OpenMS / KNIME

Oliver Alka[4], Timo Sachsenberg[4], Leon Bichmann[4], Julianus Pfeuffer[4,10], Hendrik Weisser[11], Samuel Wein[9], Eugen Netz[5], Marc Rurik[4], Oliver Kohlbacher [4,5,6,7,8], Hannes Röst[1,2,3]

1 Donnelly Centre, University of Toronto, Toronto, Canada
2 Department of Molecular Genetics, University of Toronto, Toronto, Canada
3 Department of Computer Science, University of Toronto, Toronto, Canada
4 Department for Computer Science, Applied Bioinformatics, University of Tübingen, Sand 14, 72076 Tübingen, Germany
5 Biomolecular Interactions, Max Planck Institute for Developmental Biology, Max-Planck-Ring 5, 72076 Tübingen, Germany
6 Institute for Translational Bioinformatics, University Hospital Tübingen, Hoppe-Seyler-Str. 9, 72076 Tübingen, Germany
7 Institute for Biomedical Informatics, University of Tübingen, Sand 14, 72076 Tübingen, Germany
8 Quantitative Biology Center, University of Tübingen, Auf der Morgenstelle 10, 72076 Tübingen, Germany
9 Epigenetics Institute, Department of Cell and Developmental Biology, University of Pennsylvania, 9th Floor, Smilow Center for Translational Research 3400 Civic Center Blvd Philadelphia, PA 19104, USA
10 Department for Computer Science, Algorithmic Bioinformatics, Freie Universität Berlin, Takustr. 9, 14195 Berlin, Germany
11 STORM Therapeutics Limited, Moneta Building, Babraham Research Campus, Cambridge CB22 3AT, United Kingdom

*Corresponding contributor. E-mail: hannes.rost@utoronto.ca

Introduction

Computational mass spectrometry has seen exponential growth in recent years in data size and complexity, straining the existing infrastructure of many labs as they moved towards high-performance computing (HPC) and embraced big data paradigms. Transparent and reproducible data analysis has traditionally been challenging in the field due to a highly heterogeneous software environment, while algorithms and analysis workflows have grown increasingly complex. A multitude of competing and often incompatible file formats prevented objective algorithmic comparisons and, in some cases, access to specific software or file formats relied on a vendor license. Due to the fast technology progress in the field, many novel algorithms are proposed in the literature every year, but few are implemented with reusability, robustness, cross-platform compatibility and user-friendliness in mind, creating a highly challenging software and data storage environment that in some aspects is even opaque to experts.

The OpenMS software framework addresses these issues through a set of around 175 highly robust and transparent cross-platform tools with a focus on maximal flexibility (Pfeuffer et al. 2017; Röst et al. 2016; M. Sturm et al. 2008). Modern software engineering techniques ensure reproducibility between versions and minimize code duplication and putative errors in software. OpenMS is completely open-source, uses standardized data formats extensively and is available on all three major computing platforms (macOS, Windows, Linux). Multiple layers of access to the OpenMS algorithms exist for specialist, intermediate and novice users, providing low entrance barriers through sophisticated data visualization and graphical workflow managers.

The flexibility of OpenMS allows it to support a multitude of customizable and easily transmissible workflows in multi-omics data analysis, including metabolomics, lipidomics, and proteomics setups, supporting different quantitative approaches spanning label-free, isotopic, isobaric labeling techniques, as well as targeted proteomics. Its highly flexible structure and layered design allow different scientific groups to take full advantage of the software. Developers can fully exploit the sophisticated C++ library for tool and data structure development, while advanced Python bindings (pyOpenMS) wrap most of the classes (Röst, Schmitt, et al. 2014), providing an excellent solution for fast scripting and prototyping. Users, can either work on command line tool level or take advantage of industry-grade workflows system, such as the KoNstanz Information MinEr (KNIME) (Berthold et al. 2007; Fillbrunn et al. 2017), Galaxy (Afgan et al. 2018), Nextflow (Di Tommaso et al. 2017), or Snakemake (Koster and Rahmann 2012). The framework is highly adaptable, allowing even novice users to generate complex workflows using the easy-to-learn graphical user interfaces of KNIME. Built-in support for most common workflow steps (such as popular proteomics search engines) ensures low entrance barriers while advanced users have high flexibility within the same framework. A modular and comprehensive codebase allows rapid development of novel methods as exemplified by the recent additions for metabolomics, SWATH-MS and cross-linking workflows.
In addition, a versatile visualization software (TOPPView) allows exploration of raw data as well as identification and quantification results (Marc Sturm and Kohlbacher 2009). The permissive BSD license encourages usage in commercial and academic projects, making the project especially suited for reference implementations of file formats and algorithms.

OpenMS for developers

The OpenMS framework consists of different abstraction layers. The first layer consists of external libraries (Qt, Boost, Xerces, Seqan, Eigen, Wildmagic, Coin-Or, libSVM), which add additional data structures and functionality, simplifying complex tasks such as GUI-programming or XML parsing. The next layer consists of the OpenMS core library containing algorithms, data structures and input/output processing. The third layer encloses TOPP tools and utilities (~ 175), which allow various analysis tasks, such as signal processing, filtering, identification, quantification and visualization. The core library and the TOPP tools have Python bindings, which can be used for fast scripting and prototyping (pyOpenMS) (Röst, Schmitt, et al. 2014). In the top layer, the tools are accessible from different workflow systems, for the construction of flexible, tool based workflows.

C++ library

OpenMS has a multi-level architecture with an open source C++ library at its core which implements all data structures and algorithms required for MS data analyses. Its permissive license (3-clause BSD) allows using OpenMS both in academic as well as commercial projects without any licensing fees. Since its beginning, its aim has been to provide efficient data structures and algorithms for common MS data processing tasks. As such, the library targets computational biologists and algorithm developers with sound programming skills. Using the library, developers have direct access to the richest set of functionality and generally profit from highly-optimized implementations of core data structures and algorithms when developing novel methods. More than 1,300 classes cover a broad range of functions in computational mass spectrometry, proteomics and metabolomics. These functionalities include

- file handling (mzXML, mzML, TraML, mzTab, fasta, pepxml, protxml, mzIdentML among others)
- chemistry (mass calculation, peptide fragmentation, isotopic abundances)
- signal processing and filtering (smoothing, filtering, de-isotoping, mass correction, retention time correction and peak picking)
- identification analysis (including peptide search, PTM analysis, cross-linked analytes, FDR control, RNA oligonucleotide search and small molecule search tools)
- quantitative analysis (including label-free, SILAC, iTRAQ, TMT, SWATH/DIA, and metabolomics analysis tools)
- chromatogram analysis (chromatographic peak picking, smoothing, elution profiles and peak scoring for targeted (SRM, MRM, PRM, SWATH, DIA) data)
- interaction with common tools in proteomics and metabolomics
    - search engines such as Comet, Crux, Mascot, MS-GF+, InsPecT, PepNovo, MSFragger, Myrimatch, OMSSA, Sequest, SpectraST, XTandem
    - post-processing tools such as Percolator, MSstats, Fido, EPIFANY
    - metabolomics tools such as SIRIUS

Ideally, newly developed methods, algorithms, or data structures of general interest are contributed by the community, find their way back to the library to be used by other OpenMS developers. OpenMS itself builds on other open-source projects like Qt, Xerces, COIN-OR, libSVM, Eigen, WildMagic, or boost for tasks like GUI programming, XML parsing, non-linear optimizations, machine learning or fast linear algebra. To leverage the compute power of modern processors, the OpenMP library is used to parallelize many algorithms in OpenMS.
For a short example how a multi-threaded spectrum processing algorithm can be realized using the OpenMS library see Code Example 1.

