SoftwareX 12 (2020) 100559

Contents lists available at ScienceDirect

SoftwareX

journal homepage: www.elsevier.com/locate/softx

Original software publication

HiTIME: An efficient model-selection approach for the detection of unknown drug metabolites in LC-MS data

Michael G. Leeming [a], Andrew P. Isaac [b,c], Luke Zappia [d,e], Richard A.J. O'Hair [a], William A. Donald [f], Bernard J. Pope [b,g,h,*]

[a] School of Chemistry and Bio21 Molecular Science & Biotechnology Institute, The University of Melbourne, Australia
[b] Melbourne Bioinformatics, The University of Melbourne, Australia
[c] The Walter and Eliza Hall Institute of Medical Research, Australia
[d] School of Biosciences, The University of Melbourne, Australia
[e] Murdoch Children's Research Institute, Australia
[f] School of Chemistry, University of New South Wales, Australia
[g] Department of Clinical Pathology, The University of Melbourne, Australia
[h] Department of Medicine, Central Clinical School, Monash University, Australia

ARTICLE INFO

Article history:
Received 5 December 2019
Accepted 29 June 2020

Keywords:
Liquid chromatography-mass spectrometry
Twin ions
Metabolites

ABSTRACT

The identification of metabolites plays an important role in understanding drug efficacy and safety however these compounds are often difficult to identify in complex mixtures. One approach to identify drug metabolites involves utilising differentially isotopically labelled drug compounds to create unique isotopic signals that can be detected by liquid chromatography-mass spectrometry (LC-MS). User-friendly, efficient, computational tools that allow selective detection of these signals are lacking. We have developed an efficient open-source software tool called HiTIME (High-Resolution Twin-Ion Metabolite Extraction) which filters twin-ion signals in LC-MS data. The intensity of each data point in the input is replaced by a Z-score describing how well the point matches an idealised twin-ion signal versus alternative ion signatures. Here we provide a detailed description of the algorithm and demonstrate its performance on simulated and experimental data.

Code metadata

Current code version	*1.1*
Permanent link to code/repository used of this code version	https://github.com/ElsevierSoftwareX/SOFTX_2019_370
Code Ocean compute capsule	
Legal Code License	*3-Clause BSD*
Code versioning system used	*git*
Software code languages, tools, and services used	*C++*
Compilation requirements, operating environments & dependencies	*C++ libraries: Boost, OpenMS, lru_cache. Compilation: CMake, Docker (optional)*
If available Link to developer documentation/manual	https://github.com/bjpop/HiTIME-CPP/blob/master/README.md
Support email for questions	bjpope@unimelb.edu.au

1. Motivation and significance

Liquid chromatography mass spectrometry (LC-MS) is often used to detect drug metabolites in biological cells, tissues and fluids. For metabolites that are unknown and unexpected, it can be challenging to determine which of the thousands of detected ions correspond to metabolites of administered compounds. This "needle-in-a-haystack" problem requires the development of methods to rapidly identify metabolites in complex mixtures without prior knowledge of their identity [1].

One approach to the non-targeted detection of drug metabolites is to simultaneously administer a mixture of a drug and an isotopically labelled variant in a controlled ratio [2–5]. Provided that the isotopic label is retained, when analysed by LC-MS, metabolites formed will elute simultaneously with the same

* Corresponding author at: Melbourne Bioinformatics, The University of Melbourne, Victoria, Australia, 3010, Australia.
E-mail address: bjpope@unimelb.edu.au (B.J. Pope).

https://doi.org/10.1016/j.softx.2020.100559

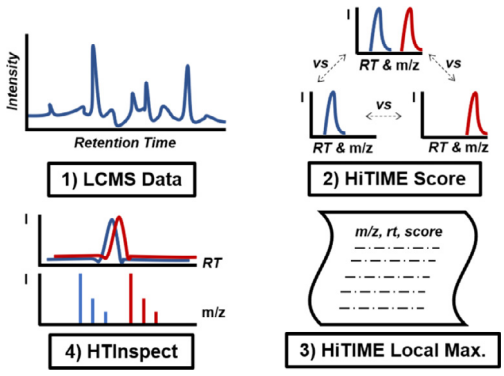

Fig. 1. The HiTIME process for detecting twin-ions in LC-MS data.

isotopic ratio as the precursor, however, the mass-to-charge ratio (m/z) values differ by the mass of the isotopic label. These "twin-ions" provide a signature unique to drug metabolites that distinguishes them from endogenous compounds.

