

Custom array sequence generation

Tomas Bjorklund

Wed Jan 13 10:18:35 2016

This script generates all AA unique AA sequences for the CustomArray production

```
suppressPackageStartupMessages(library(knitr))
suppressPackageStartupMessages(library(ShortRead))
suppressPackageStartupMessages(library(parallel))
suppressPackageStartupMessages(library(GeneGA))
suppressPackageStartupMessages(library(devtools))
suppressPackageStartupMessages(library(Hmisc))
```

Loading source files

```
source(file.path("functions", "AAtoDNA.R"))

allSequences <- readFasta("input/DNA-lib_RetrogradeTransport.fasta")
AAlist <- data.frame(Class=character(),
                      Family=character(),
                      Strain=character(),
                      Note=character(),
                      Number=character(),
                      Name=character(),
                      AAfragment=character(),
                      stringsAsFactors = FALSE)

#allSequences <- allSequences[124:129]

strt<-Sys.time()
for (i in 1:length(allSequences)){
  thisID <- as.character(ShortRead::id(allSequences[i]))
  thisSeq <- sread(allSequences[i])
  thisAA <- Biostrings::translate(thisSeq, genetic.code=GENETIC_CODE, if.fuzzy.codon="solve")
  AAlist[i,c("Class","Family","Strain","Note",
            "Number","Name","AAfragment")] <- c(BBmisc::explode(thisID, sep=","),
                                         as.character(thisAA))
}
}
```

The generateFragments function

```
generateFragments <- function(minLength,maxLength,frequency=1) {
  #Generate a table to store all AA sequences
  fragList <- data.frame(Class=character(),
                          Family=character(),
                          Strain=character(),
                          Note=character(),
                          Number=character(),
                          Name=character(),
                          AAstart=integer(),
```

```

AAstop=integer(),
AAfragment=character(),
stringsAsFactors = FALSE)

makeAllFrags <- function(k){
  thisFullAA <- AAlist[k,"AAfragment"]
  count = length(fragList[,1])+1
  for (l in seq(1,(width(thisFullAA)-minLength),frequency)){
    for (m in (l+minLength-1):(min(l+maxLength-1,width(thisFullAA)))) {
      #Truncate string to the relevant fragment:
      thisFragment <- substr(thisFullAA,l,m)
      #Take away any sequence that starts with a start codon ATG:
      if (substr(thisFragment,1,1)!="M") {
        fragList[count,c("Class","Family","Strain","Note","Number",
          "Name","AAstart","AAstop",
          "AAfragment")] <- c(AAlist[k,c("Class","Family","Strain",
            "Note","Number","Name")],l,m,thisFragment)
      }
      ## Inserts the fragment with information into the new data frame
      count <- count +1
    }
  }
  return(fragList)
}

fragList <- do.call(rbind,mcclapply(1:length(AAlist[,1]),makeAllFrags,
  mc.preschedule = TRUE, mc.cores = detectCores()))

#Control if any sequences contain non-AA characters and save them into a separate list
discardList <- fragList[grep("[[:punct:]]|X",fragList[, "AAfragment"]),]

#Remove any sequence containing non AA characters
fragList <- fragList[grep("[[:punct:]]|X",fragList[, "AAfragment"], invert = TRUE),]

#Sort the fragments, find unique strings and count number of duplicates
sortedFragments <- rev(sort(table(fragList[, "AAfragment"])))

#Run the AAtoDNA function to convert all AA sequences to human codon-optimized DNA sequences
row.names(sortedFragments) <- mcclapply(row.names(sortedFragments), fullIOPT=FALSE, species="hsa",
  AAtoDNA, mc.preschedule = TRUE, mc.set.seed = TRUE,
  mc.silent = FALSE, mc.cores = detectCores(), mc.cleanup = TRUE) #

sortedFragments <- sortedFragments[order(row.names(sortedFragments))]

return(sortedFragments)
}

```

Execution of the function

```
sortedFragments.14aa <- generateFragments(14,14,1)
```

Add the overhangs for amplification PCR and Gibson assembly into the AAV plasmid

```

fivePrime <- tolower("AACCTCCAGAGAGGCAACGCT")
threePrime <- tolower("GCCAGACAAGCAGCTACCGCA")
row.names(sortedFragments.14aa) <- paste(fivePrime, row.names(sortedFragments.14aa),
                                         threePrime, sep = "")

sortedFragments.14aa.G4S <- generateFragments(14, 14, 3)
sortedFragments.14aa.A5 <- sortedFragments.14aa.G4S

```

Add the overhangs including G4S spacers for amplication PCR and Gibson assembly into the AAV plasmid

```

fivePrime <- tolower("AACCTCCAGAGAGGCAACGGAGGCAGGAAGT")
threePrime <- tolower("GGAGGCGCGGAAGCAGACAAGCAGCTACCGCA")
row.names(sortedFragments.14aa.G4S) <- paste(fivePrime, row.names(sortedFragments.14aa.G4S),
                                              threePrime, sep = "")

```

Add the overhangs including A5 spacers for amplication PCR and Gibson assembly into the AAV plasmid

```

fivePrime <- tolower("AACCTCCAGAGAGGCAACGCTGCTGCAGCAGCC")
threePrime <- tolower("GCAGCTGCAGCTGCCAGACAAGCAGCTACCGCA")
row.names(sortedFragments.14aa.A5) <- paste(fivePrime, row.names(sortedFragments.14aa.A5),
                                              threePrime, sep = "")

```

Generate 22aa fragments

```
sortedFragments.22aa <- generateFragments(22, 22, 3)
```

Add the overhangs for amplication PCR and Gibson assembly into the AAV plasmid

```

fivePrime <- tolower("AACCTCCAGAGAGGCAACGCT")
threePrime <- tolower("GCCAGACAAGCAGCTACCGCA")
row.names(sortedFragments.22aa) <- paste(fivePrime, row.names(sortedFragments.22aa),
                                         threePrime, sep = "")

```

Merge all separate fragment lists into one complete list

```

sortedFragments <- c(sortedFragments.22aa, sortedFragments.14aa,
                      sortedFragments.14aa.A5, sortedFragments.14aa.G4S)

print(paste("Number of unique fragments:", length(unique(names(sortedFragments)))), sep = " ")

```

```
## [1] "Number of unique fragments: 92358"
```

```
write.table(c("Sequence", unique(names(sortedFragments))),
            "data/SortedFragments_all.txt", row.names=F, col.names=F, quote=F, sep="\t")
```

```
print(Sys.time() - start)
```

```
## Time difference of 2.857151 mins
```

```
devtools::session_info()
```

```
## Session info -----

```

```

## setting  value
## version  R version 3.2.3 (2015-12-10)
## system   x86_64, linux-gnu
## ui        X11
## language en_US:en
## collate  en_US.UTF-8
## tz       <NA>
## date    2016-01-13

## Packages -----
## package      * version date     source
## acepack       1.3-3.3 2014-11-24 CRAN (R 3.2.0)
## ade4          * 1.7-3   2015-11-22 CRAN (R 3.2.2)
## BBmisc         1.9     2015-02-03 CRAN (R 3.2.0)
## Biobase        * 2.30.0  2015-11-26 Bioconductor
## BiocGenerics  * 0.16.1  2015-11-26 Bioconductor
## BiocParallel   * 1.4.0   2015-11-26 Bioconductor
## Biostrings     * 2.38.2  2015-11-26 Bioconductor
## bitops          1.0-6   2013-08-17 CRAN (R 3.2.0)
## checkmate      1.6.3   2015-10-23 CRAN (R 3.2.2)
## cluster         2.0.3   2015-07-21 CRAN (R 3.2.2)
## colorspace      1.2-6   2015-03-11 CRAN (R 3.2.0)
## devtools        * 1.9.1   2015-09-11 CRAN (R 3.2.2)
## digest           0.6.8   2014-12-31 CRAN (R 3.2.0)
## evaluate         0.8     2015-09-18 CRAN (R 3.2.2)
## foreign          0.8-66  2015-08-19 CRAN (R 3.2.2)
## Formula          * 1.2-1   2015-04-07 CRAN (R 3.2.0)
## futile.logger   1.4.1   2015-04-20 CRAN (R 3.2.0)
## futile.options   1.0.0   2010-04-06 CRAN (R 3.2.0)
## GeneGA          * 1.20.0  2015-11-26 Bioconductor
## GenomeInfoDb    * 1.6.1   2015-11-26 Bioconductor
## GenomicAlignments * 1.6.1   2015-11-26 Bioconductor
## GenomicRanges   * 1.22.1  2015-11-26 Bioconductor
## ggplot2          * 1.0.1   2015-03-17 CRAN (R 3.2.0)
## gridExtra         2.0.0   2015-07-14 CRAN (R 3.2.2)
## gtable            0.1.2   2012-12-05 CRAN (R 3.2.0)
## hash              * 2.2.6   2013-02-21 CRAN (R 3.2.0)
## Hmisc             * 3.17-0  2015-09-21 CRAN (R 3.2.2)
## htmltools          0.2.6   2014-09-08 CRAN (R 3.2.0)
## hwriter            1.3.2   2014-09-10 CRAN (R 3.2.0)
## IRanges           * 2.4.4   2015-11-26 Bioconductor
## knitr             * 1.11    2015-08-14 CRAN (R 3.2.2)
## lambda.r           1.1.7   2015-03-20 CRAN (R 3.2.0)
## lattice            * 0.20-33 2015-07-14 CRAN (R 3.2.2)
## latticeExtra       0.6-26  2013-08-15 CRAN (R 3.2.0)
## magrittr            1.5     2014-11-22 CRAN (R 3.2.2)
## MASS               7.3-45  2015-11-10 CRAN (R 3.2.2)
## memoise            0.2.1   2014-04-22 CRAN (R 3.2.2)
## munsell            0.4.2   2013-07-11 CRAN (R 3.2.0)
## nnet                7.3-11  2015-08-30 CRAN (R 3.2.2)
## plyr                 1.8.3   2015-06-12 CRAN (R 3.2.2)
## proto                 0.3-10  2012-12-22 CRAN (R 3.2.0)
## RColorBrewer        1.1-2   2014-12-07 CRAN (R 3.2.0)
## Rcpp                 0.12.2  2015-11-15 CRAN (R 3.2.2)
## reshape2            1.4.1   2014-12-06 CRAN (R 3.2.0)
## rmarkdown            0.8.1   2015-10-10 CRAN (R 3.2.2)

```

```
## rpart           4.1-10  2015-06-29 CRAN (R 3.2.2)
## Rsamtools       * 1.22.0  2015-11-26 Bioconductor
## S4Vectors       * 0.8.3   2015-11-26 Bioconductor
## scales          0.3.0    2015-08-25 CRAN (R 3.2.2)
## seqinr          * 3.1-3   2014-12-17 CRAN (R 3.2.0)
## ShortRead       * 1.28.0  2015-11-26 Bioconductor
## stringi          1.0-1    2015-10-22 CRAN (R 3.2.2)
## stringr          1.0.0    2015-04-30 CRAN (R 3.2.2)
## SummarizedExperiment * 1.0.1  2015-11-26 Bioconductor
## survival        * 2.38-3  2015-07-02 CRAN (R 3.2.2)
## XVector          * 0.10.0  2015-11-26 Bioconductor
## yaml             2.1.13   2014-06-12 CRAN (R 3.2.0)
## zlibbioc         1.16.0   2015-11-26 Bioconductor
```

Library analysis output

Tomas Bjorklund

Wed Jan 13 10:21:35 2016

This workflow brings together FastQ files containing barcodes and the gene fragments synthesized with the CustomArray. Requires bbmap2. The fragments are then suitable for alignment to reference sequences using Bowtie2.

```
suppressPackageStartupMessages(library(knitr))
suppressPackageStartupMessages(library(ShortRead))
suppressPackageStartupMessages(library(parallel))
suppressPackageStartupMessages(library(doParallel))
suppressPackageStartupMessages(library(data.table))
suppressPackageStartupMessages(library(devtools))
suppressPackageStartupMessages(library(Hmisc))

opts_chunk$set(fig.width = 7.5, fig.height = 8)
opts_chunk$set(comment = NA)

config <- read.table("input/config.txt", header = FALSE, skip = 0, sep="\t",
                     stringsAsFactors = FALSE, fill=TRUE)
colnames(config) <- c("Parameter", "Value")
```

Sequencing files

```
dataDir <- config$Value[1]
in.name.P5 <- file.path(dataDir, config$Value[2])
in.name.P7 <- file.path(dataDir, config$Value[3])
name.out <- config$Value[4]
paired.alignment <- as.logical(config$Value[5])
```

Analysis parameters

```
knitr::kable(config, format = "markdown")
```

Parameter	Value
dataDir	seqFiles/DNA
in.name.P5	psc-lib-1-2UndetOldSort_S1_L001_R1_001.fastq.gz
in.name.P7	psc-lib-1-2UndetOldSort_S1_L001_R2_001.fastq.gz
name.out	AAVlibrary_complete
paired.alignment	TRUE
run.subset	FALSE
max.cores	32
subset.count	250000

```

run.subset <- as.logical(config$value[6])
max.cores <- as.integer(config$value[7])
subset.count <- as.integer(config$value[8])

strt<-Sys.time()

```

Selection of real amplicons

```

# This section searches the sequencing file and only select the files with valid amplicons
out.name.P5 <- tempfile(pattern = "P5_", tmpdir = tempdir(), fileext = ".fastq.gz")
out.name.P7 <- tempfile(pattern = "P7_", tmpdir = tempdir(), fileext = ".fastq.gz")
command.args <- paste("-Xmx12g overwrite=true k=15 rcomp=f skipr2=t qhdist=0 maskmiddle=f",
                      " hammingdistance=2 findbestmatch=f ordered=t threads=", detectCores(),
                      " in=", in.name.P5,
                      " in2=", in.name.P7,
                      " outm=", out.name.P5,
                      " outm2=", out.name.P7,
                      " fliteral=", "GTATGTTGTTCTGGAGCGGGAGGGTGCTATTTGCCTAGCGATAA", sep = "")
sys.out <- system2(path.expand("~/bbmap/bbduk2.sh"), args=command.args, stdout=TRUE, stderr=TRUE)

sys.out <- as.data.frame(sys.out)

colnames(sys.out) <- c("bbduk2 Identification of real amplicons")
invisible(sys.out[" "] <- " ")
lengthOut <- (nrow(sys.out))
knitr::kable(sys.out[3:lengthOut,], format = "markdown")

```

bbduk2 Identification of real amplicons

```

3
4   BBDuk2 version 34.79
5   Set ORDERED to true
6   Set threads to 32
7   k=15
8   hamming distance=2
9   kfiltering using 1 literal.
10
11  Initial:
12  Memory: free=12090m, used=258m
13
14  Added 30721 kmers; time: 0.083 seconds.
15  Memory: free=11639m, used=709m
16
17  Input is being processed as paired
18  Started output streams: 0.103 seconds.
19  Processing time: 84.814 seconds.

```

```
20
21 Input: 23191088 reads 3490215687 bases.
22 Contaminants: 23095890 reads (99.59%) 3475908371 bases (99.59%)
23 Result: 95198 reads (0.41%) 14307316 bases (0.41%)
24
25 Time: 85.009 seconds.
26 Reads Processed: 23191k 272.81k reads/sec
27 Bases Processed: 3490m 41.06m bases/sec
```

```
in.name.P5 <- out.name.P5
in.name.P7 <- out.name.P7
```

Extraction of a subset

```
if (run.subset){
  suppressWarnings(sampler <- FastqSampler(gsub("[\\\"]", "", in.name.P5), subset.count,
                                             readerBlockSize=1e9, ordered = TRUE))
  set.seed(123); tmp.P5 <- yield(sampler)
  in.name.P5 <- tempfile(pattern = "P5_", tmpdir = tempdir(), fileext = ".fastq.gz")
  writeFastq(tmp.P5,in.name.P5, compress=TRUE)
  rm(tmp.P5)
  suppressWarnings(sampler <- FastqSampler(gsub("[\\\"]", "", in.name.P7), subset.count,
                                             readerBlockSize=1e9, ordered = TRUE))
  set.seed(123); tmp.P7 <- yield(sampler)
  in.name.P7 <- tempfile(pattern = "P7_", tmpdir = tempdir(), fileext = ".fastq.gz")
  writeFastq(tmp.P7,in.name.P7, compress=TRUE)
  rm(tmp.P7)
}

output.Reads <- as.integer(system(paste("gunzip -c ",shQuote(gsub("[\\\"]", "", in.name.P5)),
                                     " | echo $((`wc -l`/4)) 2>&1", sep = ""), intern = TRUE,
                                     ignore.stdout = FALSE)) #Stores the read count utilized
print(paste("Utilized sequences:", output.Reads))

[1] "Utilized sequences: 11547945"
```

Extraction of barcodes

```
out.name.P5 <- tempfile(pattern = "BC_", tmpdir = tempdir(), fileext = ".fastq.gz")

sys.out <- system(paste("~/bbmap/bbduk2.sh overwrite=true k=18 mink=18 hammingdistance=2 findbestmatch=t ",
                         "rcomp=f findbestmatch=f qhdist=1 minavgquality=0 maxns=0 minlength=18 ",
                         "maxlength=22 threads=", detectCores()," in=", shQuote(in.name.P5),
                         " out=", out.name.P5," lliteral=", "GGCCTAGCGGCCGTTACTT",
                         " rliteral=", "ATAACCTCGTATAATGTATGC",
```

```

    " 2>&1", sep = ""), intern = TRUE, ignore.stdout = FALSE)
sys.out <- as.data.frame(sys.out)

in.name.P5 <- out.name.P5

colnames(sys.out) <- c("bbduk2 Extraction of barcodes")
invisible(sys.out[[" "]] <- " ")
lengthOut <- (nrow(sys.out))
knitr::kable(sys.out[3:lengthOut,], format = "markdown")

```

bbduk2 Extraction of barcodes

```

3
4   BBDuk2 version 34.79
5   Set threads to 32
6   k=18
7   maskMiddle=true
8   hamming distance=2
9   right-ktrimming using 1 literal.
10  left-ktrimming using 1 literal.
11
12 Initial:
13 Memory: free=77148m, used=2496m
14
15 Added 5104 kmers; time: 0.047 seconds.
16 Memory: free=74239m, used=5405m
17
18 Added 5104 kmers; time: 0.001 seconds.
19 Memory: free=74239m, used=5405m
20
21 Input is being processed as unpaired
22 Started output streams: 0.018 seconds.
23 Processing time: 381.797 seconds.
24
25 Input: 11547945 reads 1738273591 bases.
26 KTrimmed: 23030031 reads (199.43%) 1505373909 bases (86.60%)
27 Low quality discards: 6 reads (0.00%) 122 bases (0.00%)
28 Result: 11374106 reads (98.49%) 227257668 bases (13.07%)
29
30 Time: 381.875 seconds.
31 Reads Processed: 11547k 30.24k reads/sec
32 Bases Processed: 1738m 4.55m bases/sec

```

```

rm(sys.out)

reads.BC <- readFastq(in.name.P5)
sread(reads.BC)

A DNAStringSet instance of length 11374106
width seq
[1] 21 CTGTGCTGAATTACTTATACA
[2] 20 GTAGGTATCTAGACTCCTTT
[3] 20 GTACGTTGACTGGAAAGCTGG
[4] 20 GTGTGTAGCAGGATACGCCGG
[5] 20 CATGAGTTATTTGATGATAT
...
[11374102] 20 GTATCAACGAATGCGTCATT
[11374103] 20 GTTTATGGGAGTGCTGCCGT
[11374104] 20 GTGGGATTGAAGGATGGTGT
[11374105] 20 GAGGGCTTATGTCAATGCGT
[11374106] 20 GATGGAGTGTAGGTGTGTTG

(unique.BCs <- unique(sread(reads.BC)))

A DNAStringSet instance of length 3934570
width seq
[1] 21 CTGTGCTGAATTACTTATACA
[2] 20 GTAGGTATCTAGACTCCTTT
[3] 20 GTACGTTGACTGGAAAGCTGG
[4] 20 GTGTGTAGCAGGATACGCCGG
[5] 20 CATGAGTTATTTGATGATAT
...
[3934566] 20 GTGTGTTCATTTCATTTGTTT
[3934567] 20 GAACATGGCTTGGAGTGCCGG
[3934568] 20 GTATCAACGAATGCGTCATT
[3934569] 20 GTTTATGGGAGTGCTGCCGT
[3934570] 20 GTGGGATTGAAGGATGGTGT

output.BCs <- length(unique.BCs)
print(paste("Utilized barcodes:", output.BCs))

[1] "Utilized barcodes: 3934570"

barcodeTable <- data.table(ID=as.character(ShortRead::id(reads.BC)), BC=as.character(sread(reads.BC)))

```

Extraction of fragments

```

out.name.P7 <- tempfile(pattern = "P7_", tmpdir = tempdir(), fileext = ".fastq.gz")
command.args <- paste("-Xmx12g overwrite=true k=18 mink=18 rcomp=f qhdist=1 maskmiddle=t",
                      "hammingdistance=2 findbestmatch=t minlength=38 maxlength=78 ordered=t ",
                      "threads=", detectCores(), " in=", in.name.P7, " out=", out.name.P7,
                      " lliteral=", "AGAACCTCCAGAGAGGCAACG",
                      " rliteral=", "CAGACAAGCAGCTACCGCAGAT", sep = "")
sys.out <- system2(path.expand("~/bbmap/bbduk2.sh"), args=command.args, stdout=TRUE, stderr=TRUE) #

```

```

sys.out <- as.data.frame(sys.out)

colnames(sys.out) <- c("bbduk2 extraction of fragments")
invisible(sys.out[[" "]] <- " ")
lengthOut <- (nrow(sys.out))
knitr::kable(sys.out[3:lengthOut,], format = "markdown")

```

bbduk2 extraction of fragments	
3	
4	BBDuk2 version 34.79
5	Set ORDERED to true
6	Set threads to 32
7	k=18
8	maskMiddle=true
9	hamming distance=2
10	right-ktrimming using 1 literal.
11	left-ktrimming using 1 literal.
12	
13	Initial:
14	Memory: free=11959m, used=389m
15	
16	Added 6380 kmers; time: 0.252 seconds.
17	Memory: free=11508m, used=840m
18	
19	Added 6380 kmers; time: 0.003 seconds.
20	Memory: free=11508m, used=840m
21	
22	Input is being processed as unpaired
23	Started output streams: 0.633 seconds.
24	Processing time: 428.708 seconds.
25	
26	Input: 11547945 reads 1737634780 bases.
27	KTrimmed: 22321617 reads (193.30%) 1078844314 bases (62.09%)
28	Result: 10849152 reads (93.95%) 591375852 bases (34.03%)
29	
30	Time: 429.633 seconds.
31	Reads Processed: 11547k 26.88k reads/sec
32	Bases Processed: 1737m 4.04m bases/sec

```

in.name.P7 <- out.name.P7

out.name.P5 <- tempfile(pattern = "P5_", tmpdir = tempdir(), fileext = ".fastq.gz")

```

```

out.name.P7 <- tempfile(pattern = "P7_", tmpdir = tempdir(), fileext = ".fastq.gz")
out.name.P5_singlet <- tempfile(pattern = "P5_singlet_", tmpdir = tempdir(), fileext = ".fastq.gz")
out.name.P7_singlet <- tempfile(pattern = "P7_singlet_", tmpdir = tempdir(), fileext = ".fastq.gz")

sys.out <- system(paste("pairfq makepairs -c 'gzip' -f ", in.name.P5, " -r ", in.name.P7,
                        " -fp ", out.name.P5, " -rp ", out.name.P7, " -fs ",
                        out.name.P5_singlet, " -rs ", out.name.P7_singlet,
                        " --stats 2>&1", sep = ""), intern = TRUE, ignore.stdout = FALSE)
sys.out <- as.data.frame(sys.out)

colnames(sys.out) <- c("pairfq pair matching")
invisible(sys.out[[" "]] <- " ")
lengthOut <- (nrow(sys.out))
knitr::kable(sys.out[1:lengthOut,], format = "markdown")

```

pairfq pair matching

```

===== pairfq version : 0.15 (completion time: Wed Jan 13 10:43:39 CET 2016)
Total forward reads (/tmp/RtmpnpnI6ee/BC_195c76d2af98.fastq.gz) : 11374106
Total reverse reads (/tmp/RtmpnpnI6ee/P7_195ccf10460.fastq.gz) : 10849152
Total forward paired reads (/tmp/RtmpnpnI6ee/P5_195c3797e3fe.fastq.gz) : 10698072
Total reverse paired reads (/tmp/RtmpnpnI6ee/P7_195c75634eb1.fastq.gz) : 10698072
Total forward unpaired reads (/tmp/RtmpnpnI6ee/P5_singlet_195c585764.fastq.gz) : 676034
Total reverse unpaired reads (/tmp/RtmpnpnI6ee/P7_singlet_195c34d6b7a3.fastq.gz) : 151080

Total paired reads : 21396144
Total unpaired reads : 827114

```

```
rm(sys.out)
```

```

system(paste("mv ", out.name.P5, " ./data/barcodes_", name.out, ".fastq.gz", sep=""))
system(paste("mv ", out.name.P7, " ./data/fragments_", name.out, ".fastq.gz", sep=""))

unlink(paste(tempdir(), "/*", sep = ""), recursive = FALSE, force = FALSE) #Cleanup of temp files

print("Total execution time:")

```

[1] "Total execution time:"

```
print(Sys.time()-strt)
```

Time difference of 22.0035 mins

```
devtools::session_info()
```

Session info -----

setting value

```

version R version 3.2.3 (2015-12-10)
system x86_64, linux-gnu
ui      X11
language en_US:en
collate en_US.UTF-8
tz      <NA>
date    2016-01-13

```

Packages -----

package	*	version	date	source
acepack		1.3-3.3	2014-11-24	CRAN (R 3.2.0)
Biobase	*	2.30.0	2015-11-26	Bioconductor
BiocGenerics	*	0.16.1	2015-11-26	Bioconductor
BiocParallel	*	1.4.0	2015-11-26	Bioconductor
Biostrings	*	2.38.2	2015-11-26	Bioconductor
bitops		1.0-6	2013-08-17	CRAN (R 3.2.0)
chron		2.3-47	2015-06-24	CRAN (R 3.2.2)
cluster		2.0.3	2015-07-21	CRAN (R 3.2.2)
codetools		0.2-14	2015-07-15	CRAN (R 3.2.2)
colorspace		1.2-6	2015-03-11	CRAN (R 3.2.0)
data.table	*	1.9.6	2015-09-19	CRAN (R 3.2.2)
devtools	*	1.9.1	2015-09-11	CRAN (R 3.2.2)
digest		0.6.8	2014-12-31	CRAN (R 3.2.0)
doParallel	*	1.0.10	2015-10-14	CRAN (R 3.2.2)
evaluate		0.8	2015-09-18	CRAN (R 3.2.2)
foreach	*	1.4.3	2015-10-13	CRAN (R 3.2.2)
foreign		0.8-66	2015-08-19	CRAN (R 3.2.2)
formatR		1.2.1	2015-09-18	CRAN (R 3.2.2)
Formula	*	1.2-1	2015-04-07	CRAN (R 3.2.0)
futile.logger		1.4.1	2015-04-20	CRAN (R 3.2.0)
futile.options		1.0.0	2010-04-06	CRAN (R 3.2.0)
GenomeInfoDb	*	1.6.1	2015-11-26	Bioconductor
GenomicAlignments	*	1.6.1	2015-11-26	Bioconductor
GenomicRanges	*	1.22.1	2015-11-26	Bioconductor
ggplot2	*	1.0.1	2015-03-17	CRAN (R 3.2.0)
gridExtra		2.0.0	2015-07-14	CRAN (R 3.2.2)
gttable		0.1.2	2012-12-05	CRAN (R 3.2.0)
highr		0.5.1	2015-09-18	CRAN (R 3.2.2)
Hmisc	*	3.17-0	2015-09-21	CRAN (R 3.2.2)
htmltools		0.2.6	2014-09-08	CRAN (R 3.2.0)
hwriter		1.3.2	2014-09-10	CRAN (R 3.2.0)
IRanges	*	2.4.4	2015-11-26	Bioconductor
iterators	*	1.0.8	2015-10-13	CRAN (R 3.2.2)
knitr	*	1.11	2015-08-14	CRAN (R 3.2.2)
lambda.r		1.1.7	2015-03-20	CRAN (R 3.2.0)
lattice	*	0.20-33	2015-07-14	CRAN (R 3.2.2)
latticeExtra		0.6-26	2013-08-15	CRAN (R 3.2.0)
magrittr		1.5	2014-11-22	CRAN (R 3.2.2)
MASS		7.3-45	2015-11-10	CRAN (R 3.2.2)
memoise		0.2.1	2014-04-22	CRAN (R 3.2.2)
munsell		0.4.2	2013-07-11	CRAN (R 3.2.0)
nnet		7.3-11	2015-08-30	CRAN (R 3.2.2)
plyr		1.8.3	2015-06-12	CRAN (R 3.2.2)
proto		0.3-10	2012-12-22	CRAN (R 3.2.0)
RColorBrewer		1.1-2	2014-12-07	CRAN (R 3.2.0)
Rcpp		0.12.2	2015-11-15	CRAN (R 3.2.2)

reshape2	1.4.1	2014-12-06 CRAN (R 3.2.0)
rmarkdown	0.8.1	2015-10-10 CRAN (R 3.2.2)
rpart	4.1-10	2015-06-29 CRAN (R 3.2.2)
Rsamtools	* 1.22.0	2015-11-26 Bioconductor
S4Vectors	* 0.8.3	2015-11-26 Bioconductor
scales	0.3.0	2015-08-25 CRAN (R 3.2.2)
ShortRead	* 1.28.0	2015-11-26 Bioconductor
stringi	1.0-1	2015-10-22 CRAN (R 3.2.2)
stringr	1.0.0	2015-04-30 CRAN (R 3.2.2)
SummarizedExperiment	* 1.0.1	2015-11-26 Bioconductor
survival	* 2.38-3	2015-07-02 CRAN (R 3.2.2)
XVector	* 0.10.0	2015-11-26 Bioconductor
yaml	2.1.13	2014-06-12 CRAN (R 3.2.0)
zlibbioc	1.16.0	2015-11-26 Bioconductor

Library identification

Tomas Bjorklund

Wed Jan 13 10:43:48 2016

This workflow aligns the library fragments to the full reference sequences using Bowtie2.

```
suppressPackageStartupMessages(library(knitr))
suppressPackageStartupMessages(library(data.table))
suppressPackageStartupMessages(library(ShortRead))
suppressPackageStartupMessages(library(Hmisc))
suppressPackageStartupMessages(library(GeneGA))
suppressPackageStartupMessages(library(devtools))

opts_chunk$set(fig.width = 7.5, fig.height = 8)
opts_chunk$set(comment = NA)
```

Load sequences

```
LUT.dna <- read.table("data/SortedFragments_all.txt",
                      header = TRUE, skip = 0, sep="\t", stringsAsFactors = FALSE, fill=TRUE)
LUT.dna <- data.table(LUT.dna)
```

Remove constitutive backbone sequences

```
invisible(LUT.dna[, Sequence:=gsub("aacctccagagaggcaacg", "", Sequence)])
invisible(LUT.dna[, Sequence:=gsub("cagacaaggcgtaccgca", "", Sequence)])
invisible(LUT.dna[, Sequence:=toupper(Sequence)])
setkey(LUT.dna, "Sequence")
LUT.dna <- unique(LUT.dna)
LUT.dna$Names <- LUT.dna$Sequence
```

Split sequences based on linker and length

```
LUT.14aaG4S <- LUT.dna[substr(LUT.dna$Sequence,1,14) == "GAGGCGGAGGAAGT"]
LUT.remaining <- LUT.dna[!(substr(LUT.dna$Sequence,1,14) == "GAGGCGGAGGAAGT")]
LUT.14aaA5 <- LUT.remaining[substr(LUT.remaining$Sequence,1,14) == "CTGCTGCAGCAGCC"]
LUT.remaining <- LUT.remaining[!(substr(LUT.remaining$Sequence,1,14) == "CTGCTGCAGCAGCC")]
LUT.22aa <- LUT.remaining[nchar(LUT.remaining$Sequence) == 70L &
                           substr(LUT.remaining$Sequence,1,2) == "CT"]
LUT.remaining <- LUT.remaining[!(nchar(LUT.remaining$Sequence) == 70L &
                           substr(LUT.remaining$Sequence,1,2) == "CT")]
LUT.14aa <- LUT.remaining[nchar(LUT.remaining$Sequence) == 46L &
                           substr(LUT.remaining$Sequence,1,2) == "CT"]
rm(LUT.remaining)
LUT.dna[LUT.dna$Sequence %in% LUT.14aaG4S$Sequence, "Structure"] <- "14aaG4S"
```

```

LUT.dna[LUT.dna$Sequence %in% LUT.14aaA5$Sequence, "Structure"] <- "14aaA5"
LUT.dna[LUT.dna$Sequence %in% LUT.22aa$Sequence, "Structure"] <- "22aa"
LUT.dna[LUT.dna$Sequence %in% LUT.14aa$Sequence, "Structure"] <- "14aa"

save(LUT.dna, file = "data/LUTdna.rda")

