

# Applying Metab

Raphael Aggio

April 26, 2022

## Introduction

This document describes how to use the function included in the R package Metab.

## 1 Requirements

Metab requires 3 packages: `xcms`, `svDialogs` and `pander`. You can install these packages straight from [www.bioconductor.org](http://www.bioconductor.org).

## 2 Why should I use Metab?

Metab is an R package for processing metabolomics data previously analysed by the Automated Mass Spectral Deconvolution and Identification System (AMDIS). AMDIS can be found at: <http://chemdata.nist.gov/mass-spc/amdis/downloads/>. AMDIS is one of the most used software for deconvoluting and identifying metabolites analysed by Gas Chromatography - Mass Spectrometry (GC-MS). It is excellent in deconvoluting chromatograms and identifying metabolites based on a spectral library, which is a list of metabolites with their respective mass spectrum and their associated retention times. Although AMDIS is widely and successfully applied to chemistry and many other fields, it shows some limitations when applied to biological studies. First, it generates results in a single spreadsheet per sample, which means that one must manually merge the results provided by AMDIS in a unique spreadsheet for performing further comparisons and statistical analysis, for example, comparing the abundances of metabolites across experimental conditions. AMDIS also allows users to generate a single report containing the results for a batch of samples. However, this report contains the results of samples placed on top of each other, which also requires extensive manual process before statistical analysis. In addition, AMDIS shows some limitations when quantifying metabolites. It quantifies metabolites by calculating the area (Area) under their respective peaks or by calculating the abundance of the ion mass fragment (Base.Peak) used as model to deconvolute the peak associated with each specific metabolite. As the area of a peak may

be influenced by coelution of different metabolites, the abundance of the most abundant ion mass fragment is commonly used for quantifying metabolites in biological samples. However, AMDIS may use different ion mass fragments for quantifying the same metabolite across samples, which indicates that using AMDIS results one is not comparing the same variable across experimental conditions. Finally, according to the configurations used when applying AMDIS, it may report more than one metabolite identified for the same retention time. Therefore, AMDIS data requires manual inspection to define the correct metabolite to be assigned to each retention time.

Metab solves AMDIS limitations by selecting the most probable metabolite associated to each retention time, by correcting the Base.Peak values calculated by AMDIS and by combining results in a single spreadsheet and in a format that suits further data processing. In order to select the most probable metabolite associated to each retention time, Metab considers the number of question marks reported by AMDIS, which indicates its certainty in identification, and the difference between expected and observed retention times associated with each metabolite. For correcting abundances calculated by AMDIS, Metab makes use of an ion library containing the ion mass fragment to be used as reference when quantifying each metabolite present in the mass spectral library applied. For this, Metab collects from the AMDIS report the scan used to identify each metabolite and collects from the raw data (CDF files) the intensities of their reference ion mass fragments defined in the ion library. In addition, Metab contains functions to simply reformat AMDIS reports into a single spreadsheet containing identified metabolites and their Areas or Base.Peaks calculated by AMDIS in each analysed sample. Therefore, Metab can be used to quickly process AMDIS reports correcting or not metabolite abundances previously calculated by AMDIS. Below we demonstrate how to use each function in Metab.

### 3 How to process AMDIS results using MetReport

**MetReport** automatically process ADMIS results keeping only one compound for each retention time. In addition, **MetReport** can be used to recalculate peak intensities by assigning a fixed mass fragment for each compound across samples, or to return the Area or Base.Peaks previously calculated by AMDIS. **MetReport** may be applied to a single GC-MS file or a batch of GC-MS files.

When applied to a single file and recalculating metabolite abundances, **MetReport** requires:

1. the GC-MS sample file in CDF format. The software used by most GC-MSs include an application to convert GC-MS files to CDF format (also known as AIA format). If not available in the GC-MS software used, there are commercial software available at the market.
2. Amdis report in batch mode. It is a text file containing the results for a batch of samples and can be obtained in AMDIS through: **File > Batch Job > Create and Run Job...** Select the Analysis Type to be used, generally **Simple**, click on **Generate Report** and **Report all hits**. Click on **Add...**, select the files to be analysed, click on **Save As...**, select the folder where the report will be generated and a name for this report (any name you desire). Finally, click on **Run**. A new .TXT file with the name specified will be generated in the folder specified.

Below you can see examples of an AMDIS report:

```
> library(Metab)
> data(exampleAMDISReport)
> print(head(exampleAMDISReport, 25))
```

```

                                                                 FileName
1 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
2 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
3 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
4 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
5 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
6 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
7 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
8 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
9 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
10 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
11 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
12 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
13 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
14 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
15 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
16 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_1.FIN
17 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_2.FIN
18 M:\\Metab\\StandardSolutions_FinalSmallLib\\uL50\\130513_REF_SOL2_2_50_50_2.FIN
```

19 M:\\Metab\\StandardSolutions\_FinalSmallLib\\uL50\\130513\_REF\_SOL2\_2\_50\_50\_2.FIN  
 20 M:\\Metab\\StandardSolutions\_FinalSmallLib\\uL50\\130513\_REF\_SOL2\_2\_50\_50\_2.FIN  
 21 M:\\Metab\\StandardSolutions\_FinalSmallLib\\uL50\\130513\_REF\_SOL2\_2\_50\_50\_2.FIN  
 22 M:\\Metab\\StandardSolutions\_FinalSmallLib\\uL50\\130513\_REF\_SOL2\_2\_50\_50\_2.FIN  
 23 M:\\Metab\\StandardSolutions\_FinalSmallLib\\uL50\\130513\_REF\_SOL2\_2\_50\_50\_2.FIN  
 24 M:\\Metab\\StandardSolutions\_FinalSmallLib\\uL50\\130513\_REF\_SOL2\_2\_50\_50\_2.FIN  
 25 M:\\Metab\\StandardSolutions\_FinalSmallLib\\uL50\\130513\_REF\_SOL2\_2\_50\_50\_2.FIN

