Fur seal chemical fingerprints encode colony membership, mother-offspring similarity, relatedness and genetic quality

Stoffel, M.A., Caspers, B.A., Forcada, J., Giannakara, A., Baier, M.C., Eberhart-Phillips, L.J. , Müller, C. & Hoffman, J.I.

!!!This file is under development!!! and will soon contain all major analyses.

This documentation explains the main analyses from our paper and provides the accompanying code sections as well as figures. The complete code can be found in this github repo. We wrote two small R packages for our analysis. You can directly download them from github with the devtools package:

This document provides the code for all major analysis from our paper. For any questions just contact me under martin.adam.stoffel@gmail.com

We wrote two packages to simplify the analysis, which are both hosted on my GitHub repository. The inbreedR package provides functions for measuring inbreeding from molecular data (SNPs and microsatellites) and will soon be published. The minmodelr package is a small package for finding minimal adequate models (Crawley) and formatting outputs, but it was mainly written for personal usage. To download packages from GitHub repositories, one needs to install the `devtools` package.

Some functions have been outsourced. The raw data is in the files directory.

```
# install.packages("devtools")
library(devtools)
# install_github("mastoffel/inbreedR")
# install_github("mastoffel/minmodelr")
library(inbreedR)
library(minmodelr)
```

See `?inbreedR` and `?minmodelr` for further information on the functions. We will use them thoughout this documentation.

Loading data, standardisation and transformation

Loading the raw scent data (aligned by algorithm) and a factor data frame containing identities for colony membership (colony), mother-offspring pairs (family) and mothers and pups, respectively (age)

```
scent_raw <- as.data.frame(t(read.csv(".\\files\\scent_raw.csv", row.names = 1)))
factors <- read.csv(".\\files\\factors.csv",row.names=1)
head(factors)
```

```
##     colony family age
## M10      2     10   1
## M12      2     12   1
## M14      2     14   1
## M15      1     15   1
## M16      2     16   1
## M17      2     17   1
```

Standardising observations by total, such that within every observation compounds add up to 100 % (Thus averaging out absolute concentration differences between samples)

```
scent_stand <- as.data.frame(t(apply(scent_raw, 1, function(x) (x/sum(x)) * 100)))
```

Log(x+1) transformation of the standardised scent data.

```
scent <- log(scent_stand + 1)
```

The scent matrix contains 82 observations and 213 compounds (retention times are column names) in total

```
dim(scent)
```

```
## [1]  82 213
```

```
head(scent[1:6])
```

```
##     8.061111111     8.23 8.307142857 8.394  8.47375 8.516153846
## M10    0.000000 0.000000   0.0000000     0 0.000000   0.6562090
## M12    0.000000 0.000000   0.4864961     0 0.000000   0.0000000
## M14    3.222626 1.665421   0.0000000     0 0.000000   0.0000000
## M15    0.000000 0.000000   0.0000000     0 0.000000   0.0000000
## M16    0.000000 0.000000   0.6849915     0 1.008018   0.5654895
## M17    2.330450 0.000000   0.0000000     0 0.000000   0.0000000
```

Colony differences

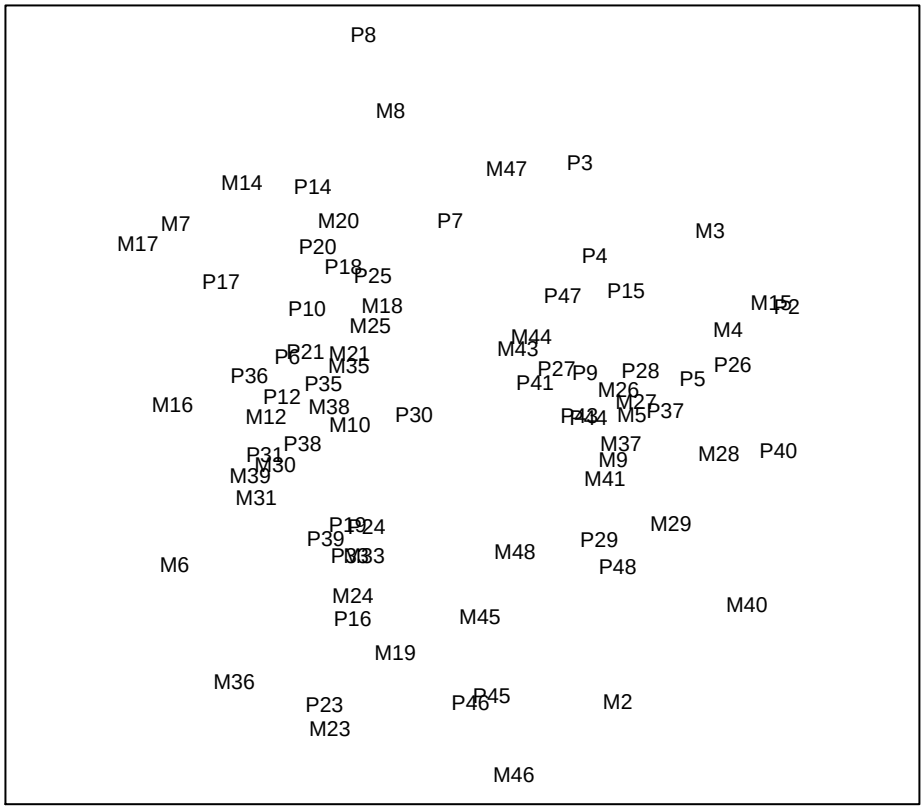
```
library(vegan)
library(MASS)
```

Non-metric multidimensional scaling (nMDS) visualizes a distance matrix (Bray-Curtis similarity). The nMDS algorithm aims to place each individual in a 2-dimensional space such that the between-individual distances are preserved as well as possible. Axis coordinates are arbitrary and not shown. The plot is better visualized with colours (see paper) and is shown here for the purpose of demonstration. Mother-offspring pairs can be identified by labels (e.g. M14, P14)

```
scent_mds <- MASS::isoMDS(vegdist(scent))
```

```
## initial  value 28.002906
## iter   5 value 21.594484
## final  value 21.345037
## converged
```

```
vegan::ordiplot(scent_mds, type = "t", ylab = "", xlab = "",axes=FALSE, frame.plot=TRUE)
```



Analysis of Similarities (ANOSIM) tests for group differences based on a Bray-curtis similarity matrix.

Dissimilarity between the two colonies.

```
vegan::anosim(dat = scent, grouping = factors$colony,
       distance = "bray", permutations = 1000)
```

```
##
## Call:
## vegan::anosim(dat = scent, grouping = factors$colony, permutations = 1000,      distance = "bray")
## Dissimilarity: bray
##
## ANOSIM statistic R: 0.5691
##       Significance: 0.000999
##
## Permutation: free
## Number of permutations: 1000
```

Dissimilarity between mothers from the two colonies.

```
vegan::anosim(dat = scent[factors$age == 1, ], grouping = factors$colony,
       distance = "bray", permutations = 1000)
```

```
##
## Call:
## vegan::anosim(dat = scent[factors$age == 1, ], grouping = factors$colony,      permutations = 1000, d
## Dissimilarity: bray
##
## ANOSIM statistic R: 0.5748
##       Significance: 0.000999
##
## Permutation: free
## Number of permutations: 1000
```

Dissimilarity between pups from the two colonies.

```
vegan::anosim(dat = scent[factors$age == 2, ], grouping = factors$colony,
       distance = "bray", permutations = 1000)
```

```
##
## Call:
## vegan::anosim(dat = scent[factors$age == 2, ], grouping = factors$colony,      permutations = 1000, d
## Dissimilarity: bray
##
## ANOSIM statistic R: 0.556
##       Significance: 0.000999
##
## Permutation: free
## Number of permutations: 1000
```

