wk5_coding_answers

Christine Hawkes

2/9/2022

Contents

Coding Exercises **1**

Load libraries 1

If you aren't continuing from class, re-load and subset data 1

1. Use phyloseq to zoom in on richness of specific phyla in the data: 2

2. Use phyloseq to examine genus-level richness: 3

3. Phyloseq acts as a wrapper for vegan for many of its community metrics Use vegan::diversity to calculate untrimmed data Shannon's H, Simpson's D: 6

4. Use vegan::radfit to determine the best model fit for rank-abundance curves (lowest AIC value) and plot 8

Session Info 11

Coding Exercises

Load libraries

```
library(phyloseq)
library(tidyverse)
library(vegan)
```

If you aren't continuing from class, re-load and subset data

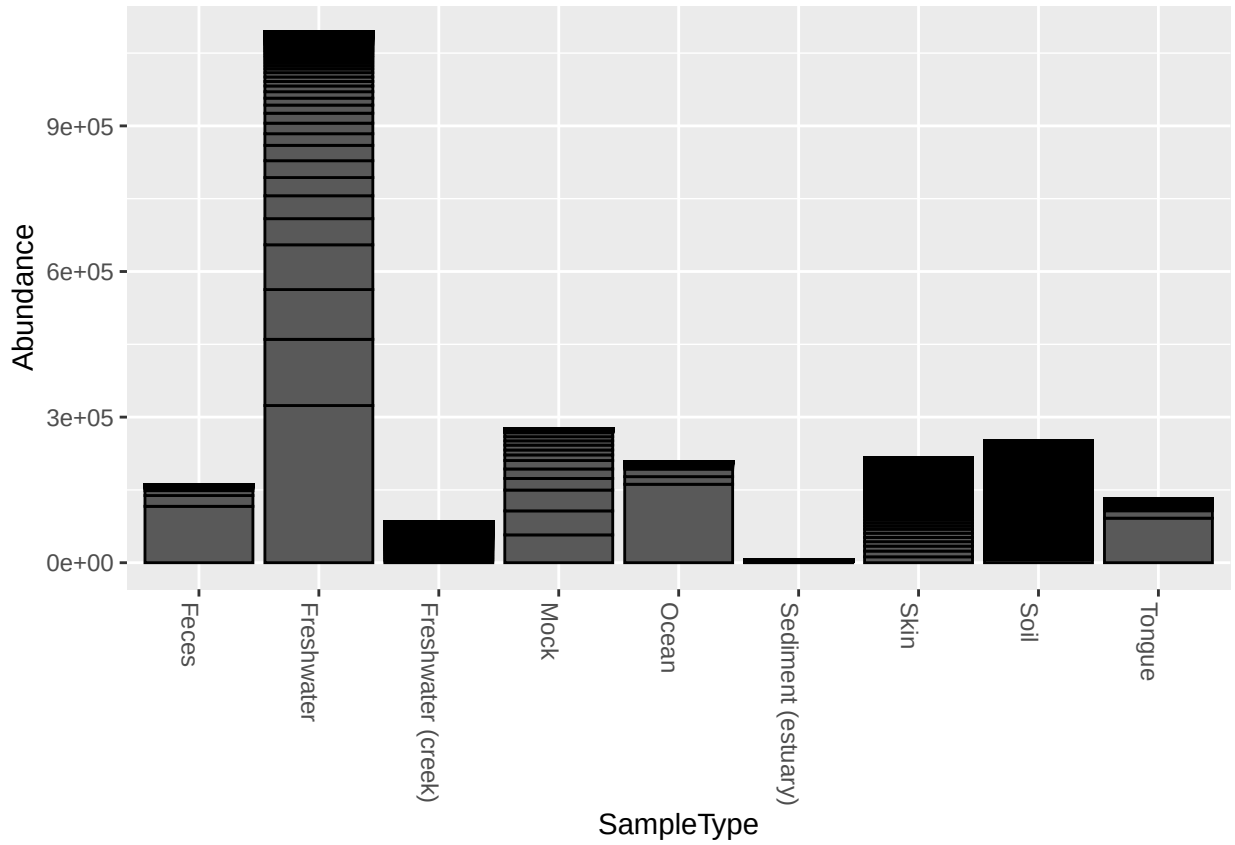
```
data("GlobalPatterns")
ps_gp <- GlobalPatterns
ps_gp_bact <-phyloseq::subset_samples(ps_gp, SampleType != "Mock")
ps_gp_bact <- phyloseq::subset_taxa(ps_gp, Kingdom=="Bacteria")
ps_gp_bact <- phyloseq::prune_taxa(taxa_sums(ps_gp_bact) > 1, ps_gp_bact)
ps_gp_bact <- phyloseq::prune_samples(sample_sums(ps_gp_bact)>0, ps_gp_bact)
```

