Exploratory analysis of RNA-seq dataset

gg

2016-01-27

To run this file: Rscript -e "rmarkdown::render('biplots_filter.Rmd')"

Setup

```
library(compositions)
library(zCompositions)
library(ALDEx2)
# you can get this from:
# https://github.com/DavidRLovell/propr
source("~/git/proprBayes/R/propr-functions.R")

library(igraph)

################ incorporating phi into ALDEx
# calculate phi on Dirichlet log-ratio distributions
# returns dataframe of the lower-triangle of symmetrical phi metric
# requires David Lovell's propr.phisym function
#
# THIS IS NOW IN THE proprBayes source
#
############# end functions
```

This will generate biplots with various things painted on top for use as figures, and will serve as a supplement for how the data was processed.

Filter the refseq to a mean count across samples of greater than 5, and remove outlier samples. These are defined as samples that have abberrant taxonomic composition or that have an abberrant phenotype for their taxonomic composition. In essence these are one-off samples. The reasons for filtering are:

Samples 3A, 31S, 13B and 15B are near the midpoint on the right, and 19A and 8B are near the midpoint on the left. Five of these six microbiomes are atypical, and the other 19A is exclusively L. iners, but phenotypically is BV. We next exclude these six samples and re-examine the partitioning.

3A near right midpoint mix of L. iners and BV expression long branch 31S near right midpoint expresses L. crispatus and BV long branch 13B near right midpoint expresses exclusively Megasphaera long branch 15B near right midpoint significant Bifidobacteria long branch 8B near left midpoint mix of L. jensenii and G. vaginalis mid branch 19A near left midpoint L.iners but expresses BV long branch

This is a form biplot that represents distances between samples with variances of components mapped onto the plot.

