NMF

William Zhang, Eva, Jerry

2025-01-09

```
# 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 min and max
5. Compute weight matrix according to Guha's paper, without LOQ

Reading the data

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

```
# PROCEDURE STEP 1:
hourly_data <- hourly_data %>% rename('co2' = 'co2_ppm')

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")


non_vocs <- c('ch4', 'co2', 'co', 'h2s', 'so2', 'nox', 'o3')

# remove row with missing obs for any chemical
hourly_nona <- hourly_data %>%
```

```
  select(c('day', 'time_utc', vocs, non_vocs, 'wdr_deg', 'wsp_ms')) %>%
  na.omit()
```

```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use ‘all_of()‘ or ‘any_of()‘ instead.
##   # Was:
##   data %>% select(vocs)
##
##   # Now:
##   data %>% select(all_of(vocs))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

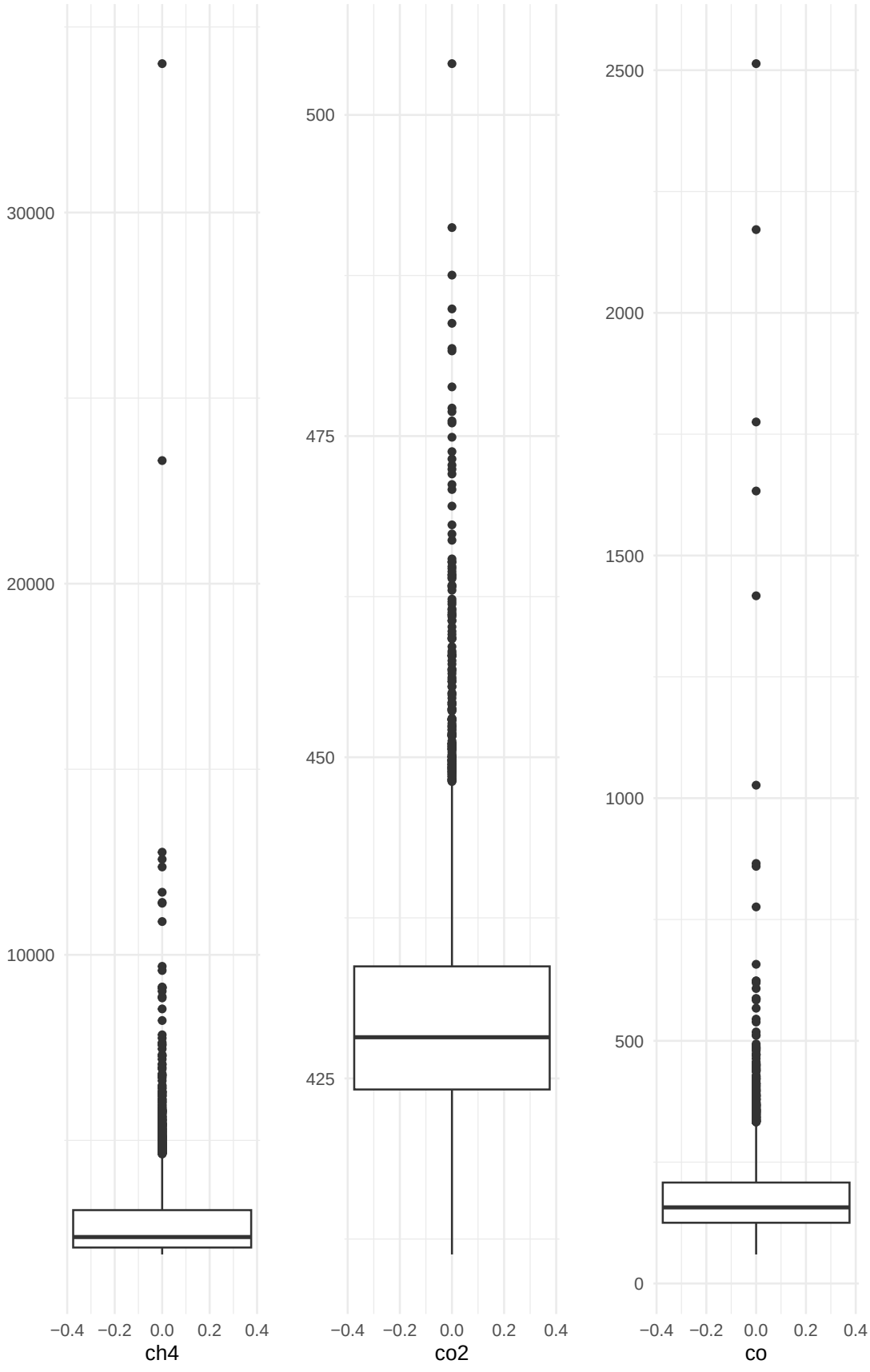
```
## Warning: Using an external vector in selections was deprecated in tidyselect 1.1.0.
## i Please use ‘all_of()‘ or ‘any_of()‘ instead.
##   # Was:
##   data %>% select(non_vocs)
##
##   # Now:
##   data %>% select(all_of(non_vocs))
##
## See <https://tidyselect.r-lib.org/reference/faq-external-vector.html>.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
# 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
hourly_non_vocs <- hourly_nona %>% select(all_of(non_vocs))

hourly_full_nona <- cbind(hourly_non_vocs, hourly_vocs)

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

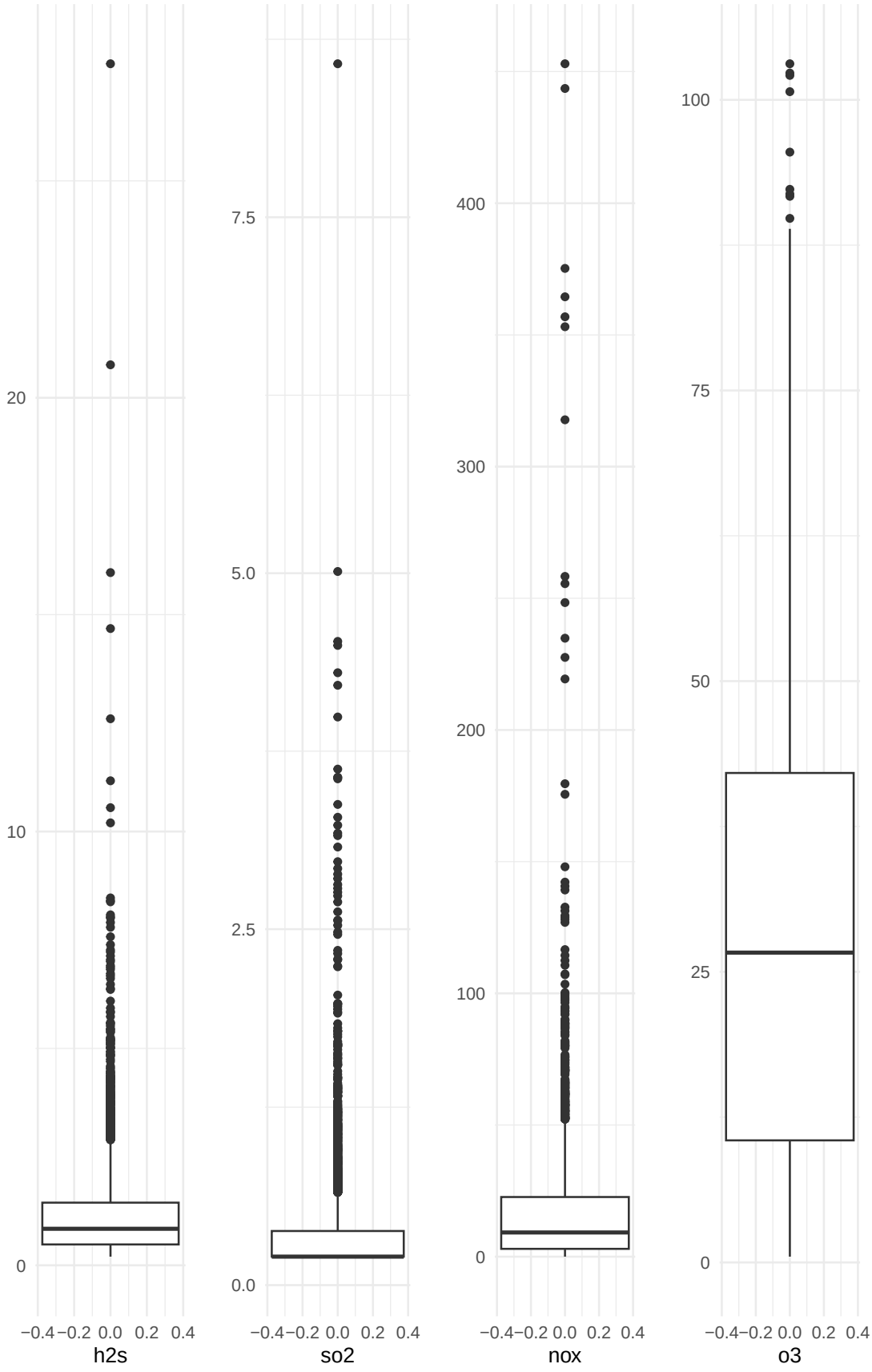
Data visualisation

