Load and Process Data from E-MTAB files and ROC fit to NCBI data

Michael A. Gilchrist

20 Jul 2020

Preliminary Information

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

To compile in emacs use M-n e

Purpose

The purpose of this document is to process empirical measurements of mRNA abundances using mRNA-seq technologies. These measurements are based on multiple measurements. The values are generally the means of the counts. Unfortunately we don't know anything about the sd of the values used to calculate the mean counts.

To remedy this, I analyze the different, fine scaled measurements of the various embryo stages in the hermaphrodites. We can use the observed relationship between the mean across treatments and their sd.

Load Libraries

```
library(Biostrings) ## process first to avoid conflicts
```

```
## Loading required package: BiocGenerics
```

```
## Loading required package: parallel
```

```
##
## Attaching package: 'BiocGenerics'
```

```
## The following objects are masked from 'package:parallel':
##
##     clusterApply, clusterApplyLB, clusterCall, clusterEvalQ,
##     clusterExport, clusterMap, parApply, parCapply, parLapply,
##     parLapplyLB, parRapply, parSapply, parSapplyLB
```

```
## The following objects are masked from 'package:stats':
##
##     IQR, mad, sd, var, xtabs
```

```
## The following objects are masked from 'package:base':
##
##     anyDuplicated, append, as.data.frame, basename, cbind, colnames,
##     dirname, do.call, duplicated, eval, evalq, Filter, Find, get, grep,
##     grepl, intersect, is.unsorted, lapply, Map, mapply, match, mget,
##     order, paste, pmax, pmax.int, pmin, pmin.int, Position, rank,
##     rbind, Reduce, rownames, sapply, setdiff, sort, table, tapply,
##     union, unique, unsplit, which, which.max, which.min

## Loading required package: S4Vectors

## Loading required package: stats4

##
## Attaching package: 'S4Vectors'

## The following object is masked from 'package:base':
##
##     expand.grid

## Loading required package: IRanges

## Loading required package: XVector

##
## Attaching package: 'Biostrings'

## The following object is masked from 'package:base':
##
##     strsplit
```

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'

## The following object is masked from 'package:S4Vectors':
##
##     expand
```

```
library(tibble)
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:Biostrings':
##
##     collapse, intersect, setdiff, setequal, union

## The following object is masked from 'package:XVector':
##
##     slice

## The following objects are masked from 'package:IRanges':
##
##     collapse, desc, intersect, setdiff, slice, union

## The following objects are masked from 'package:S4Vectors':
##
##     first, intersect, rename, setdiff, setequal, union
```

```
## The following objects are masked from 'package:BiocGenerics':
##
##     combine, intersect, setdiff, union

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringr)
library(forcats)
library(ggplot2)
library(knitr)
library(ggpubr)
library(ggpmisc)
library(purrr)
```

```
##
## Attaching package: 'purrr'

## The following object is masked from 'package:XVector':
##
##     compact

## The following object is masked from 'package:IRanges':
##
##     reduce
```

```
creationInfo <- paste0("\tDate: ", date(), "\n\tLocation: ", sub(".*/AcrossTissue", "AcrossTissue", getw

exportData=TRUE ## Flag for running save(), write.csv() and other output commands
```

Load and Shape Data

ROC

Get gene names

Extract from NCBI's FASTA file

- There are a bunch of anomolous genes that have a phi of 1 but an elevated sd relative to the rest of genes with that value Turns out they have a width which is not a multiple of 3.
- FASTA file downloaded by Lu from NCBI database. This data includes many non-verified genes.
- Note that a gene may have some isoforms that are not multiples of 3, so we don't want to exclude all isoforms of a gene.
- Data that hasn't been filtered for these anomolous genes has the prefix 'unfiltered'.
- Filtered data does not have a prefix.
- ROC analyses should be rerun with the WB fasta file:

```
## Load original FASTA file
## Using Biostrings function which is not a standard df
unfilteredSeqData <- readDNAStringSet("Input/c_elegan-NCBI.fasta")
```

```
## names are really long descriptions.
## NEed to extract relevant part
unfilteredSeqDesc  <- names(unfilteredSeqData)
unfilteredSeqLength <- width(unfilteredSeqData)
unfilteredSeqGeneNames <- sub(".*\\[gene=([^]]+).*", "\\1", unfilteredSeqDesc)

## verify there's a match for each entry
if(sum(is.na(unfilteredSeqGeneNames)) ==0) print("Every entry matches")
```

```
## [1] "Every entry matches"
```

```
## Verify that all 'names' are unique.
if(length(unfilteredSeqGeneNames) != length(unique(unfilteredSeqGeneNames))) print("Some geneNames appea
```

```
## [1] "Some geneNames appear twice due to isoforms"
```

```
## Filter out whose length is not a multiple of 3
## Keep name of filtered data simple
unfilteredFastaInfoNCBI <- tibble(info=names(unfilteredSeqData), geneName=unfilteredSeqGeneNames, lengt

## Create a vector flagging isoforms with proper gene lengths
properLengthIsoformsFlag <- ((unfilteredFastaInfoNCBI$length %% 3) == 0)

## Check for MtDNA genes
## Results indicate there are none. Good!

## note seqData uses 'names' while fastaInfoNCBI uses 'info'
seqData <- unfilteredSeqData[properLengthIsoformsFlag, ]
fastaInfoNCBI <- unfilteredFastaInfoNCBI[properLengthIsoformsFlag, ]


## Import Phi Values from ROC Output

## detailed information on phi: posterior mean, posterior mean of log10(phi), etc
## StdError really StdDev of posterior
unfilteredPhiPosteriorInfo <- readr::read_csv("Input/phi.posterior.unlabeled-NCBI.csv") %>% dplyr::renam
```

```
## Parsed with column specification:
## cols(
##   PHI = col_double(),
##   log10.PHI = col_double(),
##   Std.Error = col_double(),
##   log10.Std.Error = col_double(),
##   `0.025` = col_double(),
##   `0.975` = col_double(),
##   log10.0.025 = col_double(),
##   log10.0.975 = col_double()
## )
```

