mz5: Space- and Time-efficient Storage of Mass Spectrometry Data Sets*[S]

Mathias Wilhelm‡§¶‖, Marc Kirchner‡¶‖‡‡, Judith A. J. Steen‡§§, and Hanno Steen‡¶‡‡**

Across a host of MS-driven-omics fields, researchers witness the acquisition of ever increasing amounts of high throughput MS data and face the need for their compact yet efficiently accessible storage. Addressing the need for an open data exchange format, the Proteomics Standards Initiative and the Seattle Proteome Center at the Institute for Systems Biology independently developed the *mzData* and *mzXML* formats, respectively. In a subsequent joint effort, they defined an ontology and associated controlled vocabulary that specifies the contents of MS data files, implemented as the newer *mzML* format. All three formats are based on XML and are thus not particularly efficient in either storage space requirements or read/write speed. This contribution introduces *mz5*, a complete reimplementation of the *mzML* ontology that is based on the efficient, industrial strength storage backend HDF5. Compared with the current *mzML* standard, this strategy yields an average file size reduction to ~54% and increases linear read and write speeds ~3–4-fold. The format is implemented as part of the *ProteoWizard* project and is available under a permissive Apache license. Additional information and download links are available from http://software.steenlab.org/mz5. ***Molecular & Cellular Proteomics 11: 10.1074/mcp.O111.011379, 1–5, 2012.***

MS data are acquired on a wide variety of mass analyzer technologies and brands that deliver data sets in various proprietary data formats and make use of a multitude of architecture-dependent libraries. In the past, this situation has severely complicated the development and application of alternative, vendor-independent data analysis pipelines (1–3), highlighting the need for and fueling the development of a common and open storage format for MS data sets (4, 5).

The Proteomics Standards Initiative and the Seattle Proteome Center at the Institute for Systems Biology independently developed the *mzData* and *mzXML* formats, respectively (4, 6). They subsequently merged their efforts, leading to the development of *mzML* (7), which features a generic ontology for the representation of MS data. The *mzML* format is universally applicable, and combined with the readily available open source reference implementation *ProteoWizard* (1), it significantly simplifies data import and export as well as general data handling. Most notably, *mzML* introduces a controlled vocabulary that enables the addition of novel as well as user-defined data types without requiring changes to the underlying XML schema. As a consequence, laboratories are able to store, analyze, and share MS data using an open exchange format, even for highly specialized workflows (8–11).

Continuous improvements in mass resolution and acquisition speed pose a serious challenge for existing data formats: current MS setups commonly acquire several hundreds of megabytes of data for each run, and space requirements for complete proteomics experiments easily exceed tens of gigabytes. As a consequence, the space and time efficiency of data format implementations have become increasingly critical.

Although based on excellent ontologies, relying on the extended markup language (XML)[1] for the straightforward implementation of *mzData*, *mzXML*, and *mzML* makes for a major efficiency bottleneck. XML was designed to be a human readable, textual data format with considerable inherent verbosity and redundancy. XML was not designed for efficient bulk data storage, and the general modus operandi requires reading complete files to construct the XML parse tree. The *mzXML* and *mzML* formats partly circumvent these limitations by using base-64 encoding and (optional) compression of the raw MS scan data in combination with an application-specific indexing system. Despite the improvements gained from these efforts, vendor formats in general outperform *mzXML* and *mzML* in terms of space requirements, as well as in read and write efficiency.

