omicplotR

Daniel Giguere

2017-10-13

Contents

What is omicplotR?

`omicplotR` is an R package containing a Shiny app used to visually explore omic datasets, where the input is a table of read counts derived from high-throughput sequencing runs. It integrates `ALDEx2` for compositional data analysis of differential abundance. The Shiny user interface allows easy and fast visualization of data for both users with and without experience at the R command line. `omicplotR` speeds up the process of exploring an omic datasets by providing a graphical user interface for the user.

Introduction

This guide provides an overview of the R package `omicplotR` for visual exploration of omic datasets, using both the command line and the Shiny app. It was developed for several types of data including RNASeq, meta RNASeq, and 16s rRNA gene sequencing, and can be used with various types of data with the same underlying assumptions. `omicplotR` is especially useful for datasets that have associated metadata, which can be used to visually explore all possible associations between variables.

Installation

Example

Load omicplotR and example datasets.

```
library(omicplotR)
data(otu_table)
data(metadata)
```

Once loaded, the first step is to filter your data. If you want to compare your variables provided in a column by your metadata, omicplotR provides a way to filter your data by these variables. This will return a data frame containing only the samples that have the values selected in the metadata file. This **must** be done first if you choose to do so, it can be skipped if no metadata is available or if you don't want to filter initially.

```
d.mf <- omicplotr.metadataFilter(data = otu_table, meta = metadata,
                                 column = "probio", values = c("y", "n"))
```

Next, you can remove samples or features that are below a read count threshold. This step is done to remove sparse data, and can be based on several different parameters. In this instance, `min.count` is used to remove features (rows) with a maximum value lower than 10, removing rows that are mostly zeros or have an extremely low count. This can also be applied to samples (columns) using the `min.reads` option to remove samples with a sum lower than the input. The goal here is to remove very sprase data; different values should be tried to explore how removing sparse data affects your the structure of your biplots. Here, we will remove any rows with a maximum value lower than 10.

```
d.f <- omicplotr.filter(data = d.mf, min.reads = 10)
```

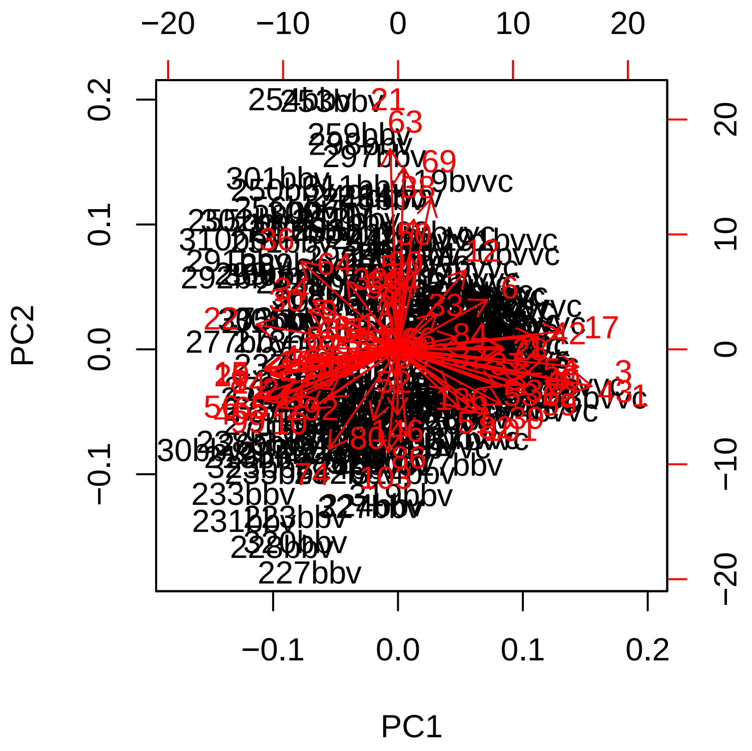
After removing sparse features, generate an object for plotting a compositional biplot. `omicplotr.clr` will replace any zeros in the counts table using the CZM method from the package `zCompositions`, transform the data to a centered log-ratio, and generate a `prcomp` object used for making the biplot.

```
d.clr <- omicplotr.clr(data = d.f)
```

```
## No. corrected values:  1921
```

You can now explore your dataset with a principal component analysis (PCA) biplot! If you don't have metadata that you want to colour by, this can be done simply with `biplot`, a function in base R.

```
biplot(d.clr)
```



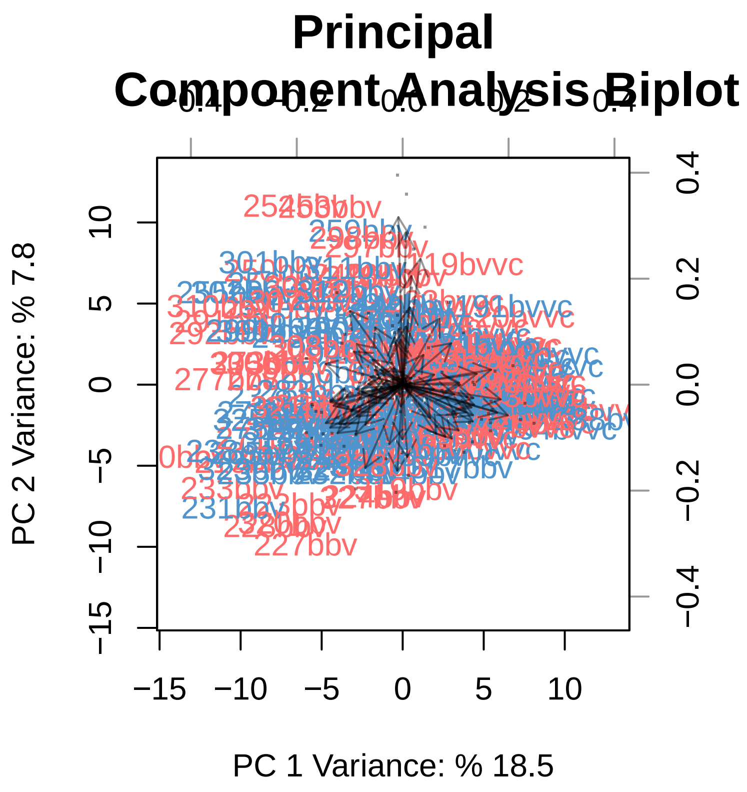
However, if you do have metadata, you will need to decide by what variable (column) in the metadata you want to colour. For this dataset, we will colour by whether the participants were taking probiotics as their treatment. Each unique variable will be assigned a colour, where black represents the value NA or not present in the dataset. `type = 3` indicates categorical variables, by default. Change this to `type = 1` for continuous data (ie, age) or by quartile.

Generate a vector of colours for your dataset using the metadata file.

```
colvec <- omicplotr.colvec(data = d.clr, meta = metadata, column = "probio", type = 3)
```

t's finally time plot! Use `omicplotr.colouredPCA` to generate the coloured biplot (this wraps colouredBiplot from `compositions`. Several graphical options are available. Here, we will display the features as their genera. You must include at a minimum the output from `omicplotr.clr` and `omicplotr.colvec`.

```
omicplotr.colouredPCA(d.clr, colourvector = colvec)
```


Principal
Component Analysis Biplot
PC 1 Variance: % 18.5
PC 2 Variance: % 7.8
−15
−10
−5
0
5
10
−0.4
−0.2
0.0
0.2
0.4
259bbv
301bbv
219bvvc
277bbv
233bbv
231bbv
227bbv