Genetic differentiation of the two colonies was assesed through the software "Structure" (Pritchard, Stephens & Donnelly (2000))

Mother offspring similarity

Overall

```
vegan::anosim(dat = scent, grouping = factors$family,
       distance = "bray", permutations = 1000)
```

```
##
## Call:
## vegan::anosim(dat = scent, grouping = factors$family, permutations = 1000,      distance = "bray")
## Dissimilarity: bray
##
## ANOSIM statistic R: 0.6723
##       Significance: 0.000999
```

```
##
## Permutation: free
## Number of permutations: 1000
```

Within colony 1 (Special study beach)

```
vegan::anosim(dat = scent[factors$colony == 1, ],
       grouping = factors[factors$colony == 1, ]$family,
       distance = "bray", permutations = 1000)
```

```
##
## Call:
## vegan::anosim(dat = scent[factors$colony == 1, ], grouping = factors[factors$colony ==      1, ]$fami
## Dissimilarity: bray
##
## ANOSIM statistic R: 0.5339
##       Significance: 0.000999
##
## Permutation: free
## Number of permutations: 1000
```

Within colony 2 (Freshwater beach)

```
vegan::anosim(dat = scent[factors$colony == 2, ],
       grouping = factors[factors$colony == 2, ]$family,
       distance = "bray", permutations = 1000)
```

```
##
## Call:
## vegan::anosim(dat = scent[factors$colony == 2, ], grouping = factors[factors$colony ==      2, ]$fami
## Dissimilarity: bray
##
## ANOSIM statistic R: 0.4532
##       Significance: 0.000999
##
## Permutation: free
## Number of permutations: 1000
```

```
coord  <- read.csv(".\\files\\coordinates_beach1.csv", row.names=1)
head(coord)
```

Olfactory similarity vs. geographic distance on special study beach (location data in meters is just available for this population as the special study beach on Bird Island provides an aerial walkway)

```
##      X  Y
## M15 10  8
## M19 10 12
## M2  25 15
## M26 23 13
## M27 26 18
## M28 26 18
```

Converting coordinates to pairwise euclidian distance matrix

```
dist_mat <- as.matrix(dist(coord, method = "euclidian"))
```

Constructing bray curtis similarity matrix of all individuals from beach 1

```
scent_bc <- as.matrix(vegdist(as.matrix(scent[factors$colony == 1, ])),
                      method = "bray")
```

Geographic distance vs. olfactory similarity in mothers

```
geo_mum <- dist_mat[1:20, 1:20]
scent_mum <- scent_bc[1:20, 1:20]
vegan::mantel(geo_mum, scent_mum, method = "spearman")
```

```
##
## Mantel statistic based on Spearman's rank correlation rho
##
## Call:
## vegan::mantel(xdis = geo_mum, ydis = scent_mum, method = "spearman")
##
## Mantel statistic r: 0.008091
##       Significance: 0.446
##
## Upper quantiles of permutations (null model):
##   90%   95% 97.5%   99%
## 0.197 0.258 0.316 0.375
## Permutation: free
## Number of permutations: 999
```

Geographic distance vs. olfactory similarity in pups

```
geo_pup <- dist_mat[21:40, 21:40]
scent_pup <- scent_bc[21:40, 21:40]
vegan::mantel(geo_pup, scent_pup, method = "spearman")
```

```
##
## Mantel statistic based on Spearman's rank correlation rho
##
## Call:
## vegan::mantel(xdis = geo_pup, ydis = scent_pup, method = "spearman")
##
## Mantel statistic r: 0.06039
```

```
##       Significance: 0.298
##
## Upper quantiles of permutations (null model):
##   90%   95% 97.5%   99%
## 0.170 0.231 0.276 0.343
## Permutation: free
## Number of permutations: 999
```

Associations between genotype and overall olfactory fingerprints.

Relatedness and overall olfactory similarity

Load pairwise relatedness (based on Queller&Goodnight, 1989) based on 41 microsatellite markers

```
relatedness <- as.matrix(read.csv(".\\files\\relatedness.csv",row.names=1))
```

Pairwise bray curtis similarity in olfactory fingerprints of all individuals

```
scent_bc <- 1-(as.matrix(vegan::vegdist(as.matrix(scent)), method = "bray"))
```

Mantel test between relatedness and bray curtis similarity of all individuals.

```
vegan::mantel(relatedness, scent_bc, method = "spearman", permutation = 1000)
```

```
##
## Mantel statistic based on Spearman's rank correlation rho
##
## Call:
## vegan::mantel(xdis = relatedness, ydis = scent_bc, method = "spearman",      permutations = 1000)
##
## Mantel statistic r: 0.07231
##       Significance: 0.006993
##
## Upper quantiles of permutations (null model):
##    90%    95%  97.5%    99%
## 0.0359 0.0456 0.0568 0.0684
## Permutation: free
## Number of permutations: 1000
```

We are likely to have a problem of pseudoreplication here. For that reason, we are analysing mothers and pups seperately.
Fur seal mothers: mantel test between genetic relatedness and bray curtis similarity of olfactory fingerprints

```
vegan::mantel(relatedness[factors$age == 1, factors$age == 1],
              scent_bc[factors$age == 1, factors$age == 1],
              method = "spearman", permutation = 1000)
```

```
##
## Mantel statistic based on Spearman's rank correlation rho
##
## Call:
## vegan::mantel(xdis = relatedness[factors$age == 1, factors$age ==      1], ydis = scent_bc[factors$ag
##
## Mantel statistic r: 0.05938
##       Significance: 0.1019
##
## Upper quantiles of permutations (null model):
##    90%    95%  97.5%    99%
## 0.0595 0.0820 0.0925 0.1174
## Permutation: free
## Number of permutations: 1000
```

Fur seal pups: mantel test between genetic relatedness and bray curtis similarity of olfactory fingerprints

```
vegan::mantel(relatedness[factors$age == 2, factors$age == 2],
              scent_bc[factors$age == 2, factors$age == 2],
              method = "spearman", permutation = 1000)
```

```
##
## Mantel statistic based on Spearman's rank correlation rho
##
## Call:
## vegan::mantel(xdis = relatedness[factors$age == 2, factors$age ==      2], ydis = scent_bc[factors$ag
##
## Mantel statistic r: 0.02985
##       Significance: 0.25475
##
## Upper quantiles of permutations (null model):
##    90%    95%  97.5%    99%
## 0.0576 0.0750 0.0880 0.1026
## Permutation: free
## Number of permutations: 1000
```

Correlation between Heterozygosity and number of compounds in odour profiles

1. Loading raw genotypes and calculating standardised multilocus heterozygosity (sMLH) based on 41 markers. The function `sMLH` is part of the `inbreedR` package, currently available on GitHub. Install with:

```
# install.packages("devtools")
library(devtools)
# install_github("mastoffel/inbreedR")
library(inbreedR)
# ?inbreedR
```

```
genotypes <- read.table(".\\files\\genotypes.txt", row.names=1)
genotypes_formatted <- inbreedR::convert_raw(genotypes, miss_val = NA)
heterozygosity <- inbreedR::sMLH(genotypes_formatted)
```

