

# Rlims : utilities for processing laboratory analyses from NIOO-CEME

**Karline Soetaert**  
NIOO-CEME  
The Netherlands

**Pieter Provoost**  
NIOO-CEME  
The Netherlands

**Pieter van Rijswijk**  
NIOO-CEME  
The Netherlands

---

## Abstract

Rpackage **Rlims** contains utilities to convert the files resulting from lab analyses performed on the GC-FID and GC-c-IRMS in NIOO-CEME.

These R-functions are meant as an alternative to the EXCEL spreadsheet with VBA macro originally developed by Pieter van Rijswijk.

In addition, it contains documents with the lab procedures.

*Keywords:* fatty acids, chromatogram, NIOO-CEME, R .

---

## 1. Introduction

R-package **Rlims** is a tool, running in R, to convert files from the analysis lab at NIOO-CEME in a more suitable format.

It contains two main functions:

- `fa.fid`, a utility function to convert chromatogram data from the GC-FID carlo erba HRGC mega 2 GC.
- `fa.irms`, a utility function to convert chromatogram data from the gas-chromatograph combustion interface isotope-ratio mass spectrometer (GC-c-IRMS)

In addition, there are three documents, so-called package “vignettes”. To open them from R: you can type

```
vignette("PLFA")  
vignette("HAA")  
vignette("GC-IRMS")
```

## 2. GC-FID data

### 2.1. Preparing GC-FID Input files

The GC-FID (Gas Chromatograph - Flame Ionization Detector) instrument produces a binary

	A	B	C	D	E	F	G	H	I	J
1										
2										
3										
4										
5										
6										
7										
8										
9										
10										
11										
12										
13										
14										
15										
16										
17										
18										
19										
20										
21										
22										
23										
24										
25										
26										
27										
28										
29										

Figure 1: Excel input sheet as produced by the CEME lab

file, which is processed with dedicated software and produces an EXCEL spreadsheet (see fig. 1).

To be used with the `fa.fid` R-function this spreadsheet has to be converted as follows:

- Remove the header with the file's metadata.
- Remove the line below the columns description with units (See fig. 2)
- Save the file as a *csv* file (comma delimited file)

This should produce an ascii file as in figure (2).

This file contains several columns, typically called: "Peak Number", "Retention Time", "Peak Height", "Area", "Component Name", "Original Conc", "Solution Conc", "Width at 50

Of these, only the columns with peak numbers, retention time, area and component name will be retained.

## 2.2. Analytical Column Input Data for GC-FID

In addition to the input data, the analytical column used also needs to be given.

The following analytical columns for GC-FID are included in the package:

- `fid.bpx70`, the BPX70 (GC-04 PTV); date June 2008
- `fid.zb5`, the ZB-5 column (GC-05 SSL LV); date March 2008
- `fid.apolar`, the "a-polar" data
- `fid.polar1`, the "polar bpx70" data from Eric Boschker
- `fid.polar2`, the "polar bpx70" data from Marco Houtekamer; date nov 2004

	A	B	C	D	E	F	G	H	I
1	Peak Num	Retention	Peak Heig	Area	Componen	Width at 5	Original Cc	Solution	Conc
2	1	17.043	7119	345753	C12:0	4.4	0.312	0.312	
3	2	22.263	2857	147175	i-C14:0	4.7	0.133	0.133	
4	3	23.96	7209	352463	C14:0	4.4	0.318	0.318	
5	4	24.238	981	57048		5.3	0	0	
6	5	25.827	10662	559331	i-C15:0/C1	4.7	0.504	0.504	
7	6	26.455	15919	846496	ai-C15:0	4.8	0.763	0.763	
8	7	27.565	2404	125645	C15:0	4.7	0.113	0.113	
9	8	29.423	2595	141177	i-C16:0/C1	4.9	0.127	0.127	
10	9	31.152	30631	1567360	C16:0	4.8	1.414	1.414	
11	10	31.622	1824	123286		5.1	0	0	
12	11	31.945	1323	65302		5.1	0	0	
13	12	32.088	5414	275143	C16:1w7t	5.4	0.248	0.248	
14	13	32.197	7410	449790		5.9	0	0	
15	14	32.44	51223	2671836	C16:1w7c/	4.8	2.41	2.41	
16	15	32.923	6366	414566	i-C17:0	6.1	0.374	0.374	
17	16	33.602	2175	120232	ai-C17:0	4.9	0.108	0.108	
18	17	34.04	3209	181624		4.9	0	0	
19	18	34.645	1957	104838	C17:0	5.1	0.095	0.095	
20	19	34.928	1415	73501	16:2w4	4.7	0.066	0.066	

Figure 2: Converted sheet; the header has been removed, as well as the line with units below the column header; this should be exported as a csv file.

It is also possible to use as input a “csv” file, which should have at least two data-columns:

- a column named “ecl” and containing the equivalent chain lengths
- a column called “name”, containing the equivalent chain with the name of the FA.

This reference data set can be read with R-function `read.csv`.

## 2.3. Other Input

You need to input also other information:

- Standards whose ecl (equivalent chain length) and retention times are known. They are used to calculate the ecl of the other peaks. Typically, there are three standards: “C12:0”, “C16:0”, and “C19:0”. If you do not know the ecls of these standards, they can be derived from the input data.
- The quantity of the standard, which has been added during the analytical procedure. For GC-FID, this quantity should be expressed in  $\mu\text{g}$  PLFA. Usually one adds “C19:0” in a quantity equal to  $\approx 2.01\mu\text{g}$  PLFA. See section 6.2. The peak of this standard will be used to scale the other peaks such as to reproduce the quantity of the other FAs. It is assumed that this fa is not present in natural samples.
- The quantity of the sample added.
- The recovery. During the analytical procedure, one adds a certain volume of chloroform, and one weights the chloroform recovered. These two measurements are also required to estimate the recovery of the analytical procedure.

