End Member Mixing Analysis

Performed with EMMAgeo in R

Author: Andrew Macumber

Table of Contents

Prep Workspace

Remove previous lists from R

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

Load libraries

```
library("EMMAgeo")
```

```
##
## EMMAgeo v. 0.9.6. When using this package please cite:
##
## Dietze, E., and Dietze, M.: Grain-size distribution unmixing using the R package EMMAgeo, E&G Quatern
##
## Dietze E, Hartmann K, Diekmann B, IJmker J, Lehmkuhl F, Opitz S, Stauch G, Wuennemann B, Borchers A.
##
## (Don't forget to mention the package and R version.)
```

```
library("functional")
```

Load Data

Bring in the dataset

```
psa_v000 <- read.table('PSA_Peroxide.csv', sep = ",")
```

Check dataframe (optional)

```
dim(psa_v000)
head(psa_v000)
```

Data Wrangling

Remove one row and two column headers

```
psa_v001 <- psa_v000[c(-1, -18), c(-c(1:3),-c(94:96))]
```

Check for grain size bins (columns) with zero sum

```
colSums(psa_v001)
```

```
##  V4  V5  V6  V7  V8  V9 V10 V11 V12 V13 V14 V15 V16 V17 V18 V19 V20 V21
##  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA
## V22 V23 V24 V25 V26 V27 V28 V29 V30 V31 V32 V33 V34 V35 V36 V37 V38 V39
##  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA
## V40 V41 V42 V43 V44 V45 V46 V47 V48 V49 V50 V51 V52 V53 V54 V55 V56 V57
##  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA
## V58 V59 V60 V61 V62 V63 V64 V65 V66 V67 V68 V69 V70 V71 V72 V73 V74 V75
##  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA
## V76 V77 V78 V79 V80 V81 V82 V83 V84 V85 V86 V87 V88 V89 V90 V91 V92 V93
##  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA  NA
```

Remove NULL values

```
psa_v002 <- na.omit(psa_v001)
```

Convert to double matrix

```
psa_v003 <- as.matrix(psa_v002)
```

Ensure rows sum to 100

```
psa_v004 <- psa_v003 / apply(psa_v003,1,sum) * 100
```

Check dataframe (optional)

```
dim(psa_v004)
head(psa_v004)
```

Data Objects

Create a grain size vector

```
mm_bins <- as.numeric(psa_v000[1,c(-c(1:3),-c(94:96))])
```

Create a depth vector

```
depth <- as.vector(psa_v000[c(-1,-2,-3,-6,-15,-18,-27,-29,-38,-39,-41,-42,-43,-51,-77,-99),1])
```

