An Efficient Data Format for Mass Spectrometry-Based Proteomics

Anuj R. Shah,[a] Jennifer Davidson,[b] Matthew E. Monroe,[a] Anoop M. Mayampurath,[c] William F. Danielson,[a] Yan Shi,[a] Aaron C. Robinson,[a] Brian H. Clowers,[d] Mikhail E. Belov,[a] Gordon A. Anderson,[a] and Richard D. Smith[a]

[a] Fundamental and Computational Sciences Directorate, Pacific Northwest National Laboratory, Richland, Washington, USA
[b] Department of Electrical Engineering and Computer Science, Oregon State University, Corvallis, Oregon, USA
[c] School of Informatics and Computing, Indiana University, Bloomington, Indiana, USA
[d] National Security Directorate, Pacific Northwest National Laboratory, Richland, Washington, USA

The diverse range of mass spectrometry (MS) instrumentation along with corresponding proprietary and nonproprietary data formats has generated a proteomics community driven call for a standardized format to facilitate management, processing, storing, visualization, and exchange of both experimental and processed data. To date, significant efforts have been extended towards standardizing XML-based formats for mass spectrometry data representation, despite the recognized inefficiencies associated with storing large numeric datasets in XML. The proteomics community has periodically entertained alternate strategies for data exchange, e.g., using a common application programming interface or a database-derived format. However, these efforts have yet to gain significant attention, mostly because they have not demonstrated significant performance benefits over existing standards, but also due to issues such as extensibility to multidimensional separation systems, robustness of operation, and incomplete or mismatched vocabulary. Here, we describe a format based on standard database principles that offers multiple benefits over existing formats in terms of storage size, ease of processing, data retrieval times, and extensibility to accommodate multidimensional separation systems. (J Am Soc Mass Spectrom 2010, 21, 1784–1788) © 2010 American Society for Mass Spectrometry

The wide range of mass spectrometers introduced in recent years by a number of different instrument manufacturers and vendors [1] and their proprietary nature of data formats makes data management (e.g., data retrieval, analysis, exchange across laboratories) and publication/verification of experimental proteomics results time consuming and expensive. As a result, multiple efforts have been dedicated to standardizing and adopting common file formats that are instrument agnostic and facilitate data exchange within and across laboratories [2–9]. The most commonly used formats for MS-based proteomics data exchange are mzXML [7], and increasingly mzML [4], both of which utilize eXtensible Markup Language (XML) documents.

By nature and design principles, XML [http://www.w3.org/TR/REC-xml/] is a simple, extensible, flexible, tag-based language primarily created for the electronic interpretation of text-based documents and data exchange across the Internet. XML documents are mostly human readable and can be opened with most text reader programs. Automated parsing of these documents is made commonplace by the availability of a plethora of software tools that read the documents in two distinct modes, either sequential or random; the latter, which utilizes a document object model, is only feasible for documents that fit into computer main memory. While devising a universal file format is a valid strategy, the choice of XML for representing large volumes of numeric data is debatable [3, 10]. For instance, processing data files that deal with large scale studies requires complex implementations that rely on memory saving techniques such as decoding into memory-efficient strings [8], memory-mapped files, user transparent memory-to-disk swapping routines, or streaming data access mechanisms [5]. Moreover, simple parsing and custom analysis tasks require the adoption and modification of complex proteomics software frameworks (e.g., proteomecommons [5], OpenMS [11], ProteoWizard [12], or Mzmine [6]) to access the algorithms and functionality. Another downside is that the existing XML-based standards do not easily accommodate for multidimensional separations techniques. For example, to store LC coupled ion mobility-based MS data in the mzXML for-

Address reprint requests to Dr. R. D. Smith, Biological Sciences Division, Pacific Northwest National Laboratory, 3335 Q Ave., (K8-98) P.O. Box 999, Richland, WA 99352, USA. E-mail: rds@pnl.gov

1044-0305/10/$32.00
doi:10.1016/j.jasms.2010.06.014

Published online July 7, 2010
Received April 20, 2010
Revised June 15, 2010
Accepted June 21, 2010

mat would require producing multiple XML files per analysis (one file for each ion mobility acquisition frame), which translates to a data management nightmare for scientists.

