# MALDIquant: Quantitative Analysis of Mass Spectrometry Data

Sebastian Gibb*

January 10, 2017

**Abstract**

MALDIquant provides a complete analysis pipeline for MALDI-TOF and other 2D mass spectrometry data.
This vignette describes the usage of the MALDIquant package and guides the user through a typical preprocessing workflow.

*mail@sebastiangibb.de

# Contents

# Foreword

`MALDIquant` is free and open source software for the `R` (R Core Team, 2014) environment and under active development. If you use it, please support the project by citing it in publications:

> Gibb, S. and Strimmer, K. (2012). MALDIquant: a versatile R package for the analysis of mass spectrometry data. *Bioinformatics*, 28(17):2270–2271

If you have any questions, bugs, or suggestions do not hesitate to contact me (`mail@sebastiangibb.de`).
Please visit `http://strimmerlab.org/software/maldiquant/`.

# 1 Introduction

MALDIquant comprising all steps from importing of raw data, preprocessing (e.g. baseline removal), peak detection and non-linear peak alignment to calibration of mass spectra.

MALDIquant was initially developed for clinical proteomics using Matrix-Assisted Laser Desorption/Ionization (MALDI) technology. However, the algorithms implemented in MALDIquant are generic and may be equally applied to other 2D mass spectrometry data.

MALDIquant was carefully designed to be independent of any specific mass spectrometry hardware. Nonetheless, a lot of open and native file formates, e.g. binary data files from Bruker flex series instruments, mzXML, mzML, etc. are supported through the associated R package MALDIquantForeign.

# 2 Setup

After starting R we could install MALDIquant and MALDIquantForeign directly from CRAN using install.packages:

```
> install.packages(c("MALDIquant", "MALDIquantForeign"))
```

Before we can use MALDIquant we have to load the package.

```
> library("MALDIquant")
```

# 3 MALDIquant objects

MALDIquant is written in an object-oriented programming approach and uses R's S4 objects. A spectrum is represented by an MassSpectrum and a list of peaks by an MassPeaks instance. To create such objects manually we could use createMassSpectrum and createMassPeaks. In general we do not need these functions because MALDIquantForeign's import routines will generate the MassSpectrum/MassPeaks objects.

```
> s <- createMassSpectrum(mass=1:10, intensity=1:10,
+                         metaData=list(name="Spectrum1"))
> s

S4 class type            : MassSpectrum
Number of m/z values     : 10
Range of m/z values      : 1 - 10
Range of intensity values: 1 - 10
Memory usage             : 1.328 KiB
Name                     : Spectrum1
```

Each `MassSpectrum`/`MassPeaks` stores the mass and intensity values of a spectrum respective of the peaks. Additionally they contain a list of metadata. To access these information we use `mass`, `intensity` and `metaData`.

```
> mass(s)

 [1]  1  2  3  4  5  6  7  8  9 10

> intensity(s)

 [1]  1  2  3  4  5  6  7  8  9 10

> metaData(s)

$name
[1] "Spectrum1"
```

# 4 Workflow

A Mass Spectrometry Analysis often follows the same workflow (see also Fig. 1). After importing the raw data (see also the `MALDIquantForeign` package) we control the quality of the spectra and draw some plots. We apply a variance-stabilizing transformation and smoothing filter. Next we remove the chemical background using a Baseline Correction method. To compare the intensities across spectra we calibrate the intensity values (often called normalization) and the mass values (warping, alignment). Subsequently we perfom a Peak Detection and do some post processing like filtering etc.

