Analogues to `plot_ordination`

Vignette Author

2015-04-15

The main inspiration for the `mvarVis` package comes from the `plot_ordination` command in package `phyloseq`. Given the result of an ordination, we should be able to plot various features onto the projected points easily. `plot_ordination` implements this idea in the case that the ordination has features from a `phyloseq` object, but the idea can be applied more generally.

To work in this wider context, we currently have a 3-step pipeline

- Convert the result of an `ade4` or `vegan` ordination into a class `mvarTable`. Having this class means we don't have to rewrite plotting or annotation code for every kind of ordination package.
- Annotate the results of the ordination. By this we mean give covariates that describe each of the projected points (like Phylum in the microbiome context).
- Plot the annotated `mvarTable` object. The goal is to provide flexibility in mapping aesthetic attributes without requiring too much work from the user.

`plot_ordination` analogues

To get an idea for how this package works, here are some examples re-implementing the methods in `plot_ordination` from the phyloseq documentation.

Loading / reshaping data

First we load the data from the linked page.

```
library("ape")
library("ade4")
```

```
##
## Attaching package: 'ade4'
##
## The following object is masked from 'package:vegan':
##
##     cca
```

```
library("phyloseq")
library("vegan")
library("mvarVis")
```

```
data("GlobalPatterns")
GP = GlobalPatterns
wh0 = genefilter_sample(GP, filterfun_sample(function(x) x > 5), A = 0.5 * nsamples(GP))
GP1 = prune_taxa(wh0, GP)
GP1 = transform_sample_counts(GP1, function(x) 1e+06 * x/sum(x))
phylum.sum = tapply(taxa_sums(GP1), tax_table(GP1)[, "Phylum"], sum, na.rm = TRUE)
```

```
top5phyla = names(sort(phylum.sum, TRUE))[1:5]
GP1 = prune_taxa((tax_table(GP1)[, "Phylum"] %in% top5phyla), GP1)
human = get_variable(GP1, "SampleType") %in% c("Feces", "Mock", "Skin", "Tongue")
sample_data(GP1)$human <- factor(human)
```

vegan methods

This is a vegan_to_mvar object that should make it easy to use this package with vegan.

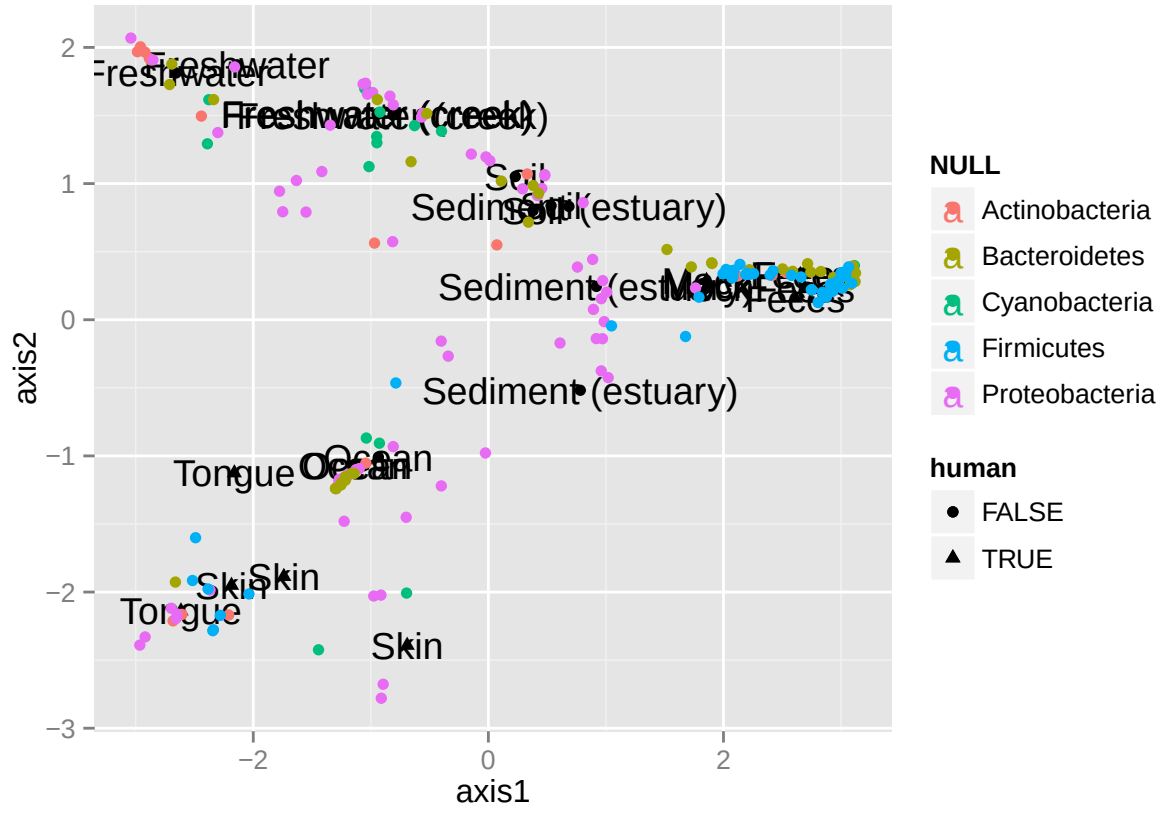
```
gp1_nmds <- vegan::metaMDS(GP1@otu_table, dist = "bray")
```

```
## Square root transformation
## Wisconsin double standardization
## Run 0 stress 0.1963138
## Run 1 stress 0.2179186
## Run 2 stress 0.2522687
## Run 3 stress 0.2353926
## Run 4 stress 0.2615103
## Run 5 stress 0.1959485
## ... New best solution
## ... procrustes: rmse 0.006836193  max resid 0.08727623
## Run 6 stress 0.1959288
## ... New best solution
## ... procrustes: rmse 0.002433995  max resid 0.02634918
## Run 7 stress 0.2243421
## Run 8 stress 0.2166755
## Run 9 stress 0.2161112
## Run 10 stress 0.196289
## ... procrustes: rmse 0.006388412  max resid 0.08741264
## Run 11 stress 0.2685716
## Run 12 stress 0.196289
## ... procrustes: rmse 0.006384503  max resid 0.08741415
## Run 13 stress 0.1959485
## ... procrustes: rmse 0.002435519  max resid 0.02635803
## Run 14 stress 0.2055421
## Run 15 stress 0.1959296
## ... procrustes: rmse 0.0002159503  max resid 0.002257008
## *** Solution reached
```

