An R script to generate Figs. 1 and 2.

Musa AA and Reinsch N

22/06/2021

import relevant packages

```
library(ggplot2)
library(reshape2)
library(corrplot)
require(RColorBrewer)
library(pheatmap)
library(ggpubr)
library(RcppCNPy)
```

Read saved data

```
msvmsc_g <- read.csv("msvmsc_g.csv", header = T)         # Mendelian (co-)variance
msvmsc_gcorr <- read.csv("msvmsc_gcorr.csv", header = T) # Mendelian correlation
```

1. DENSITY PLOTS

Extract relevant columns from data frame

```
df_msv <- data.frame(msvmsc_g[, c(3, 5, 8)])             # MS Variance
colnames(df_msv) <- c("FY", "pH", "PY")               # rename columns
head(df_msv)
```

```
##              FY          pH          PY
## 1 0.0004657436 0.001804908 0.002320987
## 2 0.0232617786 0.013224900 0.015098141
## 3 0.0216646755 0.010474963 0.013710154
## 4 0.0009494060 0.004835635 0.001351647
## 5 0.0218923149 0.012676235 0.013317761
## 6 0.0231965577 0.012756366 0.015687612
```

```
df_mscorr <- data.frame(msvmsc_gcorr[, 3:5])             # MS Correlation
colnames(df_mscorr) <- c("FY_pH", "FY_PY", "PY_pH")# rename columns
head(df_mscorr)
```

```
##       FY_pH       FY_PY       PY_pH
## 1 0.1682430  0.13035463 -0.07661897
## 2 0.8499381  0.92344974  0.78427147
## 3 0.9134121  0.90877876  0.82183644
## 4 0.1480978 -0.03885255 -0.06191213
## 5 0.8672485  0.91218654  0.77887446
## 6 0.8622727  0.90880696  0.77531606
```

Molt data frame for density plot

```
df_msv <- melt(df_msv)                                   # molted df for MS Variance
colnames(df_msv) <- c("Trait", "Variance")               # rename columns
df_mscorr <- melt(df_mscorr)                             # molted df for S Correlation
colnames(df_mscorr) <- c("Trait", "Correlation")         # rename columns
```

Create density plot for Mendelian sampling variance using ggplot

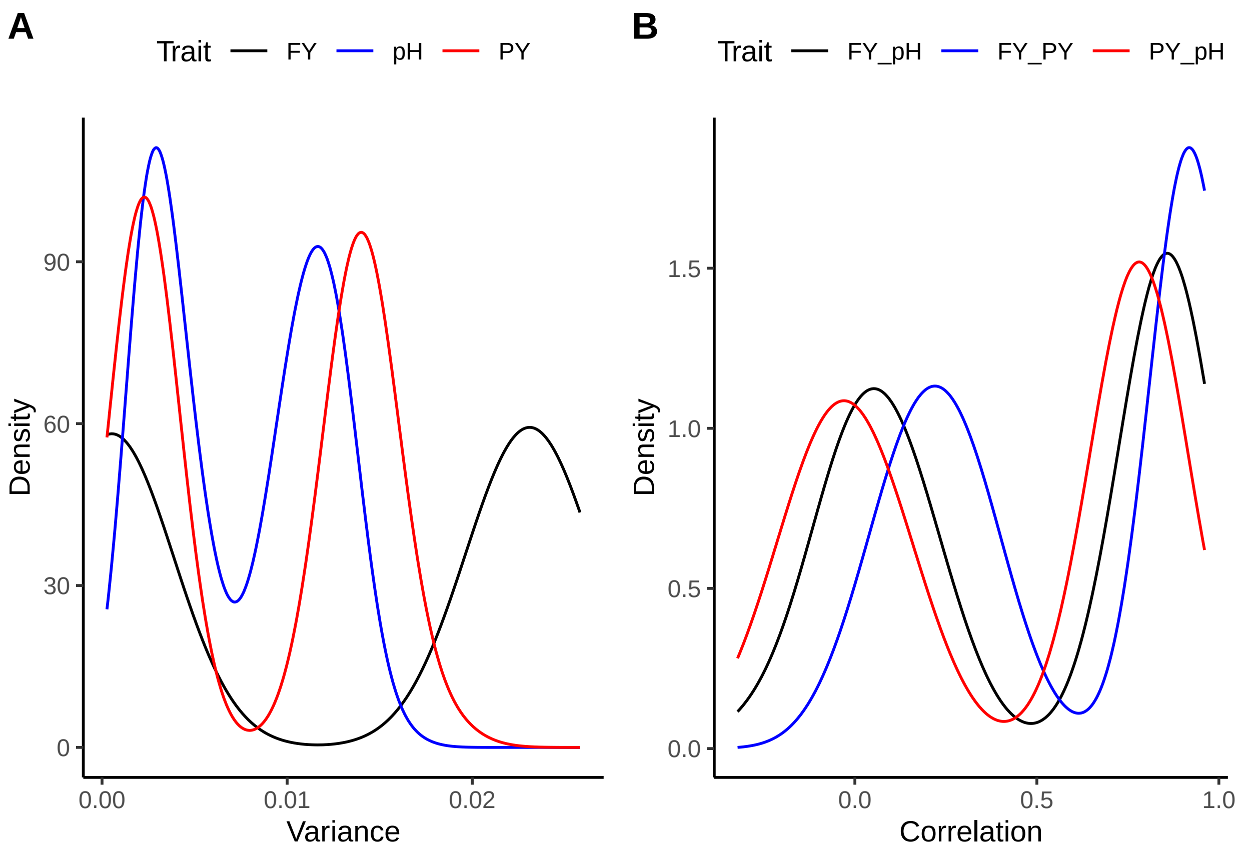
```
plot1 <- ggplot(df_msv, aes(Variance, color = Trait)) +
  stat_density(aes(x = Variance, colour = Trait),
               geom = "line",
               position = "identity") + theme_classic() +
  theme(legend.position = "top") + scale_color_manual(values = c("black", "blue", "red")) +
  ylab("Density")
```