	CAS	Name	RT	RI	Width	Purity	Model
1	130513~1-N1002	??? Ethanol	6.6513	NA	18.7 scans	85%	31 m/z
2	130513~1-N1004	??? Acetone	7.3732	NA	15.6 scans	98%	58 m/z
3	130513~1-N1006	Isopropyl alcohol	7.5762	NA	>6 scans	37%	38 m/z
4	130513~1-N1008	Acetonitril	7.9100	NA	11.3 scans	74%	39 m/z
5	130513~1-N1010	Ethyl acetate	10.6013	NA	24.1 scans	97%	43 m/z
6	130513~1-N1012	1-butanol	13.3941	NA	23.1 scans	94%	31 m/z
7	130513~1-N1014	2-pentanone	13.9695	NA	17.6 scans	93%	86 m/z
8	130513~1-N1016	Pyridine	16.4221	NA	15.2 scans	90%	51 m/z
9	130513~1-N1018	?? Zylene1	20.3983	NA	17.3 scans	89%	91 m/z
10	130513~1-N1022	Zylene3	20.3983	NA	17.3 scans	89%	91 m/z
11	130513~1-N1020	Zylene2	20.3983	NA	17.3 scans	89%	91 m/z
12	130513~1-N1020	??? Zylene2	20.6942	NA	15.1 scans	96%	92 m/z
13	130513~1-N1022	Zylene3	20.6942	NA	15.1 scans	96%	92 m/z
14	130513~1-N1018	Zylene1	20.6942	NA	15.1 scans	96%	92 m/z
15	130513~1-N1024	Benzaldehyde	25.6968	NA	12.5 scans	95%	51 m/z
16	130513~1-N1026	Indole	38.6367	NA	9.5 scans	97%	63 m/z
17	130513~1-N1002	Ethanol	6.6479	NA	17.4 scans	86%	31 m/z
18	130513~1-N1004	?? Acetone	7.3709	NA	15.5 scans	94%	42 m/z
19	130513~1-N1006	Isopropyl alcohol	7.5868	NA	15.2 scans	46%	45 m/z
20	130513~1-N1008	Acetonitril	7.9065	NA	11.0 scans	74%	41 m/z
21	130513~1-N1010	Ethyl acetate	10.6024	NA	21.9 scans	99%	45 m/z
22	130513~1-N1012	1-butanol	13.3812	NA	16.8 scans	95%	41 m/z
23	130513~1-N1014	2-pentanone	13.9654	NA	17.6 scans	92%	86 m/z
24	130513~1-N1016	Pyridine	16.4203	NA	14.7 scans	99%	79 m/z
25	130513~1-N1018	Zylene1	20.3959	NA	15.3 scans	88%	91 m/z

	Min..Abund.	Amount	Scan	Peak.Tailing	S.N..total.	Base.Peak	Max..Amount
1	0.02%	0.56%	112	3.2	127	11607327	
2	0.00%	3.76%	236	2.8	346	141000704	
3	0.01%	0.14%	270	0.0	159	31689264	0.46%
4	0.00%	0.95%	328	2.2	210	43608332	
5	0.00%	7.36%	789	3.2	442	197190784	
6	0.01%	2.23%	1267	2.7	269	34300908	
7	0.00%	5.08%	1366	2.1	437	192980720	6.69%
8	0.00%	14.30%	1786	4.8	671	330948544	
9	0.01%	0.74%	2468	2.1	163	24909594	
10	0.01%	0.74%	2468	2.1	163	24909594	
11	0.01%	0.74%	2468	2.1	163	24909594	
12	0.00%	2.56%	2518	2.3	312	72615568	
13	0.00%	2.56%	2518	2.3	312	72615568	
14	0.00%	2.56%	2518	2.3	312	72615568	
15	0.00%	6.24%	3376	0.9	552	139180208	
16	0.00%	1.41%	5593	1.4	292	70967296	
17	0.02%	0.59%	111	3.7	140	13701553	

18	0.00%	3.72%	235	3.4	377	177081840
19	0.00%	0.98%	272	3.0	196	50478668
20	0.00%	1.20%	327	2.7	235	57517692
21	0.00%	7.69%	789	3.0	482	247269568
22	0.00%	2.92%	1265	3.6	324	50734364
23	0.00%	6.46%	1365	3.3	459	223163936
24	0.00%	15.00%	1786	4.2	738	410904384
25	0.01%	0.70%	2467	2.1	168	26507924

	Area	Intgr.Signal	Max..Area	Extra.Width
1	701423866	658392192	NA	01-Apr
2	4725912300	4236371216	NA	01-May
3	174139435	164413150	571334359	1-0
4	1186617973	1091085434	NA	01-Aug
5	9249749212	8620421245	NA	02-Feb
6	2801759237	2572895876	NA	02-Jan
7	6387468112	6107379641	8400498601	2-0
8	18048017764	16633872102	NA	01-Feb
9	932247262	883317974	NA	02-Mar
10	932247262	883317974	NA	02-Mar
11	932247262	883317974	NA	02-Mar
12	3222637797	3013541655	NA	02-Mar
13	3222637797	3013541655	NA	02-Mar
14	3222637797	3013541655	NA	02-Mar
15	7845543202	7109285309	NA	03-Nov
16	1780437467	1681827509	NA	03-Mar
17	825758659	778144626	NA	01-Mar
18	5232415261	4672767401	NA	02-Feb
19	1381316279	1244850204	NA	02-Mar
20	1690150477	1535513148	NA	02-Mar
21	10809291126	10093653761	NA	02-Feb
22	4106993156	3716517247	NA	02-Feb
23	9091606473	8424578101	NA	02-Feb
24	21110512762	19030252325	NA	01-May
25	988934674	916710545	NA	03-Apr

