NMF

William Zhang, Eva, Jerry

2024-09-01

```r
# load the packages
library(NMF)
library(tidyverse)
library(gridExtra)
library(readxl)
```

Procedure

1. Remove hourly observation with missing observation for any chemical
2. Remove background noise level using min values (except for chemicals with minimum value < 2*LOD and maximum value > 100*LOD)
3. Zero values are converted to a random value between 0 and 0.5*LOD
4. Normalize using 0th & 99th quantile
5. Compute weight matrix according to Guha's paper, without LOQ

Reading the data

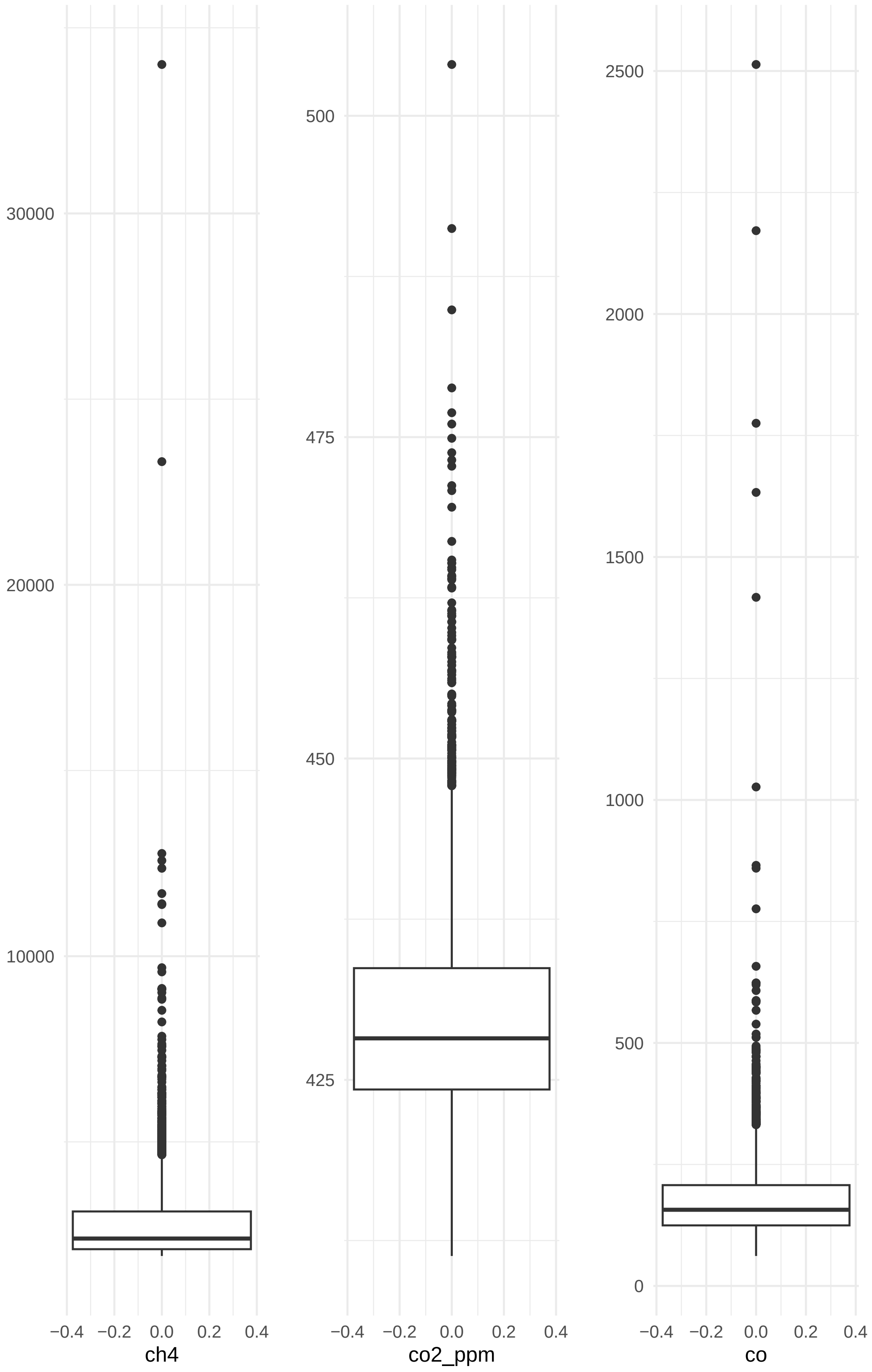
```r
# read the radon data
# Old:
# hourly_radon <- readRDS("hourly_radon.rds")
# New:
hourly_data <- readRDS("../DataProcessing/Trailer_hourly_merge_20240905.rds")
```

```r
# remove NAs
hourly_nona <- hourly_data %>% select(-c(temp_bb,rhi, esf_bb, distToLovi,inv_dist,
                                       distToLovi_wells, monthly_oil, monthly_gas)) %>% na.omit()

vocs <- c("ethane", "ethene", "propane", "propene",
                                      "1_3-butadiene", "i-butane", "n-butane",
                                      "acetylene", "cyclopentane", "i-pentane",
                                      "n-pentane", "n-hexane", "isoprene", "n-heptane",
                                      "benzene", "n-octane", "toluene", "ethyl-benzene",
                                      "m&p-xylene", "o-xylene")
# retrieving the vocs, removing everything else except the vocs
hourly_vocs <- hourly_nona %>% select(any_of(vocs))

# retrieving the non-vocs: co2_ppm, nox, ch4, h2s, so2, o3
# double check this
non_vocs <- c('ch4', 'co2_ppm', 'co', 'h2s', 'so2', 'nox', 'o3')
hourly_non_vocs <- hourly_nona %>% select(all_of(non_vocs))

hourly_full_nona <- cbind(hourly_non_vocs, hourly_vocs)
```

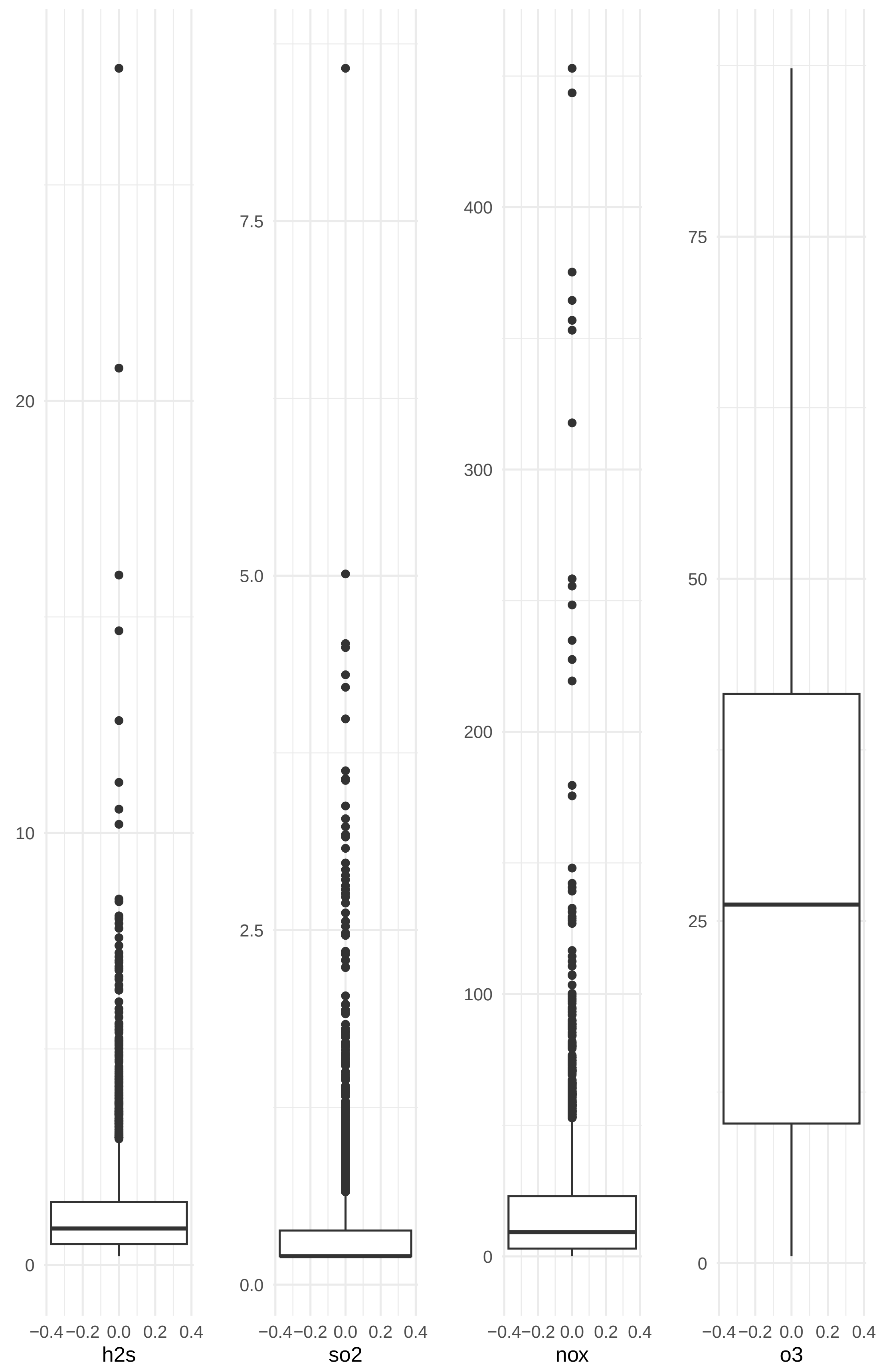
```r
# retrive a vector of yearmonth
hourly_dates <- hourly_nona %>%
  mutate(yearmonth = substring(day, 0, 7)) %>%
  pull(yearmonth)
```

Data visualisation

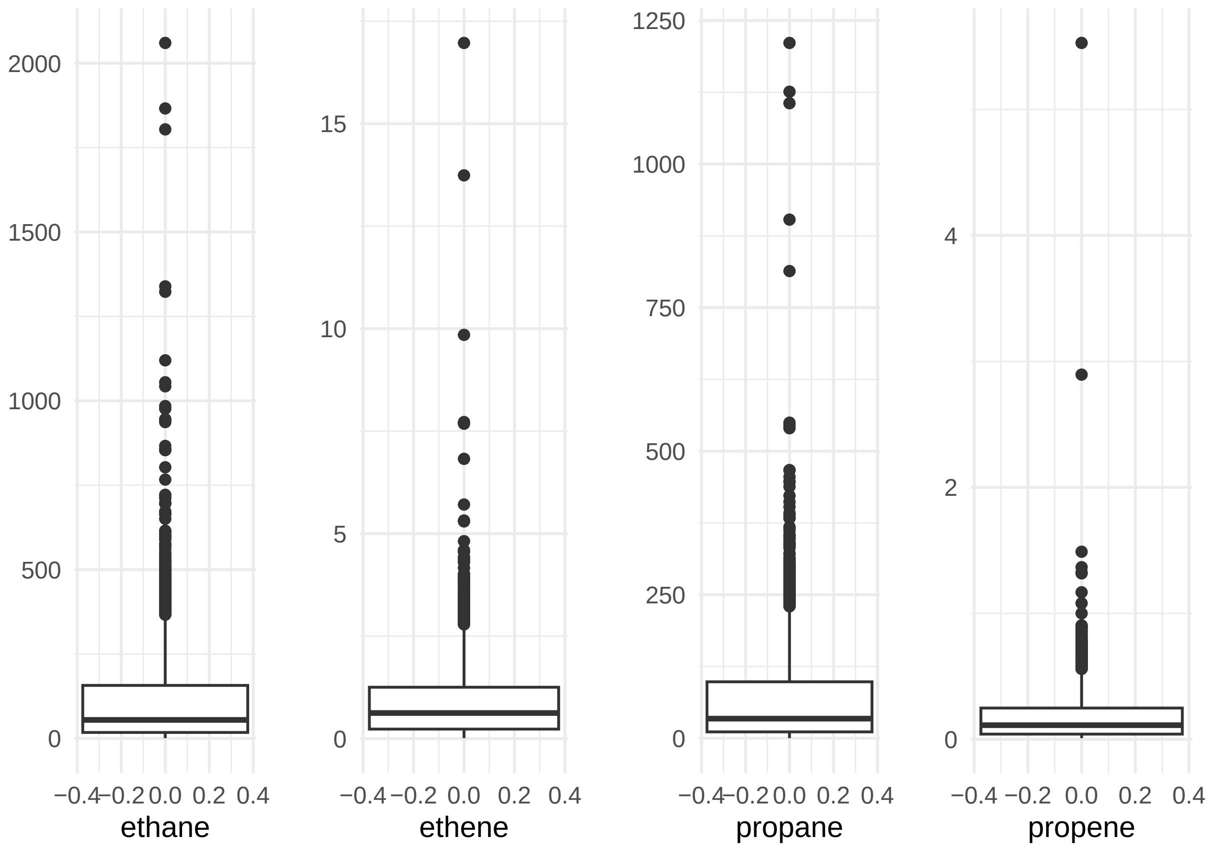
```r
for (compound in non_vocs) {
  assign(paste0(compound, '_boxplot'),
         ggplot(hourly_non_vocs) +
           geom_boxplot(aes(y = .data[[compound]])) +
           labs(x = compound, y = '') +
           theme_minimal())
}

grid.arrange(ch4_boxplot, co2_ppm_boxplot, co_boxplot, nrow = 1)
```


30000
20000
10000
−0.4 −0.2 0.0 0.2 0.4
ch4
500
475
450
425
−0.4 −0.2 0.0 0.2 0.4
co2_ppm
2500
2000
1500
1000
500
0
−0.4 −0.2 0.0 0.2 0.4
co

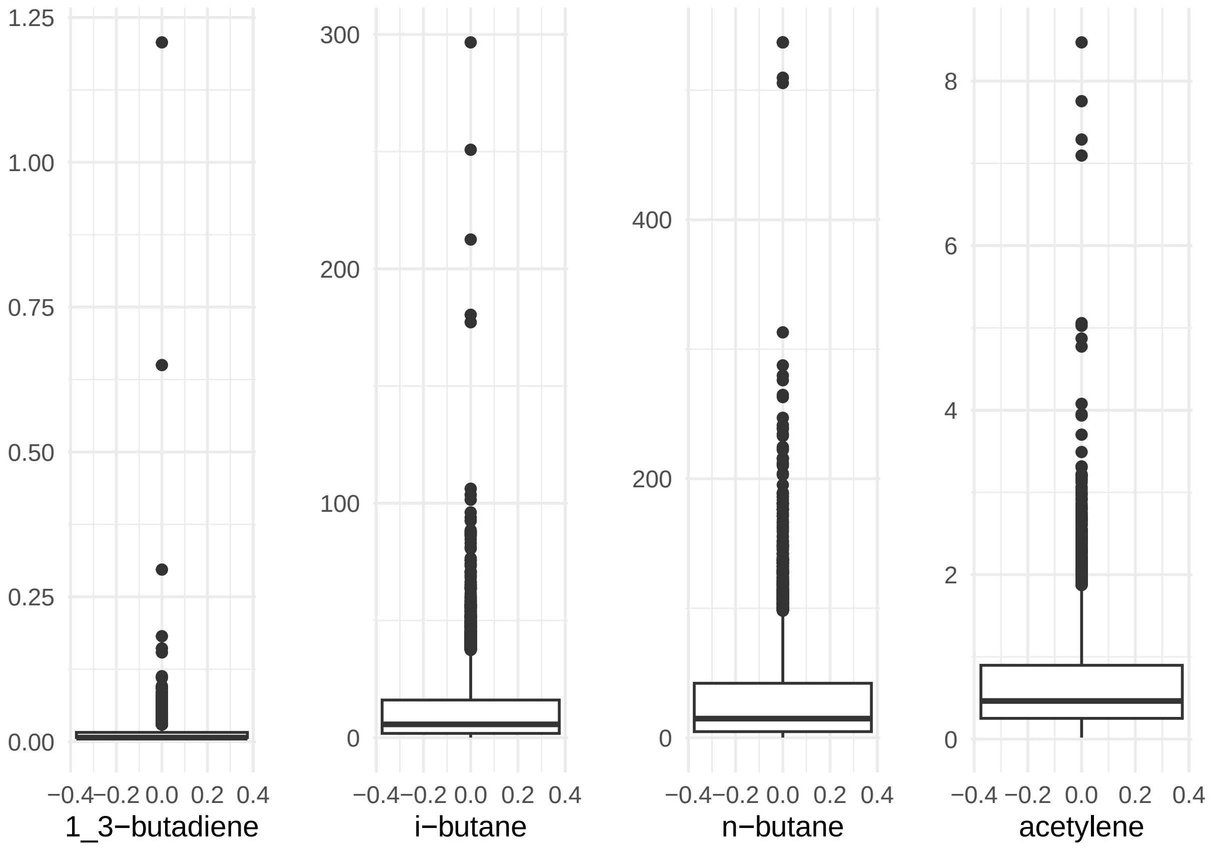
```r
grid.arrange(h2s_boxplot, so2_boxplot, nox_boxplot, o3_boxplot, nrow = 1)
```


20
10
0
7.5
5.0
2.5
0.0
400
300
200
100
0
75
50
25
0
−0.4 −0.2 0.0 0.2 0.4
h2s
so2
nox
o3

