R-code for 'GCalignR. An R package for aligning Gas-Chromatography data'

Meinolf Ottensmann, Martin A. Stoffel, Joseph Hoffman

This document provides the code for the validation of alignments presented in our paper. Some parts a computational demanding and run for several hours on a standard computer. Therefore, we provide the final dataset along with this documentation.

Prerequisites

Most functions that are used in this analysis are part of our package `GCalignR`, while some more functions are provided in form of `R` scripts that are available along with this document in the subdirectory `R`. In order to run the code you need to have a subdirectory called `data` that contains the raw datafiles.

- Install `ggplot2`, `plot3D` and `devtools` if these packages are not available

```
## install ggplot2
if (!"ggplot2" %in% rownames(install.packages())) {
    install.packages("ggplot2")
}
## install plot3D
if (!"plot3D" %in% rownames(installed.packages())) {
    install.packages("plot3D")
}
## install devtools
if (!("devtools" %in% rownames(installed.packages()))) {
    install.packages("devtools")
} else if (packageVersion("devtools") < 1.6) {
    install.packages("devtools")
}
```

- Installing the most recent version of `GCalignR` from GitHub

```
## install GCalignR
devtools::install_github("mastoffel/GCalignR")
```

- Load packages and source custom function

```
library(GCalignR)
library(ggplot2)
library(plot3D)
## small function to test parameters in
## align_chromatograms
source("R/optimal_params.R")
## calculates errors by matching aligned data to a table
## of known substances
source("R/error_rate.R")
## custom function for simulations based on chromatograms
source("R/ChromaSimFunctions.R")
```

Explore the parameter space in `align_chromatograms`

There are two parameters of major importance, namely `max_diff_peak2mean` and `min_diff_peak2peak`. While the first determines the finescale grouping of retention times the latter greatly influences the formation of substances by combining initially separated rows of similar retention times. Here, we evaluate the error rate as a function of the combination of these two parameters. The combinations are tested by iteratively running `aling_chromatograms` with the following one hundred combinations

```
## run to obtain a table of all combinations
max_diff_peak2mean = seq(0.01, 0.05, 0.01)
min_diff_peak2peak = seq(0.01, 0.2, 0.01)
expand.grid(peak2mean = max_diff_peak2mean, peak2peak = min_diff_peak2peak)
```

- Run alignments with all combinations of both parameters

```
## B. flavifrons
results_bfla <- optimal_params(data = "data/bfla.txt", rt_col_name = "RT",
    max_diff_peak2mean = seq(from = 0.01, to = 0.05, by = 0.01),
    min_diff_peak2peak = seq(from = 0.01, to = 0.2, by = 0.01))
save(results_bfla, file = "data/results_bfla.RData")

## B. bimaculatus
results_bbim <- optimal_params(data = "data/bbim.txt", rt_col_name = "RT",
    max_diff_peak2mean = seq(from = 0.01, to = 0.05, by = 0.01),
    min_diff_peak2peak = seq(from = 0.01, to = 0.2, by = 0.01))
save(results_bbim, file = "data/results_bbim.RData")

## B. ephippiatus
results_beph <- optimal_params(data = "data/beph.txt", rt_col_name = "RT",
    max_diff_peak2mean = seq(from = 0.01, to = 0.05, by = 0.01),
    min_diff_peak2peak = seq(from = 0.01, to = 0.2, by = 0.01))
save(results_beph, file = "data/results_beph.RData")
```

```
## Load data
load("data/results_bbim.RData")
load("data/results_beph.RData")
load("data/results_bfla.RData")
```

- Estimate error rates

Error rate calculations are executed with a custom function `error_rate` that uses a list of annotated substances as a reference. See the code for details.

```
errors_bbim <- data.frame(p2p = results_bbim[[2]][["p2p"]],
    p2m = results_bbim[[2]][["p2m"]])

errors_bbim[["error"]] <- unlist(lapply(X = results_bbim[[1]],
    error_rate, "data/bbim_ms.txt"))

errors_beph <- data.frame(p2p = results_beph[[2]][["p2p"]],
    p2m = results_beph[[2]][["p2m"]])

errors_beph[["error"]] <- unlist(lapply(X = results_beph[[1]],
    error_rate, "data/beph_ms.txt"))

errors_bfla <- data.frame(p2p = results_bfla[[2]][["p2p"]],
    p2m = results_bfla[[2]][["p2m"]])
```