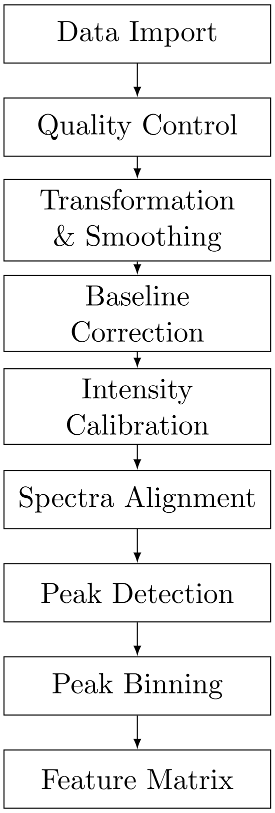


Figure 1: MS Analysis Workflow

## 4.1 Data Import

Normally we will use some of the import methods provided by `MALDIquantForeign`, e.g. `importBrukerFlex`, `importMzMl`, etc. But in this vignette we will use a small example dataset shipped with `MALDIquant`. This dataset is a subset of MALDI-TOF data described in Fiedler et al. (2009).

```
> data(fiedler2009subset)
```

`fiedler2009subset` is a `list` of 16 `MassSpectrum` objects. The 16 spectra are 8 biological samples with 2 technical replicates.

```
> length(fiedler2009subset)

[1] 16

> fiedler2009subset[1:2]

$sPankreas_HB_L_061019_G10.M19.T_0209513_0020740_18
S4 class type            : MassSpectrum
Number of m/z values     : 42388
Range of m/z values      : 1000.015 - 9999.734
Range of intensity values: 5 - 101840
Memory usage             : 505.367 KiB
Name                     : Pankreas_HB_L_061019_G10.M19
File                     : /data/set A - discovery leipzig/control/Pankreas_HB_L_

$sPankreas_HB_L_061019_G10.M20.T_0209513_0020740_18
S4 class type            : MassSpectrum
Number of m/z values     : 42388
Range of m/z values      : 1000.015 - 9999.734
Range of intensity values: 6 - 111862
Memory usage             : 505.367 KiB
Name                     : Pankreas_HB_L_061019_G10.M20
File                     : /data/set A - discovery leipzig/control/Pankreas_HB_L_
```

## 4.2 Quality Control

For a basic quality control we test whether all spectra contain the same number of data points and are not empty.

```
> any(sapply(fiedler2009subset, isEmpty))

[1] FALSE

> table(sapply(fiedler2009subset, length))


42388
   16
```

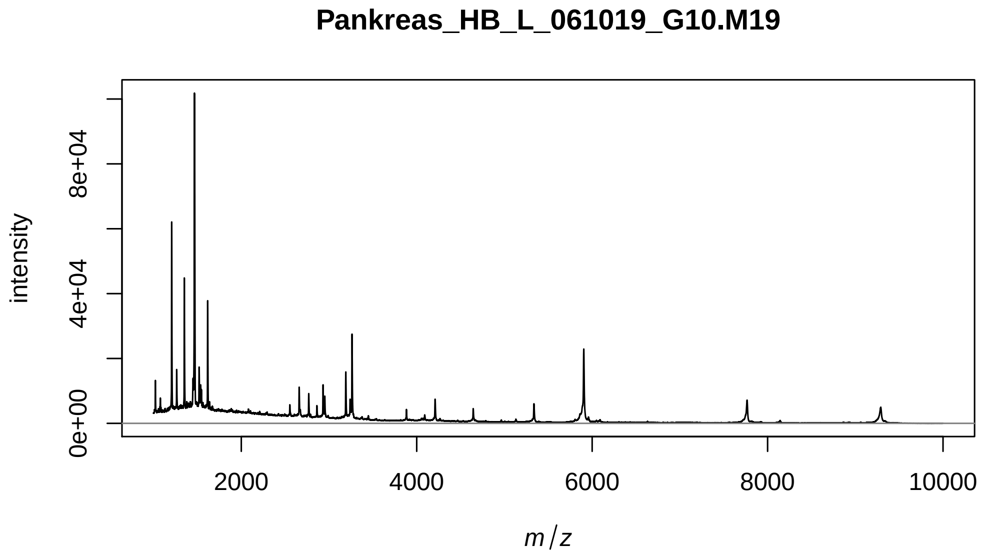
Subsequently we control the mass difference between each data point (should be equal or monotonically increasing) because `MALDIquant` is designed for profile data and not for centroided data.

```
> all(sapply(fiedler2009subset, isRegular))

[1] TRUE
```

Finally we draw some plots and inspect the spectra visually.

