

MassChroQ library documentation

version: 3.0

Olivier Langella

Contents

1	Introduction	3
2	Input format	3
3	MassChroQ methods	5
4	Output format	8
5	MSrun retention time alignment	11
6	Peak quality code	11
7	Match between run difference with legacy MassChroQ	12

Introduction

MassChroQ is designed to extract ion current of a list of peptide and measure the area under the curve on a single run.

Input format

input file format

```
{
  "project_parameters": {
    "i2MassChroQ_VERSION": {
      "category": 1,
      "value": "1.0.18"
    },
    "cvparam_SpectraDataFileFormat": {
      "category": 5,
      "value": "MS:1000544"
    },
    "cvparam_SpectraDataSpectrumIDFormat": {
      "category": 5,
      "value": "MS:1000824"
    },
    "AnalysisSoftware_name": {
      "category": 2,
      "value": "X!Tandem"
    }
  },
  "masschroq_methods": {
    "alignment_method": {
      "ms1_smoothing": 15,
      "ms2_smoothing": 5,
      "ms2_tendency": 10
    },
    "quantification_method": {
      "match_between_run": true,
      "isotope_minimum_ratio": 0.9,
      "extraction": {
        "rt_range": 300,
        "integration": "max",
        "precision": {
          "unit": "ppm",
          "up": 10,
          "down": 10
        }
      }
    },
    "prefilter": {},
    "detection": {
      "type": "zivy",
      "meanfilter": 1,
      "minmax": 3,
      "maxmin": 2,
      "threshold_on_max": 5000,

```

```
        "threshold_on_min": 3000
      }
    }
  },

  "identification_data": {
    "msrun_list": {
      "msrunb19": {
        "file": "/gorgone/pappso/data_extraction_pappso/
mzXML/20120906_balliau_extract_1_B08_teal-5.mzXML"
      }
    },
    "protein_list": {
      "protal1": {
        "description": "P02769|ALBU_BOVIN SERUM ALBUMIN PRECURSOR",
        "sequence": "SLTNDWEDHLAVK"
      }
    },
    "peptide_list": {
      "pepal1": {
        "proforma": "SLTNDWEDHLAVK",
        "proteins": ["protal1"],
        "mods": "free text",
        "label_list": { "light": { "proforma": "SLTNDWEDHLAVK" } }
      }
    },
    "msrunpeptide_list": {
      "msruna1": {
        "peptide_obs": {
          "pepal1": [
            {
              "scan_index": 2345,
              "label": "light",
              "precursor": {
                "charge": 2,
                "mz": 2456.45,
                "intensity": 4580,
                "rt": 345.67
              }
            }
          ]
        }
      }
    },
    "actions": {
      "group_list": { "g1": ["msruna1", "msruna2", "msruna3"] },
      "align_group": {
        "g1": {
          "alignment_reference": "msruna1"
        }
      }
    }
  },
```

```
    }
  },
  "quantify_all": true
}
```

2.1 JSON input root file structure

project_parameters (object) *Optional* Controlled vocabulary summarising important parameters of the project.

masschroq_methods (object) *Required* This object contains all required parameters for the MassChroQ engine, see the [Section 3](#) section

identification_data (object) *Required*

actions (object) *Required*

MassChroQ methods

masschroq_methods JSON object

```
{
  "alignment_method": {},
  "quantification_method": {}
}
```

alignment_method (object) *Required* Describes how to align retention time between MS runs, see [Section 3.1](#)

quantification_method (object) *Required* Describes how MassChroQ extracts ion current, detects peaks and then quantify ion intensities, see [Section 3.2](#)

3.1 Alignment method

alignment_method JSON object

```
{
  "ms1_smoothing": 15,
  "ms2_smoothing": 5,
  "ms2_tendency": 10
}
```

3.2 Quantification method

quantification_method JSON object

```
{
  "match_between_run": true,
  "isotope_minimum_ratio": 0.9,
  "extraction": {
    "rt_range": 300,
    "integration": "max",
    "precision": {
      "unit": "ppm",
      "up": 10,
      "down": 10
    }
  }
}
```

```

},
"prefilter": {},
"detection": {
  "type": "zivy",
  "meanfilter": 1,
  "minmax": 3,
  "maxmin": 2,
  "threshold_on_max": 5000,
  "threshold_on_min": 3000
}
}