```
## Not yet filtered for genes that have anomolous lengths (i.e. length mod 3 !=0)
unfilteredPhiData  <- as_tibble(bind_cols(geneName=unfilteredSeqGeneNames, unfilteredPhiPosteriorInfo, l

phiData <- unfilteredPhiData[properLengthIsoformsFlag, ]

anomolousLengthIsoformsFlag <- !properLengthIsoformsFlag
anomolousGeneInfo <- unfilteredFastaInfoNCBI[anomolousLengthIsoformsFlag, ]
```

```
anomolousPhiData  <-unfilteredPhiData[anomolousLengthIsoformsFlag, ]

## Problem: Isoforms of the same gene exist in the fasta file (and thus phi estimates), but are not par
## Solution: Combine separate estimates using mean or median values of phi

summaryStatsPhiData  <- phiData %>% group_by(geneName) %>% summarize(mean_phi = mean(phi), mean_sd = mea
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
comment(summaryStatsPhiData) <- "summary stats for means and sd of phi for a geneName's multiple isofor
dim(summaryStatsPhiData)
```

```
## [1] 20981    12
```

```
write.csv(unfilteredPhiData, file="Output/labeled.unfiltered.phi.data-NCBI.csv", quote=FALSE, row.names=
    write.csv(unfilteredFastaInfoNCBI, file="Output/unfiltered.fasta.info-NCBI.csv", quote= FALSE, row.
    write.csv(anomolousGeneInfo, "Output/anomolousGeneInfo-NCBI.csv", quote=FALSE, row.names=FALSE)
write.csv(summaryStatsPhiData, file="Output/summaryStatsPhiData-NCBI.csv", quote=FALSE, row.names=FALSE)
save(phiData, seqData, file = "Output/processed.ROC.data-NCBI.Rdata")
```

Import data if `exportData==FALSE`

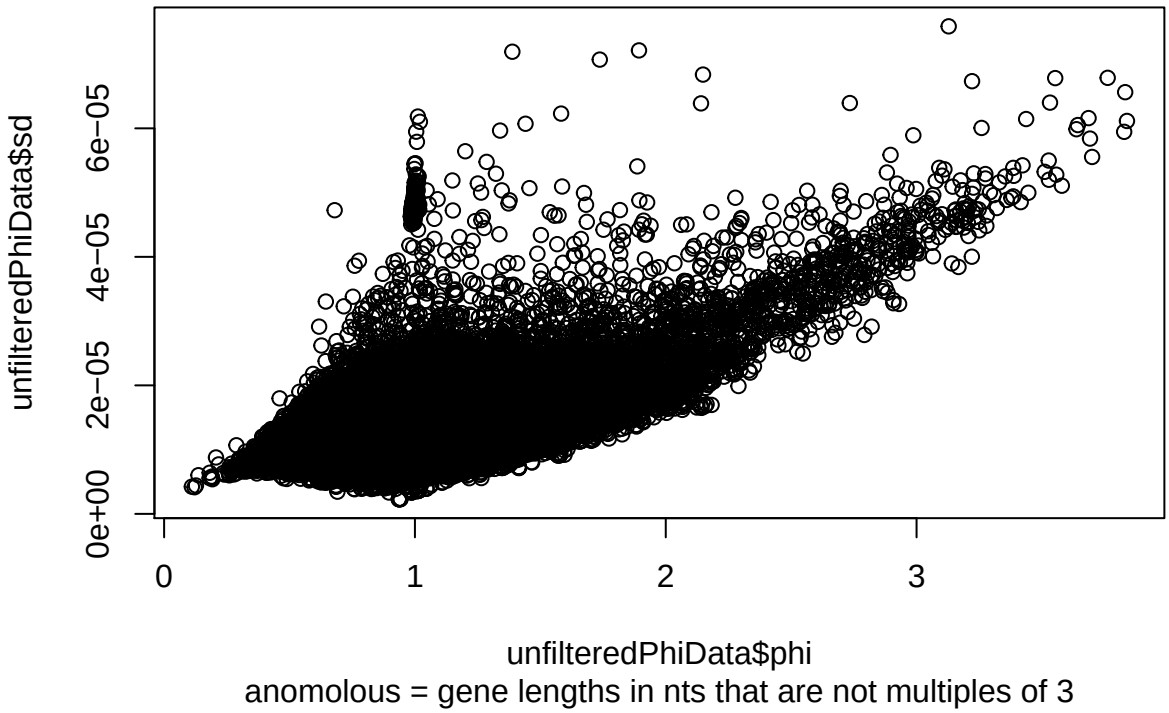
```
## Load data instead of generating and exporting it
read.csv(file="Output/labeled.unfiltered.phi.data-NCBI.csv")
read.csv(file="Output/unfiltered.fasta.info-NCBI.csv")
read.csv("Output/anomolousGeneInfo-NCBI.csv")
read.csv(file="Output/summaryStatsPhiData-NCBI.csv")
load(file = "Output/processed.ROC.data-NCBI.Rdata")
```

Plot Results

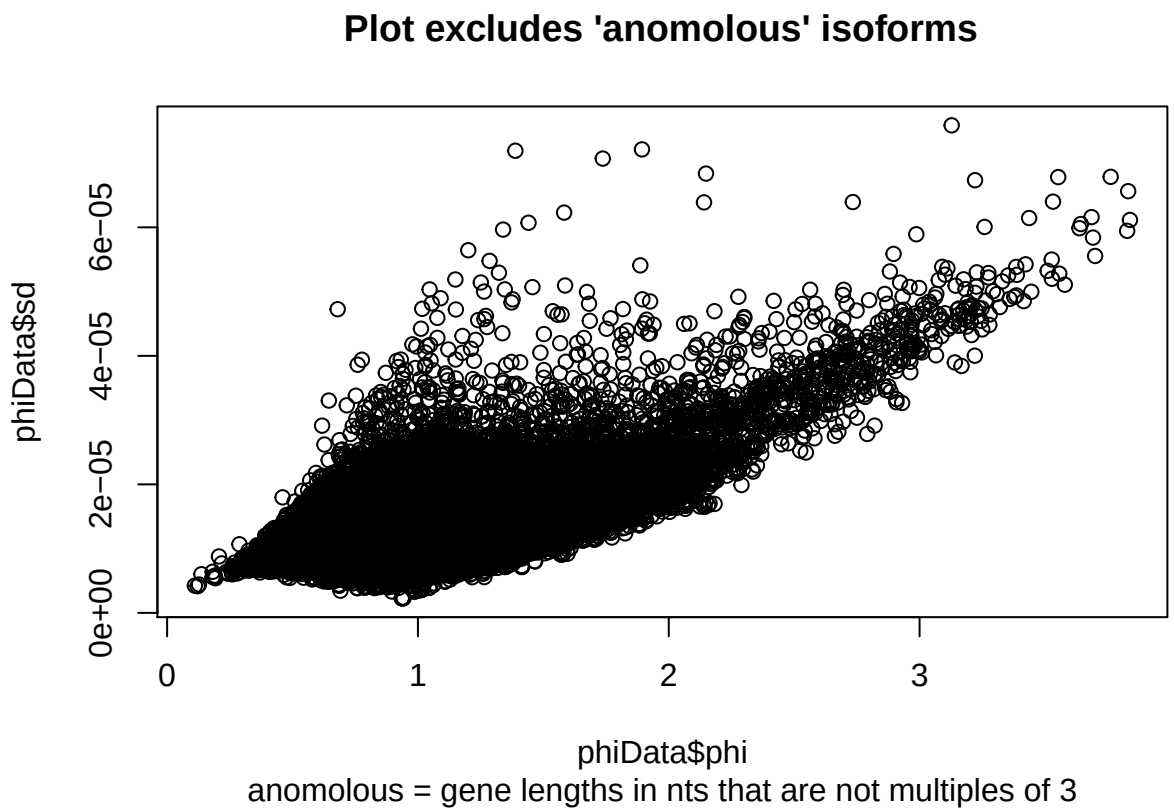
```
## Examine Unfiltered and Filtered Data
dim(unfilteredPhiData)
```