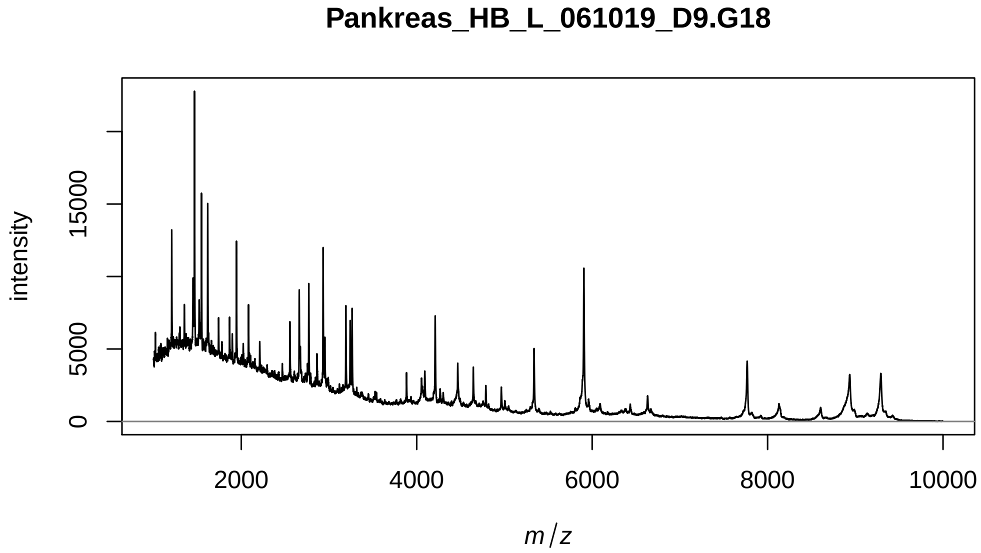
```
> plot(fiedler2009subset[[1]])
```



Pankreas_HB_L_061019_G10.M19

/data/set A − discovery leipzig/control/Pankreas_HB_L_061019_G10/0_m19/1/1SLin/fid

```
> plot(fiedler2009subset[[16]])
```



Pankreas_HB_L_061019_D9.G18

/data/set B − discovery heidelberg/tumor/Pankreas_HB_L_061019_D9/0_g18/1/1SLin/fid

## 4.3 Variance Stabilization

We use the square root transformation to simplify graphical visualization and to overcome the potential dependency of the variance from the mean.

```
> spectra <- transformIntensity(fiedler2009subset,
+                                method="sqrt")
```

## 4.4 Smoothing

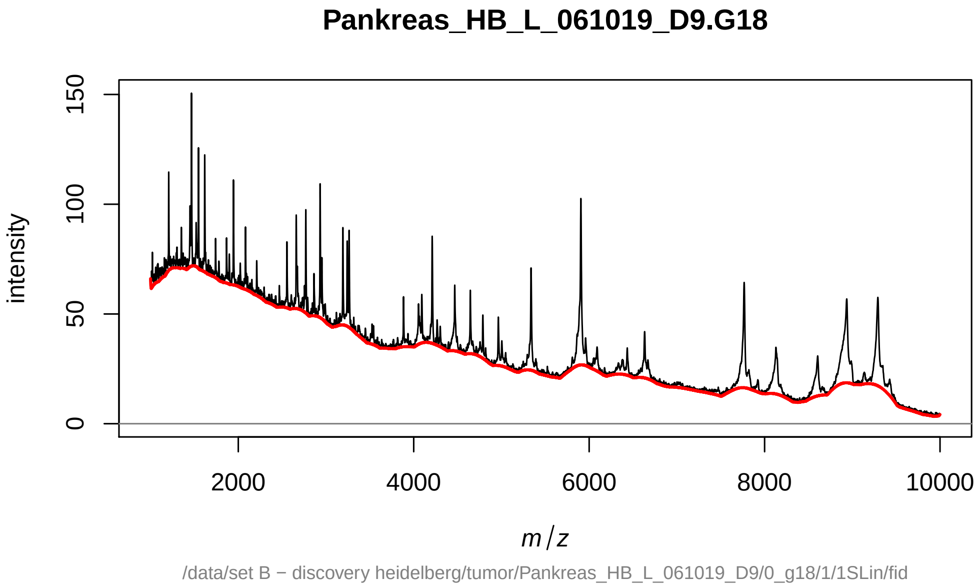
Next we use a 21 point Savitzky-Golay-Filter (Savitzky and Golay, 1964) to smooth the spectra.

```
> spectra <- smoothIntensity(spectra, method="SavitzkyGolay",
+                             halfWindowSize=10)
```