1. Use phyloseq to zoom in on richness of specific phyla in the data:

```
phyloseq::get_taxa_unique(ps_gp_bact, "Phylum")
```

```
##  [1] "Actinobacteria"   "Spirochaetes"     "MVP-15"           "Proteobacteria"
##  [5] "SBR1093"          "Fusobacteria"     "Tenericutes"      "Cyanobacteria"
##  [9] "GOUTA4"           "TG3"              "Chlorobi"         "Bacteroidetes"
## [13] "Caldithrix"       "KSB1"             "SAR406"           "LCP-89"
## [17] "Thermi"           "Gemmatimonadetes" "Fibrobacteres"    "GN06"
## [21] "AC1"              "TM6"              "OP8"              "Elusimicrobia"
## [25] "NC10"             "SPAM"             NA                 "Acidobacteria"
## [29] "CCM11b"           "Nitrospirae"      "NKB19"            "BRC1"
## [33] "Hyd24-12"         "WS3"              "PAUC34f"          "GN04"
## [37] "GN12"             "Verrucomicrobia"  "Lentisphaerae"    "LD1"
## [41] "Chlamydiae"       "OP3"              "Planctomycetes"   "OP9"
## [45] "WPS-2"            "Armatimonadetes"  "SC3"              "TM7"
## [49] "GN02"             "SM2F11"           "ABY1_OD1"         "ZB2"
## [53] "OP11"             "Chloroflexi"      "SC4"              "WS1"
## [57] "GAL15"            "AD3"              "WS2"              "Caldiserica"
## [61] "Firmicutes"       "Thermotogae"      "Synergistetes"    "SR1"
```

```
ps_gp_actino <- phyloseq::subset_taxa(ps_gp_bact, Phylum=="Actinobacteria")
plot_bar(ps_gp_actino, "SampleType", "Abundance")
```

