Additional Checks on High Sparsity Data

Anthony Duncan

```r
library(tidyverse)
library(vegan)
library(glue)
library(ggpubr)
library(patchwork)
library(LorDist)
```

In our evaluations, both LorDist and mean Bray-Curtis dissimilarity continue to function to higher level of sparsity than in the Qi et al. benchmarks.

Here we check using some examples that our sparsification procedure seems to have worked as expected, and visualise LorDist and Bray-Curtis dissimilarities to see what drives this unexpected effect.

Check sparsification

As a first check, load a source matrix and all it's levels of sparsity, and ensure non-zero entries match.

```r
sparses <- seq(0, 0.7, 0.1) |>
    as.character()
mat_path <- glue("data/simulated_new/sparsity_{sparses}/1.tsv")
mats <- mat_path |> map(read.table)

compare_mats <- function(a, b) {
    return(
        all((b == 0) |
        (a == b))
    )
}
mat_wrong <- read.table("data/simulated_new/sparsity_0.1/2.tsv")

# Ensure this function works as intended
stopifnot(compare_mats(mats[[1]], mats[[1]]))
stopifnot(compare_mats(mats[[1]], mats[[2]]))
stopifnot(!compare_mats(mats[[2]], mat_wrong))

# Check all sparsified matrices are equal to source
mats[1:length(mats)] |>
    map(\(x) compare_mats(mats[[1]], x)) |>
    unlist() |>
    all()
```

```
## [1] TRUE
```

The sparsification appears to have been done correctly.

We should try plotting the taxa for the maximum level of sparsity and see if the functions are still evident.

```
# Load metadata to link samples to subjects
sample_md <- read_delim("data/simulated_new/sample_metadata.tsv") |>
  mutate(Phenotype = Label)
```

```
## Rows: 1000 Columns: 4
## ── Column specification ──────────────────────────────────────────────
──────────────────────────────────────────────────────────────────────
──────────────────────────────
## Delimiter: "\t"
## chr (3): SubjectID, SampleID, Label
## dbl (1): Timepoint
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
plt_taxa <- mats[[8]] |>
    rownames_to_column("FeatureID") |>
    pivot_longer(-FeatureID, names_to="SampleID") |>
    left_join(sample_md, join_by(SampleID == SampleID)) |>
    mutate(
      value = ifelse(value == 0, NA, value),
      FeatureID = fct_relevel(
        FeatureID, glue("Feature{i}", i=seq(1:200))
      )
    ) |>
    ggplot(
        aes(x = Timepoint, y = value)
    ) +
    geom_line(
        aes(group = SubjectID, color = Phenotype), linewidth=0.1, alpha=0.5
    ) +
    geom_smooth(
        aes(color = Phenotype)
    ) +
    facet_wrap(~FeatureID, ncol = 10) +
    theme_pubr() +
    ylab("Count") +
    theme(
        axis.ticks = element_blank(),
        axis.text = element_blank()
    )

ggsave("results/simulated/check_sparse_taxa.png", dpi=300, width=11, height=21)
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

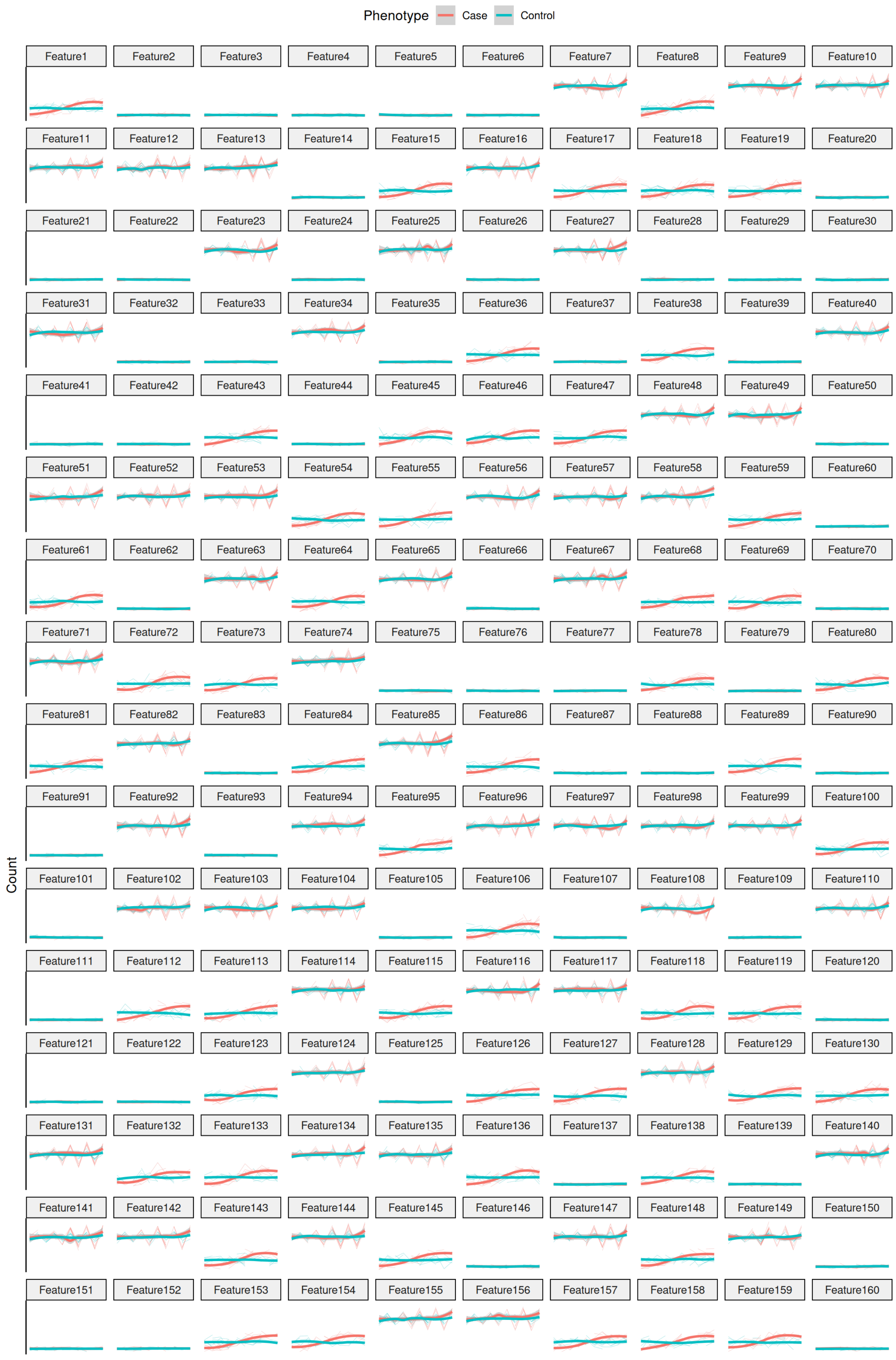
```
## Warning: Removed 140281 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

