

# In-silico Optimisation of Mass Spectrometry Fragmentation Strategies in Metabolomics - Supplementary Materials

Joe Wandy<sup>1,‡</sup>, Vinny Davies<sup>2,‡</sup>, Justin J.J. van der Hooft<sup>3</sup>,  
Stefan Weidt<sup>1</sup>, Rónán Daly<sup>1</sup> and Simon Rogers<sup>2</sup>

<sup>1</sup> Glasgow Polyomics, University of Glasgow, Glasgow, United Kingdom. <sup>2</sup> School of Computing Science, University of Glasgow, Glasgow, United Kingdom. <sup>3</sup> Bioinformatics Group, Department of Plant Sciences, Wageningen University, Wageningen, The Netherlands.

## 1 MS1 Simulations

Figure 1 shows the overall characteristics of  $m/z$ , RT and intensity values from running XCMS peak picking on the resulting simulated mzML file and 4 *multi-beer* samples for comparison (labelled *multi-beer-1*, ..., *multi-beer-4*). It can be observed from Figure 1 that all  $m/z$  and RT boxplots show similar profiles between the 4 randomly selected real beer samples and the simulated sample from ViMMS. We explain the slightly higher intensity distribution in Figure 1c due to the fact that in our sampling scheme, a chemical having high sampled maximum intensity values could be assigned a large ROI that spans a big RT range. During peak picking, these potentially-noisy ROIs are detected as multiple high-intensity peaks by XCMS, resulting in an upward shift in the intensity distribution of detected MS1 features from the simulated mzML file. Improving the regions of interest detection scheme to address this issue is a future work.

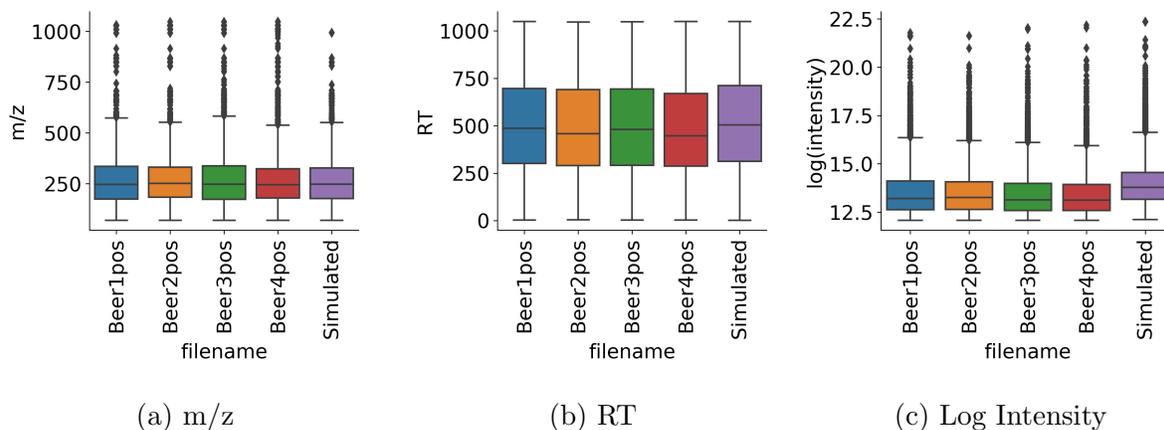


Figure 1: A figure showing boxplots of the simulated data and some of the *multi-beer* samples.

