Setting up for PCA

Wesley Burr

29/03/2022

Redo, with Normalization

Big difference in this file versus 3_Cleanup is that we will use Bromobenzene-normalized inputs instead of non-normalized inputs.

```
##
#  clean_common: Function to select, filter, clean, filter and merge
#  compounds using the logic of:
#  * unique to the samples, not the control, for ALL samples; or
#  * in both samples and control, but much stronger in the control
#
#  Inputs:
#  * dat: data.frame sourced from merging spreadsheets of GCxGC output
#  * sample_names: names of specific samples (e.g., SS5_Foot_1_a)
#  * control_names: names of specific control samples (e.g., SS5_Foot_1_Control_a)
#  * ratio_par: cut-off for the logic of "in both samples and control" - if this
#      is set very large, will eliminate cross-overs.
#
#  Returns:
#  * samples_keep: list of full data.frames for individual replicates, cleaned
#      down to relevant compounds using above logic
##
clean_common <- function(dat,
                         sample_names,
                         control_names,
                         ratio_par = 2.0) {

  samples <- vector("list", length = length(sample_names))
  names(samples) <- sample_names
  controls <- vector("list", length = length(control_names))
  names(controls) <- control_names

  # Extract specific samples and controls of interest and
  # dump all but the largest Area example of each compound
  for(j in 1:length(sample_names)) {
    samples[[j]] <- dat %>% subset(Sample == sample_names[j]) %>%
                      group_by(Name) %>%
                      filter(Area == max(Area)) %>%
                      ungroup() %>% filter(substr(Name, 1, 4) != "Peak")
    samples[[j]] <- samples[[j]][!duplicated(samples[[j]]$Name), ]
  }
  for(j in 1:length(control_names)) {
    controls[[j]] <- dat %>% subset(Sample == control_names[j]) %>%
                      group_by(Name) %>%
                      filter(Area == max(Area)) %>%
                      ungroup() %>% filter(substr(Name, 1, 4) != "Peak")
    controls[[j]] <- controls[[j]][!duplicated(controls[[j]]$Name), ]
  }
  # merge controls
  control <- do.call("rbind", controls)
  control <- control %>% group_by(Name) %>%
                      filter(Area == max(Area)) %>%
                      ungroup()
  control <- control[!duplicated(control$Name), ]

  # Find compounds that are in each sample that are also in control
  samples_keep <- samples
  for(j in 1:length(sample_names)) {
    samp <- samples[[j]] %>% filter(samples[[j]]$Name %in% control$Name)
    cont <- control %>% filter(control$Name %in% samples[[j]]$Name)

    # ratio is high enough to keep
    samp_SN <- unlist(samp[order(samp$Name), "PeakSN"])
    cont_SN <- unlist(cont[order(cont$Name), "PeakSN"])
    contrib1 <- samp %>% filter((samp_SN / cont_SN) > ratio_par)

    # also, compounds that are *not* in the controls
    contrib2 <- samples[[j]] %>% filter(!(samples[[j]]$Name %in% control$Name))
    samples_keep[[j]] <- rbind(contrib1, contrib2)
  }
  names(samples_keep) <- sample_names
  samples_keep
}


##
#
#  join_common: Function which takes output of clean_common above,
#    and merges based on common presence across all replicates of compounds.
#
#  Inputs:
#  * compounds: list of data.frames, 16 columns as in the spreadsheets
#
#  Outputs:
#  * common: merged, simplified data.frame, created via inner_join of data.frames after filtering.
##
join_common <- function(compounds) {
  n_samp <- length(compounds)
  subset_compounds <- vector("list", length = n_samp)
  for(j in 1:n_samp) {
    subset_compounds[[j]] <- compounds[[j]]
    if(n_samp > 1) {
      for(k in (1:n_samp)[-j]) {
        subset_compounds[[j]] <- subset_compounds[[j]] %>%
                                  subset(subset_compounds[[j]]$Name %in% compounds[[k]]$Name)
      }
    }
    subset_compounds[[j]] <- subset_compounds[[j]] %>% select(Name, Area, PeakSN)
  }

  # Join first two, if they exist
  if(n_samp > 1) {
    common <- inner_join(x = subset_compounds[[1]], y = subset_compounds[[2]], by = "Name")
    if(n_samp >= 3) {
      for(j in 3:n_samp) {
        common <- inner_join(x = common, y = subset_compounds[[j]], by = "Name")
      }
    }
  } else {
    common <- subset_compounds[[1]][, c("Name", "Area", "PeakSN")]
  }
  names(common) <- c("Name", paste0(c("Area_", "PeakSN_"), rep(1:n_samp, each = 2)))
  common
}
```