```

match_between_run (*boolean*) *Required* Boolean, true if a match between run is required, false otherwise

isotope_minimum_ratio (*float*) *Required* Ratio between 0 and 1 (not included): the ratio of minimum intensity of the whole theoretical isotopic pattern that must be reached. If the value is 0, then only the most theoretically intense isotope will be extracted and quantified. If the value is 0.8, it selects isotopes from the most intense to the least intense and stops when 80% of the whole theoretical intensity is reached.

extraction (*object*) *Required* Describes how the ion current is extracted at a given m/z and retention time.

prefilter (*object*) *Required* Signal process to apply on ion extracted current before the peak detection. Can be empty.

detection (*object*) *Required* Peak detection method (see [Section 3.2.1](#)) to use. This is important to separate real peaks from artefacts and to accurately determine peak boundaries.

3.2.1 Peak detection method

quantification_method JSON object

```

{
  "type": "zivy",
  "meanfilter": 1,
  "minmax": 3,
  "maxmin": 2,
  "threshold_on_max": 5000,
  "threshold_on_min": 3000
}

```

The default detection method called “zivy” differentiates real MS1 peaks from noise by performing a morphology algorithm¹. As shown in [Figure 1](#), it uses two curves:

- “minmax” is the closing curve²
- “maxmin” is the opening curve³

“minmax” minmax in detection is used to determine the boundary of the peaks : where it starts and where it ends and to validate the peak with threshold_on_max in detection.

“maxmin” maxmin in detection is used to validate the peak as a true MS1 peak using threshold_on_min in detection.

¹https://en.wikipedia.org/wiki/Mathematical_morphology

²[https://en.wikipedia.org/wiki/Closing_\(morphology\)](https://en.wikipedia.org/wiki/Closing_(morphology))

³[https://en.wikipedia.org/wiki/Opening_\(morphology\)](https://en.wikipedia.org/wiki/Opening_(morphology))

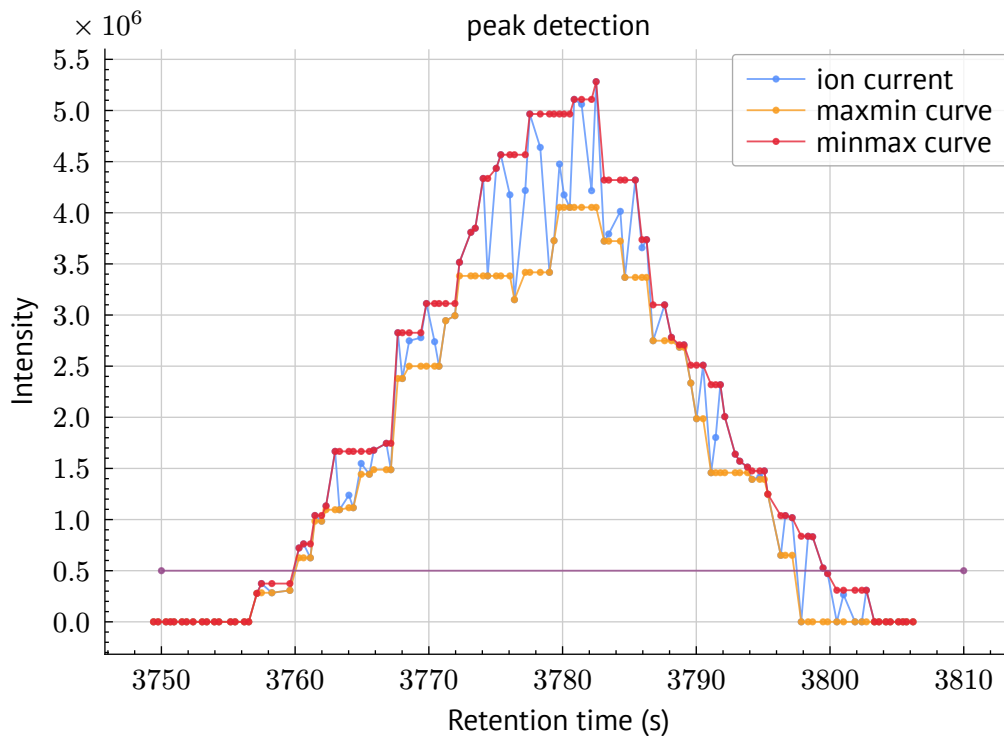


Figure 1: peak detection using minmax and maxmin filters

A peak is considered only when the `threshold_on_max` in `detection` is reached on “minmax” **and** `threshold_on_min` in `detection` is reached on “maxmin”.

type (*string*) *Required* The detection method to use. Only “zivy” is available in MassChroQ 3.

meanfilter (*integer*) *Required* Applies a mean filter on the signal before processing the minmax and maxmin process. The mean is computed in a window of width :

- $\text{meanfilter} \cdot 2$ for odd values
- $(\text{meanfilter} \cdot 2) + 1$ for even values.