### 3. GC-c-IRMS data

#### 3.1. Preparing Input for GC-c-IRMS data analysis

The GC-c-IRMS (Gas Chromatograph - combustion interface isotope-ratio mass spectrometer) instrument also produces a binary file, which is processed with dedicated software and produces an EXCEL spreadsheet.

To be used with the `fa.irms` R-function this spreadsheet has to be converted similarly as for `fa.fid` data; this means that it needs to be presented as a *csv* file with the initial part of the file removed.

From this file, the columns with peak numbers, retention time, area and component name, and stable isotopic composition will be retained.

#### 3.2. Analytical Column Input Data for GC-irms

The same analytical columns as for GC-FID can be used (????CHECK????)

#### 3.3. Other Input

The same additional input as for GC-FID is required (see also above):

- ecl standards.
- The quantity of the standard added. **\*\*NOTE\*\*** for GC-c-IRMS, this quantity should be expressed in  $\mu\text{g}$  Carbon, rather than in  $\mu\text{g}$  PLFA. Typically, this value is  $\approx 1.54$ . See section 6.2.
- The quantity of the sample added.
- The volume chloroform added, and weight recovered.

## 4. Analysing the Chromatogram Data

There are two ways in which the chromatogram data may be analysed:

- In 6 steps, using several utility functions.
- in one step, using functions `fa.fid`, or `fa.irms`

#### 4.1. Step-wise analysis

Several R-functions perform specific calculations on a chromatogram. They are defined as:

```
read.fid(filename)
read.irms(filename)
```

```
get.standard(input, standard.ecl = list(time = c(12, 16, 19),
```

```

                                ecl = c(12, 16, 19),
                                name = c("C12:0", "C16:0", "C19:0")),
                                ecl.interval = 0.02, verbose = TRUE)

add.ecl      (input, standard.ecl)

clean.peaks (input, peak.interval = 0.02, min.area = 0, verbose = TRUE)

get.peaks    (input, fa.ref, peak.interval = 0.02, verbose = TRUE)

get.conc.fid (input, standard.qty, sample.qty,
              chlor.volume, chlor.weight, standard.name = "C19:0"))
get.conc.irms (input, standard.qty, sample.qty,
              chlor.volume, chlor.weight, standard.name = "C19:0",
              d13C.meth = -45.6))

```

where:

- `filename` contains the name of a *csv* input file
- `input` contains a `data.frame` with a previously loaded input file
- `fa.ref` is the name of the reference column used. The default is the `bpx70` column
- `standard.ecl$time`, is the retention times of the standard
- `standard.ecl$ecl`, is the standard equivalent chain length
- `standard.qty`, `standard.name` is the quantity and name of the standard added to calculate concentrations.
- `sample.qty`, is the quantity of the sample added
- `chlor.volume`, is the volume chloroform added
- `chlor.weight`, is the weight of the chloroform recovered
- `peak.interval`, `ecl.interval`, are the precision or interval in equivalent chain length units with which the standard peaks or standard ecls are distinguished
- `min.area`, the minimal area below which the peaks are ignored
- `d13C.meth`, the delta 13C value of the methyl group added when using the IRMS

## 4.2. Functions `fa.fid` and `fa.irms`

Functions `fa.fid` and `fa.irms` analyse the input data produced by the GC-FID and GC-IRMS respectively in one step.

They are very similar and defined as:

```

fa.fid(input, fa.ref = fid.bpx70,
        standard.ecl = list(time = c(12, 16, 19),
                              ecl = c(12, 16, 19),
                              name = c("C12:0", "C16:0", "C19:0")),
        standard.qty, sample.qty = 1,
        chlor.volume, chlor.weight,
        peak.interval = 0.02, ecl.interval = 0.02,
        min.area = 0, accept = NULL, verbose = TRUE)

fa.irms(input, fa.ref = irms.bpx70,
        standard.ecl = list(time = c(800, 1600, 2000),
                              ecl = c(12, 16, 19),
                              name = c("C12:0", "C16:0", "C19:0")),
        standard.qty, sample.qty = 1,
        chlor.volume, chlor.weight,
        peak.interval = 0.02, ecl.interval = 20,
        min.area = 0, d13C.meth = -45.6,
        accept = NULL, verbose = TRUE)

```

where the arguments are as above, except for “input” which can be either an input file name or a `data.frame` as produced with `read.fid` or `read.irms`.

### 4.3. Example of GC-FID data

For example, in the following code, a data set and reference column is first read from file, after loading library `Rlims`. To read the data set, R-function `read.fid` is used; the reference data set is read with `read.csv`.

*reading input files*

```

> library(Rlims)
> fid.data <- read.fid("input_fid.csv")
> fid.ref  <- read.csv("reference_fid.csv")

```

We look at the first part (`head`) of these input data (`fid.data`) and reference data (`fid.ref`):

```

> head(fid.data)

  number  time   area orig.name
1      1 10.070 350902
2      2 10.898  21731
3      3 11.470 184592   C10:0
4      4 12.950  21359
5      5 14.468 319252
6      6 17.005 1652931   C12:0

> head(fid.ref)

```

	ecl	name
1	12.000	C12:0
2	12.950	C13:0
3	13.460	i-C14:0
4	13.635	2-OH C10:0
5	13.960	C14:0
6	14.473	i-C15:0/C14:1w5c

### *ecl standards*

The suggested standards from the analytical lab are:

```
> standard.ecl <- list(time = c(17.007, 31.16, 41.3),
+                       ecl = c(12, 16, 19),
+                       name = c("C12:0", "C16:0", "C19:0"))
```

We use those to get the standards compatible with the input data (`standard.new`):

```
> standard.new <- get.standard (fid.data, standard.ecl, ecl.interval = 0.02)
> standard.new
```

```
$time
[1] 17.005 31.162 41.302
```

```
$ecl
[1] 12 16 19
```

```
$name
[1] "C12:0" "C16:0" "C19:0"
```

The new values are slightly different from the original ones, but they fall within the `ecl.interval`, which is by default equal to 0.02.

