HowTo_FlimDiagRam

Julien Godet

22/04/2020

How to ?

This is a short vignette showing how to use the *flimDiagRam* package.

Contents

Installation

To be able to use the functions and utilities of the *flimDiagRam package*, you need first to install the package. This can be done by installing the last updated version directly from github.

```
if(!require(devtools)){install.packages("devtools")}
if(!require(flimDiagRam)){devtools::install_github("jgodet/flimDiagRam")}
if(!require(xtable)){install.packages('xtable')}
```

Then, call the package

```
library(flimDiagRam)
```

' **Note:** installation needs to be done only once. On the contrary, packages must be called on every new R sessions

For this vignette, we will also install two additional packages

```
if(!require(utilitR)){devtools::install_github("jgodet/utilitR")}
if(!require('tidyverse')){install.packages('tidyverse')}
```

Read the *.asc files

The first step is to read the data. For this vignette, we can use file examples available on github. These files correspond to data of a singly labelled protein (labelled with eGFP) acquired at two different days of experiment.

```
path <- "https://raw.githubusercontent.com/jgodet/flimDiagRam/master/data/"
tauFilePath <- paste(path, "PAS406_001_t1.asc", sep="")
photFilePath <- paste(path, "PAS406_001_photons.asc", sep="")

data <- getData(pathTau = tauFilePath, pathPhoton = photFilePath, label = "Image1", minPhotons = 1000)
```

Let's check everything went OK

```
head(data)
```

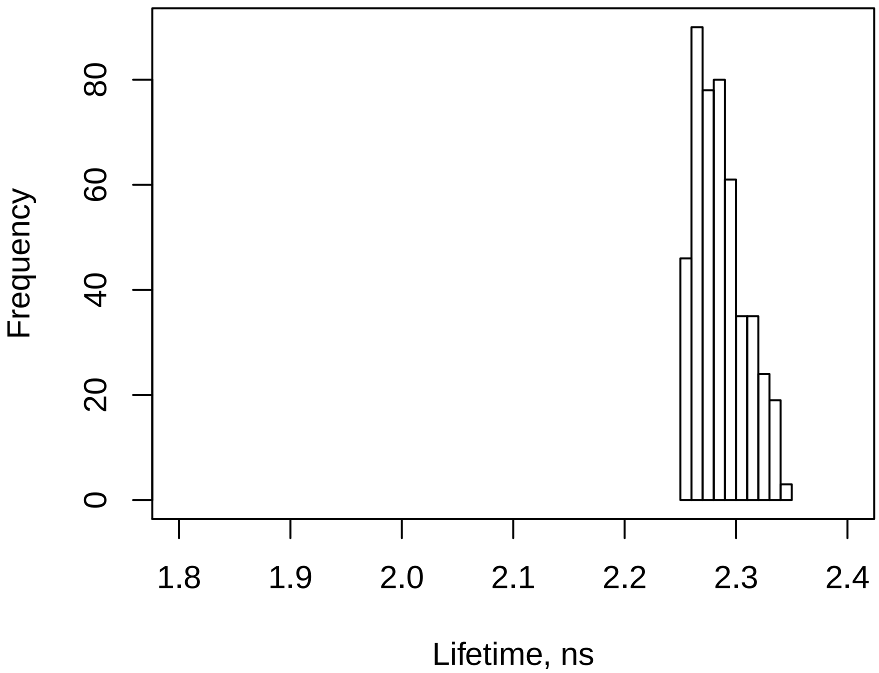
```
##       tau1d photons row col    ind
## 59  2316.74    1604  59   1 Image1
## 60  2317.43    3156  60   1 Image1
## 61  2313.05    4051  61   1 Image1
## 62  2312.28    2880  62   1 Image1
## 188 2321.20    2184  60   2 Image1
## 189 2315.67    2999  61   2 Image1
```

This table (data.frame) gives for each line the lifetime value (tau1d in ps), the intensity (photons), and the coordinates (row and column index) of a pixel of the FLIM image.