Assign an emma object

```
emma <- psa_v004
```

Graph: Grain Size Frequency Distribution

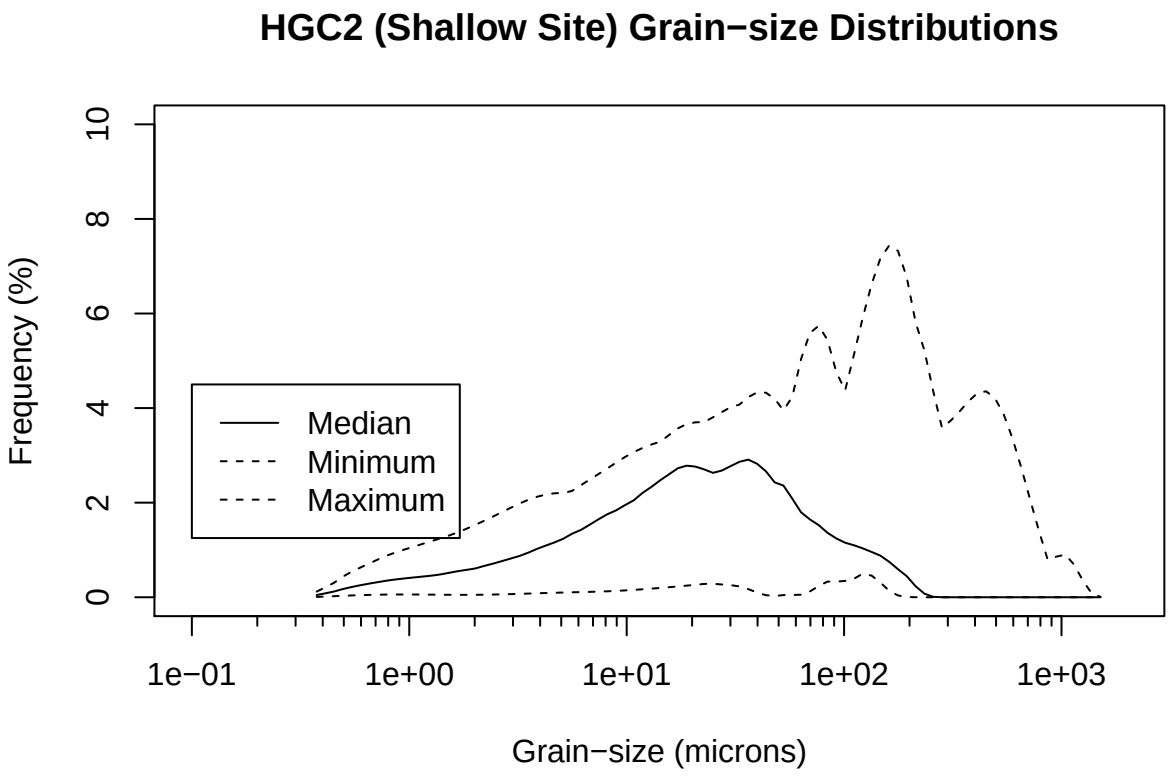
Generate a plot of median, minimum and maximum grain size (log) versus frequency (%)

Create a dataframe with grain size (column) mins, max and medians

```
library("matrixStats")
psa.range <- cbind(mm_bins,
                   colMins(emma, na.rm=TRUE),
                   colMaxs(emma, na.rm=TRUE),
                   colMedians(emma, na.rm=TRUE))
```

Create a graph showing the median, minimum and maximum grain size frequencies

```
plot(psa.range[,1],psa.range[,4], type ="l", log = "x", # plot grain-size (log) versus median
     xlim = c(0.1,2000), ylim = c(0,10), # set the axes limits
     col ="black", xaxt = "n", # colour of the line and suppress x-axis
     xlab = "Grain-size (microns)",
     ylab = "Frequency (%)",
     main = "HGC2 (Shallow Site) Grain-size Distributions") # labels
lines(psa.range[,1],psa.range[,2], col = "black", lty = 2) # add a line for minimum (grey-dash)
lines(psa.range[,1],psa.range[,3], col = "black", lty = 2) # add a line for maximum (grey-dash)
axis (side=1, at = c(0.1,1,10,100,1000)) # add major tick marks
axis (side=1, at = c(seq(0.2,0.9,by = 0.1), seq(2,9, by =1), # add minor tick marks
                     seq(20,90, by = 10), seq(200,900,by=100)),
      labels = NA, tcl=-0.25, lwd=0, lwd.ticks=1) # no labels and shorten tick marks
legend(0.1,4.5,c("Median","Minimum","Maximum"), # add a legend at specific spot with labels
       lty=c(1,2,2), col=c("black","black","black")) # ensure same as lines in plot
```



EMMA: Model Parameters

Check your dataset for errors.

```
check.data(X = emma , q = 10, l = seq(0, 1, 0.01), c = 100)
```

```
## [1] "Data matrix passed test... OK"
## [2] "End-member vector passed test... OK"
## [3] "Weight transformation limit vector passed test... OK"
## [4] "Scaling parameter passed test... OK"
## [5] "NA-test passed... OK"
## [6] "Test for zero-only values passed... OK"
## [7] "    Note: weight transformation limit(s) are out of range. Maximum value is 0.02"
## [8] "All samples sum up to constant sum... OK"
```

Create a sequence of weight transformations (l)