```
non_vocs <- c('ch4', 'co2', 'co', 'h2s', 'so2', 'nox', 'o3')
for (compound in non_vocs) {
  assign(paste0(compound, '_boxplot'),
         ggplot(hourly_non_vocs) +
           geom_boxplot(aes(y = .data[[compound]])) +
```

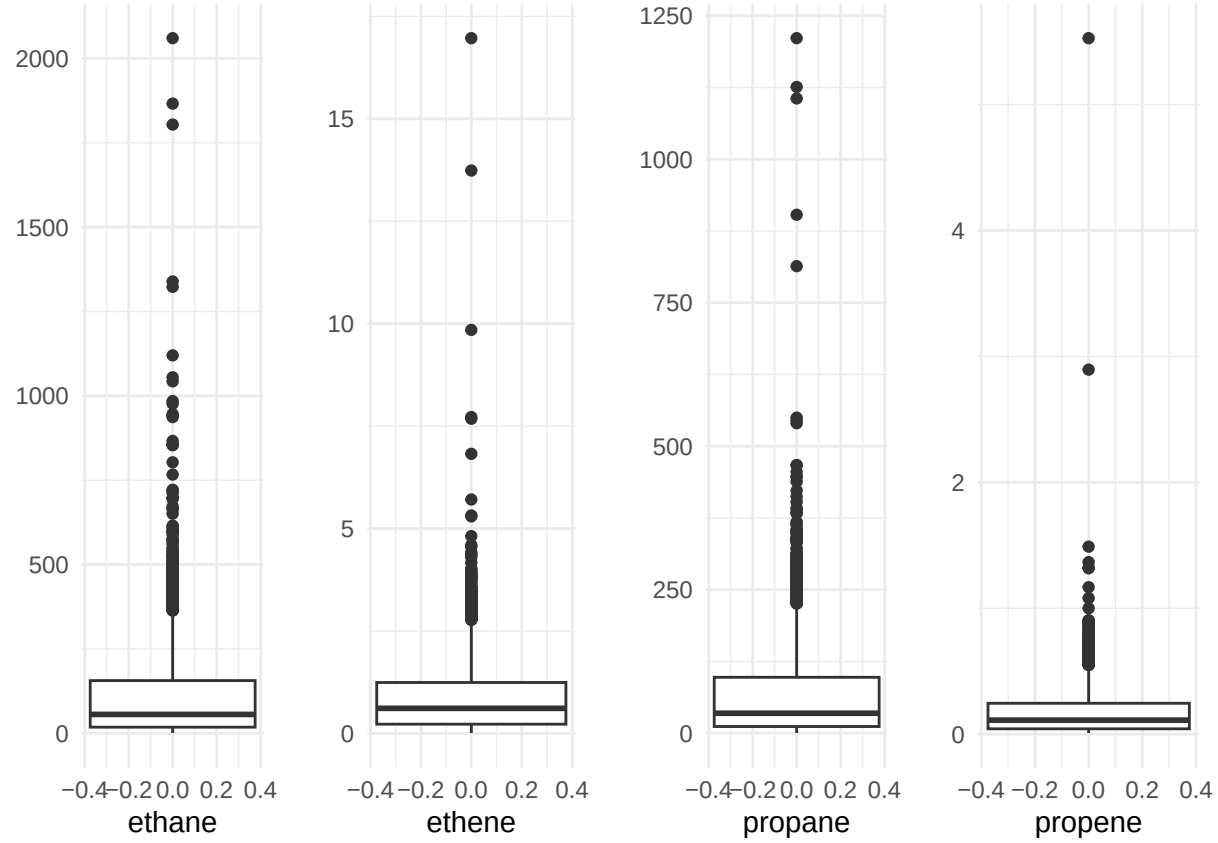
```
            labs(x = compound, y = '') +
            theme_minimal())
}

grid.arrange(ch4_boxplot, co2_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
2500
2000
1500
1000
500
0
−0.4 −0.2 0.0 0.2 0.4
co

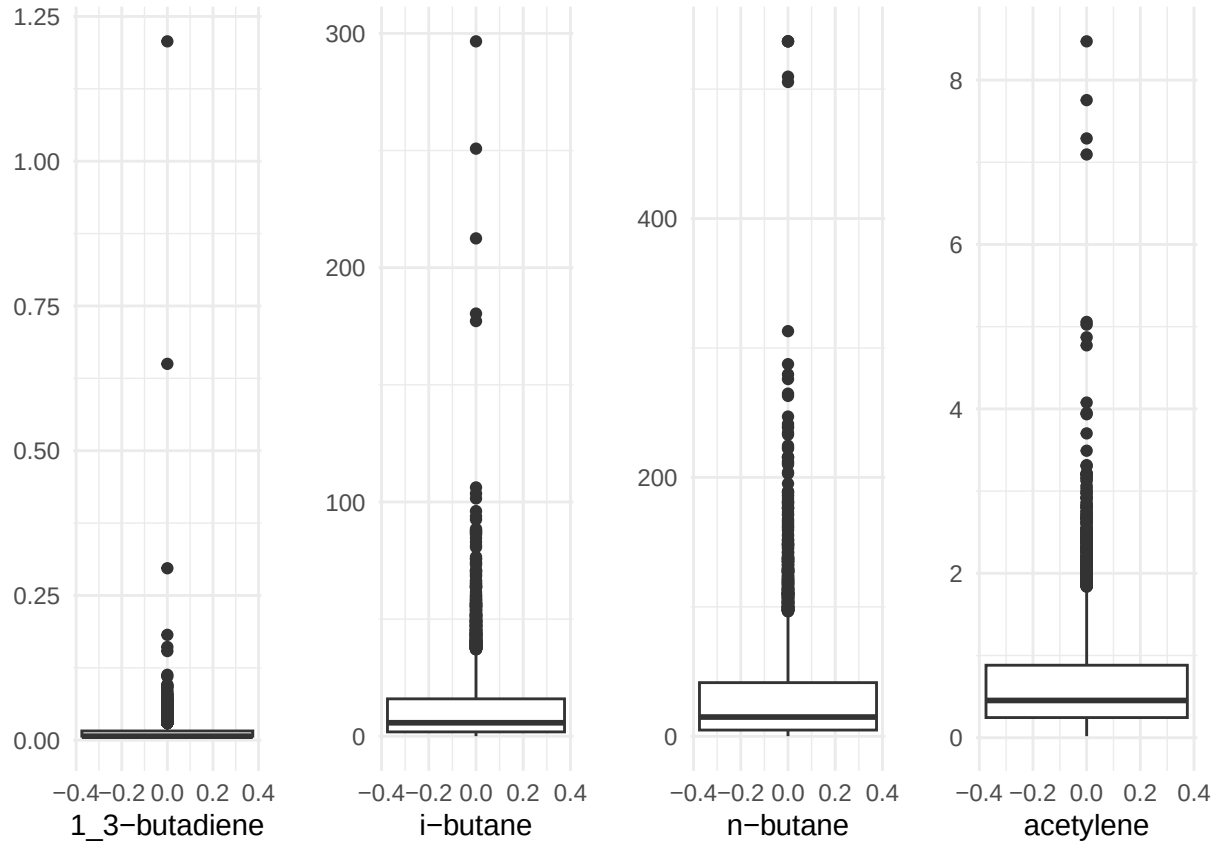
```r
grid.arrange(h2s_boxplot, so2_boxplot, nox_boxplot, o3_boxplot, nrow = 1)
```


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

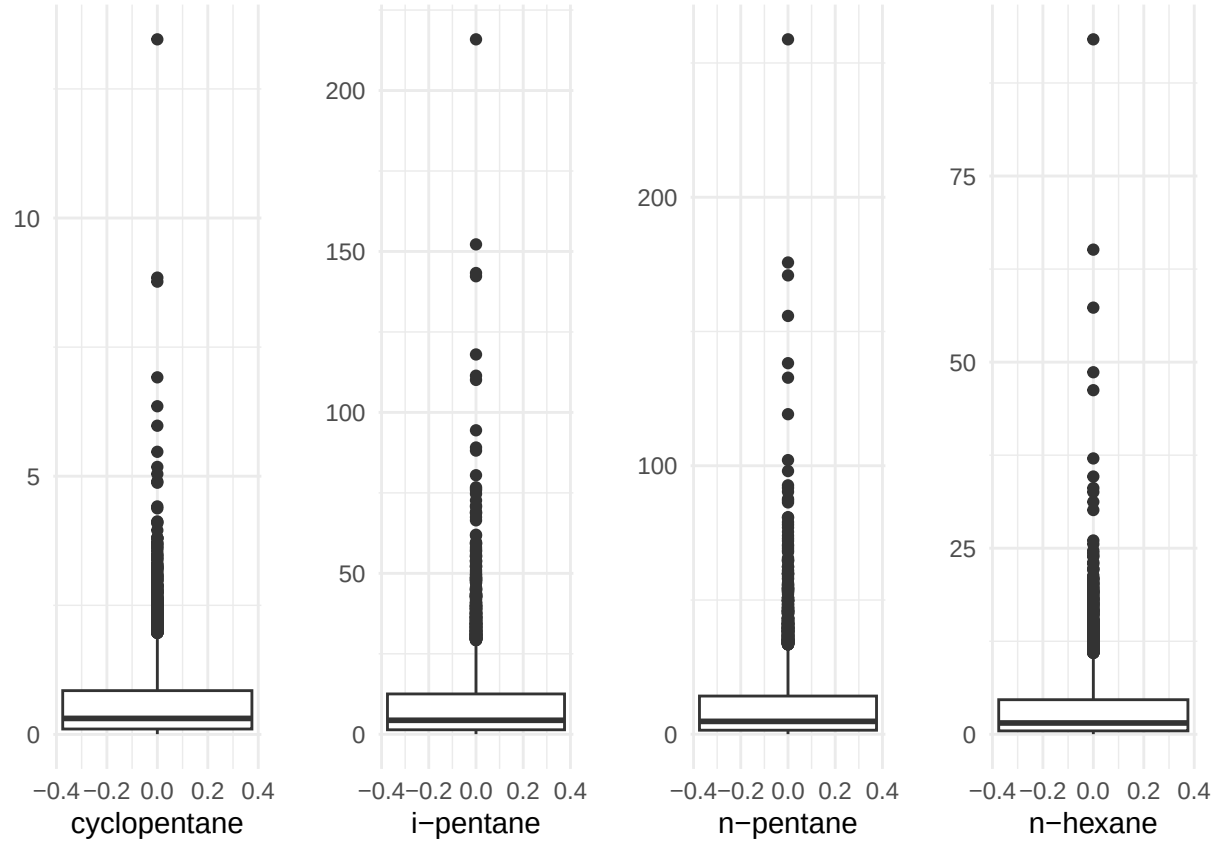
```
for (compound in vocs) {
  assign(paste0(compound, '_boxplot'),
         ggplot(hourly_vocs) +
           geom_boxplot(aes(y = .data[[compound]])) +
           labs(x = compound, y = '') +
           theme_minimal())
}
grid.arrange(get(paste0(vocs[1], '_boxplot')), get(paste0(vocs[2], '_boxplot')),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



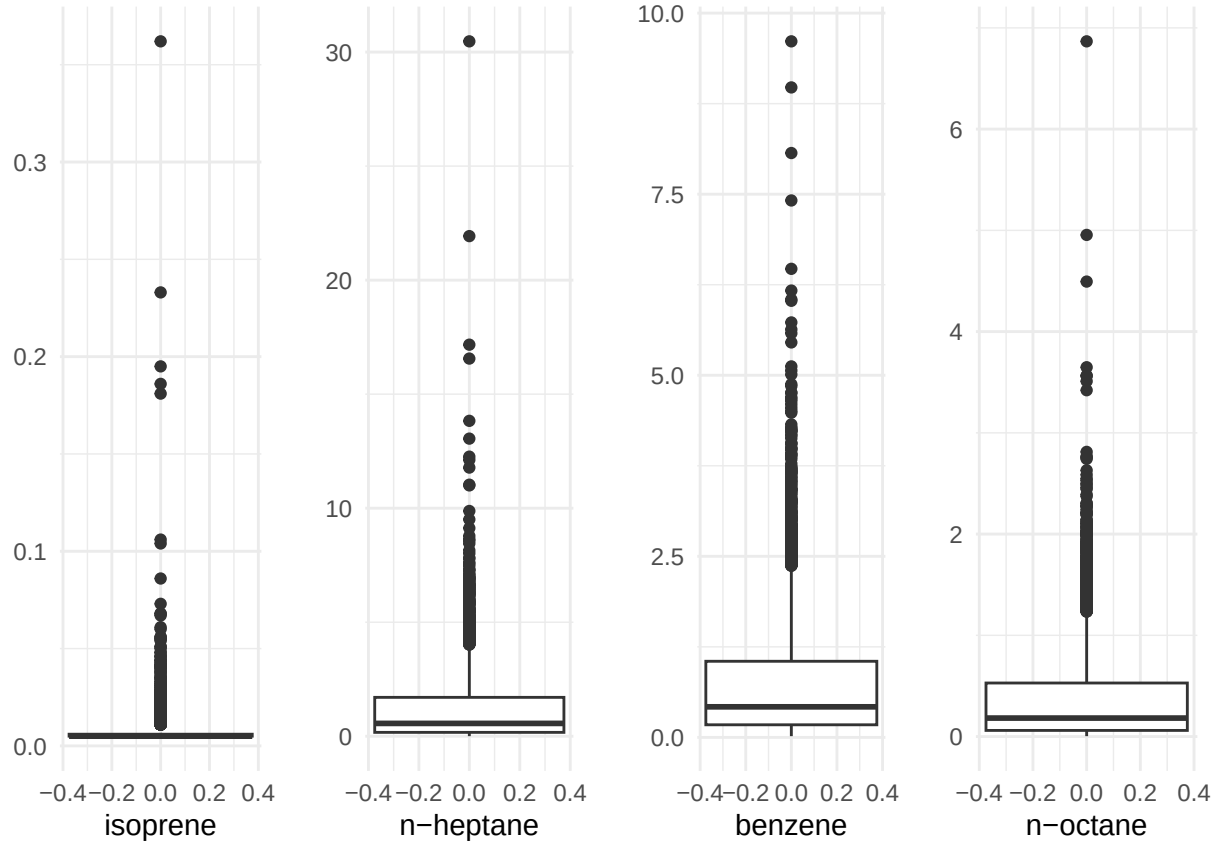