Code Example 1: Multi-threaded spectrum processing algorithm using the OpenMS library

```
// C++ example (excerpt):
// Load data, retain the 400 most intense peaks in a spectrum

MSExperiment exp;
MzMLFile().load(“file.mzML”, exp);
auto spectra = exp.getSpectra();

// construct a spectrum filter
NLargest nlargest_filter = NLargest (400);

// parallelize loop for concurrent execution using OpenMP


# pragma omp parallel for
for (auto it = spectra.begin(); it < spectra.end(); it++)
{
  // sort peaks by mass-to-charge position
  it->sortByPosition();

  // apply filter and keep only the 400 highest intensity peaks
  nlargest_filter.filterPeakSpectrum(*it);
}
```

Data formats and raw data API

Using open data formats for storing, exchanging, and deposition of primary raw data and final analysis results in public data repository are prerequisites for reproducible science. OpenMS builds on open standards for reading and writing MS data (e.g., mzML and mzXML format), transitions (traML), identifications (mzIdentML), or exporting final results (mzTab). In fact, members of the OpenMS community have been actively involved in the development of several HUPO-PSI standard formats in the past. For XML-based formats, OpenMS uses the powerful Xerces software library from the Apache project which implements both DOM and SAX parsing. For all data, in-memory data structures exist where data gets parsed from disk into these data structures, manipulated in memory and then written to disk again after transformation / manipulation. However, for raw MS data access in mzML files, OpenMS provides an advanced API that can handle access spectra and chromatograms in different ways, thus optimizing the tradeoff between memory and CPU requirements of the algorithm (Röst et al. 2015): (1) random access in memory, (2) random access on disk using indexedmzML, (3) random access on disk using a cached file format and (4) event-driven processing. While programmatically, it is easiest to program against interface (1) since all data is in memory and can be accessed very fast, this is not always possible for very large files that do not fit into the current computer’s memory. For these cases, it is possible to access spectra from disk using either mechanism (2) or (3) which allows accessing raw data that is not held in memory. Finally, we have implemented an interface that uses a call-back mechanism similar to SAX parsing in XML parsing, which works on a per-spectra basis: the call-back function gets called as soon as a new spectrum is available from the reader, allowing a consumer to process spectra as they get read from disk. The processing function can either keep all results in memory or write them back to disk such that in an ideal case only a single spectrum is in memory at any given time.

Algorithms

The OpenMS library provides a multitude of algorithms ranging from low-level algorithms, like the generation of isotope patterns, processing and filtering of raw signals, to more complex algorithms like peptide database search. In the following we picked two examples to demonstrate how these algorithms are configured and executed on data. Other algorithms provide similar interfaces.

Some algorithms with few parameters provide a simple interface and can be directly called. For isotope pattern calculation see Code Example 2.

Code Example 2: Isotope pattern calculation

```
// excerpt: generate isotope pattern (max. 10 peaks or
// 99.9% of the isotopic probability) of Glucose

EmpiricalFormula molecule("C6H12O6");
IsotopeDistribution iso;
iso = molecule.getIsotopeDistribution(CoarseIsotopePatternGenerator(10));
iso = molecule.getIsotopeDistribution(FineIsotopePatternGenerator(1e-3));
```

Note that multiple algorithms may be available through the same interface, here an isotopic distribution can be calculated using coarse (unit mass) resolution or using fine (hyperfine isotopic) resolution. A single line of code will switch between the two algorithms, but the rest of the code will work without change (note that the two algorithms take different parameter sets, either number of peaks or total isotopic probability covered). This allows OpenMS to implement new algorithms that improve performance or accuracy “under the hood” while the interface stays the same for the user and algorithms can be switched to the new interface with ease. In the above example, this may become important with high resolution instrumentation that can differentiate between the hyperfine isotopic peaks in an isotopic envelope.

Other algorithms allow to fine-tune many parameters via a so-called Param object. The signal processing algorithm PeakPickerHiRes for centroiding of profile data is one of those (see Code Example 3). Internally PeakPickerHiRes uses a spline interpolation of the raw data to determine the m/z, FWHM, and intensity of peak centroids. Several aspects, e.g, the minimum signal-to-noise ratio to call a peak, can be configured via the Param object.

Code Example 3: Centroiding of profile spectra using the PeakPichkerHiRes class

```
// excerpt: centroiding profile spectra

// load profile spectra
```

```
MSExperiment profile_data, centroided_data;
MzMLFile().load("myData.mzML"), profile_data);

// get default parameters, change as desired and store changes
PeakPickerHiRes peak_picker;
Param param = peak_picker.getParam();
param.setValue("signal_to_noise", 1.0);
peak_picker.setParameters(param);

// run algorithm
peak_picker.pickExperiment(profile_data, centroided_data);
```

Available parameters are listed in the developer documentation of each algorithm and each parameter comes with information on allowed parameter ranges and a human-readable description of the parameter.

TOPP tools (developer perspective)

TOPP tools are command line applications developed within the OpenMS framework that utilize the OpenMS C++ library to implement a specific function. OpenMS follows the UNIX philosophy which is based on providing individual tools with defined functionality that can be chained together to powerful workflows, allowing greatest amount of flexibility to the user. Currently there are around 175 TOPP tools available which range from implementing a spectral noise filter to adaptors of thirdparty tools (e.g. the Comet search engine) and full implementations of quantitative analysis of SILAC, label-free or SWATH data. These tools are the preferred way to expose novel functionality in OpenMS and are available to user directly through the command line but are also automatically integrated into workflow engines such as KNIME.

OpenMS provides a base class and a template for the creation and integration of new tools. The base class handles logging, exception handling and command line argument parsing. The template has well-defined slots for adding tool parameters, input and output as well as algorithmic functionality which can be added in a straightforward manner. Further, before a tool can be merged into the OpenMS repository, it is required to provide its own regression tests, which allow for the automatic validation of its functionality in future releases of the OpenMS framework. Once added, the tool will become available as a command-line executable as well as a node in the KNIME workflow engine where users can integrate it into their workflows.

For further instruction on tool development in OpenMS please visit the tool section in the Developer C++ Quickstart Guide (see Table 1).

Visualization

MS data exhibits a large degree of variability and complex experimental setups may result in highly heterogeneous raw data. Visual inspection is regularly done in practice to assess data quality. TOPPView is a graphical application in OpenMS that allows users to inspect spectra, chromatograms, and identification results (Marc Sturm and Kohlbacher 2009). It uses the Qt 5 framework to implement advanced visualization which includes 1D (spectra, chromatogram), 2D and 3D (peak maps) plotting. Standard and re-useable Qt classes allow graphical plotting in multiple dimensions and OpenGL (Open Graphics Library) is used for high-performance interactive 3D plotting. Developers can build their own applications using the provided SpectrumWidget and SpectrumCanvas classes, which allows advanced plotting of mass spectrometric data without having to re-invent the wheel. Furthermore, TOPPView is tightly integrated with the available TOPP tools, which can be directly executed on the currently displayed data and the output of a transformation can then be loaded back into the same graphical view, providing direct feedback for a particular choice of tool and parameters. Fine-tuning these parameters sometimes may open the chance to analyze difficult data that fail to produce acceptable results using the default settings.

Code Quality and Community

OpenMS uses modern software engineering concepts more commonly found in industry settings than in academic environments. The project places great emphasis on modularity, reusability and extensive testing (using continuous integration), resulting in high code quality. The modular architecture of OpenMS tries to build upon existing standard libraries as much as possible, relying on them for sequence analysis, XML parsing, numerical computations and statistics. Modern object-oriented C++ is used exclusively throughout the code base, encapsulating raw data structures and discouraging manual heap-based memory management (when not provably crucial for efficiency), thus providing robust and error-tolerant code. Coding conventions are enforced and extensive English documentation is available for several thousand C++ functions part of the public API. All development is performed in the open, using the public source code repository and ticketing system GitHub. Stringent code reviews and continuous integration, running a multitude of functional, unit and black-box tests, ensure continued support, robustness and correctness of the code. Automated tests verify correctness of functionality of both individual functions and whole units on all three supported platforms while also analyzing the code quality using automated tools such as cppcheck and cpplint. Additionally, the code is reviewed by other developers using the four-eyes principles and changes can be requested depending on the performance, accuracy, style and code quality.

The OpenMS development team is integrated into an international multi-site effort supported by leading labs in experimental and computational mass spectrometry across Europe and North America. It is unique in the field by providing industrial-strength high-performance algorithmic implementations for a majority of common tasks in computational proteomics as open-source software. Frequent physical meetings and training sessions educate users and transmit knowledge of established workflows to practitioners in the field, providing also opportunities for users and developers to meet and exchange ideas. The project sees high contributor activity and several downstream tools such as MSstats, aLFQ and Skyline have started to integrate their tools with OpenMS (Choi et al. 2014; MacLean et al. 2010; Rosenberger et al. 2014)

Getting started with the OpenMS library for developers

For getting started to develop a new tool or use specific classes from the OpenMS library please follow the steps in Table 1.