	Models	Frac..Good	Expec..RT	RI.RI.lib.
1	4: 31 45 29 27	0.988	6.64	NA
2	4: 58 39 38 57	1.000	7.37	NA
3	5: 38 44 46 39 37	0.997	7.58	NA
4	4: 39 38 25 12	0.999	7.91	NA
5	5: 43 29 44 30 37	1.000	10.59	NA
6	7: 31 43 45 44 53 51 13	0.999	13.38	NA
7	13: 86 71 58 39 44 26 59 62 51 57 49 30 60	1.000	13.96	NA
8	7: 51 38 48 64 36 25 83	1.000	16.43	NA
9	2: 91 51	0.998	20.40	NA
10	2: 91 51	0.998	21.80	NA
11	2: 91 51	0.998	20.70	NA
12	5: 92 79 53 38 27	0.999	20.70	NA
13	5: 92 79 53 38 27	0.999	21.80	NA
14	5: 92 79 53 38 27	0.999	20.40	NA
15	10: 51 76 62 39 29 38 26 61 90 101	1.000	25.71	NA
16	7: 63 39 87 78 56 77 55	0.999	38.63	NA

17						2: 31 43	0.994	6.64	NA
18		9: 42 27 15 44 26 29 25 55 30					1.000	7.37	NA
19						3: 45 39 31	0.914	7.58	NA
20						2: 41 13	0.998	7.91	NA
21		7: 45 70 29 26 31 87 72					1.000	10.59	NA
22						4: 41 39 27 33	1.000	13.38	NA
23		7: 86 37 50 51 67 59 25					1.000	13.96	NA
24						2: 79 80	1.000	16.43	NA
25		5: 91 79 107 64 50					0.997	20.40	NA

	Net	Weighted	Simple	Reverse	Corrections	X.m.z.	S.N..m.z.	Area....m.z.	Conc.
1	100	100	99	100	NA	31	75.7	35.318	NA
2	100	100	100	100	NA	43	263.9	58.121	NA
3	97	97	94	99	NA	NA	NA	NA	NA
4	100	100	100	100	NA	41	146.7	48.929	NA
5	100	99	99	99	NA	43	312.1	49.857	NA
6	100	100	100	100	NA	56	130.1	23.488	NA
7	100	99	98	100	NA	43	308.7	50.029	NA
8	100	100	98	100	NA	79	404.3	36.287	NA
9	100	100	100	100	NA	91	110.9	46.246	NA
10	98	97	97	97	NA	91	110.9	46.246	NA
11	97	96	96	96	NA	91	110.9	46.246	NA
12	100	100	100	100	NA	91	189.4	36.853	NA
13	100	100	100	100	NA	91	189.4	36.853	NA
14	97	96	96	96	NA	91	189.4	36.853	NA
15	100	100	100	100	NA	106	262.2	22.606	NA
16	100	100	100	100	NA	117	187.2	41.027	NA
17	100	100	99	100	NA	31	81.5	33.838	NA
18	100	100	100	100	NA	43	293.0	60.317	NA
19	98	98	96	99	NA	NA	NA	NA	NA
20	100	100	100	100	NA	41	167.0	50.556	NA
21	100	99	99	99	NA	43	346.2	51.653	NA
22	100	100	100	100	NA	56	156.8	23.468	NA
23	100	100	98	100	NA	43	328.9	51.327	NA
24	100	100	98	100	NA	79	446.3	36.614	NA
25	100	100	100	100	NA	91	113.3	45.551	NA

	RT.RT.lib.
1	0.007
2	0.000
3	-0.006
4	0.005
5	0.008
6	0.013
7	0.010
8	-0.004
9	0.003
10	-1.405
11	-0.299
12	-0.003
13	-1.109
14	0.299
15	-0.015

```

16      0.003
17      0.004
18     -0.002
19      0.005
20      0.002
21      0.009
22      0.000
23      0.006
24     -0.006
25      0.001

```

3. ion library in the specific format required by Metab. The ion library is a data frame containing the name and the reference ion mass fragment to quantify each metabolite present in the mass spectral library used by AMDIS when generating the batch report. To facilitate the process, MetReport accepts the .msl file used by AMDIS. An AMDIS library is stored in two files, a file with extension .CID and a file with extension .msl. Metab requires only the .msl file.

Below you can see examples of an ion library converted from an AMDIS library:

```

> data(exampleMSLfile)
> print(head(exampleMSLfile, 29))

```

```

1                                     V1
2                                     NAME:Ethanol
3                                     CASNO:130513~1-N1002
4                                     RI:
5                                     RW:
6                                     RT:6.644
7                                     RSN:31
8                                     COMMENT: 6.6438 min 130513_REF_SOL2_2_100_1
9 SOURCE:C:\\Program Files (x86)\\NISTMS\\AMDIS32\\LIB\\ref_sol2.msl
10                                     NUM PEAKS: 22
11          ( 13   4) ( 14  13) ( 15  29) ( 19   9) ( 24   3)
12          ( 25  14) ( 26  71) ( 27 176) ( 28  54) ( 29 249)
13          ( 30  60) ( 31 1000) ( 32  12) ( 33   2) ( 40   6)
14          ( 41  23) ( 42  79) ( 43 198) ( 44  36) ( 45 777)
15          ( 46 343) ( 47  11)
16                                     NAME:Acetone
17                                     CASNO:130513~1-N1004
18                                     RI:
19                                     RW:
20                                     RT:7.373
21                                     RSN:43
22                                     COMMENT: 7.3726 min 130513_REF_SOL2_2_100_1
23 SOURCE:C:\\Program Files (x86)\\NISTMS\\AMDIS32\\LIB\\ref_sol2.msl
24                                     NUM PEAKS: 30
25          ( 12   1) ( 13   1) ( 14   8) ( 15  28) ( 16   1)
26          ( 24   1) ( 25   5) ( 26  22) ( 27  32) ( 28   7)
27          ( 29  16) ( 30   1) ( 31   2) ( 36   5) ( 37  19)

```

```

27          ( 38  24) ( 39  44) ( 40  10) ( 41  23) ( 42  76)
28          ( 43 1000) ( 44  26) ( 45   3) ( 52   1) ( 53   4)
29          ( 55   3) ( 57   7) ( 58 262) ( 59  10) ( 60   1)