```
grid.arrange(get(paste0(vocs[5], '_boxplot')), get(paste0(vocs[6], '_boxplot')),
             get(paste0(vocs[7], '_boxplot')), get(paste0(vocs[8], '_boxplot')), nrow = 1)
```



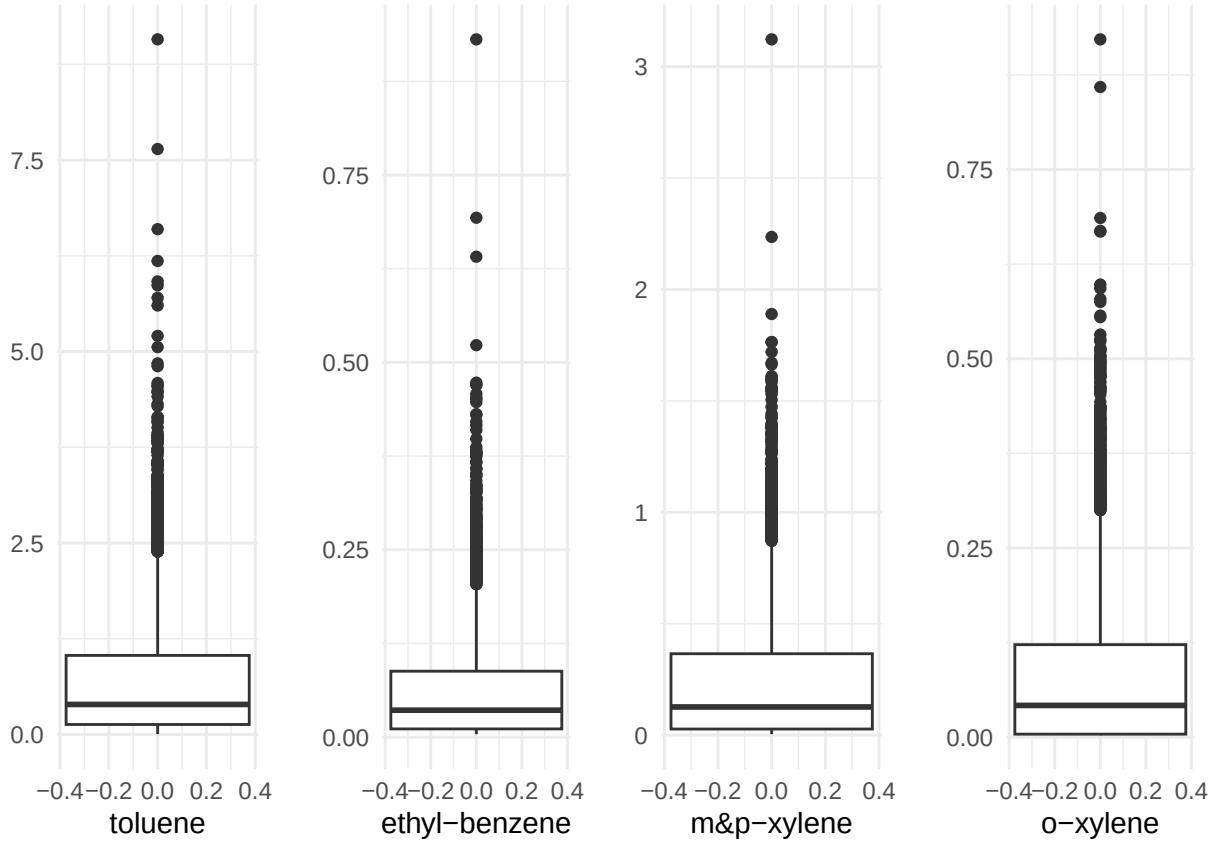

```
grid.arrange(get(paste0(vocs[9], '_boxplot')), get(paste0(vocs[10], '_boxplot')),
             get(paste0(vocs[11], '_boxplot')), get(paste0(vocs[12], '_boxplot')), nrow = 1)
```




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




```
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

```
# Define LOD for each chemical
LOD_non_voc <- c('ch4' = 0.9,
                 'co2' = 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            co           h2s           so2
##      1928.000       411.300        59.910         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	4788	1962.98700	6286.12400	34010.900	1928.000	1
co2	4788	416.47870	460.62260	503.990	411.300	1
co	4788	84.23050	442.08860	2513.440	59.910	1
h2s	4788	0.20000	5.20986	27.700	0.200	829
so2	4788	0.20000	1.78686	8.578	0.200	3266
nox	4788	0.22974	89.72371	452.959	0.025	2
o3	4788	0.50000	76.02600	103.100	0.500	259
ethane	4788	1.84422	526.44700	2060.000	0.916	1

name	N	1st percentile	99th percentile	Max	Background	# Background
ethene	4788	0.01100	3.50826	16.970	0.011	163
propane	4788	0.84674	300.79000	1211.000	0.224	1
propene	4788	0.00900	0.69739	5.528	0.009	411
1_3-butadiene	4788	0.00700	0.05900	1.207	0.007	3357
i-butane	4788	0.15148	60.89400	296.600	0.035	1
n-butane	4788	0.37248	166.52100	536.900	0.090	1
acetylene	4788	0.04900	2.61304	8.471	0.019	2
cyclopentane	4788	0.00500	3.06899	13.460	0.005	96
i-pentane	4788	0.10987	49.60210	215.900	0.038	1
n-pentane	4788	0.10487	55.95980	258.800	0.042	1
n-hexane	4788	0.04300	18.17780	93.360	0.021	2
isoprene	4788	0.00500	0.03313	0.362	0.005	2816
n-heptane	4788	0.01500	6.57669	30.470	0.004	5
benzene	4788	0.02800	3.78693	9.610	0.017	3
n-octane	4788	0.00400	2.00839	6.867	0.004	100
toluene	4788	0.01300	3.52165	9.077	0.004	11
ethyl-benzene	4788	0.00400	0.31613	0.931	0.004	918
m&p-xylene	4788	0.00400	1.29156	3.123	0.004	851
o-xylene	4788	0.00400	0.45700	0.922	0.004	1330

```
# PROCEDURE STEP 2:
#adjustments that were made according to paper
#William: I'm guessing this refers to Gunnar's paper section 2.2 and Guha 3.3

# Check whether chemical has background noise level that needs to be removed
# i.e, NO ADJUSTMENT if minimum value < 2*LOD and maximum value > 100*LOD
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)
}
```

```
# 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_full_nona[chemical], na.rm = TRUE),
         LODx2 = 2 * LOD,
         criterion1 = min(hourly_full_nona[chemical], na.rm = TRUE) < 2 * LOD,
         max = max(hourly_full_nona[chemical], na.rm = TRUE),
         LODx100 = 100 * LOD,
         criterion2 = max(hourly_full_nona[chemical], na.rm = TRUE) > 100 * LOD,
```

```
         adjusted_background = adjusting_neg_bg_from_lod(chemical, LOD, background, hourly_full_nona))
```

```
# 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)`
```

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

```
# So now we have the adjusted background concentrations
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[ac
```

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

```
##           ch4           co2            co           h2s           so2
##             1             1             1           829          3266
##           nox            o3        ethane        ethene       propane
##             0             0             1             0             1
##       propene 1_3-butadiene      i-butane      n-butane     acetylene
##             0          3357             1             1             0
##  cyclopentane     i-pentane     n-pentane      n-hexane      isoprene
##             0             1             1             2          2816
##     n-heptane       benzene      n-octane       toluene ethyl-benzene
##             0             0             0             0             0
##    m&p-xylene      o-xylene
##             0             0
```

```
# PROCEDURE STEP 3
# replace zero 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