Create density plot for Mendelian sampling correlation using ggplot

```
plot2 <- ggplot(df_mscorr, aes(Correlation, color = Trait)) +
  stat_density(aes(x = Correlation, colour = Trait),
               geom = "line",
               position = "identity") + theme_classic() +
  theme(legend.position = "top") + scale_color_manual(values = c("black", "blue", "red")) +
  ylab("Density")
```

Combine plots and save to file

```
# tiff("Density plot.tif", width = 8, height = 3, units = 'in', compression = 'lzw', res = 700)
ggarrange(plot1, plot2, labels = c("A", "B"), ncol = 2, nrow = 1)
```



```
# dev.off()
```

2. SIMILARITY MATRICES

Read saved data

```
ped <- read.csv("pedigree.csv", header = F)              # Pedigree
mat <- npyLoad("sim_g.npy")                              # Similarity matrix
# standardize similarity matrix
matstd <- cov2cor(mat)                                   # Standardized similarity matrix
# rename rows and columns of the matrices
row.names(mat) <- colnames(mat) <- 1:nrow(mat)
row.names(matstd) <- colnames(matstd) <- 1:nrow(matstd)
```

Within-family clustering of similarities

```
ind <- no <- NULL
for (i in 1:5) {
  first <- which(ped[, 1] == i)[1]                           # 1st ind in a family
  last <- which(ped[, 1] == i)[length(which(ped[, 1] == i))] # last ind in a family
  no <- c(no, length(which(ped[, 1] == i)))                  # no. of inds in a family
  m <- mat[first:last, first:last]                           # subset of matrix
  out <- pheatmap(m, show_rownames = T, cluster_cols = T,
    cluster_rows = T, cex = 1,
    clustering_distance_rows = "euclidean",
    clustering_distance_cols = "euclidean",
    clustering_method = "complete", border_color = F, silent = T)
  ind <- c(ind, (rownames(m[out$tree_row[["order"]], ])))
}
ind <- as.numeric(ind)
sim_clus <- mat[ind, ind]                                    # clustered inds within family
```