```

```
> testLib <- buildLib(exampleMSLfile, save = FALSE, verbose = FALSE)
```

```

-----
Names      RT      Ion
-----
Zylene1    20.39  *91*
Zylene2    20.7   *91*
-----

```

```
> print(testLib)
```

	Name	RT	ref_ion1	ref_ion2	ref_ion3	ref_ion4	ion2to1	ion3to1
1	Ethanol	6.644	31	45	46	29	0.777	0.343
2	Acetone	7.373	43	58	42	39	0.262	0.076
3	Isopropyl alcohol	7.582	45	41	27	39	0.107	0.090
4	Acetonitril	7.905	41	40	39	38	0.546	0.223
5	Ethyl acetate	10.593	43	45	70	61	0.137	0.116
6	1-butanol	13.381	56	41	43	31	0.720	0.543
7	2-pentanone	13.959	43	86	41	71	0.249	0.127
8	Pyridine	16.426	79	52	51	50	0.564	0.275
9	Zylene1	20.395	91	106	77	51	0.327	0.080
10	Zylene2	20.697	91	106	105	77	0.533	0.223
11	Zylene3	21.803	91	106	105	77	0.488	0.189
12	Benzaldehyde	25.712	106	105	77	51	0.990	0.935
13	Indole	38.634	117	90	89	63	0.414	0.313
	ion4to1							
1		0.249						
2		0.044						
3		0.072						
4		0.137						
5		0.105						
6		0.346						
7		0.109						
8		0.205						
9		0.077						
10		0.115						
11		0.109						
12		0.404						
13		0.103						

When all the requirements described above are ready and available, MetReport can be applied. If an essential argument is missing, a dialog box will pop up allowing the user to point and click on the missing file. Here is an example of MetReport applied to a single file and recalculating metabolite abundances. We use a test file distributed



with the package, unzip it and store the file name in the `testfile` variable. This file will also be used in the subsequent examples.

```
> ##### Load exampleAMDISReport #####
> data(exampleAMDISReport)
> ##### Load exampleIonLib #####
> data(exampleIonLib)
> ##### Analyse a single file #####
> testfile <- unzip(system.file("extdata/130513_REF_SOL2_2_50_50_1.CDF.zip", package = "Metab"))
> test <- MetReport(inputData = testfile,
+                   singleFile = TRUE, AmdisReport = exampleAMDISReport,
+                   ionLib = exampleIonLib, abundance = "recalculate",
+                   TimeWindow = 0.5, save = FALSE)
> ##### Show results #####
> print(test)
```

	Name	130513_REF_SOL2_2_50_50_1
1	Replicates	A
2	1-butanol	34874681
3	2-pentanone	195503137
4	Acetone	140289057
5	Acetonitril	44105593
6	Benzaldehyde	143276433
7	Ethanol	11756469
8	Ethyl acetate	201696289
9	Indole	70889473
10	Isopropyl alcohol	38373933
11	Pyridine	369485217
12	Zylene1	73424897
13	Zylene2	25606145

Note that the first line of the resulting data.frame is used to represent sample meta-data (for example replicates).

The argument "abundance" defines the way metabolite abundances will be reported. If abundance = "recalculated", the abundances of metabolites will be corrected by fixing a single mass fragment as reference. If abundance = "Area", the area associated with each compound will be extracted from the AMDIS report indicated by "AmdisReport". And finally, if abundance = "Base.Peak", the Base.Peak associated with each compound will be extracted from the AMDIS report. Below you can find an example when extracting the area:

```
> ##### Load exampleAMDISReport #####
> data(exampleAMDISReport)
> ##### Analyse a single file #####
> test <- MetReport(inputData = testfile,
+                   singleFile = TRUE, AmdisReport = exampleAMDISReport,
+                   abundance = "Area", TimeWindow = 0.5, save = FALSE)
> ##### Show results #####
> print(test)
```

	Name	130513_REF_SOL2_2_50_50_1
1	Replicates	A
2	1-butanol	2801759237
3	2-pentanone	6387468112
4	Acetone	4725912300
5	Acetonitril	1186617973
6	Benzaldehyde	7845543202
7	Ethanol	701423866
8	Ethyl acetate	9249749212
9	Indole	1780437467
10	Isopropyl alcohol	174139435
11	Pyridine	18048017764
12	Zylene1	3222637797
13	Zylene2	932247262

Note that in this case the ion library is not required, as the abundances of metabolites will be extracted directly from the AMDIS report.

When applied to a batch of GC-MS files, **MetReport** can be used to automatically detect the name of experimental conditions under study. For this, GC-MS files in CDF format must be organised in subfolders according to their experimental condition, as follows:

```

-----
Experiment1
---Condition1
-----Sample1.cdf
-----Sample2.cdf
-----Sample3.cdf
---Condition2
-----Sample1.cdf
-----Sample2.cdf
-----Sample3.cdf
---Condition3
-----Sample1.cdf
-----Sample2.cdf
-----Sample3.cdf
-----

```

The folder Experiment1 is the main folder containing one subfolder for each experimental condition. Each subfolder contains the CDF files associated with this specific experimental condition. Alternatively, all the CDF files can be placed in a single folder and MetReport will analyse every sample as belonging to the same experimental condition.