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

```
# 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               co                h2s         
##  Min.   :0.000000   Min.   :0.0000   Min.   :0.00000   Min.   :0.00000  
##  1st Qu.:0.005795   1st Qu.:0.1384   1st Qu.:0.02592   1st Qu.:0.01022  
##  Median :0.014603   Median :0.1823   Median :0.03884   Median :0.02335  
##  Mean   :0.026837   Mean   :0.2000   Mean   :0.04761   Mean   :0.03500  
##  3rd Qu.:0.037200   3rd Qu.:0.2418   3rd Qu.:0.05970   3rd Qu.:0.04525  
##  Max.   :1.000000   Max.   :1.0000   Max.   :1.00000   Max.   :1.00000  
##       so2                nox                o3              ethane         
##  Min.   :0.000000   Min.   :0.000000   Min.   :0.00000   Min.   :0.000000  
##  1st Qu.:0.007997   1st Qu.:0.006534   1st Qu.:0.09747   1st Qu.:0.008386  
##  Median :0.016114   Median :0.020262   Median :0.25487   Median :0.026672  
##  Mean   :0.026320   Mean   :0.036440   Mean   :0.26676   Mean   :0.050993  
##  3rd Qu.:0.023633   3rd Qu.:0.049978   3rd Qu.:0.40546   3rd Qu.:0.075376  
##  Max.   :1.000000   Max.   :1.000000   Max.   :1.00000   Max.   :1.000000  
##      ethene            propane            propene          1_3-butadiene      
##  Min.   :0.00000   Min.   :0.000000   Min.   :0.000000   Min.   :0.000000  
##  1st Qu.:0.01268   1st Qu.:0.009285   1st Qu.:0.005979   1st Qu.:0.001667  
##  Median :0.03547   Median :0.028411   Median :0.018482   Median :0.004167  
##  Mean   :0.05042   Mean   :0.053805   Mean   :0.028772   Mean   :0.007368  
##  3rd Qu.:0.07266   3rd Qu.:0.080132   3rd Qu.:0.042761   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.:0.006153   1st Qu.:0.008783   1st Qu.:0.02674   1st Qu.:0.007432  
##  Median :0.019261   Median :0.027528   Median :0.05135   Median :0.022668  
##  Mean   :0.038384   Mean   :0.054906   Mean   :0.07436   Mean   :0.043730  
##  3rd Qu.:0.053703   3rd Qu.:0.077047   3rd Qu.:0.10211   3rd Qu.:0.062653  
##  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.   :0.000000   Min.   :0.000000  
##  1st Qu.:0.006293   1st Qu.:0.005681   1st Qu.:0.004725   1st Qu.:0.002801  
##  Median :0.019932   Median :0.018371   Median :0.016060   Median :0.005602
```

```
##  Mean   :0.041085   Mean   :0.038859   Mean   :0.035000   Mean   :0.010304
##  3rd Qu.:0.057848   3rd Qu.:0.054837   3rd Qu.:0.049564   3rd Qu.:0.011204
##  Max.   :1.000000   Max.   :1.000000   Max.   :1.000000   Max.   :1.000000
##    n-heptane           benzene           n-octane           toluene
##  Min.   :0.000000   Min.   :0.00000   Min.   :0.000000   Min.   :0.00000
##  1st Qu.:0.005473   1st Qu.:0.01637   1st Qu.:0.008269   1st Qu.:0.01389
##  Median :0.018348   Median :0.04222   Median :0.026009   Median :0.04276
##  Mean   :0.039328   Mean   :0.07655   Mean   :0.054341   Mean   :0.07825
##  3rd Qu.:0.055866   3rd Qu.:0.10779   3rd Qu.:0.076497   3rd Qu.:0.11333
##  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.:0.007551   1st Qu.:0.007374   1st Qu.:0.00000
##  Median :0.034520   Median :0.039115   Median :0.04139
##  Mean   :0.062378   Mean   :0.077508   Mean   :0.08650
##  3rd Qu.:0.090615   3rd Qu.:0.115742   3rd Qu.:0.12881
##  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

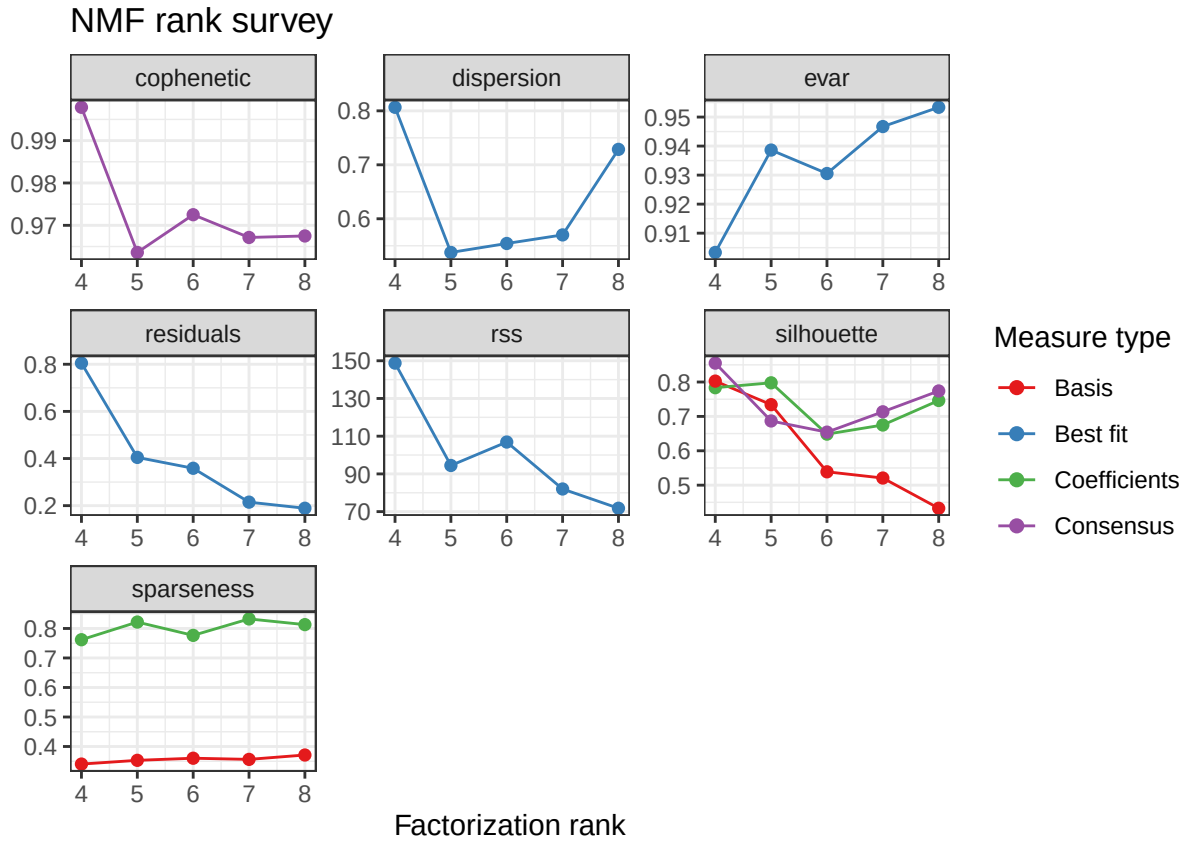
```
# compute weight matrix (uncertainties)
# Based on the Guha paper
# next comment is from the other nmf R file
weight_matrix <- matrix(0, nrow = nrow(normalized_matrix), ncol = ncol(normalized_matrix))
LOD_merged <- tibble(chemical = c(adjusted_background_non_voc$chemical, adjusted_background_voc$chemical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

# write_csv(as_tibble(weight_matrix) %>% setNames(colnames(normalized_matrix)) %>% select(-o3) %>% muta
```

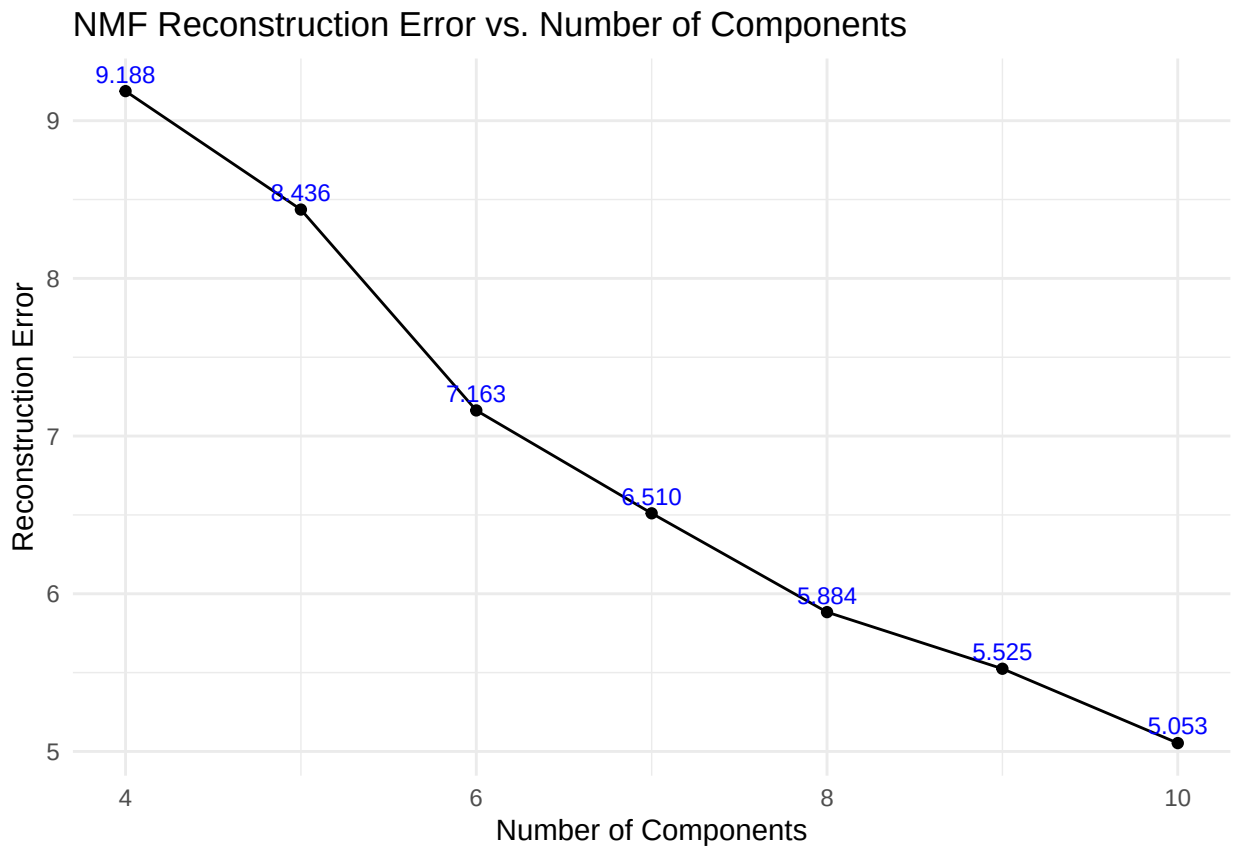
```
# plots the NMF rank survey
plot(estimate_rank)
```



Source contributions

NMF - SVD seed