Basic data exploration

Histogram of the lifetime distribution

```
hist(data$tau1d/1000, main="",
     xlab="Lifetime, ns",
     xlim=c(1.8, 2.4))
box()
```

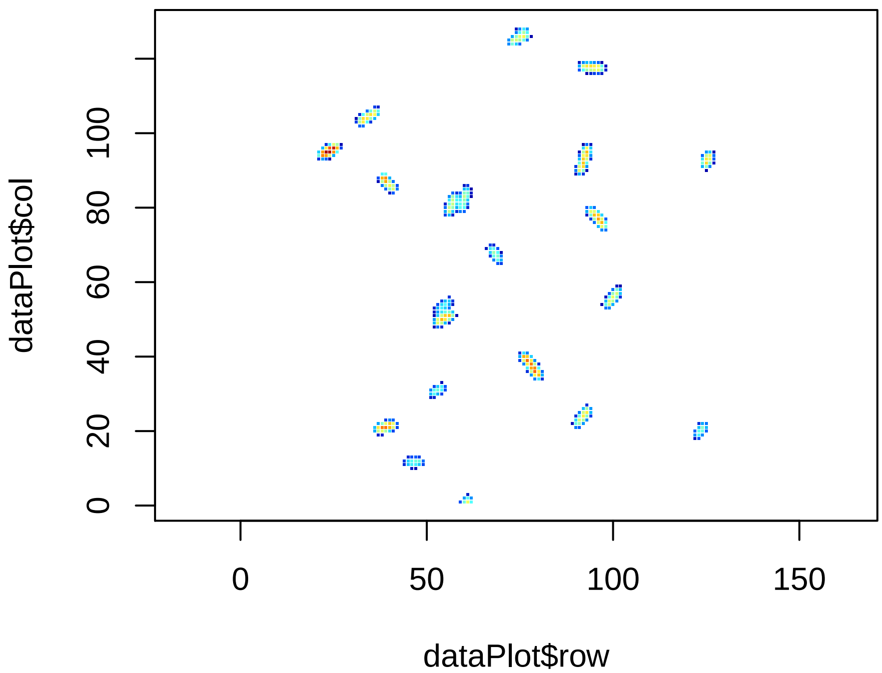


Aggregate data by bacteria

Plot the image

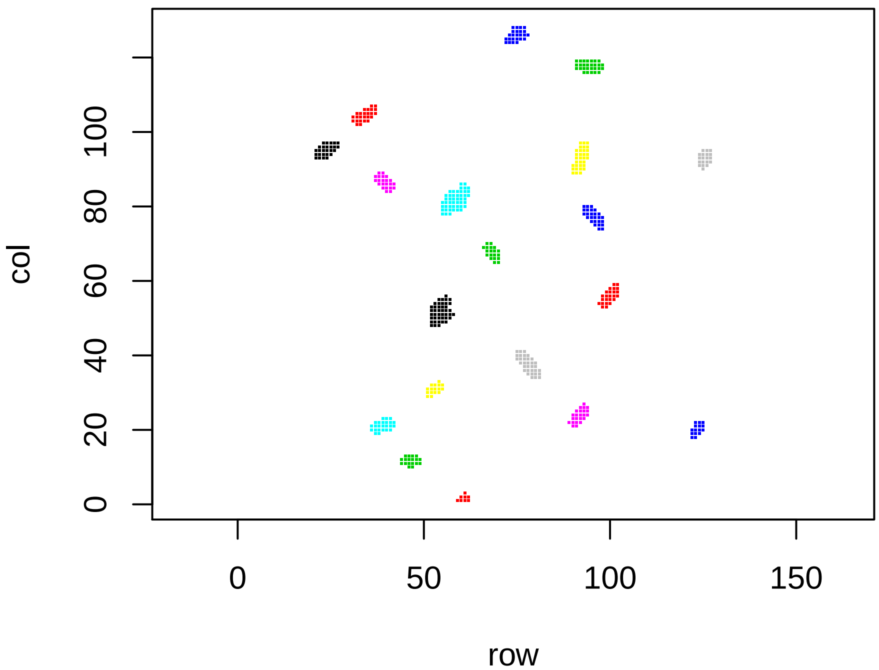
```
dataPlot <- data %>% dplyr::filter(photons>1000)
plot(x = dataPlot$row, y = dataPlot$col ,pch='.' ,
     col=utilitR::contColor(dataPlot$photons),asp=1)
```

```
## Loading required package: colorRamps
```



We will use the *dbscan* package to cluster pixels belonging to the same bacteria and *tidyverse* package to explore the data

```r
if(!require('dbscan')){install.packages('dbscan')}
require(dbscan)
res <- dbscan(data[,c("row","col")], eps = 2, minPts = 5)
data$bacteria <- res$cluster
plot(data[,c("row","col")], col = res$cluster + 1L, pch='.', asp=1)
```



Note : The cluster labelled "0" gathers pixel that were not attributed to any cluster.