```

Trim sequences

```

LUT.14aa$Sequence <- substr(LUT.14aa$Sequence, 3, 44)
LUT.14aaG4S$Sequence <- substr(LUT.14aaG4S$Sequence, 15, 56)
LUT.14aaA5$Sequence <- substr(LUT.14aaA5$Sequence, 15, 56)
LUT.22aa$Sequence <- substr(LUT.22aa$Sequence, 3, 68)

```

Save fasta files for Bowtie alignments

```

LUT.14aa.fa <- tempfile(pattern = "LUT_14aa_", tmpdir = tempdir(), fileext = "fa")
LUT.14aa.seq = ShortRead(DNAStringSet(LUT.14aa$Sequence), BStringSet(LUT.14aa$Names))
writeFasta(LUT.14aa.seq, LUT.14aa.fa)

LUT.14aaG4S.fa <- tempfile(pattern = "LUT_14aaG4s_", tmpdir = tempdir(), fileext = "fa")
LUT.14aaG4S.seq = ShortRead(DNAStringSet(LUT.14aaG4S$Sequence), BStringSet(LUT.14aaG4S$Names))
writeFasta(LUT.14aaG4S.seq, LUT.14aaG4S.fa)

LUT.14aaA5.fa <- tempfile(pattern = "LUT_14aaA5_", tmpdir = tempdir(), fileext = "fa")
LUT.14aaA5.seq = ShortRead(DNAStringSet(LUT.14aaA5$Sequence), BStringSet(LUT.14aaA5$Names))
writeFasta(LUT.14aaA5.seq, LUT.14aaA5.fa)

LUT.22aa.fa <- tempfile(pattern = "LUT_14aaA5_", tmpdir = tempdir(), fileext = "fa")
LUT.22aa.seq = ShortRead(DNAStringSet(LUT.22aa$Sequence), BStringSet(LUT.22aa$Names))
writeFasta(LUT.22aa.seq, LUT.22aa.fa)

```

Build Bowtie index

```

seqs.original <- readFasta("input/DNA-lib_RetrogradeTransport.fasta")

seqs.AA <- Biostrings::translate(sread(seqs.original), genetic.code=GENETIC_CODE,
                                   if.fuzzy.codon="error")

source("functions/AAtoDNA.R")
seqs.optimized = ShortRead(DNAStringSet(sapply(seqs.AA, function(x) AAtoDNA(x, species="hsa"))),
                           BStringSet(gsub("[ ]", "_", ShortRead::id(seqs.original)))))

bowtie.fasta <- tempfile(pattern = "bowtie_", tmpdir = tempdir(), fileext = ".fa")

writeFasta(seqs.optimized, bowtie.fasta)

bowtie.idx <- tempfile(pattern = "IDX_bowtie_", tmpdir = tempdir(), fileext = "")

```

```
sys.out <- system(paste("bowtie2-build", bowtie.fasta, bowtie.idx, "2>&1", sep = " "),  
                  intern = TRUE, ignore.stdout = FALSE)
```

Align fragments to reference

Align 14aa sequences

```
name.bowtie <- tempfile(pattern = "bowtie_", tmpdir = tempdir(), fileext = "")  
  
sys.out <- system(paste("bowtie2 --non-deterministic --threads ", detectCores(),  
                        " --local --score-min 'C,0,-1' -f -a",  
                        " -x ", bowtie.idx, " -U ", LUT.14aa.fa, " -S ",  
                        name.bowtie, ".sam 2>&1", sep = ""),  
                  intern = TRUE, ignore.stdout = FALSE)  
  
sys.out <- as.data.frame(sys.out)  
  
colnames(sys.out) <- c("Bowtie 2 alignment to library")  
invisible(sys.out[" "] <- "")  
lengthOut <- (nrow(sys.out))  
knitr::kable(sys.out[1:lengthOut], format = "markdown")
```

Bowtie 2 alignment to library

44708 reads; of these:

44708 (100.00%) were unpaired; of these:

0 (0.00%) aligned 0 times

34582 (77.35%) aligned exactly 1 time

10126 (22.65%) aligned >1 times

100.00% overall alignment rate

```
system(paste("samtools view -@ ", detectCores(), " -Sb ", name.bowtie, ".sam > ",  
            name.bowtie, ".bam", sep = ""))  
system(paste("samtools sort -@ ", detectCores(), " ", name.bowtie, ".bam ",  
            name.bowtie, "_sort", sep = ""))  
  
frag14aa.ranges <- readGAlignments(paste(name.bowtie, "_sort.bam", sep = ""), use.names=TRUE)  
length(names(frag14aa.ranges))
```

[1] 61314

```
length(unique(names(frag14aa.ranges)))
```

[1] 44708

```
length(unique(LUT.14aa$Sequence))
```

[1] 44708

Align 14aaG4S sequences

```
name.bowtie <- tempfile(pattern = "bowtie_", tmpdir = tempdir(), fileext = "")  
  
sys.out <- system(paste("bowtie2 --non-deterministic --threads ", detectCores(),  
                         " --local --score-min 'C,0,-1' -f -a",  
                         " -x ", bowtie.idx, " -U ", LUT.14aaG4S.fa, " -S ",  
                         name.bowtie, ".sam 2>&1", sep = ""),  
                         intern = TRUE, ignore.stdout = FALSE)  
  
sys.out <- as.data.frame(sys.out)  
  
colnames(sys.out) <- c("Bowtie 2 alignment to library")  
invisible(sys.out[" "] <- " ")  
lengthOut <- (nrow(sys.out))  
knitr::kable(sys.out[1:lengthOut,], format = "markdown")
```

Bowtie 2 alignment to library

15796 reads; of these:

15796 (100.00%) were unpaired; of these:

0 (0.00%) aligned 0 times

11498 (72.79%) aligned exactly 1 time

4298 (27.21%) aligned >1 times

100.00% overall alignment rate

```
system(paste("samtools view -@ ", detectCores(), " -Sb ", name.bowtie, ".sam > ",  
            name.bowtie, ".bam", sep = ""))  
system(paste("samtools sort -@ ", detectCores(), " ", name.bowtie, ".bam ",  
            name.bowtie, "_sort", sep = ""))  
  
frag14aaG4S.ranges <- readGAlignments(paste(name.bowtie, "_sort.bam", sep = ""), use.names=TRUE)  
length(names(frag14aaG4S.ranges))
```

[1] 22866

```
length(unique(names(frag14aaG4S.ranges)))
```

[1] 15796

```
length(unique(LUT.14aaG4S$Sequence))
```

[1] 15796

Align 14aaaA5 sequences

```
name.bowtie <- tempfile(pattern = "bowtie_", tmpdir = tempdir(), fileext = "")  
  
sys.out <- system(paste("bowtie2 --non-deterministic --threads ", detectCores(),  
                         " --local --score-min 'C,0,-1' -f -a",  
                         " -x ", bowtie.idx, " -U ", LUT.14aaA5.fa, " -S ",
```

```

        name.bowtie, ".sam 2>&1", sep = ""),
intern = TRUE, ignore.stdout = FALSE)

sys.out <- as.data.frame(sys.out)

colnames(sys.out) <- c("Bowtie 2 alignment to library")
invisible(sys.out[" "] <- " ")
lengthOut <- (nrow(sys.out))
knitr::kable(sys.out[1:lengthOut,], format = "markdown")

```

Bowtie 2 alignment to library
15796 reads; of these:
15796 (100.00%) were unpaired; of these:
0 (0.00%) aligned 0 times
11498 (72.79%) aligned exactly 1 time
4298 (27.21%) aligned >1 times
100.00% overall alignment rate

```

system(paste("samtools view -@ ", detectCores(), " -Sb ", name.bowtie, ".sam > ",
            name.bowtie, ".bam", sep = ""))
system(paste("samtools sort -@ ", detectCores(), " ", name.bowtie, ".bam ",
            name.bowtie, "_sort", sep = ""))
frag14aaA5.ranges <- readGAlignments(paste(name.bowtie, "_sort.bam", sep = ""), use.names=TRUE)
length(names(frag14aaA5.ranges))

```

[1] 22866

```
length(unique(names(frag14aaA5.ranges)))
```

[1] 15796

```
length(unique(LUT.14aaA5$Sequence))
```

[1] 15796

Align 22aa sequences

```

name.bowtie <- tempfile(pattern = "bowtie_", tmpdir = tempdir(), fileext = "")

sys.out <- system(paste("bowtie2 --non-deterministic --threads ", detectCores(),
                        " --local --score-min 'C,0,-1' -f -a",
                        " -x ", bowtie.idx, " -U ", LUT.22aa.fa, " -S ",
                        name.bowtie, ".sam 2>&1", sep = ""),
                        intern = TRUE, ignore.stdout = FALSE)

sys.out <- as.data.frame(sys.out)

colnames(sys.out) <- c("Bowtie 2 alignment to library")

```

```

invisible(sys.out[" "] <- " ")
lengthOut <- (nrow(sys.out))
knitr::kable(sys.out[1:lengthOut,], format = "markdown")

```

Bowtie 2 alignment to library

16058 reads; of these:

16058 (100.00%) were unpaired; of these:

0 (0.00%) aligned 0 times

12384 (77.12%) aligned exactly 1 time

3674 (22.88%) aligned >1 times

100.00% overall alignment rate

```

system(paste("samtools view -@ ", detectCores(), " -Sb ", name.bowtie, ".sam > ",
            name.bowtie, ".bam", sep = ""))
system(paste("samtools sort -@ ", detectCores(), " ", name.bowtie, ".bam ",
            name.bowtie, "_sort", sep = ""))

```

```

frag22aa.ranges <- readGAlignments(paste(name.bowtie, "_sort.bam", sep = ""), use.names=TRUE)
length(names(frag22aa.ranges))

```

[1] 21924

```
length(unique(names(frag22aa.ranges)))
```

[1] 16058

```
length(unique(LUT.22aa$Sequence))
```

[1] 16058

Merge and annotate aligned sequences

```

mcols(frag14aa.ranges)$structure <- "14aa"
mcols(frag22aa.ranges)$structure <- "22aa"
mcols(frag14aaA5.ranges)$structure <- "14aaaA5"
mcols(frag14aaG4S.ranges)$structure <- "14aaG4S"
allFragments.ranges <- append(frag14aa.ranges, frag22aa.ranges)
allFragments.ranges <- append(allFragments.ranges, frag14aaA5.ranges)
allFragments.ranges <- append(allFragments.ranges, frag14aaG4S.ranges)

save(allFragments.ranges, file="data/alignedLibraries.rda")

devtools::session_info()

```

Session info -----

```

setting  value
version  R version 3.2.3 (2015-12-10)
system   x86_64, linux-gnu
ui        X11
language en_US:en
collate  en_US.UTF-8
tz       <NA>
date     2016-01-13

```

Packages -----

package	*	version	date	source
acepack	*	1.3-3.3	2014-11-24	CRAN (R 3.2.0)
ade4	*	1.7-3	2015-11-22	CRAN (R 3.2.2)
Biobase	*	2.30.0	2015-11-26	Bioconductor
BiocGenerics	*	0.16.1	2015-11-26	Bioconductor
BiocParallel	*	1.4.0	2015-11-26	Bioconductor
Biostrings	*	2.38.2	2015-11-26	Bioconductor
bitops		1.0-6	2013-08-17	CRAN (R 3.2.0)
chron		2.3-47	2015-06-24	CRAN (R 3.2.2)
cluster		2.0.3	2015-07-21	CRAN (R 3.2.2)
colorspace		1.2-6	2015-03-11	CRAN (R 3.2.0)
data.table	*	1.9.6	2015-09-19	CRAN (R 3.2.2)
devtools	*	1.9.1	2015-09-11	CRAN (R 3.2.2)
digest		0.6.8	2014-12-31	CRAN (R 3.2.0)
evaluate		0.8	2015-09-18	CRAN (R 3.2.2)
foreign		0.8-66	2015-08-19	CRAN (R 3.2.2)
formatR		1.2.1	2015-09-18	CRAN (R 3.2.2)
Formula	*	1.2-1	2015-04-07	CRAN (R 3.2.0)
futile.logger		1.4.1	2015-04-20	CRAN (R 3.2.0)
futile.options		1.0.0	2010-04-06	CRAN (R 3.2.0)
GeneGA	*	1.20.0	2015-11-26	Bioconductor
GenomeInfoDb	*	1.6.1	2015-11-26	Bioconductor
GenomicAlignments	*	1.6.1	2015-11-26	Bioconductor
GenomicRanges	*	1.22.1	2015-11-26	Bioconductor
ggplot2	*	1.0.1	2015-03-17	CRAN (R 3.2.0)
gridExtra		2.0.0	2015-07-14	CRAN (R 3.2.2)
gttable		0.1.2	2012-12-05	CRAN (R 3.2.0)
hash	*	2.2.6	2013-02-21	CRAN (R 3.2.0)
highr		0.5.1	2015-09-18	CRAN (R 3.2.2)
Hmisc	*	3.17-0	2015-09-21	CRAN (R 3.2.2)
htmltools		0.2.6	2014-09-08	CRAN (R 3.2.0)
hwriter		1.3.2	2014-09-10	CRAN (R 3.2.0)
IRanges	*	2.4.4	2015-11-26	Bioconductor
knitr	*	1.11	2015-08-14	CRAN (R 3.2.2)
lambda.r		1.1.7	2015-03-20	CRAN (R 3.2.0)
lattice	*	0.20-33	2015-07-14	CRAN (R 3.2.2)
latticeExtra		0.6-26	2013-08-15	CRAN (R 3.2.0)
magrittr		1.5	2014-11-22	CRAN (R 3.2.2)
MASS		7.3-45	2015-11-10	CRAN (R 3.2.2)
memoise		0.2.1	2014-04-22	CRAN (R 3.2.2)
munsell		0.4.2	2013-07-11	CRAN (R 3.2.0)
nnet		7.3-11	2015-08-30	CRAN (R 3.2.2)
plyr		1.8.3	2015-06-12	CRAN (R 3.2.2)
proto		0.3-10	2012-12-22	CRAN (R 3.2.0)
RColorBrewer		1.1-2	2014-12-07	CRAN (R 3.2.0)
Rcpp		0.12.2	2015-11-15	CRAN (R 3.2.2)

reshape2	1.4.1	2014-12-06 CRAN (R 3.2.0)
rmarkdown	0.8.1	2015-10-10 CRAN (R 3.2.2)
rpart	4.1-10	2015-06-29 CRAN (R 3.2.2)
Rsamtools	* 1.22.0	2015-11-26 Bioconductor
S4Vectors	* 0.8.3	2015-11-26 Bioconductor
scales	0.3.0	2015-08-25 CRAN (R 3.2.2)
seqinr	* 3.1-3	2014-12-17 CRAN (R 3.2.0)
ShortRead	* 1.28.0	2015-11-26 Bioconductor
stringi	1.0-1	2015-10-22 CRAN (R 3.2.2)
stringr	1.0.0	2015-04-30 CRAN (R 3.2.2)
SummarizedExperiment	* 1.0.1	2015-11-26 Bioconductor
survival	* 2.38-3	2015-07-02 CRAN (R 3.2.2)
XVector	* 0.10.0	2015-11-26 Bioconductor
yaml	2.1.13	2014-06-12 CRAN (R 3.2.0)
zlibbioc	1.16.0	2015-11-26 Bioconductor

Barcoded plasmid library translation

Tomas Bjorklund

Wed Jan 13 10:44:38 2016

This workflow clusters every read from each unique barcode and determines the consensus fragmnt from the CustomArray and the barcode fidelity. i.e., if the barcode was monoclonal or not..

```
suppressPackageStartupMessages(library(knitr))
suppressPackageStartupMessages(library(ShortRead))
suppressPackageStartupMessages(library(parallel))
suppressPackageStartupMessages(library(doParallel))
suppressPackageStartupMessages(library(data.table))
suppressPackageStartupMessages(library(BSgenome))
suppressPackageStartupMessages(library(GenomicFeatures))
suppressPackageStartupMessages(library(GenomicAlignments))
suppressPackageStartupMessages(library(GenomicRanges))
suppressPackageStartupMessages(library(biovizBase))
suppressPackageStartupMessages(library(Gviz))
suppressPackageStartupMessages(library(plyr))
suppressPackageStartupMessages(library(devtools))
suppressPackageStartupMessages(library(Hmisc))
suppressPackageStartupMessages(library(matrixStats))
suppressPackageStartupMessages(library(stringdist))
suppressPackageStartupMessages(library(scales))

strt1<-Sys.time()

load("data/LUTdna.rda")

fragments.file <- "data/fragments_AAVlibrary_complete.fastq.gz"
barcodes.file <- "data/barcodes_AAVlibrary_complete.fastq.gz"

reads.trim <- readFastq(fragments.file)
reads.BC <- readFastq(barcodes.file)
```

Make CustomArray reference index for Blast

```
LUT.fa <- tempfile(pattern = "LUT_14aa_", tmpdir = tempdir(), fileext = ".fa")
LUT.seq = ShortRead(DNAStringSet(LUT.dna$Sequence), BStringSet(1:length(LUT.dna$Names)))
writeFasta(LUT.seq,LUT.fa)
```

Save unique fragments as fasta file

```
unique.reads <- unique(sread(reads.trim))
```

Select subset

```
#unique.reads <- unique.reads[sample(length(unique.reads), 50000)]  
  
unique.reads <- ShortRead(DNAStringSet(unique.reads), BStringSet(1:length(unique.reads)))  
fragments.unique.fa <- tempfile(pattern = "FragUnique_", tmpdir = tempdir(), fileext = ".fa")  
writeFasta(unique.reads, fragments.unique.fa)
```

Align against the library using blast

```
blast.db <- tempfile(pattern = "blastDB_", tmpdir = tempdir(), fileext = ".db")  
blast.out <- tempfile(pattern = "blastOut_", tmpdir = tempdir(), fileext = ".txt")  
  
sys.out <- system(paste("makeblastdb -in ", LUT.fa,  
                         " -out ",blast.db," -dbtype nucl -title LUT -parse_seqids 2>&1", sep = ""),  
                     intern = TRUE, ignore.stdout = FALSE)  
  
sys.out <- as.data.frame(sys.out)  
  
colnames(sys.out) <- c("blastn database generation")  
invisible(sys.out[" "] <- "")  
knitr::kable(sys.out[1:(nrow(sys.out)),], format = "markdown")
```

blastn database generation

Building a new DB, current time: 01/13/2016 10:46:12
New DB name: /tmp/RtmpoKLYWb/blastDB_1e425fd88164.db
New DB title: LUT
Sequence type: Nucleotide
Keep Linkouts: T
Keep MBits: T
Maximum file size: 1000000000B
Adding sequences from FASTA; added 92358 sequences in 2.99395 seconds.

```
sys.out <- system(paste("export SHELL=/bin/sh; cat ",fragments.unique.fa," | parallel --block ",  
                      floor(length(unique.reads)/detectCores()),  
                      " --recstart '>' --pipe blastn -max_target_seqs 10 -word_size 7",  
                      " -num_threads 1 -outfmt 10 -db ", blast.db,  
                      " -query - > ", blast.out, " 2>&1", sep = ""),  
                     intern = TRUE, ignore.stdout = FALSE) # -word_size 7  
  
table.blastn <- data.table(read.table(blast.out, header = FALSE, skip = 0, sep="; ",  
                                         stringsAsFactors = FALSE, fill=FALSE) , keep.rownames=FALSE, key="V1")  
  
system(paste("mv", blast.out, "./data/blastOutput.csv", sep=" "))
```

```

if (length(grep("Warning",table.blastn$V1)) != 0) {
  warnings.out <- unique(table.blastn[grep("Warning",table.blastn$V1),])
  table.blastn <- table.blastn[-grep("Warning",table.blastn$V1),]
  setnames(warnings.out,"V1", c("blastn Warnings"))
  knitr::kable(warnings.out[1:(nrow(warnings.out)),], format = "markdown")
}

|blastn Warnings |:-----| |Warning: lcl|Query_1439 7994: Warning: Could not
calculate ungapped Karlin-Altschul parameters due to an invalid query sequence or its translation. Please verify the
query sequence(s) and/or filtering options |



```

Starcode based barcode reduction

```

out.name.BC.star <- tempfile(pattern = "BCsc_", tmpdir = tempdir(), fileext = ".txt")

system(paste("gunzip -c ",barcodes.file," | starcode -t ",detectCores()-1," --print-clusters -d",
  1," -r5 -q -o ", out.name.BC.star, " 2>&1", sep = ""),
  intern = TRUE, ignore.stdout = FALSE)

## character(0)

table.BC.sc <- data.table(read.table(out.name.BC.star, header = FALSE, row.names = 1, skip = 0, sep="\t",
  stringsAsFactors = FALSE, fill=FALSE),keep.rownames=TRUE, key="rn") #

```

```

table.BC.sc[,V2 := NULL]

table.BC.sc <- table.BC.sc[, strsplit(as.character(V3), ", ", fixed=TRUE), by=rn]

SC.droppedBC <- length(unique(sread(reads.BC))) - length(unique(table.BC.sc$V1)) %in% unique(sread(reads.BC))
print(paste("Dropped BCs in Starcode:", SC.droppedBC))

## [1] "Dropped BCs in Starcode: 16603"

#rm(reads.BC, reads.trim)

setnames(table.BC.sc, c("V1", "rn"), c("BC", "scBC"))

```

Replacing barcodes with Starcode reduced versions

```

setkey(full.table,BC)
setkey(table.BC.sc,BC)
full.table <- full.table[table.BC.sc, nomatch=0]
#full.table <- merge(full.table, table.BC.sc, by="BC", all = FALSE, all.x = FALSE)
#rm(table.BC.sc)

setnames(full.table, c("BC", "scBC"), c("oldBC", "BC"))

setkey(full.table,BC)

RetainedBC <- length(unique(full.table$oldBC))
scBC <- length(unique(full.table$BC))
print(paste("Original unique barcodes:", RetainedBC))

## [1] "Original unique barcodes: 3576777"

print(paste("SC reduced unique barcodes:", scBC))

## [1] "SC reduced unique barcodes: 3185961"

table.frag <- data.table(as.data.frame((rev(sort(table(full.table$oldBC))))[1:10]), keep.rownames=TRUE)
setnames(table.frag, colnames(table.frag), c("Original BC", "Count"))
knitr::kable(table.frag, format = "markdown")

```

Original BC	Count
GTGTCCATCTGGACGCGTAG	165
CATGCATGGATGACTTATGT	137
GATGGTATATATGCGGATGG	133
GTTTAAAGCAATATGCACAC	118
AATCGCTGGATGGTTATGG	118
GCGCAATCGTGGAACGCAC	108
GTACATTGCTGTGAGCCTGC	101

Original BC	Count
GTGCCTTATTGGCGGCATT	99
GCGGGCGGGTGGAATGGCGT	99
GCGTATGTGCAGGCTCATTG	97

```
table.frag <- data.table(as.data.frame((rev(sort(table(full.table$BC))))[1:10]), keep.rownames=TRUE)
setnames(table.frag, colnames(table.frag), c("SC reduced BC", "Count"))
knitr::kable(table.frag, format = "markdown")
```

SC reduced BC	Count
GTGTCCATCTGGACGCGTAG	173
GATGGTATATATGCGGATGG	151
CATGCATGGATGACTTATGT	151
AATCGCTGGATGGTTATGG	125
GTAAAGCAATATGCACAC	122
GCGCAATCGTGGAACGCAC	119
GCGGGCGGGTGGAATGGCGT	117
GTACATTGCTGTGAGCCTGC	110
GCGTATGTGCAGGCTCATTG	105
CTGCGCACGCTTGTACGCCG	103

```
invisible(full.table[,oldBC:=NULL])
```

Splitting reads into single-read and multi-read barcodes

```
full.table <- full.table[order(full.table$BC),]
full.table[,mismatches:= as.numeric(mismatches)]

temp.table.single <- full.table[full.table[, .I[N == 1], by="BC"]$V1]
temp.table.multi <- full.table[full.table[, .I[N > 1], by="BC"]$V1]

temp.table.single[,c("mCount","tCount"):=1]
temp.table.single$Mode <- "Amb"
setkeyv(temp.table.multi,c("BC","LUTnr"))

temp.table.multi[,c("bitScore","mismatches" , "tCount"):= list(mean(bitScore),
                                                       median(mismatches), .N),
                 by=key(temp.table.multi)]

temp.table.multi$Mode <- "Def"
temp.table.multi <- unique(temp.table.multi)

print("Utilized reads.....")
```

```

## [1] "Utilized reads....."

print(nrow(full.table))

## [1] 10638907

print("Whereof single reads.....")

## [1] "Whereof single reads....."

print(nrow(temp.table.single))

## [1] 1351591

```

Splitting multi-read barcodes into clean and chimeric

```

setkeyv(temp.table.multi, "BC")

temp.table.multi.clean <- temp.table.multi[temp.table.multi[, .I[N == 1], by="BC"]$V1]
temp.table.multi <- temp.table.multi[temp.table.multi[, .I[N > 1], by="BC"]$V1]

temp.table.multi.clean[,mCount:=tCount]

print("Clean multi-read barcodes.....")

## [1] "Clean multi-read barcodes....."

print(nrow(temp.table.multi.clean))

## [1] 454777

print("Chimeric multi-read barcodes.....")

## [1] "Chimeric multi-read barcodes....."

print(length(unique(temp.table.multi$BC)))

## [1] 1379593

```

Calculate consensus alignment of chimeric barcodes

```

setkey(temp.table.multi, "BC")
temp.table.multi[, "mCount":=tCount]
temp.table.multi[, "tCount":=sum(tCount), by="BC"]

setkey(temp.table.multi, "Reads")

```

```

temp.table.multi[,c("LUTnr","bitScore","mismatches"):=NULL]
setkey(table.blastn,"Reads")
temp.table.multi <- temp.table.multi[table.blastn, nomatch=0, allow.cartesian=TRUE]

setkeyv(temp.table.multi,c("BC","LUTnr"))
temp.table.multi[,c("bitScore","mismatches" , "mCount"):= list(max(bitScore),
                           median(mismatches),
                           sum(mCount)), by=key(temp.table.multi)]
temp.table.multi <- unique(temp.table.multi)

setkeyv(temp.table.multi,"BC")
temp.table.multi <- temp.table.multi[temp.table.multi[, .I[mCount == max(mCount)],
                                         by=key(temp.table.multi)]$V1]
# Select only rows with the highest mCount

temp.table.multi <- temp.table.multi[temp.table.multi[, .I[which.max(bitScore)],
                                         by=key(temp.table.multi)]$V1]
# Select only rows with the highest bitScore

temp.table.multi[temp.table.multi$mCount==1]$Mode <- "Amb"

print(paste("Number of barcodes with false mCount:",
           nrow(temp.table.multi[mCount > tCount])))

## [1] "Number of barcodes with false mCount: 87365"

temp.table.multi.consensus <- rbind(temp.table.multi, temp.table.multi.clean)

print(paste("Total number of definitive Barcodes:",
           length(grep("Def", temp.table.multi.consensus$Mode)))))

## [1] "Total number of definitive Barcodes: 1642247"

print(paste("Total number of ambiguous Barcodes:",
           length(grep("Amb", temp.table.multi.consensus$Mode))))

## [1] "Total number of ambiguous Barcodes: 192123"

print(paste("Total number of single-read Barcodes:",
           nrow(temp.table.single)))

## [1] "Total number of single-read Barcodes: 1351591"

output.Table <- rbind(temp.table.multi.consensus,temp.table.single)
save(output.Table, file="data/multipleContfragmentsComplete.rda")

print("Total analysis time:")

## [1] "Total analysis time:"

print(Sys.time()-strt1)

## Time difference of 2.187571 hours

```

```
devtools::session_info()
```

```
## Session info ----

## setting value
## version R version 3.2.3 (2015-12-10)
## system x86_64, linux-gnu
## ui X11
## language en_US:en
## collate en_US.UTF-8
## tz <NA>
## date 2016-01-13

## Packages ----

## package * version date source
## acepack 1.3-3.3 2014-11-24 CRAN (R 3.2.0)
## AnnotationDbi * 1.32.0 2015-11-26 Bioconductor
## Biobase * 2.30.0 2015-11-26 Bioconductor
## BiocGenerics * 0.16.1 2015-11-26 Bioconductor
## BiocParallel * 1.4.0 2015-11-26 Bioconductor
## biomaRt 2.26.1 2015-11-26 Bioconductor
## Biostrings * 2.38.2 2015-11-26 Bioconductor
## biovizBase * 1.18.0 2015-11-26 Bioconductor
## bitops 1.0-6 2013-08-17 CRAN (R 3.2.0)
## BSgenome * 1.38.0 2015-11-26 Bioconductor
## chron 2.3-47 2015-06-24 CRAN (R 3.2.2)
## cluster 2.0.3 2015-07-21 CRAN (R 3.2.2)
## codetools 0.2-14 2015-07-15 CRAN (R 3.2.2)
## colorspace 1.2-6 2015-03-11 CRAN (R 3.2.0)
## data.table * 1.9.6 2015-09-19 CRAN (R 3.2.2)
## DBI 0.3.1 2014-09-24 CRAN (R 3.2.0)
## devtools * 1.9.1 2015-09-11 CRAN (R 3.2.2)
## dichromat 2.0-0 2013-01-24 CRAN (R 3.2.0)
## digest 0.6.8 2014-12-31 CRAN (R 3.2.0)
## doParallel * 1.0.10 2015-10-14 CRAN (R 3.2.2)
## evaluate 0.8 2015-09-18 CRAN (R 3.2.2)
## foreach * 1.4.3 2015-10-13 CRAN (R 3.2.2)
## foreign 0.8-66 2015-08-19 CRAN (R 3.2.2)
## formatR 1.2.1 2015-09-18 CRAN (R 3.2.2)
## Formula * 1.2-1 2015-04-07 CRAN (R 3.2.0)
## futile.logger 1.4.1 2015-04-20 CRAN (R 3.2.0)
## futile.options 1.0.0 2010-04-06 CRAN (R 3.2.0)
## GenomeInfoDb * 1.6.1 2015-11-26 Bioconductor
## GenomicAlignments * 1.6.1 2015-11-26 Bioconductor
## GenomicFeatures * 1.22.5 2015-11-26 Bioconductor
## GenomicRanges * 1.22.1 2015-11-26 Bioconductor
## ggplot2 * 1.0.1 2015-03-17 CRAN (R 3.2.0)
## gridExtra 2.0.0 2015-07-14 CRAN (R 3.2.2)
## gtable 0.1.2 2012-12-05 CRAN (R 3.2.0)
## Gviz * 1.14.0 2015-11-26 Bioconductor
## highr 0.5.1 2015-09-18 CRAN (R 3.2.2)
## Hmisc * 3.17-0 2015-09-21 CRAN (R 3.2.2)
## htmltools 0.2.6 2014-09-08 CRAN (R 3.2.0)
## hwriter 1.3.2 2014-09-10 CRAN (R 3.2.0)
## IRanges * 2.4.4 2015-11-26 Bioconductor
```

```

## iterators      * 1.0.8    2015-10-13 CRAN (R 3.2.2)
## knitr         * 1.11     2015-08-14 CRAN (R 3.2.2)
## lambda.r      1.1.7     2015-03-20 CRAN (R 3.2.0)
## lattice        * 0.20-33   2015-07-14 CRAN (R 3.2.2)
## latticeExtra    0.6-26    2013-08-15 CRAN (R 3.2.0)
## magrittr       1.5       2014-11-22 CRAN (R 3.2.2)
## MASS           7.3-45    2015-11-10 CRAN (R 3.2.2)
## matrixStats    * 0.15.0   2015-10-27 CRAN (R 3.2.2)
## memoise        0.2.1     2014-04-22 CRAN (R 3.2.2)
## munsell        0.4.2     2013-07-11 CRAN (R 3.2.0)
## nnet            7.3-11    2015-08-30 CRAN (R 3.2.2)
## plyr           * 1.8.3     2015-06-12 CRAN (R 3.2.2)
## proto          0.3-10    2012-12-22 CRAN (R 3.2.0)
## RColorBrewer   1.1-2     2014-12-07 CRAN (R 3.2.0)
## Rcpp            0.12.2    2015-11-15 CRAN (R 3.2.2)
## RCurl           1.95-4.7   2015-06-30 CRAN (R 3.2.2)
## reshape2        1.4.1     2014-12-06 CRAN (R 3.2.0)
## rmarkdown       0.8.1     2015-10-10 CRAN (R 3.2.2)
## rpart          4.1-10    2015-06-29 CRAN (R 3.2.2)
## Rsamtools      * 1.22.0   2015-11-26 Bioconductor
## RSQLite         1.0.0     2014-10-25 CRAN (R 3.2.0)
## rtracklayer    * 1.30.1   2015-11-26 Bioconductor
## S4Vectors      * 0.8.3    2015-11-26 Bioconductor
## scales          * 0.3.0    2015-08-25 CRAN (R 3.2.2)
## ShortRead      * 1.28.0   2015-11-26 Bioconductor
## stringdist     * 0.9.4    2015-10-26 CRAN (R 3.2.2)
## stringi          1.0-1    2015-10-22 CRAN (R 3.2.2)
## stringr          1.0.0    2015-04-30 CRAN (R 3.2.2)
## SummarizedExperiment * 1.0.1    2015-11-26 Bioconductor
## survival        * 2.38-3   2015-07-02 CRAN (R 3.2.2)
## VariantAnnotation 1.16.3   2015-11-26 Bioconductor
## XML             3.98-1.3  2015-06-30 CRAN (R 3.2.2)
## XVector          * 0.10.0   2015-11-26 Bioconductor
## yaml            2.1.13    2014-06-12 CRAN (R 3.2.0)
## zlibbioc        1.16.0    2015-11-26 Bioconductor