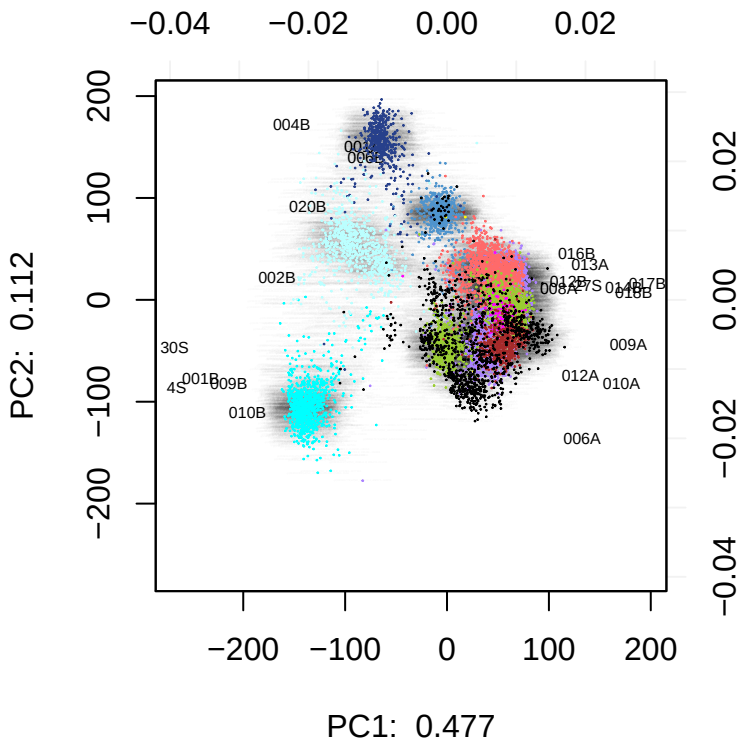
```
# this is the taxon table for samples
o <- read.table(
   "~/git/twntyfr/data/merged_readcounts_taxonomy_qiime_L6_16Sorder_color_sum.txt", header=T, row.names=
)

# this is the taxon table for reference sequences
tax <- read.table("data/cluster_tax_lookup.txt", header=T, row.names=1,
   check.names=F, sep="\t", comment.char="", quote=""
```

```
)

# this is the refseq set
d <- read.table("data/merged_readcounts_subsys4.txt", header=T,
   row.names=2, check.names=F, sep="\t", comment.char="", quote="")

sub4 <- d$subsys4
d$subsys4 <- NULL

refseq <- d$refseqIDfaa
d$refseqIDfaa <- NULL

len <- d$length
d$length <- NULL

d.min <- d

# remove samples
d.min[,"003A"] <- NULL
d.min[,"31S"] <- NULL
d.min[,"013B"] <- NULL
d.min[,"015B"] <- NULL
d.min[,"019A"] <- NULL
d.min[,"008B"] <- NULL

# filter to a mean count of 5 per refseq.
# in the perfect scenario, this would differ from 0-10
# counts between two equally sized groups
# reduces the set from

# interesting, the rare stuff (mean less than 5, is what
# differentiates the different BV types) When I use a cutoff
# of mean >44 then there is a lot more spread between the samples, but
# the percent variance goes down
d.min <- d.min[which(apply(d.min,1,mean)> 44),]

# we need to replace 0 values with a best estimate
# use zCompositions CZM by default
# but samples must be by row, so use t()

d.n0.CZM <- cmultRepl(t(d.min), label=0, method="CZM")
```

```
## No. corrected values:  1111
```

```
# turn this into a centered log-ratio transform
# samples are by row
# remember that apply by row rotates the data
# R is terrible, so we need to use t() again

d.clr <- t( apply(d.n0.CZM, 1, function(x){log(x) - mean(log(x))}) )

d.mvar.clr <- mvar(d.clr)
```

```
d.pcx <- prcomp(d.clr)

biplot(d.pcx, cex=c(0.5,0.05), col=c("black",rgb(0,0,0,0.05)), var.axes=F,
    xlab=paste("PC1: ", round(sum(d.pcx$sdev[1]^2)/d.mvar.clr, 3)),
    ylab=paste("PC2: ", round(sum(d.pcx$sdev[2]^2)/d.mvar.clr, 3)),