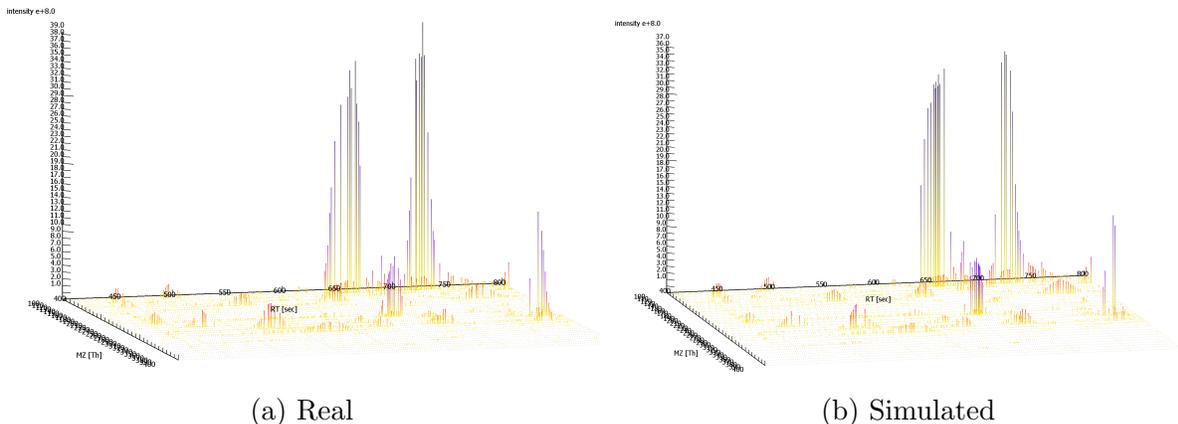


Figure 2: ToppView visualisation of m/z range (100 - 400) ppm and RT range (400 - 800)s for a Top-10 DDA fragmentation strategy in ToppView for (a) a *multi-beer* sample and (b) for the same file generated using ViMMS

## 2 Top-N Simulations

From the actual *multi-beer-1* fragmentation mzML, we extracted 512,540 ROIs across all scans using a 10 ppm tolerance that determines when peaks should be grouped under one ROI. Only ROIs with at least one peak above the minimum intensity threshold of  $1.75E5$  are kept. This results in 10,190 ROIs, which are converted into chemical objects in ViMMS, having their corresponding normalised chromatographic peak shapes derived from the ROIs. Simulated Top-10 DDA fragmentation was performed using the same fragmentation parameters used to generate the actual beer1 data (N=10, DEW=15s). Simulation took approximately 2 minutes in ViMMS running on an Intel Core i9 laptop. The generated mzML file and simulator state were loaded into ToppView and Jupyter Notebook for further analysis.

Visual inspection of the resulting spectra in ToppView shows that the mzML files from ViMMS and the *multi-beer-1* data look broadly similar (Figure 2). Comparing the number of scans, we obtained a total of 9,891 scans from the simulator to 9,406 scans in the actual beer mzML, resulting in 3,027 fewer MS1 features that can be picked by XCMS from the generated mzML file (Table 1).

To evaluate these differences quantitatively, we matched precursor ion information from the real *multi-beer 1* mzML to the generated mzML from ViMMS. 6,757 out of 7,655 (88%) precursor ions can be matched from the actual *multi-beer 1* file to the generated file when matching up to 2 decimal places for the m/z values and using 15s tolerance window for retention time. It can be seen from Figure 3 that fragmentation events from both the real and simulated files are in close proximity to each other.

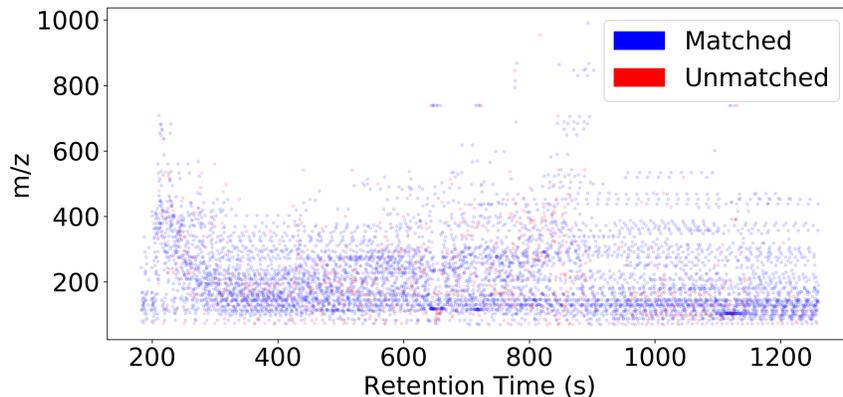


Figure 3: Locations of precursor peaks extracted from MS/MS scans in original *multi-beer-1* data that could be matched to the generated mzML data in ViMMS. The colours indicate where the MS/MS occurred in close proximity to an MS/MS in the real beer data (blue for matched fragmentation events and red for unmatched ones). Scans were performed in both cases using a Top-10 DDA fragmentation strategy.