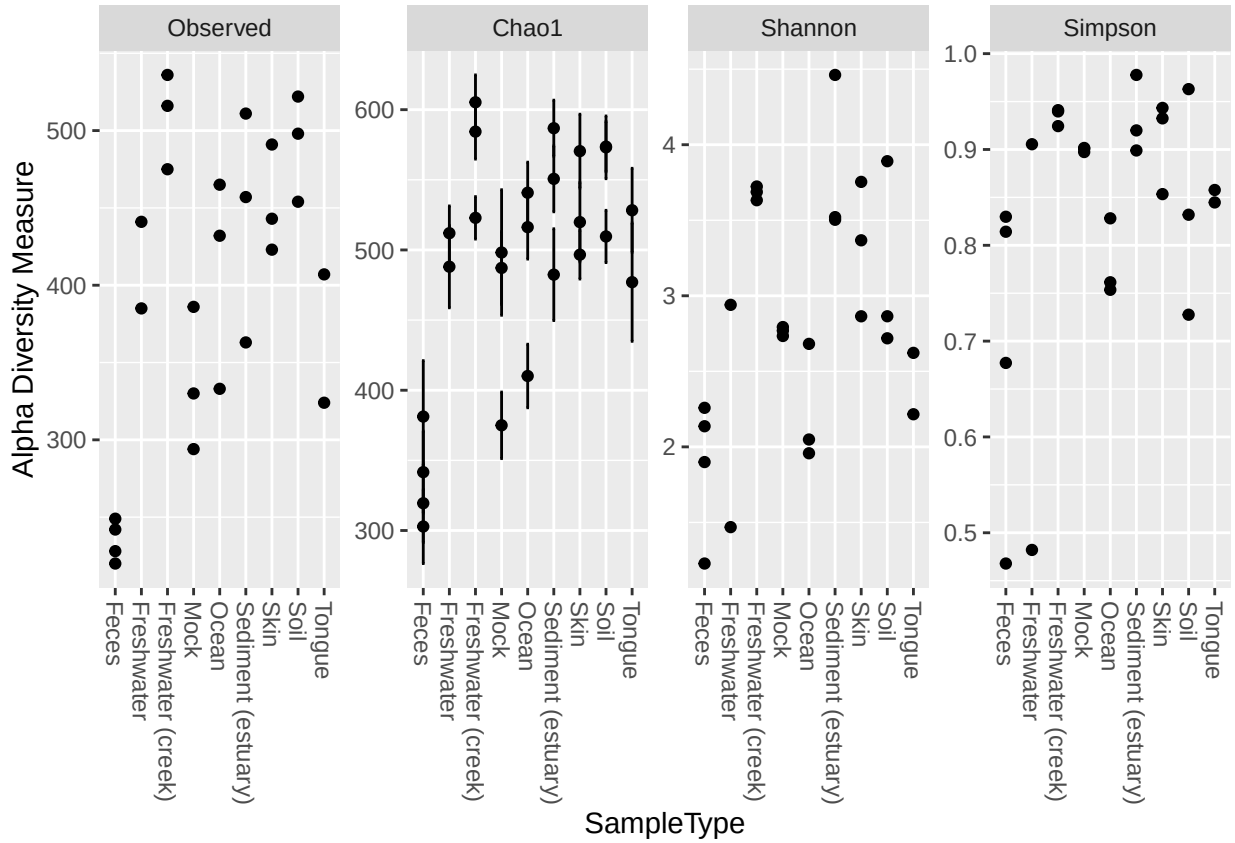


2. Use phyloseq to examine genus-level richness:

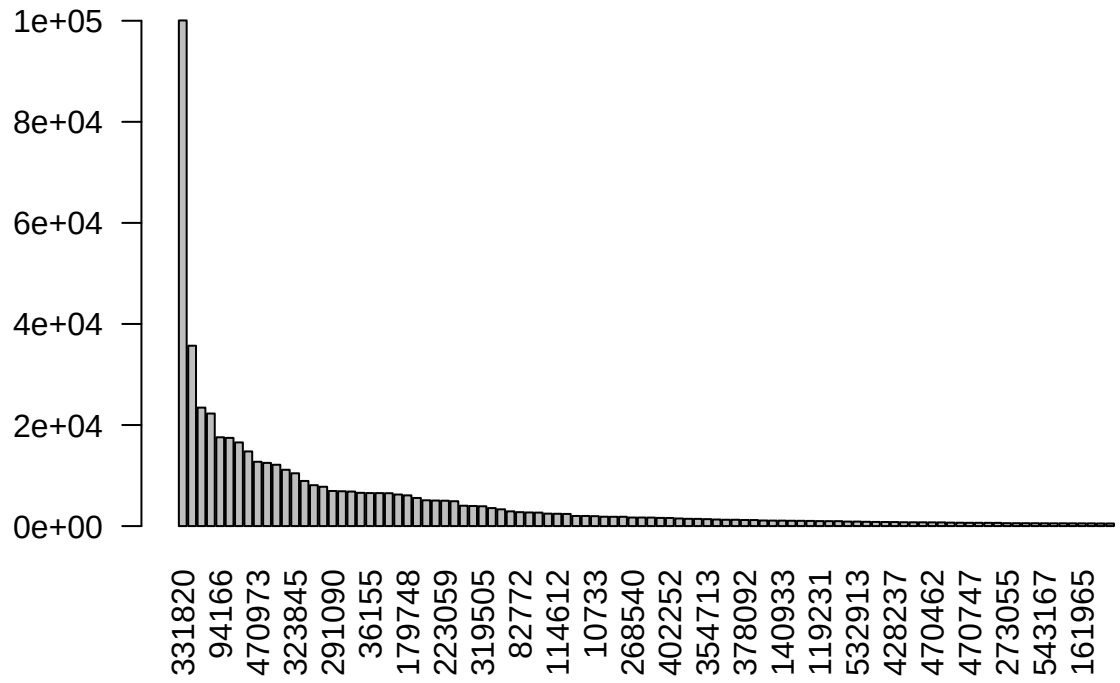
```
ps_gp_genus <- phyloseq::tax_glom(ps_gp_bact, "Genus")
gp_alpha_genus <- phyloseq::estimate_richness(ps_gp_genus, split=TRUE, measures=c("Observed", "Chao1",
gp_alpha_genus #can be exported as .csv file for later use
```

```
##          Observed    Chao1 se.chao1  Shannon   Simpson
## CL3           498 573.1463 22.34065 2.864509 0.8319046
## CC1           522 573.6562 17.87999 2.718544 0.7275739
## SV1           454 509.6176 18.62158 3.890766 0.9629980
## M31Fcsw       220 302.8333 26.38120 1.899463 0.6772423
## M11Fcsw       228 319.4375 28.00997 1.227820 0.4678816
## M31Plmr       443 496.6250 17.34019 2.864311 0.8534122
## M11Plmr       491 570.4444 26.32357 3.754009 0.9433673
## F21Plmr       423 519.8333 28.31852 3.367316 0.9324953
## M31Tong       407 528.2727 29.92244 2.215924 0.8447398
## M11Tong       324 477.0303 42.16450 2.622263 0.8577195
## LMEpi24M      441 511.9492 19.39292 1.468766 0.4820889
## SLEpi20M      385 488.0526 29.28946 2.940187 0.9054914
## AQC1cm        536 605.2778 19.48688 3.723236 0.9243866
## AQC4cm        516 584.3529 19.57881 3.687320 0.9410144
## AQC7cm        475 522.8776 