```

Library analysis output

Tomas Bjorklund

Wed Jan 13 12:56:17 2016

This workflow brings together FastQ files containing barcodes and 5'/3' ends of a suitable insert and alignmen them using Bowtie2. It also includes starcode based false barcode reduction and a MapReduce based hierarchical clustering

```
suppressPackageStartupMessages(library(knitr))
suppressPackageStartupMessages(library(ShortRead))
suppressPackageStartupMessages(library(ggplot2))
suppressPackageStartupMessages(library(ggbio))
suppressPackageStartupMessages(library(beanplot))
suppressPackageStartupMessages(library(parallel))
suppressPackageStartupMessages(library(doParallel))
suppressPackageStartupMessages(library(data.table))
suppressPackageStartupMessages(library(scales)) #Gives the log2 ability to ggplot2
suppressPackageStartupMessages(library(formatR))
suppressPackageStartupMessages(library(BSgenome))
suppressPackageStartupMessages(library(Rsamtools))
suppressPackageStartupMessages(library(rtracklayer))
suppressPackageStartupMessages(library(GenomicFeatures))
suppressPackageStartupMessages(library(GenomicAlignments))
suppressPackageStartupMessages(library(GenomicRanges))
suppressPackageStartupMessages(library(biovizBase))
suppressPackageStartupMessages(library(Gviz))
suppressPackageStartupMessages(library(plyr))
suppressPackageStartupMessages(library(devtools))
strt <- Sys.time()
load("data/multipleContfragmentsComplete.rda")
load("data/alignedLibraries.rda")
load("data/LUTdna.rda")

load.list <- read.table("input/loadlist.txt", header = FALSE, skip = 0, sep="\t",
stringsAsFactors = FALSE, fill=TRUE)

dataDir <- "seqFiles/RNA"
colnames(load.list) <- c("Name", "BaseName", "GroupName")

log.table <- data.table(Name="Name",
                         Reads=NA,
                         Purity=NA,
                         BCs=NA,
                         SCdroppedBC=NA,
                         allBCs=NA,
                         scBCs=NA)

analyzeTissue <- function(indexNr) {
#indexNr <- 15

  name <- unlist(strsplit(load.list$BaseName[indexNr], "/"))
  name <- name[!is.na(name)]
  if (length(name)==2){
    in.files <- list.files(paste(gsub("(\\[\\])", "", dataDir),name[1],sep="/"),
                           pattern=paste(name[2],"*", sep=""), full.names=TRUE)
  } else {
```

```

in.files <- list.files(gsub("(\\[\\])", "", dataDir),
                       pattern=paste(name[1], "*", sep=""), full.names=TRUE)
}
in.files.P5 <- in.files[grep("R1", in.files)]
in.files.P7 <- in.files[grep("R2", in.files)]

in.name.P5 <- tempfile(pattern = "P5_", tmpdir = tempdir(), fileext = ".fastq.gz")
in.name.P7 <- tempfile(pattern = "P7_", tmpdir = tempdir(), fileext = ".fastq.gz")
system(paste("cat '", paste(as.character(in.files.P5), collapse="' '"), "' > ",
            in.name.P5, " 2>&1", sep = ""), intern = TRUE, ignore.stdout = FALSE)

system(paste("cat '", paste(as.character(in.files.P7), collapse="' '"), "' > ",
            in.name.P7, " 2>&1", sep = ""), intern = TRUE, ignore.stdout = FALSE)

log.table$Name <- load.list>Name[indexNr]
name.out <- log.table$Name

# Selection of real amplicons
# =====

out.name.P5 <- tempfile(pattern = "P5_", tmpdir = tempdir(), fileext = ".fastq.gz")
out.name.P7 <- tempfile(pattern = "P7_", tmpdir = tempdir(), fileext = ".fastq.gz")
command.args <- paste("-Xmx12g overwrite=true k=10 rcomp=f skipr1=t qhdist=0 maskmiddle=t ",
                      "hammingdistance=1 findbestmatch=t ordered=t threads=", detectCores(),
                      " in=", in.name.P5,
                      " in2=", in.name.P7,
                      " outm=", out.name.P5,
                      " outm2=", out.name.P7,
                      " fliteral=", "CGCCACAACATCGAGGACGGCAGCGTG", sep = "")

sys.out <- system2(path.expand("~/bbmap/bbduk2.sh"), args=command.args, stdout=TRUE, stderr=TRUE)
log.table$Purity <- strsplit(sys.out[grep("Contaminants", sys.out)], split = "\t")[[1]][2]

in.name.P5 <- out.name.P5
in.name.P7 <- out.name.P7

log.table$Reads <- as.integer(system(paste("gunzip -c ", shQuote(gsub("(\\[\\])", "", in.name.P5)),
                                         " | echo $((`wc -l`/4)) 2>&1", sep = ""), intern = TRUE,
                                         ignore.stdout = FALSE)) #Stores the read count utilized

# Extraction of barcodes
# =====

out.name.BC <- tempfile(pattern = "BC_", tmpdir = tempdir(), fileext = ".fastq.gz")

sys.out <- system(paste("~/bbmap/bbduk2.sh overwrite=true k=12 mink=12 hammingdistance=2 ",
                        "findbestmatch=t trd=t rcomp=f skipr2=t findbestmatch=f qhdist=0 ",
                        "minavgquality=0 ordered=t maxns=0 minlength=18 maxlength=22 threads=",
                        detectCores(), " in=", shQuote(in.name.P5), " out=", out.name.BC,
                        " lliteral=", "GGCCTAGCGGCCGCTTACTT", " rliteral=", "ATAACTTCGTATA",
                        " 2>&1", sep = ""), intern = TRUE, ignore.stdout = FALSE)

log.table$BCs <- strsplit(sys.out[grep("Result:", sys.out)], split = "\t")[[1]][2]

reads.BC <- readFastq(out.name.BC)
barcodeTable <- data.table(ID=as.character(ShortRead::id(reads.BC)),

```

```

BC=as.character(sread(reads.BC)), key="BC")

# Starcode based barcode reduction
# =====

out.name.BC.star <- tempfile(pattern = "BCsc_", tmpdir = tempdir(), fileext = ".txt")

system(paste("gunzip -c ",out.name.BC," | starcode -t ",detectCores()-1," --print-clusters -d",
           "1," -r5 -q -o ", out.name.BC.star, " 2>&1", sep = ""),
       intern = TRUE, ignore.stdout = FALSE)

table.BC.sc <- data.table(read.table(out.name.BC.star, header = FALSE, row.names = 1, skip = 0, sep="\t",
                                      stringsAsFactors = FALSE, fill=FALSE),keep.rownames=TRUE, key="rn")
table.BC.sc[,V2 := NULL]

table.BC.sc <- table.BC.sc[, strsplit(as.character(V3),",",fixed=TRUE), by=rn]

log.table$SCdroppedBC <- length(unique(sread(reads.BC))) - length(unique(table.BC.sc$V1)) %in% unique(sread(r

setnames(table.BC.sc,c("V1","rn"),c("BC","scBC"))

# Replacing barcodes with Starcode reduced versions
# =====

setkey(table.BC.sc,BC)

barcodeTable <- barcodeTable[table.BC.sc,nomatch=0]

setnames(barcodeTable,c("BC","scBC"),c("oldBC","BC"))

setkey(barcodeTable,BC)

log.table$allBCs <- length(unique(barcodeTable$oldBC))
log.table$scBCs <- length(unique(barcodeTable$BC))

invisible(barcodeTable[,oldBC:=NULL])
setkey(output.Table,"BC")

BCcount <- rev(sort(table(barcodeTable$BC)))

foundFrags <- output.Table[match(names(BCcount), output.Table$BC),]
foundFrags$RNACount <- as.integer(BCcount)
foundFrags <- na.omit(foundFrags)
foundFrags$fragment <- LUT.dna$Sequence[as.integer(foundFrags$LUTnr)]

matchRange <- function(idxFrag) {
  #idxFrag <- 23
  matchRanges <- which(names(allFragments.ranges) == foundFrags$fragment[idxFrag])
  return(cbind(matchRanges, idxFrag))
}

match.ranges.list <- mclapply(1:nrow(foundFrags), matchRange, mc.preschedule = TRUE,
                               mc.cores = detectCores())
match.ranges <- do.call(rbind, match.ranges.list)
foundFragments.ranges <- allFragments.ranges[match.ranges[,1]]
if (ncol(match.ranges) >= 2) {

```

```

foundFrgs <- foundFrgs[match.ranges[, "idxFrag"], ]
foundFrgs[, c("Reads", "fragment"):=NULL]
mcols(foundFrgs.ranges) <- c(mcols(foundFrgs.ranges),
                                foundFrgs[match.ranges[, "idxFrag"], ])
o = order(-mcols(foundFrgs.ranges)$RNAcount)
foundFrgs.ranges <- foundFrgs.ranges[o]
saveRDS(foundFrgs.ranges, file=paste("output/", "found.", name.out, ".rds", sep=""),
        compress = TRUE)
}
return(log.table)
}

```

Analysis summary

```

all.logs <- lapply(1:nrow(load.list), analyzeTissue)
all.logs <- rbindlist(all.logs, use.names=FALSE )
knitr::kable(all.logs, format = "markdown")

```

Name	Reads	Purity	BCs	SCdroppedBC	allBCs
RatNr18_4wks_1000x_Sc-12	1556292	3112584 reads (99.48%)	813311 reads (52.26%)	11	3586
RatNr25_4wks_Untreat_Mu-13	1968380	3936760 reads (99.28%)	277622 reads (14.10%)	3	1200
RatNr25_4wks_Untreat_Sc-14	2120478	4240956 reads (99.18%)	200004 reads (9.43%)	0	1305
RatNr2_4wks_100x_Ctx-1	1955085	3910170 reads (99.25%)	591146 reads (30.24%)	3	5607
RatNr2_4wks_100x_SN-2	1921383	3842766 reads (99.39%)	1210727 reads (63.01%)	10	18161
RatNr2_4wks_100x_Str-3	2101122	4202244 reads (100.00%)	1563614 reads (74.42%)	24	84742
RatNr2_4wks_100x_Th-4	2177483	4354966 reads (95.47%)	752026 reads (34.54%)	0	16068
RatNr5_4wks_100x_Mu-5	2035694	4071388 reads (99.50%)	403591 reads (19.83%)	1	6078
RatNr5_4wks_100x_Sc-6	2441451	4882902 reads (97.96%)	632745 reads (25.92%)	10	2174
RatNr13_4wks_1000x_Ctx-7	2070763	4141526 reads (99.61%)	183573 reads (8.86%)	1	1403
RatNr13_4wks_1000x_SN-8	1809233	3618466 reads (99.47%)	105370 reads (5.82%)	2	1696
RatNr13_4wks_1000x_Str-9	1693037	3386074 reads (99.32%)	1096589 reads (64.77%)	2	17281
RatNr13_4wks_1000x_Th-10	1954529	3909058 reads (99.61%)	1079929 reads (55.25%)	27	12266
RatNr18_4wks_1000x_Mu-11	2069527	4139054 reads (99.35%)	358745 reads (17.33%)	3	3191
RatNr7_100x_SN-1	1983137	3966274 reads (99.65%)	1614014 reads (81.39%)	41	24691
RatNr7_100x_Ctx-2	1994467	3988934 reads (99.60%)	1771085 reads (88.80%)	13	16108
RatNr7_100x_Str-3	2867972	5735944 reads (99.65%)	1382945 reads (48.22%)	12	30889
RatNr7_100x_Th-4	1596468	3192936 reads (99.62%)	991144 reads (62.08%)	15	60701
RatNr1_100x_SN-5	1611759	3223518 reads (99.67%)	1361550 reads (84.48%)	47	11471
RatNr1_100x_Ctx-6	1541657	3083314 reads (99.67%)	1203038 reads (78.04%)	9	11542
RatNr1_100x_Str-7	2359538	4719076 reads (99.67%)	1026288 reads (43.50%)	1	17976
RatNr1_100x_Th-8	1505088	3010176 reads (99.60%)	619506 reads (41.16%)	6	37192
RatNr8_100x_SN-9	2054931	4109862 reads (99.70%)	1215848 reads (59.17%)	80	22685
RatNr8_100x_Ctx-10	2102816	4205632 reads (99.69%)	1081123 reads (51.41%)	18	13376

Name	Reads	Purity	BCs	SCdroppedBC	allBCs
RatNr8_100x_Str-11	2105768	4211536 reads (99.69%)	856547 reads (40.68%)	9	29605
RatNr8_100x_Th-12	1623686	3247372 reads (99.64%)	612950 reads (37.75%)	9	45404
RatNr15_1000x_SN-13	1436268	2872536 reads (99.64%)	1435201 reads (99.93%)	17	4226
RatNr15_1000x_Ctx-14	1260242	2520484 reads (99.65%)	1062278 reads (84.29%)	12	6434
RatNr15_1000x_Str-15	1105966	2211932 reads (99.65%)	944716 reads (85.42%)	3	5642
RatNr15_1000x_Th-16	948187	1896374 reads (99.65%)	684628 reads (72.20%)	2	22301
RatNr21_1000x_SN-17	1115267	2230534 reads (99.75%)	1112638 reads (99.76%)	3	4489
RatNr21_1000x_Ctx-18	1201263	2402526 reads (99.73%)	788753 reads (65.66%)	2	3920
RatNr21_1000x_Str-19	1234915	2469830 reads (99.73%)	1063068 reads (86.08%)	0	7414
RatNr21_1000x_Th-20	1151549	2303098 reads (99.71%)	845867 reads (73.45%)	3	29847
RatNr19_1000x_Ctx-21	1743952	3487904 reads (99.61%)	1738647 reads (99.70%)	15	6568
RatNr19_1000x_Str-22	1738722	3477444 reads (99.68%)	1174178 reads (67.53%)	21	10753
RatNr19_1000x_Th-23	1645506	3291012 reads (99.67%)	1196277 reads (72.70%)	5	34666
RatNr20_1000x_Str-24	788088	1576176 reads (99.61%)	682035 reads (86.54%)	3	5098
RatNr20_1000x_Th-25	983263	1966526 reads (99.59%)	827836 reads (84.19%)	3	23426
Cells293Nr2_1000x_cDNA-26	1588150	3176300 reads (99.63%)	1016131 reads (63.98%)	11	7051
Cells293Nr3_100x_cDNA-27	1920390	3840780 reads (99.65%)	864987 reads (45.04%)	13	20920
primNeuronsNr6_1000x_cDNA-28	1143395	2286790 reads (99.56%)	552817 reads (48.35%)	4	6313
primNeuronsNr7_100x_cDNA-29	1652464	3304928 reads (99.66%)	942060 reads (57.01%)	13	24858
primNeuronsNr6_1000x_RNA-30	1739141	3478282 reads (99.60%)	780217 reads (44.86%)	9	6942

```
knitr::kable(all.logs, format = "markdown")
```

Name	Reads	Purity	BCs	SCdroppedBC	allBCs
RatNr18_4wks_1000x_Sc-12	1556292	3112584 reads (99.48%)	813311 reads (52.26%)	11	3586
RatNr25_4wks_Untreat_Mu-13	1968380	3936760 reads (99.28%)	277622 reads (14.10%)	3	1200
RatNr25_4wks_Untreat_Sc-14	2120478	4240956 reads (99.18%)	200004 reads (9.43%)	0	1305
RatNr2_4wks_100x_Ctx-1	1955085	3910170 reads (99.25%)	591146 reads (30.24%)	3	5607
RatNr2_4wks_100x_SN-2	1921383	3842766 reads (99.39%)	1210727 reads (63.01%)	10	18161
RatNr2_4wks_100x_Str-3	2101122	4202244 reads (100.00%)	1563614 reads (74.42%)	24	84742
RatNr2_4wks_100x_Th-4	2177483	4354966 reads (95.47%)	752026 reads (34.54%)	0	16068
RatNr5_4wks_100x_Mu-5	2035694	4071388 reads (99.50%)	403591 reads (19.83%)	1	6078
RatNr5_4wks_100x_Sc-6	2441451	4882902 reads (97.96%)	632745 reads (25.92%)	10	2174
RatNr13_4wks_1000x_Ctx-7	2070763	4141526 reads (99.61%)	183573 reads (8.86%)	1	1403
RatNr13_4wks_1000x_SN-8	1809233	3618466 reads (99.47%)	105370 reads (5.82%)	2	1696
RatNr13_4wks_1000x_Str-9	1693037	3386074 reads (99.32%)	1096589 reads (64.77%)	2	17281
RatNr13_4wks_1000x_Th-10	1954529	3909058 reads (99.61%)	1079929 reads (55.25%)	27	12266
RatNr18_4wks_1000x_Mu-11	2069527	4139054 reads (99.35%)	358745 reads (17.33%)	3	3191
RatNr7_100x_SN-1	1983137	3966274 reads (99.65%)	1614014 reads (81.39%)	41	24691

Name	Reads	Purity	BCs	SCdroppedBC	allBCs
RatNr7_100x_Ctx-2	1994467	3988934 reads (99.60%)	1771085 reads (88.80%)	13	16108
RatNr7_100x_Str-3	2867972	5735944 reads (99.65%)	1382945 reads (48.22%)	12	30889
RatNr7_100x_Th-4	1596468	3192936 reads (99.62%)	991144 reads (62.08%)	15	60701
RatNr1_100x_SN-5	1611759	3223518 reads (99.67%)	1361550 reads (84.48%)	47	11471
RatNr1_100x_Ctx-6	1541657	3083314 reads (99.67%)	1203038 reads (78.04%)	9	11542
RatNr1_100x_Str-7	2359538	4719076 reads (99.67%)	1026288 reads (43.50%)	1	17976
RatNr1_100x_Th-8	1505088	3010176 reads (99.60%)	619506 reads (41.16%)	6	37192
RatNr8_100x_SN-9	2054931	4109862 reads (99.70%)	1215848 reads (59.17%)	80	22685
RatNr8_100x_Ctx-10	2102816	4205632 reads (99.69%)	1081123 reads (51.41%)	18	13376
RatNr8_100x_Str-11	2105768	4211536 reads (99.69%)	856547 reads (40.68%)	9	29605
RatNr8_100x_Th-12	1623686	3247372 reads (99.64%)	612950 reads (37.75%)	9	45404
RatNr15_1000x_SN-13	1436268	2872536 reads (99.64%)	1435201 reads (99.93%)	17	4226
RatNr15_1000x_Ctx-14	1260242	2520484 reads (99.65%)	1062278 reads (84.29%)	12	6434
RatNr15_1000x_Str-15	1105966	2211932 reads (99.65%)	944716 reads (85.42%)	3	5642
RatNr15_1000x_Th-16	948187	1896374 reads (99.65%)	684628 reads (72.20%)	2	22301
RatNr21_1000x_SN-17	1115267	2230534 reads (99.75%)	1112638 reads (99.76%)	3	4489
RatNr21_1000x_Ctx-18	1201263	2402526 reads (99.73%)	788753 reads (65.66%)	2	3920
RatNr21_1000x_Str-19	1234915	2469830 reads (99.73%)	1063068 reads (86.08%)	0	7414
RatNr21_1000x_Th-20	1151549	2303098 reads (99.71%)	845867 reads (73.45%)	3	29847
RatNr19_1000x_Ctx-21	1743952	3487904 reads (99.61%)	1738647 reads (99.70%)	15	6568
RatNr19_1000x_Str-22	1738722	3477444 reads (99.68%)	1174178 reads (67.53%)	21	10753
RatNr19_1000x_Th-23	1645506	3291012 reads (99.67%)	1196277 reads (72.70%)	5	34666
RatNr20_1000x_Str-24	788088	1576176 reads (99.61%)	682035 reads (86.54%)	3	5098
RatNr20_1000x_Th-25	983263	1966526 reads (99.59%)	827836 reads (84.19%)	3	23426
Cells293Nr2_1000x_cDNA-26	1588150	3176300 reads (99.63%)	1016131 reads (63.98%)	11	7051
Cells293Nr3_100x_cDNA-27	1920390	3840780 reads (99.65%)	864987 reads (45.04%)	13	20920
primNeuronsNr6_1000x_cDNA-28	1143395	2286790 reads (99.56%)	552817 reads (48.35%)	4	6313
primNeuronsNr7_100x_cDNA-29	1652464	3304928 reads (99.66%)	942060 reads (57.01%)	13	24858
primNeuronsNr6_1000x_RNA-30	1739141	3478282 reads (99.60%)	780217 reads (44.86%)	9	6942

```
unlink(paste(tempdir(), "/*", sep = ""), recursive = FALSE, force = FALSE) #Cleanup of temp files
print("Total execution time:")
```

```
## [1] "Total execution time:"
```

```
print(Sys.time()-strt)
```

```
## Time difference of 18.12098 mins
```

```
devtools::session_info()
```

```
## Session info ----

## setting value
## version R version 3.2.3 (2015-12-10)
## system x86_64, linux-gnu
## ui X11
## language en_US:en
## collate en_US.UTF-8
## tz <NA>
## date 2016-01-13

## Packages ----

## package * version date source
## acepack 1.3-3.3 2014-11-24 CRAN (R 3.2.0)
## AnnotationDbi * 1.32.0 2015-11-26 Bioconductor
## beanplot * 1.2 2014-09-19 CRAN (R 3.2.0)
## Biobase * 2.30.0 2015-11-26 Bioconductor
## BiocGenerics * 0.16.1 2015-11-26 Bioconductor
## BiocInstaller 1.20.1 2015-11-26 Bioconductor
## BiocParallel * 1.4.0 2015-11-26 Bioconductor
## biomaRt 2.26.1 2015-11-26 Bioconductor
## Biostrings * 2.38.2 2015-11-26 Bioconductor
## biovizBase * 1.18.0 2015-11-26 Bioconductor
## bitops 1.0-6 2013-08-17 CRAN (R 3.2.0)
## BSgenome * 1.38.0 2015-11-26 Bioconductor
## chron 2.3-47 2015-06-24 CRAN (R 3.2.2)
## cluster 2.0.3 2015-07-21 CRAN (R 3.2.2)
## codetools 0.2-14 2015-07-15 CRAN (R 3.2.2)
## colorspace 1.2-6 2015-03-11 CRAN (R 3.2.0)
## data.table * 1.9.6 2015-09-19 CRAN (R 3.2.2)
## DBI 0.3.1 2014-09-24 CRAN (R 3.2.0)
## devtools * 1.9.1 2015-09-11 CRAN (R 3.2.2)
## dichromat 2.0-0 2013-01-24 CRAN (R 3.2.0)
## digest 0.6.8 2014-12-31 CRAN (R 3.2.0)
## doParallel * 1.0.10 2015-10-14 CRAN (R 3.2.2)
## evaluate 0.8 2015-09-18 CRAN (R 3.2.2)
## foreach * 1.4.3 2015-10-13 CRAN (R 3.2.2)
## foreign 0.8-66 2015-08-19 CRAN (R 3.2.2)
## formatR * 1.2.1 2015-09-18 CRAN (R 3.2.2)
## Formula 1.2-1 2015-04-07 CRAN (R 3.2.0)
## futile.logger 1.4.1 2015-04-20 CRAN (R 3.2.0)
## futile.options 1.0.0 2010-04-06 CRAN (R 3.2.0)
## GenomeInfoDb * 1.6.1 2015-11-26 Bioconductor
## GenomicAlignments * 1.6.1 2015-11-26 Bioconductor
## GenomicFeatures * 1.22.5 2015-11-26 Bioconductor
## GenomicRanges * 1.22.1 2015-11-26 Bioconductor
## GGally 0.5.0 2014-12-02 CRAN (R 3.2.0)
## ggbio * 1.18.1 2015-11-26 Bioconductor
## ggplot2 * 1.0.1 2015-03-17 CRAN (R 3.2.0)
## graph 1.48.0 2015-11-26 Bioconductor
## gridExtra 2.0.0 2015-07-14 CRAN (R 3.2.2)
## gtable 0.1.2 2012-12-05 CRAN (R 3.2.0)
## Gviz * 1.14.0 2015-11-26 Bioconductor
```

```

##  highr          0.5.1   2015-09-18 CRAN (R 3.2.2)
##  Hmisc          3.17-0  2015-09-21 CRAN (R 3.2.2)
##  htmltools       0.2.6   2014-09-08 CRAN (R 3.2.0)
##  hwriter         1.3.2   2014-09-10 CRAN (R 3.2.0)
##  IRanges         * 2.4.4   2015-11-26 Bioconductor
##  iterators       * 1.0.8   2015-10-13 CRAN (R 3.2.2)
##  knitr           * 1.11    2015-08-14 CRAN (R 3.2.2)
##  lambda.r        1.1.7   2015-03-20 CRAN (R 3.2.0)
##  lattice          0.20-33  2015-07-14 CRAN (R 3.2.2)
##  latticeExtra     0.6-26   2013-08-15 CRAN (R 3.2.0)
##  magrittr         1.5      2014-11-22 CRAN (R 3.2.2)
##  MASS             7.3-45   2015-11-10 CRAN (R 3.2.2)
##  matrixStats      0.15.0   2015-10-27 CRAN (R 3.2.2)
##  memoise          0.2.1    2014-04-22 CRAN (R 3.2.2)
##  munsell          0.4.2    2013-07-11 CRAN (R 3.2.0)
##  nnet              7.3-11   2015-08-30 CRAN (R 3.2.2)
##  OrganismDbi      1.12.0   2015-11-26 Bioconductor
##  plyr             * 1.8.3   2015-06-12 CRAN (R 3.2.2)
##  proto             0.3-10   2012-12-22 CRAN (R 3.2.0)
##  RBGL              1.46.0   2015-11-26 Bioconductor
##  RColorBrewer     1.1-2    2014-12-07 CRAN (R 3.2.0)
##  Rcpp              0.12.2   2015-11-15 CRAN (R 3.2.2)
##  RCurl             1.95-4.7 2015-06-30 CRAN (R 3.2.2)
##  reshape            0.8.5   2014-04-23 CRAN (R 3.2.0)
##  reshape2          1.4.1    2014-12-06 CRAN (R 3.2.0)
##  rmarkdown          0.8.1    2015-10-10 CRAN (R 3.2.2)
##  rpart             4.1-10   2015-06-29 CRAN (R 3.2.2)
##  Rsamtools         * 1.22.0   2015-11-26 Bioconductor
##  RSQLite            1.0.0    2014-10-25 CRAN (R 3.2.0)
##  rtracklayer       * 1.30.1   2015-11-26 Bioconductor
##  S4Vectors         * 0.8.3   2015-11-26 Bioconductor
##  scales             * 0.3.0   2015-08-25 CRAN (R 3.2.2)
##  ShortRead         * 1.28.0   2015-11-26 Bioconductor
##  stringi            1.0-1    2015-10-22 CRAN (R 3.2.2)
##  stringr            1.0.0    2015-04-30 CRAN (R 3.2.2)
##  SummarizedExperiment * 1.0.1   2015-11-26 Bioconductor
##  survival           2.38-3   2015-07-02 CRAN (R 3.2.2)
##  VariantAnnotation 1.16.3   2015-11-26 Bioconductor
##  XML                3.98-1.3 2015-06-30 CRAN (R 3.2.2)
##  XVector            * 0.10.0   2015-11-26 Bioconductor
##  yaml               2.1.13   2014-06-12 CRAN (R 3.2.0)
##  zlibbioc          1.16.0   2015-11-26 Bioconductor

```

Generate a complete library range object

Tomas Björklund

Wed Jan 13 13:14:49 2016

This workflow clusters every read from each unique barcode from the library and matches them to the relevant ranges.

```
suppressPackageStartupMessages(library(knitr))
suppressPackageStartupMessages(library(ShortRead))
suppressPackageStartupMessages(library(parallel))
suppressPackageStartupMessages(library(doParallel))
suppressPackageStartupMessages(library(data.table))
suppressPackageStartupMessages(library(devtools))

load("data/alignedLibraries.rda")
load("data/LUTdna.rda")
load("data/multipleContfragmentsComplete.rda")
output.Table$fragment <- LUT.dna$Sequence[as.integer(output.Table$LUTnr)]
setkey(output.Table,fragment)

range.idx <- data.table(fragment=names(allFragments.ranges),
                         idxFrag=1:length(allFragments.ranges), key="fragment")
output.Table <- output.Table[range.idx, nomatch=0, allow.cartesian=TRUE]

foundFragments.ranges <- allFragments.ranges[output.Table$idxFrag]
output.Table[,c("Reads","fragment","idxFrag"):=NULL]
mcols(foundFragments.ranges) <- c(mcols(foundFragments.ranges), output.Table)

saveRDS(foundFragments.ranges, file="output/completeLibraryRanges.rds")

devtools::session_info()

## Session info -----
## setting  value
## version  R version 3.2.3 (2015-12-10)
## system   x86_64, linux-gnu
## ui        X11
## language en_US:en
## collate  en_US.UTF-8
## tz       <NA>
## date     2016-01-13

## Packages -----
## package      * version date      source
## Biobase      * 2.30.0  2015-11-26 Bioconductor
## BiocGenerics * 0.16.1  2015-11-26 Bioconductor
## BiocParallel * 1.4.0   2015-11-26 Bioconductor
## Biostrings    * 2.38.2  2015-11-26 Bioconductor
## bitops        1.0-6   2013-08-17 CRAN (R 3.2.0)
## chron         2.3-47   2015-06-24 CRAN (R 3.2.2)
## codetools     0.2-14   2015-07-15 CRAN (R 3.2.2)
## data.table    * 1.9.6   2015-09-19 CRAN (R 3.2.2)
```

```
## devtools          * 1.9.1   2015-09-11 CRAN (R 3.2.2)
## digest            0.6.8   2014-12-31 CRAN (R 3.2.0)
## doParallel        * 1.0.10  2015-10-14 CRAN (R 3.2.2)
## evaluate           0.8     2015-09-18 CRAN (R 3.2.2)
## foreach            * 1.4.3   2015-10-13 CRAN (R 3.2.2)
## futile.logger      1.4.1   2015-04-20 CRAN (R 3.2.0)
## futile.options     1.0.0   2010-04-06 CRAN (R 3.2.0)
## GenomeInfoDb       * 1.6.1   2015-11-26 Bioconductor
## GenomicAlignments * 1.6.1   2015-11-26 Bioconductor
## GenomicRanges      * 1.22.1  2015-11-26 Bioconductor
## htmltools           0.2.6   2014-09-08 CRAN (R 3.2.0)
## hwriter             1.3.2   2014-09-10 CRAN (R 3.2.0)
## IRanges             * 2.4.4   2015-11-26 Bioconductor
## iterators           * 1.0.8   2015-10-13 CRAN (R 3.2.2)
## knitr               * 1.11    2015-08-14 CRAN (R 3.2.2)
## lambda.r            1.1.7   2015-03-20 CRAN (R 3.2.0)
## lattice              0.20-33 2015-07-14 CRAN (R 3.2.2)
## latticeExtra         0.6-26  2013-08-15 CRAN (R 3.2.0)
## magrittr             1.5     2014-11-22 CRAN (R 3.2.2)
## memoise              0.2.1   2014-04-22 CRAN (R 3.2.2)
## RColorBrewer          1.1-2   2014-12-07 CRAN (R 3.2.0)
## rmarkdown             0.8.1   2015-10-10 CRAN (R 3.2.2)
## Rsamtools            * 1.22.0  2015-11-26 Bioconductor
## S4Vectors            * 0.8.3   2015-11-26 Bioconductor
## ShortRead            * 1.28.0  2015-11-26 Bioconductor
## stringi               1.0-1   2015-10-22 CRAN (R 3.2.2)
## stringr               1.0.0   2015-04-30 CRAN (R 3.2.2)
## SummarizedExperiment * 1.0.1   2015-11-26 Bioconductor
## XVector               * 0.10.0  2015-11-26 Bioconductor
## yaml                  2.1.13  2014-06-12 CRAN (R 3.2.0)
## zlibbioc              1.16.0  2015-11-26 Bioconductor
```