Alternate strategies include using a common application programming interface (API) like mzAPI [3]; a netCDF [13], or HDF5 [http://www.hdfgroup.org/HDF5/] based file format; or a relational database management systems (RDBMS)-based strategy. The use of such data access mechanisms is common in fields that generate large datasets. For example, standardized OpenGL API is used in the computer graphics field, MPI-based programming in high-performance computing codes; HDF5-based file formats have found acceptance in financial engineering [http://www.hdfgroup.org/HDF5/users5.html], satellite climate monitoring [14], and computational grids [www.globus.org], among others; and netCDF-based formats are widely used in international geosciences and education communities [15]. While viable, these approaches have not gathered sufficient momentum for acceptance in the MS proteomics community largely because none, including mzML, have demonstrated significant processing benefits compared with mzXML. Relational databases on the other hand are an attractive alternative. Databases have been studied in scientific literature for over 40 years, are very stable and relatively well understood by entry-level engineers and scientists. More importantly, they support the incorporation of binary data as blobs (a binary large object that can be stored as a single entity in a database), which translates to significant space savings when it comes to storing raw spectra. Additional advantages are a standard query language (SQL) that serves the purpose of a common API for accessing underlying data. The introduction of SQLite [http://www.sqlite.org/], a library of C functions that allows creation of relatively light-weight serverless databases as a single cross platform file, has opened up additional avenues for efficient data exchange formats.

Here we introduce a light RDBMS-based data format referred to as Yet Another Format for Mass Spectrometry (YAFMS) for evaluation and discussion among the proteomics community. The proposed format generates a file that is modified with minimal effort to include additional data, and accommodates for multidimensional experimental setups with ease. Additionally, the format includes the flexibility to leverage the ontologies developed as part of existing XML-based formats and can work as the underlying implementation for an API-based approach like mzAPI [3].

Keep It Simple

Raw mass spectrometry data are not complex data in that at the fundamental level they are a series of high precision numbers, i.e., mass/charge (m/z) ratios and intensity values. Data exchange challenges arise when an entire collection of spectra (e.g., collected across multiple separations, sample sets, etc.) needs to be repeatedly manipulated in a time- and space-efficient manner. In YAFMS, a four table relational schema captures all data elements involved in MS and tandem MS experiments (Figure 1).

The "Dataset_Info" table contains details about the experiment as key-value pairs, (e.g., the operator, the instrument, the laboratory).

The "Spectra_Info" table contains details about the acquired spectra and the separations methods, (e.g., liquid chromatography elution time or scan type).

The "Spectra_Data" table contains the raw spectra as binary objects (blobs) along with additional parameters such as the number of peaks, base peak intensity, base peak mass/charge (m/z), and total ion current (TIC), among others. These parameters are directly calculated from the spectra when creating YAFMS files and are easily updatable using standard SQL procedures and queries.

Importantly, the "Ontology" table affords cross compatibility with mzXML and mzML standards where the terms from the XML-based ontology can be mapped back to the names used in the individual tables. The software API to write YAFMS files allows the addition of terms that are not present in the controlled vocabulary, though a warning is issued for custom terminology.

The driving principles behind a minimalistic table structure were simplicity, ease of use, and flexibility on demand. Each of the four tables is indexed on specific columns that allow for fast data retrieval. The flexibility of YAFMS comes from the ability to extend the schema by adding tables or columns to existing tables to incorporate custom information. Extending the format to include multiple experiments is easy via a new "dataset_id" column in all tables. So, rather than transferring a set of small data files related to a single study, one could encode all files in a single YAFMS file. YAFMS files are also easy to update and modify. For example, researchers can disseminate custom information in the form of images or calculate more parameters from the spectra by creating new tables and/or appropriate columns using standard SQL queries. This has the obvious advantage that YAFMS files do not need to be recreated every time a new parameter is required. Also, updates to YAFMS files do not affect the indexing mechanism and data retrieval (in contrast, XML formats require re-indexing after updates or recreation of a new file). Another advantage of using this kind of relational model is the ability to represent multidimensional data. For example, the format is readily extensible to experiments that use multiple separation techniques like strong cation exchange chromatography, liquid chromatography, ion mobility separations, and combinations of such techniques. The added flexibility and extensibility of YAFMS could lead to incompatibility issues if sufficient safeguards are not introduced that prevent the ad hoc modification of column names, tables, or the core four-table schema, thereby rendering the format unusable. To this effect, we have included methods that check for compatibility with the proposed