15.11768 3.632608 0.9397151
## NP2           333 410.1429 22.64958 1.957668 0.7614112
## NP3           465 540.7826 21.78728 2.681671 0.8281457
## NP5           432 516.2453 22.75164 2.048734 0.7535721
## TRRsed1       363 482.3846 32.70026 4.461892 0.9778045
## TRRsed2       511 586.7969 19.96644 3.503201 0.8989789
## TRRsed3       457 550.6719 23.46538 3.520589 0.9199038
## TS28          242 341.6000 29.20589 2.258391 0.8297806
## TS29          249 381.2222 39.85195 2.136559 0.8141659
## Even1         386 487.1818 25.96038 2.793140 0.9008417
## Even2         330 498.1714 44.72739 2.733895 0.8973283
## Even3         294 375.0000 23.68776 2.769324 0.9015793
```

```
plot_richness(ps_gp_genus, x="SampleType", measures=c("Observed", "Chao1", "Shannon", "Simpson"))
```




```
# observed alpha diversity of genera is << that of ASVs
# sample types are more similar at genus level
N <- 100
barplot(sort(taxa_sums(ps_gp_genus), TRUE) [1:N]/nsamples(ps_gp_genus), las=2)
```




```
# rank-abundance similar to ASV level for top 100
# suggests dominance at ASV level is maintained at genus level
```

3. Phyloseq acts as a wrapper for vegan for many of its community metrics Use vegan::diversity to calculate untrimmed data Shannon's H, Simpson's D:

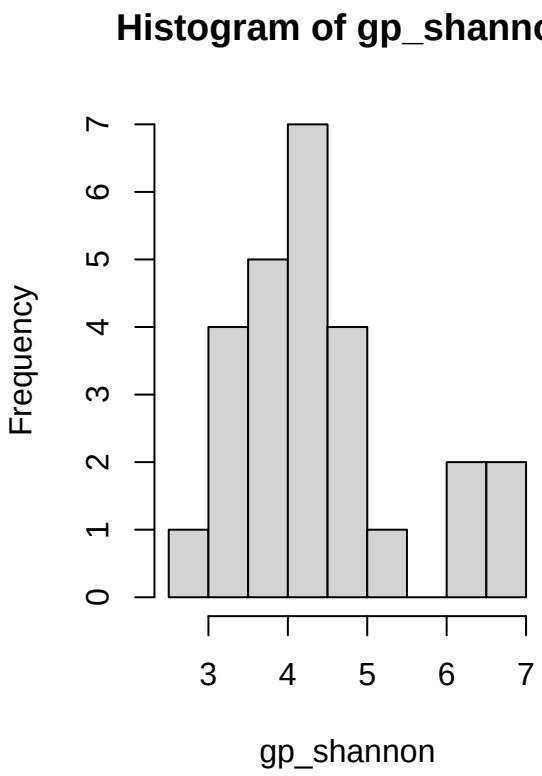
```
votu_all <- otu_table(GlobalPatterns)
votu_all <- as.matrix(t(votu_all))
gp_shannon <- vegan::diversity(votu_all, index="shannon")
gp_shannon
```

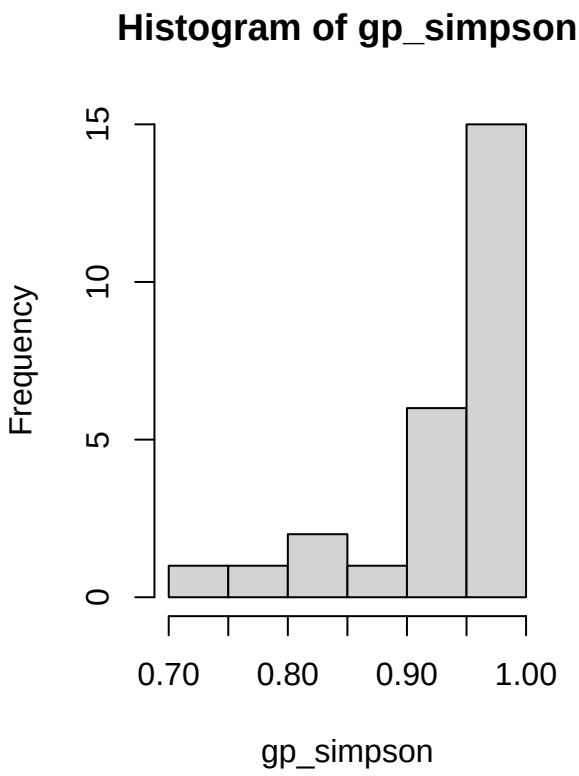
```
##      CL3      CC1      SV1  M31Fcsw  M11Fcsw  M31Plmr  M11Plmr  F21Plmr
## 6.576517 6.776603 6.498494 3.828368 3.287666 4.289269 4.849999 4.874747
##  M31Tong  M11Tong LMEpi24M SLEpi20M   AQC1cm   AQC4cm   AQC7cm      NP2
## 2.672103 3.905419 3.093981 3.651142 3.552736 3.372495 4.027716 4.230515
##      NP3      NP5  TRRsed1  TRRsed2  TRRsed3     TS28     TS29    Even1
## 4.483806 4.563943 6.157462 4.869817 5.461840 4.126538 3.452772 4.083665
##    Even2    Even3
## 3.956909 4.006375
```

```
gp_simpson <- vegan::diversity(votu_all, index="simpson")
gp_simpson
```

```
##       CL3       CC1       SV1   M31Fcsw   M11Fcsw   M31Plmr   M11Plmr   F21Plmr
## 0.9946561 0.9952117 0.9962900 0.9275989 0.9097382 0.9379114 0.9518733 0.9777509
##   M31Tong   M11Tong  LMEpi24M  SLEpi20M    AQC1cm    AQC4cm    AQC7cm       NP2
## 0.8625384 0.9358927 0.8023279 0.9072187 0.7648870 0.7397659 0.8179374 0.9532320
##       NP3       NP5   TRRsed1   TRRsed2   TRRsed3      TS28      TS29     Even1
## 0.9718016 0.9748733 0.9924388 0.9640962 0.9815843 0.9651752 0.9180976 0.9681981
##     Even2     Even3
## 0.9639157 0.9673405
```

```
#run the next 3 code lines together
par(mfrow = c(1,2))
hist(gp_shannon)
hist(gp_simpson)
```


