Processing short nanopore reads with dada2

Marko Suokas

The document utilizes previously computed objects saved as RDS files. These objects are reloaded as needed. The original code can be executed by setting eval = TRUE or by running the chunk directly in RStudio

#### **Preprocess Ion Torrent Adapter Reads**

1. **Trim forward reads using Adapter A and the reverse complement of the trP1 sequence:**

* cutadapt --trimmed-only -e 0.05 -o ev\_forward.fastq.gz \  
  -g "CCATCTCATCCCTGCGTGTCTCCGACTCAG;o=30...ATCACCGACTGCCCATAGAGAGG;o=23" \  
  ev\_reads\_hq.fastq.gz

1. **Trim reverse reads using trP1 and the reverse complement of Adapter A:**

* cutadapt --trimmed-only -e 0.05 -o ev\_reverse.fastq.gz \  
  -g "CCTCTCTATGGGCAGTCGGTGAT;o=23...CTGAGTCGGAGACACGCAGGGATGAGATGG;o=30" \  
  ev\_reads\_hq.fastq.gz

1. **Generate reverse complements of the trimmed reverse reads:**

* seqkit seq -rp -t DNA -o ev\_rcomp.fasta.gz ev\_reverse.fastq.gz

1. **Merge forward reads with reverse-complemented reverse reads:**

* cat ev\_forward.fastq.gz ev\_rcomp.fastq.gz > raw\_005.fastq.gz

#### Import Data into QIIME2

1. **Import the merged reads into QIIME2:**

* qiime tools import --type 'MultiplexedSingleEndBarcodeInSequence' \  
  --input-path raw\_005.fastq.gz --output-path raw\_005.qza

#### Demultiplexing

1. **Demultiplex the reads:**

* qiime cutadapt demux-single --i-seqs raw\_005.qza \  
  --m-barcodes-file jt\_meta.tsv --m-barcodes-column Barcode\_seq \  
  --output-dir demuxed --p-error-rate 0 --p-anchor-barcode

#### Trim PCR Primers

1. **Trim the PCR primers (519F and 926R):**
   * **Trim forward primers (519F):**

* qiime cutadapt trim-single --p-front ACAGCMGCCGCGGTAATWC --p-overlap 15 \  
  --i-demultiplexed-sequences demuxed/per\_sample\_sequences.qza \  
   --p-discard-untrimmed --o-trimmed-sequences trim1.qza
  + **Trim reverse primers (926R):**
* qiime cutadapt trim-single --i-demultiplexed-sequences trim1.qza \  
  --p-adapter AAACTCAAAKGAATTGACGG \  
  --o-trimmed-sequences trimmed-sequences.qza

#### Decompress QIIME Artifact

1. **Extract the sequences from the QIIME2 artifact:**

* unzip trimmed-sequences.qza

**Notes**

* **Error allowances**: The parameters allow one sequencing error in the adapters, no errors in barcode sequences, and 1-2 errors in the PCR primers.
* **Command formatting**: Some options use double dashes (--), but they may not render correctly in some text formats.

#### Load libraries

library(dada2)  
library(mia)  
library(vegan)  
library(scater)  
library(Biostrings)  
library(tidyverse)  
library(knitr)  
library(kableExtra)  
library(ggthemes)  
library(ggsci)  
library(patchwork)

#### Set parameters

# Path variables  
path <- "data/reads/"  
silva <- "~/feature\_classifiers/silva\_nr99\_v138.1\_train\_set.fa.gz"  
species <- "~/feature\_classifiers/silva\_species\_assignment\_v138.1.fa.gz"  
training <- "~/feature\_classifiers/SILVA\_SSU\_r138\_2019.RData"  
meta\_file <- "data/jt\_meta.tsv"  
exportloc <- "results/"  
# Variables: truncation length, phix (Illumina)  
truncation <- 350  
#Creates results directory  
dir.create(exportloc)  
#metadata  
metadata <- data.frame(read\_tsv(meta\_file, show\_col\_types = F))  
#set knitr cache path  
knitr::opts\_chunk$set(cache.path = "cache/")

