associations_4panels

Jieqi Tu (jt3098)

1/19/2020

Import dataset

```r
# Import dataset
CL = readxl::read_excel("./data_new/ABC_Cord Blood_Metabolomics_CL data_15Jan2020.xlsx")
BA = readxl::read_excel("./data_new/ABC_Cord Blood_Metabolomics_BA data_15Jan2020.xlsx")
PM = readxl::read_excel("./data_new/ABC_Cord Blood_Metabolomics_PM data_15Jan2020.xlsx")
```

```
## New names:
## * lactamide -> lactamide...94
## * lactamide -> lactamide...95
```

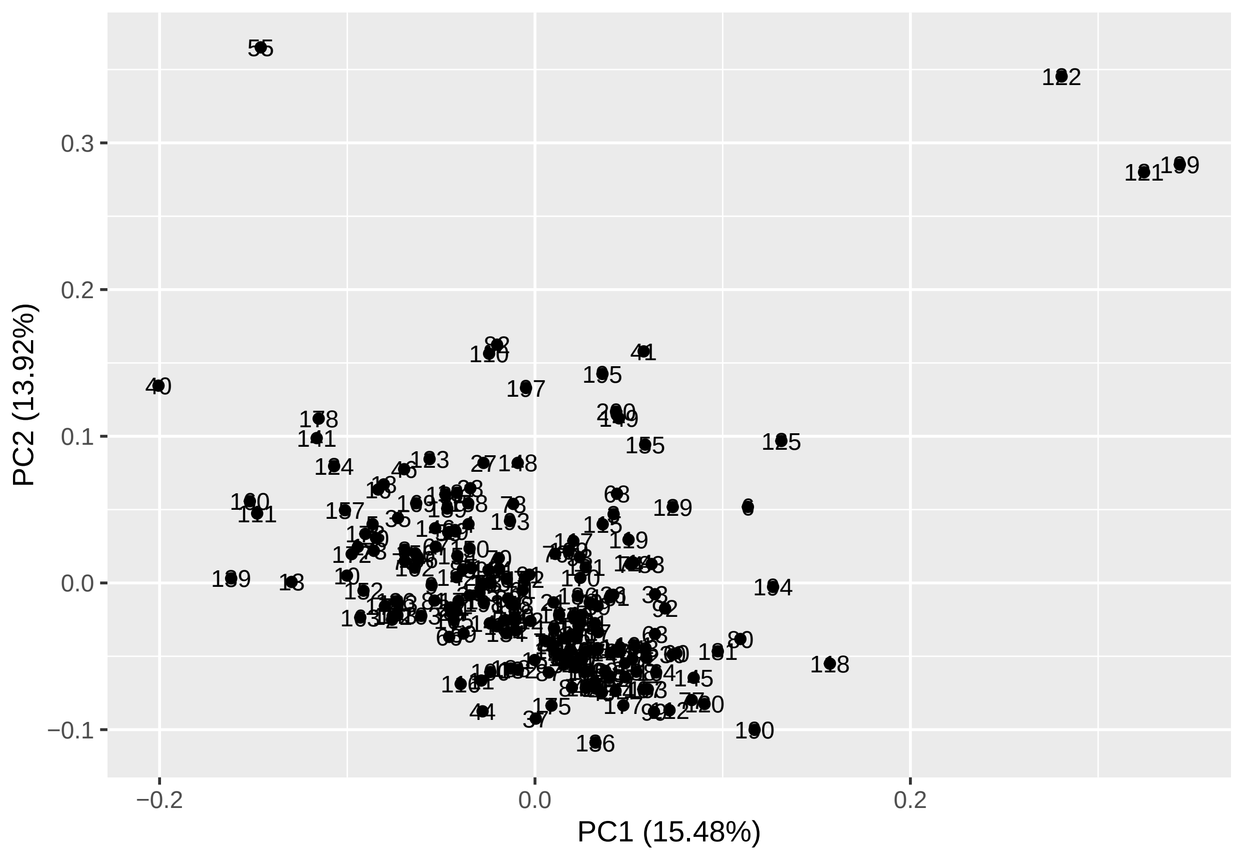
```r
OL =  readxl::read_excel("./data_new/ABC_Cord Blood_Metabolomics_OL data_15Jan2020.xlsx")
```