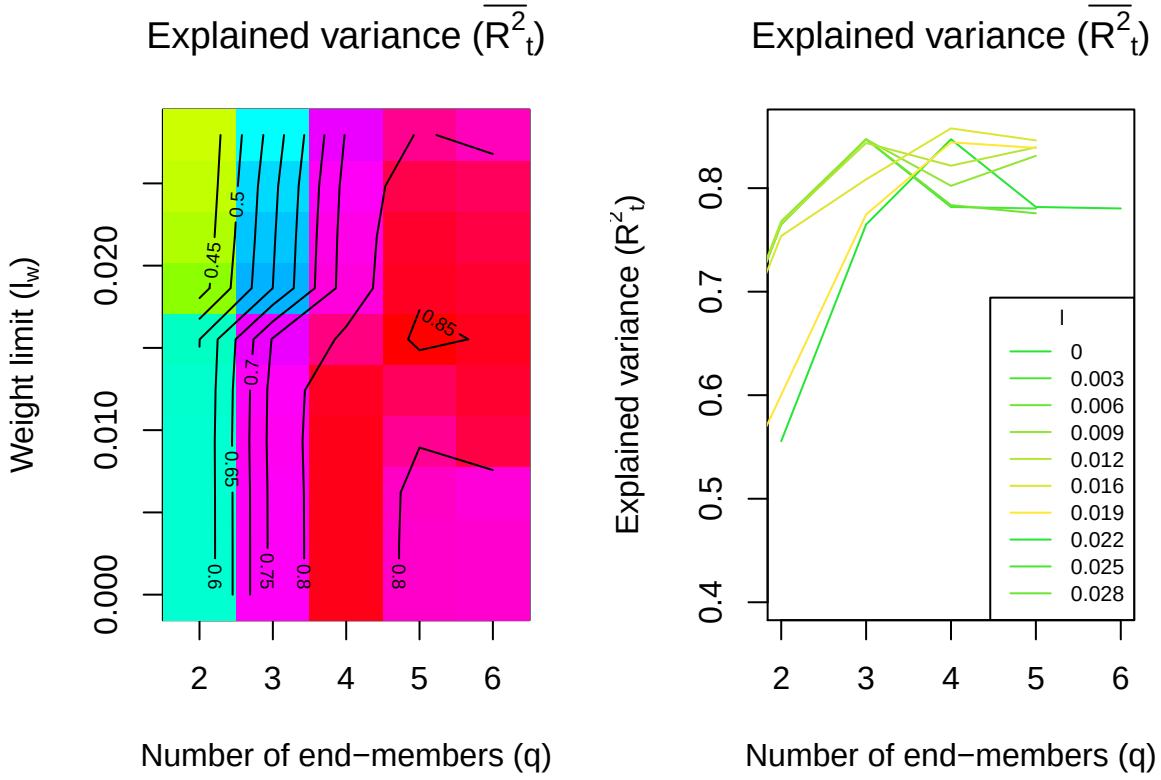
```
l <- get.l(X = emma, n = 10, max = 0.95, min = 0)
```

Combine the range of end members and weight transformations (Q). Create a range of end members only (q)

```
Q <- get.q(X = emma, l = l)
q <- 2:6
```

See a graph showing how weight transformation (l) varies with number of end Members

```
EM.tp <- test.parameters(X = emma, q = q, l = l, plot = "mRt",
                         legend = "bottomright", cex = 0.7,
                         multicore = FALSE,
                         colour = rgb((1:7) / 7, 0.9, 0.2, 1)
                         )
```



See a table showing how weight transformation (l) varies with number of end-members