```
# 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')
```



Method comparisons

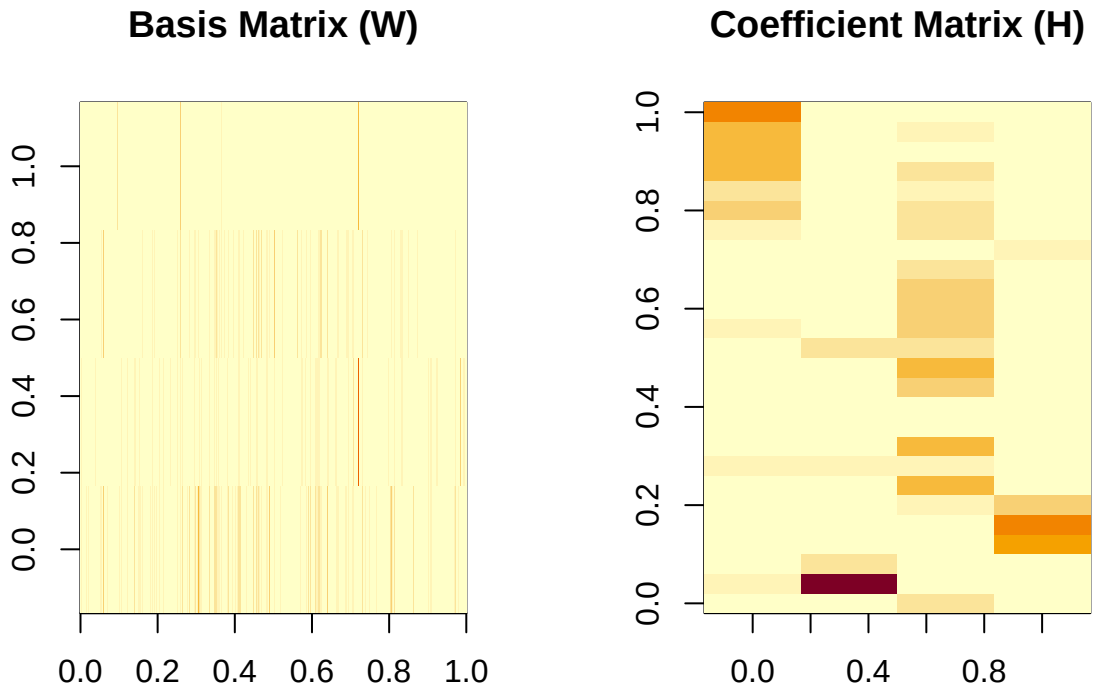
Remove Ozone

4 Components

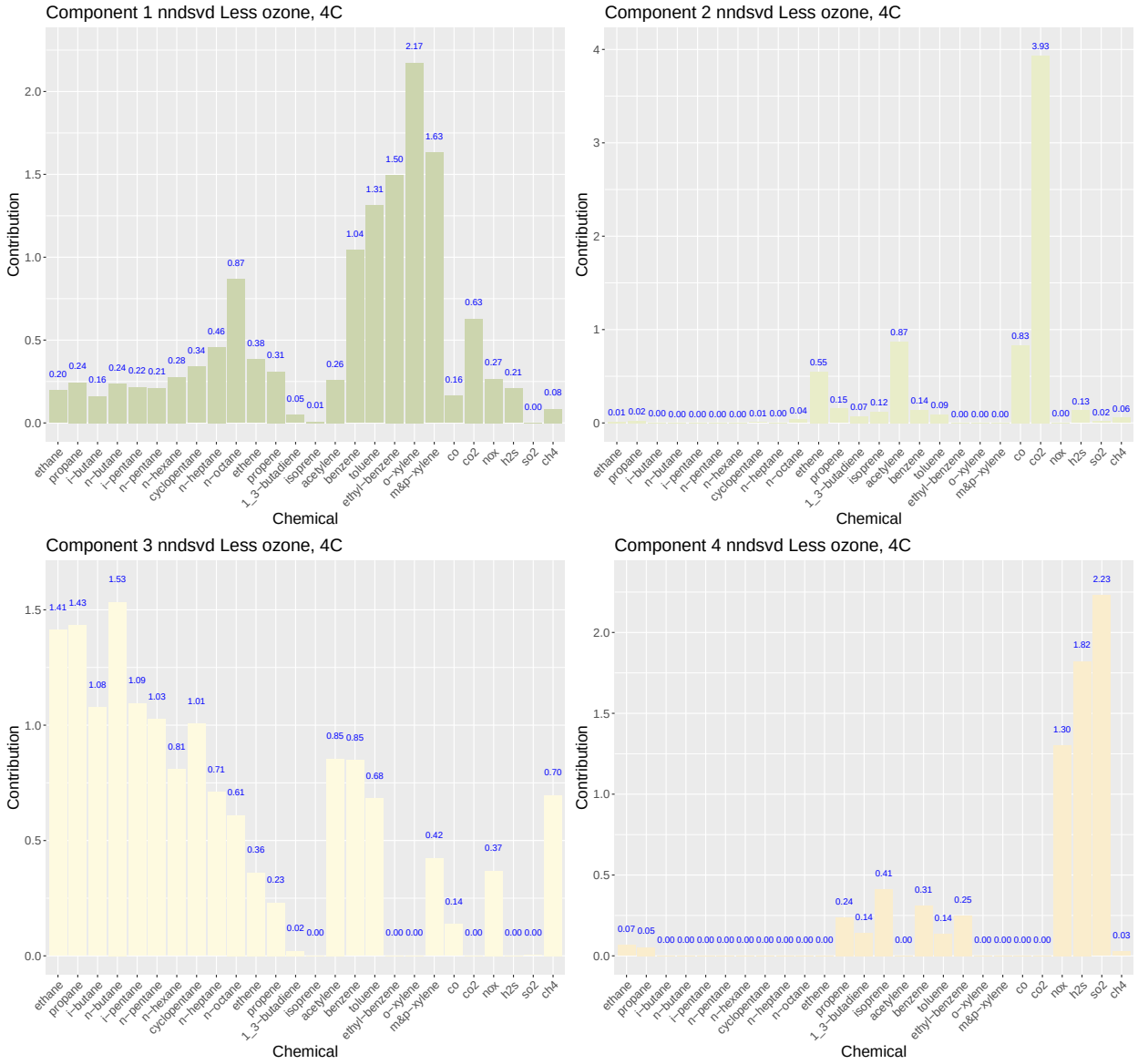
```
normalized_matrix_less_o3 <- normalized_matrix[ ,setdiff(colnames(normalized_matrix), "o3")]
# Store this for other use
# write_csv(as_tibble(normalized_matrix_less_o3) %>% mutate(time_utc = hourly_nona$time_utc), 'normaliz

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.
```

```
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)")
```




```
# Convert H to a data frame for ggplot
H_df_4c_less_o3 <- as.data.frame(coef_matrix_4c_less_o3)
# Add a column for chemicals
H_df_4c_less_o3$Component <- rownames(H_df_4c_less_o3)

# Reshape data to long format
H_long_4c_less_o3 <- pivot_longer(H_df_4c_less_o3, cols = -Component, names_to = "Chemical", values_to =