Below you can see an example of MetReport applied to a batch of samples:

```

> MetReport(
+   dataFolder = "/Users/ThePathToTheMainFolder/",
+   AmdisReport = "/Users/MyAMDISreport.TXT",

```

```

+       ionLib = "/Users/MyIonLibrary.csv",
+       save = TRUE,
+       output = "metabData",
+       TimeWindow = 2.5,
+       Remove = c("Ethanol", "Pyridine"))

```

As a result, MetReport generates a data frame containing the metabolites identified in the first column and their abundances in the different samples analysed in the following columns. See below an example:

```

> data(exampleMetReport)
> print(exampleMetReport)

```

	Name	130513_REF_SOL2_2_100_1	130513_REF_SOL2_2_100_2
1	Replicates	100ul	100ul
2	1-butanol	169279488	176668672
3	2-pentanone	358105088	412483584
4	Acetone	247545856	285147136
5	Acetonitril	89587712	96366592
6	Benzaldehyde	534659072	580452352
7	Ethanol	23259136	24012800
8	Ethyl acetate	342671360	422952960
9	Indole	157777920	163397632
10	Isopropyl alcohol	82120704	77467648
11	Pyridine	731381760	861339648
12	Zylene1	29983744	53530624
13	Zylene2	86278144	138510336
14	Zylene3	<NA>	<NA>
	130513_REF_SOL2_2_100_3	130513_REF_SOL2_2_100_4	130513_REF_SOL2_2_100_5
1	100ul	100ul	100ul
2	181108736	192888832	208617472
3	388415488	363429888	456081408
4	271532032	307740672	308297728
5	92360704	108470272	107765760
6	589234176	654049280	649789440
7	22847488	25887744	26106880
8	427343872	501448704	494567424
9	163446784	167837696	186777600
10	80994304	93126656	95952896
11	843120640	916586496	889716736
12	41664512	57958400	55349248
13	118910976	152977408	146456576
14	20529152	<NA>	49307648
	130513_REF_SOL2_2_50_50_1	130513_REF_SOL2_2_50_50_2	
1	50ul	50ul	
2	34881536	51818496	
3	195510272	231931904	
4	140296192	183975936	
5	44122112	60628992	
6	143278080	160907264	
7	11761664	13939712	

8	201703424	242745344
9	70889472	80273408
10	38379520	53235712
11	369508352	415711232
12	73424896	27378688
13	25606144	79704064
14	<NA>	<NA>
	130513_REF_SOL2_2_50_50_3	130513_REF_SOL2_2_50_50_4
1	50u1	50u1
2	76873728	66592768
3	291504128	227393536
4	211861504	194805760
5	65150976	65810432
6	207470592	162250752
7	15432704	15524864
8	316309504	252280832
9	86499328	84062208
10	61800448	65531904
11	457539584	438960128
12	35684352	20614144
13	100077568	57167872
14	<NA>	<NA>
	130513_REF_SOL2_2_50_50_5	
1	50u1	
2	69951488	
3	258048000	
4	207060992	
5	64122880	
6	165134336	
7	14892032	
8	272318464	
9	83632128	
10	60612608	
11	427327488	
12	25833472	
13	76689408	
14	45645824	

## 4 What if I have the AMDIS report but not the CDF files?

The function `MetReportNames` is used to process an AMDIS report by choosing a single compound per RT and extracting the AREA or the BASE.PEAK reported by AMDIS for each compound. `MetReportNames` only requires the names of the files or samples to be extracted from the AMDIS report and the AMDIS report in batch mode. It is applied as follows:

```
> ### Load the example of AMDIS report #####
> data(exampleAMDISReport)
> ### Extract the Area of compounds in samples
> # 130513_REF_SOL2_2_100_1 and 130513_REF_SOL2_2_100_2 ##
> test <- MetReportNames(
+   c("130513_REF_SOL2_2_100_1", "130513_REF_SOL2_2_100_2"),
+   exampleAMDISReport,
+   save = FALSE,
+   TimeWindow = 0.5,
+   base.peak = FALSE)
> print(test)
```

	Name	130513_REF_SOL2_2_100_1	130513_REF_SOL2_2_100_2
1	1-butanol	12764249729	13106120736
2	2-pentanone	11073801529	14219281161
3	Acetone	7450198663	7664120070
4	Acetonitril	2415421513	2619137294
5	Benzaldehyde	24017979717	27158354783
6	Ethanol	1298487467	1310635238
7	Ethyl acetate	14504720058	18031280625
8	Indole	4150927824	3048110943
9	Isopropyl alcohol	1863002758	509048091
10	Pyridine	13248571706	54766105482
11	Zylene1	977285068	1873141055
12	Zylene2	3484655661	4098512121

## 5 Normalisations and further analysis: `removeFalsePositives`, `normalizeByInternalStandard`, `normalizeByBiomass`, `Htest`

Normalisations and statistical analysis are commonly applied to metabolomics data. Therefore, `Metab` contains few functions to facilitate these processes. Every function described in this section uses an input data in the same format as the results generated by the previously described functions. In the first row, it contains the names of the experimental conditions associated with each sample. Removing metabolites considered false positives: In some metabolomics experiments it is ideal to consider only those metabolites detected in a minimum proportion of the samples analysed for a specific experimental condition. For example, if an experimental condition contains 6 sample, or replicates, one may consider that metabolites present in only 2 samples are potential miss identifications or contaminations. Thus, they must be removed before further analysis. The function `removeFalsePositives` uses a data set generated by `MetReport`, `MetReportArea` or `MetReportBasePeak` to automatically remove these compounds. `removeFalsePositives` only requires the data frame to be processed, which can be a vector in R or a CSV file, and the percentage of samples to be used as cut off. For example:

```
> ### Load the inputData ###
> data(exampleMetReport)
> ### Normalize ####
> normalizedData <- removeFalsePositives(exampleMetReport, truePercentage = 40, save = FALSE)
> #####
> # The abundances of compound Zylene3 will be replaced by NA in samples from experimental
> #condition 50ul, as it is present in less than 40 per cent of the samples from this
> #experimental condition.
> ### Show results ####
> print(normalizedData)
```

	Name	130513_REF_SOL2_2_100_1	130513_REF_SOL2_2_100_2
1	Replicates	100ul	100ul
2	Isopropyl alcohol	82120704	77467648
3	Pyridine	731381760	861339648
4	Zylene1	29983744	53530624
5	Zylene2	86278144	138510336
6	Zylene3	<NA>	<NA>
7	1-butanol	169279488	176668672
8	2-pentanone	358105088	412483584
9	Acetone	247545856	285147136
10	Acetonitril	89587712	96366592
11	Benzaldehyde	534659072	580452352
12	Ethanol	23259136	24012800
13	Ethyl acetate	342671360	422952960
14	Indole	157777920	163397632
	130513_REF_SOL2_2_100_3	130513_REF_SOL2_2_100_4	130513_REF_SOL2_2_100_5
1	100ul	100ul	100ul