2.Number of compounds per individual

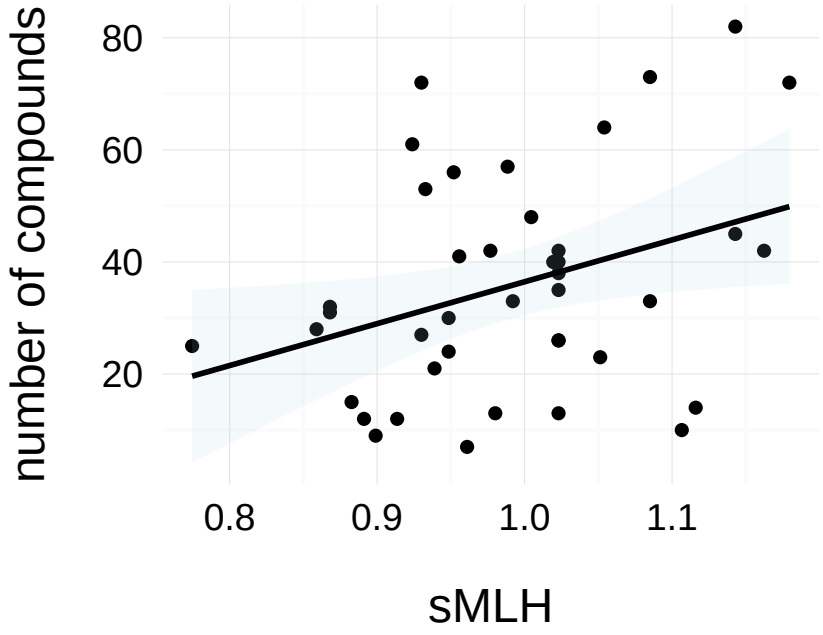
```
num_comp <- as.vector(apply(scent, 1, function(x) length(x[x>0])))
```

3.Correlation between heterozygosity and number of compounds in mothers

```
het_mum <- heterozygosity[factors$age == 1]
num_comp_mum <- num_comp[factors$age==1]
summary(lm(het_mum ~ num_comp_mum))
```

```
##
## Call:
## lm(formula = het_mum ~ num_comp_mum)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.199148 -0.049220  0.005588  0.047729  0.161623
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.9339147  0.0282437  33.066   <2e-16 ***
## num_comp_mum 0.0015914  0.0006936   2.294   0.0272 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08633 on 39 degrees of freedom
## Multiple R-squared:  0.1189, Adjusted R-squared:  0.09633
## F-statistic: 5.264 on 1 and 39 DF,  p-value: 0.02724
```

```
## Warning: package 'ggplot2' was built under R version 3.2.0
```



4. Correlation between heterozygosity and number of compounds in pups

```
het_pup <- heterozygosity[factors$age == 2]
num_comp_pup <- num_comp[factors$age==2]
summary(lm(het_pup ~ num_comp_pup))
```

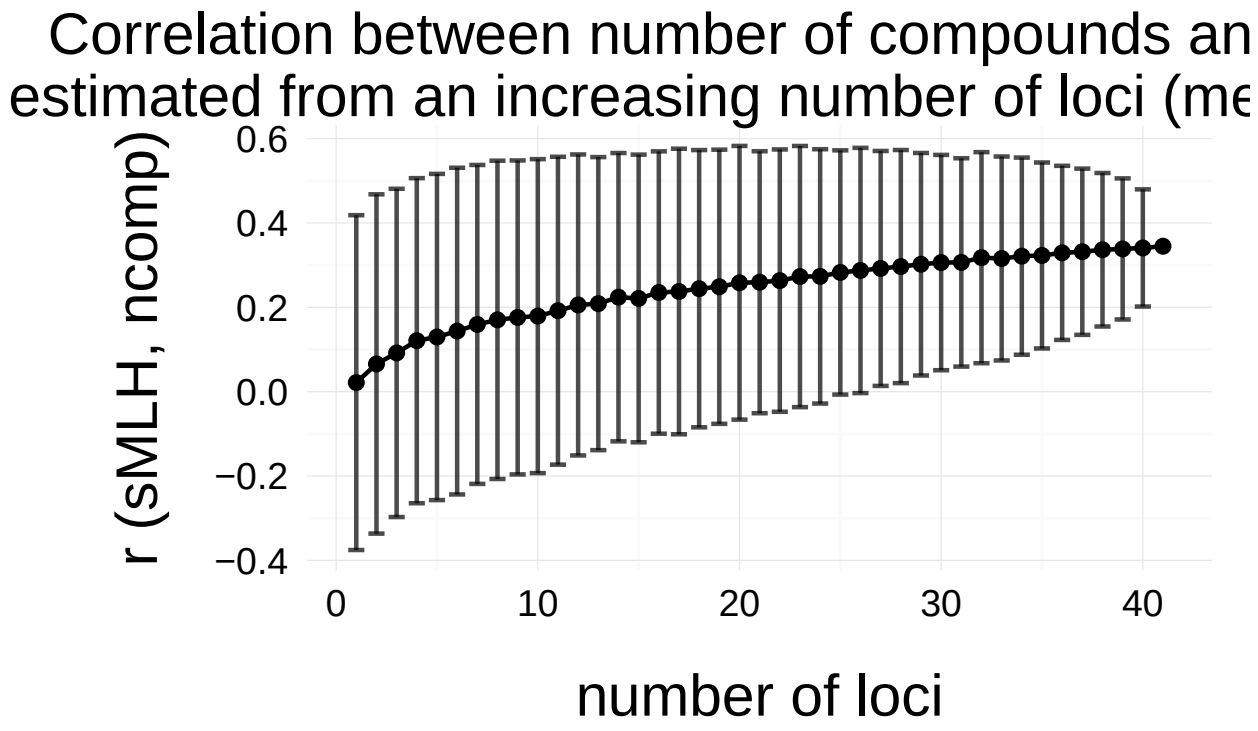
```
##
## Call:
## lm(formula = het_pup ~ num_comp_pup)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.199070 -0.067303 -0.001971  0.050500  0.152902
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.9940863  0.0235812  42.156   <2e-16 ***
## num_comp_pup 0.0003928  0.0005542   0.709    0.483
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08035 on 39 degrees of freedom
## Multiple R-squared:  0.01272,    Adjusted R-squared:  -0.0126
## F-statistic: 0.5025 on 1 and 39 DF,  p-value: 0.4826
```

```
# probably put that function in a seperate package
resample_loci <- function(genotypes, y, num_iter = 1000) {
# genotypes in inbreedR format. See ?inbreedR
# y is a vector to correlate with sMLH
# num_iter is the number of resamplings per added locus

        # calculate number of loci
        num_loci <- ncol(genotypes)
        # initialize results df
        results <- data.frame(matrix(nrow = num_iter, ncol = num_loci))

        for (i in seq_along((1: num_loci))){
                for (k in seq_along(1:num_iter)) {
                loci_ind <- sample(1:num_loci, i, replace = FALSE)
                het <- inbreedR::sMLH(genotypes[, loci_ind])
                results[k, i] <- cor(het[1:41],y) # heterozygosity subsetted for mothers
                }
        }
results
}

# resampling with 1000 iterations
genotypes_formatted <- inbreedR::convert_raw(genotypes, miss_val = NA)
resample_mums <- resample_loci(genotypes_formatted, num_comp_mum, num_iter = 1000)
```