Normalize Library counts

Tomas Bjorklund

Wed Jan 13 13:15:41 2016

This workflow normalizes read counts between samples.

```
suppressPackageStartupMessages(library(knitr))
suppressPackageStartupMessages(library(GenomicAlignments))
suppressPackageStartupMessages(library(data.table))
suppressPackageStartupMessages(library(scales))
suppressPackageStartupMessages(library(ShortRead))
suppressPackageStartupMessages(library(multicore))
suppressPackageStartupMessages(library(plyr))
suppressPackageStartupMessages(library(Hmisc))
```

Generate load list and grouping names

```
in.names.all <- list.files("output", pattern="*.rds", full.names=TRUE)
load.list <- read.table("input/loadlist.txt", header = FALSE, skip = 0, sep="\t",
                        stringsAsFactors = FALSE, fill=TRUE)
colnames(load.list) <- c("Name", "BaseName", "GroupName")
load.list <- rbind(load.list,c("completeLibraryRanges","","totalLib"))
load.list <- load.list[-grep("Untreat",load.list$Name),]

select.Cases <- c(unlist(sapply(load.list>Name, function(x) grep(x,in.names.all), simplify = TRUE)))

(in.names.all <- in.names.all[select.Cases])

## [1] "output/found.RatNr18_4wks_1000x_Sc-12.rds"
## [2] "output/found.RatNr2_4wks_100x_Ctx-1.rds"
## [3] "output/found.RatNr2_4wks_100x_SN-2.rds"
## [4] "output/found.RatNr2_4wks_100x_Str-3.rds"
## [5] "output/found.RatNr2_4wks_100x_Th-4.rds"
## [6] "output/found.RatNr5_4wks_100x_Mu-5.rds"
## [7] "output/found.RatNr5_4wks_100x_Sc-6.rds"
## [8] "output/found.RatNr13_4wks_1000x_Ctx-7.rds"
## [9] "output/found.RatNr13_4wks_1000x_SN-8.rds"
## [10] "output/found.RatNr13_4wks_1000x_Str-9.rds"
## [11] "output/found.RatNr13_4wks_1000x_Th-10.rds"
## [12] "output/found.RatNr18_4wks_1000x_Mu-11.rds"
## [13] "output/found.RatNr7_100x_SN-1.rds"
## [14] "output/found.RatNr7_100x_Ctx-2.rds"
## [15] "output/found.RatNr7_100x_Str-3.rds"
## [16] "output/found.RatNr7_100x_Th-4.rds"
## [17] "output/found.RatNr1_100x_SN-5.rds"
## [18] "output/found.RatNr1_100x_Ctx-6.rds"
## [19] "output/found.RatNr1_100x_Str-7.rds"
## [20] "output/found.RatNr1_100x_Th-8.rds"
## [21] "output/found.RatNr8_100x_SN-9.rds"
## [22] "output/found.RatNr8_100x_Ctx-10.rds"
## [23] "output/found.RatNr8_100x_Str-11.rds"
```

```

## [24] "output/found.RatNr8_100x_Th-12.rds"
## [25] "output/found.RatNr15_1000x_SN-13.rds"
## [26] "output/found.RatNr15_1000x_Ctx-14.rds"
## [27] "output/found.RatNr15_1000x_Str-15.rds"
## [28] "output/found.RatNr15_1000x_Th-16.rds"
## [29] "output/found.RatNr21_1000x_SN-17.rds"
## [30] "output/found.RatNr21_1000x_Ctx-18.rds"
## [31] "output/found.RatNr21_1000x_Str-19.rds"
## [32] "output/found.RatNr21_1000x_Th-20.rds"
## [33] "output/found.RatNr19_1000x_Ctx-21.rds"
## [34] "output/found.RatNr19_1000x_Str-22.rds"
## [35] "output/found.RatNr19_1000x_Th-23.rds"
## [36] "output/found.RatNr20_1000x_Str-24.rds"
## [37] "output/found.RatNr20_1000x_Th-25.rds"
## [38] "output/found.Cells293Nr2_1000x_cDNA-26.rds"
## [39] "output/found.Cells293Nr3_100x_cDNA-27.rds"
## [40] "output/found.primNeuronsNr6_1000x_cDNA-28.rds"
## [41] "output/found.primNeuronsNr7_100x_cDNA-29.rds"
## [42] "output/found.primNeuronsNr6_1000x_RNA-30.rds"
## [43] "output/completeLibraryRanges.rds"

grouping <- data.frame(Sample=gsub("-", "_", gsub("found.|(output/)|(.rds)", "", in.names.all)),
                        Group=load.list[match(names(select.Cases), load.list$Name), "GroupName"],
                        stringsAsFactors = FALSE)

```

Load the desired alignment files and annotating group

```

loadRDS <- function(in.name) {
  #in.name <- in.names.all[3]
  this.sample <- readRDS(in.name)
  this.name <- gsub("-", "_", gsub("found.|(output/)|(.rds)", "", in.name))
  this.group <- grouping[match(this.name, grouping$Sample), "GroupName"]

  if (this.name == "completeLibraryRanges"){
    mcols(this.sample) <- cbind(mcols(this.sample),
                                data.frame(RNACount=mcols(this.sample)$tCount,
                                           Sample = this.name, Group=this.group,
                                           stringsAsFactors = FALSE))
    #This is required, as the total library alignment does not have any RNA counts
  } else {
    mcols(this.sample) <- cbind(mcols(this.sample),
                                data.frame(Sample = this.name, Group=this.group,
                                           stringsAsFactors = FALSE))
  }
  return(this.sample)
}

out.range <- lapply(in.names.all, loadRDS)

```

Normalizing read counts to correct for variable read depth

```

readCounts <- lapply(out.range, function(x) sum(mcols(x)$RNACount))
maxCount <- max(unlist(readCounts))
readCounts <- lapply(readCounts, function(x) maxCount/x)

makeNormCount <- function(inIndex){
  #inIndex <- 38
  thisRange <- out.range[[inIndex]]
  mcols(thisRange)$RNACount <- mcols(thisRange)$RNACount*readCounts[[inIndex]]
  return(thisRange)
}

out.range <- lapply(1:length(readCounts), makeNormCount)

```

Flatten into one GAlignments object

```

out.range <- do.call(GAlignmentsList, unlist(out.range))
out.range <- cbind(unlist(out.range))[[1]]
mcols(out.range)$NormCount <- 1
out.range <- out.range[mcols(out.range)$Mode == "Def"] #Selects only defined i.e., trusted reads

```

Split the GAlignments into list based on group

```

out.range.split <- split(out.range, c(mcols(out.range)$Group))
out.range.split <- lapply(out.range.split, function(x) split(x, seqnames(x)))
out.range.split <- mclapply(out.range.split, function(x) lapply(x, function(y) split(y, names(y))),
                           mc.preschedule = TRUE, mc.cores = detectCores())

MergeCounts <- function(inRanges) {
  outRanges <- inRanges[1]
  mcols(outRanges) <- data.frame(structure=mcols(inRanges)$structure[1],
                                   Group=mcols(inRanges)$Group[1],
                                   bitScore=sum(mcols(inRanges)$bitScore*mcols(inRanges)$tCount)/sum(mcols(inRanges)$tCount),
                                   mismatches=median(mcols(inRanges)$mismatches),
                                   mCount=sum(mcols(inRanges)$mCount),
                                   tCount=sum(mcols(inRanges)$tCount),
                                   BC=paste(unique(mcols(inRanges)$BC), collapse = ","),
                                   Animals=paste(unique(mcols(inRanges)$Sample), collapse = ","),
                                   RNACount=sum(mcols(inRanges)$RNACount),
                                   NormCount=log2(sum(mcols(inRanges)$RNACount)+1)*length(inRanges),
                                   stringsAsFactors=FALSE)
  return(outRanges)
}

out.range.split <- lapply(out.range.split, function(x) mclapply(x, function(y) lapply(y, MergeCounts), mc.pre)

out.range.split <- mclapply(out.range.split, function(x) unlist(do.call(GAlignmentsList, unlist(x))), use.names=FALSE)
out.range.split <- unlist(do.call(GAlignmentsList, unlist(out.range.split)), use.names=FALSE)
saveRDS(out.range.split, file="data/normalizedSampleRangesDefined.RDS")

```

Pairwise sample analysis output

Tomas Bjorklund

Wed Jan 13 17:57:29 2016

This is the final script presenting top candidates and overview plots.

```
suppressPackageStartupMessages(library(knitr))
```

Generation of infective library

```
all.samples <- readRDS("data/normalizedSampleRangesDefined.RDS")
total.AAV.samples <- all.samples[!(mcols(all.samples)$Group %in% "totalLib")]
total.AAV.samples <- total.AAV.samples[-grep("4wks", mcols(total.AAV.samples)$Group)]
mcols(total.AAV.samples)$Group <- "infectiveLib"
all.samples <- append(all.samples, total.AAV.samples)
```

Plotting function

```
plotPair <- function(topSample, bottomSample, filterBC=FALSE, filterAnimal=FALSE,
                      AnimaladjustPlot=FALSE, NormalizePlot=TRUE) {
  # Select samples
  #=====

  #   topSample <- "CNS100x_Ctx"
  #   bottomSample <- "CNS100x_Str"
  #   filterBC <- FALSE
  #   filterAnimal <- FALSE
  #   AnimaladjustPlot <- FALSE
  #   NormalizePlot <- TRUE

  fill.values <- eval(parse(text=paste("c(", topSample, "= rgb(38,64,135, maxColorValue = 255), ",
                                       bottomSample, "= rgb(157,190,217, maxColorValue = 255)", ",sep=""))))

  select.samples <- all.samples[mcols(all.samples)$Group %in% names(fill.values)]
  trim.names <- data.table(GeneName=levels(seqnames(select.samples)))
  trim.names[, c("Category", "Protein", "Origin",
                "Extra", "Number", "GeneName") := tstrsplit(GeneName, " ", fixed=TRUE)]
  trim.names$GeneName <- gsub("/|_", "-", trim.names$GeneName)
  levels(seqnames(select.samples)) <- trim.names$GeneName

  geneTable <- data.frame(seqlengths(select.samples))
  colnames(geneTable) <- "SeqLength"
  geneTable$GeneName <- row.names(geneTable)

  plot.data <- data.frame(GeneName=seqnames(select.samples),
                           Group=factor(mcols(select.samples)$Group, levels = names(fill.values)),
                           APos=start(select.samples)+(width(select.samples)/2),
                           Animals=mcols(select.samples)$Animals,
                           BC=mcols(select.samples)$BC,
```

```

        RNAcount=mcols(select.samples)$RNAcount,
        NormCount=mcols(select.samples)$NormCount,
        stringsAsFactors = FALSE)
plot.data <- merge(plot.data,geneTable)
plot.data$AAproc <- (plot.data$AApos/plot.data$SeqLength)*100

#=====
#Binning of data
#=====

size.bin <- 2
FullLength <- 100
position <- seq(0,FullLength,size.bin)
plot.data.dt <- data.table(plot.data)
plot.data.dt[,bin:=findInterval(AAproc, position)]
plot.data.bin <- plot.data.dt[, list(.N,SeqLength=min(SeqLength),
                                         AAproc = position[findInterval(min(AAproc),position)],
                                         BCmean=unlist(lapply(strsplit(paste(BC, collapse=","), ","),function(x) as.numeric(unlist(strsplit(x,","))))),
                                         AnimalCount = length(table(strsplit(paste(t(Animals), collapse=","))),
                                         NormCount = log2((sum(RNAcount)/SeqLength*FullLength)+1)
                                         ), by=c("bin","GeneName","Group"))]

#=====
#Filtration parameters
#=====

if (NormalizePlot) {
  plot.data.bin[plot.data.bin$Group == names(fill.values)[1]]$NormCount <- plot.data.bin[plot.data.bin$Group == names(fill.values)[1]]$NormCount + 1
  plot.data.bin[plot.data.bin$Group == names(fill.values)[2]]$NormCount <- plot.data.bin[plot.data.bin$Group == names(fill.values)[2]]$NormCount + 1
}

if (filterBC) {
  plot.data.bin <- plot.data.bin[plot.data.bin$BCmean > 1,]
  select.samples <- select.samples[mcols(select.samples)$BC>1]
}

if (filterAnimal) {
  plot.data.bin <- plot.data.bin[plot.data.bin$AnimalCount > 1,]
}

if (AnimaladjustPlot) {
  plot.data.bin$NormCount <- plot.data.bin$NormCount*plot.data.bin$AnimalCount/plot.data.bin$SeqLength*FullLength
}

plot.data.bin[plot.data.bin$Group == names(fill.values)[2]]$NormCount <- plot.data.bin[plot.data.bin$Group == names(fill.values)[2]]$NormCount + 1

#=====
# Sort and select top samples
#=====

select.samples.gr <- granges(select.samples)

mcols(select.samples.gr) <- cbind(mcols(select.samples)[,c(1,2,4,5,6,7,10)], 
                                    DataFrame(Animals=unlist(lapply(strsplit(mcols(select.samples)$Animals, " "),""))
                                             )
                                    )

o = order(-mcols(select.samples.gr)$NormCount)

```

```

select.samples.gr <- select.samples.gr[o]
top.sample <- select.samples.gr[mcols(select.samples.gr)$Group %in% names(fill.values)][1]
bottom.sample <- select.samples.gr[mcols(select.samples.gr)$Group %in% names(fill.values)][2]

if (length(top.sample) >=1){
  out <- top.sample
  out.2 <- data.table(Fragment=names(out),
                        SeqCount=0,
                        Gene=as.character(seqnames(out)),
                        startAA=(start(out)+2)/3)
  out.2 <- data.frame(out.2,as.data.frame(mcols(out)))
  out.2$BC <- unlist(lapply(strsplit(out.2$BC, ","),function(x) length(unique(x))))
  out.2[match(names(table(out.2$Fragment)),out.2$Fragment),"SeqCount"] <- as.integer(table(out.2$Fragment))
  out.2 <- out.2[out.2$SeqCount>=1]
  setnames(out.2, "Fragment", paste(topSample,"- Fragment"))
  top.sample.out.1 <- out.2[1:min(15,nrow(out.2)),c(1,3)]
  top.sample.out.2 <- out.2[1:min(15,nrow(out.2)),c(-1,-2,-6)]
} else {top.sample.out.1 <- top.sample.out.2 <- NA}

if (length(bottom.sample) >=1) {
  out <- bottom.sample
  out.2 <- data.table(Fragment=names(out),
                        SeqCount=0,
                        Gene=as.character(seqnames(out)),
                        startAA=(start(out)+2)/3)
  out.2 <- data.frame(out.2,as.data.frame(mcols(out)))
  out.2$BC <- unlist(lapply(strsplit(out.2$BC, ","),function(x) length(unique(x))))
  out.2[match(names(table(out.2$Fragment)),out.2$Fragment),"SeqCount"] <- as.integer(table(out.2$Fragment))
  out.2 <- out.2[out.2$SeqCount>=1]
  setnames(out.2, "Fragment", paste(bottomSample,"- Fragment"))
  bottom.sample.out.1 <- out.2[1:min(15,nrow(out.2)),c(1,3,6)]
  bottom.sample.out.2 <- out.2[1:min(15,nrow(out.2)),c(-1,-2,-6)]
} else {bottom.sample.out.1 <- bottom.sample.out.2 <- NA}

#=====
#Output plot
#=====

plot.out <- ggplot(plot.data.bin,aes(x=AProc,y=NormCount, fill = Group)) +
  geom_bar(stat="identity", position="identity") + theme_bw() +
  scale_fill_manual(name = "Library", values = fill.values) +
  scale_colour_manual(name = "Library", values = fill.values) +
  scale_x_continuous(limit=c(0,100), breaks=c(seq(0,100,20)), expand =c(0,0)) +
  facet_wrap(~ GeneName, ncol=5) +
  theme(plot.margin=unit(x=c(0,0,0,0),units="mm"),
        legend.position="bottom",
        legend.margin=unit(-0.5,"cm"),
        legend.key.height=unit(0, "cm"),
        plot.background=element_rect(fill="white"),
        axis.text = element_text(size = rel(0.5)),
        axis.ticks = element_line(size = rel(0.5)),
        axis.ticks.length = unit(.05, "cm"),
        strip.text.x = element_text(size = rel(0.5), colour = "black",
                                    angle = 0, lineheight=0.1, vjust=0.1),
        strip.background = element_blank(),
        strip.background = element_rect(size = 0),
        panel.margin.y = unit(-0.15, "cm"),

```

```
panel.margin.x = unit(0, "cm"))

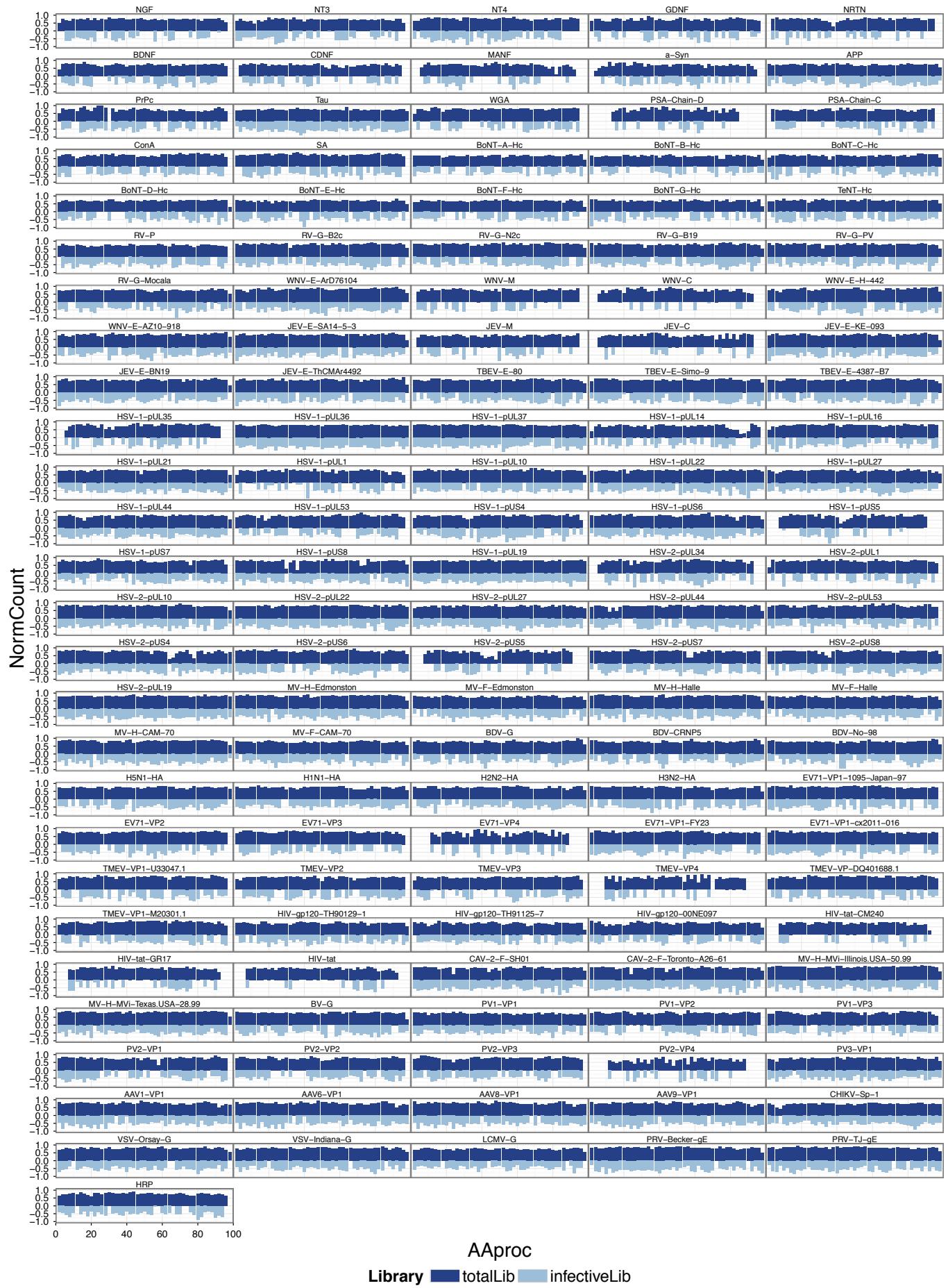
out.list <- list(plot=plot.out,
                 plotBin=plot.data.bin,
                 top1=top.sample.out.1,
                 top2=top.sample.out.2,
                 bottom1=bottom.sample.out.1,
                 bottom2=bottom.sample.out.2)

return(out.list)
}
```

Analyze samples

```
out.plot.list <- plotPair("totalLib","infectiveLib",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=FALSE,
                           NormalizePlot=TRUE)

out.plot.list$plot
```



```
knitr::kable(out.plot.list$top2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	HSV-1-pUL36	2882	14aaA5	0.5	6351	3712	26	8941.211	1
2	HSV-2-pUL53	214	22aa	1.0	3033	3155	573	6660.484	1
3	JEV-E-ThCMAr4492	475	22aa	0.0	3191	2842	550	6310.259	1
4	JEV-E-ThCMAr4492	478	22aa	0.0	2611	2298	428	4779.386	1
5	HSV-1-pUS8	415	22aa	1.0	2104	2037	382	4199.302	1
6	HSV-1-pUL36	2881	14aaG4S	0.0	1650	1650	13	4168.759	1
7	WNV-E-AZ10-918	151	22aa	0.0	1886	1943	379	4140.504	1
8	GDNF	58	22aa	0.0	2042	1998	370	4057.073	1
9	HSV-1-pUS8	412	22aa	2.0	2271	2009	361	3961.246	1
10	HSV-1-pUL37	376	22aa	1.0	2424	1883	352	3829.613	1
11	PrPc	34	22aa	0.5	2043	1907	349	3803.348	1
12	HSV-2-pUS4	217	22aa	0.0	1846	1804	337	3645.593	1
13	HSV-1-pUL22	520	22aa	0.5	1788	1820	331	3584.901	1
14	TMEV-VP1-U33047.1	238	22aa	1.0	1585	1647	321	3430.367	1
16	HSV-1-pUL36	1519	22aa	0.0	1688	1731	307	3302.775	1

```
knitr::kable(out.plot.list$bottom2, format = "markdown")
```

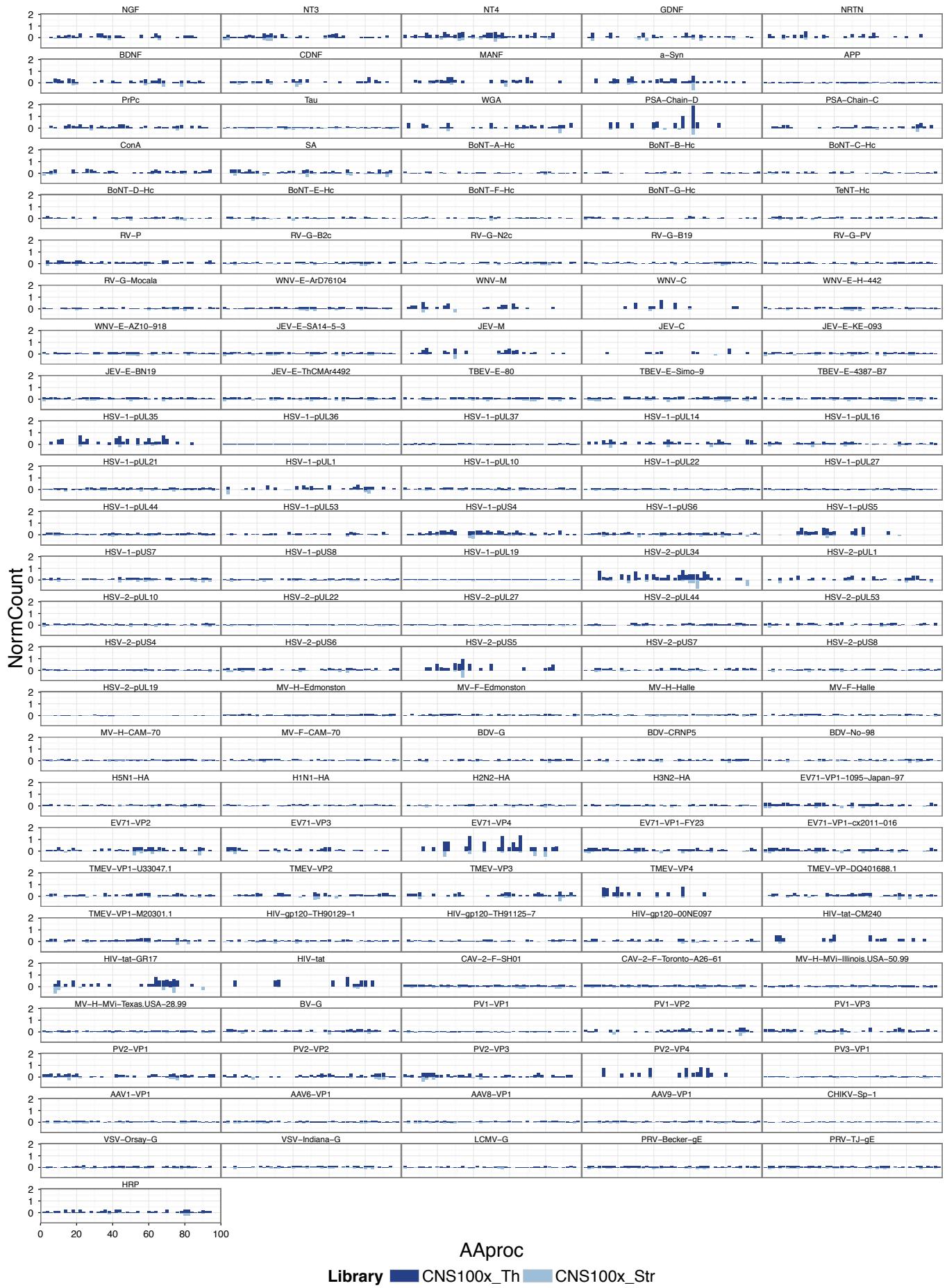
	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	HSV-2-pUL10	425	14aa	0.00	36	54	12	169.1715	3
2	HSV-1-pUL36	1801	14aa	0.00	49	52	11	162.3208	3
3	PV2-VP1	54	14aa	0.00	47	69	11	155.1726	3
5	Tau	251	14aa	0.00	38	59	11	153.8395	3
6	BV-G	302	14aa	0.00	37	63	8	146.9449	4
7	HSV-1-pUL14	100	14aa	0.25	48	81	10	141.3028	3
8	HSV-1-pUL36	2680	14aa	0.00	48	62	10	139.9025	3
9	WNV-M	16	14aa	0.00	48	72	10	137.8492	3
10	HSV-1-pUL36	1519	22aa	0.00	49	74	9	122.9765	3
11	HSV-1-pUL36	2595	14aa	1.00	44	53	9	122.5534	3
12	HSV-1-pUL36	2596	14aa	0.50	44	62	8	111.3385	3
13	HSV-1-pUS8	29	14aa	0.00	45	53	8	110.9723	3
14	HSV-1-pUL36	770	14aa	0.00	34	47	5	110.3266	4
15	PSA-Chain-D	26	14aa	0.25	41	49	8	109.4910	3
16	HSV-1-pUL19	417	14aa	0.25	55	81	8	108.7193	3

```
out.plot.list <- plotPair("CNS100x_Th","CNS100x_Str",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=FALSE,
                           NormalizePlot=TRUE)
```

```
out.plot.list$plot
```



```
out.plot.list <- plotPair("CNS100x_Th","CNS100x_Str",
                         filterBC=FALSE,
                         filterAnimal=FALSE,
                         AnimaladjustPlot=TRUE,
                         NormalizePlot=TRUE)
out.plot.list$plot
```



AAproc

Library CNS100x_Th CNS100x_Str

```
knitr::kable(out.plot.list$plotBin[1:10], format = "markdown")
```

bin	GeneName	Group	N	SeqLength	AAproc	BCmean	AnimalCount	NormCount
13	AAV1-VP1	CNS100x_Th	9	2208	24	13	3	0.1001386
13	AAV1-VP1	CNS100x_Th	9	2208	24	13	3	0.1001386
13	AAV1-VP1	CNS100x_Th	9	2208	24	13	3	0.1001386
13	AAV1-VP1	CNS100x_Th	9	2208	24	13	3	0.1001386
13	AAV1-VP1	CNS100x_Th	9	2208	24	13	3	0.1001386
13	AAV1-VP1	CNS100x_Th	9	2208	24	13	3	0.1001386
13	AAV1-VP1	CNS100x_Th	9	2208	24	13	3	0.1001386
13	AAV1-VP1	CNS100x_Th	9	2208	24	13	3	0.1001386
13	AAV1-VP1	CNS100x_Th	9	2208	24	13	3	0.1001386
44	AAV1-VP1	CNS100x_Th	8	2208	86	10	3	0.0970851

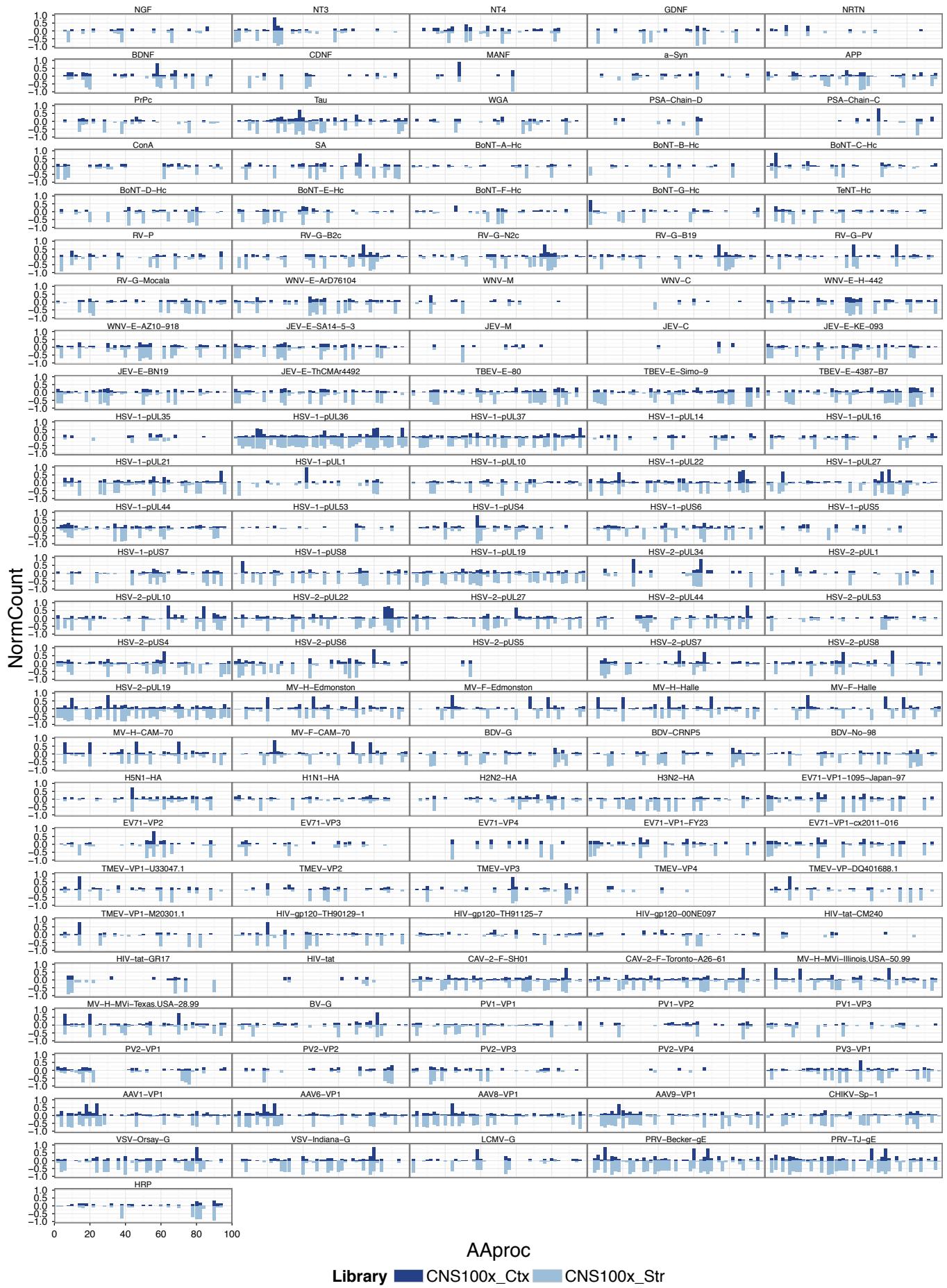
```
knitr::kable(out.plot.list$top2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	HSV-2-pUL10	425	14aa	0.00	36	54	12	169.1715	3
2	HSV-1-pUL36	1801	14aa	0.00	49	52	11	162.3208	3
3	PV2-VP1	54	14aa	0.00	47	69	11	155.1726	3
5	Tau	251	14aa	0.00	38	59	11	153.8395	3
6	HSV-1-pUL14	100	14aa	0.25	48	81	10	141.3028	3
7	HSV-1-pUL36	2680	14aa	0.00	48	62	10	139.9025	3
8	WNV-M	16	14aa	0.00	48	72	10	137.8492	3
9	HSV-1-pUL36	1519	22aa	0.00	49	74	9	122.9765	3
10	HSV-1-pUL36	2595	14aa	1.00	44	53	9	122.5534	3
11	HSV-1-pUL36	2596	14aa	0.50	44	62	8	111.3385	3
12	HSV-1-pUS8	29	14aa	0.00	45	53	8	110.9723	3
13	PSA-Chain-D	26	14aa	0.25	41	49	8	109.4910	3
14	HSV-1-pUL19	417	14aa	0.25	55	81	8	108.7193	3
16	HSV-2-pUL19	755	14aa	0.00	37	48	8	107.7478	3
17	HSV-1-pUL36	1059	14aa	0.00	36	44	8	107.4727	3

