logFC data munging

Nate Olson

2017-09-28

```
library(ProjectTemplate)
cwd <- getwd()
setwd("../")
load.project()
```

```
## Warning in .load.config(override.config): Your configuration file
## is missing the following entries: data_loading_header, data_ignore,
## logging_level, cache_loaded_data, sticky_variables. Defaults will be used.
```

```
## Warning in .check.version(config): Your configuration is compatible with version 0.7 of the ProjectT
##   Please run ProjectTemplate::migrate.project() to migrate to the installed version 0.8.
```

```
setwd(cwd)
pipeline_dir <- "../../mgtst_pipelines"
mrexp <- get_mrexp(pipeline_dir)
```

Objective

Workout code for generating data frame with estimated logFC and variance between titrations for the different pipelines and differential abundance methods. Currently using the default normalization for each differential abundance detection method.

Pipelines: 1. DADA2
2. Mothur
3. QIIME
4. Unclustered

Differential Abundance Methods:
1. metagenomeSeq - fitFeatureModel
2. DESeq2
3. EdgeR

Approach

Use a data_frame object and list columns to apply differential abundance methods to different datasets

Outline

1. Define parameter fields for titration comparisons
2. Develop function for each differential abundance method
3. Save data frames with log fold-change calculations for each differential abundance detection method.

Titration Comparisons

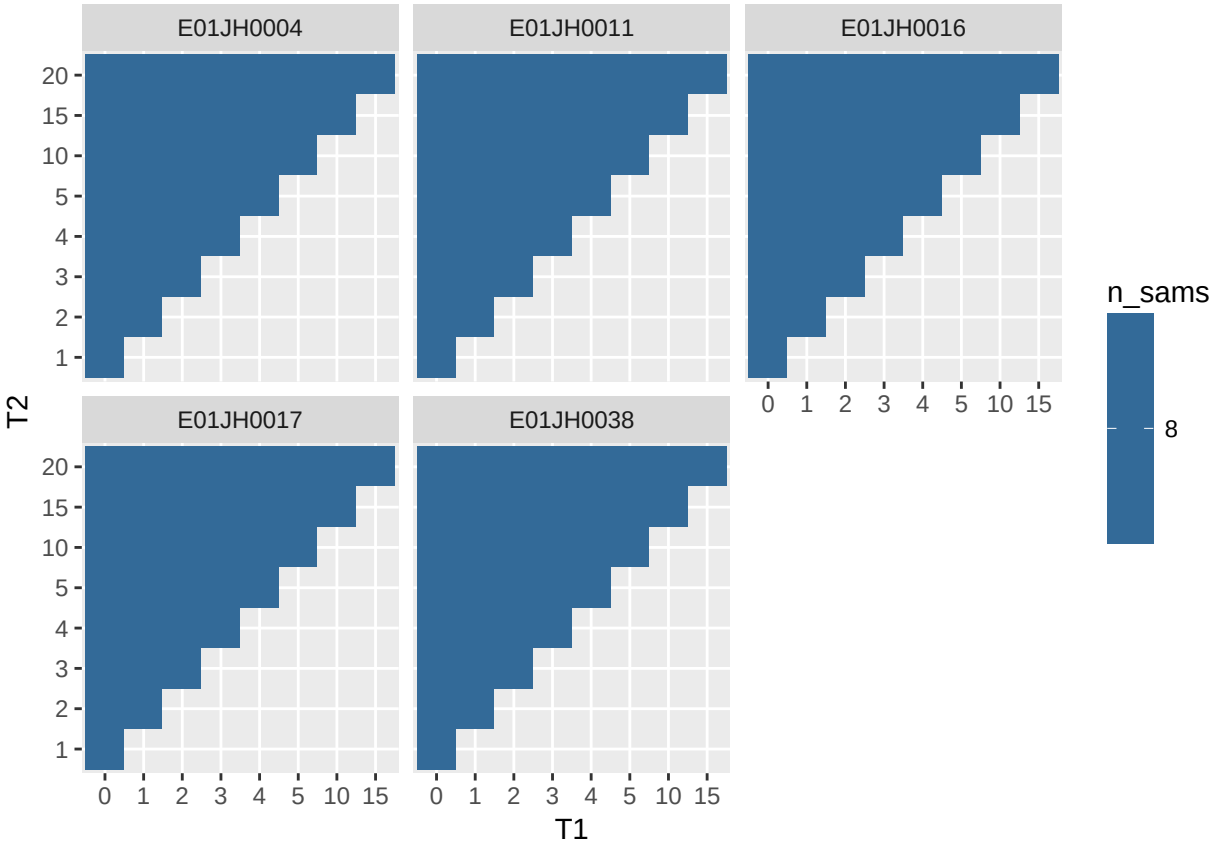
```
pdat <- mrexp$dada2 %>% pData %>%
      mutate(t_fctr = factor(t_fctr, level = c(0:5,10,15,20)))
t1_pdat <- pdat %>% select(biosample_id, id, t_fctr) %>%
      rename(T1 = t_fctr, T1_id = id)
t2_pdat <- pdat %>% select(biosample_id, id, t_fctr) %>%
      rename(T2 = t_fctr, T2_id = id)
titration_comp_df <- left_join(t1_pdat, t2_pdat) %>%
      filter(as.numeric(T1) < as.numeric(T2)) %>%
      group_by(biosample_id, T1, T2) %>%
      summarise(sam_names = c(T1_id,T2_id) %>% unique() %>% list(),
                n_sams = c(T1_id,T2_id) %>% unique() %>% length())
```

```
## Joining, by = "biosample_id"
```