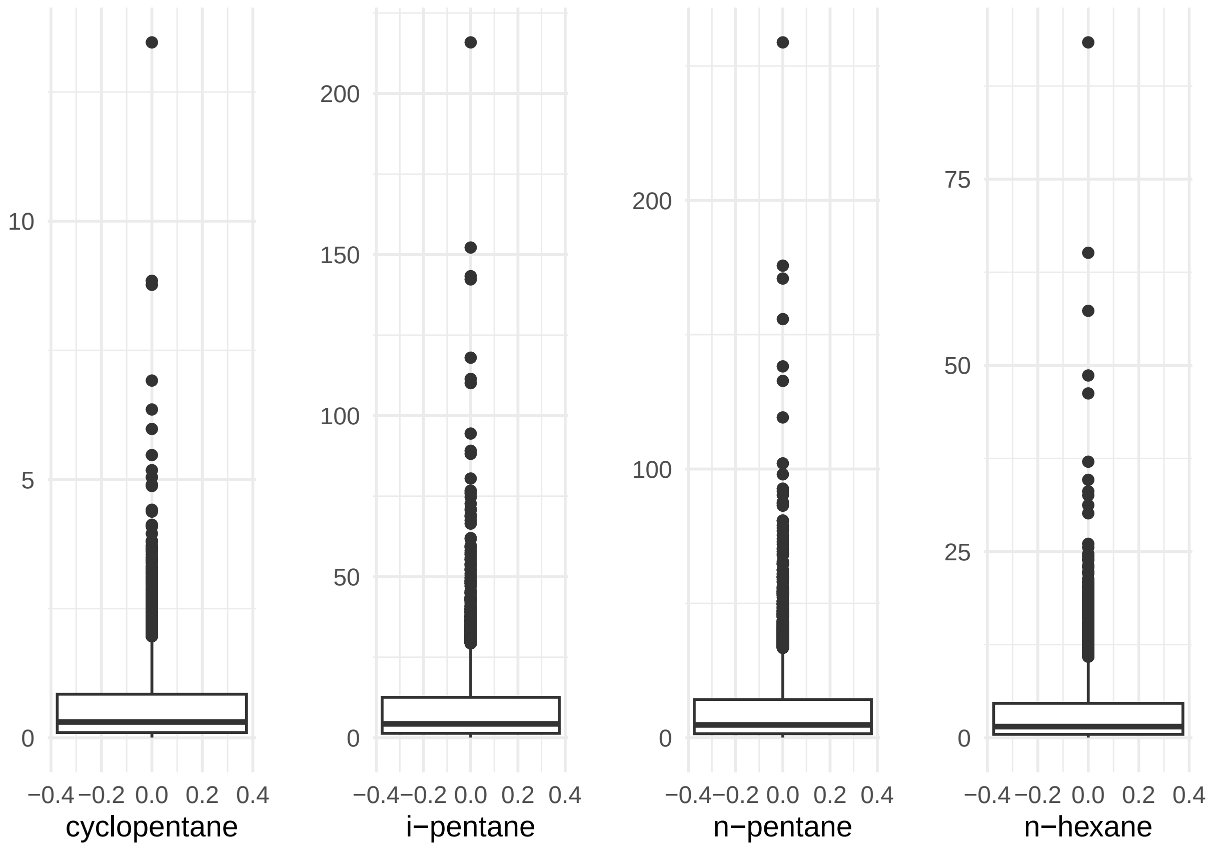
```r
for (compound in vocs) {
  assign(paste0(compound, '_boxplot'),
         ggplot(hourly_vocs) +
           geom_boxplot(aes(y = .data[[compound]])) +
           labs(x = compound, y = '') +
           theme_minimal())
}
grid.arrange(get(paste0(vocs[1], '_boxplot')), get(paste0(vocs[2], '_boxplot')),
             get(paste0(vocs[3], '_boxplot')), get(paste0(vocs[4], '_boxplot')), nrow = 1)
```



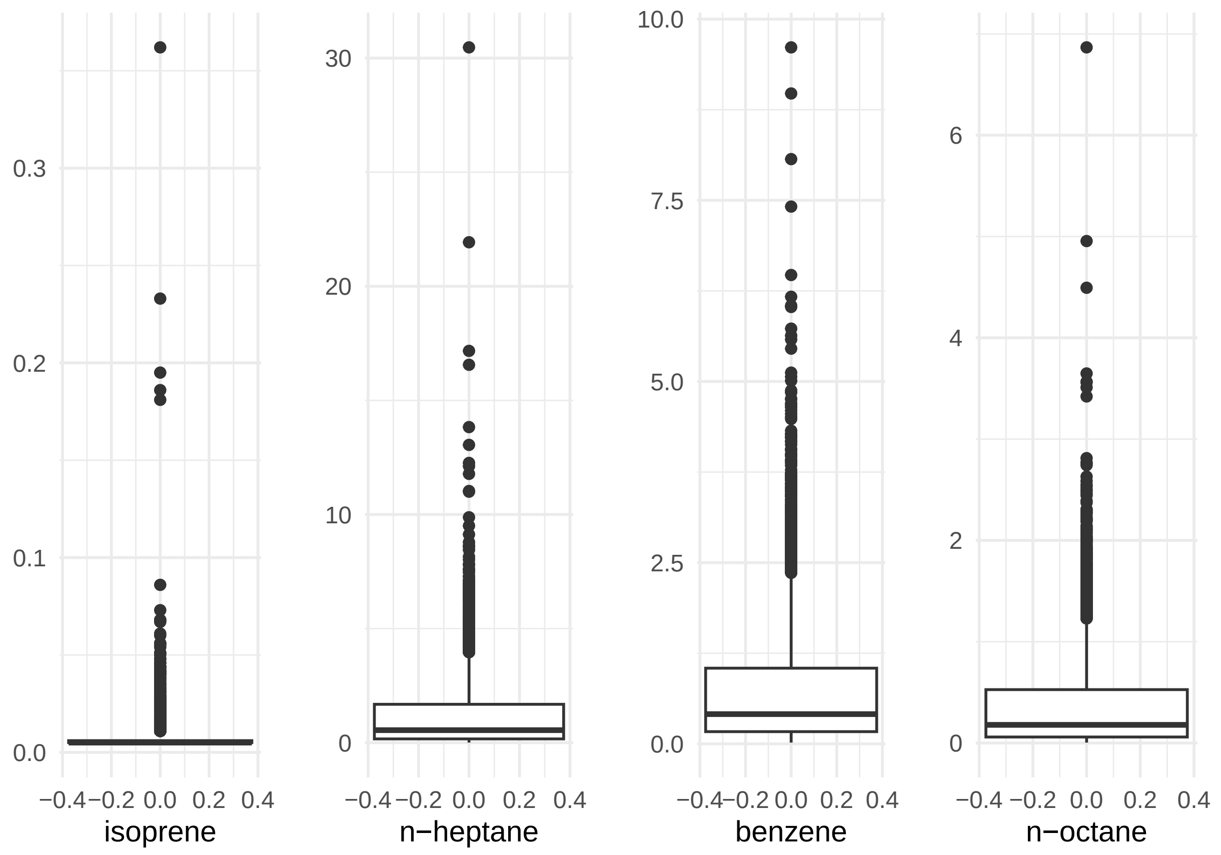
```r
grid.arrange(get(paste0(vocs[5], '_boxplot')), get(paste0(vocs[6], '_boxplot')),
             get(paste0(vocs[7], '_boxplot')), get(paste0(vocs[8], '_boxplot')), nrow = 1)
```



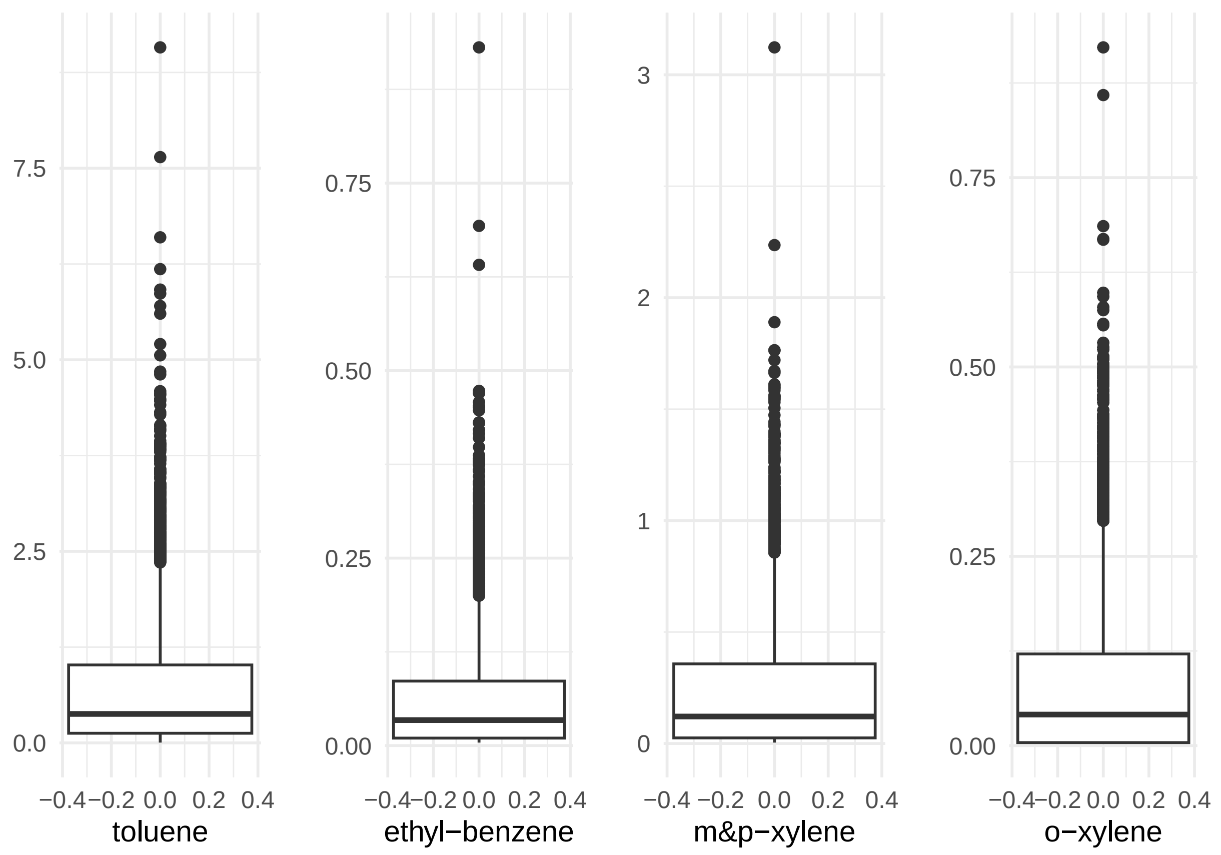
```
grid.arrange(get(paste0(vocs[9], '_boxplot')), get(paste0(vocs[10], '_boxplot')),
             get(paste0(vocs[11], '_boxplot')), get(paste0(vocs[12], '_boxplot')), nrow = 1)
```



```
grid.arrange(get(paste0(vocs[13], '_boxplot')), get(paste0(vocs[14], '_boxplot')),
             get(paste0(vocs[15], '_boxplot')), get(paste0(vocs[16], '_boxplot')), nrow = 1)
```



```r
grid.arrange(get(paste0(vocs[17], '_boxplot')), get(paste0(vocs[18], '_boxplot')),
             get(paste0(vocs[19], '_boxplot')), get(paste0(vocs[20], '_boxplot')), nrow = 1)
```



Data preprocessing

```r
# Define LOD for each chemical
LOD_non_voc <- c('ch4' = 0.9,
             'co2_ppm' = 0.0433,
             'co' = 40,
             'h2s' = 0.4,
             'so2' = 0.4,
             'nox' = 0.05,
             'o3' = 1)

LOD_voc_monthly <- read_csv('../data/LNM_VOC_LOD_Rounded.csv') %>% select(-1)
# extract the yearmonth from date variables
LOD_voc_monthly <- LOD_voc_monthly %>%
  mutate(yearmonth = strftime(as.POSIXct(start_date, format = '%Y-%m-%d %H:%M:%S', tz = 'UTC'), '%Y-%m'

LOD_voc_monthly <- LOD_voc_monthly %>%
  select(-c(start_date, end_date)) %>%
  select(!any_of(ends_with('half_ldl')))

colnames(LOD_voc_monthly) <- str_replace_all(names(LOD_voc_monthly), '_ldl', '')

LOD_voc_avg <- read_xlsx('../data/LNM_VOC_Uncertainties.xlsx', skip = 1)
LOD_voc_avg <- LOD_voc_avg %>%
  select(1, 4) %>%
  rename('LOD' = 2, 'chemical' = 1) %>%
  head(20)
```

```
# find the min for background-levels
background_levels <- sapply(hourly_full_nona, min)
background_levels
```

```
##           ch4      co2_ppm           co          h2s          so2
##      1928.000      411.300       61.630        0.200        0.200
##           nox           o3       ethane       ethene      propane
##         0.025        0.500        0.916        0.011        0.224
##       propene 1_3-butadiene     i-butane     n-butane    acetylene
##         0.009        0.007        0.035        0.090        0.019
##  cyclopentane     i-pentane    n-pentane     n-hexane     isoprene
##         0.005        0.038        0.042        0.021        0.005
##     n-heptane      benzene     n-octane      toluene ethyl-benzene
##         0.004        0.017        0.004        0.004        0.004
##    m&p-xylene     o-xylene
##         0.004        0.004
```

```
get_info <- function(column) {
  N <- length(column)
  background <- quantile(column, 0)
  quantile1 <- quantile(column, 0.01)
  quantile99 <- quantile(column, 0.99)
  n_background <- sum(column == background)
  max <- max(column)
  return(c(N, quantile1, quantile99, max, background, n_background))
}

info_table <- hourly_full_nona %>%
  reframe(across(everything(), ~ get_info(.x)))

info_table <- info_table %>%
  mutate(rownames = c('N', '1st percentile', '99th percentile', 'Max', 'Background', '# Background')) %
  pivot_longer(-rownames) %>%
  pivot_wider(names_from = rownames, values_from = value)

knitr::kable(info_table)
```

name	N	1st percentile	99th percentile	Max	Background	# Background
ch4	4497	1963.40000	6318.81200	34010.900	1928.000	1
co2_ppm	4497	417.09000	457.87120	503.990	411.300	1
co	4497	84.90720	444.04320	2513.440	61.630	1
h2s	4497	0.20000	5.18084	27.700	0.200	777
so2	4497	0.20000	1.83896	8.578	0.200	3065
nox	4497	0.22700	92.01080	452.959	0.025	2
o3	4497	0.50000	72.11200	87.300	0.500	255
ethane	4497	1.80852	536.67200	2060.000	0.916	1
ethene	4497	0.01100	3.52212	16.970	0.011	163
propane	4497	0.81700	305.54000	1211.000	0.224	1
propene	4497	0.00900	0.70228	5.528	0.009	401
1_3-butadiene	4497	0.00700	0.05904	1.207	0.007	3126