We have developed a software system called HiTIME that searches LC-MS data for twin-ion signatures. HiTIME has been applied successfully to the detection of metabolites of the drugs paracetamol (APAP) and valproic acid in blood plasma of drug-treated rats [3,6], endogenous proteins covalently bound to electrophilic APAP metabolites [7], and nucleobase adducts of APAP [8]. This paper presents the first complete description of HiTIME system, including a detailed exposition of its underlying algorithm and a demonstration of its performance on synthetic and real data sets.

2. Software description

The HiTIME process for identifying twin-ions involves 4 steps (Fig. 1): (1) twin-ion LC-MS data is produced by an experimental assay yielding raw data in mzML format (in profile or centroid mode); (2) the data are scored based on model selection; (3) scores are filtered to local maxima; and (4) candidate results are visualised and reviewed. We have developed two new software tools to facilitate this process: HiTIME for scoring and filtering the input LC-MS data and HTInspect for visualisation and review.

The core HiTIME algorithm employs a novel model-selection approach, whereby sequential "neighbourhoods" of LC-MS data are compared to a pair of two-dimensional (Gaussian) elution peaks and contrasted against single ion ("heavy" or "light") features. For a data point at a given retention time (rt) and mass-to-charge ratio (m/z), the first model peak is centred at that point and the second at (rt, $m/z + \Delta m/z$). The model peaks are parameterised by the full-width at half-maximum (FWHM) in the rt dimension and the FWHM in the m/z dimension. The goodness-of-fit of this model to the surrounding data is determined by computing the correlation to the target signal and is scored in comparison to alternative non-target signals (single peaks). The overall score assigned to a given data point is calculated using a modified Z-test [9]. Two scores are calculated, one for the twin-ion model compared to the "light" isotope only model, and one for the twin-ion model compared to the heavy isotope only model. The minimum Z-score is used as the final score. A score greater than zero indicates that the data in the surrounding region has a higher correlation to the twin-ion model while a score less than zero indicates a better match to a single peak (either heavy or light). Scored data is written to an mzML file which can be further processed in open source software. By comparison, previous methods to detect twin-ions employ a two-step process of agnostic feature finding and subsequent filtering of the results [10–13]. HiTIME filters the scored data to find local maxima, outputting a list of candidate twin-ions as a CSV file containing the rt, m/z and score data at the local maxima. User visualisation and review of candidate twin-ions is then conducted using our auxiliary graphical tool, HTInspect (https://github.com/mgleeming/HTInspect), which plots extracted ion chromatograms and mass spectra in a format that is familiar to researchers with experience in mass spectrometry. A final list of confirmed twin-ions can be exported.

The HiTIME algorithm is illustrated by pseudocode in Fig. 2, consisting of a score function for evaluating the data points in an individual spectrum, and a thread_worker function for parallelisation. The algorithm is parameterised by the experimentally defined ratio of natural-to-isotope abundance (R), the m/z differences (Δmz) that correspond to the mass difference introduced by isotopic enrichment and by the full-width at half-maximum in the retention time (rt) dimension ($FWHM\_rt$, in number of scans) and the full-width at half-maximum in the m/z dimension ($FWHM\_mz$, in parts per million).

The provided $FWHM\_rt$ and $FWHM\_mz$ are used to determine the bounds to the data regions and Gaussian models used to calculate the score at each point (rt, mz). The retention time bound is defined to be $rt +/- 2\sigma_{rt}$ and the mass-to-charge bound is defined to be $mz \cdot (1 +/- 2\sigma_{mz})$. Where σ_{rt} and σ_{mz} are derived by scaling $FWHM\_rt$ and $FWHM\_mz$ respectively.

The correlation between the data and the fitted model is calculated in two parts, one for the natural and one for the isotope regions of $rt +/- 2\sigma_{rt}$ by $mz \cdot (1 +/- 2\sigma_{mz})$ and $rt +/- 2\sigma_{rt}$ by $(mz + \Delta mz) \cdot (1 +/- 2\sigma_{mz})$ respectively. The correlation is calculated to give equal weight to each region, irrespective of differences in sample size between the regions. Three correlations are calculated, one for each model, and are used to produce a final Z-score for each data point.