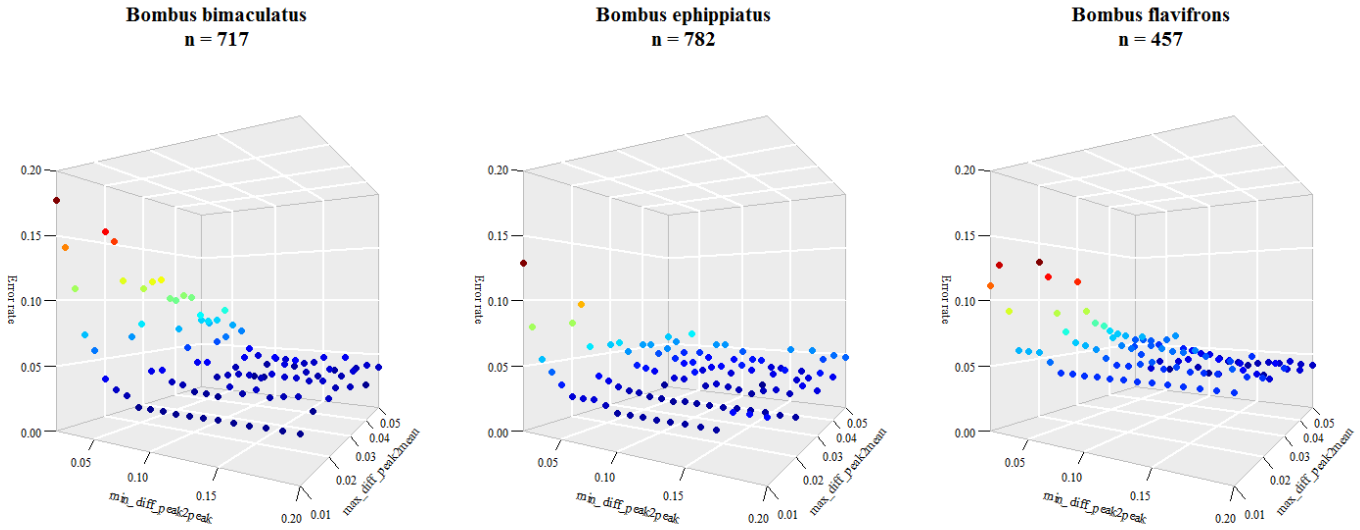

Figure 1: Influence of paramter combinations on the error rate of alignments. Sample size refer to the number of retention times considered.

```
errors_bfla[["error"]] <- unlist(lapply(X = results_bfla[[1]],
    error_rate, "data/bfla_ms.txt"))
```

- Plot results using package `plot3D`

```
## Set up the margins
par(mfrow = c(1, 3), family = "serif", mai = c(0.1, 0.3,
    0.5, 0.15))
## plotting
with(errors_bbim, scatter3D(x = p2p, y = p2m, z = error,
    pch = 19, size = 2, theta = 30, phi = 0, ticktype = "detailed",
    main = "Bombus bimaculatus\nn = 717", xlab = "min_diff_peak2peak",
    ylab = "max_diff_peak2mean", zlab = "Error rate", bty = "g",
    colkey = FALSE, cex = 1.5, cex.lab = 1.25, cex.axis = 1.25,
    cex.main = 2, zlim = c(0, 0.2)))

with(errors_beph, scatter3D(x = p2p, y = p2m, z = error,
    pch = 19, size = 2, theta = 30, phi = 0, ticktype = "detailed",
    main = "Bombus ephippiatus\nn = 782", xlab = "min_diff_peak2peak",
    ylab = "max_diff_peak2mean", zlab = "Error rate", bty = "g",
    colkey = FALSE, cex = 1.5, cex.lab = 1.25, cex.axis = 1.25,
    cex.main = 2, zlim = c(0, 0.2)))

with(errors_bfla, scatter3D(x = p2p, y = p2m, z = error,
    pch = 19, size = 2, theta = 30, phi = 0, ticktype = "detailed",
    main = "Bombus flavifrons\nn = 457", xlab = "min_diff_peak2peak",
    ylab = "max_diff_peak2mean", zlab = "Error rate", bty = "g",
    colkey = FALSE, cex = 1.5, cex.lab = 1.25, cex.axis = 1.25,
    cex.main = 2, zlim = c(0, 0.2)))
```

The exploration of the parameter space shows that the parameter `min_diff_peak2peak` has the greatest

potential to change the error rate and thereby the accuracy of an alignment. Using small values allows a fine scalled grouping of retention times by penalising deviations strongly, but seems not appropriate for the three datasets. Allowing larger differences between unique peaks favours the correct assignment of more variable substances.

