Code for plotting Map, PCA, and UMAP

TLP

Code below sets chunk width so code wraps and doesn't run off the page

```
library(knitr)
opts_chunk$set(tidy.opts=list(width.cutoff=60),tidy=TRUE)
```

Here we are going to take population location information and a genotype matrix to plot a map, run and plot PCA, and run and plot UMAP

Loading necessary libraries

```
library(data.table)
library(ggplot2)
library(ggsci)
library(umap)
library(LEA)
library(readr)
library(ggpubr)
```

Function for running PCA, written by Trevor Faske

- PCA for 012 coded vcf files
- Following method in Patterson et al 2006

Input files:

`df_gen`: genotypic data with individuals as rows and snps as columns. Can include missing data. Either genotype probabilities or 012 format

Output:

`df_out`:
`$pca_df`: dataframe with rows as individuals and columns as PC1-X, Pop, ID
`$pve`: list of proportion of variance explained for each PC

Function:

```
PCA_gen <- function(df_gen, num = 10, tw = FALSE, tw_pvalue = 0.01) {

    df_gen <- apply(df_gen, 2, function(df) gsub(-1, NA, df,
        fixed = TRUE))
    df_gen <- apply(df_gen, 2, function(df) as.numeric(df))
    colmean <- apply(df_gen, 2, mean, na.rm = TRUE)

    normalize <- matrix(nrow = nrow(df_gen), ncol = ncol(df_gen))
    af <- colmean/2

    for (m in 1:length(af)) {
        nr <- df_gen[, m] - colmean[m]
        dn <- sqrt(af[m] * (1 - af[m]))
        normalize[, m] <- nr/dn
    }

    normalize[is.na(normalize)] <- 0

    method1 <- prcomp(normalize, scale. = FALSE, center = FALSE)
    pve <- summary(method1)$importance[2, ]
    print(pve[1:5])

    ### adjust number of PC axes ###

    if (nrow(df_gen) < num) {
        num <- nrow(df_gen)
    }

    #### Tracy Widom, PC axes ####
    if (tw == TRUE) {
        cat("\nRunning Tracy Widom test....\n\n")
        write.lfmm(normalize, "temp.lfmm")
        pca_tw <- pca("temp.lfmm", center = FALSE)
        tw <- tracy.widom(pca_tw)
        tw_sign <- tw$pvalues[tw$pvalues <= tw_pvalue]
        cat("\nNumber of TW sig. PC axes: ", length(tw_sign),
            "\n\n")
        num = length(tw_sign)
        unlink("temp.lfmm")
    }

    pca_X <- method1$x[, 1:num]

    pca_X <- as.data.frame(pca_X)

    pca_out <- list(pca_df = pca_X, pve = pve)

    return(pca_out)
}
```

EXAMPLE: All sampled populations of *Pinus muricata*

```
#### setwd ####
setwd("/Users/thomasparchman/Documents/GitHub/lab/parchman_sub/map_PCA_umap")

#### read in files ####
```

```r
g <- fread("PM_gl_matrix_miss30_maf05_noBadInds_noHighCov_noParalogs_noWeird.recode.csv",
    sep = ",", data.table = F)
g <- g[, -c(1:2)]

Pop_ID_Sum <- read.csv("PM_pop_ids.csv")

##### Run PCA ####
pca_out <- PCA_gen(g, tw = TRUE)
pve <- pca_out$pve[1:5]
pve
# PC1 PC2 PC3 PC4 PC5
g  #    0.07045 0.02406 0.01839 0.01187 0.01119

ncol(pca_out$pca_df)  # 14, number of tw PC axes

pca_df <- pca_out$pca_df
pca_df <- cbind(Pop_ID_Sum, pca_df)
```