Structurally, the HiTIME algorithm is a stencil convolution. Each output data point at (rt, mz) is computed from the local neighbourhoods of two data points in the input file at (rt, mz) and (rt, Δmz) respectively. Stencil convolution algorithms are data parallel and relatively easy to parallelise if one assumes that all input and output data are kept in memory throughout the entire computation, resulting in 2N memory usage for an input file of size N (one copy for the input, and one equal-sized copy for the output). One of our aims is to enable HiTIME to be useable on commodity hardware (e.g. laptops), therefore such a simple parallelisation approach is not feasible because high-resolution LC-MS files can easily be many gigabytes in size. Therefore, a significant contribution of this work is the development of a memory-efficient parallelisation strategy. To do this we adopt a "leapfrog" approach, such that each CPU thread operates on the next unsolved spectrum, in retention time order.

A global variable keeps track of the ID of the next unsolved input spectrum. Threads can only read from and update this variable via the get_next_spectrum_todo() method, which prevents race conditions by protecting the variable with a C++ mutex lock. When a thread needs to read from a spectrum it does so via its ID, described below.

This leapfrog approach allows us to spread the processing work evenly across the available CPU threads, however it presents two major challenges in terms of file input and output handling:

1. Threads will be working on overlapping neighbourhood windows of spectra (via the get_neighbourhood() function). We seek to avoid having to re-read the same spectrum from the input multiple times and we do not want to have multiple copies of the same spectrum in memory simultaneously.

```
global FWHM_rt, FWHM_mz, R

def score(spectrum_id):
        rt, spectrum = lookup(spectrum_id)
        output_spectrum = new(spectrum)
        for (mz, intensity) in mass_sorted(spectrum):
                low_data = get_neighbourhood(rt, mz)
                high_data = get_neighbourhood(rt, mz + Δmz)
                COR_twin = fit_gaussians(low_data, high_data, 1, R)
                COR_nat = fit_gaussians(low_data, high_data, 1, 0)
                COR_iso = fit_gaussians(low_data, high_data, 0, 1)
                score = min( MengZ(COR_twin, COR_nat), MengZ(COR_twin, COR_iso))
                output_spectrum.write(rt, mz, score)
        return output_spectrum

def thread_worker():
        next_spectrum_id = get_next_spectrum_todo()
        while (next_spectrum_id < num_spectra):
                output_spectrum = score(next_spectrum_id)
                put_spectrum(next_spectrum_id, output_spectrum)
                next_spectrum_id = get_next_spectrum_todo()
```

Fig. 2. Pseudo code of the HiTIME algorithm. Each data point in a single spectrum is scored based on its goodness of fit to the twin ion model by the score function. Each parallel thread executes the thread_worker function which iterates over the available spectra in the input file in retention time order.

2. We cannot guarantee that threads will complete scoring of spectra in retention time order. Therefore we may not necessarily be able to write an output spectrum to file immediately when it has been scored; some in-memory reordering may be necessary.

Our solution to the first challenge is to maintain a cache of spectra that are currently being processed by the program and share this between threads. The cache is indexed by spectrum IDs. We employ a least-recently-used (LRU) cache implementation (from https://github.com/lamerman/cpp-lru-cache under the 3-Clause BSD license). The LRU retention policy enables fast access to data elements while maintaining good temporal locality. When a thread needs to access a spectrum, it must make a request via the cache using the ID of the spectrum. If the spectrum is not already in the cache then it is fetched from the input file using the IndexedMzMLFileLoader method provided by OpenMS [14], that provides random access to indexed mzML files. The cache refers to individual spectra via thread-safe C++ shared pointers (shared_ptr) that are reference counted. This ensures that spectra are kept in memory while they are being processed by one or more threads and are automatically freed (only) when no longer needed. Importantly, this enables correct memory allocation of spectra even if they are evicted from the cache while still being processed by the program — a situation that can arise with small cache sizes and/or large neighbourhood windows. The size of the cache defaults to 50 spectra, however it can be overridden by a command-line parameter.

