Data parsing in mass spectrometry imaging using R Studio and Cardinal: A tutorial

Cameron J. Shedlock a, Katherine A. Stumpo a,b,c,*

a Department of Chemistry, University of Scranton, Scranton, PA 18510, United States
b Russell H. Morgan Department of Radiology and Radiological Science, The Johns Hopkins University School of Medicine, Baltimore, MD 21205, United States
c Bruker Scientific, Billerica, MA 01821, United States

ARTICLE INFO

Keywords:
Mass spectrometry imaging
Data validation
R Studio
Cardinal

ABSTRACT

Mass spectrometry imaging (MSI) has emerged as a rapidly expanding field in the MS community. The analysis of large spectral data is further complicated by the added spatial dimension of MSI. A plethora of resources exist for expert users to begin parsing MSI data in R, but there is a critical lack of guidance for absolute beginners. This tutorial is designed to serve as a one-stop guide to start using R with MSI data and describe the possibilities that data science can bring to MSI analysis.

Introduction

Mass spectrometry imaging (MSI) is a versatile tool for the analysis and identification of molecules of interest within biological tissues [1]. MSI [2] is an application of mass spectrometry [3,4], which allows scientists to analyze non-volatile biomolecules by ionizing them into the gas phase. This label-free technique can determine the spatial distribution of hundreds of compounds in a highly heterogeneous sample in one experiment. In MSI, frozen tissue sections are spray-coated with a matrix solution and data are then collected at discrete locations on the sample surface [5,6].

In the past ten years, the use of MSI has expanded from basic research [2,7] to biomedical applications, particularly proteomics and lipidomics [8] due to the ease of ionization of these classes of compounds [9]. MSI has the potential to transform translational science and general medicine and to improve health outcomes in a number of ways. For example, MSI has enhanced pharmaceutical drug discovery and development by facilitating high-resolution evaluation of drug disposition, metabolism, and toxicity [10–13].

MSI can be employed to discover and track the presence of predictive tissue biomarkers and aid in disease diagnosis, patient risk stratification, and disease management [14]. Because MSI can be carried out within hours to days, it can provide data that can be co-registered with concurrent histopathology [15,16] and, thus, has real potential for future routine integration into clinical workflows. As an example of how MSI has been used in patient diagnostics, desorption electrospray ionization (DESI) MSI and, more recently, MALDI-MSI have been integrated into the preoperative decision-making phase of tumor resection [17–21]. Intraoperative analysis of tissues has been interfaced with real-time surgical workflows and resulted in identifying peaks of interest that can be informative for pathological determinations. MALDI-MSI in the surgical suite has been demonstrated to deliver near-real-time peptide and protein information within 30 min of tissue resection [20].

Exciting recent studies that acquire metadata and utilize machine learning are paving the way for novel and unexpected correlations with disease states and ways to benefit patient outcomes [22,23]. Integration of MSI-based data with proteomic, transcriptomic, and genomic information [22] suggests that MSI could have far-reaching impacts and could become a clinical assay in and of itself. MSI is a highly efficient technique for high-throughput collection of large quantities of data, making it an optimal tool for creating accessible data sets within the scientific community.

Even as MSI usage has soared in the past two decades, there are several aspects that require further development in order to fully implement MSI in routine clinical research and practice. For example, while MSI works well for visualizing peptides, proteins, and lipids, its
application for other categories of small molecules (typically defined as mass less than 500 Daltons) is underdeveloped in comparison. This limits the use of MSI in metabolomics, which is increasingly of interest to researchers to understand the distribution and metabolism of natural and synthetic small molecules in tissue [24], as well as in applying metabolic signature profiling to precision medicine practices [25,26]. As biochemical reactions and metabolites vary among individuals, environmental conditions, and disease states, the ability of a high-throughput technique such as MSI to obtain metabolomics data will be extremely important in establishing precision medicine [27,28], as well as population trends and predictive patterns. Furthermore, consistent protocols must be established for reproducibility across laboratories and data analysis within a clinical timescale; computational and statistical strategies are needed for unbiased assessment of data quality. Across all of these challenges, it is also necessary to ensure that MSI can be applied by non-experts in order to maximize its utility [29].