Note 2 : Some bacteria are not correctly segmented using this basic clustering approach. It might be a good option to filter clusters according to their area (or number of pixels in the cluster) to discard the one corresponding to multiple bacteria

Now we need to aggregate data by clusters of pixels corresponding to one bacteria

```r
Im1_sumup <- data %>% filter(bacteria != 0) %>%
  group_by(bacteria) %>%
  dplyr::summarise( nPixels =n(),
            tau_mean = mean(tau1d, na.rm = T),
            tau_sd = sd(tau1d, na.rm = T),
            tau_med = median(tau1d, na.rm = T),
            label = unique(ind)) %>%
  filter(nPixels<40)

Im1_sumup
```

```
## # A tibble: 17 x 6
##    bacteria nPixels tau_mean tau_sd tau_med label
##       <int>   <int>    <dbl>  <dbl>   <dbl> <chr>
##  1        1       8    2318.   4.38   2317. Image1
##  2        2      18    2310.   3.91   2309. Image1
##  3        3      15    2270.   4.85   2270. Image1
##  4        4      24    2290.   1.76   2290. Image1
##  5        5      23    2272.   2.08   2272. Image1
##  6        6      16    2279.   4.97   2278. Image1
```

```
## 7          7       33     2259.    2.78    2258. Image1
## 8          9       24     2291.    3.63    2289. Image1
## 9         10       19     2323.    2.71    2325. Image1
## 10        11       26     2263.    3.82    2263. Image1
## 11        13       22     2335.    3.30    2335. Image1
## 12        14       32     2295.   13.0     2295. Image1
## 13        15       19     2283.    3.59    2283. Image1
## 14        16       26     2276.    3.22    2277. Image1
## 15        17       25     2259.    4.08    2259. Image1
## 16        18       28     2293.    3.31    2294. Image1
## 17        19       24     2281.    1.88    2281. Image1
```

We have some nice descriptive statistics of 17 individual bacteria.

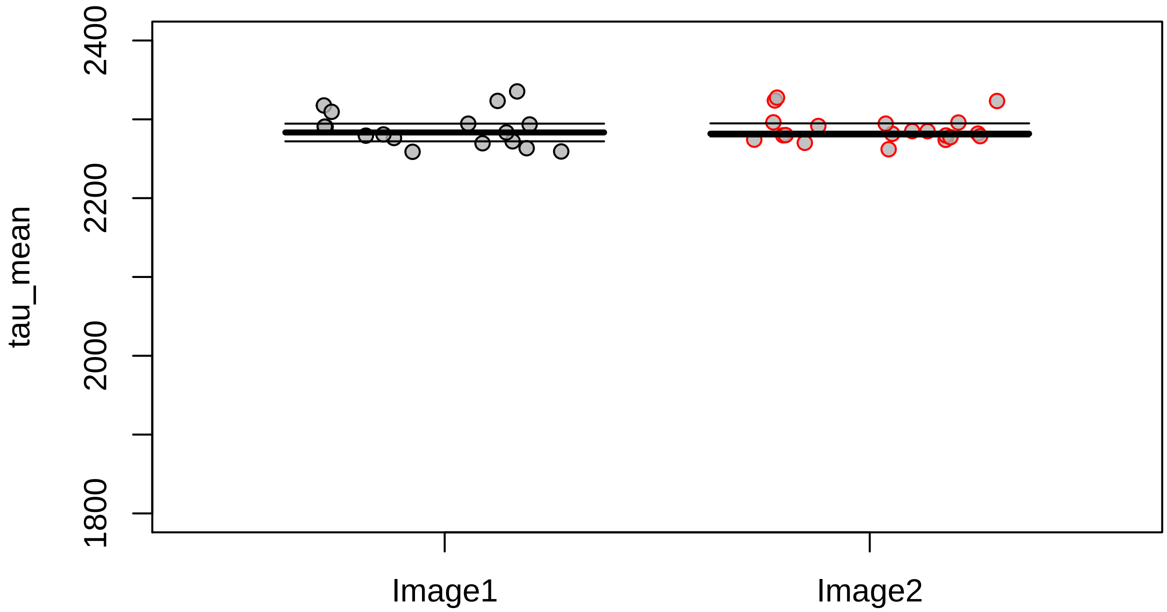
We can repeat the same procedure for a second FLIM image

```
tauFilePath2 <-  paste(path, "PAS406_002_t1.asc", sep="")
photFilePath2 <-  paste(path, "PAS406_002_photons.asc", sep="")

data2 <- getData(pathTau = tauFilePath2, pathPhoton = photFilePath2,
                 label = "Image2", minPhotons = 1000)

res2 <- dbscan(data2[,c("row","col")], eps = 2, minPts = 3)
data2$bacteria <- res2$cluster

Im2_sumup <- data2 %>% filter(bacteria != 0) %>%
  group_by(bacteria) %>%
  dplyr::summarise( nPixels =n(),
             tau_mean = mean(tau1d, na.rm = T),
             tau_sd = sd(tau1d, na.rm = T),
             tau_med = median(tau1d, na.rm = T),
             label = unique(ind)) %>%
  filter(nPixels<40)

#Im2_sumup
```