Our solution to the second challenge is to keep a priority queue of output spectra, ordered by retention time. Threads place output spectra into the queue via the put_spectrum() method. A queued spectrum is written to disk only when it is sequentially the next unwritten one in retention time order. The leapfrog parallelisation strategy generally ensures that the output queue does not grow very large. On typical LC-MS test data sets it is rarely longer than one element.

The combination of the leapfrog parallelisation strategy along with input caching and output queueing results in a *memory-efficient* algorithm that scales very well with the number of available CPU threads. Fig. 3 demonstrates linear speedup and memory usage of HiTIME versus number of threads on a 14 GB representative full-profile (not centroided) data set.

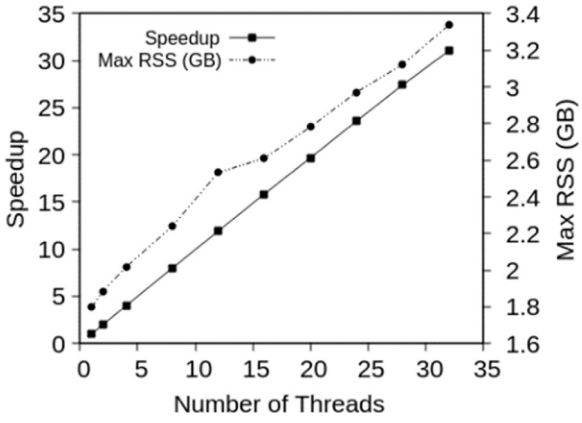

Fig. 3. Runtime speedup and memory usage (maximum resident set size (RSS)) versus number of threads for HiTIME data analysis. Input test data is a 14 GB mzML file tested on 1 to 32 CPU cores on 2.3 GHz Intel Xeon E5-2698 v3, with 32 cores per node.

3. Illustrative examples

Two different approaches are taken to demonstrate the HiTIME process (in addition to previous publications). Firstly, synthetic LC-MS data was produced that contain exactly 17 twin-ions of varying abundance in addition to a number of non-twin-ion signals. Running HiTIME on this synthetic data resulted in the successful identification of all 17 true signals and no additional signals. Secondly, we tested HiTIME against real LC-MS data using a pure protein reacted with the chemically synthesised metabolite of APAP, N-acetyl-p-quinoneimine, (NAPQI) and an equal portion of isotopically enriched $^{13}C_6$-NAPQI.

3.1. Synthetic LC-MS data

A small test case dataset generated *in silico* called synthetic17.mzML has been generated using MSSimulator [15] along with a custom Python script to generate a file containing an exactly known number of twin-ions. The synthetic17.mzML file can be obtained from:

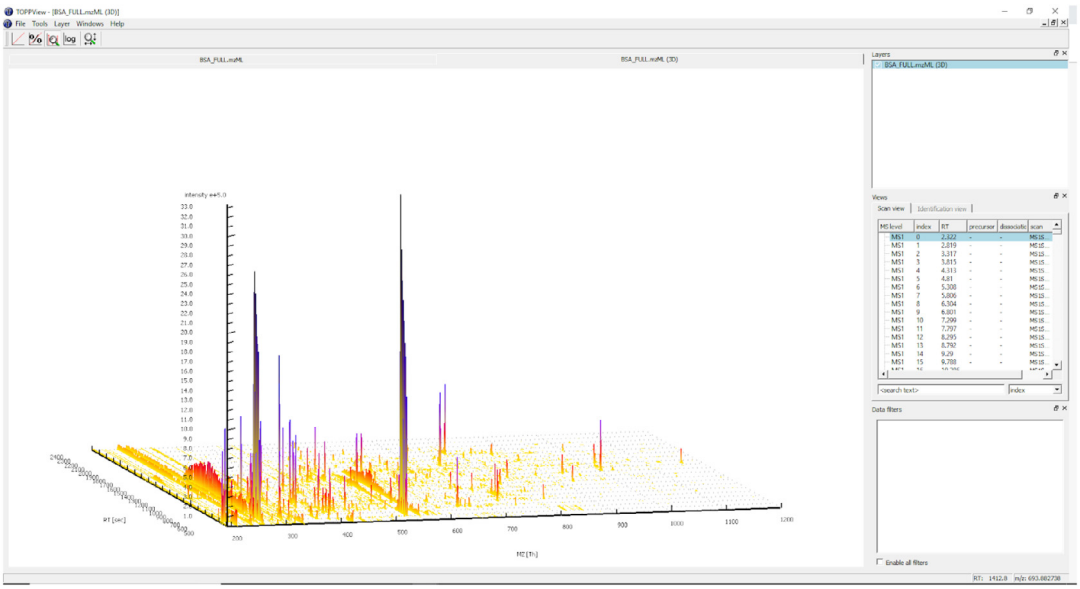

Fig. 4. Unprocessed LC-MS data visualised in TOPPView generated by analysis of the NAPQI/$^{13}C_6$-NAPQI-treated BSA digest.

https://melbourne.figshare.com/ndownloader/files/16247657

We provide a convenient wrapper script called hitime-docker.sh to run the HiTIME Docker container. Using the wrapper script, the program can be run on the test case like so, yielding a new reweighted mzML file results.mzML as output:

./hitime-docker.sh -i synthetic17.mzML -o results.mzML – -d 6.0201 -r 3 -m 30

In the above example the parameters -d 6.0201 -r 3 -m 30 specify a twin-ion *m/z* difference of 6.0201, retention time full width half maximum (FWHM) size of 3 steps, and m/z FWHM of 30 parts-per-million respectively. These parameters must be chosen to match the input dataset, and, depending on the experiment, may be known in advance by the user, or may be determined empirically by examining the spectra for key molecules, such as the precursor.

HiTIME can also be used to filter the output data to only include the points that have the largest value in a region defined by the retention time full width half maximum (FWHM) size, and the m/z FWHM bounds. For example, this can be applied to the re-scaled output from above, like so:

./hitime-docker.sh -i results.mzML -o max.results.mzML – -r 3 -l 0.3 -u 0.7

The -r 3 parameter retains its meaning from above, while the -l 0.3 -u 0.7 specify lower and upper m/z offsets for the local maxima window; these can also be determined empirically by examining the spectra for key molecules, such as the precursor. This command will produce two files, max.results.mzML and max.results.csv. The CSV file is a comma separated text file listing the local maxima. This list can be sorted to help identify the strongest twin-ion signal matches.

The graphical tool, HTInspect, is provided as a Python package which allows interactive review of the results. In the graphical user interface, select the CSV output (max.results.csv) generated above as "HiTIME Output File" field. This will display an *m/z* vs. retention time plot of the local maxima points. Select the original mzML file (synthetic17.mzML) in the "mzML File" field. The "mzDelta" field is the mass difference between heavy and light isotopes as above and "EIC Width" refers to the mass tolerance (in *m/z*) to use when extracting mass spectral data for EIC plots. An output file can be set as desired and processing is started by clicking "Run Postprocessing". This will produce an output file containing EIC traces and mass spectra for each of the local maxima detected. To interactively review these results, select the "Results Viewer" tab of the user interface, click "Load Results" and select the output file generated in the above step. EIC traces for both "heavy" and "light" isotopes can then be inspected interactively for each hit. The full-width at half-maximum in the rt dimension can be measured by counting the number of scans on a known compound signal, such as that of the precursor compound.

3.2. Real LC-MS data: bovine serum albumin

To test HiTIME on a sample containing many twin-ions, a pure protein (bovine serum albumin, BSA) was reacted with the chemically synthesised metabolite of APAP, N-acetyl-p-quinoneimine, (NAPQI) and an equal portion of isotopically enriched $^{13}C_6$-NAPQI. Experimental details for the preparation of this sample are provided in Appendix A. The protein was then digested and analysed by LC-MS. An image of the resulting LC-MS data is given in Fig. 4. The data is clearly dominated by a small number of high abundance ions making inspection of possible twin-ion signals, that are of much lower abundance, difficult.

To identify potential twin-ions, this data was processed with HiTIME in $<$ 1 min using a desktop computer. Upon visualisation using HTInspect, these are confirmed to correspond to true twin-ion signals (Fig. 5(a)). By comparison, identical treatment of a control BSA digest dataset that had not been exposed to $^{13}C_6$-NAPQI, no convincing twin-ion is found (Fig. 5(b)).