Working Through All the REST DONORS

```
load("/home/wburr/Darshil_Compounds/dat/Normalized_RDA/RH1newn.rda")
RH1 <- RH1newm
RH1$Sample <- str_replace_all(RH1$Sample, " ", "_")

RH1 <- join_common( clean_common(RH1,
                sample_names = c("D0D11010", "D0D11042","D0D11055"),
                control_names = c("RC"),
                ratio_par = 2.0) )
```

Donor 2 has multiple days so separating each Day into its own code and doing same thing for the rest of donors

```
# Load the normalized donor data
load("/home/wburr/Darshil_Compounds/dat/Normalized_RDA/RH2n.rda")
# simple object name
RH2 <- RH2m
# fix the sample space name issue
RH2$Sample <- str_replace_all(RH2$Sample, " ", "_")

#Day 0 - extract, clean, join
RH2_0 <- join_common( clean_common(RH2,
                sample_names = c("D0D21257", "D0D21450","D0D21700"),
                control_names = c("RC"),
                ratio_par = 2.0) )
#DAY 1
RH2_1 <- join_common( clean_common(RH2,
                sample_names = c("D1D2945", "D1D21210","D1D21400", "D1D211600"),
                control_names = c("D1D2RC"),
                ratio_par = 2.0) )
```

```r
#DAY 2
RH2_2 <- join_common( clean_common(RH2,
                sample_names = c("D2D21051", "D2D21248","D2D21450", "D2D21650"),
                control_names = c("D2D2RC"),
                ratio_par = 2.0) )

#DAY 3
RH2_3 <- join_common( clean_common(RH2,
                sample_names = c("D3D2930", "D3D21130","D3D21328", "D3D21529"),
                control_names = c("D3D2RC"),
                ratio_par = 2.0) )

#DAY 4
RH2_4 <- join_common( clean_common(RH2,
                sample_names = c("H2D4R1"),
                control_names = c("D4601934"),
                ratio_par = 2.0) )
```

Continue for the rest of the donors in the same way.

```r
load("/home/wburr/Darshil_Compounds/dat/Normalized_RDA/RH3N1n.rda")
RH3 <- RH3N1m
RH3$Sample <- str_replace_all(RH3$Sample, " ", "_")

#Day 0
RH3_0 <- join_common( clean_common(RH3,
                sample_names = c("D0D3R1", "D0D3R2","D0D3R3"),
                control_names = c("D0D3RC"),
                ratio_par = 2.0) )

#DAY 1 Samples were not collected due to the rainfall during the sampling day
RH3_1 <- NULL

#Day 2
RH3_2 <- join_common( clean_common(RH3,
                sample_names = c("D2D3R1", "D2D3R2","D3D3R3"),
                control_names = c("D2D3RC_2"),
                ratio_par = 2.0) )
#Day 3
RH3_3 <- join_common( clean_common(RH3,
                sample_names = c("D3D3R1", "D3D3R2","D3D3R3"),
                control_names = c("D3D3RC"),
                ratio_par = 2.0) )
#Day 4
RH3_4 <- join_common( clean_common(RH3,
                sample_names = c("D4D3R1", "D4D3R2"),
                control_names = c("D4D3RC"),
                ratio_par = 2.0) )
```

