SALS microsatellite population structure manuscript - breeding marsh landscape characteristics PCA

The below information contains a complete record of the data sources, input file generation and structure, analysis and Rcode, output data and/or figures generated, and session information I used to complete the above title-referenced analysis for publication.

The following can also be compiled in RStudio from the file SALSusat-EnvPCA.Rnw

Motivation: Presented below is a PCA to visualize the environmental differences among saltmarsh sparrow breeding marshes and the population clusters identified in this paper. Apparently this was done at some point by Jen Walsh, but was lost, so I have run it here in order to generate the final figure in vector format for publication; as well as to more fully document the data & analysis for posterity.

Add required packages:

```
> library(extrafont)
> library(ggbiplot)
```

Read in data*:

```
> ARHab <- read.table("~/Documents/ModernSparrowGenomics/Copies of Associated ELN
    files/SALS usat ms/Data and Analyses/Rhabitat4.txt", header = TRUE)
```

***Documentation of data sources and structure:**

1.) File Rhabitat4.txt.

Source: These data were generated in Genalex (verified using GenAlEx v. 6.51b2) based on the decimal degree coordinates for each marsh found on the 'Coords' tab in the SHARP Marsh Coords.xlsx file. I'm not sure who originally generated this file (dated July 29, 2016) or supplied the marsh coordinates, presumably Jen Walsh, Adrienne Kovach or maybe Bri Benvenuti. The output (GGD tab of above file) was generated using the Genalex options Distance -> Geographic... and options as shown below to calculate the pairwise distances according to the formula used by Genalex noted in the manual (Genalex 6.502 Appendix 1): "GenAlEx uses a modification of the Haversine Formula developed by R.W. Sinnott (Virtues of the Haversine (1984) Sky and Telescope 68,159) following computer code published online by Bob Chamberlain from JPL, NASA. (http://www.usenet-replayer.com/faq/comp.infosystems.gis.html still available on 12/12/12). Distances calculated via Lat/Long coordinators are returned in km." According to Wikipedia, the Haversine formula calculates great-circle distance, *not* Euclidean distance. This text file is a copy of the GGD tab, formatted for R (Genalex-specific formatting and column headers removed; blank fields in the upper triangle and self-pair distances of 0.000 changed to 'NA').

Data structure: Triangular matrix of pairwise geographic distance values.

Data units: Kilometers

Data snippet:

```
> head(ARHab, n = 5L)
```

```
        AR     Ne              Pop         Site Hybrid_Zone Patch_Size
1 7.168615  219.2 NJ-Long_IslandNY        OC-MW           0 5807.21753
2 7.016077 1632.9 NJ-Long_IslandNY          ATT           0 2174.62543
3 7.046231   93.6              SAW          SAW           0   87.52990
4 6.434769   11.1     Four_Sparrow Four_Sparrow           0   16.80693
5 6.598769   62.9              IDL          IDL           0   94.72314
     perim_m Sea_level_trend_mm Proportion_high_marsh
1 476101.767               3.99              0.438064
2 245159.557               3.99              0.199796
3  18507.434               2.77              0.298727
4   3302.812               2.77              0.181286
5  29036.173               2.77              0.221040
  Proportion_Natural_Lands_1000_m_buffer_around_patch  Ag_1000
1                                            0.361336 0.038586
2                                            0.381055 0.012047
3                                            0.380643 0.055066
4                                            0.362183 0.000000
```

```
5                                          0.142985 0.000000
  Proportion_Developed_Lands_1000_m_buffer_around_patch OpenW_1000 Marsh_1000
1                                             0.137132   0.462944   0.165333
2                                             0.122077   0.484819   0.077523
3                                             0.402336   0.161954   0.222026
4                                             0.470330   0.167485   0.086903
5                                             0.796739   0.060275   0.057732
  Proportion_Roads_1000_m_buffer_around_patch   X_Coord  Y_Coord
1                                    42.00075 -74.44168 39.46946
2                                    25.66446 -74.19568 39.71249
3                                    58.06001 -74.19110 40.61006
4                                    47.97375 -73.90542 40.60117
5                                    89.31336 -73.74525 40.64821
  Proportion_nonhigh_marsh Mean_High_Waterm Mean_High_Waterm.1 Proximity_Index
1                 0.561935             1.60               1.60      48.1441463
2                 0.800203             1.60               1.60       3.2063057
3                 0.701272             1.65               1.65       1.8916592
4                 0.818713             1.64               1.64       0.1040257
5                 0.778959             1.64               1.64       0.4395862
  Proximity_Index2 NEW_DISTANCE_TO_Atlantic_Coast_km
1     2073.8397580                             1.606
2       26.9582350                             2.222
3       95.2570008                            21.704
4        0.2548494                             4.831
5        4.2026605                             5.489
```