For MSI to enter routine biomedical and clinical practice, it must be “user-friendly” for non-experts. Instrument operation has evolved to be driven by workflows and case-of-use software integrations, making it more hands off; scientists who outsource MSI to fee-for-service labs often receive data they cannot adequately interpret. Most data analysis methodologies are geared towards expert users with coding knowledge in order to extract the most information, and many research teams include computational biologists and/or biostatisticians for this purpose. In addition, current computational strategies for MSI often require the purchase of expensive proprietary software that requires extensive training to use effectively. While there is open-source software, such as the statistical programming language R, that can provide the workflows and code for anyone to use, open-source code often lacks the instructions and details required for non-experts to process and interpret the data.

One of the current challenges for MSI in pre-/clinical and pharmacological applications is a consistent application of experimental replicates and a subsequent routine data validation. Further compounding the issue is that there is no broadly accepted way to assess data quality in MSI, which impacts non-expert users the most, as they do not have enough MSI experience to judge data informally. This tutorial is a simple proposed data analysis workflow in R that will allow new pre-/clinical users to determine the requirements of data analysis in MSI and will provide concrete explanations behind the code. The base requirements for a new user are to have the current version of R and the R Studio integrated development environment installed on their workstation. Additionally, the package, Cardinal, and its dependencies will need to be downloaded (see Section 3.1 for information on how to get started). Cardinal requires all data to be converted to the vendor neutral imzML format as it is not designed to read the native file format from instruments. Nearly all vendor instrument software conversion to imzML features to export imaging runs in this format. This is intended not to be a tutorial on how to use Cardinal, as those already exist [30,31], but rather to introduce how to use the tools for data analysis needs in pre-/clinical experimentation.

The tutorial workflow has two sample scenarios in order to meet the demands of data quality in pre-/clinical analysis for which we will show data and the subsequent outcomes: (i) rapid comparison between tissue sections in order to account for the question of biological variance (i.e., cellular heterogeneity), and (ii) rapid assessment of sample preparation parameters/environmental factors to confirm data quality. By using the same R Notebook with the same commands to perform basic data validation, a complex MSI data, our goal is to show that anyone can use R and the sample dataset to assess these outcomes. Fig. 1, shown below, describes the analyses to be performed, the sample requirements, the validity tests, and the processed data outcomes.

**Methods for sample prep in MSI**

Zebradish embryos were placed in a 10 mm × 10 mm × 5 mm biopsy cryomold (Ted Pella, Redding, CA) and embedded in Thermo Scientific Zebreish husbandry

The Institutional Animal Care and Use Committee (IACUC) at the University of Scranton approved protocols #9–19 and #1–20, which includes all animal handling, breeding, and euthanasia methods. Adult zebrafish (*Danio rerio*) were purchased from Carolina Biological Supply (Burlington, NC) and bred, and embryos were collected within 1 h of fertilization. Embryos were transferred to Petri dishes containing embryo medium (E3 buffer) and kept in an incubator at 28.5 °C. E3 buffer was changed daily until 5 days postfertilization when embryos were sacrificed using either a 600 mg/L solution of tricaine methanesulfonate or immersion in liquid nitrogen.

**Instrumentation and data handling**

All imaging experiments were performed on Bruker Rapiflex MALDI TOF/TOF (Billerica, MA) in reflection positive mode at a lateral spatial resolution of 20 μm. Data collection occurred at the Applied Imaging Mass Spectrometry Core Facility at Johns Hopkins University School of Medicine with full software capabilities (i.e., FlexImaging, SGLIS lab, and other Bruker software). However, data analysis was done off-site on a workstation at the University of Scranton (see workstation requirements in section 3.1). All data was exported from FlexImaging into an .imzML file for use in R, which is required by Cardinal for analysis.