Convert 0 to half of the minimum values

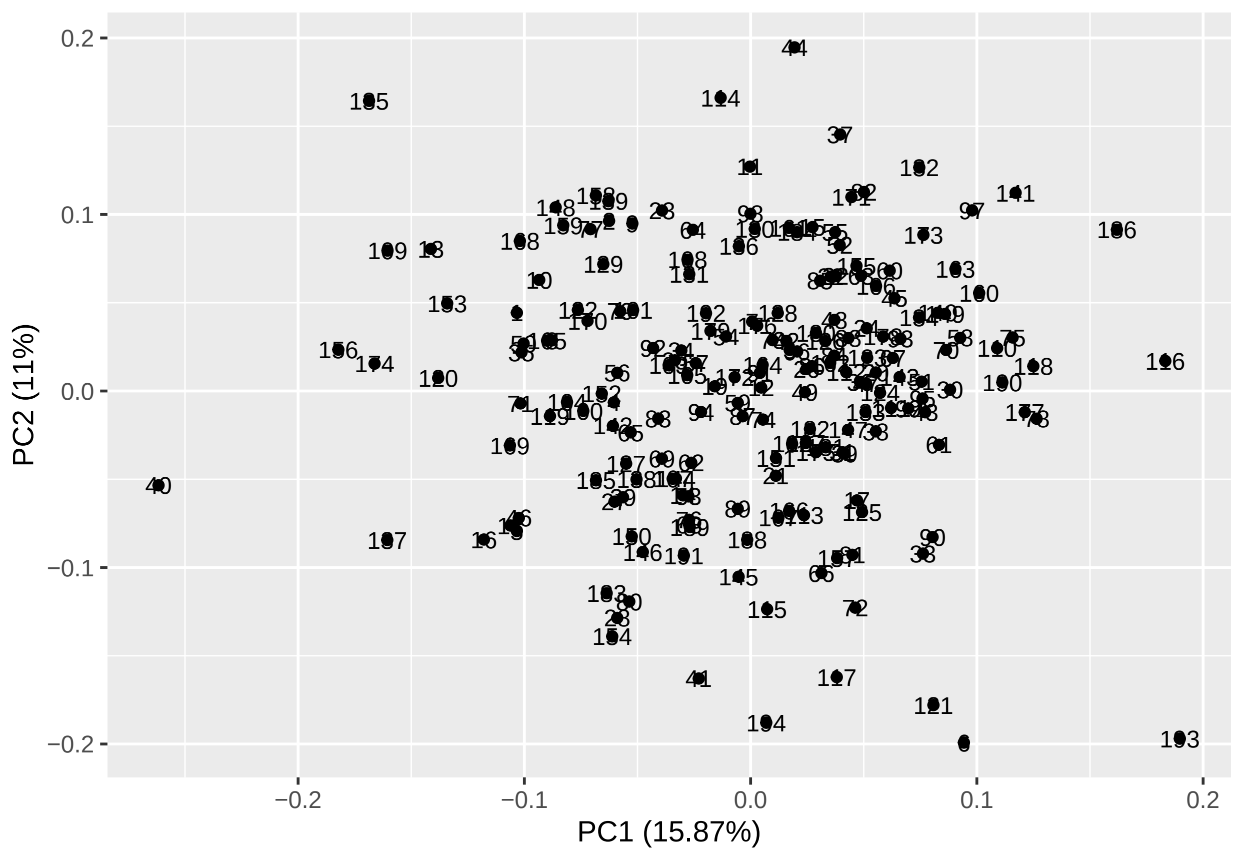
```r
CL_data = CL[11:491]
BA_data = BA[11:266]
PM_data = PM[11:193]
OL_data = OL[11:81]
CL_data[CL_data == 0] = NA
BA_data[BA_data == 0] = NA
PM_data[PM_data == 0] = NA
OL_data[OL_data == 0] = NA
CL_min = sapply(CL_data[1:481], function(x) min(x, na.rm = T))
BA_min = sapply(BA_data[1:256], function(x) min(x, na.rm = T))
PM_min = sapply(PM_data[1:183], function(x) min(x, na.rm = T))
OL_min = sapply(OL_data[1:71], function(x) min(x, na.rm = T))

CL_data = CL[11:491]
BA_data = BA[11:266]
PM_data = PM[11:193]
OL_data = OL[11:81]
# Convert 0 to half of the minimum value
for(i in 1:481) {
  CL_data[i][CL_data[i]==0] = 0.5*CL_min[i]
}

for(i in 1:256) {
  BA_data[i][BA_data[i]==0] = 0.5*BA_min[i]
}

for(i in 1:183) {
  PM_data[i][PM_data[i]==0] = 0.5*PM_min[i]
}

for(i in 1:71) {
  OL_data[i][OL_data[i]==0] = 0.5*OL_min[i]
}

CL_info = CL[1:10]
CL_new = cbind.data.frame(CL_info, CL_data)

BA_info = BA[1:10]
BA_new = cbind.data.frame(BA_info, BA_data)

PM_info = PM[1:10]
PM_new = cbind.data.frame(PM_info, PM_data)

OL_info = OL[1:10]
OL_new = cbind.data.frame(OL_info, OL_data)
```