### *equivalent chain lengths*

Next we add the equivalent chain lengths; the new data set is now called `fid.ecl`.

```
> fid.ecl <- add.ecl(fid.data, standard.new)
> head(fid.ecl)
```

	number	time	area	orig.name	ecl
1	1	10.070	350902		10.04055
2	2	10.898	21731		10.27449
3	3	11.470	184592	C10:0	10.43611
4	4	12.950	21359		10.85428
5	5	14.468	319252		11.28318
6	6	17.005	1652931	C12:0	12.00000

A new column with equivalent chain lengths (ecl) of each peak has been added.

### *cleanup peaks*

Now we cleanup the input, by removing peaks too close to one another and removing peaks whose “area” is too low, and considered noise. The new data set is called `fid.clean`

We first assume nothing is noise (hence `min.area = 0`).

```
> fid.clean <- clean.peaks(fid.ecl, peak.interval = 0.02, min.area = 0)
```

```
removed 0 peaks that were too small; 77 peaks remaining
peaks nrs 16 and 17 are too close; removed peak number 16
peaks nrs 17 and 18 are too close; removed peak number 17
peaks nrs 30 and 31 are too close; removed peak number 30
peaks nrs 55 and 56 are too close; removed peak number 55
peaks nrs 64 and 65 are too close; removed peak number 64
peaks nrs 71 and 72 are too close; removed peak number 71
removing smallest peaks; 71 peaks remaining
```

Note the warnings.

The first warning means that no peaks are below the `min.area`.

Next warnings mean that some peaks are too close to be distinguished. By default, the function then removes the smaller peak. Here peaks 16, 17, and 18 were too close; so both peak 16 and 17 were removed, while peak 18, which has the largest area, was kept:

```
> fid.clean[12:17,]
```

	number	time	area	orig.name	ecl
12	12	31.162	914972	C16:0	16.00000
13	13	32.112	23720	C16:1w7t	16.28107
14	14	32.432	160586	C16:1w7c/10MeC	16.37574
17	17	32.917	253454	i-C17:0	16.51923
18	18	34.653	92770	C17:0	17.03284
19	19	35.765	24302	cy C17:0	17.36183

Alternatively, we can also cleanup the chromatogram by using a “noise level” to get rid of peaks that are too small; thus we exclude all peaks whose area is lower than a minimal value (`min.area`); here we use a minimal value of 100000.

```
> fid.clean <- clean.peaks(fid.ecl, peak.interval = 0.02, min.area = 100000)
```

```
removed 35 peaks that were too small; 42 peaks remaining
```

```
> head(fid.clean)
```



	number	time	area	orig.name	ecl
1	1	10.070	350902		10.04055
3	3	11.470	184592	C10:0	10.43611
5	5	14.468	319252		11.28318
6	6	17.005	1652931	C12:0	12.00000
7	7	20.333	263481	C13:0	12.94031
8	8	23.973	140943	C14:0	13.96878

### *identify peaks*

Now we identify peaks by comparing with the reference column, using function `get.peaks`:

```
> fid.new <- get.peaks(fid.clean, fa.ref = fid.ref, peak.interval = 0.02)
```

some peaks are unknown - run `edit.chrom` or `accept.chrom` on the input ...

Note the warning that some peaks are unknown, i.e. they cannot be given names, because they are not within the `peak.interval` from a known peak.

```
> fid.new[5:10,]
```

	number	time	area	orig.name	ecl	name	numpeaks
7	7	20.333	263481	C13:0	12.94031	C13:0	1
8	8	23.973	140943	C14:0	13.96878	C14:0	1
9	9	26.668	226872	ai-C15:0	14.73024	UNKN ai-C15:0	0
12	12	31.162	914972	C16:0	16.00000	C16:0	1
14	14	32.432	160586	C16:1w7c/10MeC	16.37574	C16:1w7c	1
17	17	32.917	253454	i-C17:0	16.51923	i-C17:0	1
		distance					
7		0.009687787					
8		-0.008778696					
9		-0.085239458					
12		0.000000000					
14		-0.003739645					
17		-0.014230769					

A number of new columns have been added:

- column “name” contains the suggested name of the peak, based on the reference column; unknown peaks have the prefix “UNKN” followed by a suggested name (the peak that was closest).
- Column “numpeaks” contains the number of peaks from the reference column that were close enough (i.e. whose ecl was < `peak.interval`).
- Column “distance” refers to the distance of the peak to the nearest reference peak.

Note that the new name does not always correspond with the original name - if that happens, and it is important - check and ask the head of the analytical lab what to do!

Other new names have the label "UNKN" in front, while the number of peaks ("numpeaks") equals 0. In the example this applies to peak number 9 and the closest reference peak, at distance 0.021 is named "ai-C15:0".

In the next section, it is shown how this can be remediated.