Run PCA:

```
> HAB.pca <- prcomp(ARHab[,c(8:14,21,23)], center = TRUE, scale. = TRUE)
```

Printout of results:

```
> HAB.pca
```

```
Mantel statistic based on Pearson's product-moment correlation

Call:
mantel(xdis = GeoDist, ydis = GenDist, method = "pearson", permutations = 10000,
        strata = NULL, na.rm = TRUE, parallel = getOption("mc.cores"))

Mantel statistic r: 0.2099
      Significance: 0.022298

Upper quantiles of permutations (null model):
  90%   95% 97.5%   99%
0.134 0.173 0.205 0.239
Permutation: free
Number of permutations: 10000
```

Format final figure for publication:

```
> rownames(HAB.pca$rotation) <- c("Sea Level Trend", "% High Marsh", "%
    Surrounding Natural Lands", "% Surrounding Ag Lands", "% Surrounding Developed
    ", "% Surrounding Open Water", "% Surrounding Marsh Lands", "Proximity Index",
     "Distance to Atlantic Coast")
> Pop.reorder <-  factor(ARHab$Pop, levels = c("NJ-Long_IslandNY", "SAW", "Four_
    Sparrow", "IDL", "CTMonomoy", "RI", "Great_MarshNHMA_Furb_Scarb", "GreatBayNH_
    Eldridge", "SouthernME"))
> labels <- c("OC-Mullica"="OC-Mullica", "ATT"="ATT", "Sawmill"="Sawmill", "Four_
    Sparrow"="Four Sparrow", "Idlewild"="Idlewild", "Marine_Nature_Center"="Marine
     Nature Center", "Long_Island"="Wertheim", "East_River"="East River", "
    Hammonasset"="Hammonasset", "Barn_Island"="Barn Island", "Chaffee"="Chafee", "
    Sachuest"="Sachuest", "Monomoy"="Monomoy", "Parker_River"="Parker River", "
    Hampton"="Hampton", "Fairhill"="Fairhill", "Chapman's_Landing"="Chapman's
    Landing", "Lubberland_Creek"="Lubberland Creek", "Furbish"="Furbish", "
    Eldridge"="Eldridge", "Little_River"="Little River", "Jones"="Jones", "
    Scarborough"="Scarborough", "Spurwink"="Spurwink")
> #How cool! Learned how to edit R programs/functions - see https://stackoverflow.
    com/questions/25995173/specifying-colour-transparency-and-position-of-arrows-
    line-segments-in-ggbipl
> #Get the function code, copy to text editor to edit as needed.
> #Assign function code with edits to new function:
> ggbiplot2 <- function (pcobj, choices = 1:2, scale = 1, pc.biplot = TRUE,
+       obs.scale = 1 - scale, var.scale = scale, groups = NULL,
+       ellipse = FALSE, ellipse.prob = 0.68, labels = NULL, labels.size = 3,
+       alpha = 1, var.axes = TRUE, circle = FALSE, circle.prob = 0.69,
+       varname.size = 3, varname.adjust = 1.5, varname.abbrev = FALSE, color = ("
    darkgray"),
+             linetype = "solid",
+             alpha_arrow = 1,
+       ...)
+  {
+       library(ggplot2)
+       library(plyr)
+       library(scales)
+       library(grid)
+       stopifnot(length(choices) == 2)
+       if (inherits(pcobj, "prcomp")) {
```

```
+         nobs.factor <- sqrt(nrow(pcobj$x) - 1)
+         d <- pcobj$sdev
+         u <- sweep(pcobj$x, 2, 1/(d * nobs.factor), FUN = "*")
+         v <- pcobj$rotation
+     }
+     else if (inherits(pcobj, "princomp")) {
+         nobs.factor <- sqrt(pcobj$n.obs)
+         d <- pcobj$sdev
+         u <- sweep(pcobj$scores, 2, 1/(d * nobs.factor), FUN = "*")
+         v <- pcobj$loadings
+     }
+     else if (inherits(pcobj, "PCA")) {
+         nobs.factor <- sqrt(nrow(pcobj$call$X))
+         d <- unlist(sqrt(pcobj$eig)[1])
+         u <- sweep(pcobj$ind$coord, 2, 1/(d * nobs.factor), FUN = "*")
+         v <- sweep(pcobj$var$coord, 2, sqrt(pcobj$eig[1:ncol(pcobj$var$coord),
+             1]), FUN = "/")
+     }
+     else if (inherits(pcobj, "lda")) {
+         nobs.factor <- sqrt(pcobj$N)
+         d <- pcobj$svd
+         u <- predict(pcobj)$x/nobs.factor
+         v <- pcobj$scaling
+         d.total <- sum(d^2)
+     }
+     else {
+         stop("Expected a object of class prcomp, princomp, PCA, or lda")
+     }
+     choices <- pmin(choices, ncol(u))
+     df.u <- as.data.frame(sweep(u[, choices], 2, d[choices]^obs.scale,
+         FUN = "*"))
+     v <- sweep(v, 2, d^var.scale, FUN = "*")
+     df.v <- as.data.frame(v[, choices])
+     names(df.u) <- c("xvar", "yvar")
+     names(df.v) <- names(df.u)
+     if (pc.biplot) {
+         df.u <- df.u * nobs.factor
+     }
+     r <- sqrt(qchisq(circle.prob, df = 2)) * prod(colMeans(df.u^2))^(1/4)
+     v.scale <- rowSums(v^2)
+     df.v <- r * df.v/sqrt(max(v.scale))
+     if (obs.scale == 0) {
+         u.axis.labs <- paste("standardized PC", choices, sep = "")
+     }
+     else {
+         u.axis.labs <- paste("PC", choices, sep = "")
+     }
+     u.axis.labs <- paste(u.axis.labs, sprintf("(%0.1f%% explained var.)",
+         100 * pcobj$sdev[choices]^2/sum(pcobj$sdev^2)))
+     if (!is.null(labels)) {
+         df.u$labels <- labels
+     }
+     if (!is.null(groups)) {
+         df.u$groups <- groups
+     }
+     if (varname.abbrev) {
+         df.v$varname <- abbreviate(rownames(v))
+     }
+     else {
+         df.v$varname <- rownames(v)
+     }
```