```
## [1] 30167    10
```

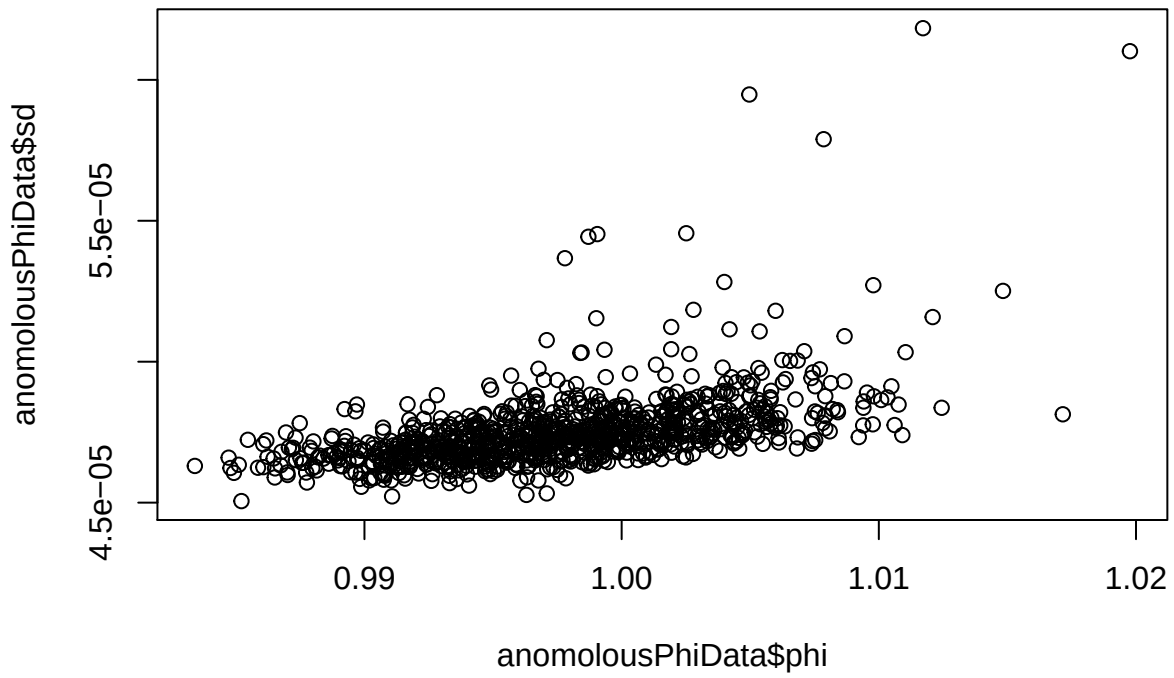
```
plot(unfilteredPhiData$phi, unfilteredPhiData$sd,
     main="Plot Includes 'anomolous' isoforms",
     sub = "anomolous = gene lengths in nts that are not multiples of 3"
     )
```