### FA concentrations

Finally, we can estimate the concentration of each FA, using the input parameters obtained during the analytical procedure. The quantity of the standard added (`standard.qty`) was 2.01, sample was added in quantity = 1 (`sample.qty`) and the volume chloroform used (`chlor.volume`) was 15; the weight of chloroform recovered `chlor.weight` was 20.

```
> fid.final <- get.conc.fid(fid.new, standard.qty = 2.01, sample.qty = 1,
+                           chlor.volume = 15, chlor.weight = 20)
> head(fid.final)
```

	number	time	area	orig.name	ecl	name	numpeaks
1	1	10.070	350902		10.04055	UNKN C12:0	0
3	3	11.470	184592	C10:0	10.43611	UNKN C12:0	0
5	5	14.468	319252		11.28318	UNKN C12:0	0
6	6	17.005	1652931	C12:0	12.00000	C12:0	1
7	7	20.333	263481	C13:0	12.94031	C13:0	1
8	8	23.973	140943	C14:0	13.96878	C14:0	1

	distance	conc	perc
1	1.959454687	1.0223649	1.3922122
3	1.563890655	0.5378150	0.7323732
5	0.716818535	0.9301515	1.2666400
6	0.000000000	4.8158705	6.5580438
7	0.009687787	0.7676608	1.0453673
8	-0.008778696	0.4106422	0.5591948

Note the two extra columns with the concentration ("conc") and the percentage ("perc").

### 4.4. Same example, one step

It is also possible to use function `fa.fid` to convert a file to the proper format in one step.

```
> fid.final <- fa.fid(input = "input_fid.csv", fa.ref = "reference_fid.csv",
+                     standard.ecl = list(time = c(17.007, 31.16, 41.3),
+                                             ecl = c(12, 16, 19),
+                                             name = c("C12:0", "C16:0", "C19:0")),
+                     standard.qty = 2.01,
+                     chlor.volume = 15, chlor.weight = 20,
+                     min.area = 100000)
```

removed 35 peaks that were too small; 42 peaks remaining  
 some peaks are unknown - run `edit.chrom` or `accept.chrom` on the input ...

#### 4.5. Troubleshooting ambiguous peaks

In the above example, the functions `get.peaks` and `fa.fid` complain that some peaks cannot unambiguously be attributed to a certain fatty acid and need your input.

There are two options when that happens:

- you accept the defaults, as already present in the input file (not recommended). You do this by running function `accept`.
- you go through all problems manually, using the `edit` command.

With the `accept` command, you can either use the default name as estimated by function `get.peaks`, or give precedence to the original name from the input file ("`orig.name`"):

```
> PLFA1 <- accept(fid.final, accept = "name")
> PLFA2 <- accept(fid.final, accept = "orig.name")
> PLFA1[5:10,]
```

	number	time	area	orig.name	ecl	name	numpeaks
7	7	20.333	263481	C13:0	12.94031	C13:0	1
8	8	23.973	140943	C14:0	13.96878	C14:0	1
9	9	26.668	226872	ai-C15:0	14.73024	ai-C15:0	0
12	12	31.162	914972	C16:0	16.00000	C16:0	1
14	14	32.432	160586	C16:1w7c/10MeC	16.37574	C16:1w7c	1
17	17	32.917	253454	i-C17:0	16.51923	i-C17:0	1
		distance	conc	perc			
7		0.009687787	0.7676608	1.0453673			
8		-0.008778696	0.4106422	0.5591948			
9		-0.085239458	0.6609993	0.9001202			
12		0.000000000	2.6658019	3.6301736			
14		-0.003739645	0.4678728	0.6371288			
17		-0.014230769	0.7384468	1.0055849			

```
> PLFA2[5:10,]
```

	number	time	area	orig.name	ecl	name	numpeaks
7	7	20.333	263481	C13:0	12.94031	C13:0	1
8	8	23.973	140943	C14:0	13.96878	C14:0	1
9	9	26.668	226872	ai-C15:0	14.73024	ai-C15:0	0
12	12	31.162	914972	C16:0	16.00000	C16:0	1
14	14	32.432	160586	C16:1w7c/10MeC	16.37574	C16:1w7c	1
17	17	32.917	253454	i-C17:0	16.51923	i-C17:0	1
		distance	conc	perc			

	row.names	ecl	name	distance	orig.name
1	1	10.04055	UNKN C12:0	1.959455	
2	2	10.27449	UNKN C12:0	1.725507	
3	3	10.43611	UNKN C12:0	1.563891	C10:0
4	4	10.85428	UNKN C12:0	1.145723	
5	5	11.28318	UNKN C12:0	0.7168185	
6	6	12	C12:0	0	C12:0
7	7	12.94031	C13:0	0	C13:0
8	8	13.96878	C14:0	0	C14:0
9	9	14.73024	ai-C15:0	-0.08523946	ai-C15:0
10	10	14.99216	C15:0	0	C15:0
11	11	15.79798	UNKN 2-OH C12:0	-0.0209798	2-OH C12:0
12	12	16	C16:0	0	C16:0
13	13	16.28107	10Me-C16:0	0	C16:1w7e
14	14	16.37574	C16:1w7e	0	C16:1w7c/10MeC
15	17	16.51923	i-C17:0	0	i-C17:0
16	18	17.03284	C17:0/3-OH C12:0	0	C17:0
17	19	17.36183	cy C17:0	0	cy C17:0
18	20	17.54911	UNKN i-C18:0/C16:3w4	-0.02511243	i-C18:0/C16:3w
19	21	18.0358	C18:0	0	C18:0

Figure 3: The edit screen that shows the problems for which a decision should be taken

```

7  0.009687787 0.7676608 1.0453673
8  -0.008778696 0.4106422 0.5591948
9  -0.085239458 0.6609993 0.9001202
12  0.000000000 2.6658019 3.6301736
14  -0.003739645 0.4678728 0.6371288
17  -0.014230769 0.7384468 1.0055849

```

Alternatively, we go through each problem manually, each time taking a decision. This is done using the `edit` command, which opens R's spreadsheet-like interface

```
PLFA <- edit(PLFA1)
```

This command opens a window like in figure 3, which shows the problems.