name	N	1st percentile	99th percentile	Max	Background	# Background
i-butane	4497	0.14496	63.53760	296.600	0.035	1
n-butane	4497	0.34792	171.37600	536.900	0.090	1
acetylene	4497	0.04900	2.66204	8.471	0.019	1
cyclopentane	4497	0.00500	3.12356	13.460	0.005	96
i-pentane	4497	0.10396	51.02080	215.900	0.038	1
n-pentane	4497	0.10300	58.10280	258.800	0.042	1
n-hexane	4497	0.04196	18.32640	93.360	0.021	2
isoprene	4497	0.00500	0.03204	0.362	0.005	2815
n-heptane	4497	0.01500	6.58924	30.470	0.004	5
benzene	4497	0.02700	3.87512	9.610	0.017	3
n-octane	4497	0.00400	2.01452	6.867	0.004	100
toluene	4497	0.01296	3.53640	9.077	0.004	11
ethyl-benzene	4497	0.00400	0.31604	0.931	0.004	898
m&p-xylene	4497	0.00400	1.31824	3.123	0.004	814
o-xylene	4497	0.00400	0.45912	0.922	0.004	1266

```r
#adjustments that were made according to paper
#William: I'm guessing this refers to Gunnar's paper section 2.2 and Guha 3.3
adjusting_neg_bg_from_lod <- function(chemical, LOD, background, hourly_data){
    # get min and max
    min_value <- min(hourly_data[chemical], na.rm = TRUE)
    max_value <- max(hourly_data[chemical], na.rm = TRUE)
    # if min less than double LOD or max > 100 times LOD
    # adjust to -100 (for entire column???)
    if (min_value < 2 * LOD & max_value > 100 * LOD ){
      return (0)
    }
  return (background)
}
```

```r
# Check if background is negligible for non voc
# merge background and LOD
background_lod_non_voc <- tibble(chemical = non_vocs,
                                 LOD = LOD_non_voc,
                                 background = unname(background_levels[non_vocs]))
adjusted_background_non_voc <- background_lod_non_voc %>%
  rowwise() %>%
  mutate(min = min(hourly_data[chemical], na.rm = TRUE),
         LODx2 = 2 * LOD,
         criterion1 = min(hourly_data[chemical], na.rm = TRUE) < 2 * LOD,
         max = max(hourly_data[chemical], na.rm = TRUE),
         LODx100 = 100 * LOD,
         criterion2 = max(hourly_data[chemical], na.rm = TRUE) > 100 * LOD,
         adjusted_background = adjusting_neg_bg_from_lod(chemical, LOD, background, hourly_data))
```

```r
# Check if background is negligible for voc
# merge background and LOD
background_lod_voc <- LOD_voc_avg %>%
  left_join(tibble(chemical = setdiff(names(background_levels), non_vocs),
                   background = background_levels[setdiff(names(background_levels), non_vocs)]))
```

```
## Joining with `by = join_by(chemical)`
```

```r
adjusted_background_voc <- background_lod_voc %>%
  rowwise() %>%
  mutate(min = min(hourly_data[chemical], na.rm = TRUE),
         LODx2 = 2 * LOD,
         criterion1 = min(hourly_data[chemical], na.rm = TRUE) < 2 * LOD,
         max = max(hourly_data[chemical], na.rm = TRUE),
         LODx100 = 100 * LOD,
         criterion2 = max(hourly_data[chemical], na.rm = TRUE) > 100 * LOD,
         adjusted_background = adjusting_neg_bg_from_lod(chemical, LOD, background, hourly_data))
```

```r
# So now we have the adjusted background concentrations
subtract_adj_bg <- function(column, chemical) {
  print(chemical)
  result <-
  return (result)
}
hourly_nona_bgrm <- hourly_full_nona %>%
  mutate(across(adjusted_background_non_voc$chemical, ~  .x - adjusted_background_non_voc$adjusted_backg
hourly_nona_bgrm <- hourly_nona_bgrm %>%
  mutate(across(adjusted_background_voc$chemical, ~  .x - adjusted_background_voc$adjusted_background[ad
```

```r
# look at zero values
colSums(hourly_nona_bgrm == 0)
```

```
##           ch4        co2_ppm             co            h2s            so2
##             1              1              1            777           3065
##           nox             o3         ethane         ethene        propane
##             0              0              1              0              1
##       propene 1_3-butadiene       i-butane       n-butane      acetylene
##             0           3126              1              1              0
##  cyclopentane      i-pentane      n-pentane       n-hexane       isoprene
##             0              1              1              0           2815
##     n-heptane        benzene       n-octane        toluene ethyl-benzene
##             0              0              0              0              0
##    m&p-xylene       o-xylene
##             0              0
```

```r
# replace negative values with random values between 0 and 0.5*LOD
set.seed(123)
replace_zero_with_random <- function(column, name, LOD_df){
  LOD <- LOD_df$LOD[LOD_df$chemical == name]
  column <- if_else(column == 0, round(runif(length(column), 0, 0.5 * LOD), 3), column)
  return (column)
}

hourly_nona_bgrm_zerorepl <- hourly_nona_bgrm %>%
  mutate(across(adjusted_background_non_voc$chemical, ~ replace_zero_with_random(.x, cur_column(), adju

hourly_nona_bgrm_zerorepl <- hourly_nona_bgrm_zerorepl %>%
  mutate(across(adjusted_background_voc$chemical, ~ replace_zero_with_random(.x, cur_column(), adjusted_
```

Normalize the non-vocs

```r
#normalizing function
normalize_column <- function(column){
  background <- quantile(column, 0)
  max <- quantile(column, 1) # this could be adjusted
  return ((column - background)/(max - background))
}
```

```r
# normalize all
hourly_nona_bgrm_zerorepl_norm <- as_tibble(sapply(as.list(hourly_nona_bgrm_zerorepl), normalize_column
summary(hourly_nona_bgrm_zerorepl_norm)
```

```
##       ch4              co2_ppm             co                h2s
##  Min.   :0.000000   Min.   :0.0000   Min.   :0.00000   Min.   :0.00000
##  1st Qu.:0.005697   1st Qu.:0.1397   1st Qu.:0.02347   1st Qu.:0.01022
##  Median :0.014615   Median :0.1826   Median :0.03663   Median :0.02338
##  Mean   :0.027164   Mean   :0.1998   Mean   :0.04550   Mean   :0.03517
##  3rd Qu.:0.037362   3rd Qu.:0.2415   3rd Qu.:0.05734   3rd Qu.:0.04564
##  Max.   :1.000000   Max.   :1.0000   Max.   :1.00000   Max.   :1.00000
##       so2               nox                o3              ethane
##  Min.   :0.000000   Min.   :0.000000   Min.   :0.0000   Min.   :0.000000
##  1st Qu.:0.007997   1st Qu.:0.006515   1st Qu.:0.1118   1st Qu.:0.008171
##  Median :0.015756   Median :0.020376   Median :0.2961   Median :0.026203
##  Mean   :0.026295   Mean   :0.036923   Mean   :0.3079   Mean   :0.051422
##  3rd Qu.:0.023633   3rd Qu.:0.050500   3rd Qu.:0.4735   3rd Qu.:0.075895
##  Max.   :1.000000   Max.   :1.000000   Max.   :1.0000   Max.   :1.000000
##      ethene            propane            propene          1_3-butadiene
##  Min.   :0.00000   Min.   :0.000000   Min.   :0.000000   Min.   :0.000000
##  1st Qu.:0.01303   1st Qu.:0.009005   1st Qu.:0.005798   1st Qu.:0.001667
##  Median :0.03615   Median :0.028067   Median :0.018663   Median :0.004167
##  Mean   :0.05116   Mean   :0.054283   Mean   :0.028932   Mean   :0.007500
##  3rd Qu.:0.07324   3rd Qu.:0.081075   3rd Qu.:0.043305   3rd Qu.:0.007500
##  Max.   :1.00000   Max.   :1.000000   Max.   :1.000000   Max.   :1.000000
##     i-butane           n-butane          acetylene         cyclopentane
##  Min.   :0.000000   Min.   :0.000000   Min.   :0.00000   Min.   :0.000000
##  1st Qu.:0.005999   1st Qu.:0.008556   1st Qu.:0.02769   1st Qu.:0.007284
##  Median :0.019042   Median :0.027302   Median :0.05265   Median :0.022445
##  Mean   :0.038815   Mean   :0.055538   Mean   :0.07600   Mean   :0.043917
##  3rd Qu.:0.054019   3rd Qu.:0.078140   3rd Qu.:0.10400   3rd Qu.:0.062356
##  Max.   :1.000000   Max.   :1.000000   Max.   :1.00000   Max.   :1.000000
##    i-pentane          n-pentane           n-hexane           isoprene
##  Min.   :0.000000   Min.   :0.000000   Min.   :0.00000   Min.   :0.000000
##  1st Qu.:0.006092   1st Qu.:0.005596   1st Qu.:0.00466   1st Qu.:0.002801
##  Median :0.019717   Median :0.018229   Median :0.01579   Median :0.005602
##  Mean   :0.041448   Mean   :0.039184   Mean   :0.03512   Mean   :0.010377
##  3rd Qu.:0.057853   3rd Qu.:0.054789   3rd Qu.:0.04926   3rd Qu.:0.011204
##  Max.   :1.000000   Max.   :1.000000   Max.   :1.00000   Max.   :1.000000
##    n-heptane          benzene            n-octane           toluene
##  Min.   :0.000000   Min.   :0.00000   Min.   :0.000000   Min.   :0.00000
##  1st Qu.:0.005317   1st Qu.:0.01574   1st Qu.:0.008014   1st Qu.:0.01345
##  Median :0.017889   Median :0.04097   Median :0.025499   Median :0.04122
##  Mean   :0.039271   Mean   :0.07624   Mean   :0.054094   Mean   :0.07760
##  3rd Qu.:0.055078   3rd Qu.:0.10706   3rd Qu.:0.076206   3rd Qu.:0.11165
##  Max.   :1.000000   Max.   :1.00000   Max.   :1.000000   Max.   :1.00000
##  ethyl-benzene        m&p-xylene          o-xylene
##  Min.   :0.000000   Min.   :0.000000   Min.   :0.00000  
##  1st Qu.:0.006472   1st Qu.:0.006733   1st Qu.:0.00000  
##  Median :0.032363   Median :0.037512   Median :0.04031  
##  Mean   :0.061284   Mean   :0.076834   Mean   :0.08577  
##  3rd Qu.:0.088457   3rd Qu.:0.113177   3rd Qu.:0.12745  
##  Max.   :1.000000   Max.   :1.000000   Max.   :1.00000  
```

Combine and Transpose

```
normalized_matrix <- as.matrix(hourly_nona_bgrm_zerorepl_norm) #important: using the normalized VOCs fo

# Transpose <- cbind(Normalized_Data, Merged_VOCs)
# rownames(Transpose) <- as.character(Transpose[,1]) # I'm not able to run this line, but it shouldn't 
# transpose_normalized_matrix <- t(as.matrix(normalized_matrix))

number_row<- dim(normalized_matrix)[1] #store number of rows (used for checking)
number_column<- dim(normalized_matrix)[2] #store number of columns
```

NMF section

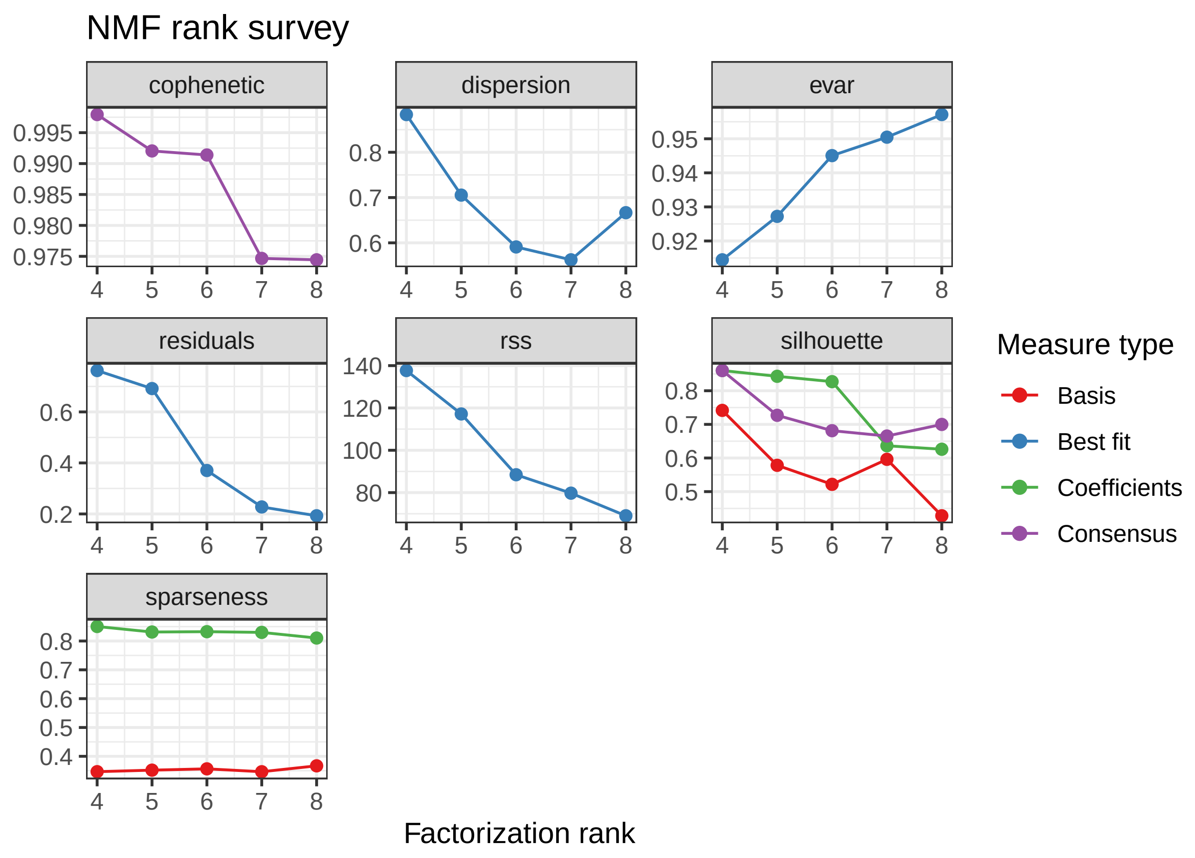
```
# compute weight matrix (uncertainties)
# Based on the Guha paper
# next comment is from the other nmf R file
weight_matrix <- matrix(0, nrow = nrow(normalized_matrix), ncol = ncol(normalized_matrix))
LOD_merged <- tibble(chemical = c(adjusted_background_non_voc$chemical, adjusted_background_voc$chemical
                     LOD = c(adjusted_background_non_voc$LOD, adjusted_background_voc$LOD))

LOD_merged <- tibble(chemical = names(hourly_nona_bgrm_zerorepl_norm)) %>%
  left_join(LOD_merged)
```

```
## Joining with `by = join_by(chemical)`
```

```
# creating uncertainty Matrix
for (i in 1:number_row) {
  for (j in 1:number_column) {
    xij <- normalized_matrix[i, j]
    LOD <- LOD_merged$LOD[[j]]
    # Get LOD value for this row
    if (j == 1) {
      # based on equation 6, we sqrt ch4 (at column = 1) and times by 1
      weight_matrix[i, j] <- sqrt(xij)
    } else if (j == 2) {
      # 0.25 for co2
      weight_matrix[i, j] <- 0.25 * sqrt(xij)
    } else if (j == 3) {
      # 0.5 for CO
      weight_matrix[i, j] <- 0.5 * sqrt(xij)
    } else if (xij <= LOD) {
      weight_matrix[i, j] <- 2 * LOD # equation 5a) in reference paper
    } else {
      weight_matrix[i, j] <- sqrt(((0.1 * xij)**2 + LOD**2))  #equation 5c) in reference paper
    }
  }
}
```

```
# set a seed for nmf
# set.seed(123)
# #function below used to estimate the optimal rank and will be used in the nmf() function.
# # takes around 20-30 mins to run
# estimate_rank <- nmfEstimateRank(normalized_matrix, 4:8, method = "ls-nmf", weight = weight_matrix, 3
# # # changing the range of rank to 2:20 from 4:20
# saveRDS(estimate_rank, 'estimate_rank.rds')

estimate_rank <- readRDS('estimate_rank.rds')
measures <- estimate_rank$measures
fit <- estimate_rank$fit
consensus <- estimate_rank$consensus
```

```
# plots the NMF rank survey
plot(estimate_rank)
```

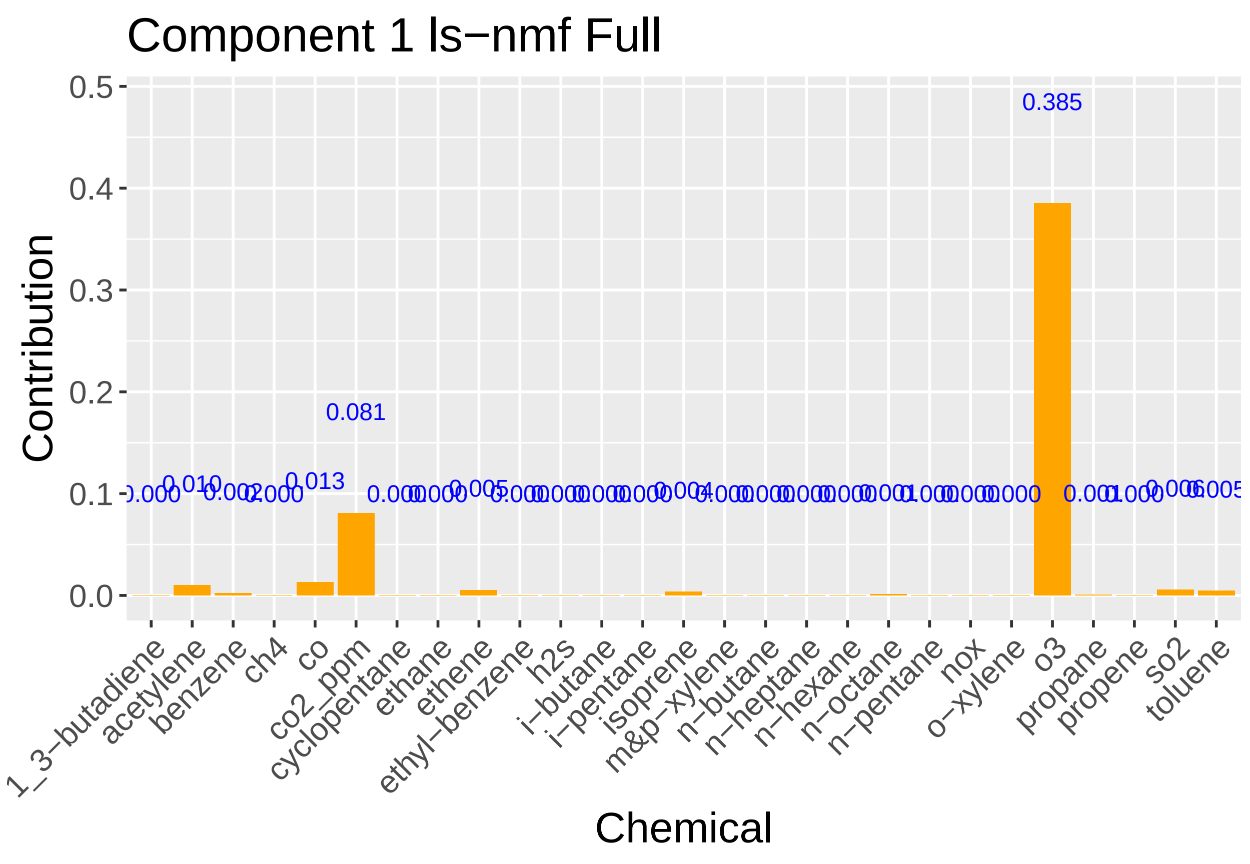


```
# fitting the optimal rank based on the above plots
# the choice of the optimal rank needs to be discussed
output <- nmf(normalized_matrix, rank = 4, weight = weight_matrix, method = "ls-nmf")
W <- basis(output)
H <- coef(output)
```