This contribution introduces *mz5*, a novel data representation that combines the merits of the *mzML* ontology with the efficiency of the Hierarchical Data Format (HDF5) (12). HDF5 is an established industrial standard for efficient storage and

From the ‡Proteomics Center, Children's Hospital Boston, Boston, Massachusetts, the §Faculty of Technology, University Bielefeld, Bielefeld, Germany, the ¶Department of Pathology, Children's Hospital Boston, Boston, Massachusetts, the ‡‡Department of Pathology, Harvard Medical School, Boston, Massachusetts, and the §§Department of Neurobiology, Harvard Medical School and F. M. Kirby Neurobiology Center, Children's Hospital, Boston, Massachusetts

Received June 13, 2011, and in revised form, September 14, 2011
Published, MCP Papers in Press, September 29, 2011, DOI 10.1074/mcp.O111.011379

[1] The abbreviations used are: XML, extended markup language; HDF5, Hierarchical Data Format, version 5; I/O, Input/Output.

retrieval of large amounts of complex data and is based on a portable binary representation. HDF5 was specifically designed for large data sets; the key features used in our *mz5* implementation are its native support for compression and cached partial read and write access. HDF5 library implementations are available for a range of languages and computational platforms, and the format is distributed under a permissive licensing scheme. Introduced more than 20 years ago, the continued use of HDF5 as the standard format for the NASA earth observing system, as well as its adaption as a prime format for data intensive applications in fields as diverse as astronomy, geology, remote sensing, and avionics (13–16), ensures broad, ongoing use and technical support for many years to come.

Our analyses show that *mz5* dramatically outperforms the XML-based representations of the *mzXML* and *mzML* formats in terms of both space requirements and Input/Output (I/O) speed and is competitive with proprietary vendor formats in terms of space requirements (Fig. 1). Because *mz5* is implemented inside the *ProteoWizard* (1) framework, it is already available for use by the mass spectrometry community, and *mz5* files can immediately be created from all open or proprietary formats supported by *ProteoWizard* (Fig. 2).

EXPERIMENTAL PROCEDURES AND RESULTS

Twenty LC-MS/MS runs of fractionated HeLa S3 cell lysate were acquired in data-dependent acquisition mode on an LTQ/Orbitrap classic (Thermo Scientific) system hyphenated to a nanoflow HPLC system (Eksigent). The survey scan range was limited to *m/z* 400–2000 at a nominal resolution of 60,000, and MS/MS scans were limited to the eight most abundant precursor ions. The resulting raw data files were converted to *mzML* using the *ProteoWizard* (1) tool set, *mzML* files were converted to *mz5*, and the identity of the *mzML* and *mz5* file contents was confirmed. For all formats, the data were stored in compressed double precision. For the *mzXML* and *mzML* formats, indexing was turned on. All of the experiments were conducted using the same hardware, with all resources dedicated to the test runs (no parallel jobs). Overall, we consider the selected data set to be a reasonably common example of LC-MS/MS data acquisition and hence expect that the results generalize to a large majority of practical use cases.

To assess *mz5* performance, we conducted three types of comparative experiments, testing (i) write performance; (ii) read performance; and (iii) space requirements. For each of the 20 input files, we generated a set of 10 subsets of increasing sizes (sets of 800, 1600, up to 8000 spectra, and the complete set of all spectra present in the respective input file). This strategy yielded 220 test cases. All of the time measurements were averaged over 10 repeats for every test case, after discarding the minimum and maximum timings. For performance comparisons between different formats, we fitted linear models with intercepts and determined relative slopes over time and space (see supplemental materials). *mz5* files were used to generate each test case for *mzXML*, *mzML*, and *mz5*, to guarantee that all three file formats contain the same data.

To test linear read performance, we measure the time it takes to read each of the test cases from the file it is stored in. Fig. 1*a* shows the linear read times of all test cases *versus* the number of (*m/z*, abundance) pairs. The *mz5* format outperforms the *mzXML* and *mzML* formats by factors of ~2.3 and ~3.7, respectively. We note in passing that the *mz5* gains over *mzML* are not strictly linear; this indicates optimization opportunities for the parsing of smaller *mzML* files. Interestingly, the *mz5* linear read times also exhibit smaller variation compared with *mzXML* and, in particular, *mzML* read times.

For strictly random reading, *mz5* is significantly faster than *mzML* but performs slightly slower than *mzXML*. This is due to the underlying HDF5 caching/compression tradeoff. We focused on the vast majority of conceivable use cases and geared the tradeoff toward efficient linear reading. The random read and initialization performance tests, as well as benchmarks for different mass analyzer types, can be found in the supplemental materials.

