RCode - MHC class II genotype does not contribute towards the chemical encoding of heterozygosity and relatedness in a wild vertebrate population

J. Tebbe

16/03/23

Packages

```r
if (!require("magrittr", quietly = TRUE)) {
install.packages("magrittr")
library(magrittr)
} else {
library(magrittr) # pipe operators
}

if (!require("tidyverse", quietly = TRUE)) {
install.packages("tidyverse")
library(tidyverse)
} else {
library(tidyverse) # package collection for easy and pretty data science with R
}


if (!require("phyloseq", quietly = TRUE)) {
if (!require("BiocManager", quietly = TRUE)) {
install.packages("BiocManager")
}
BiocManager::install(pkgs = "phyloseq")
library(phyloseq) # phyloseq objects
} else {
library(phyloseq) # phyloseq objects
}

if (!require("GCalignR", quietly = TRUE)) {
install.packages("GCalignR")
library(GCalignR)
} else {
library(GCalignR) # handling/aligning chromatograms
}

if (!require("inbreedR", quietly = TRUE)) {
install.packages("inbreedR")
library(inbreedR)
} else {
```

```
library(inbreedR) # population genetic analyses
}

if (!require("vegan", quietly = TRUE)) {
install.packages("vegan")
library(vegan)
} else {
library(vegan) # statistical tools
}

if (!require("ggpubr", quietly = TRUE)) {
install.packages("ggpubr")
library(ggpubr)
} else {
library(ggpubr) # ggplot grid and plot alignment functions
}

if (!require("ape", quietly = TRUE)) {
install.packages("ape")
library(ape)
} else {
library(ape) # handling phylogenetic tree data
}

if (!require("performance", quietly = TRUE)) {
install.packages("performance")
library(performance)
} else {
library(performance) # tools for models
}

if (!require("MuMIn", quietly = TRUE)) {
install.packages("MuMIn")
library(MuMIn)
} else {
library(MuMIn) # tools for models
}


# archived package as is dependend on `fts` package
# for execution of the code, users need to manually install Rtools to be able
# to install packages `Demerelate` and `fts`

library(fts)

library(Demerelate)
```

Packages for relatedness calculations

Not supported on newer versions of R, to execute code you must have Rtools installed on your machine in order to load older version of the `Demerelate` and `fts` package.

```
if (!require("remotes", quietly = TRUE)) {
install.packages("remotes")
```

```r
library(remotes)
} else {
library(remotes) # tools for models
}

if (!require("fts", quietly = TRUE)) {
install_version("fts", "0.9.9.2")
  library(fts)
} else {
library(fts) # tools for models
}

if (!require("Demerelate", quietly = TRUE)) {
install_version("Demerelate", "0.9.9.2")
  library(Demerelate)
} else {
library(Demerelate) # tools for models
}
```

Subset scent data to correlate same individuals

```r
## read in meta data
meta <- read.table(file = "data/arga_metadata.txt", sep = "\t") %>%
  `colnames<-`(unlist(.[1,])) %>%
  .[-1,]


## normalise area and return a data frame
scent <- norm_peaks(aligned_peak_data,
                    conc_col_name = "area",
                    rt_col_name = "time",
                    out = "data.frame")
## common transformation for abundance data to reduce the extent of mean-variance trends
scent <- log(scent + 1)

n_scnt <- rownames(scent)

keep_i <- match(meta$id, n_scnt)

scent %<>%
  .[keep_i, ] %>%
  `rownames<-`(meta$real_id)


## NMDS with reduced data
## GCalignR contains factors for the chemical dataset
data("peak_factors")
peak_factors <- peak_factors[match(meta$id, rownames(peak_factors)),] %>%
  `rownames<-`(meta$real_id)

## keep order of rows consistent
scent <- scent[match(rownames(peak_factors),rownames(scent)),]
## NMDS using Bray-Curtis dissimilarities
```

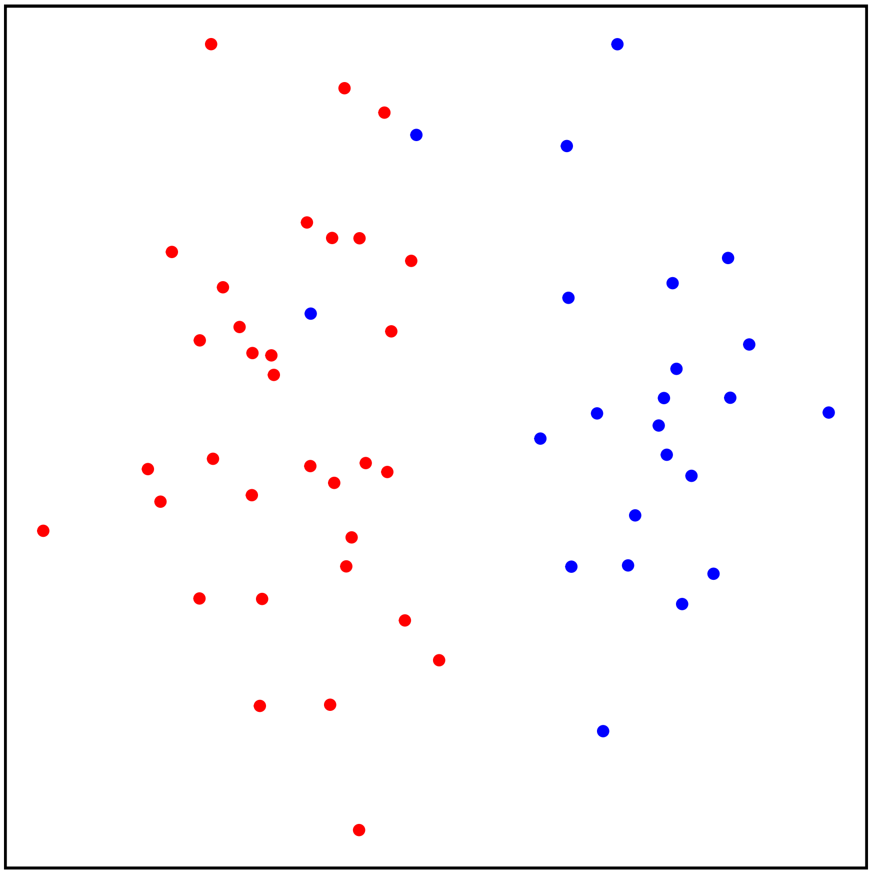
```r
scent_nmds.obj <- vegan::metaMDS(comm = scent, distance = "bray")
```

```
## Run 0 stress 0.2373122
## Run 1 stress 0.2619666
## Run 2 stress 0.2372465
## ... New best solution
## ... Procrustes: rmse 0.003287153  max resid 0.01771477
## Run 3 stress 0.2410723
## Run 4 stress 0.2611266
## Run 5 stress 0.2372465
## ... Procrustes: rmse 1.271277e-05  max resid 8.75837e-05
## ... Similar to previous best
## Run 6 stress 0.2691489
## Run 7 stress 0.2372465
## ... New best solution
## ... Procrustes: rmse 7.211019e-06  max resid 4.205237e-05
## ... Similar to previous best
## Run 8 stress 0.2527846
## Run 9 stress 0.2372465
## ... Procrustes: rmse 3.296201e-06  max resid 1.79671e-05
## ... Similar to previous best
## Run 10 stress 0.2915174
## Run 11 stress 0.2373122
## ... Procrustes: rmse 0.0032866  max resid 0.01771751
## Run 12 stress 0.2706668
## Run 13 stress 0.405117
## Run 14 stress 0.2373122
## ... Procrustes: rmse 0.003286632  max resid 0.01771787
## Run 15 stress 0.2924491
## Run 16 stress 0.2471852
## Run 17 stress 0.2659259
## Run 18 stress 0.25261
## Run 19 stress 0.2372465
## ... Procrustes: rmse 7.88252e-06  max resid 3.716299e-05
## ... Similar to previous best
## Run 20 stress 0.2602675
## *** Best solution repeated 3 times
```

```r
## get x and y coordinates
scent_nmds <- as.data.frame(scent_nmds.obj[["points"]])
## add the colony as a factor to each sample
scent_nmds <- cbind(scent_nmds,colony = peak_factors[["colony"]])
## quick plotting
scent_plot <- ggplot(data = scent_nmds,aes(MDS1,MDS2,color = colony)) +
  geom_point() +
  theme_void() +
  scale_color_manual(values = c("blue","red")) +
  theme(panel.background = element_rect(colour = "black",
                                        size   = 1,
                                        fill   = NA),
        aspect.ratio    =  1,
        legend.position = "none")
```

```
## Warning: The `size` argument of `element_rect()` is deprecated as of ggplot2 3.4.0.
## i Please use the `linewidth` argument instead.
```

```
scent_plot
```