Histogram of gp_shannon
Frequency
0
1
2
3
4
5
6
7
3
4
5
6
7
gp_shannon


Histogram of gp_simpson
Frequency
0
5
10
15
0.70
0.80
0.90
1.00
gp_simpson

4. Use vegan::radfit to determine the best model fit for rank-abundance curves (lowest AIC value) and plot

```
votu <- otu_table(ps_gp_bact)
colnames(votu) #need to transpose for vegan
```

```
##  [1] "CL3"      "CC1"      "SV1"      "M31Fcsw"  "M11Fcsw"  "M31Plmr"
##  [7] "M11Plmr"  "F21Plmr"  "M31Tong"  "M11Tong"  "LMEpi24M" "SLEpi20M"
## [13] "AQC1cm"   "AQC4cm"   "AQC7cm"   "NP2"      "NP3"      "NP5"
## [19] "TRRsed1"  "TRRsed2"  "TRRsed3"  "TS28"     "TS29"     "Even1"
## [25] "Even2"    "Even3"
```

```
votu <- t(votu)
row.names(votu) #confirm samples are rows
```

```
##  [1] "CL3"      "CC1"      "SV1"      "M31Fcsw"  "M11Fcsw"  "M31Plmr"
##  [7] "M11Plmr"  "F21Plmr"  "M31Tong"  "M11Tong"  "LMEpi24M" "SLEpi20M"
## [13] "AQC1cm"   "AQC4cm"   "AQC7cm"   "NP2"      "NP3"      "NP5"
## [19] "TRRsed1"  "TRRsed2"  "TRRsed3"  "TS28"     "TS29"     "Even1"
## [25] "Even2"    "Even3"
```

```
votu.df <- as.data.frame(votu)
rf <- radfit(votu)
```