PCA

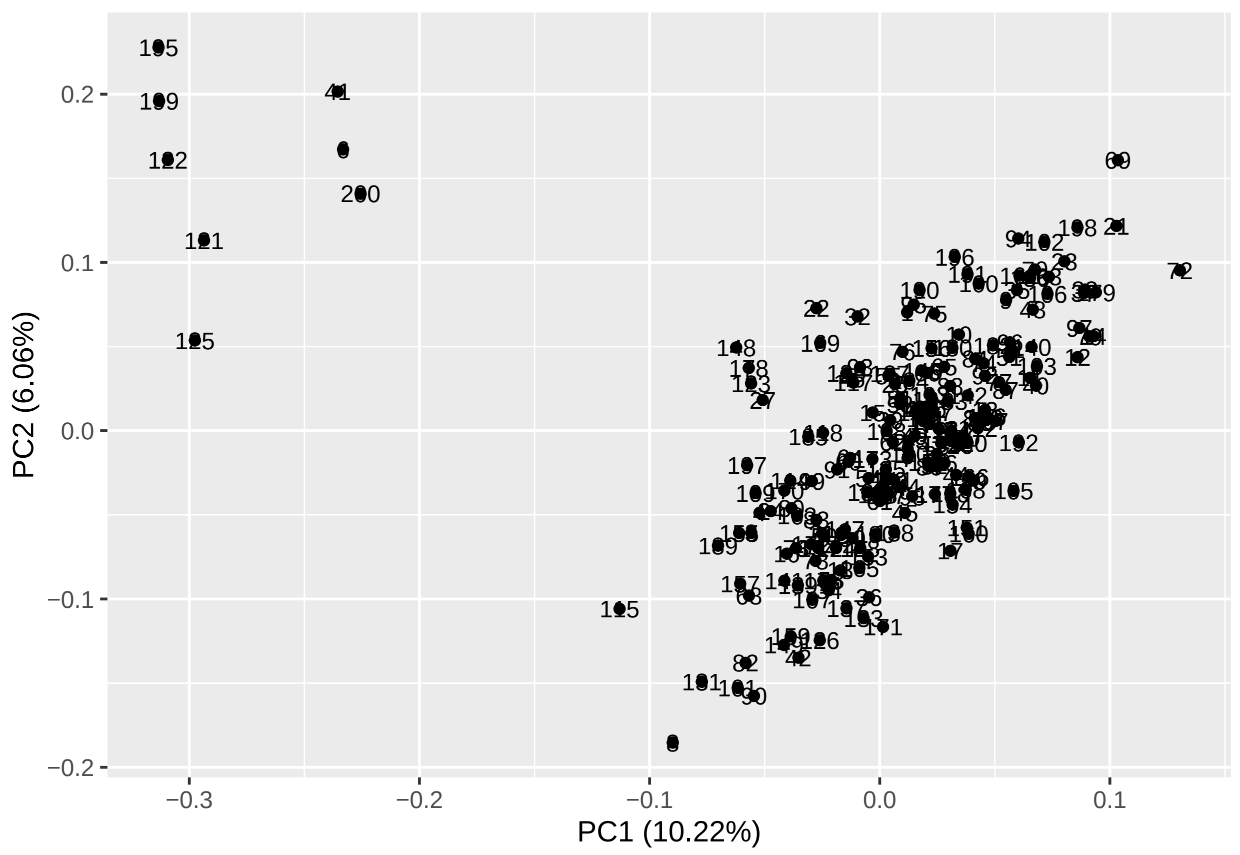
```r
# Calculate z-scores
mean_CL = mean_control = sapply(CL_data[1:481], function(x) mean(x))
sd_CL = sapply(CL_data[1:481], function(x) sd(x))
CL_data_z = CL_data
for(i in 1:481) {
  CL_data_z[i] = (CL_data[i] - mean_CL[i])/sd_CL[i]
}

CL_pca = prcomp(CL_data_z[c(1:481)], center = F, scale. = F)

# Plot PC1/PC2
library(ggfortify)
autoplot(CL_pca, label = T, label.size = 3)
```



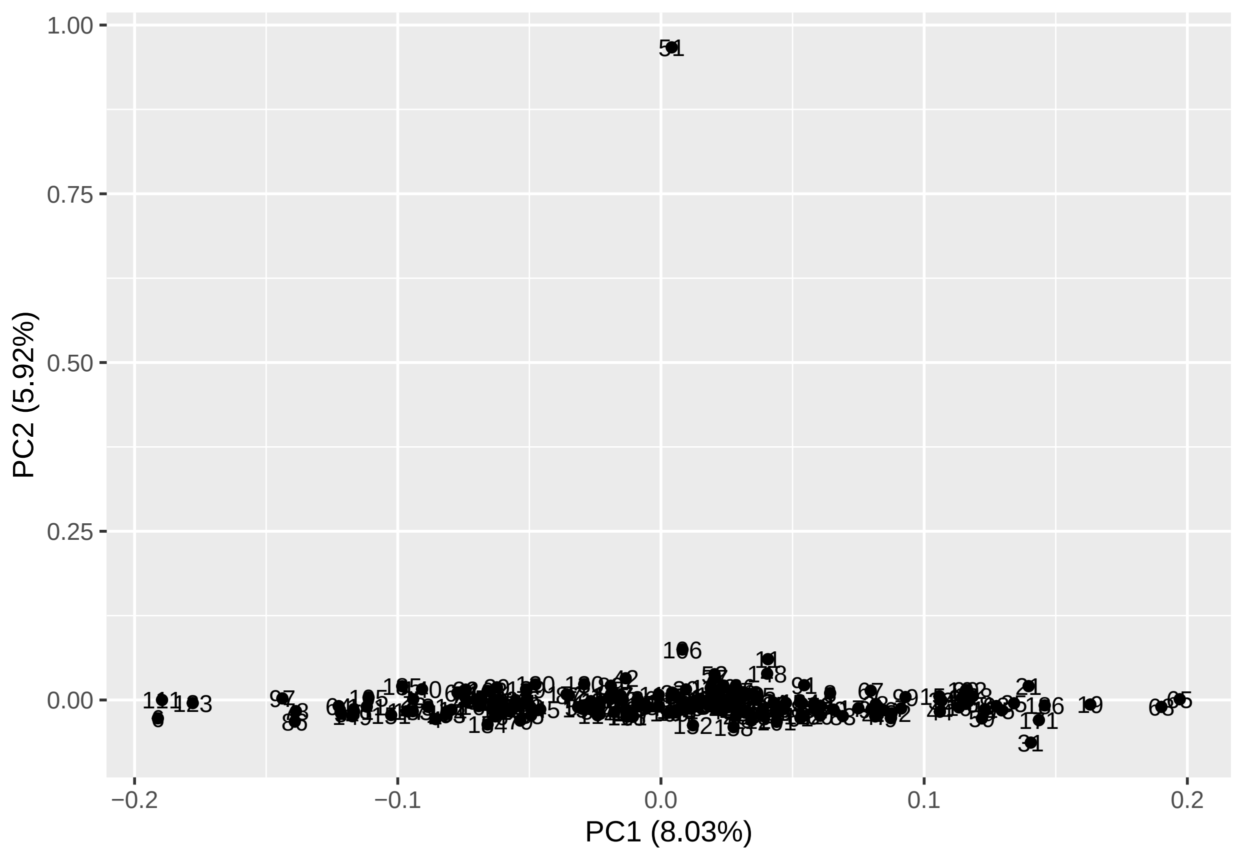
```
CL_data_new = CL_data_z[-c(55, 56, 121, 122, 195, 196),]
CL_pca_2 = prcomp(CL_data_new[c(1:481)], center = F, scale. = F)
autoplot(CL_pca_2, label = T, label.size = 3)
```