Comparison for all titration combinations for each biological replicate. Total of 180 sets of logFC calculations.

Pairwise comparisons sanity check, expectation T1 always less than T2 with 8 samples for each pairwise comparison

```
titration_comp_df %>% ggplot() +
      geom_raster(aes(x = T1, y = T2, fill = n_sams)) +
      facet_wrap(~biosample_id)
```




```
titration_comp_df %>% filter(biosample_id == "E01JH0004", T1 == 0)
```

```
## # A tibble: 8 x 5
## # Groups:   biosample_id, T1 [1]
```

```
##   biosample_id     T1     T2 sam_names n_sams
##          <chr> <fctr> <fctr>    <list>  <int>
## 1    E01JH0004      0      1 <chr [8]>      8
## 2    E01JH0004      0      2 <chr [8]>      8
## 3    E01JH0004      0      3 <chr [8]>      8
## 4    E01JH0004      0      4 <chr [8]>      8
## 5    E01JH0004      0      5 <chr [8]>      8
## 6    E01JH0004      0     10 <chr [8]>      8
## 7    E01JH0004      0     15 <chr [8]>      8
## 8    E01JH0004      0     20 <chr [8]>      8
```

Subset MRexperiments

For each set of logFC calculations subset the MRexperiment objects

```
make_titration_comp_subset_df <- function(mrexp_obj, titration_comp_df){

      ## Subsetting mrexp for titrations
      subset_mrexp <- function(sam_names){
            mrexp_obj %>% {.[,which(colnames(.) %in% sam_names)]}
      }

      ## Dataframe with list of subsetted mrexp
     titration_comp_df %>%
            mutate(mrexp_sub = map(sam_names, subset_mrexp))
}

### DADA2
titration_comp_dada2_df <- mrexp$dada2 %>%
  make_titration_comp_subset_df(titration_comp_df)

### Mothur
titration_comp_mothur_df <- mrexp$mothur %>%
      make_titration_comp_subset_df(titration_comp_df)

### QIIME
titration_comp_qiime_df <- mrexp$qiime %>%
      make_titration_comp_subset_df(titration_comp_df)

### Unclustered
titration_comp_unclustered_df <- mrexp$unclustered %>%
      make_titration_comp_subset_df(titration_comp_df)
```

Estimate logFC

MetagenomeSeq - fitFeatureModel

Functions for calculating logFC using fitFeatureModel

```
fit_model <- function(mrexp_sub, T1, T2, css_p = 0.75, present = 8, depth = 1){
      ## Normalize count data
      mrexp_sub <-  cumNorm(mrexp_sub, p = css_p)
```

```
      # Exclude features with counts not observed in all PCR replicates
      mrexp_sub2 <- filterData(mrexp_sub, present = present, depth = depth)

      pd <- pData(mrexp_sub2)

      ## Fitting Model
      mod <- model.matrix(~1 + t_fctr, data = pd)
      fitFeatureModel(mrexp_sub2, mod)
}

fit_safely <- safely(fit_model)
mrcoefs_safely <- safely(MRcoefs)
```