```
## Verify we've filtered correctly
plot(phiData$phi, phiData$sd, main="Plot excludes 'anomolous' isoforms",
     sub = "anomolous = gene lengths in nts that are not multiples of 3"
     )
```

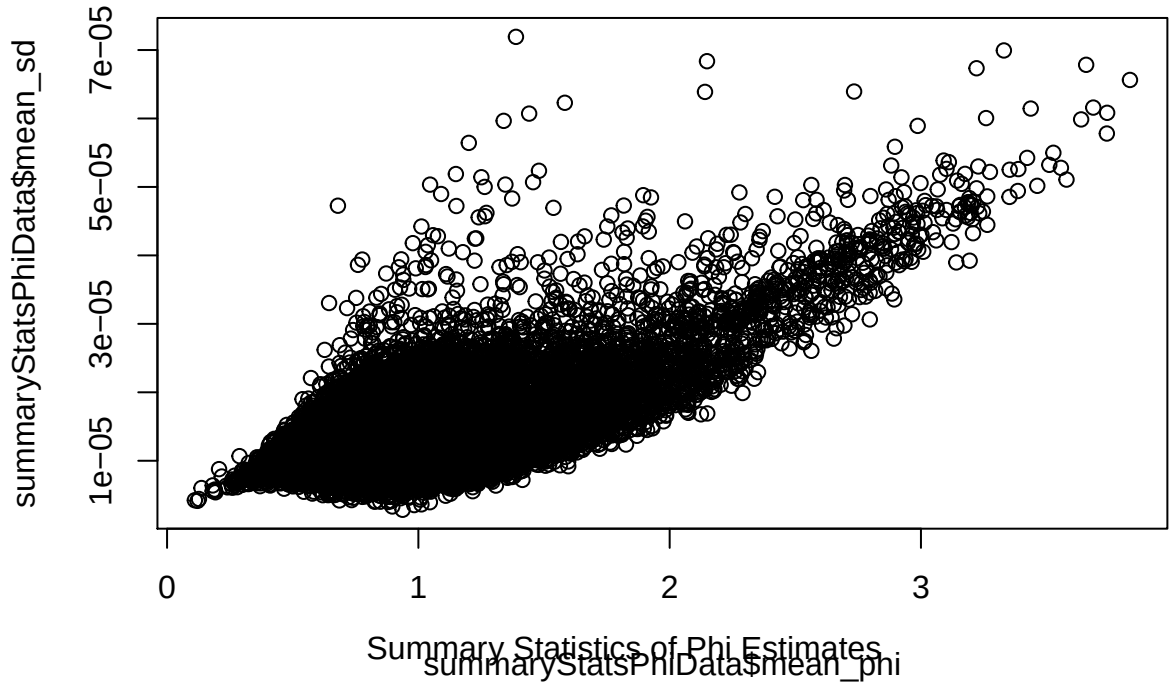



```
plot(anomolousPhiData$phi, anomolousPhiData$sd, main="Isoforms Whose Lengths are Not Multiples of Three
```

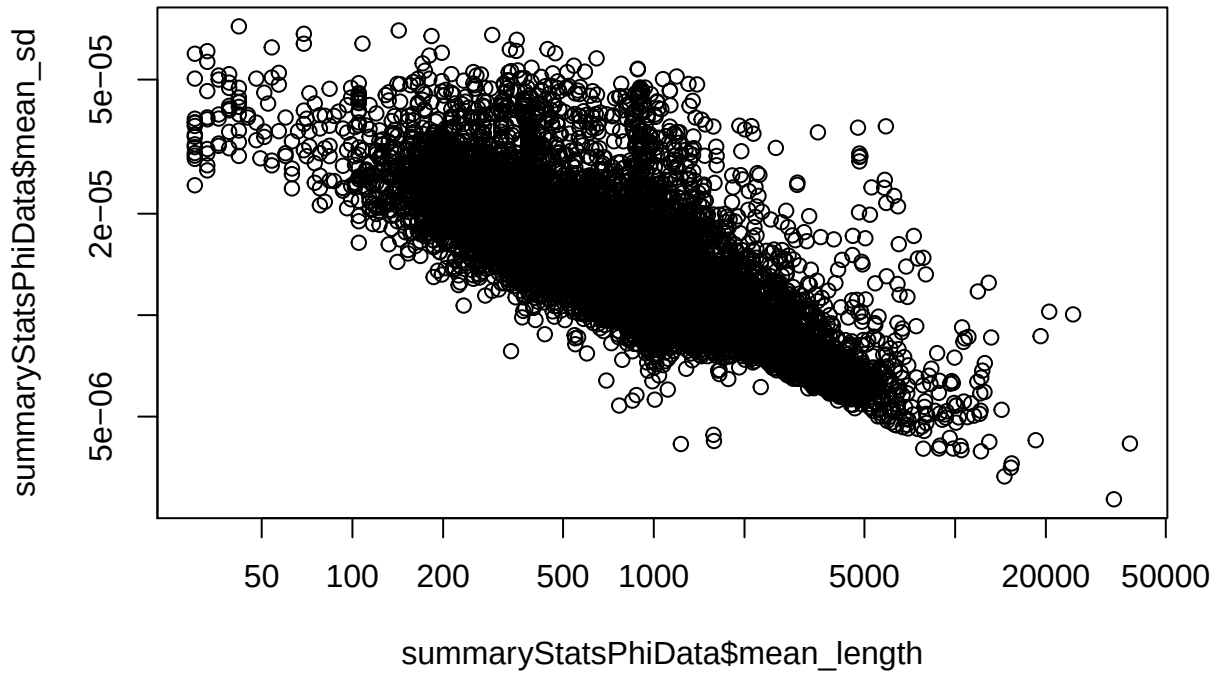