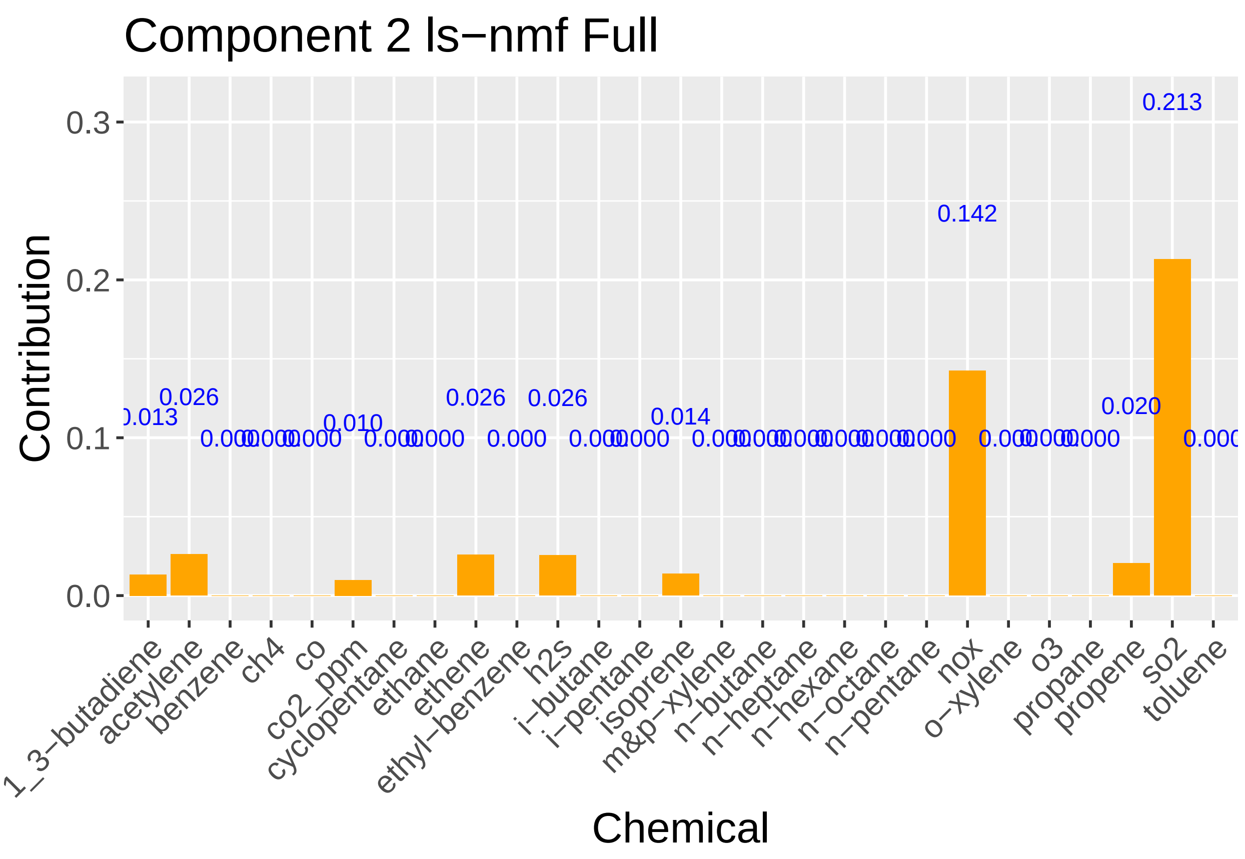
Source contributions

```
# Convert H to a data frame for ggplot
H_df <- as.data.frame(H)
```

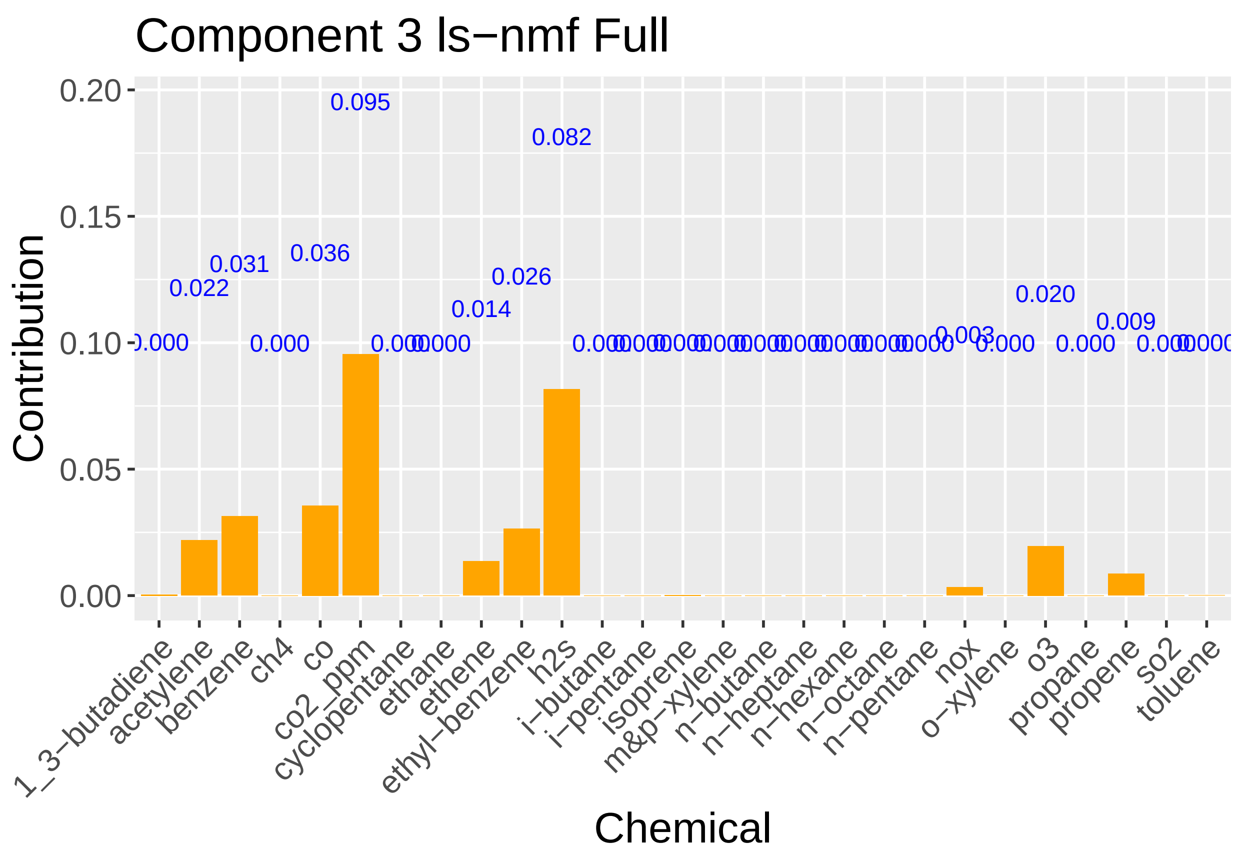
```r
# Add a column for component
H_df$Component <- names(as.data.frame(W))

# Reshape data to long format
H_long <- pivot_longer(H_df, cols = -Component, names_to = "Chemical", values_to = "Contribution")

get_component_plot <- function(data, component, title) {
  component_data <- subset(data, Component == component)
  plot <- ggplot(component_data, aes(x = Chemical, y = Contribution)) +
            geom_bar(stat = "identity", position = "dodge", fill = "orange") +
            theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
            geom_text(aes(label = sprintf("%.3f", Contribution)), color = "blue", size = 3, nudge_y = 0
            labs(x = "Chemical", y = "Contribution", title = title)+
            theme(
            text = element_text(size = 14), # Base text size for all text elements
            axis.title = element_text(size = 16), # Size of axis titles
            axis.text = element_text(size = 12), # Size of axis text (tick labels)
            plot.title = element_text(size = 18) # Size of the plot title
            )
  return(plot)
}
nmfplt_1_ls <- get_component_plot(H_long, 'V1', 'Component 1 ls-nmf Full')
nmfplt_1_ls
```



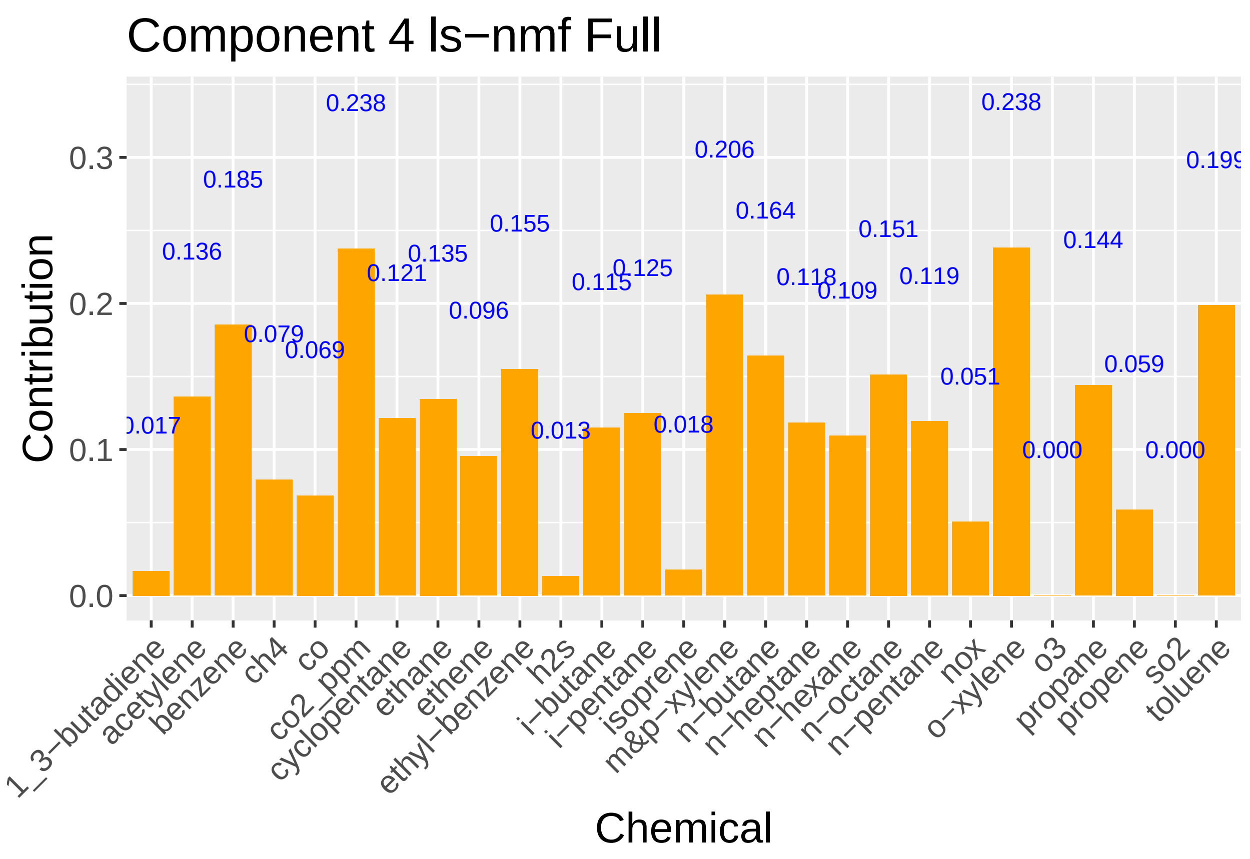
```r
nmfplt_2_ls <- get_component_plot(H_long, 'V2', 'Component 2 ls-nmf Full')
nmfplt_2_ls
```



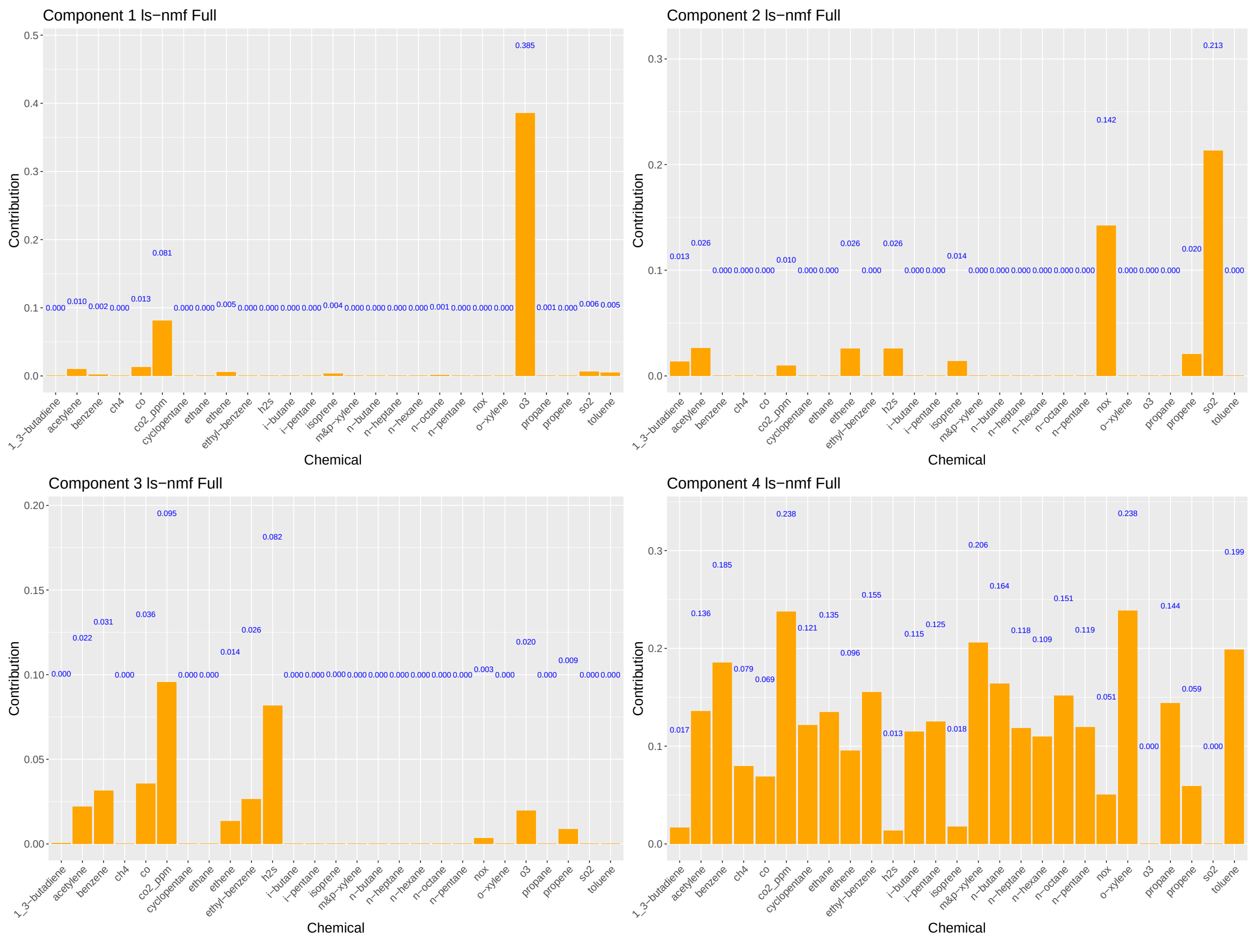
```r
nmfplt_3_ls <- get_component_plot(H_long, 'V3', 'Component 3 ls-nmf Full')
nmfplt_3_ls
```



```
nmfplt_4_ls <- get_component_plot(H_long, 'V4', 'Component 4 ls-nmf Full')
nmfplt_4_ls
```