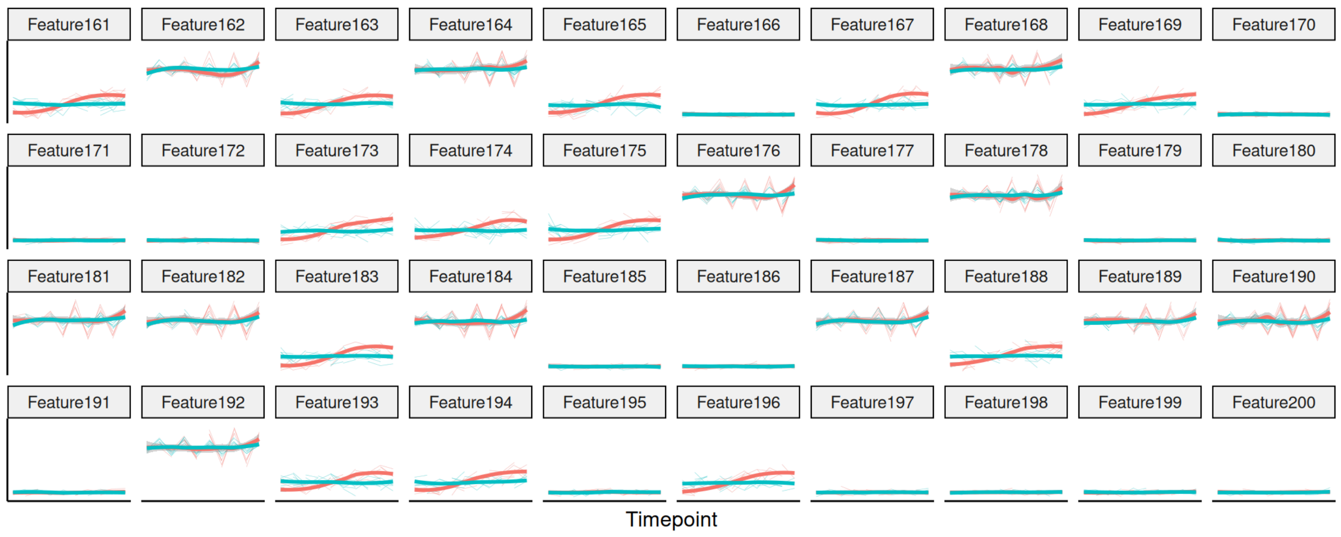
```
## Warning: Removed 507 rows containing missing values or values outside the scale
## range (`geom_line()`).
```

```
plt_taxa
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

```
## Warning: Removed 140281 rows containing non-finite outside the scale range (`stat_smooth()
`).
## Removed 507 rows containing missing values or values outside the scale range (`geom_line()
`).
```


Phenotype
Case
Control
Count
Feature1
Feature2
Feature3
Feature4
Feature5
Feature6
Feature7
Feature8
Feature9
Feature10
Feature11
Feature12
Feature13
Feature14
Feature15
Feature16
Feature17
Feature18
Feature19
Feature20
Feature21
Feature22
Feature23
Feature24
Feature25
Feature26
Feature27
Feature28
Feature29
Feature30
Feature31
Feature32
Feature33
Feature34
Feature35
Feature36
Feature37
Feature38
Feature39
Feature40
Feature41
Feature42
Feature43
Feature44
Feature45
Feature46
Feature47
Feature48
Feature49
Feature50
Feature51
Feature52
Feature53
Feature54
Feature55
Feature56
Feature57
Feature58
Feature59
Feature60
Feature61
Feature62
Feature63
Feature64
Feature65
Feature66
Feature67
Feature68
Feature69
Feature70
Feature71
Feature72
Feature73
Feature74
Feature75
Feature76
Feature77
Feature78
Feature79
Feature80
Feature81
Feature82
Feature83
Feature84
Feature85
Feature86
Feature87
Feature88
Feature89
Feature90
Feature91
Feature92
Feature93
Feature94
Feature95
Feature96
Feature97
Feature98
Feature99
Feature100
Feature101
Feature102
Feature103
Feature104
Feature105
Feature106
Feature107
Feature108
Feature109
Feature110
Feature111
Feature112
Feature113
Feature114
Feature115
Feature116
Feature117
Feature118
Feature119
Feature120
Feature121
Feature122
Feature123
Feature124
Feature125
Feature126
Feature127
Feature128
Feature129
Feature130
Feature131
Feature132
Feature133
Feature134
Feature135
Feature136
Feature137
Feature138
Feature139
Feature140
Feature141
Feature142
Feature143
Feature144
Feature145
Feature146
Feature147
Feature148
Feature149
Feature150
Feature151
Feature152
Feature153
Feature154
Feature155
Feature156
Feature157
Feature158
Feature159
Feature160



Functions are still visually apparent in the 70% sparsity matrix. The fitted line doesn't capture the periodic taxa very well, but they are visible in the individual traces.

We can also check that it is about 70% sparse.