DADA2

```
logFC_MgSeq_dada2_df <- titration_comp_dada2_df %>%
      mutate(fit = pmap(list(mrexp_sub, T1, T2), fit_safely))

logFC_MgSeq_dada2_coefs_df <- logFC_MgSeq_dada2_df %>%
      mutate(fit_coefs = map(fit, ~mrcoefs_safely(.$result, number = Inf)))

logFC_MgSeq_dada2_coefs_tidy <- logFC_MgSeq_dada2_coefs_df %>%
      select(biosample_id, T1, T2, fit_coefs) %>%
      mutate(fit_coefs = map(fit_coefs, ~.$result),
             fit_coefs_class = map_chr(fit_coefs, class)) %>%
      filter(fit_coefs_class == "data.frame") %>%
      select(-fit_coefs_class) %>%
      mutate(fit_coefs = map(fit_coefs, rownames_to_column, var = "featureNames")) %>%
      unnest()
```

Mothur

```
logFC_MgSeq_mothur_df <- titration_comp_mothur_df %>%
      mutate(fit = pmap(list(mrexp_sub, T1, T2), fit_safely))

logFC_MgSeq_mothur_coefs_df <- logFC_MgSeq_mothur_df %>%
      mutate(fit_coefs = map(fit, ~mrcoefs_safely(.$result, number = Inf)))

logFC_MgSeq_mothur_coefs_tidy <- logFC_MgSeq_mothur_coefs_df %>%
      select(biosample_id, T1, T2, fit_coefs) %>%
      ## filtering titration comparisons where the model failed
      mutate(fit_coefs = map(fit_coefs, ~.$result),
             fit_coefs_class = map_chr(fit_coefs, class)) %>%
      filter(fit_coefs_class == "data.frame") %>% select(-fit_coefs_class) %>%
      mutate(fit_coefs = map(fit_coefs, rownames_to_column, var = "featureNames")) %>%
      unnest()
```

QIIME

```
logFC_MgSeq_qiime_df <- titration_comp_qiime_df %>%
      mutate(fit = pmap(list(mrexp_sub, T1, T2), fit_safely))
```

```
logFC_MgSeq_qiime_coefs_df <- logFC_MgSeq_qiime_df %>%
      mutate(fit_coefs = map(fit, ~mrcoefs_safely(.$result, number = Inf)))

logFC_MgSeq_qiime_coefs_tidy <- logFC_MgSeq_qiime_coefs_df %>%
      select(biosample_id, T1, T2, fit_coefs) %>%
      ## excluding titration comparisons where the model failed
      mutate(fit_coefs = map(fit_coefs, ~.$result),
             fit_coefs_class = map_chr(fit_coefs, class)) %>%
      filter(fit_coefs_class == "data.frame") %>% select(-fit_coefs_class) %>%
      mutate(fit_coefs = map(fit_coefs, rownames_to_column, var = "featureNames")) %>%
      unnest()
```

Unclustered

```
logFC_MgSeq_unclustered_df <- titration_comp_unclustered_df %>%
      mutate(fit = pmap(list(mrexp_sub, T1, T2), fit_safely))

logFC_MgSeq_unclustered_coefs_df <- logFC_MgSeq_unclustered_df %>%
      mutate(fit_coefs = map(fit, ~mrcoefs_safely(.$result, number = Inf)))

logFC_MgSeq_unclustered_coefs_tidy <- logFC_MgSeq_unclustered_coefs_df %>%
      select(biosample_id, T1, T2, fit_coefs) %>%
      ## excluding titration comparisons where the model failed
      mutate(fit_coefs = map(fit_coefs, ~.$result),
             fit_coefs_class = map_chr(fit_coefs, class)) %>%
      filter(fit_coefs_class == "data.frame") %>% select(-fit_coefs_class) %>%
      mutate(fit_coefs = map(fit_coefs, rownames_to_column, var = "featureNames")) %>%
      unnest()
```

Cleanup fitfeatureModel

```
rm(logFC_MgSeq_dada2_df)
rm(logFC_MgSeq_dada2_coefs_df)
rm(logFC_MgSeq_mothur_df)
rm(logFC_MgSeq_mothur_coefs_df)
rm(logFC_MgSeq_qiime_df)
rm(logFC_MgSeq_qiime_coefs_df)
rm(logFC_MgSeq_unclustered_df)
rm(logFC_MgSeq_unclustered_coefs_df)
```