In the control data, by chance there are still ions that match the twin-ion profile. The Sensitivity of HiTIME allows the detection of these signals and visualisation allows the analyst to accept or reject the feature as a target twin-ion.

4. Impact

Assessing the metabolic profile of a xenobiotic can be critical to understanding the molecule's therapeutic and toxic effects on living organisms however due to the complexity of biological fluids such as blood (that contain thousands of different compounds), differentiating xenobiotic metabolites from endogenous species can be difficult. As a result, isotopically labelled xenobiotics can be employed to provide an isotopic signature in routine LC-MS analysis [5]. HiTIME allows researchers to detect

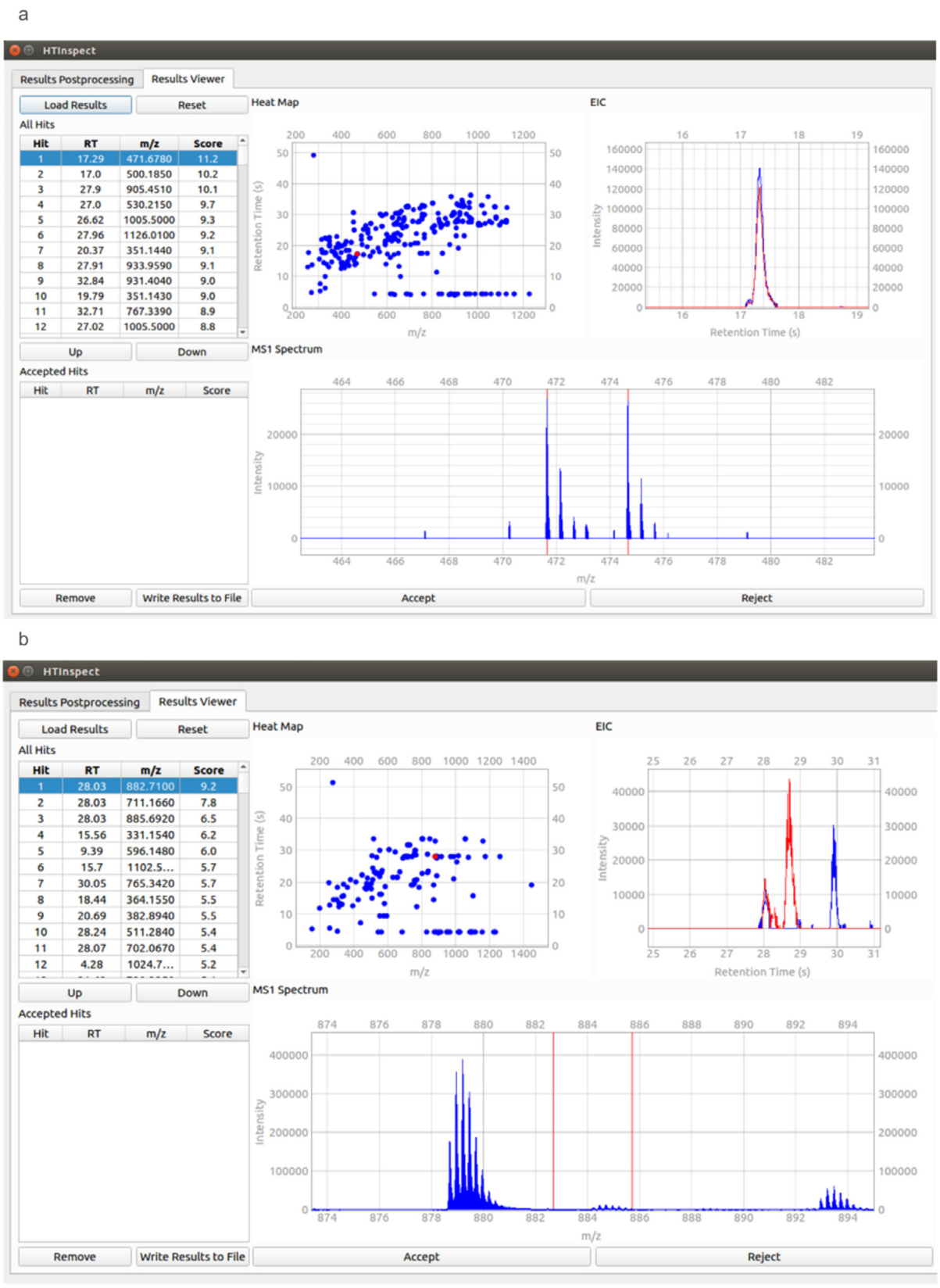

Fig. 5. Screenshot of HTInspect visualisation of (a) the NAPQI/$^{13}C_6$-NAPQI-treated BSA digest and (b) the control BSA digest.

