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

J. Tebbe

09/11/23

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
```

```r
}

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
}

if (!require("partR2", quietly = TRUE)) {
  install.packages("partR2")
  library(partR2)
} else {
  library(partR2) # tools for models
}

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

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


# archived package as is dependent 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.

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

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

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

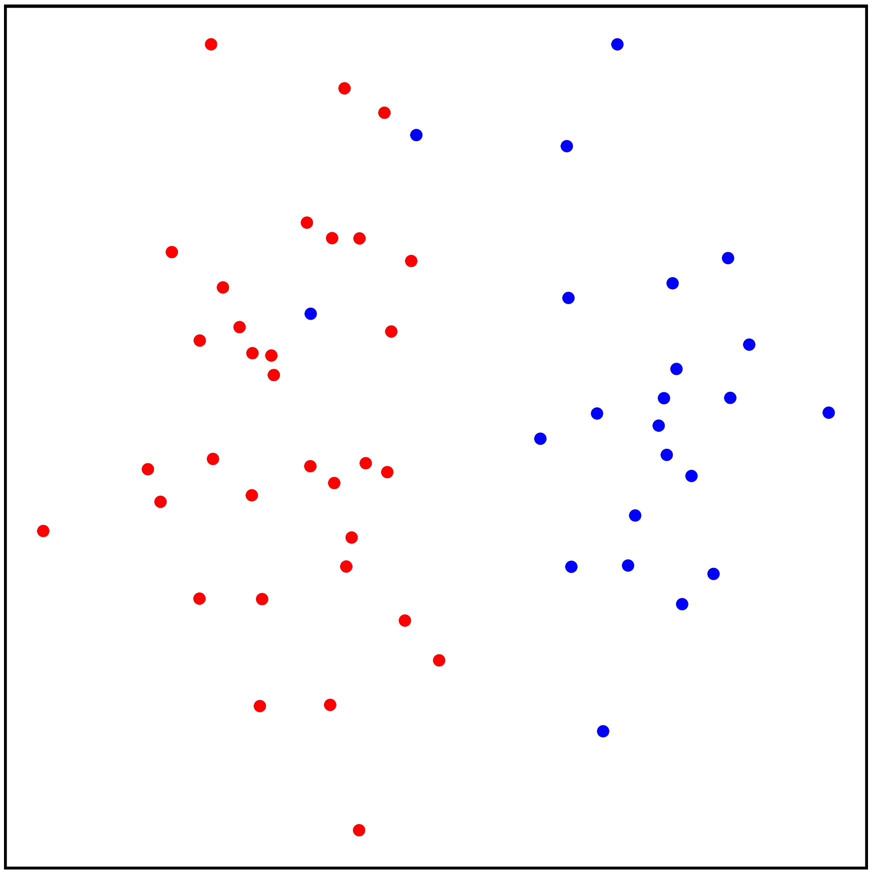
```r
options(digits = 12)
```

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")
```

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

```r
## 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",
                                        linewidth = 1,
                                        fill   = NA),
        aspect.ratio    =  1,
        legend.position = "none")
scent_plot
```



Calculate MHC heterozygosity relatedness between individuals

```r
## 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,]

## 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 mark
##   Especially, rxy and ritland estimator are not defined when bi-allelic estimates are used with alle
##   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 oc
```

```
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

```
# 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

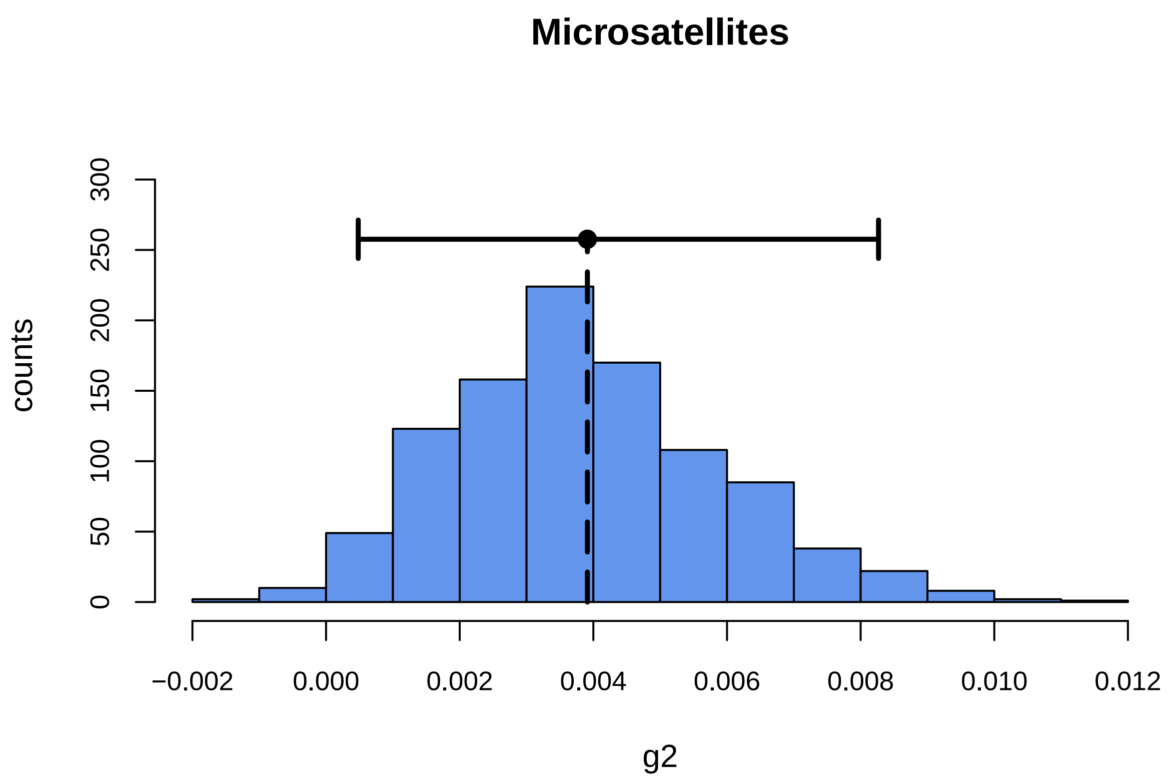
```
# handle genotypes as otu table
phylo_mat <- read.table("data/phyloseq-mat.txt") %>%
  as.matrix()

# 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)
```

```
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)
```

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
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

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

Transform model variables

```r
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()

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 + agePaired + (1|familyBool) +
                        (1|pairID1) + (1|pairID2),
                      data = model_rel.df)
# relatedness
a2 <- lmerTest::lmer(scent_mds ~ rel + colonyBool + agePaired + (1|familyBool) + (1|pairID1) +
                        (1|pairID2),
                      data = model_rel.df)
# mhc & relatedness
a3 <- lmerTest::lmer(scent_mds ~ rel + ufrac + colonyBool + agePaired + (1|familyBool) +
                        (1|pairID1) + (1|pairID2),
                      data = model_rel.df)
# no genetic effect
a4 <- lmerTest::lmer(scent_mds ~ colonyBool + agePaired + (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 weights
## ---------------------------------------------------------------------------------------------------------
## a1   | lmerModLmerTest |      0.696 |      0.140 | 0.646 | 0.060 | 0.062 |       0.198 |        0.19
## a2   | lmerModLmerTest |      0.690 |      0.143 | 0.638 | 0.060 | 0.062 |       0.212 |        0.21
## a3   | lmerModLmerTest |      0.689 |      0.144 | 0.636 | 0.060 | 0.062 |       0.082 |        0.08
## a4   | lmerModLmerTest |      0.698 |      0.139 | 0.649 | 0.060 | 0.062 |       0.508 |        0.51
```

```r
summary(a2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_mds ~ rel + colonyBool + agePaired + (1 | familyBool) +
##     (1 | pairID1) + (1 | pairID2)
##    Data: model_rel.df
##
## REML criterion at convergence: -3928.2
##
## Scaled residuals:
##          Min           1Q       Median           3Q          Max
## -3.108444259 -0.690869875 -0.068233408  0.583996354  3.762633304
##
## Random effects:
##  Groups     Name        Variance      Std.Dev.
##  pairID2    (Intercept) 0.00136165106 0.0369005564
##  pairID1    (Intercept) 0.00132283514 0.0363708006
##  familyBool (Intercept) 0.00404648919 0.0636120208
##  Residual               0.00381948877 0.0618020126
## Number of obs: 1540, groups:  pairID2, 55; pairID1, 55; familyBool, 2
##
## Fixed effects:
##                   Estimate     Std. Error             df  t value Pr(>|t|)
## (Intercept)  3.15186662e-01 4.73683692e-02 1.14156516e+00  6.65395 0.074931 .
## rel          8.71026170e-03 2.08593513e-02 1.33740535e+03  0.41757 0.676328
## colonyBool1  8.37421175e-02 8.06181663e-03 1.20111837e+02 10.38750  < 2e-16 ***
## agePairedM/P 2.39466928e-03 8.06276850e-03 1.28046621e+02  0.29700 0.766945
## agePairedP/P 1.21988991e-02 1.53157945e-02 1.03077423e+02  0.79649 0.427577
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) rel    clnyB1 agPM/P
## rel         -0.090
## colonyBool1 -0.152  0.012
## agePairdM/P -0.145 -0.016 -0.023
## agePairdP/P -0.137 -0.004 -0.026  0.915
```

```r
summary(a4)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_mds ~ colonyBool + agePaired + (1 | familyBool) + (1 |
##     pairID1) + (1 | pairID2)
##    Data: model_rel.df
```

```
##
## REML criterion at convergence: -3933.9
##
## Scaled residuals:
##          Min           1Q       Median           3Q          Max
## -3.113665362 -0.687292046 -0.067704903  0.586076727  3.745905490
##
## Random effects:
##  Groups     Name        Variance      Std.Dev.
##  pairID2    (Intercept) 0.00136551721 0.0369529053
##  pairID1    (Intercept) 0.00132698089 0.0364277489
##  familyBool (Intercept) 0.00435745709 0.0660110377
##  Residual               0.00381639653 0.0617769903
## Number of obs: 1540, groups:  pairID2, 55; pairID1, 55; familyBool, 2
##
## Fixed effects:
##                   Estimate     Std. Error             df  t value Pr(>|t|)
## (Intercept)  3.17056257e-01 4.87992888e-02 1.13632796e+00  6.49715 0.077614 .
## colonyBool1  8.36965038e-02 8.07029419e-03 1.20281473e+02 10.37094  < 2e-16 ***
## agePairedM/P 2.44412815e-03 8.07041285e-03 1.28202135e+02  0.30285 0.762495
## agePairedP/P 1.22231134e-02 1.53347232e-02 1.03314754e+02  0.79709 0.427228
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) clnyB1 agPM/P
## colonyBool1 -0.147
## agePairdM/P -0.142 -0.023
## agePairdP/P -0.134 -0.026  0.915
```

```
# 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
## -4.441714289 -0.385476875  0.272592692  0.681993885  1.501197368
##
## Random effects:
##  Groups   Name        Variance      Std.Dev.
##  pairID1  (Intercept) 0.00216331204 0.0465114183
##  pairID2  (Intercept) 0.00101136839 0.0318020186
##  Residual             0.03589648773 0.1894636844
## Number of obs: 1540, groups:  pairID1, 55; pairID2, 55
##
## Fixed effects:
##                   Estimate      Std. Error              df  t value   Pr(>|t|)
## (Intercept)  7.97297459e-01  9.69913681e-03  6.44203957e+01 82.20293 < 2.22e-16
## rel         -2.38911481e-01  5.12089926e-02  1.42412461e+03 -4.66542 3.3692e-06
##
## (Intercept) ***
## rel         ***
## ---
## 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
## -4.473364209 -0.379228923  0.272204191  0.676906270  2.150144511
##
## Random effects:
##  Groups     Name        Variance      Std.Dev.
##  pairID1    (Intercept) 0.00215194602 0.0463890722
##  pairID2    (Intercept) 0.00107608003 0.0328036587
##  familyBool (Intercept) 0.01743074553 0.1320255488
##  Residual               0.03554491277 0.1885335853
## Number of obs: 1540, groups:  pairID1, 55; pairID2, 55; familyBool, 2
##
## Fixed effects:
##                  Estimate     Std. Error             df  t value Pr(>|t|)
## (Intercept)   0.7107349721   0.0967753027   0.9921110956  7.34418 0.087397 .
## rel          -0.1266874397   0.0594151597 922.8588459187 -2.13224 0.033251 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##     (Intr)
## rel -0.126
```

```r
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 w
## ----------------------------------------------------------------------------------------------------------
## 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 |
```

```r
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
## -4.473094103 -0.378995809  0.272905728  0.676506794  2.148306764
##
## Random effects:
##  Groups     Name        Variance      Std.Dev.
##  pairID1    (Intercept) 0.00217394338 0.0466255657
##  pairID2    (Intercept) 0.00108902249 0.0330003408
##  familyBool (Intercept) 0.01740183944 0.1319160318
##  Residual               0.03555503513 0.1885604283
## Number of obs: 1540, groups:  pairID1, 55; pairID2, 55; familyBool, 2
##
## Fixed effects:
##                   Estimate      Std. Error              df  t value Pr(>|t|)
## (Intercept)  7.11336570e-01  9.72959228e-02  1.01443951e+00  7.31106 0.084313 .
## rel         -1.26595370e-01  5.94527269e-02  9.16826722e+02 -2.12935 0.033492 *
## colonyBool1 -7.42382735e-04  1.31644657e-02  2.09024905e+02 -0.05639 0.955083
## ---
## 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
```

