analytical chemistry

Comment

pubs.acs.org/ac

Comment on "Computation of Isotopic Peak Center-Mass Distribution by Fourier Transform"

Han Hu,[†,‡,$] Piotr Dittwald,[§,∥,$] Joseph Zaia,[‡] and Dirk Valkenborg*[,⊥,⊗,#]

[†]Bioinformatics Program, Boston University, Boston, Massachusetts 02118, United States

[‡]Center for Biomedical Mass Spectrometry, Boston University School of Medicine, Boston University, Boston, Massachusetts 02118, United States

[§]College of Inter-Faculty Individual Studies in Mathematics and Natural Sciences, University of Warsaw, Warsaw, Poland

[∥]Institute of Informatics, University of Warsaw, Warsaw, Poland

[⊥]Applied Bio & Molecular Systems, Vlaamse Instelling Voor Technologisch Onderzoek (VITO), Mol, Belgium

[⊗]Interuniversity Institute for Biostatistics and Statistical Bioinformatics, Hasselt University, Diepenbeek, Belgium

[#]Center for Proteomics, Antwerp, Belgium

Anal. Chem. **2012**, *84* (16), 7052−7056. DOI: 10.1021/ac301296a

Supporting Information

Jorge Fernandez-de-Cossio Diaz and Jorge Fernandez-de-Cossio recently published[1] an algorithm to calculate the aggregated isotope distribution and center-masses based on the molecular formula and elemental isotope distribution that could serve their ISOTOPICA software.[2−4] The presented approach is rooted in the polynomial methods as proposed by Brownawell and Fillippo[5] and Yergey et al.[6] Furthermore, Rockwood and Haimi suggested to utilize the polynomial methods to calculate accurate masses of isotopic peaks.[7] Recently, Claesen et al. generalized the method of Rockwood and Haimi as a polynomial generating function to calculate exact center-masses as applied in the BRAIN method.[8−11] Fernandez-de-Cossio Diaz and Fernandez-de-Cossio solve the polynomial specified in eqs 18−23 by Claesen et al.[8] for the center-masses using the fast Fourier transform approach (FFT) instead of employing an iterative scheme as the case in BRAIN. The operation of the FFT-approach to solve the polynomial generating function opened the possibility for additional improvements in their procedure. The FFT-approach,[1] which implements the center-mass calculation will be referred to as FTMC in the remainder of this comment.

Fernandez-de-Cossio Diaz and Fernandez-de-Cossio compare the BRAIN method with the FFT-approach in terms of a computational performance study. To facilitate a better insight in the various developments in FTMC and their impact on computational performance, we will make a list of all these aspects separately and explain how they benefit numerical efficiency. Further, we will provide an implementation of BRAIN in C++ and compare the speed of the algorithm with the BRAIN implementation in R. We also contrast the performance of BRAIN in C++ and FTMC, for which latter is implemented C#. The most important adjustments introduced in FTMC that have an influence on the computational speed are listed.

(1) Algorithm to Solve the Polynomial Generating Function. Fernandez-de-Cossio Diaz and Fernandez-de-Cossio proposed to solve the polynomial generating function by means of a FFT-based approach. A comparison in terms of the theoretical limits was conducted to illustrate the worst-case running time complexity of the algorithm in function of the number of computed peaks N. As a result, FTMC obtained an order complexity of $N\log(N)$ while the iterative formulas in the BRAIN method had an order complexity of N.[2]

This analysis indicates that FTMC has a better order of complexity than BRAIN. However, the actual running times of the exact algorithm does depend on multiple factors and does not reflect the theoretical order complexity of the FFT algorithm necessarily. For example, the FTMC method transduces the mass m of a molecule to a number of peaks that needs to be returned by the calculation, resulting in a complexity of $\sqrt{m}\log(m)$. The number of computed peaks and molecular mass are related parameters that are derived from each other with a particular heuristic. Obviously, this heuristic has an influence on the speed of the algorithm and depends on the application or the molecular classes for which an isotope distribution is required. For this purpose, we evaluate some heuristics that relate the mass to the number of computed peaks.