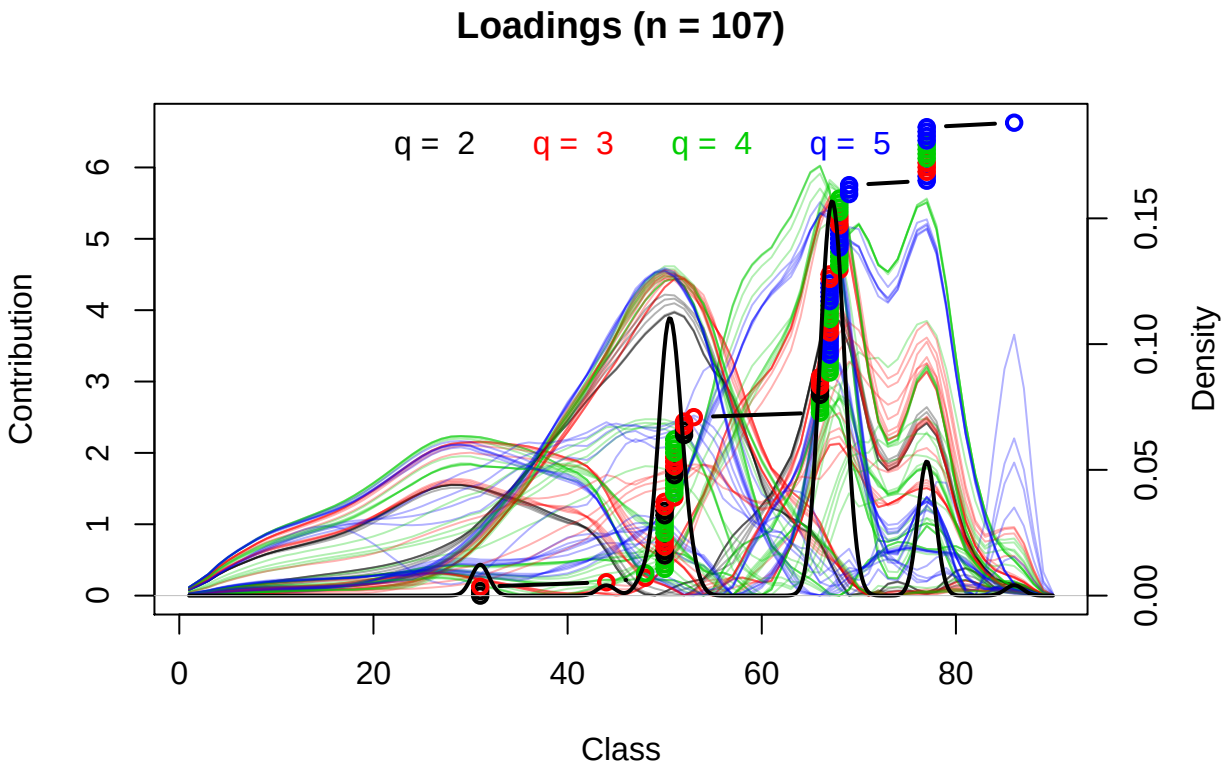
```
EM.tp$mRt
```

```
##           0 0.00310457515182371 0.00620915030364741 0.00931372545547112
## 2 0.5555440           0.5555440           0.5557328           0.5559270
## 3 0.7649179           0.7649179           0.7653085           0.7680057
## 4 0.8471807           0.8471807           0.8473190           0.8471425
## 5 0.7817873           0.7817873           0.7835755           0.8022342
## 6 0.7803654           0.7803654           0.7756399           0.8311790
##   0.0124183006072948 0.0155228757591185 0.0186274509109422
## 2          0.5534594          0.5494117          0.4261947
## 3          0.7662134          0.7537822          0.6002804
## 4          0.8435341          0.8084498          0.7745527
```

```
## 5          0.8216908          0.8577023          0.8442327
## 6          0.8394078          0.8461027          0.8388877
##   0.021732026062766 0.0248366012145897 0.0279411763664134
## 2          0.4127676          0.4048466          0.4009334
## 3          0.5940486          0.5870033          0.5721732
## 4          0.7716211          0.7660430          0.7539492
## 5          0.8398002          0.8296871          0.8038257
## 6          0.8315690          0.8215965          0.7872191
```