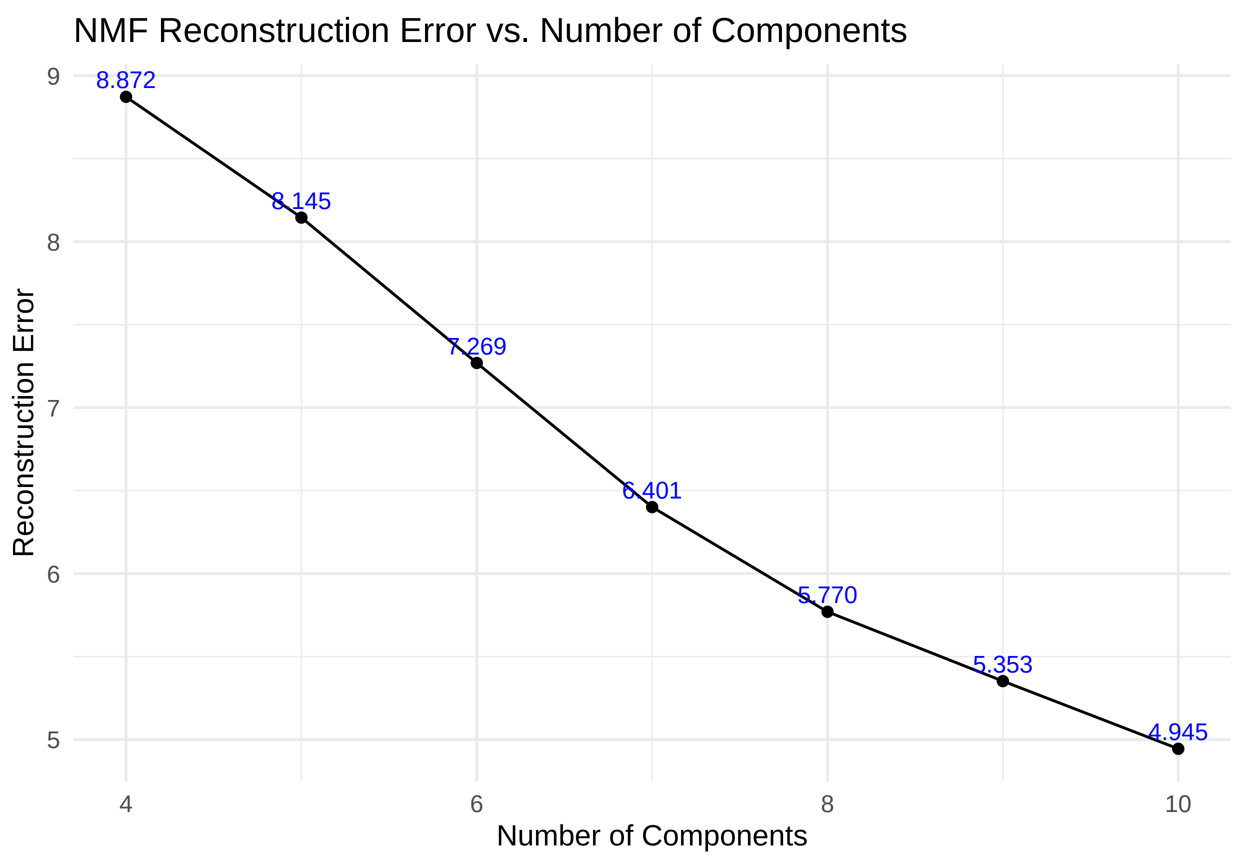


Component 4 ls−nmf Full
Contribution
Chemical
0.0
0.1
0.2
0.3
1_3−butadiene
acetylene
benzene
ch4
co
co2_ppm
cyclopentane
ethane
ethene
ethyl−benzene
h2s
i−butane
i−pentane
isoprene
m&p−xylene
n−butane
n−heptane
n−hexane
n−octane
n−pentane
nox
o−xylene
o3
propane
propene
so2
toluene
.017
0.136
0.185
0.079
0.069
0.238
0.121
0.135
0.096
0.155
0.013
0.115
0.125
0.018
0.206
0.164
0.118
0.109
0.151
0.119
0.051
0.238
0.000
0.144
0.059
0.000
0.199

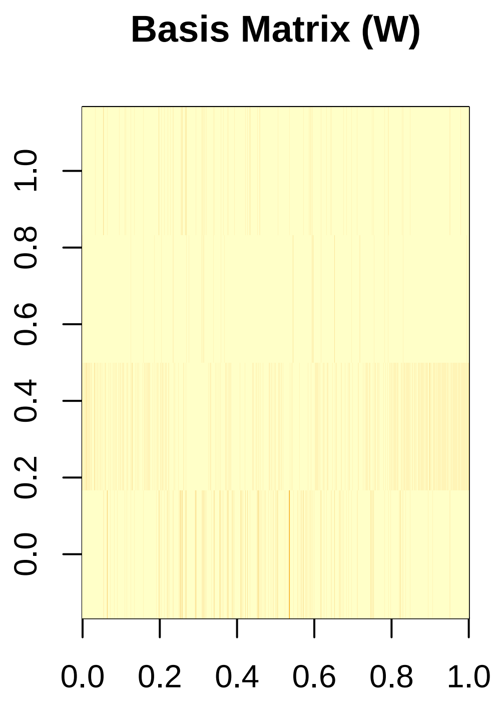


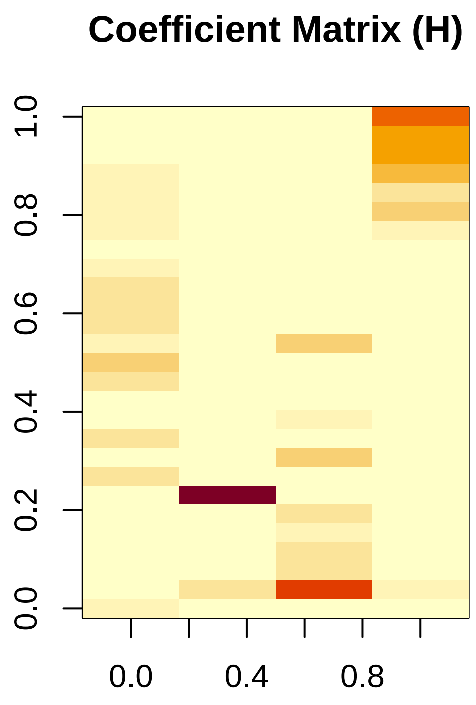
NMF - Eva

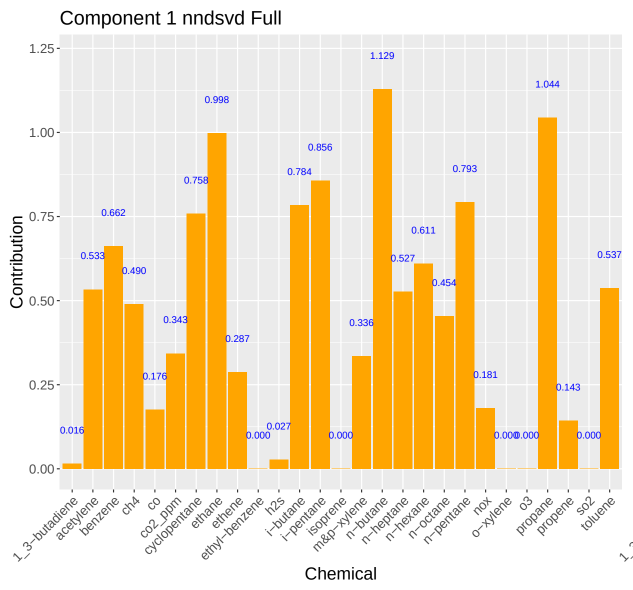
```r
# Try Eva's approach
components <- 4:10
errors <- numeric(length(components) - 4)

# Loop over the number of components
# for (n in components) {
#   nmf_result <- nmf(normalized_matrix, rank = n, method = "KL", seed='nndsvd')
#   reconstruction <- basis(nmf_result) %*% coef(nmf_result)
#   error <- norm(normalized_matrix - reconstruction, type = "F")
#   errors[n-3] <- error
#   print(paste0('Completed ', n - 3, ' out of 7'))
# }
#
# saveRDS(errors, 'errors_norm.rds')

errors <- readRDS('errors_norm.rds')
```

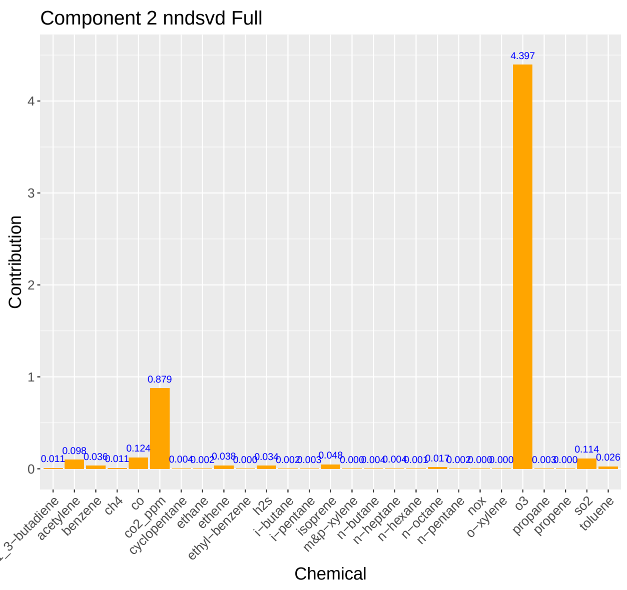


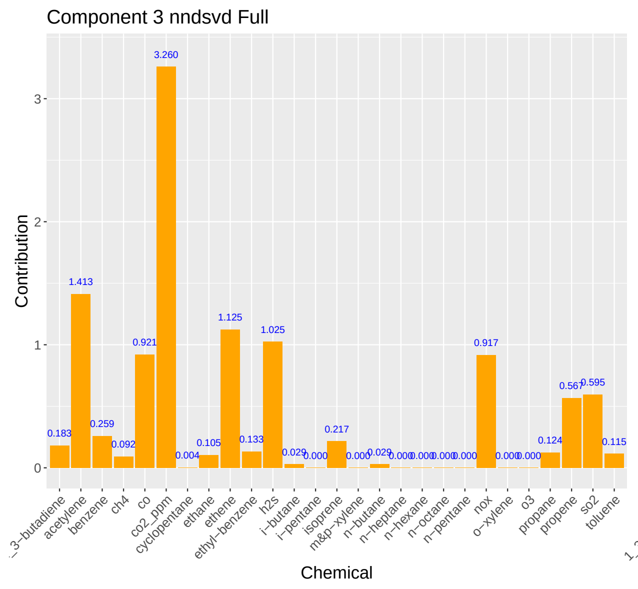
```
## [1] 8.872266
## Warning in sqrt(S[i] * termp) * uup: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
## Warning in sqrt(S[i] * termp) * vvp: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
## Warning in sqrt(S[i] * termp) * uup: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
## Warning in sqrt(S[i] * termp) * vvp: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
## Warning in sqrt(S[i] * termn) * uun: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
## Warning in sqrt(S[i] * termn) * vvn: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
```

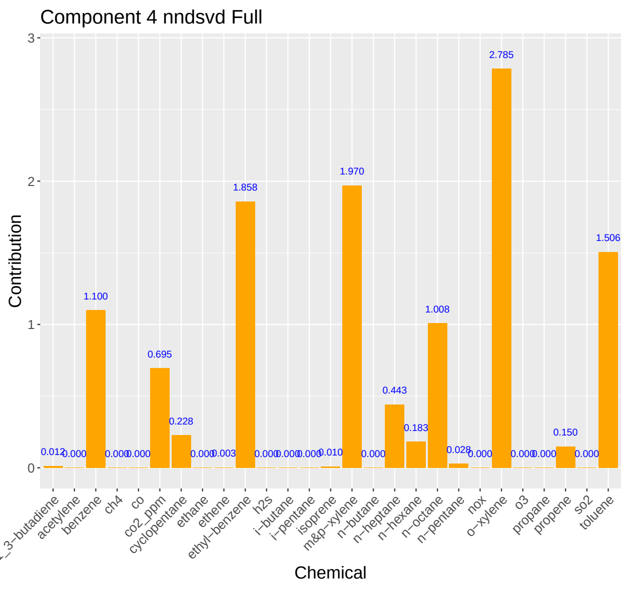

Basis Matrix (W)
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.2
0.4
0.6
0.8
1.0


Coefficient Matrix (H)
1.0
0.8
0.6
0.4
0.2
0.0
0.0
0.4
0.8


Component 1 nndsvd Full
Contribution
Chemical

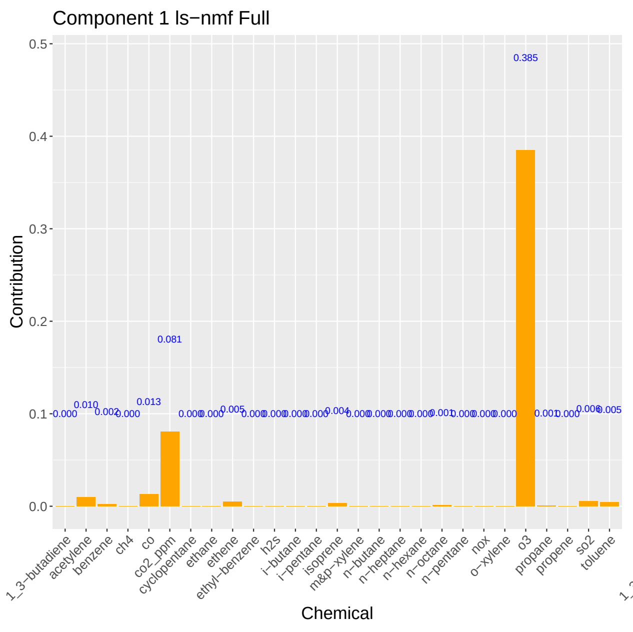

Component 2 nndsvd Full
Contribution
Chemical

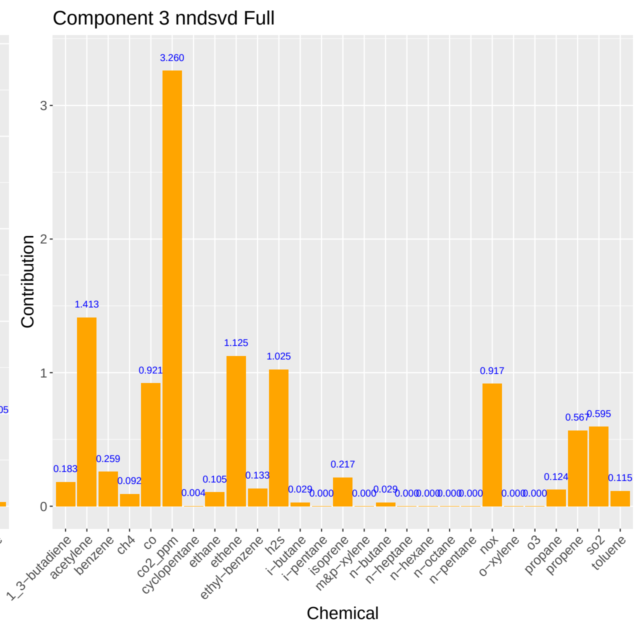

Component 3 nndsvd Full
Contribution
Chemical


Component 4 nndsvd Full
Contribution
Chemical

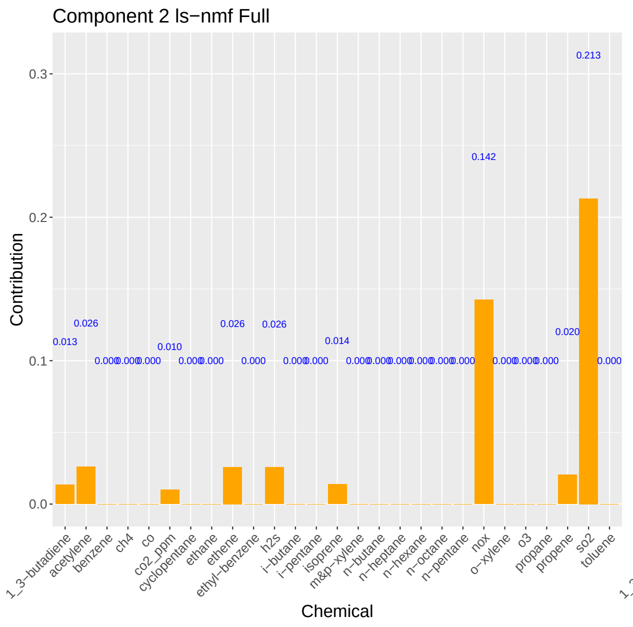
Method comparisons

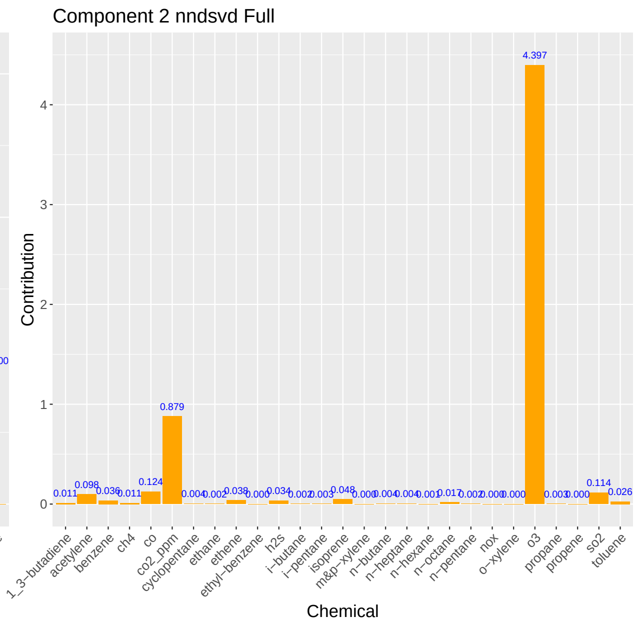
```
grid.arrange(nmfplt_1_ls, nmfplt_3_svd, ncol = 2)
```