Calculate MHC heterozygosity relatedness between individuals

```
## read in mhc genotype data
mhc_het_dat <- read.table("data/clone_mhc_het.txt")
## restructure `mhc_het_dat`to fit `Demerelate()::inputdata)
## id and colony as factors; alleles as integers or numeric
## otherwise `rxy`cannot handle computations
mhc_het_dat %<>%
  rownames_to_column(., var = "id") %>%
  # mutate(., a1 = str_pad(a1, 2, pad = "0")) %>%
  # mutate(., a2 = str_pad(a2, 2, pad = "0")) %>%
  mutate(., colony = as.factor(rep("col", 56))) %>%
  mutate(., id = as.factor(id)) %>%
  .[,-4] %>%
  relocate(., colony, .before = a1)
  ## order mhc_het_dat$id after meta$real_id
  ## so data is consistently ordered same in all data.frames

## get matching indeces
id_index <- match(meta$real_id, mhc_het_dat$id)
## sort correspondingly
mhc_het_dat %<>% .[id_index,]
```

```r
## calculate relatedness after Queller & Goodnight
mhc_relatedness_res <- Demerelate(inputdata = mhc_het_dat,
                                  value = "rxy",
                                  object = T,
                                  NA.rm = F,
                                  Fis = F)
```

```
## Warning in Demerelate(inputdata = mhc_het_dat, value = "rxy", object = T, : Careful, bi-allelic marke
##   Especially, rxy and ritland estimator are not defined when bi-allelic estimates are used with allel
##   You should consider removing bi-allelics which tend to have very evenly distributed alleles or swi
##   Be careful even if allele frequencies are not perfectly 0.5, during randomizations problems may occ
```

```r
mhc_relatedness <- unlist(mhc_relatedness_res$Empirical_Relatedness)

## fill distant matrix / make sure that it follows same systematics as previous distance matrices
## create empty matrix with equal rows and cols similar to sample size of indidivuals
relate_mat_mhc <- matrix(nrow = 56, ncol = 56)
## fill distance matrix row wise, thus fill upper.tri
relate_mat_mhc[upper.tri(relate_mat_mhc)] <- mhc_relatedness
## transpose to keep consistency with other distance matrices
relate_mat_mhc <- t(relate_mat_mhc)
relate_mat_mhc %<>% `colnames<-`(meta$real_id) %>% `rownames<-`(meta$real_id)

## vectorize again to identify whether relatedness pairs were consistent in the first place
a <- relate_mat_mhc %>% as.vector() %>% na.omit()
```

Create vectorized distance measurements for scent data

```r
# bray-curtis distance measurement on scent profiles
scent_dist <- vegdist(scent) %>% as.matrix()
scent_dist[upper.tri(scent_dist, diag = T)] <- NA
b <- scent_dist %>% as.vector() %>% na.omit()
```

Generate UniFrac distances from MHC DQB II individual genotypes

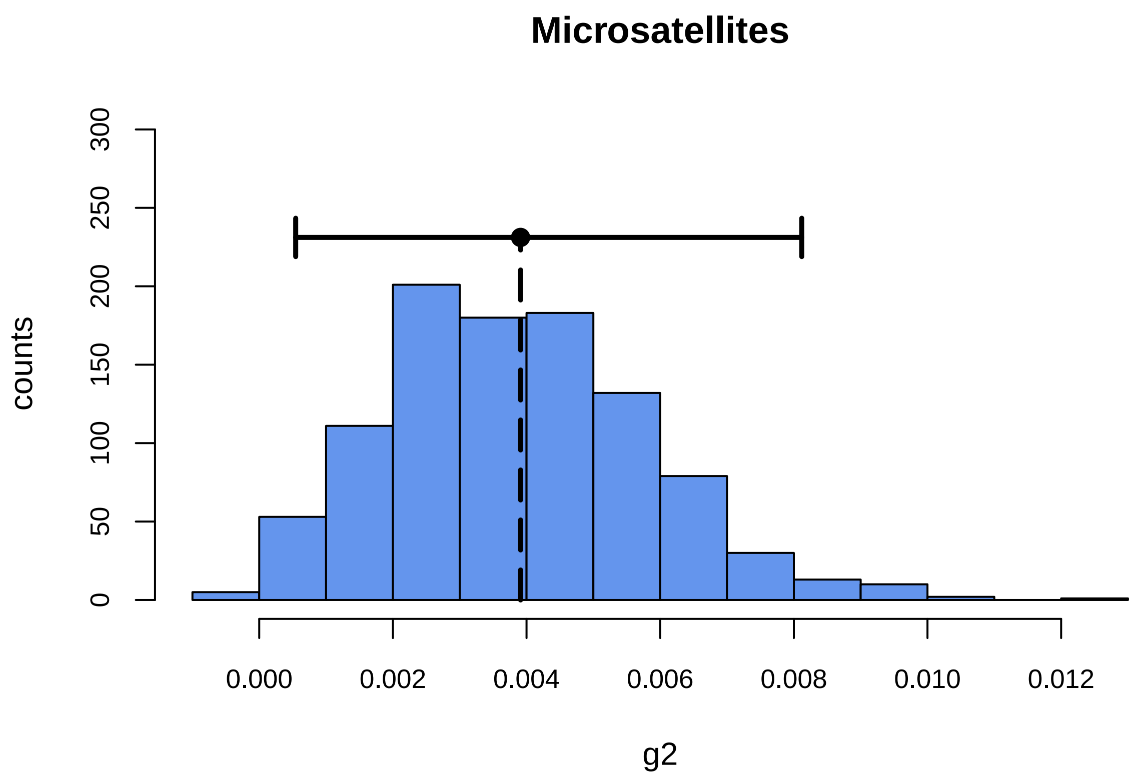
```r
# handle genotypes as otu table
phylo_mat <- read.table("data/phyloseq-mat.txt") %>%
  as.matrix()

# make sample names consistent
n <- match(meta$real_id, colnames(phylo_mat))

phylo_mat %<>% .[, n] %>%
  otu_table(., taxa_are_rows = T)

# create phylogenetic tree from file
phylo_tree <- ape::read.tree("data/unifrac_tree_p.nwk")

# merge into Formal class phyloseq
arga_phylseq <- merge_phyloseq(phylo_mat, phylo_tree)

# create UniFrac as genetic diversity measurement for single locus data
mhc_dqb2_ufrac <- UniFrac(arga_phylseq, weighted = F) %>%
  # distances to distance matrix
  as.matrix()

# vectorize distances matrices
mhc_dqb2_ufrac[upper.tri(mhc_dqb2_ufrac, diag = T)] <- NA
c <- mhc_dqb2_ufrac %>% as.vector() %>% na.omit()
```

Calculate identity disequilibirum g2

```
msats_g2 <- read.table("data/msats_genotypes_inbreedR.txt", sep = "\t") %>%
  convert_raw()

g2 <- g2_microsats(msats_g2, nperm = 1000, nboot = 1000, CI = 0.95)
```

```
##
##  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, hell yeah!! ###
```

```
plot(g2, main = "Microsatellites",
     col = "cornflowerblue", cex.axis=0.85)
```



Calculate microsatellite relatedness values

create `data.frame` in correspondence to `Demerelate` input format

```
# read in genotype data table
msats_df <- read.table("data/msats_genotypes_inbreedR.txt", sep = "\t")


# update data.frame with additional info
# "delete" colony info, otherwise relatedness is only calculated for individuals
# within their own colonies -> no complete pairwise comparison
msats_df <- cbind(id = as.factor(rownames(msats_df)),
                  # colony = meta$colony,
                  colony = rep("col", 56),
                  msats_df[1:56,]) %>%
  # clear df from rownames/ only keep colnames/ variable names
  `rownames<-`(NULL)

msats_df[is.na(msats_df)] = 0

str(msats_df)

write.table(msats_df, file = "data/msats_genotypes_demerelate.txt",
            sep = "\t",
            row.names = F)
```