(2) Number of Computed Peaks. The number of aggregated isotope variants that are required in the calculation has an enormous impact on the computational efficiency. The BRAIN method[8] uses a simple but coarse heuristic to calculate the number of computed peaks:

$$N = \max(\lceil 2x(\mathrm{mass}_{\mathrm{average}} - \mathrm{mass}_{\mathrm{monoisotopic}})\rceil, 5) \quad (1)$$

where $\lceil x\rceil$ indicates the ceiling function to the smallest following integer. However, Fernandez-de-Cossio Diaz and Fernandez-de-Cossio[1] pointed out that the proposed heuristic in eq 1 for small molecules is not good enough since the coverage of the aggregated distribution is too small. To overcome this limitation we propose following formula

$$N = \max(\lceil 2x(\mathrm{mass}_{\mathrm{average}} - \mathrm{mass}_{\mathrm{monoisotopic}})\rceil, 50) \quad (2)$$

Published: November 4, 2013

The heuristics in eqs 1 and 2 cause the range spanned by BRAIN to scale linearly in function of the mass. This heuristic is chosen such that it returns a balanced number of aggregated isotope variants before and after the average mass of a molecule. The rationale for this heuristic is that the BRAIN algorithm was tested by comparing the theoretical average mass with the empirical average mass obtained from a probability weighted sum of the returned isotope distribution.[8,12] Note that this heuristic has little practical value in a real application and is impaired for efficient calculations.

A restriction imposed by the iterative algorithm of the BRAIN method is that it requires to start the calculation from the lightest isotope variant (i.e., *N* equals the heaviest isotope variant). This requirement is regarded as a limitation of the BRAIN method. It should be noted that FTMC which solves the polynomial generating function by a Fourier transform approach is not constrained to this requirement and can choose which region of the isotope distribution to be computed.[13−16] On the other hand, the iterative representation in BRAIN is flexible and allows for a variety of advanced stopping criteria, as implemented in the Bioconductor package.[10] For instance, the cumulative probability, i.e., coverage, can be calculated during the recursion and evaluated whether it is close to a predefined threshold. For small molecules the calculation can be timely halted after the first three or four peaks resulting in a lower number of peaks that have to be computed. Figure 1 illustrates

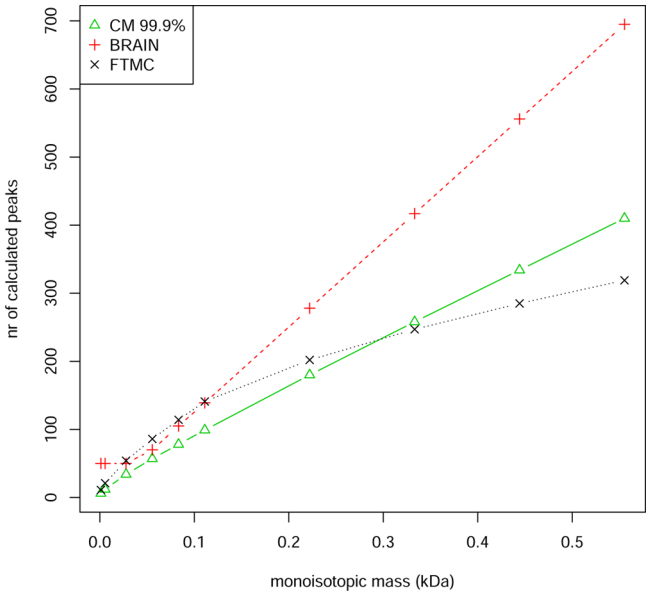

Figure 1. Number of peaks that are required according to the different heuristics: CM, coverage of 99.9% starting from the monoisotopic variant, cfr; BRAIN (triangle), BRAIN heuristics based on eq 2 (plus-sign); heuristic used by FTMC. All heuristics yield a coverage of minimum 99.9%.

the relation between the number of peaks (*y*-axis) required for the calculation of a molecule with a particular monoisotopic mass (*x*-axis) for several heuristics: default FTMC, BRAIN from eq 2, and the 99.9% coverage starting from the monoisotopic variant (CM). It is remarkable that BRAIN and to some extent FTMC overestimate the number of peaks required to obtain a sufficient coverage, leaving room for improvements in both heuristics.

(3) Software Platform and Implementation. Fernandez-de-Cossio Diaz and Fernandez-de-Cossio implemented their FTMC algorithm in C#,[1] while BRAIN was originally implemented in R,[10] which is an interpreted language and therefore less efficient. To facilitate a direct comparison, we have ported the BRAIN method to C++. It should be noted that the performance of C++ and C# for standard bioinformatics algorithms occurred to be close.[17] The source code and binaries of BRAIN in C++ are available at https://code.google.com/p/brain-isotopic-distribution/.