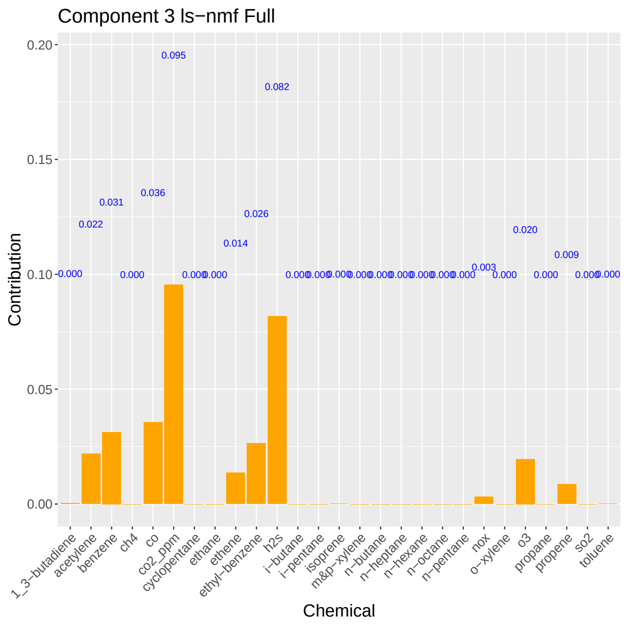


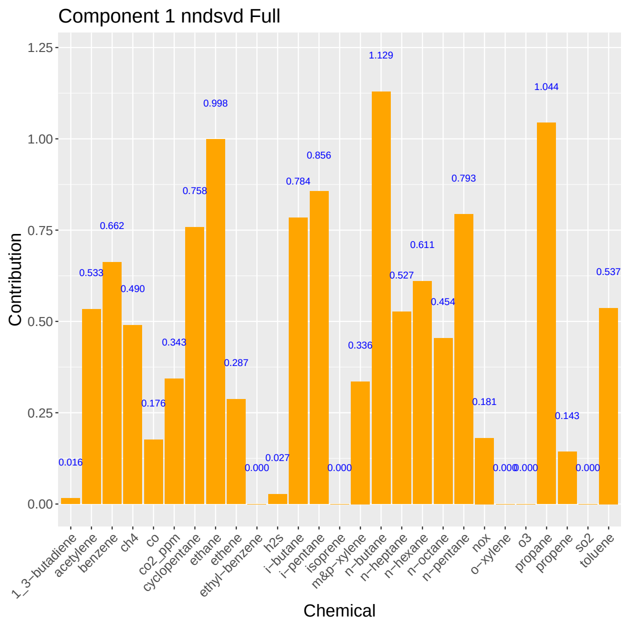
```
grid.arrange(nmfplt_2_ls, nmfplt_2_svd, ncol = 2)
```



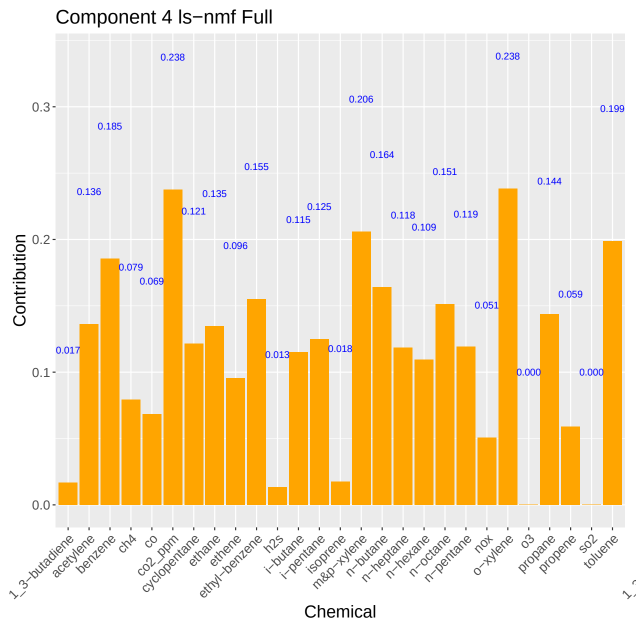


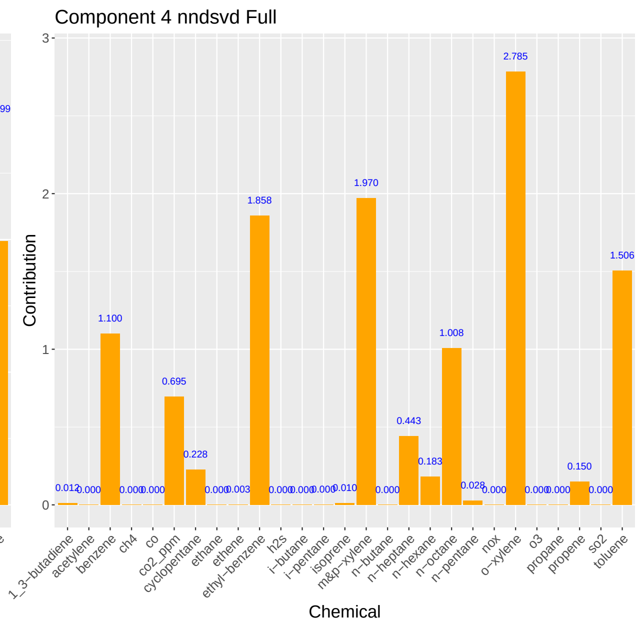
```
grid.arrange(nmfplt_3_ls, nmfplt_1_svd, ncol = 2)
```





```
grid.arrange(nmfplt_4_ls, nmfplt_4_svd, ncol = 2)
```





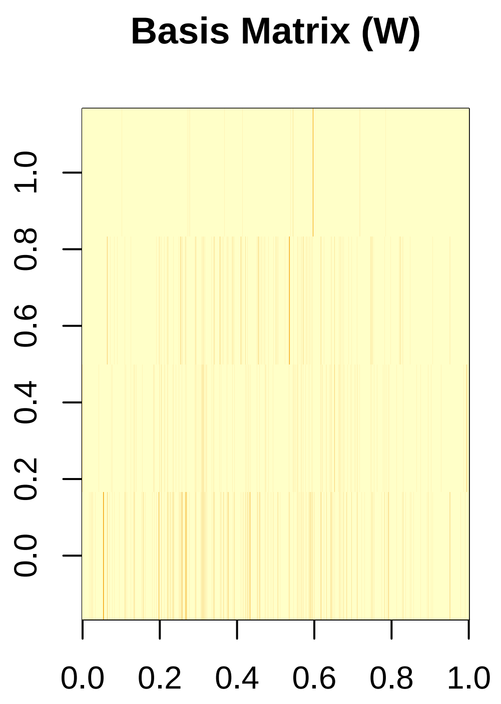
Remove Ozone

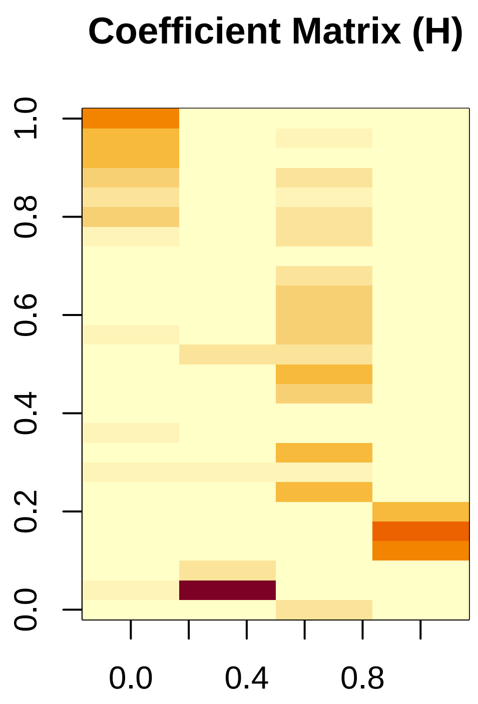
4 Components

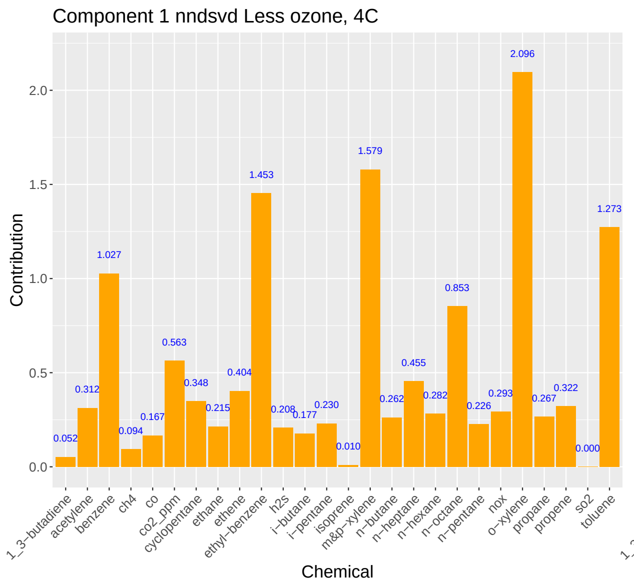
```
normalized_matrix_less_o3 <- normalized_matrix[ ,setdiff(colnames(normalized_matrix), "o3")]

nmf_result_4c_less_o3 <- nmf(normalized_matrix_less_o3, rank = 4, method = "KL", seed='nndsvd')
```

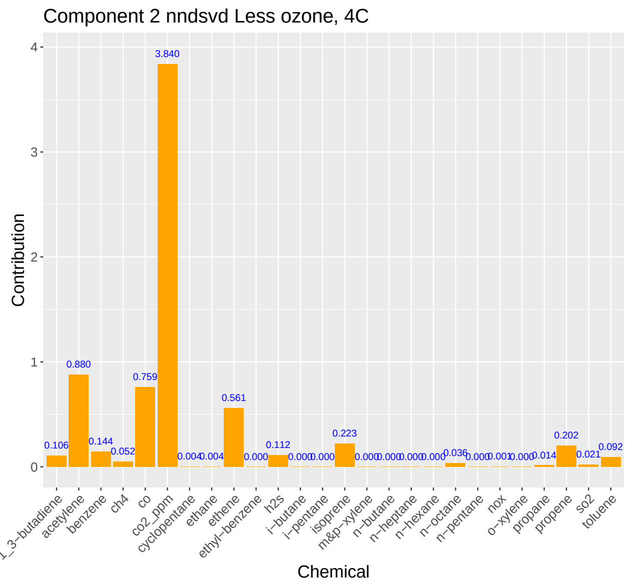
```
## Warning in sqrt(S[i] * termn) * uun: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
## Warning in sqrt(S[i] * termn) * vvn: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * uun: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * vvn: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * uun: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * vvn: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
```

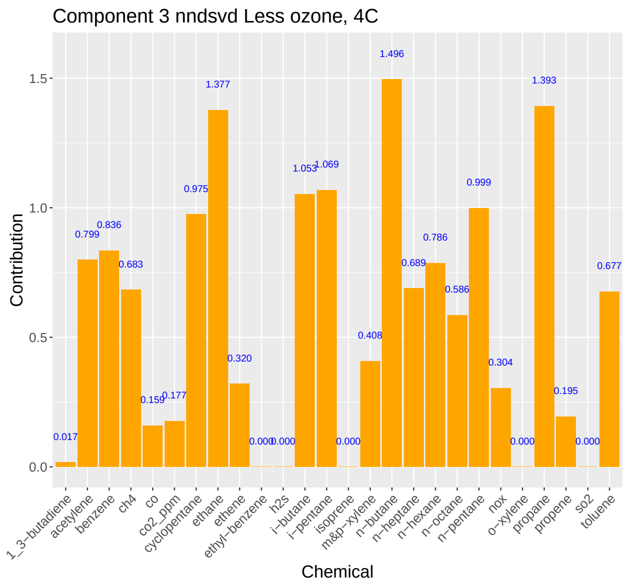
```r
basis_matrix_4c_less_o3 <- basis(nmf_result_4c_less_o3)
coef_matrix_4c_less_o3 <- coef(nmf_result_4c_less_o3)

par(mfrow = c(1, 2))
image(basis_matrix_4c_less_o3, main = "Basis Matrix (W)")
image(coef_matrix_4c_less_o3, main = "Coefficient Matrix (H)")
```

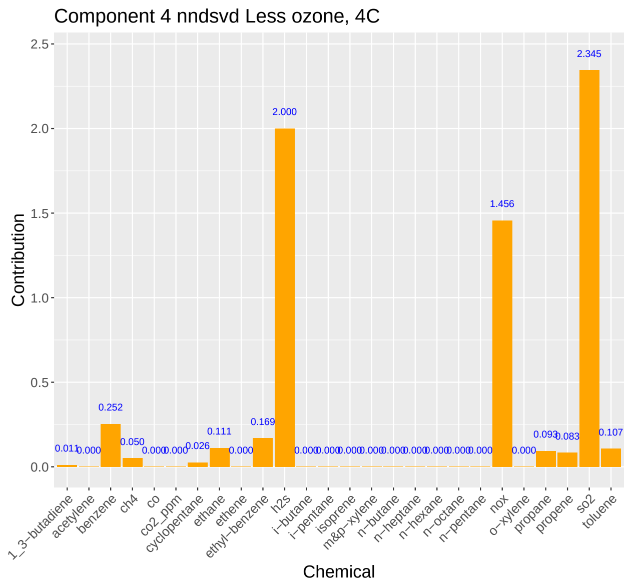










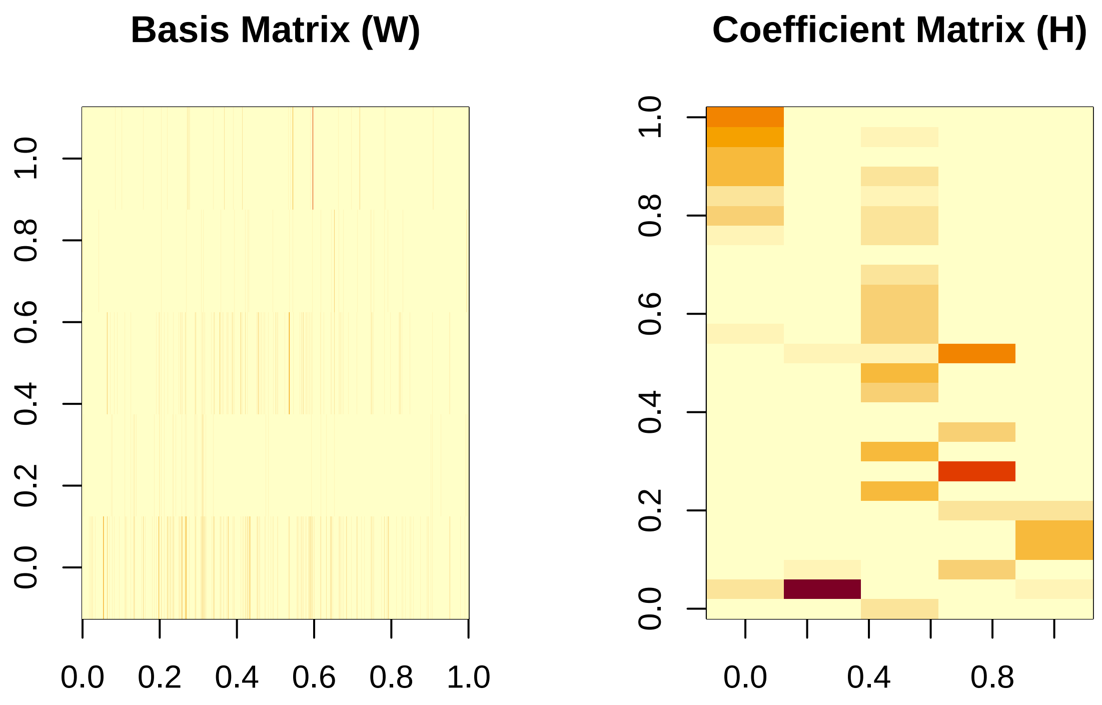


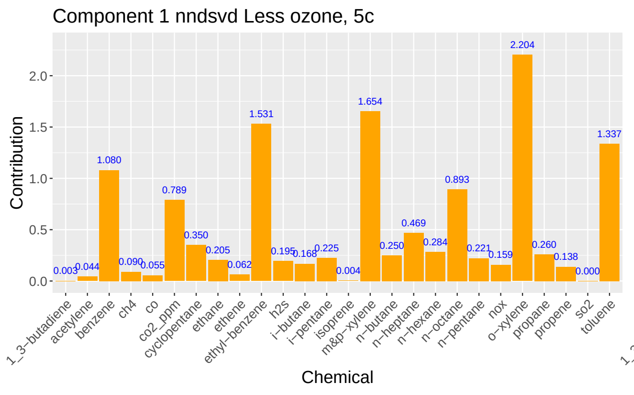
5 Components

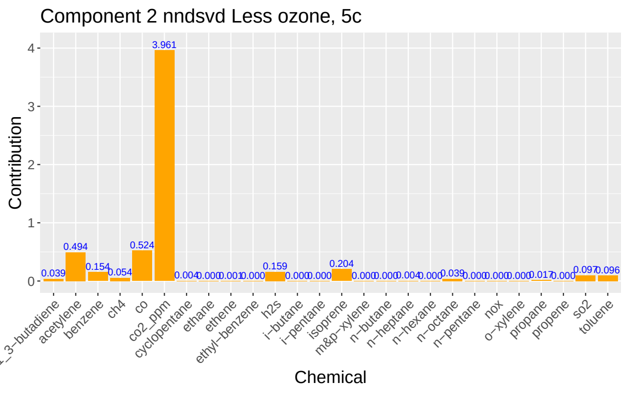
```r
nmf_result_5c_less_o3 <- nmf(normalized_matrix_less_o3, rank = 5, method = "KL", seed='nndsvd')
```