    scale=0
)

# plot the taxon associated with each refseq
# colors need some love
tax.subset <- tax[rownames(d.min),]

li <- grep("Lactobacillus;iners",tax.subset$common_taxonomy)
lc <- grep("Lactobacillus;crispatus",tax.subset$common_taxonomy)
lj <- c(grep("Lactobacillus;johnsonii",tax.subset$common_taxonomy),
    grep("Lactobacillus;gasseri",tax.subset$common_taxonomy))
lje <- c(grep("Lactobacillus;jensenii",tax.subset$common_taxonomy))
bvab <- grep("BVAB",tax.subset$common_taxonomy)
pb <- grep("Prevotella",tax.subset$common_taxonomy)
gv <- grep("Gardnerella",tax.subset$common_taxonomy)
me <- grep("Megasphaera",tax.subset$common_taxonomy)
sn <- grep("Sneathia",tax.subset$common_taxonomy)
di <- grep("Dialister",tax.subset$common_taxonomy)
le <- grep("Leptotrichia",tax.subset$common_taxonomy)
as <- grep("NA.",rownames(tax.subset))

points(d.pcx$rotation[,1][lc], d.pcx$rotation[,2][lc], pch=19, cex=0.05, col="cyan")
points(d.pcx$rotation[,1][li], d.pcx$rotation[,2][li], pch=19, cex=0.05, col="steelblue3")
points(d.pcx$rotation[,1][lj], d.pcx$rotation[,2][lj], pch=19, cex=0.05, col="royalblue4")
points(d.pcx$rotation[,1][lje], d.pcx$rotation[,2][lje], pch=19, cex=0.05, col="paleturquoise1")
points(d.pcx$rotation[,1][bvab], d.pcx$rotation[,2][bvab], pch=19, cex=0.05, col="yellow")
points(d.pcx$rotation[,1][pb], d.pcx$rotation[,2][pb], pch=19, cex=0.05, col="mediumpurple1")
points(d.pcx$rotation[,1][gv], d.pcx$rotation[,2][gv], pch=19, cex=0.05, col="indianred1")
points(d.pcx$rotation[,1][me], d.pcx$rotation[,2][me], pch=19, cex=0.05, col="olivedrab3")
points(d.pcx$rotation[,1][sn], d.pcx$rotation[,2][sn], pch=19, cex=0.05, col="pink")
points(d.pcx$rotation[,1][di], d.pcx$rotation[,2][di], pch=19, cex=0.05, col="magenta")
points(d.pcx$rotation[,1][le], d.pcx$rotation[,2][le], pch=19, cex=0.05, col="brown")
points(d.pcx$rotation[,1][as], d.pcx$rotation[,2][as], pch=19, cex=0.05, col="black")
```




```
#####
```