Table 1: Developer instructions

Follow the steps to start programming on the library:

Working with your own fork:

- Fork the OpenMS repository (https://github.com/OpenMS/OpenMS)
- Clone the respective fork locally (git clone https://github.com/username/OpenMS.git)
- Compile OpenMS (build instructions below)
- Have fun working with and on the library

Working on OpenMS/OpenMS:

- Clone or download the source (https://github.com/OpenMS/OpenMS.git)
- Compile OpenMS (build instructions below)
- Have fun working with and on the library

Build instructions for Linux:

https://openms.de/documentation/install_linux.html

Build instructions for OS X:

https://openms.de/documentation/install_mac.html

Build instructions for Windows:

https://openms.de/documentation/install_win.html

For further instructions and information about coding conventions, please check out our WIKI:

https://github.com/OpenMS/OpenMS/wiki

OpenMS Developer C++ Guide:
https://openms.de/documentation/OpenMS_tutorial.html

OpenMS for users

TOPP tools (user perspective)

TOPP tools are individual tools, which usually perform one specific task. For example, the FeatureFinderCentroided can be used to detect two-dimensional features in centroided LC-MS data. A multitude of different tools exists ranging from simple format conversion, data filtering, to new data analysis, and data reduction algorithms. Additionally, wrapper exist and can be added upon request, which allow the usage of well-established third-party tools developed by non-

OpenMS developers, such as MS-GF+ (Kim and Pevzner 2014) or SIRIUS (Dührkop et al. 2019) within the OpenMS Framework. All OpenMS tools provide a detailed choice of parameters that go beyond what classical software in the field offers and allow to tailor its function to the specific needs of the user. The tools can be used individually or in an analysis pipeline either using the command line or a workflow engine. In the following, we would like to present a few examples of how to tackle common problem settings in mass spectrometry based multi-omics research with OpenMS tools and workflows.

Getting started with OpenMS for users

Interested in using OpenMS for your application and your research, have a look at the installation instructions and further tutorials on the usage of OpenMS (Table 2).

Table 2: OpenMS installation instruction

Follow the steps to start using OpenMS TOPP and Command Line Tools: Current release binary installer for MacOS, Windows and Linux (debian) can be obtained at https://www.openms.de/download/openms-binaries/ OpenMS Quickstart Guide: https://openms.de/documentation/Quickstart.html

Workflows in MS

A bioinformatics workflow defines a series of computational steps which can be applied to single or multiple data sets. Therefore, the concept of a workflow clearly separates the computation from the data on which it operates, describing a reproducible set of steps with a well-defined input and output. In theory, if the same software is run on the same data in the same order with the same parameters then the user should obtain the same output. In practice, it is often difficult to exactly reproduce these conditions on a different computer or at a later date, leading to challenges with reproducible data analysis. This is the problem which a computational workflow solves.
The workflow describes the software, the order in which the software operates on the data and the chosen parameters, which allows the replication of the work with the same data. This will lead to consistent results after reanalysis, since the workflow contains all information on how to process the data - information that is often lost if only an input file and an output file is provided. It is therefore crucial to store and submit workflow files alongside any data output files for other users to have a machine-readable and reproducible description of how the scientific result was computed. Working without workflows can lead to irreproducible computational results when

other researchers try to re-analyse data and potentially end up with different results because the computational methods were not well described.

OpenMS in KNIME

The workflow platform KNIME implements such workflows in a graphical user interface by connecting so called nodes (see Fig. 1). The node is the smallest entity in a workflow, which represents a single operation. Nodes can be connected in the workflow using input and output ports. In KNIME different port types exists depending on the action to be performed. It distinguishes between tables (table ports - black triangle) and whole files (file ports - blue box).

KNIME supports a plugin system which allows the usage of a multitude of analysis software in synergy with OpenMS. These cover a wide area of applications, such as machine learning, hypothesis testing, data visualization and chemoinformatics methods. Here, for example RDKit can be used for the visualization of small molecule structures encoded in SMILES. Additionally, KNIME supports scripting nodes for R, Python and other languages, which can be used to run custom scripts within the workflow. Finished workflows can be saved and shared with the community using the KNIME Community Workflow Hub (https://hub.knime.com/). Further a large collection of plugins is provided by the user community, which can be integrated into the workflow.

Getting started with OpenMS in KNIME

Interested in using OpenMS in combination with the workflow engine KNIME have a look at the installation instructions and further tutorials on the usage of OpenMS with KNIME (Table 3).

Table 3: KNIME with OpenMS plugin installation instructions and tutorials

Follow the steps to start using KNIME with the OpenMS plugin:

Installing KNIME
  Download KNIME (https://www.knime.com/downloads)
  Follow the installation instructions

Installing the OpenMS plugin in KNIME:
  Go to “Help” -> “Install New Software...”
  Select “KNIME Community Contributions (3.7) – http://update.knime.com/community-contributions/trusted/3.7” in “Work with”
  Open the “KNIME Community Contributions - Bioinformatics & NGS” field
  Select “OpenMS”
  Click “Next” and follow the instructions.

In general, KNIME will automatically detect missing plugins upon opening of a workflow and directs the user to the installation process.

OpenMS / KNIME Tutorial (with handout, example data and workflows): https;//www.openms.de/tutorials/

Other workflow systems with OpenMS integrations

In addition to KNIME, OpenMS can be used in various other workflow systems due to the common easily wrappable command line interface of its tools. Other popular workflow systems with graphical user interfaces (GUI) for which wrappers of OpenMS tools exist are the following:

- Galaxy: One of the most commonly used server-based workflow editors and managers in bioinformatics (Afgan et al. 2018).
- gUSE: A workflow system for high-performance computing clusters.

Another workflow language that also provides software for automatic execution of OpenMS tools on various hardware backends (local, remote, HPC clusters or clouds) is nextflow (Di Tommaso et al. 2017). Although nextflow does not provide a GUI yet, users can either script their own workflows or make use of community-made and well-maintained workflows available in its public workflow collection nf-core (Ewels et al. 2019). Lastly, OpenMS was successfully used in Pachyderm (Novella et al. 2019) and snakemake (Koster and Rahmann 2012) pipelines as well. Easy installation even in restricted HPC environments can be achieved through ready-made OpenMS containers (see the chapter about Containerization and Reproducibility) or through the Bioconda (Grüning et al. 2018) package manager.

Peptide identification and protein inference

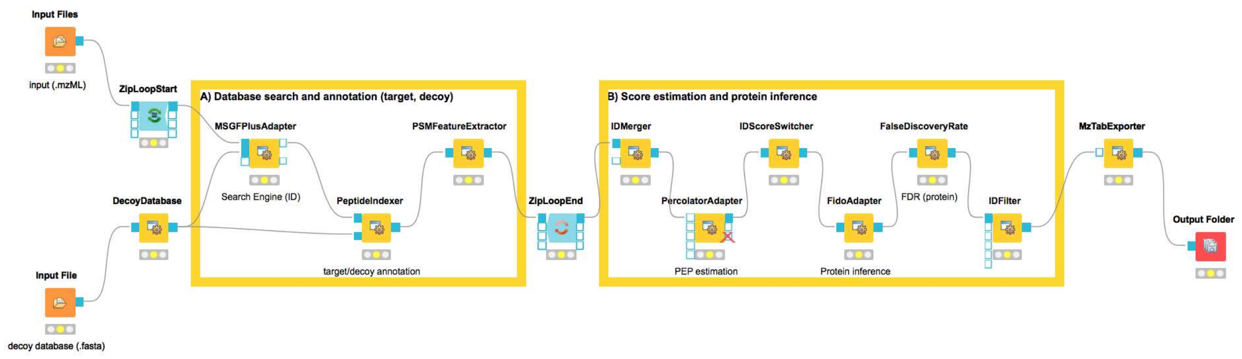

Figure 1: Identification workflow using OpenMS and KNIME can be applied for peptide identification and protein inference.

Search engine choice

One task that is commonly needed for the analysis of shotgun proteomics data is the identification of peptides and inference of their proteins of origin. Identification of proteomics data can be performed in a workflow as depicted in Figure 1. Here, mass spectrometry input files (.mzML) are loaded in the “Input Files” node. All files are processed iteratively by all tools between the ZipLoopStart and ZipLoopEnd nodes (see Figure 1A). Here, the search engine MS-GF+ (Kim and Pevzner 2014) is applied using the MSGFPlusAdapter to identify MS2 spectra. Search parameters, such as mass error, fragmentation method, possible fixed and variable modifications, as well as charge range and peptide length can be specified. As an alternative to the MSGFPlusAdapter, OpenMS provides a multitude of different wrappers for classic proteomic search engines, such as Comet (Jimmy K. Eng, Jahan, and Hoopmann 2013), Crux (Park et al. 2008), InsPecT (Tanner et al. 2005), Mascot (Perkins et al. 1999), MSFragger (Kong et al. 2017), MyriMatch (Tabb, Fernando, and Chambers 2007), OMSSA (Geer et al. 2004), Sequest (J K Eng, McCormack, and Yates 1994), PepNovo (Frank and Pevzner 2005), and XTandem (Craig and Beavis 2004). Hence, the search engine node within the workflow can be conveniently exchanged for other tools and thus one could test and find out the best performing method in a concise benchmark.

