RMassBank: Non-standard usage

Michael Stravs

October 30, 2017

Contents

1 Introduction

This vignette assumes you are familiar with the standard usage of *RMassBank*, which is documented in

```
> vignette("RMassBank")
```

2 Skipping recalibration

For instances where recalibration is not wanted, e.g. there is not enough data, or the user wants to use non-recalibrated data, recalibration can be deactivated. To do this, the `recalibrator` entry in the settings must be set to `recalibrate.identity`. This can be done in the settings file directly (preferred):

```
# [...]
recalibrator:
    MS1: recalibrate.identity
    MS2: recalibrate.identity
# [...]
```

Or, alternatively, the settings can be adapted directly via R code.

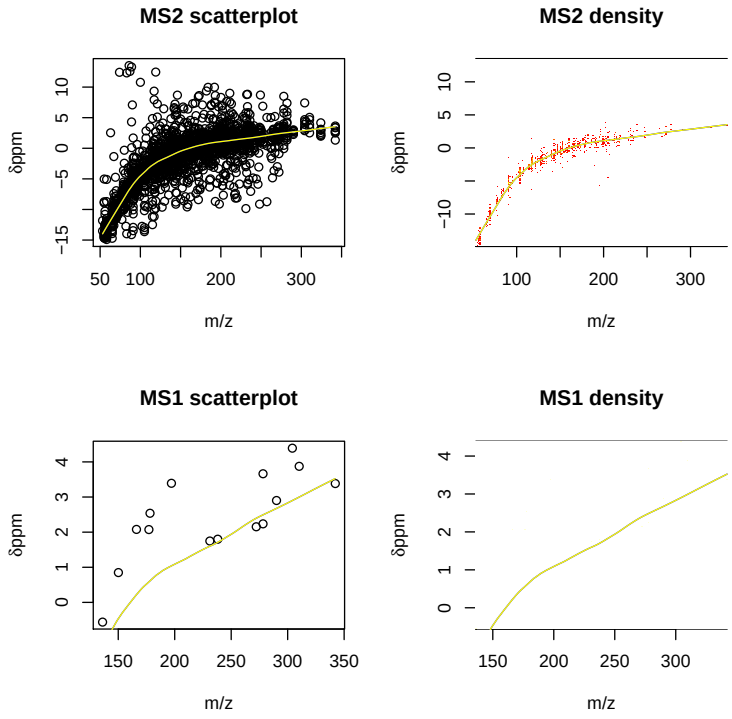
```
> RmbDefaultSettings()
> rmbo <- getOption("RMassBank")
> rmbo$recalibrator <- list(
+                 "MS1" = "recalibrate.identity",
+                 "MS2" = "recalibrate.identity"
+         )
> options("RMassBank" = rmbo)
```

To show the results of using a non-recalibrated workflow, we load a workspace with pre-processed data:

```
> w <- loadMsmsWorkspace(system.file("results/pH_narcotics_RF.RData",
+                                    package="RMassBankData"))
```

The recalibration curve:

```
> recal <- makeRecalibration(w@parent, "pH",
+                                    recalibrateBy = rmbo$recalibrateBy,
+                                    recalibrateMS1 = rmbo$recalibrateMS1,
+                                    recalibrator = list(MS1="recalibrate.loess",MS2="re
+                                    recalibrateMS1Window = 15)
> w@rc <- recal$rc
> w@rc.ms1 <- recal$rc.ms1
> w@parent <- w
> plotRecalibration(w)
```



Some example peaks to show the effect of recalibration:

```
> w@spectra[[1]]@parent@mz[30:32]
```

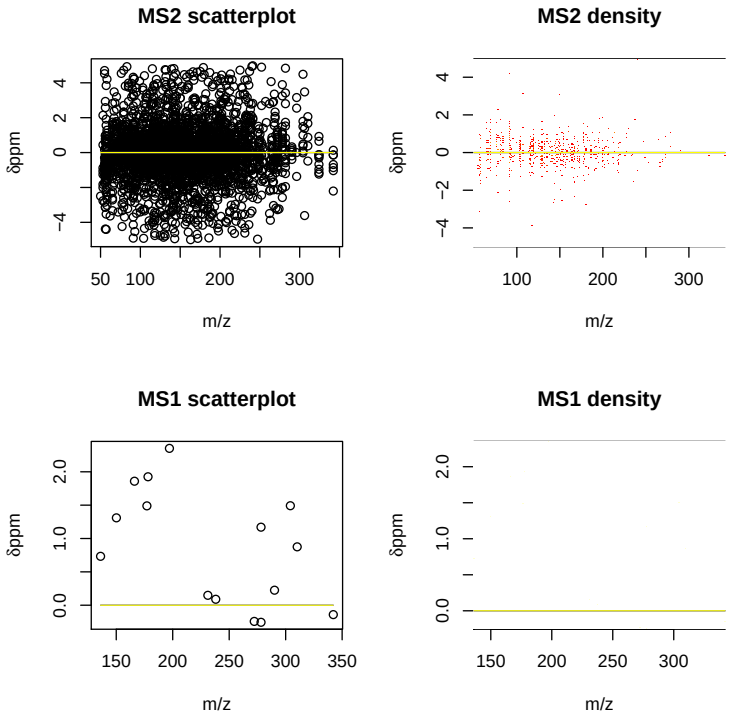
```
[1] 133.0346 133.0381 133.9772
```

```
> w@spectra[[1]]@children[[1]]@mz[15:17]
```

```
[1] 197.0844 138.0113 162.1139
```