This is a good representation of the data explaining 0.477 proportion of the variance in the data on the first component, and a much smaller amount on the second and later components. This reduced dataset contains 10052 genes. We can see that the samples cluster into a few fairly discrete groups. These groups contain fairly discrete sets of genes, and we can infer that the sample set distributions are driven by gene occurrence in different genomes.

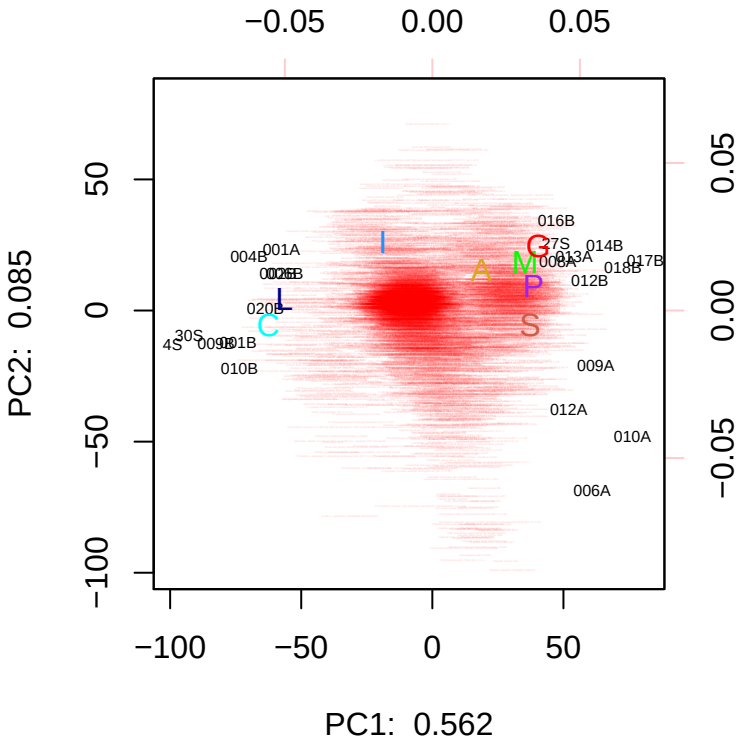
To get around the correlation between gene occurrence and expression, we aggregate the data to the SEED subsystem 4 level. This is done on the same set of samples.

```
e <- read.table("data/AitchisonTransformedDataForALDExInput.txt", header=T,
   row.names=1, check.names=F, sep="\t", comment.char="", quote="")

e.min <- e

e.min[,"003A"] <- NULL
e.min[,"31S"] <- NULL
e.min[,"013B"] <- NULL
e.min[,"015B"] <- NULL
e.min[,"019A"] <- NULL
e.min[,"008B"] <- NULL

e.min.n0.CZM <- cmultRepl(t(e.min), label=0, method="CZM")
```

```
## No. corrected values:  545
```

```
# turn this into a centered log-ratio transform
# samples are by row
```

```
# remember that apply by row rotates the data
# R is terrible, so we need to use t() again

e.min.clr <- t( apply(e.min.n0.CZM, 1, function(x){log(x) - mean(log(x))}) )

e.min.mvar.clr <- mvar(e.min.clr)

e.min.pcx <- prcomp(e.min.clr)

# TODO color samples by most abundant organism
biplot(e.min.pcx, cex=c(0.5,0.05), col=c("black",rgb(1,0,0,0.2)), var.axes=F,
    xlab=paste("PC1: ", round(sum(e.min.pcx$sdev[1]^2)/e.min.mvar.clr, 3)),
    ylab=paste("PC2: ", round(sum(e.min.pcx$sdev[2]^2)/e.min.mvar.clr, 3)),
    scale=0
)

# filter the taxon table by what is in e.min
o.min <- o[,colnames(e.min)]
o.CZM <- cmultRepl(t(o.min), label=0, method="CZM")
o.clr <- t( apply(o.CZM, 1, function(x){log(x) - mean(log(x))}) )

# overplot taxon correlations with first two components
cor.scale.0 <-  max(abs(1/range(e.min.pcx$rotation[,1:2])))

points(cor(e.min.pcx$x[,1], o.clr[,1], method="kendall") /cor.scale.0,
    cor(e.min.pcx$x[,2], o.clr[,1], method="kendall")/cor.scale.0, pch="I",
    col="DodgerBlue")
points(cor(e.min.pcx$x[,1], o.clr[,6], method="kendall") /cor.scale.0,
    cor(e.min.pcx$x[,2], o.clr[,6], method="kendall")/cor.scale.0, pch="L",
    col="DarkBlue")

points(cor(e.min.pcx$x[,1], o.clr[,2], method="kendall") /cor.scale.0,
    cor(e.min.pcx$x[,2], o.clr[,2], method="kendall")/cor.scale.0, pch="C", col="cyan")

points(cor(e.min.pcx$x[,1], o.clr[,10], method="kendall") /cor.scale.0,
    cor(e.min.pcx$x[,2], o.clr[,10], method="kendall")/cor.scale.0, pch="M", col="green")

points(cor(e.min.pcx$x[,1], o.clr[,8], method="kendall") /cor.scale.0,
    cor(e.min.pcx$x[,2], o.clr[,8], method="kendall")/cor.scale.0, pch="P", col="purple")

points(cor(e.min.pcx$x[,1], o.clr[,7], method="kendall") /cor.scale.0,
    cor(e.min.pcx$x[,2], o.clr[,7], method="kendall")/cor.scale.0, pch="G", col="red")

points(cor(e.min.pcx$x[,1], o.clr[,9], method="kendall") /cor.scale.0,
    cor(e.min.pcx$x[,2], o.clr[,9], method="kendall")/cor.scale.0, pch="A",
    col="Goldenrod")

points(cor(e.min.pcx$x[,1], o.clr[,12], method="kendall") /cor.scale.0,
    cor(e.min.pcx$x[,2], o.clr[,12], method="kendall")/cor.scale.0, pch="S", col="coral3")
```