Strength of correlation between sMLH and number of compounds increases with an increasing number of genetic markers in mothers. Plotting mean correlation of heterozygosity (estimated by an increasing number of markers) with number of compounds

```
sum_results <- function(resampling_output) {
        mean.cor <- apply(resampling_output,2,mean, na.rm=T)
        sd.cor <- apply(resampling_output,2,sd, na.rm=T)
        se.cor <- sd.cor/(sqrt(sd.cor))
        sum_results <- data.frame(locnum = 1:ncol(resampling_output),
                                  cormean = mean.cor, corsd = sd.cor, corse = se.cor)
}

results_mums <- sum_results(resample_mums)

# plotting
ggplot(results_mums, aes(x = locnum, y = cormean)) +
        geom_errorbar(aes(ymin = cormean-corse, ymax = cormean+corse),
                      width=0.8, alpha=0.7, size = 0.8) +
        geom_point(size = 3, shape = 16) +
        geom_line(size = 0.8) +
        theme_minimal(base_size = 18) +
        theme(axis.title.x = element_text(vjust= -2 ,size = 22),
              axis.title.y = element_text(vjust=3,size = 22),
              axis.ticks.x = element_blank(),
              axis.ticks.y = element_blank(),
              plot.margin = (unit(c(.5, .5, 2, 2), "cm"))) +
        #geom_hline(yintercept=0.305) +
```

```
        ylab("r (sMLH, ncomp)") +
        xlab("number of loci") +
        ggtitle("Correlation between number of compounds and sMLH \nestimated from an increasing number
```




```
inbreedR::g2_microsats(genotypes_formatted, nperm = 1000, nboot = 1000, CI = 0.95)
```

Estimation of identity disequilibrium g2 with the inbreedR package. (can diverge slightly from the RMES program)

```
##
##  20 permutations done
##  40 permutations done
##  60 permutations done
##  80 permutations done
##  100 permutations done
##  120 permutations done
##  140 permutations done
##  160 permutations done
##  180 permutations done
##  200 permutations done
##  220 permutations done
##  240 permutations done
```

```
##  260 permutations done
##  280 permutations done
##  300 permutations done
##  320 permutations done
##  340 permutations done
##  360 permutations done
##  380 permutations done
##  400 permutations done
##  420 permutations done
##  440 permutations done
##  460 permutations done
##  480 permutations done
##  500 permutations done
##  520 permutations done
##  540 permutations done
##  560 permutations done
##  580 permutations done
##  600 permutations done
##  620 permutations done
##  640 permutations done
##  660 permutations done
##  680 permutations done
##  700 permutations done
##  720 permutations done
##  740 permutations done
##  760 permutations done
##  780 permutations done
##  800 permutations done
##  820 permutations done
##  840 permutations done
##  860 permutations done
##  880 permutations done
##  900 permutations done
##  920 permutations done
##  940 permutations done
##  960 permutations done
##  980 permutations done
##  ### permutations finished ###
##  20 bootstraps done
##  40 bootstraps done
##  60 bootstraps done
##  80 bootstraps done
##  100 bootstraps done
##  120 bootstraps done
##  140 bootstraps done
##  160 bootstraps done
##  180 bootstraps done
##  200 bootstraps done
##  220 bootstraps done
##  240 bootstraps done
##  260 bootstraps done
##  280 bootstraps done
##  300 bootstraps done
##  320 bootstraps done
```

```
##  340 bootstraps done
##  360 bootstraps done
##  380 bootstraps done
##  400 bootstraps done
##  420 bootstraps done
##  440 bootstraps done
##  460 bootstraps done
##  480 bootstraps done
##  500 bootstraps done
##  520 bootstraps done
##  540 bootstraps done
##  560 bootstraps done
##  580 bootstraps done
##  600 bootstraps done
##  620 bootstraps done
##  640 bootstraps done
##  660 bootstraps done
##  680 bootstraps done
##  700 bootstraps done
##  720 bootstraps done
##  740 bootstraps done
##  760 bootstraps done
##  780 bootstraps done
##  800 bootstraps done
##  820 bootstraps done
##  840 bootstraps done
##  860 bootstraps done
##  880 bootstraps done
##  900 bootstraps done
##  920 bootstraps done
##  940 bootstraps done
##  960 bootstraps done
##  980 bootstraps done
##  ### bootstrapping finished, hells yeah!! ###
```

```
##
## Data: 82 observations at 41 markers
## Function call = inbreedR::g2_microsats(genotypes = genotypes_formatted, nperm = 1000,     nboot = 10
##
## g2 = 0.00241214, se = 0.001390797
##
## confidence interval
##         2.5%        97.5%
## -0.000151049  0.005274310
##
## p (g2 > 0) = 0.026 (based on 1000 permutations)
```

potentially make figure for increasing markers here

Factor analysis on the chemical compounds data with the package HDMD.

HDMD allows for doing a factoranalysis with high dimensional data, where more variables than observations are present by calculating a general inverse matrix.

```
library(HDMD)
```

```
## Warning: package 'HDMD' was built under R version 3.2.0
```

```
## Loading required package: psych
```

```
## Warning: package 'psych' was built under R version 3.2.0
```

```
##
## Attaching package: 'psych'
##
## The following object is masked from 'package:ggplot2':
##
##     %+%
```

```
library(minmodelr)
source("get_pairdiff.R")
```

Factor analysis and extraction of factor scores for the first 4 factors. Promax rotation of the factors allows them to be non-orthogonal and thus correlated.

```
# factor analysis with 4 factors, promax rotation -----------------------------
scent_fa <- factor.pa.ginv(scent, nfactors = 4,
                           prerotate = T,rotate = "promax",
                           scores = T, m = 3)
```

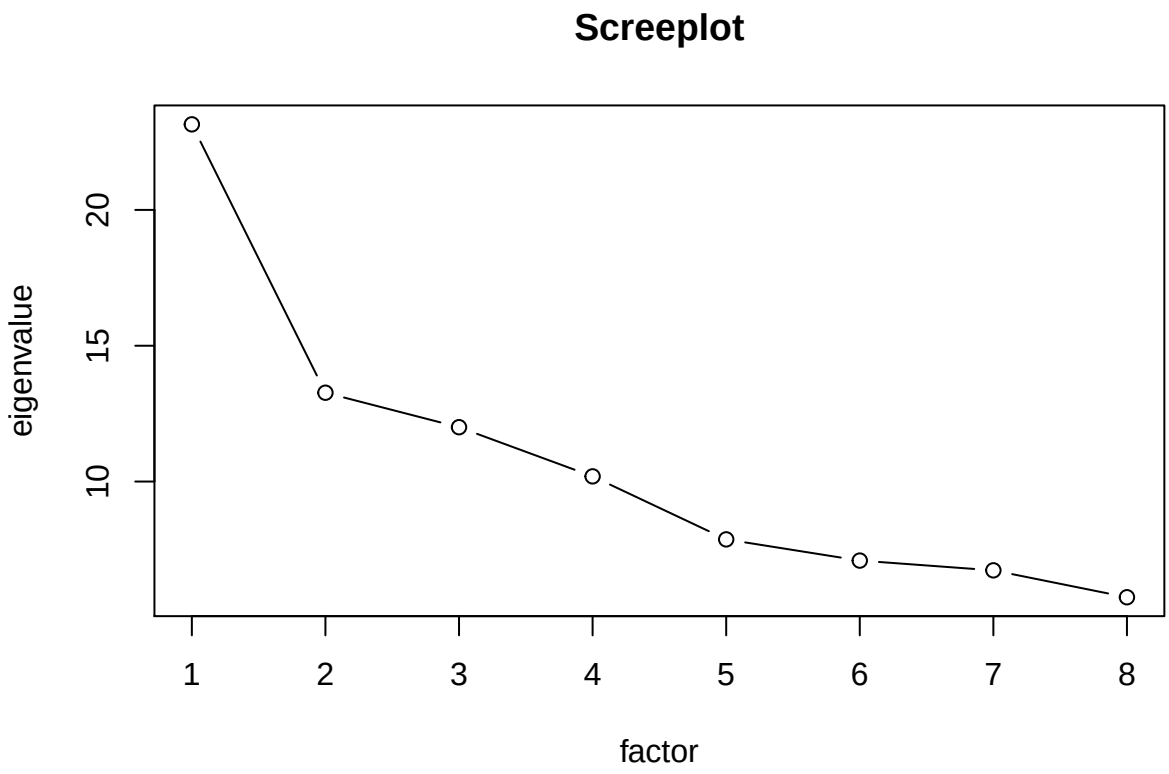
```
## Warning in cor.smooth(R): Matrix was not positive definite, smoothing was
## done
```

```
## Could not solve for inverse correlation.  Using general inverse ginv(r)
```

```
fa_scores <- as.data.frame(scent_fa$scores)
```

The eigenvalue course seen in the screeplot allows for decisions on the number of factors to retain.