And on and on we go . . .

```r
load("/home/wburr/Darshil_Compounds/dat/Normalized_RDA/RH4n.rda")
RH4 <- RH4m
RH4$Sample <- str_replace_all(RH4$Sample, " ", "_")
```

```r
# DAY 0
RH4_0 <- join_common( clean_common(RH4,
                sample_names = c("D0D4R1", "D0D4R2","D0D4R3"),
                control_names = c("D4RC"),
                ratio_par = 2.0) )
#DAY 1
RH4_1 <- join_common( clean_common(RH4,
                sample_names = c("D1D4R1", "D1D4R2","D1D4R3"),
                control_names = c("D1D4RC"),
                ratio_par = 2.0) )

#Day 2 Sampling was Samples were not collected due to the rainfall during the sampling day
RH4_2 <- NULL

#Day 3
RH4_3 <- join_common( clean_common(RH4,
                sample_names = c("D3D4R1", "D3D4R2","D3D4R3"),
                control_names = c("D3D4RC"),
                ratio_par = 2.0) )

# Day 4, only control collected
RH4_4 <- NULL
```

```r
#Day 0
load("/home/wburr/Darshil_Compounds/dat/Normalized_RDA/RH6n.rda")
RH6 <- RH6m
RH6$Sample <- str_replace_all(RH6$Sample, " ", "_")

#Day 0
RH6_0 <- join_common( clean_common(RH6,
                sample_names = c("D6D0R1", "D6D0R2", "D6D0R3"),
                control_names = c("D6RC"),
                ratio_par = 2.0) )

#Day 1
RH6_1 <- join_common( clean_common(RH6,
                sample_names = c("D6D111_15", "D6D113_10", "D6D114_10"),
                control_names = c("D6D1RC"),
                ratio_par = 2.0) )

#Day 2
RH6_2 <- join_common( clean_common(RH6,
                sample_names = c("D6D211_40", "D6D213_44", "D6D215_30"),
                control_names = c("D6D2RC"),
                ratio_par = 2.0) )

#Day 3
RH6_3 <- join_common( clean_common(RH6,
                sample_names = c("D6D311_41", "D6D313_42", "D6D315_14"),
                control_names = c("D6D3RC"),
                ratio_par = 2.0) )

#Day 4
RH6_4 <- join_common( clean_common(RH6,
                sample_names = c("D6D411_15", "D6D413", "D6D415"),
                control_names = c("D6D4RC"),
                ratio_par = 2.0) )

#Day 5
RH6_5 <- join_common( clean_common(RH6,
                sample_names = c("D6D511_15", "D6D513_45", "D6D515_20"),
                control_names = c("D6D5RC"),
                ratio_par = 2.0) )
```

```
load("/home/wburr/Darshil_Compounds/dat/Normalized_RDA/RH7n.rda")
RH7 <- RH7m
RH7$Sample <- str_replace_all(RH7$Sample, " ", "_")

#Day 0
RH7_0 <- join_common( clean_common(RH7,
                sample_names = c("D0H713_25", "D0H715_30"),
                control_names = c("D1RC"),
                ratio_par = 2.0) )

#Day 1
RH7_1 <- join_common( clean_common(RH7,
                sample_names = c("D1H710_58", "D1H713_21", "D1H715_21"),
                control_names = c("D1RC"),
                ratio_par = 2.0) )

#Day 2
RH7_2 <- join_common( clean_common(RH7,
                sample_names = c("D2H710_45", "D2H713_33", "D2H715_15"),
                control_names = c("D2RC"),
                ratio_par = 2.0) )

#Day 3 Sampling was Samples were not collected due to the rainfall during the sampling day
RH7_3 <- NULL

#Day 4
RH7_4 <- join_common( clean_common(RH7,
                sample_names = c("D4H711_30", "D4H713_40", "D4H715_10"),
                control_names = c("D4RC"),
                ratio_par = 2.0) )
#Day 5
RH7_5 <- join_common( clean_common(RH7,
                sample_names = c("D5H79_45", "D5H712_45", "D5H714_45"),
                control_names = c("D5RC"),
                ratio_par = 2.0) )
```

```
load("/home/wburr/Darshil_Compounds/dat/Normalized_RDA/RH8n.rda")
RH8 <- RH8m
RH8$Sample <- str_replace_all(RH8$Sample, " ", "_")

#Day 0
RH8_0 <- join_common( clean_common(RH8,
                sample_names = c("H8D0R1","H8D0R2"),
                control_names = c("H8D0RC"),
                ratio_par = 2.0) )
#Day 1
RH8_1 <- join_common( clean_common(RH8,
                sample_names = c("H8D1R1","H8D1R2", "H8D1R3"),
                control_names = c("H8D1RC"),
                ratio_par = 2.0) )
#Day 2
RH8_2 <- join_common( clean_common(RH8,
                sample_names = c("H8D2R1","H8D2R2", "H8D2R3"),
                control_names = c("D2H8RC"),
                ratio_par = 2.0) )
#Day 3
RH8_3 <- join_common( clean_common(RH8,
                sample_names = c("H8D3R1","H8D3R2", "H8D3R3"),
                control_names = c("H8D3RC"),
                ratio_par = 2.0) )
#Day 4
RH8_4 <- join_common( clean_common(RH8,
#                sample_names = c("RD_RC_Tube_3_AM_sample", "H8D4R3"),  #### NOTE: nothing survives fro
                sample_names = "H8D4R3",
                control_names = c("H8D4RC"),
                ratio_par = 2.0) )
#Day 5
RH8_5 <- join_common( clean_common(RH8,
                sample_names = c("H8D5R1"),
                control_names = c("H8D5RC"),
                ratio_par = 2.0) )
```

```
load("/home/wburr/Darshil_Compounds/dat/Normalized_RDA/RH9n.rda")
RH9 <- RH9m
RH9$Sample <- str_replace_all(RH9$Sample, " ", "_")

#
#  Special fixes: the names aren't consistent!
#
names(RH9) <- c("Peak3", "PeakNum", "Sample", "Name",
                "Formula", "RTs", "Similarity", "Area",
                "Height", "QuantMass", "BaseMass", "QuantSN",
                "PeakSN", "RT1", "RT2")

#Day 0
RH9_0 <- join_common( clean_common(RH9,
                sample_names = c("D0H910_55", "D0H913_18", "D0H915_17"),
                control_names = c("D0RC"),
                ratio_par = 2.0) )

#Day 1
RH9_1 <- join_common( clean_common(RH9,
                sample_names = c("D1H910_36", "D1H913_12", "D1H915_34"),
                control_names = c("D1RC_(2)"),
                ratio_par = 2.0) )

#Day 2
RH9_2 <- join_common( clean_common(RH9,
                sample_names = c("D2H910_53", "D2RC13_32", "D2RC15_11"),
                control_names = c("D2RC"),
                ratio_par = 2.0) )

#Day 3
RH9_3 <- join_common( clean_common(RH9,
                sample_names = c("D3H911_1", "D3H913_25", "D3H915_21"),
                control_names = c("D3RC"),
                ratio_par = 2.0) )

#Day 4
RH9_4 <- join_common( clean_common(RH9,
                sample_names = c("D4H910_50", "D4H913_18", "D4H915_35"),
                control_names = c("D4RC"),  # cheating here? Nooooooo just an copy paste error...noobie
                ratio_par = 2.0) )

#Day 5
RH9_5 <- join_common( clean_common(RH9,
                sample_names = c("D5H910:58"),
                control_names = c("D5RC"),
                ratio_par = 2.0) )
```