EdgeR

Function for running EdgeR

```
## Function based on phyloseq_to_edgeR code
## http://joey711.github.io/phyloseq-extensions/edgeR.html
mrexp_to_edgeR <- function(mrexp_obj, group, method = "RLE", ...){
      require(edgeR)
      ## Extracting count data - no scaling or transformation
      x <- mrexp_obj %>%
            metagenomeSeq::MRcounts(norm = FALSE, log = FALSE, sl = 1) %>%
            as.matrix()
      x <- x + 1 # add 1 to prevent log(0) issues
```

```
      ## Check `group` argument
      if(identical(all.equal(length(group), 1), TRUE) & ncol(mrexp_obj)> 1){
            ## Assumes grouop is a categorical sample variable name
            group <- pData(mrexp_obj) %>% .[,group]
            group <- as.numeric(as.character(group))
            T1 <- min(group)
            T2 <- max(group)
            group <- factor(group, levels = c(as.character(T1), as.character(T2)))
      }

      ## Use taxonomy information at gene annotations
      ## - Where OTUname is incorporated into the results
      taxonomy <- fData(mrexp_obj)
      if(!is.null(taxonomy)){
            taxonomy <- taxonomy %>% as.matrix() %>% data.frame()
      }

      ## Convert into a DGEList
      y <- DGEList(counts = x, group = group, genes = taxonomy,
                   remove.zeros = TRUE, ...)

      ## Calc normalization factors
      z <- edgeR::calcNormFactors(y, method = method)

      ## Check for division by zero inside `calcNormFactors`
      if( !all(is.finite(z$amples$norm.factors))){
            stop("Something wrong with edgeR::calcNormFactors on this data,
                 non-finite $norm.factors, consider changing `method` argument.")
      }

      ## Estimate dispersions
      z %>% estimateCommonDisp() %>% estimateTagwiseDisp()
}
```

DADA2

```
require(edgeR)
logFC_edgeR_dada2_df <- titration_comp_dada2_df %>%
      mutate(fit = map(mrexp_sub, mrexp_to_edgeR, group = "t_fctr"),
             fit = map(fit, exactTest))

logFC_edgeR_dada2_coefs_df <- logFC_edgeR_dada2_df %>%
      mutate(fit_coefs = map(fit, topTags, n = Inf, adjust.method = "BH"))

logFC_edgeR_dada2_coefs_tidy <- logFC_edgeR_dada2_coefs_df %>%
      select(biosample_id, T1, T2, fit_coefs) %>%
      mutate(fit_coefs = map(fit_coefs, ~.@.Data[[1]])) %>%
      unnest()
```

Mothur

```
logFC_edgeR_mothur_df <- titration_comp_mothur_df %>%
 mutate(fit = map(mrexp_sub, mrexp_to_edgeR, group = "t_fctr"),
```

```
             fit = map(fit, exactTest))

logFC_edgeR_mothur_coefs_df <- logFC_edgeR_mothur_df %>%
      mutate(fit_coefs = map(fit, topTags, n = Inf, adjust.method = "BH"))

logFC_edgeR_mothur_coefs_tidy <- logFC_edgeR_mothur_coefs_df %>%
      select(biosample_id, T1, T2, fit_coefs) %>%
      mutate(fit_coefs = map(fit_coefs, ~.@.Data[[1]])) %>%
      unnest()
```

QIIME

```
logFC_edgeR_qiime_df <- titration_comp_qiime_df %>%
      mutate(fit = map(mrexp_sub, mrexp_to_edgeR, group = "t_fctr"),
             fit = map(fit, exactTest))

logFC_edgeR_qiime_coefs_df <- logFC_edgeR_qiime_df %>%
      mutate(fit_coefs = map(fit, topTags, n = Inf, adjust.method = "BH"))

logFC_edgeR_qiime_coefs_tidy <- logFC_edgeR_qiime_coefs_df %>%
      select(biosample_id, T1, T2, fit_coefs) %>%
      mutate(fit_coefs = map(fit_coefs, ~.@.Data[[1]])) %>%
      unnest()
```

Unclustered

```
logFC_edgeR_unclustered_df <- titration_comp_unclustered_df %>%
      mutate(fit = map(mrexp_sub, mrexp_to_edgeR, group = "t_fctr"),
             fit = map(fit, exactTest))

logFC_edgeR_unclustered_coefs_df <- logFC_edgeR_unclustered_df %>%
      mutate(fit_coefs = map(fit, topTags, n = Inf, adjust.method = "BH"))

logFC_edgeR_unclustered_coefs_tidy <- logFC_edgeR_unclustered_coefs_df %>%
      select(biosample_id, T1, T2, fit_coefs) %>%
      mutate(fit_coefs = map(fit_coefs, ~.@.Data[[1]])) %>%
      unnest()
```