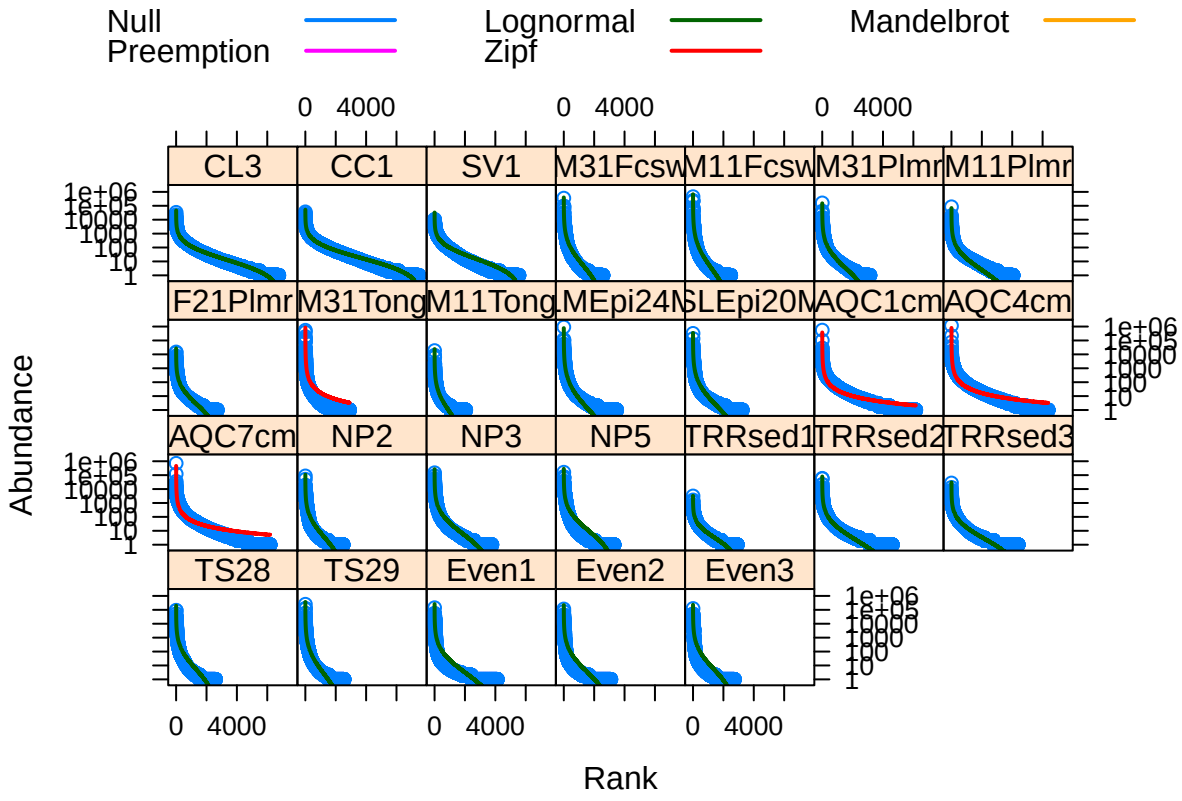
```
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
```

```
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
## Error in glm.fit(x = structure(c(1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,  :
##   NA/NaN/Inf in 'x'
```

```
# rf_single <- radfit(votu[1,]) # alt is to run for one sample
rf
```

```
##
## Deviance for RAD models:
##
##                   CL3       CC1       SV1    M31Fcsw    M11Fcsw   M31Plmr
## Null        1577477.0 1922595.4 1071432.4  7065261.9 11267780.1 2987598.4
## Preemption   694514.5  892280.3  287086.7  1816872.9  2493790.3 1141267.6
## Lognormal     13099.7   30184.5   67943.2   112129.7   323373.7   48327.8
## Zipf         256889.1  338901.7  334945.9   335060.7   476151.2  154957.8
## Mandelbrot
##               M11Plmr   F21Plmr   M31Tong    M11Tong   LMEpi24M  SLEpi20M
## Null        1603166.0  561111.2 13531474.6  413668.4 13418436.2 6396253.5
## Preemption   922233.1  233151.5  3481255.9  207652.2  4827996.8 1968974.9
## Lognormal     23109.7   22200.5   469978.8   14145.2   173485.5   92793.5
## Zipf          47236.9   55981.3   454618.1   15137.3   319672.1  230522.6
## Mandelbrot    38247.1                                  319672.1
##                AQC1cm    AQC4cm    AQC7cm        NP2        NP3       NP5
## Null        7722316.2 16437604.7 9838266.3  2029557.0  5931903.4 6210832.8
## Preemption  5254659.6 10716324.2 6349904.6   533170.8  1461751.2 1022087.3
## Lognormal    431023.9   932516.7  583652.4    60112.8   236412.8  326797.7
## Zipf         288890.0   706747.9  491763.8   163433.3   565466.7  779271.0
## Mandelbrot   288890.0   706747.9  491763.8
##               TRRsed1   TRRsed2   TRRsed3       TS28       TS29     Even1
## Null          76563.9 1893435.4  760335.5  3864449.3  6288350.7 6050776.2
## Preemption    43869.9  937966.0  368474.7   735374.8  1654116.2 1257725.5
## Lognormal      1780.6   26665.2    6243.0   250081.9   155127.3  587895.1
## Zipf          11702.7   68015.9   49962.7   458773.3   247587.7  900123.8
## Mandelbrot
##                 Even2   Even3
## Null        4563999.6 4768262
## Preemption   743501.6  591936
## Lognormal    398691.3  497836
## Zipf         630325.7  792608
## Mandelbrot
```

```
plot(rf, xlab="Rank",  ylab="Abundance", log="y")
```




```
# best fit is lognormal in the majority of samples
# AQC samples (all from one creek) are best fit by Zipf (power law)
# as is one human tongue sample
# you may get an error about glm fit NA/NaN/Inf for some samples
# this is due to lack of fit for the Mandelbrot distribution
```