Darshil Review

```
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor1_Filtered.csv", x = RH1)
# Donor 2, days 0 through 4
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor2_Filtered_0.csv", x = RH2_0)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor2_Filtered_1.csv", x = RH2_1)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor2_Filtered_2.csv", x = RH2_2)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor2_Filtered_3.csv", x = RH2_3)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor2_Filtered_4.csv", x = RH2_4)
# Donor 3, days 0, 2-4
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor3_Filtered_0.csv", x = RH3_0)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor3_Filtered_2.csv", x = RH3_2)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor3_Filtered_3.csv", x = RH3_3)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor3_Filtered_4.csv", x = RH3_4)
# Donor 4, days 0, 1, 3
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor4_Filtered_0.csv", x = RH4_0)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor4_Filtered_1.csv", x = RH4_1)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor4_Filtered_3.csv", x = RH4_3)
# Donor 6, days 0-5
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor6_Filtered_0.csv", x = RH6_0)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor6_Filtered_1.csv", x = RH6_1)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor6_Filtered_2.csv", x = RH6_2)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor6_Filtered_3.csv", x = RH6_3)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor6_Filtered_4.csv", x = RH6_4)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor6_Filtered_5.csv", x = RH6_5)
# Donor 7, days 0-2,4,5
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor7_Filtered_0.csv", x = RH7_0)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor7_Filtered_1.csv", x = RH7_1)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor7_Filtered_2.csv", x = RH7_2)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor7_Filtered_4.csv", x = RH7_4)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor7_Filtered_5.csv", x = RH7_5)
# Donor 8, days 0-5
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor8_Filtered_0.csv", x = RH8_0)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor8_Filtered_1.csv", x = RH8_1)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor8_Filtered_2.csv", x = RH8_2)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor8_Filtered_3.csv", x = RH8_3)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor8_Filtered_4.csv", x = RH8_4)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor8_Filtered_5.csv", x = RH8_5)
# Donor 9, days 0-4
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor9_Filtered_0.csv", x = RH9_0)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor9_Filtered_1.csv", x = RH9_1)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor9_Filtered_2.csv", x = RH9_2)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor9_Filtered_3.csv", x = RH9_3)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor9_Filtered_4.csv", x = RH9_4)
write.csv(file = "/home/wburr/Darshil_Compounds/REST_Review/Donor9_Filtered_5.csv", x = RH9_5)
```