# Plot
nmfplt_1_svd_4c_less_o3 <- get_component_plot(H_long_4c_less_o3,
                                              '1', 'Component 1 nndsvd Less ozone, 4C')
nmfplt_2_svd_4c_less_o3 <- get_component_plot(H_long_4c_less_o3,
                                              '2', 'Component 2 nndsvd Less ozone, 4C')
nmfplt_3_svd_4c_less_o3 <- get_component_plot(H_long_4c_less_o3,
                                              '3', 'Component 3 nndsvd Less ozone, 4C')
nmfplt_4_svd_4c_less_o3 <- get_component_plot(H_long_4c_less_o3,
                                              '4', 'Component 4 nndsvd Less ozone, 4C')
```



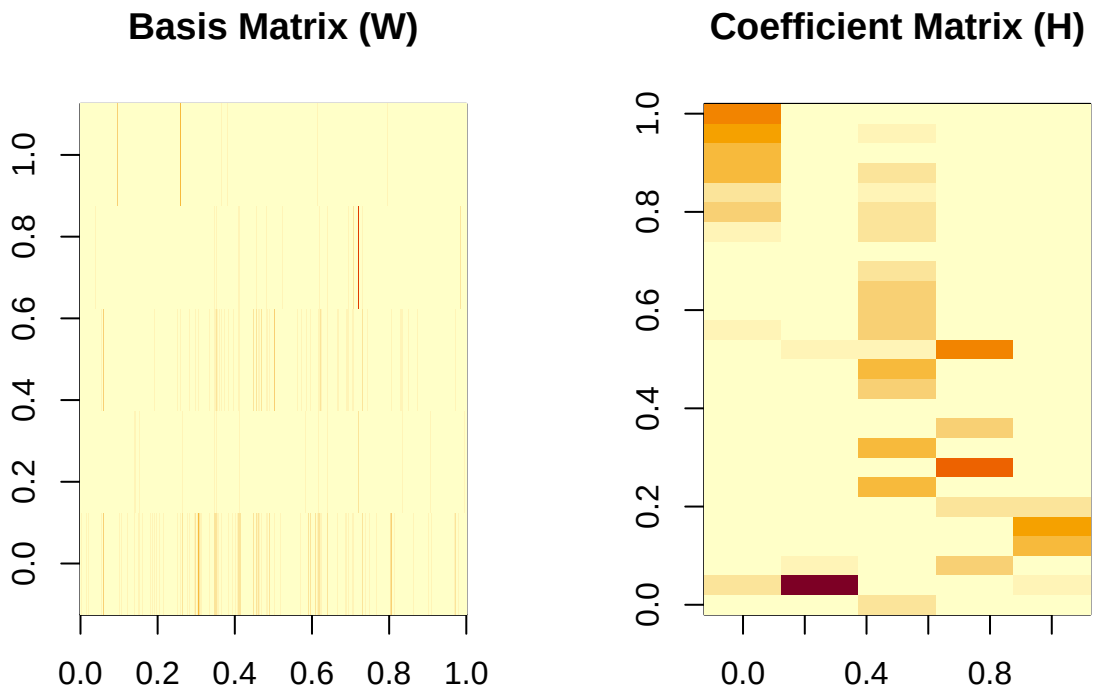
5 Components

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

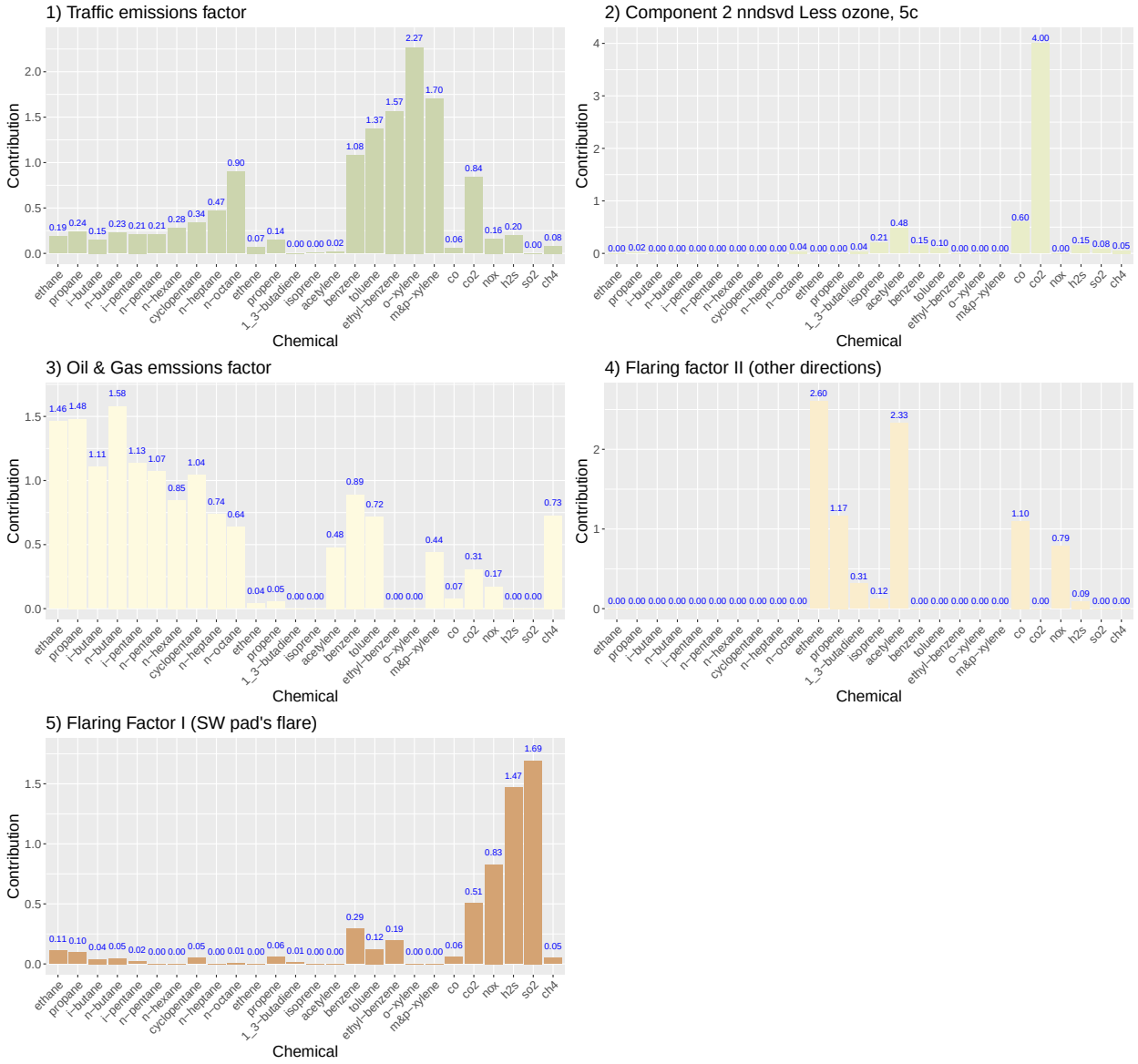
```
## Warning in sqrt(S[i] * termn) * uun: Recycling array of length 1 in array-vector arithmetic is depre
##   Use c() or as.vector() instead.

## 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.
```

```
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)")
```



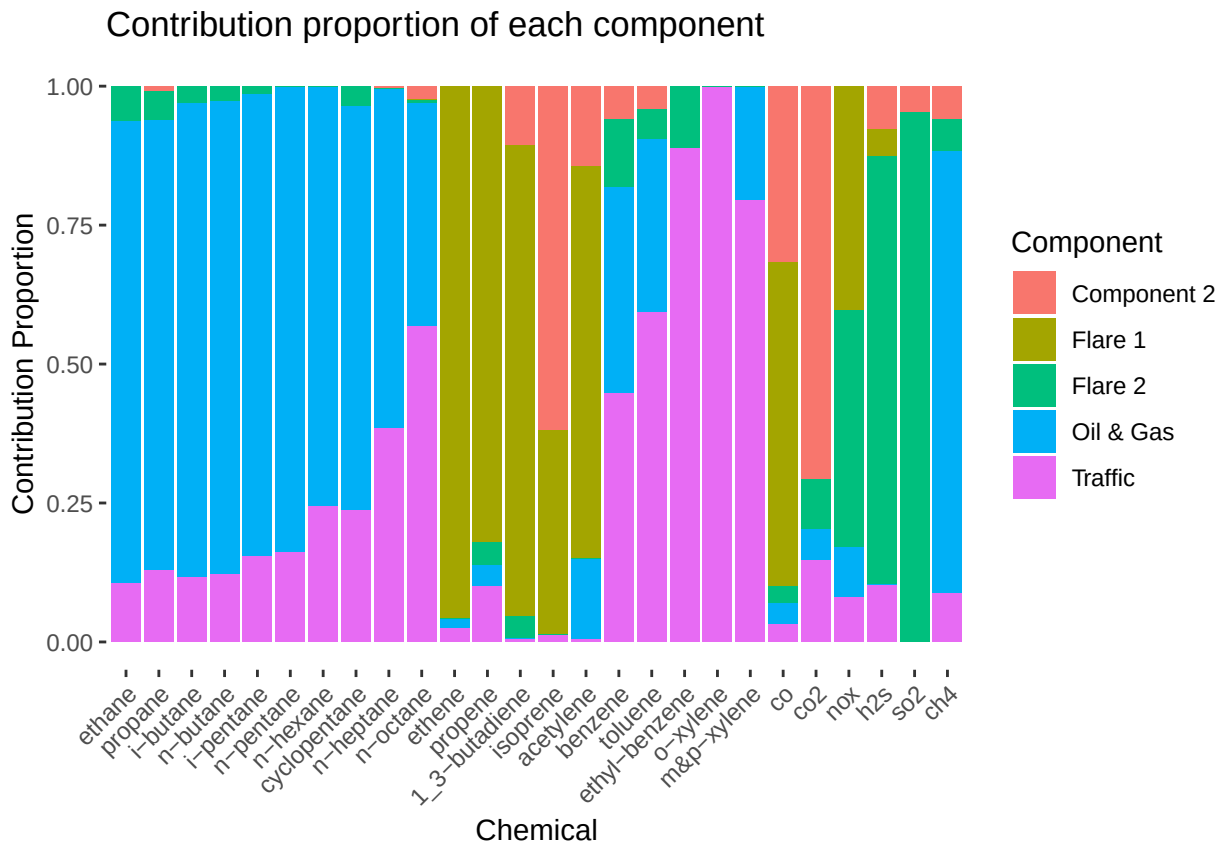
```r
# Convert H to a data frame for ggplot
H_df_5c_less_o3 <- as.data.frame(coef_matrix_5c_less_o3)
# Add a column for chemicals
H_df_5c_less_o3$Component <- rownames(H_df_5c_less_o3)

# Reshape data to long format
H_long_5c_less_o3 <- pivot_longer(H_df_5c_less_o3, cols = -Component, names_to = "Chemical", values_to =