#### Import reads

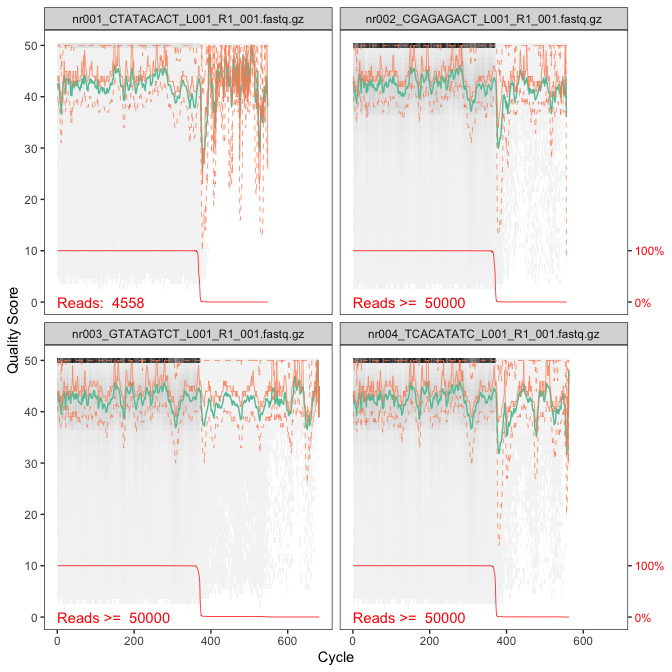
Sample nr072 was removed from dataset (0 reads)

# Forward fastq filenames have format: SAMPLENAME\_R1\_001.fastq  
fnFs <- sort(list.files(path, pattern="L001\_R1\_001.fastq.gz", full.names = T))  
# Extract sample names, assuming filenames have format: SAMPLENAME\_XXX.fastq.gz  
sample.names <- sapply(strsplit(basename(fnFs), "\_"), `[`, 1)

Check read quality of first samples

# Base quality plot  
p <- plotQualityProfile(fnFs[1:4], n = 50000)  
p

# Load base quality plot from saved object  
p <- readRDS("rds/qplot.rds")  
p



#### Filter and trim reads

# Filtered files are placed in filtered subdirectory  
filtFs <- file.path(path, "filtered", paste0(sample.names,  
 "\_F\_filt.fastq.gz"))  
# For single end data sets, maxEE default values  
names(filtFs) <- sample.names  
out <- filterAndTrim(fnFs, filtFs, truncLen=truncation,  
 maxN = 0, maxEE = 2 , truncQ = 2,  
 compress = T, multithread = F,  
 rm.phix = F)  
saveRDS(out, file = "rds/out.rds")

# Filtered files are placed in filtered subdirectory  
filtFs <- file.path(path, "filtered", paste0(sample.names,  
 "\_F\_filt.fastq.gz"))  
# For single end data sets without phix control  
names(filtFs) <- sample.names  
# Load previously saved object  
out <-readRDS("rds/out.rds")

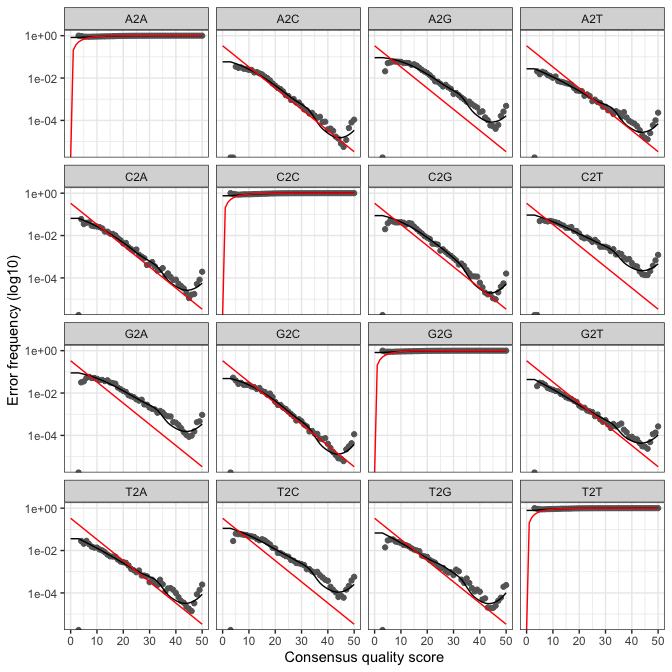
#### Learn and plot error profile

# Forward read error rate  
errF <- learnErrors(filtFs, multithread = T)  
saveRDS(errF, file = "rds/errF.rds")

# Load previously saved object  
errF <- readRDS("rds/errF.rds")

Plot error profile

# Plotting error rate profile for forward reads  
plotErrors(errF, nominalQ = T)



#### Denoise sequences

dadaFs <- dada(filtFs, err = errF, multithread = T, verbose = F)  
saveRDS(dadaFs, file = "rds/dadaFs.rds")