Table 1: Table showing XCMS peak picking results on the real vs simulated files for the original beer data.

	Generated mzML	Real mzML
Number of MS1 scans	1,267	1,751
Number of MS2 scans	8,624	7,655
Number of peaks picked by XCMS	12,801	15,828

### 3 Alternative scenario: varying $N_s$ when both full-scan and Top- $N$ data are available.

Consider a scenario where within an experimental batch, full-scan and Top- $N$  data are acquired. In a typical analysis of this scenario, the MS1 peaks detected by XCMS from the full-scan MS1 file serve as the ground truth of peaks we wish to fragment. To compute performance in this scenario, XCMS' CentWave peak detection is performed on the full-scan input files mzML files, resulting in the number of ground truth MS1 features shown in Table 2.

Table 2: Table showing the number of ground truth MS1 features from XCMS peak picking for the two full-scan (MS1-only) *multi-beer* and *multi-urine* data in the Top-N experiments.

	Number of ground truth MS1 features
<i>multi-beer-1</i>	9,198
<i>multi-beer-2</i>	9,805
<i>multi-urine-2</i>	7,188
<i>multi-urine-3</i>	7,664

For evaluation, we provide the following definition of positive and negative instances (illustrated in Figure 4):

**True Positives (TP):** peaks from ground truth (found in full-scan files) that are fragmented above the minimum intensity threshold.

**False Positives (FP):** peaks from ground truth that are not fragmented + peaks from ground truth that are fragmented below the minimum intensity threshold.

**False Negatives (FN):** peaks not from ground truth that are fragmented above the minimum intensity threshold.

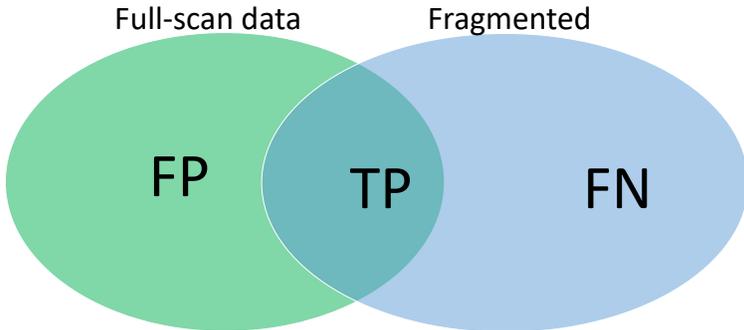
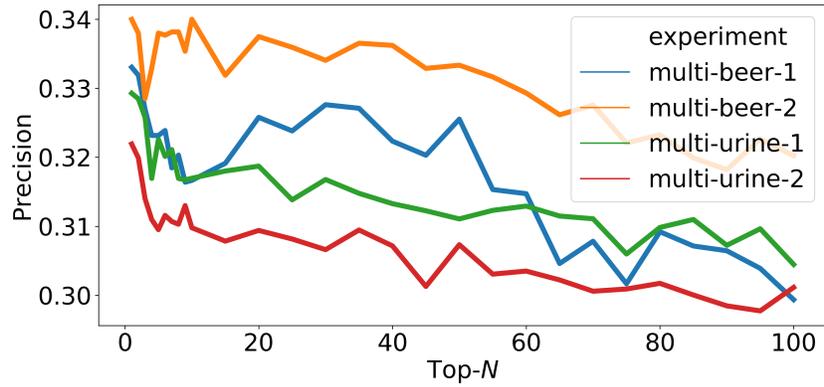
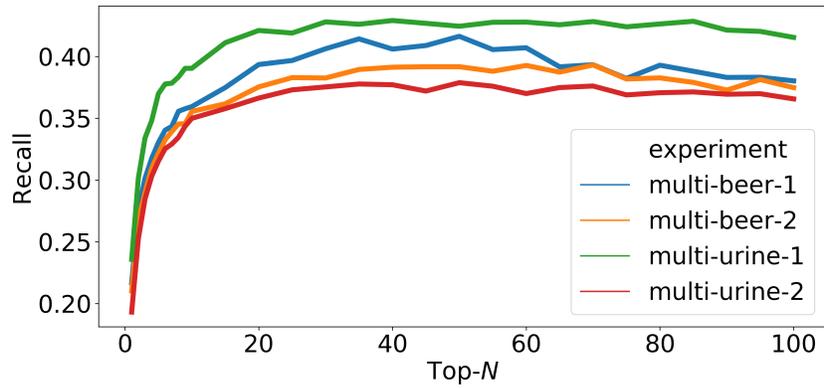