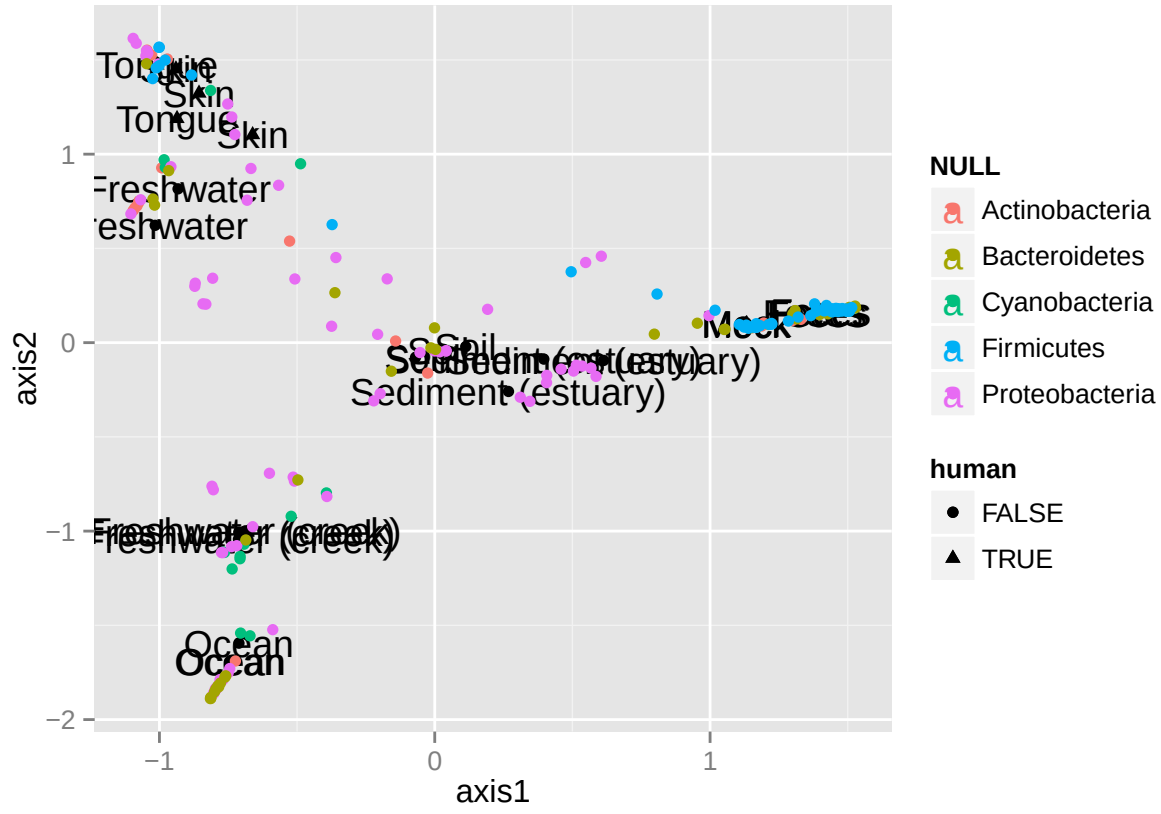
```
gp1_nmds_mvar <- vegan_to_mvar(gp1_nmds)
gp1_nmds_mvar <- mvar_annotate(gp1_nmds_mvar, list(GP1@tax_table, GP1@sam_data))
p <- plot_mvar(gp1_nmds_mvar, color_vars = list("Phylum", NULL),
               arrow_ix = numeric(0), text_ix = 2, text_vars = list(NULL, "SampleType"),
               shape_vars = list(NULL, "human"))

# DCA
gp1_dca <- vegan::decorana(GP1@otu_table)
gp1_dca_mvar <- vegan_to_mvar(gp1_dca)
gp1_dca_mvar <- mvar_annotate(gp1_dca_mvar, list(GP1@tax_table, GP1@sam_data))
p <- plot_mvar(gp1_dca_mvar, color_vars = list("Phylum", NULL),
               arrow_ix = numeric(0), text_ix = 2, text_vars = list(NULL, "SampleType"),
```