# Load previously saved object  
dadaFs <- readRDS("rds/dadaFs.rds")

#### Build ASV table

seqtab <- makeSequenceTable(dadaFs)  
# Dimensions of ASV table  
dim(seqtab)

[1] 131 8404

Remove chimeric variants

seqtab.nochim <- removeBimeraDenovo(seqtab, method = "consensus",  
 multithread = T)  
dim(seqtab.nochim)

[1] 131 7634

Amount of data remaining after chimera removal

sum(seqtab.nochim)/sum(seqtab)

[1] 0.9959133

#### Summary table

#If processing a single sample, remove the sapply calls  
getN <- function(x) sum(getUniques(x))  
track <- cbind(out, sapply(dadaFs, getN), rowSums(seqtab.nochim),  
 rowSums(seqtab.nochim != 0))  
colnames(track) <- c("Input", "Filtered", "DenoisedF", "Nonchimeric",  
 "N:o of variants")  
rownames(track) <- sample.names  
#table  
kable(track, caption="Summary table", booktabs = T, longtable = T) %>%  
 kable\_styling(latex\_options=c("striped", "HOLD\_position", "repeat\_header"),  
 font\_size = 11) %>%  
 row\_spec(0,background = "teal", color = "ivory")

Summary table

|  | Input | Filtered | DenoisedF | Nonchimeric | N:o of variants |
| --- | --- | --- | --- | --- | --- |
| nr001 | 4558 | 3737 | 3589 | 3589 | 96 |
| nr002 | 101469 | 82272 | 82017 | 82017 | 449 |
| nr003 | 139054 | 113872 | 113654 | 113612 | 359 |
| nr004 | 256204 | 207763 | 207374 | 207160 | 361 |
| nr005 | 104757 | 85575 | 85340 | 85153 | 413 |
| nr006 | 125276 | 101867 | 101438 | 101135 | 407 |
| nr007 | 16776 | 13466 | 13352 | 13352 | 179 |
| nr008 | 60779 | 50372 | 50261 | 50261 | 259 |
| nr009 | 57496 | 46090 | 45844 | 45777 | 250 |
| nr010 | 58884 | 48417 | 48302 | 48287 | 239 |
| nr011 | 57804 | 46828 | 46648 | 46648 | 315 |
| nr012 | 63184 | 51624 | 51355 | 51279 | 291 |
| nr013 | 184184 | 151003 | 150506 | 150337 | 305 |
| nr014 | 37034 | 30255 | 30059 | 30059 | 167 |
| nr015 | 52921 | 42498 | 42295 | 42214 | 256 |
| nr016 | 57780 | 47377 | 47221 | 47198 | 184 |
| nr017 | 32249 | 27015 | 26761 | 26761 | 223 |
| nr018 | 32343 | 26457 | 26235 | 26204 | 238 |
| nr019 | 211 | 182 | 152 | 152 | 8 |
| nr020 | 17466 | 14548 | 14312 | 14309 | 271 |
| nr021 | 13454 | 10843 | 10407 | 10368 | 298 |
| nr022 | 36958 | 30203 | 29950 | 29916 | 330 |
| nr023 | 5863 | 4939 | 4760 | 4760 | 169 |
| nr024 | 39062 | 32944 | 32682 | 32675 | 287 |
| nr025 | 18358 | 15039 | 14839 | 14839 | 234 |
| nr026 | 54181 | 45020 | 44705 | 44689 | 461 |
| nr027 | 57462 | 46962 | 46734 | 46734 | 281 |
| nr028 | 6385 | 5190 | 4976 | 4976 | 135 |
| nr029 | 1149 | 915 | 809 | 809 | 57 |
| nr030 | 2978 | 2463 | 2280 | 2280 | 72 |
| nr031 | 55110 | 44610 | 44399 | 44191 | 303 |
| nr032 | 53814 | 42979 | 42625 | 42284 | 219 |
| nr033 | 41001 | 32923 | 32685 | 32664 | 287 |
| nr034 | 25124 | 20580 | 20298 | 20296 | 288 |
| nr035 | 1028 | 835 | 761 | 761 | 34 |
| nr036 | 55738 | 45478 | 45334 | 45302 | 332 |
| nr037 | 52126 | 42022 | 41630 | 41550 | 325 |
| nr038 | 63365 | 52054 | 51762 | 51391 | 404 |
| nr039 | 62495 | 50482 | 50215 | 50090 | 428 |
| nr040 | 66486 | 54623 | 54394 | 54325 | 364 |
| nr041 | 50049 | 41199 | 41048 | 41028 | 288 |
| nr042 | 56406 | 47601 | 47412 | 47302 | 201 |
| nr043 | 62973 | 52026 | 51849 | 51794 | 248 |
| nr044 | 51860 | 42460 | 42297 | 42212 | 259 |
| nr045 | 62931 | 51359 | 50996 | 50953 | 410 |
| nr046 | 57953 | 47598 | 47331 | 46941 | 259 |
| nr047 | 59138 | 49654 | 49488 | 49442 | 260 |
| nr048 | 55998 | 46407 | 46245 | 46231 | 284 |
| nr049 | 59015 | 49029 | 48942 | 48940 | 252 |
| nr050 | 58770 | 48128 | 47929 | 47881 | 367 |