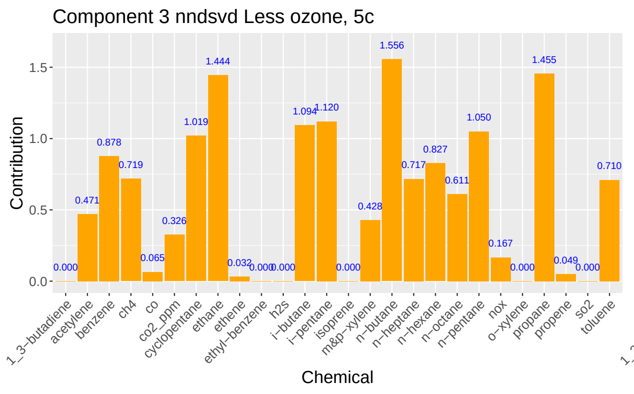
```
## Warning in sqrt(S[i] * termn) * uun: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * vvn: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * uun: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * vvn: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * uun: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
## Warning in sqrt(S[i] * termn) * vvn: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termp) * uup: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termp) * vvp: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
```

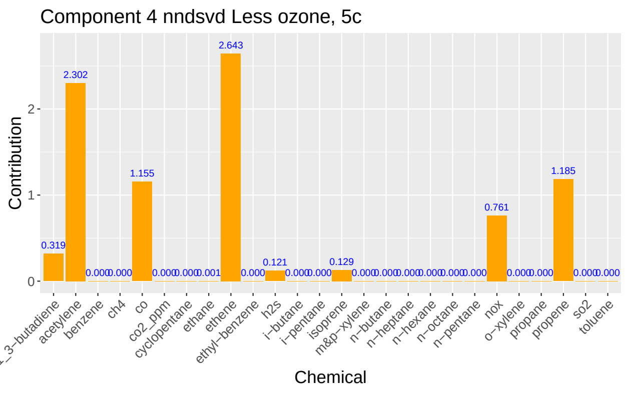
```r
basis_matrix_5c_less_o3 <- basis(nmf_result_5c_less_o3)
coef_matrix_5c_less_o3 <- coef(nmf_result_5c_less_o3)

par(mfrow = c(1, 2))
image(basis_matrix_5c_less_o3, main = "Basis Matrix (W)")
image(coef_matrix_5c_less_o3, main = "Coefficient Matrix (H)")
```

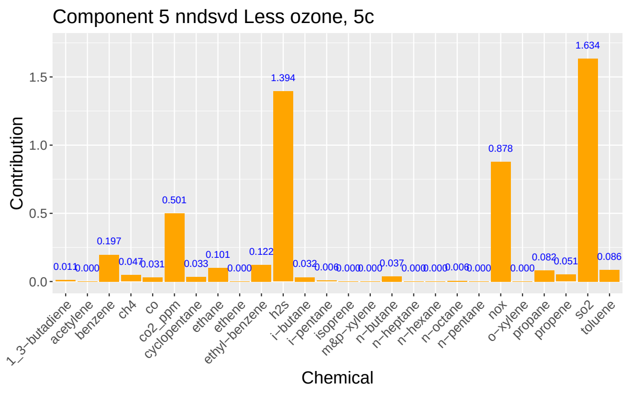



Component 1 nndsvd Less ozone, 5c
Contribution
Chemical
0.0
0.5
1.0
1.5
2.0
1_3−butadiene 0.003
acetylene 0.044
benzene 1.080
ch4 0.090
co 0.055
co2_ppm 0.789
cyclopentane 0.350
ethane 0.205
ethene 0.062
ethyl−benzene 1.531
h2s 0.195
i−butane 0.168
i−pentane 0.225
isoprene 0.004
m&p−xylene 1.654
n−butane 0.250
n−heptane 0.469
n−hexane 0.284
n−octane 0.893
n−pentane 0.221
nox 0.159
o−xylene 2.204
propane 0.260
propene 0.138
so2 0.000
toluene 1.337

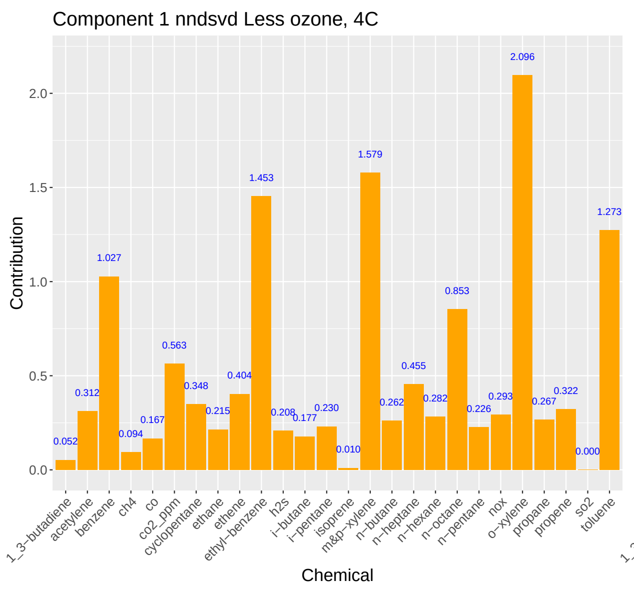

Component 2 nndsvd Less ozone, 5c
Contribution
Chemical
0
1
2
3
4
1_3−butadiene 0.039
acetylene 0.494
benzene 0.154
ch4 0.054
co 0.524
co2_ppm 3.961
cyclopentane 0.004
ethane 0.000
ethene 0.001
ethyl−benzene 0.000
h2s 0.159
i−butane 0.000
i−pentane 0.000
isoprene 0.204
m&p−xylene 0.000
n−butane 0.000
n−heptane 0.004
n−hexane 0.000
n−octane 0.039
n−pentane 0.000
nox 0.000
o−xylene 0.000
propane 0.017
propene 0.000
so2 0.097
toluene 0.096

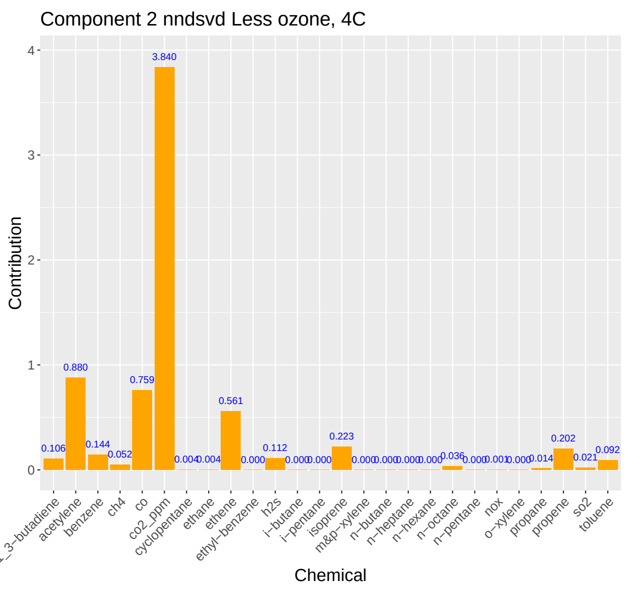

Component 3 nndsvd Less ozone, 5c
Contribution
Chemical
0.0
0.5
1.0
1.5
1_3−butadiene 0.000
acetylene 0.471
benzene 0.878
ch4 0.719
co 0.065
co2_ppm 0.326
cyclopentane 1.019
ethane 1.444
ethene 0.032
ethyl−benzene 0.000
h2s 0.000
i−butane 1.094
i−pentane 1.120
isoprene 0.000
m&p−xylene 0.428
n−butane 1.556
n−heptane 0.717
n−hexane 0.827
n−octane 0.611
n−pentane 1.050
nox 0.167
o−xylene 0.000
propane 1.455
propene 0.049
so2 0.000
toluene 0.710

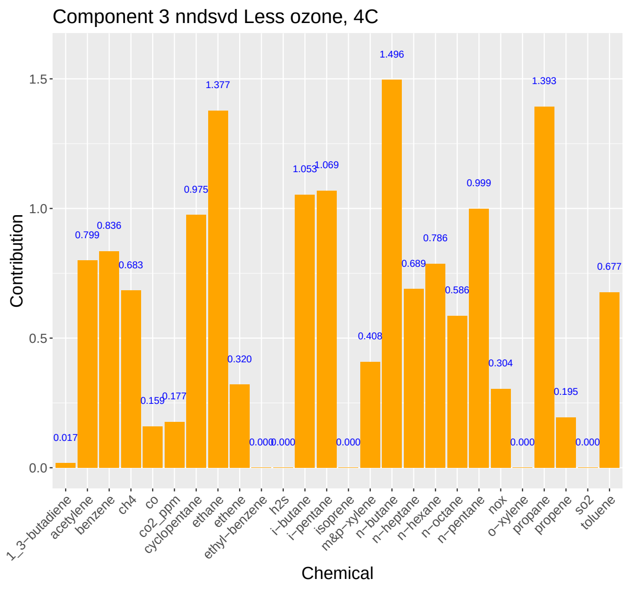

Component 4 nndsvd Less ozone, 5c
Contribution
Chemical
0
1
2
1_3−butadiene 0.319
acetylene 2.302
benzene 0.000
ch4 0.000
co 1.155
co2_ppm 0.000
cyclopentane 0.000
ethane 0.001
ethene 2.643
ethyl−benzene 0.000
h2s 0.121
i−butane 0.000
i−pentane 0.000
isoprene 0.129
m&p−xylene 0.000
n−butane 0.000
n−heptane 0.000
n−hexane 0.000
n−octane 0.000
n−pentane 0.000
nox 0.761
o−xylene 0.000
propane 0.000
propene 1.185
so2 0.000
toluene 0.000

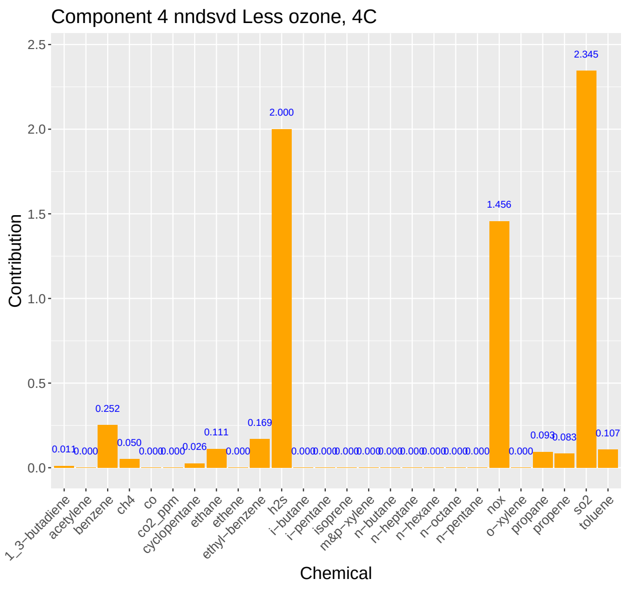

Component 5 nndsvd Less ozone, 5c
Contribution
Chemical
0.0
0.5
1.0
1.5
1_3−butadiene 0.011
acetylene 0.000
benzene 0.197
ch4 0.047
co 0.031
co2_ppm 0.501
cyclopentane 0.033
ethane 0.101
ethene 0.000
ethyl−benzene 0.122
h2s 1.394
i−butane 0.032
i−pentane 0.006
isoprene 0.000
m&p−xylene 0.000
n−butane 0.037
n−heptane 0.000
n−hexane 0.000
n−octane 0.006
n−pentane 0.000
nox 0.878
o−xylene 0.000
propane 0.082
propene 0.051
so2 1.634
toluene 0.086

Compare to 4 components









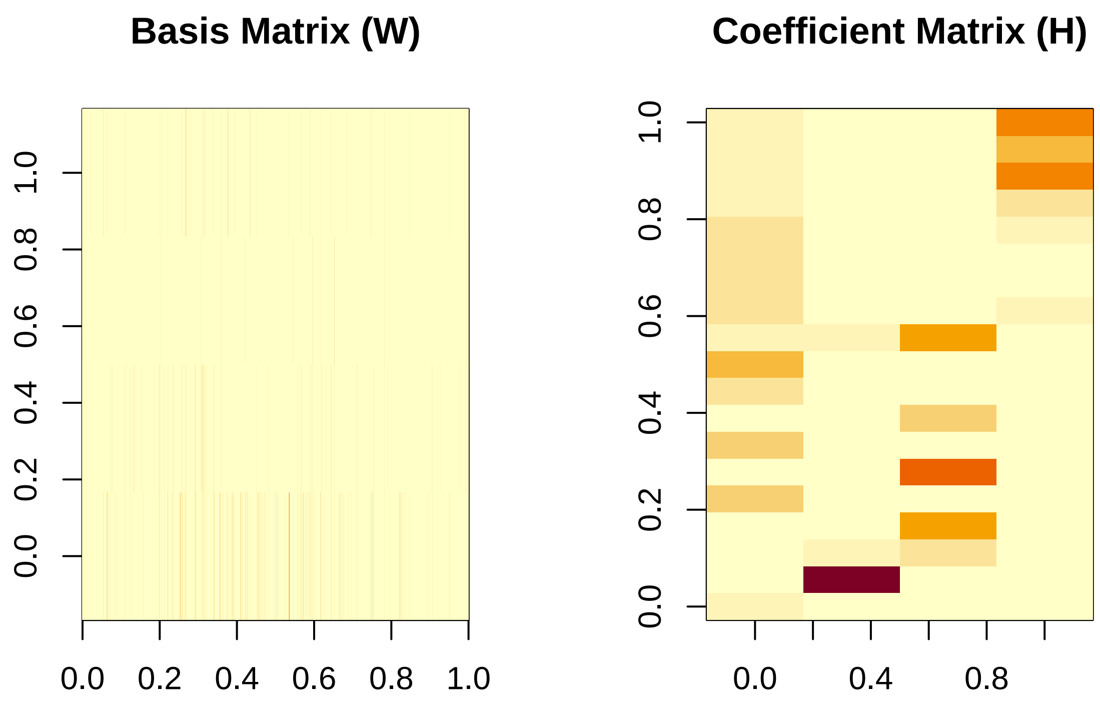
Remove Ozone + chemicals with more than 500+ background values

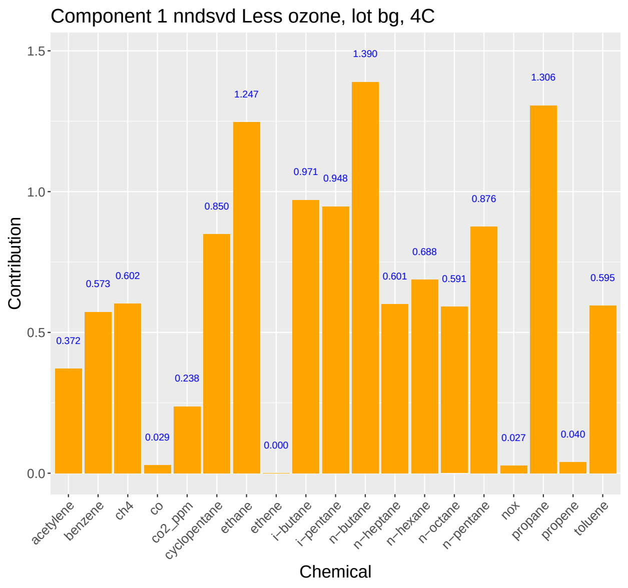
4 Components

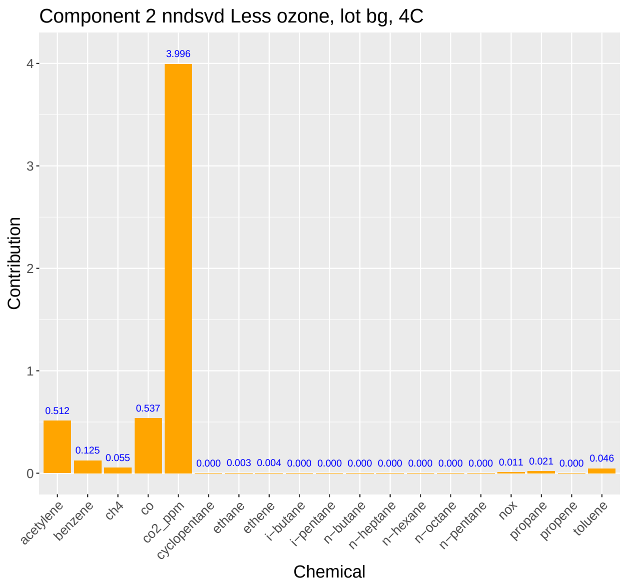
```
normalized_matrix_less_o3_lotbg <- normalized_matrix[ ,setdiff(colnames(normalized_matrix), c('o3', 'h2

nmf_result_4c_less_o3_lotbg <- nmf(normalized_matrix_less_o3_lotbg, rank = 4, method = "KL", seed='nnds
```