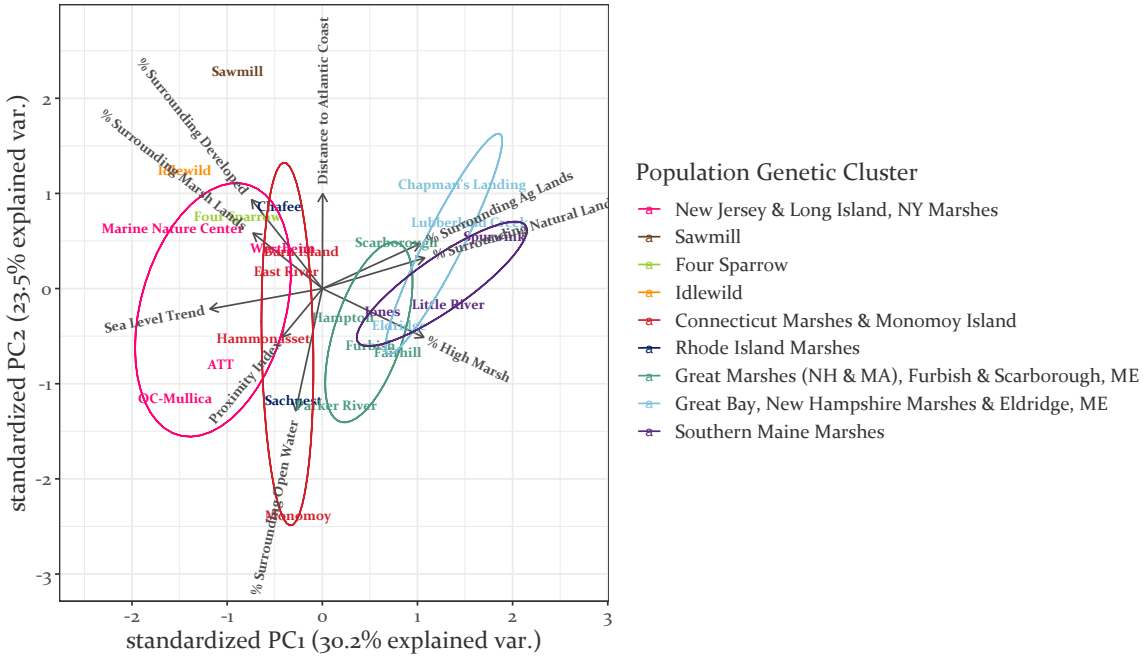
```
+      df.v$angle <- with(df.v, (180/pi) * atan(yvar/xvar))
+      df.v$hjust = with(df.v, (1 - varname.adjust * sign(xvar))/2)
+      g <- ggplot(data = df.u, aes(x = xvar, y = yvar)) + xlab(u.axis.labs[1]) +
+          ylab(u.axis.labs[2]) + coord_equal()
+      if (var.axes) {
+          if (circle) {
+              theta <- c(seq(-pi, pi, length = 50), seq(pi, -pi,
+                  length = 50))
+              circle <- data.frame(xvar = r * cos(theta), yvar = r *
+                  sin(theta))
+              g <- g + geom_path(data = circle, color = muted("white"),
+                  size = 1/2, alpha = 1/3)
+          }
+          g <- g + geom_segment(data = df.v, aes(x = 0, y = 0,
+              xend = xvar, yend = yvar), arrow = arrow(length = unit(1/2,
+              "picas")), color = color, linetype = linetype, alpha = alpha_arrow)
+      }
+      if (!is.null(df.u$labels)) {
+          if (!is.null(df.u$groups)) {
+              g <- g + geom_text(aes(label = labels, color = groups),
+                  size = labels.size)
+          }
+          else {
+              g <- g + geom_text(aes(label = labels), size = labels.size)
+          }
+      }
+      else {
+          if (!is.null(df.u$groups)) {
+              g <- g + geom_point(aes(color = groups), alpha = alpha)
+          }
+          else {
+              g <- g + geom_point(alpha = alpha)
+          }
+      }
+      if (!is.null(df.u$groups) && ellipse) {
+          theta <- c(seq(-pi, pi, length = 50), seq(pi, -pi, length = 50))
+          circle <- cbind(cos(theta), sin(theta))
+          ell <- ddply(df.u, "groups", function(x) {
+              if (nrow(x) <= 2) {
+                  return(NULL)
+              }
+              sigma <- var(cbind(x$xvar, x$yvar))
+              mu <- c(mean(x$xvar), mean(x$yvar))
+              ed <- sqrt(qchisq(ellipse.prob, df = 2))
+              data.frame(sweep(circle %*% chol(sigma) * ed, 2,
+                  mu, FUN = "+"), groups = x$groups[1])
+          })
+          names(ell)[1:2] <- c("xvar", "yvar")
+          g <- g + geom_path(data = ell, aes(color = groups, group = groups))
+      }
+      if (var.axes) {
+          g <- g + geom_text(data = df.v, aes(label = varname,
+              x = xvar, y = yvar, angle = angle, hjust = hjust),
+              color = color, size = varname.size)
+      }
+      return(g)
+  }
> #Apply new functionality!
> ##PC12A <- ggbiplot2(HAB.pca, ellipse=TRUE, labels=labels, groups=Pop.reorder,
    color = "blue", varname.adjust=1.1) + aes(family = "Constantia", fontface = "
    bold")
```

```
>
> ##PC12B <- PC12A + xlim(-2.15, 2.6) + ylim(-2.75, 2.5) + theme_bw() + scale_