To test write performance, we extracted every test case from the respective *mz5* input file and measured the time it takes to write the data to disk, using *mzXML*, *mzML*, and *mz5*. Consequently, potential read time delays only cause a constant offset that is identical between the different formats and does not affect the slopes. supplemental Fig. 1*a* illustrates the results for all 220 test cases. Time measurements for *mzXML*, *mzML*, and *mz5* are shown as *diamonds*, *triangles*, and *circles*, respectively. It is apparent that *mz5* is consistently faster than *mzXML* and *mzML*. The read time-corrected relative slopes show improvements of a factor of ~4.7 for *mz5* over *mzXML* and ~3.9 over *mzML* (Table I).

Fig. 1*b* illustrates storage consumption *versus* the number of (*m/z*, abundance) pairs. It is evident that the *mz5* format requires little more than half the disk space (54%) of the *mzXML* and *mzML* formats. The *mzXML* and *mzML* storage requirements are practically identical. Fig. 1*c* illustrates that *mz5* even supersedes XML-based formats that are compressed at the file and spectrum levels for off-line long term storage. Hence, *mz5* provides better compression levels while still offering immediate data access and avoiding the need for time-consuming file level compression and decompression.

Fig. 1*c* also provides a rough comparison of space requirements for three popular proprietary vendor formats (Thermo raw, Bruker yep, and ABSciex wiff) against the open *mzXML*, *mzML*, and *mz5* formats. The *mz5* format compares very favorably against the Thermo and Bruker formats and is the best open alternative for the ABSciex format. The Bruker yep file data was generated from *Escherichia coli* whole cell lysate and was acquired on a Bruker amaZon instrument. The ABSciex wiff file data stems from subcellular fractions of a HeLa cell lysate and was analyzed by LC/MS on an TripleTOF 5600.

Implementation—The *mz5* format maps the *mzML* ontology, as implemented in the *ProteoWizard* library, into a collection of HDF5 objects. The HDF5 library provides HDF5 object persistence.

The design of *mz5* is tailored to meet two specific goals: to create an open exchange format with both I/O speed and storage space requirements that rival existing vendor formats. The practical challenge was to develop data representations that enable the use of the HDF5 properties conducive to the above goals. In *mz5*, we represent *mzML* tags as HDF5 compound data types, which correspond to classical abstract data types. Collections of compounds are stored in instances of a multidimensional array type (H5::data set). Instances are compressed and stored to disk. It should be noted that the straightforward representation of *mzML* tags in terms of HDF5 groups is not a viable solution, because such an approach would prohibit data compression for technical reasons.

The HDF5 library is already highly optimized for file I/O (17). The *mz5* implementation guarantees optimized buffer sizes to minimize the number of necessary I/O operations, collecting raw data before bulk writes are performed. HDF5 natively supports compression filters, and to optimize compression rates, our implementation makes heavy use of HDF5 data types that are amenable to compression, in particular avoiding variable length data types where possible. In addition, *mz5* removes zero intensity scans and encodes *m/z* measurements in a delta mass representation, storing distances between consecutive *m/z* observations. The latter two measures yield a stor-