| nr051 | 58199 | 48129 | 48011 | 47938 | 241 |
| nr052 | 59722 | 48648 | 48421 | 48387 | 456 |
| nr053 | 52295 | 42909 | 42812 | 42810 | 206 |
| nr054 | 54222 | 44000 | 43814 | 43761 | 255 |
| nr055 | 44896 | 36482 | 36250 | 36211 | 327 |
| nr056 | 56564 | 46213 | 46048 | 46037 | 340 |
| nr057 | 57220 | 44611 | 44384 | 44384 | 340 |
| nr058 | 29071 | 21680 | 21628 | 21628 | 101 |
| nr059 | 49673 | 37678 | 37585 | 37585 | 184 |
| nr060 | 56441 | 44838 | 44765 | 44720 | 204 |
| nr061 | 39464 | 31746 | 31592 | 31558 | 203 |
| nr062 | 57638 | 47417 | 47311 | 47311 | 121 |
| nr063 | 41622 | 34042 | 33991 | 33847 | 116 |
| nr064 | 57665 | 45751 | 45672 | 45672 | 151 |
| nr065 | 31057 | 24009 | 23923 | 23923 | 115 |
| nr066 | 46974 | 36323 | 36192 | 36192 | 160 |
| nr067 | 28068 | 23098 | 22969 | 22967 | 116 |
| nr068 | 29823 | 24238 | 24203 | 24203 | 86 |
| nr069 | 30168 | 24183 | 24129 | 24127 | 128 |
| nr070 | 60493 | 50010 | 49839 | 49598 | 200 |
| nr071 | 53242 | 41175 | 41058 | 40969 | 179 |
| nr073 | 49981 | 40478 | 40360 | 40357 | 179 |
| nr074 | 57395 | 46109 | 46003 | 46003 | 153 |
| nr075 | 56378 | 46525 | 46362 | 46362 | 181 |
| nr076 | 50105 | 39042 | 38965 | 38965 | 198 |
| nr077 | 53839 | 45055 | 44930 | 44903 | 272 |
| nr078 | 49264 | 40522 | 40432 | 40428 | 324 |
| nr079 | 1023 | 865 | 775 | 775 | 25 |
| nr080 | 76644 | 64132 | 63928 | 63928 | 348 |
| nr081 | 77967 | 62167 | 61521 | 61008 | 1391 |
| nr082 | 39055 | 32857 | 32731 | 32731 | 161 |
| nr083 | 33607 | 27830 | 27742 | 27719 | 227 |
| nr084 | 6891 | 5655 | 5515 | 5515 | 88 |
| nr085 | 63220 | 52028 | 51855 | 51853 | 218 |
| nr086 | 28118 | 21641 | 21539 | 21515 | 165 |
| nr087 | 1547 | 1240 | 1214 | 1214 | 44 |
| nr088 | 355 | 291 | 248 | 248 | 19 |
| nr089 | 94933 | 75244 | 74715 | 73698 | 62 |
| nr090 | 49193 | 39043 | 38697 | 38660 | 56 |
| nr091 | 43492 | 34421 | 34160 | 34137 | 35 |
| nr092 | 87487 | 69403 | 69153 | 69153 | 64 |
| nr093 | 170090 | 138581 | 137950 | 137467 | 90 |
| nr094 | 183908 | 136075 | 135427 | 135250 | 92 |
| nr095 | 187778 | 149971 | 149441 | 149364 | 86 |
| nr096 | 152388 | 120640 | 120247 | 120041 | 86 |
| nr097 | 151106 | 122605 | 122173 | 122018 | 69 |
| nr098 | 61905 | 49312 | 49062 | 49001 | 50 |
| nr099 | 6228 | 5126 | 5003 | 5003 | 32 |
| nr100 | 181 | 137 | 91 | 91 | 11 |
| nr101 | 118 | 86 | 47 | 47 | 7 |
| nr102 | 65 | 48 | 23 | 23 | 4 |
| nr103 | 196 | 145 | 78 | 78 | 9 |
| nr104 | 198 | 153 | 100 | 100 | 13 |
| nr105 | 179 | 145 | 82 | 82 | 11 |
| nr106 | 107 | 83 | 42 | 42 | 5 |
| nr107 | 176 | 144 | 68 | 68 | 7 |
| nr108 | 92 | 71 | 29 | 29 | 5 |
| nr109 | 139 | 113 | 67 | 67 | 9 |
| nr110 | 447 | 360 | 269 | 269 | 32 |
| nr111 | 122 | 96 | 52 | 52 | 4 |
| nr112 | 212 | 174 | 114 | 114 | 11 |
| nr113 | 144 | 116 | 74 | 74 | 5 |
| nr114 | 127 | 103 | 57 | 57 | 7 |
| nr115 | 502 | 393 | 378 | 378 | 11 |
| nr116 | 73 | 52 | 16 | 16 | 2 |
| nr117 | 127 | 103 | 46 | 46 | 6 |
| nr118 | 120 | 94 | 47 | 47 | 5 |
| nr119 | 170311 | 133480 | 132659 | 129636 | 87 |
| nr120 | 157 | 126 | 64 | 64 | 8 |
| nr121 | 161 | 131 | 72 | 72 | 11 |
| nr122 | 199 | 161 | 117 | 117 | 14 |
| nr123 | 60 | 48 | 14 | 14 | 2 |
| nr124 | 100 | 83 | 47 | 47 | 10 |
| nr125 | 84 | 66 | 35 | 35 | 6 |
| nr126 | 81 | 63 | 25 | 25 | 2 |
| nr127 | 208327 | 163285 | 162177 | 159439 | 112 |
| nr128 | 72407 | 59091 | 58754 | 58652 | 56 |
| nr129 | 45752 | 37244 | 37008 | 36922 | 44 |
| nr130 | 192968 | 154678 | 153444 | 145777 | 122 |
| nr131 | 19 | 17 | 13 | 13 | 2 |
| nr132 | 14 | 11 | 1 | 1 | 1 |