# Plot
nmfplt_1_svd_5c_less_o3 <- get_component_plot(H_long_5c_less_o3,
                                              '1', '1) Traffic emissions factor')
nmfplt_2_svd_5c_less_o3 <- get_component_plot(H_long_5c_less_o3,
                                              '2', '2) Component 2 nndsvd Less ozone, 5c')
nmfplt_3_svd_5c_less_o3 <- get_component_plot(H_long_5c_less_o3,
                                              '3', '3) Oil & Gas emssions factor')
nmfplt_4_svd_5c_less_o3 <- get_component_plot(H_long_5c_less_o3,
                                              '4', '4) Flaring factor II (other directions)')
nmfplt_5_svd_5c_less_o3 <- get_component_plot(H_long_5c_less_o3,
                                              '5', '5) Flaring Factor I (SW pad\'s flare)')
```



Fingerprint plot

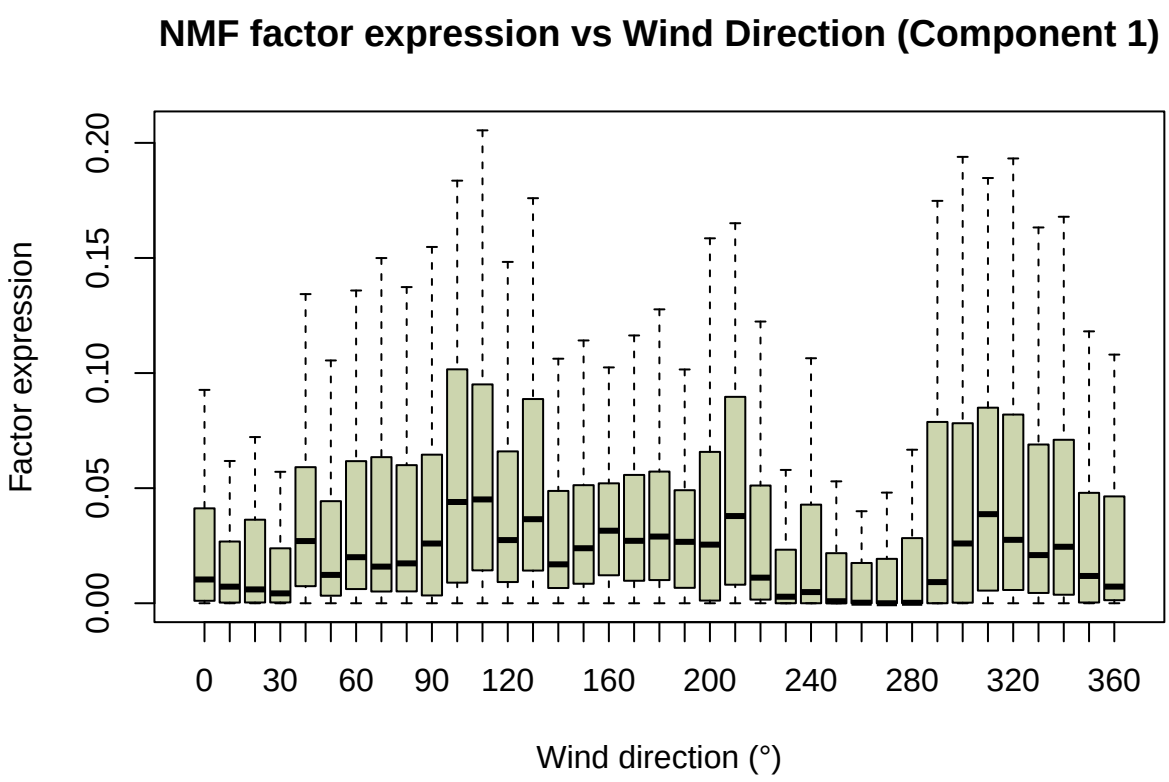
```
contrib_prop <- apply(H_df_5c_less_o3[,1:(length(H_df_5c_less_o3)-1)], MARGIN = 2, FUN = function(x) {x,
contrib_prop %>%
  as_tibble() %>%
  mutate(Component = c('Traffic', 'Component 2', 'Oil & Gas', 'Flare 1', 'Flare 2')) %>%
  pivot_longer(cols = -Component, names_to = "Chemical", values_to = "Contribution_prop") %>%
  mutate(Chemical = factor(Chemical, levels = desired_order)) %>%
  ggplot(aes(fill=Component, y=Contribution_prop, x=Chemical)) +
  geom_bar(position="fill", stat="identity") +
  theme(axis.text.x = element_text(angle = 45, hjust = 1)) +
  labs(x = "Chemical", y = "Contribution Proportion",
       title = 'Contribution proportion of each component') +
  theme(panel.grid.major = element_blank(),
```

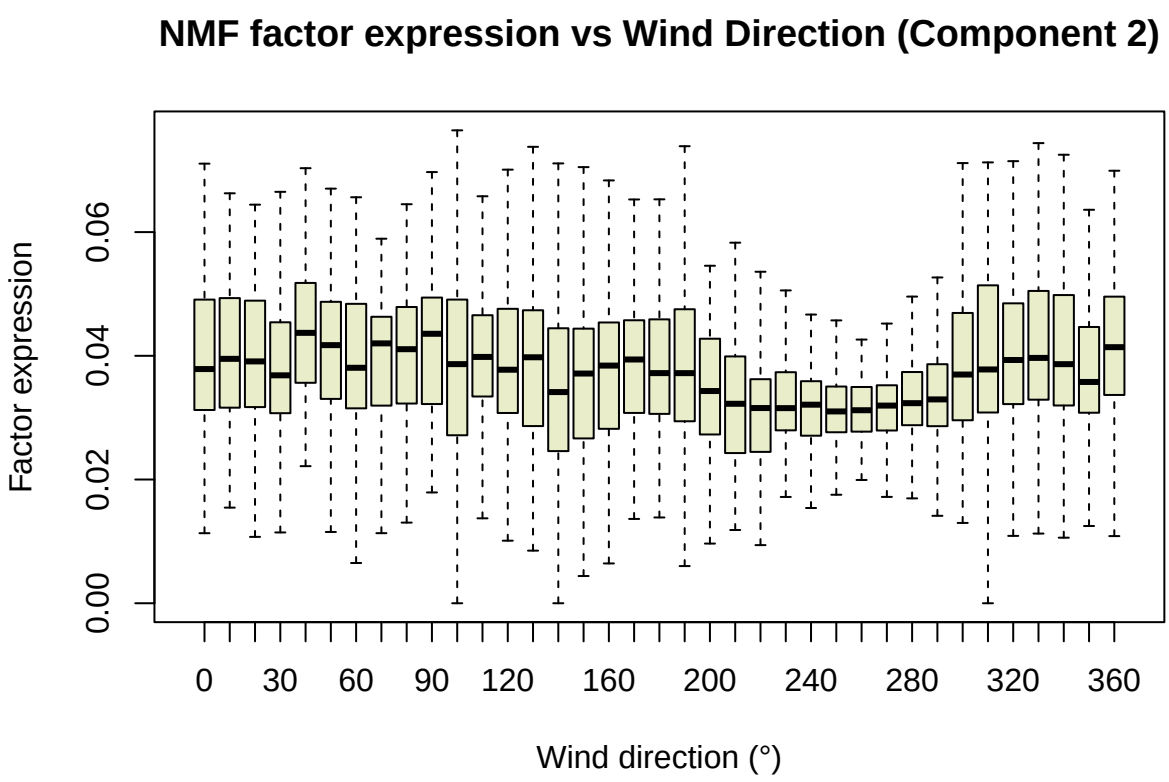
```
        panel.grid.minor = element_blank(),
        panel.background = element_blank())
```

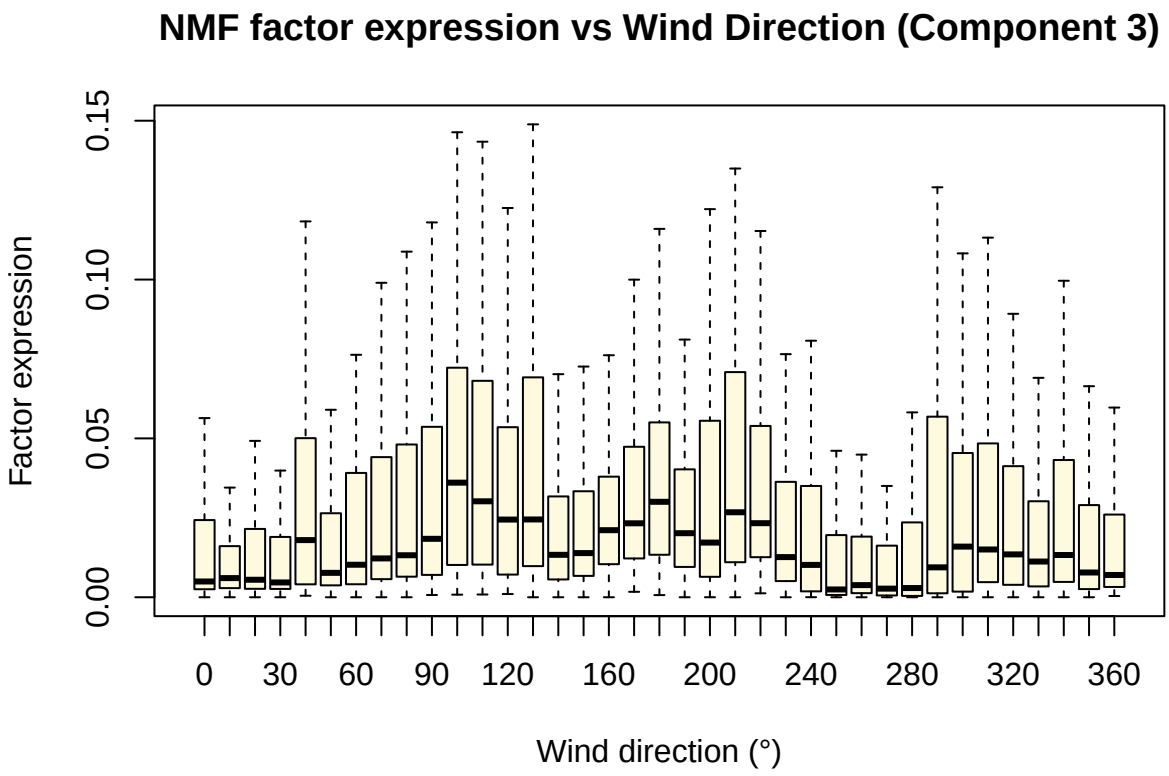


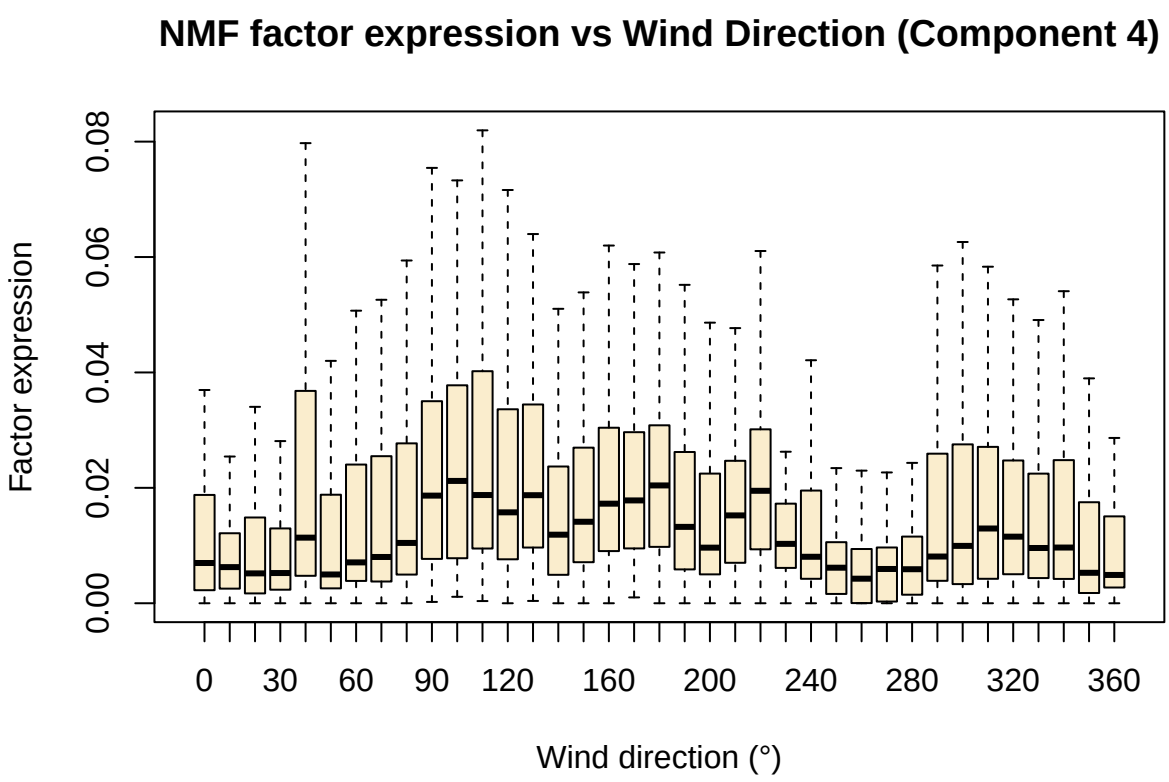
Wind plots

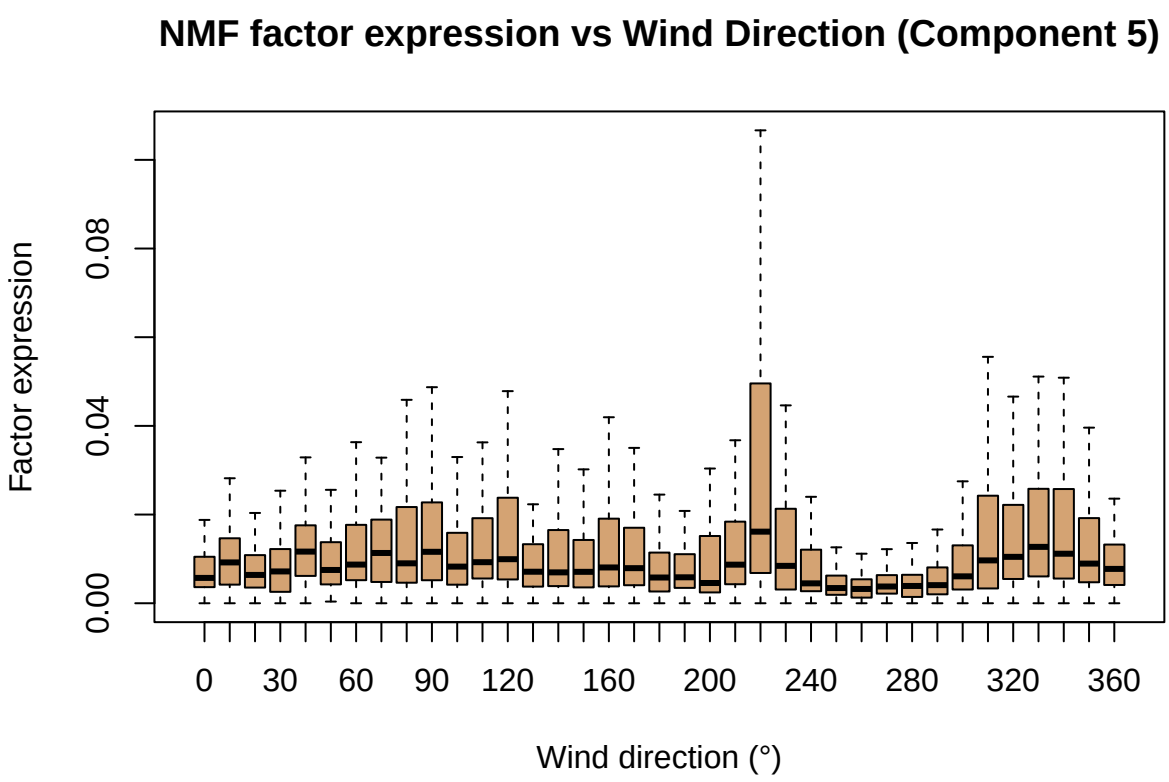
```
hourly_wind_nona <- hourly_nona %>%
  select(wdr_deg, wsp_ms)