```r
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 |
```

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

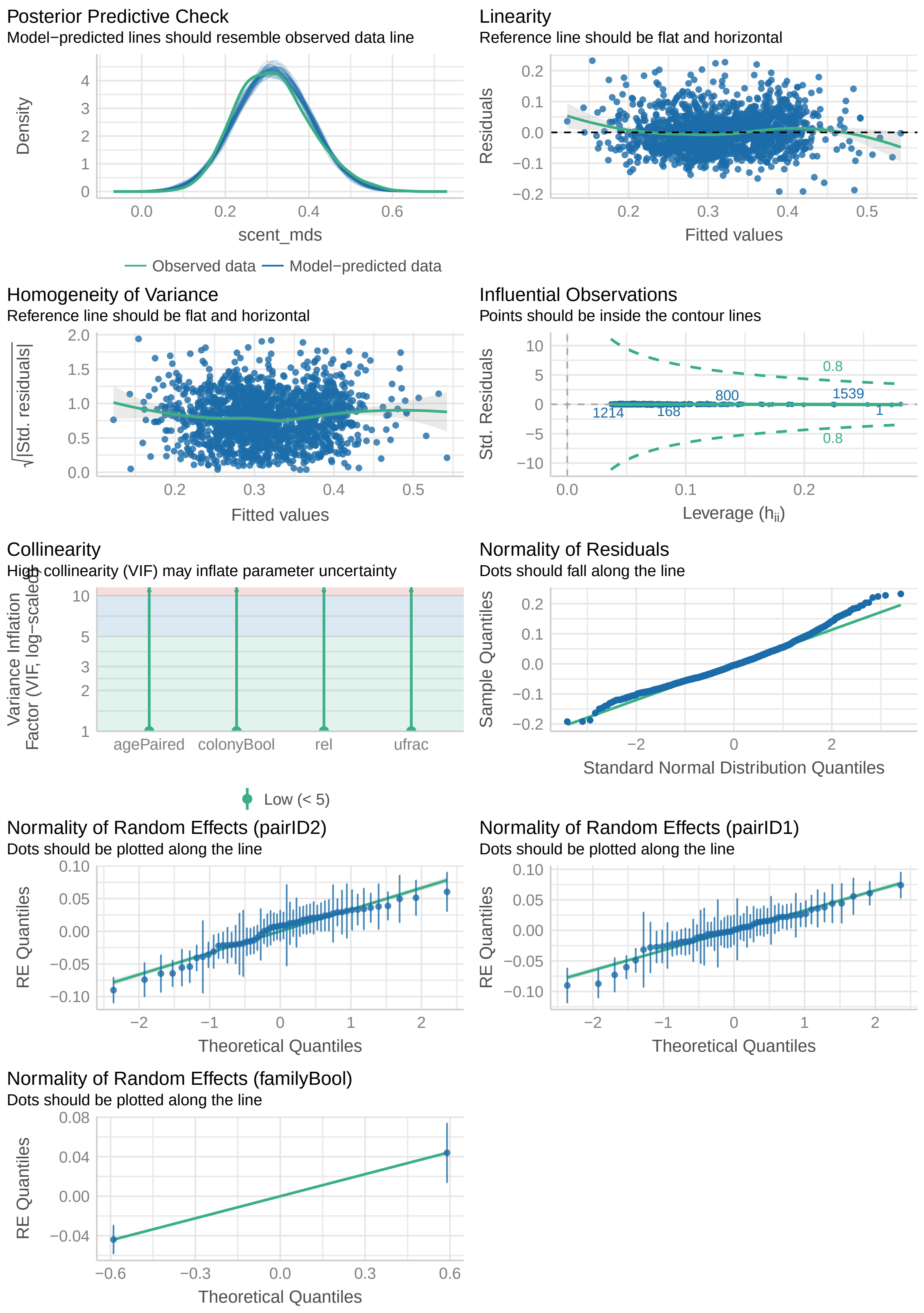
```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ufrac ~ rel + (1 | pairID1) + (1 | pairID2) + (1 | familyBool)
##    Data: model_rel.df
##
## REML criterion at convergence: -674.6
##
## Scaled residuals:
##         Min          1Q      Median          3Q         Max
## -4.473364209 -0.379228923  0.272204191  0.676906270  2.150144511
##
## Random effects:
##  Groups     Name        Variance      Std.Dev.
##  pairID1    (Intercept) 0.00215194602 0.0463890722
##  pairID2    (Intercept) 0.00107608003 0.0328036587
##  familyBool (Intercept) 0.01743074553 0.1320255488
##  Residual               0.03554491277 0.1885335853
## Number of obs: 1540, groups:  pairID1, 55; pairID2, 55; familyBool, 2
##
## Fixed effects:
##                  Estimate     Std. Error            df  t value Pr(>|t|)
## (Intercept)   0.7107349721   0.0967753027   0.9921110956  7.34418 0.087397 .
## rel          -0.1266874397   0.0594151597 922.8588459187 -2.13224 0.033251 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##     (Intr)
## rel -0.126
```

```
(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 0.1616032287 0.1616032287     1 922.8588459 4.54645 0.033251 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Adressing collinearity

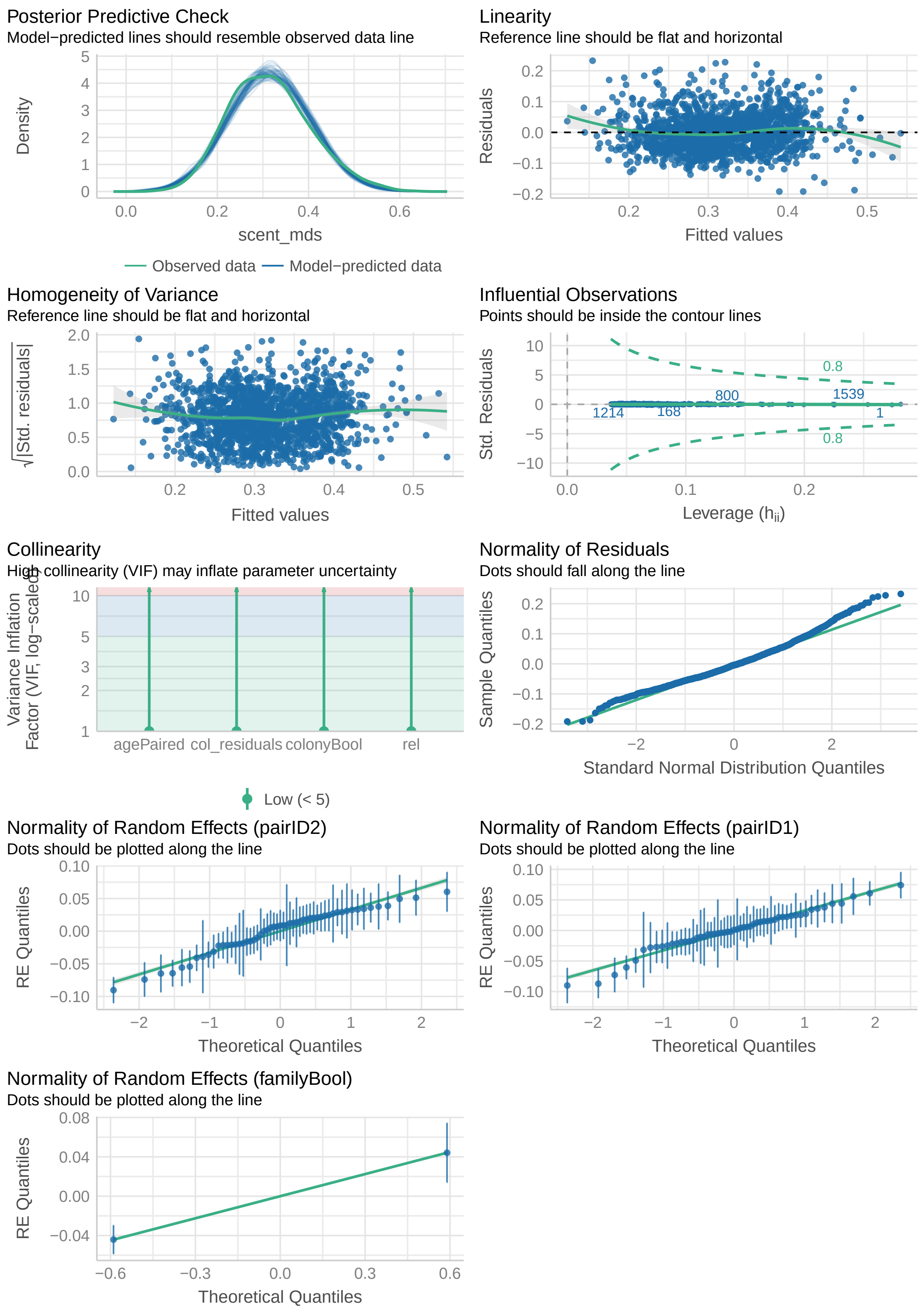
```
# use performance tools to check model goodness

# model with all genetic effects that were
# evaluated in chemical similarity associations:
check_model(a3)
```


Posterior Predictive Check
Model−predicted lines should resemble observed data line
Density
scent_mds
Observed data
Model−predicted data
Linearity
Reference line should be flat and horizontal
Residuals
Fitted values
Homogeneity of Variance
Reference line should be flat and horizontal
√|Std. residuals|
Fitted values
Influential Observations
Points should be inside the contour lines
Std. Residuals
Leverage (h_{ii})
0.8
1214
168
800
1539
1
Collinearity
High collinearity (VIF) may inflate parameter uncertainty
Variance Inflation Factor (VIF, log−scaled)
agePaired
colonyBool
rel
ufrac
Low (< 5)
Normality of Residuals
Dots should fall along the line
Sample Quantiles
Standard Normal Distribution Quantiles
Normality of Random Effects (pairID2)
Dots should be plotted along the line
RE Quantiles
Theoretical Quantiles
Normality of Random Effects (pairID1)
Dots should be plotted along the line
RE Quantiles
Theoretical Quantiles
Normality of Random Effects (familyBool)
Dots should be plotted along the line
RE Quantiles
Theoretical Quantiles

```
# check by re-running a3 model with residuals for relatedness effect
res_u_r <- residuals(u_r_model2)

temp_df <- cbind(model_rel.df, col_residuals = res_u_r)

a3_residual <- lmerTest::lmer(scent_mds ~ col_residuals + rel + colonyBool + agePaired +
                                (1|familyBool) + (1|pairID1) + (1|pairID2),
                              data = temp_df)

summary(a3_residual)
```

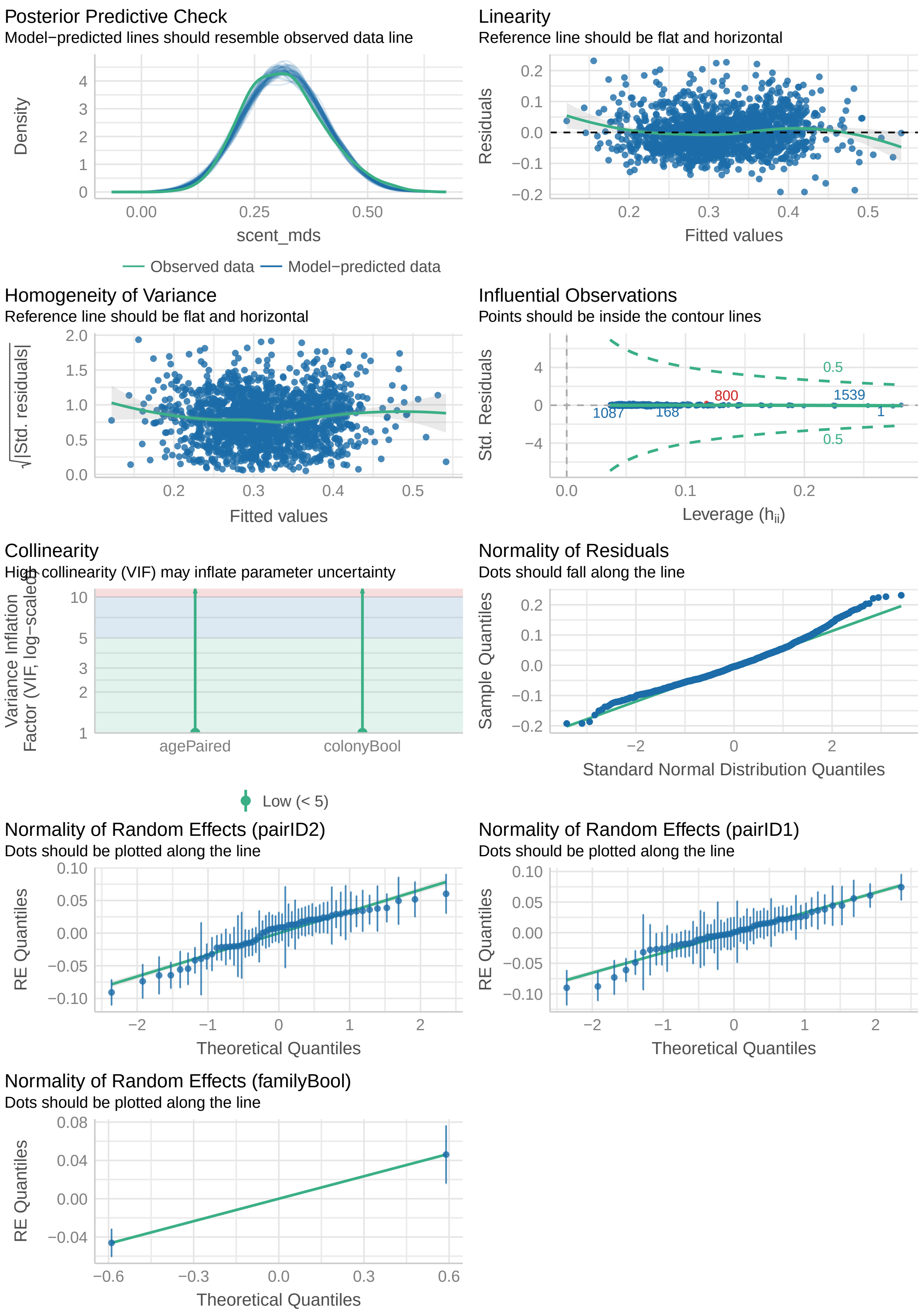
```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_mds ~ col_residuals + rel + colonyBool + agePaired + (1 |
##     familyBool) + (1 | pairID1) + (1 | pairID2)
##    Data: temp_df
##
## REML criterion at convergence: -3920.6
##
## Scaled residuals:
##          Min           1Q       Median           3Q          Max
## -3.103258827 -0.690769473 -0.068686908  0.585558921  3.761156979
##
## Random effects:
##  Groups     Name        Variance      Std.Dev.
##  pairID2    (Intercept) 0.00136213257 0.0369070802
##  pairID1    (Intercept) 0.00132201584 0.0363595358
##  familyBool (Intercept) 0.00404272164 0.0635824004
##  Residual               0.00382206362 0.0618228406
## Number of obs: 1540, groups:  pairID2, 55; pairID1, 55; familyBool, 2
##
## Fixed effects:
##                     Estimate      Std. Error              df  t value Pr(>|t|)
## (Intercept)    3.15169410e-01  4.73489770e-02  1.14124101e+00  6.65631 0.074941
## col_residuals -1.72524676e-03  8.62343567e-03  1.43952773e+03 -0.20006 0.841458
## rel            8.77922600e-03  2.08681133e-02  1.33588284e+03  0.42070 0.674041
## colonyBool1    8.37532200e-02  8.06204794e-03  1.20109868e+02 10.38858  < 2e-16
## agePairedM/P   2.38288843e-03  8.06317648e-03  1.28076263e+02  0.29553 0.768070
## agePairedP/P   1.21929643e-02  1.53155887e-02  1.03075963e+02  0.79611 0.427795
##
## (Intercept)   .
## col_residuals
## rel
## colonyBool1   ***
## agePairedM/P
## agePairedP/P
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) cl_rsd rel    clnyB1 agPM/P
## col_residls  0.002
## rel         -0.090 -0.014
## colonyBool1 -0.152 -0.008  0.012
## agePairdM/P -0.145  0.008 -0.016 -0.023
## agePairdP/P -0.137  0.002 -0.004 -0.026  0.915
```

```
check_model(a3_residual)
```


Posterior Predictive Check
Model−predicted lines should resemble observed data line
Density
scent_mds
Observed data
Model−predicted data
Linearity
Reference line should be flat and horizontal
Residuals
Fitted values
Homogeneity of Variance
Reference line should be flat and horizontal
√|Std. residuals|
Fitted values
Influential Observations
Points should be inside the contour lines
Std. Residuals
Leverage (hii)
0.8
800
1539
1214
168
1
Collinearity
High collinearity (VIF) may inflate parameter uncertainty
Variance Inflation Factor (VIF, log−scaled)
agePaired
col_residuals
colonyBool
rel
Low (< 5)
Normality of Residuals
Dots should fall along the line
Sample Quantiles
Standard Normal Distribution Quantiles
Normality of Random Effects (pairID2)
Dots should be plotted along the line
RE Quantiles
Theoretical Quantiles
Normality of Random Effects (pairID1)
Dots should be plotted along the line
RE Quantiles
Theoretical Quantiles
Normality of Random Effects (familyBool)
Dots should be plotted along the line
RE Quantiles
Theoretical Quantiles