```
sum(mats[[8]] == 0) / (dim(mats[[8]])[[1]] * dim(mats[[8]])[[2]])
```

```
## [1] 0.701405
```

Is this also what the package simulated data looks like?

There is some sample data provided with the LorDist package. Does this sparsified data look similar to what we have simulated?

```
load("data/source/simdata.RData")
ld_sim <- SimData.mat
ld_meta <- SimSample_information |>
    mutate(Phenotype = ifelse(Label == 1, "Case", "Control"))
plt_ld_taxa <- ld_sim |>
    rownames_to_column("FeatureID") |>
    pivot_longer(-FeatureID, names_to="SampleID") |>
    left_join(ld_meta, join_by(SampleID == SampleID)) |>
    mutate(
      value = ifelse(value == 0, NA, value),
      FeatureID = fct_relevel(
        FeatureID, glue("Feature{i}", i=seq(1:200))
      )
    ) |>
    ggplot(
        aes(x = Timepoint, y = value)
    ) +
    geom_line(
        aes(group = SubjectID, color = Phenotype), linewidth=0.1, alpha=0.7
    ) +
    geom_smooth(
        aes(color = Phenotype)
    ) +
    facet_wrap(~FeatureID, ncol = 10) +
    theme_pubr() +
    ylab("Count") +
    theme(
        axis.ticks = element_blank(),
        axis.text = element_blank()
    )

ggsave(
  "results/simulated/check_lordist_taxa.png",
  dpi = 300,
  width = 11,
  height = 21
)
```

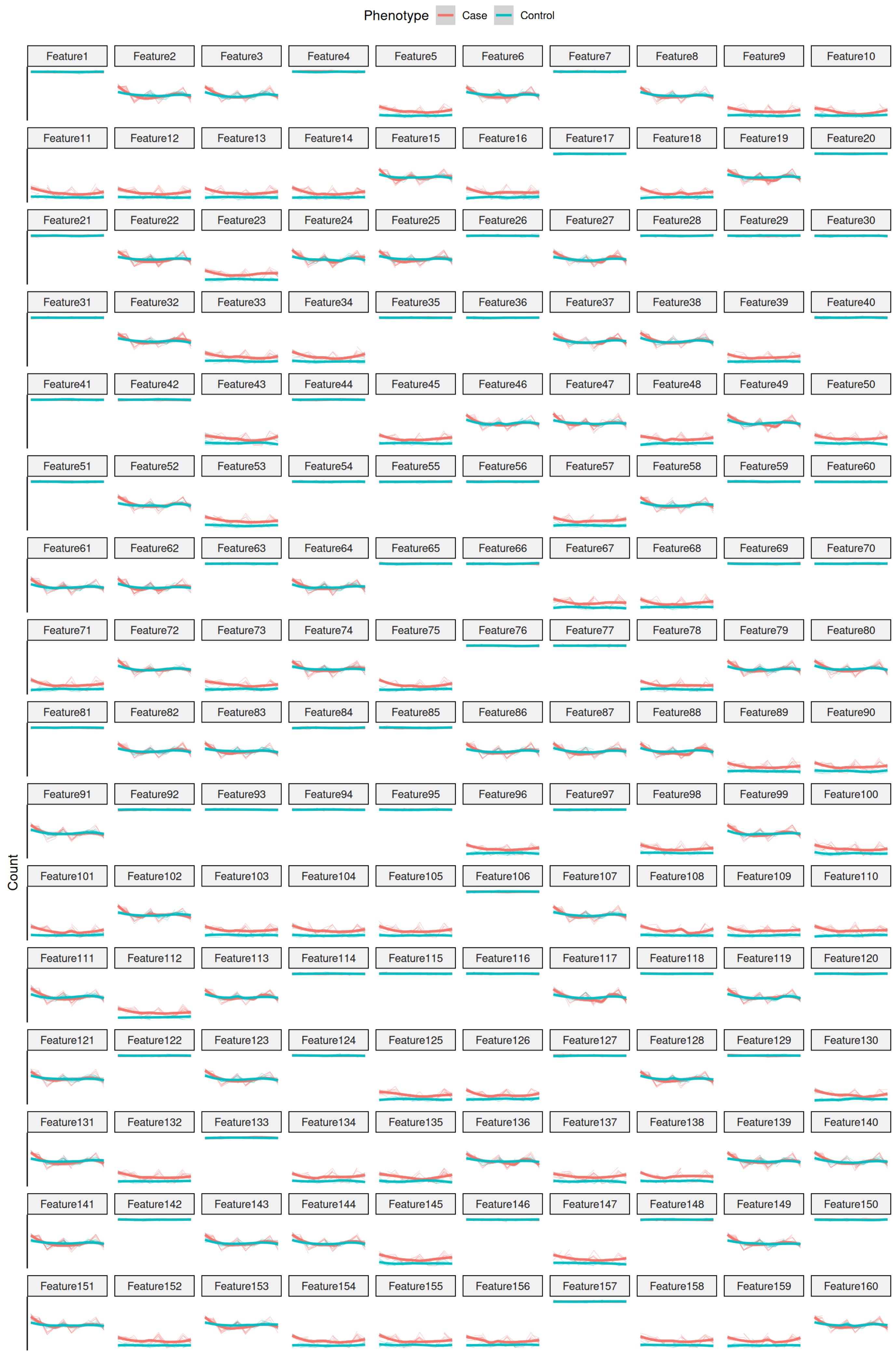
```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
```

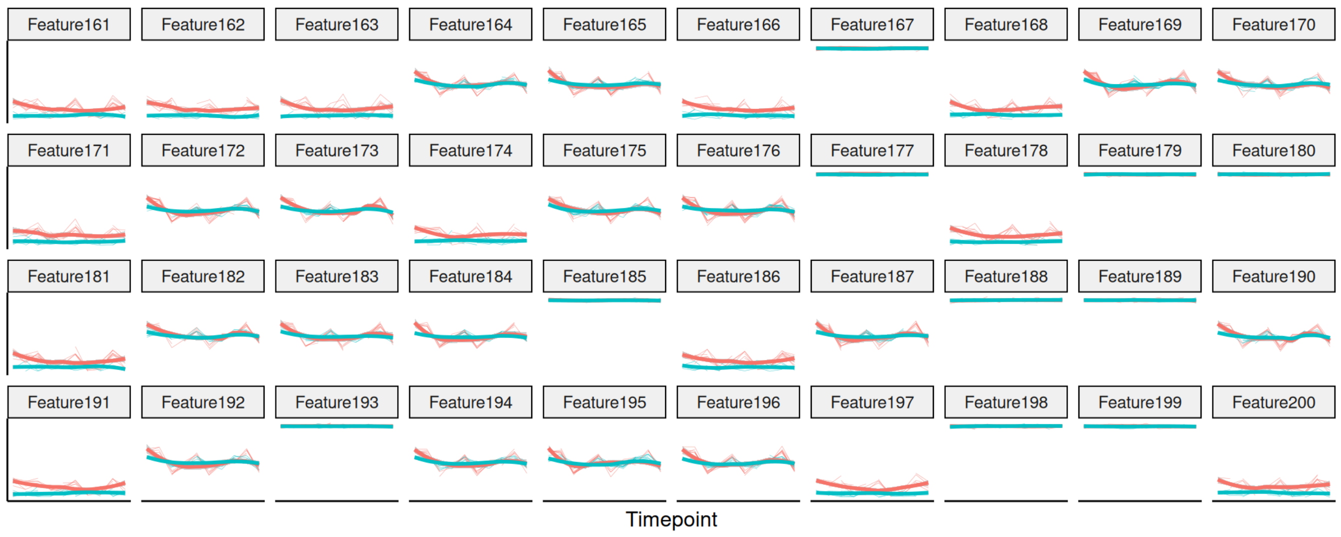
```
## Warning: Removed 25004 rows containing non-finite outside the scale range
## (`stat_smooth()`).
```