```
#par(mfrow=c(2,2))
plot(summaryStatsPhiData$mean_phi, summaryStatsPhiData$mean_sd)
```

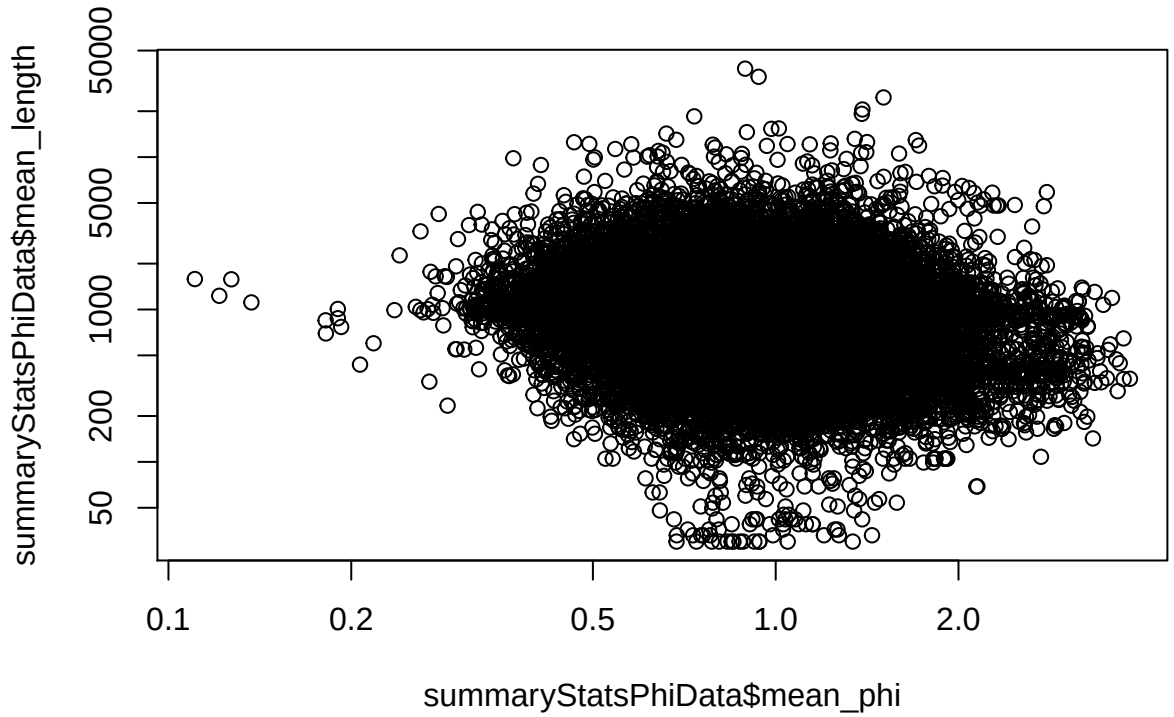
```
mtext("Summary Statistics of Phi Estimates", side = 3, line = -21, outer = TRUE)
```



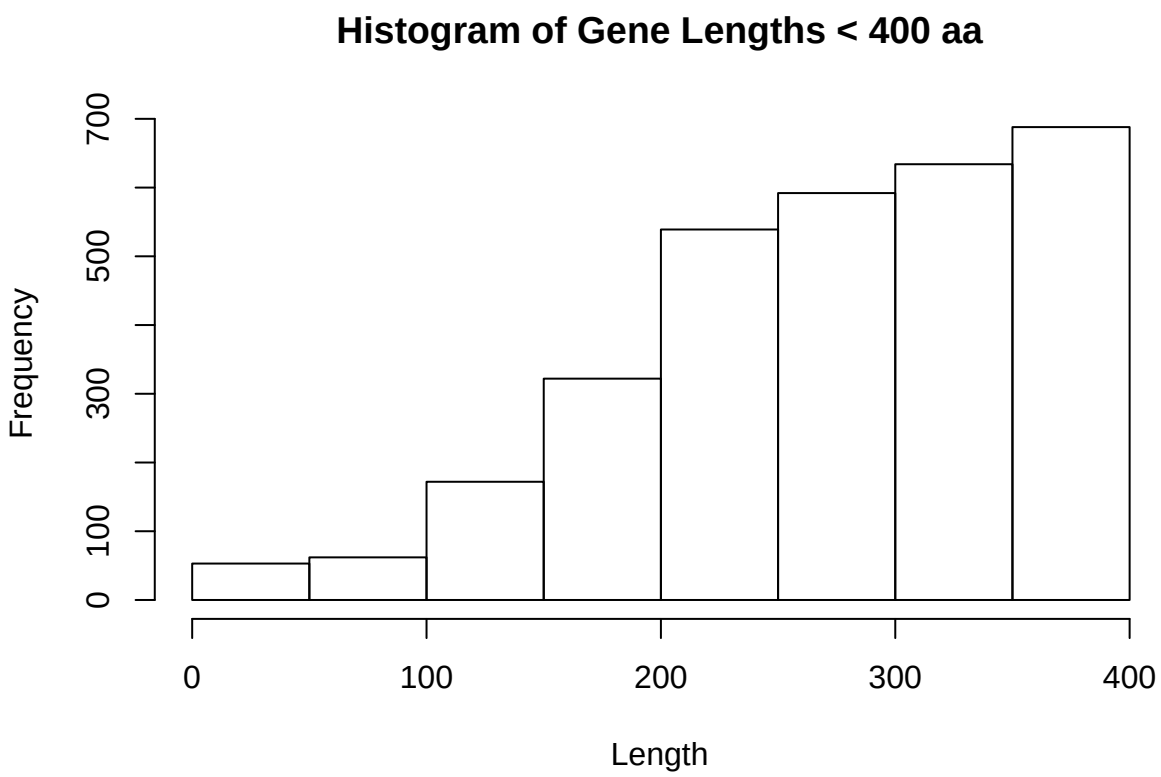

```
plot(summaryStatsPhiData$mean_length, summaryStatsPhiData$mean_sd, log="xy")
```




```
plot(summaryStatsPhiData$mean_phi, summaryStatsPhiData$mean_length, log="xy")
```

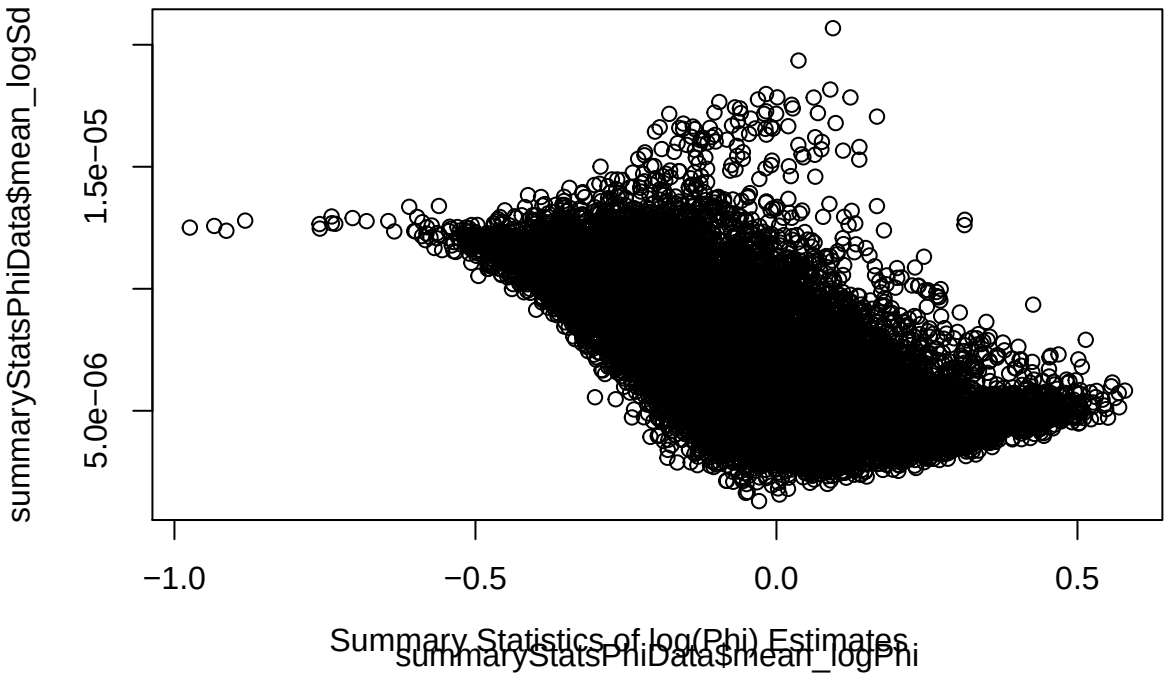