#### Assign taxonomy

#Create a DNAStringSet from the ASV sequences  
#taxonomy <- assignTaxonomy(seqtab.nochim, silva, multithread=3)  
taxonomy <- addSpecies(taxonomy, species, n = 500)  
saveRDS(taxonomy, "rds/taxonomy.rds")

# Load previously saved object  
taxonomy <- readRDS("rds/taxonomy.rds")

#### Create tse object

#Counts table  
counts <- t(seqtab.nochim)  
repseq <- DNAStringSet(rownames(counts))  
rownames(counts) <- NULL  
ASV\_names <- paste0("ASV", seq(nrow(counts)))  
#Taxonomy table  
rownames(taxonomy) <- NULL  
#Metadata  
#Create tse  
tse <- TreeSummarizedExperiment(assays = list(counts = counts),  
 rowData = DataFrame(taxonomy),  
 colData = DataFrame(metadata))  
names(repseq) <- ASV\_names  
referenceSeq(tse) <- repseq  
#View  
tse

class: TreeSummarizedExperiment   
dim: 7634 131   
metadata(0):  
assays(1): counts  
rownames: NULL  
rowData names(7): Kingdom Phylum ... Genus Species  
colnames(131): nr001 nr002 ... nr131 nr132  
colData names(5): SampleID Name Source Type Barcode\_seq  
reducedDimNames(0):  
mainExpName: NULL  
altExpNames(0):  
rowLinks: NULL  
rowTree: NULL  
colLinks: NULL  
colTree: NULL  
referenceSeq: a DNAStringSet (7634 sequences)

Some additional pruning

#remove taxa with unknown kingdom, discard empty  
tse <- tse[rowData(tse)$Kingdom %in% "Bacteria" |  
 rowData(tse)$Kingdom %in% "Archaea" |  
 !is.na(rowData(tse)$Kingdom)]  
#remove chloroplastic taxa, keep empty  
tse <- tse[!rowData(tse)$Order %in% "Chloroplast" | is.na(rowData(tse)$Order)]  
#remove mitochondrial taxa, keep empty  
tse <- tse[!rowData(tse)$Family %in% "Mitochondria" | is.na(rowData(tse)$Family)]  
#final object dimensions  
dim(tse)