Calculate relatedness of individuals based on Queller & Goodnight

```
relatedness_results <- Demerelate(inputdata = msats_df,
                                  value = "rxy",
                                  object = T,
                                  NA.rm = F,
                                  Fis = F)
```

```
## Warning in Demerelate(inputdata = msats_df, value = "rxy", object = T, NA.rm = F, : Careful, bi-allel
##   Especially, rxy and ritland estimator are not defined when bi-allelic estimates are used with allel
##   You should consider removing bi-allelics which tend to have very evenly distributed alleles or swi
##   Be careful even if allele frequencies are not perfectly 0.5, during randomizations problems may occ
```

Coerce output to a vector

```
relatedness <- unlist(relatedness_results$Empirical_Relatedness)

## fill distant matrix / make sure that it follows same systematics as previous distance matrices
## create empty matrix with equal rows and cols similar to sample size of indidivuals
relate_mat <- matrix(nrow = 56, ncol = 56)
## fill distance matrix row wise, thus fill upper.tri
relate_mat[upper.tri(relate_mat)] <- relatedness
## transpose to keep consistency with other distance matrices
relate_mat <- t(relate_mat)
relate_mat %<>% `colnames<-`(meta$real_id) %>% `rownames<-`(meta$real_id)

## vectorize again to identify whether relatedness pairs were consistent in the first place
d <- relate_mat %>% as.vector() %>% na.omit()
```

Analyse Odour and genetic association by MHC DQB II and neutral genomic background

Create data.frame to plot in ggplot2

```
## substitute once tested correctly
## scent_mds shall contain similarity values but `b` contains
## dissimilarity values based on Bray-Curtis -> substracting
## dissmilarities from 1 returns similarities

model_rel.df <- cbind(mhc_rel = a, scent_mds = 1-b, ufrac = c, rel = d) %>%
  as.data.frame()
```

Custom theme to make plot aesthetics consistent

```r
# custom theme to ease figure creation
custom_theme <- ggplot2::theme_classic(base_size =20,
                                       base_line_size = 1,
                                       base_rect_size = 1) +
  ggplot2::theme(
    #c(top, right, bottom, left)
    plot.margin = margin(5.5,6.5,8,5.5, "pt"),
    panel.grid = element_blank(),
    axis.text = element_text(color = "black"),
    axis.title.x = element_text(vjust = -.75),
    axis.title.y = element_text(vjust = +2),
    axis.ticks = element_line(color = "black"),
    aspect.ratio = 1,
    legend.position = "none"
  )
```

Plot odour by mhc similarity

```r
# odour by mhc sim
panel1.a <- ggplot(data = model_rel.df,
       aes(x = ufrac, y = scent_mds)) +
  geom_point(size = 3.5,
             alpha = 0.25) +
  # geom_smooth(method = "lm",
  #             color = "orange") +
  scale_x_continuous(name = "MHC Unifrac distance") +
  scale_y_continuous(name = "Chemical similarity") +
  # labs(tag = "A") +
  custom_theme
```

Plot odour by relatedness

```r
# odour by relatedness
panel1.b <- ggplot(data = model_rel.df,
       aes(x = rel, y = scent_mds)) +
  geom_point(size = 3.5,
             alpha = 0.25) +
  # geom_smooth(method = "lm",
  #             color = "orange") +
  scale_x_continuous(name = "Relatedness") +
  scale_y_continuous(name = "Chemical similarity") +
  custom_theme
```

Model odour relationship on MHC and neutral genetic background

Pool underlying data dependencies

Create a function that generates pairwise variables in a systematic matter for pairwise comparisons

```r
# Function specification ---------------------------------------------------

## make into function, to create age, col and family ids for the pairs
## in similar manner

# for function: row and col names need then to be the values to cross in the right
# order

# Code execution ---------------------------------------------------------------
create_pair_vars <-function(row_cross, col_cross, split_vars = F){
  require(stringr)

  rc <- row_cross
  cc <- col_cross

  # create empy matrix
  # keep row and col names from existing distance matrices

  empty_mat <- matrix(nrow = length(rc),
                      ncol = length(cc)) %>%
    `colnames<-`(cc) %>%
    `rownames<-`(rc)

  # fill each matrix i,j-th cell with the crossing from their corresponding
  # i-th rowname and j-th colname
  for (i in 1:dim(empty_mat)[1]) {
    for (j in 1:dim(empty_mat)[2]) {

      empty_mat[i,j] <- paste0(rc[i], "/", cc[j])

    } # end j
  } # end i


  # delete `upper.tri()` of `empty_mat` to resemble structure of the other
  # distance matrices in use

  empty_mat[upper.tri(empty_mat, diag = T)] <- NA
  pair_vars <- empty_mat %>% as.vector() %>% na.omit()

  # split `pair_vars` if needed
  if (split_vars == T) {

    pair_vars1 <- sapply(pair_vars,
                         function(x){
                           str_split(x, pattern = "/")[[1]][1]
                         })

    pair_vars2 <- sapply(pair_vars,
                         function(x){
                           str_split(x, pattern = "/")[[1]][2]
                         })

    pair_vars_split <- list(pair_variable1 = pair_vars1,
                            pair_variable2 = pair_vars2)

    return(pair_vars_split)

  } else {
    return(pair_vars)
  }

} #end create_pair_vars
```

Helper function to combine double entries

```
## for x, overwrite specified replacer with specified value
f <- function(x, replacer, overwrite){
  if (x == replacer) {
    x <- overwrite
  } else {
    x <- x
  }
}
```

Transform model variables

```
agePaired <- create_pair_vars(row_cross = meta$maturity,
                              col_cross = meta$maturity) %>%
  sapply(., f, "P/M", "M/P")

colonyPaired <- create_pair_vars(row_cross = meta$colony,
                                 col_cross = meta$colony) %>%
  sapply(., f, "FWB/SSB", "SSB/FWB")

colonyID1 <- create_pair_vars(row_cross = meta$colony,
                              col_cross = meta$colony,
                              split_vars = T)[1] %>%
  unlist() %>%
  paste0("f", .) %>%
  as.vector()

colonyID2 <- create_pair_vars(row_cross = meta$colony,
                              col_cross = meta$colony,
                              split_vars = T)[2] %>%
  unlist() %>%
  paste0("f", .) %>%
  as.vector()

colonyBool <- ifelse(colonyID1 == colonyID2, 1, 0)

familyPaired <- create_pair_vars(row_cross = meta$family,
                                 col_cross = meta$family)

familyID1 <- create_pair_vars(row_cross = meta$family,
                              col_cross = meta$family,
                              split_vars = T)[1] %>%
  unlist() %>%
  paste0("f", .) %>%
  as.vector()
```

```r
familyID2 <- create_pair_vars(row_cross = meta$family,
                              col_cross = meta$family,
                              split_vars = T)[2] %>%
  unlist() %>%
  paste0("f", .) %>%
  as.vector()

pairID1 <- create_pair_vars(row_cross = meta$real_id,
                            col_cross = meta$real_id,
                            split_vars = T)[1] %>%
  unlist() %>%
  as.vector()

pairID2 <- create_pair_vars(row_cross = meta$real_id,
                            col_cross = meta$real_id,
                            split_vars = T)[2] %>%
  unlist() %>%
  as.vector()

familyBool <- ifelse(familyID1 == familyID2, 1, 0)
```

Update data.frame with model variables

```r
model_rel.df <- data.frame(model_rel.df,
                      agePaired = as.factor(agePaired),
                      colonyPaired = as.factor(colonyPaired),
                      colonyBool = as.factor(colonyBool),
                      familyPaired = as.factor(familyPaired),
                      familyID1 = as.factor(familyID1),
                      familyID2 = as.factor(familyID2),
                      familyBool = as.factor(familyBool),
                      pairID1 = as.factor(pairID1),
                      pairID2 = as.factor(pairID2))
```

Color Chemical similarity by same or different beach

'colonyBool' encodes whether individual from same colonies (SSB vs SSB and FWB vs FWB) are compared or from different colonies