## 4.5 Baseline Correction

Before we correct the baseline we visualize it. Here we use the SNIP algorithm (Ryan et al., 1988).

```
> baseline <- estimateBaseline(spectra[[16]], method="SNIP",
+                               iterations=100)
> plot(spectra[[16]])
> lines(baseline, col="red", lwd=2)
```

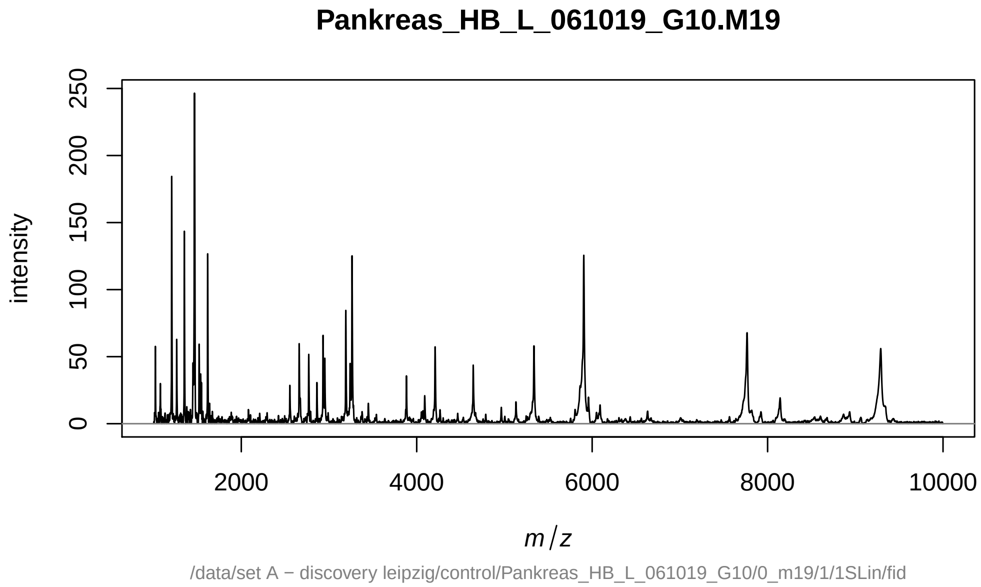


Pankreas_HB_L_061019_D9.G18

/data/set B − discovery heidelberg/tumor/Pankreas_HB_L_061019_D9/0_g18/1/1SLin/fid

If we are satisfied with our estimated baseline we remove it.

```
> spectra <- removeBaseline(spectra, method="SNIP",
+                           iterations=100)
> plot(spectra[[1]])
```



## 4.6 Intensity Calibration/Normalization

For better comparison and to overcome (very) small batch effects we equalize the intensity values using the Total-Ion-Current-Calibration (often called normalization).

```
> spectra <- calibrateIntensity(spectra, method="TIC")
```

## 4.7 Warping/Alignment

Now we (re)calibrate the mass values. Our alignment procedure is a peak based warping algorithm. If you need a finer control or want to investigate the impact of different parameters please use `determineWarpingFunctions` instead of the easier `alignSpectra`.

```
> spectra <- alignSpectra(spectra,
+                         halfWindowSize=20,
+                         SNR=2,
+                         tolerance=0.002,
+                         warpingMethod="lowess")
```

Before we call the Peak Detection we want to average the technical replicates. Therefore we look for the sample name that is stored in the metadata because each technical replicate has the same sample name.

```
> samples <- factor(sapply(spectra,
+                          function(x)metaData(x)$sampleName))
```