For instance,

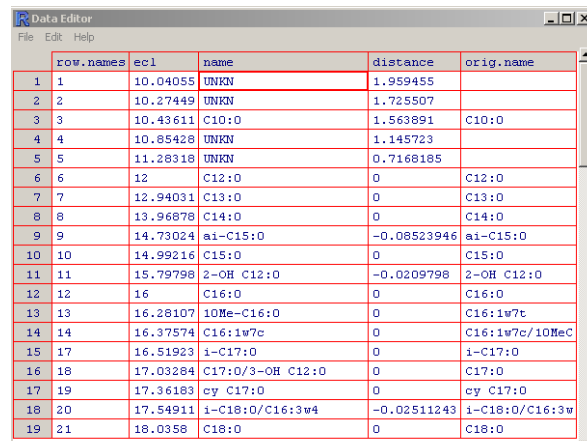
- on the line 11, the equivalent chain length of *15.79798* is closest to the peak corresponding to *2-OH C12:0*, but it differs from the ecl of this plfa with 0.021, which is more than 0.02 (the inputted precision).

As it is likely that this ecl is indeed *2-OH C12:0*, we accept this name and replace the item *UNKN 2-OH C12:0* with *2-OH C12:0*.

- On line 2, the suggested name is *C12:0*, but clearly it should be *C10:0*.
- Another uncertainty of a different kind is the occurrence of multiple peaks, where we have to select the most likely plfa. (KARLINE: VOORBEELD VERZINNEN)

The edited PLFA file is in figure 4.

Note that the editor works similar as excel, but is not as flexible; use ctrl C and ctrl V (copy-paste), and double click on a cell.



	row.names	ecl	name	distance	orig.name
1	1	10.04055	UNKN	1.959455	
2	2	10.27449	UNKN	1.725507	
3	3	10.43611	C10:0	1.563891	C10:0
4	4	10.85428	UNKN	1.145723	
5	5	11.28318	UNKN	0.7168185	
6	6	12	C12:0	0	C12:0
7	7	12.94031	C13:0	0	C13:0
8	8	13.96878	C14:0	0	C14:0
9	9	14.73024	ai-C15:0	-0.08523946	ai-C15:0
10	10	14.99216	C15:0	0	C15:0
11	11	15.79798	2-OH C12:0	-0.0209798	2-OH C12:0
12	12	16	C16:0	0	C16:0
13	13	16.28107	10Me-C16:0	0	C16:1w7t
14	14	16.37574	C16:1w7c	0	C16:1w7c/10MeC
15	15	16.51923	i-C17:0	0	i-C17:0
16	16	17.03284	C17:0/3-OH C12:0	0	C17:0
17	17	17.36183	cy C17:0	0	cy C17:0
18	18	17.54911	i-C18:0/C16:3w4	-0.02511243	i-C18:0/C16:3w
19	19	18.0358	C18:0	0	C18:0

Figure 4: The edit screen after which a decision has been taken for all problems

#### 4.6. Outputting the results

After that we can print the results to the screen

```
> PLFA2 [,c("name", "conc")]
```

	name	conc
1		1.0223649
3	C10:0	0.5378150
5		0.9301515
6	C12:0	4.8158705
7	C13:0	0.7676608
8	C14:0	0.4106422
9	ai-C15:0	0.6609993
12	C16:0	2.6658019
14	C16:1w7c	0.4678728
17	i-C17:0	0.7384468
21	C18:0	3.7015020
22	10Me-C18:0	0.7900572
23	C18:1w9c	0.9603794
24	C18:1w7c	0.9082971
25		0.9581972
26	C18:2w6c	0.3532864
27	C19:0	1.8231293
28	C18:3w6/cy C19:0	0.2933812
31	C18:3w3	0.6196679
32	2-OH C16:0	0.6877630
33	C20:0	0.7237160
34	C18:4w3	0.3300102
36	C20:1w9c	1.7339576
37		1.2457251
38		0.5229881
39	C20:2w9	0.7676870
41		0.6902861
43	C20:4w6	10.8093505
44	C20:3w3	1.8159736
45		0.2946020
48	C22:1w11	2.5700078
49	C22:1w9c	1.5707415
50	C20:5w3	8.2819968
55		0.9032159
56	C22:4w6	7.6300965
60	C22:5w3	3.7455139
61	C22:6w3	0.9543717
62		0.9828224
65	C27:0	0.9030090
74		0.9022399

```
76          0.7073244
77          1.2356355
```

print the inputs of the analysis:

```
> get.diagnostics(PLFA2)
```

standards

```
      time ecl  name
1 17.005  12 C12:0
2 31.162  16 C16:0
3 41.302  19 C19:0
```

other inputs

```
      value
peak.interval 0.02
min.area      1e+05
fa.ref        "input.chrom"
standard.qty   2.01
sample.qty     1
chlor.volume   15
chlor.weight   20
standard.name  "C19:0"
name.accept    "orig.name"
```

and make a figure:

```
> plot(PLFA2, xlab = "ECL", ylab = "peak area", main = "a plfa dataset",
+      marker = fa.marker, lwd = 2)

> barplot(PLFA2, main = "barplot", marker = fa.marker, lwd = 2,
+      col = c("grey", "brown", "brown", "brown",
+      "green2", "darkgreen", "orange", "blue"))
```

Note that we use a data.frame with biomarker information to give colors to the different peaks. In the package is one such data.frame present, called “fa.marker”. Its contents can be visualised with:

```
> fa.marker
```

```
      name      group
1   C12:0  standard
2   C16:0  standard
3   C19:0  standard
4  C16:2w7   diatoms
5  C16:2w4   diatoms
6  C16:3w4   diatoms
```