Sequence database

Database search engines make use of a provided sequence database fasta file (“Input File” node), which should contain all target proteins of the organism of choice and possible contaminants. The database and its size highly depend on the research question and the experimental design. In order to assess a false discovery rate in a later step, the database should additionally contain decoy proteins - shuffled or reversed sequences - of all provided targets. The OpenMS tool DecoyDatabase provides an option to concatenate multiple databases (e.g. Swiss-Prot human, cRAP) and generate decoys using different methods (e.g. protein- or peptide-based shuffling or pseudo-reversing) that will be appended to the provided database and tagged with a specified name prefix or suffix (e.g. “DECOY_”).

Identification post-processing

After the database search (see Figure 1A), identified PSMs (peptide spectrum matches) undergo a series of consecutive post-processing steps to yield the FDR annotated lists of peptide and protein identifications. First target and decoy annotations are assigned to the respective PSM based on the fasta file decoy name tag entries (PeptideIndexer). Next, a number of descriptive features are computed and annotated in a standardized manner to each PSM in order to run a multivariate discrimination of target and decoy space at a later stage (PSMFeatureExtractor). Advanced users can also add specific or customized PSM features in this step. While each MS run is processed separately, it is commonly recommended to co-process the runs and compute a false discovery rate globally over the merged set of all identifications (Serang and Käll 2015). Hence, the ZipLoop ends here and IDMerger performs a merging of all identifications. Subsequently, the tool Percolator (The et al. 2016) is employed and computes a global FDR based on target and decoy PSM feature scores annotated previously. The Percolator version in OpenMS additionally supports basic protein inference capabilities which we will skip in favor of more advanced methods using ambiguous peptides.

Protein Inference

Carrying out a robust protein inference and probabilistically distributing evidence of shared peptides in this workflow (see Figure 1B), requires the application of the FidoAdapter tool. However, as FidoAdapter was designed to work with OpenMS' own estimation tool for PEPs (IDPosteriorErrorProbability), it is necessary to pick and rename the right score from the PercolatorAdapter. Hence, IDScoreSwitcher, FidoAdapter and subsequently the FalseDiscoveryRate node is applied on protein-level to calculate a protein inference-based target-decoy FDR. As an alternative to Fido (Serang, MacCoss, and Noble 2010) one could instead employ the tool EPIFANY, which has recently been added to the OpenMS toolbox or PIA which is provided in a separate KNIME plugin (Uszkoreit et al. 2015). Ultimately, the resulting peptide and protein identifications can be filtered by various criteria on both levels, including q-value, other metavalues or blacklists from fasta/text files (IDFilter) and exported in the community standard format (mzTab).

Further peptide identification methods

De novo peptide search

Apart from database search, OpenMS also provides tools for *de novo* peptide identification for example the CompNovo or CompNovoCID (Bertsch et al. 2009) tool, as well as the NovorAdapter tool which supports *de novo* peptide search using Rapid Novor (Ma 2015).

Spectral library search

Additionally, peptides can be identified via a spectral library search. Here, we provide a tool called SpecLibSearcher, which is able to identify MS2 spectra (.mzML) based on an input spectral library (.msp). Alternatively, we also support the SpectraST tool from the TPP through the SpectraSTSearchAdapter, which can be used for spectral library search (Lam et al. 2007).

Additional supported methods

Additional tools are available which are able to use the identification data, for example for phosphosite localization and spectral clustering.

Phosphosite localization

The LuciphorAdapter uses the thirdparty tool LuciPHOr2 (Fermin et al. 2015) for the assessment of the phosphosite localization on a phosphopeptide with multiple possible sites, which can be critical in phosphorylation studies. It estimates a false localization rate based on a target decoy approach, which can be used for filtering later on.

Spectral clustering

OpenMS also provides an adapter to MaRaCluster (The and Käll 2016) a thirdparty tool to cluster spectra into groups of similar spectra or to create consensus spectra. Applications range from yielding better and faster identification rates for database search to unsupervised clustering to identify spectra.

Peptide and Protein quantification

The identification workflow above can be extended to perform identification and quantification. Here, depending on the experimental method (e.g. label free, SILAC, TMT) the respective nodes can be plugged into the workflow.

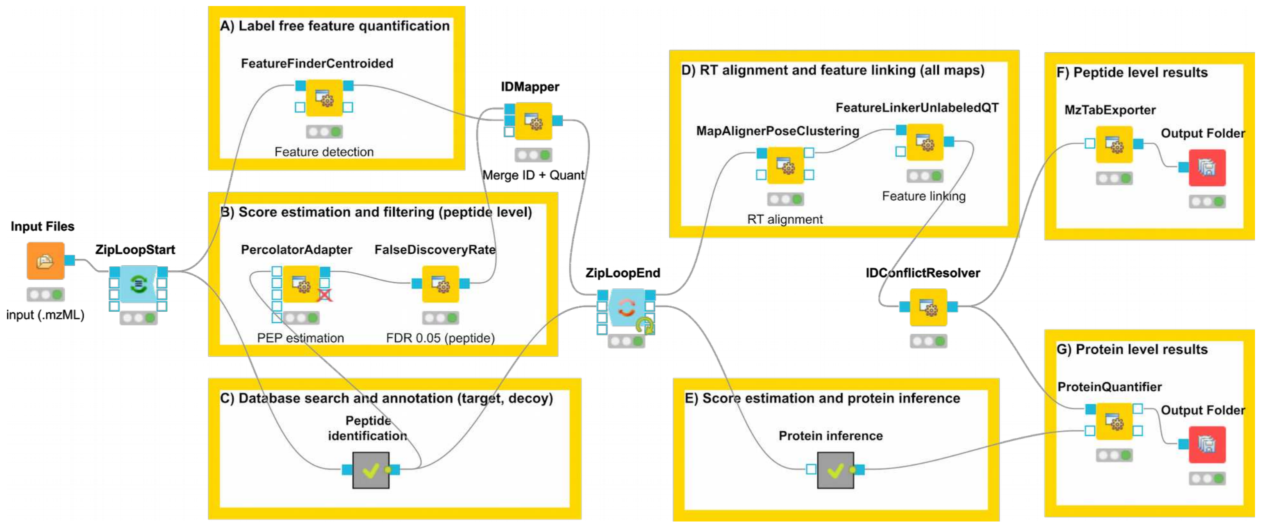

Figure 2: Workflow using OpenMS and KNIME which can be applied for peptide, protein identification and label free quantification. The peptide identification (C) and protein inference (E) are described in the workflow in Figure 1.

The workflow in see Figure 2 depicts how identification and label free quantification can be performed with OpenMS in KNIME.

Feature finding

The tools, which are able to perform label free quantification are named FeatureFinder in OpenMS. There are several different implementations available such as FeatureFinderCentroided (Weisser et al. 2013) (see Figure 2A) or the FeatureFinderMultiplex. A FeatureFinder recognizes features in LC-MS maps, corresponding to peptides, by their characteristic isotope pattern. Quantification is carried out based on the sum of intensities within the feature region. Here, the position in m/z and rt as well as the charge of the analyte is computed. Resulting features are then represented with scores based on an isotope profile and retention time model. Data points which do not fit the model are removed from the feature region.

Combining post-processed identification with quantification data (peptide level)

Afterwards, a score estimation and filtering of peptide identifications found in an individual file is performed (see Figure 2). For peptide level results, posterior error probability estimation is performed using Percolator (The et al. 2016) samplewise and a user-defined FDR filter is applied (e.g. 5% FDR). In order to combine quantitative and identification information per sample, peptide identifications are generated samplewise and mapped to their respective feature by the IDMapper.

Retention time alignment, feature linking and generation of peptide level results

The mapped information is further processed by the MapAlignerPoseClustering, which performs a linear retention time alignment of input maps to correct retention time shifts and distortions. This is based on a pose clustering algorithm, which uses affine transformation and later refinement based on feature grouping. The FeatureLinkerUnlabeledQT (Weisser et al. 2013) uses QT-based clustering and linking to group corresponding features from multiple maps for label free data (see Figure 2). Additionally, OpenMS provides the FeatureLinkerUnlabeledKD, which uses a faster KD-tree based approach for linking. The KD-tree based algorithm has a speed advantage, which becomes apparent for larger datasets (hundreds of samples upwards). The IDConflictResolver is used to ensure that every feature is associated with one single identification based on its score. The peptide level results are then exported as MzTab (see Figure 2F).

Generate protein level results

The peptide level information from the IDConflictResolver can also be used in conjunction with the protein inference information (see Figure 2C and Figure 2E) to quantify on protein level (see Figure 1G). In this workflow the ProteinQuantifier (Weisser et al. 2013) accumulates feature intensities to peptide abundances based on the identification. Afterwards it uses the protein inference information to average over the abundances the peptides referring to a protein.