Chemical similarity models

```r
# mhc
a1 <- lmerTest::lmer(scent_mds ~ ufrac + colonyBool + (1|familyBool) +
                       (1|pairID1) + (1|pairID2),
                     data = model_rel.df)
# relatedness
a2 <- lmerTest::lmer(scent_mds ~ rel + colonyBool + (1|familyBool) + (1|pairID1) +
                       (1|pairID2),
                     data = model_rel.df)
# mhc & relatedness
a3 <- lmerTest::lmer(scent_mds ~ rel + ufrac + colonyBool + (1|familyBool) +
                       (1|pairID1) + (1|pairID2),
                     data = model_rel.df)
```

```
# no genetic effect
a4 <- lmerTest::lmer(scent_mds ~ colonyBool + (1|familyBool) + (1|pairID1) +
                       (1|pairID2),
                     data = model_rel.df)

# compare model performance scores
compare_performance(a1, a2, a3, a4, rank = T) %>%
  arrange(Name)
```

```
## # Comparison of Model Performance Indices
##
## Name |           Model | R2 (cond.) | R2 (marg.) |   ICC |  RMSE | Sigma | AIC weights | AICc weight
## -------------------------------------------------------------------------------------------------------
## a1   | lmerModLmerTest |      0.692 |      0.142 | 0.641 | 0.060 | 0.062 |       0.198 |        0.19
## a2   | lmerModLmerTest |      0.686 |      0.145 | 0.633 | 0.060 | 0.062 |       0.210 |        0.20
## a3   | lmerModLmerTest |      0.685 |      0.145 | 0.631 | 0.060 | 0.062 |       0.081 |        0.08
## a4   | lmerModLmerTest |      0.694 |      0.141 | 0.643 | 0.060 | 0.062 |       0.511 |        0.51
```

```
summary(a2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_mds ~ rel + colonyBool + (1 | familyBool) + (1 | pairID1) +
##     (1 | pairID2)
##    Data: model_rel.df
##
## REML criterion at convergence: -3942.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.1298 -0.6832 -0.0735  0.5746  3.7786
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  pairID2    (Intercept) 0.001346 0.03669
##  pairID1    (Intercept) 0.001322 0.03636
##  familyBool (Intercept) 0.003931 0.06270
##  Residual               0.003820 0.06181
## Number of obs: 1540, groups:  pairID2, 55; pairID1, 55; familyBool, 2
##
## Fixed effects:
##              Estimate Std. Error        df t value Pr(>|t|)
## (Intercept) 3.179e-01  4.624e-02 1.096e+00   6.875    0.078 .
## rel         8.184e-03  2.085e-02 1.325e+03   0.393    0.695
## colonyBool1 8.396e-02  8.041e-03 1.219e+02  10.441   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) rel
## rel         -0.095
## colonyBool1 -0.159  0.012
```

```
summary(a4)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_mds ~ colonyBool + (1 | familyBool) + (1 | pairID1) + (1 |
##     pairID2)
##    Data: model_rel.df
##
## REML criterion at convergence: -3948.3
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.1384 -0.6829 -0.0753  0.5819  3.7627
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  pairID2    (Intercept) 0.001349 0.03673
##  pairID1    (Intercept) 0.001326 0.03641
##  familyBool (Intercept) 0.004214 0.06492
##  Residual               0.003817 0.06178
## Number of obs: 1540, groups:  pairID2, 55; pairID1, 55; familyBool, 2
##
## Fixed effects:
##               Estimate Std. Error        df t value Pr(>|t|)
## (Intercept) 3.197e-01  4.755e-02 1.094e+00   6.723   0.0801 .
## colonyBool1 8.392e-02  8.049e-03 1.220e+02  10.426   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## colonyBool1 -0.154
```

```
# if interested
# check model performance by
# check_model(a2)
#
```

Correlations of genetic main effects

```
# correlation of ufrac and relatedness
u_r_model1 <- lmerTest::lmer(ufrac ~ rel  + (1|pairID1) + (1|pairID2),
                             data = model_rel.df)
summary(u_r_model1)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ufrac ~ rel + (1 | pairID1) + (1 | pairID2)
##    Data: model_rel.df
##
## REML criterion at convergence: -663.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.4417 -0.3855  0.2726  0.6820  1.5012
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  pairID1  (Intercept) 0.002163 0.04651
##  pairID2  (Intercept) 0.001011 0.03180
##  Residual             0.035896 0.18946
## Number of obs: 1540, groups:  pairID1, 55; pairID2, 55
##
## Fixed effects:
##               Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)  7.973e-01  9.699e-03  6.442e+01  82.203  < 2e-16 ***
## rel         -2.389e-01  5.121e-02  1.424e+03  -4.665 3.37e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##     (Intr)
## rel -0.009
```

```
u_r_model2 <- lmerTest::lmer(ufrac ~ rel  + (1|pairID1) + (1|pairID2) + (1|familyBool),
                             data = model_rel.df)
summary(u_r_model2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ufrac ~ rel + (1 | pairID1) + (1 | pairID2) + (1 | familyBool)
##    Data: model_rel.df
##
## REML criterion at convergence: -674.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.4734 -0.3792  0.2722  0.6769  2.1501
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  pairID1    (Intercept) 0.002152 0.04639
##  pairID2    (Intercept) 0.001076 0.03280
##  familyBool (Intercept) 0.017431 0.13203
##  Residual               0.035545 0.18853
## Number of obs: 1540, groups:  pairID1, 55; pairID2, 55; familyBool, 2
##
## Fixed effects:
##              Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)   0.71073    0.09678   0.99211   7.344   0.0874 .
## rel          -0.12669    0.05942 922.85885  -2.132   0.0333 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##     (Intr)
## rel -0.126
```

```
compare_performance(u_r_model1, u_r_model2, rank = T)
```

```
## Some of the nested models seem to be identical and probably only vary in
##   their random effects.
```

```
## # Comparison of Model Performance Indices
```

```
##
## Name       |           Model | R2 (cond.) | R2 (marg.) |   ICC |  RMSE | Sigma | AIC weights | AICc
## ------------------------------------------------------------------------------------------------------
## u_r_model2 | lmerModLmerTest |      0.369 |      0.003 | 0.368 | 0.185 | 0.189 |       0.874 |
## u_r_model1 | lmerModLmerTest |      0.095 |      0.015 | 0.081 | 0.186 | 0.189 |       0.126 |
```

```
u_r_model3 <- lmerTest::lmer(ufrac ~ rel + colonyBool + (1|pairID1) + (1|pairID2) + (1|familyBool),
                             data = model_rel.df)
summary(u_r_model3)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ufrac ~ rel + colonyBool + (1 | pairID1) + (1 | pairID2) + (1 |
##     familyBool)
##    Data: model_rel.df
##
## REML criterion at convergence: -667.8
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.4731 -0.3790  0.2729  0.6765  2.1483
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  pairID1    (Intercept) 0.002174 0.04663
##  pairID2    (Intercept) 0.001089 0.03300
##  familyBool (Intercept) 0.017402 0.13192
##  Residual               0.035555 0.18856
## Number of obs: 1540, groups:  pairID1, 55; pairID2, 55; familyBool, 2
##
## Fixed effects:
##               Estimate Std. Error         df t value Pr(>|t|)
## (Intercept)  7.113e-01  9.730e-02  1.014e+00   7.311   0.0843 .
## rel         -1.266e-01  5.945e-02  9.168e+02  -2.129   0.0335 *
## colonyBool1 -7.424e-04  1.316e-02  2.090e+02  -0.056   0.9551
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) rel
## rel         -0.126
## colonyBool1 -0.110  0.013
```

```
compare_performance(u_r_model1, u_r_model2, u_r_model3, rank = T)
```

```
## Some of the nested models seem to be identical and probably only vary in
##   their random effects.
```

```
## # Comparison of Model Performance Indices
##
## Name       |           Model | R2 (cond.) | R2 (marg.) |   ICC |  RMSE | Sigma | AIC weights | AICc
## ------------------------------------------------------------------------------------------------------
## u_r_model2 | lmerModLmerTest |      0.369 |      0.003 | 0.368 | 0.185 | 0.189 |       0.661 |
## u_r_model3 | lmerModLmerTest |      0.369 |      0.003 | 0.368 | 0.185 | 0.189 |       0.243 |
## u_r_model1 | lmerModLmerTest |      0.095 |      0.015 | 0.081 | 0.186 | 0.189 |       0.096 |
```

```r
summary(u_r_model2)# colony effect unsubstantial but family important!
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ufrac ~ rel + (1 | pairID1) + (1 | pairID2) + (1 | familyBool)
##    Data: model_rel.df
##
## REML criterion at convergence: -674.6
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -4.4734 -0.3792  0.2722  0.6769  2.1501
##
## Random effects:
##  Groups     Name        Variance Std.Dev.
##  pairID1    (Intercept) 0.002152 0.04639
##  pairID2    (Intercept) 0.001076 0.03280
##  familyBool (Intercept) 0.017431 0.13203
##  Residual               0.035545 0.18853
## Number of obs: 1540, groups:  pairID1, 55; pairID2, 55; familyBool, 2
##
## Fixed effects:
##              Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)   0.71073    0.09678   0.99211   7.344   0.0874 .
## rel          -0.12669    0.05942 922.85885  -2.132   0.0333 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##     (Intr)
## rel -0.126
```

```r
(aov_u_r <- anova(u_r_model2))
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##     Sum Sq Mean Sq NumDF  DenDF F value  Pr(>F)
## rel 0.1616  0.1616     1 922.86  4.5465 0.03325 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Model relationship between chemical diversity and mhc plus msats diversity

update data frame with meta data

Include information about MHC heterozygosity, sMLH from microsatellite data and chemical diversity by number of compounds per individual

