HMM search output analyses

Executing HMM search output analyses with max e-val:s

LJM

2020-01-03

Contents

This document contains the final analyses and visualialisations of HMMer search with eggNOG viral HMM profiles on fOTUs.

1 Load libraries

```
library(tidyverse)
#library(devtools)
#install_github("vqv/ggbiplot")
library(ggbiplot)
#install.packages("vegan")
library(vegan)
```

2 Read intermediary csv file

In the intermediary csv file the $-log_{10}$(e-val):s are the maximum e-values among all the hits to the fOTUs instead (and in contrast to the previous run in file 2019-11-25_hmmsearch_output_analyses.Rmd) of the average of all the HMMsearch hits to the fOTUs.

```
columns <- paste("c", strrep("d",8315), sep = "")
fOTUsHMM <- read_csv(file = "../analyses/HMMsearch/max_fOTUsHMM.csv", col_types = columns)
```

3 Exchange NA, Inf and -Inf with 0

```
is.na(fOTUsHMM) <- sapply(fOTUsHMM, is.infinite)
fOTUsHMM[is.na(fOTUsHMM)] <- 0
```

4 Move fOTU names to row names and execute PCA

```
fOTUsHMM_pca <- fOTUsHMM %>%
  column_to_rownames(var = "fOTU_name") %>%
  prcomp(.,
         center = TRUE)
```

5 Visualise the results

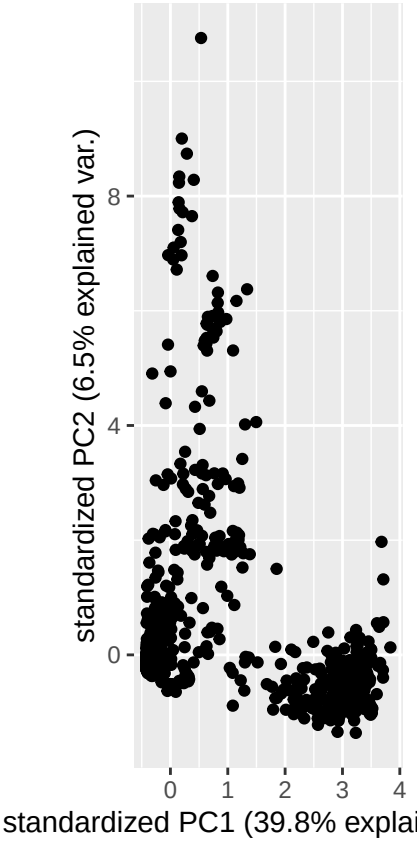
5.1 PCA plots

These PCA plots can be used to visually determine potential groupings.

```
file_name <- "../visualisations/HMMsearch/ggbiplot_max_PC1-2.png"
png(filename = file_name)
ggbiplot(fOTUsHMM_pca, var.axes = F, choices = 1:2)
dev.off()
```

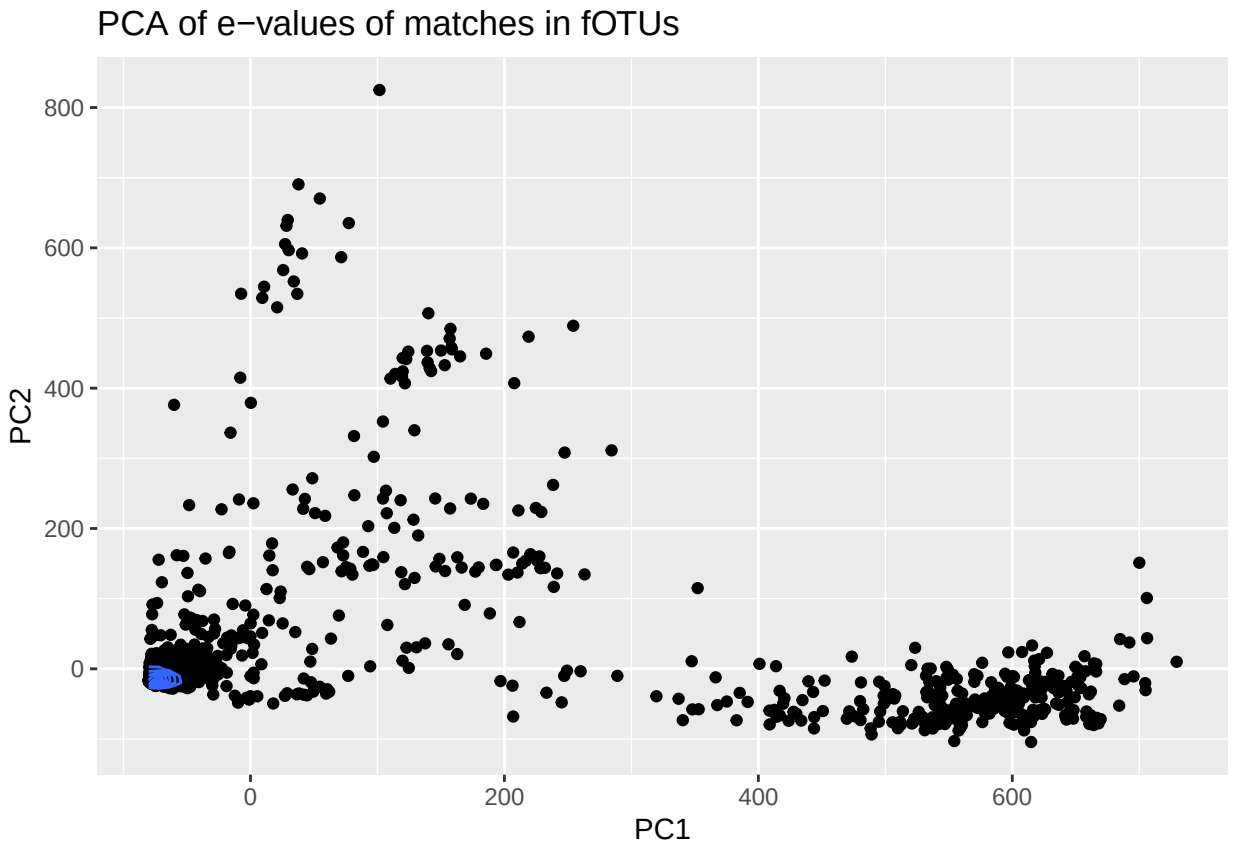
```
## pdf
##   2
```

```
ggbiplot(fOTUsHMM_pca, var.axes = F, choices = 1:2)
```