Additional supported quantification methods

Label free quantification based on identification data

The quantification and retention time alignment can also be performed based on previous identifications. The FeatureFinderIdentification (Weisser and Choudhary 2017) can be used for label free quantification of the MS1 features, based on prior peptide identification. This is based on the intuition that a high confidence identification on MS2 level from a specific precursor will produce a corresponding feature at that position in all LC-MS maps of an experiment. Similar, the MapAlignerIdentification performs retention time alignment based on previous peptide identification.

Quantification using chemical or isotopic labeling

Further, quantification can be performed for experiments using isotopically or chemically labelled peptides, such as dimethyl labeling or SILAC, by applying the FeatureFinderMultiplex. First, the algorithm finds pairs in a MS1 scan by using the mass difference based on the labeling, the charge and the intensity profile, its correlation and the averagine model. The assessed features are then filtered, and clusters are formed in the rt and m/z range. These clusters correspond to the monoisotopic trace referring to the lightest peptide of a pair (for example SILAC). Afterwards hierarchical clustering is used to assign the peaks to a specific cluster. Then linear regression is used to determine the relative amount of the peptide based on their labels.

Quantification using isobaric labeling

In addition, quantification for isobaric labeling can be performed with the IsobaricAnalyzer, which is able to extract and normalize TMT and iTRAQ quantitative information. It is able to extract the data from centroided MS2 and MS3 spectra and performs an isotope correction based on the specified correction matrix (as provided by the manufacturer).

Targeted Analysis

Targeted proteomics is a field of proteomic analysis where accurate and reproducible quantification is required and is often used in clinical settings or laboratory experiments where quantification is of utmost importance. Multiple types of targeted proteomics workflows exist, traditional workflows on a triple quadrupole instrument (SRM or MRM) use a list of Q1 (precursor m/z) and Q3 (fragment m/z) for a set of peptides, through which the instrument will then cycle deterministically. These Q1-Q3 m/z pairs are called transitions and the instrument will typically measure 3-6 transitions per peptide over the course of an experiment, producing a chromatographic measurement for each transition. In more advanced setup, a set of transitions will not be measured during the whole LC-MS/MS experiment but only during a fixed amount of time centered around the putative elution of the target peptide. Similarly, PRM measurements use a list of target peptides, but acquire a high resolution full MS/MS scan for each target peptide (independent of whether a precursor signal was detected or not), resulting in a set of full MS/MS scans for a given precursor acquired deterministically over time. Software is then used to extract a set of N (typically 3-6) transitions from these scans to determine the elution time-point of a peptide. One of the drawback of both SRM and PRM is that only a limited set of peptides can be targeted and no quantitative data is acquired for peptides that are not on the target list.
This limitation is addressed by DIA (or SWATH-MS) approaches, which partition the precursor m/z space into small windows of ca. 10 to 25 m/z and deterministically fragment all precursors in that window and record a full high resolution MS/MS scan of the resulting fragment ions. Similar to PRM, software is used to extract fragment ion traces from these data, but unlike PRM, no target list of peptides is required since the whole mass range is targeted. In the original SWATH-MS implementation, 32 windows of 25 m/z width were used to target the mass range of 400 - 1200 Da, which covers most of the human tryptic peptides.

OpenMS supports targeted analysis through the OpenSWATH (Röst, Rosenberger, et al. 2014) module which allows automated analysis of targeted proteomics (SRM and PRM) as well as DIA / SWATH-MS data. For all targeted proteomics experiments, the analysis requires the raw MS data as well as an assay library that contains information (precursor m/z, fragment ion m/z, retention time, fragment intensity) about the target peptides. Like the rest of OpenMS, OpenSWATH works with standard file formats (mzML for raw MS data, TraML for the assay library) in order to provide interoperability with other software and standard compliance. The OpenSWATH module performs automatic retention time and m/z calibration using a set of anchor peptides (e.g. spiked in standards such as iRT peptides (Escher et al. 2012) or endogenous peptides (Parker et al. 2015) using either a linear or non-linear function.
Next, OpenSWATH will perform chromatographic extraction, where extracted ion chromatograms are constructed for all peptides in the assay library and then performs chromatographic peak picking and scoring. For DIA data, it will perform chromatographic analysis of the data, but it will also consult the full scan data to obtain additional information such as mass accuracy and isotopic envelope information. Statistical analysis of the data using the target-decoy approach can then either be performed using Percolator (The et al. 2016) or the specifically designed pyProphet software (Teleman et al. 2015), which will compute false discovery rate (FDR) estimates. Each task described above can be performed by individual TOPP tools, but for convenience we offer an integrated tool OpenSwathWorkflow that performs all steps at once which speeds up execution.
As mentioned above, each targeted analysis requires an assay library. OpenMS provides multiple tools to generate such assays including OpenSwathAssayGenerator which can take spectral library input (e.g. in SpectraST format) and generate an assay library output using specific transition-level criteria. Similarly, OpenSwathDecoyGenerator offers several methods ('shuffle', 'pseudo-reverse', 'reverse', 'shift') for generating spectral decoys that are required for target-decoy approaches to FDR estimation.

Extensive documentation is provided at http://www.openswath.org/en/latest/ with detailed information on parameters, example data and extended tutorials on how to run OpenSWATH.

Metabolomics

OpenMS can be used for label-free LC-MS metabolomics data analysis. It supports quantification and different identification techniques, such as accurate mass search, *de novo* identification and spectral library search. Similar to label-free proteomics, feature detection and adduct grouping are first performed for each LC-MS map. Retention time alignment and grouping of features across multiple maps are then performed using MapAlignerPoseClustering and FeatureLinkerUnlabeledQT (see “Peptide and Protein quantification”).

Metabolite quantification

OpenMS provides several tools with the prefix “FeatureFinder” that are used to perform quantification on MS1-level. FeatureFinderMetabo (Kenar et al. 2014) was specifically developed to detect small molecules in LC-MS samples with high sensitivity and specificity. As before, a feature contains all signals that are caused by the same metabolite in a certain charge state.

The algorithm first detects continuous mass traces by starting a new trace at the most intense peak and extending it in both retention time directions. Since low-intensity peaks are often less accurate, a heteroscedastic noise model is used to decide if an additional peak is added. Often, small molecules with the same mass elute at similar retention times and appear on a single continuous mass trace. FeatureFinderMetabo can detect these cases based on the elution profile and splits the mass trace at the local minimum of the elution profile. Finally, all mass traces that are caused by the same metabolite are assembled into a single feature, i.e. the monoisotopic trace and additional isotopic traces. To decide which mass traces are assembled, they need to co-elute at the correct m/z distance and exhibit the expected isotope abundance ratios. Due to the vast diversity of metabolites, it is not possible to use the averagine model. Instead, a support vector machine was trained to detect isotope abundance ratios that are likely to be observed for metabolites.

Each feature corresponds to a metabolite with a certain adduct. The same compound can be observed multiple times at a similar retention time with different adducts or neutral losses (e.g. sodium adduct or water loss). The OpenMS tool MetaboliteAdductDecharger can be used to group these features and annotate them with their adduct and charge state, which can be useful for subsequent analysis steps. For this, a list of potential adducts and their probabilities has to be provided. The MetaboliteAdductDecharger then builds a connected graph of co-eluting features, which is resolved using a corresponding Integer Linear Programming approach (ILP). With this, the solution that maximizes the overall probabilities is chosen in the end.

Metabolite identification

Compound identification remains one of the major challenges in metabolomics. OpenMS supports commonly used approaches based on compound databases and spectral libraries. In addition, it integrates SIRIUS (Dührkop et al. 2019) for the *de novo* identification of metabolites.

Compound databases

The tool AccurateMassSearch is the first step towards compound identification and can be used to annotate detected features with putative compound identifications using only their accurate mass. The tool considers arbitrary adducts for positive and negative polarity and a compound database providing access to the Human Metabolome Database (HMDB) by default (Wishart et al. 2018).

Spectral library search

Searching the accurate mass of unidentified metabolites against a compound database will provide insight into which compounds could be present in the sample, but the results are often ambiguous. To arrive at more confident identifications, MetaboliteSpectralMatcher can be used to search MS/MS spectra against spectral libraries containing experimentally acquired reference spectra. Spectrum matches are scored using a modified version of the hyperscore introduced by Fenyö and Beavis (Fenyö and Beavis 2003). Any spectral library in the mzML file format can be used, by default MassBank provides access MassBank (Horai et al. 2010).