    color_manual(name = "Population Genetic Cluster", labels = c("New Jersey &
    Long Island, NY Marshes", "Sawmill", "Four Sparrow", "Idlewild", "Connecticut
    Marshes & Monomoy Island", "Rhode Island Marshes", "Great Marshes (NH & MA),
    Furbish & Scarborough, ME", "Great Bay, New Hampshire Marshes & Eldridge, ME",
     "Southern Maine Marshes"), values = c("#FF0075", "#6C3E1E", "yellowgreen", "
    darkorange", "#D11F2A", "#002060", "#4A9B82", "#7FC1DB", "#57277C")) + theme(
    text = element_text(family = "Constantia", color = "grey20",size=15))
>
> PC12A <- ggbiplot2(HAB.pca,ellipse=TRUE, choices=c(1,2), labels=labels, groups=
    Pop.reorder, labels.size = 3, varname.size = 3, color = "grey30", varname.
    adjust=1.1, varname.color = "grey30")+ aes(family = "Constantia", fontface = "
    bold")
> PC12B <- PC12A + xlim(-2.5, 2.75) + ylim(-3, 2.7) + theme_bw() + scale_color_
    manual(name = "Population Genetic Cluster", labels = c("New Jersey & Long
    Island, NY Marshes", "Sawmill", "Four Sparrow", "Idlewild", "Connecticut
    Marshes & Monomoy Island", "Rhode Island Marshes", "Great Marshes (NH & MA),
    Furbish & Scarborough, ME", "Great Bay, New Hampshire Marshes & Eldridge, ME",
     "Southern Maine Marshes"), values = c("#FF0075", "#6C3E1E", "yellowgreen", "
    darkorange", "#D11F2A", "#002060", "#4A9B82", "#7FC1DB", "#57277C")) + theme(
    text = element_text(family = "Constantia", color = "grey20",size=15))
> PC13A <- ggbiplot2(HAB.pca,ellipse=TRUE, choices=c(1,3), labels=labels, groups=
    Pop.reorder, labels.size = 3, varname.size = 3, color = "grey30", varname.
    adjust=1.1, varname.color = "grey30")+ aes(family = "Constantia", fontface = "
    bold")
> PC13B <- PC13A + xlim(-2.5, 2.75) + ylim(-2.25, 3.25) + theme_bw() + scale_color
    _manual(name = "Population Genetic Cluster", labels = c("New Jersey & Long
    Island, NY Marshes", "Sawmill", "Four Sparrow", "Idlewild", "Connecticut
    Marshes & Monomoy Island", "Rhode Island Marshes", "Great Marshes (NH & MA),
    Furbish & Scarborough, ME", "Great Bay, New Hampshire Marshes & Eldridge, ME",
     "Southern Maine Marshes"), values = c("#FF0075", "#6C3E1E", "yellowgreen", "
    darkorange", "#D11F2A", "#002060", "#4A9B82", "#7FC1DB", "#57277C")) + theme(
    text = element_text(family = "Constantia", color = "grey20",size=15))
> var_explained_df <- data.frame(PC= paste0("PC",1:9), var_explained=(HAB.pca$sdev
    )^2/sum((HAB.pca$sdev)^2))
> Scree <- ggplot(var_explained_df, aes(x=PC, y=var_explained)) + geom_bar(stat =
    "identity") + theme_bw() + theme(text = element_text(family = "Constantia",
    color = "grey20",size=15)) + xlab("PC Axis") + ylab("Percent Variance
    Explained")
```