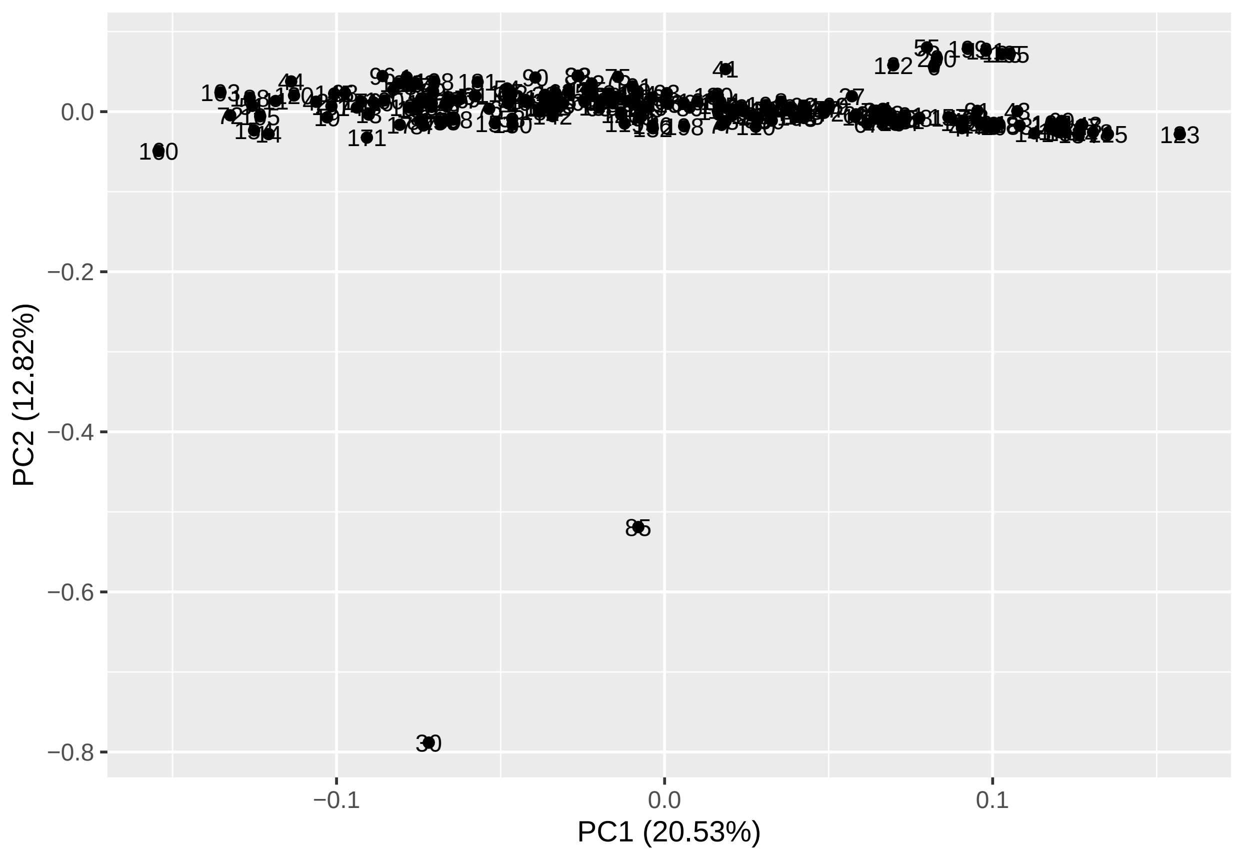
```r
# Calculate z-scores
mean_BA = mean_control = sapply(BA_data[1:256], function(x) mean(x))
sd_BA = sapply(BA_data[1:256], function(x) sd(x))
BA_data_z = BA_data
for(i in 1:256) {
  BA_data_z[i] = (BA_data[i] - mean_BA[i])/sd_BA[i]
}

BA_pca = prcomp(BA_data_z[c(1:256)], center = F, scale. = F)

# Plot PC1/PC2
library(ggfortify)
autoplot(BA_pca, label = T, label.size = 3)
```