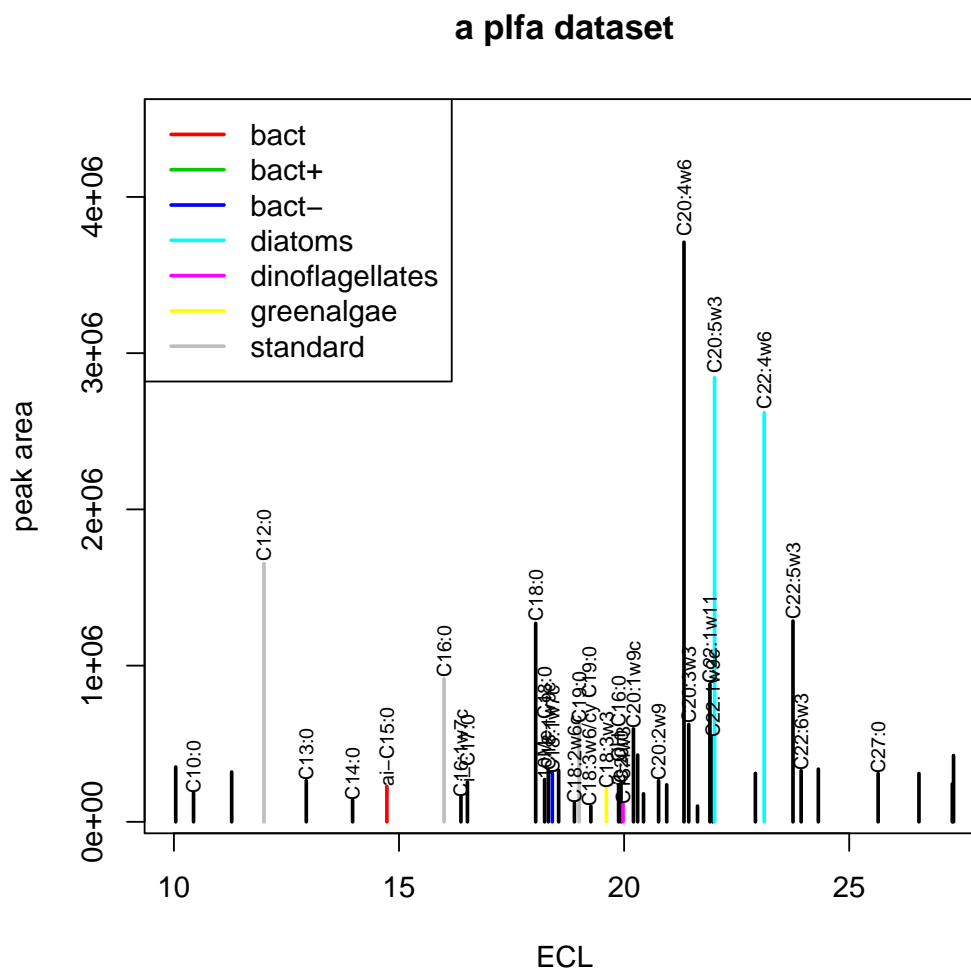


Figure 5: Plot of the plfa analysis - see text for R-code

7	C16:4W1	diatoms
8	C20:5w3	diatoms
9	C22:4w6	diatoms
10	C16:3w3	greenalgae
11	C16:4w3	greenalgae
12	C18:3w3	greenalgae
13	C18:4w3	dinoflagellates
14	i-C14:0	bact+
15	i-C15:0	bact+
16	ai-C15:0	bact
17	C18:1w7c	bact-



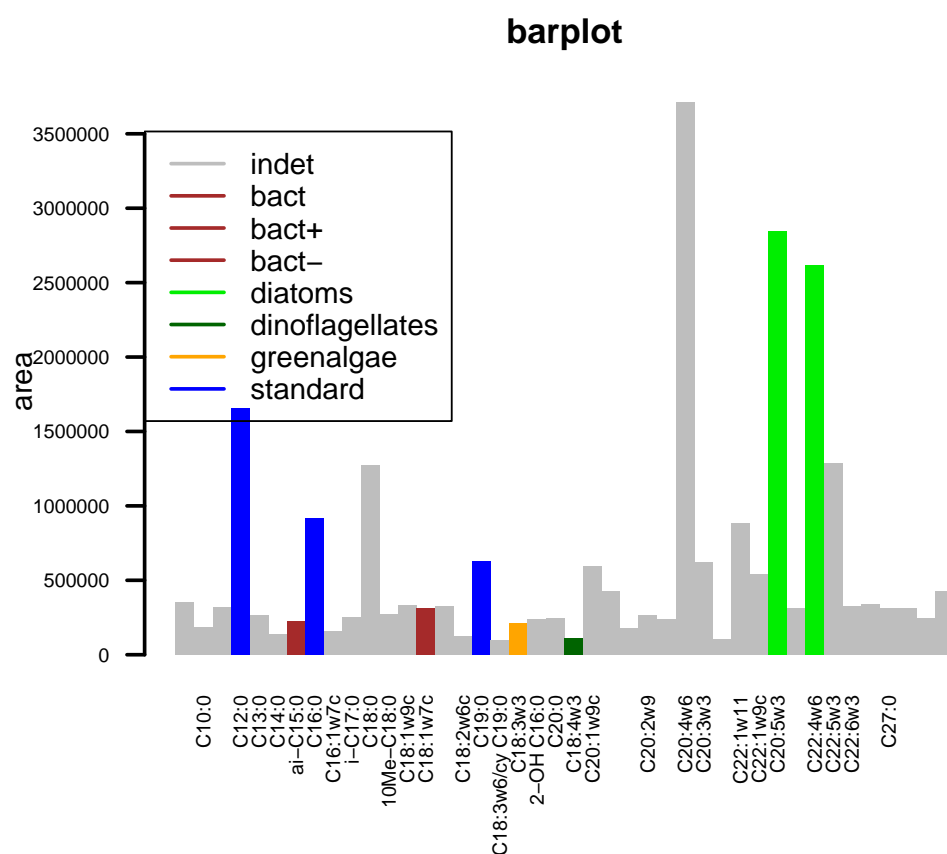


Figure 6: Barplot of the plfa analysis - see text for R-code

#### 4.7. Example of GC-c-IRMS data

We now analyse a GC-c-IRMS data set, in one step.

```
> standard.ecl <- list(time = c(883.5, 1637.8, 2227.2),
+                       ecl = c(12, 16, 19),
+                       name = c("C12:0", "C16:0", "C19:0"))
> irms.final <- fa.irms(input = "input_irms.csv", fa.ref = "reference_irms.csv",
+   standard.ecl = standard.ecl, ecl.interval = 20,
+   standard.qty = 1.55, #accept = "name",
+   chlor.volume = 15, chlor.weight = 20,
+   min.area = 0.2)
```