Let's bind table 1 and table 2 to check for example visually if we obtained the same results and lifetime values in these two images.

```
finalTable <- bind_rows( Im1_sumup, Im2_sumup)
utilitR::stripPlot(data = finalTable, tau_mean ~ label,
                   addBoxplot = T, ylim=c(1800, 2400))
```

```
## Loading required package: beeswarm
```



In this plot, each point corresponds to the average lifetime of a bacteria measured either in image 1 or in image 2. Horizontal lines correspond to the median value of cells (bold line) and the interquartile range (IQR). The lifetimes obtained for the two images look very similar. Indeed, a null hypothesis statistical test fails to demonstrate any difference of the bacteria cell average lifetime in these two images - confirming that reproducible results can be easily obtained in between independant experiements realized at different dates of experiments.

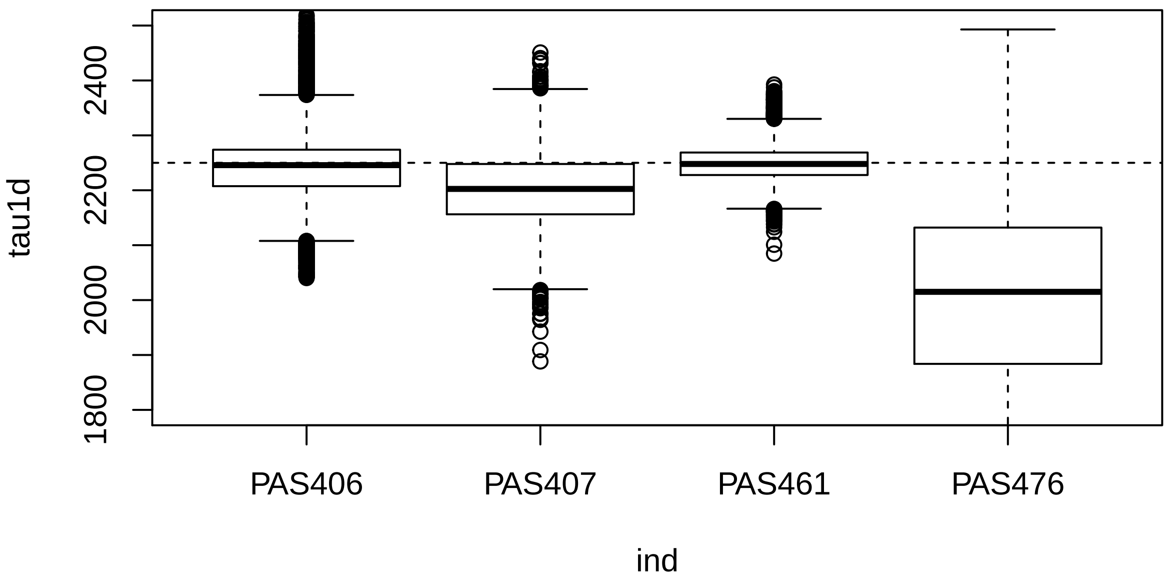
```
mod <- glm(tau_mean ~ label, data = finalTable, family = "gaussian")
options(xtable.comment = FALSE)
xtable(mod)
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	2288.1159	4.8939	467.54	0.0000
labelImage2	-0.0155	6.6564	-0.00	0.9982

Empirical distribution functions

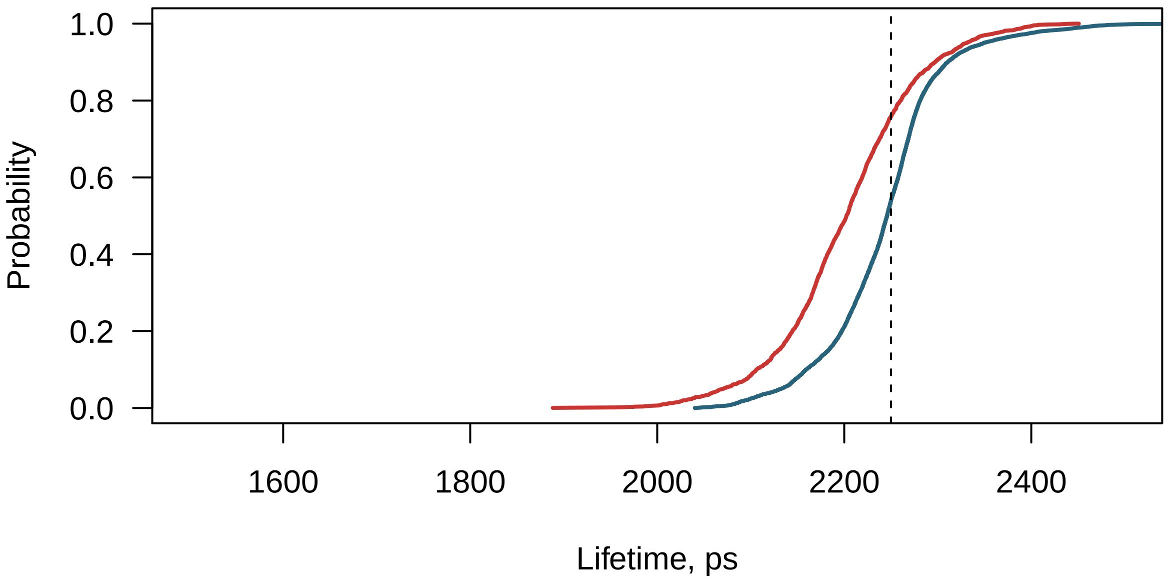
When FRET occurs, the lifetime of the donor is shortened and the lifetime distribution of the pixels of the FLIM images is shifted towards shorter times.

```
data406 <- getData(pathTau = paste(path, "PAS406_003_t1.asc", sep=""),
                   pathPhoton = paste(path, "PAS406_003_photons.asc", sep=""),
                   label = "PAS406")
data407 <- getData(pathTau = paste(path, "PAS407_001_t1.asc", sep=""),
                   pathPhoton = paste(path, "PAS407_001_photons.asc", sep=""),
                   label = "PAS407")
data461 <- getData(pathTau = paste(path, "PAS461_001_t1.asc", sep=""),
                   pathPhoton = paste(path, "PAS461_001_photons.asc", sep=""),
                   label = "PAS461")
data476 <- getData(pathTau = paste(path, "PAS476_001_t1.asc", sep=""),
                   pathPhoton = paste(path, "PAS476_001_photons.asc", sep=""),
                   label = "PAS476")