Neither model, performs noticeably better, and by VIC both models (with and without residuals fitted) show no high amount of collinearity. However, ‘a4’ is within this same goodness and while retaining the same information with less fixed effects, thus being more parsimonious.

```r
check_model(a4)
```


Posterior Predictive Check
Model−predicted lines should resemble observed data line
Density
scent_mds
Observed data
Model−predicted data
Linearity
Reference line should be flat and horizontal
Residuals
Fitted values
Homogeneity of Variance
Reference line should be flat and horizontal
√|Std. residuals|
Fitted values
Influential Observations
Points should be inside the contour lines
Std. Residuals
Leverage (hii)
0.5
800
1539
1087
168
1
Collinearity
High collinearity (VIF) may inflate parameter uncertainty
Variance Inflation Factor (VIF, log−scaled)
agePaired
colonyBool
Low (< 5)
Normality of Residuals
Dots should fall along the line
Sample Quantiles
Standard Normal Distribution Quantiles
Normality of Random Effects (pairID2)
Dots should be plotted along the line
RE Quantiles
Theoretical Quantiles
Normality of Random Effects (pairID1)
Dots should be plotted along the line
RE Quantiles
Theoretical Quantiles
Normality of Random Effects (familyBool)
Dots should be plotted along the line
RE Quantiles
Theoretical Quantiles

```r
# 1540 samples,
# number of coefficients = 4,
# coefficients minus Intercept = 3
pwr_a <- pwr.f2.test(v = 1540, u = 3, f2 = 0.1439/(1-0.1439), sig.level = 0.05)
pwr_a
```

Power analysis

```
##
##      Multiple regression power calculation
##
##               u = 3
##               v = 1540
##              f2 = 0.168087840206
##       sig.level = 0.05
##           power = 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
```

```r
# 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)
)
```

General meta data information

```
# 32 adult female individuals
meta_mom <- meta %>% filter(maturity == "M")

# check n for each beach colony
n_fwb_mom <- meta_mom %>% filter(colony == "FWB") %>% dim() %>% .[1]
n_ssb_mom <- meta_mom %>% filter(colony == "SSB") %>% dim() %>% .[1]

# test for difference in number of chemicals by beach
chems_fwb_mom <- meta_mom %>% filter(., colony == "FWB") %>%
  select(compound_n) %>%
  unlist()

chems_ssb_mom <- meta_mom %>% filter(., colony == "SSB") %>%
  select(compound_n) %>%
  unlist()

(colony_t_test_mom <- t.test(chems_fwb_mom, chems_ssb_mom))
```

```
##
##  Welch Two Sample t-test
##
## data:  chems_fwb_mom and chems_ssb_mom
## t = -1.522037389, df = 22.29327201, p-value = 0.142058931
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -29.16483741389   4.46483741389
## sample estimates:
## mean of x mean of y
##     52.15     64.50
```

```
# 24 pups (mixed sexes)
meta_pup <- meta %>% filter(maturity == "P")

# check n for each beach colony
n_fwb_pup <- meta_pup %>% filter(colony == "FWB") %>% dim() %>% .[1]
n_ssb_pup <- meta_pup %>% filter(colony == "SSB") %>% dim() %>% .[1]

# test for difference in number of chemicals by beach
chems_fwb_pup <- meta_pup %>% filter(., colony == "FWB") %>%
  select(compound_n) %>%
  unlist()

chems_ssb_pup <- meta_pup %>% filter(., colony == "SSB") %>%
  select(compound_n) %>%
  unlist()

(colony_t_test_pup <- t.test(chems_fwb_pup, chems_ssb_pup))
```

```
##
##  Welch Two Sample t-test
##
## data:  chems_fwb_pup and chems_ssb_pup
## t = 1.175783211, df = 20.29290702, p-value = 0.253292061
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -6.90357099378 24.77769686791
## sample estimates:
##     mean of x     mean of y
## 58.8461538462 49.9090909091
```

```
# standard error function
se <- function(x) sd(x)/sqrt(length(x))

mean_chems_by_colony <- list(chems_fwb_mom, chems_fwb_pup, chems_ssb_mom, chems_ssb_pup)
sapply(mean_chems_by_colony, mean)
```

```
## [1] 52.1500000000 58.8461538462 64.5000000000 49.9090909091
```

```
sapply(mean_chems_by_colony, se)
```

```
## [1] 4.80037005153 6.31324021184 6.54182348779 4.23288971884
```

```
mean(meta$compound_n)
```

```
## [1] 55.9107142857
```

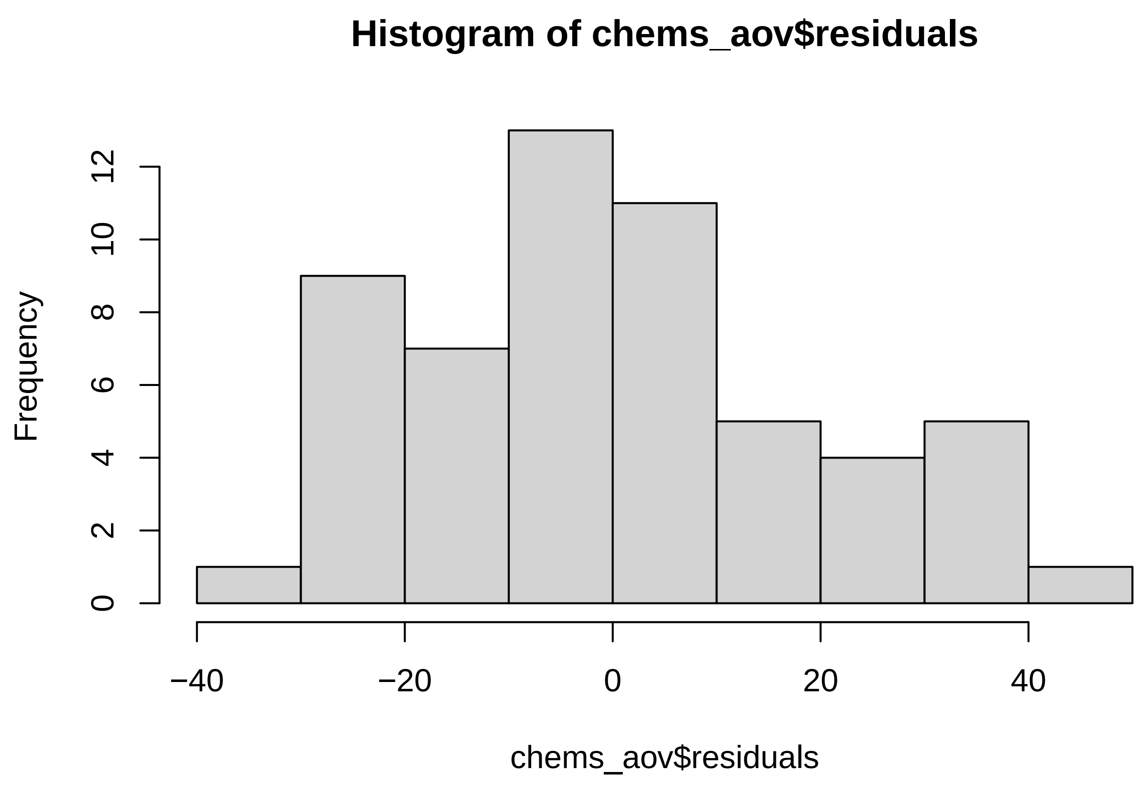
```
se(meta$compound_n)
```

```
## [1] 2.80714265061
```

```
# neither in pups nor moms compound_n differs for colony, calculate anova

# check whether compound number varies for different ages and different colonies

chems_aov <- aov(compound_n ~ maturity * colony, data = meta)
summary(chems_aov)
```

```
##                  Df      Sum Sq     Mean Sq F value  Pr(>F)  
## maturity          1    56.58482   56.584821 0.13023 0.71966  
## colony            1   115.19930  115.199303 0.26513 0.60880  
## maturity:colony   1  1504.61805 1504.618048 3.46285 0.06842 .
## Residuals        52 22594.15140  434.502912                  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
hist(chems_aov$residuals)
```



```r
# post-hoc analysis
TukeyHSD(chems_aov)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = compound_n ~ maturity * colony, data = meta)
##
## $maturity
##         diff           lwr           upr        p adj
## P-M -2.03125 -13.326105701 9.26360570104 0.7196568197
##
## $colony
##                  diff            lwr           upr          p adj
## SSB-FWB 2.90513833992 -8.45648556259 14.2667622424 0.610058289387
##
## $`maturity:colony`
##                     diff             lwr            upr          p adj
## P:FWB-M:FWB   6.69615384615 -13.01371483797 26.40602253028 0.803961243091
## M:SSB-M:FWB  12.35000000000  -7.85146210548 32.55146210548 0.375145035385
## P:SSB-M:FWB  -2.24090909091 -23.00834798997 18.52652980816 0.991714931867
## M:SSB-P:FWB   5.65384615385 -16.49347177283 27.80116408052 0.905141795957
## P:SSB-P:FWB  -8.93706293706 -31.60181836421 13.72769249008 0.723034927926
## P:SSB-M:SSB -14.59090909091 -37.68444251969  8.50262433787 0.346056221246
```

```r
#### Permanova
```

```r
perm_chem_data <- vegan::adonis2(
  # term
  scent ~ colony + family + smlh + maturity,
  by = "terms",
  # data for grouping
  data = meta)

perm_chem_data
```

```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## vegan::adonis2(formula = scent ~ colony + family + smlh + maturity, data = meta, by = "terms")
##          Df     SumOfSqs           R2        F Pr(>F)
## colony    1  1.671427973 0.1273987963 11.47634  0.001 ***
## family   34  8.546158183 0.6514012466  1.72587  0.001 ***
## smlh      1  0.120713508 0.0092009682  0.82884  0.679
## maturity  1  0.159812162 0.0121811274  1.09730  0.306
## Residual 18  2.621540965 0.1998178615
## Total    55 13.119652790 1.0000000000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

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.

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

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

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

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


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

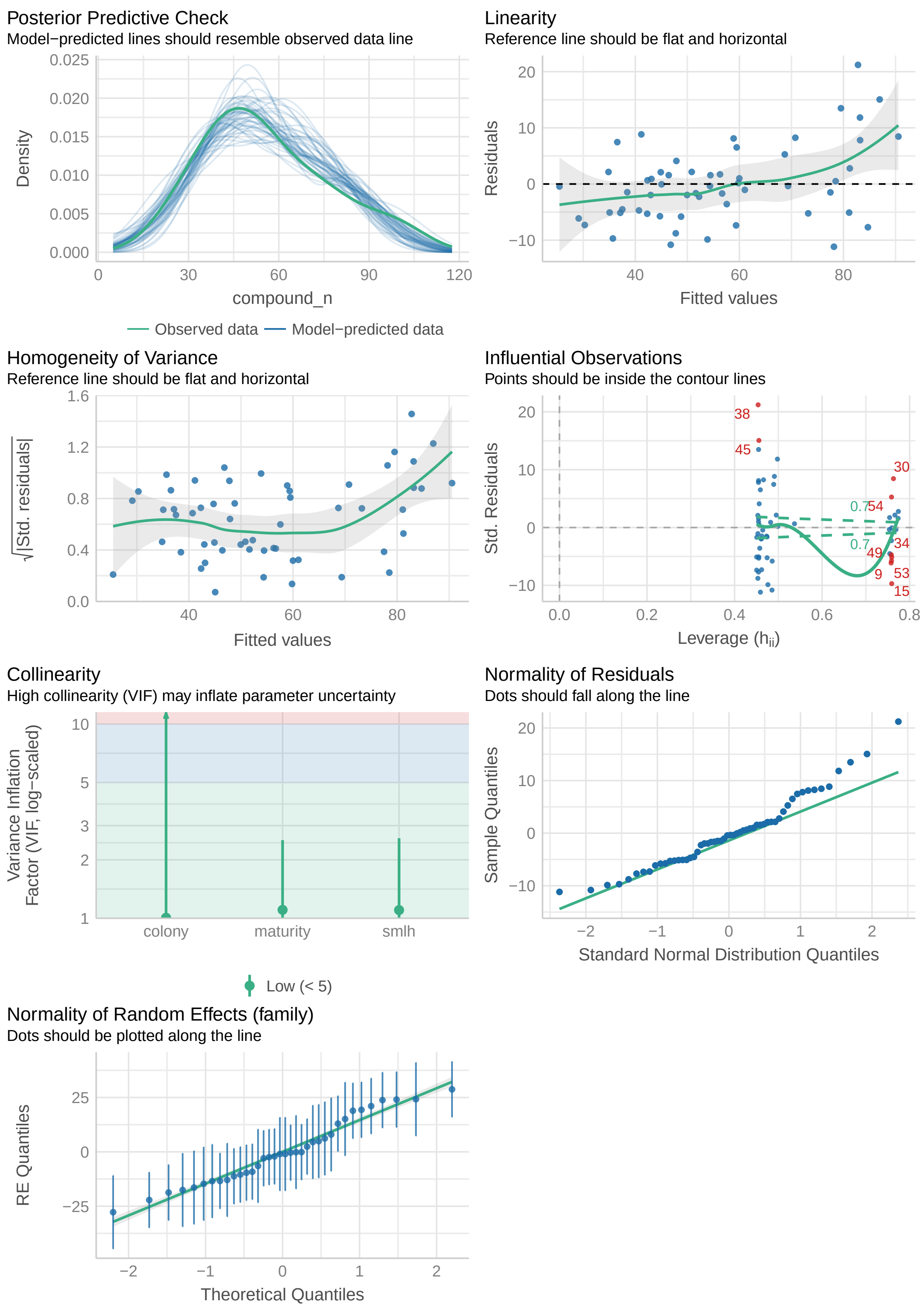
```
## # Comparison of Model Performance Indices
##
## Name |           Model | R2 (cond.) | R2 (marg.) |   ICC |  RMSE |  Sigma | AIC weights | AICc weigh
## ----------------------------------------------------------------------------------------------------
## b1   | lmerModLmerTest |      0.747 |      0.006 | 0.746 | 7.248 | 10.847 |       0.005 |         0.00
## b2   | lmerModLmerTest |      0.772 |      0.119 | 0.741 | 6.696 |  9.994 |       0.713 |         0.70
## b3   | lmerModLmerTest |      0.769 |      0.117 | 0.738 | 6.706 | 10.127 |       0.269 |         0.2
## b4   | lmerModLmerTest |      0.752 |      0.006 | 0.751 | 7.202 | 10.670 |       0.014 |         0.0
```

```r
summary(b2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: compound_n ~ smlh + maturity + colony + (1 | family)
##    Data: meta
##
## REML criterion at convergence: 450.6
##
## Scaled residuals:
##           Min            1Q        Median            3Q           Max
## -1.1174803761 -0.5102355060 -0.0397205379  0.2311200062  2.1221413149
##
## Random effects:
##  Groups   Name        Variance    Std.Dev.
##  family   (Intercept) 285.8932945 16.90837942
##  Residual              99.8834741  9.99417201
## Number of obs: 56, groups:  family, 36
##
## Fixed effects:
##                Estimate   Std. Error           df  t value  Pr(>|t|)
## (Intercept) -21.15641099  24.20114640  44.84179770 -0.87419 0.3866725
## smlh         76.53192951  24.10779559  43.94521713  3.17457 0.0027408 **
## maturityP    -4.62266359   3.17801210  24.39984244 -1.45458 0.1585308
## colonySSB     4.10983380   6.36424962  34.10616354  0.64577 0.5227505
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##           (Intr) smlh   mtrtyP
## smlh      -0.984
## maturityP  0.257 -0.303
## colonySSB -0.165  0.055 -0.076
```

Check model fit