```r
scent.abs <- ifelse(scent != 0, 1, 0)
compound_n <- apply(scent.abs, 1, sum)

names(compound_n) == meta$real_id
```

```
##  [1] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [16] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [31] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
## [46] TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE TRUE
```

```
# read in heterzygosity information
het_table <- read.table("data/arga_mhc_het.txt", sep = "\t")

# keep names consistent
match_het <- match(meta$real_id, rownames(het_table))
het_table %<>% .[match_het,]

# generate sMLH with microsatellite data
# table is pre-prepped, thus rows correspond to same individuals in meta data
smlh_res <- read.table("data/msats_genotypes_inbreedR.txt", sep = "\t") %>%
  # convert to inbreedR format
  convert_raw() %>%
  # generate sMLH
  sMLH()

meta %<>% cbind(., compound_n = compound_n,
                mhc_het = het_table$het,
                smlh = smlh_res)

meta %<>% mutate(
  real_id = as.factor(real_id),
  colony = as.factor(colony),
  maturity = as.factor(maturity),
  family = as.factor(family)
)
```

Compare chemical diversity models

Correlate Chemical diversity per sample with their sMLH and MHC, respectively. Also accounting maturity and family as fixed and random effect.

```
b1 <- lmerTest::lmer(compound_n ~ mhc_het + maturity + (1|family),
                     data = meta)

b2 <- lmerTest::lmer(compound_n ~ smlh + maturity + (1|family),
                     data = meta)

b3 <- lmerTest::lmer(compound_n ~ mhc_het + smlh + maturity + (1|family),
                     data = meta)

b4 <- lmerTest::lmer(compound_n ~ maturity + (1|family),
                     data = meta)


compare_performance(b1, b2, b3, b4, rank = T) %>% arrange(Name)
```

```
## # Comparison of Model Performance Indices
##
## Name |           Model | R2 (cond.) | R2 (marg.) |   ICC |  RMSE |  Sigma | AIC weights | AICc weigh
## ----------------------------------------------------------------------------------------------------
## b1   | lmerModLmerTest |      0.741 |      0.001 | 0.741 | 7.295 | 10.842 |       0.006 |        0.0
## b2   | lmerModLmerTest |      0.768 |      0.115 | 0.738 | 6.731 |  9.987 |       0.710 |        0.7
## b3   | lmerModLmerTest |      0.765 |      0.113 | 0.735 | 6.742 | 10.117 |       0.269 |        0.2
## b4   | lmerModLmerTest |      0.746 |      0.001 | 0.746 | 7.249 | 10.668 |       0.015 |        0.0
```

```
summary(b2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: compound_n ~ smlh + maturity + (1 | family)
##    Data: meta
##
## REML criterion at convergence: 456.5
##
## Scaled residuals:
##      Min       1Q   Median       3Q      Max
## -1.11347 -0.50879 -0.06924  0.29280  2.17312
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  family   (Intercept) 280.25   16.741
##  Residual              99.74    9.987
## Number of obs: 56, groups:  family, 36
##
## Fixed effects:
##             Estimate Std. Error      df t value Pr(>|t|)
## (Intercept)  -18.698     23.772  44.956  -0.787  0.43567
## smlh          75.812     23.976  44.618   3.162  0.00282 **
## maturityP     -4.473      3.164  24.671  -1.414  0.16991
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##           (Intr) smlh
## smlh      -0.990
## maturityP  0.248 -0.299
```

Correlate zygosity effects

```
smlh_het_m1 <- lmerTest::lmer(smlh ~ mhc_het + (1|family), data = meta)
summary(smlh_het_m1)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: smlh ~ mhc_het + (1 | family)
##    Data: meta
##
## REML criterion at convergence: -98.4
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.1936 -0.6565  0.1024  0.7186  1.9076
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  family   (Intercept) 0.001313 0.03624
##  Residual             0.007200 0.08486
## Number of obs: 56, groups:  family, 36
##
## Fixed effects:
```

```
##              Estimate Std. Error       df t value Pr(>|t|)
## (Intercept)  1.02866    0.02684 51.33505  38.329   <2e-16 ***
## mhc_het     -0.03813    0.03017 53.02191  -1.264    0.212
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##         (Intr)
## mhc_het -0.876
```

```
# check performance for including colony as fixed effect, as well
smlh_het_m2 <- lmerTest::lmer(smlh ~ mhc_het + colony + (1|family), data = meta)
summary(smlh_het_m2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: smlh ~ mhc_het + colony + (1 | family)
##    Data: meta
##
## REML criterion at convergence: -93.1
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.0951 -0.6533  0.1137  0.6874  1.8474
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  family   (Intercept) 0.001546 0.03932
##  Residual             0.007141 0.08451
## Number of obs: 56, groups:  family, 36
##
## Fixed effects:
##               Estimate Std. Error        df t value Pr(>|t|)
## (Intercept)  1.031014   0.028716 45.359094  35.903   <2e-16 ***
## mhc_het     -0.036797   0.030541 52.119411  -1.205    0.234
## colonySSB   -0.008389   0.026810 26.335039  -0.313    0.757
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##           (Intr) mhc_ht
## mhc_het   -0.799
## colonySSB -0.329 -0.074
```

```
(aov <- anova(smlh_het_m2))
```

```
## Type III Analysis of Variance Table with Satterthwaite's method
##            Sum Sq   Mean Sq NumDF  DenDF F value Pr(>F)
## mhc_het 0.0103668 0.0103668     1 52.119  1.4517 0.2337
## colony  0.0006993 0.0006993     1 26.335  0.0979 0.7568
```

```
compare_performance(smlh_het_m1, smlh_het_m2, rank = T)
```

```
## # Comparison of Model Performance Indices
##
## Name        |           Model | R2 (cond.) | R2 (marg.) |   ICC |  RMSE | Sigma | AIC weights | AICc
## ------------------------------------------------------------------------------------------------
## smlh_het_m2 | lmerModLmerTest |      0.202 |      0.029 | 0.178 | 0.076 | 0.085 |       0.277 |
## smlh_het_m1 | lmerModLmerTest |      0.178 |      0.028 | 0.154 | 0.077 | 0.085 |       0.723 |
```

Plot chemical complexity by mhc heterozygosity

```
panel2.a <- ggplot(data = meta,
                   aes(y = compound_n,
                       x = as.factor(mhc_het),
                       fill = as.factor(mhc_het),
                       color = as.factor(mhc_het))) +
  scale_fill_manual(values = c("darkgrey", "orange")) +
  geom_boxplot(width = 0.4,
               color = "black",
               size = 1) +
  geom_jitter(height = 0.02,
              width = 0.1,
              color = "black",
              size = 3.5,
             alpha = 0.25) +
  scale_x_discrete(name = "MHC heterozygosity",
                   breaks = c(0,1),
                   labels = c("homozygous", "heterozygous")) +
  scale_y_continuous(name = "Chemical diversity") +
  custom_theme
```

Plot chemical complexity by sMLH

```
panel2.b <- ggplot(data = meta,
                   aes(y = compound_n,
                       x = smlh)) +
  geom_point(size = 3.5,
             alpha = 0.25) +
  geom_smooth(method = "lm",
              se = T,
              color = "orange") +
  scale_x_continuous(name = "sMLH") +
  scale_y_continuous(name = "Chemical diversity") +
  scale_color_manual(name = "Senescence",
    values = c("#E8B54D", "#000000"),
    labels = c("Mother", "Pup")) +
  scale_fill_manual(name = "Senescence",
    values = c("#E8B54D", "#000000"),
    labels = c("Mother", "Pup")) +
  custom_theme
```

PERMANOVA for individual genotypes and alleles respectively

Create workable dataframe

```
# create data frame containing of:
  # individual substance count for every animal
  # an animals individual genotype, represented by 0 and 1 for a given number
  # of alleles (here ranging from 1 to 19)
```