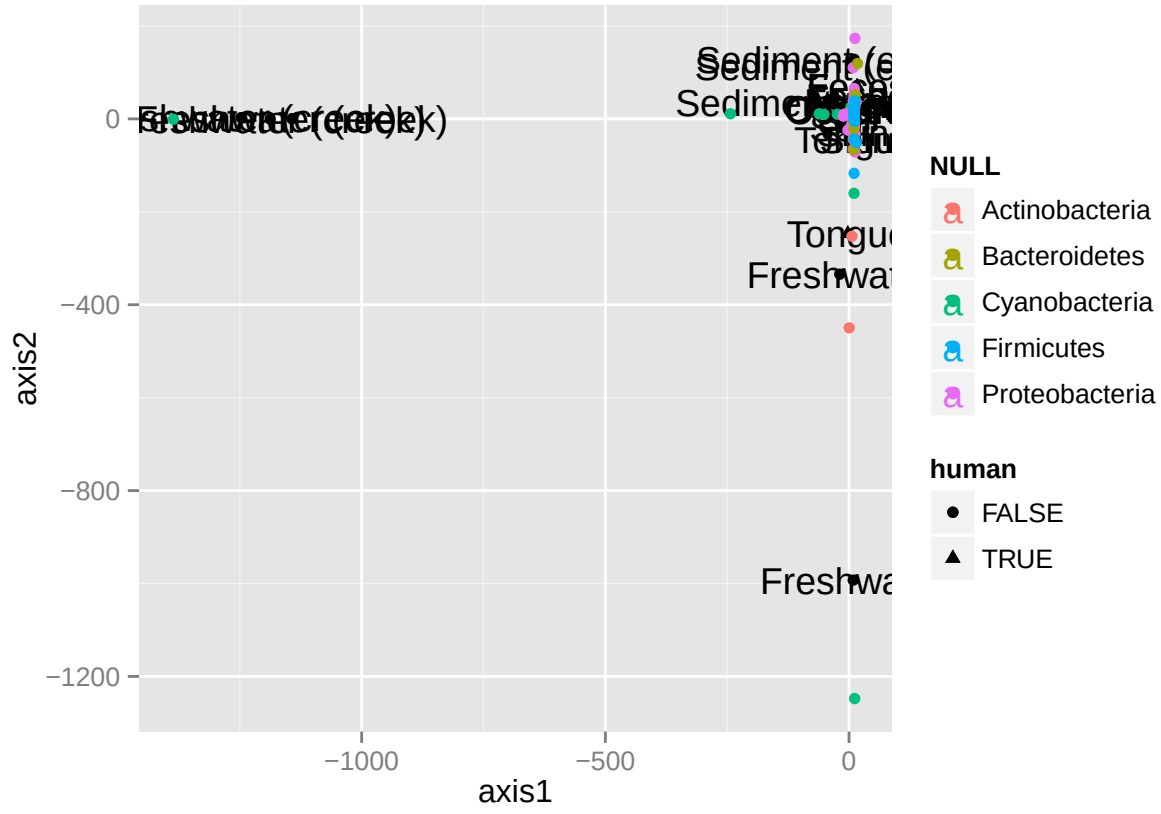
```
                shape_vars = list(NULL, "human"))
p
```




```
# cca
gp1_cca <- vegan::cca(GP1@otu_table)
gp1_cca_mvar <- vegan_to_mvar(gp1_cca)
gp1_cca_mvar <- mvar_annotate(gp1_cca_mvar, list(GP1@tax_table, GP1@sam_data))
p <- plot_mvar(gp1_cca_mvar, color_vars = list("Phylum", NULL),
               arrow_ix = numeric(0), text_ix = 2, text_vars = list(NULL, "SampleType"),
               shape_vars = list(NULL, "human"))
p
```