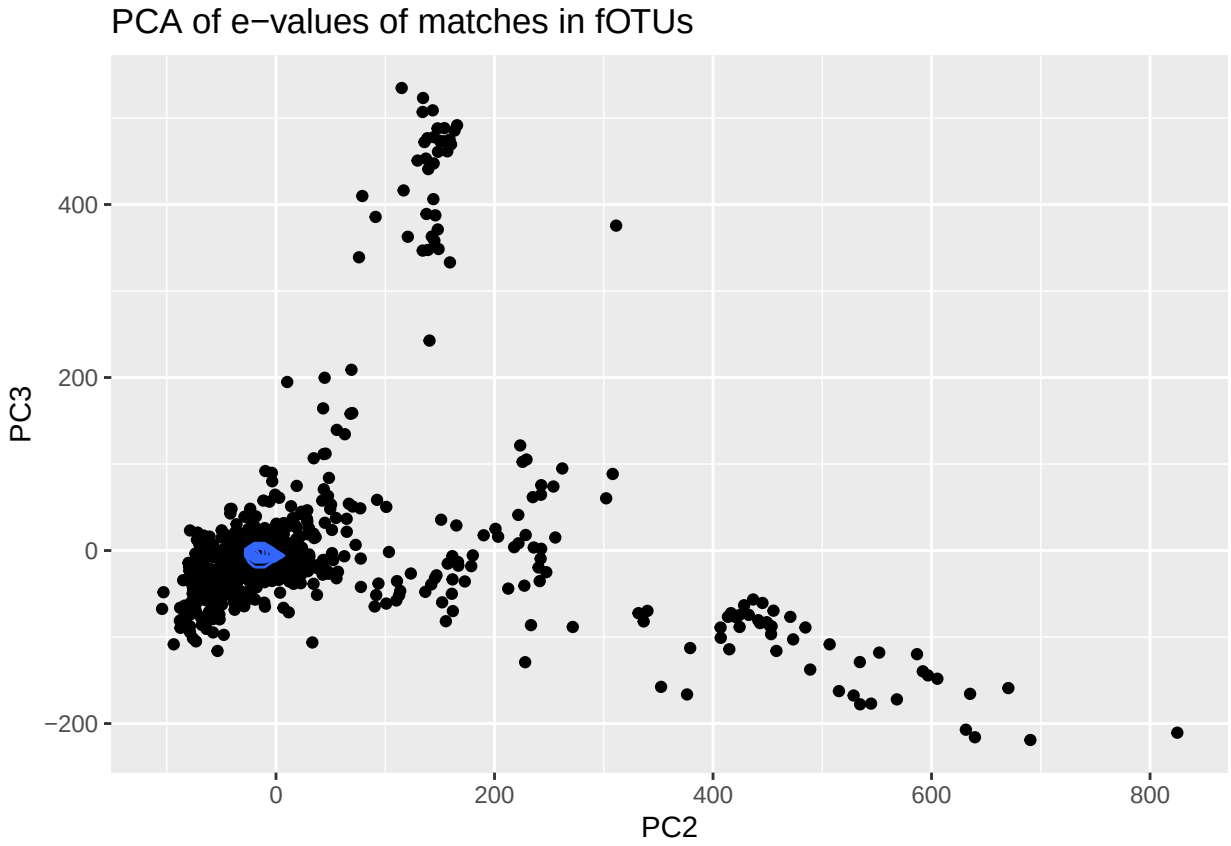


Here above the plot is slightly squashed but the following aren't:

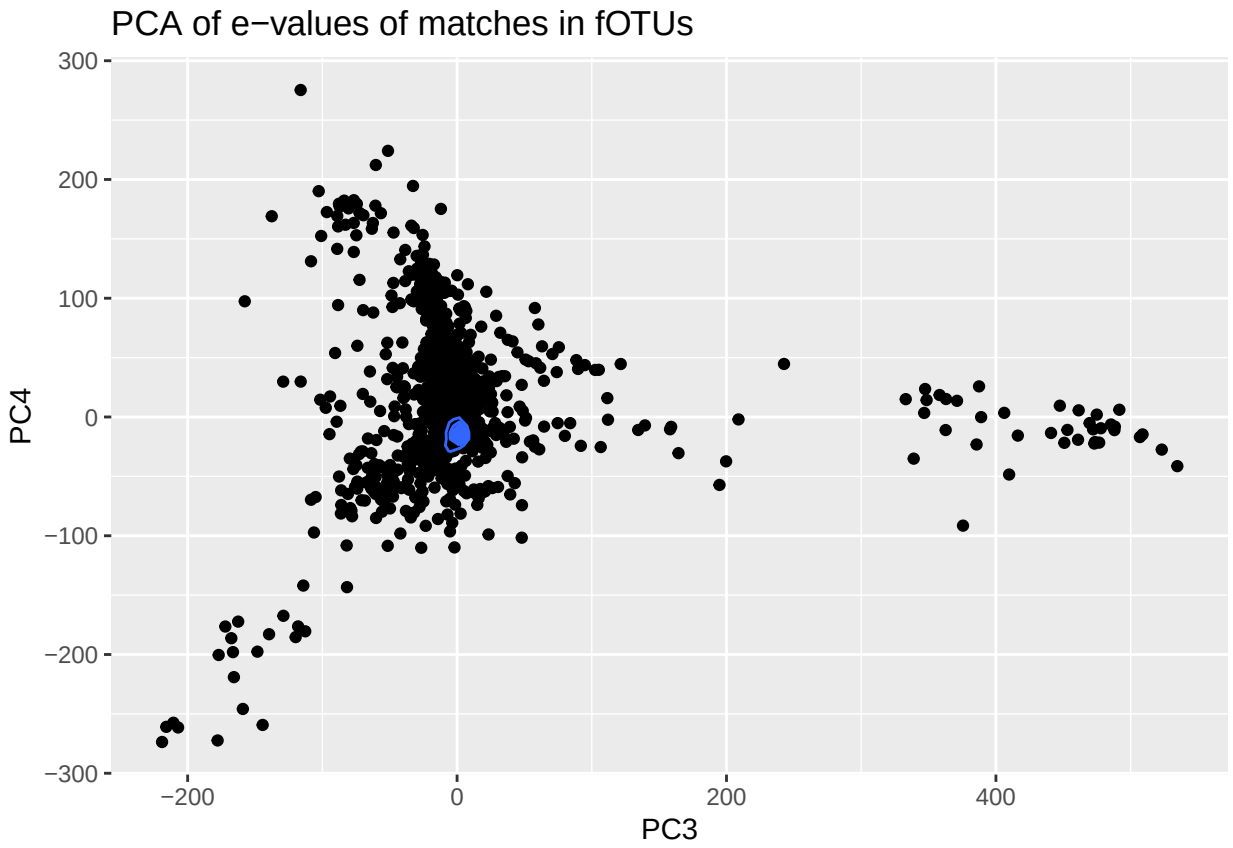
```
PC12s <- tibble(fOTUs = names(fOTUsHMM_pca$x[,1]),
                PC1 = unname(fOTUsHMM_pca$x[,1]),
                PC2 = unname(fOTUsHMM_pca$x[,2]))

title <- "PCA of e-values of matches in fOTUs"
qplot(PC1,
      PC2,
      data = PC12s,
      geom = c("point", "density2d")) + ggtitle(title)
```




```
ggsave(filename = "../visualisations/HMMsearch/pca_hmmsearch_max_PC1-2.png",
       device = png,
       width = 1000,
       height = 750,
       units = "in",
       limitsize = FALSE,
       dpi = 320)

PC23s <- tibble(fOTUs = names(fOTUsHMM_pca$x[,1]),
                PC2 = unname(fOTUsHMM_pca$x[,2]),
                PC3 = unname(fOTUsHMM_pca$x[,3]))

title <- "PCA of e-values of matches in fOTUs"
qplot(PC2,
      PC3,
      data = PC23s,
      geom = c("point", "density2d")) + ggtitle(title)
```




```
ggsave(filename = "../visualisations/HMMsearch/pca_hmmsearch_max_PC2-3.png",
       device = png,
       width = 1000,
       height = 750,
       units = "in",
       limitsize = FALSE,
       dpi = 320)

PC34s <- tibble(fOTUs = names(fOTUsHMM_pca$x[,1]),
                PC3 = unname(fOTUsHMM_pca$x[,3]),
                PC4 = unname(fOTUsHMM_pca$x[,4]))

title <- "PCA of e-values of matches in fOTUs"
qplot(PC3,
      PC4,
      data = PC34s,
      geom = c("point", "density2d")) + ggtitle(title)
```




```
ggsave(filename = "../visualisations/HMMsearch/pca_hmmsearch_max_PC3-4.png",
       device = png,
       width = 1000,
       height = 750,
       units = "in",
       limitsize = FALSE,
       dpi = 320)
```

5.2 Heatmaps

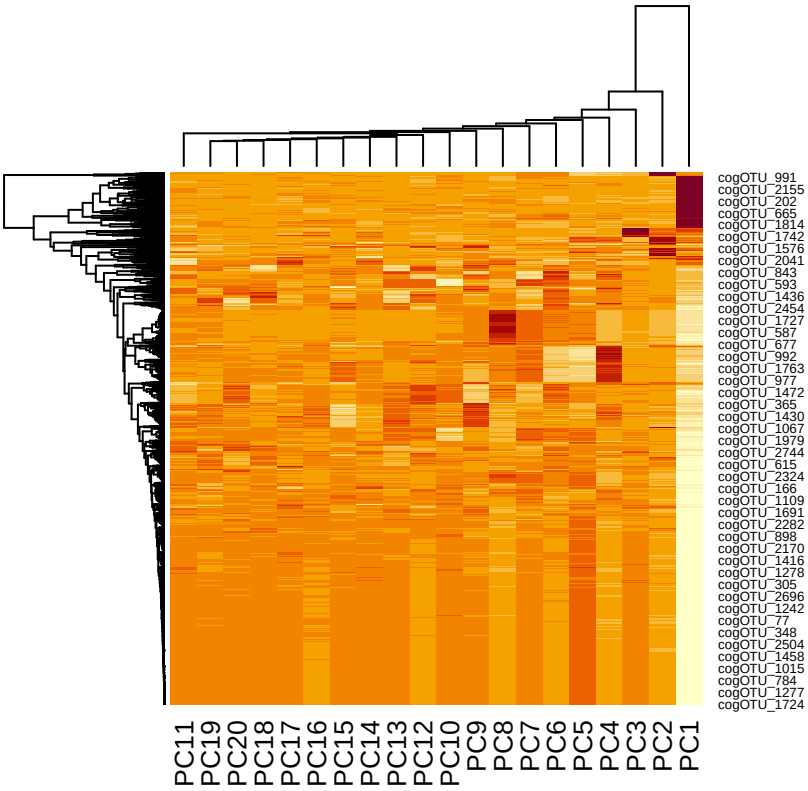
```
dim(fOTUsHMM_pca)
```

```
## NULL
```

```
prediction <- predict(fOTUsHMM_pca)[,1:20]
file_name <- "../visualisations/HMMsearch/heatmap_max_20_first_PCs.png"
png(filename = file_name)
heatmap(as.matrix(prediction))
dev.off()
```

```
## pdf
##   2
```