Combine Them All Into a List

```
all_SS <- vector("list", 35)  # total number of donor+day

# DONOR 1
all_SS[[1]] <- RH1
# DONOR 2
all_SS[[2]] <- RH2_0
all_SS[[3]] <- RH2_1
all_SS[[4]] <- RH2_2
all_SS[[5]] <- RH2_3
all_SS[[6]] <- RH2_4
# DONOR 3
all_SS[[7]]  <- RH3_0
all_SS[[8]]  <- RH3_2
all_SS[[9]]  <- RH3_3
all_SS[[10]] <- RH3_4
# DONOR 4
all_SS[[11]]  <- RH4_0
all_SS[[12]]  <- RH4_1
all_SS[[13]]  <- RH4_3
# DONOR 6
all_SS[[14]]  <- RH6_0
all_SS[[15]]  <- RH6_1
all_SS[[16]]  <- RH6_2
all_SS[[17]]  <- RH6_3
all_SS[[18]]  <- RH6_4
all_SS[[19]]  <- RH6_5
# DONOR 7
all_SS[[20]]  <- RH7_0
all_SS[[21]]  <- RH7_1
all_SS[[22]]  <- RH7_2
all_SS[[23]]  <- RH7_4
all_SS[[24]]  <- RH7_5
# DONOR 8
all_SS[[25]]  <- RH8_0
```

```r
all_SS[[26]]  <- RH8_1
all_SS[[27]]  <- RH8_2
all_SS[[28]]  <- RH8_3
all_SS[[29]]  <- RH8_4
all_SS[[30]]  <- RH8_5
# DONOR 9
all_SS[[31]]  <- RH9_0
all_SS[[32]]  <- RH9_1
all_SS[[33]]  <- RH9_2
all_SS[[34]]  <- RH9_3
all_SS[[35]]  <- RH9_4
all_SS[[36]]  <- RH9_5

names(all_SS) <- c("D1",
                   "D2_d0", "D2_d1", "D2_d2", "D2_d3", "D2_d4",
                   "D3_d0", "D3_d2", "D3_d3", "D3_d4",
                   "D4_d0", "D4_d1", "D4_d3",
                   "D6_d0", "D6_d1", "D6_d2", "D6_d3", "D6_d4", "D6_d5",
                   "D7_d0", "D7_d1", "D7_d2", "D7_d4", "D7_d5",
                   "D8_d0", "D8_d1", "D8_d2", "D8_d3", "D8_d4", "D8_d5",
                   "D9_d0", "D9_d1", "D9_d2", "D9_d3", "D9_d4", "D9_d5")
# add more names as you go!!

save(file = "./all_SS_PCA.rda", all_SS)
# This cleans the environment and reloads ONLY this object to continue
rm(list = ls())
load("./all_SS_PCA.rda")
```

Cleanup Noted Compounds Throughout

There are a number of compounds that have to do with the column, the reference standard (somehow coming through), and other definitely not-related things. We will strip them out now, in preparation for PCA.