The peak area is always computed on the original signal, not the signal after the mean filter. But keep in mind that this filter must be used only on very noisy signals, with extremely high random variations.

Do not use a value more than 1, and if the signal is stable, prefer a 0 value (no mean filter at all).

minmax (*integer*) *Required* Computes the “closing” signal to fit the upper shape of the peak, filling random noise (see Figure 1). It uses a window of width :

- $\text{minmax} \cdot 2$ for odd values
- $(\text{minmax} \cdot 2) + 1$ for even values.

maxmin (*integer*) *Required* Computes the “opening” signal to fit the lower shape (from below) of the peak, filling random noise (see Figure 1). It uses a window of width :

- $\text{maxmin} \cdot 2$ for odd values
- $(\text{maxmin} \cdot 2) + 1$ for even values.

threshold_on_max (*integer*) *Required* Intensity threshold of the maxmin in detection.

threshold_on_min (*integer*) *Required* Intensity threshold of the minmax in detection.

CSV input file format

peptide proformat, charge, retention time

SLTNDWEDHLAVK, 2, 853.78

SLTNDWEDHLAVK, 3, 853.78

Output format

Wouldn't be nice to try CBOR ? CBOR is supported in QT6

It could be something liket that :

output file format

```
{
  "informations": {
    "cpu_used": 8,
    "executable": "mcql",
    "masschroq_version": "3.0.1",
    "pappsomsp_version": "0.9.46",
    "sysinfo_machine_hostname": "proteus1",
    "sysinfo_product_name": "Debian GNU/Linux 12 (bookworm)",
    "timestamp": "2025-06-25T10:33:51"
  },
  "alignment_data" : [
    {
      "alignment_id": "a1",
      "group_id": "g1",
      "timestamp1": "ISODate"
      "alignment": {
        "msrun_ref": "msruna2",
        "corrections": {
          "msrunal": {
            "original": [1,2,3,4.5],
            "aligned": [1,2,3,4.5],
            "ms2_delta_rt": {
              "x": [123, 145, 157, 189],
              "y": [0.2354, 2.2345, -4.23, -3.875]
            },
            "ms2_median": [1,2,3,4.5],
            "ms2_mean": [1,2,3,4.5]
          },
          "msruna2": {
            "original": [1,2,3,4.5],
            "aligned": [1,2,3,4.5]
          }
        }
      },
      "timestamp2": "ISODate"
    }
  ],
  "quantification_data" : [
    {
      "timestamp1": "ISODate",
      "quantify_id": "q1",
      "group_id": "g1",
      "first_pass": {"msrunal": "qr_data_block", "msruna2": "qr_data_block"},
    }
  ]
}
```



```

        "timestamp2": "ISODate",
        "second_pass": {"msrun1": "qr_data_block", "msrun2": "qr_data_block"},
        "timestamp3": "ISODate"
    }
],
"end":{"timestamp":"2025-01-27T15:30:11"}
}

```

4.1 QrDataBlock element

QrDataBlock stands for **quantification run data block** it handles quantification data for a single MSrun.

qr_data_block element format

```

{
  "msrunal": {
    "msrun": {
      "id": "msrunal",
      "filename": "/gorgone/truc",
      "sample": "échantillon"
    },
    "peptide_measurements": {
      "pepalal": {
        "proforma": "SLTNDWEDHLAVK",
        "mods": "free text",
        "rt_target": 853.78,
        "xics": [
          {
            "mz": 764.3755373,
            "xic_coord": {
              "mz_range": [764.344, 765.56666],
              "ims_im_index": [150, 175]
            },
            "charge": 2,
            "isotope": 0,
            "rank": 1,
            "th_ratio": 0.568912,
            "quality": "a",
            "label": "light",
            "trace": {
              "x": [851.78, 852.78, 853.78, 854.78],
              "y": [851.78, 852.78, 853.78, 854.78]
            },
            "peak_shape": {
              "trace": {
                "x": [851.78, 852.78, 853.78, 854.78],
                "y": [851.78, 852.78, 853.78, 854.78]
              }
            },
            "peak": {
              "area": 450245623,
              "max_intensity": 2345.456,
              "rt": [851.78, 852.78, 853.78],