In order to evaluate the performance of the compiled programs, the database presented in the BRAIN application note[10] composed of 57 930 proteins is processed using BRAIN in C++ (version 0.9.6) and in R (version 1.6.4). The calculated number of peaks are set corresponding to the heuristic in eq 2. The output file is generated in 80 s. In the case of the C++ implementation, while in R, the calculations took 15.5 min. Both calculations were conducted on a Core i7 870 2.93 GHz (4 GB RAM) machine. Furthermore, the 10 molecules based on the averagine peptide model[18] as describe in the analysis of Fernandez-de-Cossio Diaz and Jorge Fernandez-de-Cossio[1] are used to compare the performance of FTMC with the BRAIN implementation in C++. Note that large averagine molecules can be helpful to observe the trend in performance; however, in practice, heavy molecules can be less precise due to the fluctuations of isotope abundances.[9,11]

In our comparison, we ran both software programs for each averagine molecule separately using the default heuristics, i.e., eq 1 and eq 6 in ref 1, and repeated this procedure 100 times enabling us to determine the minimum time that the system required to calculate the isotope distribution for a molecule. The minimum time from 100 calculations is chosen to minimize inevitable side effects such as garbage collection, interrupts, etc. It should be noted that this time also includes the standard operations required to start the algorithmic procedure, e.g., reading the input file, writing the result file, loading user-defined parameters, etc. These standard operations will not change the theoretical asymptotic complexity but can obscure the timing results when not appropriately accounted for, as seen in Figure 2. Indeed, the time complexity plots in Figure 1 presented by Fernandez-de-Cossio Diaz and Fernandez-de-Cossio[1] are calculated as "the average times required for 100 repetitions of these computation"; however, it is not specified how this running time was measured exactly in the case of FTMC. We assume that the authors have run the program with option "−r100" specifying the number of repetitions for the calculations. In this procedure, the time required for data initialization and system operation is only measured once for a batch of 100 repetitions. Hence, the standard operation time for starting the calculation is divided over the 100 runs. Note that this procedure is very different than the one presented in Figure 2, where the two programs were run 100 times separately for each molecule, so that system time is included in the timing. On the other hand, for the time measurement of the useBRAIN function from the R Bioconductor package, the authors measured the system time for 100 separate runs of the function (code at http://bioinformatica.cigb.edu.cu/isotopica/centermass.html). Of note, this analysis is not timing the same procedure as the standard system operations are executed 100 times separately, cfr. Figure 2.

It seems that software optimization is an important factor when analyzing the numerical efficiency of an algorithm,

dx.doi.org/10.1021/ac402731h | *Anal. Chem.* 2013, 85, 12189−12192

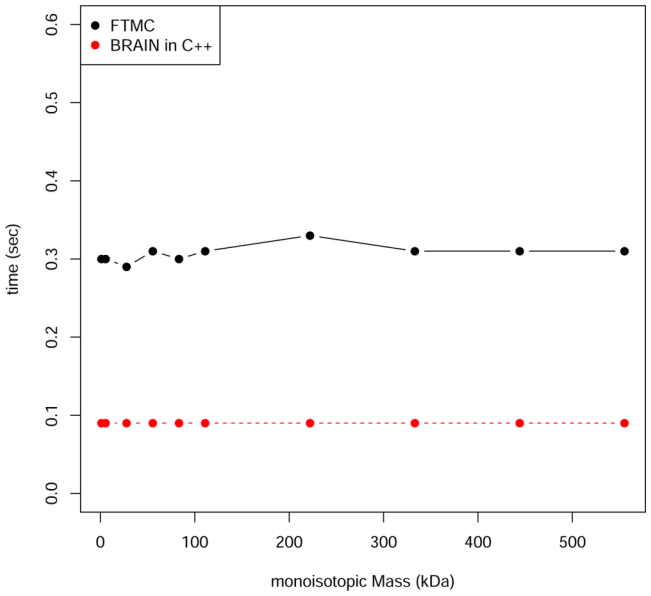

Figure 2. Average system time elapse for FTMC and BRAIN on each averagine molecules separately. For every molecule the software is called 100 times.

especially since both BRAIN and FTMC allow for batch processing. During batch processing tedious and recurrent calculations can be computed in advance and stored into the memory for later usage. To make the comparison in terms of the efficiency of batch processing, we run both software packages for each molecule in the averagine data set separately using the default heuristics and specify the molecule 100 times in the input file. For each molecule a result file is written as output. We argue that this comparison is more transparent than using the "−r100" option. The obtained system time elapse is divided by 100 and presented in Figure 3. Indeed, we can see that the standard operations have an impact on the timing