```
## Warning: Removed 44 rows containing missing values or values outside the scale
## range (`geom_line()`).
```

```
plt_ld_taxa
```

```
## `geom_smooth()` using method = 'loess' and formula = 'y ~ x'
## Warning: Removed 25004 rows containing non-finite outside the scale range (`stat_smooth()
`).
## Removed 44 rows containing missing values or values outside the scale range (`geom_line()
`).
```


Phenotype
Case
Control
Count
Feature1
Feature2
Feature3
Feature4
Feature5
Feature6
Feature7
Feature8
Feature9
Feature10
Feature11
Feature12
Feature13
Feature14
Feature15
Feature16
Feature17
Feature18
Feature19
Feature20
Feature21
Feature22
Feature23
Feature24
Feature25
Feature26
Feature27
Feature28
Feature29
Feature30
Feature31
Feature32
Feature33
Feature34
Feature35
Feature36
Feature37
Feature38
Feature39
Feature40
Feature41
Feature42
Feature43
Feature44
Feature45
Feature46
Feature47
Feature48
Feature49
Feature50
Feature51
Feature52
Feature53
Feature54
Feature55
Feature56
Feature57
Feature58
Feature59
Feature60
Feature61
Feature62
Feature63
Feature64
Feature65
Feature66
Feature67
Feature68
Feature69
Feature70
Feature71
Feature72
Feature73
Feature74
Feature75
Feature76
Feature77
Feature78
Feature79
Feature80
Feature81
Feature82
Feature83
Feature84
Feature85
Feature86
Feature87
Feature88
Feature89
Feature90
Feature91
Feature92
Feature93
Feature94
Feature95
Feature96
Feature97
Feature98
Feature99
Feature100
Feature101
Feature102
Feature103
Feature104
Feature105
Feature106
Feature107
Feature108
Feature109
Feature110
Feature111
Feature112
Feature113
Feature114
Feature115
Feature116
Feature117
Feature118
Feature119
Feature120
Feature121
Feature122
Feature123
Feature124
Feature125
Feature126
Feature127
Feature128
Feature129
Feature130
Feature131
Feature132
Feature133
Feature134
Feature135
Feature136
Feature137
Feature138
Feature139
Feature140
Feature141
Feature142
Feature143
Feature144
Feature145
Feature146
Feature147
Feature148
Feature149
Feature150
Feature151
Feature152
Feature153
Feature154
Feature155
Feature156
Feature157
Feature158
Feature159
Feature160



The counts in the provided simulated data do not appear to resemble the proposed generating function closely. None of the plots clearly resemble taxon 3, which is static in control and sigmoid in case. Some such as `Feature10` are similar, but have a sine-like wave with constant amplitude in case, rather than a sigmoid pattern. For taxon 2, the amplitude is less distinctly increasing over `t`, and is quite similar between case and control.

However, it is unclear whether this data from the repository was generated using the simulation method described, so perhaps it is just an alternative, similar set of example data.

How simple is it to distinguish phenotype?

We will look at how well a simple PCoA on mean B-C dissimilarity separates the two phenotypes in a high sparsity matrices, and compare this to LorDist. This repeats the procedure implemented in `03b_bc_benchmark.R` and `02_lordist_benchmark`, wrapped to a few functions.