```
## Create histogram of short genes
tmp <- summaryStatsPhiData$mean_length[ summaryStatsPhiData$mean_length < 400]
hist(tmp, xlim=c(0, max(tmp)), main="Histogram of Gene Lengths < 400 aa", xlab="Length" )
```

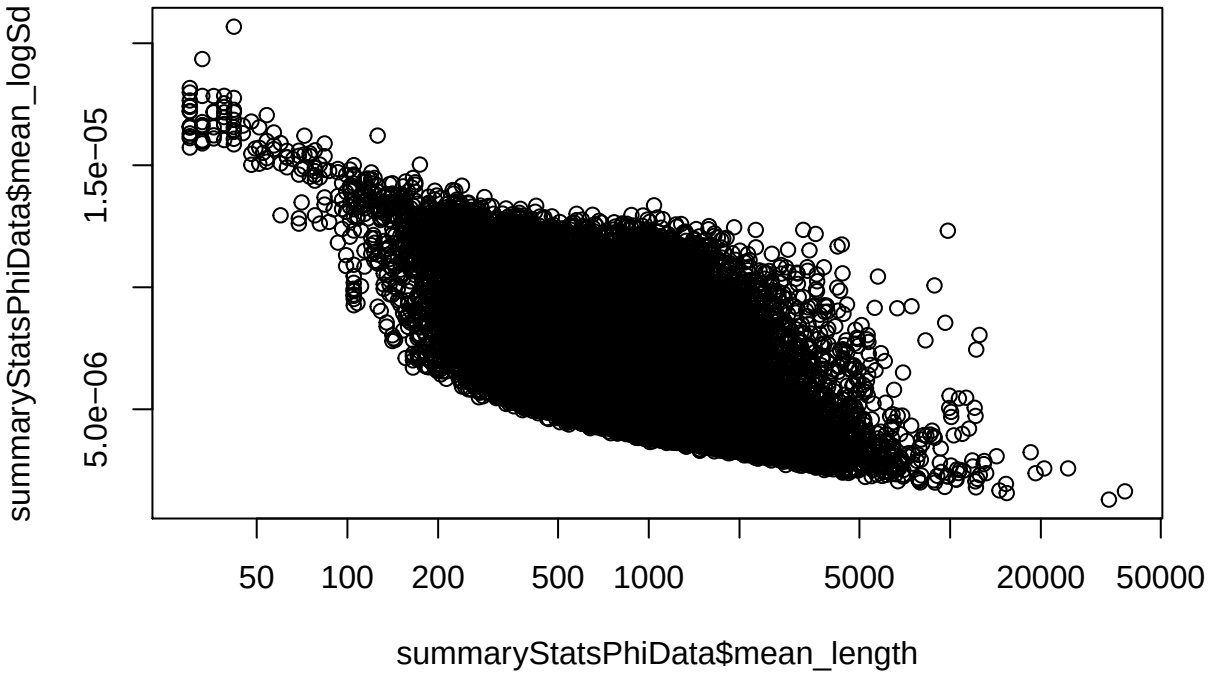



```
## Log metrics
#par(mfrow=c(2,2))
plot(summaryStatsPhiData$mean_logPhi, summaryStatsPhiData$mean_logSd)
```

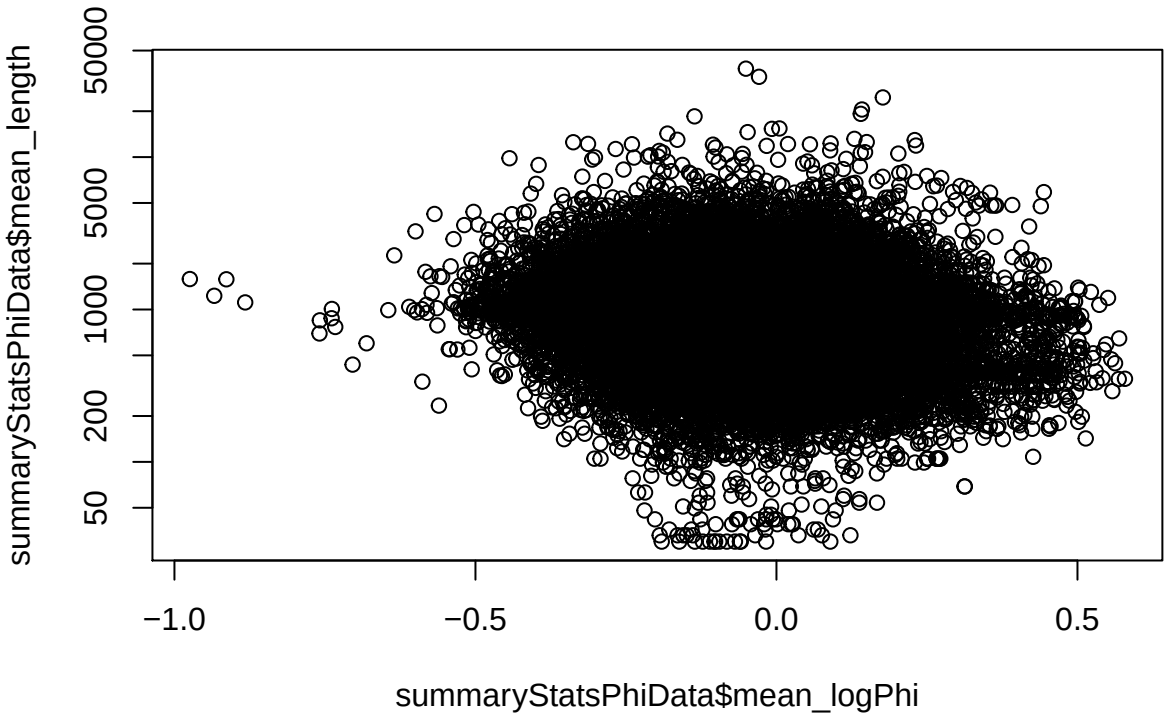
```
mtext("Summary Statistics of log(Phi) Estimates", side = 3, line = -21, outer = TRUE)
```