globalData <- bind_rows(data406, data407, data461, data476)
boxplot(data = globalData, tau1d ~ ind,  ylim=c(1800, 2500))
abline(h=2250, lty=2)
```

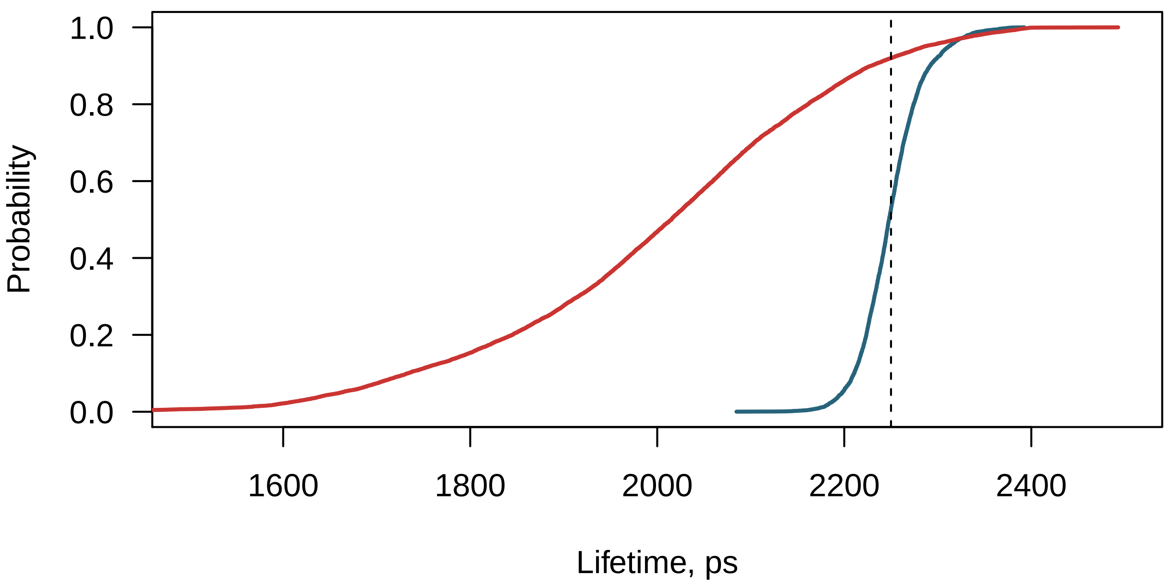


An easy way for comparing visually distributions is to use their empirical cumulative distribution functions. This can be done with the *plotEcdf* function

```
plotEcdf(AA =data406$tau1d, BB = data407$tau1d,
         xlim=c(1500,2500), xlab="Lifetime, ps",
         las=1,