Now reprocess the recalibration step with the above set `recalibration.identity`:

```
> w <- msmsWorkflow(w, steps=4)
```



The recalibration graph shows that the recalibration "curve" will do no recalibration. To verify, we can show the same peaks as before:

```
> w@spectra[[1]]@parent@mz[30:32]
```

```
[1] 133.0346 133.0381 133.9772
```

```
> w@spectra[[1]]@children[[1]]@mz[15:17]
```

```
[1] 87.00286 88.13592 97.10767
```

3 Combining multiplicities

Standard multiplicity filtering, which is configurable in the settings, eliminates peaks which are observed only once for a compound. This eliminates spurious formula matches for random noise efficiently. It works well if either many spectra are recorded per compound, or if the same collision energy is present twice (e.g. with different resolutions). It sometimes fails for spectra on the "outer end" of the recorded collision energies when that spectrum is only present once – peaks which appear only in the highest or only in the lowest recorded energy can be erroneously deleted. To prevent this, one can re-run the workflow, read a second set of spectra for every compound (the second most intense) and combine the peak multiplicities of the two analyzed runs. (Mutiplicity filtering can also be switched off completely.)

Example:

```
> RmbDefaultSettings()
> getOption("RMassBank")$multiplicityFilter
```

```
[1] 2
```

```
> # to make processing faster, we only use 3 spectra per compound
> rmbo <- getOption("RMassBank")
> rmbo$spectraList <- list(
+     list(mode="CID", ces = "35%", ce = "35 % (nominal)", res = 7500),
+     list(mode="HCD", ces = "15%", ce = "15 % (nominal)", res = 7500),
+     list(mode="HCD", ces = "30%", ce = "30 % (nominal)", res = 7500)
+ )
> options(RMassBank = rmbo)
> loadList(system.file("list/NarcoticsDataset.csv",
+         package="RMassBankData"))
> w <- newMsmsWorkspace()
```

```
> files <- list.files(system.file("spectra", package="RMassBankData"),
+         ".mzML", full.names = TRUE)
> w@files <- files[1:2]
```

First, the spectra are read and processed until reanalysis (step 7) normally:

```
> w1 <- msmsWorkflow(w, mode="pH", steps=c(1))
> # Here we artificially cut spectra out to make the workflow run faster for the vign
> w1@spectra <- as(lapply(w1@spectra, function(s)
+     {
+                 s@children <- s@children[1:3]
+                 s
+     }),"SimpleList")
> w1 <- msmsWorkflow(w1, mode="pH", steps=c(2:7))
```

Subsequently, we re-read and process the "confirmation spectra", i.e. the second-best spectra from the files. Therefore, we will have two sets of spectra for each compound and every real peak should in theory occur twice.

```
> w2 <- msmsWorkflow(w, mode="pH", steps=c(1), confirmMode = 1)
> # Here we artificially cut spectra out to make the workflow run faster for the vign
>
> w2@spectra <- as(lapply(w2@spectra, function(s)
+     {
+                 s@children <- s@children[1:3]
+                 s
+     }),"SimpleList")
> w2 <- msmsWorkflow(w2, mode="pH", steps=c(2:7))
```

Finally, we combine the two workspaces for multiplicity filtering, and apply the last step in the workflow (multiplicity filtering).

```
> wTotal <- combineMultiplicities(c(w1, w2))
> wTotal <- msmsWorkflow(wTotal, steps=8, mode="pH", archivename = "output")
```

Subsequently, we can proceed as usual with `mbWorkflow`:

```
> mb <- newMbWorkspace(wTotal)
> # [...] load lists, execute workflow etc.
```

4 Session information

```
> sessionInfo()
```

```
R version 3.4.2 (2017-09-28)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: OS X El Capitan 10.11.6

Matrix products: default
BLAS: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/3.4/Resources/lib/libRlapack.dylib

locale:
[1] C/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] gplots_3.0.1         RMassBankData_1.15.0 RMassBank_2.6.0
[4] Rcpp_0.12.13

loaded via a namespace (and not attached):
 [1] compiler_3.4.2        BiocInstaller_1.28.0  plyr_1.8.4
 [4] rcdklibs_2.0          bitops_1.0-6          ProtGenerics_1.10.0
 [7] iterators_1.0.8       tools_3.4.2           zlibbioc_1.24.0
[10] MALDIquant_1.16.4     digest_0.6.12         tibble_1.3.4
[13] preprocessCore_1.40.0 gtable_0.2.0          lattice_0.20-35
[16] png_0.1-7             rlang_0.1.2           foreach_1.4.3
[19] yaml_2.1.14           parallel_3.4.2        rJava_0.9-9
[22] caTools_1.17.1        gtools_3.5.0          S4Vectors_0.16.0
[25] IRanges_2.12.0        stats4_3.4.2          grid_3.4.2
[28] Biobase_2.38.0        impute_1.52.0         rcdk_3.4.3
[31] XML_3.98-1.9          BiocParallel_1.12.0   gdata_2.18.0
```

```
[34] limma_3.34.0          ggplot2_2.2.1         mzR_2.12.0
[37] itertools_0.1-3       scales_0.5.0          pcaMethods_1.70.0
[40] codetools_0.2-15      BiocGenerics_0.24.0   fingerprint_3.5.6
[43] mzID_1.16.0           MSnbase_2.4.0         colorspace_1.3-2
[46] KernSmooth_2.23-15    affy_1.56.0           RCurl_1.95-4.8
[49] lazyeval_0.2.1        munsell_0.4.3         doParallel_1.0.11
[52] rjson_0.2.15          vsn_3.46.0            affyio_1.48.0
```