```
plot(summaryStatsPhiData$mean_length, summaryStatsPhiData$mean_logSd, log="x")
```




```
plot(summaryStatsPhiData$mean_logPhi, summaryStatsPhiData$mean_length, log="y")
```



Import Phi Values based on NCBI File

- Don't use Lu's file

```
## NOTE: rocNames is corrupt.  It has replaced some gene names with date formats (e.g. apr-1 has been c
rocNamesBroken <- read_csv("Input/phi.mean.by.names-NCBI-corrupted.csv", col_names = c("geneName","phi2
rocNamesBroken[rocNamesBroken$geneName=='1-Apr',]
## THere are 17 corrupted names
length(unfilteredSeqGeneNames) - sum(rocNamesBroken==unfilteredSeqGeneNames)

## Need to run 'Get estimates from ROC' code below to evaluate following line
phiBrokenDataCheck  <- bind_cols(rocNamesBroken, unfilteredPhiPosteriorInfo)
## Verify that phi values line up between the two datasets
plot(phiBrokenDataCheck$phi2, phiDataBrokenCheck$phi)
```

E-MTAB

Load Data

```
emtabFile <- "Input/E-MTAB-2812-query-results.tpms.tsv"
flatData  <- readr::read_tsv(emtabFile, skip=4) %>%
    dplyr::rename(WBID = `Gene ID`, geneName=`Gene Name` ) ## WBID is the WormBaseID
```

```
## Parsed with column specification:
## cols(
##   .default = col_double(),
##   `Gene ID` = col_character(),
##   `Gene Name` = col_character()
## )
```

```
## See spec(...) for full column specifications.
```

```
## use pivot_longer command (not gather which is depricated)
tmpData <- flatData %>% pivot_longer(-c(WBID, geneName), names_to = "long_description", values_to = "co


## Now separate entry in descriptor column into separate column entries and filter out tissue specific
tmpFastaGeneName <- unique(fastaInfoNCBI$geneName)
countData  <- separate(tmpData, long_description, into=c("sex", "tissue", "stage"), sep=", ") %>% filte
    filter(geneName %in% tmpFastaGeneName) %>%
    replace_na(list(count = 0)) ## replace NAs with 0.
## Before doing so the min value in the tibble was 0.1

comment(countData) <- paste0("Data from ", emtabFile, "in tidy format. Empty cells filled with '0', gen
```

Create stageCounts

Create data: sex='hermaphrodite' stage='embryo Ce'.

- Data includes a new set of summary measurements of gene expression from the multitude of data from hermphroditic, embryonic stages
- This data is created from the mean values of all sex = 'hermaphrodite' and stage = '* embryo Ce' data
- See file `plot.moments.of.embryo.data.Rmd` for analysis of this data.

```
## Get hermphrodite embryo stage names
## NB: male, 'embryo Ce' lacks a space at the start
embryoStageName  <- unique(countData$stage[grepl(' embryo Ce', countData$stage )])

separateEmbryoStageCount  <-  filter(countData, stage %in% embryoStageName)

##
## Make things simple and focus on just the 'main' lifestage measurements
## note that the stage "embryo Ce" is specific to sex = male
## Thus need to combine embryo measurements some how
##
## Only include data columns well find elsewhere
## See separate chunk for looking at moments of data
embryoStageCount <- separateEmbryoStageCount %>%
    group_by(WBID) %>%
    summarise(geneName, sex, tissue, stage="embryo Ce", count =mean(count), .groups="drop") %>%
    unique()
#### Combine Counts Across Stages

nonEmbryoStageCount <- filter(countData, !(stage %in% embryoStageName)) %>% select(-tissue)

stageCount <- bind_rows(embryoStageCount, nonEmbryoStageCount)
comment(stageCount) <-  paste0("Data from ", emtabFile, "in tidy format. Empty cells filled with '0'\n"

## Be sure to use count = func(count) last in summarise().
## Otherwise it will redefine count to a single value which will screw up the other functions
## Use mean() rather than median for summarizing count
## It seems less sensitive to 0 values
embryoStageCountMoments <- separateEmbryoStageCount %>%
    group_by(WBID) %>%
```

```
    summarise(mean=mean(count), median=median(count), sd=sd(count), logMean=log(mean), logSd = log(sd),

comment(embryoStageCountMoments) <- paste0("Various measures of mean and variation in gene counts across

## This will include both sex = c('male', 'hermaphrodite')
## Really only lacks 'newly molted young adult hermaphrodite Ce'

lifeStages <- c("embryo Ce", "L1 larva Ce", "L2 larva Ce", "L3 larva Ce", "L4 larva Ce", "adult Ce", "da
comment(lifeStages)=paste0("List of main lifestages to use in regression.\n", creationInfo)