This is a much better representation of the data. We did not bother filtering by abundance since there is a much smaller number of features (3874). Now we can see a fairly clear split in the data that is less driven by gene occurrence and more by functions within the microbiome (cite Beiko review regarding ecosystems of genes rather than organisms here!).

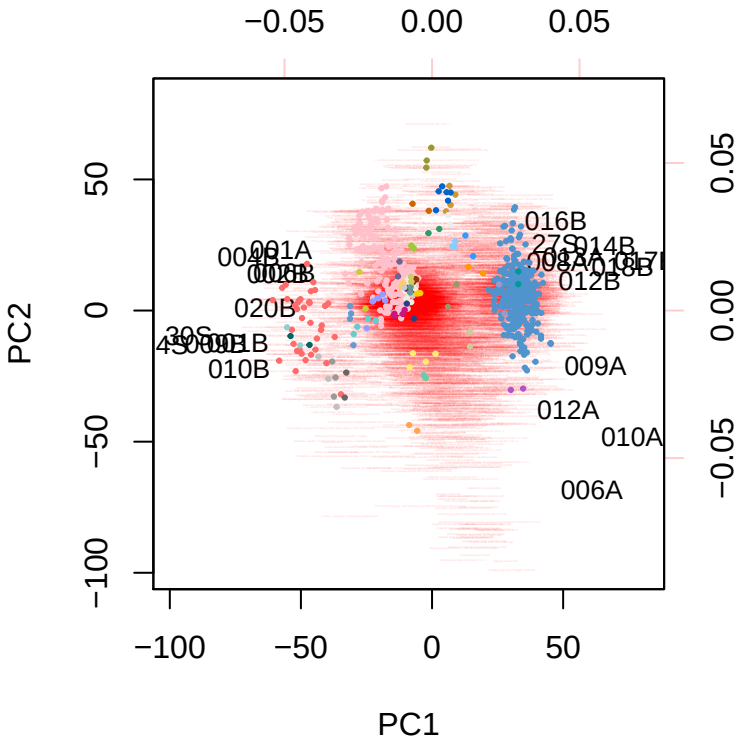
```
###### phi with Bayesian estimation
# the problem is that a low phi can easily arise between functions
# that have the same distribution of 0 values across samples
# so we need to estimate the value of 0 (and other low-count functions)

# generate random DIR clr instances with ALDEx2 in the minimum function set
e.x <- aldex.clr(e.min)
```

```
## [1] "operating in serial mode"
```

```
# calculate phi divided by number of random instances
e.min.sma.df <- aldex.phi(e.x)

# find the set of connections with phi less than some value
# we choose an arbitrary cutoff, but it is higher after Bayesian estimation
# obviously
 phi.cutoff <- 0.03

e.min.sma.lo.phi <- subset(e.min.sma.df, phi < phi.cutoff)

# generate a graphical object
g <- graph.data.frame(e.min.sma.lo.phi, directed=FALSE)

# overview of all the proportional relationships
# this can take a long time!!!
# plot(g, layout=layout.fruchterman.reingold.grid(g, weight=0.05/E(g)$phi), vertex.size=1,
```

```
#   vertex.color="black", vertex.label=NA)

# # get the clusters from the graph object
 g.clust <- clusters(g)

# write them to a file

# # data frame containing the names and group memberships of each cluster
g.df <- data.frame(Systematic.name=V(g)$name, cluster=g.clust$membership,
    cluster.size=g.clust$csize[g.clust$membership])

attach(g.df)
g.order <- g.df[order(cluster),]
detach(g.df)
write.table(g.order, file="SEED.g.df.txt", sep="\t", quote=F, col.names=NA)

max.clust <- induced.subgraph(g, which(g.clust$membership %in% which(g.clust$csize ==
    max(g.clust$csize))))
max.clust.names <- V(max.clust)$name
```

```
par(mfrow=c(1,1))

biplot(e.min.pcx,col=c("black", rgb(1,0,0,0.2)),cex=c(0.8,0.05),
var.axes=F, scale=0)

#nms <- rownames(g.df)[g.df$cluster==max.clust.names]

#points(e.min.pcx$rotation[max.clust.names,][,1],e.min.pcx$rotation[max.clust.names,][,2],
#    col=rgb(0,0,1,0.2),pch=19,cex=0.2)

colours <- c("steelblue3", "skyblue1", "indianred1", "mediumpurple1", "olivedrab3",
   "pink", "#FFED6F", "mediumorchid3", "ivory2", "tan1", "aquamarine3", "#C0C0C0",
   "royalblue4", "mediumvioletred", "#999933", "#666699", "#CC9933", "#006666", "#3399FF",
   "#993300", "#CCCC99", "#666666", "#FFCC66", "#6699CC", "#663366", "#9999CC", "#CCCCCC",
   "#669999", "#CCCC66", "#CC6600", "#9999FF", "#0066CC", "#99CCCC", "#999999", "#FFCC00",
   "#009999", "#FF9900", "#999966", "#66CCCC", "#339966", "#CCCC33", "#EDEDED"
)

big <- g.df[which(g.df$cluster.size >= 2),]
colnames(big) <- colnames(g.df)

lev <- factor(big$cluster)
for(i in as.numeric(levels(lev))){
nms <- rownames(big)[big$cluster==i]
#print(rownames(big)[big$cluster==i])
#print("")
points(e.min.pcx$rotation[nms,][,1],e.min.pcx$rotation[nms,][,2], col=colours[i],
    pch=19, cex=0.3)
}
```



Here the vast majority of low phi values come from core functions that are constant ratios in all samples, but not differential. A few smaller groups are both differential and compositionally associated.