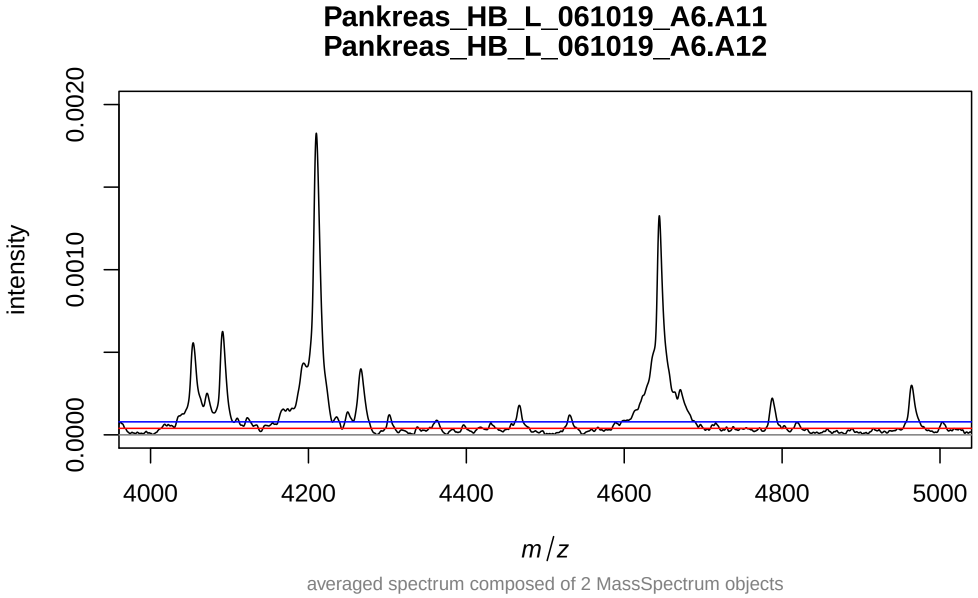
Next we use `averageMassSpectra` to create a mean spectrum for each biological sample.

```
> avgSpectra <- averageMassSpectra(spectra, labels=samples,
+                                  method="mean")
```

## 4.8 Peak Detection

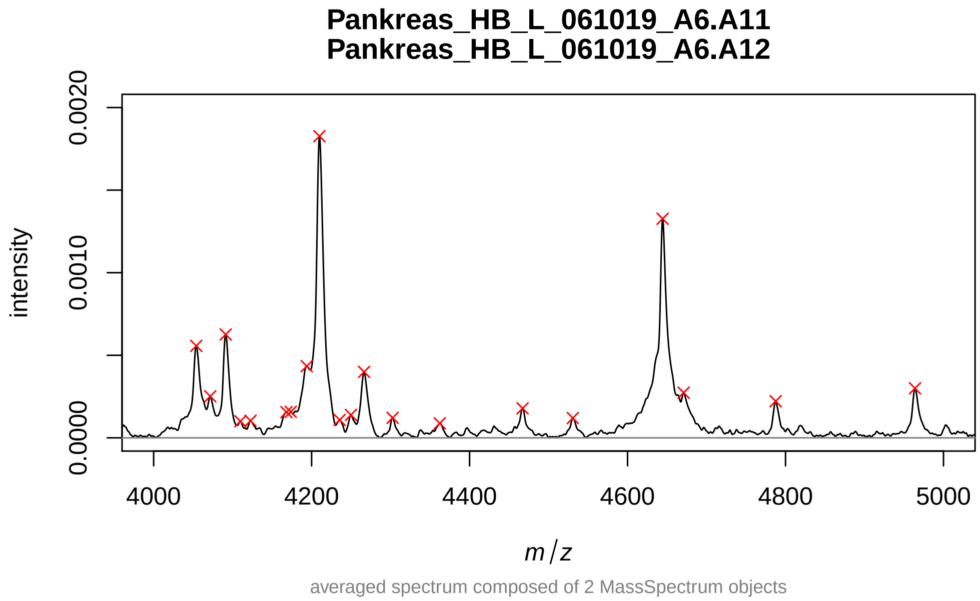
The next crucial step is the Peak Detection. Before we perform the peak detection algorithm we estimate the noise of the spectra to get a feeling for the signal-to-noise ratio.

```
> noise <- estimateNoise(avgSpectra[[1]])
> plot(avgSpectra[[1]], xlim=c(4000, 5000), ylim=c(0, 0.002))
> lines(noise, col="red")
> lines(noise[,1], noise[, 2]*2, col="blue")
```