## Note that 'count' are standardized means of biological and technical replicates
## 'organism' scale data for different lifestages
## Includes data for 'sex' =  male and hermphrodite
lifeStageCount <- filter(stageCount, (stage %in% lifeStages))
comment(lifeStageCount) <- c("RNASeq counts for each of the\n\t- hermaphrodite non-embryo lifestages: L

lifeStageCountMoments <- lifeStageCount %>% group_by(sex, stage)%>% summarize(stage_sd = sd(count), stag

## `summarise()` regrouping output by 'sex' (override with `.groups` argument)
geneNameAndWormBaseID <- unique(select(lifeStageCount, c(geneName, WBID)))
comment(geneNameAndWormBaseID) <- paste("Mapping between WormBase IDs and geneName in FASTA file")

## Export values
## Don't use write_csv because that only works with dataframes,not tibbles
## write.csv converts the tibble to a dataframe


write.csv(separateEmbryoStageCount, file = "Output/separate.embryo.stage.count.csv", quote=FALSE, row.na
write.csv(embryoStageCountMoments, file="Output/summary.stats.of.embryo.stage.data.csv", quote=FALSE, ro
## Export data using 'geneName' name or WormBaseID
write.csv(geneNameAndWormBaseID, file="Output/geneName.and.WBID.csv", quote=FALSE, row.names=FALSE)

save(countData, file = "Output/countData.from.E-MTAB-data.Rdata")
save(lifeStageCount, lifeStages, file = "Output/processed.E-MTAB.data.Rdata")
```

Import the data instead of generating it.

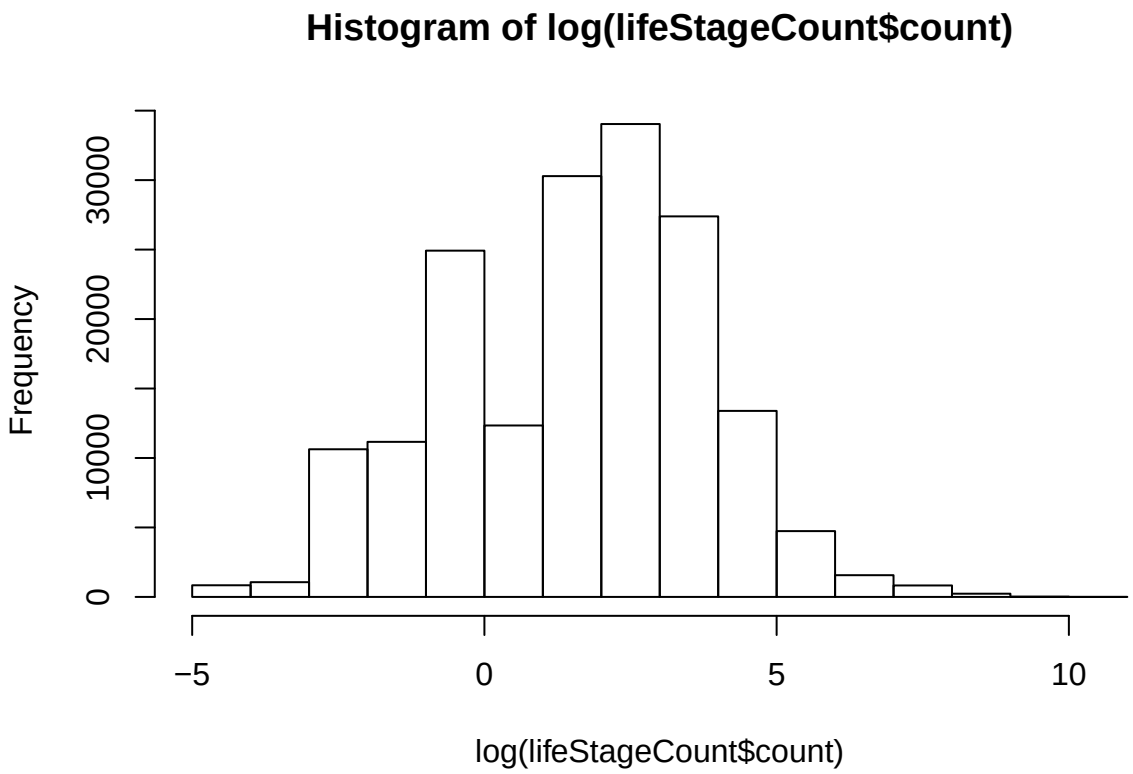
```
load("Output/countData.from.E-MTAB-data.Rdata")
read.csv(unfilteredPhiData, file="Output/summary.stats.of.embryo.stage.data.csv")
read.csv(file="Output/summary.stats.of.embryo.stage.data.csv")
## Export data using 'geneName' name or WormBaseID
write.csv(geneNameAndWormBaseID, file="Output/geneName.and.WBID.csv", quote=FALSE, row.names=FALSE)

load(file = "Output/countData.from.E-MTAB-data.Rdata")
load(file = "Output/processed.E-MTAB.data.Rdata")
```

```
## Plot stuff
summary(lifeStageCount$count)
```

```
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
##     0.00     0.30     4.00    36.94    19.00 36316.00
```

```
## hist(lifeStageCount$count)
hist(log(lifeStageCount$count))
```



Create database linking geneName and WormBaseID

Evaluate these lines manually outside of knitr

Combine empirical E-MTAB and ROC data

- This used to work, but doesn't now. Not worth fixing right now.

```
## Combine phi

phiNames <- summaryStatsPhiData$geneName

namesToWBID <- lapply(phiNames, function(x) {index = which(geneNameAndWormBaseID$geneName == x); ifelse

unmatchedPhiNames <- phiNames[is.na(namesToWBID)]
geneName <- geneNameAndWormBaseID$geneName
WBID <- geneNameAndWormBaseID$WBID



tidyJoinedData <- bind_rows(lifeStageCount, tidyFilteredAndWeightedPhiData) %>%
    mutate_if(is.character, as.factor)

joinedData <- pivot_wider(tidyJoinedData, names_from=stage, values_from=c(count, weight))
```