[1] 7533 131

#### Write data

Last step is to save data to suitable file formats.

saveRDS(tse, "rds/tse.rds")

Variant sequences are saved to fasta

tse %>% referenceSeq() %>% writeXStringSet(paste0(exportloc,"/repseq.fasta"),  
 append=FALSE, compress=FALSE,  
 format="fasta")

Taxonomy is read from rowData and written

taxfile <- as.data.frame(rowData(tse))  
taxfile %>% rownames\_to\_column(var = "Variant") %>%  
 write\_tsv(file=paste0(exportloc,"/taxonomy.tsv"))

Metadata is read from colData and written

metadf <- data.frame(Sampleid = rownames(colData(tse)), colData(tse))  
write\_tsv(metadf, paste0(exportloc, "/metadata.tsv"))

Counts are read from counts and written

ASV\_counts <- as.data.frame(assays(tse)$counts)  
ASV\_counts %>% rownames\_to\_column(var= "Variant") %>%  
write\_tsv(file = paste0(exportloc, "/asvs.tsv"))

#### Vsearch denovo clustering (99%) data from Qiime

#Counts  
counts <- read\_tsv("data/qiime/feature-table.tsv", show\_col\_types =F)  
counts <- counts[,2:ncol(counts)]  
ASV\_names <- paste0("ASV", seq(1:nrow(counts)))  
#Taxonomy table  
taxonomy <- readRDS("data/qiime/taxid.rds")  
rownames(taxonomy) <- NULL  
colnames(taxonomy) <- c("Kingdom", "Phylum", "Class", "Order", "Family", "Genus", "Species")  
#Project metadata  
metadata <- read\_tsv("data/qiime/jt\_meta2.tsv", show\_col\_types = F)  
rownames(metadata) <- NULL  
#Create TSE  
tseq <- TreeSummarizedExperiment(assays = list(counts = counts),  
 rowData = DataFrame(taxonomy),  
 colData = DataFrame(metadata))  
rownames(tseq) <- ASV\_names  
#View  
tseq

class: TreeSummarizedExperiment   
dim: 127854 132   
metadata(0):  
assays(1): counts  
rownames(127854): ASV1 ASV2 ... ASV127853 ASV127854  
rowData names(7): Kingdom Phylum ... Genus Species  
colnames(132): nr001 nr002 ... nr132 nr072  
colData names(5): SampleID Name Source Type Barcode\_seq  
reducedDimNames(0):  
mainExpName: NULL  
altExpNames(0):  
rowLinks: NULL  
rowTree: NULL  
colLinks: NULL  
colTree: NULL

Pruning data before comparison

#Remove sampleid nr072  
tseq <- tseq[, colnames(tseq) != "nr072"]  
tseq

class: TreeSummarizedExperiment   
dim: 127854 131   
metadata(0):  
assays(1): counts  
rownames(127854): ASV1 ASV2 ... ASV127853 ASV127854  
rowData names(7): Kingdom Phylum ... Genus Species  
colnames(131): nr001 nr002 ... nr118 nr132  
colData names(5): SampleID Name Source Type Barcode\_seq  
reducedDimNames(0):  
mainExpName: NULL  
altExpNames(0):  
rowLinks: NULL  
rowTree: NULL  
colLinks: NULL  
colTree: NULL

#remove taxa with unknown kingdom, discard empty  
tseq <- tseq[rowData(tseq)$Kingdom %in% "Bacteria" | !is.na(rowData(tseq)$Kingdom),]  
#remove chloroplastic taxa, keep empty  
tseq <- tseq[!rowData(tseq)$Order %in% "Chloroplast" | is.na(rowData(tseq)$Order),]  
#remove mitochondrial taxa, keep empty  
tseq <- tseq[!rowData(tseq)$Family %in% "Mitochondria" | is.na(rowData(tseq)$Family),]  
#final object dimensions  
dim(tseq)