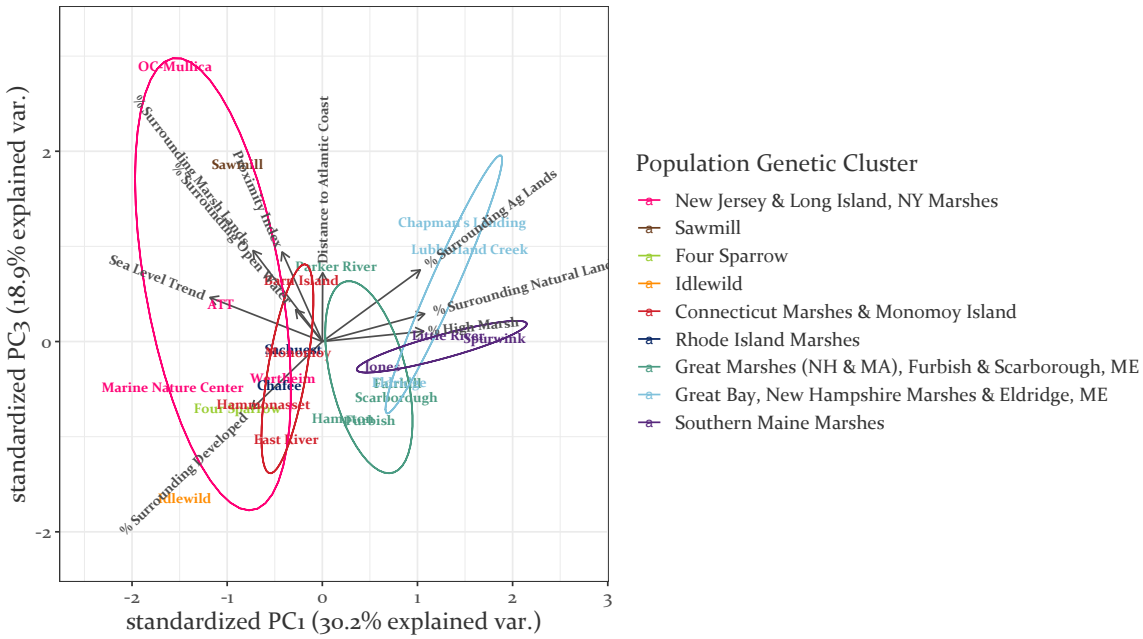
Save final figure:

```
> ggsave(file = "/Users/Lindsey/Documents/ModernSparrowGenomics/Copies of
    Associated ELN files/SALS usat ms/Final Figures/EnvPCAFinalPC12.pdf", plot =
    PC12B, device = cairo_pdf, width = 10, units = "in")
> ggsave(file = "/Users/Lindsey/Documents/ModernSparrowGenomics/Copies of
    Associated ELN files/SALS usat ms/Final Figures/EnvPCAFinalPC13.pdf", plot =
    PC13B, device = cairo_pdf, width = 10, units = "in")
> ggsave(file = "/Users/Lindsey/Documents/ModernSparrowGenomics/Copies of
    Associated ELN files/SALS usat ms/Final Figures/EnvPCAFinalScree.pdf", plot =
    Scree, device = cairo_pdf, width = 6.5, units = "in")
```