2	80994304	93126656	95952896
3	843120640	916586496	889716736
4	41664512	57958400	55349248
5	118910976	152977408	146456576
6	20529152	<NA>	49307648
7	181108736	192888832	208617472
8	388415488	363429888	456081408
9	271532032	307740672	308297728
10	92360704	108470272	107765760
11	589234176	654049280	649789440
12	22847488	25887744	26106880
13	427343872	501448704	494567424
14	163446784	167837696	186777600
	130513_REF_SOL2_2_50_50_1	130513_REF_SOL2_2_50_50_2	
1	50u1	50u1	
2	38379520	53235712	
3	369508352	415711232	
4	73424896	27378688	
5	25606144	79704064	
6	<NA>	<NA>	
7	34881536	51818496	
8	195510272	231931904	
9	140296192	183975936	
10	44122112	60628992	
11	143278080	160907264	
12	11761664	13939712	
13	201703424	242745344	
14	70889472	80273408	
	130513_REF_SOL2_2_50_50_3	130513_REF_SOL2_2_50_50_4	
1	50u1	50u1	
2	61800448	65531904	
3	457539584	438960128	
4	35684352	20614144	
5	100077568	57167872	
6	<NA>	<NA>	
7	76873728	66592768	
8	291504128	227393536	
9	211861504	194805760	
10	65150976	65810432	
11	207470592	162250752	
12	15432704	15524864	
13	316309504	252280832	
14	86499328	84062208	
	130513_REF_SOL2_2_50_50_5		
1	50u1		
2	60612608		
3	427327488		
4	25833472		
5	76689408		
6	<NA>		
7	69951488		

```

8           258048000
9           207060992
10          64122880
11          165134336
12          14892032
13          272318464
14          83632128

```

Normalising by internal standard: The use of internal standards is a common practice in metabolomics. In order to normalise a data set by a specific internal standard, the abundance or intensity of each metabolite must be divided by the abundance of the internal standard at the sample where each metabolite was detected. The function `normalizeByInternalStandard` normalises a data set generated by `Metab` functions according to an internal standard defined by the user. For example:

```

> ### Load the inputData ###
> data(exampleMetReport)
> ### Normalize #####
> normalizedData <- normalizeByInternalStandard(
+   exampleMetReport,
+   internalStandard = "Acetone",
+   save = FALSE)
> ### Show results #####
> print(normalizedData)

```

	Name	130513_REF_S0L2_2_100_1	130513_REF_S0L2_2_100_2
1	Replicates	100ul	100ul
2	1-butanol	0.683830829307036	0.619570213743967
3	2-pentanone	1.44662121914091	1.44656400827396
4	Acetone	1	1
5	Acetonitril	0.361903501224436	0.337953918639393
6	Benzaldehyde	2.15983850685022	2.03562399448403
7	Ethanol	0.0939588986696671	0.08421196276718
8	Ethyl acetate	1.38427427361175	1.48327970581476
9	Indole	0.63736845588722	0.573029188692255
10	Isopropyl alcohol	0.331739360645973	0.271676051482418
11	Pyridine	2.95453041233702	3.02068490002298
12	Zylene1	0.121123998941029	0.187729832222478
13	Zylene2	0.348533986365742	0.485750402206389
14	Zylene3	<NA>	<NA>
	130513_REF_S0L2_2_100_3	130513_REF_S0L2_2_100_4	130513_REF_S0L2_2_100_5
1	100ul	100ul	100ul
2	0.666988475230797	0.626790182611936	0.676675346760908
3	1.43045918059494	1.18096150774637	1.47935377584099
4	1	1	1
5	0.340146624027032	0.352472980886972	0.349550937981612
6	2.17003560007241	2.12532609274344	2.10766859754477
7	0.0841428830024739	0.084121945376138	0.0846807408194718
8	1.57382489591504	1.62945216419102	1.60418770260934
9	0.601942919205937	0.545386785923441	0.605835149067333



10	0.298286369396006	0.302614065910664	0.31123452197481
11	3.10505038315332	2.97843794920939	2.88590104692565
12	0.153442345984433	0.188335196720439	0.179531806345326
13	0.437926144934532	0.497098440078795	0.475049157676569
14	0.0756048995353889	<NA>	0.159935165010363
	130513_REF_SOL2_2_50_50_1	130513_REF_SOL2_2_50_50_2	
1	50u1	50u1	
2	0.248627817353731	0.281659096981031	
3	1.39355366110008	1.26066435123341	
4	1	1	
5	0.314492584374635	0.329548490515629	
6	1.02125423332944	0.874610383827589	
7	0.0838345206119351	0.0757692136432452	
8	1.4376970687843	1.31944073381423	
9	0.505284362956908	0.436325585537448	
10	0.27356066799019	0.289362365304123	
11	2.63377321032348	2.25959568973194	
12	0.523356300362023	0.148816680024935	
13	0.182514889641481	0.433230919939443	
14	<NA>	<NA>	
	130513_REF_SOL2_2_50_50_3	130513_REF_SOL2_2_50_50_4	
1	50u1	50u1	
2	0.362848967597247	0.341841883936081	
3	1.37591833578223	1.167283431455	
4	1	1	
5	0.307516820044853	0.33782590412111	
6	0.979274611398964	0.832884777123633	
7	0.0728433609156291	0.079694070647603	
8	1.49300131467017	1.29503784693019	
9	0.408282422086459	0.431518082422204	
10	0.29170211120563	0.336396131202691	
11	2.1596164256438	2.25332211942809	
12	0.168432449153198	0.105818965517241	
13	0.472372592993581	0.293460891505467	
14	<NA>	<NA>	
	130513_REF_SOL2_2_50_50_5		
1	50u1		
2	0.33783035290394		
3	1.24624149390726		
4	1		
5	0.309681120430448		
6	0.797515429656591		
7	0.0719209922456085		
8	1.31516062668144		
9	0.40390093369204		
10	0.292728279791106		
11	2.06377591391043		
12	0.124762620667827		
13	0.37037110302263		
14	0.220446273144485		