[1] 125822 131

Filter low abundance otus

# Extract the counts matrix from tseq and convert to data frame  
qiime\_data <- assays(tseq)$counts  
  
# Add row sums, filter rows where sum > 9, and retain row names  
qiime\_data <- data.frame(ASV = rownames(tseq),qiime\_data) %>%  
 mutate(sum = rowSums(qiime\_data)) %>%  
 filter(sum > 9) %>% column\_to\_rownames("ASV")  
tseq\_filtered <- tseq[rownames(qiime\_data),]  
# Check the result  
tseq\_filtered

class: TreeSummarizedExperiment   
dim: 15581 131   
metadata(0):  
assays(1): counts  
rownames(15581): ASV1 ASV2 ... ASV126176 ASV127537  
rowData names(7): Kingdom Phylum ... Genus Species  
colnames(131): nr001 nr002 ... nr118 nr132  
colData names(5): SampleID Name Source Type Barcode\_seq  
reducedDimNames(0):  
mainExpName: NULL  
altExpNames(0):  
rowLinks: NULL  
rowTree: NULL  
colLinks: NULL  
colTree: NULL

Remove samples containing less than 10 000 counts in total

#create column for total counts  
colData(tse)$Sum <- colSums(assays(tse)$counts)  
colData(tseq)$Sum <- colSums(assays(tseq)$counts)  
colData(tseq\_filtered)$Sum <- colSums(assays(tseq\_filtered)$counts)  
#filter samples that have over 10 000 counts  
tse <- tse[,colData(tse)$Sum >= 10000]  
tseq <- tseq[, colData(tseq)$Sum >= 10000]  
tseq\_filtered <- tseq\_filtered[, colData(tseq\_filtered)$Sum >= 10000]  
dim(tse)

[1] 7533 91

dim(tseq)

[1] 125822 91

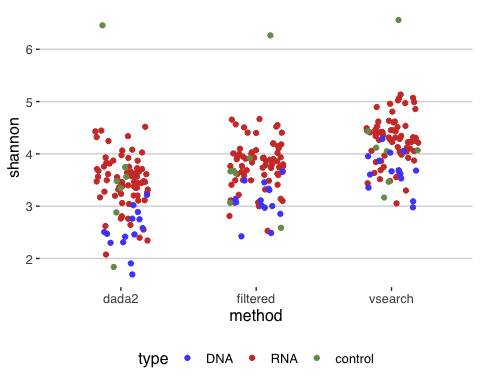
dim(tseq\_filtered)

[1] 15581 91

#create second copies  
#tse2 <- tse  
#tseq2 <- tseq

Calculate shannon index and compare denoising ja clustering results

#calculate shannon indexes  
tse <- estimateDiversity(tse, index = "shannon")  
tseq <- estimateDiversity(tseq, index = "shannon")  
tseq\_filtered <- estimateDiversity(tseq\_filtered, index = "shannon")  
#create df with values and sample type  
shannon <- data.frame(dada2 = colData(tse)$shannon, vsearch = colData(tseq)$shannon,  
 filtered = colData(tseq\_filtered)$shannon,  
 type = colData(tse)$Type)  
#pivot table  
long\_diversity <- shannon %>% pivot\_longer(col = c(dada2,vsearch,filtered),  
 names\_to = "method",  
 values\_to = "shannon")  
#plot data  
plot\_shannon <- ggplot(long\_diversity, aes(method,shannon, color=type)) +  
 geom\_point(position = position\_jitter(width=0.2)) + theme\_hc() +  
 scale\_color\_igv()  
plot\_shannon