We decide to use a signal-to-noise ratio of 2 (blue line).

```
> peaks <- detectPeaks(avgSpectra, method="MAD",
+                      halfWindowSize=20, SNR=2)
> plot(avgSpectra[[1]], xlim=c(4000, 5000), ylim=c(0, 0.002))
> points(peaks[[1]], col="red", pch=4)
```



## 4.9 Peak Binning

After the alignment the peak positions (mass) are very similar but not identical. The binning is needed to make similar peak mass values identical.

```
> peaks <- binPeaks(peaks, tolerance=0.002)
```

## 4.10 Feature Matrix

We choose a very low signal-to-noise ratio to keep as much features as possible. To remove some false positive peaks we remove less frequent peaks.

```
> peaks <- filterPeaks(peaks, minFrequency=0.25)
```

At the end of the analysis we create a feature matrix that could be used in further statistical analysis. Please note that missing values (not detected peaks) are imputed/interpolated from the corresponding spectrum.

```
> featureMatrix <- intensityMatrix(peaks, avgSpectra)
> head(featureMatrix[, 1:3])

     1011.73182227583 1020.6748082171 1029.40115131151
[1,]     0.0001894947    0.0007715987     0.0001093035
[2,]     0.0002144354    0.0015030560     0.0001422394
[3,]     0.0002117147    0.0004555688     0.0001303326
[4,]     0.0002314181    0.0005260977     0.0001441254
[5,]     0.0001562401    0.0024054031     0.0001198008
[6,]     0.0001600630    0.0020315191     0.0001090484
```

# 5 Summary

We shortly described a complete example workflow of a mass spectrometry data analysis. Please note that this workflow is only an example and could not cover every use case.
`MALDIquant` provides a lot of more functions than we mentioned in this vignette. The described functions are the most used ones but they have a lot of more parameters which could/need adjust to your data (e.g. `halfWindowSize`, `SNR`, `tolerance`, etc.). That's why we suggest the user to read the manual pages of theses functions carefully.
We also provide more examples in the demo directory and at:

`http://strimmerlab.org/software/maldiquant/`

Please do not hesitate to contact me (`mail@sebastiangibb.de`) if you have any questions.

# 6 Session Information

- R version 3.3.2 (2016-10-31), `x86_64-pc-linux-gnu`
- Base packages: base, datasets, grDevices, graphics, methods, stats, utils
- Other packages: MALDIquant 1.16.1, knitr 1.15.1

- Loaded via a namespace (and not attached): evaluate 0.10, highr 0.6, magrittr 1.5, parallel 3.3.2, stringi 1.1.2, stringr 1.1.0, tools 3.3.2

# References

Fiedler, G. M., Leichtle, A. B., Kase, J., Baumann, S., Ceglarek, U., Felix, K., Conrad, T., Witzigmann, H., Weimann, A., Schtte, C., Hauss, J., Bchler, M., and Thiery, J. (2009). Serum peptidome profiling revealed platelet factor 4 as a potential discriminating peptide associated with pancreatic cancer. *Clin Cancer Res*, 15(11):3812–3819.

Gibb, S. and Strimmer, K. (2012). MALDIquant: a versatile R package for the analysis of mass spectrometry data. *Bioinformatics*, 28(17):2270–2271.

R Core Team (2014). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Ryan, C., Clayton, E., Griffin, W., Sie, S., and Cousens, D. (1988). Snip, a statistics-sensitive background treatment for the quantitative analysis of pixe spectra in geoscience applications. *Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms*, 34(3):396 – 402.

Savitzky, A. and Golay, M. J. E. (1964). Smoothing and differentiation of data by simplified least squares procedures. *Analytical Chemistry*, 36(8):1627–1639.