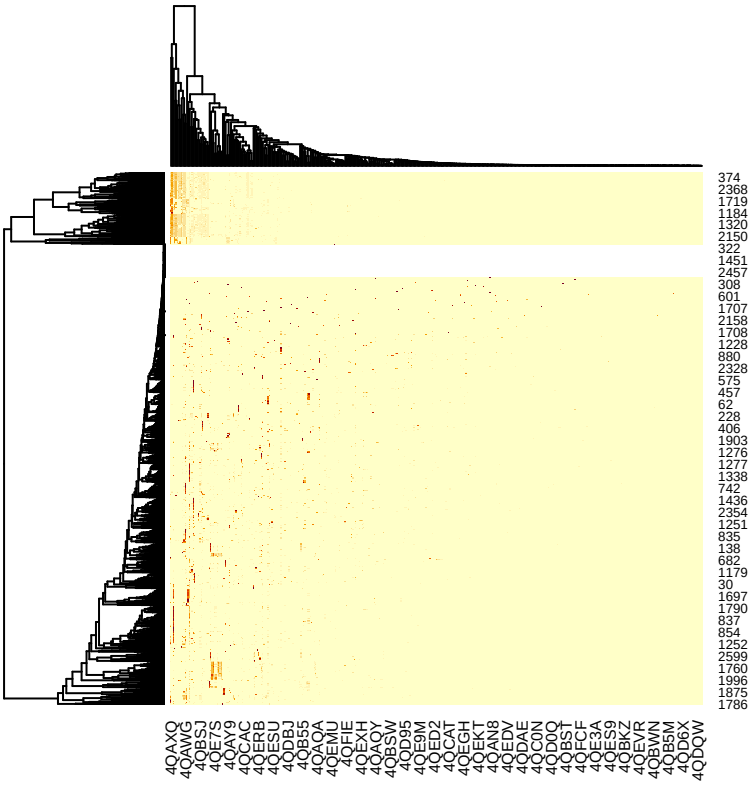
```
heatmap(as.matrix(prediction))
```




```
file_name <- "../visualisations/HMMsearch/heatmap_max_400_first_fOTUs.png"
png(filename = file_name)
heatmap(as.matrix(fOTUsHMM[,-1])[,1:400])
dev.off()
```

```
## pdf
##   2
```

```
heatmap(as.matrix(fOTUsHMM[,-1])[,1:400])
```



6 NMDS

Let's perform non-metric multidimensional scaling:

```
width <- dim(fOTUsHMM)[2]
ord <- metaMDS(fOTUsHMM[,2:width])
```

7 Save/Load R data

Save the data once again to speed-up things.

```
#saveRDS(ord, file = "../analyses/HMMsearch/R_data/max_ord.rds")
ord <- readRDS(file = "../analyses/HMMsearch/R_data/max_ord.rds")
```

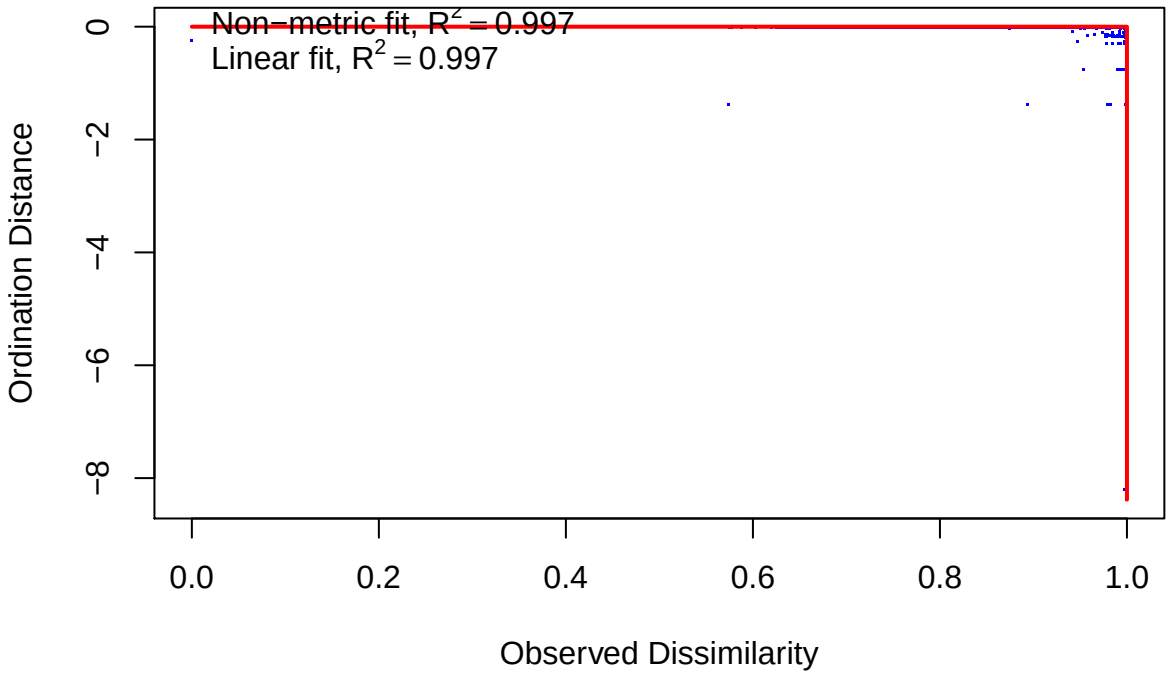
How well does the 2-D representation of multidimensional space represent the actual multidimensional space is done by using statistic called stress. Stress < 0.1 indicates that the 2-D representation is a good reprerepresentation of the multi-D space. Let's check that out now:

```
ord$stress
```

```
## [1] 0.05091335
```