- Determining the best parameters

The plot clearly indicates a large area in the parametric space that yields to similarly small errors and therefore demonstrates the robustness of `align_chromatograms`.

```
## combine estimates in a data frame
df <- data.frame(p2p = errors_bbim[["p2p"]], p2m = errors_bbim[["p2m"]],
    bbim = errors_bbim[["error"]], beph = errors_beph[["error"]],
    bfla = errors_bfla[["error"]])
## calculate mean error rates
df[["mean"]] <- apply(df[, 3:5], 1, mean)
## 10 best parameter combinations
head(df[order(df[["mean"]]), ], n = 10)
#>     p2p  p2m      bbim       beph       bfla
#> 71 0.11 0.04 0.0278940 0.03196931 0.03719912
#> 68 0.08 0.04 0.0348675 0.03196931 0.03282276
#> 96 0.16 0.05 0.0320781 0.03069054 0.03938731
#> 10 0.10 0.01 0.0278940 0.02557545 0.05032823
#> 11 0.11 0.01 0.0278940 0.02557545 0.05032823
#> 12 0.12 0.01 0.0278940 0.02557545 0.05032823
#> 13 0.13 0.01 0.0278940 0.02557545 0.05032823
#> 14 0.14 0.01 0.0278940 0.02557545 0.05032823
#> 15 0.15 0.01 0.0278940 0.02557545 0.05032823
#> 16 0.16 0.01 0.0278940 0.02557545 0.05032823
#>          mean
#> 71 0.03235415
#> 68 0.03321986
#> 96 0.03405198
#> 10 0.03459923
#> 11 0.03459923
#> 12 0.03459923
#> 13 0.03459923
#> 14 0.03459923
#> 15 0.03459923
#> 16 0.03459923
```

Simulate the effect of additional noise

The performance of GCalignR is clearly dependend on the quality of the raw data. To show the effect of noise (i.e. bad quality chromatograms) we used again the raw datasets of (Dellicour and Lecocq 2013) and added small errors to a varying proportion of peaks and repeated the error calculation as shown above. This time we choose default values for `min__diff_peak2peak`and `max_diff_peak2mean`that were shown to be robust.

- Aligning the raw datasets with default settings

```
## we align that untreated datasets in order to extract
## input retention times
bbim_zero <- align_chromatograms(data = "data/bbim.txt",
    rt_col_name = "RT")
save(bbim_zero, file = "data/bbim_zero.RData")
```

```
beph_zero <- align_chromatograms(data = "data/beph.txt",
    rt_col_name = "RT")
save(beph_zero, file = "data/beph_zero.RData")
bfla_zero <- align_chromatograms(data = "data/bfla.txt",
    rt_col_name = "RT")
save(bfla_zero, file = "data/bfla_zero.RData")
```

```
load("data/bbim_zero.RData")
load("data/beph_zero.RData")
load("data/bfla_zero.RData")
```

- Convert to lists

```
bfla_chroma <- lapply(bfla_zero[["input_list"]], na.remove)  # remove NAs
bbim_chroma <- lapply(bbim_zero[["input_list"]], na.remove)  # remove NAs
beph_chroma <- lapply(beph_zero[["input_list"]], na.remove)  # remove NAs
```

- Do the simulations
- Dataset *Bombus flavifrons*

```
bfla_out <- sim_linear_shift(bfla_chroma, rt_col_name = "RT",
    shifts = c(-0.03, 0.03))
bfla_shifted <- bfla_out[["Chromas"]]

p <- rep(seq(from = 0, to = 1, by = 0.05), each = 3)
bfla_data <- list()
names <- character()
for (i in 1:length(p)) {
    ## add errors
    temp <- lapply(bfla_shifted, add_peak_error, p = p[i],
        rt_col_name = "RT", conc_col_name = "Area", distr = c(-0.02,
            -0.01, 0.01, 0.02))
    ## extract peak list
    temp <- lapply(temp, FUN = function(x) x[["chroma"]])
    aligned <- align_chromatograms(temp, rt_col_name = "RT",
        max_linear_shift = 0.05)
    ## We need the 'true' retention times for referencing
    ## purposes
    aligned <- original_rt(org = bfla_chroma, aligned = aligned,
        rt_col_name = "RT")
    bfla_data <- append(bfla_data, list(aligned))
    names <- c(names, paste0("no_", as.character(i), "_noise_",
        as.character(p[i])))
}
names(bfla_data) <- names
bfla_simulations <- list(OptAlign = bfla_zero, SimAlign = bfla_data,
    noise = p)
save(x = bfla_simulations, file = paste0("data/", "bfla_simulations",
    ".RData"))
```