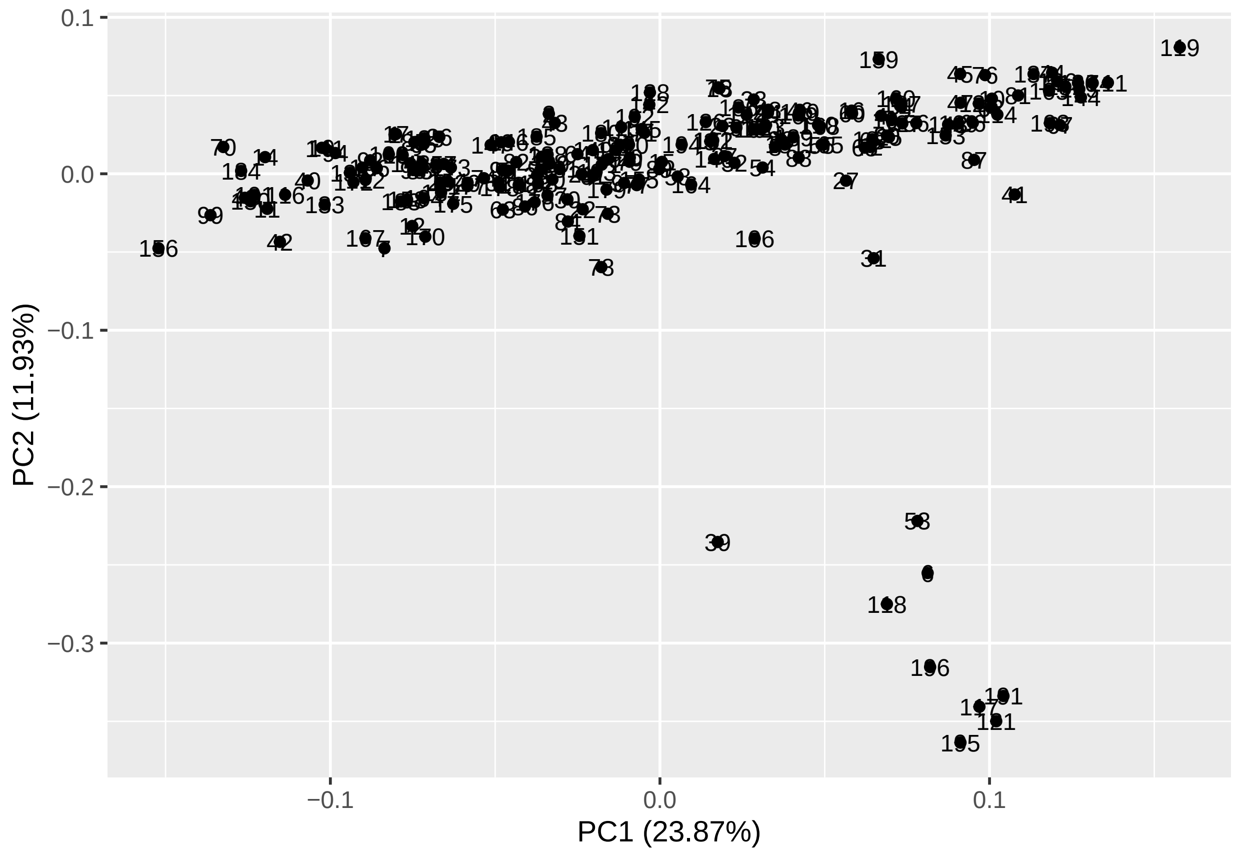
```
BA_data_new = BA_data_z[-c(5,6,41, 42, 191, 192, 121, 122, 125, 126, 195, 196, 199, 200),]
BA_pca_new = prcomp(BA_data_new[c(1:256)], center = F, scale. = F)
autoplot(BA_pca_new, label = T, label.size = 3)
```



```
# Calculate z-scores
mean_OL = mean_control = sapply(OL_data[1:71], function(x) mean(x))
sd_OL = sapply(OL_data[1:71], function(x) sd(x))
OL_data_z = OL_data
for(i in 1:71) {
  OL_data_z[i] = (OL_data[i] - mean_OL[i])/sd_OL[i]
}

OL_pca = prcomp(OL_data_z[c(1:71)], center = F, scale. = F)

# Plot PC1/PC2
library(ggfortify)
autoplot(OL_pca, label = T, label.size = 3)
```



```r
OL_data_new = OL_data_z[-c(29, 30, 85, 86),]
OL_pca_new = prcomp(OL_data_new[c(1:71)], center = F, scale. = F)
autoplot(OL_pca_new, label = T, label.size = 3)
```



Delete outliers

```r
# Delete outliers that identified by PCA
CL_new = CL_new[-c(55, 56, 121, 122, 195, 196),]
BA_new = BA_new[-c(5,6,41, 42, 191, 192, 121, 122, 125, 126, 195, 196, 199, 200),]
PM_new = PM_new
OL_new = OL_new[-c(29, 30, 85, 86),]
```