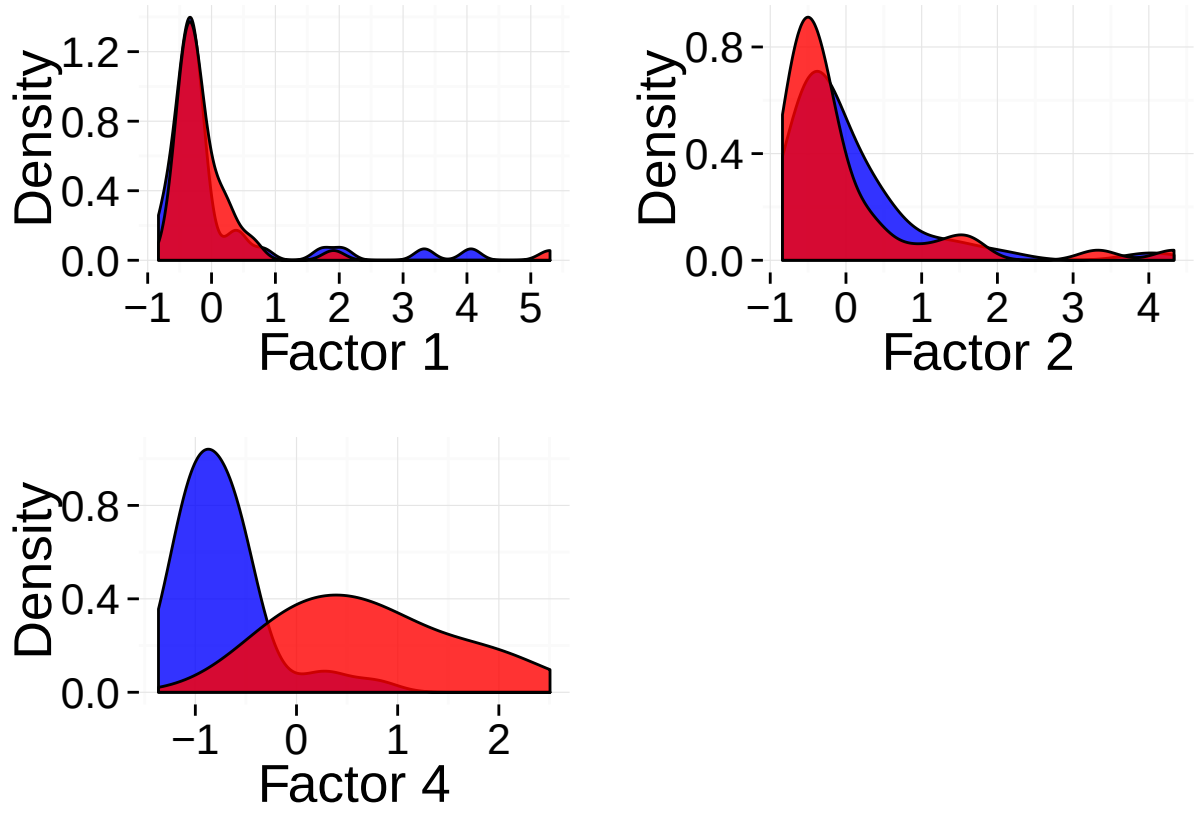
```
# screeplot, 4 factors left to the "scree"
plot(scent_fa$values[1:8], type="b", ylab = "eigenvalue", xlab = "factor",
     main = "Screeplot")
```



Plotting the distribution of factor scores seperately for each colony. Similar distributions suggest the compounds which are represented by a given factor to be similarly distributed across colonies and could thus be of potential genetic origin, while different distributions as in factor 4 suggest this factor to represent environmentally influenced compounds.

```
# distribution of factor scores
df <- cbind(fa_scores, factors["colony"])
df$colony <- as.factor(df$colony)

for (i in c(1,2,4)) {
plot <- ggplot(df, aes_string(x = paste("F", i, sep = ""), fill = "colony")) +
        geom_density(alpha=0.8, size=0.5, aes(fill = colony),adjust=1.5) +
        scale_fill_manual(values = c("blue","red")) +
        guides(fill=guide_legend(title=NULL)) +
        theme_minimal(base_size = 20) +
        theme(legend.position="none") +
        scale_x_continuous(breaks = c(seq(from = -1, to = 6, by = 1))) +
        scale_y_continuous(breaks = c(seq(from = 0, to = 1.4, by = 0.4))) +
        #scale_y_continuous(breaks = c(seq(from = 0, to = 1, by = 0.2))) +
        xlab(paste("Factor", i, sep = " ")) +
        ylab("Density")

assign(paste("f", i, "_plot", sep = ""), plot)
}

# using multiplot function from cookbook-r.com for plotting multiple ggplots
```

```
source("multiplot.R")
multiplot(f1_plot, f2_plot, f4_plot, cols = 2)
```



Linear model for associations between heterozygosity and factor scores as explanatory variables in mothers.

```
# bind heterozygosity and the factor scores in one data.frame and subset mothers
het_df <- cbind(heterozygosity, fa_scores)[factors$age == 1, ]
het_model <- lm(heterozygosity ~., data=het_df)
summary(het_model)
```

```
##
## Call:
## lm(formula = heterozygosity ~ ., data = het_df)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.19439 -0.06271  0.01210  0.04769  0.14674
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.987971   0.013037  75.780   <2e-16 ***
## F1           0.029393   0.012288   2.392   0.0221 *
## F2           0.027521   0.011898   2.313   0.0265 *
## F3           0.004033   0.012964   0.311   0.7575
## F4          -0.009617   0.014180  -0.678   0.5020
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08315 on 36 degrees of freedom
## Multiple R-squared:  0.2455, Adjusted R-squared:  0.1616
## F-statistic: 2.928 on 4 and 36 DF,  p-value: 0.03406
```