Calculate bray-curtis and compare denoising ja clustering results

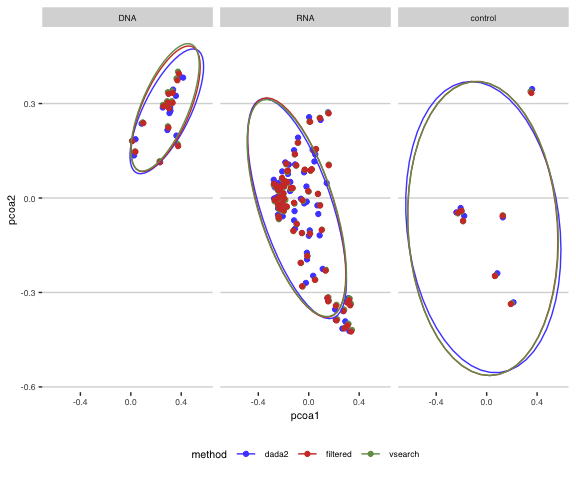
#transform dada counts to relabundance  
tse <- transformAssay(tse, assay.type = "counts", method = "relabundance",  
 name = "relabundance")  
#create bray-curtis distance matrix  
tse <- runMDS(tse, FUN = vegan::vegdist, method = "bray",  
 name="PCoA\_BC", exprs\_values = "relabundance")  
dada\_bray <- plotReducedDim(tse, "PCoA\_BC")  
#create df for plot  
dada\_df <- data.frame(pcoa1 = dada\_bray$data[,1],  
 pcoa2 = dada\_bray$data[,2],  
 type = colData(tse)$Type)  
#transform vsearch counts to relabundance  
tseq <- transformAssay(tseq, assay.type = "counts", method = "relabundance",  
 name = "relabundance")  
#create bray-curtis distance matrix  
tseq <- runMDS(tseq, FUN = vegan::vegdist, method = "bray",  
 name="PCoA\_BC", exprs\_values = "relabundance")  
vsearch\_bray <- plotReducedDim(tseq, "PCoA\_BC")  
#create df for plot  
vsearch\_df <- data.frame(pcoa1 = vsearch\_bray$data[,1],  
 pcoa2 = vsearch\_bray$data[,2],  
 type = colData(tseq)$Type)  
#transform vsearch counts to relabundance  
tseq\_filtered <- transformAssay(tseq\_filtered, assay.type = "counts", method = "relabundance",  
 name = "relabundance")  
tseq\_filtered <- runMDS(tseq\_filtered, FUN = vegan::vegdist, method = "bray",  
 name="PCoA\_BC", exprs\_values = "relabundance")  
filtered\_bray <- plotReducedDim(tseq\_filtered, "PCoA\_BC")  
#create df for plot  
filtered\_df <- data.frame(pcoa1 = filtered\_bray$data[,1],  
 pcoa2 = filtered\_bray$data[,2],  
 type = colData(tseq\_filtered)$Type)

Combine dataframes

#combined df  
bray\_both <- data.frame(dada1 = dada\_df$pcoa1,  
 dada2 = dada\_df$pcoa2,  
 vsearch1 = vsearch\_df$pcoa1,  
 vsearch2 = vsearch\_df$pcoa2,  
 filtered1 = filtered\_df$pcoa1,  
 filtered2 = filtered\_df$pcoa2,  
 type = dada\_df$type)  
bray\_both = data.frame(pcoa1=c(bray\_both$dada1, bray\_both$vsearch1,  
 bray\_both$filtered1),  
 pcoa2=c(bray\_both$dada2, bray\_both$vsearch2,  
 bray\_both$filtered2),  
 type=c(dada\_df$type,dada\_df$type, dada\_df$type))  
bray\_both$method <- "dada2"   
bray\_both$method[92:182] <- "vsearch"  
bray\_both$method[183:273] <- "filtered"  
plot\_pcoa <- ggplot(bray\_both, aes(pcoa1,pcoa2, color=method)) +  
 geom\_point() + facet\_wrap(~ type) + theme\_hc(base\_size=8) + scale\_color\_igv() + stat\_ellipse()

Plot Bray-Curtis

(plot\_pcoa)



**Observations**

* Customized sup basecalling produces high-quality reads.
* The error profiles of short amplicons (truncated to 350 bp) align with expected frequencies using the DADA2 learnErrors function.
* The proportion of unique reads is much smaller compared to long amplicons.
* Number of unique perfect matches found by DADA2addSpecies is well within expected range
* Overall, the findings suggest that denoising can effectively process short ONT reads.
* Shannon diversity is higher when using VSEARCH. Clustering at 99% identity produces a large number of variants, potentially causing overestimation. Filtering OTUs with fewer than 10 counts brings the Shannon index closer to DADA2 results.
* Bray-Curtis plots are very similar for samples with more than 10,000 counts.

**Advantages of Nanopore**

* Read length is not a limiting factor when designing amplicon targets.
* Base quality does not decrease with read length.
* Low-diversity libraries are not a problem for nanopore sequencing chemistry.
* Libraries prepared for other platforms can be conveniently converted to nanopore.
* Live basecalling allows real-time control of sequencing throughput, and in some cases, the flow cell can be reused.
* Cost per base in amplicon sequencing is competitive compared to MiSeq.

**Disadvantages of Nanopore**

* Homopolymer region accuracy is not as good as Illumina.
* High-accuracy basecalling is computationally intensive.
* Software tools are less mature compared to other platforms and require more technical expertise.
* The consistency of flow cells (e.g., number of functional pores) and sequencing repeatability are not yet fully clear. The shelf life of flow cells is uncertain; Oxford Nanopore provides a 3-month guarantee from the delivery date.
* Pores may degrade if the library preparation contains contaminants from samples (especially a concern for genomic or transcriptomic sequencing).