```r
check_model(b2)
```


Posterior Predictive Check
Model−predicted lines should resemble observed data line
Density
compound_n
Observed data
Model−predicted data
Linearity
Reference line should be flat and horizontal
Residuals
Fitted values
Homogeneity of Variance
Reference line should be flat and horizontal
√|Std. residuals|
Fitted values
Influential Observations
Points should be inside the contour lines
Std. Residuals
Leverage (h_ii)
38
45
30
0.7
54
0.7
49
34
9
53
15
Collinearity
High collinearity (VIF) may inflate parameter uncertainty
Variance Inflation Factor (VIF, log−scaled)
colony
maturity
smlh
Low (< 5)
Normality of Residuals
Dots should fall along the line
Sample Quantiles
Standard Normal Distribution Quantiles
Normality of Random Effects (family)
Dots should be plotted along the line
RE Quantiles
Theoretical Quantiles

#### Power analysis

```
# 56 samples,
# number of coefficients = 4,
# coefficients minus Intercept = 3
pwr_b <- pwr.f2.test(v = 56, u = 3, f2 = 0.1168/(1-0.1168), sig.level = 0.05)
pwr_b
```

```
##
##      Multiple regression power calculation
##
##               u = 3
##               v = 56
##              f2 = 0.132246376812
##       sig.level = 0.05
##           power = 0.61633398636
```

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
## -2.193649022 -0.656470901  0.102448217  0.718635430  1.907640611
##
## Random effects:
##  Groups   Name        Variance      Std.Dev.
##  family   (Intercept) 0.00131322013 0.0362383793
##  Residual             0.00720048654 0.0848556806
## Number of obs: 56, groups:  family, 36
##
## Fixed effects:
##                   Estimate    Std. Error            df  t value Pr(>|t|)
## (Intercept)   1.0286614945  0.0268377389 51.3350533536 38.32892  < 2e-16 ***
## mhc_het      -0.0381338548  0.0301679498 53.0219141496 -1.26405  0.21174
## ---
## 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
## -2.095136459 -0.653315112  0.113695726  0.687418314  1.847436161
##
## Random effects:
##  Groups   Name        Variance      Std.Dev.
##  family   (Intercept) 0.00154644902 0.0393249160
##  Residual             0.00714119814 0.0845056101
## Number of obs: 56, groups:  family, 36
##
## Fixed effects:
##                  Estimate     Std. Error             df  t value Pr(>|t|)
## (Intercept)  1.03101391140  0.02871644361 45.35909433637 35.90326   <2e-16 ***
## mhc_het     -0.03679744212  0.03054090182 52.11941131342 -1.20486   0.2337
## colonySSB   -0.00838946614  0.02680960458 26.33503941544 -0.31293   0.7568
## ---
## 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 0.010366750137 0.010366750137     1 52.11941131 1.45168 0.2337
## colony  0.000699292576 0.000699292576     1 26.33503942 0.09792 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
```

Effect size boostrap of linear mixed effects models

Demanding code and results are prerendered to optimize run-time of this markdown script. Bootstraps take several hours, only run, if you're interested in a different seed as a different randomization should switch up the data points but not overall results.

```
# subsample 30 individuals

model_r2_btrap <- function(x){

  n <- 1:dim(meta)[1]
  n_btrap <- sample(n, 30, replace = F)
  btrap_meta <- meta[n_btrap, ]
  div_trap <- lmerTest::lmer(compound_n ~ mhc_het + smlh + maturity + colony + (1|family),
                             data = btrap_meta)

  # bootstrap effect size with partR2
  partR2(div_trap, partvars = c("mhc_het", "smlh", "maturity", "colony"), nboot = 100, CI = 0.95)

}

# amount of times boostrap shall be repeated
nbtrap <- 1000
```

```
# repeat function call as often as specified for bootstrap iterations
system.time(
  div_r2_btrap <- lapply(1:nbtrap, model_r2_btrap)
)



sim_r2_btrap <- function(x){

n <- 1:dim(model_rel.df)[1]
n_btrap <- sample(n, 800, replace = F)
btrap_sim_df <- model_rel.df[n_btrap, ]

sim_trap <- lmerTest::lmer(scent_mds ~ rel + ufrac + colonyBool + agePaired + (1|familyBool) +
                          (1|pairID1) + (1|pairID2),
                        data = btrap_sim_df)


partR2(sim_trap, partvars = c("rel", "ufrac", "colonyBool", "agePaired"),
       nboot = 100, CI = 0.95)
}

nboot = 1000
sim_r2_btrap_res <- lapply(1:nboot, sim_r2_btrap)
```

Plot effect size results

Load in the pre-rendered data.

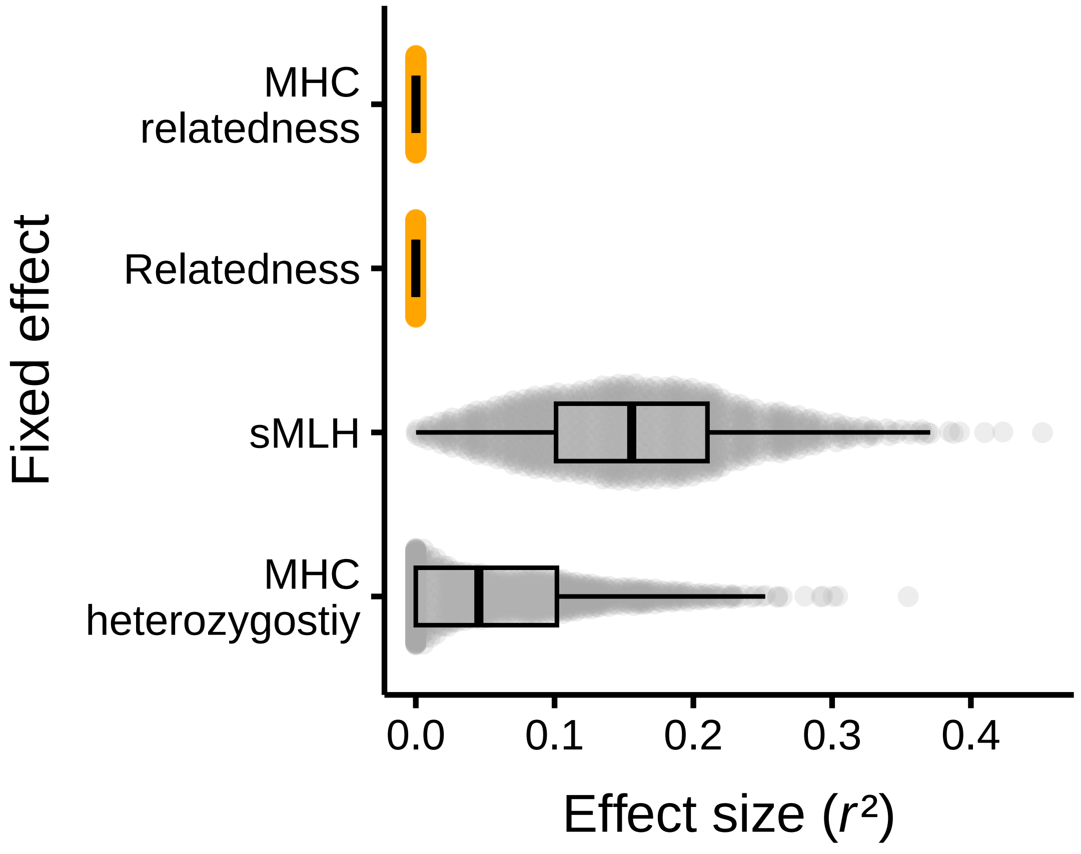
```
load(file = "RData/div_r2_btrap.RData")
load(file = "RData/sim_r2_btrap_res.RData")
```

Create a ggplot workable data frame

```
# extract r2 values only for the mhc heterozygosity fixed effect
r2_mhc_het <- sapply(1:length(div_r2_btrap), function(i) div_r2_btrap[[i]]$R2$estimate[2])
r2_smlh <- sapply(1:length(div_r2_btrap), function(i) div_r2_btrap[[i]]$R2$estimate[3])
r2_relatedness <- sapply(1:length(sim_r2_btrap_res), function(i) sim_r2_btrap_res[[1]]$R2$estimate[2])
r2_ufrac <- sapply(1:length(sim_r2_btrap_res), function(i) sim_r2_btrap_res[[1]]$R2$estimate[3])



group_factor <- c(rep("mhc_het", 1000),
                  rep("r2_smlh", 1000),
                  rep("relatedness", 1000),
                  rep("ufrac", 1000))

boot_estimate <- c(r2_mhc_het,
                   r2_smlh,
                   r2_relatedness,
                   r2_ufrac)

plot_div_btrap_df <- data.frame(boot_estimate = boot_estimate,
                                group = as.factor(group_factor))
```

Plot effect sizes

```
effectsize_best_model_gg <- ggplot(plot_div_btrap_df, aes(y = boot_estimate,
                                                           x = group,
                                                           color = group)) +
  # this arranges the points according to their density
  geom_quasirandom(alpha = 0.2, size = 3, width = 0.3, bandwidth = .95) +
  scale_color_manual(values = c("darkgrey","darkgrey", "orange","orange")) +
  # makes the boxplots
  geom_boxplot(width = 0.35, outlier.shape = NA, color = "black",
               alpha = 0.1, lwd=0.8, notch = F) +
  labs(y = expression(paste("Effect size (", italic("r"), "²)")),
       x = "Fixed effect") +
  scale_x_discrete(labels = c(
    "ufrac" = "MHC\nrelatedness",
     "relatedness" = "Relatedness",
    "r2_smlh" = "sMLH",
    "mhc_het" = "MHC\nheterozygostiy")) +
  coord_flip() +
custom_theme
effectsize_best_model_gg
```



```
ggsave(filename = "figures/figure3.png",dpi = 400,
       width = 22, height = 20,
       units = "cm",
       bg = "white")
```

Addition to model selection process by ‘dredge’ and ‘partR2’

Similarity model selection and power

```
# premise for 'dredge' to work correctly
options(na.action = "na.fail")

# model with all effects
sim_m <- lmerTest::lmer(scent_mds ~ rel + ufrac + colonyBool + agePaired + (1|familyBool) +
                        (1|pairID1) + (1|pairID2),
                      data = model_rel.df)

# calculate different versions
m1 <- dredge(sim_m)
```

Find appropiate models

```
## Warning in dredge(sim_m): comparing models fitted by REML
```

```
## Fixed term is "(Intercept)"
```

```
subset(m1, delta < 4)
```

```
## Global model call: lmerTest::lmer(formula = scent_mds ~ rel + ufrac + colonyBool +
##     agePaired + (1 | familyBool) + (1 | pairID1) + (1 | pairID2),
##     data = model_rel.df)
## ---
## Model selection table
##   (Intrc) clnyB df   logLik    AICc delta weight
## 3  0.3197     +  6 1974.155 -3936.3     0      1
## Models ranked by AICc(x)
## Random terms (all models):
##   1 | familyBool, 1 | pairID1, 1 | pairID2
```

```
# chose best
summary(get.models(m1, 1)[[1]])
```

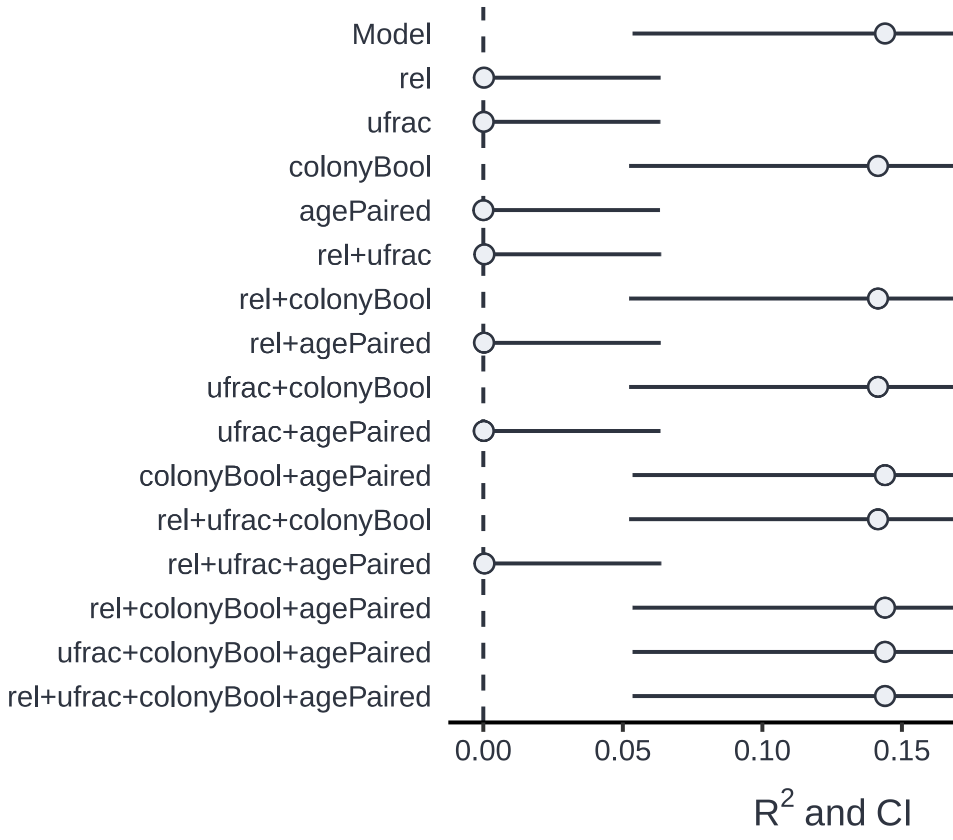
select best

```
## 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
## -3.138411025 -0.682850860 -0.075254444  0.581948282  3.762734892
##
## Random effects:
##  Groups     Name        Variance    Std.Dev.
##  pairID2    (Intercept) 0.00134929136 0.0367327015
##  pairID1    (Intercept) 0.00132586922 0.0364124872
##  familyBool (Intercept) 0.00421404024 0.0649156394
##  Residual                0.00381737000 0.0617848687
## Number of obs: 1540, groups:  pairID2, 55; pairID1, 55; familyBool, 2
##
## Fixed effects:
##                     Estimate     Std. Error             df  t value Pr(>|t|)
## (Intercept) 3.19681994e-01 4.75484750e-02 1.09434933e+00  6.72329 0.080084 .
## colonyBool1 8.39162891e-02 8.04874742e-03 1.22040771e+02 10.42601  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## colonyBool1 -0.154
```

```
r2_sim_m <- partR2(sim_m, partvars = c("rel", "ufrac", "colonyBool", "agePaired"),
              nboot = 100, CI = 0.95)
```

Calculate effect size by partR2

```
forestplot(r2_sim_m, type = "R2", line_size = 0.7, text_size = 14, point_size = 3)
```