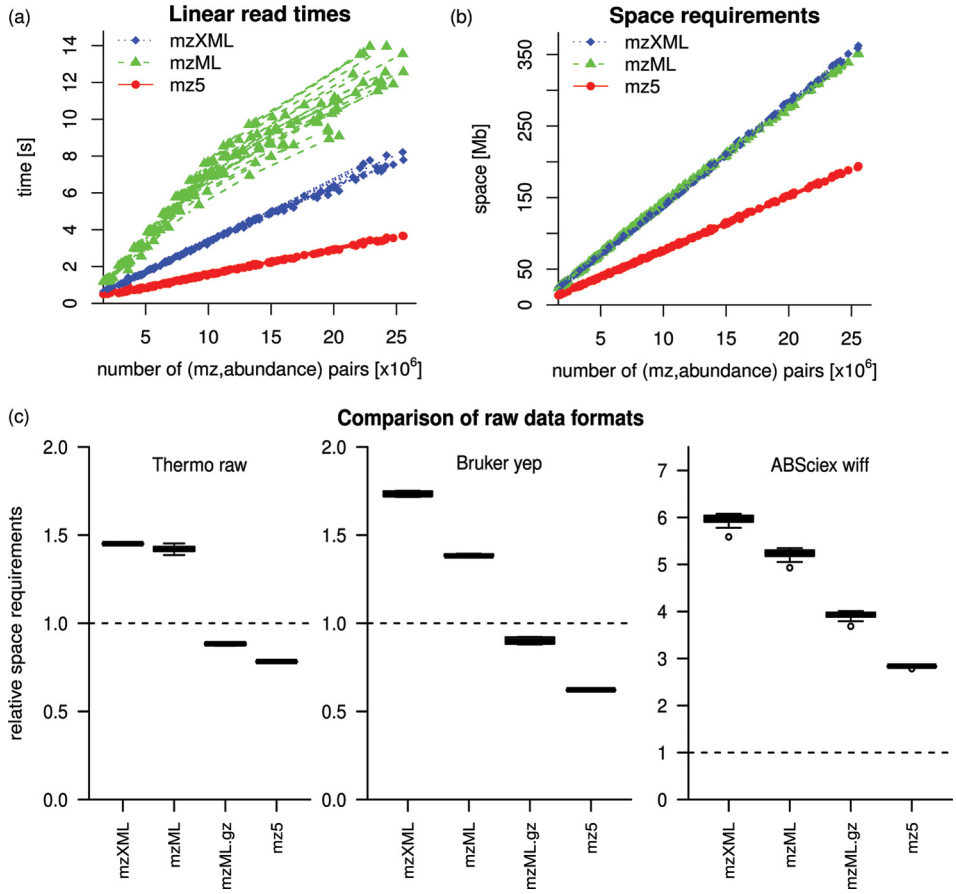

FIG. 1. **Linear read/write times and storage space requirements for different file formats on 22 file fragments over 10 repeats.** Linear read and write speeds are measured in 10^6 (*m/z*, abundance) pairs/s; storage space is measured in MB/10^6 (*m/z*, abundance) pairs. All *mz5*, *mzXML*, and *mzML* measurements are shown as *circles*, *diamonds*, and *triangles*, respectively. *a*, the *mz5* format exhibits a 2.28-fold increase in reading speed compared with *mzXML* and a 3.65-fold increase compared with *mzML*. *b*, *mz5* roughly halves the storage space requirements when compared with *mzXML* (53%) and *mzML* (55%). *c*, space requirements for *mzXML*, *mzML*, *mzML* with file level compression (gzip), and *mz5* relative to Thermo raw, Bruker yep, and ABSciex wiff format (*dashed line*). The *mz5* format minimizes storage requirements while providing fast raw data I/O.

TABLE I

Average read/write times and space requirements for 10^6 (m/z, abundance) pairs for the mz5, mzML, and mzXML file formats

The rows correspond to the linear read time (t_{lr}), the write time (t_w), the *mz5* read time − corrected write time ($t_w - t_{lr}^{mz5}$) and the space requirements. All of the timing values are in seconds or MB/10^6 data points, respectively. Columns 4 and 5 show that *mz5* outperforms the *mzML* and *mzXML* formats in all respects.

	mz5	*mzML*	*mzXML*	*mzML/mz5*	*mzXML/mz5*
t_{lr}	0.13	0.49	0.3	3.65	2.28
t_w	0.63	2.08	2.46	3.28	3.9
$t_w - t_{lr}^{mz5}$	0.5	1.95	2.33	3.89	4.66
Space	7.57	13.72	14.13	1.81	1.87

age requirement reduction of 55%. Tests showed that there were no numerical errors introduced by delta storage.