**Experimental planning and rationale**

This tutorial and data set were originally used for teaching within the Stumpo research group at the University of Scranton, a primarily undergraduate institution, with the goal of introducing new undergraduate students to the broad utility of MSI and data science. Without easy access
to an instrument, the main dataset utilized for that purpose was redesigned into this tutorial. A common question in the zebrafish community is if sacrifice method has an impact on detectable biochemical processes, especially for embryos [32–34], hence the two different sacrifice methods. The foundational publication of the Stumpo group centered around using nanoparticles for enhanced ionization of small molecules [35], hence the multiple comparisons of matrices.

**Tutorial for R using R Studio**

This tutorial starts with the basic information needed to setup a workstation for data processing. The minimum recommended computing power is an Intel Core i7 processor (equivalent or better), Microsoft Windows 7 operating system or Linux operating system (Ubuntu recommended), 32 GB of RAM, a 1 TB SATA solid state drive, and a graphics card supporting OpenGL 3.2. The best way to improve on this set of minimum recommendations is to add more RAM. If more storage space is needed, a less expensive option is to have a larger non-SSD for long-term storage and data for immediate processing can be stored on the SATA SSD. Due to the inherent large file size of MSI datasets this minimum setup is suggested, but it is worth noting that a weaker system such as a laptop is capable of processing small imaging runs and that size of the dataset will dictate minimum performance requirements.

**Basic information to get setup for data processing**

A host of powerful R packages have been created to enable efficient and relevant analysis of MSI data in R Studio. This tutorial will guide users through a simple step-by-step workflow to allow MSI data to be analyzed in R Studio without requiring a strong background in R or MSI. A sample dataset has been provided with different sacrifice and spray conditions, enabling many routes for comparison and analysis. R packages have been optimized to load complex MSI data with a few simple commands that are contained within a notebook that can be used repeatedly. Depending on your familiarity with R, the following resources are designed to guide beginners through the user interface to start using this notebook (https://education.rstudio.com/learn/beginner/). You will need at least a basic understanding of how to "talk" to R, such as the introductory information shown in Fig. 3 (some small suggestions on how to work within your notebook are given in the figure as well).

With R and R Studio installed, the user interface can be quickly understood through freely available videos and guides, such as the resources provided by the R Studio team. Use your own dataset or download the example from the GitHub data section, which includes this dataset and the tutorial notebook. Metaspace (a free online collection of MSI datasets available to download) [36] is another resource for finding and using MSI datasets. As this tutorial progresses, you will see code displayed in a code chunk (shown below), which you can copy and paste into your own R notebook or run natively in the provided introductory notebook. MSI datasets are ready to be loaded into R once they are in the imzML format. Imaging runs are either stored as continuous or processed imzMLs, which Cardinal treats differently. In a continuous file the m/z values will be the same for every spectrum in an image, while in a

---

```r
# This is a chunk. All code must be run in these.
Name <- './file/path'
# Anything after a # sign is a note to self that R will ignore. Every chunk should have a description of what it does in this format

Think of everything outside the chunk like a word docx. R only uses code found in chunks and this space is for thinking or explaining errors.
```

---

Fig. 2. Optical image of imaged slide with regions of interest highlighted and sample preparation conditions defined.

Fig. 3. Sample R Studio notebook screenshot with some basic introductory information.
processed type each spectrum keeps a respective $m/z$ array. Cardinal documentation explains the features of both types clearly, but an understanding of which type of file is being used is required to start. The code that is utilized throughout is available as a downloadable R notebook on GitHub: https://github.com/Camber27/MSI-R-Tutorial.

Now the object data is your MSI dataset loaded into R for further analysis. Many packages and data parsing methods can be used through R to make discoveries from the data.

```r
# install packages
if (!requireNamespace("BiocManager", quietly = TRUE))
  install.packages("BiocManager")
BiocManager::install("Cardinal")

# load MSI package Cardinal
library(Cardinal)
```

With the Cardinal package prepared and a MSI dataset in imzML format downloaded locally, analysis can proceed.