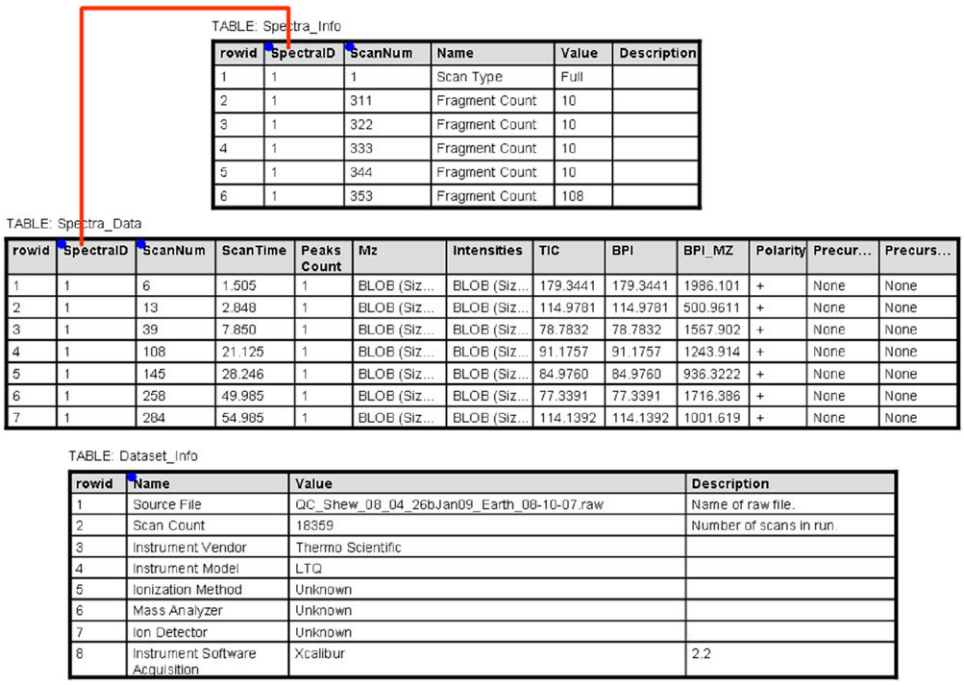

TABLE: Spectra_Info

rowid	SpectraID	ScanNum	Name	Value	Description
1	1	1	Scan Type	Full	
2	1	311	Fragment Count	10	
3	1	322	Fragment Count	10	
4	1	333	Fragment Count	10	
5	1	344	Fragment Count	10	
6	1	353	Fragment Count	108	

TABLE: Spectra_Data

rowid	SpectraID	ScanNum	ScanTime	Peaks Count	Mz	Intensities	TIC	BPI	BPI_MZ	Polarity	Precur...	Precurs...
1	1	6	1.505	1	BLOB (Siz...	BLOB (Siz...	179.3441	179.3441	1986.101	+	None	None
2	1	13	2.848	1	BLOB (Siz...	BLOB (Siz...	114.9781	114.9781	500.9611	+	None	None
3	1	39	7.850	1	BLOB (Siz...	BLOB (Siz...	78.7832	78.7832	1567.902	+	None	None
4	1	108	21.125	1	BLOB (Siz...	BLOB (Siz...	91.1757	91.1757	1243.914	+	None	None
5	1	145	28.246	1	BLOB (Siz...	BLOB (Siz...	84.9760	84.9760	936.3222	+	None	None
6	1	258	49.985	1	BLOB (Siz...	BLOB (Siz...	77.3391	77.3391	1716.386	+	None	None
7	1	284	54.985	1	BLOB (Siz...	BLOB (Siz...	114.1392	114.1392	1001.619	+	None	None

TABLE: Dataset_Info

rowid	Name	Value	Description
1	Source File	QC_Shew_08_04_26bJan09_Earth_08-10-07.raw	Name of raw file.
2	Scan Count	18359	Number of scans in run.
3	Instrument Vendor	Thermo Scientific	
4	Instrument Model	LTQ	
5	Ionization Method	Unknown	
6	Mass Analyzer	Unknown	
7	Ion Detector	Unknown	
8	Instrument Software Acquisition	Xcalibur	2.2

Figure 1. YAFMS Schema. Three tables are shown with sample data elements. The Dataset_info table stores experimental setup details, the instrument used and other metadata. The Spectra_Info table stores sparse information related to spectra that are not stored as columns under the Spectra_Data table. The Spectra_Data table stores the mass/charge ratios and intensities as two binary large objects (blobs) in the database, in conjunction with the total ion count, base peak intensity, base peak mass/charge, precursor mass/charge, scan time, etc. The red line indicates the link between Spectra_Info and Spectra_Data table. The primary key for this linking is a combination of SpectraID and ScanNum. The blue dot alongside a column indicates indexes built on those columns for fast retrieval. The ontology table is not shown/used in this example.

schema in the API software to inform users about the validity of the underlying YAFMS file(s). An empty database schema file is distributed with the API that contains the basic four-table structure, the correct column names, and primary keys for tables and indexes. Methods are provided in the YAFMS library to validate whether the schema conforms to the four-table structure proposed in this document. Fatal errors are raised when one of the four tables/columns is missing or the columns are named incorrectly while missing indexes are created on the fly. Extensions to the basic table structure are not reported as errors/warnings.