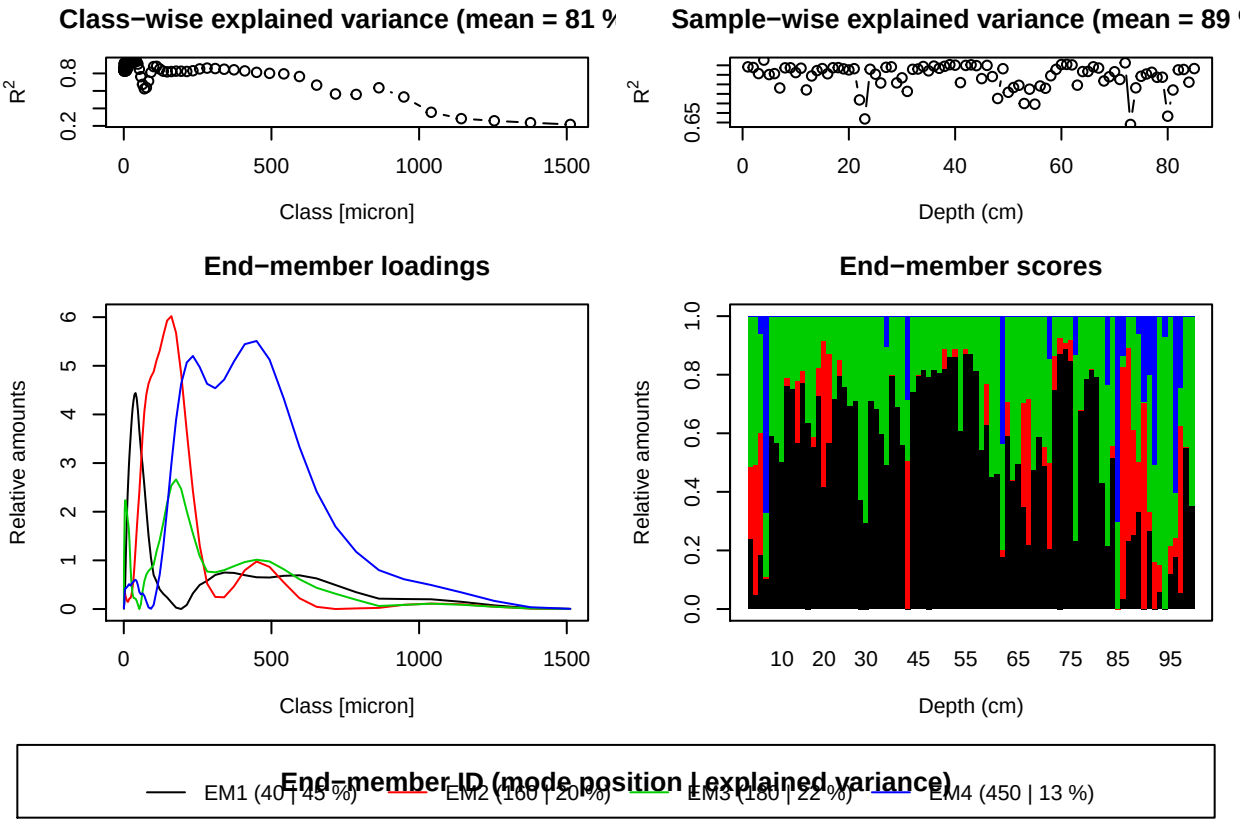
EMMA: Run All Models

```
EM.pot <- model.EM(X = emma, q = Q)
```



EMMA: Run Specific Model

```
EM.opt <- EMMA(X = emma,
               q = 4, l = 0.006,
               c = 100,
               plot = TRUE,
               classunits = mm_bins,
               ID = as.numeric(depth),
               xlab = c(expression(paste("Class [", micron, "]")), "Depth (cm)"))
```



EMMA: Run Robust Model

Get the bin size ranges for end members (i.e., limits)

```
# Visualize limits
stem(EM.pot$modes, scale = 2)
```