Figure 4: Definition of True Positives (TP), False Positives (FP) and False Negatives (FN) for performance evaluation of Top-N DDA fragmentation strategy (Scenario 1). The blue circle in the Venn diagram refers to all peaks that are fragmented above the minimum MS1 intensity threshold, while the green circle refers to all ground truth peaks (MS1 features) found by XCMS’ CentWave from the full-scan files.

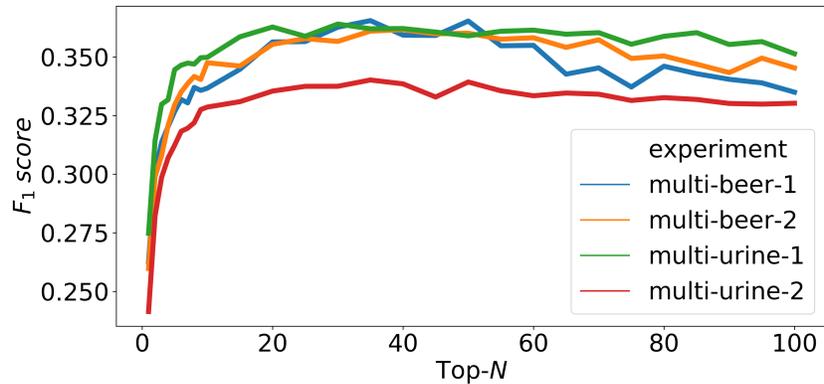
The results in this experiment shows that increasing  $N$  results in lower precision (Figure 5a), while recall increases with  $N$  initially but flattens (Figure 5b). Assessing the  $F_1$  score (Figure 5c), which is the harmonic average of precision and recall, we see that the best fragmentation performance (as represented by the  $F_1$  score) is reached between  $N$  from 10 to 20 and plateaus after 20, suggesting that no further performance gain is obtained from increasing the number of precursor ions to fragment.



(a) Precision



(b) Recall



(c)  $F_1$

Figure 5: Figures showing (a) precision, (b) recall and (d)  $F_1$  score for peak picking performance as  $N$  changes in Top- $N$  DDA experiments in ViMMS based on the classification specifications given in Figure 4.

## 4 Varying Multiple Parameters in the Top- $N$ Simulations

Figure 6 shows boxplots comparing the  $F_1$ -scores (representative of fragmentation performance) of all parameter combinations from the real and simulated data. The results from our simulator generally match the results from the real data, although with a slightly greater spread in the simulated fragmentation performance. Exploring the results in detail, we see that the best fragmentation performance is obtained at  $N = 20$  and  $DEW = 30$  for both datasets and the worst at  $N = 1$  and  $DEW = (15, 30, 60)$  (Table 3). We explain our findings by the fact that the best performance is obtained at the parameter combinations with the lowest trade-off between fragmentation performance and MS2 peak picking quality. The poor results came about from when both  $N$  and  $DEW$  are too small. In the former, not enough unique MS1 peaks are fragmented due to dynamic exclusion effect, whereas in the latter case, the quality of peak picking from fragmentation files decrease significantly, affecting the number of true positives obtained.