Remove Explicitly

The following compounds are definitely not related to decomp, and are related to the process or environment, and need to be removed:

- Bromobenzene (or Benzene, bromo)
- Oxygen
- Acetone
- Methyl alcohol / Methanol
- Carbon dioxide

Before we start that, we'll make a list of the actual "appearances" of compounds, and then check against this to determine the actual filtering arguments (e.g., first 8 characters or full name, etc.).

```r
all_compounds <- sort(unique(
  unlist(lapply(all_SS, FUN = function(x) { unlist(x$Name) }))))
length(all_compounds)
```

```
## [1] 1438
```

So there are 1438 unique compounds present across the 35 total samples, after Controlling (but with the 2.0 ratio argument in place). Let's look for each of the above compounds:

```r
loc1 <- grep(pattern = "Benzene, bromo-", all_compounds,
             ignore.case = TRUE)
```

```r
loc2 <- grep(pattern = "oxygen", all_compounds, ignore.case = TRUE)
# all_compounds[loc2]
```

```r
loc3 <- grep(pattern = "acetone", all_compounds, ignore.case = TRUE)
# all_compounds[loc3]
```

```r
loc4 <- grep(pattern = "methyl alc", all_compounds, ignore.case = TRUE)
# all_compounds[loc4]
loc5 <- grep(pattern = "methanol, TMS", all_compounds,
             ignore.case = TRUE)
# all_compounds[loc5]
```

```r
loc6 <- grep(pattern = "carbon diox", all_compounds, ignore.case = TRUE)
# all_compounds[loc5]
```

Put them together as indexes, then extract these from the list of compounds as actual names.

```r
remove_specifics <- c(loc1, loc2, loc3, loc4, loc5, loc6)
remove1 <- all_compounds[remove_specifics]
```

Remove Via Keyword

There are three keywords that show up that we should also strip out:

- Sil
- TMS
- TBDMS

Let's grab these now:

```r
loc1 <- grep(pattern = "Sil", all_compounds, ignore.case = TRUE)
# all_compounds[loc1]
loc2 <- grep(pattern = "TMS", all_compounds)
# all_compounds[loc2]
loc3 <- grep(pattern = "TBDMS", all_compounds)
# all_compounds[loc3]
remove2 <- all_compounds[c(loc1, loc2, loc3)]
```

Merge Compounds

We need some logic to look for things that are the same compound, but different in only stereochemistry. The indicator seems to be brackets: (E), (Z), (S), popping up in one or more of the variants. So there might be, for example, 2-Octene, (E)- as a compound, and then another sample might have 2-Octene, (Z)-. These should just be merged.

Let's try to look for them first:

```r
loc1 <- grep(pattern = "\\(E\\)-$", all_compounds)
loc2 <- grep(pattern = "\\(Z\\)-$", all_compounds)
loc3 <- grep(pattern = "\\(S\\)-$", all_compounds)
```

```
loc4 <- grep(pattern = "\\(R\\)-$", all_compounds)
to_clean <- c(loc1, loc2, loc3, loc4)
```

Now, the tricky bit: how to fix this up. What we want are these compounds, and their corresponding compounds which **don't** have the (S), (Z), (R) or (E); or have a different one. In all cases, we'll merge them down to the **doesn't have brackets** version if it exists, or if it doesn't, we'll make one.

```
mappings <- data.frame(Original = NA, Transformed = NA)
for(j in 1:length(to_clean)) {
  orig <- all_compounds[to_clean[j]]
  fixed <- strsplit(orig, "\\(")[[1]][1]
  fixed <- substr(fixed, 1, nchar(fixed) - 2)
  mappings[j, ] <- c(orig,
                     fixed)
}
```