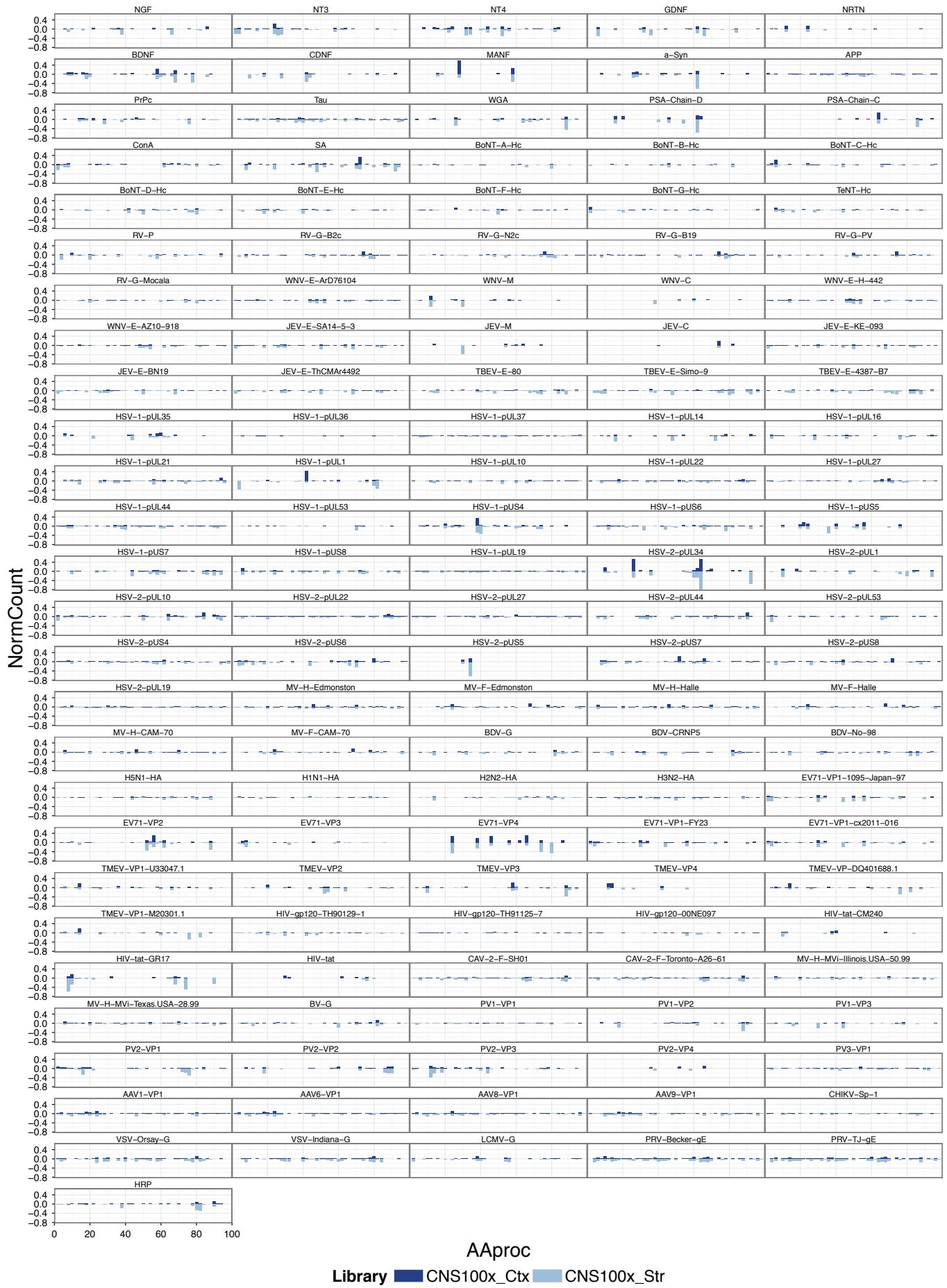
```
knitr::kable(out.plot.list$bottom2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	HSV-1-pUL36	1519	22aa	0.0	26	25	6	94.41806	3
2	PV1-VP2	236	14aa	0.0	13	17	5	83.54495	3
3	HSV-2-pUL27	393	14aa	1.0	16	21	5	80.69311	3
4	Tau	241	14aaG4S	0.5	16	25	4	68.21062	3
5	HSV-1-pUL21	75	14aa	0.5	20	35	4	64.98599	3
6	HSV-2-pUS6	275	14aa	1.0	31	44	4	64.03911	3
7	HSV-2-pUL19	419	14aa	0.0	19	29	4	63.95275	3
8	HRP	250	14aaG4S	0.0	15	22	4	59.97958	3
9	HSV-1-pUL36	2552	14aa	0.5	33	45	4	59.55352	2
10	HSV-2-pUS8	235	14aaA5	0.0	15	18	4	59.51165	3
11	VSV-Orsay-G	387	14aaA5	0.0	24	32	4	55.92369	3
13	TMEV-VP2	133	22aa	0.0	15	20	4	54.28445	2
14	HSV-1-pUL37	920	14aa	0.0	8	15	3	52.57127	2
15	HSV-1-pUL36	2521	22aa	0.0	13	17	3	51.46529	3
16	VSV-Orsay-G	192	14aa	0.0	11	18	3	51.34442	2

```
out.plot.list <- plotPair("CNS100x_Ctx", "CNS100x_Str",
                         filterBC=FALSE,
                         filterAnimal=FALSE,
                         AnimaladjustPlot=FALSE,
                         NormalizePlot=TRUE)
out.plot.list$plot
```



```
out.plot.list <- plotPair("CNS100x_Ctx", "CNS100x_Str",
                         filterBC=FALSE,
                         filterAnimal=FALSE,
                         AnimaladjustPlot=TRUE,
                         NormalizePlot=TRUE)
out.plot.list$plot
```



```
knitr::kable(out.plot.list$top2, format = "markdown")
```

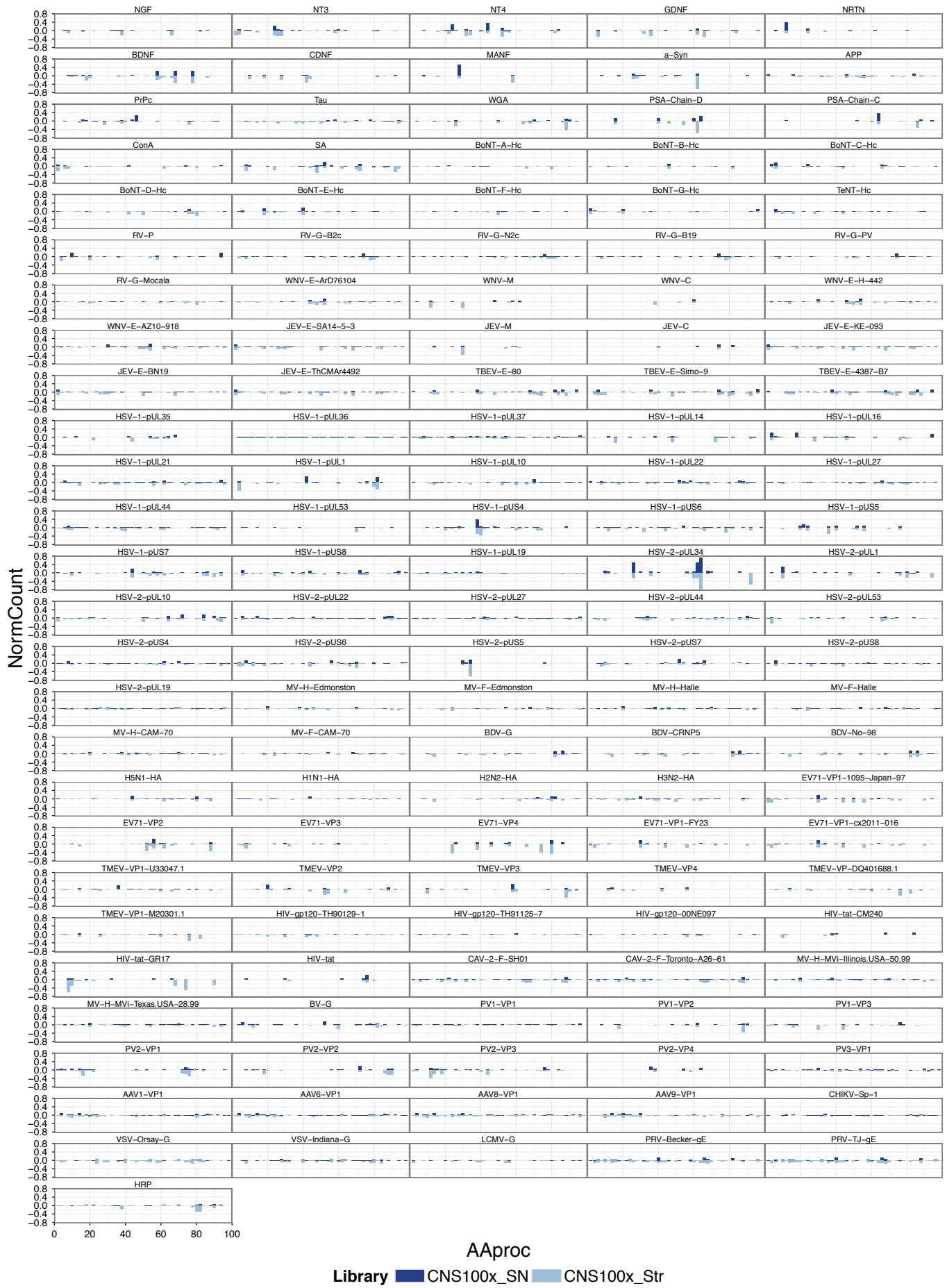
	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	HSV-2-pUL22	743	14aa	0.0	15	36	5	95.79975	3
2	BV-G	318	14aa	0.0	25	36	5	89.79143	2
3	HSV-2-pUS7	194	14aa	1.5	31	47	4	72.82153	3
4	HSV-1-pUL22	160	14aaaA5	0.5	19	30	4	68.82554	3
5	HSV-2-pUL19	419	14aa	0.0	7	7	3	63.54107	2
6	HSV-1-pUL1	88	14aaaA5	0.0	14	17	3	63.13008	3
7	HSV-1-pUL27	628	14aaaA5	1.0	15	22	3	60.86283	3
8	Tau	480	14aa	2.0	26	38	5	58.50914	2
9	HSV-2-pUS8	391	14aa	0.0	12	11	3	57.97992	3
10	BoNT-C-Hc	21	14aa	2.0	28	45	3	57.95954	3
11	MV-F-Edmonston	378	14aa	0.0	26	21	3	55.34030	3
14	HSV-1-pUL36	2601	14aa	0.0	10	14	3	54.69235	2
15	HSV-1-pUL21	499	14aa	0.0	8	9	3	54.41724	3
16	HSV-1-pUL27	608	14aa	0.0	16	22	3	54.32717	3
17	SA	177	14aa	0.0	11	13	3	54.07652	3

```
out.plot.list <- plotPair("CNS100x_SN","CNS100x_Str",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=FALSE,
                           NormalizePlot=TRUE)
```

```
out.plot.list$plot
```



```
out.plot.list <- plotPair("CNS100x_SN","CNS100x_Str",
                         filterBC=FALSE,
                         filterAnimal=FALSE,
                         AnimaladjustPlot=TRUE,
                         NormalizePlot=TRUE)
out.plot.list$plot
```



AAproc

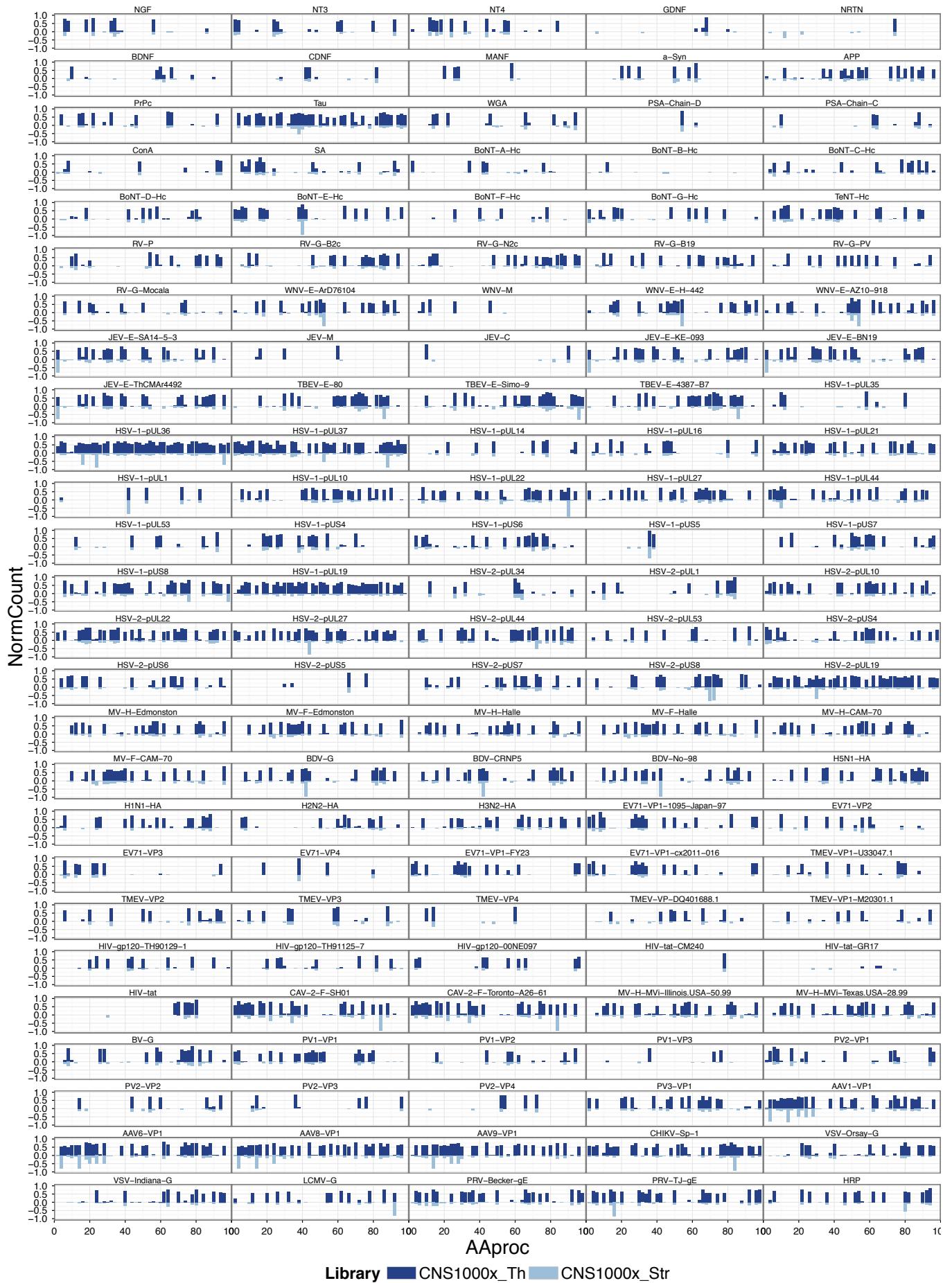
Library CNS100x_SN CNS100x_Str

```
knitr::kable(out.plot.list$top2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	CHIKV-Sp-1	431	14aa	2.50	12	21	4	73.79785	2
2	HSV-2-pUS8	28	14aaaA5	0.00	17	41	4	70.89160	3
3	HSV-1-pUS8	394	14aaaA5	0.00	27	28	4	67.18122	3
4	HSV-2-pUS4	55	14aa	0.25	11	25	4	66.23569	3
5	HSV-2-pUL22	743	14aa	0.50	20	33	4	64.01023	3
6	HSV-2-pUS7	194	14aa	0.00	15	22	4	62.29724	3
7	BDNF	128	14aa	0.50	25	32	4	61.72544	2
8	BoNT-G-Hc	10	14aa	0.00	9	15	3	57.07397	2
9	H5N1-HA	244	14aaaA5	3.00	10	12	3	55.34925	3
10	HSV-1-pUS4	88	14aa	0.00	10	11	3	53.72732	3
11	PRV-Becker-gE	228	14aaaA5	0.50	18	28	3	53.72688	3
13	HSV-1-pUS4	88	14aaaG4S	0.50	13	21	3	52.73960	3
14	HSV-2-pUL10	418	22aa	1.00	35	28	3	50.90180	2
15	BoNT-E-Hc	184	14aa	2.00	16	25	3	50.19565	3
16	HSV-2-pUL10	393	14aa	1.00	13	20	3	49.96192	3

```
out.plot.list <- plotPair("CNS1000x_Th","CNS1000x_Str",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=FALSE,
                           NormalizePlot=TRUE)
```

```
out.plot.list$plot
```



```
knitr::kable(out.plot.list$top2, format = "markdown")
```

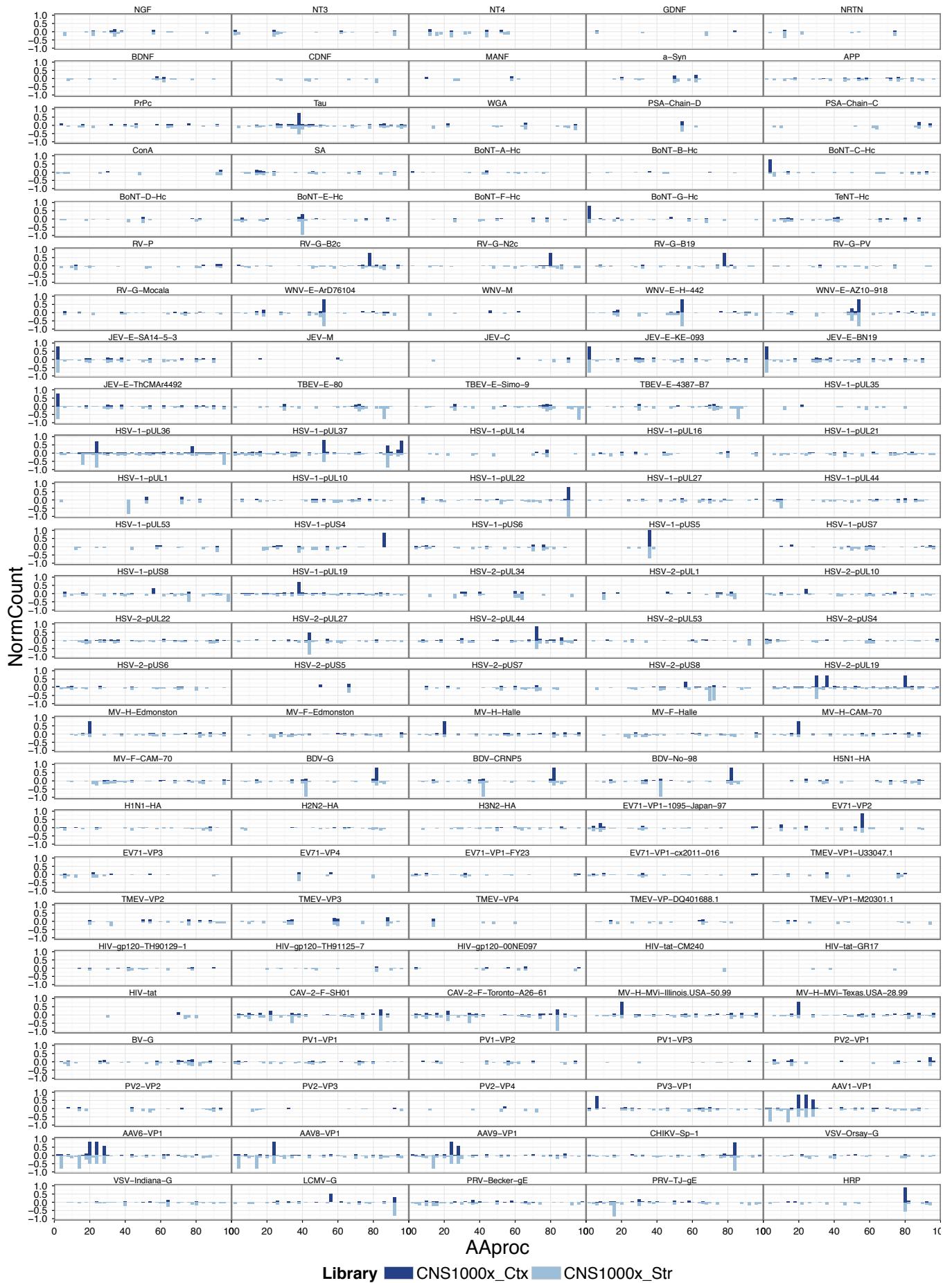
	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	BV-G	302	14aa	0.0	37	63	8	146.94491	4
2	BDV-G	409	14aaaA5	0.0	15	20	6	99.55255	4
4	AAV8-VP1	515	14aa	0.5	39	37	6	97.49597	4
5	EV71-VP1-1095-Japan-97	20	14aa	0.0	28	39	5	89.75568	4
6	TMEV-VP3	136	14aa	0.0	12	22	5	82.66370	4
7	H3N2-HA	367	14aa	1.0	22	38	5	82.05983	4
8	HSV-2-pUL22	302	14aa	0.5	21	28	5	81.86378	3
9	TBEV-E-80	147	14aa	1.0	20	25	5	81.10393	4
12	HIV-gp120-TH91125-7	367	22aa	0.0	27	35	5	80.29551	4
13	TMEV-VP1-U33047.1	207	14aa	0.0	18	37	5	79.60364	3
14	AAV1-VP1	144	14aa	0.5	15	29	5	78.98733	4
16	HSV-1-pUL10	212	14aa	0.0	25	28	5	77.23914	3
17	HSV-1-pUL37	139	22aa	0.0	18	21	5	75.93768	4
18	HSV-2-pUL1	185	14aa	0.5	13	19	4	73.55901	4
19	WNV-E-AZ10-918	250	14aa	0.0	15	31	4	71.44031	4

```
knitr::kable(out.plot.list$bottom2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	HSV-1-pUL36	770	14aa	0.0	34	47	5	110.32660	4
2	HSV-1-pUL22	756	14aa	0.0	16	29	4	91.64609	4
3	CHIKV-Sp-1	1051	14aaaA5	0.0	16	27	4	88.16866	4
4	BoNT-E-Hc	184	14aa	0.0	14	26	4	84.10735	4
5	BV-G	302	14aa	1.0	48	61	7	58.12730	3
6	HSV-1-pUS5	28	14aa	0.0	22	28	4	55.36772	4
7	AAV1-VP1	174	14aa	0.0	12	24	4	55.23572	4
11	Tau	268	14aa	0.0	8	13	4	55.12702	4
12	HRP	243	14aa	0.0	21	28	4	55.05112	3
13	HSV-2-pUL19	419	14aa	1.0	21	28	3	54.98727	2
14	AAV1-VP1	144	14aa	0.5	16	36	4	54.50202	3
16	BDV-G	409	14aaaA5	0.0	30	58	6	54.28026	4
18	HSV-1-pUL37	580	22aa	0.0	16	14	3	44.16113	3
19	HSV-2-pUS4	402	14aa	0.0	28	38	6	42.37686	4
20	BDV-G	210	14aa	0.5	10	22	2	42.03579	2

```
out.plot.list <- plotPair("CNS1000x_Ctx", "CNS1000x_Str",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=FALSE,
                           NormalizePlot=TRUE)
```

```
out.plot.list$plot
```



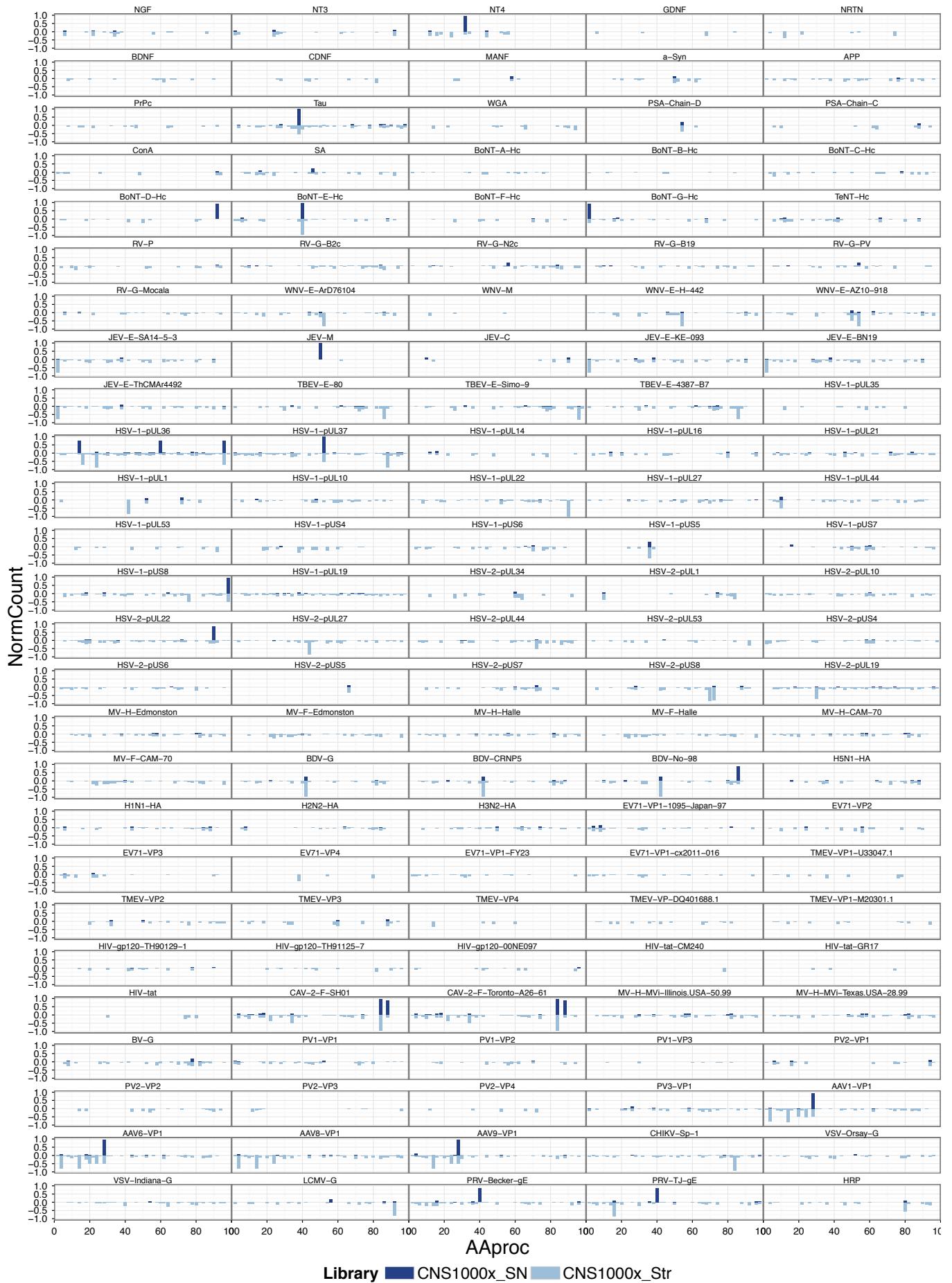
Library █ CNS1000x_Ctx █ CNS1000x_Str

```
knitr::kable(out.plot.list$top2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	HSV-1-pUS5	28	14aa	0.00	8	18	2	40.44128	2
2	AAV1-VP1	174	14aa	0.00	7	12	2	40.44117	2
6	HRP	243	14aa	0.00	5	8	2	40.44117	2
7	AAV1-VP1	144	14aa	0.50	8	16	2	40.09822	2
9	HSV-1-pUL22	756	14aa	0.50	11	17	2	39.27394	2
10	CHIKV-Sp-1	1051	14aaaA5	0.25	4	4	2	39.12554	2
11	WNV-E-ArD76104	264	14aaG4S	1.25	8	13	2	37.48570	2
14	PV3-VP1	52	14aa	1.00	11	17	2	37.48566	2
15	BoNT-G-Hc	10	14aa	0.00	8	14	2	37.48564	2
16	Tau	268	14aa	0.00	7	10	2	36.70942	2
17	AAV1-VP1	205	14aaaA5	0.00	4	8	2	30.17197	2
20	HSV-2-pUL27	393	14aa	0.00	7	10	2	26.65307	2
21	BV-G	302	14aa	0.50	39	38	4	24.85594	2
22	CAV-2-F-SH01	459	14aa	0.50	6	8	2	20.96141	2
24	HSV-2-pUL44	348	14aa	0.00	2	5	1	20.22056	1

```
out.plot.list <- plotPair("CNS1000x_SN","CNS1000x_Str",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=FALSE,
                           NormalizePlot=TRUE)
```

```
out.plot.list$plot
```



```
knitr::kable(out.plot.list$top2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	HSV-1-pUL37	580	22aa	0.0	4	4	1	44.30619	2
2	BoNT-E-Hc	184	14aa	0.5	7	16	2	40.04651	2
3	Tau	268	14aa	0.0	2	3	1	21.14548	1
4	HSV-1-pUS8	535	14aa	0.0	2	2	1	20.51469	1
5	CAV-2-F-SH01	459	14aa	0.0	2	2	1	20.51469	1
7	AAV1-VP1	205	14aaaA5	0.0	2	2	1	20.51469	1
10	NT4	52	22aa	1.5	14	11	1	19.02290	1
11	BoNT-D-Hc	376	22aa	1.0	3	4	1	19.02290	1
12	BoNT-G-Hc	10	14aa	0.0	2	2	1	19.02290	1
13	JEV-M	79	14aa	0.0	2	3	1	19.02290	1
14	HSV-1-pUL36	494	14aa	0.0	5	7	1	19.02290	1
15	HSV-1-pUL36	3041	14aa	0.0	4	5	1	19.02290	1
16	HSV-1-pUL36	1886	14aa	0.0	6	8	1	19.02290	1
17	HSV-2-pUL22	748	14aaaA5	0.0	6	11	1	19.02290	1
18	BDV-No-98	427	14aaaG4S	0.0	10	11	1	19.02290	1

```
out.plot.list <- plotPair("CNS1000x_Str","CNS100x_Str",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=FALSE,
                           NormalizePlot=TRUE)