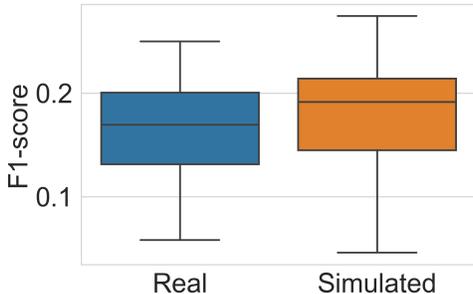


Figure 6: Distributions of F1-scores for all parameter combinations for the real and simulated *BeerQCB* data.

Table 3: The highest and lowest fragmentation performance for real and simulated results.

Rank	Real					Simulated				
	N	DEW	Prec	Rec	F1	N	DEW	Prec	Rec	F1
<b>1</b>	20	30	0.820	0.147	0.250	20	30	0.799	0.166	0.274
<b>2</b>	10	30	0.722	0.135	0.228	20	60	0.761	0.158	0.262
<b>3</b>	15	30	0.769	0.132	0.226	15	30	0.739	0.157	0.258
<b>-1</b>	1	15	0.771	0.030	0.058	1	15	0.793	0.024	0.046
<b>-2</b>	1	30	0.757	0.039	0.074	1	30	0.743	0.033	0.062
<b>-3</b>	1	60	0.738	0.048	0.091	1	60	0.741	0.046	0.087

## 5 Implementation Details

ViMMS is implemented in Python (Van Rossum et al., 2007), with numerical and matrix computations performed on top of the NumPY (Van Der Walt et al., 2011), SciPy (Jones et al., 2001) and Scikit-learn (Pedregosa et al., 2011) libraries. Prototyping as well as running interactive examples of ViMMS functionalities are performed in Jupyter Notebook environment (Kluyver et al., 2016). For DsDA, we re-use the original script from DsDA (Broeckling et al., 2018) available in the R programming language (Ihaka and Gentleman, 1996) to perform the prioritisation and scoring process. mzML files are read using the pymzML library (Kösters et al., 2018) and written using the psims library (Klein and Zaia, 2019).

## References

- Broeckling, C. D., E. Hoyes, K. Richardson, J. M. Brown, and J. E. Prenni (2018). Comprehensive tandem-mass-spectrometry coverage of complex samples enabled by data-set-dependent acquisition. *Analytical Chemistry* 90(13), 8020–8027.
- Ihaka, R. and R. Gentleman (1996). R: a language for data analysis and graphics. *Journal of computational and graphical statistics* 5(3), 299–314.
- Jones, E., T. Oliphant, P. Peterson, et al. (2001). Scipy: Open source scientific tools for python.
- Klein, J. and J. Zaia (2019). psims-a declarative writer for mzml and mzidentml for python. *Molecular & Cellular Proteomics* 18(3), 571–575.
- Kluyver, T., B. Ragan-Kelley, F. Pérez, B. E. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. B. Hamrick, J. Grout, S. Corlay, et al. (2016). Jupyter notebooks-a publishing format for reproducible computational workflows. In *ELPUB*, pp. 87–90.
- Kösters, M., J. Leufken, S. Schulze, K. Sugimoto, J. Klein, R. Zahedi, M. Hippler, S. Leidel, and C. Fufezan (2018). pymzml v2. 0: introducing a highly compressed and seekable gzip format. *Bioinformatics* 34(14), 2513–2514.
- Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, et al. (2011). Scikit-learn: Machine learning in python. *Journal of machine learning research* 12, 2825–2830.
- Van Der Walt, S., S. C. Colbert, and G. Varoquaux (2011). The numpy array: a structure for efficient numerical computation. *Computing in Science & Engineering* 13(2), 22.
- Van Rossum, G. et al. (2007). Python programming language. In *USENIX annual technical conference*, Volume 41, pp. 36.