         ylab= "Probability")
abline(v=2250,lty=2)
```




```
plotEcdf(AA =data461$tau1d, BB = data476$tau1d,
         xlim=c(1500,2500), xlab="Lifetime, ps",
         las=1,
         ylab= "Probability")
abline(v=2250,lty=2)
```



See figure Y in Manko et al. 2020 for the interpretation of these data.

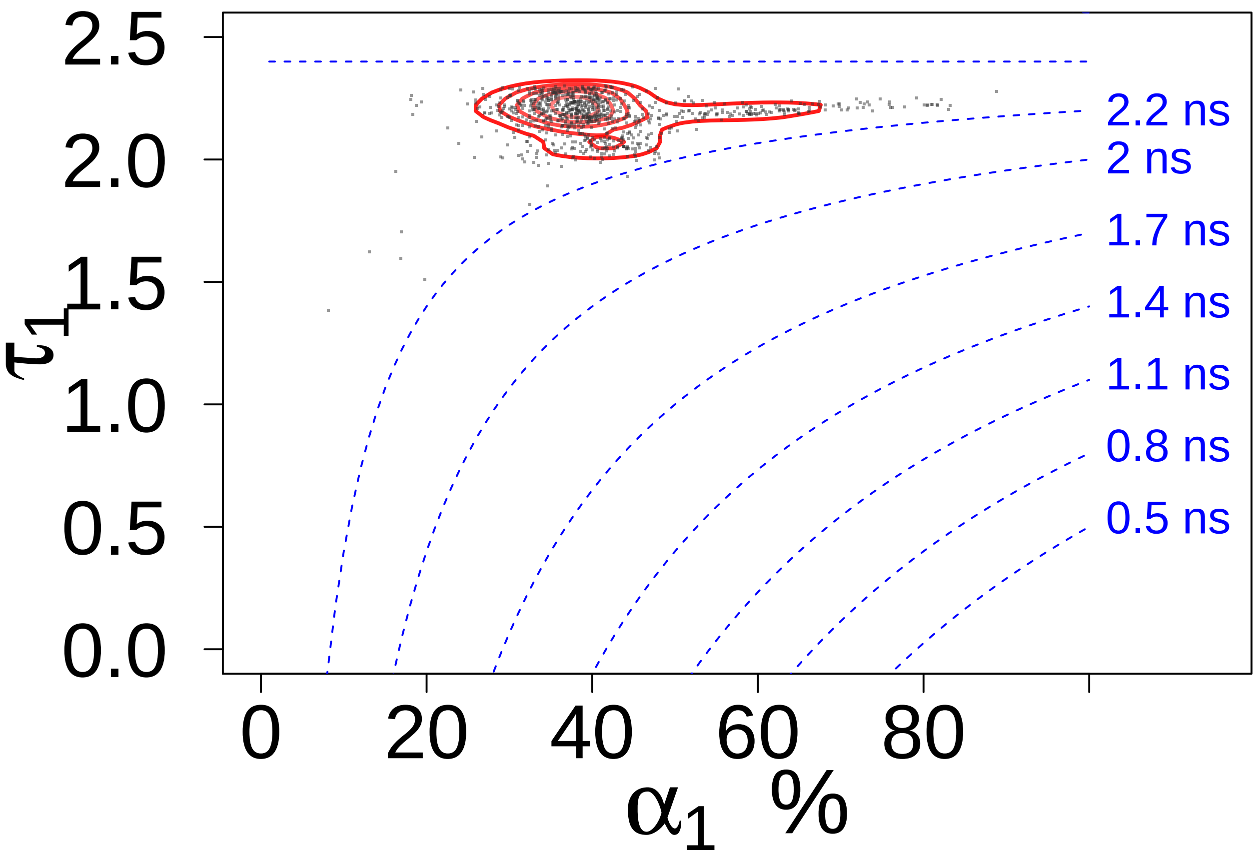
Advanced data exploration for double-explonential fit

First upload the data corresponding to double-explonential fit and write them locally.

