` structures (see section 4.4), containing information about fragmentation events.

Remarks: MSMS spectra are not cached, therefore they are available only from vendor's format files. To get MSMS spectra call GetSpectrabyScanNumber function with provided Scan Number.

5.7 GetChromatogramArray

```
public double[][] GetChromatogramArray(string[] FileNames, double[] MZLow, double[] MZHigh, double[] RTLow, double[] RTHigh, out string ErrorMessage, bool Cache = true)
```

Description: Batch analog of GetChromatogram function. Gets Retention time/Intensity pairs for particular parameters calling underlying function of instrumental API. Equal length of incoming array is assumed.

Returns: Array of flat arrays of data in form `of [RT1; Intensity1; RT2; Intensity2;…; RTN; IntensityN]` as described in section 3.1.. If error occurs in any of the file operations, exception will be generated

5.8 GetSpectrumArray

```
public double[][] GetSpectrumArray(string[] FileNames, double[] MZLow, double[] MZHigh, double[] RTLow, double[] RTHigh, out string ErrorMessage, bool Profile = false);
```

Description: Batch analog of GetAverageSpectrum function. Gets Mass/Intensity pairs for averaged (summed) spectra for retention time range calling underlying function of instrumental API. Equal length of incoming array is assumed.

Returns: Returns array of flat arrays of data in form of `[Mass1; Intensity1; Mass2; Intensity2;…; MassN; IntensityN]` as described in section 3.2. If error occurs in any of the file operations, exception will be generated.

Remarks: Averaging/summing procedure is Instrumental API depended. Resulting Intensities can differ from what have been found with single spectra or chromatogram function.

5.9 GetAreaArray

```
public double[][] GetAreaArray(string[] FileNames, double[] MZLow, double[] MZHigh, double[] RTLow, double[] RTHigh, out string ErrorMessage, bool Cache = true , bool Profile = false);
```

Description: Batch analog of GetArea function. : Gets [Mass/Retention time/Intensity] triplets for particular parameters. Internally, function collects the data calling single spectra function of instrumental API for all the spectra in desired range of retention times and filtering them according to desired mass range. Equal length of incoming array is assumed.

Returns: Returns array of flat arrays of data in form `of [Mass1; RT1; Intensity1; Mass2; RT2; Intensity2;…; MassN; RTN; IntensityN]` as described in section 3.3. If error occurs in any of the file operations, exception will be generated.

5.10 GetScanNumberFromRT

```
public int GetScanNumberFromRT(string FileName, double RT, out string ErrorMessage, bool Cache = false)
```

Description: Gets scan number for of nearest MS^1 spectrum with less or equal retention time

5.11 GetRTFromScanNumber

```
public double GetRTFromScanNumber(string FileName, int ScanNumber, out string ErrorMessage, bool Cache = false);
```

Description: Gets exact retention time for particular scan number.

Returns:

5.12 GetMZRange

```
public double[] GetMZRange(string FileName, out string ErrorMessage);
```

Description: Get full mass range for specified file name.

Returns: array of two doubles `[LowMZ, HighMZ]`, or `null` if exception has been thrown.

5.13 GetRTRange

```
public double[] GetRTRange(string FileName, out string ErrorMessage);
```

Description: Get full retention time range for specified file name

Returns: array of two doubles `[LowRT;HighRT]` or `null` if exception has been thrown.