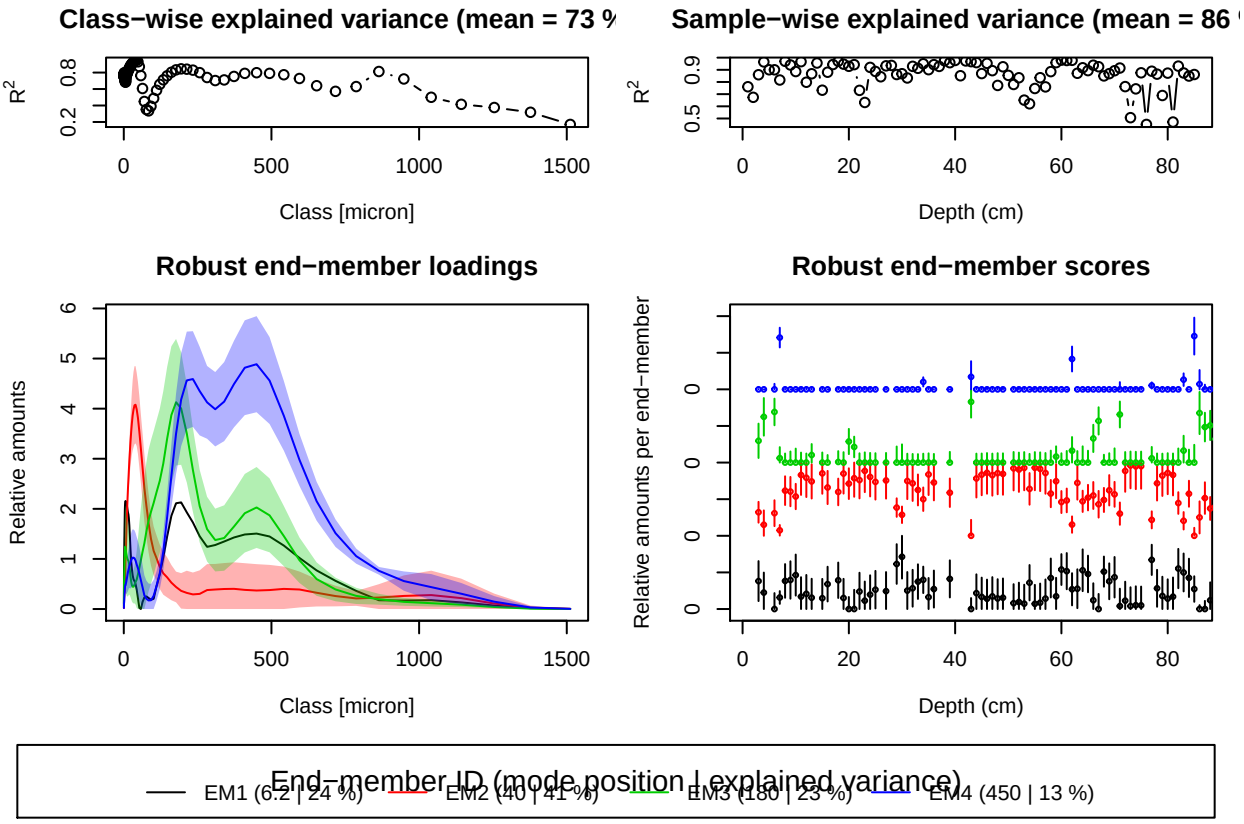
```
##
##   The decimal point is at the |
##
##   30 | 000
##   32 |
##   34 |
##   36 |
##   38 |
##   40 |
##   42 |
##   44 | 0
##   46 |
##   48 | 00
##   50 | 00000000000000000000000000000000
##   52 | 00000
##   54 |
##   56 |
##   58 |
```

```
##   60 |
##   62 |
##   64 |
##   66 | 00000000000000000000000000000000
##   68 | 0000000000000000000
##   70 |
##   72 |
##   74 |
##   76 | 0000000000000
##   78 |
##   80 |
##   82 |
##   84 |
##   86 | 0
```

```
# Assign limits
v1 <- c(28,48,64,74)
v2 <- c(32,54,68,78)
limits = cbind(v1,v2)
```

Run robust model

```
EM.rob <- robust.EM(em = EM.pot
                    , l = 0.006
                    , limits = limits
                    , c = 100
                    , plot = TRUE
                    , classunits = mm_bins
                    , ID = as.numeric(depth)
                    , xlab = c(expression(paste("Class [", micron, "]")), "Depth (cm)")
                    )
```



EMMA: Results

We tested 107 models to find the initial parameters. The mean total explained variance:

- For all models: 0.8577023
- For the optimal model: 0.8484582
- For the robust model: 0.7970022

The optimal model had the following end member modes: 39.7813, 161.177, 176.935, 449.672 each explaining 44.6805146, 20.3727398, 21.6014092, 13.3453364 of the variance, respectively.