This is quite low which is great! Let's now plot out the relationship between the ordination dissimilarities and the distances in the ordination space:

```
stressplot(ord)
```



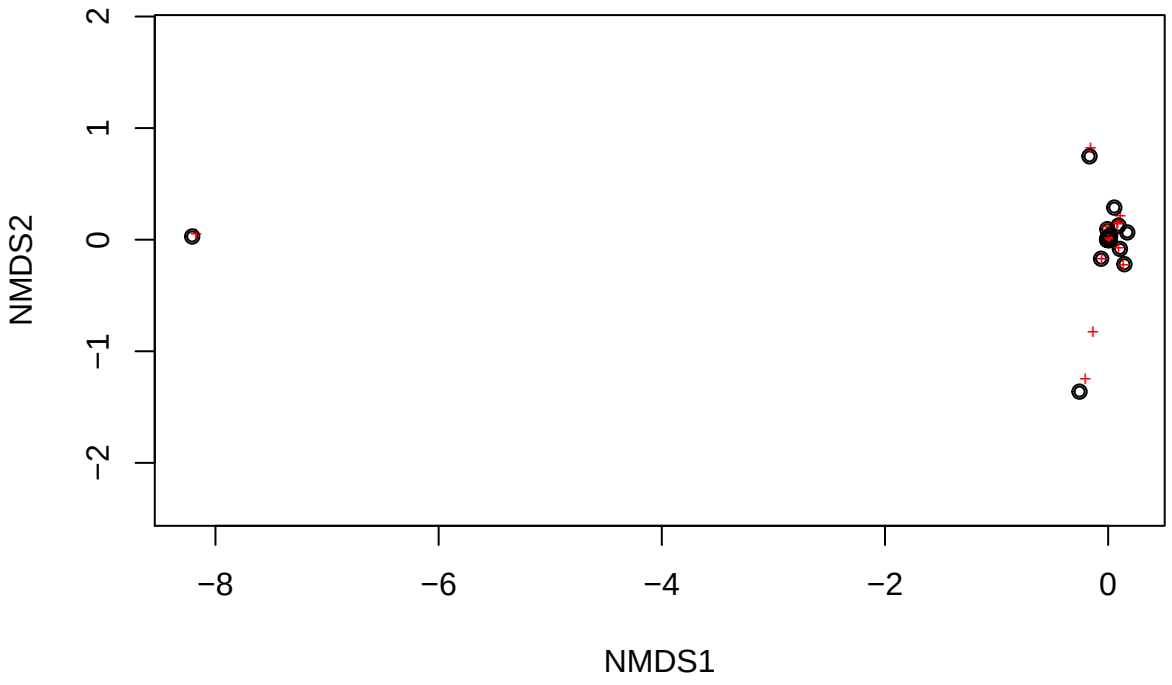
Observed dissimilarity of 1 means the fOTU doesn't have any HMMs incommon with the others.

Let's plot out the ordination space in 2-d:

```
file_name <- "../visualisations/HMMsearch/nmds_ordiplot_all_data.png"
png(filename = file_name)
ordiplot(ord)
points(ord)
dev.off()
```

```
## pdf
##   2
```

```
ordiplot(ord)
points(ord)
```