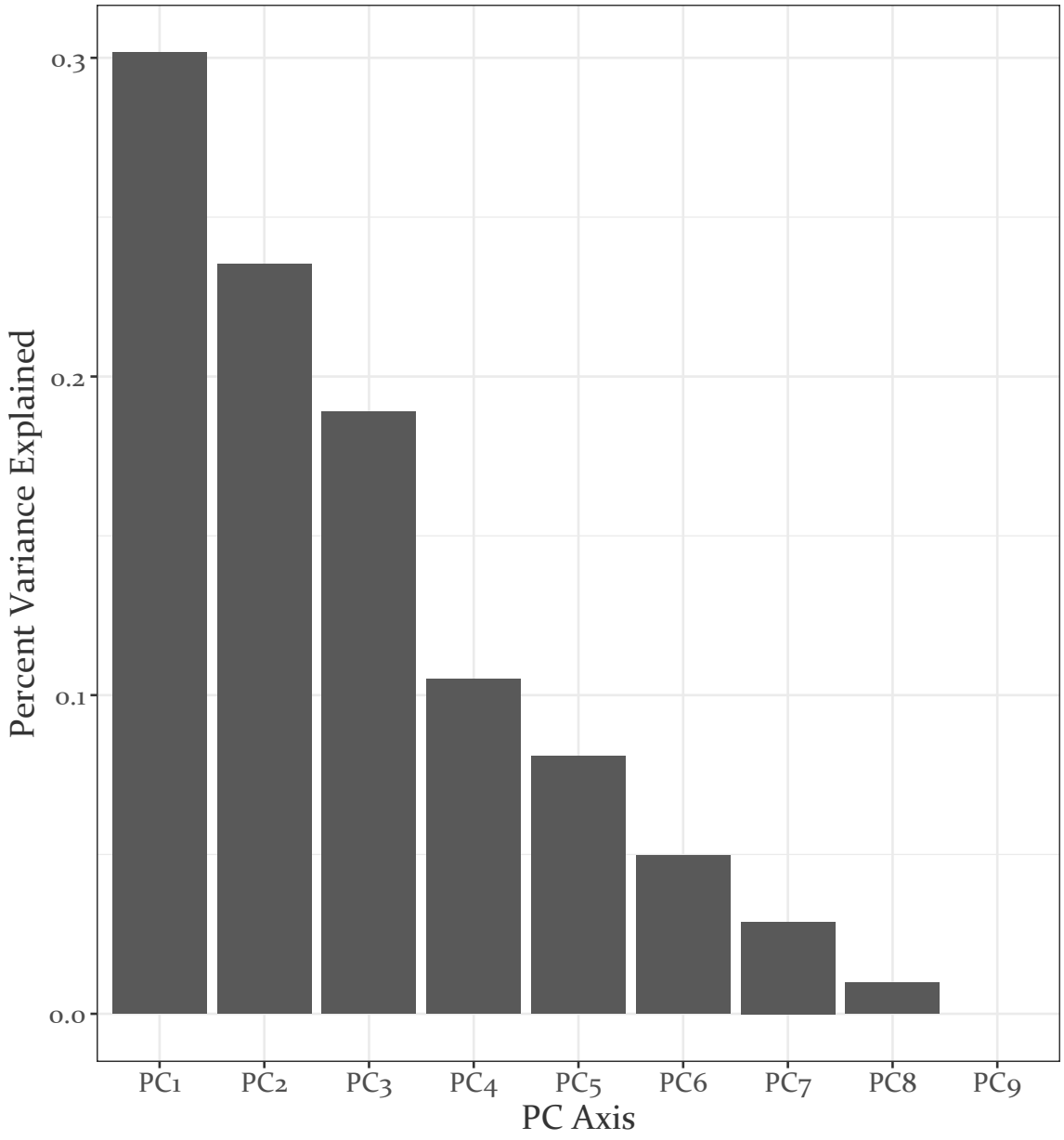
Copy of resulting EnvPCAFinalPC12.pdf plot:



Copy of resulting EnvPCAFinalPC13.pdf plot:



Copy of resulting EnvPCAFinalScree.pdf plot:


0.3
0.2
0.1
0.0
Percent Variance Explained
PC1
PC2
PC3
PC4
PC5
PC6
PC7
PC8
PC9
PC Axis

Record of Session Info:

```
> sessionInfo()

R version 4.0.3 (2020-10-10)
Platform: x86_64-apple-darwin17.0 (64-bit)
Running under: macOS Big Sur 10.16

Matrix products: default
BLAS:   /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRblas.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRlapack.dylib

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] grid      stats     graphics  grDevices utils     datasets  methods
[8] base

other attached packages:
[1] ggbiplot_0.55  scales_1.1.1   plyr_1.8.6     ggplot2_3.3.5  extrafont_0.17

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.8       rstudioapi_0.13  Rttf2pt1_1.3.9   magrittr_2.0.2
 [5] tidyselect_1.1.0 munsell_0.5.0    colorspace_2.0-3 R6_2.5.1
 [9] rlang_1.0.1      fansi_1.0.2      dplyr_1.0.2      tools_4.0.3
[13] gtable_0.3.0     utf8_1.2.2       cli_3.2.0        withr_2.4.3
[17] extrafontdb_1.0  ellipsis_0.3.2   digest_0.6.29    tibble_3.1.6
[21] lifecycle_1.0.1  crayon_1.5.0     farver_2.1.0     purrr_0.3.4
[25] vctrs_0.3.8      glue_1.4.2       labeling_0.4.2   compiler_4.0.3
[29] pillar_1.7.0     generics_0.1.2   pkgconfig_2.0.3
```