Smaller Is Better

The current rates of data generation are growing faster than data storage capabilities [16]. Surrounding large numeric data elements with tags in XML formats only exacerbates the problem as the file sizes are simply too large for most software programs and come coupled with costs for disk space usage, data compression/decompression, as well as with data transfer bottlenecks across the Internet/LAN.

Another shortcoming of XML formats is that they are text-based and cannot accommodate binary data. As a result, both mzXML and mzML represent spectra in a Base64 encoded hexadecimal compressed form, which is larger than the original binary data itself. Conversely, a relational database has the facility to store various data types and binary data are directly stored as blobs. The most efficient formats, such as the Thermo .RAW files are binary in nature and strive hard to produce the minimalist file necessary in terms of size. Storing the raw spectra as compressed binary objects in YAFMS further reduces the size of these files (relative to XML-based files), which procures a significant cost savings. We utilize the freely available LibLZF (http://oldhome.schmorp.de/marc/liblzf.html) compression libraries to achieve fast lossless compression/decompression of raw spectra. To read spectra from the YAFMS file, the blobs are decompressed using the libLZF libraries and converted to double precision *mz* values. One must note that the compression and decompression algorithm is loosely coupled with the API and as such its underlying implementation can be replaced with a different algorithm by simply changing the method signature.

Figure 2 compares file sizes across five datasets (Table 1) of varying spectral density (i.e., the number of peaks in a spectrum). These datasets, available from our website, were analyzed on different resolution instruments and scan counts range from 5000 (dataset A) to 90,000 (dataset E). Note that YAFMS file sizes were 25% to 60% smaller than their corresponding mzXML and mzML representations. The smallest gains in file sizes

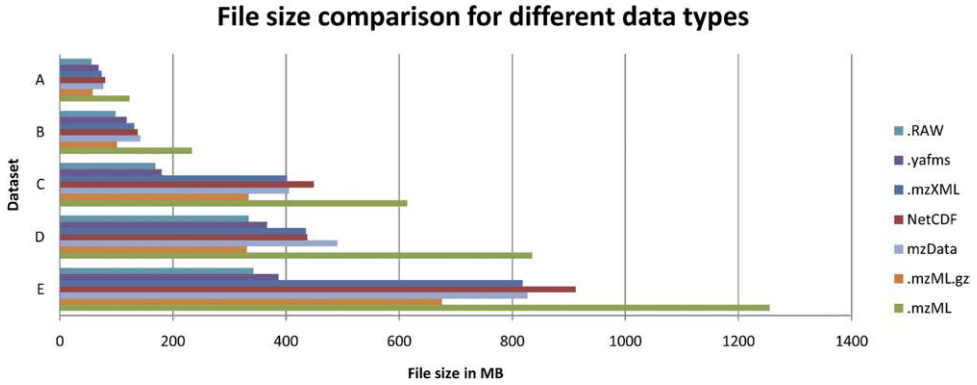

Figure 2. File size comparison for different data file formats. The YAFMS file sizes are comparable to the .RAW data formats and always significantly smaller than the mzXML and mzML data files (as much as 25%–30% samples with dense spectra and more than 50% in cases of sparse spectral density). The mzXML and mzML files were created using ProteoWizard release 1.4.0 [12]. The NetCDF files were created using Xcalibur 2.1.0 as distributed by ThermoFinnigan. The compressed mzML files (.mzmL.gz) were created with the Trans-proteomics pipeline (http://tools.proteomecenter.org/wiki/index.php?title = Software:TPP) and offer file sizes comparable to the .RAW files in some cases (A, B, and D). The mzData files were created using the openMS software framework [11].

(25%) are observed for samples that have larger numbers of peaks per spectra (datasets C and D), whereas relatively sparse spectra (dataset E) translate to highly compressed (60%) YAFMS files. In general, the largest compression in file sizes will be obtained for samples that contain large numbers of relatively sparse spectra, whereas small but significant gains will be obtained in cases of dense spectra. It is interesting to note that the compressed mzML data files are comparable in file size on some datasets (A, B, and D) but clearly underperform on datasets C and E.