```r
remove_empty_samples <- function(mat, sample_metadata) {
  #' Remove any completely empty samples from the abundance and metadata matrix
  #' This is more likely to happen at very high levels of sparsity
  empty_samples <- (mat |> colSums()) > 0
  if(sum(!empty_samples) > 0) {
    log_warn("Empty samples dues to sparsity: {sum(!empty_samples)}")
    log_warn("These samples are dropped from both the abundance and metadata")
    mat <- mat[, empty_samples]
    filt_md <- sample_metadata |>
        filter(SampleID %in% colnames(mat))
    return(list(matrix = mat, metadata = filt_md))
  }
  return(list(matrix = mat, metadata = sample_metadata))
}

make_subject_md <- function(sample_metadata) {
  #' Reduce sample metadata to one row per subject
  subject_md <- sample_md |>
    group_by(SubjectID) |>
    slice_head(n = 1) |>
    column_to_rownames("SubjectID")
  return(subject_md)
}

get_dist_meanbc <- function(mat, sample_metadata) {
  # Get subject rather thank sample metadata
  subject_md <- make_subject_md(sample_metadata)
  # Drop any completely empty samples
  drop_z <- remove_empty_samples(mat, sample_metadata)
  mat <- drop_z$matrix
  sample_metadata <- drop_z$metadata

  # Get per-sample Bray-Curtis
  dist <- vegdist(mat |> t(), method = "bray")

  # Calculate the average between subject Bray-Curtis
  dist_df <- dist |>
    as.matrix() |>
    as.data.frame() |>
    rownames_to_column("id_a") |>
    pivot_longer(
      !id_a,
      names_to = "id_b",
      values_to = "distance"
    ) |>
    dplyr::left_join(
      sample_md |> select(SampleID, SubjectID),
      join_by(id_a == SampleID)
    ) |>
    rename(group_a = SubjectID) |>
    dplyr::left_join(
      sample_md |> select(SampleID, SubjectID),
      join_by(id_b == SampleID)
    ) |>
    rename(group_b = SubjectID)
  # Easiest way to group these is to make this complete
  # Inelegant but achieves the goal
  dist_df2 <- dist_df
  dist_df2$group_b <- dist_df$group_a
  dist_df2$group_a <- dist_df$group_b
  dist_df2$id_a <- dist_df$id_b
  dist_df2$id_b <- dist_df$id_a

  dist_df <- bind_rows(dist_df, dist_df2)

  # Turn back into a distance matrix
  subject_dist <- dist_df |>
    group_by(group_a, group_b) |>
    summarise(distance = mean(distance)) |>
    pivot_wider(names_from = group_a, values_from = distance) |>
    column_to_rownames("group_b") |>
    as.matrix() |>
    as.dist()

  return(subject_dist)
}

get_dist_lordist <- function(mat, sample_metadata) {
  # Drop any completely empty samples
  drop_z <- remove_empty_samples(mat, sample_metadata)
  mat <- drop_z$matrix
  sample_metadata <- drop_z$metadata

  ldist <- LorDist(
    mat,
    sample_metadata |> as.data.frame(),
    SampleID = "SampleID",
    SubjectID = "SubjectID",
    Timepoint = "Timepoint"
  )

  return(ldist)
}

plot_pcoa <- function(dmat, sample_metadata, k = 20, ax1 = "V1", ax2 = "V2") {
  #' Plot two axes of PCoA, and boxplots showing separation of phenotype
  #' on different axes

  subject_md <- make_subject_md(sample_metadata)

  pcoa <- cmdscale(dmat, k = k)
  pcoa_df <- pcoa |>
      as.data.frame() |>
      rownames_to_column("SubjectID") |>
      left_join(subject_md |> rownames_to_column("SubjectID"))

  # Scatter plot
  plt_pcoa_bc <- pcoa_df |>
      ggplot(
          aes(x=!!sym(ax1), y=!!sym(ax2), color=Label, shape=Label)
      ) +
      geom_point() +
      xlab(gsub("V", "PC", ax1)) +
      ylab(gsub("V", "PC", ax2)) +
      theme_pubr()

  # Box plot
  pcoa_df_long <- pcoa |>
    as.data.frame() |>
    rownames_to_column("SubjectID") |>
    pivot_longer(-SubjectID) |>
    left_join(subject_md |> rownames_to_column("SubjectID")) |>
    mutate(name = fct_relevel(name, glue("V{i}", i=1:k)))
  plt_box <- pcoa_df_long |>
    ggplot(
      aes(x=Phenotype, y=value, fill=Phenotype)
    ) +
    geom_boxplot() +
    facet_wrap(~name) +
    scale_y_continuous(expand = expansion(mult=c(0.05, 0.1))) +
    stat_compare_means()
  return(list(
    scatter = plt_pcoa_bc,
    box = plt_box
  ))
}

combine_pcoa_plots <- function(
    ld, bc, sparsity
) {
  plt_combined <- (
    bc + ggtitle("Mean B-C Dissimilarities")
  ) + (
    ld + ggtitle("LorDist Distances")
  ) +
    plot_annotation(
      title = glue("PCoA at {sparsity}% Sparsity"),
      tag_levels = "A"
    ) +
    plot_layout(guides = 'collect') &
    theme(legend.position = "top")
  return(plt_combined)
}
```

70% Sparsity

```r
mat <- mats[[8]]

ldist <- get_dist_lordist(mat, sample_md)
bcdist <- get_dist_meanbc(mat, sample_md)
```