- Dataset *Bombus bimaculatus*

```
bbim_out <- sim_linear_shift(bbim_chroma, rt_col_name = "RT",
    shifts = c(-0.03, 0.03))
bbim_shifted <- bbim_out[["Chromas"]]  # linearly shifted sample
```

```
p <- rep(seq(from = 0, to = 1, by = 0.05), each = 3)
bbim_data <- list()
names <- character()
for (i in 1:length(p)) {
    ## add errors
    temp <- lapply(bbim_shifted, add_peak_error, p = p[i],
        rt_col_name = "RT", conc_col_name = "Area", distr = c(-0.02,
            -0.01, 0.01, 0.02))
    ## extract peak list
    temp <- lapply(temp, FUN = function(x) x[["chroma"]])
    aligned <- align_chromatograms(temp, rt_col_name = "RT",
        max_linear_shift = 0.05)
    ## We need the 'true' retention times for referencing
    ## purposes
    aligned <- original_rt(org = bbim_chroma, aligned = aligned,
        rt_col_name = "RT")
    bbim_data <- append(bbim_data, list(aligned))
    names <- c(names, paste0("no_", as.character(i), "_noise_",
        as.character(p[i])))
}
names(bbim_data) <- names
bbim_simulations <- list(OptAlign = bbim_zero, SimAlign = bbim_data,
    noise = p)
save(x = bbim_simulations, file = paste0("data/", "bbim_simulations",
    ".RData"))
```

- Dataset *Bombus ephippiatus*

```
beph_out <- sim_linear_shift(beph_chroma, rt_col_name = "RT",
    shifts = c(-0.03, 0.03))
beph_shifted <- beph_out[["Chromas"]]  # linearly shifted sample

p <- rep(seq(from = 0, to = 1, by = 0.05), each = 3)
beph_data <- list()
names <- character()
for (i in 1:length(p)) {
    ## add errors
    temp <- lapply(beph_shifted, add_peak_error, p = p[i],
        rt_col_name = "RT", conc_col_name = "Area", distr = c(-0.02,
            -0.01, 0.01, 0.02))
    ## extract peak list
    temp <- lapply(temp, FUN = function(x) x[["chroma"]])
    aligned <- align_chromatograms(temp, rt_col_name = "RT",
        max_linear_shift = 0.05)
    ## We need the 'true' retention times for referencing
    ## purposes
    aligned <- original_rt(org = beph_chroma, aligned = aligned,
        rt_col_name = "RT")
    beph_data <- append(beph_data, list(aligned))
    names <- c(names, paste0("no_", as.character(i), "_noise_",
        as.character(p[i])))
}
names(beph_data) <- names
beph_simulations <- list(OptAlign = beph_zero, SimAlign = beph_data,
```

```
    noise = p)
save(x = beph_simulations, file = paste0("data/", "beph_simulations",
    ".RData"))
```

- Estimate errors

```
load("data/bfla_simulations.RData")
load("data/beph_simulations.RData")
load("data/bbim_simulations.RData")
## set up data frames
bfla <- data.frame(data.frame(noise = bfla_simulations[["noise"]]))
bbim <- data.frame(data.frame(noise = bbim_simulations[["noise"]]))
beph <- data.frame(data.frame(noise = beph_simulations[["noise"]]))
## calculate errors
bfla[["error"]] <- unlist(lapply(X = bfla_simulations[["SimAlign"]],
    error_rate, Reference = "data/bfla_ms.txt", rt_col_name = "RT",
    linshift = FALSE))
bbim[["error"]] <- unlist(lapply(X = bbim_simulations[["SimAlign"]],
    error_rate, Reference = "data/bbim_ms.txt", rt_col_name = "RT",
    linshift = FALSE))
beph[["error"]] <- unlist(lapply(X = beph_simulations[["SimAlign"]],
    error_rate, Reference = "data/beph_ms.txt", rt_col_name = "RT",
    linshift = FALSE))
## Combine data into one data frame
df <- rbind(bbim, bfla, beph)
df[["id"]] <- rep(c("bbim", "bfla", "beph"), each = nrow(df)/3)
save(df, file = "data/df.RData")
```

- Plotting results

```
load("data/df.RData")
plot <- ggplot2::ggplot(data = df, aes(x = noise, y = error,
    group = id, col = id)) + geom_smooth(level = 0.95) +
    theme_bw(base_size = 14) + theme(aspect.ratio = 1) +
    xlab("Additional noise level") + ylab("Error rate") +
    scale_x_continuous(breaks = seq(0, 1, 0.1)) + scale_y_continuous(breaks = seq(0,
    0.3, 0.02)) + scale_colour_manual(values = c("#1B9E77",
    "#D95F02", "#7570B3"), name = "", breaks = c("bbim",
    "beph", "bfla"), labels = c("Bombus bimaculatus", "B. ephippiatus",
    "B. flavifrons"), guide = guide_legend(label.theme = element_text(face = "italic",
    angle = 0, size = 11)))
```