**MSI data loading**

The defining feature of MSI is the spatial component of the data. To generate $m/z$ images, data first needs to be read into R and processed.

```r
# use your terminal or OS to find the file path of the .imzml file of the MSI data from the downloaded github files
data_path_example <- "the/path/to/your/.imzml/imaging/data"

# make sure the imzml and ibd file are in the same folder. Use the file path for the .imzml file. It ends with .imzml in the path. Download the dataset from metaspace https://www.metaspace2020.eu/datasets?subm=e89830bd-cell-11ea-96db-8faddc24c30c&organism=Danio%20rerio%20zebrafish%29 or use the preprocessed .rds file provided in the github repository and skip past creating it.
data_path <- "insert the path to the imzml downloaded from metaspace here"
```

Important note: depending on the OS of the workstation, different symbols are needed for file paths. In Windows the symbol \ is the default for outputting file paths, but R will read this as its own command. This is why a second \ is added manually. In UNIX based OS like Mac or Linux the native / file paths are acceptable.

Particular peak is desired, it is useful to make an $m/z$ image and display its spectrum before and after preprocessing to verify this step retains the target.

For more information, readers are directed to several recent reviews that give a good overview of the choices available and the mechanics of preprocessing.[37–40]
Target confirmation

By visually inspecting the before and after preprocessing dataset objects, a target m/z can be identified by comparing images and spectra. The imaging of before preprocessing will be a time intensive process as it is performing analysis on unprocessed data (Fig. 4).

Target <- 146

# verify target remains after preprocessing
# create image of target before preprocessing
before <- image(data, mz=target, plusminus=0.5)
# create image of target after preprocessing
after <- image(data_proc, mz=target, plusminus=0.5)

# plotting the spectrum from a random (here number 5) pixel of the processed data. Adjust x and y to contain an appropriate intensity for y and the m/z value in the x range
plot(data_proc, pixel=11000, xlim=c(140,150), ylim=c(0,30))
This m/z image shows the total amount of signal for all pixels. By changing the parameters in the code, specific m/z values can be used along with changing the color theme. Cardinal offers dark and light themes along with cividis, magma, inferno, and plasma for colorscales. An m/z image of the whole dataset is useful for quickly visualizing signal hotspots and trends without further analysis. Furthermore, the image generated can be used to estimate coordinates for particular areas of the slide.

### Selecting desired regions of interest (ROI)

This code enables the user to choose coordinates from the previous m/z image of the full imaging run. By using estimated coordinates a select region can be turned into an R object. This is useful for targeted analysis of specific sections and having a smaller object for later processing.

```r
# cut an imaging run into a smaller object
# save features of data
features <- features(data_proc)
# select desired pixel range to form square around region of interest
pixels <- pixels(data_proc,
                  x>= 2250 & x<=3500,
                  y>= 625 & y<=850)
# make object with features of selected area
data_cut <- data_proc[features,pixels]
```

Now, when the smaller code chunk is projected as an image it only includes the desired region.

```r
# image of selected area
image(data_cut,
      mz=104,
      colorscale=magma)
```

```r
# image any m/z value
# simply change the mz= parameter to target
image(data_cut, mz=86.21196)
```
Data interpretation

Visual interrogation of images for quality/consistency is subjective but is a first pass at assessing data quality. For these data shown in Fig. 7, the left set shows no distinctive features from the zebrafish embryo, while the center and right sets show outlines of the eye and notochord. For method development experiments, this is a useful early indicator of which preparation is working best (Fig. 8).