Visualize effect sizes of fixed effects

```
r2_sim_m
```

```
##
##
```

```
## R2 (marginal) and 95% CI for the full model:
##  R2     CI_lower CI_upper nboot ndf
##  0.1439 0.0535   0.2507   100   6
##
## ----------
##
## Part (semi-partial) R2:
##  Predictor(s)                    R2     CI_lower CI_upper nboot ndf
##  Model                           0.1439 0.0535   0.2507   100   6
##  rel                             0.0002 0.0000   0.0634   100   5
##  ufrac                           0.0001 0.0000   0.0633   100   5
##  colonyBool                      0.1414 0.0523   0.2473   100   5
##  agePaired                       0.0000 0.0000   0.0631   100   4
##  rel+ufrac                       0.0004 0.0000   0.0636   100   4
##  rel+colonyBool                  0.1414 0.0523   0.2473   100   4
##  rel+agePaired                   0.0003 0.0000   0.0634   100   3
##  ufrac+colonyBool                0.1414 0.0523   0.2473   100   4
##  ufrac+agePaired                 0.0002 0.0000   0.0633   100   3
##  colonyBool+agePaired            0.1439 0.0535   0.2507   100   3
##  rel+ufrac+colonyBool            0.1414 0.0523   0.2473   100   3
##  rel+ufrac+agePaired             0.0004 0.0000   0.0636   100   2
##  rel+colonyBool+agePaired        0.1439 0.0535   0.2507   100   2
##  ufrac+colonyBool+agePaired      0.1439 0.0535   0.2507   100   2
##  rel+ufrac+colonyBool+agePaired 0.1439 0.0535   0.2507   100   1
```

Diversity model selection

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

m3 <- dredge(div_m)
```

Find appropiate models

```
## Warning in dredge(div_m): comparing models fitted by REML
```

```
## Fixed term is "(Intercept)"
```

```
subset(m3, delta < 4)
```

```
## Global model call: lmerTest::lmer(formula = compound_n ~ mhc_het + smlh + maturity +
##     colony + (1 | family), data = meta)
## ---
## Model selection table
##     (Int) cln mtr mhc_het   sml df   logLik  AICc delta weight
## 16 -22.62   +   +  1.1860 77.18  7 -222.645 461.6  0.00  0.620
## 12 -21.16   +   +         76.53  6 -225.280 464.3  2.65  0.165
## 15 -20.32       +  1.2930 76.51  6 -225.621 465.0  3.33  0.117
## 14 -12.22   +     -0.2606 66.23  6 -225.800 465.3  3.69  0.098
## Models ranked by AICc(x)
## Random terms (all models):
##   1 | family
```

```
summary(get.models(m3, 1)[[1]])
```

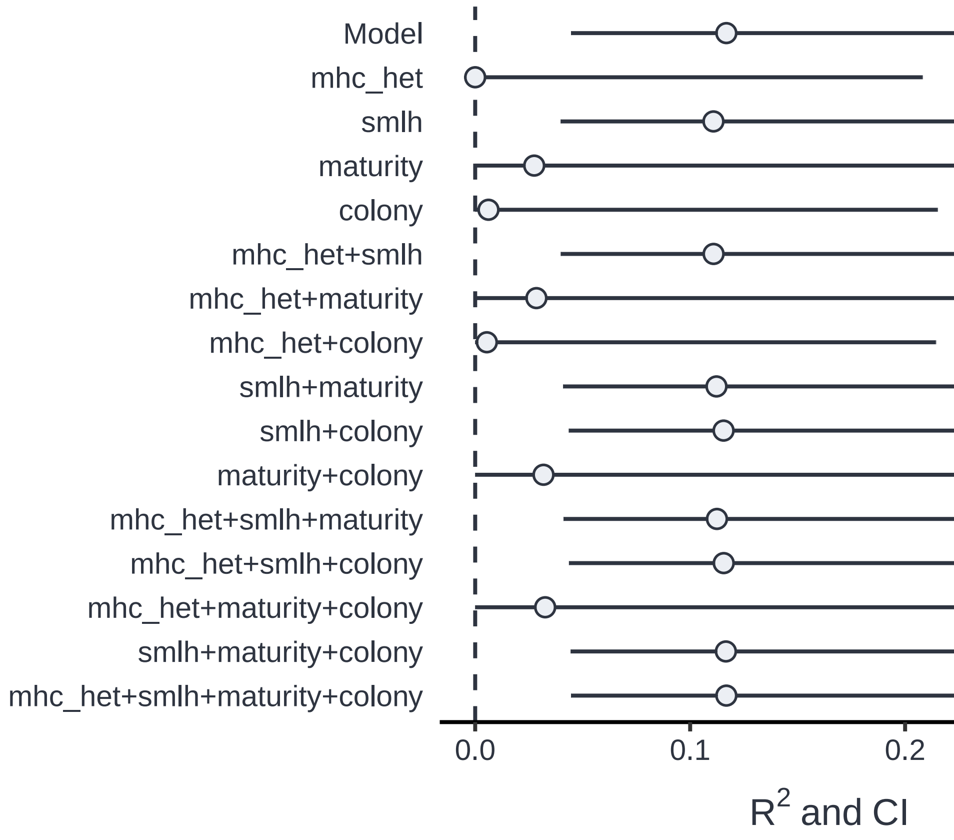
select best

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: compound_n ~ colony + maturity + mhc_het + smlh + (1 | family)
##    Data: meta
##
## REML criterion at convergence: 445.3
##
## Scaled residuals:
##           Min            1Q        Median            3Q           Max
## -1.0928313033 -0.4734130794 -0.0457156417  0.2360966398  2.0910996889
##
## Random effects:
##  Groups   Name        Variance   Std.Dev.
##  family   (Intercept) 289.098507 17.0028970
##  Residual             102.550738 10.1267338
## Number of obs: 56, groups:  family, 36
##
## Fixed effects:
##                 Estimate   Std. Error          df  t value  Pr(>|t|)
## (Intercept) -22.62052081  25.23757770  41.63258609 -0.89630 0.3752445
## colonySSB     4.06383229   6.41330434  33.84088527  0.63366 0.5305648
## maturityP    -4.75717833   3.27180878  24.21976233 -1.45399 0.1587859
## mhc_het       1.18588993   5.45889047  42.15362665  0.21724 0.8290693
## smlh         77.17560246  24.48424644  42.14207673  3.15205 0.0029835 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##           (Intr) clnSSB mtrtyP mhc_ht
## colonySSB -0.152
## maturityP  0.289 -0.068
## mhc_het   -0.247 -0.034 -0.181
## smlh      -0.973  0.051 -0.313  0.100
```

```r
r2_div_m <- partR2(div_m,
                  partvars = c("mhc_het", "smlh", "maturity", "colony"),
                  nboot = 100, CI = 0.95)
```

Calculate effect size by partR2

```r
forestplot(r2_div_m, type = "R2", line_size = 0.7, text_size = 14, point_size = 3)
```



Visualize effect sizes of fixed effects

```
r2_div_m
```

```
##
##
## R2 (marginal) and 95% CI for the full model:
##  R2     CI_lower CI_upper nboot ndf
##  0.1168 0.0446   0.3295   100   5
##
## ----------
##
## Part (semi-partial) R2:
##  Predictor(s)                 R2     CI_lower CI_upper nboot ndf
##  Model                        0.1168 0.0446   0.3295   100   5
##  mhc_het                      0.0000 0.0000   0.2080   100   4
##  smlh                         0.1108 0.0398   0.3233   100   4
##  maturity                     0.0275 0.0000   0.2368   100   4
##  colony                       0.0062 0.0000   0.2150   100   4
##  mhc_het+smlh                 0.1109 0.0398   0.3233   100   3
##  mhc_het+maturity             0.0285 0.0000   0.2379   100   3
##  mhc_het+colony               0.0054 0.0000   0.2142   100   3
##  smlh+maturity                0.1122 0.0409   0.3247   100   3
##  smlh+colony                  0.1155 0.0436   0.3282   100   3
##  maturity+colony              0.0318 0.0000   0.2413   100   3
##  mhc_het+smlh+maturity        0.1125 0.0411   0.3250   100   2
##  mhc_het+smlh+colony          0.1156 0.0436   0.3282   100   2
##  mhc_het+maturity+colony      0.0326 0.0000   0.2421   100   2
##  smlh+maturity+colony         0.1166 0.0444   0.3293   100   2
##  mhc_het+smlh+maturity+colony 0.1168 0.0446   0.3295   100   1
```

Correlation of genetic effects

```
gen_m <- lmerTest::lmer(ufrac ~ rel + colonyBool + agePaired + (1|pairID1) + (1|pairID2) + (1|familyBoo
                        data = model_rel.df)

m2 <- dredge(gen_m)
```

Find appropiate models

```
## Warning in dredge(gen_m): comparing models fitted by REML
```

```
## Fixed term is "(Intercept)"
```

```
subset(m2, delta < 4)
```

```
## Global model call: lmerTest::lmer(formula = ufrac ~ rel + colonyBool + agePaired +
##     (1 | pairID1) + (1 | pairID2) + (1 | familyBool), data = model_rel.df)
## ---
## Model selection table
##   (Intrc)     rel df  logLik   AICc delta weight
## 1  0.6825          5 337.011 -664.0  0.00  0.675
## 5  0.7107 -0.1267  6 337.288 -662.5  1.46  0.325
## Models ranked by AICc(x)
## Random terms (all models):
##   1 | pairID1, 1 | pairID2, 1 | familyBool
```

```
summary(get.models(m2, 1)[[1]])
```

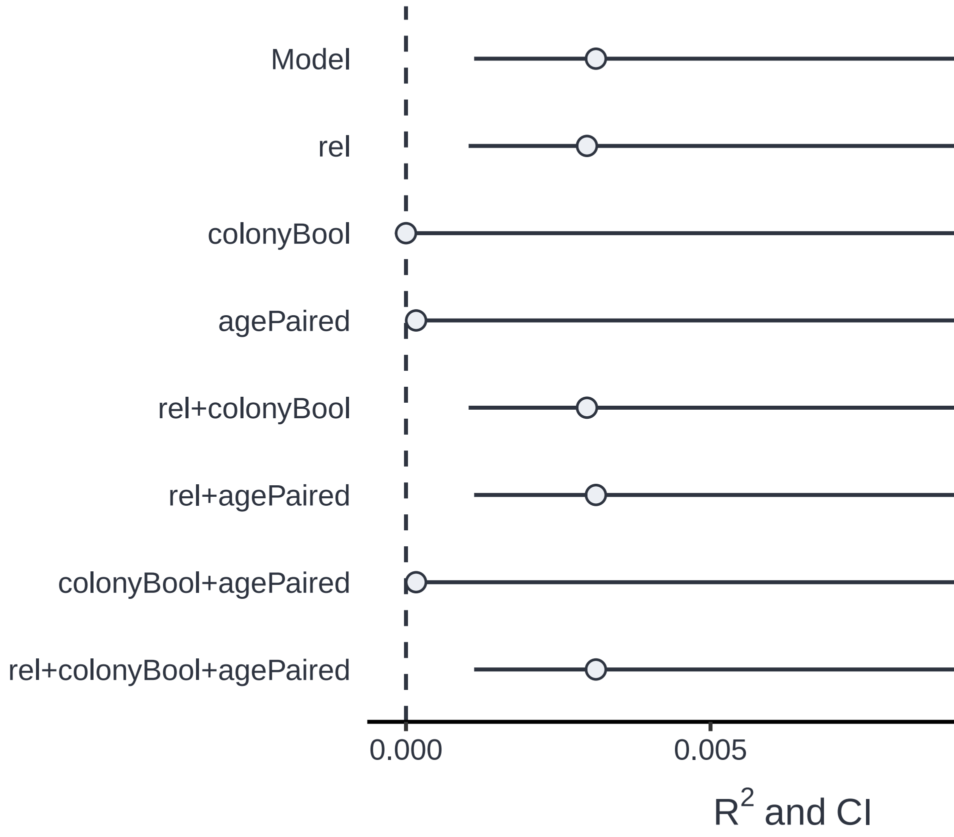
select best

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: ufrac ~ (1 | pairID1) + (1 | pairID2) + (1 | familyBool)
##    Data: model_rel.df
##
## REML criterion at convergence: -674
##
## Scaled residuals:
##          Min           1Q       Median           3Q          Max
## -4.485507072 -0.371589025  0.270089938  0.677691079  2.138879974
##
## Random effects:
##  Groups     Name        Variance      Std.Dev.
##  pairID1    (Intercept) 0.00218749622 0.0467706769
##  pairID2    (Intercept) 0.00115356500 0.0339641723
##  familyBool (Intercept) 0.02938341728 0.1714159190
##  Residual               0.03556353542 0.1885829669
## Number of obs: 1540, groups:  pairID1, 55; pairID2, 55; familyBool, 2
##
## Fixed effects:
##                Estimate  Std. Error          df t value Pr(>|t|)
## (Intercept) 0.682529592 0.123310856 1.004398662 5.53503  0.11302
```

```
r2_gen_m <- partR2(gen_m, partvars = c("rel", "colonyBool", "agePaired"),
                   nboot = 100, CI = 0.95)
```

Calculate effect size by partR2

```
forestplot(r2_gen_m, type = "R2", line_size = 0.7, text_size = 14, point_size = 3)
```



Visualize effect sizes of fixed effects

```
r2_gen_m
```

```
##
##
## R2 (marginal) and 95% CI for the full model:
##  R2     CI_lower CI_upper nboot ndf
##  0.0031 0.0011   0.0127   100   5
##
## ----------
##
## Part (semi-partial) R2:
##  Predictor(s)             R2     CI_lower CI_upper nboot ndf
##  Model                    0.0031 0.0011   0.0127   100   5
##  rel                      0.0030 0.0010   0.0126   100   4
##  colonyBool               0.0000 0.0000   0.0097   100   4
##  agePaired                0.0002 0.0000   0.0099   100   3
##  rel+colonyBool           0.0030 0.0010   0.0126   100   3
##  rel+agePaired            0.0031 0.0011   0.0127   100   2
##   colonyBool+agePaired      0.0002 0.0000    0.0099   100   2
##   rel+colonyBool+agePaired 0.0031 0.0011    0.0127   100   1
```

PERMANOVA for individual genotypes and alleles respectively

Create workable dataframe

```r
# 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

```r
# 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 0.0151593876 0.0084655024 0.46315  0.558
## a2        1 0.0096098366 0.0053664500 0.29360  0.642
## a3        1 0.0069896633 0.0039032587 0.21355  0.744
## a4        1 0.0378931425 0.0211607815 1.15771  0.283
## a5        1 0.0026201425 0.0014631741 0.08005  0.873
## a6        1 0.0442421031 0.0247062507 1.35169  0.240
## a7        1 0.0221723994 0.0123817997 0.67741  0.433
## a8        1 0.0564621645 0.0315303363 1.72503  0.207
## a9        1 0.0022475191 0.0012550888 0.06867  0.910
## a10       1 0.0239470526 0.0133728246 0.73163  0.442
## a11       1 0.0201670701 0.0112619576 0.61615  0.446
## a12       1 0.0751690749 0.0419768926 2.29657  0.131
## a13       1 0.0171102090 0.0095549055 0.52275  0.486
## a14       1 0.0578941065 0.0323299800 1.76878  0.199
## a15       1 0.1338806176 0.0747633560 4.09033  0.052 .
## a16       1 0.0026532032 0.0014816363 0.08106  0.867
## a17       1 0.0129836481 0.0072504976 0.39668  0.594
## a18       1 0.0591946266 0.0330562334 1.80852  0.194
## a19       1 0.0120118860 0.0067078336 0.36699  0.564
## Residual 36 1.1783172402 0.6580112405
## Total    55 1.7907250935 1.0000000000
## ---
## 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  0.806142857143
## 2  0.813200000000
## 3  0.883500000000
## 4  0.768142857143
## 5  0.910000000000
## 6  0.760000000000
## 7  0.806142857143
## 8  0.760000000000
## 9  0.910000000000
## 10 0.806142857143
## 11 0.806142857143
## 12 0.760000000000
## 13 0.806142857143
## 14 0.760000000000
## 15 0.760000000000
## 16 0.910000000000
## 17 0.806142857143
## 18 0.760000000000
## 19 0.806142857143
```