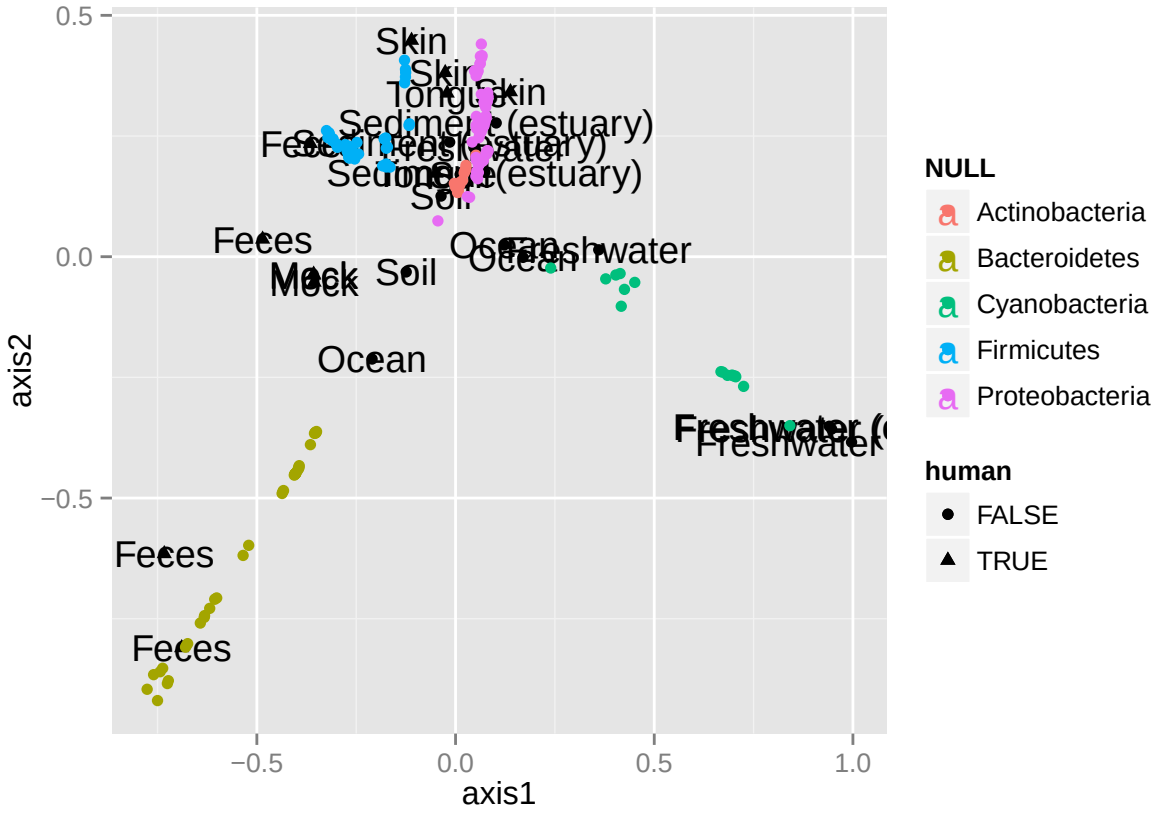

```
# rda
gp1_rda <- vegan::rda(GP1@otu_table)
gp1_rda_mvar <- vegan_to_mvar(gp1_rda)
gp1_rda_mvar <- mvar_annotate(gp1_rda_mvar, list(GP1@tax_table, GP1@sam_data))
p <- plot_mvar(gp1_rda_mvar, color_vars = list("Phylum", NULL),
               arrow_ix = numeric(0), text_ix = 2, text_vars = list(NULL, "SampleType"),
               shape_vars = list(NULL, "human"))
p
```