De novo

De novo identification has the advantage that it does not rely on a spectral library of previously measured compounds. SIRIUS reported identification rates of more than 70% (Dührkop et al. 2019). *De novo* approaches are computationally expensive making their application in large high-throughput studies cumbersome. To this end SIRIUS has been integrated into OpenMS (SiriusAdapter), which allows the pre-processing and complexity reduction of mass spectrometry data, by providing feature, charge and adduct information.

Metaproteomics

The field of Metaproteomics studies communities of (micro-) organisms like the gut microbiome at the proteome level. Besides host-pathogen interaction, central topics are the degradation of substrates and nutrients - including feeding on other organisms. The characterization of organisms helps to understand clinical relevant processes in microbiomes and potentially associated diseases. In OpenMS, we provide the tool MetaProSIP to perform stable isotope probing of metaproteomic communities. It determines to what extent isotopes from the labeled substrate were incorporated into newly synthesized proteins and the labeling ratio to characterize the speed of protein biosynthesis (protein turnover). Carefully designed experiments and time-series analysis of MetaProSIP results allows reconstructing the elemental flow between functional groups of organisms in a complex community (Sachsenberg et al. 2015).

Cross-linking MS

Structural proteomics is an emerging field combining different experimental methodologies with mass spectrometry analysis to gain insights into the structures of biomolecular complexes. Cross-linking is one of these methods and involves inducing non-native covalent bonds between different molecules or different moieties within the same molecule using either chemical reagents, UV light or both. In protein-protein cross-linking usually side chains of protein residues are bound using a chemical cross-linker. The linker has a specific length that it can span, so the identification of the two linked residues gives us an upper bound for the distance between these residues. They can be part of the same protein or two separate proteins interacting with each other, therefore cross-linking MS yields information about the structures of single proteins as well as protein complexes of any size (Leitner et al. 2016). In nucleic acid to protein cross-linking experiments, covalent bonds are induced between proteins and RNA or DNA strands in close proximity. These bonds are induced while the molecules are as close as possible to their native state or a specific state of interest. Afterwards the proteins are digested with enzymes and the linked peptide pairs or peptides linked with nucleotide oligos are analyzed by mass spectrometry. Because of the increased search space and more complex MS2 fragmentation patterns, linear peptide search algorithms can not be effectively used to analyze these types of data.
OpenMS enables the analysis of both of these types of data through the dedicated search algorithms OpenPepXL for protein-protein cross-linking and RNP^{XL} for protein-RNA cross-linking.

OpenPepXL requires the centroided MS data from a protein-protein cross-linking experiment and a fasta database of the targeted proteins and decoys. If an isotopically labeled cross-linking reagent was used, it is possible to combine the information from two MS2 spectra (the same peptide pair linked by the light and the heavy linker) to reduce search time and increase the specificity of the search. In this case an additional consensusXML file produced by the tool FeatureFinderMultiplex is necessary to link together MS1 features from the light and heavy cross-linkers. OpenPepXL will then digest the fasta database and preprocess the spectra or spectra pairs by deisotoping and filtering. For each spectrum a list of candidate peptide pairs is generated according to its precursor mass and the given precursor tolerance. Theoretical spectra are generated by considering both peptides and the cross-linker as a single molecule to accurately model the fragment masses of cross-linked peptides. The experimental spectrum and the theoretical spectrum are matched and scored against each other using the OpenPepXL scoring function. The hits for all MS2 spectra in the input mzML file are then written out in idXML or mzIdentML. These can be further processed by XFDR, the dedicated false discovery rate estimation tool for protein-protein cross-linking based on xProphet (Walzthoeni et al. 2012).
RNP^{XL} identifies protein-RNA cross-links in UV-induced cross-linking experiments. Input files are centroided spectra and a fasta database of the targeted proteins and decoys. A detailed description on data analysis is given in the Supplementary material of Kramer et al. (Kramer et al. 2014).

RNA (modification) analysis

Besides proteomics and metabolomics, another important application of biological mass spectrometry is the sequence analysis of nucleic acids. Before the advent of "next generation" sequencing approaches, mass spectrometry was being pursued as a potential tool for high-throughput DNA sequencing (Apffel et al. 1997). Recently, the nascent field of epitranscriptomics (RNA epigenetics) has spurred a growing interest in chemical modifications on RNA, and an appreciation of their varied biological roles. Mass spectrometric analysis of intact RNA oligonucleotides has become an important method in this area; it has the unique advantages of allowing the detection and localization of multiple different modifications at the same time, with single-nucleotide resolution. The experimental and computational analysis workflows are largely analogous to shotgun proteomics: Purified RNA samples are enzymatically digested (typically with RNase T1, which cuts after guanosines), the oligonucleotides separated by liquid chromatography (typically ion-pair HPLC) and characterized by tandem mass spectrometry in negative ion mode.
OpenMS now includes tools to analyze data from such experiments. The NucleicAcidSearchEngine (NASE) provides functionality for the identification of RNA oligonucleotides based on tandem mass spectra (mzML) and a sequence database (fasta) (Wein et al. 2018). Analogously to database search engines for shotgun proteomics, NASE has a variety of options that can be adjusted by users, including support for a plethora of ribonucleotide modifications and different digestion enzymes. During development NASE has been tested on tRNA, rRNA, and miRNA samples.
Beyond RNA identification, OpenMS offers basic capabilities for label-free quantification of RNA analytes. To this end, NASE can produce a file with "target coordinates" for its search results,

which the FeatureFinderMetaboIdent tool can use to perform targeted quantification of the identified oligonucleotides.

Visualization capabilities (user perspective)

The graphical application TOPPView provides advanced visualization of mass spectrometric data. It allows examining raw spectra and chromatograms, the effects of data processing steps as well as identification and quantification results in a graphical user interface (GUI). TOPPView offers one dimensional visualization of spectra (m/z vs intensities) and chromatograms (RT vs intensities). Additionally, whole experimental maps can be visualized in 2D (RT vs m/z) and 3D (RT vs m/z vs intensity) allowing visual quality inspection of the current experiment. From the 2D view, projections on either the RT axis (extracted ion chromatograms, XIC) or projections on the m/z axis (integrated spectra) can be computed (see Figure 3).
Furthermore, chromatographic data as acquired in SRM or extracted ion chromatograms (from PRM / DIA or SWATH-MS data) can be visualized using TOPPView's chromatographic visualization module (RT vs intensities). DIA or SWATH-MS data can be visualized using full high-resolution MS/MS spectra by displaying SWATH maps individually in 2D or 3D (fragment ion m/z vs RT). TOPPView is also capable of displaying ion mobility (IM) data is individual spectra ("frames") contain additional ion mobility data, either annotated as meta-data or in 2D or 3D (m/z vs IM) by right-clicking on a spectrum and selecting "Switch to ion mobility view".

Identification data from search engines can be superimposed on individual spectra to annotate fragment ion peaks and display the highest-scoring peptide identification in the same graphical frameworks. These peptide-spectrum matches (PSMs) can be visualized and manually curated.

The TOPPView application is tightly integrated with the TOPP tools provided by OpenMS, offering graphical dialogues to conveniently configure and run TOPP tools without resorting to executing the tool on the command line. Possible applications are optimizing tool configurations to find the best parameters for a particular type of instrument or data.

TOPPView is highly configurable, where the user can select the colors of the display, the position of the axes and the scaling of the data (log, relative, absolute). Furthermore, the user can choose to not load all data into memory at once, which can be suitable for memory-constrained situations or in the case of very large files. In this case, TOPPView is capable of only loading the requested spectra into memory using the indexedmzML data standard that allows random-access to individual spectra even in large XML files (Röst et al. 2015).