and review twin-ion signatures formed in LC-MS data in twin-ion experiments. The use of the HiTIME approach can result in substantially more metabolites being detected in complex mixtures by LC-MS than by use of an alternative centroiding approach [3].

HiTIME has been successfully employed to investigate the in vivo metabolism of paracetamol (an analgesic) [3] and valproic acid (an anti-seizure medication) [6] leading to the successful detection of a structurally diverse range of metabolites including

a range of Phase 1 oxidation products as well as a number of polar Phase II conjugation products. In addition, HiTIME and twin-ion experiments has been employed to aid in the identification of the protein covalent modification targets of chemically reactive metabolites formed via *in vitro* [7] which are thought to be an important mediator of idiosyncratic adverse drug reactions [16]. In a different approach to investigating DNA drug targets, we have reacted the isotopically labelled nucleobase deoxycytidine with NAPQI to identify different types of adducts [8]. We are currently using HiTIME and LC-MS to identify the *in vivo* protein targets that can be covalently modified upon exposure to different classes of pesticides to investigate the mechanism of delayed and acute pesticide toxicity. We have also received some interest from a large pharmaceutical company to use HiTIME in their drug development programs.

5. Conclusions

HiTIME is a sensitive tool for the detection of twin-ion signals in LC-MS data that has been successfully demonstrated in previous work for the detection of paracetamol (APAP) metabolites in blood plasma of APAP-treated rats and endogenous proteins covalently bound to electrophilic APAP metabolites. HiTIME accepts inputs and produces outputs in standard mzML format, facilitating integration with other tools and workflows. Additionally, our HTInspect tool allows users to visualise, review and filter detected ions for improved specificity. A significant advantage of HiTIME is that it supports inputs in both profile and centroid modes, and its novel memory efficient and scalable parallelisation algorithm allows timely processing of large data sets on commodity computing hardware.

CRediT authorship contribution statement

Michael G. Leeming: Conceptualization, Methodology, Software, Validation, Writing - original draft. **Andrew P. Isaac:** Conceptualization, Methodology, Software, Writing - original draft, Formal analysis, Supervision. **Luke Zappia:** Software, Writing - review & editing. **Richard A.J. O'Hair:** Conceptualization, Methodology, Writing - review & editing, Supervision, Funding acquisition. **William A. Donald:** Conceptualization, Methodology, Writing - review & editing, Supervision. **Bernard J. Pope:** Conceptualization, Methodology, Software, Writing - original draft, Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

We gratefully thank Kin Kuan Hoi for evaluating early versions of HiTIME. We thank Prof. Michael Small for useful discussions on mathematical approaches to identifying twin-ion peaks. We thank the University of Melbourne Interdisciplinary Seed Grant program for funding. This research was further supported by the Victorian Life Sciences Computation Initiative (VLSCI). BP is supported by a Victorian Health and Medical Research Fellowship, Australia. MGL thanks the Elizabeth and Vernon Puzey foundation for the award of a PhD scholarship and The University of Melbourne for a Norma Hilda Schuster scholarship.

Appendix A

N-Acetyl-*p*-aminophenol (APAP, 28.9 mg), *ring*-$^{13}C_6$-APAP (30.0 mg) and freshly prepared Ag_2O (117.8 mg) were combined in dry dichloromethane (4.5 mL) and stirred under a nitrogen atmosphere for 1 h at ambient temperature with the exclusion of light. The reaction mixture was filtered, and the filtrate was applied immediately to a silica gel column and eluted with dry Et_2O. The product was collected as a bright yellow band. Dry acetonitrile (8 mL) was added to the N-acetyl-*p*-quinoneimine (NAPQI) fraction and the Et_2O was removed *in vacuo*.