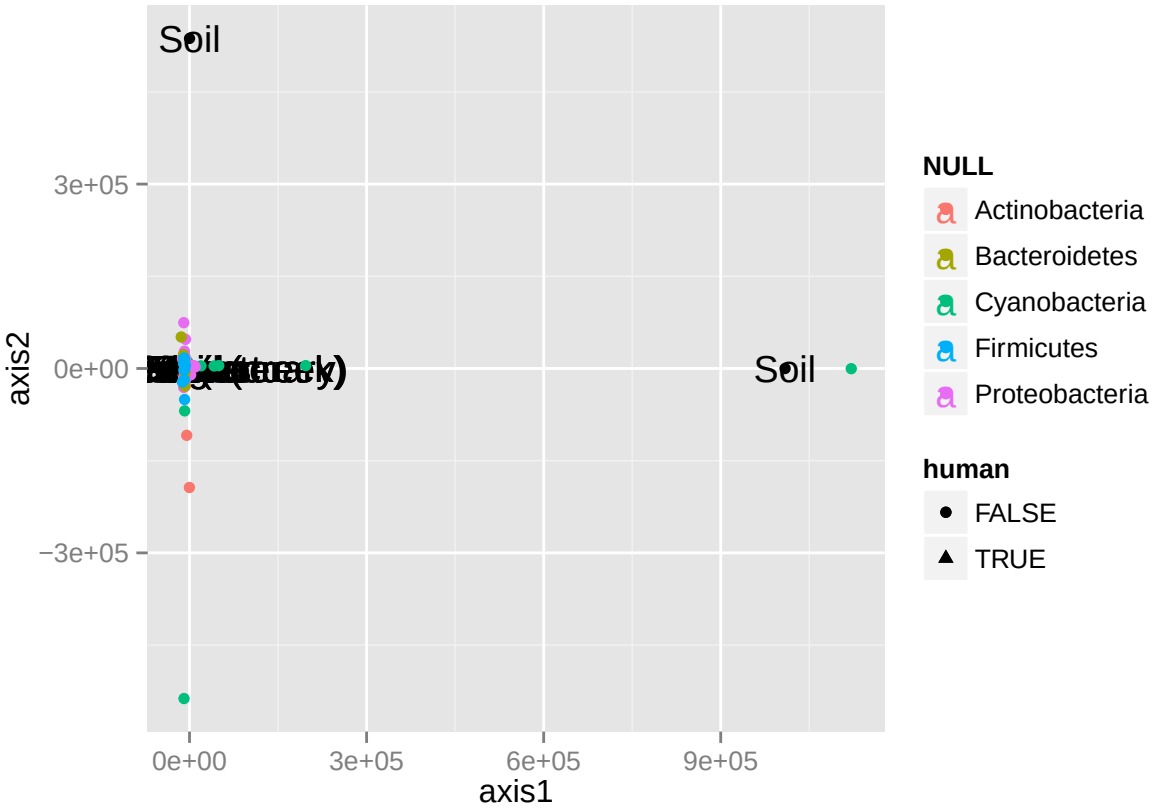
ade4 methods

There is an analogous function for ade4.

```
# DPCoA
patristicDist <- cailliez(as.dist(cophenetic.phylo(phy_tree(GP1))))
gp1_dpcoa <- dpcoa(data.frame(GP1@otu_table), patristicDist, scannf = F, nf = 2)
gp1_dpcoa_mvar <- ade4_to_mvar(gp1_dpcoa, tables_to_include = c("l1", "l2"))
gp1_dpcoa_mvar <- mvar_annotate(gp1_dpcoa_mvar, list(GP1@tax_table, GP1@sam_data))
plot_mvar(gp1_dpcoa_mvar, color_vars = list("Phylum", NULL), arrow_ix = numeric(0),
          text_ix = 2, text_vars = list(NULL, "SampleType"), shape_vars = list(NULL, "human"))
```




```
# MDS
gp1_mds <- dudi.pco(dist(GP1@otu_table), scannf = F, nf = 2)
gp1_mds_mvar <- ade4_to_mvar(gp1_mds)
gp1_mds_mvar <- mvar_annotate(gp1_mds_mvar, list(GP1@tax_table, GP1@sam_data))
plot_mvar(gp1_mds_mvar, color_vars = list("Phylum", NULL), arrow_ix = numeric(0),
          text_ix = 2, text_vars = list(NULL, "SampleType"), shape_vars = list(NULL, "human"))
```



I haven't implemented the pcoa example yet, since this needs a conversion from an `ape` specific format.