Back to the Original Data, Ready to Rock & Roll

So we have remove1 - compounds to remove. We have remove2 - more compounds to remove. And we have mappings, which have compounds that need to be renamed. Then, at the end, we need to check for duplicates, because the renaming may have resulted in more than one compound surviving in a single sample due to the stereochemistry issue.

```
test <- lapply(all_SS, FUN = function(x) {
    y <- x %>% filter(!(Name %in% remove1 | Name %in% remove2))
    which_rows <- which(y$Name %in% mappings$Original)
    if(length(which_rows) > 0) {
      for(j in 1:length(which_rows)) {
        orig <- unlist(y[which_rows[j], "Name"])
        y[which_rows[j], "Name"] <- mappings[mappings$Original == orig,
                                             "Transformed"]
      }
    }
    y
  })
```

Now, look for duplicates, and remove if any now exist:

```
test <- lapply(test, FUN = function(x) {
    dupes <- which(duplicated(x$Name))
    if(length(dupes) > 0) {
      x[-dupes, ]
    } else {
      x
    }
  })
```

We're done! All fixed up. Let's write this back out to an Excel file for Darshil to take a look at.

```
library("xlsx")
names(test)[1]
```

```
## [1] "D1"
```

```
write.xlsx(x = test[[1]],
           file = "/home/wburr/Darshil_Compounds/all_REST_cleaned.xlsx",
           sheetName = names(test)[1],
           col.names = TRUE,
           row.names = TRUE,
           append = FALSE)
for(j in 2:length(test)) {
  write.xlsx(x = test[[j]],
             file = "/home/wburr/Darshil_Compounds/all_REST_cleaned.xlsx",
             sheetName = names(test)[j],
             col.names = TRUE,
             row.names = TRUE,
             append = TRUE)
}
```

Extract Top Compounds

```
compounds <- unlist(lapply(test, FUN = function(x) { x$Name }))
summary_table <- table(compounds)
summary_table <- sort(summary_table, decreasing = TRUE)
top_compounds <- summary_table[1:(min(which(summary_table < 2))-1)]
output_df <- data.frame(top_compounds)
names(output_df) <- c("Compound", "Frequency (out of 35)")
write.csv(file = "./top_compounds.csv", output_df,
          row.names = FALSE)
```

Darshil also wanted to know which samples they appeared in, which is less easy to do. We have the compound names at least, so now we need to create 35 variables to track things, and then fill it in.

```
output_df2 <- as.data.frame(matrix(data = NA, nrow = nrow(output_df),
                                   ncol = 36))
names(output_df2) <- names(test)
output_df2 <- cbind(output_df, output_df2)
for(j in 1:nrow(output_df)) {
    comp <- output_df[j, "Compound"]
    fit <- unlist(lapply(test, FUN = function(x) {
      if(comp %in% x$Name) { 1 } else { 0 }}))
    output_df2[j, -(1:2)] <- fit
}
write.csv(file = "./top_compounds_withSpecifics.csv", output_df2,
          row.names = FALSE)
```

Back to PCA

Now, the objective is to create "blanked" vectors, with 0s inserted in the spots where detection was not found. This requires a full list of unique compounds (there's a LOT - 1357!), then we PCA away.

```
unique_compounds <- sort(unique(
      unlist(lapply(test, FUN = function(x) { x$Name }))))
```

Then, let's create a set of vectors, 37 in total (36 samples, 1 name list).

```
pca_dat <- as.data.frame(matrix(data = 0.0, nrow = length(unique_compounds),
                                ncol = 37)) #darshil change this value
pca_dat[, 1] <- unique_compounds
```

```r
names(pca_dat) <- c("Name", names(test))
for(j in 1:length(test)) {
  x <- test[[j]]
  x_names <- x$Name
  x_area <- apply(x, MAR = 1, FUN = function(y) {
    z <- y[seq(2, length(y), 2)];
    z <- as.numeric(z);
    mean(z) })
 for(k in 1:length(x_names)) {
    pca_dat[pca_dat$Name == x_names[k], j+1] <- x_area[k]
  }
  #pca_dat[pca_dat$Name %in% x_names, j+1] <- x_area
}
extracting <- pca_dat[, -1]
extracting <- apply(extracting, MAR = 2, FUN = function(x) { as.numeric(x) })
row.names(extracting) <- pca_dat$Name
pca_dat <- extracting

write.csv(file = "pca_ready_REST.csv", pca_dat)
```