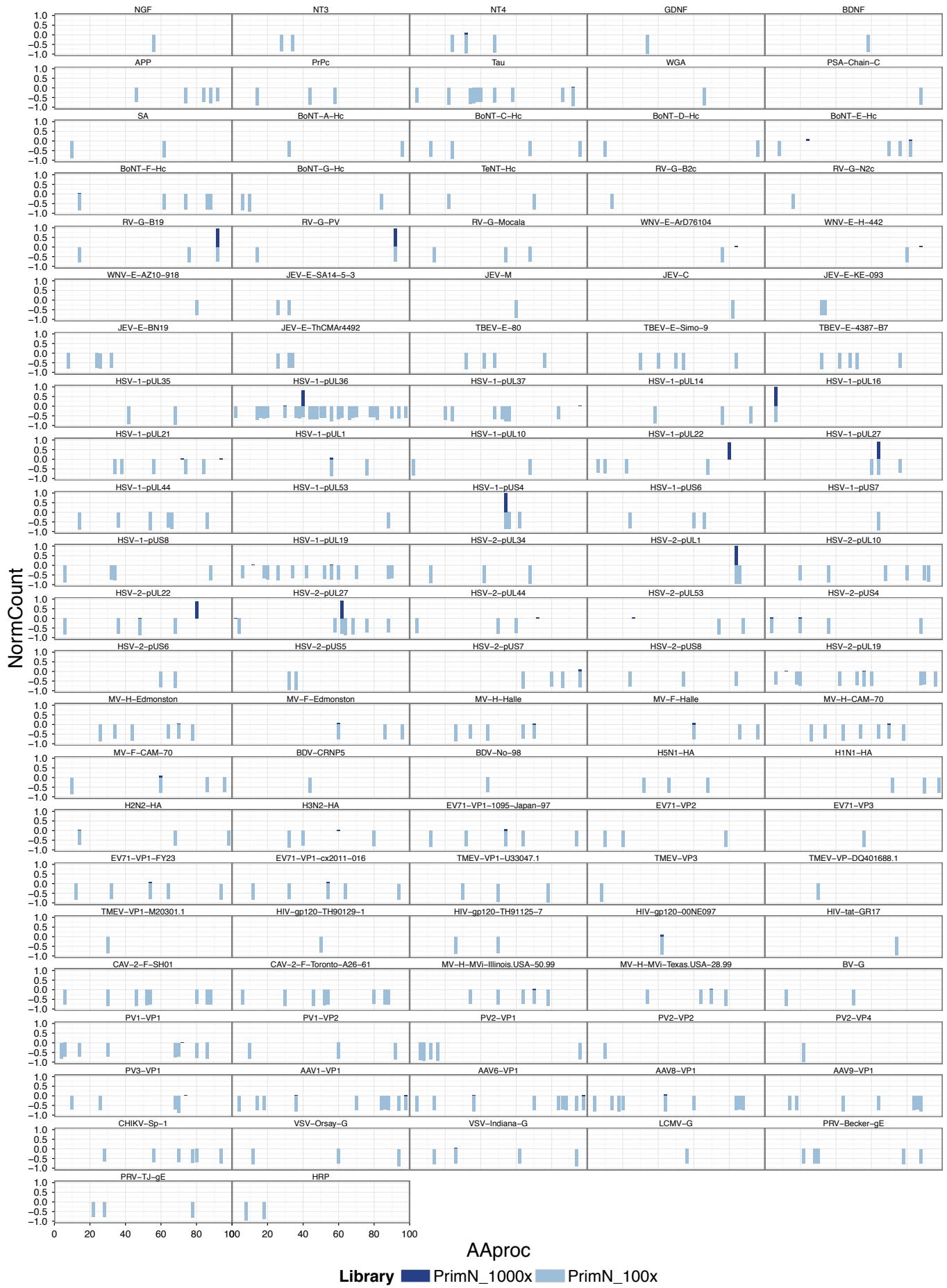
out.plot.list <- plotPair("CNS1000x_Th","CNS100x_Th",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=FALSE,
                           NormalizePlot=TRUE)

out.plot.list <- plotPair("CNS1000x_Ctx","CNS100x_Ctx",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=TRUE,
                           NormalizePlot=TRUE)

out.plot.list <- plotPair("CNS1000x_SN","CNS100x_SN",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=FALSE,
                           NormalizePlot=TRUE)

out.plot.list <- plotPair("PrimN_1000x","PrimN_100x",
                           filterBC=FALSE,
                           filterAnimal=FALSE,
                           AnimaladjustPlot=FALSE,
                           NormalizePlot=TRUE)

out.plot.list$plot
```



```
knitr::kable(out.plot.list$top2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	HSV-1-pUL36	1286	14aa	0.0	5	7	1	20.861401	1
2	HSV-1-pUL16	13	22aa	3.5	10	14	1	20.854037	1
3	HSV-1-pUL27	575	14aa	3.5	10	14	1	20.854037	1
5	RV-G-B19	463	22aa	0.0	2	3	1	20.733697	1
7	HSV-1-pUS4	123	14aa	0.0	2	3	1	19.925512	1
8	HSV-2-pUL1	185	14aa	0.0	2	3	1	19.925512	1
9	HSV-1-pUL22	670	14aa	0.0	6	11	1	19.814446	1
11	BoNT-E-Hc	111	14aa	0.0	2	2	1	4.814156	1
12	HSV-2-pUS7	355	14aa	1.5	6	7	1	4.814156	1
13	MV-F-Edmonston	335	14aa	0.0	2	2	1	4.814156	1
16	HIV-gp120-00NE097	153	14aa	3.0	4	9	1	4.814156	1
17	AAV8-VP1	323	14aa	0.0	6	7	1	4.814156	1
18	NT4	52	22aa	0.0	2	2	1	3.864548	1
19	Tau	630	14aa	0.0	3	4	1	3.864548	1
20	BoNT-E-Hc	385	14aaaA5	0.0	5	8	1	3.864548	1

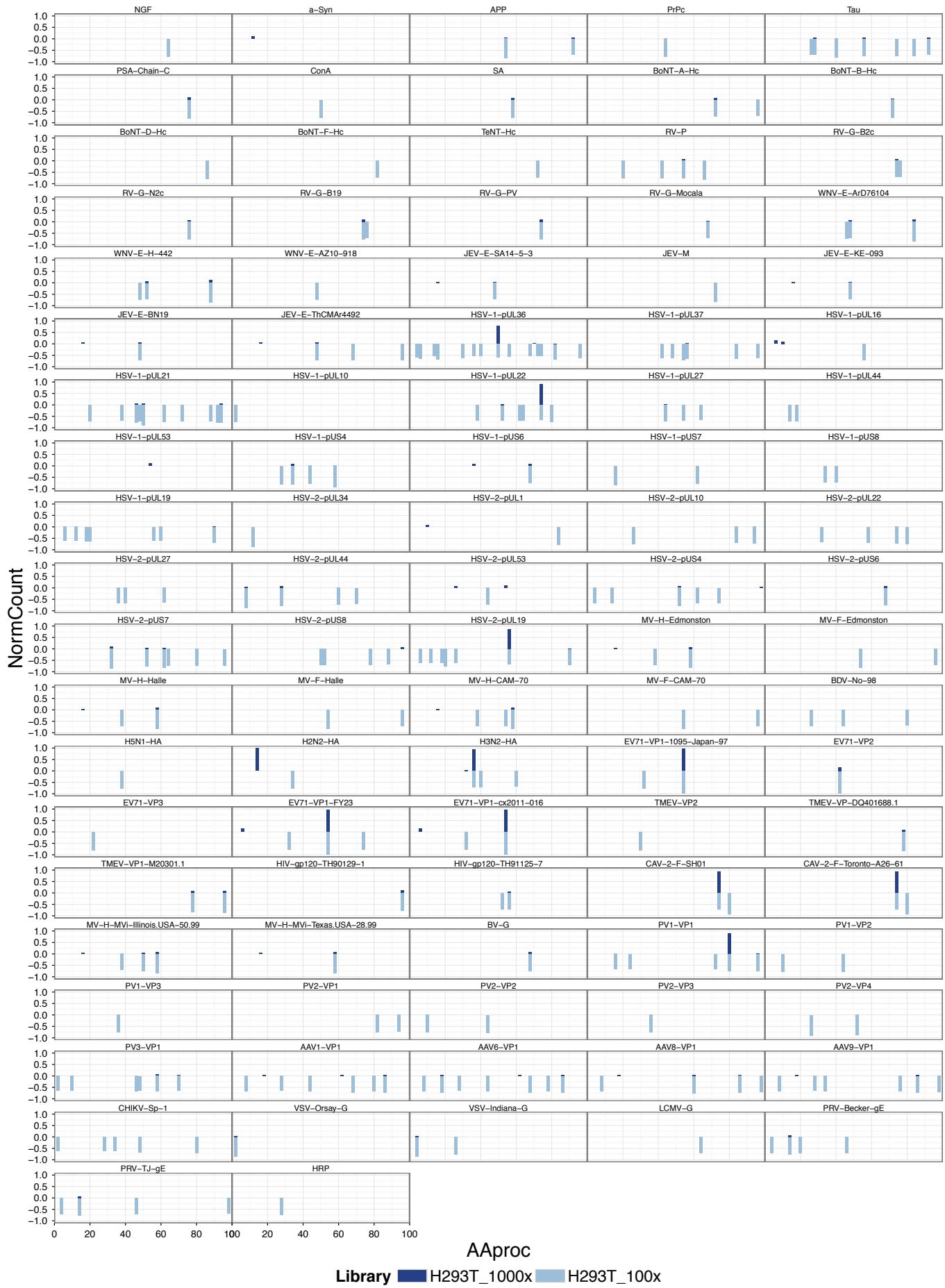
```
knitr::kable(out.plot.list$bottom2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	HSV-1-pUL44	272	14aa	0.00	9	12	3	49.80159	1
2	Tau	630	14aa	0.00	7	10	2	33.55231	1
3	HSV-1-pUS7	242	14aa	0.00	4	4	1	32.50720	1
4	HSV-1-pUL44	66	14aa	2.00	15	18	2	31.79270	1
5	PV1-VP1	43	22aa	0.00	6	6	2	31.61406	1
8	HSV-1-pUL27	575	14aa	0.50	10	13	2	31.33214	1
10	HSV-1-pUL36	1529	14aa	0.00	10	15	2	31.21353	1
11	TMEV-VP1-U33047.1	133	14aa	0.00	12	20	2	31.06380	1
12	CAV-2-F-SH01	163	14aa	1.00	8	9	2	30.99296	1
14	HSV-1-pUL36	1429	14aa	0.25	7	9	2	30.82106	1
15	HSV-2-pUL1	185	14aa	0.50	15	18	2	30.68263	1
16	HSV-1-pUL36	2971	14aa	2.25	7	10	2	30.43577	1
17	HSV-2-pUL27	574	14aa	0.00	5	8	1	16.43275	1
18	BoNT-C-Hc	100	22aa	0.00	6	7	1	16.16740	1
19	MV-H-Edmonston	161	14aa	0.00	6	7	1	16.16740	1

```
mcols(all.samples)$Group[mcols(all.samples)$Group== "293T_1000x"] <- "H293T_1000x"  
mcols(all.samples)$Group[mcols(all.samples)$Group== "293T_100x"] <- "H293T_100x"
```

```
out.plot.list <- plotPair("H293T_1000x", "H293T_100x",  
    filterBC=FALSE,  
    filterAnimal=FALSE,  
    AnimaladjustPlot=FALSE,  
    NormalizePlot=TRUE)
```

```
out.plot.list$plot
```



```
knitr::kable(out.plot.list$top2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	H2N2-HA	75	14aa	0	2	4	1	21.237898	1
2	PV1-VP1	769	14aaaA5	0	9	4	1	20.250610	1
3	HSV-1-pUL36	1596	14aa	0	3	7	1	20.092428	1
4	H3N2-HA	202	22aa	0	10	13	1	20.086854	1
5	HSV-2-pUL19	789	14aa	0	3	3	1	20.064739	1
6	HSV-1-pUL22	618	14aa	1	4	5	1	19.907224	1
7	EV71-VP1-1095-Japan-97	155	14aa	0	2	2	1	19.719528	1
10	CAV-2-F-SH01	396	14aa	0	2	2	1	19.719528	1
12	WNV-E-ArD76104	437	14aa	0	5	8	1	5.531846	1
14	HSV-1-pUL16	13	22aa	1	3	10	1	5.531846	1
15	EV71-VP2	105	14aa	0	5	8	1	5.531846	1
16	HIV-gp120-TH90129-1	466	14aa	0	5	8	1	5.531846	1
17	MV-H-Edmonston	355	14aa	1	6	12	1	5.127165	1
20	EV71-VP1-FY23	12	14aa	0	2	3	1	5.127165	1
24	BoNT-A-Hc	336	14aa	0	6	7	1	4.562697	1

```
knitr::kable(out.plot.list$bottom2, format = "markdown")
```

	Gene	startAA	structure	mismatches	mCount	tCount	BC	NormCount	Animals
1	PV1-VP1	924	14aa	3.5	4	8	2	32.49186	1
2	MV-H-Edmonston	353	14aa	0.5	4	6	2	31.68891	1
7	HSV-2-pUL44	35	14aa	0.0	8	13	2	31.65288	1
8	HSV-2-pUL44	34	14aa	0.5	10	12	2	31.39757	1
9	EV71-VP1-1095-Japan-97	155	14aa	0.0	5	8	1	18.09313	1
10	EV71-VP2	105	14aa	0.0	5	8	1	18.09313	1
13	CAV-2-F-SH01	432	14aa	0.0	5	8	1	18.09313	1
15	APP	409	22aa	0.0	6	9	1	17.45588	1
16	HSV-1-pUL21	262	14aaaA5	0.0	6	9	1	17.45588	1
17	VSV-Orsay-G	8	14aa	1.0	3	10	1	17.18657	1
19	HSV-1-pUS4	135	14aa	3.0	2	2	1	17.11910	1
20	HSV-2-pUL19	270	14aa	2.0	12	21	1	16.93788	1
21	WNV-E-ArD76104	437	14aa	1.0	3	4	1	16.87334	1
23	MV-F-Edmonston	298	14aa	1.0	3	4	1	16.87334	1
26	Tau	274	14aaaA5	1.0	8	14	1	16.81694	1

```
devtools::session_info()
```

Session info -----

```
setting  value
version  R version 3.2.3 (2015-12-10)
system   x86_64, linux-gnu
ui       X11
language (EN)
collate  en_US.UTF-8
tz       <NA>
date    2016-01-13
```

Packages -----

package	* version	date	source
acepack	1.3-3.3	2014-11-24	CRAN (R 3.2.0)
AnnotationDbi	1.32.0	2015-11-26	Bioconductor
Biobase	* 2.30.0	2015-11-26	Bioconductor
BiocGenerics	* 0.16.1	2015-11-26	Bioconductor
BiocInstaller	1.20.1	2015-11-26	Bioconductor
BiocParallel	1.4.0	2015-11-26	Bioconductor
biomaRt	2.26.1	2015-11-26	Bioconductor
Biostrings	* 2.38.2	2015-11-26	Bioconductor
biovizBase	1.18.0	2015-11-26	Bioconductor
bitops	1.0-6	2013-08-17	CRAN (R 3.2.0)
BSgenome	1.38.0	2015-11-26	Bioconductor
chron	2.3-47	2015-06-24	CRAN (R 3.2.2)
cluster	2.0.3	2015-07-21	CRAN (R 3.2.2)
codetools	0.2-14	2015-07-15	CRAN (R 3.2.2)
colorspace	1.2-6	2015-03-11	CRAN (R 3.2.0)
data.table	* 1.9.6	2015-09-19	CRAN (R 3.2.2)
DBI	0.3.1	2014-09-24	CRAN (R 3.2.0)
devtools	* 1.9.1	2015-09-11	CRAN (R 3.2.2)
dichromat	2.0-0	2013-01-24	CRAN (R 3.2.0)
digest	0.6.8	2014-12-31	CRAN (R 3.2.0)
doParallel	* 1.0.10	2015-10-14	CRAN (R 3.2.2)
evaluate	0.8	2015-09-18	CRAN (R 3.2.2)
foreach	* 1.4.3	2015-10-13	CRAN (R 3.2.2)
foreign	0.8-66	2015-08-19	CRAN (R 3.2.2)
formatR	1.2.1	2015-09-18	CRAN (R 3.2.2)
Formula	1.2-1	2015-04-07	CRAN (R 3.2.0)
futile.logger	1.4.1	2015-04-20	CRAN (R 3.2.0)
futile.options	1.0.0	2010-04-06	CRAN (R 3.2.0)
GenomeInfoDb	* 1.6.1	2015-11-26	Bioconductor
GenomicAlignments	* 1.6.1	2015-11-26	Bioconductor
GenomicFeatures	1.22.5	2015-11-26	Bioconductor
GenomicRanges	* 1.22.1	2015-11-26	Bioconductor
GGally	0.5.0	2014-12-02	CRAN (R 3.2.0)
ggbio	* 1.18.1	2015-11-26	Bioconductor
ggplot2	* 1.0.1	2015-03-17	CRAN (R 3.2.0)
graph	1.48.0	2015-11-26	Bioconductor
gridExtra	2.0.0	2015-07-14	CRAN (R 3.2.2)
gtable	0.1.2	2012-12-05	CRAN (R 3.2.0)
highr	0.5.1	2015-09-18	CRAN (R 3.2.2)
Hmisc	3.17-0	2015-09-21	CRAN (R 3.2.2)
htmltools	0.2.6	2014-09-08	CRAN (R 3.2.0)
IRanges	* 2.4.4	2015-11-26	Bioconductor
iterators	* 1.0.8	2015-10-13	CRAN (R 3.2.2)
knitr	* 1.11	2015-08-14	CRAN (R 3.2.2)
labeling	0.3	2014-08-23	CRAN (R 3.2.0)
lambda.r	1.1.7	2015-03-20	CRAN (R 3.2.0)
lattice	0.20-33	2015-07-14	CRAN (R 3.2.2)
latticeExtra	0.6-26	2013-08-15	CRAN (R 3.2.0)
magrittr	1.5	2014-11-22	CRAN (R 3.2.2)
MASS	7.3-45	2015-11-10	CRAN (R 3.2.2)
memoise	0.2.1	2014-04-22	CRAN (R 3.2.2)
munsell	0.4.2	2013-07-11	CRAN (R 3.2.0)
nnet	7.3-11	2015-08-30	CRAN (R 3.2.2)
OrganismDbi	1.12.0	2015-11-26	Bioconductor
plyr	* 1.8.3	2015-06-12	CRAN (R 3.2.2)
proto	0.3-10	2012-12-22	CRAN (R 3.2.0)

RBGL	1.46.0	2015-11-26	Bioconductor
RColorBrewer	1.1-2	2014-12-07	CRAN (R 3.2.0)
Rcpp	0.12.2	2015-11-15	CRAN (R 3.2.2)
RCurl	1.95-4.7	2015-06-30	CRAN (R 3.2.2)
reshape	0.8.5	2014-04-23	CRAN (R 3.2.0)
reshape2	1.4.1	2014-12-06	CRAN (R 3.2.0)
rmarkdown	0.8.1	2015-10-10	CRAN (R 3.2.2)
rpart	4.1-10	2015-06-29	CRAN (R 3.2.2)
Rsamtools	* 1.22.0	2015-11-26	Bioconductor
RSQLite	1.0.0	2014-10-25	CRAN (R 3.2.0)
rtracklayer	1.30.1	2015-11-26	Bioconductor
S4Vectors	* 0.8.3	2015-11-26	Bioconductor
scales	0.3.0	2015-08-25	CRAN (R 3.2.2)
stringi	1.0-1	2015-10-22	CRAN (R 3.2.2)
stringr	1.0.0	2015-04-30	CRAN (R 3.2.2)
SummarizedExperiment	* 1.0.1	2015-11-26	Bioconductor
survival	2.38-3	2015-07-02	CRAN (R 3.2.2)
VariantAnnotation	1.16.3	2015-11-26	Bioconductor
XML	3.98-1.3	2015-06-30	CRAN (R 3.2.2)
XVector	* 0.10.0	2015-11-26	Bioconductor
yaml	2.1.13	2014-06-12	CRAN (R 3.2.0)
zlibbioc	1.16.0	2015-11-26	Bioconductor

Top 10 heatmap analysis output

Tomas Bjorklund

Wed Jan 13 13:34:59 2016

This is the final script presenting top 10 candidates as heatmap plots.

```
suppressPackageStartupMessages(library(knitr))
```

Setup parameters

```
select.samples <- readRDS("data/normalizedSampleRangesDefined.RDS")
mcols(select.samples)$Sequence <- names(select.samples)
names(select.samples) <- make.names(names(select.samples), unique=TRUE)
length.Table <- data.table(seqnames=names(seqlengths(select.samples)),
                           seqlength=seqlengths(select.samples), key="seqnames")
select.samples <- data.table(as.data.frame(select.samples), key="seqnames")
select.samples[,c("strand","qwidth","cigar","njunc","end"):=NULL]
select.samples <- select.samples[length.Table] #A data.table berge to match seqlengths to their respective .
```

Selection of relevant samples

```
select.samples <- select.samples[-grep("4wks|PrimN_1000x_RNA",select.samples$Group),]
select.samples[, c("Category", "Protein", "Origin", "Extra", "Number","GeneName") := tstrsplit(seqnames, ",",
select.samples[,c("seqnames","Extra"):=NULL]

select.samples.binCat <- select.samples
setkeyv(select.samples.binCat,c("Group","Category"))
select.samples.binCat[,c("BCcount","NormCount")]:=list(unlist(lapply(strsplit(paste(BC, collapse=","), ","),
mean(NormCount)), by=key(select.samples.binCat)))
select.samples.binCat <- unique(select.samples.binCat)
select.samples.binCat <- select.samples.binCat[,c("Group","Category",
"BCcount","NormCount"), with = FALSE]
select.samples.binCat[,totBC:=sum(BCcount), by="Group"]
max.count <- max(select.samples.binCat$totBC)
select.samples.binCat[,BCcountN:=BCcount/totBC*max.count]

length.Table[, c("Category", "Protein", "Origin", "Extra", "Number","GeneName") := tstrsplit(seqnames, ",",
setkey(length.Table,"Category")
length.Table[,seqlength:=sum(seqlength), by="Category"]
length.Table <- length.Table[,c("Category","seqlength"), with = FALSE]
length.Table <- unique(length.Table)
setkey(select.samples.binCat,"Category")
select.samples.binCat <- select.samples.binCat[length.Table,nomatch=0]
select.samples.binCat[,Category:=gsub("/|_|'","-",Category)]
select.samples.binCat[,BCcountNseq:=BCcountN/seqlength]
select.samples.binCat[,NormCountBC:=BCcountNseq*NormCount]

select.samples.binGene <- select.samples
```

```

setkeyv(select.samples.binGene,c("Group","Category","GeneName"))
select.samples.binGene[,c("BCcount","NormCount"):=list(unlist(lapply(strsplit(paste(BC, collapse=",",), ","),
                                                               mean(NormCount))), by=key(select.samples.binGene))]
select.samples.binGene <- unique(select.samples.binGene)
select.samples.binGene <- select.samples.binGene[,c("Group","GeneName","Category","BCcount","seqlength","NormCount")]
select.samples.binGene[,GeneName:=gsub("/|_|'","-",GeneName)]
select.samples.binGene[,totBC:=sum(BCcount), by="Group"]
max.count <- max(select.samples.binGene$totBC)
select.samples.binGene[,BCcountN:=BCcount/totBC*max.count]
select.samples.binGene[,BCcountNseq:=BCcountN/seqlength]
select.samples.binGene[,NormCountBC:=BCcountNseq*NormCount]

select.samples.binPos <- select.samples
setkeyv(select.samples.binPos,c("Group","structure","Sequence"))
select.samples.binPos <- unique(select.samples.binPos)
#Due to key, this removes replicates if identical sequence mapped to multiple genes

select.samples.binPos[,AA:=((start+2)/3)+floor((width/3)/2)]
setkeyv(select.samples.binPos,c("Group","Category","GeneName","AA"))
select.samples.binPos[,c("BCcount","NormCount","AnimalCount","mainStruct","mismatches"):=
  list(unlist(lapply(strsplit(paste(BC, collapse=",",), ","),function(x) length(unique(
    mean(NormCount),
    unlist(lapply(strsplit(paste(Animals, collapse=",",), ","),function(x) length(unlist(
      paste(unique(structure), collapse=","),
      median(mismatches))), by=key(select.samples.binPos)]))

select.samples.binPos <- unique(select.samples.binPos)
select.samples.binPos <- select.samples.binPos[,c("Group","GeneName","AA","NormCount",
                                                 "BCcount","AnimalCount","mainStruct",
                                                 "mismatches","seqlength"), with = FALSE]
select.samples.binPos[,totBC:=sum(BCcount), by="Group"]
max.count <- max(select.samples.binPos$totBC)
select.samples.binPos[,BCcountN:=BCcount/totBC*max.count]
select.samples.binPos[,BCcountNanim:=BCcountN*AnimalCount]
select.samples.binPos[,BCcountNseq:=BCcountN/seqlength]
select.samples.binPos[,NormCountBC:=BCcountNanim*NormCount]
select.samples.binPos[,GeneAA:=paste(GeneName, " [",AA,"]",sep="")]

```

Plot Heatmaps split by Category

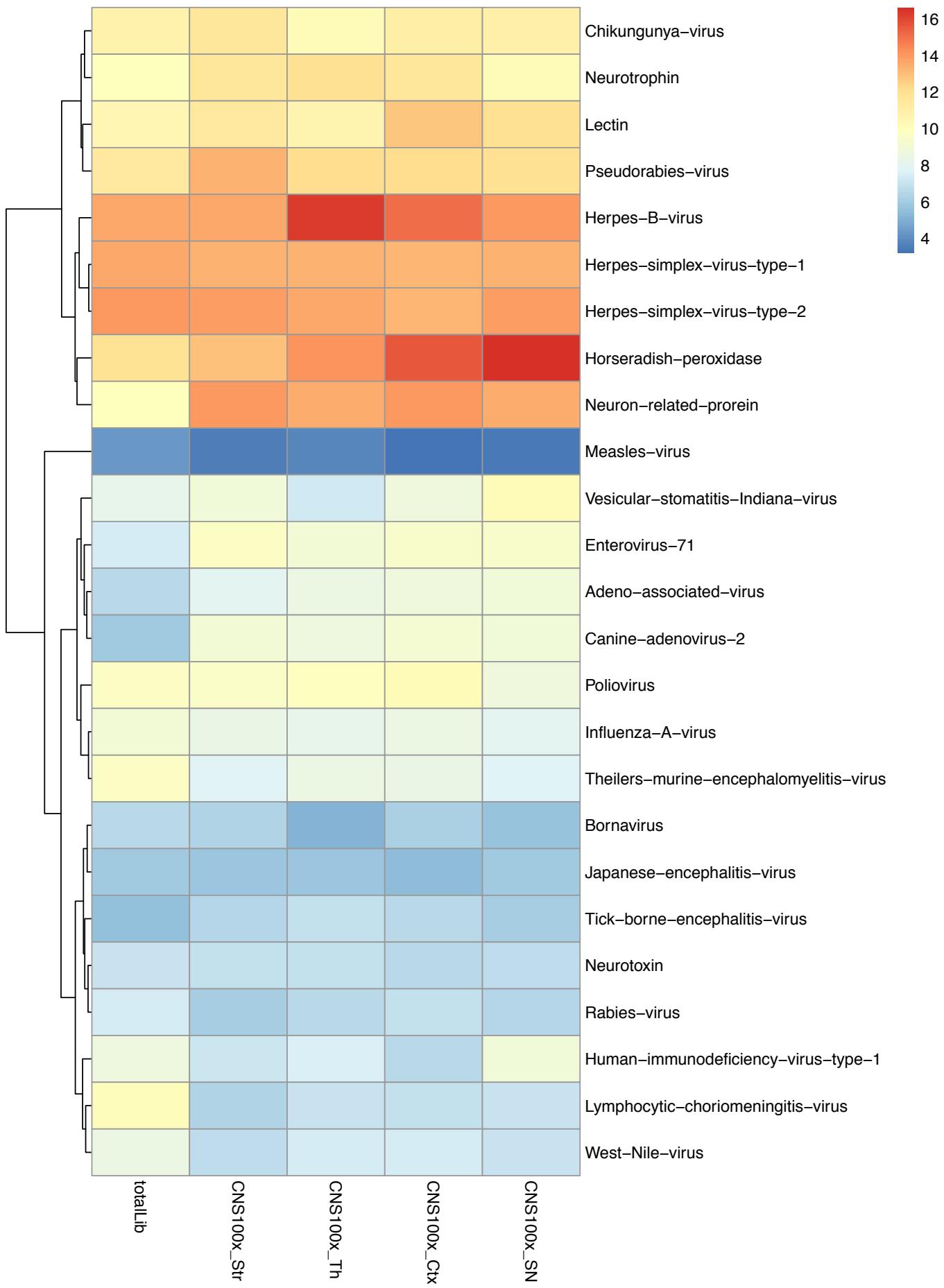
```

plotCategory <- function(select.samples.table,plot.col,sample.select){
  setkey(select.samples.table,Group)
  select.samples.select <- select.samples.table[sample.select]
  eval(parse(text=paste("setorder(select.samples.select,Group, -", plot.col,")", sep="")))
  select.samples.matrix <- acast(select.samples.select, Category~Group, value.var=plot.col)
#Utilizes reshape 2 to make matrix for heatmap

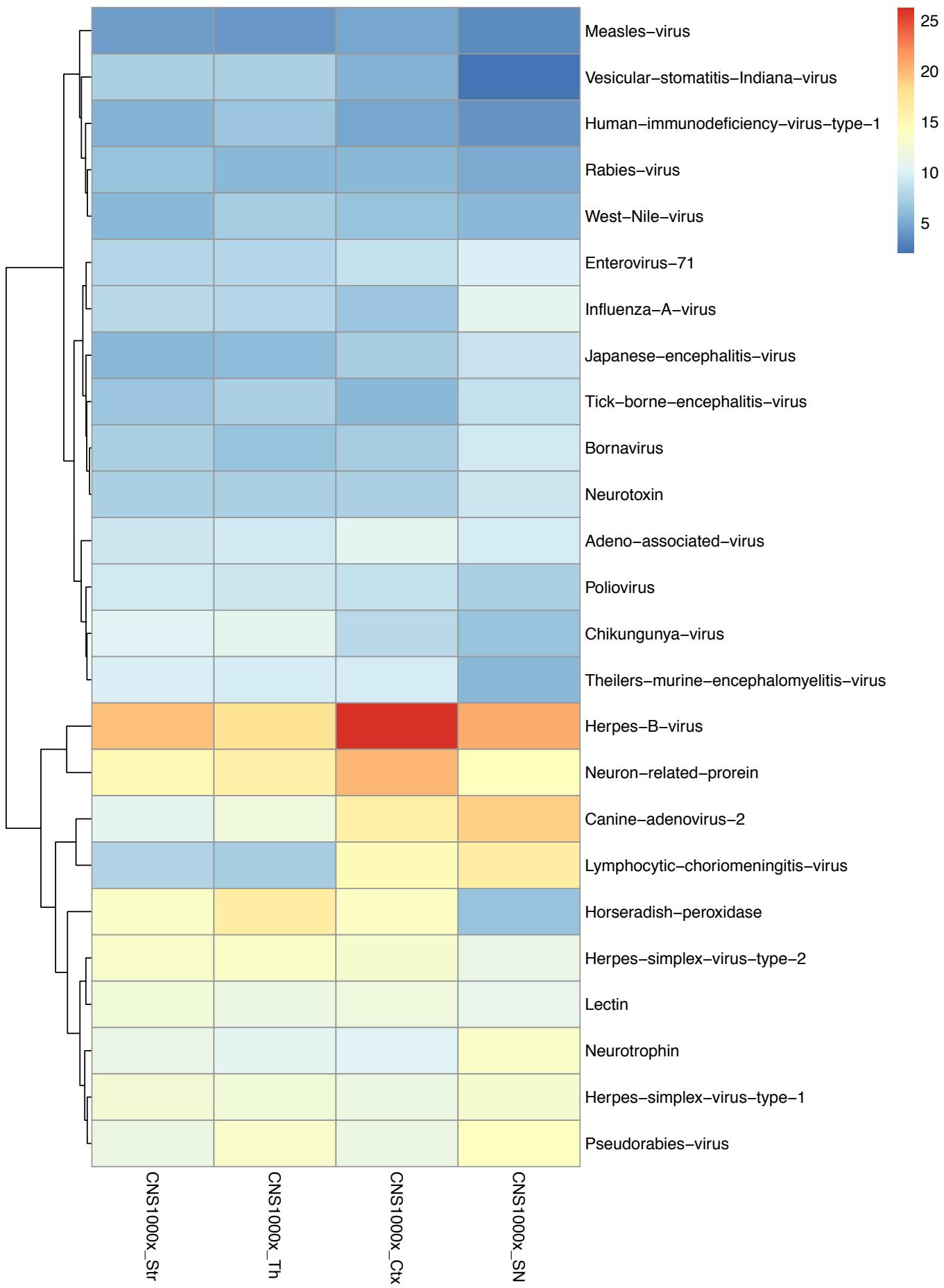
  select.samples.matrix[is.na(select.samples.matrix)] <- 0
  select.samples.matrix <- select.samples.matrix[,sample.select]
  return(pheatmap(select.samples.matrix, cluster_rows=TRUE, show_rownames=TRUE, cluster_cols=FALSE))
}

plotCategory(select.samples.binCat,"BCcountNseq",c("totalLib","CNS100x_Str","CNS100x_Th","CNS100x_Ctx","CNS100x_Raw"))

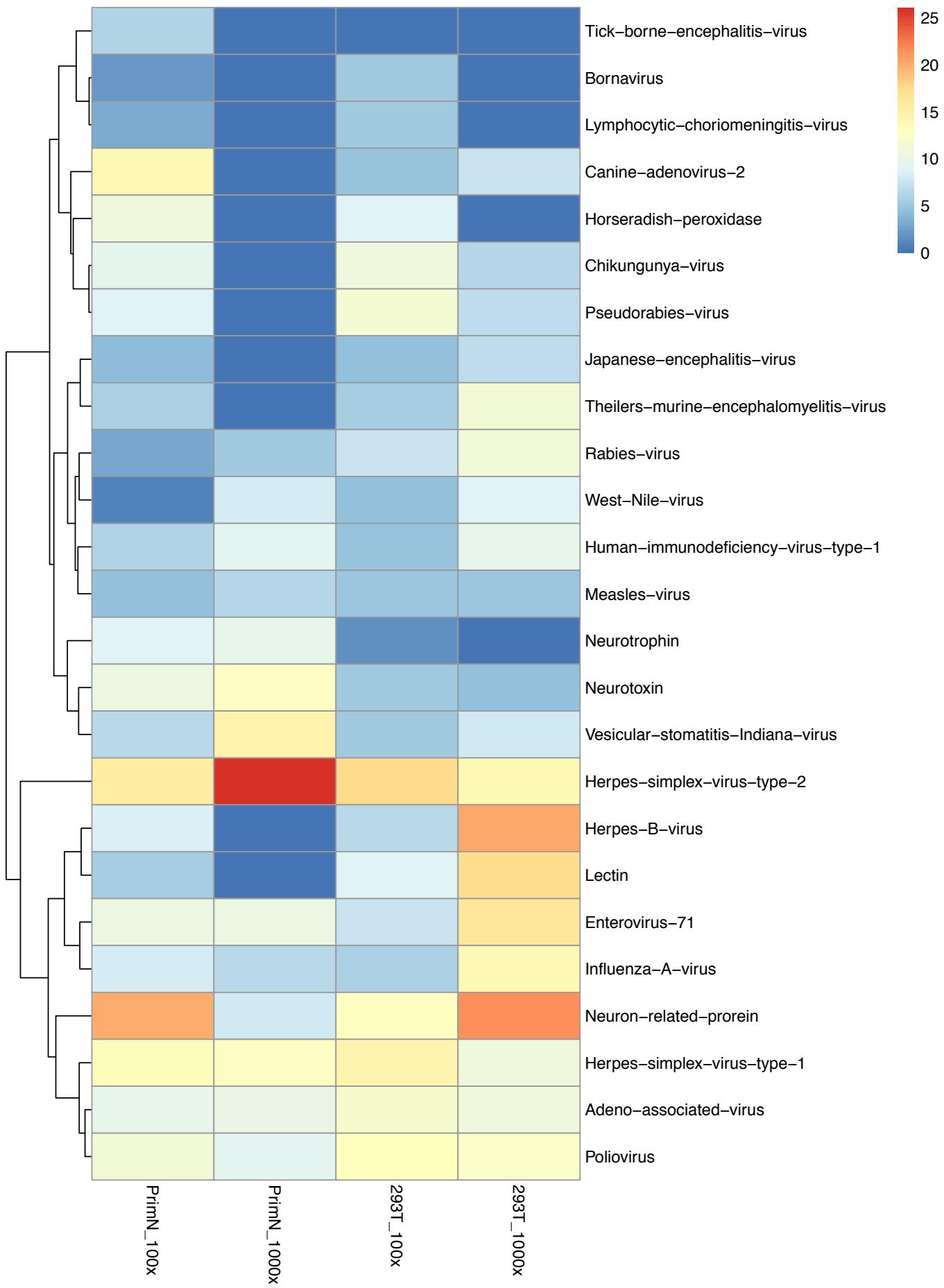
```



```
plotCategory(select.samples.binCat,"BCcountNseq",c("CNS1000x_Str","CNS1000x_Th","CNS1000x_Ctx","CNS1000x_SN
```