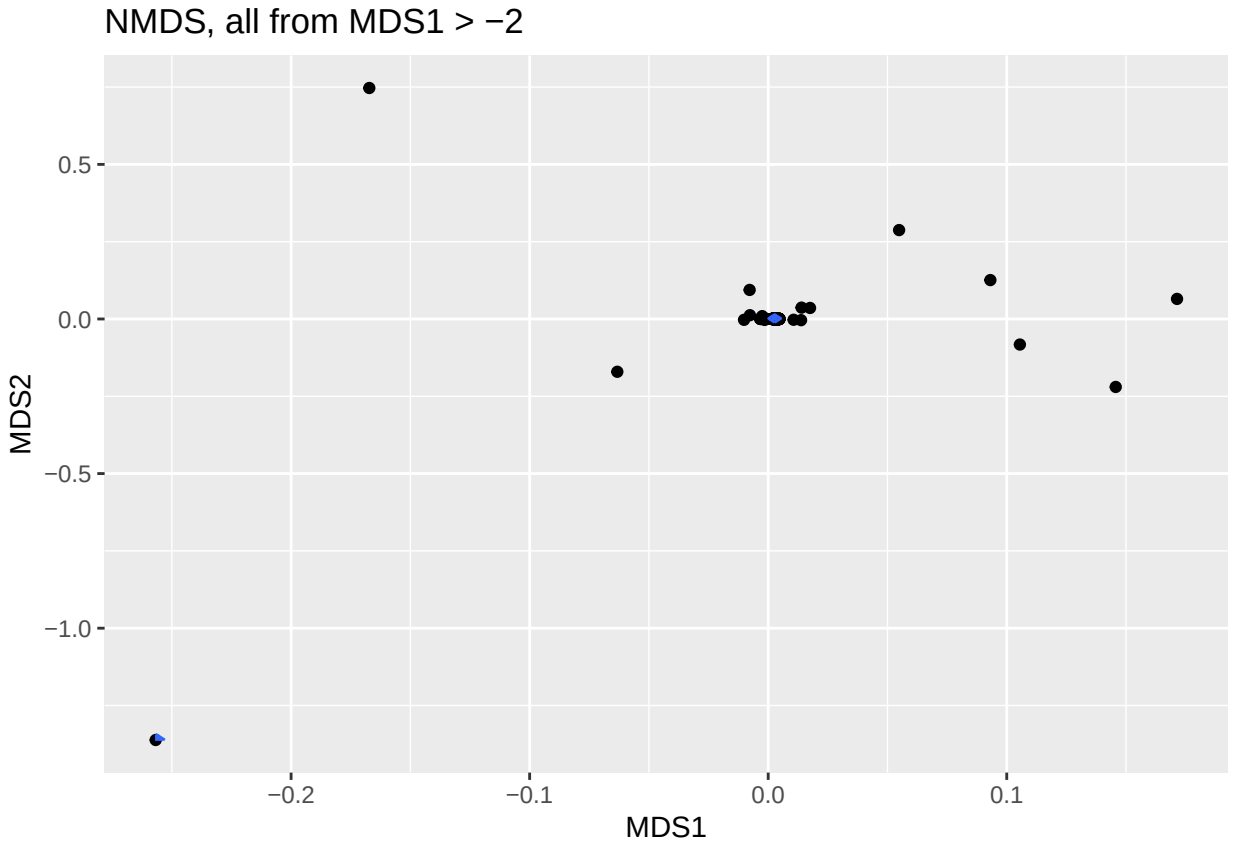
8 Remove outlier(s)

These are just removed so that it's easier to see the better the whole data.

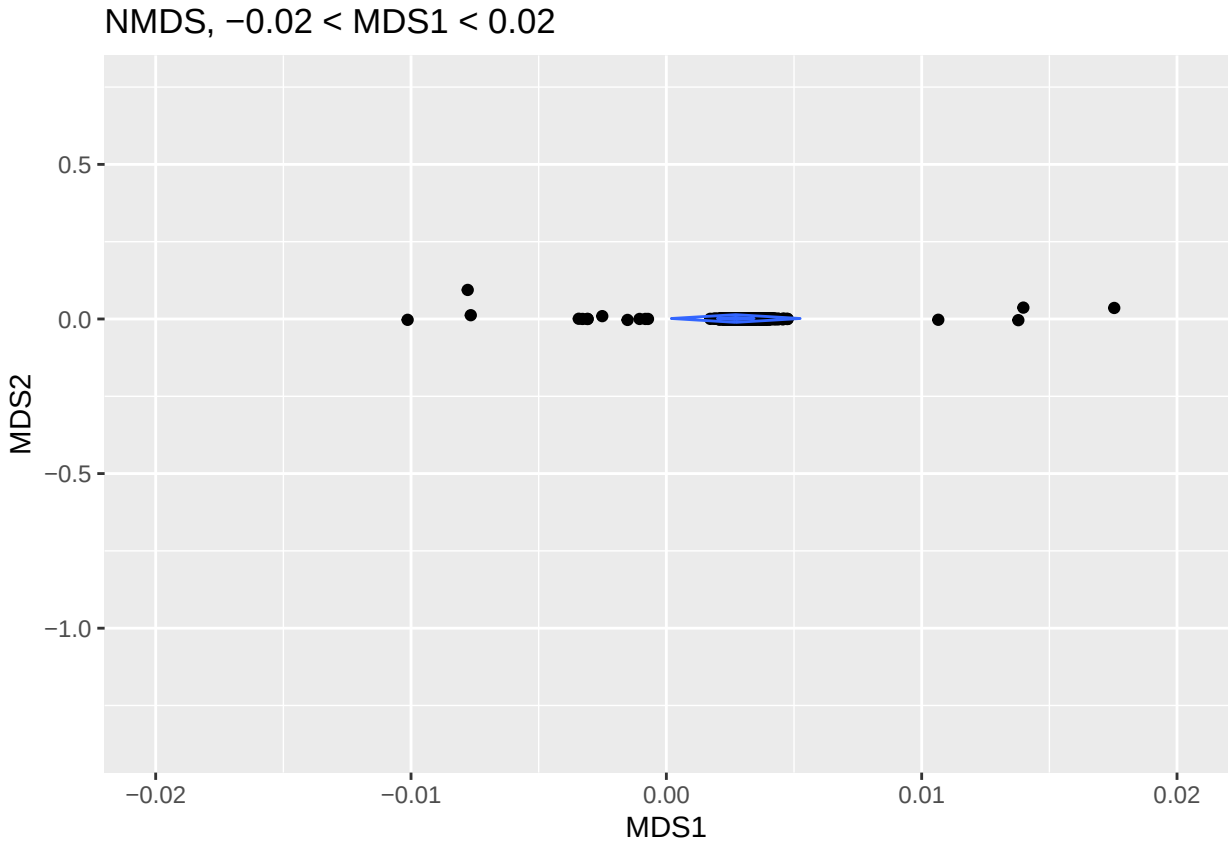
```
NMDS_points <- as_tibble(ord$points) %>%
  filter(MDS1 > -2)
```