Scaling

```r
# Calculate the standard deviation of the control group for each dataset
BA_c = BA_new %>%
  group_by(Strata) %>%
  filter(PROT_ASD_2015 == 0)
CL_c = CL_new %>%
  group_by(Strata) %>%
  filter(PROT_ASD_2015 == 0)
PM_c = PM_new %>%
  group_by(Strata) %>%
  filter(PROT_ASD_2015 == 0)
OL_c = OL_new %>%
  group_by(Strata) %>%
  filter(PROT_ASD_2015 == 0)
sd_BA = sapply(BA_c[11:266], function(x) sd(x))
sd_CL = sapply(CL_c[11:491], function(x) sd(x))
```

```r
sd_PM = sapply(PM_c[11:193], function(x) sd(x))
sd_OL = sapply(OL_c[11:81], function(x) sd(x))
# Divide the standard deviation
for(i in 1:256) {
  BA_new[i+10] = BA_new[i+10]/sd_BA[i]
}
for(i in 1:481) {
  CL_new[i+10] = CL_new[i+10]/sd_CL[i]
}
for(i in 1:183) {
  PM_new[i+10] = PM_new[i+10]/sd_PM[i]
}
for(i in 1:71) {
  OL_new[i+10] = OL_new[i+10]/sd_OL[i]
}
```

Log transformation

```r
# base 10 log transformation
BA_log = BA_new
CL_log = CL_new
PM_log = PM_new
OL_log = OL_new
for(i in 1:256) {
  BA_log[i+10] = log10(BA_new[i+10])
}

for(i in 1:481) {
  CL_log[i+10] = log10(CL_new[i+10])
}

for(i in 1:183) {
  PM_log[i+10] = log10(PM_new[i+10])
}

for(i in 1:71) {
  OL_log[i+10] = log10(OL_new[i+10])
}
```

Recode birth year

```r
CL_log =
  CL_log %>%
  mutate(year_new = ifelse(BirthYear <= 2000, "before 2000", BirthYear))

PM_log =
  PM_log %>%
  mutate(year_new = ifelse(BirthYear <= 2000, "before 2000", BirthYear))

BA_log =
  BA_log %>%
  mutate(year_new = ifelse(BirthYear <= 2000, "before 2000", BirthYear))
```

```
OL_log =
  OL_log %>%
  mutate(year_new = ifelse(BirthYear <= 2000, "before 2000", BirthYear))
```

Test for associations using conditional logistic regression

```
# CL data associations
library(survival)
column_name = colnames(CL_log)
predictor_name = column_name[11:491]
store = data.frame(matrix(ncol = 7, nrow = 0))
colnames(store) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.high")
for(i in 1:481) {
  data = CL_log[i+10]
  a = unlist(data)
  model = clogit(PROT_ASD_2015 ~ a + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata(Stra
  result = broom::tidy(model)
  store[i, ] = result[1,]
}

CL_result =
  store %>%
  mutate(term = predictor_name)
```

```
# BA data associations
column_name = colnames(BA_log)
predictor_name = column_name[11:266]

store_BA = data.frame(matrix(ncol = 7, nrow = 0))
colnames(store_BA) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.hi
for(i in 1:256) {
  data = BA_log[i+10]
  a = unlist(data)
  model = clogit(PROT_ASD_2015 ~ a + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata(Stra
  result = broom::tidy(model)
  store_BA[i, ] = result[1,]
}
BA_result =
  store_BA %>%
  mutate(term = predictor_name)
```

```
# PM data associations
column_name = colnames(PM_log)
predictor_name = column_name[11:193]

store_PM = data.frame(matrix(ncol = 7, nrow = 0))
colnames(store_PM) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.hi
for(i in 1:183) {
  data = PM_log[i+10]
  a = unlist(data)
  model = clogit(PROT_ASD_2015 ~ a + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata(Stra
  result = broom::tidy(model)
  store_PM[i, ] = result[1,]
}
```

```r
PM_result =
  store_PM %>%
  mutate(term = predictor_name)
```

```r
# PM data associations
column_name = colnames(OL_log)
predictor_name = column_name[11:81]

store_OL = data.frame(matrix(ncol = 7, nrow = 0))
colnames(store_OL) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.hig
for(i in 1:71) {
  data = OL_log[i+10]
  a = unlist(data)
  model = clogit(PROT_ASD_2015 ~ a + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata(Stra
  result = broom::tidy(model)
  store_OL[i, ] = result[1,]
}

OL_result =
  store_OL %>%
  mutate(term = predictor_name)
```

False Discovery Rate Correction

```r
# Calculate the adjusted p-values
# CL data
CL_result$adjusted_pval = p.adjust(p = CL_result$p.value, method = "BH")

# PM data
PM_result$adjusted_pval = p.adjust(p = PM_result$p.value, method = "BH")

# BA data
BA_result$adjusted_pval = p.adjust(p = BA_result$p.value, method = "BH")

# OL data
OL_result$adjusted_pval = p.adjust(p = OL_result$p.value, method = "BH")
```