```
idv_allele <- t(phylo_mat) %>%
  # coerce to data.frame
  as.data.frame() %>%
  # combine individual compound number with mhc genotype
  cbind(., compound_n) %>%
  # rename columns
  `colnames<-`(c(paste0("a",1:19), "compound_n"))
```

Run PERMANOVA on each allele

```
# run permanova to associate individual alleles to compound complexity
allele_permanova <-
  vegan::adonis2(compound_n ~ a1 + a2 + a3 + a4 + a5 + a6 + a7 + a8 + a9 + a10 +
                   a11 + a12 + a13 + a14 + a15 + a16 + a17 + a18 + a19,
               data = idv_allele)
# View results
allele_permanova
```

```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## vegan::adonis2(formula = compound_n ~ a1 + a2 + a3 + a4 + a5 + a6 + a7 + a8 + a9 + a10 + a11 + a12 +
##           Df SumOfSqs      R2      F Pr(>F)
## a1         1  0.01516 0.00847 0.4632  0.510
## a2         1  0.00961 0.00537 0.2936  0.671
## a3         1  0.00699 0.00390 0.2135  0.719
## a4         1  0.03789 0.02116 1.1577  0.271
## a5         1  0.00262 0.00146 0.0801  0.880
## a6         1  0.04424 0.02471 1.3517  0.242
## a7         1  0.02217 0.01238 0.6774  0.424
## a8         1  0.05646 0.03153 1.7250  0.164
## a9         1  0.00225 0.00126 0.0687  0.899
## a10        1  0.02395 0.01337 0.7316  0.404
## a11        1  0.02017 0.01126 0.6161  0.452
## a12        1  0.07517 0.04198 2.2966  0.130
## a13        1  0.01711 0.00955 0.5228  0.503
## a14        1  0.05789 0.03233 1.7688  0.165
## a15        1  0.13388 0.07476 4.0903  0.046 *
## a16        1  0.00265 0.00148 0.0811  0.888
## a17        1  0.01298 0.00725 0.3967  0.559
## a18        1  0.05919 0.03306 1.8085  0.178
## a19        1  0.01201 0.00671 0.3670  0.585
## Residual 36  1.17832 0.65801
## Total    55  1.79073 1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# give out p-values for each individual allele
pvals <- allele_permanova[1:19,5]

# correct p-values by fdr
pvals_corrected <- p.adjust(pvals, method = "fdr") %>% as.data.frame()
pvals_corrected
```

```
##             .
## 1  0.7939286
## 2  0.8499333
## 3  0.8538125
## 4  0.7355714
## 5  0.8990000
## 6  0.7355714
## 7  0.7939286
## 8  0.6764000
## 9  0.8990000
## 10 0.7939286
## 11 0.7939286
## 12 0.6764000
## 13 0.7939286
## 14 0.6764000
## 15 0.6764000
## 16 0.8990000
## 17 0.7939286
## 18 0.6764000
## 19 0.7939286
```

PERMANOVA for associated odour nmds profiles with genotypes

```r
# combine individuals alleles for each individual to genotype in same dataframe
het_table %<>% mutate(gtype = as.factor(paste0(a1, "/", a2)))

vegan::adonis2(scent ~ het_table$gtype)
```

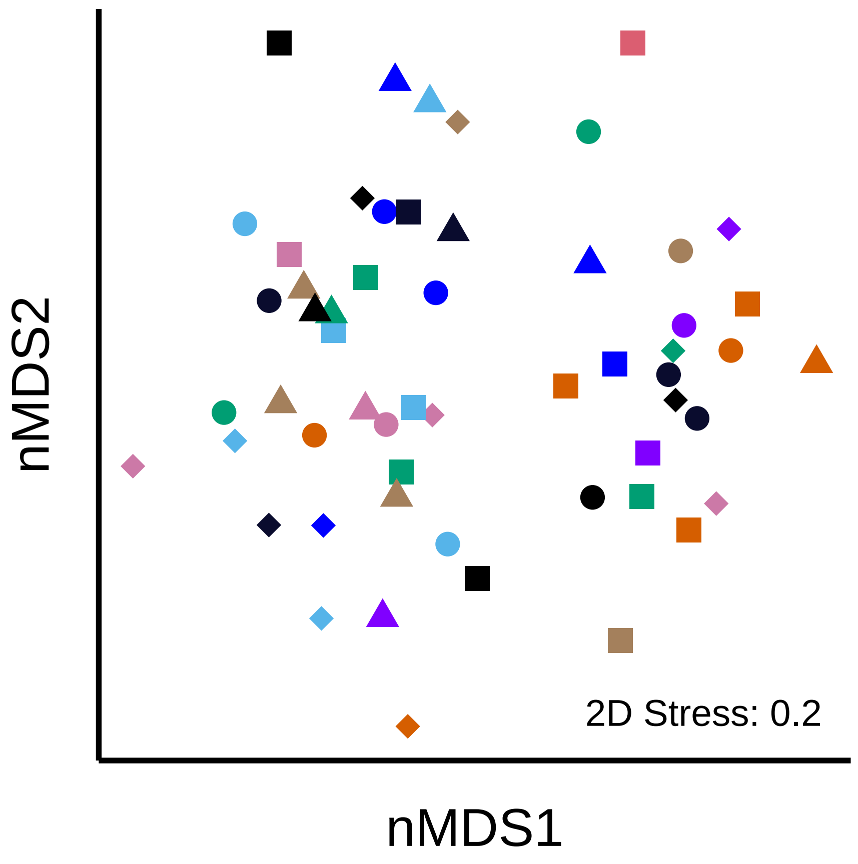
```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## vegan::adonis2(formula = scent ~ het_table$gtype)
##                 Df SumOfSqs      R2      F Pr(>F)
## het_table$gtype 36   8.8036 0.67102 1.0765  0.183
## Residual        19   4.3160 0.32898
## Total           55  13.1197 1.00000
```

```r
scent_nmds %<>% cbind(., gtype = as.factor(het_table$gtype))
```

Plot PERMANOVA results

```r
# create color palette for the plot
clr <- c("#D55E00", "#0000ff", "#56B4E9", "#009E73","#000000", "#CC79A7", "#a4805c",
         "turquoise", "#ed0c2e", "#8000ff", "#ffb700", "#ffff00", "#0a0c2e", "#db5e71")

# assign pch values for plotting
shp <- c(17, 15, 16, 18)

color_shape_pairs <- crossing(clr,shp)

shape_pair_df <- data.frame(fam = levels(scent_nmds$gtype),
                            color_shape_pairs[1:length(levels(scent_nmds$gtype)),])

cross_ref <- match(scent_nmds$gtype, shape_pair_df$fam)
```

```
shape_pair_df %<>% .[cross_ref,]

scent_nmds %<>%  cbind(.,
                     shape_pair_df[,2:3])

scent_nmds %<>% mutate(across(clr:shp, as.factor))