The implementation at hand features considerable versatility: it is possible to adapt the data type of raw data (float/double), configure compression level and filtering strategies, and configure HDF5 buffer sizes. The default configuration provides a reasonable parameter setting for most common use cases. *mz5* is currently limited to collections of (*m/z*, abundance) and (time, abundance) for spectral and chromatographic measurements, respectively. The extension to other forms of measurements is possible and straightforward.

Application—For practical applications, the main impact of the *mz5* format is likely going to be the significant decrease in storage space requirements. Many proteomics core facilities have substantial data storage and archiving costs, and the efficient use of available space can help to reduce these costs. Although standard database searches as they are currently performed are unlikely to benefit from increased read and write speeds, we expect that *mz5* will be of particular interest for spectral library searches, which are likely to become more relevant in the future (18, 19). Furthermore, *mz5* will be of interest for data processing pipeline setups because the read and

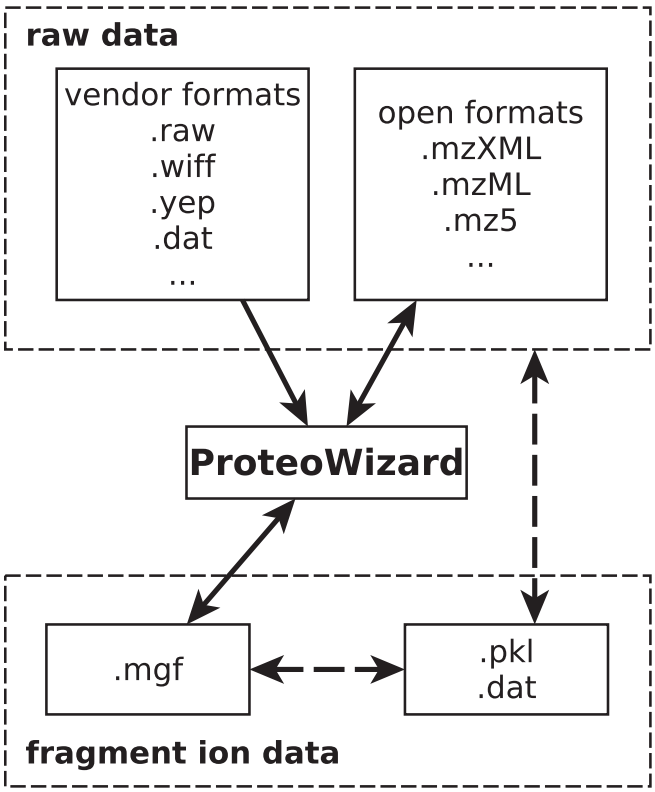

FIG. 2. **The *mz5* format and its relation to common input and output formats.** Format conversions that are supported by the *ProteoWizard* (1) framework are represented by *solid arrows*; format conversions that currently require additional software are shown with *dashed arrows*. Because the *mz5* format has been implemented within the *ProteoWizard* framework, it can readily be generated from a wide range of proprietary vendor raw formats.

write speeds and storage space requirements offered by *mz5* will increase the efficiency of data exchange between different components of a tool chain and reduce the impact caused by data conversion from proprietary vendor formats to an open standard.

To enable the transparent, straightforward use of the *mz5* format in existing and future software applications, we have implemented *mz5* within the *ProteoWizard* library; hence, any application that already makes use of *ProteoWizard*-based transparent data file access will automatically support *mz5*. This eliminates the need for any *mz5*-specific conversion tools or software adaptions because the *ProteoWizard* interface already enables the use of many proprietary and open formats and requires only a single implementation effort on the user side. We have successfully used the *mz5*/*ProteoWizard* combination under Windows, Linux, and Mac OSX.