Add quadratic terms for each analytes

```r
# Add squared term in CL data
column_name = colnames(CL_log)
predictor_name = column_name[11:491]
store = data.frame(matrix(ncol = 7, nrow = 0))
p_chisq = data.frame(matrix(ncol = 4, nrow = 0))
colnames(store) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.high")
colnames(p_chisq) = c("loglik", "Chisq", "Df", "P(>|Chi|)")
for(i in 1:481) {
  data = CL_log[i+10]
  a = unlist(data)
  CL_log$b = a*a
  model1 = clogit(PROT_ASD_2015 ~ a + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata(Str
  model2 = clogit(PROT_ASD_2015 ~ a + b + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata
  result = broom::tidy(model2)
  lrt = anova(model1, model2) %>% as.data.frame()
  store[i, ] = result[2,]
  p_chisq[i, ] = lrt[2, ]
}

store =
  store %>%
  mutate(term = predictor_name)

CL_quadratic_result =
  cbind.data.frame(store, p_chisq)
```

```
# Add squared term in CL data
column_name = colnames(BA_log)
predictor_name = column_name[11:266]
store = data.frame(matrix(ncol = 7, nrow = 0))
p_chisq = data.frame(matrix(ncol = 4, nrow = 0))
colnames(store) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.high"
colnames(p_chisq) = c("loglik", "Chisq", "Df", "P(>|Chi|)")
for(i in 1:256) {
  data = BA_log[i+10]
  a = unlist(data)
  BA_log$b = a*a
  model1 = clogit(PROT_ASD_2015 ~ a + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata(Stra
  model2 = clogit(PROT_ASD_2015 ~ a + b + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata
  result = broom::tidy(model2)
  lrt = anova(model1, model2) %>% as.data.frame()
  store[i, ] = result[2,]
  p_chisq[i, ] = lrt[2, ]
}

store =
  store %>%
  mutate(term = predictor_name)

BA_quadratic_result =
  cbind.data.frame(store, p_chisq)
```

```
# Add squared term in CL data
column_name = colnames(PM_log)
predictor_name = column_name[11:193]
store = data.frame(matrix(ncol = 7, nrow = 0))
p_chisq = data.frame(matrix(ncol = 4, nrow = 0))
colnames(store) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.high"
colnames(p_chisq) = c("loglik", "Chisq", "Df", "P(>|Chi|)")
for(i in 1:183) {
  data = PM_log[i+10]
  a = unlist(data)
  PM_log$b = a*a
  model1 = clogit(PROT_ASD_2015 ~ a + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata(Stra
  model2 = clogit(PROT_ASD_2015 ~ a + b + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata
  result = broom::tidy(model2)
  lrt = anova(model1, model2) %>% as.data.frame()
  store[i, ] = result[2,]
  p_chisq[i, ] = lrt[2, ]
}

store =
  store %>%
  mutate(term = predictor_name)

PM_quadratic_result =
  cbind.data.frame(store, p_chisq)
```

```
# Add squared term in CL data
column_name = colnames(OL_log)
predictor_name = column_name[11:81]
store = data.frame(matrix(ncol = 7, nrow = 0))
p_chisq = data.frame(matrix(ncol = 4, nrow = 0))
colnames(store) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.high"
colnames(p_chisq) = c("loglik", "Chisq", "Df", "P(>|Chi|)")
for(i in 1:71) {
  data = OL_log[i+10]
  a = unlist(data)
  OL_log$b = a*a
  model1 = clogit(PROT_ASD_2015 ~ a + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata(Str
  model2 = clogit(PROT_ASD_2015 ~ a + b + factor(BirthSeason) + factor(year_new) + factor(Sex) + strata
  result = broom::tidy(model2)
  lrt = anova(model1, model2) %>% as.data.frame()
  store[i, ] = result[2,]
  p_chisq[i, ] = lrt[2, ]
}

store =
  store %>%
  mutate(term = predictor_name)

OL_quadratic_result =
  cbind.data.frame(store, p_chisq)
```

FWER correction using Hockberg step-up procedure

```
# FWER correction for CL results
CL_quadratic_result$p_adjust = p.adjust(CL_quadratic_result$`P(>|Chi|)`, method = "hochberg")
BA_quadratic_result$p_adjust = p.adjust(BA_quadratic_result$`P(>|Chi|)`, method = "hochberg")
PM_quadratic_result$p_adjust = p.adjust(PM_quadratic_result$`P(>|Chi|)`, method = "hochberg")
OL_quadratic_result$p_adjust = p.adjust(OL_quadratic_result$`P(>|Chi|)`, method = "hochberg")
```