removed 15 peaks that were too small; 37 peaks remaining  
 some peaks are unknown - run edit.chrom or accept.chrom on the input ...

```
> barplot(irms.final, main = "IRMS", marker = fa.marker, lwd = 2,
+   col = c("grey", "brown", "brown", "brown",
+   "green2", "darkgreen", "orange", "blue"), poslegend = "top")
```

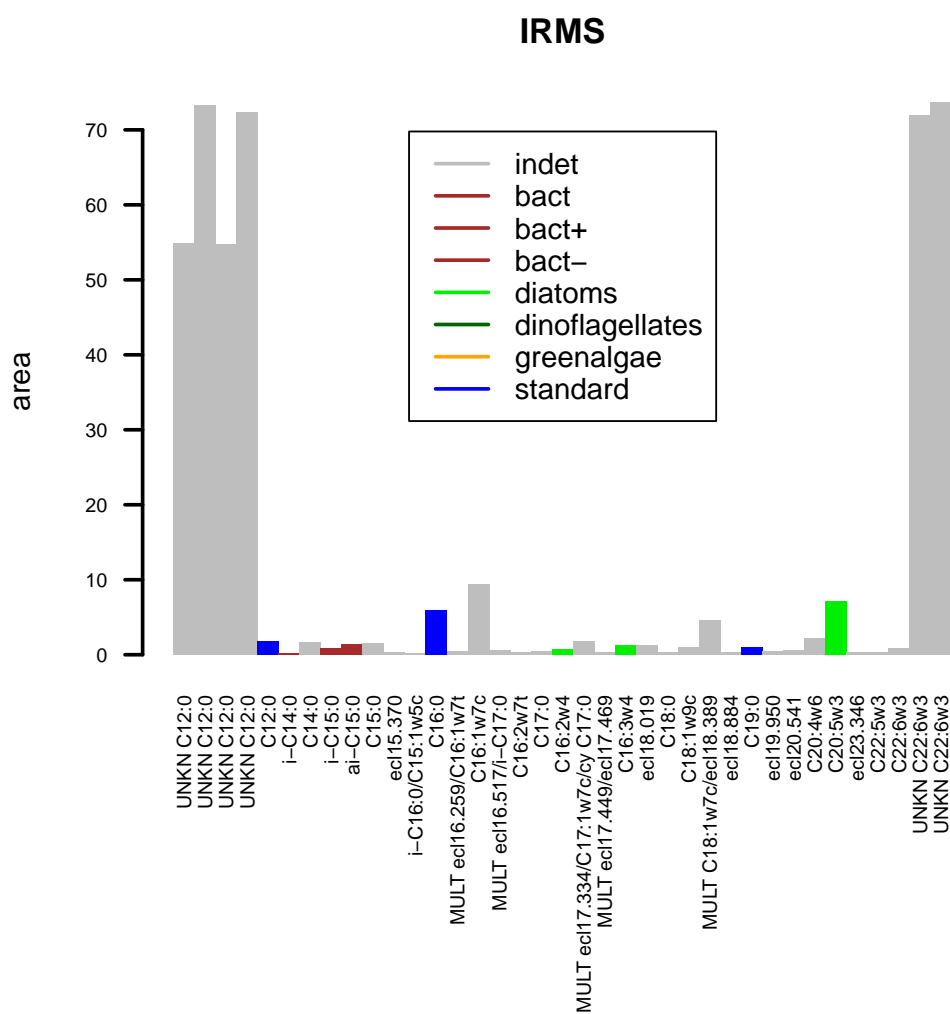


Figure 7: Plot of the irms analysis - see text for R-code

## 5. equations

The chromatograms obtained from the analytic instruments, are converted to PLFA concentrations using several inputs:

- the retention times of the standards and their equivalent chain length
- the quantity of the standard added
- the quantity of the sample
- the volume chloroform added
- the weight of chloroform recovered
- the equivalent chain lengths of known PLFAs for the reference column used.

The analysis of individual PLFAs is done based on the relative retention times (RT) of each peak, standardized for the RTs of the standards. The PLFAs C12:0 and C19:0 are added as internal standards and C16:0 is always present in samples as one of the main peaks.

The calculation proceeds in two (fid) or three (irms) steps:

- In a first step, the retention times of peaks are recalculated in equivalent chain lengths (ECL)
- In a second step, the PLFA concentration is calculated, based on the amount sample and standard added.
- For irms data, the delta values of the PLFAs are then calculated.

### 5.1. equivalent chain lengths from retention times

Given the retention times (RT) for the standards “C12:0”, “C16:0”, “C19:0”, which have known equivalent chain lengths (ECL) of 12, 16, and 19 respectively, the ECL of the other components are estimated as:

$$ECL_{PLFA} = 12 + (16 - 12) \cdot \frac{RT_{PLFA} - RT_{C12:0}}{RT_{C16:0} - RT_{C12:0}} \quad (1)$$

$$ECL_{PLFA} = 16 + (19 - 16) \cdot \frac{RT_{PLFA} - RT_{C16:0}}{RT_{C19:0} - RT_{C16:0}} \quad (2)$$

for components inbetween C12:0 and C16:0 and inbetween C16:0 and C19:0 respectively. Components before C12:0 or after C19:0 are estimated by extrapolation, using the nearest gradient.