Unsupervised data exploration

Data analysis is typically broken up into “supervised” and “unsupervised” analyses. The former focuses on looking at specific target molecules and performing desired statistical analyses. The latter aims to look at the underlying data and any patterns that can be discerned, and has been the focus of numerous papers and reviews, as the field of MSI data analysis has grown significantly.[41–43] We will focus here on two techniques that can be quickly utilized for our two main data analysis questions: are the samples of interest similar enough to each other for comparison, and are the samples prepared in an adequately similar fashion to yield similar results? Data dimension reducing techniques, such as spatial shrunken centroids (SSC) and principal component analysis (PCA) can be used to parse MSI datasets without the need for targeted user inputs.

Data dimension reducing techniques, such as principal component analysis and spatial shrunken centroids, can be used to parse MSI datasets without the need for targeted user inputs.

Spatial shrunken centroid (SSC) function

In the spatial shrunken centroid function, there are four user changeable parameters, which are “method,” “r,” “s,” and “k.” The method picks the type of spatial weights to be used and the typical options are “adaptive” or “gaussian.” Here we use “adaptive” to attempt to preserve the image along the tissue edges better. The “r” value defines the smoothing radius. The “s” value alters the way peaks are chosen and is also called the shrinkage parameter; higher values result in fewer peaks being used for the segmentation analysis and it is typical to start in the range of 0–10. The “k” parameter is the number of segments desired and is what is most often changed by the user; this will have the greatest effect on your visual output. While it is beyond the scope of this tutorial,
Fig. 6. Image of intensity for all pixels in imaging run. Sum signal intensity shown by colorscale is not associated with a particular \( m/z \) value.

Fig. 7. A \( m/z \) image for the right (tricaine sacrifice) side of the slide demonstrating how to selectively image ROIs for optimal visualization.
SSC is capable of powerful classification models between specified regions and is explained in the Cardinal documentation and other review articles.[41,43,44]

Interpretation of SSC

In this dataset, 4 classes were specified and they show underlying trends in the data. Visual analysis reveals the far left DHB section in orange and purple appears to spread into the first 5 nm AuNP region, which also displays green and blue. The sections on the right that span DHB and AuNPs do not have any overspraying. This was not a purposeful overspraying and was difficult to see by visual inspection of the actual slide; overspraying would likely have not been noticed without this calculation. Since the trends that are analyzed are not correlated with biological conditions or sample preparation, the user must inspect this output to confirm the veracity. Beyond visual interpretation, numerical analysis is also possible. Using the topFeatures() function the top 10 m/z values used to determine each class are displayed. The class can be changed in the class parameter to get a listing for each of the classes.

```
# ssc
data_proc.ssc <- spatialShrunkenCentroids(data_proc, method="adaptive", r=2, s=15, k=4)

# save created object
saveRDS(data_proc.ssc, file="data_proc_ssc_new.rds")
```

```
# image results of ssc
ssc <- image(data_proc.ssc,
             model=list(s=15))
ssc
```

Fig. 8. A m/z image of a target m/z of 86.2 showing strong intensity in different preparation conditions.
that have been defined. This listing is a good start for looking for features of interest within your dataset and generating m/z images is a common next step.

```r
# determine top m/z features

topFeatures(data_proc_ssc, model=list(s=15), class == 2)

## Top-ranked features:

| # | mz     | count  | freq | r  | k | s | class | centers | statistic |
|---|--------|--------|------|----|---|---|-------|---------|-----------|
| 1 | 184.28504 | 43745  | 0.34335118 | 2  | 4 | 15 | 2     | 256.14417 | 646.9344  |
| 2 | 147.21064 | 49406  | 0.38778393 | 2  | 4 | 15 | 2     | 114.14313 | 614.1169  |
| 3 | 86.27770  | 44807  | 0.35168673 | 2  | 4 | 15 | 2     | 318.15212 | 611.2457  |
| 4 | 72.27624  | 25251  | 0.19819318 | 2  | 4 | 15 | 2     | 72.62282  | 454.9673  |
| 5 | 87.30807  | 37284  | 0.29263928 | 2  | 4 | 15 | 2     | 56.68679  | 439.2428  |
| 6 | 185.31779 | 25215  | 0.19791062 | 2  | 4 | 15 | 2     | 29.83917  | 391.6039  |
| 7 | 146.26699 | 46318  | 0.36354646 | 2  | 4 | 15 | 2     | 41.13210  | 373.6848  |
| 8 | 166.29970 | 20051  | 0.15737877 | 2  | 4 | 15 | 2     | 21.61741  | 369.6334  |
| 9 | 71.26435  | 21731  | 0.17056497 | 2  | 4 | 15 | 2     | 24.55040  | 346.7850  |
| 10| 206.33796 | 9703   | 0.07615811 | 2  | 4 | 15 | 2     | 24.92451  | 311.0347  |
```