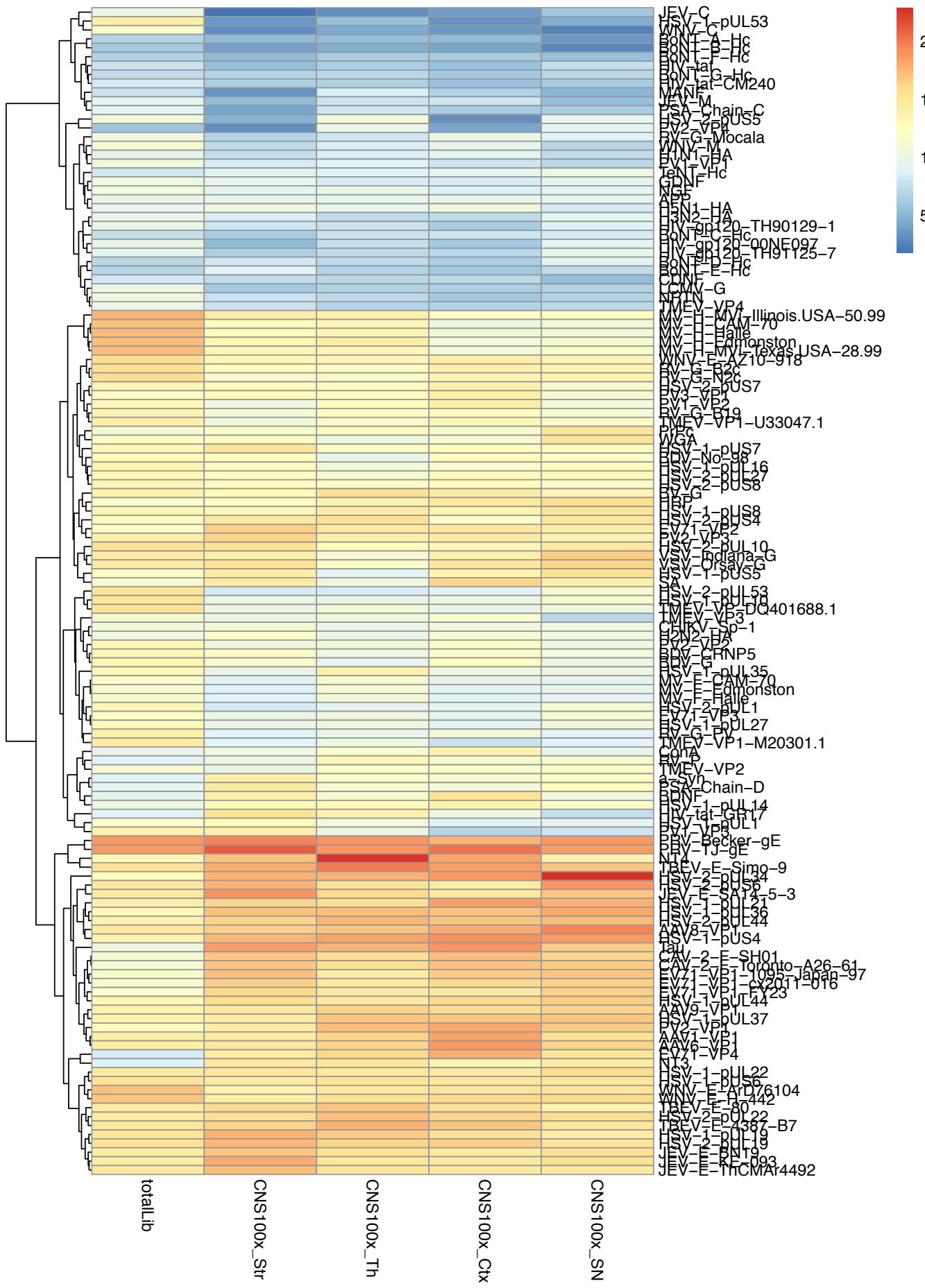


```
plotCategory(select.samples.binCat,"BCcountNseq",c("PrimN_100x","PrimN_1000x","293T_100x","293T_1000x"))
```

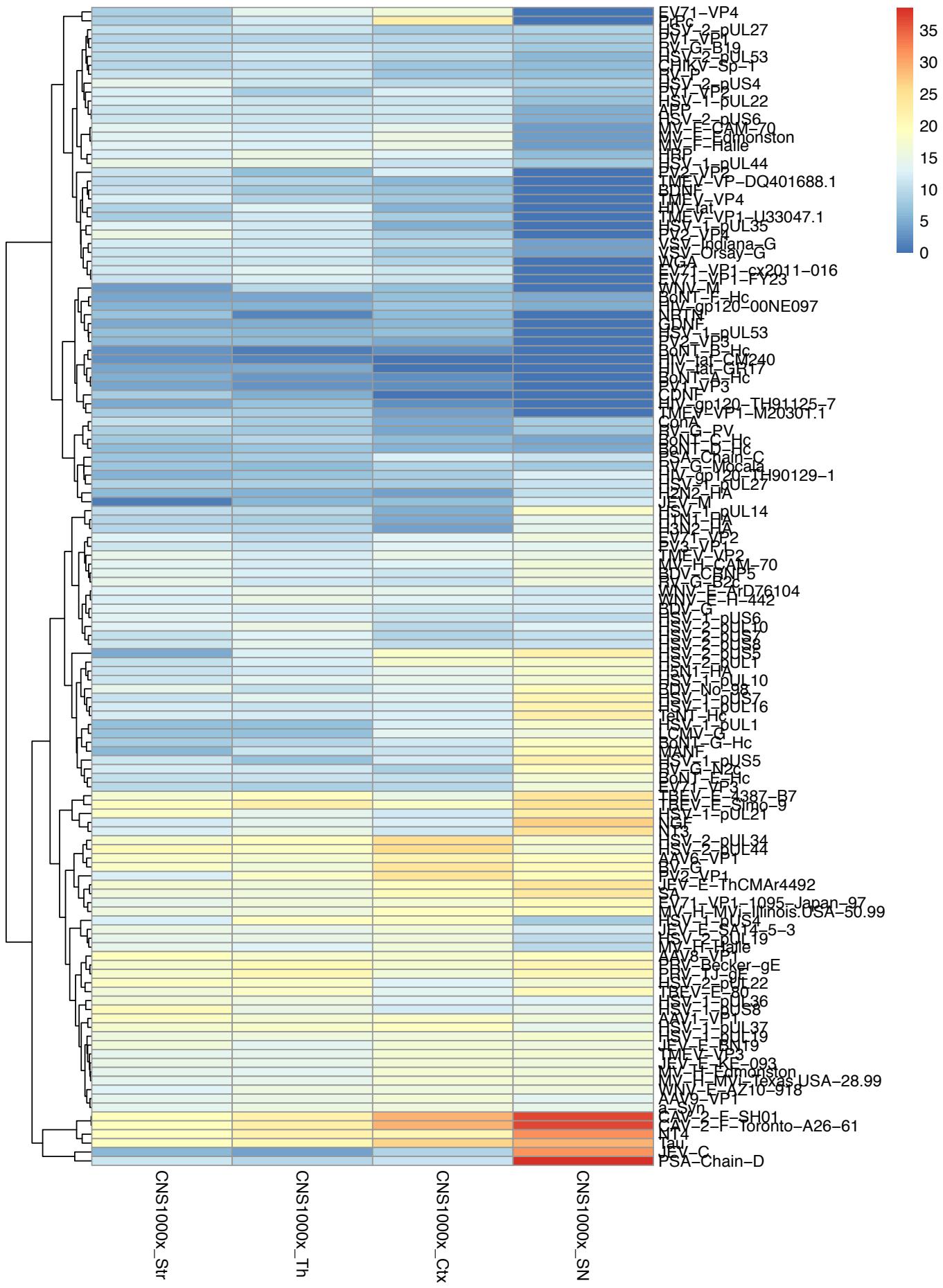


Plot Heatmaps split by GeneName

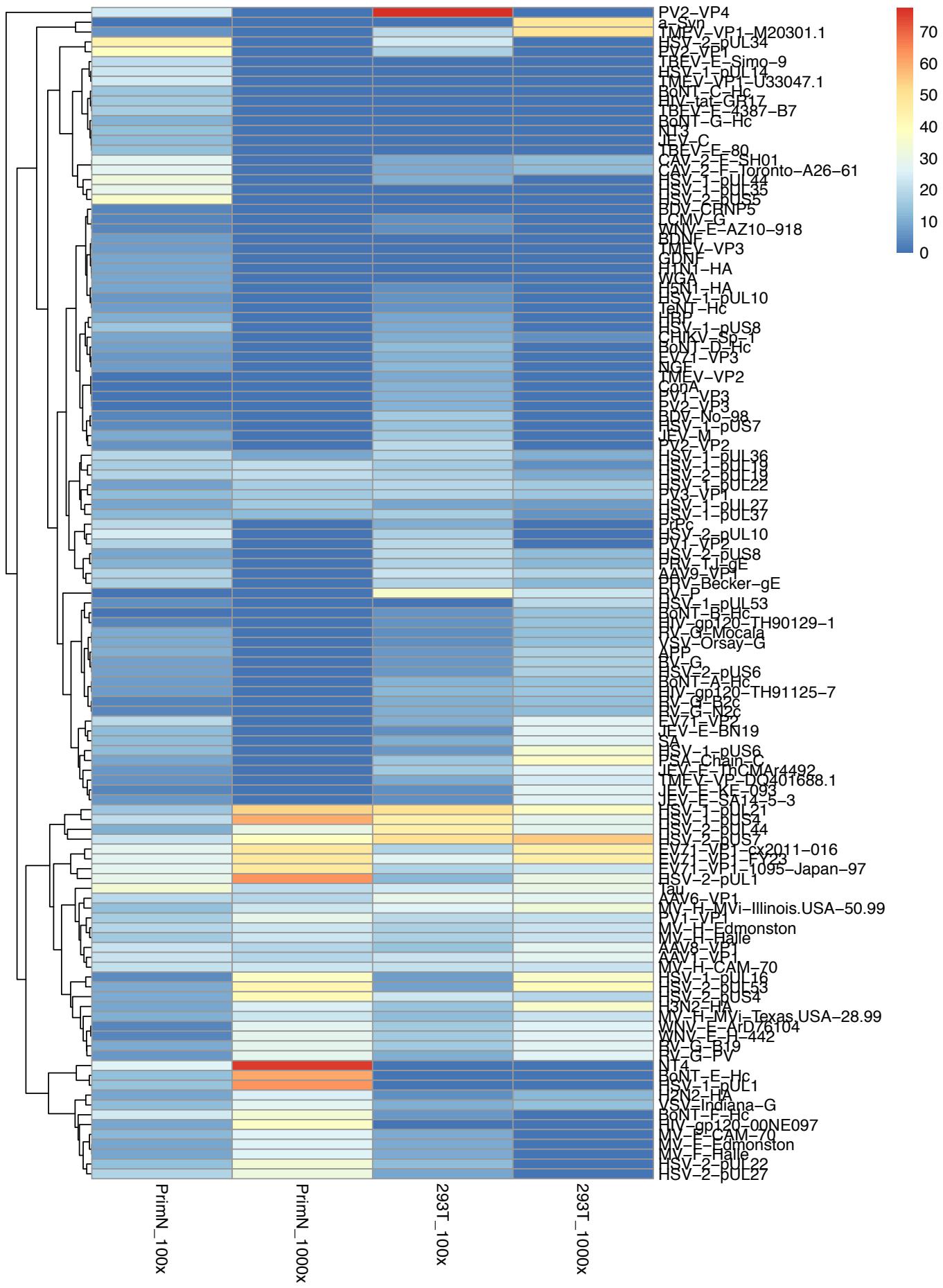
```
plotGene <- function(select.samples.table,plot.col,sample.select){  
  setkey(select.samples.table,Group)  
  select.samples.select <- select.samples.table[sample.select]  
  eval(parse(text=paste("setorder(select.samples.select,Group, -", plot.col,")", sep="")))  
  select.samples.matrix <- acast(select.samples.select, GeneName~Group, value.var=plot.col)  
  #Utilizes reshape 2 to make matrix for heatmap  
  
  select.samples.matrix[is.na(select.samples.matrix)] <- 0  
  select.samples.matrix <- select.samples.matrix[,sample.select]  
  return(pheatmap(select.samples.matrix, cluster_rows=TRUE, show_rownames=TRUE, cluster_cols=FALSE))  
}  
  
plotGene(select.samples.binGene,"BCcountNseq",c("totalLib","CNS100x_Str","CNS100x_Th","CNS100x_Ctx","CNS100x_Nseq"))
```



```
plotGene(select.samples.binGene,"BCcountNseq",c("CNS1000x_Str","CNS1000x_Th","CNS1000x_Ctx","CNS1000x_SN"))
```

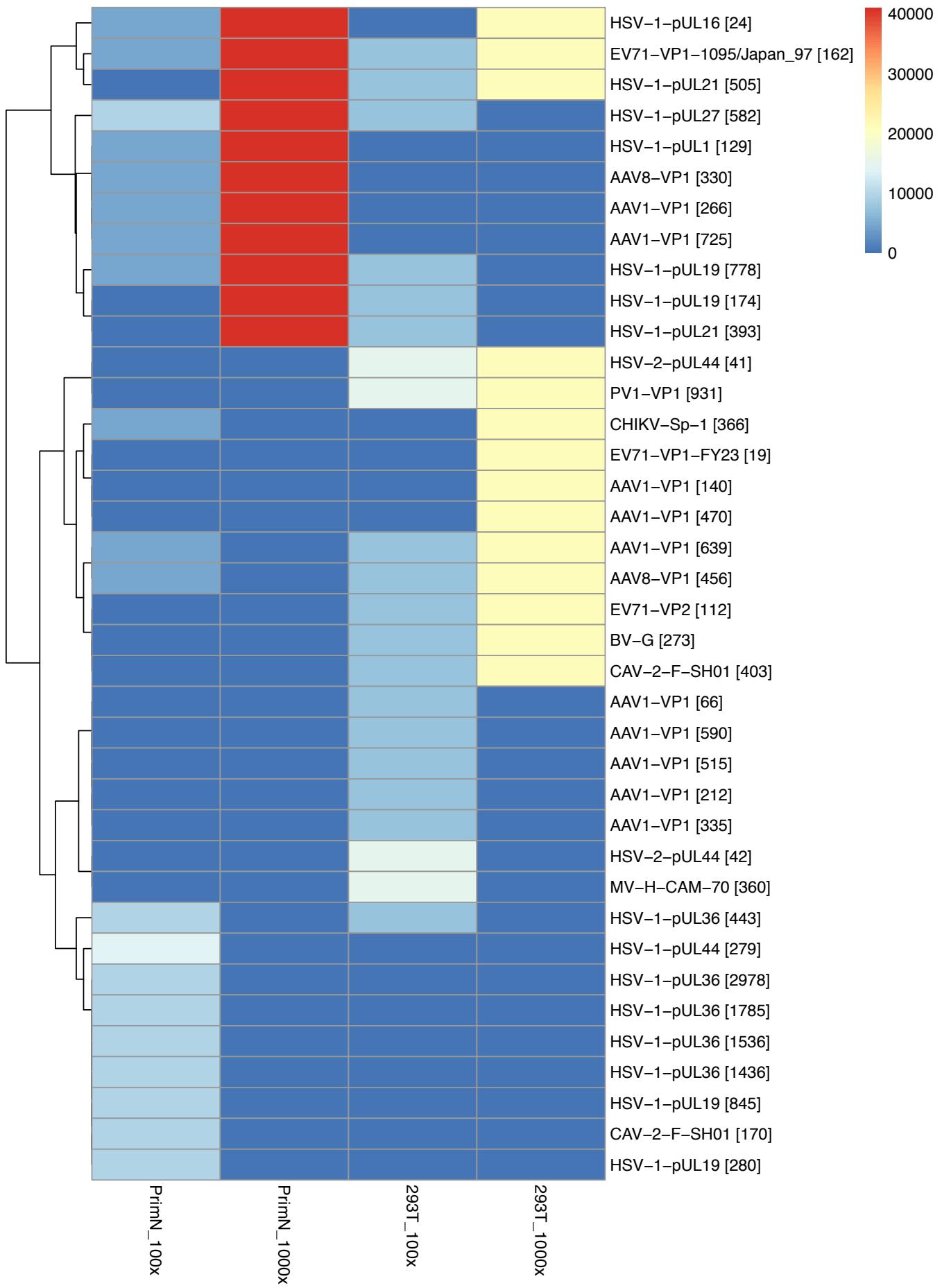


```
plotGene(select.samples.binGene,"BCcountNseq",c("PrimN_100x","PrimN_1000x","293T_100x","293T_1000x"))
```



Selection of top ten fragments per sample

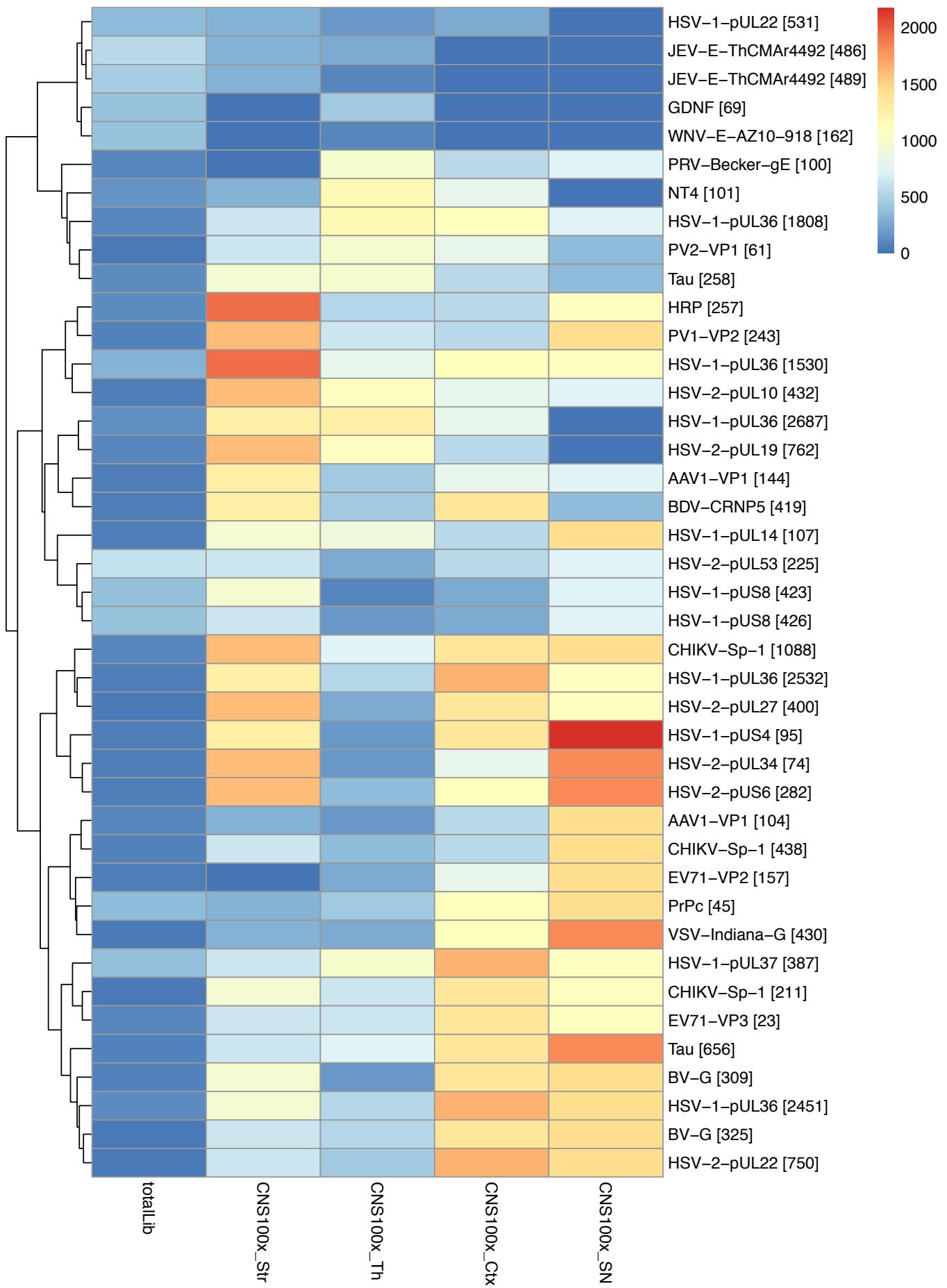
```
plotPos <- function(select.samples.table,plot.col,sample.select){  
  setkeyv(select.samples.table,"Group")  
  select.samples.select <- select.samples.table[sample.select]  
  eval(parse(text=paste("setorder(select.samples.select,Group, -", plot.col,")", sep="")))  
  select.samples.topTen <- select.samples.select[, head(.SD, 10), by=Group]  
  select.samples.out <- select.samples.select[select.samples.select$GeneAA %in% select.samples.topTen$GeneAA]  
  select.samples.out <- acast(select.samples.out, GeneAA~Group, value.var=plot.col) #Utilizes reshape 2 to many  
  select.samples.out[is.na(select.samples.out)] <- 0  
  select.samples.out <- select.samples.out[,sample.select]  
  pheatmap(select.samples.out, cluster_rows=TRUE, show_rownames=TRUE, cluster_cols=FALSE)  
  select.samples.topTen[,c("totBC","seqlength","BCcountN","GeneAA","BCcountNseq")]:=NULL]  
  return(knitr::kable(select.samples.topTen, format = "markdown"))  
}  
  
plotPos(select.samples.binPos,"BCcountN",c("PrimN_100x","PrimN_1000x","293T_100x","293T_1000x"))
```



Group	GeneName	AA	NormCount	BCcount	AnimalCount	mainStruct	mismatches	BC
293T_1000x	AAV1-VP1	140	3.793435	1	1	14aaG4S	1.00	
293T_1000x	AAV1-VP1	470	3.793435	1	1	14aaaA5	0.00	
293T_1000x	AAV1-VP1	639	3.793435	1	1	14aa	1.00	
293T_1000x	AAV8-VP1	456	3.793435	1	1	14aa	2.00	
293T_1000x	CAV-2-F-SH01	403	19.719528	1	1	14aa	0.00	
293T_1000x	CHIKV-Sp-1	366	3.622488	1	1	14aa	0.00	
293T_1000x	EV71-VP1-1095/Japan_97	162	12.490793	1	1	14aa	0.00	
293T_1000x	EV71-VP1-FY23	19	12.490793	1	1	14aa	0.00	
293T_1000x	EV71-VP2	112	12.490793	1	1	14aa	0.00	
293T_1000x	BV-G	273	4.562697	1	1	14aa	1.00	
293T_100x	HSV-2-pUL44	41	15.694712	2	1	14aa	0.50	
293T_100x	HSV-2-pUL44	42	15.694712	2	1	14aa	0.00	
293T_100x	MV-H-CAM-70	360	19.003300	2	1	14aa	0.50	
293T_100x	PV1-VP1	931	15.539818	2	1	14aa	3.50	
293T_100x	AAV1-VP1	66	15.059856	1	1	14aa	0.00	
293T_100x	AAV1-VP1	212	15.059856	1	1	14aaaA5	0.00	
293T_100x	AAV1-VP1	335	15.059856	1	1	14aa	0.00	
293T_100x	AAV1-VP1	515	15.059856	1	1	14aa	7.00	
293T_100x	AAV1-VP1	590	15.059856	1	1	14aaaA5	0.50	
293T_100x	AAV1-VP1	639	15.059856	1	1	14aa	0.00	
PrimN_1000x	AAV1-VP1	266	4.054469	1	1	14aaG4S	0.00	
PrimN_1000x	AAV1-VP1	725	4.054469	1	1	14aaaA5	0.00	
PrimN_1000x	AAV8-VP1	330	4.054469	1	1	14aa	0.00	
PrimN_1000x	EV71-VP1-1095/Japan_97	162	3.864548	1	1	14aa	2.00	
PrimN_1000x	HSV-1-pUL1	129	10.780106	1	1	22aa	1.00	
PrimN_1000x	HSV-1-pUL16	24	10.780106	1	1	22aa	3.50	
PrimN_1000x	HSV-1-pUL19	174	10.780106	1	1	14aa	0.00	
PrimN_1000x	HSV-1-pUL19	778	10.780106	1	1	14aa	0.00	
PrimN_1000x	HSV-1-pUL21	393	10.780106	1	1	22aa	0.00	
PrimN_1000x	HSV-1-pUL21	505	10.780106	1	1	14aa	0.00	
PrimN_100x	HSV-1-pUL44	279	15.960704	3	1	14aa	0.00	
PrimN_100x	CAV-2-F-SH01	170	16.583186	2	1	14aa	1.00	
PrimN_100x	HSV-1-pUL19	280	15.960704	2	1	14aa	1.00	
PrimN_100x	HSV-1-pUL19	845	15.960704	2	1	14aa,14aaaA5	0.00	
PrimN_100x	HSV-1-pUL27	582	15.960704	2	1	14aa	0.50	
PrimN_100x	HSV-1-pUL36	443	15.960704	2	1	14aa,14aaaA5	0.50	
PrimN_100x	HSV-1-pUL36	1436	15.960704	2	1	14aa	0.25	
PrimN_100x	HSV-1-pUL36	1536	15.960704	2	1	14aa	0.00	
PrimN_100x	HSV-1-pUL36	1785	15.960704	2	1	14aa,22aa	0.00	

Group	GeneName	AA	NormCount	BCcount	AnimalCount	mainStruct	mismatches	BC
PrimN_100x	HSV-1-pUL36	2978	15.960704	2	1	14aa		2.25

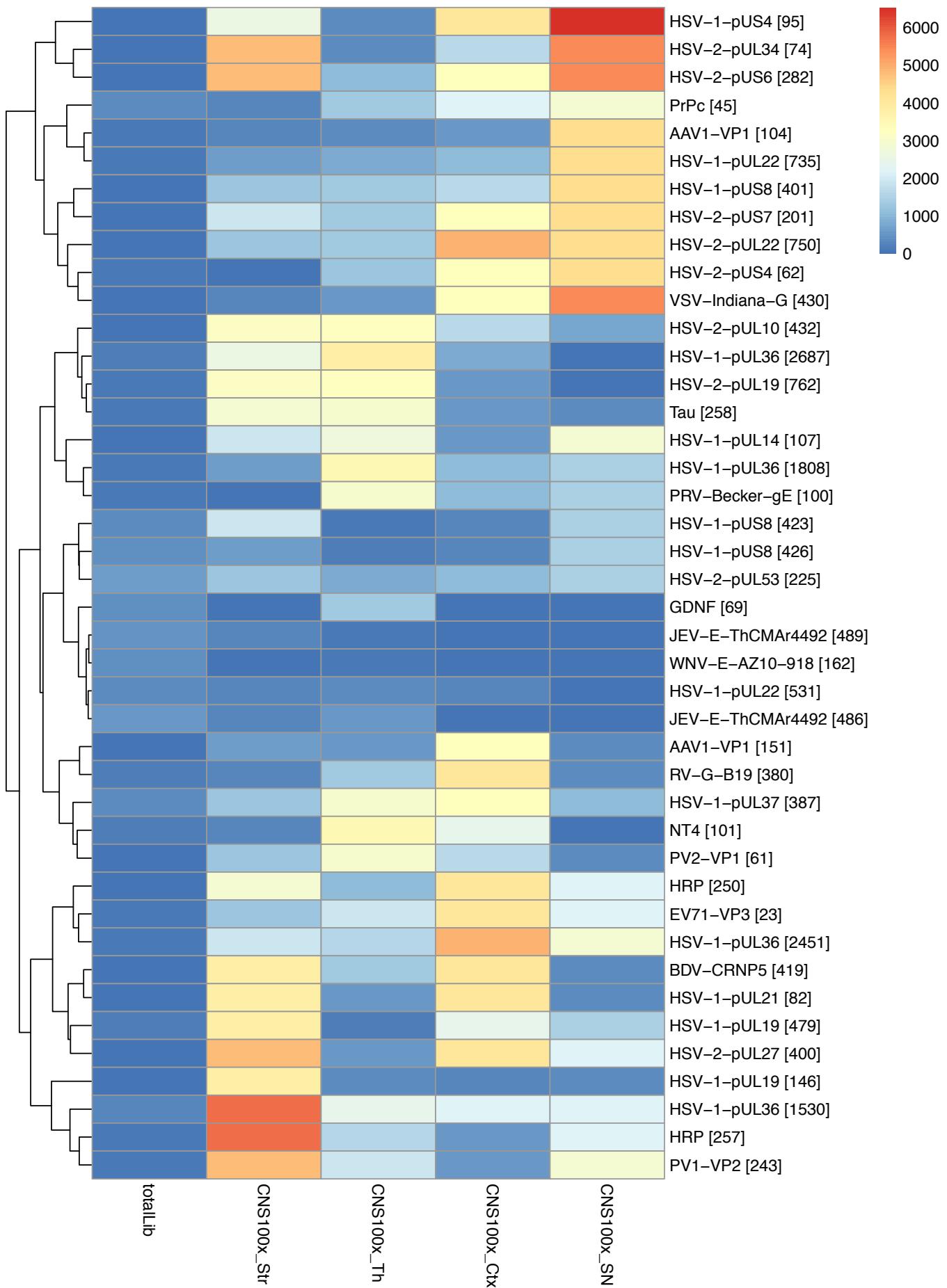
```
plotPos(select.samples.binPos,"BCcountN",c("totalLib","CNS100x_Str","CNS100x_Th","CNS100x_Ctx","CNS100x_SN"))
```