Normalising by biomass: Normalisation by biomass (e.g. number of cells or O.D.) is also a common practice in metabolomics. In order to normalise a data set by the biomass associated with each sample, the abundance or intensity of each metabolite must be divided by the biomass associated with the sample where each metabolite was detected. The function `normalizeByBiomass` normalises a data set generated by `Metab` functions according to a list of biomasses defined by the user. For this, the user must provide a data frame or a CSV file containing the name of each sample in the first column and their respective biomass in the second column. See below an example of the data frame specifying biomasses:

```
> data(exampleBiomass)
> print(exampleBiomass)
```

	Sample	Biomass
1	130513_REF_SOL2_2_100_1	0.5
2	130513_REF_SOL2_2_100_2	0.5
3	130513_REF_SOL2_2_100_3	0.5
4	130513_REF_SOL2_2_100_4	0.5
5	130513_REF_SOL2_2_100_5	0.5
6	130513_REF_SOL2_2_50_50_1	0.5
7	130513_REF_SOL2_2_50_50_2	0.5
8	130513_REF_SOL2_2_50_50_3	0.5
9	130513_REF_SOL2_2_50_50_4	0.5
10	130513_REF_SOL2_2_50_50_5	0.5

For example:

```
> ### Load the inputData ###
> data(exampleMetReport)
> ### Load the list of biomasses ###
> data(exampleBiomass)
> ### Normalize ####
> normalizedData <- normalizeByBiomass(
+   exampleMetReport,
+   biomass = exampleBiomass,
+   save = FALSE)
> ### Show results ###
> print(normalizedData)
```

	Name	130513_REF_SOL2_2_100_1	130513_REF_SOL2_2_100_2
1	Replicates	100ul	100ul
2	1-butanol	338558976	353337344
3	2-pentanone	716210176	824967168
4	Acetone	495091712	570294272
5	Acetonitril	179175424	192733184
6	Benzaldehyde	1069318144	1160904704
7	Ethanol	46518272	48025600
8	Ethyl acetate	685342720	845905920
9	Indole	315555840	326795264

10	Isopropyl alcohol	164241408	154935296
11	Pyridine	1462763520	1722679296
12	Zylene1	59967488	107061248
13	Zylene2	172556288	277020672
14	Zylene3	<NA>	<NA>
	130513_REF_SOL2_2_100_3	130513_REF_SOL2_2_100_4	130513_REF_SOL2_2_100_5
1	100ul	100ul	100ul
2	362217472	385777664	417234944
3	776830976	726859776	912162816
4	543064064	615481344	616595456
5	184721408	216940544	215531520
6	1178468352	1308098560	1299578880
7	45694976	51775488	52213760
8	854687744	1002897408	989134848
9	326893568	335675392	373555200
10	161988608	186253312	191905792
11	1686241280	1833172992	1779433472
12	83329024	115916800	110698496
13	237821952	305954816	292913152
14	41058304	<NA>	98615296
	130513_REF_SOL2_2_50_50_1	130513_REF_SOL2_2_50_50_2	
1	50ul	50ul	
2	69763072	103636992	
3	391020544	463863808	
4	280592384	367951872	
5	88244224	121257984	
6	286556160	321814528	
7	23523328	27879424	
8	403406848	485490688	
9	141778944	160546816	
10	76759040	106471424	
11	739016704	831422464	
12	146849792	54757376	
13	51212288	159408128	
14	<NA>	<NA>	
	130513_REF_SOL2_2_50_50_3	130513_REF_SOL2_2_50_50_4	
1	50ul	50ul	
2	153747456	133185536	
3	583008256	454787072	
4	423723008	389611520	
5	130301952	131620864	
6	414941184	324501504	
7	30865408	31049728	
8	632619008	504561664	
9	172998656	168124416	
10	123600896	131063808	
11	915079168	877920256	
12	71368704	41228288	
13	200155136	114335744	
14	<NA>	<NA>	
	130513_REF_SOL2_2_50_50_5		

```

1           50ul
2           139902976
3           516096000
4           414121984
5           128245760
6           330268672
7           29784064
8           544636928
9           167264256
10          121225216
11          854654976
12          51666944
13          153378816
14          91291648

```

Performing ANOVA or t-Test: The statistical tests ANOVA and t-Test are widely applied in metabolomics studies. The function `Htest` can be used to quickly calculate the p-values associated with each metabolite when performing ANOVA or t-Test. For example:

```

> ### Load the inputData ###
>     data(exampleMetReport)
> ### Perform t-test ###
>     tTestResults <- htest(
+         exampleMetReport,
+         signif.level = 0.05,
+         StatTest = "T",
+         save = FALSE
+     )
> ### Show results ###
>     print(tTestResults)

```

	Name	130513_REF_SOL2_2_100_1	130513_REF_SOL2_2_100_2
1	Replicates	100ul	100ul
4	Zylene2	86278144	138510336
5	Zylene3	<NA>	<NA>
6	1-butanol	169279488	176668672
13	Indole	157777920	163397632
	130513_REF_SOL2_2_100_3	130513_REF_SOL2_2_100_4	130513_REF_SOL2_2_100_5
1		100ul	100ul
4		118910976	152977408
5		20529152	<NA>
6		181108736	192888832
13		163446784	167837696
	130513_REF_SOL2_2_50_50_1	130513_REF_SOL2_2_50_50_2	
1		50ul	50ul
4		25606144	79704064
5		<NA>	<NA>
6		34881536	51818496

```

13          70889472          80273408
130513_REF_SOL2_2_50_50_3 130513_REF_SOL2_2_50_50_4
1          50ul          50ul
4          100077568          57167872
5          <NA>          <NA>
6          76873728          66592768
13         86499328          84062208
130513_REF_SOL2_2_50_50_5          pvalues
1          50ul          bonferroni
4          76689408   -1.54383074750421
5          45645824          0
6          69951488  0.00184334081091174
13         83632128          0