Another metric of comparison is data retrieval within specific *m/z* and time ranges. For example, in the case of label-free quantitation, the most common operation would be the comparison of peak areas for extracted ion chromatographs. Table 1 lists the best read times for truly random *m/z* range access for different dataset representations across the same five datasets presented in Figure 2. The YAFMS format offers significantly lower read times on range queries compared with mzXML and .RAW formats as they are directly related to the time it takes to decompress binary spectra into numbers and more complex spectra require higher access times. Neither the .RAW data format nor the mzXML data formats facilitate fast range queries and their performance degrades with increasing file sizes and/or spectral complexity. It is harder to benchmark the read access times for the compressed mzML files as the files need to be first decompressed for reading; thus, a comparison is not presented in this manuscript.

An Easy to Use Software Suite

YAFMS is accompanied by downloadable, user friendly software processing modules that facilitate individual process steps. We developed a C# dynamic link library (dll) that provides functions for reading, writing and updating YAFMS files. A comprehensive set of instructions that articulate the minimal software setup required to work with this format are outlined in the README file distributed with the library. Additionally, we extended the FileConverter program from the OpenMS proteomics pipeline with modifications that facilitate conversion of mzData, mzXML, and mzML to YAFMS, developed a command line utility that converts .RAW files to this new file format, and extended the Decon2LS software [17] to read and process YAFMS files, allowing for visualization and deisotoping of

Table 1. Random access read times for popular data formats

Sample[a]	Scans	Spectra density (KB/scan)[b]	YAFMS (ms)[c]	RAW (ms)[d]	mzXML (ms)[e]
A	6878	24.00	0.106	6.432	0.570
B	18,359	5.18	0.869	4.825	5.129
C	5548	60.26	2.891	40.062	41.873
D	13,844	23.58	0.704	4.317	4.490
E	90,766	0.60	2.390	48.130	49.591

[a]Read times for five different samples were calculated using the average of 1000 randomly generated scan numbers and *m/z* ranges to the nearest microsecond using high resolution timing calls.
[b]Spectra density is calculated by dividing the total file size by the total number of scans. The average spectra density shows that samples C and D are highly complex samples.
[c,d,e]The Decon2LS application was used to read all data formats. Decon2LS uses the RAMP mzXML parser to read mzXML files, ThermoFinnigan's proprietary libraries for .RAW files and our custom dynamic link library to read YAFMS files.

spectra in both manual and automated modes. Users can manipulate and interact directly with YAFMS files via numerous reader utilities that are freely available on the Internet as popular browser add-ons (recommendations on our website); the one exception is the actual spectral data, since it is stored as a compressed blob. All of the datasets discussed in this manuscript and software, including source code for OpenMS extensions, the reader/writer dll, the .raw to YAFMS conversion utility, and the Decon2LS software package, can be downloaded from http://omics.pnl.gov/software/YAFMS.php and is released under the Open Source license agreement. Users can obtain the necessary source code from our website and integrate it with their existing applications.

Conclusion

We emphasize the advantages of using a light, serverless, relational database format such as YAFMS for proteomics data exchange purposes. We have illustrated that this file format is highly efficient in processing time, as well as in storage space. YAFMS not only allows novices to interact via simple browser add-ons, but also allows for data extraction and updates by writing simple SQL queries. Additionally, this format provides the flexibility to add tables that contain, for example, processed data, deconvolution results, or even images used in publications.

A potential challenge associated with the widespread adoption of YAFMS is that its ease of use could result in many evolving and mutually exclusive file formats and vocabularies (e.g., used in the key-value pairs in YAFMS) for representing metadata information associated with spectra and the experiment. However, adopting the controlled vocabulary (ontology) as defined by the HUPO-PSI for MS data and keeping the minimal table structure intact would ensure common semantics and to that effect we have provided methods in the API to ensure basic compliance. Additionally, our future tasks include the extension of the FileConverter program from openMS to convert from YAFMS to mzML and mzXML files. It is our hope that researchers, practitioners, and computer scientists/programmers evaluate this approach and use it for data exchange within small laboratories, consortiums, and collaborations.