ggplot(data = scent_nmds,aes(MDS1,MDS2, color = clr, shape = shp)) +
  geom_point(size = 4) +
  scale_shape_manual(values = as.numeric(levels(scent_nmds$shp))) +
  theme_void() +
  scale_color_manual(values = levels(as.factor(scent_nmds$clr))) +
  annotate("text", x = 0.48, y = -0.75, label = "2D Stress: 0.2", size = 5) +
  scale_x_continuous(name = "nMDS1") +
  scale_y_continuous(name = "nMDS2") +
  custom_theme +
  theme(
    legend.position = "none",
    axis.ticks = element_blank(),
    axis.text = element_blank()
  )
```



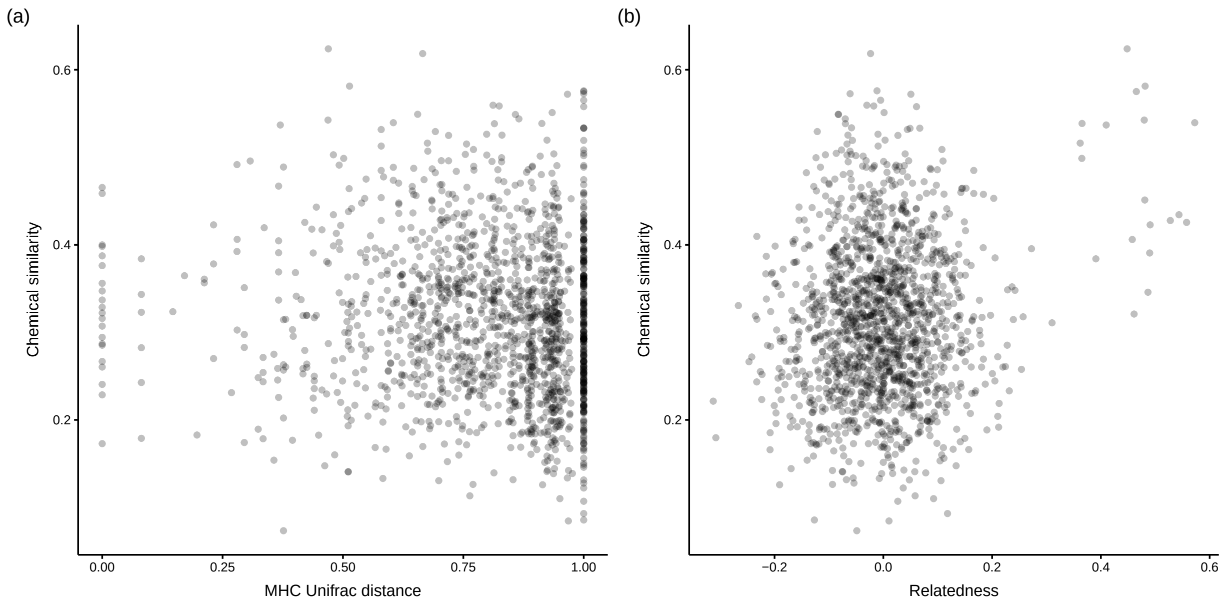
```
# save output
ggsave(filename = "figures/genotype_pairs_nmds.png",
       width = 32, height = 16,
       units = "cm", dpi = 400)
```

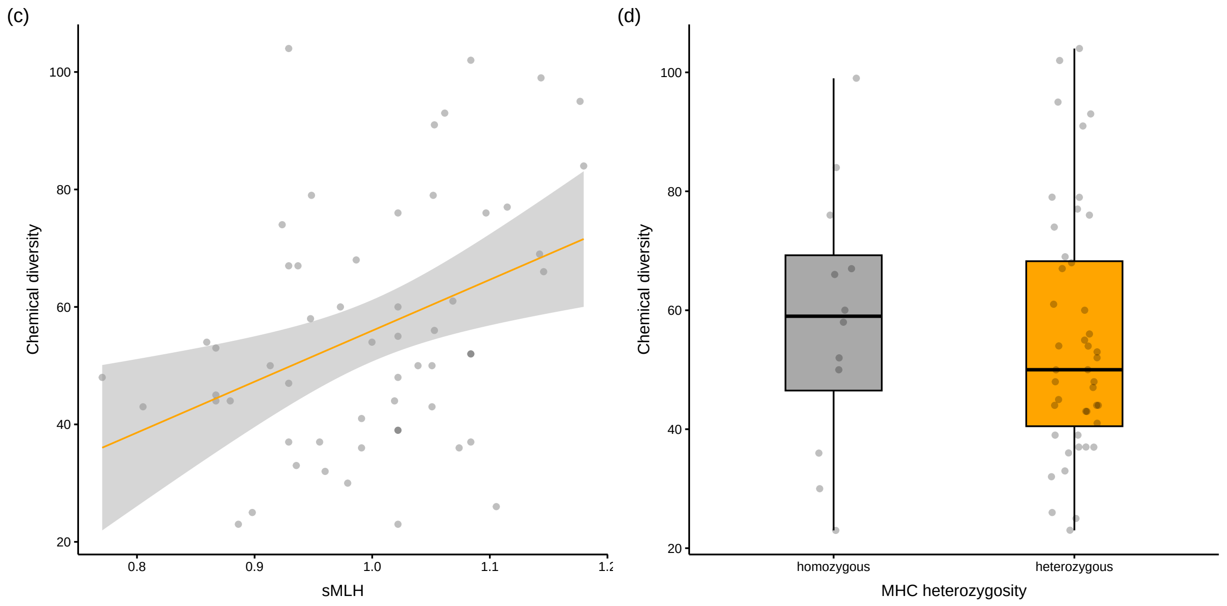
Create manuscript panel figure

```
# tag is according to final manuscript structure
panel1.a <- panel1.a + labs(tag = "(a)")
panel1.b <- panel1.b + labs(tag = "(b)")
panel2.b <- panel2.b + labs(tag = "(c)")
panel2.a <- panel2.a + labs(tag = "(d)")


# arrange figures in 2x2 grid and align horizontally and vertically
panel_final <- ggpubr::ggarrange(panel1.a, panel1.b,
                                 panel2.b, panel2.a,
                                 nrow = 2, ncol = 2, align = "hv")
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

```
# print
panel_final
```


(a)
Chemical similarity
0.6
0.4
0.2
0.00
0.25
0.50
0.75
1.00
MHC Unifrac distance
(b)
Chemical similarity
0.6
0.4
0.2
−0.2
0.0
0.2
0.4
0.6
Relatedness


(c)
Chemical diversity
100
80
60
40
20
0.8
0.9
1.0
1.1
1.2
sMLH
(d)
Chemical diversity
100
80
60
40
20
homozygous
heterozygous
MHC heterozygosity

```r
# save high resolution
ggsave(filename = "figures/final_panel.png",
       panel_final,dpi = 400,
       width = 33.97, height = 31.04,
       units = "cm",
       bg = "white"
       )
```

Session information

```r
sessionInfo()
```

```
## R version 4.2.2 (2022-10-31 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 22621)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=German_Germany.utf8  LC_CTYPE=German_Germany.utf8
## [3] LC_MONETARY=German_Germany.utf8 LC_NUMERIC=C
## [5] LC_TIME=German_Germany.utf8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] remotes_2.4.2      Demerelate_0.9-3   fts_0.9.9.2        zoo_1.8-11
##  [5] MuMIn_1.47.1       performance_0.10.2 ape_5.6-2          ggpubr_0.5.0
##  [9] vegan_2.6-4        lattice_0.20-45    permute_0.9-7      inbreedR_0.3.3
## [13] GCalignR_1.0.5     phyloseq_1.42.0    forcats_0.5.2      stringr_1.5.0
## [17] dplyr_1.0.10       purrr_1.0.1        readr_2.1.3        tidyr_1.2.1
## [21] tibble_3.1.8       ggplot2_3.4.0      tidyverse_1.3.2    magrittr_2.0.3
##
## loaded via a namespace (and not attached):
##   [1] minqa_1.2.5            googledrive_2.0.0      colorspace_2.0-3
##   [4] ggsignif_0.6.4         ellipsis_0.3.2         XVector_0.38.0
##   [7] fs_1.5.2               rstudioapi_0.14        farver_2.1.1
##  [10] fansi_1.0.3            lubridate_1.9.0        xml2_1.3.3
##  [13] codetools_0.2-18       splines_4.2.2          knitr_1.41
##  [16] ade4_1.7-20            Formula_1.2-4          jsonlite_1.8.4
##  [19] nloptr_2.0.3           broom_1.0.2            cluster_2.1.4
##  [22] dbplyr_2.2.1           sfsmisc_1.1-14         compiler_4.2.2
##  [25] httr_1.4.4             backports_1.4.1        assertthat_0.2.1
##  [28] Matrix_1.5-1           fastmap_1.1.0          gargle_1.2.1
##  [31] cli_3.6.0              htmltools_0.5.4        tools_4.2.2
##  [34] lmerTest_3.1-3         igraph_1.3.5           gtable_0.3.1
##  [37] glue_1.6.2             GenomeInfoDbData_1.2.9 reshape2_1.4.4
##  [40] Rcpp_1.0.9             carData_3.0-5          Biobase_2.58.0
##  [43] cellranger_1.1.0       vctrs_0.5.1            Biostrings_2.66.0
##  [46] rhdf5filters_1.10.0    mlogit_1.1-1           multtest_2.54.0
##  [49] nlme_3.1-160           iterators_1.0.14       lmtest_0.9-40
##  [52] insight_0.19.0         xfun_0.36              rbibutils_2.2.13
##  [55] lme4_1.1-31            rvest_1.0.3            timechange_0.2.0
##  [58] lifecycle_1.0.3        statmod_1.5.0          rstatix_0.7.1
##  [61] googlesheets4_1.0.1    zlibbioc_1.44.0        MASS_7.3-58.1
##  [64] scales_1.2.1           ragg_1.2.5             hms_1.1.2
##  [67] parallel_4.2.2         biomformat_1.26.0      rhdf5_2.42.0
##  [70] yaml_2.3.6             stringi_1.7.12         highr_0.10
##  [73] S4Vectors_0.36.1       foreach_1.5.2          BiocGenerics_0.44.0
##  [76] boot_1.3-28            GenomeInfoDb_1.34.6    systemfonts_1.0.4
##  [79] Rdpack_2.4             rlang_1.0.6            pkgconfig_2.0.3
##  [82] bitops_1.0-7           evaluate_0.19          Rhdf5lib_1.20.0
##  [85] labeling_0.4.2         cowplot_1.1.1          dfidx_0.0-5
##  [88] tidyselect_1.2.0       plyr_1.8.8             R6_2.5.1
##  [91] IRanges_2.32.0         generics_0.1.3         DBI_1.1.3
##  [94] pillar_1.8.1           haven_2.5.1            withr_2.5.0
##  [97] mgcv_1.8-41            survival_3.4-0         abind_1.4-5
## [100] RCurl_1.98-1.9         modelr_0.1.10          crayon_1.5.2
## [103] car_3.1-1              utf8_1.2.2             tzdb_0.3.0
## [106] rmarkdown_2.19         grid_4.2.2             readxl_1.4.1
## [109] data.table_1.14.6      reprex_2.0.2           digest_0.6.31
## [112] numDeriv_2016.8-1.1    textshaping_0.3.6      stats4_4.2.2
## [115] munsell_0.5.0
```