Group	GeneName	AA	NormCount	BCcount	AnimalCount	mainStruct	mismatches
CNS100x_Ctx	HSV-1-pUL36	2451	6.764485	6	3	14aa	0.000
CNS100x_Ctx	HSV-1-pUL36	2532	6.764485	6	2	14aa,22aa	0.125
CNS100x_Ctx	HSV-1-pUL37	387	6.764485	6	2	14aa,22aa	0.500
CNS100x_Ctx	HSV-2-pUL22	750	7.015719	6	3	14aa,22aa	0.000
CNS100x_Ctx	BDV-CRNP5	419	6.943438	5	3	14aa,14aaG4S	0.000
CNS100x_Ctx	CHIKV-Sp-1	211	6.557930	5	2	14aa	0.000
CNS100x_Ctx	CHIKV-Sp-1	1088	6.557930	5	2	14aa,14aaaA5	2.500
CNS100x_Ctx	EV71-VP3	23	7.696183	5	3	14aa,14aaaA5	0.000
CNS100x_Ctx	BV-G	309	8.800267	5	2	14aa,22aa	0.250
CNS100x_Ctx	BV-G	325	8.800267	5	2	14aa	0.000
CNS100x_SN	HSV-1-pUS4	95	6.622249	6	3	14aa,14aaG4S	0.250
CNS100x_SN	HSV-2-pUL34	74	7.087182	5	3	14aa,14aaaA5	0.750
CNS100x_SN	HSV-2-pUS6	282	7.087182	5	3	14aa	0.000
CNS100x_SN	Tau	656	6.452116	5	2	14aa,14aaaA5,14aaG4S	0.000
CNS100x_SN	VSV-Indiana-G	430	5.260092	5	3	14aa,22aa	0.000
CNS100x_SN	AAV1-VP1	104	6.460376	4	3	14aa	0.625
CNS100x_SN	CHIKV-Sp-1	438	6.462385	4	2	14aa	2.500
CNS100x_SN	CHIKV-Sp-1	1088	6.462385	4	2	14aa,14aaaA5	1.000
CNS100x_SN	EV71-VP2	157	6.458255	4	2	14aa	0.000
CNS100x_SN	BV-G	309	7.966180	4	2	14aa,22aa	0.000
CNS100x_Str	HSV-1-pUL36	1530	8.971415	6	3	22aa	0.000
CNS100x_Str	HRP	257	11.478122	6	3	14aa,14aaG4S	0.000
CNS100x_Str	CHIKV-Sp-1	1088	8.790423	5	2	14aa,14aaaA5	0.125
CNS100x_Str	HSV-2-pUL10	432	9.005401	5	2	14aa	0.000
CNS100x_Str	HSV-2-pUL19	762	9.005401	5	2	14aa,22aa	0.000
CNS100x_Str	HSV-2-pUL27	400	9.005401	5	3	14aa	1.000
CNS100x_Str	HSV-2-pUL34	74	9.005401	5	3	14aa,14aaaA5	0.500
CNS100x_Str	HSV-2-pUS6	282	9.005401	5	3	14aa,22aa	1.000
CNS100x_Str	PV1-VP2	243	8.497670	5	3	14aa	0.000
CNS100x_Str	AAV1-VP1	144	9.759704	4	2	22aa	0.000
CNS100x_Th	HSV-1-pUL36	2687	16.590961	14	3	14aa,14aaaA5	0.000
CNS100x_Th	HSV-1-pUL36	1808	16.590961	13	3	14aa,14aaG4S	0.000
CNS100x_Th	NT4	101	17.830703	13	3	14aa,14aaaA5,14aaG4S	0.500
CNS100x_Th	HSV-2-pUL10	432	16.450100	12	3	14aa	0.000
CNS100x_Th	HSV-2-pUL19	762	16.450100	12	3	14aa,22aa	0.000
CNS100x_Th	HSV-1-pUL37	387	16.590961	11	3	14aa,22aa	0.000
CNS100x_Th	Tau	258	17.749381	11	3	14aa	0.000
CNS100x_Th	PV2-VP1	61	15.425411	11	3	14aa	0.000
CNS100x_Th	PRV-Becker-gE	100	16.534930	11	3	14aa,14aaaA5,14aaG4S	0.000

Group	GeneName	AA	NormCount	BCcount	AnimalCount	mainStruct	mismatches
CNS100x_Th	HSV-1-pUL14	107	16.590961	10	3	14aa	0.250
totalLib	HSV-2-pUL53	225	147.112080	603	1	14aa,22aa	0.500
totalLib	JEV-E-ThCMAr4492	486	121.229213	556	1	14aa,22aa	0.250
totalLib	JEV-E-ThCMAr4492	489	121.229213	463	1	14aa,22aa	0.000
totalLib	WNV-E-AZ10-918	162	132.096465	412	1	14aa,22aa	0.500
totalLib	HSV-1-pUS8	426	148.363716	402	1	14aa,22aa	0.500
totalLib	GDNF	69	115.597454	393	1	14aa,22aa	0.500
totalLib	HSV-1-pUL37	387	148.363716	388	1	14aa,22aa	1.250
totalLib	HSV-1-pUS8	423	148.363716	377	1	14aa,22aa	1.000
totalLib	HSV-1-pUL22	531	148.363716	363	1	14aa,22aa	0.250
totalLib	PrPc	45	115.234755	360	1	14aa,22aa	0.250

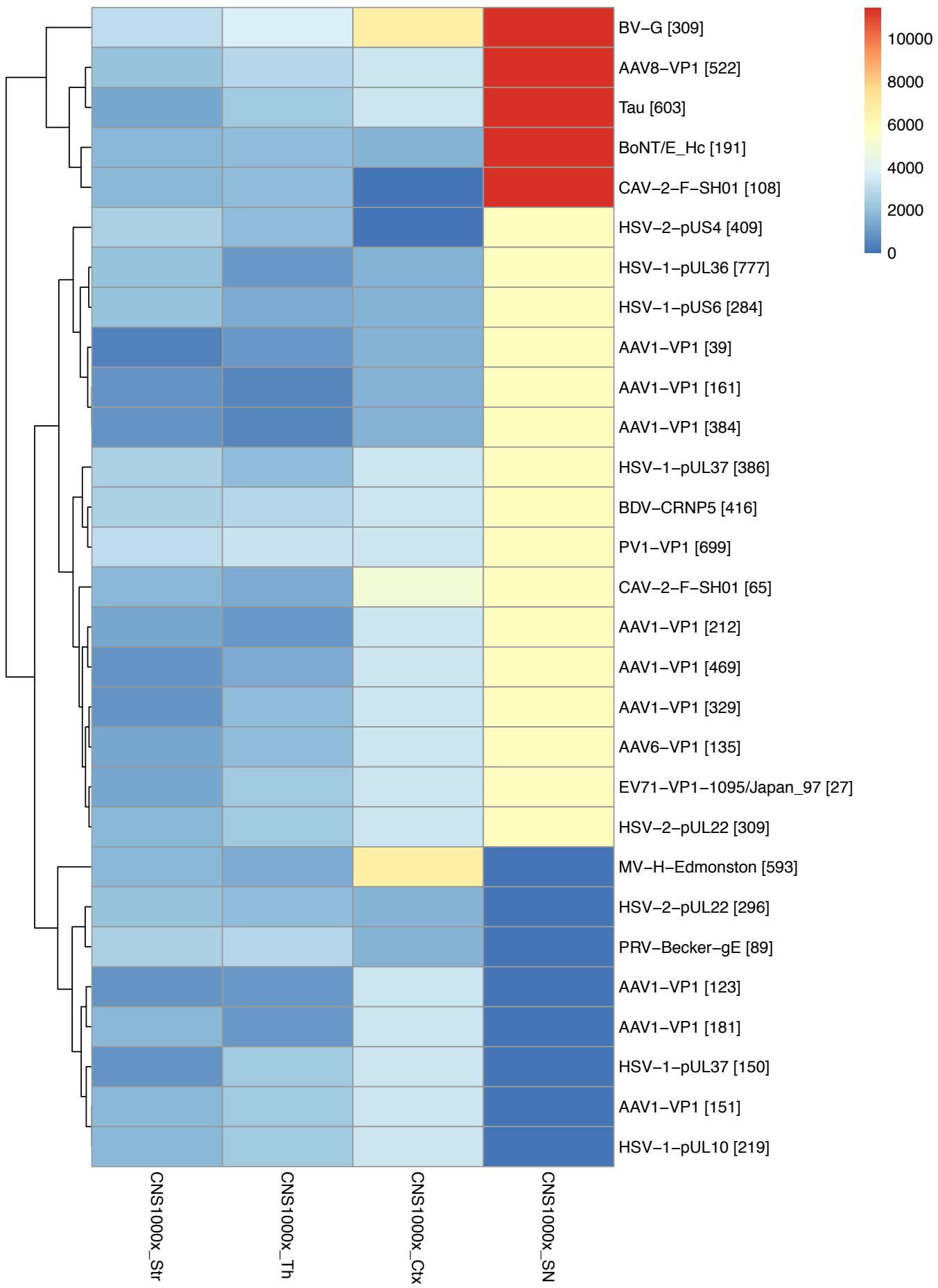
```
plotPos(select.samples.binPos,"BCcountNanim",c("totalLib","CNS100x_Str","CNS100x_Th","CNS100x_Ctx","CNS100x
```



Group	GeneName	AA	NormCount	BCcount	AnimalCount	mainStruct	mismatches
CNS100x_Ctx	HSV-1-pUL36	2451	6.764485	6	3	14aa	0.000
CNS100x_Ctx	HSV-2-pUL22	750	7.015719	6	3	14aa,22aa	0.000
CNS100x_Ctx	BDV-CRNP5	419	6.943438	5	3	14aa,14aaG4S	0.000
CNS100x_Ctx	EV71-VP3	23	7.696183	5	3	14aa,14aaaA5	0.000
CNS100x_Ctx	HSV-1-pUL21	82	6.764485	5	3	14aa	0.000
CNS100x_Ctx	HSV-1-pUS4	95	6.764485	5	3	14aa,14aaG4S	0.500
CNS100x_Ctx	HSV-2-pUL27	400	7.015719	5	3	14aa	1.000
CNS100x_Ctx	HRP	250	6.682046	5	3	14aa	0.000
CNS100x_Ctx	RV-G-B19	380	6.440835	5	3	14aaaA5,14aaG4S	0.500
CNS100x_Ctx	AAV1-VP1	151	7.169508	4	3	14aa	0.000
CNS100x_SN	HSV-1-pUS4	95	6.622249	6	3	14aa,14aaG4S	0.250
CNS100x_SN	HSV-2-pUL34	74	7.087182	5	3	14aa,14aaaA5	0.750
CNS100x_SN	HSV-2-pUS6	282	7.087182	5	3	14aa	0.000
CNS100x_SN	VSV-Indiana-G	430	5.260092	5	3	14aa,22aa	0.000
CNS100x_SN	AAV1-VP1	104	6.460376	4	3	14aa	0.625
CNS100x_SN	HSV-1-pUL22	735	6.622249	4	3	14aa	0.250
CNS100x_SN	HSV-1-pUS8	401	6.622249	4	3	14aaaA5	0.000
CNS100x_SN	HSV-2-pUL22	750	7.087182	4	3	14aa	0.500
CNS100x_SN	HSV-2-pUS4	62	7.087182	4	3	14aa	0.250
CNS100x_SN	HSV-2-pUS7	201	7.087182	4	3	14aa	0.000
CNS100x_Str	HSV-1-pUL36	1530	8.971415	6	3	22aa	0.000
CNS100x_Str	HRP	257	11.478122	6	3	14aa,14aaG4S	0.000
CNS100x_Str	HSV-2-pUL27	400	9.005401	5	3	14aa	1.000
CNS100x_Str	HSV-2-pUL34	74	9.005401	5	3	14aa,14aaaA5	0.500
CNS100x_Str	HSV-2-pUS6	282	9.005401	5	3	14aa,22aa	1.000
CNS100x_Str	PV1-VP2	243	8.497670	5	3	14aa	0.000
CNS100x_Str	BDV-CRNP5	419	10.235977	4	3	14aa,14aaG4S	0.000
CNS100x_Str	HSV-1-pUL19	146	8.971415	4	3	14aa,14aaG4S	1.625
CNS100x_Str	HSV-1-pUL19	479	8.971415	4	3	14aa,14aaG4S	0.000
CNS100x_Str	HSV-1-pUL21	82	8.971415	4	3	14aa	0.500
CNS100x_Th	HSV-1-pUL36	2687	16.590961	14	3	14aa,14aaaA5	0.000
CNS100x_Th	HSV-1-pUL36	1808	16.590961	13	3	14aa,14aaG4S	0.000
CNS100x_Th	NT4	101	17.830703	13	3	14aa,14aaaA5,14aaG4S	0.500
CNS100x_Th	HSV-2-pUL10	432	16.450100	12	3	14aa	0.000
CNS100x_Th	HSV-2-pUL19	762	16.450100	12	3	14aa,22aa	0.000
CNS100x_Th	HSV-1-pUL37	387	16.590961	11	3	14aa,22aa	0.000
CNS100x_Th	Tau	258	17.749381	11	3	14aa	0.000
CNS100x_Th	PV2-VP1	61	15.425411	11	3	14aa	0.000
CNS100x_Th	PRV-Becker-gE	100	16.534930	11	3	14aa,14aaaA5,14aaG4S	0.000

Group	GeneName	AA	NormCount	BCcount	AnimalCount	mainStruct	mismatches
CNS100x_Th	HSV-1-pUL14	107	16.590961	10	3	14aa	0.250
totalLib	HSV-2-pUL53	225	147.112080	603	1	14aa,22aa	0.500
totalLib	JEV-E-ThCMAr4492	486	121.229213	556	1	14aa,22aa	0.250
totalLib	JEV-E-ThCMAr4492	489	121.229213	463	1	14aa,22aa	0.000
totalLib	WNV-E-AZ10-918	162	132.096465	412	1	14aa,22aa	0.500
totalLib	HSV-1-pUS8	426	148.363716	402	1	14aa,22aa	0.500
totalLib	GDNF	69	115.597454	393	1	14aa,22aa	0.500
totalLib	HSV-1-pUL37	387	148.363716	388	1	14aa,22aa	1.250
totalLib	HSV-1-pUS8	423	148.363716	377	1	14aa,22aa	1.000
totalLib	HSV-1-pUL22	531	148.363716	363	1	14aa,22aa	0.250
totalLib	PrPc	45	115.234755	360	1	14aa,22aa	0.250

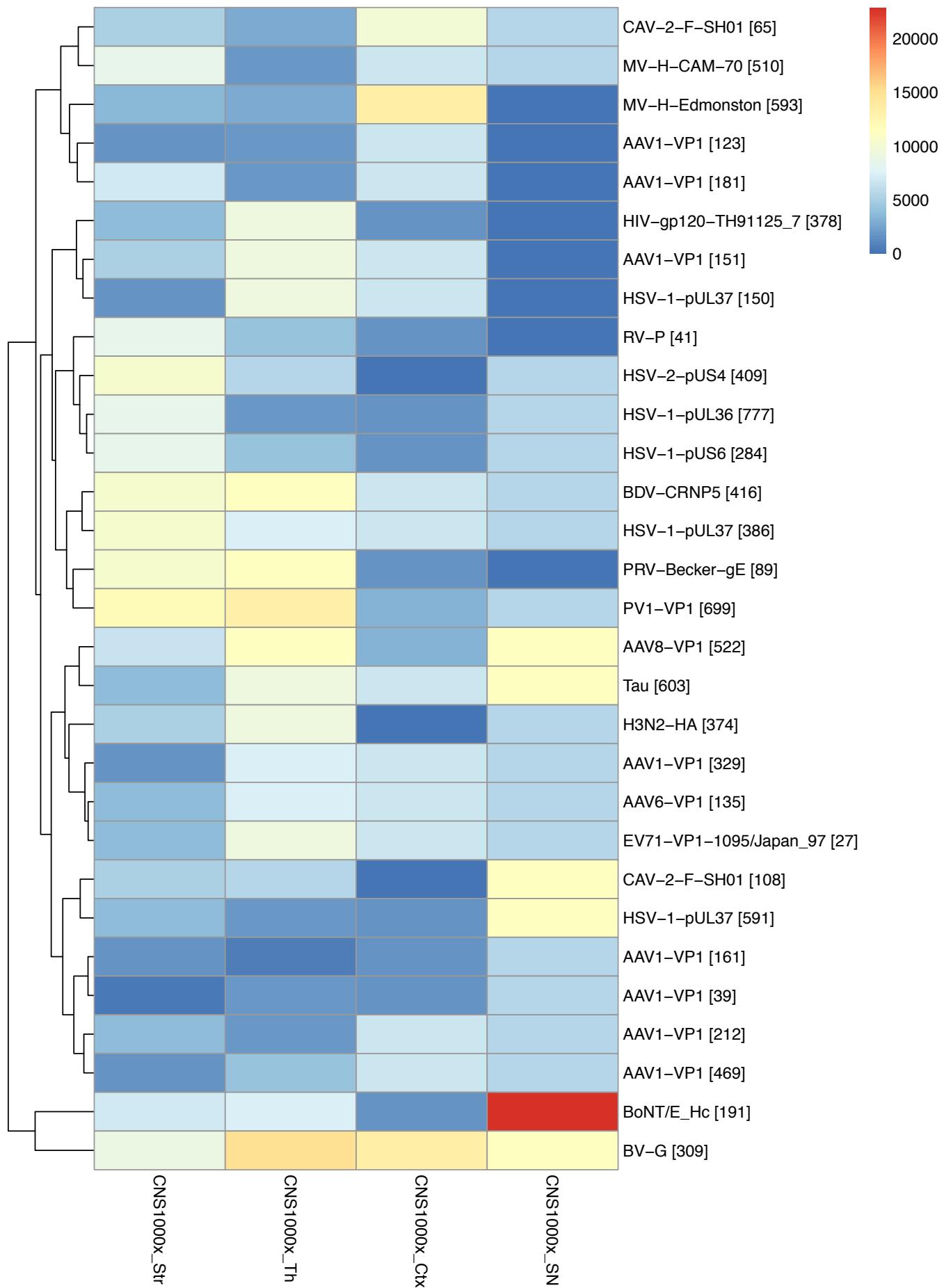
```
plotPos(select.samples.binPos, "BCcountN", c("CNS1000x_Str", "CNS1000x_Th", "CNS1000x_Ctx", "CNS1000x_SN"))
```



Group	GeneName	AA	NormCount	BCcount	AnimalCount	mainStruct	mismatches	E
CNS1000x_Ctx	BV-G	309	5.866275	4	2	14aa	0.500	
CNS1000x_Ctx	MV-H-Edmonston	593	5.397283	4	2	14aa,14aaaA5	0.375	
CNS1000x_Ctx	CAV-2-F-SH01	65	6.278143	3	2	14aaaA5	1.000	
CNS1000x_Ctx	AAV1-VP1	123	8.399157	2	2	14aa	0.000	
CNS1000x_Ctx	AAV1-VP1	151	8.399157	2	2	14aa	0.500	
CNS1000x_Ctx	AAV1-VP1	181	8.399157	2	2	14aa	0.000	
CNS1000x_Ctx	AAV1-VP1	212	8.399157	2	2	14aaaA5	0.000	
CNS1000x_Ctx	AAV1-VP1	329	8.399157	2	2	14aa	1.250	
CNS1000x_Ctx	AAV1-VP1	469	8.399157	2	2	14aa	0.000	
CNS1000x_Ctx	AAV6-VP1	135	8.399157	2	2	14aa	0.500	
CNS1000x_SN	AAV8-VP1	522	5.808903	2	1	14aa	0.000	
CNS1000x_SN	CAV-2-F-SH01	108	7.265575	2	1	14aa	0.000	
CNS1000x_SN	BV-G	309	6.566840	2	1	14aa	0.500	
CNS1000x_SN	Tau	603	4.875844	2	1	14aa,22aa	0.000	
CNS1000x_SN	BoNT/E_Hc	191	7.920536	2	2	14aa	0.500	
CNS1000x_SN	AAV1-VP1	39	5.808903	1	1	14aa	0.000	
CNS1000x_SN	AAV1-VP1	161	5.808903	1	1	14aa	0.000	
CNS1000x_SN	AAV1-VP1	212	5.808903	1	1	14aaaA5	0.000	
CNS1000x_SN	AAV1-VP1	329	5.808903	1	1	14aa	0.000	
CNS1000x_SN	AAV1-VP1	384	5.808903	1	1	14aa	2.000	
CNS1000x_Str	BV-G	309	9.321071	7	3	14aa	1.000	
CNS1000x_Str	PV1-VP1	699	5.781098	7	4	14aa,22aa	0.000	
CNS1000x_Str	BDV-CRNP5	416	7.189508	6	4	14aaaA5	0.000	
CNS1000x_Str	HSV-1-pUL37	386	6.236153	6	4	14aa,14aaaG4S	0.000	
CNS1000x_Str	HSV-2-pUS4	409	5.981587	6	4	14aa	0.000	
CNS1000x_Str	PRV-Becker-gE	89	6.322660	6	4	14aa,22aa	0.000	
CNS1000x_Str	AAV8-VP1	522	8.164010	5	3	14aa	0.000	
CNS1000x_Str	HSV-1-pUL36	777	6.236153	5	4	14aa	0.000	
CNS1000x_Str	HSV-1-pUS6	284	6.236153	5	4	14aa,14aaaA5	0.125	
CNS1000x_Str	HSV-2-pUL22	296	5.981587	5	3	14aa,14aaaA5	0.125	
CNS1000x_Th	BV-G	309	25.314057	8	4	14aa	0.000	
CNS1000x_Th	PV1-VP1	699	15.265347	7	4	14aa,22aa	0.000	
CNS1000x_Th	AAV8-VP1	522	16.541045	6	4	14aa	0.500	
CNS1000x_Th	BDV-CRNP5	416	15.145850	6	4	14aaaA5	0.000	
CNS1000x_Th	PRV-Becker-gE	89	17.024871	6	4	14aa,22aa	0.875	
CNS1000x_Th	AAV1-VP1	151	16.541045	5	4	14aa	0.500	
CNS1000x_Th	EV71-VP1-1095/Japan_97	27	14.101092	5	4	14aa	0.000	
CNS1000x_Th	HSV-1-pUL10	219	15.049748	5	3	14aa	0.000	
CNS1000x_Th	HSV-1-pUL37	150	15.049748	5	4	22aa	0.000	

Group	GeneName	AA	NormCount	BCcount	AnimalCount	mainStruct	mismatches	E
CNS1000x_Th	HSV-2-pUL22	309	14.677593	5	3	14aa		0.500

```
plotPos(select.samples.binPos, "BCcountNanim", c("CNS1000x_Str", "CNS1000x_Th", "CNS1000x_Ctx", "CNS1000x_SN"))
```



Group	GeneName	AA	NormCount	BCcount	AnimalCount	mainStruct	mismatches	E
CNS1000x_Ctx	BV-G	309	5.866275	4	2	14aa	0.500	
CNS1000x_Ctx	MV-H-Edmonston	593	5.397283	4	2	14aa,14aaA5	0.375	
CNS1000x_Ctx	CAV-2-F-SH01	65	6.278143	3	2	14aaA5	1.000	
CNS1000x_Ctx	AAV1-VP1	123	8.399157	2	2	14aa	0.000	
CNS1000x_Ctx	AAV1-VP1	151	8.399157	2	2	14aa	0.500	
CNS1000x_Ctx	AAV1-VP1	181	8.399157	2	2	14aa	0.000	
CNS1000x_Ctx	AAV1-VP1	212	8.399157	2	2	14aaaA5	0.000	
CNS1000x_Ctx	AAV1-VP1	329	8.399157	2	2	14aa	1.250	
CNS1000x_Ctx	AAV1-VP1	469	8.399157	2	2	14aa	0.000	
CNS1000x_Ctx	AAV6-VP1	135	8.399157	2	2	14aa	0.500	
CNS1000x_SN	BoNT/E_Hc	191	7.920536	2	2	14aa	0.500	
CNS1000x_SN	AAV8-VP1	522	5.808903	2	1	14aa	0.000	
CNS1000x_SN	CAV-2-F-SH01	108	7.265575	2	1	14aa	0.000	
CNS1000x_SN	BV-G	309	6.566840	2	1	14aa	0.500	
CNS1000x_SN	HSV-1-pUL37	591	4.918278	1	2	22aa	0.000	
CNS1000x_SN	Tau	603	4.875844	2	1	14aa,22aa	0.000	
CNS1000x_SN	AAV1-VP1	39	5.808903	1	1	14aa	0.000	
CNS1000x_SN	AAV1-VP1	161	5.808903	1	1	14aa	0.000	
CNS1000x_SN	AAV1-VP1	212	5.808903	1	1	14aaaA5	0.000	
CNS1000x_SN	AAV1-VP1	329	5.808903	1	1	14aa	0.000	
CNS1000x_Str	PV1-VP1	699	5.781098	7	4	14aa,22aa	0.000	
CNS1000x_Str	BDV-CRNP5	416	7.189508	6	4	14aaaA5	0.000	
CNS1000x_Str	HSV-1-pUL37	386	6.236153	6	4	14aa,14aaG4S	0.000	
CNS1000x_Str	HSV-2-pUS4	409	5.981587	6	4	14aa	0.000	
CNS1000x_Str	PRV-Becker-gE	89	6.322660	6	4	14aa,22aa	0.000	
CNS1000x_Str	BV-G	309	9.321071	7	3	14aa	1.000	
CNS1000x_Str	HSV-1-pUL36	777	6.236153	5	4	14aa	0.000	
CNS1000x_Str	HSV-1-pUS6	284	6.236153	5	4	14aa,14aaaA5	0.125	
CNS1000x_Str	MV-H-CAM-70	510	5.781607	5	4	14aa,22aa	0.250	
CNS1000x_Str	RV-P	41	5.264432	5	4	14aaaA5	0.000	
CNS1000x_Th	BV-G	309	25.314057	8	4	14aa	0.000	
CNS1000x_Th	PV1-VP1	699	15.265347	7	4	14aa,22aa	0.000	
CNS1000x_Th	AAV8-VP1	522	16.541045	6	4	14aa	0.500	
CNS1000x_Th	BDV-CRNP5	416	15.145850	6	4	14aaaA5	0.000	
CNS1000x_Th	PRV-Becker-gE	89	17.024871	6	4	14aa,22aa	0.875	
CNS1000x_Th	AAV1-VP1	151	16.541045	5	4	14aa	0.500	
CNS1000x_Th	EV71-VP1-1095/Japan_97	27	14.101092	5	4	14aa	0.000	
CNS1000x_Th	HSV-1-pUL37	150	15.049748	5	4	22aa	0.000	
CNS1000x_Th	HIV-gp120-TH91125_7	378	12.531731	5	4	22aa	0.000	

Group	GeneName	AA	NormCount	BCcount	AnimalCount	mainStruct	mismatches	E
CNS1000x_Th	H3N2-HA	374	14.171908	5	4	14aa		1.000

```
devtools::session_info()
```

Session info -----

```
setting  value
version  R version 3.2.3 (2015-12-10)
system   x86_64, linux-gnu
ui        X11
language en_US:en
collate  en_US.UTF-8
tz       <NA>
date     2016-01-13
```

Packages -----

package	* version	date	source
acepack	1.3-3.3	2014-11-24	CRAN (R 3.2.0)
annotate	1.48.0	2016-01-07	Bioconductor
AnnotationDbi	1.32.0	2015-11-26	Bioconductor
Biobase	* 2.30.0	2015-11-26	Bioconductor
BiocGenerics	* 0.16.1	2015-11-26	Bioconductor
BiocInstaller	1.20.1	2015-11-26	Bioconductor
BiocParallel	1.4.0	2015-11-26	Bioconductor
biomaRt	2.26.1	2015-11-26	Bioconductor
Biostrings	* 2.38.2	2015-11-26	Bioconductor
biovizBase	1.18.0	2015-11-26	Bioconductor
bitops	1.0-6	2013-08-17	CRAN (R 3.2.0)
BSgenome	1.38.0	2015-11-26	Bioconductor
chron	2.3-47	2015-06-24	CRAN (R 3.2.2)
cluster	2.0.3	2015-07-21	CRAN (R 3.2.2)
codetools	0.2-14	2015-07-15	CRAN (R 3.2.2)
colorspace	1.2-6	2015-03-11	CRAN (R 3.2.0)
data.table	* 1.9.6	2015-09-19	CRAN (R 3.2.2)
DBI	0.3.1	2014-09-24	CRAN (R 3.2.0)
DESeq2	* 1.10.1	2016-01-07	Bioconductor
devtools	* 1.9.1	2015-09-11	CRAN (R 3.2.2)
dichromat	2.0-0	2013-01-24	CRAN (R 3.2.0)
digest	0.6.8	2014-12-31	CRAN (R 3.2.0)
doParallel	* 1.0.10	2015-10-14	CRAN (R 3.2.2)
evaluate	0.8	2015-09-18	CRAN (R 3.2.2)
foreach	* 1.4.3	2015-10-13	CRAN (R 3.2.2)
foreign	0.8-66	2015-08-19	CRAN (R 3.2.2)
formatR	1.2.1	2015-09-18	CRAN (R 3.2.2)
Formula	1.2-1	2015-04-07	CRAN (R 3.2.0)
futile.logger	1.4.1	2015-04-20	CRAN (R 3.2.0)
futile.options	1.0.0	2010-04-06	CRAN (R 3.2.0)
genefilter	1.52.0	2016-01-07	Bioconductor
geneplotter	1.48.0	2016-01-07	Bioconductor
GenomeInfoDb	* 1.6.1	2015-11-26	Bioconductor
GenomicAlignments	* 1.6.1	2015-11-26	Bioconductor
GenomicFeatures	1.22.5	2015-11-26	Bioconductor

GenomicRanges	* 1.22.1	2015-11-26 Bioconductor
GGally	0.5.0	2014-12-02 CRAN (R 3.2.0)
ggbio	* 1.18.1	2015-11-26 Bioconductor
ggplot2	* 1.0.1	2015-03-17 CRAN (R 3.2.0)
graph	1.48.0	2015-11-26 Bioconductor
gridExtra	2.0.0	2015-07-14 CRAN (R 3.2.2)
gttable	0.1.2	2012-12-05 CRAN (R 3.2.0)
highr	0.5.1	2015-09-18 CRAN (R 3.2.2)
Hmisc	3.17-0	2015-09-21 CRAN (R 3.2.2)
htmltools	0.2.6	2014-09-08 CRAN (R 3.2.0)
IRanges	* 2.4.4	2015-11-26 Bioconductor
iterators	* 1.0.8	2015-10-13 CRAN (R 3.2.2)
knitr	* 1.11	2015-08-14 CRAN (R 3.2.2)
lambda.r	1.1.7	2015-03-20 CRAN (R 3.2.0)
lattice	0.20-33	2015-07-14 CRAN (R 3.2.2)
latticeExtra	0.6-26	2013-08-15 CRAN (R 3.2.0)
locfit	1.5-9.1	2013-04-20 CRAN (R 3.2.2)
magrittr	1.5	2014-11-22 CRAN (R 3.2.2)
MASS	7.3-45	2015-11-10 CRAN (R 3.2.2)
memoise	0.2.1	2014-04-22 CRAN (R 3.2.2)
munsell	0.4.2	2013-07-11 CRAN (R 3.2.0)
nnet	7.3-11	2015-08-30 CRAN (R 3.2.2)
OrganismDbi	1.12.0	2015-11-26 Bioconductor
pheatmap	* 1.0.8	2015-12-11 CRAN (R 3.2.3)
plyr	* 1.8.3	2015-06-12 CRAN (R 3.2.2)
proto	0.3-10	2012-12-22 CRAN (R 3.2.0)
RBGL	1.46.0	2015-11-26 Bioconductor
RColorBrewer	1.1-2	2014-12-07 CRAN (R 3.2.0)
Rcpp	* 0.12.2	2015-11-15 CRAN (R 3.2.2)
RcppArmadillo	* 0.6.400.2.2	2015-12-16 CRAN (R 3.2.3)
RCurl	1.95-4.7	2015-06-30 CRAN (R 3.2.2)
reshape	0.8.5	2014-04-23 CRAN (R 3.2.0)
reshape2	* 1.4.1	2014-12-06 CRAN (R 3.2.0)
rmarkdown	0.8.1	2015-10-10 CRAN (R 3.2.2)
rpart	4.1-10	2015-06-29 CRAN (R 3.2.2)
Rsamtools	* 1.22.0	2015-11-26 Bioconductor
RSQLite	1.0.0	2014-10-25 CRAN (R 3.2.0)
rtracklayer	1.30.1	2015-11-26 Bioconductor
S4Vectors	* 0.8.3	2015-11-26 Bioconductor
scales	0.3.0	2015-08-25 CRAN (R 3.2.2)
stringi	1.0-1	2015-10-22 CRAN (R 3.2.2)
stringr	1.0.0	2015-04-30 CRAN (R 3.2.2)
SummarizedExperiment	* 1.0.1	2015-11-26 Bioconductor
survival	2.38-3	2015-07-02 CRAN (R 3.2.2)
VariantAnnotation	1.16.3	2015-11-26 Bioconductor
XML	3.98-1.3	2015-06-30 CRAN (R 3.2.2)
xtable	1.8-0	2015-11-02 CRAN (R 3.2.2)
XVector	* 0.10.0	2015-11-26 Bioconductor
yaml	2.1.13	2014-06-12 CRAN (R 3.2.0)
zlibbioc	1.16.0	2015-11-26 Bioconductor