Fig. 9. SSC generated from R. The five data regions (from left to right) correspond to: DHB liquid nitrogen sacrifice, 5 nm AuNPs liquid nitrogen sacrifice, 2 nm AuNPs liquid nitrogen sacrifice, DHB tricaine sacrifice, 5 nm AuNPs tricaine sacrifice.
Principal component analysis function

Dimension reduction by orthogonal transformation is accomplished using principal component analysis (PCA), which has emerged as a common approach for unsupervised exploration of MSI data. The first PC is defined so that it explains the largest possible variance in the dataset. Each subsequent PC is defined so that it explains the largest possible variance for the remainder of the dataset, and so on and so on. The more PCs that are defined, the less impact they should have on the overall dataset. However, this does mean that how you define your regions of interest are of critical importance, as a whole-body section will naturally have high heterogeneity, but a single organ tissue section should be more consistent, unless there is a disease state present. The command lines for generating a PCA and subsequent image are below, with the results shown in Fig. 10.

```r
# PCA
# ncomp parameter is number of principal components
data_proc_pca <- PCA(data_proc, ncomp=4)
# save object due to longer run time
saveRDS(data_proc_pca, file="data_proc_pca_new.rds")
# image results of PCA
image(data_proc_pca, contrast.enhance="histogram", normalize.image="linear")
```

Analysis of PCA

There are several points to note in this analysis. First, the unsupervised method again picked up on accidental overspray of DHB matrix into the 5 nm AuNP zone. For the AuNP regions without overspraying, small differences can still be observed between 5 nm AuNPs and 2 nm AuNPs; interestingly, the tissue and background are difficult to distinguish from each other on the far-right set (2 nm AuNPs tricaine sacrifice). This was not observed using SSC, making both tools useful. For the DHB sprayed sections, it is easy to tell matrix only area from tissue area, allowing for a quick assessment of spray consistency, which is one of the most important MSI sample preparation parameters for successfully acquiring data.

Conclusion

This R tutorial for introductory use in MSI has been intended to guide newer users through an open-source option for assessment of MSI data, as well as the tools to progress into more complex data analysis. Any manuscript dealing with MSI data analysis must limit its scope because of the plethora of data science methods that are available, although references have been provided throughout to aid in further knowledge acquisition.
[42] H. Hu, R. Yin, H.M. Brown, J. Laskin, Spatial segmentation of mass spectrometry imaging data by combining multivariate clustering and univariate thresholding, Anal. Chem. 93 (7) (2021) 3477–3485.

[43] D. Guo, K. Bemis, C. Rawlins, J. Agar, O. Vitek, Unsupervised segmentation of mass spectrometric ion images characterizes morphology of tissues, Bioinformatics 35 (2019) i208–i217, https://doi.org/10.1093/BIOINFORMATICS/BTZ345.

[44] W. Zhang, M. Claesen, T. Moerman, M.R. Groseclose, E. Waelkens, B. De Moor, N. Verbeeck, Spatially aware clustering of ion images in mass spectrometry imaging data using deep learning, Anal. Bioanal. Chem. 413 (10) (2021) 2803–2819, https://doi.org/10.1007/s00216-021-03179-w.