```
## `summarise()` has grouped output by 'group_a'. You can override using the
## `.groups` argument.
```

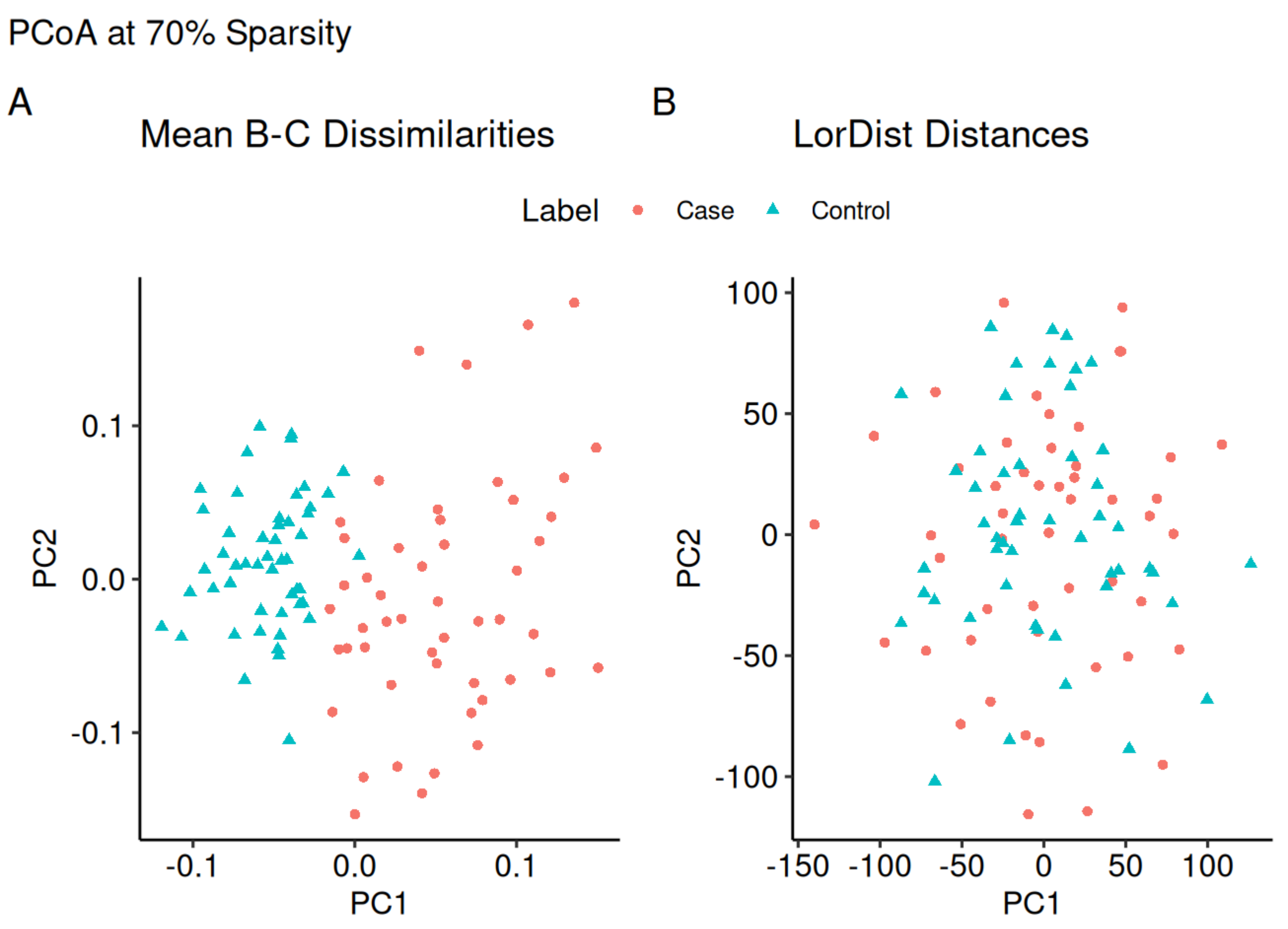
```
plt_ldist <- plot_pcoa(ldist, sample_metdata, k = 20)
```

```
## Joining with `by = join_by(SubjectID)`
## Joining with `by = join_by(SubjectID)`
```

```
plt_bcdist <- plot_pcoa(bcdist, sample_metdata, k = 20)
```

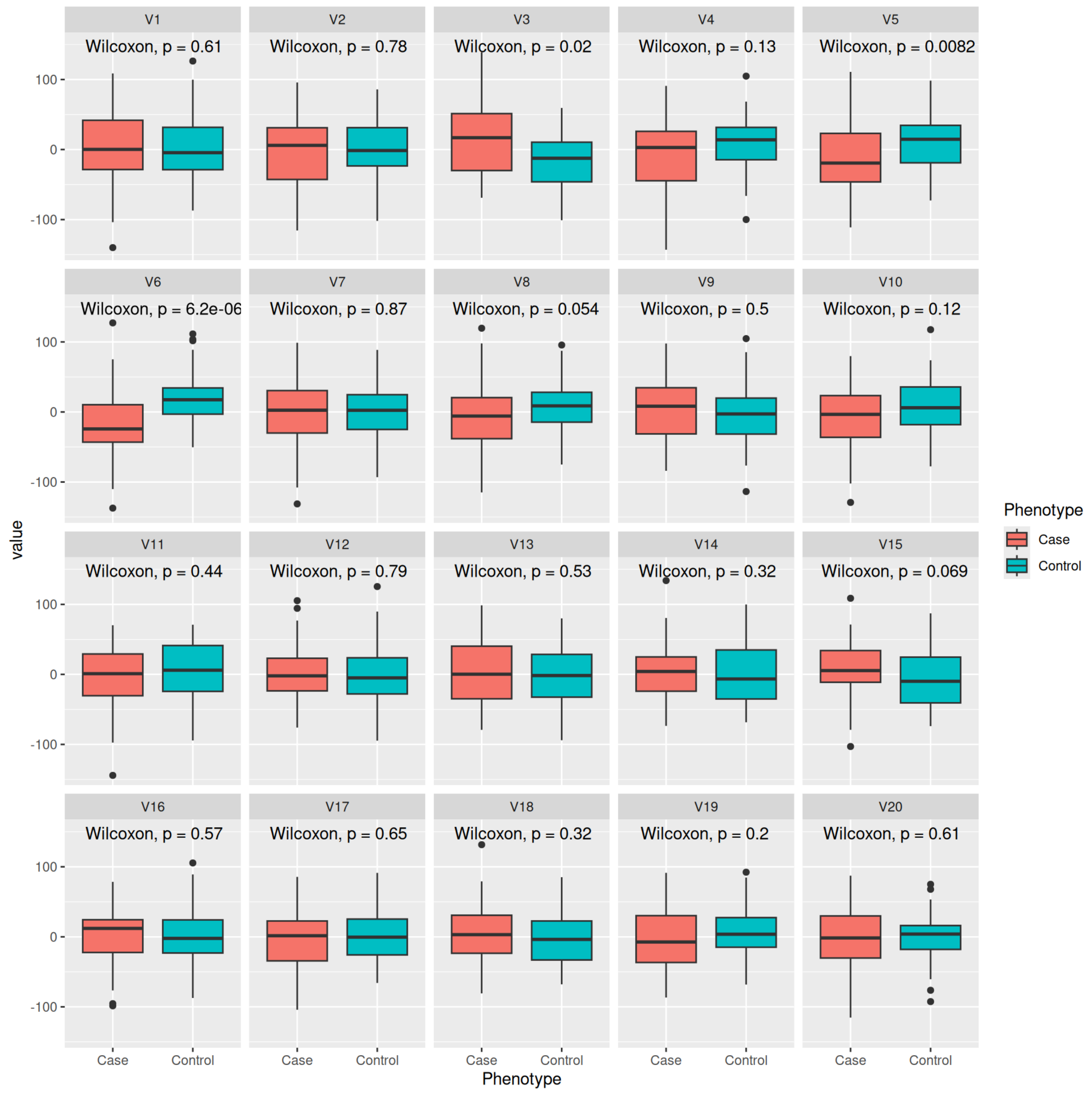
```
## Joining with `by = join_by(SubjectID)`
## Joining with `by = join_by(SubjectID)`
```