PERMANOVA for associated odour nmds profiles with genotypes

```
# 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)
```

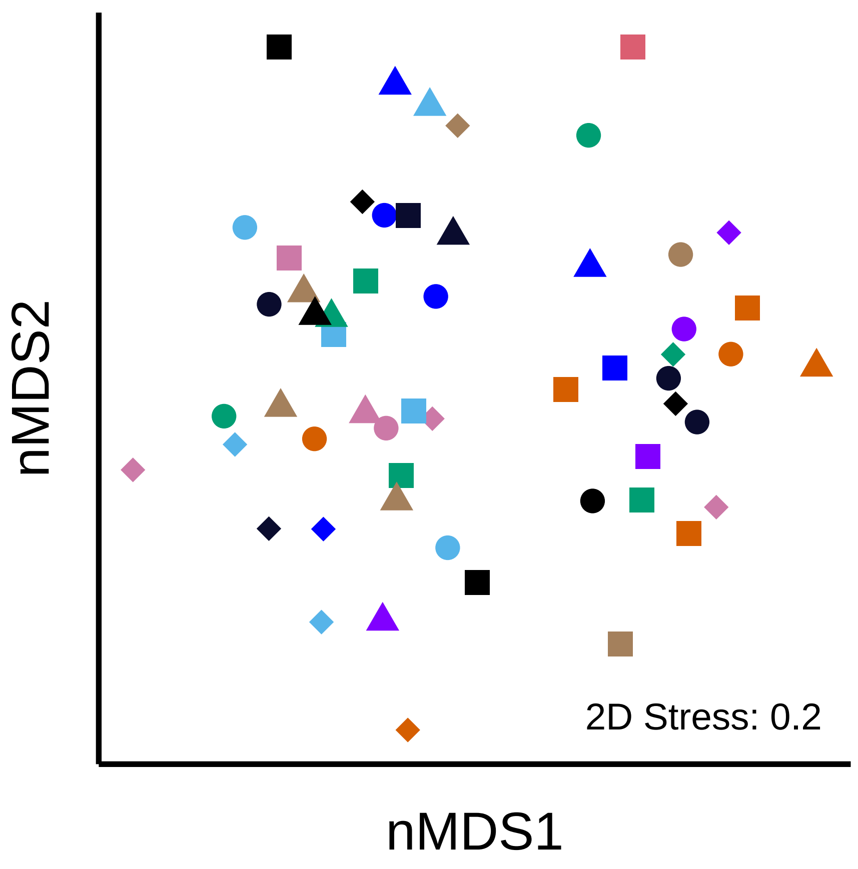
```
## Permutation test for adonis under reduced model
## Terms added sequentially (first to last)
## Permutation: free
## Number of permutations: 999
##
## vegan::adonis2(formula = scent ~ het_table$gtype)
##                 Df     SumOfSqs           R2       F Pr(>F)
## het_table$gtype 36  8.803613197 0.6710248615 1.07653  0.166
## Residual        19  4.316039594 0.3289751385
## Total           55 13.119652790 1.0000000000
```

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

Plot PERMANOVA results

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

```r
# 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()
  )
```



```r
# save output
ggsave(filename = "figures/supplementary_figure1.png",
       width = 32, height = 16,
       units = "cm", dpi = 400)
```

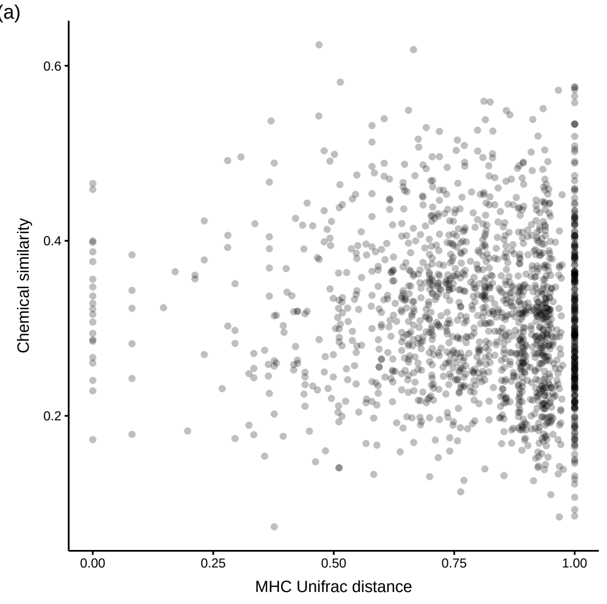
Create manuscript panel figure

```r
# 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'
```

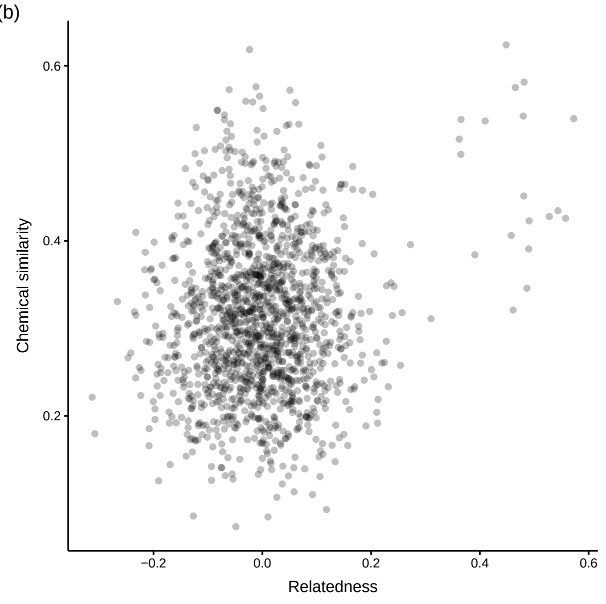
```r
# print
panel_final
```

(a)



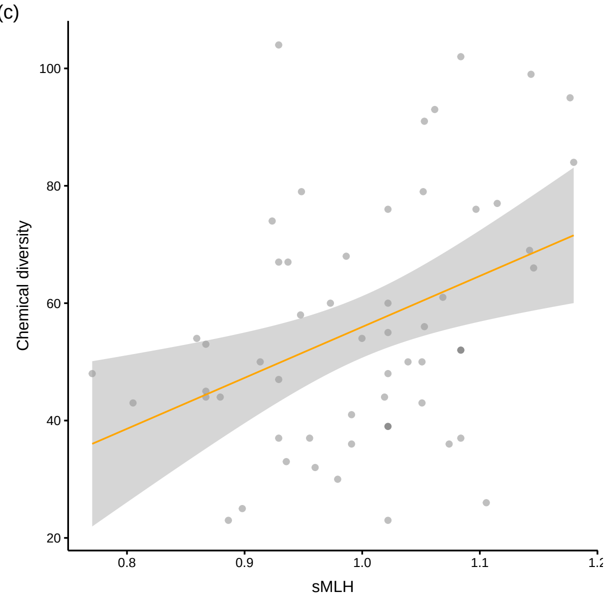
Chemical similarity
0.6
0.4
0.2
0.00
0.25
0.50
0.75
1.00
MHC Unifrac distance

(b)



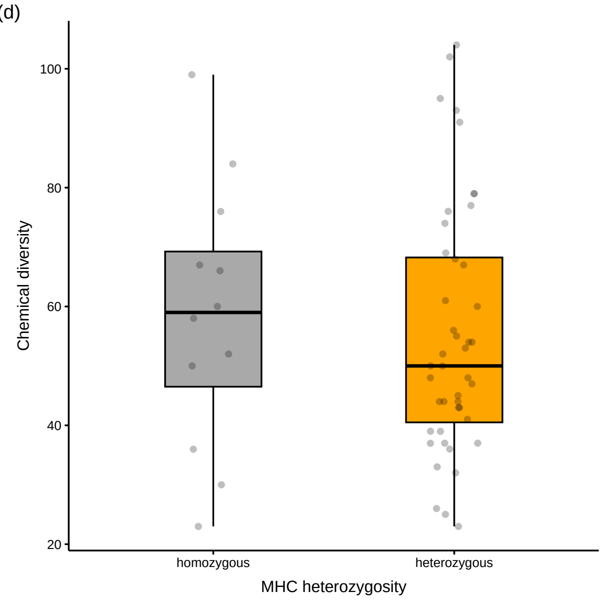
Chemical similarity
0.6
0.4
0.2
−0.2
0.0
0.2
0.4
0.6
Relatedness

(c)



Chemical diversity
100
80
60
40
20
0.8
0.9
1.0
1.1
sMLH

(d)



Chemical diversity
100
80
60
40
20
homozygous
heterozygous
MHC heterozygosity

```
# save high resolution
ggsave(filename = "figures/figure2.png",
       panel_final,dpi = 400,
       width = 33.97, height = 31.04,
       units = "cm",
       bg = "white"
       )
```

Repeat for best subset by SIMPER analysis

```
# simper analysis
min_range <- 12
max_range <- 42
scent_range <- scent[,as.numeric(colnames(scent)) >= min_range & as.numeric(colnames(scent)) <= max_rang

simp_colony <- vegan::simper(scent_range, meta$colony)

# getting  best substances and their contribution to colony dissimilarity

#set numbers of best compounds to retain
keep_compounds <- 15

simp_colony_names <- summary(simp_colony, ordered = T)[[1]] %>%
        rownames(.) %>%
        .[1:keep_compounds]

contribution <- summary(simp_colony, ordered = TRUE)[[1]] %>%
        .$average %>%
        .[1:keep_compounds]

# indices of colony substances (58,62,68,74,86,89,90,98,106,107,110,164,181,189,211)
ind_col <- which(names(scent) %in% simp_colony_names)
# 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  15.4498717948718 3.05575894307
## 2  19.7148780487805 2.48839019949
## 3  19.5174358974359 2.37671240496
## 4  16.3895238095238 2.30226998568
## 5  26.7808450704225 2.27321897210
## 6  21.3982051282051 2.06493210540
## 7              21.34 1.87869623710
## 8  30.7885294117647 1.51069283280
## 9  37.5734782608696 1.49482594047
## 10 38.4982142857143 1.47661837144
## 11 17.4137142857143 1.38914137040
## 12  20.509347826087 1.35687404025
## 13 15.5008108108108 1.27741681524
## 14           38.5256 1.23455984162
## 15 19.6568571428571 1.21537246870
```

```r
print(
  paste0(
    "Subset of ", keep_compounds, " chemicals accounts for ", round(sum(col_simp$contrib), digits = 2),
    "% of colony similarity"))
```

```
## [1] "Subset of 15 chemicals accounts for 27.4% of colony similarity"
```

adjust scent matrix, include only 15 most explanatory compounds

```r
scent_simper <- scent[, ind_col]

## keep order of rows consistent
scent <- scent[match(rownames(peak_factors),rownames(scent)),]
## NMDS using Bray-Curtis dissimilarities
scent_nmds_simper.obj <- vegan::metaMDS(comm = scent, distance = "bray")
```

```
## Run 0 stress 0.237312208882
## Run 1 stress 0.23724653363
## ... New best solution
## ... Procrustes: rmse 0.00328866268999  max resid 0.0177259193894
## Run 2 stress 0.237319083301
## ... Procrustes: rmse 0.00471342410828  max resid 0.019275855762
## Run 3 stress 0.237246533538
## ... New best solution
## ... Procrustes: rmse 9.83131807562e-06  max resid 5.0480851187e-05
## ... Similar to previous best
## Run 4 stress 0.237312208145
## ... Procrustes: rmse 0.00328744946049  max resid 0.0177186706632
## Run 5 stress 0.260781284422
## Run 6 stress 0.256616629286
## Run 7 stress 0.23731220795
## ... Procrustes: rmse 0.00328701300242  max resid 0.0177121882667
## Run 8 stress 0.247114064892
## Run 9 stress 0.246650951445
## Run 10 stress 0.237312207968
## ... Procrustes: rmse 0.00328871873412  max resid 0.0177259445398
## Run 11 stress 0.237259771122
## ... Procrustes: rmse 0.00329480984218  max resid 0.0184474395207
## Run 12 stress 0.237319083493
## ... Procrustes: rmse 0.00471493927975  max resid 0.0192730934228
## Run 13 stress 0.237319083107
## ... Procrustes: rmse 0.004713652389  max resid 0.0192711729731
## Run 14 stress 0.260151039362
## Run 15 stress 0.237312208045
## ... Procrustes: rmse 0.00328867379457  max resid 0.0177248516597
## Run 16 stress 0.297268721257
## Run 17 stress 0.265869960872
## Run 18 stress 0.237312208092
## ... Procrustes: rmse 0.00328663078623  max resid 0.0177145971122
## Run 19 stress 0.247114062462
## Run 20 stress 0.251568953819
## *** Best solution repeated 1 times
```

```r
## get x and y coordinates
scent_nmds_simper <- as.data.frame(scent_nmds.obj[["points"]]) %>%
## add the colony as a factor to each sample
cbind(.,colony = peak_factors[["colony"]])
```