Incorporating the Bayesian estimate of 0 gives a much clearer picture, than using point estimates. When point estimates are used, then the majority of the clusters are near 0,0 - simply because of sets of functions that share the same distribution of 0 across samples.

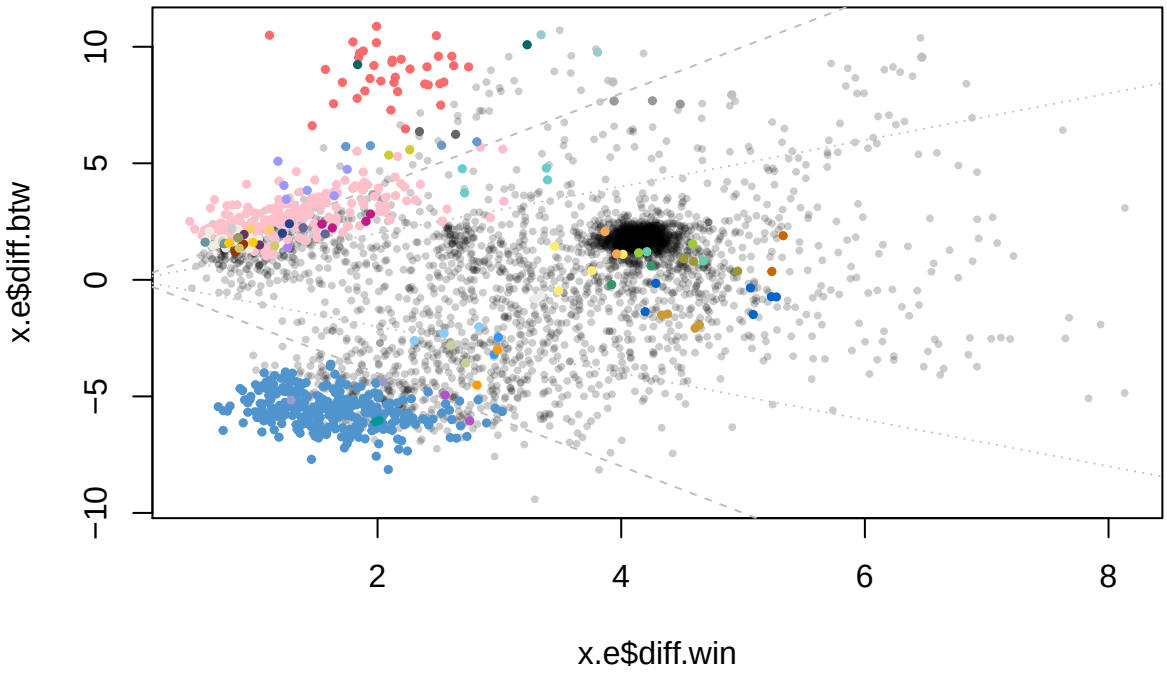
The relationship between phi and effect size.

```
    B <- match(rownames(e.min.pcx$x)[e.min.pcx$x[,1] > 0], rownames(e.min.pcx$x))
    H <- match(rownames(e.min.pcx$x)[e.min.pcx$x[,1] < 0], rownames(e.min.pcx$x))

    conds <- vector()
    conds[B] <- "B"
    conds[H] <- "H"

    x.e <- aldex.effect(e.x, conds, verbose=FALSE)
```

```
## [1] "operating in serial mode"
```

```
    plot(x.e$diff.win, x.e$diff.btw, pch=19,cex=0.4, col=rgb(0,0,0,0.2))

    abline(0,2, lty=2, col="grey")
    abline(0,-2, lty=2, col="grey")
    abline(0,1, lty=3, col="grey")
    abline(0,-1, lty=3, col="grey")

    for(i in as.numeric(levels(lev))){
    nms <- rownames(big)[big$cluster==i]
    #print(rownames(big)[big$cluster==i])
    #print("")
```

```
    points(x.e[nms,"diff.win"],x.e[nms,"diff.btw"], col=colours[i], pch=19, cex=0.5)
}
```