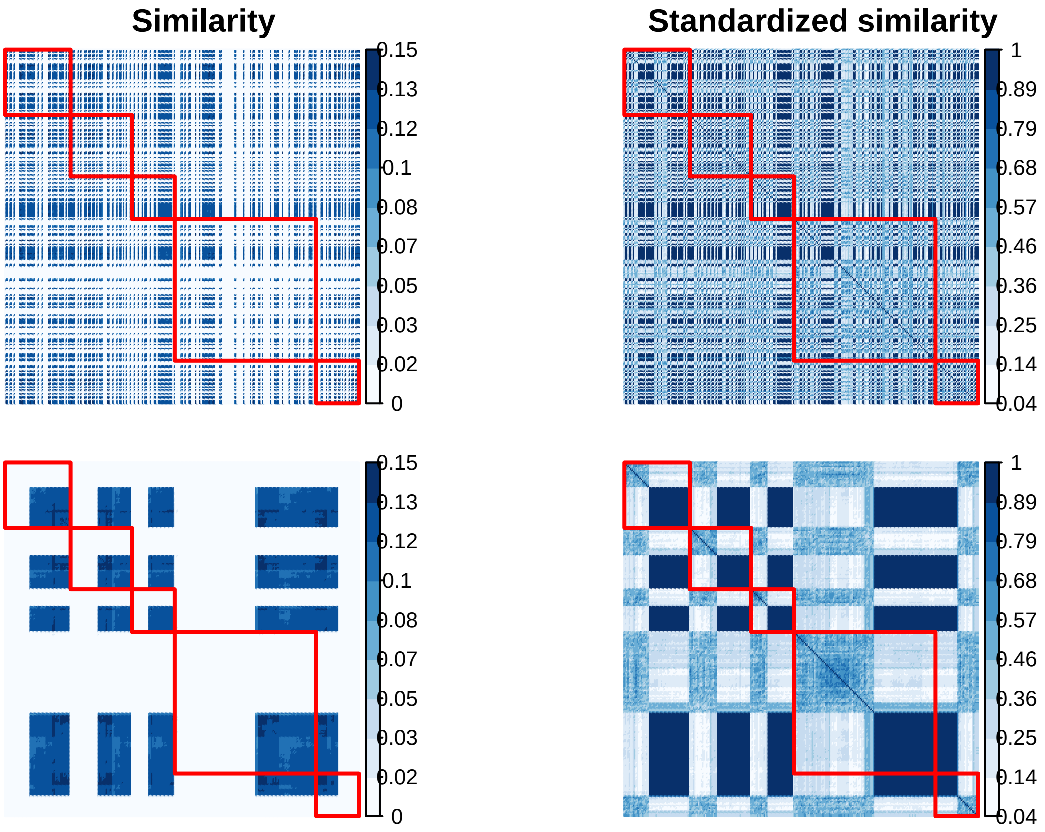
Within-family clustering of standardized similarities

```
ind <- no <- NULL
for (i in 1:5) { #for first two half-sib families only
  first <- which(ped[, 1] == i)[1]                           # 1st ind in a family
  last <- which(ped[, 1] == i)[length(which(ped[, 1] == i))] # last ind in a family
  no <- c(no, length(which(ped[, 1] == i)))                  # no. of inds in a family
  m <- matstd[first:last, first:last]                        # subset of matrix
  out <- pheatmap(m, show_rownames = T, cluster_cols = T,
    cluster_rows = T, cex = 1,
    clustering_distance_rows = "euclidean",
    clustering_distance_cols = "euclidean",
    clustering_method = "complete", border_color = F, silent = T)
  ind <- c(ind, rev(rownames(m[out$tree_row[["order"]], ]))) # reverse order of inds
}
ind <- as.numeric(ind)
stdsim_clus <- matstd[ind, ind]                              # clustered inds within family
```

Plot similarity matrices

```
# tiff("sim mats.tif", width = 9, height = 8, units = 'in', compression = 'lzw', res = 700)
par(mfrow = c(2, 2), oma = c(0, 0, 1, 0.1) + 0.1,
    mar = c(0, 0, 0, 0.5) + 0.1)
cols <- brewer.pal(9,"Blues")
corrplot(mat, is.corr = F, method = "color", cl.lim=range(mat),
         cl.cex = 0.80, tl.col = "black", tl.pos = "n", col = cols,
         cl.align.text = "c", mar = c(0, 0, 1, 0), title = "Similarity")
corrRect(no, col = "red") # draw diagonal boxes to demarcate families
corrplot(matstd, is.corr = F, method = "color", cl.lim=range(matstd),
         cl.cex = 0.80,tl.col = "black",tl.pos = "n", col = cols,
         cl.align.text = "c", mar = c(0, 0, 1, 0),
         title = "Standardized similarity")
corrRect(no, col = "red")
corrplot(sim_clus, is.corr = F, method = "color", cl.lim=range(sim_clus),
         cl.cex = 0.80,tl.col = "black",tl.pos = "n", col = cols,
         cl.align.text = "c", mar = c(0, 0, 1, 0))
corrRect(no, col = "red")
corrplot(stdsim_clus, is.corr = F, method = "color", cl.lim=range(stdsim_clus),
         cl.cex = 0.80,tl.col = "black",tl.pos = "n",
         col = cols, cl.align.text = "c", mar = c(0, 0, 1, 0))
corrRect(no, col = "red")

mtext('A', at=.75, side=2, outer=T, cex=1, las=0, line=0)
mtext('B', at=.25, side=2, outer=T, cex=1, las=0, line=0)
```



```
# dev.off()
```