Model simplification via deletion testing (Crawley, Statistics). The minmodelr package contains some helper functions for this task. See ?MinMod, ?DelTestVar

```
het_reduced <- MinMod(het_df)
```

```
##
## Call:
## glm(formula = depVar ~ ., family = family, data = bestmodeldf)
##
## Deviance Residuals:
##       Min         1Q     Median         3Q        Max
## -0.191202  -0.060032   0.009789   0.057485   0.155757
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.98814    0.01278  77.328   <2e-16 ***
## F1           0.02783    0.01167   2.385   0.0222 *
## F2           0.02809    0.01164   2.414   0.0207 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.006649744)
##
##     Null deviance: 0.32986  on 40  degrees of freedom
## Residual deviance: 0.25269  on 38  degrees of freedom
## AIC: -84.303
##
## Number of Fisher Scoring iterations: 2
```

```
# extract data frame
het_reduced_df <- het_reduced[[1]]
# extract reduced model
het_reduced_mod <- het_reduced[[2]]
# deletion testing for both variables in the reduced model. See ?DelTestVar
table <- DelTestVar(het_reduced_df)
```

```
##               Estimate Deviance Explained        F P (F-test)
## (Intercept) 0.98814218                NA       NA         NA
## F1          0.02783316           11.46936 5.689346 0.02215967
## F2          0.02808759           11.74642 5.826780 0.02071002
##             P (Chisquared-test)
## (Intercept)                  NA
## F1                   0.01706822
## F2                   0.01578399
```

```r
# deviance explained by the reduced model
dev_expl <- (het_reduced_mod$null.deviance - het_reduced_mod$deviance) / het_reduced_mod$null.deviance

summary(het_reduced_mod)
```

```
##
## Call:
## glm(formula = depVar ~ ., family = family, data = bestmodeldf)
##
## Deviance Residuals:
##       Min         1Q     Median         3Q        Max
## -0.191202  -0.060032   0.009789   0.057485   0.155757
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.98814    0.01278  77.328   <2e-16 ***
## F1           0.02783    0.01167   2.385   0.0222 *
## F2           0.02809    0.01164   2.414   0.0207 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.006649744)
##
##     Null deviance: 0.32986  on 40  degrees of freedom
## Residual deviance: 0.25269  on 38  degrees of freedom
## AIC: -84.303
##
## Number of Fisher Scoring iterations: 2
```

Creating a new variable F1F2 which is the sum of the two factor scores and using this variable as predictor in a linear model.

```r
# sum of factors as variable
het_df$F1F2 <- het_df$F1 + het_df$F2
table <- DelTestVar(as.data.frame(cbind(het_df$heterozygosity, het_df$F1F2)))
```

```
##               Estimate Deviance Explained        F  P (F-test)
## (Intercept) 0.98813169                 NA       NA          NA
## V2          0.02796073           23.39391 11.90979 0.001356642
##             P (Chisquared-test)
## (Intercept)                  NA
## V2                0.0005583969
```

```r
summary(lm(heterozygosity ~ F1F2, data = het_df))
```

```
##
## Call:
## lm(formula = heterozygosity ~ F1F2, data = het_df)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.191204 -0.060060  0.009766  0.057466  0.155764
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.988132   0.012596  78.449  < 2e-16 ***
## F1F2        0.027961   0.008102   3.451  0.00136 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.08049 on 39 degrees of freedom
## Multiple R-squared:  0.2339, Adjusted R-squared:  0.2143
## F-statistic: 11.91 on 1 and 39 DF,  p-value: 0.001357
```

Linear model for associations between genetic relatedness and factor scores as explanatory variables - Mothers Pairwise genetic relatedness is represented as a matrix. To model the relationship between relatedness and factor scores we created a matrix for each factor, whereby each pairwise value represents the difference in factor scores for a pair of seals. `get_pairdiff()` creates these matrices. We based these analysis on mothers and pups seperately.

Creation of 4 pairwise distance matrices for each factor.

```
fa_diff_mums <- get_pairdiff(relatedness[factors$age == 1, factors$age == 1],
                             fa_scores[factors$age == 1, ], df = F)

# assign pairwise difference factor matrices to names
for (i in seq_along(1:4)) {
        assign(paste("f", i, "_diff", sep=""), fa_diff_mums[, i+1])
}
```

The ecodist package can handle multiple distance matrices by doing a partial mantel test.

```
rel_dist <- as.dist(relatedness[factors$age == 1, factors$age == 1])
ecodist::mantel(rel_dist ~ f1_diff + f2_diff + f3_diff + f4_diff, mrank = T, nperm = 1000) # tests for
```

```
##     mantelr       pval1       pval2       pval3   llim.2.5%  ulim.97.5%
## -0.12256667  0.98700000  0.01400000  0.03000000 -0.16041250 -0.06567327
```

```
ecodist::mantel(rel_dist ~ f2_diff + f1_diff + f3_diff + f4_diff, mrank = T)              # tests for
```

```
##      mantelr        pval1        pval2        pval3    llim.2.5%
## -0.047920124  0.791000000  0.210000000  0.397000000 -0.081627429
##   ulim.97.5%
## -0.009656023
```

```
ecodist::mantel(rel_dist ~ f3_diff + f2_diff + f1_diff + f4_diff, mrank = T)              # tests for
```

```
##    mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
## 0.08900545 0.05000000 0.95100000 0.12000000 0.04327876 0.13048902
```

```
ecodist::mantel(rel_dist ~ f4_diff + f3_diff + f2_diff + f1_diff, mrank = T)              # tests for
```

```
##     mantelr       pval1       pval2       pval3   llim.2.5%  ulim.97.5%
## 0.052327208 0.124000000 0.877000000 0.236000000 0.006083401 0.090950478
```

```
fa_diff_pups <- get_pairdiff(relatedness[factors$age == 2, factors$age == 2],
                             fa_scores[factors$age == 2, ], df = F)

for (i in seq_along(1:4)) {
        assign(paste("f", i, "_diff", sep=""), fa_diff_pups[, i+1])
}

rel_dist <- as.dist(relatedness[factors$age == 2, factors$age == 2])
ecodist::mantel(rel_dist ~ f1_diff + f2_diff + f3_diff + f4_diff, mrank = T)                # tests for
```

Linear model for associations between genetic relatedness and factor scores as explanatory variables - Pups

```
##     mantelr       pval1       pval2       pval3   llim.2.5%  ulim.97.5%
##  0.02435999  0.34200000  0.65900000  0.65100000 -0.01870531  0.06230953
```

```
ecodist::mantel(rel_dist ~ f2_diff + f1_diff + f3_diff + f4_diff, mrank = T)                # tests for
```

```
##     mantelr       pval1       pval2       pval3   llim.2.5%  ulim.97.5%
##  0.01340994  0.40500000  0.59600000  0.82400000 -0.02536771  0.04755427
```

```
ecodist::mantel(rel_dist ~ f3_diff + f2_diff + f1_diff + f4_diff, mrank = T)                # tests for
```

```
##    mantelr      pval1      pval2      pval3  llim.2.5% ulim.97.5%
## 0.08487309 0.06500000 0.93600000 0.16100000 0.03487686 0.13285137
```

```
ecodist::mantel(rel_dist ~ f4_diff + f3_diff + f2_diff + f1_diff, mrank = T)                # tests for
```

```
##     mantelr       pval1       pval2       pval3   llim.2.5%  ulim.97.5%
## -0.06239643  0.92000000  0.08100000  0.18200000 -0.10742584 -0.01830814
```

```
# create data frame
col_df <- cbind(factors["colony"], fa_scores)
# reduce model by deletion testing
col_reduced <- MinMod(col_df)
```

Colony differences in factor scores: Just factor 4 shows significant differences.

```
##
## Call:
## glm(formula = depVar ~ ., family = family, data = bestmodeldf)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.8127  -0.2569  -0.1038   0.2719   0.7243
##
```

```
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.51220    0.03696   40.91  < 2e-16 ***
## F4           0.38454    0.03791   10.14 5.08e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for gaussian family taken to be 0.1120233)
##
##     Null deviance: 20.4878  on 81  degrees of freedom
## Residual deviance:  8.9619  on 80  degrees of freedom
## AIC: 57.179
##
## Number of Fisher Scoring iterations: 2
```

```
col_reduced_df <- col_reduced[[1]]
# dev explained
dev_expl <- (col_reduced_df$null.deviance - col_reduced_df$deviance) / col_reduced_df$null.deviance
# deletion test single variable
table <- DelTestVar(col_reduced[[1]])
```

```
##              Estimate Deviance Explained        F   P (F-test)
## (Intercept) 1.5121951                 NA       NA           NA
## F4          0.3845353           56.25756 102.8887 5.082064e-16
##             P (Chisquared-test)
## (Intercept)                  NA
## F4                3.545071e-24
```