Summary—This contribution introduces an efficient open data format for bulk mass spectrometry storage termed *mz5*. It combines the merits of HDF5, an established industry standard, and the *mzML* ontology developed by the HUPO Proteomics Standards Initiative. The *mz5* format offers dramatically faster I/O than the *mzXML* and *mzML* formats and requires only approximately half (54%) the storage space. The current *mz5* implementation is fully integrated into the *ProteoWizard* library and supports conversion operations for all proprietary data formats supported by *ProteoWizard* itself. All software is available from: http://software.steenlab.org/mz5.

The *mz5* format is a first step toward providing extended capabilities for MS data storage. Depending on future *mzML* ontology development and forthcoming HDF functionality, this may include physical and logical merging of large experimental data sets, distributed read and write access for high throughput workflows, and novel strategies for large data repositories (20, 21).

Acknowledgments—We thank Matthew Chambers from the Tabb Lab (funded by National Institutes of Health Grant R01-CA126218-05) for substantial *ProteoWizard* integration support and Fiona Pachl and Bernhard Kuster for providing Bruker yep data sets.

* This work was supported, in whole or in part, by National Institutes of Health Grants R01-NS066973-01 (to J. S.) and R01-GM094844-01 (to H. S.). This work was also supported by Alexander von Humboldt Foundation Grant DEU/1134241 (to M. K.). The costs of publication of this article were defrayed in part by the payment of page charges. This article must therefore be hereby marked "*advertisement*" in accordance with 18 U.S.C. Section 1734 solely to indicate this fact.

S This article contains supplemental text, Table 1, and Figs. 1–5.

‖ These authors contributed equally to this work.

** To whom correspondence should be addressed. E-mails: mathias.wilhelm@childrens.harvard.edu, marc.kirchner@childrens.harvard.edu, judith.steen@childrens.harvard.edu, hanno.steen@childrens.harvard.edu.

REFERENCES

1. Kessner, D., Chambers, M., Burke, R., Agus, D., and Mallick, P. (2008) ProteoWizard: Open source software for rapid proteomics tools development. *Bioinformatics* **24,** 2534–2536
2. Deutsch, E. W., Mendoza, L., Shteynberg, D., Farrah, T., Lam, H., Tasman, N., Sun, Z., Nilsson, E., Pratt, B., Prazen, B., Eng, J. K., Martin, D. B., Nesvizhskii, A. I., and Aebersold R. (2010) A guided tour of the trans-proteomic pipeline. *Proteomics* **10,** 1150–1159
3. Bertsch, A., Gröpl, C., Reinert, K., and Kohlbacher, O. (2011) OpenMS and TOPP: Open source software for LC-MS data analysis. *Methods Mol. Biol.* **696,** 353–367
4. Pedrioli, P. G., 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., and Aebersold, R. (2004) A common open representation of mass spectrometry data and its application to proteomics research. *Nat. Biotechnol.* **22,** 1459–1466
5. Orchard, S., Montechi-Palazzi, L., Deutsch, E. W., Binz, P. A., Jones, A. R., Paton, N., Pizarro, A., Creasy, D. M., Wojcik, J., and Hermjakob, H. (2007) Five years of progress in the standardization of proteomics data 4th annual spring workshop of the HUPO-proteomics standards initiative April 23–25, 2007 Ecole Nationale Suprieure (ens), Lyon, France. *Proteomics* **7,** 3436–3440
6. PSI-MS: Mass Spectrometer Standards Working Group (2010) http://www.psidev.info/index.php?q=node/80
7. Martens, L., Chambers, M., Sturm, M., Kessner, D., Levander, F., Shofstahl, J., Tang, W. H., Römpp, A., Neumann, S., Pizarro, A. D., Montecchi-Palazzi, L., Tasman, N., Coleman, M., Reisinger, F., Souda, P., Hermjakob, H., Binz, P. A., and Deutsch, E. W. (2011) mzML: A community standard for mass spectrometry data. *Mol. Cell. Proteomics* **10,** R110.000133
8. Orchard, S. (2009) Data deposition as an integral part of the publication process. *J. Proteomics Bioinf.* **2,** 334–335
9. Jones, P., Côté, R. G., Cho, S. Y., Klie, S., Martens, L., Quinn, A. F., Thorneycroft, D., and Hermjakob, H. (2008) PRIDE: New developments and new datasets. *Nucleic Acids Res.* **36,** D878–D883
10. Desiere, F., Deutsch, E. W., Nesvizhskii, A. I., Mallick, P., King, N. L., Eng, J. K., Aderem, A., Boyle, R., Brunner, E., Donohoe, S., Fausto, N., Hafen, E., Hood, L., Katze, M. G., Kennedy, K. A., Kregenow, F., Lee, H., Lin, B., Martin, D., Ranish, J. A., Rawlings, D. J., Samelson, L. E., Shiio, Y., Watts, J. D., Wollscheid, B., Wright, M. E., Yan, W., Yang, L., Yi, E. C., Zhang, H., and Aebersold, R. (2005) Integration with the human genome of peptide sequences obtained by high-throughput mass spectrometry. *Genome Biol.* **6,** R9