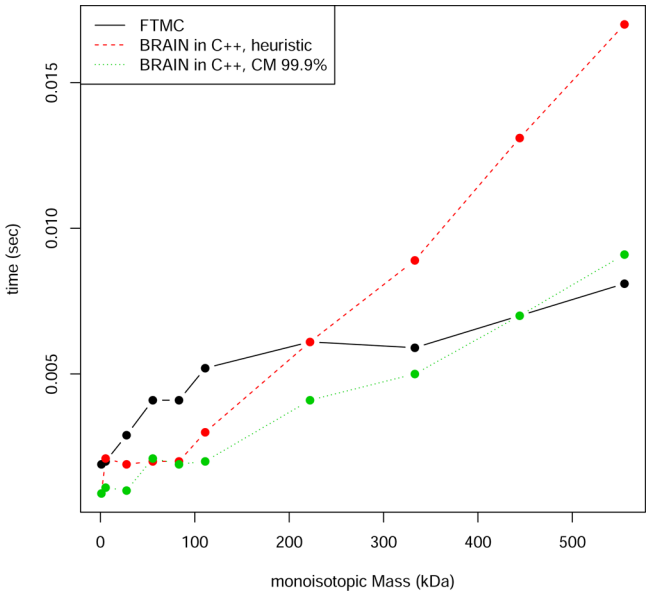

Figure 3. Average system time elapsed in function of the monoisotopic mass for FTMC and BRAIN in C++ run with an input file containing 100 times the same molecules for each averagine molecule (shown system times are divided by 100).

procedure. The figure displays now a quadratic trend for BRAIN in function of the molecular mass (red dots), as expected from the theoretical order complexity. This quadratic trend is also visible in Figure S1 in the Supporting Information that presents the timing in function of the number of peak per molecule. The explanation for this repeated trend is that the BRAIN heuristic has a linear relation between the number of peaks and the mass of a molecule, as can be observed in Figure 1. It should be noted that in Figure S1 in the Supporting Information, data points corresponding to the same number of peaks does not correspond to the same molecule. Figure 3 displays a timing trend for FTMC that is proportional to the square root of the mass as presented in Figure 1 by Fernandez-de-Cossio Diaz and Fernandez-de-Cossio.[1] However, a linear relation (in fact, Nlog(N)) becomes apparent when presenting the timing trend in terms of the number of computed peaks as seen from Figure S1 in the Supporting Information. Note that the presentation in terms of the number of computed peaks makes the timing invariant to the employed heuristic.

In addition, we run BRAIN in C++ with the number of peaks corresponding to a coverage of 99.9% (CM) to emphasize the importance of good heuristics. The green line in Figure 3 and Figure S1 in the Supporting Information shows the decrease in time when using fewer peaks. Clearly, this information could also be retrieved from interpolating the red line in Figure S1 based on the heuristics presented in Figure 1. Further, it should be stressed that the time measurements in this study are subjected to choices in hardware configuration and input/output operations. In our results we encounter fluctuation which we could not explain. For example, a strict monotonic trend should be expected in Figure 3. The code used for the comparison is available online http://www.mimuw.edu.pl/~pd219416/AnChemComment/.

To conclude, both BRAIN and FTMC are methods that are based on the same theoretical principles. Differences in the algorithm become very apparent in terms of computational efficiency. However, it is not always trivial to quantify which improvements are responsible for the observed increase in efficiency. The user has to be aware that variations in both numerical and technical aspects between the algorithms can lead to discrepancies in the results. A case in point are the several heuristics that can be employed to determine the prominent part of the isotope distribution. The argument we want to convey is that different heuristics and implementations have an impact on the numerical performance and therefore have to be controlled in a comparison. In this comment, we have evaluated the impact of a solution (FFT vs iterative scheme), heuristics (centering vs starting from monoisotopic variant), and implementation (R vs C++) separately.

Finally, compared to the BRAIN implementation in R, we regard the implementation in C++ particularly useful, not only because of the favorable time properties but also because of the ease of integration into desired workflows and batch processing. For smaller molecules, it appears that both FTMC and BRAIN in C++ are fast. For very large molecules above 100 kDa, FTMC is a faster method; however, isotope calculation can become imprecise in general due to fluctuations in the elemental isotope distribution.[9,11] For completeness, we should mention that methods such as Emass,[7] NeutronCluster,[19] SIRIUS,[20] etc. are also capable to perform these isotope calculations. More insight on methods that calculate the isotope distribution are provided by Valkenborg et al.,[21] Rockwood and Palmblad,[22] and Scheubert et al.[23]

dx.doi.org/10.1021/ac402731h | *Anal. Chem.* 2013, 85, 12189−12192

■ ASSOCIATED CONTENT

Supporting Information

Additional information as noted in text. This material is available free of charge via the Internet at http://pubs.acs.org.