Save logFC tables

Combine tables for the different pipelines into individual data frames for each differential abundance methods and save as RDS files.

```
list(dada2 = logFC_MgSeq_dada2_coefs_tidy,
     mothur = logFC_MgSeq_mothur_coefs_tidy,
     qiime = logFC_MgSeq_qiime_coefs_tidy,
     unclustered = logFC_MgSeq_unclustered_coefs_tidy) %>%
      bind_rows(.id = "pipe") %>% saveRDS("~/Desktop/logFC_MgSeq_df.rds")

list(dada2 = logFC_edgeR_dada2_coefs_tidy,
     mothur = logFC_edgeR_mothur_coefs_tidy,
     qiime = logFC_edgeR_qiime_coefs_tidy,
```

```
    unclustered = logFC_edgeR_unclustered_coefs_tidy) %>%
     bind_rows(.id = "pipe") %>% saveRDS("~/Desktop/logFC_edgeR_df.rds")
```

Session information

Git repo commit information

```
library(git2r)
```

```
##
## Attaching package: 'git2r'
```

```
## The following object is masked from 'package:foreach':
##
##     when
```

```
## The following objects are masked from 'package:Biobase':
##
##     content, notes
```

```
## The following object is masked from 'package:dplyr':
##
##     pull
```

```
## The following objects are masked from 'package:purrr':
##
##     is_empty, when
```

```
repo <- repository(path = "../")
last_commit <- commits(repo)[[1]]
```

The current git commit of this file is 7e8ca07dde4fb89e2fb672ae206fd0156a75ff6a, which is on the master branch and was made by nate-d-olson on 2017-09-25 16:44:19. The current commit message is revised intro and methods. The repository is online at https://github.com/nate-d-olson/mgtst-pub

Platform Information

```
s_info <- devtools::session_info()
print(s_info$platform)
```

```
##  setting  value
##  version  R version 3.4.1 (2017-06-30)
##  system   x86_64, darwin16.7.0
##  ui       unknown
##  language (EN)
##  collate  en_US.UTF-8
##  tz       America/New_York
##  date     2017-09-28
```

Package Versions

```
s_info$packages %>% filter(`*` == "*") %>% select(-`*`) %>%
      knitr::kable()
```

package	version	date	source
base	3.4.1	2017-08-18	local
bindrcpp	0.2	2017-06-17	CRAN (R 3.4.0)
Biobase	2.36.2	2017-06-21	Bioconductor
BiocGenerics	0.22.0	2017-05-04	Bioconductor
datasets	3.4.1	2017-08-18	local
dplyr	0.7.2	2017-07-20	CRAN (R 3.4.1)
edgeR	3.18.1	2017-06-21	Bioconductor
forcats	0.2.0	2017-01-23	CRAN (R 3.4.0)
foreach	1.4.3	2015-10-13	CRAN (R 3.4.0)
ggplot2	2.2.1	2016-12-30	CRAN (R 3.4.0)
git2r	0.19.0	2017-07-19	CRAN (R 3.4.1)
glmnet	2.0-10	2017-05-06	CRAN (R 3.4.0)
graphics	3.4.1	2017-08-18	local
grDevices	3.4.1	2017-08-18	local
knitr	1.17	2017-08-10	CRAN (R 3.4.1)
limma	3.32.3	2017-07-19	Bioconductor
Matrix	1.2-10	2017-05-03	CRAN (R 3.4.1)
metagenomeSeq	1.18.0	2017-05-04	Bioconductor
methods	3.4.1	2017-08-18	local
modelr	0.1.1	2017-07-24	CRAN (R 3.4.1)
parallel	3.4.1	2017-08-18	local
ProjectTemplate	0.8	2017-08-09	CRAN (R 3.4.1)
purrr	0.2.3	2017-08-02	CRAN (R 3.4.1)
RColorBrewer	1.1-2	2014-12-07	CRAN (R 3.4.0)
readr	1.1.1	2017-05-16	CRAN (R 3.4.0)
readxl	1.0.0	2017-04-18	CRAN (R 3.4.0)
stats	3.4.1	2017-08-18	local
stringr	1.2.0	2017-02-18	CRAN (R 3.4.0)
tibble	1.3.3	2017-05-28	CRAN (R 3.4.0)
tidyr	0.6.3	2017-05-15	CRAN (R 3.4.0)
tidyverse	1.1.1	2017-01-27	CRAN (R 3.4.0)
utils	3.4.1	2017-08-18	local