```
combine_pcoa_plots(plt_ldist$scatter, plt_bcdist$scatter, sparsity = 70)
```



Bray-Curtis shows clear separation on axis 1. None evident for LorDist. We can check the box plots to see if there is a better separation available on other axes.

```
plt_ldist$box
```

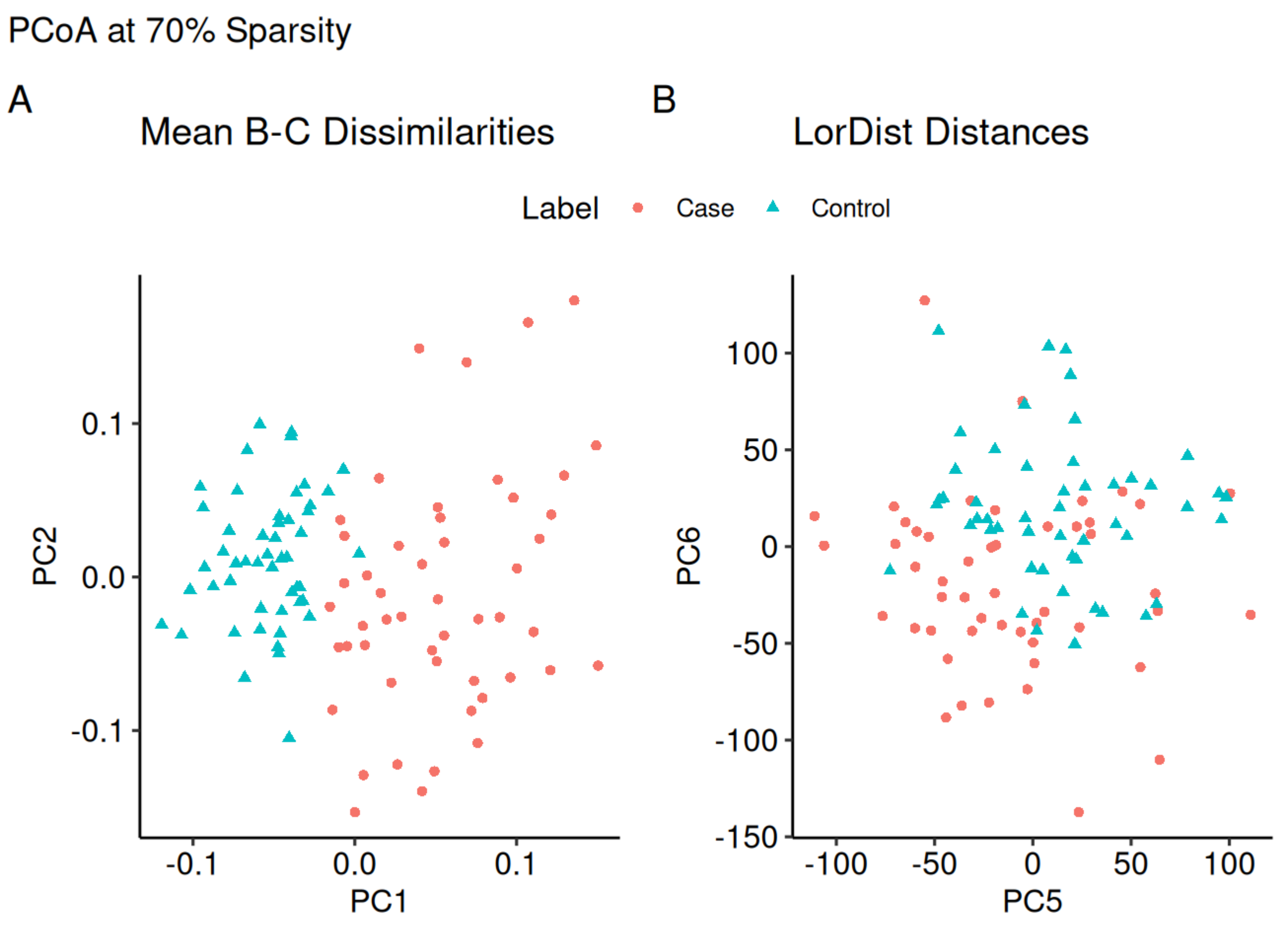


There is a significant difference ($p < 0.05$) on axes 3, 5, and 6, so we should plot these instead. 5 and 6 show the strongest effect so plot those.

```
plt_ldist <- plot_pcoa(ldist, sample_metdata, k = 20, ax1 = "V5", ax2 = "V6")
```

```
## Joining with `by = join_by(SubjectID)`
## Joining with `by = join_by(SubjectID)`
```

```
combine_pcoa_plots(plt_ldist$scatter, plt_bcdist$scatter, sparsity = 70)
```



80% Sparsity

We have generated some additional data, which were not included in the benchmarks, with 80%, 90% and 95% sparse data.