```

```

> ### Perform ANOVA ###
> AnovaResults <- htest(
+   exampleMetReport,
+   signif.level = 0.05,
+   StatTest = "Anova",
+   save = FALSE
+ )
> ### Show results ###
> print(AnovaResults)

```

```

          Name 130513_REF_SOL2_2_100_1 130513_REF_SOL2_2_100_2
1   Replicates          100ul          100ul
2   Pyridine          731381760          861339648
5   Zylene3          <NA>          <NA>
6   1-butanol          169279488          176668672
7   2-pentanone          358105088          412483584
8   Acetone          247545856          285147136
10 Benzaldehyde          534659072          580452352
11 Ethanol          23259136          24012800
13 Indole          157777920          163397632
130513_REF_SOL2_2_100_3 130513_REF_SOL2_2_100_4 130513_REF_SOL2_2_100_5
1          100ul          100ul          100ul
2          843120640          916586496          889716736
5          20529152          <NA>          49307648
6          181108736          192888832          208617472
7          388415488          363429888          456081408
8          271532032          307740672          308297728
10         589234176          654049280          649789440
11         22847488          25887744          26106880
13         163446784          167837696          186777600
130513_REF_SOL2_2_50_50_1 130513_REF_SOL2_2_50_50_2
1          50ul          50ul
2          369508352          415711232
5          <NA>          <NA>
6          34881536          51818496
7          195510272          231931904
8          140296192          183975936
10         143278080          160907264

```

11	11761664	13939712
13	70889472	80273408
	130513_REF_SOL2_2_50_50_3	130513_REF_SOL2_2_50_50_4
1	50ul	50ul
2	457539584	438960128
5	<NA>	<NA>
6	76873728	66592768
7	291504128	227393536
8	211861504	194805760
10	207470592	162250752
11	15432704	15524864
13	86499328	84062208
	130513_REF_SOL2_2_50_50_5	pvalues
1	50ul	bonferroni
2	427327488	0.00158164810074067
5	45645824	0
6	69951488	0.0014872976434156
7	258048000	0.0035539783363238
8	207060992	0.0382389971487311
10	165134336	1.87190063707585e-05
11	14892032	0.000793716738135416
13	83632128	0.000312485332764501

## Session information

```
> print(sessionInfo(), locale = FALSE)
```

```
R version 4.2.0 RC (2022-04-21 r82226)  
Platform: x86_64-pc-linux-gnu (64-bit)  
Running under: Ubuntu 20.04.4 LTS
```

```
Matrix products: default
```

```
BLAS: /home/biocbuild/bbs-3.16-bioc/R/lib/libRblas.so
```

```
LAPACK: /home/biocbuild/bbs-3.16-bioc/R/lib/libRlapack.so
```

```
attached base packages:
```

```
[1] stats4      stats      graphics  grDevices  utils      datasets  methods  
[8] base
```

```
other attached packages:
```

```
[1] Metab_1.31.0          svDialogs_1.0.3      xcms_3.19.0  
[4] MSnbase_2.23.0       ProtGenerics_1.29.0 S4Vectors_0.35.0  
[7] mzR_2.31.0           Rcpp_1.0.8.3         Biobase_2.57.0  
[10] BiocGenerics_0.43.0 BiocParallel_1.31.0
```

```
loaded via a namespace (and not attached):
```

```
[1] MatrixGenerics_1.9.0      vsn_3.65.0  
[3] foreach_1.5.2            svGUI_1.0.1  
[5] assertthat_0.2.1        BiocManager_1.30.17  
[7] affy_1.75.0              pander_0.6.5  
[9] GenomeInfoDbData_1.2.8   robustbase_0.95-0  
[11] impute_1.71.0            pillar_1.7.0  
[13] lattice_0.20-45          glue_1.6.2  
[15] limma_3.53.0             digest_0.6.29  
[17] GenomicRanges_1.49.0     RColorBrewer_1.1-3  
[19] XVector_0.37.0           colorspace_2.0-3  
[21] preprocessCore_1.59.0    Matrix_1.4-1  
[23] plyr_1.8.7               MALDIquant_1.21  
[25] XML_3.99-0.9             pkgconfig_2.0.3  
[27] zlibbioc_1.43.0          purrr_0.3.4  
[29] scales_1.2.0             RANN_2.6.1  
[31] affyio_1.67.0           tibble_3.1.6  
[33] generics_0.1.2          IRanges_2.31.0  
[35] ggplot2_3.3.5           ellipsis_0.3.2  
[37] SummarizedExperiment_1.27.0 cli_3.3.0  
[39] MassSpecWavelet_1.63.0   magrittr_2.0.3
```

[41]	crayon_1.5.1	ncdf4_1.19
[43]	fansi_1.0.3	doParallel_1.0.17
[45]	MASS_7.3-57	MsFeatures_1.5.0
[47]	tools_4.2.0	lifecycle_1.0.1
[49]	matrixStats_0.62.0	munsell_0.5.0
[51]	cluster_2.1.3	DelayedArray_0.23.0
[53]	pcaMethods_1.89.0	compiler_4.2.0
[55]	GenomeInfoDb_1.33.0	mzID_1.35.0
[57]	rlang_1.0.2	grid_4.2.0
[59]	RCurl_1.98-1.6	iterators_1.0.14
[61]	rstudioapi_0.13	MsCoreUtils_1.9.0
[63]	bitops_1.0-7	gtable_0.3.0
[65]	codetools_0.2-18	DBI_1.1.2
[67]	R6_2.5.1	dplyr_1.0.8
[69]	utf8_1.2.2	clue_0.3-60
[71]	parallel_4.2.0	vctrs_0.4.1
[73]	DEoptimR_1.0-11	tidyselect_1.1.2