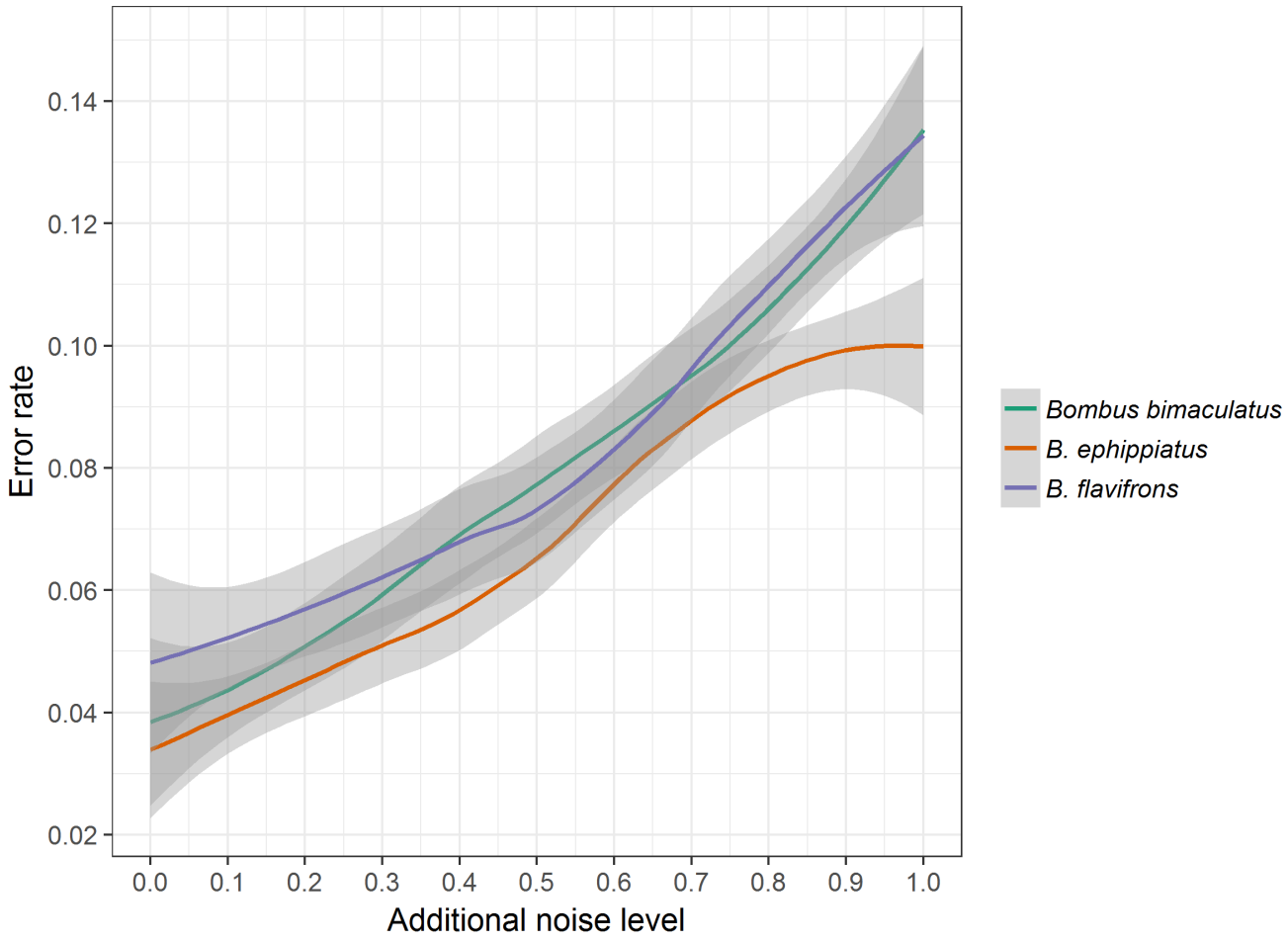

Figure 2: Influence of additional noise levels on the error rate of alignments. Shaded areas around the lines indicate confidence intervals at a level of 0.95

References

Dellicour, Simon, and Thomas Lecocq. 2013. "GCALIGNER 1.0: An Alignment Program to Compute a Multiple Sample Comparison Data Matrix from Large Eco-Chemical Datasets Obtained by Gc." *Journal of Separation Science* 36 (19): 3206–9. doi:10.1002/jssc.201300388.