9 Visualise the filtered data

```
title <- "NMDS, all from MDS1 > -2"
qplot(MDS1,
      MDS2,
      data = NMDS_points,
      geom = c("point", "density2d")) + ggtitle(title)
```




```
path <- "../visualisations/HMMsearch/nmds_hmmsearch_max_MDS1_greater_than_-2.png"

ggsave(filename = path,
       device = png,
       width = 1000,
       height = 750,
       units = "in",
       limitsize = FALSE,
       dpi = 320)

# Let's zoom in to the main cluster
zoom <- coord_cartesian(xlim = c(-0.02,0.02))

title <- "NMDS, -0.02 < MDS1 < 0.02"
qplot(MDS1,
      MDS2,
      data = NMDS_points,
      geom = c("point", "density2d")) + ggtitle(title) + zoom
```




```
path <- "../visualisations/HMMsearch/nmds_hmmsearch_max_MDS1_greater_than_-0_02_and_less_than_0_02.png"

ggsave(filename = path,
       device = png,
       width = 1000,
       height = 750,
       units = "in",
       limitsize = FALSE,
       dpi = 320)
```

10 Create lists of fOTUs of interest

10.1 PCA lists

Based on the previewing of PCA plots in 5.1 we can now manually curate a lists of fOTUs approximately based on clustering in them.

Let's first create file names:

```
# Path to files
path <- "../analyses/HMMsearch/lists_of_PCA_clusters/"

# Names of files
PC12a_filename <- paste(path,
                        "fOTU_max_PC1-and-2_x--100-to-0-and-y--50-to-+50.txt",
                        sep = "",
                        collapse = NULL)
PC12b_filename <- paste(path,
```

```
                        "fOTU_max_PC1-and-2_x-+200-to-+500-and-y--100-to-0.txt",
                        sep = "",
                        collapse = NULL)


PC23a_filename <- paste(path,
                        "fOTU_max_PC2-and-3_x--100-to-+75-and-y--150-to-+80.txt",
                        sep = "",
                        collapse = NULL)
PC23b_filename <- paste(path,
                        "fOTU_max_PC2-and-3_x-+100-to-+250-and-y--80-to-+125.txt",
                        sep = "",
                        collapse = NULL)
PC23c_filename <- paste(path,
                        "fOTU_max_PC2-and-3_x-+250-to-+425-and-y--200-to-0.txt",
                        sep = "",
                        collapse = NULL)
PC23d_filename <- paste(path,
                        "fOTU_max_PC2-and-3_x-+400-to-+650-and-y--210-to--80.txt",
                        sep = "",
                        collapse = NULL)
PC23e_filename <- paste(path,
                        "fOTU_max_PC2-and-3_x-0-to-+200-and-y-+200-to-+450.txt",
                        sep = "",
                        collapse = NULL)


PC34a_filename <- paste(path,
                        "fOTU_max_PC3-and-4_x--80-to-+80-and-y--50-to-+75.txt",
                        sep = "",
                        collapse = NULL)
PC34b_filename <- paste(path,
                        "fOTU_max_PC3-and-4_x--50-to-0-and-y-+75-to-+175.txt",
                        sep = "",
                        collapse = NULL)
PC34c_filename <- paste(path,
                        "fOTU_max_PC3-and-4_x-+200-to-+450-and-y--25-to-+50.txt",
                        sep = "",
                        collapse = NULL)
```

and then we'll filter tibbles based on their coordinates with a simple function:

```
# This function filters a tibble based on given limit values
filterator <- function(data,
                       x_low = -9999,
                       x_high = 9999,
                       y_low = -9999,
                       y_high = 9999){
  data %>%
    filter(data[,2] > x_low & data[,2] < x_high & data[,3] > y_low & data[,3] < y_high) %>%
    select(1)
}
```

and the actual filtering:

```
PC12_cluster1 <- filterator(PC12s,-100,0,-50,50)
PC12_cluster2 <- filterator(PC12s,200,500,-100,0)

PC23_cluster1 <- filterator(PC23s,-100,75,-150,80)
PC23_cluster2 <- filterator(PC23s,100,250,-80,125)
PC23_cluster3 <- filterator(PC23s,250,425,-200,0)
PC23_cluster4 <- filterator(PC23s,400,650,-210,80)
PC23_cluster5 <- filterator(PC23s,0,200,200,450)

PC34_cluster1 <- filterator(PC34s,-80,80,-50,75)
PC34_cluster2 <- filterator(PC34s,-50,0,75,175)
PC34_cluster3 <- filterator(PC34s,200,450,-25,50)
```

Lastly, we'll just write the files:

```
# PC1-PC2 manually curated clusters
write_delim(PC12_cluster1,
            PC12a_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(PC12_cluster2,
            PC12b_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")

# PC2-PC3 manually curated clusters
write_delim(PC23_cluster1,
            PC23a_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(PC23_cluster2,
            PC23b_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(PC23_cluster3,
            PC23c_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(PC23_cluster4,
```

```
            PC23d_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(PC23_cluster5,
            PC23e_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")

# PC3-PC4 manually curated clusters
write_delim(PC34_cluster1,
            PC34a_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(PC34_cluster2,
            PC34b_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(PC34_cluster3,
            PC34c_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
```

10.2 NMDS lists

Based on the previewing of NMDS plots in 9 we can now manually curate a lists of fOTUs approximately based on various putative clusters in them.