```
download.file(url = paste(path, "407_20170302_002_t1.asc", sep=""),
              destfile = file.path(getwd(),"407_20170302_002_t1.asc"))
download.file(url = paste(path, "407_20170302_002_a1%5B%25%5D.asc", sep=""),
              destfile = file.path(getwd(),"407_20170302_002_a1[%].asc"))
download.file(url = paste(path, "407_20170302_002_a2%5B%25%5D.asc", sep=""),
              destfile = file.path(getwd(),"407_20170302_002_a2[%].asc"))
```

Then to plot the FLIM diagram, two lines are necessary. First load and read the data, then render the plot.

```
pathlowFret <- paste(getwd(),  "407_20170302_002", sep="/")
qsD <- getDataD(pathEGFP = pathlowFret, label = "PAS407" , ref = 2300)#load
plotDiagram(data = qsD, silence =TRUE)
```



This plot corresponds to a situation for which the number of donors is much higher than the number of acceptors. The fraction of transfering species is low (about 25-30 %). The lifetime values distribution is narrow with τ_1 values centerer at about 2.3 ns.

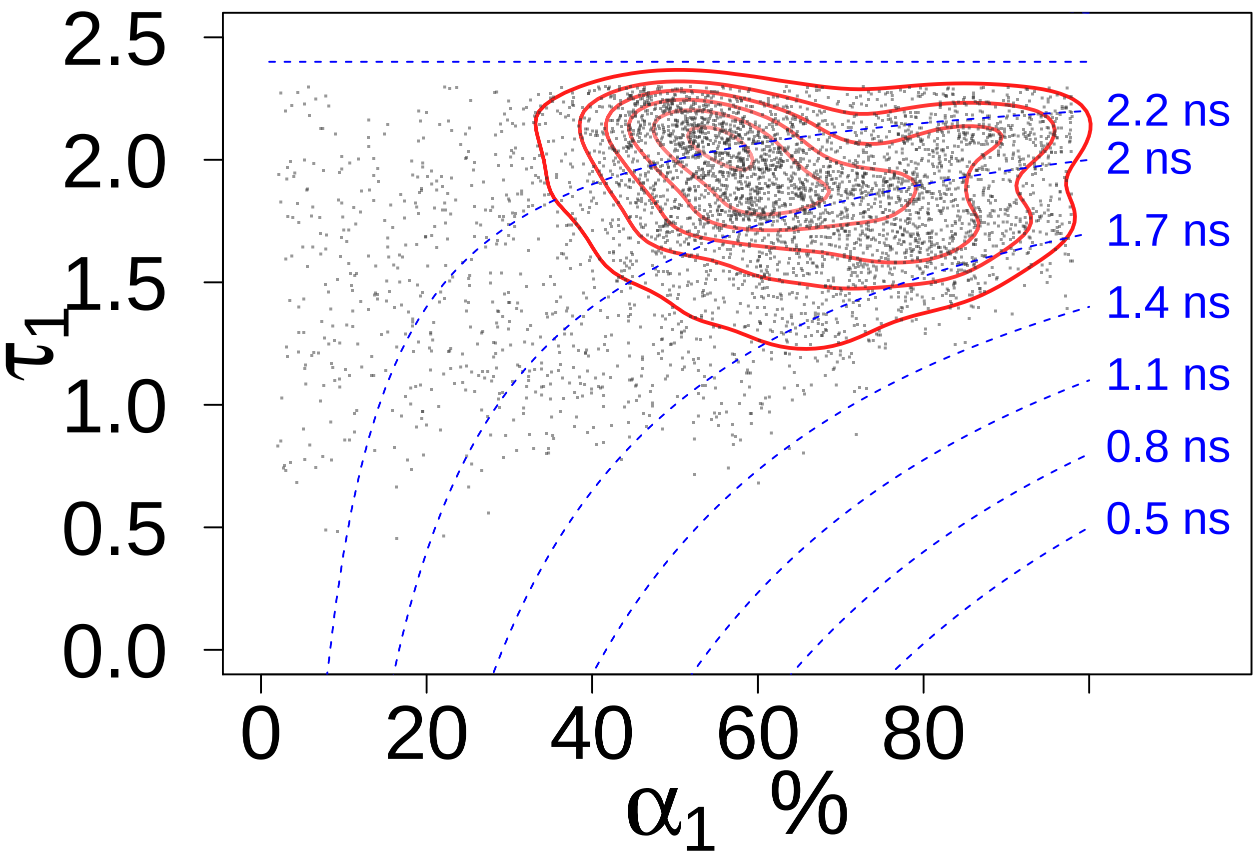
If we now look at data corresponding to a strain with the opposite labelling strategy (the donor is now on the lowest expressed enzyme), the diagram plot looks very different.

```
download.file(url = paste(path, "476_20170428_001_t1.asc", sep=""),
              destfile = file.path(getwd(),"476_20170428_001_t1.asc"))
download.file(url = paste(path, "476_20170428_001_a1%5B%25%5D.asc", sep=""),
              destfile = file.path(getwd(),"476_20170428_001_a1[%].asc"))