Session Info

```
sessionInfo()
```

```
## R version 4.1.2 (2021-11-01)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 19042)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] vegan_2.5-7     lattice_0.20-45 permute_0.9-7   forcats_0.5.1
##  [5] stringr_1.4.0   dplyr_1.0.7     purrr_0.3.4     readr_2.1.1
##  [9] tidyr_1.1.4     tibble_3.1.6    ggplot2_3.3.5   tidyverse_1.3.1
## [13] phyloseq_1.38.0
##
## loaded via a namespace (and not attached):
##  [1] nlme_3.1-155           bitops_1.0-7           fs_1.5.2
##  [4] lubridate_1.8.0        httr_1.4.2             GenomeInfoDb_1.30.0
##  [7] tools_4.1.2            backports_1.4.1        utf8_1.2.2
## [10] R6_2.5.1               DBI_1.1.2              BiocGenerics_0.40.0
## [13] mgcv_1.8-38            colorspace_2.0-2       rhdf5filters_1.6.0
## [16] ade4_1.7-18            withr_2.4.3            tidyselect_1.1.1
## [19] compiler_4.1.2         cli_3.1.1              rvest_1.0.2
## [22] Biobase_2.54.0         xml2_1.3.3             labeling_0.4.2
## [25] scales_1.1.1           digest_0.6.29          rmarkdown_2.11
## [28] XVector_0.34.0         pkgconfig_2.0.3        htmltools_0.5.2
## [31] highr_0.9              dbplyr_2.1.1           fastmap_1.1.0
## [34] rlang_0.4.12           readxl_1.3.1           rstudioapi_0.13
## [37] farver_2.1.0           generics_0.1.2         jsonlite_1.7.3
## [40] RCurl_1.98-1.5         magrittr_2.0.1         GenomeInfoDbData_1.2.7
## [43] biomformat_1.22.0      Matrix_1.4-0           Rcpp_1.0.8
## [46] munsell_0.5.0          S4Vectors_0.32.3       Rhdf5lib_1.16.0
## [49] fansi_0.5.0            ape_5.6-1              lifecycle_1.0.1
## [52] stringi_1.7.6          yaml_2.2.1             MASS_7.3-54
## [55] zlibbioc_1.40.0        rhdf5_2.38.0           plyr_1.8.6
## [58] grid_4.1.2             parallel_4.1.2         crayon_1.4.2
## [61] Biostrings_2.62.0      haven_2.4.3            splines_4.1.2
## [64] multtest_2.50.0        hms_1.1.1              knitr_1.37
## [67] pillar_1.7.0           igraph_1.2.11          reshape2_1.4.4
## [70] codetools_0.2-18       stats4_4.1.2           reprex_2.0.1
## [73] glue_1.6.0             evaluate_0.14          data.table_1.14.2
## [76] modelr_0.1.8           vctrs_0.3.8            tzdb_0.2.0
```

```
## [79] foreach_1.5.1          cellranger_1.1.0       gtable_0.3.0
## [82] assertthat_0.2.1       xfun_0.29              broom_0.7.11
## [85] survival_3.2-13        iterators_1.0.13       IRanges_2.28.0
## [88] cluster_2.1.2          ellipsis_0.3.2
```