```

```

        "aligned_rt": [851.78, 852.78, 853.78]
    },
    {
        "mz": 764.8769,
        "charge": 2,
        "isotope": 1,
        "quality": "a",
        "trace": {
            "x": [851.78, 852.78, 853.78, 854.78],
            "y": [851.78, 852.78, 853.78, 854.78]
        },
        "peak": {
            "area": 550245623
        }
    }
]
},
"pepala2": {}
}
}

```

output file format

```

{
    "informations": {
        "cpu_used": 8,
        "executable": "mcql",
        "masschroq_version": "3.0.1",
        "pappsomsp_version": "0.9.46",
        "sysinfo_machine_hostname": "proteus1",
        "sysinfo_product_name": "Debian GNU/Linux 12 (bookworm)",
        "timestamp": "2025-06-25T10:33:51"
    },
    "alignment_data" : [
        {
            "alignment_id": "a1",
            "group_id": "g1",
            "timestamp1": "ISODate"
            "alignment": {
                "msrun_ref": "msruna2",
                "corrections": {
                    "msrunal": {
                        "original": [1,2,3,4.5],
                        "aligned": [1,2,3,4.5],
                        "ms2_delta_rt": {
                            "x": [123, 145, 157, 189],
                            "y": [0.2354, 2.2345, -4.23, -3.875]
                        },
                        "ms2_median": [1,2,3,4.5],
                        "ms2_mean": [1,2,3,4.5]
                    },
                },
            },
        },
    ],
}

```

```

        "msruna2": {
            "original": [1,2,3,4.5],
            "aligned": [1,2,3,4.5]
        }
    },
    "timestamp2": "ISODate"
}
],
"quantification_data" : [
    {
        "timestamp1": "ISODate",
        "quantify_id": "q1",
        "group_id": "g1",
        "first_pass": {"msruna1": "qr_data_block", "msruna2": "qr_data_block"},
        "timestamp2": "ISODate",
        "second_pass": {"msruna1": "qr_data_block", "msruna2": "qr_data_block"},
        "timestamp3": "ISODate"
    }
],
"end": {"timestamp": "2025-01-27T15:30:11"}
}

```

MSrun retention time alignment

MS run retention time function

```

std::shared_ptr<pappso::MsRunRetentionTime<QString>> &
mcql::MsRunPeptideList::buildMsRunRetentionTimeSp(const mcql::AlignmentMethodSp
&alignment_method)

```

Peak quality code

- aa** best quality : many MS2 fragmentation event, only one peak directly detected
- zaa** same as aa, but this charge state was not directly observed in MS2 fragmentation events in this MSrun
- a** good quality, single MS2 fragmentation event, one peak detected
- za** same as a, but this charge state was not directly observed in MS2 fragmentation events in this MSrun
- ab** many MS2 fragmentation event, but more than one peak detected, the greater peak (area) is chosen, it is obviously fragmented... perhaps a hint to check for peak detection parameters
- zab** same as ab, but this charge state was not directly observed in MS2 fragmentation events in this MSrun
- b** peak obtained by “match between run” on the mean aligned observed retention times in MS2 fragmentation events **and** also matching with the retention time given by other detected and quantified MS1 apex peaks in other MS runs
- c** peak obtained by “match between run” only on the mean of aligned observed retention times in MS2 fragmentation events

d peak obtained by “match between run” only matching with the retention time given by other detected and quantified MS1 apex peaks in other MS runs

missed no peak detected, no quantification

Match between run difference with legacy

MassChroQ

The match between run process behaviour is slightly different between legacy MassChroQ and MassChroQ3. The new process is more conservative as it will not try to find different peptide charge state if the peptide was observed for an MSrun in an other charge state.

An important difference is that MassChroQ3 reports missed peaks : this lead to a natural data cleaning when used with MCQR. Indeed, if the most theoretical abundant isotope is not found in **any** msruns, then it will be discarded. This lead to less quantified peptides but with a better quality. indentation