■ AUTHOR INFORMATION

Corresponding Author

*E-mail: dirk.valkenborg@vito.be.

Author Contributions

$H.H. and P.D. contributed equally to this work.

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

This research is supported in part by the Polish National Science Center Grant 2011/01/B/NZ2/00864 and by the EU through the European Social Fund, Contract Number UDAPOKL. 04.01.01-00-072/09-00. D.V. and P.D. gratefully acknowledge the support of the bilateral FWO-PAS Grant VS.005.13N/Innovative algorithms to detect protein modifications in mass spectrometry data. P.D. is supported by a START fellowship from the Foundation for Polish Science. D.V. acknowledges the support of the SBO Grant 'InSPECtor' (120025) of the Flemish Agency for Innovation by Science and Technology (IWT).

■ REFERENCES

(1) Fernandez-de-Cossio Diaz, J.; Fernandez-de-Cossio, J. *Anal. Chem.* **2012**, *84*, 7052−7056.
(2) Fernandez-de-Cossio, J. *Anal. Chem.* **2010**, *82*, 6726−6729.
(3) Fernandez-de-Cossio, J. *Anal. Chem.* **2010**, *82*, 1759−1765.
(4) Fernandez-de-Cossio, J.; Gonzalez, L. J.; Satomi, Y.; Betancourt, L.; Ramos, Y.; Huerta, V.; Amaro, A.; Besada, V.; Padron, G.; Minamino, N.; Takao, T. *Nucleic Acids Res.* **2004**, *32*, W674−W678.
(5) Brownawell, M. L.; San Filippo, J. *J. Chem. Educ.* **1982**, *59*, 663.
(6) Yergey, J. A. *Int. J. Mass Spectrom. Ion Phys.* **1983**, *52*, 337−349.
(7) Rockwood, A. L.; Haimi, P. *J. Am. Soc. Mass Spectrom.* **2006**, *17*, 415−419.
(8) Claesen, J.; Dittwald, P.; Burzykowski, T.; Valkenborg, D. *J. Am. Soc. Mass Spectrom.* **2012**, *23*, 753−763.
(9) Claesen, J.; Dittwald, P.; Burzykowski, T.; Valkenborg, D. *J. Am. Soc. Mass Spectrom.* **2012**, *23*, 1828−1829.
(10) Dittwald, P.; Claesen, J.; Burzykowski, T.; Valkenborg, D.; Gambin, A. *Anal. Chem.* **2013**, *85*, 1991−1994.
(11) Böcker, S. *J. Am. Soc. Mass Spectrom.* **2012**, *23*, 1826−1827.
(12) Roussis, S. G.; Proulx, R. *Anal. Chem.* **2003**, *75*, 1470−1482.
(13) Rockwood, A. L.; Van Orden, S. L. *Anal. Chem.* **1996**, *68*, 2027−2030.
(14) Rockwood, A. L.; Van Orden, S. L.; Smith, R. D. *Anal. Chem.* **1995**, *67*, 2699−2704.
(15) Rockwood, A. L.; Van Orden, S. L.; Smith, R. D. *Rapid Commun. Mass Spectrom.* **1996**, *10*, 54−59.
(16) Rockwood, A. L. *Rapid Commun. Mass Spectrom.* **1995**, *9*, 103−105.
(17) Fourment, M.; Gillings, M. R. *BMC Bioinf.* **2008**, *9*, 82.
(18) Senko, M. W.; Beu, S. C.; McLafferty, F. W. *J. Am. Soc. Mass Spectrom.* **1995**, *6*, 229−233.
(19) Olson, M. T.; Yergey, A. L. *J. Am. Soc. Mass Spectrom.* **2009**, *20*, 295−302.
(20) Böcker, S.; Letzel, M. C.; Lipták, Z.; Pervukhin, A. *Bioinformatics* **2009**, *25*, 218−224.
(21) Valkenborg, D.; Mertens, I.; Lemière, F.; Witters, E.; Burzykowski, T. *Mass Spectrom. Rev.* **2012**, *31*, 96−109.
(22) Rockwood, A. L.; Palmblad, M. In *Mass Spectrometry Data Analysis in Proteomics*, 2nd ed.; Springer: Secaucus, NJ, 2013; pp 65−99.
(23) Scheubert, K.; Hufsky, F.; Böcker, S. *J. Cheminf.* **2013**, *5*, 1−24.