The robust model had the following end member modes: 6.15902, 39.7813, 176.935 and 449.672. Each explaining 23.9468218, 40.6538317, 22.7753838, 12.6239627 of the variance, respectively.

Save Data

Create a dataframe with robust end member mean abundance and standard deviation.

```
EM.rob.mean <- EM.rob$scores$mean
EM.rob.sd <- EM.rob$scores$sd
EM.rob.scores <- cbind(EM.rob.mean, EM.rob.sd)
colnames(EM.rob.scores) <- c("EM_06_Mean", "EM_40_Mean", "EM_177_Mean", "EM_450_Mean",
                             "EM_06_SD", "EM_40_SD", "EM_177_SD", "EM_450_SD")
row.names(EM.rob.scores) <- depth
```

Export File

```
write.csv(EM.rob.scores, file ="Chap01_EMMA_Q4_scores.csv")
```

Graph: Robust End Member Distributions

This figure will show the average grain size distribution with standard deviations for each robust end member. Each end member will be plotted on its on graph. The grain size distributions of all samples will be displayed as grey lines on each graph. *Note: Creates Figure 7 of Chapter 1. Markdown creates an improperly formatted graph. Not run here*

Format sample data

```
dietze <- rbind(mm_bins, emma)
dietze <- t(dietze)
```

Format robust end member means

```
robust <- rbind(mm_bins, EM.rob$loadings$mean)
robust <- t(robust)
```

Format robust end member standard deviations

```
robust.sd <- t(EM.rob$loadings$sd)
```