References

Armstrong, Whit. 2018. *Fts: R Interface to Tslib (a Time Series Library in c++).* https://CRAN.R-project.org/package=fts.

Bache, Stefan Milton, and Hadley Wickham. 2022. *Magrittr: A Forward-Pipe Operator for r.* https://CRAN.R-project.org/package=magrittr.

Bartoń, Kamil. 2022. *MuMIn: Multi-Model Inference.* https://CRAN.R-project.org/package=MuMIn.

Csárdi, Gábor, Jim Hester, Hadley Wickham, Winston Chang, Martin Morgan, and Dan Tenenbaum. 2021. *Remotes: R Package Installation from Remote Repositories, Including GitHub.* https://CRAN.R-project.org/package=remotes.

Kassambara, Alboukadel. 2022. *Ggpubr: Ggplot2 Based Publication Ready Plots.* https://rpkgs.datanovia.com/ggpubr/.

Kraemer, Philipp, and Gabriele Gerlach. 2017. *Demerelate: Functions to Calculate Relatedness on Diploid Genetic Data.* https://www.r-project.org.

Lüdecke, Daniel, Mattan S. Ben-Shachar, Indrajeet Patil, Philip Waggoner, and Dominique Makowski. 2021. "performance: An R Package for Assessment, Comparison and Testing of Statistical Models." *Journal of Open Source Software* 6 (60): 3139. https://doi.org/10.21105/joss.03139.

Lüdecke, Daniel, Dominique Makowski, Mattan S. Ben-Shachar, Indrajeet Patil, Philip Waggoner, and Brenton M. Wiernik. 2023. *Performance: Assessment of Regression Models Performance.* https://easystats.github.io/performance/.

McMurdie, Paul J., and Susan Holmes. 2013. "Phyloseq: An r Package for Reproducible Interactive Analysis and Graphics of Microbiome Census Data." *PLoS ONE* 8 (4): e61217. http://dx.plos.org/10.1371/journal.pone.0061217.

McMurdie, Paul J., Susan Holmes, with contributions from Gregory Jordan, and Scott Chamberlain. 2022. *Phyloseq: Handling and Analysis of High-Throughput Microbiome Census Data.* http://dx.plos.org/10.1371/journal.pone.0061217.

Müller, Kirill, and Hadley Wickham. 2022. *Tibble: Simple Data Frames.* https://CRAN.R-project.org/package=tibble.

Oksanen, Jari, Gavin L. Simpson, F. Guillaume Blanchet, Roeland Kindt, Pierre Legendre, Peter R. Minchin, R. B. O'Hara, et al. 2022. *Vegan: Community Ecology Package.* https://github.com/vegandevs/vegan.

Ottensmann, Meinolf, Martin A. Stoffel, Hazel J. Nichols, and Joseph I. Hoffman. 2018. "GCalignR: An r Package for Aligning Gas-Chromatography Data for Ecological and Evolutionary Studies." *PLOS ONE*. https://doi.org/10.1371/journal.pone.0198311.

Ottensmann, Meinolf, Martin Stoffel, Hazel J. Nichols, and Joseph I. Hoffman. 2023. *GCalignR: Simple Peak Alignment for Gas-Chromatography Data*. https://github.com/mottensmann/GCalignR.

Paradis, Emmanuel, Simon Blomberg, Ben Bolker, Joseph Brown, Santiago Claramunt, Julien Claude, Hoa Sien Cuong, et al. 2022. *Ape: Analyses of Phylogenetics and Evolution*. http://ape-package.ird.fr/.

Paradis, E., and K. Schliep. 2019. "Ape 5.0: An Environment for Modern Phylogenetics and Evolutionary Analyses in R." *Bioinformatics* 35: 526–28.

R Core Team. 2022. *R: A Language and Environment for Statistical Computing*. Vienna, Austria: R Foundation for Statistical Computing. https://www.R-project.org/.

Sarkar, Deepayan. 2008. *Lattice: Multivariate Data Visualization with r*. New York: Springer. http://lmdvr.r-forge.r-project.org.

———. 2021. *Lattice: Trellis Graphics for r*. http://lattice.r-forge.r-project.org/.

Simpson, Gavin L. 2022. *Permute: Functions for Generating Restricted Permutations of Data*. https://github.com/gavinsimpson/permute.

Stoffel, Martin A., Mareike Esser, Joseph Hoffman, and Marty Kardos. 2022. *inbreedR: Analysing Inbreeding Based on Genetic Markers*. https://CRAN.R-project.org/package=inbreedR.

Stoffel, Martin A., Mareike Esser, Marty Kardos, Emily Humble, Hazel Nichols, Patrice David, and Joseph I. Hoffman. 2016. "inbreedR: An r Package for the Analysis of Inbreeding Based on Genetic Markers." *Methods in Ecology and Evolution*. https://doi.org/10.1111/2041-210X.12588.

Wickham, Hadley. 2016. *Ggplot2: Elegant Graphics for Data Analysis*. Springer-Verlag New York. https://ggplot2.tidyverse.org.

———. 2022a. *Forcats: Tools for Working with Categorical Variables (Factors)*. https://CRAN.R-project.org/package=forcats.

———. 2022b. *Stringr: Simple, Consistent Wrappers for Common String Operations*. https://CRAN.R-project.org/package=stringr.

———. 2022c. *Tidyverse: Easily Install and Load the Tidyverse*. https://CRAN.R-project.org/package=tidyverse.

Wickham, Hadley, Mara Averick, Jennifer Bryan, Winston Chang, Lucy D'Agostino McGowan, Romain François, Garrett Grolemund, et al. 2019. "Welcome to the tidyverse." *Journal of Open Source Software* 4 (43): 1686. https://doi.org/10.21105/joss.01686.

Wickham, Hadley, Winston Chang, Lionel Henry, Thomas Lin Pedersen, Kohske Takahashi, Claus Wilke, Kara Woo, Hiroaki Yutani, and Dewey Dunnington. 2022. *Ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. https://CRAN.R-project.org/package=ggplot2.

Wickham, Hadley, Romain François, Lionel Henry, and Kirill Müller. 2022. *Dplyr: A Grammar of Data Manipulation*. https://CRAN.R-project.org/package=dplyr.

Wickham, Hadley, and Maximilian Girlich. 2022. *Tidyr: Tidy Messy Data*. https://CRAN.R-project.org/package=tidyr.

Wickham, Hadley, and Lionel Henry. 2023. *Purrr: Functional Programming Tools*. https://CRAN.R-project.org/package=purrr.

Wickham, Hadley, Jim Hester, and Jennifer Bryan. 2022. *Readr: Read Rectangular Text Data*. https://CRAN.R-project.org/package=readr.

Zeileis, Achim, and Gabor Grothendieck. 2005. "Zoo: S3 Infrastructure for Regular and Irregular Time Series." *Journal of Statistical Software* 14 (6): 1–27. https://doi.org/10.18637/jss.v014.i06.

Zeileis, Achim, Gabor Grothendieck, and Jeffrey A. Ryan. 2022. *Zoo: S3 Infrastructure for Regular and Irregular Time Series (z's Ordered Observations)*. https://zoo.R-Forge.R-project.org/.