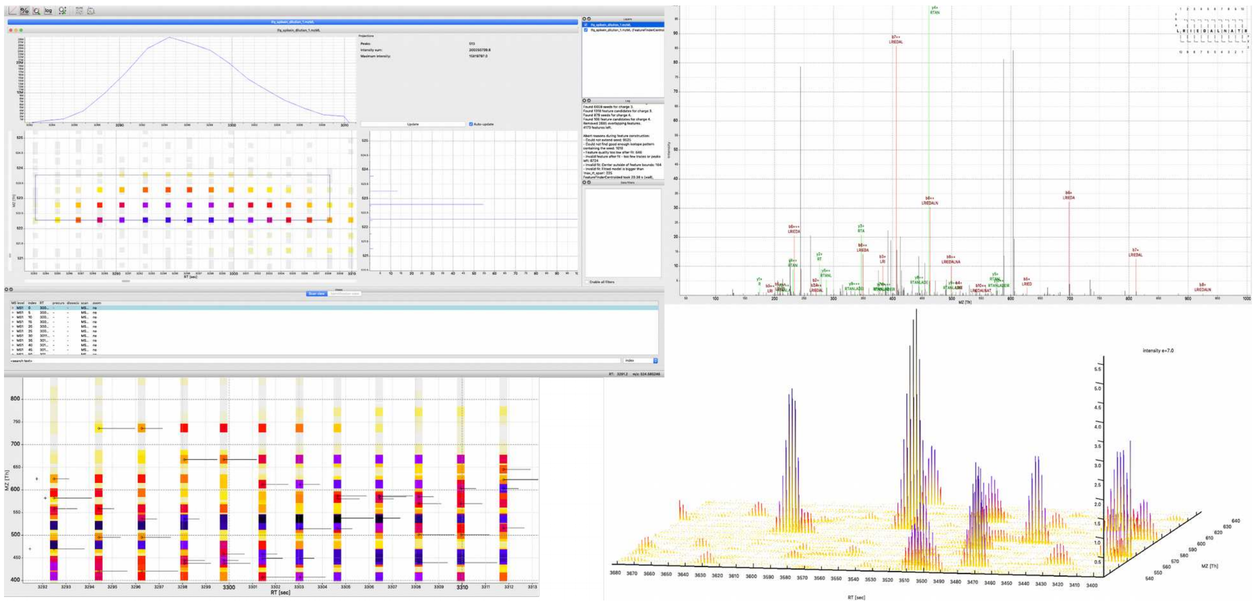
Figure 3: TOPPView - Tool for data visualization. (Top-left) Extracted ion chromatogram view of a specific feature in a segment of the current 2D map. (Bottom-left) Time points of a fragmentation events in the 2D view. (Top-right) MS2 spectrum in a 1D view overlaid with the identification based on database search (Identification View). (Bottom-right) 3D representation of a segment from a mass spectrometry map.

Containerization and reproducibility

Containerization of software is usually defined as a method to bundle code, configurations and dependencies into one object that is quickly and reliably deployable on different computing environments. Leading providers of software that is able to create and run such containers include Docker and Singularity. OpenMS regularly provides updated containers and recipes (see Table 4) to create such containers for different configurations and scenarios. Those scenarios include but are not limited to development, usage in container-enabled workflow-systems (e.g. nextflow) or spawning workers in cloud or HPC environments to scale up analyses. Even full workflows can be containerized (e.g. with the deNBI-CIBI plugin from KNIME) to create a snapshot of the environment with which a certain analysis was performed. This is useful e.g. for generating referenceable identifiers in scientific journals and enabling reproducible research.

Table 4: Information about containerization

Further information about OpenMS containers can be found here: OpenMS DockerHub pages: https://hub.docker.com/u/openms and https://hub.docker.com/u/hroest Biocontainers collection: https://biocontainers.pro Containerization of KNIME workflows: https://www.knime.com/denbicibi-contributions https://github.com/OpenMS/OpenMS/wiki/Exporting-a-KNIME-workflow-as-a-Docker-container

Acknowledgements

O.K., L.B., J.P. and T.S. were supported by a grant from the German Federal Ministry of Education and Research (BMBF) under grant no. 031A535A (German Network for Bioinformatics, de.NBI/CIBI).

This Project was funded by the Government of Canada through Genome Canada and the Ontario Genomics Institute (OGI-164). This work was supported by the Canadian Institutes for Health Research, the Natural Sciences and Engineering Research Council of Canada, and the Canada Research Coordinating Committee. H.L.R. is supported by the Canadian Foundation for Innovation and the John R. Evans Leaders Fund and is the Canada Research Chair in Mass Spectrometry-based Personalized Medicine.

References

Afgan, Enis et al. 2018. “The Galaxy Platform for Accessible, Reproducible and Collaborative Biomedical Analyses: 2018 Update.” *Nucleic Acids Research* 46(W1): W537–44. https://academic.oup.com/nar/article/46/W1/W537/5001157.

Apffel, Alex et al. 1997. “Analysis of Oligonucleotides by HPLC−Electrospray Ionization Mass Spectrometry.” *Analytical Chemistry* 69(7): 1320–25. https://pubs.acs.org/doi/10.1021/ac960916h.

Berthold, Michael R et al. 2007. Studies in Classification, Data Analysis, and Knowledge Organization (GfKL 2007) *KNIME: The Konstanz Information Miner*. Springer.

Bertsch, Andreas et al. 2009. “De Novo Peptide Sequencing by Tandem MS Using Complementary CID and Electron Transfer Dissociation.” *Electrophoresis* 30(21): 3736–47. http://www.ncbi.nlm.nih.gov/pubmed/19862751.

Choi, Meena et al. 2014. “MSstats: An R Package for Statistical Analysis of Quantitative Mass Spectrometry-Based Proteomic Experiments.” *Bioinformatics (Oxford, England)* 30(17): 2524–26. http://www.ncbi.nlm.nih.gov/pubmed/24794931.

Craig, Robertson, and Ronald C Beavis. 2004. “TANDEM: Matching Proteins with Tandem Mass

Spectra." *Bioinformatics (Oxford, England)* 20(9): 1466–67. http://www.ncbi.nlm.nih.gov/pubmed/14976030.

Dührkop, Kai et al. 2019. "SIRIUS 4: A Rapid Tool for Turning Tandem Mass Spectra into Metabolite Structure Information." *Nature Methods* 16(4): 299–302. http://www.nature.com/articles/s41592-019-0344-8.

Eng, J K, A L McCormack, and J R Yates. 1994. "An Approach to Correlate Tandem Mass Spectral Data of Peptides with Amino Acid Sequences in a Protein Database." *Journal of the American Society for Mass Spectrometry* 5(11): 976–89. http://www.ncbi.nlm.nih.gov/pubmed/24226387.

Eng, Jimmy K., Tahmina A. Jahan, and Michael R. Hoopmann. 2013. "Comet: An Open-Source MS/MS Sequence Database Search Tool." *PROTEOMICS* 13(1): 22–24. http://doi.wiley.com/10.1002/pmic.201200439.

Escher, Claudia et al. 2012. "Using IRT, a Normalized Retention Time for More Targeted Measurement of Peptides." *Proteomics* 12(8): 1111–21.

Ewels, Philip A et al. 2019. "Nf-Core: Community Curated Bioinformatics Pipelines." *bioRxiv*: 610741. http://biorxiv.org/content/early/2019/04/16/610741.abstract.

Fenyö, David, and Ronald C. Beavis. 2003. "A Method for Assessing the Statistical Significance of Mass Spectrometry-Based Protein Identifications Using General Scoring Schemes." *Analytical Chemistry* 75(4): 768–74. https://pubs.acs.org/doi/10.1021/ac0258709.

Fermin, Damian, Dmitry Avtonomov, Hyungwon Choi, and Alexey I. Nesvizhskii. 2015. "LuciPHOr2: Site Localization of Generic Post-Translational Modifications from Tandem Mass Spectrometry Data." *Bioinformatics* 31(7): 1141–43.

Fillbrunn, Alexander et al. 2017. "KNIME for Reproducible Cross-Domain Analysis of Life Science Data." *Journal of Biotechnology* 261(February): 149–56.

Frank, Ari, and Pavel Pevzner. 2005. "PepNovo: De Novo Peptide Sequencing via Probabilistic Network Modeling." *Analytical Chemistry* 77(4): 964–73. https://pubs.acs.org/doi/10.1021/ac048788h.

Geer, Lewis Y et al. 2004. "Open Mass Spectrometry Search Algorithm." : 958–64.

Grüning, Björn et al. 2018. "Bioconda: Sustainable and Comprehensive Software Distribution for the Life Sciences." *Nature Methods* 15(7): 475–76. http://www.nature.com/articles/s41592-018-0046-7.

Horai, Hisayuki et al. 2010. "MassBank: A Public Repository for Sharing Mass Spectral Data for Life Sciences." *Journal of Mass Spectrometry* 45(7): 703–14.

Kenar, Erhan et al. 2014. "Automated Label-Free Quantification of Metabolites from Liquid Chromatography-Mass Spectrometry Data." *Molecular &amp; Cellular Proteomics* 13(1): 348–59. http://www.mcponline.org/content/13/1/348.full %5Cnpapers3://publication/doi/10.1074/mcp.M113.031278.

Kim, Sangtae, and Pavel A Pevzner. 2014. "Database Search Tool for Proteomics." *Nature Communications* 5: 1–10. http://dx.doi.org/10.1038/ncomms6277.

Kong, Andy T et al. 2017. "MSFragger: Ultrafast and Comprehensive Peptide Identification in Mass Spectrometry–Based Proteomics." *Nature Methods* 14(5): 513–20. http://www.nature.com/articles/nmeth.4256.