Identification of substance subsets.

```
# subsets and identification
library(vegan)
library(ggplot2)
require(dplyr)
```

```
## Loading required package: dplyr
```

```
## Warning: package 'dplyr' was built under R version 3.2.0
```

```
##
## Attaching package: 'dplyr'
##
## The following object is masked from 'package:MASS':
##
##     select
##
## The following object is masked from 'package:stats':
##
##     filter
##
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
require(magrittr)
```

```
## Loading required package: magrittr
```

```
## Warning: package 'magrittr' was built under R version 3.2.0
```

```
library(vegan)
library(reshape2)
```

```
## Warning: package 'reshape2' was built under R version 3.2.0
```

Similarity percentages analysis (SIMPER) identifies the contribution of a specific compound to group similarity / dissimilarity. ANOSIM was used to test whether a small subset of the compounds with the highest contributions shows significant patterns.

Identification of best substances encoding mother-offspring similarity.

```
# results from simper analysis in Primer-E
mp_simp <- read.csv(".\\files\\simper_mp_results.csv", colClasses = c("character", "numeric"))

# mother offspring similarity

# overall
vegan::anosim(dat = scent[mp_simp$comp], grouping = factors$family,
       distance = "bray", permutations = 1000)
```

```
##
## Call:
## vegan::anosim(dat = scent[mp_simp$comp], grouping = factors$family,      permutations = 1000, distan
## Dissimilarity: bray
##
## ANOSIM statistic R: 0.6787
##       Significance: 0.000999
##
## Permutation: free
## Number of permutations: 1000
```

```
# within colony 1 (Special study beach)
vegan::anosim(dat = scent[factors$colony == 1, mp_simp$comp],
       grouping = factors[factors$colony == 1, ]$family,
       distance = "bray", permutations = 1000)
```

```
##
## Call:
## vegan::anosim(dat = scent[factors$colony == 1, mp_simp$comp],      grouping = factors[factors$colony
## Dissimilarity: bray
##
```

```
## ANOSIM statistic R: 0.5304
##       Significance: 0.000999
##
## Permutation: free
## Number of permutations: 1000
```

```
# within colony 2 (Freshwater beach)
vegan::anosim(dat = scent[factors$colony == 2, mp_simp$comp],
       grouping = factors[factors$colony == 2, ]$family,
       distance = "bray", permutations = 1000)
```

```
##
## Call:
## vegan::anosim(dat = scent[factors$colony == 2, mp_simp$comp],      grouping = factors[factors$colony
## Dissimilarity: bray
##
## ANOSIM statistic R: 0.3066
##       Significance: 0.001998
##
## Permutation: free
## Number of permutations: 1000
```

Identification of best substances encoding colony dissimilarity.

```
# colony dissimilarity, best substances ----------------------------------------
simp_colony <- vegan::simper(scent, factors$colony)

# getting 15 best substances and their contribution to colony dissimilarity
simp_colony_names <- rownames(summary(simp_colony, ordered = TRUE)[[1]])[1:15]
contribution <- summary(simp_colony, ordered = TRUE)[[1]]$contr[1:15]

# indices of colony substances (58,62,68,74,86,89,90,98,106,107,110,164,181,189,211)
ind_col <- paste(which(names(scent)%in%simp_colony_names), collapse = ",")

# connect to data frame and compute contribution in percent
col_simp <- data.frame(comp = simp_colony_names, contrib = contribution*100, stringsAsFactors = FALSE)
col_simp
```

```
##           comp  contrib
## 1  15.45769231 3.006785
## 2   16.3974359 2.419310
## 3  26.78859155 2.069715
## 4  19.52538462 1.965096
## 5       21.405 1.894868
## 6  21.34820513 1.671251
## 7  19.72268293 1.669923
## 8  30.80365385 1.482931
## 9   38.5183871 1.440400
## 10 17.40942623 1.330158
## 11 20.51086207 1.288827
## 12 33.63655172 1.266612
## 13 21.57529412 1.208037
## 14 15.74219178 1.180692
## 15 19.66514286 1.128883
```

```r
# overall (number of permutations is 1000 instead of 10,000 in the paper)
anosim(dat = scent[col_simp$comp], grouping = factors$colony,
       distance = "bray", permutations = 1000)
```

```
##
## Call:
## anosim(dat = scent[col_simp$comp], grouping = factors$colony,      permutations = 1000, distance = "
## Dissimilarity: bray
##
## ANOSIM statistic R: 0.7726
##       Significance: 0.000999
##
## Permutation: free
## Number of permutations: 1000
```

```r
################ BIO-ENV bootstrap procedure ###################################
#### run seperately on multicore server #####
#### aim: resampling test for finding the substances associated with genetic
#### relatedness. Basic assumption: Each variable will be tested in many different
#### environments (individuals, other variables), which will prevent spurious
#### correlations, as the really important substances will occur in best subsets
#### in many different constellations. (see methods section)

# parallel computing using 40 cores, takes some days nevertheless and is just
# shown here.

library(vegan)
library(stringr)
library(dplyr)
library(snow)
library(snowfall)
source("bio.env.R")

# number of cores
ncores <- 2
# subset
scent_mum <- filter(scent, factors$age == 1)
relate_mum <-  relatedness[factors$age == 1, factors$age == 1]

# initialise results vector
all_best <- vector()

# initialise cluster
sfInit(parallel=TRUE, cpus=ncores, type="SOCK")

# export libraries and main function to all cores
sfSource("bio.env.R")
sfLibrary(vegan)
sfLibrary(stringr)
sfLibrary(dplyr)
```

```
bootstrap <- function(iter_comp) { # main resampling function

        for (i in 1:500) {
                # sample 20 out of 41 mothers, indices
                ind_obs <- sort(sample(1:41, size = 20, replace = F))
                # subset relate_mum and scent_mum
                reltemp <- 1-as.dist(relate_mum[ind_obs, ind_obs])
                abundtemp <- scent_mum[ind_obs, ]

                for (i in iter_comp) {
                        # sample 10 compounds
                        index_comps <- sort(sample(1:213, size = 10, replace = F))
                        abundtemp_sub <- abundtemp[, index_comps]
                        # get vector with 0 for null-column and 1 for non-null column
                        nullcomps <- apply(abundtemp_sub, 2, function(x) sum(x>0))
                        abundtemp_sub <- subset(abundtemp_sub,
                                                subset = c(rep(TRUE,nrow(abundtemp_sub))),
                                                select = (nullcomps >= 2))
                        # new iteration if too less substances left
                        if (ncol(abundtemp_sub) <= 2) next

                        # main function: bio.env finds subset that mostly correlates
                        # with relatedness
                        results <- bio.env(reltemp, abundtemp_sub,
                                           var.dist.method = "bray",
                                           scale.fix = F, scale.var = F)

                        mods <- results$best.model.vars
                        best <- unlist(str_split(mods, ","))
                        all_best <- append(all_best, best)
                        # write(best, file = "best.txt", append = TRUE, sep = " ")
                }
        }
        return(all_best)
}
# export objects
sfExportAll(except = NULL, debug = FALSE)
sfClusterEval(ls())

# create list of 500 iterations for all cores
vals <- list()
for (i in 1:ncores) {
        vals[[i]] <- 1:500
}
# run analysis
# best is a list of all best subsets
best <- sfLapply(vals, bootstrap)
# stop cluster
sfStop()
# bring all results
results <- unlist(best)

############################## END ##############################################
```