11. Deutsch, E. W., Lam, H., and Aebersold, R. (2008) PeptideAtlas: A resource for target selection for emerging targeted proteomics workflows. *EMBO Rep.* **9,** 429–434
12. The HDF Group (2000–2010), Hierarchical data format version 5. http://www.hdfgroup.org/HDF5
13. Millard, B. L., Niepel, M., Menden, M. P., Muhlich, J. L., and Sorger, P. K. (2011) Adaptive informatics for multifactorial and high-content biological data. *Nat. Methods* **8,** 487–493
14. Anderson, K., Alexov, A., Baehren, L., Griessmeier, J. M., Wise, M., and Renting, A. (2010) LOFAR and HDF5: Toward a new radio data standard. *Int. SKA Forum* **2010**
15. Bauer, B., Carr, L. D., Evertz, H. G., Feiguin, A., Freire, J., Fuchs, S., Gamper, L., Gukelberger, J., Gull, E., Guertler, S., Hehn, A., Igarashi, R., Isakov, S. V., Koop, D., Ma, P. N., Mates, P., Matsuo, H., Parcollet, O., Pawlowski, G., Picon, J. D., Pollet, L., Santos, E., Scarola, V. W., Schollwöck, U., Silva, C., Surer, B., Todo, S., Trebst, S., Troyer, M., Wall, M. L., Werner, P., and Wessel, S. (2011) The ALPS project release 2.0: Open source software for strongly correlated systems. *J. Stat. Mech. Theory Exp.* **2011,** P05001
16. Dougherty, M. T., Folk, M. J., Zadok, E., Bernstein, H. J., Bernstein, F. C., Eliceiri, K. W., Benger, W., and Best, C. (2009) Unifying biological image formats with HDF5. *Commun. ACM* **52,** 42–47
17. Howison, M., Koziol, Q., Knaak, D., Mainzer, J., and Shalf, J. (2010) Tuning HDF5 for lustre file systems. *Workshop on Interfaces and Abstractions for Scientific Data Storage (IASDS10)*
18. Bandeira, N., Tsur, D., Frank, A., and Pevzner, P. A. (2007) Protein identification by spectral networks analysis. *Proc. Natl. Acad. Sci. U.S.A.* **104,** 6140–6145
19. Lam, H., Deutsch, E. W., Eddes, J. S., Eng, J. K., King, N., Stein, S. E., and Aebersold, R. (2007) Development and validation of a spectral library searching method for peptide identification from ms/ms. *Proteomics* **7,** 655–667
20. Askenazi, M., Webber, J. T., and Marto, J. A. (2011) mzServer: Web-based programmatic access for mass spectrometry data analysis. *Mol. Cell. Proteomics* **10,** M110.003988
21. Webber, J. T., Askenazi, M., and Marto, J. A. (2011) mzResults: An interactive viewer for interrogation and distribution of proteomics results. *Mol. Cell. Proteomics* **10,** M110.003970