Koster, J., and S. Rahmann. 2012. "Snakemake--a Scalable Bioinformatics Workflow Engine." *Bioinformatics* 28(19): 2520–22. https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/bts480.

Kramer, Katharina et al. 2014. "Photo-Cross-Linking and High-Resolution Mass Spectrometry for Assignment of RNA-Binding Sites in RNA-Binding Proteins." *Nature Methods* 11(10): 1064–70. http://www.nature.com/articles/nmeth.3092.

Lam, Henry et al. 2007. "Development and Validation of a Spectral Library Searching Method for Peptide Identification from MS/MS." *Proteomics* 7(5): 655–67.

Leitner, Alexander, Marco Faini, Florian Stengel, and Ruedi Aebersold. 2016. "Crosslinking and

Mass Spectrometry: An Integrated Technology to Understand the Structure and Function of Molecular Machines." *Trends in Biochemical Sciences* 41(1): 20–32. https://linkinghub.elsevier.com/retrieve/pii/S0968000415002078.

Ma, Bin. 2015. "Novor: Real-Time Peptide de Novo Sequencing Software." *Journal of The American Society for Mass Spectrometry* 26(11): 1885–94. http://link.springer.com/10.1007/s13361-015-1204-0.

MacLean, Brendan et al. 2010. "Skyline: An Open Source Document Editor for Creating and Analyzing Targeted Proteomics Experiments." *Bioinformatics (Oxford, England)* 26(7): 966–68. http://www.ncbi.nlm.nih.gov/pubmed/20147306.

Novella, Jon Ander et al. 2019. "Container-Based Bioinformatics with Pachyderm" ed. Jonathan Wren. *Bioinformatics* 35(5): 839–46. https://academic.oup.com/bioinformatics/article/35/5/839/5068160.

Park, Christopher Y. et al. 2008. "Rapid and Accurate Peptide Identification from Tandem Mass Spectra." *Journal of Proteome Research* 7(7): 3022–27. http://pubs.acs.org/doi/abs/10.1021/pr800127y.

Parker, Sarah J et al. 2015. "Identification of a Set of Conserved Eukaryotic Internal Retention Time Standards for Data-Independent Acquisition Mass Spectrometry." *Molecular & cellular proteomics : MCP* 14(10): 2800–2813. http://www.ncbi.nlm.nih.gov/pubmed/26199342.

Perkins, David N., Darryl J. C. Pappin, David M. Creasy, and John S. Cottrell. 1999. "Probability-Based Protein Identification by Searching Sequence Databases Using Mass Spectrometry Data." *Electrophoresis* 20(18): 3551–67. http://doi.wiley.com/10.1002/%28SICI%291522-2683%2819991201%2920%3A18%3C3551%3A%3AAID-ELPS3551%3E3.0.CO%3B2-2.

Pfeuffer, Julianus et al. 2017. "OpenMS - A Platform for Reproducible Analysis of Mass Spectrometry Data." *Journal of Biotechnology* 261(February): 142–48.

Rosenberger, George et al. 2014. "ALFQ: An R-Package for Estimating Absolute Protein Quantities from Label-Free LC-MS/MS Proteomics Data." *Bioinformatics (Oxford, England)* 30(17): 2511–13. http://www.ncbi.nlm.nih.gov/pubmed/24753486.

Röst, Hannes L., Uwe Schmitt, Ruedi Aebersold, and Lars Malmström. 2014. "PyOpenMS: A Python-Based Interface to the OpenMS Mass-Spectrometry Algorithm Library." *Proteomics* 14(1): 74–77.

Röst, Hannes L, George Rosenberger, et al. 2014. "OpenSWATH Enables Automated, Targeted Analysis of Data-Independent Acquisition MS Data." *Nature biotechnology* 32(3): 219–23. http://www.ncbi.nlm.nih.gov/pubmed/24727770.

Röst, Hannes L et al. 2016. "OpenMS: A Flexible Open-Source Software Platform for Mass Spectrometry Data Analysis." *Nature Methods* 13(9): 741–48. http://www.nature.com/articles/nmeth.3959.

Röst, Hannes L, Uwe Schmitt, Ruedi Aebersold, and Lars Malmström. 2015. "Fast and Efficient XML Data Access for Next-Generation Mass Spectrometry." *PloS one* 10(4): e0125108. http://www.ncbi.nlm.nih.gov/pubmed/25927999.

Sachsenberg, Timo et al. 2015. "MetaProSIP: Automated Inference of Stable Isotope Incorporation Rates in Proteins for Functional Metaproteomics." *Journal of proteome research* 14(2): 619–27. http://www.ncbi.nlm.nih.gov/pubmed/25412983.

Serang, Oliver, and Lukas Käll. 2015. "Solution to Statistical Challenges in Proteomics Is More Statistics, Not Less." *Journal of proteome research* 14(10): 4099–4103. http://www.ncbi.nlm.nih.gov/pubmed/26257019.

Serang, Oliver, Michael J. MacCoss, and William Stafford Noble. 2010. "Efficient Marginalization to Compute Protein Posterior Probabilities from Shotgun Mass Spectrometry Data." *Journal of Proteome Research* 9(10): 5346–57. http://pubs.acs.org/doi/abs/10.1021/pr100594k.

Sturm, M. et al. 2008. "OpenMS - an Open-Source Software Framework for Mass

Spectrometry." *BMC Bioinformatics* 9: 163.

Sturm, Marc, and Oliver Kohlbacher. 2009. "TOPPView: An Open-Source Viewer for Mass Spectrometry Data." *Journal of Proteome Research* 8(7): 3760–63. http://pubs.acs.org/doi/abs/10.1021/pr900171m.

Tabb, David L., Christopher G. Fernando, and Matthew C. Chambers. 2007. "MyriMatch: Highly Accurate Tandem Mass Spectral Peptide Identification by Multivariate Hypergeometric Analysis." *Journal of Proteome Research* 6(2): 654–61. http://pubs.acs.org/doi/abs/10.1021/pr0604054.

Tanner, Stephen et al. 2005. "InsPecT: Identification of Posttranslationally Modified Peptides from Tandem Mass Spectra." *Analytical Chemistry* 77(14): 4626–39. https://pubs.acs.org/doi/10.1021/ac050102d.

Teleman, Johan et al. 2015. "DIANA-Algorithmic Improvements for Analysis of Data-Independent Acquisition MS Data." *Bioinformatics* 31(4): 555–62.

The, Matthew, and Lukas Käll. 2016. "MaRaCluster: A Fragment Rarity Metric for Clustering Fragment Spectra in Shotgun Proteomics." *Journal of proteome research* 15(3): 713–20. http://www.ncbi.nlm.nih.gov/pubmed/26653874.

The, Matthew, Michael J. MacCoss, William S. Noble, and Lukas Käll. 2016. "Fast and Accurate Protein False Discovery Rates on Large-Scale Proteomics Data Sets with Percolator 3.0." *Journal of The American Society for Mass Spectrometry* 27(11): 1719–27. http://link.springer.com/10.1007/s13361-016-1460-7.

Di Tommaso, Paolo et al. 2017. "Nextflow Enables Reproducible Computational Workflows." *Nature Biotechnology* 35(4): 316–19. http://www.nature.com/articles/nbt.3820.

Uszkoreit, Julian et al. 2015. "PIA: An Intuitive Protein Inference Engine with a Web-Based User Interface." *Journal of Proteome Research* 14(7): 2988–97. http://pubs.acs.org/doi/10.1021/acs.jproteome.5b00121.

Walzthoeni, Thomas et al. 2012. "False Discovery Rate Estimation for Cross-Linked Peptides Identified by Mass Spectrometry." *Nature Methods* 9(9): 901–3. http://www.nature.com/articles/nmeth.2103.

Wein, Samuel et al. 2018. "A Computational Platform for High-Throughput Analysis of RNA Sequences and Modifications by Mass Spectrometry." *bioRxiv*: 501668. http://biorxiv.org/content/early/2018/12/20/501668.abstract.

Weisser, Hendrik et al. 2013. "An Automated Pipeline for High-Throughput Label-Free Quantitative Proteomics." *Journal of Proteome Research* 12(4): 1628–44.

Weisser, Hendrik, and Jyoti S Choudhary. 2017. "Targeted Feature Detection for Data-Dependent Shotgun Proteomics." *Journal of proteome research* 16(8): 2964–74. http://www.ncbi.nlm.nih.gov/pubmed/28673088.

Wishart, David S. et al. 2018. "HMDB 4.0: The Human Metabolome Database for 2018." *Nucleic Acids Research* 46(D1): D608–17.