### 5.2. PLFA concentration for FID

The concentrations of the PLFAs, expressed in  $\mu g$  PLFA, are estimated based on the peak areas of the respective PLFA ( $A_{PLFA}$ ), the peak area of the standard “C19:0” ( $A_{19:0}$ ) and the

PLFA concentration of the standard that was added ( $C_{19:0}$ ):

$$C_{PLFA} = \frac{A_{Fame}/A_{19:0} \cdot C_{19:0}}{g_S \cdot f} \quad (3)$$

where  $g_S$  is the total amount of the sample,  $f$  is the fraction of chloroform that is recovered during the extraction.

At NIOO, the amount of C19:0 added is  $\approx 2.01 \mu g$  PLFA (see below).

### 5.3. C concentration for IRMS

The concentrations of the PLFAs, in  $\mu g$  Carbon, are estimated based on the peak areas of the respective PLFA ( $A_{PLFA}$ ), the peak area of the standard “C19:0” ( $A_{19:0}$ ) and the carbon concentration of the standard that was added ( $C_{19:0}$ ):

$$C_{PLFA} = \frac{A_{Fame}/A_{19:0} \cdot C_{19:0}}{g_S \cdot f} \cdot \frac{n}{n+1} \quad (4)$$

where  $g_S$  is the total amount of the sample,  $f$  is the fraction of chloroform that is recovered during the extraction and  $n$  is the number of C-atoms in the PLFA. The last factor is to correct for the methyl group that was added during the analytical procedure.

At NIOO, the amount of C19:0 added is  $\approx 1.54 \mu g$  C (see below).

### 5.4. PLFA isotopic compositions for IRMS

The delta values for the PLFAs also have to be corrected for the methyl group that was added.

$$\delta^{13}C_{PLFA} = \frac{(n+1)\delta^{13}C_{FAME} - 1\delta^{13}C_{methanol}}{n} \quad (5)$$

At NIOO, the methanol group has a  $\delta$  value of  $\delta^{13}C_{methanol} -46\text{‰}$

## 6. Important notes

### 6.1. Units of the GC-FID and GC-c-IRMS

The GC-FID apparatus gives output in weight *PLFA*, while the GC-c-IRMS measures in *Carbon*.

These are also the units that are returned by the R-functions !

It is very important that the weight of the standard added is expressed in the same unit, for instance  $\mu g$  PLFA for GC-FID and  $\mu g$  C for GC-C-IRMS.

### 6.2. Weight of the Standard added

The molar weight of “C19:0” is 312 g/mol, and can be used to convert from one unit to the other.

Assume that, in the procedure 20  $\mu\text{l}$  C19:0 of a concentration 0.1  $\text{mg ml}^{-1}$  has been added to the solution. This amounts to  $20 * 0.1 = 2\mu\text{g}$  C19 added. This is the quantity that has to be used in the procedure using the GC-FID. The calculations for this machine return concentrations also expressed in  $\mu\text{g}$  PLFA. To convert these numbers to  $\mu\text{g C}$ , one needs to take into account the molar weight of the PLFA.

#### *GC-c-IRMS standard*

Dividing by the molar weight, we obtain the concentration of C19:0 added, expressed in  $\mu$  moles. This is  $2/312 = 0.00641\mu\text{mol}$  C19:0 added. C19:0 contains 19 carbon atoms; one more C atom is added in the form of methylester. As the molar weight of carbon = 12, the amount of carbon added is  $0.00641 * 20 * 12 = 1.5385\mu\text{g}$  Carbon. All calculations for the GC-c-IRMS also return concentrations in gram Carbon.

#### *Current standard concentration*

The standard solution is made once in a while, and its concentration can vary slightly.

As from Februari 2010, the concentration of the standard is 0.1053. With 20  $\mu\text{l}$  we calculate the required input for both machines.

```
> conc.standard <- 0.1053
> (standard.fid <- 20*conc.standard)

[1] 2.106

> (standard.irms <- 20*conc.standard/312*20*12)

[1] 1.62
```

### 6.3. ecls that require attention in GC-c-IRMS

- C17:1w7c, ecl  $\approx$  17.339
- C16:3w3, ecl  $\approx$  17.373; tends to move -> 17.31
- C16:3w4, ecl  $\approx$  17.47; tends to move -> 17.41

## 7. Things to do

- A function to merge several plfa upgraded data sets into one comprehensive table?
- An export function to LIMS!
- add links and references to help files + vignettes....
- check use of standard.ecl: should input values overwrite the original ones????
- Should peaks before first peak from reference column or after last peak be removed?

**Affiliation:**

Karline Soetaert

Centre for Estuarine and Marine Ecology (CEME)

Netherlands Institute of Ecology (NIOO)

4401 NT Yerseke, Netherlands

E-mail: [k.soetaert@nioo.knaw.nl](mailto:k.soetaert@nioo.knaw.nl)

URL: <http://www.nioo.knaw.nl/users/ksoetaert>

Pieter Provoost

Centre for Estuarine and Marine Ecology (CEME)

Netherlands Institute of Ecology (NIOO)

4401 NT Yerseke, Netherlands

E-mail: [p.provoost@nioo.knaw.nl](mailto:p.provoost@nioo.knaw.nl) Pieter van Rijswijk

Centre for Estuarine and Marine Ecology (CEME)

Netherlands Institute of Ecology (NIOO)

4401 NT Yerseke, Netherlands

E-mail: [p.vanRijswijk@nioo.knaw.nl](mailto:p.vanRijswijk@nioo.knaw.nl)