Create vectorized distance measurements for scent data

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

Repeat chemical similarity models with best chemicals subset

```r
model_rel.df %<>% cbind(., scent_simper_mds = b_simper)

# mhc
z1 <- lmerTest::lmer(scent_simper_mds ~ ufrac + colonyBool + agePaired + (1|familyBool) +
                       (1|pairID1) + (1|pairID2),
                     data = model_rel.df)
# relatedness
z2 <- lmerTest::lmer(scent_simper_mds ~ rel + colonyBool + agePaired + (1|familyBool) + (1|pairID1) +
                       (1|pairID2),
                     data = model_rel.df)
# mhc & relatedness
z3 <- lmerTest::lmer(scent_simper_mds ~ rel + ufrac + colonyBool + agePaired + (1|familyBool) +
                       (1|pairID1) + (1|pairID2),
                     data = model_rel.df)
# no genetic effect
z4 <- lmerTest::lmer(scent_simper_mds ~ colonyBool + agePaired + (1|familyBool) + (1|pairID1) +
                       (1|pairID2),
                     data = model_rel.df)

# compare model performance scores
compare_performance(z1, z2, z3, z4, rank = T) %>%
  arrange(Name)
```

```
## # Comparison of Model Performance Indices
##
## Name |           Model | R2 (cond.) | R2 (marg.) |   ICC |  RMSE | Sigma | AIC weights | AICc weight
## ----------------------------------------------------------------------------------------------------
## z1   | lmerModLmerTest |      0.670 |      0.350 | 0.493 | 0.096 | 0.099 |       0.157 |        0.15
## z2   | lmerModLmerTest |      0.658 |      0.365 | 0.461 | 0.096 | 0.099 |       0.304 |        0.30
## z3   | lmerModLmerTest |      0.658 |      0.365 | 0.462 | 0.096 | 0.099 |       0.113 |        0.11
## z4   | lmerModLmerTest |      0.670 |      0.351 | 0.492 | 0.096 | 0.099 |       0.426 |        0.43
```

```r
summary(z2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_simper_mds ~ rel + colonyBool + agePaired + (1 | familyBool) +
##     (1 | pairID1) + (1 | pairID2)
##    Data: model_rel.df
##
```

```
## REML criterion at convergence: -2489.9
##
## Scaled residuals:
##          Min           1Q       Median           3Q          Max
## -3.361727005 -0.601762806  0.044879466  0.656512064  4.481398163
##
## Random effects:
##  Groups     Name        Variance      Std.Dev.
##  pairID2    (Intercept) 0.00371962719 0.0609887464
##  pairID1    (Intercept) 0.00222537088 0.0471738368
##  familyBool (Intercept) 0.00248148577 0.0498145136
##  Residual               0.00985985813 0.0992968183
## Number of obs: 1540, groups:  pairID2, 55; pairID1, 55; familyBool, 2
##
## Fixed effects:
##                   Estimate     Std. Error             df   t value Pr(>|t|)
## (Intercept)    0.5656545688   0.0416705362   1.4157079114  13.57445 0.018025
## rel           -0.0339705052   0.0329143620 473.1835094380  -1.03209 0.302558
## colonyBool1   -0.2041643091   0.0118146357 112.2879453844 -17.28063  < 2e-16
## agePairedM/P  -0.0165814738   0.0119597238 128.9256179904  -1.38644 0.168005
## agePairedP/P  -0.0317056901   0.0224867649  99.4223955740  -1.40997 0.161669
##
## (Intercept)  *
## rel
## colonyBool1  ***
## agePairedM/P
## agePairedP/P
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) rel    clnyB1 agPM/P
## rel         -0.149
## colonyBool1 -0.256  0.016
## agePairdM/P -0.250 -0.024 -0.002
## agePairdP/P -0.237 -0.009 -0.003  0.900
```

```r
summary(z4)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_simper_mds ~ colonyBool + agePaired + (1 | familyBool) +
##     (1 | pairID1) + (1 | pairID2)
##    Data: model_rel.df
##
## REML criterion at convergence: -2493.8
##
## Scaled residuals:
##          Min           1Q       Median           3Q          Max
## -3.369275911 -0.607272157  0.045559137  0.657635847  4.508880002
##
## Random effects:
##  Groups     Name        Variance      Std.Dev.
##  pairID2    (Intercept) 0.00373524835 0.0611166781
##  pairID1    (Intercept) 0.00223267976 0.0472512409
##  familyBool (Intercept) 0.00356932632 0.0597438392
##  Residual               0.00985536801 0.0992742062
## Number of obs: 1540, groups:  pairID2, 55; pairID1, 55; familyBool, 2
##
## Fixed effects:
##                   Estimate     Std. Error             df    t value Pr(>|t|)
## (Intercept)    0.5579296640   0.0474009636   1.3493551125  11.77043 0.025259
## colonyBool1   -0.2038667903   0.0118300291 112.1681349190 -17.23299  < 2e-16
## agePairedM/P  -0.0167874180   0.0119726612 128.7456942475  -1.40215 0.163278
## agePairedP/P  -0.0318598080   0.0225198624  99.4410644101  -1.41474 0.160268
##
## (Intercept)  *
## colonyBool1  ***
## agePairedM/P
## agePairedP/P
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##              (Intr) clnyB1 agPM/P
## colonyBool1 -0.224
## agePairdM/P -0.224 -0.001
## agePairdP/P -0.210 -0.003  0.900
```

```
# if interested
# check model performance by
# check_model(z2)
#
```

Repeat analyses for moms only

empty workspace to repeat analyses for moms

```
rm(list=ls())
```

Subset scent data to correlate same individuals

```
## read in meta data
meta <- read.table(file = "data/arga_metadata.txt", sep = "\t") %>%
  `colnames<-`(unlist(.[1,])) %>%
  .[-1,] %>%
  # subset for only moms
  filter(., maturity == "M")



## 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)
```

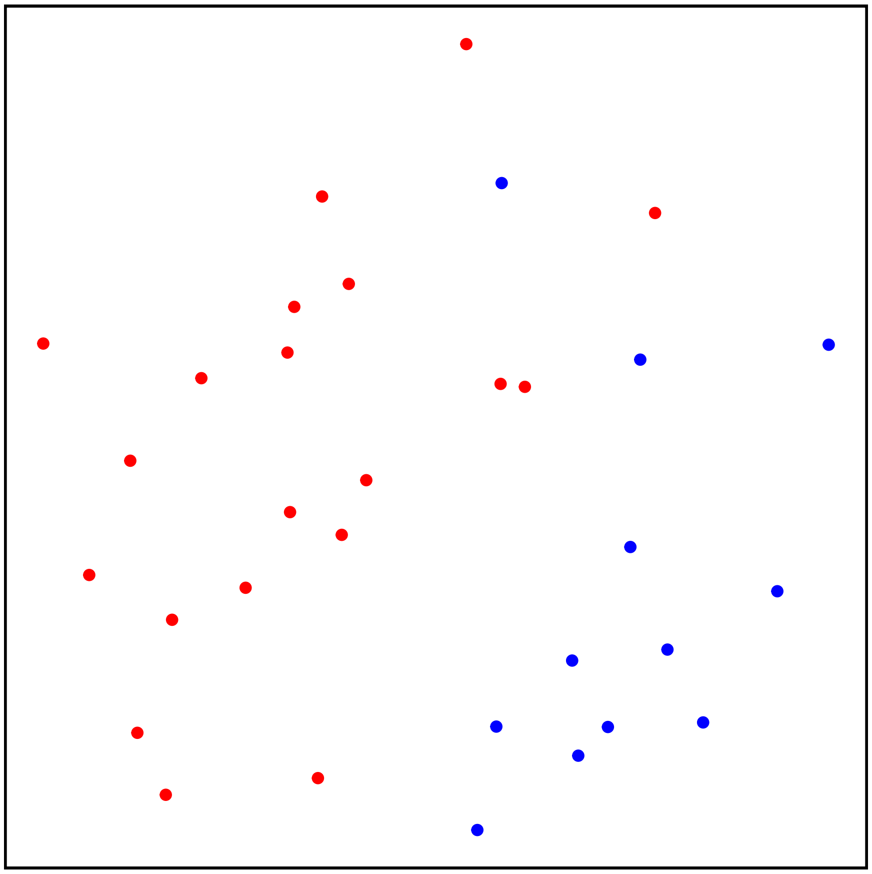
```r
n_scnt <- rownames(scent)

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

```
## Run 0 stress 0.236250449583
## Run 1 stress 0.240516333127
## Run 2 stress 0.234374193559
## ... New best solution
## ... Procrustes: rmse 0.056523221584  max resid 0.202907648393
## Run 3 stress 0.236896136491
## Run 4 stress 0.258284912855
## Run 5 stress 0.257447061896
## Run 6 stress 0.234468815214
## ... Procrustes: rmse 0.00904827880853  max resid 0.0403112319561
## Run 7 stress 0.25758403711
## Run 8 stress 0.250551297651
## Run 9 stress 0.254928903427
## Run 10 stress 0.258775754585
## Run 11 stress 0.235605108844
## Run 12 stress 0.250551297513
## Run 13 stress 0.267275477033
## Run 14 stress 0.234662722242
## ... Procrustes: rmse 0.0347701850277  max resid 0.173995175855
## Run 15 stress 0.250551297661
## Run 16 stress 0.265090470062
## Run 17 stress 0.289911853996
## Run 18 stress 0.257447055237
## Run 19 stress 0.267368015967
## Run 20 stress 0.285105249604
## *** Best solution was not repeated -- monoMDS stopping criteria:
##     19: stress ratio > sratmax
##      1: scale factor of the gradient < sfgrmin
```

```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
```

```r
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.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Calculate MHC heterozygosity relatedness between individuals

```r
## 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,]

## 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 mark
##   Especially, rxy and ritland estimator are not defined when bi-allelic estimates are used with alle
##   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 oc
```

```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 = dim(mhc_het_dat)[1],
                         ncol = dim(mhc_het_dat)[1])
## 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()
```

```r
# 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 microsatellite relatedness values

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

```r
# 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)

msats_df %<>% .[match(meta$real_id, .$id),]
```

Calculate relatedness of individuals based on Queller & Goodnight

```r
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

## Warning in prop.test(c(emp, non), c(sum(table(empirical.list)),
## sum(table(relate.non.X.mean))), : Chi-Quadrat-Approximation kann inkorrekt sein
```

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 = dim(msats_df)[1],
                     ncol = dim(msats_df)[1])
## 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

```
# custom theme to ease figure creation
custom_theme <- ggplot2::theme_classic(base_size = 16,
                                       base_line_size = 1,
                                       base_rect_size = 1) +
  ggplot2::theme(
    # plot.margin = unit(c(5.5, 8.5, 5.5, 5.5), "pt"), #c(top, right, bottom, left)
    plot.margin = margin(5.5,6.5,5.5,5.5, "pt"),
    panel.grid = element_blank(),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black"),
    aspect.ratio = 1
  )
```

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
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

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

Transform model variables

```r
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()

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

```
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))
```

update data frame with meta data

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

```
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
```

```
# 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
```

```r
# table is pre-prepped, thus rows correspond to same individuals in meta data
smlh_res <- read.table("data/msats_genotypes_inbreedR.txt", sep = "\t")
smlh_res <- smlh_res[match(meta$real_id, rownames(smlh_res)), ] %>%
  # 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)
)
```

Repeat models for chemical similarity

```r
# without age and indicator for family groupings
# as grouping factors must have >1 sampled level

# mhc
c1 <- lmerTest::lmer(scent_mds ~ ufrac + colonyBool + (1|pairID1) + (1|pairID2),
                     data = model_rel.df)
# relatedness
c2 <- lmerTest::lmer(scent_mds ~ rel + colonyBool + (1|pairID1) + (1|pairID2),
                     data = model_rel.df)
# mhc & relatedness
c3 <- lmerTest::lmer(scent_mds ~ rel + ufrac + colonyBool + (1|pairID1) + (1|pairID2),
                     data = model_rel.df)
# no genetic effect
c4 <- lmerTest::lmer(scent_mds ~ colonyBool + (1|pairID1) + (1|pairID2),
                     data = model_rel.df)


# compare model performance scores
compare_performance(c1, c2, c3, c4, rank = T) %>%
  arrange(Name)
```

```
## # Comparison of Model Performance Indices
##
## Name |           Model | R2 (cond.) | R2 (marg.) |   ICC |  RMSE | Sigma | AIC weights | AICc weight
## -----------------------------------------------------------------------------------------------------------
## c1   | lmerModLmerTest |      0.524 |      0.245 | 0.370 | 0.060 | 0.064 |       0.200 |        0.198
## c2   | lmerModLmerTest |      0.524 |      0.245 | 0.370 | 0.060 | 0.064 |       0.196 |        0.194
## c3   | lmerModLmerTest |      0.524 |      0.244 | 0.369 | 0.060 | 0.064 |       0.074 |        0.07
## c4   | lmerModLmerTest |      0.525 |      0.245 | 0.370 | 0.060 | 0.064 |       0.531 |        0.538
```

```r
summary(c2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_mds ~ rel + colonyBool + (1 | pairID1) + (1 | pairID2)
##    Data: model_rel.df
##
## REML criterion at convergence: -1211.3
##
## Scaled residuals:
##          Min           1Q       Median           3Q          Max
## -2.434739031 -0.668546769 -0.125072431  0.516608517  3.362588564
##
## Random effects:
##  Groups   Name        Variance      Std.Dev.
##  pairID1  (Intercept) 0.00129394285 0.0359714171
##  pairID2  (Intercept) 0.00108030441 0.0328679845
##  Residual             0.00404600916 0.0636082475
## Number of obs: 496, groups:  pairID1, 31; pairID2, 31
##
## Fixed effects:
##                   Estimate     Std. Error             df  t value   Pr(>|t|)
## (Intercept) 2.64239187e-01 1.31288102e-02 4.08268587e+01 20.12667 < 2.22e-16
## rel         1.70656495e-03 3.59170911e-02 4.83363527e+02  0.04751    0.96212
## colonyBool1 9.11773096e-02 1.13258476e-02 7.04830509e+01  8.05037 1.4125e-11
##
## (Intercept) ***
## rel
## colonyBool1 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) rel
## rel          0.030
## colonyBool1 -0.666  0.045
```

```r
summary(c4)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_mds ~ colonyBool + (1 | pairID1) + (1 | pairID2)
##    Data: model_rel.df
##
## REML criterion at convergence: -1216.1
##
## Scaled residuals:
##          Min           1Q       Median           3Q          Max
## -2.438554364 -0.669736096 -0.127401072  0.518947513  3.362266186
##
## Random effects:
##  Groups   Name        Variance      Std.Dev.
##  pairID1  (Intercept) 0.00129455037 0.0359798606
##  pairID2  (Intercept) 0.00108126826 0.0328826437
##  Residual             0.00403675432 0.0635354572
## Number of obs: 496, groups:  pairID1, 31; pairID2, 31
##
## Fixed effects:
```

```
##                   Estimate    Std. Error            df  t value   Pr(>|t|)
## (Intercept)  0.2642215710  0.0131250678 41.6332801738 20.13106 < 2.22e-16 ***
## colonyBool1  0.0911511887  0.0113126069 71.5115810251  8.05749 1.2502e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## colonyBool1 -0.668
```

Repeat models for chemical diversity

```
# exclude family as random effects (no accounting of same family in different
# individuals)
# exlude age as only one 'age' group being moms is modelled


d1 <- lm(compound_n ~ mhc_het + colony , data = meta)

d2 <- lm(compound_n ~ smlh + colony, data = meta)

d3 <- lm(compound_n ~ mhc_het + smlh + colony, data = meta)

d4 <- lm(compound_n ~ colony, data = meta)


compare_performance(d1, d2, d3, d4, rank = T) %>% arrange(Name)
```

```
## # Comparison of Model Performance Indices
##
## Name | Model |    R2 | R2 (adj.) |   RMSE |  Sigma | AIC weights | AICc weights | BIC weights | Perf
## -----------------------------------------------------------------------------------------------------
## d1   |    lm | 0.091 |     0.029 | 21.014 | 22.075 |       0.086 |        0.086 |       0.079 |
## d2   |    lm | 0.190 |     0.134 | 19.841 | 20.842 |       0.540 |        0.543 |       0.500 |
## d3   |    lm | 0.191 |     0.104 | 19.831 | 21.200 |       0.202 |        0.134 |       0.090 |
## d4   |    lm | 0.074 |     0.043 | 21.217 | 21.913 |       0.172 |        0.236 |       0.331 |
```

```
summary(d2)
```

```
##
## Call:
## lm(formula = compound_n ~ smlh + colony, data = meta)
##
## Residuals:
##         Min          1Q      Median          3Q         Max
## -36.56900864 -15.29316134  -0.55473784  12.51322130  45.43810011
##
## Coefficients:
##                 Estimate   Std. Error  t value Pr(>|t|)
## (Intercept) -27.91731762  39.51389375 -0.70652 0.485504
## smlh         80.82602303  39.60992119  2.04055 0.050493 .
## colonySSB    10.39704337   7.67024625  1.35550 0.185721
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## Residual standard error: 20.8416688 on 29 degrees of freedom
## Multiple R-squared:  0.189883606,	Adjusted R-squared:  0.13401351
## F-statistic:  3.3986626 on 2 and 29 DF,  p-value: 0.0471995222
```