The Ag_2O used in the above reaction was prepared as follows: $AgNO_3$ (2.25 g) was dissolved in H_2O (15 mL) then a solution of KOH (0.95 g) in H_2O (10 mL) was added with vigorous stirring. The brown precipitate was filtered and washed with H_2O (3 $\times$ 10 mL) followed by MeOH (3 $\times$ 10 mL). The product was dried under vacuum at 50 °C for *ca.* 3 h prior to use.

A stock solution of bovine serum albumin (BSA) was prepared to 2 mg mL^{-1} in 50 mM Triethylammonium bicarbonate (TEAB) buffer (pH 8.5), then 25 μL was diluted to 100 μL with 50 mM TEAB. Tris(2-carboxyethyl)phosphine hydrochlroide (2 μL, 500 mM) was added to each solution and incubated for 30 min at 60 °C with shaking then allowed to cool to room temperature. Either NAPQI/$^{13}C_6$NAPQI solution in acetonitrile (40 μL) or acetonitrile (40 μL) was added then the sample was incubated in the dark for 1 h at ambient temperature. Iodoacetamide (3 μL, 500 mM) was then added and the sample was incubated in the dark for 30 min. The solvent was removed by centrifugal evaporation and the residue resuspended in H_2O (100 μL) and stored at −20 °C until the time of analysis.

BSA digests were analysed using an Agilent 1100 liquid chromatography system coupled to an Agilent 6520 quadrupole time-of-flight mass spectrometer. Peptides were separated on an Agilent ZORBAX Exlipse XDB C_{18} reversed phase column (150 $\times$ 4.6 mm, 5 μm) using gradient elution at a solvent flow rate of 0.3 mL min^{-1} as follows [time (min), %B]: [0,0, [1,0], [40, 70], [42, 100], [48, 100], [0,49], 55,0]. Mass spectra were acquired over an *m/z* range of 100–2500 in profile mode and 15 μL of sample was injected for analysis. The Agilent '.d' data files were converted to mzML format using MSConvert with a lower intensity filter of 1000 counts.

References

[1] Bueschl C, Kluger B, Berthiller F, Lirk G, Winkler S, Krska R, et al. Bioinformatics 2012;28:736–8.
[2] Hoi KK, Daborn PJ, Battlay P, Robin C, Batterham P, O'Hair RAJ, et al. Anal Chem 2014;86:3525–32.
[3] Leeming MG, Isaac AP, Pope BJ, Cranswick N, Wright CE, Ziogas J, et al. Anal Chem 2015;87:4104–9.
[4] Mutlib AE. Chem Res Toxicol 2008;21:1672–89.
[5] Schellekens RCA, Stellaard F, Woerdenbag HJ, Frijlink HW, Kosterink JGW. Br J Clin Pharmacol 2011;72:879–97.
[6] Leeming MG, Cranswick N, Wright CE, Ziogas J, O'Brien TJ, Donald WA, et al. Int J Mass Spectrom 2019;444:116187.
[7] Leeming MG, Donald WA, O'Hair RAJ. Anal Chem 2017;89:5748–56.
[8] Li S, Leeming M, Chan B, O'Hair RAJ. Aust J Chem 2020;73:180–8.
[9] Meng X-L, Rosenthal R, Rubin DB. Psychol Bull 1992;111:172.
[10] Capellades J, Navarro M, Samino S, Garcia-Ramirez M, Hernandez C, Simo R, et al. Anal Chem 2016;88:621–8.
[11] Mahieu NG, Huang X, Chen Y-J, Patti GJ. Anal Chem 2014;86:9583–9.
[12] Reaser BC, Yang S, Fitz BD, Parsons BA, Lidstrom ME, Synovec RE. J Chromatogr A 2016;1432:111–21.
[13] Weindl D, Wegner A, Hiller K. Methods Enzymol 2015;561:277–302.
[14] Pfeuffer J, Sachsenberg T, Alka O, Walzer M, Fillbrunn A, Nilse L, et al. J Biotechnol 2017;261:142–8.
[15] Bielow C, Aiche S, Andreotti S, Reinert K. J Proteome Res 2011;10:2922–9.
[16] Nakayama S, Atsumi R, Takakusa H, Kobayashi Y, Kurihara A, Nagai Y, et al. Drug Metab Dispos 2009;37:1970–7.