Now we do it all over again with KO numbers

```
k <- read.table("data/KO_reads_aitchison_sum.txt", header=T,
    row.names=1, sep="\t", check.names=FALSE)

k.min <- k[grep("^K", rownames(k)),]

k.min[,"003A"] <- NULL
k.min[,"31S"] <- NULL
k.min[,"013B"] <- NULL
k.min[,"015B"] <- NULL
k.min[,"019A"] <- NULL
k.min[,"008B"] <- NULL
dim(k.min)
```

```
## [1] 2956   22
```

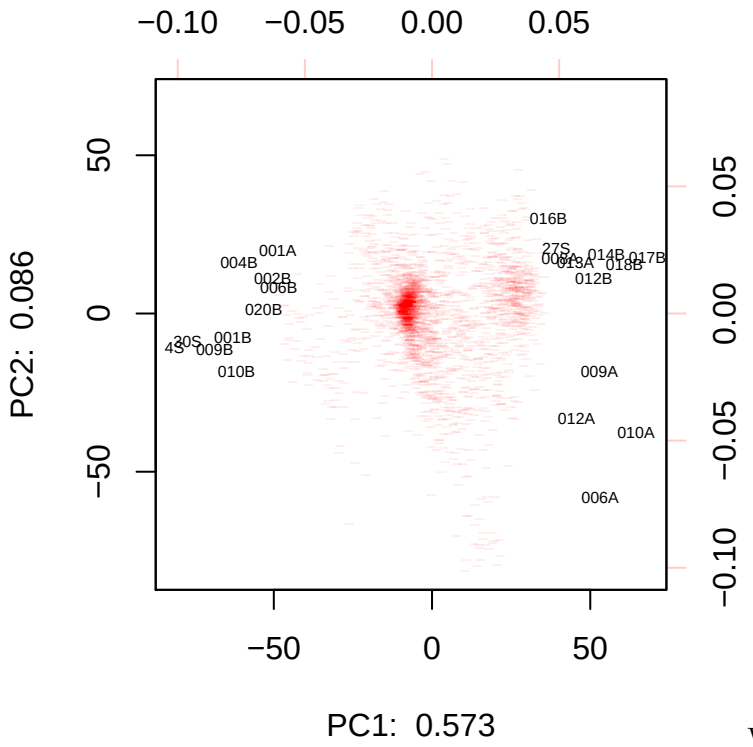
```
k.min <- k.min[apply(k.min,1,max) > 0,]

dim(k.min)
```

```
## [1] 2842   22
```

```
k.min.n0.CZM <- cmultRepl(t(k.min), label=0, method="CZM")
```

```
## No. corrected values:  460
```

```
# turn this into a centered log-ratio transform
# samples are by row
# remember that apply by row rotates the data
# R is terrible, so we need to use t() again

k.min.clr <- t( apply(k.min.n0.CZM, 1, function(x){log(x) - mean(log(x))}) )

k.min.mvar.clr <- mvar(k.min.clr)

k.min.pcx <- prcomp(k.min.clr)

# TODO color samples by most abundant organism
biplot(k.min.pcx, cex=c(0.5,0.05), col=c("black",rgb(1,0,0,0.2)), var.axes=F,
    xlab=paste("PC1: ", round(sum(k.min.pcx$sdev[1]^2)/k.min.mvar.clr, 3)),
    ylab=paste("PC2: ", round(sum(k.min.pcx$sdev[2]^2)/k.min.mvar.clr, 3)),
    scale=0
)
```