data_to_plot <- tibble(
  component1 = basis(nmf_result_5c_less_o3)[,1],
  component2 = basis(nmf_result_5c_less_o3)[,2],
  component3 = basis(nmf_result_5c_less_o3)[,3],
  component4 = basis(nmf_result_5c_less_o3)[,4],
  component5 = basis(nmf_result_5c_less_o3)[,5],
  wd = round(hourly_wind_nona$wdr_deg, -1)
)
```


NMF factor expression vs Wind Direction (Component 1)
Factor expression
0.00
0.05
0.10
0.15
0.20
0
30
60
90
120
160
200
240
280
320
360
Wind direction (°)


NMF factor expression vs Wind Direction (Component 2)
Factor expression
0.00
0.02
0.04
0.06
0
30
60
90
120
160
200
240
280
320
360
Wind direction (°)


NMF factor expression vs Wind Direction (Component 3)
Factor expression
0.00
0.05
0.10
0.15
0
30
60
90
120
160
200
240
280
320
360
Wind direction (°)


NMF factor expression vs Wind Direction (Component 4)
Factor expression
0.00
0.02
0.04
0.06
0.08
0
30
60
90
120
160
200
240
280
320
360
Wind direction (°)



Factor analysis

```
# First look at how well this approximates
fitted_5c_less_o3 <- fitted(nmf_result_5c_less_o3)
sum(abs(normalized_matrix_less_o3-fitted_5c_less_o3))
```

```
## [1] 1060.414
```

```
# NMF factorizes V = WH
# Store Basis matrix (W) and Coef Matrix (H)
saveRDS(basis_matrix_5c_less_o3, 'result_rfiles/nmf_norm_5c_less_o3_basis.rds')
saveRDS(coef_matrix_5c_less_o3, 'result_rfiles/nmf_norm_5c_less_o3_coef.rds')

# Merge basis matrix into hourly observations
basis_matrix_5c_less_o3 <- as_tibble(basis_matrix_5c_less_o3) %>%
  setNames(c('Factor1', 'Factor2', 'Factor3', 'Factor4', 'Factor5'))
```

```
## Warning: The ‘x‘ argument of ‘as_tibble.matrix()‘ must have unique column names if
## ‘.name_repair‘ is omitted as of tibble 2.0.0.
## i Using compatibility ‘.name_repair‘.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```r
normalized_hourly_data_5c_less_o3 <- hourly_nona[,c('day', 'time_utc')] %>%
  cbind(normalized_matrix_less_o3) %>%
  cbind(basis_matrix_5c_less_o3) %>%
  right_join(hourly_data %>% select(-'day'), join_by(time_utc), suffix = c('_norm', ''))

# saveRDS(normalized_full_data_5c_less_o3, 'result_rfiles/normalized_hourly_data_5c_less_o3.rds')
```

```r
# Also compute a daily dataset
normalized_daily_data_5c_less_o3 <- normalized_hourly_data_5c_less_o3 %>%
  group_by(day) %>%
  summarise(across(where(is.numeric), ~ mean(.x, na.rm = T)))

# saveRDS(normalized_daily_data_5c_less_o3, 'result_rfiles/normalized_daily_data_5c_less_o3.rds')
```

```r
# Check if relationship between # flares and flare factor (4 & 5)

# Linear model
flare_factor <- lm(n_flare_100 ~ Factor4 + Factor5, data = normalized_daily_data_5c_less_o3)
summary(flare_factor)
```

```
##
## Call:
## lm(formula = n_flare_100 ~ Factor4 + Factor5, data = normalized_daily_data_5c_less_o3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.4658 -3.0946 -0.3795  2.2016 17.1266
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    3.592      0.456   7.878 7.71e-14 ***
## Factor4        6.625     20.357   0.325    0.745
## Factor5       42.500     27.706   1.534    0.126
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.787 on 276 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.01269,    Adjusted R-squared:  0.005536
## F-statistic: 1.774 on 2 and 276 DF,  p-value: 0.1716
```

```r
flare_factor_weighted <- lm(weighted.count ~ Factor4 + Factor5, data = normalized_daily_data_5c_less_o3)
summary(flare_factor_weighted)
```

```
##
## Call:
## lm(formula = weighted.count ~ Factor4 + Factor5, data = normalized_daily_data_5c_less_o3)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -10.209  -3.167  -0.377   1.832 120.250
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    5.030      1.007   4.996 1.04e-06 ***
## Factor4     -103.752     44.944  -2.308  0.02171 *
## Factor5      193.540     61.168   3.164  0.00173 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 8.361 on 276 degrees of freedom
##   (1 observation deleted due to missingness)
## Multiple R-squared:  0.03869,    Adjusted R-squared:  0.03173
## F-statistic: 5.554 on 2 and 276 DF,  p-value: 0.004316
```

```
# Poisson model
flare_factor_pois <- glm(n_flare_100 ~ Factor4 + Factor5, family = 'poisson', data = normalized_daily_da
summary(flare_factor_pois)
```

```
##
## Call:
## glm(formula = n_flare_100 ~ Factor4 + Factor5, family = "poisson",
##     data = normalized_daily_data_5c_less_o3)
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   1.2970     0.0586  22.132  < 2e-16 ***
## Factor4       1.6290     2.5155   0.648  0.51724
## Factor5       9.2161     3.2936   2.798  0.00514 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 1022.1  on 278  degrees of freedom
## Residual deviance: 1010.7  on 276  degrees of freedom
##   (1 observation deleted due to missingness)
## AIC: 1758
##
## Number of Fisher Scoring iterations: 5
```

```
# Check relationship between avg flare distance and flare factor (4 & 5)
# Linear model
flare_factor_dist <- lm(distToLovi ~ Factor4 + Factor5, data = normalized_daily_data_5c_less_o3)
summary(flare_factor_dist)
```

```
##
## Call:
## lm(formula = distToLovi ~ Factor4 + Factor5, data = normalized_daily_data_5c_less_o3)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17.8872  -4.0924  -0.6397   3.1281  15.8871
##
## Coefficients:
```

```
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  20.3055     0.8902  22.809   <2e-16 ***
## Factor4      78.3034    40.2421   1.946    0.053 .
## Factor5     -61.7593    51.8998  -1.190    0.235
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.536 on 218 degrees of freedom
##   (59 observations deleted due to missingness)
## Multiple R-squared:  0.01769,    Adjusted R-squared:  0.008681
## F-statistic: 1.963 on 2 and 218 DF,  p-value: 0.1429
```

6 Components

Wind plots

Remove Ozone + chemicals with more than 500+ background values

4 Components

5 Components

Compare to 4 components