Plot the graph

```
plot.new()

# Design the layout of the graph
layout(matrix(c(1,1,2,2,3,3,4,4), 4, 2, byrow = TRUE),
       widths=c(3,3,3,3), heights=c(2,2,2,2))

# First Graph

## Samples
plot(dietze[,1],dietze[,2], type = "l", log ="x", ylim = c(0,10)
     , xlim = c(0.1,2000),col="grey"
     , xaxt="n", xlab = ""
     , ylab = "Frequency (%)")
for (i in 3:ncol(dietze)) lines(dietze[,1],dietze[,i],col="grey") # plot for every column

## The End Member
lines(robust[,1],robust[,2],type="l",col="black",lty = 1, lwd = 2) # mean
lines(robust[,1],robust[,2]-robust.sd[,1],type="l",col="black",lty = 2, lwd = 2) # lower standard devia
lines(robust[,1],robust[,2]+robust.sd[,1],type="l",col="black",lty = 2, lwd = 2)

## Graph Details
axis (side=1, at = c(0.1,1,10,100,1000), labels = NA) # add major tick marks
axis (side=1, at = c(seq(0.2,0.9,by = 0.1), seq(2,9, by =1), # add minor tick marks
                     seq(20,90, by = 10), seq(200,900,by=100))
      , labels = NA, tcl=-0.25, lwd=0, lwd.ticks=1) # no labels and shorten tick marks
```

```
legend(200,9,c("Samples","End Member Mean","End Member St. Deviation"), # add a legend at specific spot
       lty=c(1,1,2), col=c("grey","black","black")) # ensure same as lines in plot
text(0.1, 8, adj = 0, "Fine Silt End Member (6 microns) Explaining 24% of Variance", cex = 1.5)
title(main = "Robust End Members of Harvey Lake Shallow Core (HGC2)", cex.main = 2)


# Second Graph

## Samples
plot(dietze[,1],dietze[,2], type = "l", log ="x", ylim = c(0,10),
     xlim = c(0.1,2000),col="grey", xaxt="n", xlab = "",
     ylab = "Frequency (%)")
for (i in 3:ncol(dietze)) lines(dietze[,1],dietze[,i],col="grey") # plot for every column

## The End Member
lines(robust[,1],robust[,3],type="l",col="black", lty = 1, lwd = 2)
lines(robust[,1],robust[,3]-robust.sd[,2],type="l",col="black", lty = 2, lwd = 2) # upper standard devi
lines(robust[,1],robust[,3]+robust.sd[,2],type="l",col="black", lty = 2, lwd = 2)

## Graph Details
axis (side=1, at = c(0.1,1,10,100,1000), labels = NA) # add major tick marks
axis (side=1, at = c(seq(0.2,0.9,by = 0.1), seq(2,9, by =1), # add minor tick marks
                     seq(20,90, by = 10), seq(200,900,by=100))
      , labels = NA, tcl=-0.25, lwd=0, lwd.ticks=1) # no labels and shorten tick marks
legend(200,9,c("Samples","End Member Mean","End Member St. Deviation"), # add a legend at specific spot
       lty=c(1,1,2), col=c("grey","black","black")) # ensure same as lines in plot
text(0.1, 8, adj = 0, "Very Coarse Silt End Member (40 microns) Explaining 42% of Variance", cex = 1.5)


# Third Graph

## Samples
plot(dietze[,1],dietze[,2], type = "l", log ="x", ylim = c(0,10),
     xlim = c(0.1,2000),col="grey", xaxt="n", xlab = "",
     ylab = "Frequency (%)")
for (i in 3:ncol(dietze)) lines(dietze[,1],dietze[,i],col="grey") # plot for every column

## The End Member
lines(robust[,1],robust[,4],type="l",col="black",lty = 1, lwd = 2)
lines(robust[,1],robust[,4]-robust.sd[,3],type="l",col="black",lty = 2, lwd = 2)
lines(robust[,1],robust[,4]+robust.sd[,3],type="l",col="black",lty = 2, lwd = 2)

## Graph Details
axis (side=1, at = c(0.1,1,10,100,1000), labels = NA) # add major tick marks
axis (side=1, at = c(seq(0.2,0.9,by = 0.1), seq(2,9, by =1), # add minor tick marks
                     seq(20,90, by = 10), seq(200,900,by=100))
      , labels = NA, tcl=-0.25, lwd=0, lwd.ticks=1) # no labels and shorten tick marks
legend(200,9,c("Samples","End Member Mean","End Member St. Deviation"), # add a legend at specific spot
       lty=c(1,1,2), col=c("grey","black","black")) # ensure same as lines in plot
text(0.1, 8, adj = 0, "Fine Sand End Member (177) microns) Explaining 22% of Variance", cex = 1.5)


# Fourth Graph
```

```r
## Samples
plot(dietze[,1],dietze[,2], type = "l", log ="x", ylim = c(0,10),
     xlim = c(0.1,2000),col="grey", xaxt="n", xlab = "",
     ylab = "Frequency (%)")
for (i in 3:ncol(dietze)) lines(dietze[,1],dietze[,i],col="grey") # plot for every column

## The End Member
lines(robust[,1],robust[,5],type="l",col="black",lty = 1, lwd = 2)
lines(robust[,1],robust[,5]-robust.sd[,4],type="l",col="black",lty = 2, lwd = 2)
lines(robust[,1],robust[,5]+robust.sd[,4],type="l",col="black",lty = 2, lwd = 2)

## Graph Details
axis (side=1, at = c(0.1,1,10,100,1000)) # add major tick marks
axis (side=1, at = c(seq(0.2,0.9,by = 0.1), seq(2,9, by =1), # add minor tick marks
                     seq(20,90, by = 10), seq(200,900,by=100))
      , labels = NA, tcl=-0.25, lwd=0, lwd.ticks=1) # no labels and shorten tick marks
legend(200,9,c("Samples","End Member Mean","End Member St. Deviation"), # add a legend at specific spot
       lty=c(1,1,2), col=c("grey","black","black")) # ensure same as lines in plot
text(0.1, 8, adj = 0, "Medium Sand End Member (58 microns) Explaining 9% of Variance", cex = 1.5)
title(xlab = "Grain size (microns)")
```