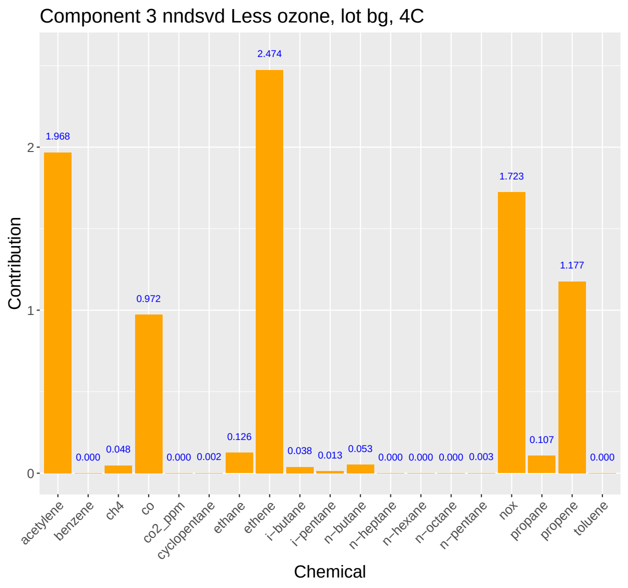
```
## Warning in sqrt(S[i] * termp) * uup: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termp) * vvp: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termp) * uup: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termp) * vvp: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * uun: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * vvn: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
```

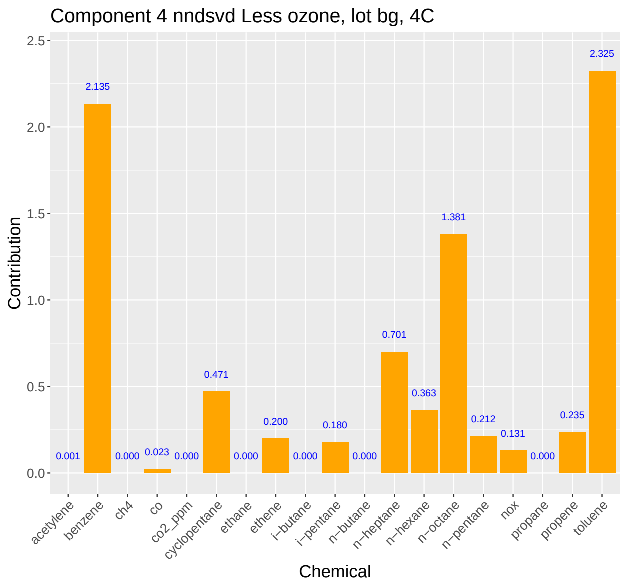
```
basis_matrix_4c_less_o3_lotbg <- basis(nmf_result_4c_less_o3_lotbg)
coef_matrix_4c_less_o3_lotbg <- coef(nmf_result_4c_less_o3_lotbg)

par(mfrow = c(1, 2))
image(basis_matrix_4c_less_o3_lotbg, main = "Basis Matrix (W)")
image(coef_matrix_4c_less_o3_lotbg, main = "Coefficient Matrix (H)")
```









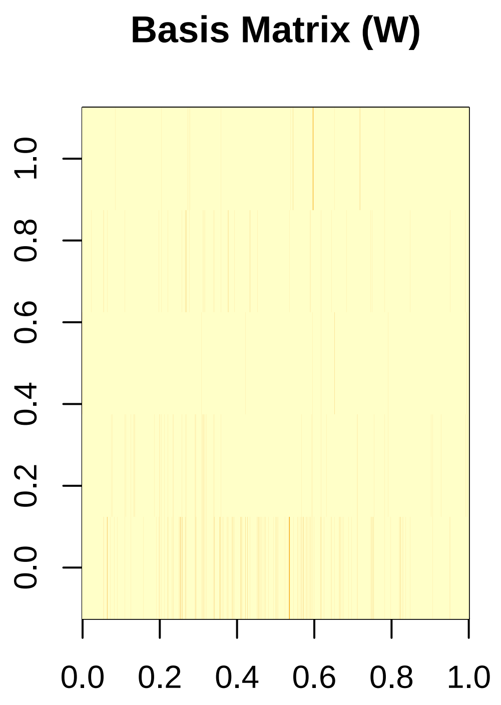


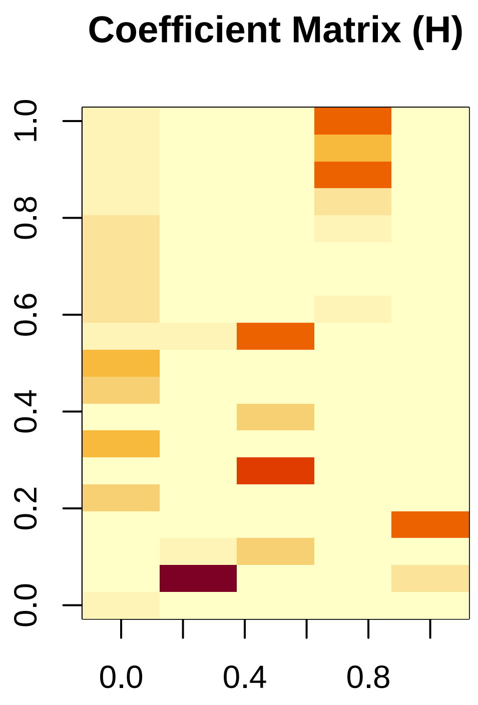
5 Components

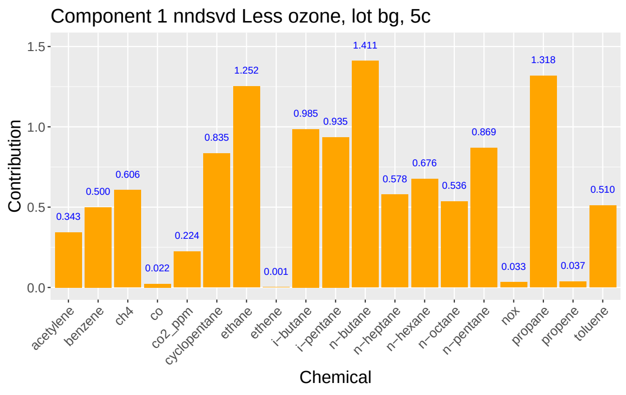
```
nmf_result_5c_less_o3_lotbg <- nmf(normalized_matrix_less_o3_lotbg, rank = 5, method = "KL", seed='nnds
```

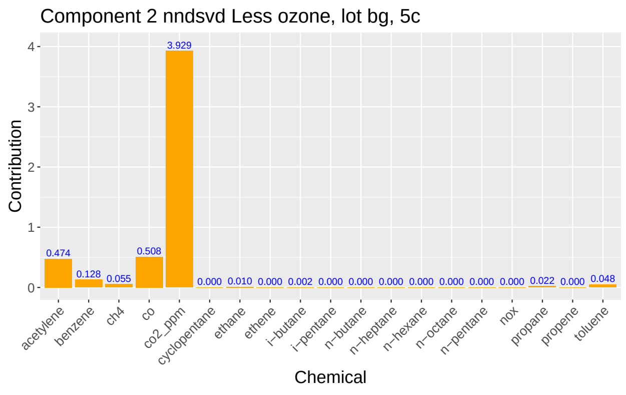
```
## Warning in sqrt(S[i] * termp) * uup: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termp) * vvp: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termp) * uup: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termp) * vvp: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termn) * uun: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
## Warning in sqrt(S[i] * termn) * vvn: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termp) * uup: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## Warning in sqrt(S[i] * termp) * vvp: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.
```

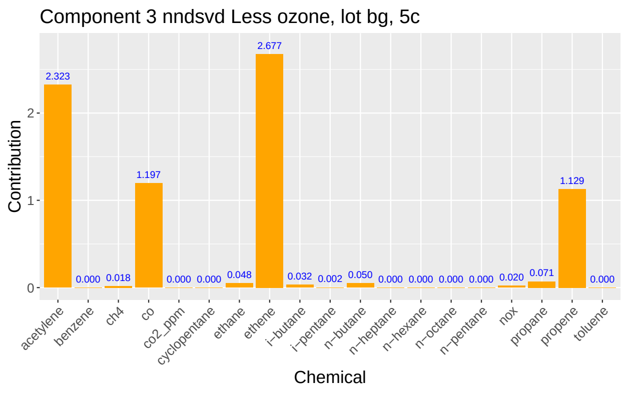
```r
basis_matrix_5c_less_o3_lotbg <- basis(nmf_result_5c_less_o3_lotbg)
coef_matrix_5c_less_o3_lotbg <- coef(nmf_result_5c_less_o3_lotbg)

par(mfrow = c(1, 2))
image(basis_matrix_5c_less_o3_lotbg, main = "Basis Matrix (W)")
image(coef_matrix_5c_less_o3_lotbg, main = "Coefficient Matrix (H)")
```

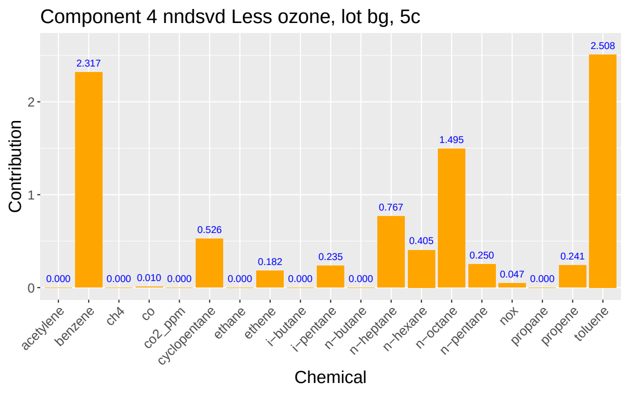


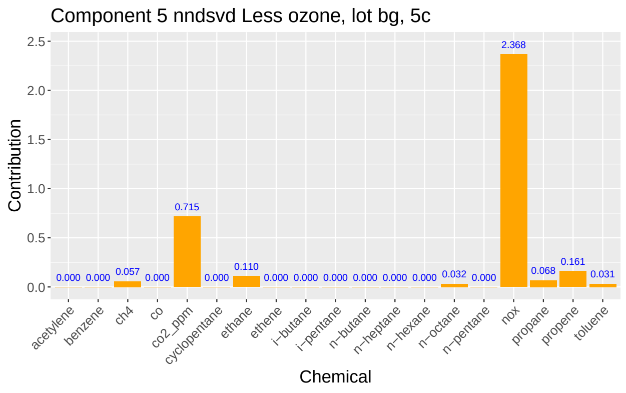




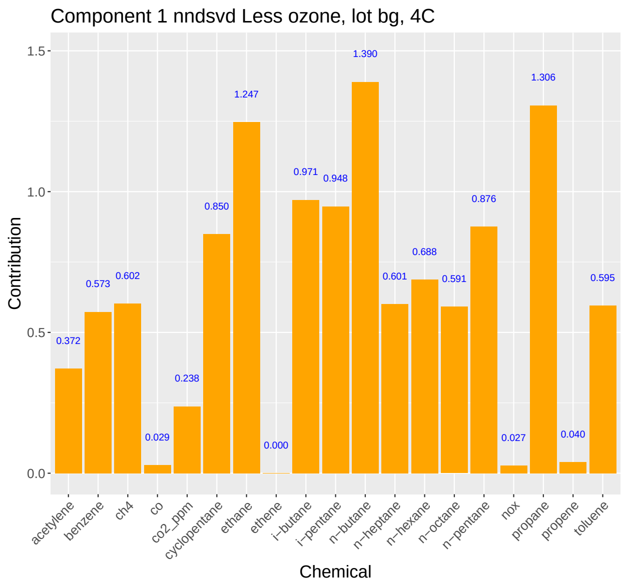


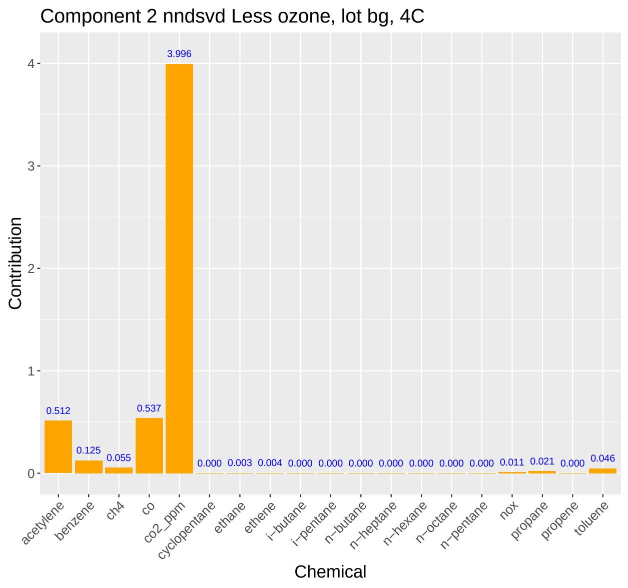


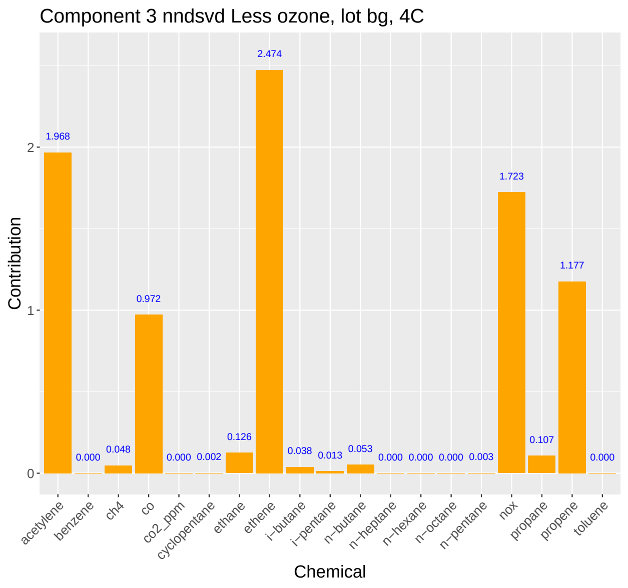


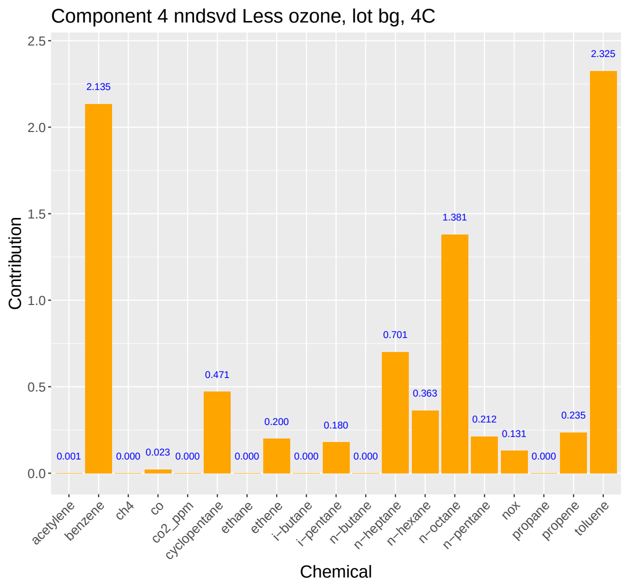


Compare to 4 components


Component 1 nndsvd Less ozone, lot bg, 4C
Contribution
Chemical
0.0
0.5
1.0
1.5
acetylene 0.372
benzene 0.573
ch4 0.602
co 0.029
co2_ppm 0.238
cyclopentane 0.850
ethane 1.247
ethene 0.000
i−butane 0.971
i−pentane 0.948
n−butane 1.390
n−heptane 0.601
n−hexane 0.688
n−octane 0.591
n−pentane 0.876
nox 0.027
propane 1.306
propene 0.040
toluene 0.595


Component 2 nndsvd Less ozone, lot bg, 4C
Contribution
Chemical
0
1
2
3
4
acetylene 0.512
benzene 0.125
ch4 0.055
co 0.537
co2_ppm 3.996
cyclopentane 0.000
ethane 0.003
ethene 0.004
i−butane 0.000
i−pentane 0.000
n−butane 0.000
n−heptane 0.000
n−hexane 0.000
n−octane 0.000
n−pentane 0.000
nox 0.011
propane 0.021
propene 0.000
toluene 0.046


Component 3 nndsvd Less ozone, lot bg, 4C
Contribution
Chemical
0
1
2
acetylene 1.968
benzene 0.000
ch4 0.048
co 0.972
co2_ppm 0.000
cyclopentane 0.002
ethane 0.126
ethene 2.474
i−butane 0.038
i−pentane 0.013
n−butane 0.053
n−heptane 0.000
n−hexane 0.000
n−octane 0.000
n−pentane 0.003
nox 1.723
propane 0.107
propene 1.177
toluene 0.000


Component 4 nndsvd Less ozone, lot bg, 4C
Contribution
Chemical
0.0
0.5
1.0
1.5
2.0
2.5
acetylene 0.001
benzene 2.135
ch4 0.000
co 0.023
co2_ppm 0.000
cyclopentane 0.471
ethane 0.000
ethene 0.200
i−butane 0.000
i−pentane 0.180
n−butane 0.000
n−heptane 0.701
n−hexane 0.363
n−octane 1.381
n−pentane 0.212
nox 0.131
propane 0.000
propene 0.235
toluene 2.325