```
sample_md <- read_delim("data/simulated_high/sample_metadata.tsv") |>
  mutate(Label = Phenotype)
```

```
## Rows: 1000 Columns: 4
## ── Column specification ────────────────────────────────────────────────────────
──────────────────────────────────────────────────────────────────────────────────
─────────────────────────────────────

## Delimiter: " "
## chr (3): SubjectID, SampleID, Phenotype
## dbl (1): Timepoint
##
## ℹ Use `spec()` to retrieve the full column specification for this data.
## ℹ Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
high_80 <- read.table("data/simulated_high/sparsity_0.8/1.tsv")
high_90 <- read.table("data/simulated_high/sparsity_0.9/1.tsv")
high_95 <- read.table("data/simulated_high/sparsity_0.95/1.tsv")
```

```
ld_80 <- get_dist_lordist(high_80, sample_md)
bc_80 <- get_dist_meanbc(high_80, sample_md)
```

```
## `summarise()` has grouped output by 'group_a'. You can override using the
## `.groups` argument.
```

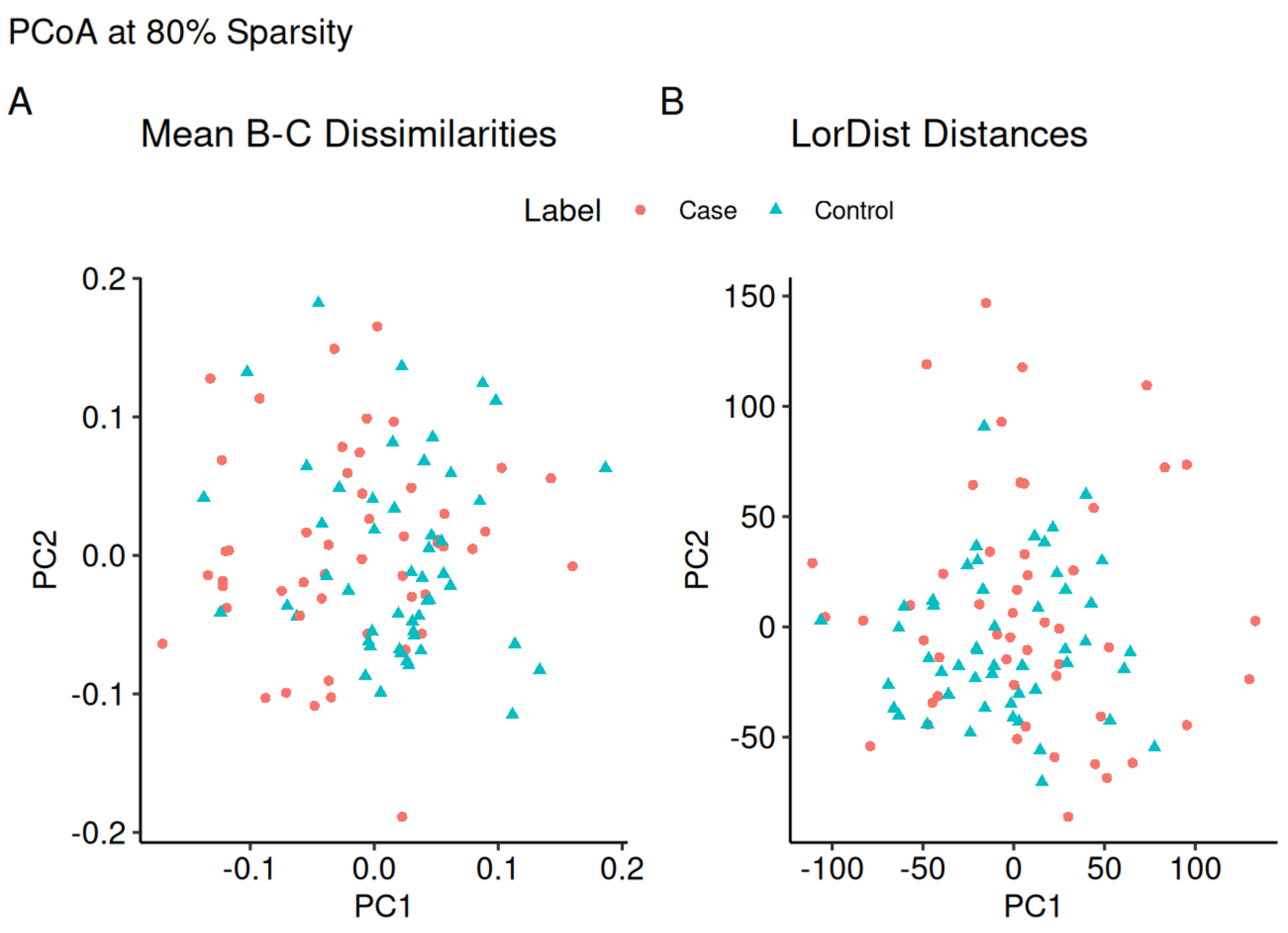
```
plt_ld_80 <- plot_pcoa(ld_80, sample_metdata, k = 20)
```

```
## Joining with `by = join_by(SubjectID)`
## Joining with `by = join_by(SubjectID)`
```

```
plt_bc_80 <- plot_pcoa(bc_80, sample_metdata, k = 20)
```

```
## Joining with `by = join_by(SubjectID)`
## Joining with `by = join_by(SubjectID)`
```

```
combine_pcoa_plots(plt_ld_80$scatter, plt_bc_80$scatter, sparsity = 80)
```



Poor results at 80% sparsity. We can check whether there's any evidence of a difference with PERMANOVA

```
prbc <- adonis2(
  bc_80 ~ Label,
  make_subject_md(sample_md),
  permutations = 1000
)
prld <- adonis2(
  ld_80 ~ Label,
  make_subject_md(sample_md),
  permutations = 1000
)
prbc['Model', 'R2']
```

```
## [1] 0.01266628
```

```
prld['Model', 'R2']
```

```
## [1] 0.01223502
```

There is some, explaining only 1.2% of variance.