90th quantile analysis

```
# Calculate 90th quantiles of control group for each variable in each panel
BA_c = BA_new %>%
  group_by(Strata) %>%
  filter(PROT_ASD_2015 == 0)
CL_c = CL_new %>%
  group_by(Strata) %>%
  filter(PROT_ASD_2015 == 0)
```

```r
PM_c = PM_new %>%
  group_by(Strata) %>%
  filter(PROT_ASD_2015 == 0)
OL_c = OL_new %>%
  group_by(Strata) %>%
  filter(PROT_ASD_2015 == 0)
CL_quantile = sapply(CL_c[11:491], function(x) quantile(x, probs = 0.9))
BA_quantile = sapply(BA_c[11:266], function(x) quantile(x, probs = 0.9))
PM_quantile = sapply(PM_c[11:193], function(x) quantile(x, probs = 0.9))
OL_quantile = sapply(OL_c[11:81], function(x) quantile(x, probs = 0.9))

CL_converted = data.frame(matrix(ncol = 481, nrow = 194))
CL_names = colnames(CL_new)
colnames(CL_converted) = CL_names[11:491]
# Convert variables from continuous to binary
for (i in 1:481) {
  CL_converted[i] = as.numeric(CL_new[i+10]> CL_quantile[i])
}

BA_converted = data.frame(matrix(ncol = 256, nrow = 186))
BA_names = colnames(BA_new)
colnames(BA_converted) = BA_names[11:266]
for (i in 1:256) {
  BA_converted[i] = as.numeric(BA_new[i+10]> BA_quantile[i])
}

PM_converted = data.frame(matrix(ncol = 183, nrow = 196))
PM_names = colnames(PM_new)
colnames(PM_converted) = PM_names[11:193]
for (i in 1:183) {
  PM_converted[i] = as.numeric(PM_new[i+10]> PM_quantile[i])
}


OL_converted = data.frame(matrix(ncol = 71, nrow = 196))
OL_names = colnames(OL_new)
colnames(OL_converted) = OL_names[11:81]
for (i in 1:71) {
  OL_converted[i] = as.numeric(OL_new[i+10]> OL_quantile[i])
}

# Combine the data part and the demographic part
CL_info_2 = CL_new[1:10]
BA_info_2 = BA_new[1:10]
PM_info_2 = PM_new[1:10]
OL_info_2 = OL_new[1:10]

CL_binary= cbind.data.frame(CL_info_2, CL_converted)
BA_binary= cbind.data.frame(BA_info_2, BA_converted)
PM_binary= cbind.data.frame(PM_info_2, PM_converted)
OL_binary= cbind.data.frame(OL_info_2, OL_converted)

# Recode year again
```

```
CL_binary =
  CL_binary %>%
  mutate(year_new = ifelse(BirthYear <= 2000, "before 2000", BirthYear))

PM_binary =
  PM_binary %>%
  mutate(year_new = ifelse(BirthYear <= 2000, "before 2000", BirthYear))

BA_binary =
  BA_binary %>%
  mutate(year_new = ifelse(BirthYear <= 2000, "before 2000", BirthYear))

OL_binary =
  OL_binary %>%
  mutate(year_new = ifelse(BirthYear <= 2000, "before 2000", BirthYear))
# Perform conditional logistic regression again with converted values
# CL data
column_name = colnames(CL_binary)
predictor_name = column_name[11:491]
CL_store = data.frame(matrix(ncol = 7, nrow = 0))
colnames(CL_store) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.hi
for(i in 1:481) {
  data = CL_binary[i+10]
  a = unlist(data)
  model = clogit(PROT_ASD_2015 ~ factor(a) + factor(BirthSeason) + factor(year_new) + factor(Sex) + str
  result = broom::tidy(model)
  CL_store[i, ] = result[1,]
}

CL_binary_result =
  CL_store %>%
  mutate(term = predictor_name)

# BA data
column_name = colnames(BA_binary)
predictor_name = column_name[11:266]
BA_store = data.frame(matrix(ncol = 7, nrow = 0))
colnames(BA_store) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.hi
for(i in 1:256) {
  data = BA_binary[i+10]
  a = unlist(data)
  model = clogit(PROT_ASD_2015 ~ factor(a) + factor(BirthSeason) + factor(year_new) + factor(Sex) + str
  result = broom::tidy(model)
  BA_store[i, ] = result[1,]
}

BA_binary_result =
  BA_store %>%
  mutate(term = predictor_name)

# PM data
column_name = colnames(PM_binary)
predictor_name = column_name[11:193]
```

```r
PM_store = data.frame(matrix(ncol = 7, nrow = 0))
colnames(PM_store) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.hi
for(i in 1:183) {
  data = PM_binary[i+10]
  a = unlist(data)
  model = clogit(PROT_ASD_2015 ~ factor(a) + factor(BirthSeason) + factor(year_new) + factor(Sex) + stra
  result = broom::tidy(model)
  PM_store[i, ] = result[1,]
}

PM_binary_result =
  PM_store %>%
  mutate(term = predictor_name)

# OL data
column_name = colnames(OL_binary)
predictor_name = column_name[11:81]
OL_store = data.frame(matrix(ncol = 7, nrow = 0))
colnames(OL_store) = c("term", "estimate", "std.error", "statistic", "p.value",   "conf.low",  "conf.hi
for(i in 1:71) {
  data = OL_binary[i+10]
  a = unlist(data)
  model = clogit(PROT_ASD_2015 ~ factor(a) + factor(BirthSeason) + factor(year_new) + factor(Sex) + stra
  result = broom::tidy(model)
  OL_store[i, ] = result[1,]
}

OL_binary_result =
  OL_store %>%
  mutate(term = predictor_name)
```

```r
# Calculate the adjusted p-values
CL_binary_result$adjusted_pval = p.adjust(p = CL_binary_result$p.value, method = "BH")

# PM data
PM_binary_result$adjusted_pval = p.adjust(p = PM_binary_result$p.value, method = "BH")

# BA data
BA_binary_result$adjusted_pval = p.adjust(p = BA_binary_result$p.value, method = "BH")

# OL data
OL_binary_result$adjusted_pval = p.adjust(p = OL_binary_result$p.value, method = "BH")
```