download.file(url = paste(path, "476_20170428_001_a2%5B%25%5D.asc", sep=""),
              destfile = file.path(getwd(),"476_20170428_001_a2[%].asc"))
```

```r
download.file(url = paste(path, "476_20170428_002_t1.asc", sep=""),
              destfile = file.path(getwd(),"476_20170428_002_t1.asc"))
download.file(url = paste(path, "476_20170428_002_a1%5B%25%5D.asc", sep=""),
              destfile = file.path(getwd(),"476_20170428_002_a1[%].asc"))
download.file(url = paste(path, "476_20170428_002_a2%5B%25%5D.asc", sep=""),
              destfile = file.path(getwd(),"476_20170428_002_a2[%].asc"))
```

In this exemple, the data of two images are combined

```r
pathhighFret1 <- paste(getwd(),  "476_20170428_001", sep="/")
pathhighFret2 <- paste(getwd(),  "476_20170428_002", sep="/")
qsF1 <- getDataF(pathFRET =pathhighFret1, label = "PAS476" )#load
qsF2 <- getDataF(pathFRET =pathhighFret2, label = "PAS476" )#load
qsF <- bind_rows(qsF1, qsF2)
plotDiagram(data = qsF, silence =TRUE,nl = 8)
```



Here, α_1 values are much higher than in the previous plot, ranging from 40% to nearly 100%. The large distribution of lifetimes shows the coexistence of complexes with different numbers of acceptors and different FRET efficiencies.

The graphical analysis of the same complexe but with two different labelling order demonstrates how the FLIM diagram can find very interesting visualization properties regarding complexes with unbalanced stoichiometry - a situation usually challenging to interpretate in FLIM-FRET data.