RMassBank: Non-standard usage

Michael Stravs

April 16, 2015

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.

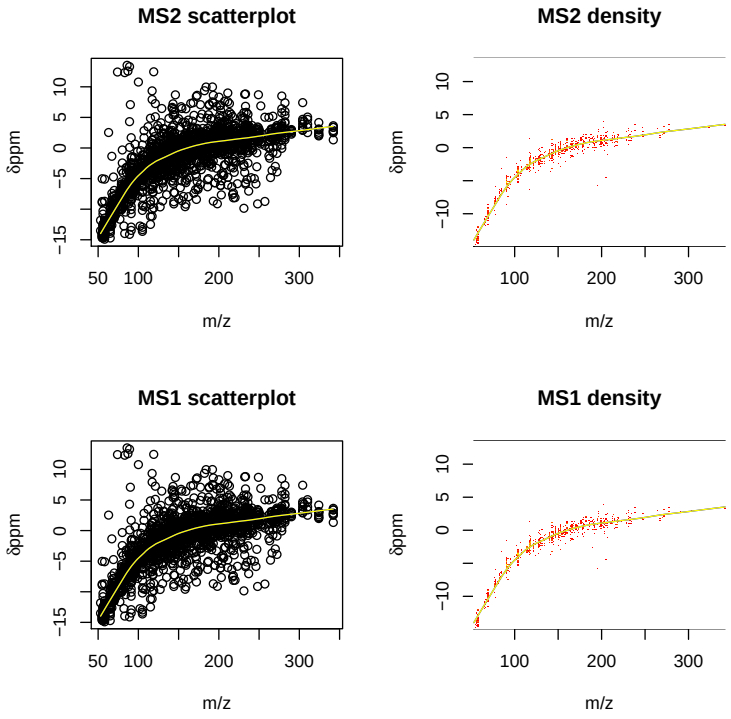
```
> RmbDefaultSettings()
> rmbo <- getOption("RMassBank")
> rmbo$recalibrator <- list(
+                 "MS1" = "recalibrate.identity",
+                 "MS2" = "recalibrate.identity"
+         )
> options("RMassBank" = rmbo)
```

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 stored recalibration curve:

```
> plotRecalibration(w)
```



Some example peaks to show the effect of recalibration:

```
> w@recalibratedSpecs[[1]]$parentPeak[30:32,]
```

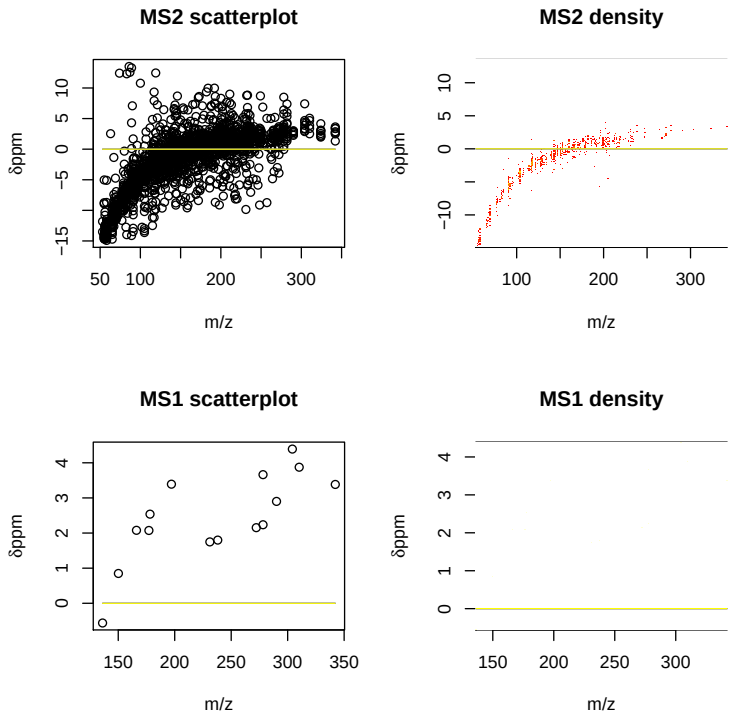
```
           mz        int  mzRecal
[1,] 133.0344 135120.094 133.0346
[2,] 133.0379   1278.020 133.0381
[3,] 133.9770   1347.847 133.9772
```

```
> w@recalibratedSpecs[[1]]$peaks[[1]][15:17,]
```

```
           mz      int  mzRecal
[1,] 87.00228 11738.93 87.00286
[2,] 88.13535 10237.29 88.13592
[3,] 97.10719 12170.58 97.10767
```

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@recalibratedSpecs[[1]]$parentPeak[30:32,]
```

```
           mz        int  mzRecal
[1,] 133.0344 135120.094 133.0344
```

```
[2,] 133.0379   1278.020 133.0379
[3,] 133.9770   1347.847 133.9770
```

```
> w@recalibratedSpecs[[1]]$peaks[[1]][15:17,]
```

```
           mz      int  mzRecal
[1,] 87.00228 11738.93 87.00228
[2,] 88.13535 10237.29 88.13535
[3,] 97.10719 12170.58 97.10719
```

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@specs <- lapply(w1@specs, function(s)
+     {
+       s$childScans <- s$childScans[1:3]
+       s$childHeaders <- s$childHeaders[1:3,]
+       s$peaks <- s$peaks[1:3]
+       s
+     })
> 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@specs <- lapply(w2@specs, function(s)
+     {
+       s$childScans <- s$childScans[1:3]
+       s$childHeaders <- s$childHeaders[1:3,]
+       s$peaks <- s$peaks[1:3]
+       s
+     })
> 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 3.2.0 RC (2015-04-08 r68161)
Platform: x86_64-apple-darwin10.8.0 (64-bit)
Running under: OS X 10.6.8 (Snow Leopard)

locale:
[1] en_US.UTF-8/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_2.16.0       RMassBankData_1.5.0 RMassBank_1.10.0
[4] Rcpp_0.11.5

loaded via a namespace (and not attached):
 [1] codetools_0.2-11    XML_3.98-1.1        gtools_3.4.2
 [4] png_0.1-7           bitops_1.0-6        rcdk_3.3.2
 [7] KernSmooth_2.23-14  gdata_2.13.3        mzR_2.2.0
[10] rjson_0.2.15        iterators_1.0.7     tools_3.2.0
[13] Biobase_2.28.0      ProtGenerics_1.0.0  RCurl_1.95-4.5
```

```
[16] fingerprint_3.5.2   rcdklibs_1.5.8.4    yaml_2.1.13
[19] parallel_3.2.0      BiocGenerics_0.14.0 caTools_1.17.1
[22] rJava_0.9-6
```