Let's first create file names for the putative clusters:

```
# Path to files
path <- "../analyses/HMMsearch/lists_of_clusters/"

# Names of files
NMDSa_filename <- paste(path,
                        "fOTU_max_NMDS_x--10-to--7.txt",
                        sep = "",
                        collapse = NULL)
NMDSb_filename <- paste(path,
                        "fOTU_max_NMDS_x--1-to--0_2.txt",
```

```
                        sep = "",
                        collapse = NULL)
NMDSc_filename <- paste(path,
                        "fOTU_max_NMDS_x--0_2-to--0.05txt",
                        sep = "",
                        collapse = NULL)
NMDSd_filename <- paste(path,
                        "fOTU_max_NMDS_x--0_02-to-0.txt",
                        sep = "",
                        collapse = NULL)
NMDSe_filename <- paste(path,
                        "fOTU_max_NMDS_x-0-to-+0_01.txt",
                        sep = "",
                        collapse = NULL)
NMDSf_filename <- paste(path,
                        "fOTU_max_NMDS_x-+0_01-to-+0_02.txt",
                        sep = "",
                        collapse = NULL)
NMDSg_filename <- paste(path,
                        "fOTU_max_NMDS_x-+0_05-to-+9999.txt",
                        sep = "",
                        collapse = NULL)
```

Now we need to have a data frame with 3 columns, first of which contains the fOTU names. This so because then the already made function `filterator`'s indexing is correct.

```
unfiltered <- as_tibble(ord$points)
# Prepend the fOTU_names before first column
unfiltered <- cbind(select(fOTUsHMM,1),
                    unfiltered)
```

and execute the actual filtering:

```
# The outlier
NMDS_cluster1 <- filterator(unfiltered,
                            x_low = -10,
                            x_high = -7)
# And the rest
NMDS_cluster2 <- filterator(unfiltered,
                            x_low = -1,
                            x_high = -0.2)
NMDS_cluster3 <- filterator(unfiltered,
                            x_low = -0.2,
                            x_high = -0.05)
NMDS_cluster4 <- filterator(unfiltered,
                            x_low = -0.02,
                            x_high = 0)
NMDS_cluster5 <- filterator(unfiltered,
                            x_low = 0,
                            x_high = 0.01)
NMDS_cluster6 <- filterator(unfiltered,
                            x_low = 0.01,
                            x_high = 0.02)
NMDS_cluster7 <- filterator(unfiltered,
                            x_low = 0.05)
```

Lastly, we'll just write the cluster list files:

```
write_delim(NMDS_cluster1,
            NMDSa_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(NMDS_cluster2,
            NMDSb_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(NMDS_cluster3,
            NMDSc_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(NMDS_cluster4,
            NMDSd_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(NMDS_cluster5,
            NMDSe_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(NMDS_cluster6,
            NMDSf_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
write_delim(NMDS_cluster7,
            NMDSg_filename,
            delim = "",
            na = "NA",
            append = FALSE,
            col_names = FALSE,
            quote_escape = "double")
```

11 Session info

```
sessionInfo()
```

```
## R version 3.6.1 (2019-07-05)
## Platform: x86_64-pc-linux-gnu (64-bit)
## Running under: Debian GNU/Linux 9 (stretch)
##
## Matrix products: default
## BLAS/LAPACK: /usr/lib/libopenblasp-r0.2.19.so
##
## locale:
##  [1] LC_CTYPE=C.UTF-8       LC_NUMERIC=C           LC_TIME=C.UTF-8
##  [4] LC_COLLATE=C.UTF-8     LC_MONETARY=C.UTF-8    LC_MESSAGES=C
##  [7] LC_PAPER=C.UTF-8       LC_NAME=C              LC_ADDRESS=C
## [10] LC_TELEPHONE=C         LC_MEASUREMENT=C.UTF-8 LC_IDENTIFICATION=C
##
## attached base packages:
## [1] grid      stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
##  [1] vegan_2.5-6     lattice_0.20-38 permute_0.9-5   ggbiplot_0.55
##  [5] scales_1.0.0    plyr_1.8.4      forcats_0.4.0   stringr_1.4.0
##  [9] dplyr_0.8.3     purrr_0.3.2     readr_1.3.1     tidyr_0.8.3
## [13] tibble_2.1.3    ggplot2_3.2.1   tidyverse_1.2.1
##
## loaded via a namespace (and not attached):
##  [1] tidyselect_0.2.5 xfun_0.9         splines_3.6.1    haven_2.1.1
##  [5] colorspace_1.4-1 generics_0.0.2   vctrs_0.2.0      htmltools_0.3.6
##  [9] mgcv_1.8-28      yaml_2.2.0       rlang_0.4.0      pillar_1.4.2
## [13] glue_1.3.1       withr_2.1.2      modelr_0.1.5     readxl_1.3.1
## [17] munsell_0.5.0    gtable_0.3.0     cellranger_1.1.0 rvest_0.3.4
## [21] evaluate_0.14    labeling_0.3     knitr_1.24       parallel_3.6.1
## [25] broom_0.5.2      Rcpp_1.0.2       backports_1.1.4  jsonlite_1.6
## [29] hms_0.5.1        digest_0.6.20    stringi_1.4.3    bookdown_0.13
## [33] cli_1.1.0        tools_3.6.1      magrittr_1.5     lazyeval_0.2.2
## [37] cluster_2.1.0    crayon_1.3.4     pkgconfig_2.0.2  zeallot_0.1.0
## [41] Matrix_1.2-17    MASS_7.3-51.4    xml2_1.2.2       lubridate_1.7.4
## [45] assertthat_0.2.1 rmarkdown_1.15   httr_1.4.1       rstudioapi_0.10
## [49] R6_2.4.0         nlme_3.1-140     compiler_3.6.1
```

12 References