Identification of substanced encoding relatedness. Analysing results from the BIO-ENV bootstrap analysis.

```
# substance occurences are counted in sorted in a table
best_mums <- read.csv("files/bootstrap_mums.csv",row.names=1)

# subset
scent_mum <- dplyr::filter(scent, factors$age == 1)
relate_mum <-  1-relatedness[factors$age == 1, factors$age == 1]

# get vectors of best substance names
sub_names_mums <- row.names(best_mums)

statm <- vector()
sigm <- vector()

# compute mantelR for an increasing set of best substances
for (i in 2:100) {
        bc_dist <- vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method = "bray")
        mod <- vegan::mantel(relate_mum, bc_dist, na.rm = T, method = "spearman")
        statm <- append(statm, mod$statistic)
        sigm <- append(sigm, mod$sig)
}
```

```
## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): you have empty rows: their dissimilarities may be meaningless in
## method "bray"

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): missing values in results

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): you have empty rows: their dissimilarities may be meaningless in
## method "bray"

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): missing values in results

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): you have empty rows: their dissimilarities may be meaningless in
## method "bray"

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): missing values in results

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): you have empty rows: their dissimilarities may be meaningless in
## method "bray"

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): missing values in results
```

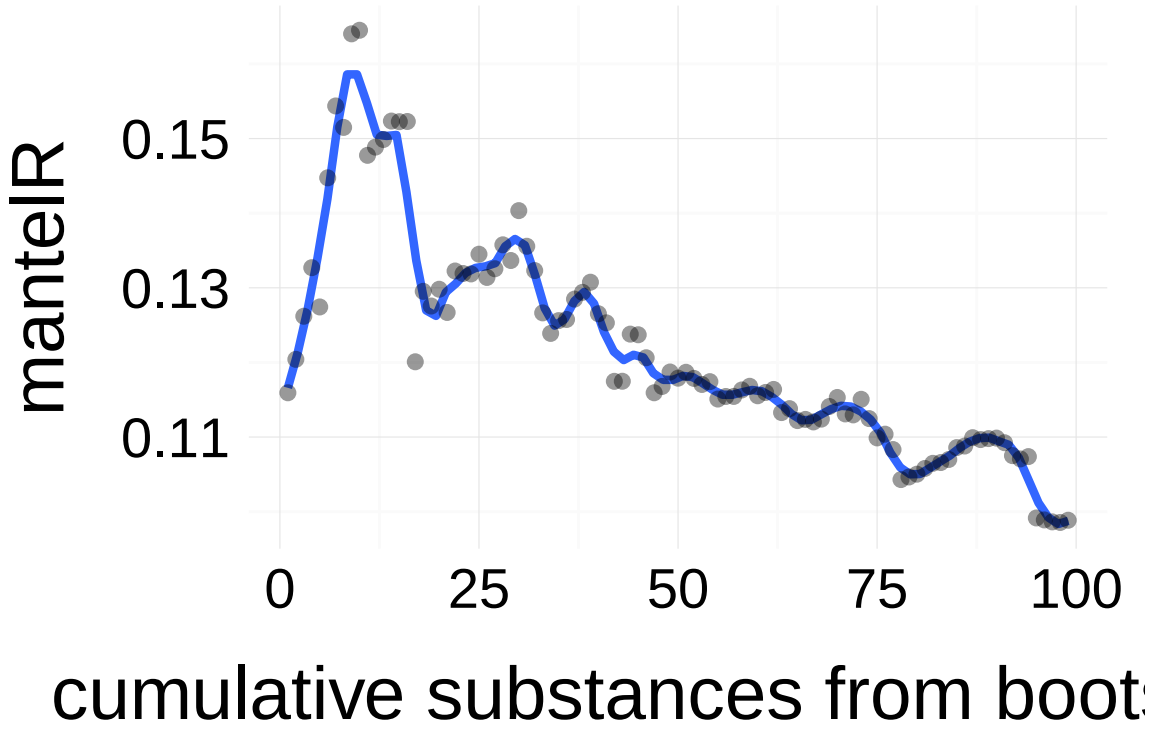
```
## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): you have empty rows: their dissimilarities may be meaningless in
## method "bray"

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): missing values in results

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): you have empty rows: their dissimilarities may be meaningless in
## method "bray"

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): missing values in results

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): you have empty rows: their dissimilarities may be meaningless in
## method "bray"

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): missing values in results

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): you have empty rows: their dissimilarities may be meaningless in
## method "bray"

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): missing values in results

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): you have empty rows: their dissimilarities may be meaningless in
## method "bray"

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): missing values in results

## Warning in vegan::vegdist(scent_mum[, sub_names_mums[1:i]], method =
## "bray"): you have empty rows: their dissimilarities may be meaningless in
## method "bray"
```

```
stat_df <- data.frame(num_comps = 1:length(statm), mantelR = statm)
```

Plotting results.

```
library(grid)
# simple plot
ggplot(stat_df, aes(x = num_comps, y = mantelR)) +
        stat_smooth(se = FALSE, span = 0.11, size = 1.5, method = "loess") +
        geom_point(colour = "black", size = 3, alpha = 0.4) +
        theme_minimal(base_size = 26) +
        theme(strip.text.x = element_text(vjust=1,size = 18),
```

```
        axis.title.x = element_text(vjust= -2 ,size = 28),
        axis.title.y = element_text(vjust=3,size = 28),
        axis.ticks.x = element_blank(),
        axis.ticks.y = element_blank(),
        plot.margin = (unit(c(.5, .5, 2, 2), "cm"))) +
    # scale_x_continuous(breaks=c(seq(from = 0.8, to = 1.20, by = 0.1))) +
    #geom_text(aes(0.85,80, label="(a) r = 0.34, p = 0.027"),size=4) +
    xlab("cumulative substances from bootstrap") +
    ylab("mantelR")
```



Check whether scent similarity at the most important compounds is associated with genetic relatedness.

```
# indices of the 10 best compounds associated with relatedness -----------------
comp_ind_m <- c(36,52,86,88,96,103,110,203,206,207)

# bray curtis similarity matrix based on this 10 compounds
scent_bc <- 1-(as.matrix(vegan::vegdist(as.matrix(scent[factors$age == 1, comp_ind_m])),
                         method = "bray"))
```

```
## Warning in vegan::vegdist(as.matrix(scent[factors$age == 1, comp_ind_m])):
## you have empty rows: their dissimilarities may be meaningless in method
## "bray"
```

```
## Warning in vegan::vegdist(as.matrix(scent[factors$age == 1, comp_ind_m])):
## missing values in results
```

```r
# relatedness matrix
rel_m <- relatedness[1:41, 1:41]

# mantel test for association between both
vegan::mantel(rel_m, scent_bc, method = "spearman", permutation = 1000, na.rm = TRUE)
```

```
##
## Mantel statistic based on Spearman's rank correlation rho
##
## Call:
## vegan::mantel(xdis = rel_m, ydis = scent_bc, method = "spearman",      permutations = 1000, na.rm = T
##
## Mantel statistic r: 0.164
##       Significance: 0.001998
##
## Upper quantiles of permutations (null model):
##    90%    95%  97.5%    99%
## 0.0652 0.0891 0.1122 0.1319
## Permutation: free
## Number of permutations: 1000
```