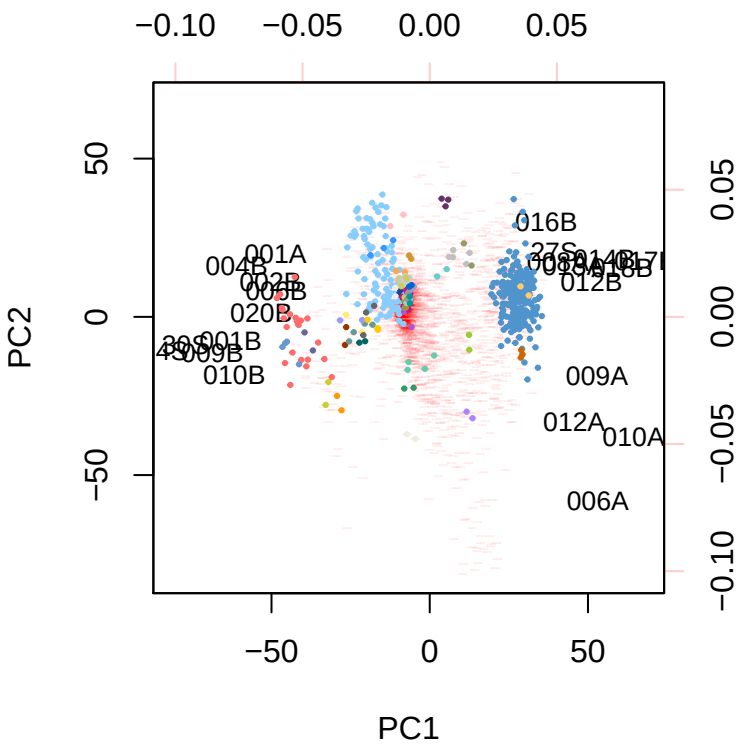
Wow, this is indistinguishable from SEED level 4 in terms of variance explained and associations between samples.

```
# generate random DIR clr instances with ALDEx2 in the minimum function set
k.x <- aldex.clr(k.min)
```

```
## [1] "operating in serial mode"
```

```
# calculate phi divided by number of random instances
k.min.sma.df <- aldex.phi(k.x)

# find the set of connections with phi less than some value
# we choose an arbitrary cutoff, but it is higher after Bayesian estimation
```

```
# obviously
 phi.cutoff <- 0.03

k.min.sma.lo.phi <- subset(k.min.sma.df, phi < phi.cutoff)

# generate a graphical object
g <- graph.data.frame(k.min.sma.lo.phi, directed=FALSE)

# overview of all the proportional relationships
# this can take a long time!!!
# plot(g, layout=layout.fruchterman.reingold.grid(g, weight=0.05/E(g)$phi), vertex.size=1,
#   vertex.color="black", vertex.label=NA)

# # get the clusters from the graph object
 g.clust <- clusters(g)

# write them to a file

# # data frame containing the names and group memberships of each cluster
KO.g.df <- data.frame(Systematic.name=V(g)$name, cluster=g.clust$membership,
    cluster.size=g.clust$csize[g.clust$membership])

attach(KO.g.df)
k.order <- KO.g.df[order(cluster),]
detach(KO.g.df)

write.table(k.order, file="KO.g.df.txt", sep="\t", quote=F, col.names=NA)

max.clust <- induced.subgraph(g, which(g.clust$membership %in% which(g.clust$csize ==
    max(g.clust$csize))))
max.clust.names <- V(max.clust)$name

biplot(k.min.pcx,col=c("black", rgb(1,0,0,0.2)),cex=c(0.8,0.05),
var.axes=F, scale=0)

#nms <- rownames(g.df)[g.df$cluster==max.clust.names]

big <- KO.g.df[which(KO.g.df$cluster.size >= 2),]
colnames(big) <- colnames(KO.g.df)

lev <- factor(big$cluster)
for(i in as.numeric(levels(lev))){
nms <- rownames(big)[big$cluster==i]
#print(rownames(big)[big$cluster==i])
#print("")
points(k.min.pcx$rotation[nms,][,1],k.min.pcx$rotation[nms,][,2], col=colours[i],
    pch=19, cex=0.3)
}
```


−0.10
−0.05
0.00
0.05
50
0
−50
PC2
016B
001A
004B
012B
020B
001B
009B
010B
009A
012A
010A
006A
−50
0
50
PC1
0.05
0.00
−0.05
−0.10