References

1. Watson, J. T.; Sparkman, O. D. *Introduction to Mass Spectrometry: Instrumentation, Applications and Strategies for Data Interpretation*, 4th Edition; Wiley InterScience: 2007.
2. Garden, P.; Alm, R.; Hakkinen, J. PROTEIOS: An Open Source Proteomics Initiative. *Bioinformatics* **2005,** *21,* 2085–2087.
3. Askenazi, M.; Parikh, J. R.; Marto, J. A. mzAPI: A New Strategy for Efficiently Sharing Mass Spectrometry Data. *Nat. Methods* **2009,** *6,* 240–241.
4. Deutsch, E. mzML: A Single, Unifying Data Format for Mass Spectrometer Output. *Proteomics* **2008,** *8,* 2776–2777.
5. Falkner, J. A.; Falkner, J. W.; Andrews, P. C. ProteomeCommons.org IO Framework: Reading and Writing Multiple Proteomics Data Formats. *Bioinformatics* **2007,** *23,* 262–263.
6. Katajamaa, M.; Miettinen, J.; Oresic, M. MZmine: Toolbox for Processing and Visualization of Mass Spectrometry-Based Molecular Profile Data. *Bioinformatics* **2006**, btk039.
7. Pedrioli, P. G. A.; Eng, J. K.; Hubley, R.; Vogelzang, M.; Deutsch, E. W.; Raught, B.; Pratt, B.; Nilsson, E.; Angeletti, R. H.; Apweiler, R.; Cheung, K.; Costello, C. E.; Hermjakob, H.; Huang, S.; Julian, R. K.; Kapp, E.; McComb, M. E.; Oliver, S. G.; Omenn, G.; Paton, N. W.; Simpson, R.; Smith, R.; Taylor, C. F.; Zhu, W.; Aebersold, R. A Common Open Representation of Mass Spectrometry Data and Its Application to Proteomics Research. *Nat. Biotech.* **2004,** *22,* 1459–1466.
8. Prince, J. T.; Marcotte, E. M. mspire: Mass Spectrometry Proteomics in Ruby. *Bioinformatics* **2008,** *24,* 2796–2797.
9. Cannataro, M.; Veltri, P. SpecDB: A Database for Storing and Managing Mass Spectrometry Proteomics Data. In *Fuzzy Logic and Applications*. Springer: Berlin/Heidelberg, 2006;236–245.
10. Lin, S. M.; Zhu, L.; Winter, A. Q.; Sasinowski, M.; Kibbe, W. A. What is mzXML Good For? *Expert Rev. Proteom.* **2005,** *2,* 839–845.
11. Sturm, M.; Bertsch, A.; Gropl, C.; Hildebrandt, A.; Hussong, R.; Lange, E.; Pfeifer, N.; Schulz-Trieglaff, O.; Zerck, A.; Reinert, K.; Kohlbacher, O. OpenMS—An Open-Source Software Framework for Mass Spectrometry. *BMC Bioinformatics* **2008,** *9,* 163.
12. Kessner, D.; Chambers, M.; Burke, R.; Agus, D.; Mallick, P. ProteoWizard: Open Source Software for Rapid Proteomics Tools Development. *Bioinformatics* **2008,** *24,* 2534–2536.
13. Rew, R.; Davis, G. NetCDF: An Interface for Scientific Data Access. Computer Graphics and Applications. *IEEE* **1990,** *10,* 76–82.
14. Kaspar, F.; Schultz, J.; Hollmann, R.; Schroder, M.; Muller, R.; Karlsson, K. G.; Roebeling, R.; Riihela, A.; Paepe, B.d.; Stockli, R. Satellite-Based Datasets for Validation of Regional Climate Models: CM-SAF Product Suite and New Tools for Processing. *Geophys. Res. Abstracts* **2009,** *11,* 11948.
15. Nativi, S.; Caron, J.; Davis, E.; Domenico, B. Design and Implementation of netCDF Markup Language (NcML) and Its GML-Based Extension (NcML-GML). *Comput. Geosci.* **2005,** *31,* 1104–1118.
16. Kouzes, R. T.; Anderson, G. A.; Elbert, S. T.; Gorton, I.; Gracio, D. K. The Changing Paradigm of Data-Intensive Computing. *Computer* **2009,** *42,* 26–34.
17. Jaitly, N.; Mayampurath, A. M.; Littlefield, K.; Adkins, J. N.; Anderson, G. A.; Smith, R. D. Decon2LS: An Open-Source Software Package for Automated Processing and Visualization of High Resolution Mass Spectrometry Data. *BMC Bioinformatics* **2009,** *10,* 87.