Repeat analyses for pups only

Empty Workspace to analyse pups isolated

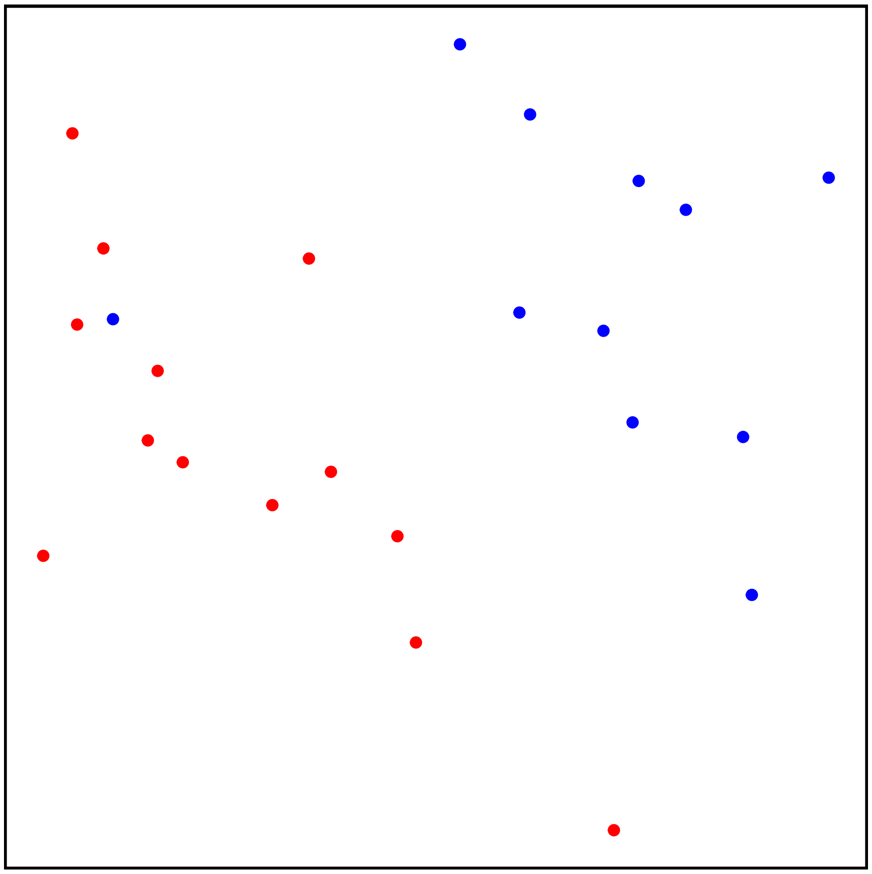
```
rm(list=ls())
```

Subset scent data to correlate same individuals

```
## read in meta data
meta <- read.table(file = "data/arga_metadata.txt", sep = "\t") %>%
  `colnames<-`(unlist(.[1,])) %>%
  .[-1,] %>%
  # subset for only pups
  filter(., maturity == "P")



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

```
## Run 0 stress 0.212491025469
## Run 1 stress 0.218956786124
## Run 2 stress 0.220554496683
## Run 3 stress 0.216633348678
## Run 4 stress 0.237815408693
## Run 5 stress 0.240483042564
## Run 6 stress 0.23567938312
## Run 7 stress 0.21895681306
## Run 8 stress 0.216772443295
## Run 9 stress 0.216633356099
## Run 10 stress 0.21663341619
## Run 11 stress 0.21895683149
## Run 12 stress 0.245675546245
## Run 13 stress 0.222517644162
## Run 14 stress 0.238043598229
## Run 15 stress 0.216633437494
## Run 16 stress 0.211374945833
## ... New best solution
## ... Procrustes: rmse 0.0907127311886  max resid 0.287770629192
## Run 17 stress 0.236799774697
## Run 18 stress 0.242361765569
## Run 19 stress 0.219797441423
## Run 20 stress 0.212490930928
## *** Best solution was not repeated -- monoMDS stopping criteria:
##     20: stress ratio > sratmax
```

```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
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")
```



Calculate MHC heterozygosity relatedness between individuals

```r
## 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,]

## 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 mark
##   Especially, rxy and ritland estimator are not defined when bi-allelic estimates are used with alle
##   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 oc
```

```
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 = dim(mhc_het_dat)[1],
                         ncol = dim(mhc_het_dat)[1])
## 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

```
# 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

```
# 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 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)

msats_df %<>% .[match(meta$real_id, .$id),]
```

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 swic
##   Be careful even if allele frequencies are not perfectly 0.5, during randomizations problems may occ
```

```
## Warning in prop.test(c(emp, non), c(sum(table(empirical.list)),
## sum(table(relate.non.X.mean))), : Chi-Quadrat-Approximation kann inkorrekt sein
```

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
```

```r
relate_mat <- matrix(nrow = dim(msats_df)[1],
                     ncol = dim(msats_df)[1])
## 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

```r
## 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 = 16,
                                       base_line_size = 1,
                                       base_rect_size = 1) +
  ggplot2::theme(
    # plot.margin = unit(c(5.5, 8.5, 5.5, 5.5), "pt"), #c(top, right, bottom, left)
    plot.margin = margin(5.5,6.5,5.5,5.5, "pt"),
    panel.grid = element_blank(),
    axis.text = element_text(color = "black"),
    axis.ticks = element_line(color = "black"),
    aspect.ratio = 1
  )
```

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
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

```r
## 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()

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] %>%
```

```r
  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))
```

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
```

```r
# 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")
smlh_res <- smlh_res[match(meta$real_id, rownames(smlh_res)), ] %>%
  # 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)
)
```

Repeat models for chemical similarity

```
# without age and indicator for family groupings
# as grouping factors must have >1 sampled level

# mhc
e1 <- lmerTest::lmer(scent_mds ~ ufrac + colonyBool + (1|pairID1) + (1|pairID2),
                     data = model_rel.df)
# relatedness
e2 <- lmerTest::lmer(scent_mds ~ rel + colonyBool + (1|pairID1) + (1|pairID2),
                     data = model_rel.df)
# mhc & relatedness
e3 <- lmerTest::lmer(scent_mds ~ rel + ufrac + colonyBool + (1|pairID1) + (1|pairID2),
                     data = model_rel.df)
# no genetic effect
e4 <- lmerTest::lmer(scent_mds ~ colonyBool + (1|pairID1) + (1|pairID2),
                     data = model_rel.df)


# compare model performance scores
compare_performance(e1, e2, e3, e4, rank = T) %>%
  arrange(Name)
```

```
## # Comparison of Model Performance Indices
##
## Name |           Model | R2 (cond.) | R2 (marg.) |   ICC |  RMSE | Sigma | AIC weights | AICc weights
## -----------------------------------------------------------------------------------------------------
## e1   | lmerModLmerTest |      0.517 |      0.188 | 0.404 | 0.058 | 0.062 |       0.287 |        0.28
## e2   | lmerModLmerTest |      0.525 |      0.186 | 0.417 | 0.058 | 0.062 |       0.207 |        0.20
## e3   | lmerModLmerTest |      0.520 |      0.191 | 0.407 | 0.058 | 0.062 |       0.228 |        0.21
## e4   | lmerModLmerTest |      0.522 |      0.184 | 0.414 | 0.058 | 0.062 |       0.278 |        0.29
```

```
summary(e2)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_mds ~ rel + colonyBool + (1 | pairID1) + (1 | pairID2)
##    Data: model_rel.df
##
## REML criterion at convergence: -669.2
##
## Scaled residuals:
##           Min            1Q        Median            3Q           Max
## -2.8734024086 -0.6551149100 -0.0127392185  0.5907406140  2.6919018000
##
## Random effects:
##  Groups   Name        Variance      Std.Dev.
##  pairID1  (Intercept) 0.00114373121 0.0338190954
##  pairID2  (Intercept) 0.00158812032 0.0398512273
##  Residual             0.00382618866 0.0618561934
## Number of obs: 276, groups:  pairID1, 23; pairID2, 23
##
## Fixed effects:
##                   Estimate     Std. Error             df  t value   Pr(>|t|)
## (Intercept)   0.2930925713   0.0159793884  29.8480662643 18.34191 < 2.22e-16
## rel           0.0634645257   0.0534097404 267.2637395374  1.18826    0.23579
## colonyBool1   0.0761977589   0.0136127102  51.9474648776  5.59755 8.2768e-07
##
## (Intercept) ***
## rel
## colonyBool1 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) rel
## rel          0.108
## colonyBool1 -0.639 -0.031
```

```r
summary(e4)
```

```
## Linear mixed model fit by REML. t-tests use Satterthwaite's method [
## lmerModLmerTest]
## Formula: scent_mds ~ colonyBool + (1 | pairID1) + (1 | pairID2)
##    Data: model_rel.df
##
## REML criterion at convergence: -671.8
##
## Scaled residuals:
##           Min            1Q        Median            3Q           Max
## -2.8754878831 -0.6724901336 -0.0272566346  0.5651130445  2.6590829271
##
## Random effects:
##  Groups   Name        Variance      Std.Dev.
##  pairID1  (Intercept) 0.00114074594 0.0337749306
##  pairID2  (Intercept) 0.00156895325 0.0396100145
##  Residual             0.00383649427 0.0619394403
## Number of obs: 276, groups:  pairID1, 23; pairID2, 23
##
## Fixed effects:
##                  Estimate    Std. Error            df  t value   Pr(>|t|)
## (Intercept)  0.2910051000  0.0158359198 29.1599058492 18.37627 < 2.22e-16 ***
## colonyBool1  0.0767098586  0.0135826001 52.0344825538  5.64766 6.8802e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr)
## colonyBool1 -0.640
```

Repeat models for chemical diversity

```r
# exclude family as random effects (no accounting of same family in different
# individuals)
# exlude age as only one 'age' group being moms is modelled


f1 <- lm(compound_n ~ mhc_het + colony , data = meta)

f2 <- lm(compound_n ~ smlh + colony, data = meta)

f3 <- lm(compound_n ~ mhc_het + smlh + colony, data = meta)

f4 <- lm(compound_n ~ colony, data = meta)


compare_performance(f1, f2, f3, f4, rank = T) %>% arrange(Name)
```

```
## # Comparison of Model Performance Indices
##
## Name | Model |    R2 | R2 (adj.) |   RMSE |  Sigma | AIC weights | AICc weights | BIC weights | Perf
## ----------------------------------------------------------------------------------------------------
## f1   |    lm | 0.055 |    -0.035 | 18.471 | 19.746 |       0.071 |        0.070 |       0.067 |
## f2   |    lm | 0.201 |     0.125 | 16.980 | 18.152 |       0.536 |        0.526 |       0.503 |
## f3   |    lm | 0.202 |     0.082 | 16.973 | 18.593 |       0.199 |        0.106 |       0.104 |
## f4   |    lm | 0.055 |     0.012 | 18.471 | 19.293 |       0.193 |        0.298 |       0.327 |
```

```r
summary(f2)
```

```
##
## Call:
## lm(formula = compound_n ~ smlh + colony, data = meta)
##
## Residuals:
##          Min           1Q       Median           3Q          Max
## -33.84774747 -10.72822530  -2.63667025  13.89024838  32.64888040
##
## Coefficients:
##                 Estimate   Std. Error  t value Pr(>|t|)
## (Intercept) -32.76603326  46.95021186 -0.69789  0.49290
## smlh         89.23084048  45.46613021  1.96258  0.06308 .
## colonySSB    -3.59128851   7.91958429 -0.45347  0.65486
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 18.1520925 on 21 degrees of freedom
## Multiple R-squared:  0.201400231,    Adjusted R-squared:  0.12534311
## F-statistic: 2.64801282 on 2 and 21 DF,  p-value: 0.0942879933
```

Session information

```r
sessionInfo()
```

```
## R version 4.3.1 (2023-06-16 ucrt)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 11 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
##
## time zone: Europe/Berlin
## tzcode source: internal
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] remotes_2.4.2.1    Demerelate_0.9-3   fts_0.9.9.2        zoo_1.8-12
##  [5] ggbeeswarm_0.7.2   pwr_1.3-0          partR2_0.9.1       MuMIn_1.47.5
##  [9] performance_0.10.4 ape_5.7-1          ggpubr_0.6.0       vegan_2.6-4
## [13] lattice_0.21-8     permute_0.9-7      inbreedR_0.3.3     GCalignR_1.0.5
## [17] phyloseq_1.44.0    lubridate_1.9.2    forcats_1.0.0      stringr_1.5.0
## [21] dplyr_1.1.2        purrr_1.0.1        readr_2.1.4        tidyr_1.3.0
## [25] tibble_3.2.1       ggplot2_3.4.2      tidyverse_2.0.0    magrittr_2.0.3
##
## loaded via a namespace (and not attached):
##   [1] rstudioapi_0.15.0      jsonlite_1.8.7         datawizard_0.8.0
##   [4] farver_2.1.1           nloptr_2.0.3           rmarkdown_2.23
##   [7] ragg_1.2.5             zlibbioc_1.46.0        vctrs_0.6.3
##  [10] multtest_2.56.0        minqa_1.2.5            RCurl_1.98-1.12
##  [13] rstatix_0.7.2          htmltools_0.5.5        broom_1.0.5
##  [16] Rhdf5lib_1.22.0        Formula_1.2-5          rhdf5_2.44.0
##  [19] plyr_1.8.8             sfsmisc_1.1-15         igraph_1.5.0.1
##  [22] lifecycle_1.0.3        iterators_1.0.14       pkgconfig_2.0.3
##  [25] Matrix_1.6-0           R6_2.5.1               fastmap_1.1.1
##  [28] GenomeInfoDbData_1.2.10 rbibutils_2.2.13      digest_0.6.33
##  [31] numDeriv_2016.8-1.1    colorspace_2.1-0       patchwork_1.1.2
##  [34] S4Vectors_0.38.1       textshaping_0.3.6      labeling_0.4.2
##  [37] fansi_1.0.4            timechange_0.2.0       abind_1.4-5
##  [40] mgcv_1.8-42            compiler_4.3.1         withr_2.5.0
##  [43] backports_1.4.1        carData_3.0-5          mlogit_1.1-1
##  [46] highr_0.10             ggsignif_0.6.4         MASS_7.3-60
##  [49] biomformat_1.28.0      tools_4.3.1            vipor_0.4.5
##  [52] lmtest_0.9-40          beeswarm_0.4.0         glue_1.6.2
##  [55] nlme_3.1-162           rhdf5filters_1.12.1    grid_4.3.1
##  [58] cluster_2.1.4          reshape2_1.4.4         ade4_1.7-22
##  [61] see_0.8.0              generics_0.1.3         gtable_0.3.3
##  [64] tzdb_0.4.0             data.table_1.14.8      hms_1.1.3
##  [67] car_3.1-2              utf8_1.2.3             XVector_0.40.0
##  [70] BiocGenerics_0.46.0    ggrepel_0.9.3          foreach_1.5.2
##  [73] pillar_1.9.0           splines_4.3.1          survival_3.5-5
##  [76] tidyselect_1.2.0       pbapply_1.7-2          Biostrings_2.68.1
##  [79] knitr_1.43             IRanges_2.34.1         stats4_4.3.1
##  [82] xfun_0.39              Biobase_2.60.0         statmod_1.5.0
##  [85] dfidx_0.0-5            stringi_1.7.12         yaml_2.3.7
##  [88] boot_1.3-28.1          evaluate_0.21          codetools_0.2-19
##  [91] cli_3.6.1              systemfonts_1.0.4      Rdpack_2.4
##  [94] munsell_0.5.0         Rcpp_1.0.11           GenomeInfoDb_1.36.1
##  [97] parallel_4.3.1        bayestestR_0.13.1     bitops_1.0-7
## [100] lme4_1.1-34           lmerTest_3.1-3        scales_1.2.1
## [103] insight_0.19.3        crayon_1.5.2          rlang_1.1.1
## [106] cowplot_1.1.1
```

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/.