1 piRNA and Degradome Count Generation

Bryan Teefy

08/09/2019

Load Required Libraries

```
library(dplyr)
library(reshape2)
library(ggplot2)
library(ggpubr)
```

Classifying Transcripts in the *Hydra* Transcriptome

To determine the category of transcript to which piRNAs and degradome reads align, transcripts were classified as TEs, ncRNAs, uncharacterized transcripts, and genes.

The *Hydra* transcriptome was BLASTed against the *Hydra* Repbase, Swissprot, and nr databases with an e-value of 1e-5.

HMMER suite 3.1b2 (February 2015, http://www.hmmer.org/) and Pfam v31.0 database were used to identify protein domains in the transcriptome using an e-value of 1e-6.

Uniprot protein descriptions were added to any transcripts that had a match in the Swissprot database using Uniprot's Retrieve ID/mapping tool (https://www.uniprot.org/uploadlists/).

Open reading frames were identified using Transdecoder.

Results are summarized in Table S1.

Load Transcriptome Annotation Matrix

```
Transcript_Characterization <- read.table("objects/Transcriptome_Annotation_Matrix.txt",
    sep = "\t", check.names = FALSE, header = TRUE)
```

Transposon Annotation

Transcripts that met the following criteria were classified as TEs:

Transcripts with significant similarities to entries in the Repbase database.

Transcripts with Swissprot protein descriptions or nr sequence descriptions containing the strings "transpos", "J/jerky", and "mobile element".

Transcripts with Pfam domain descriptions predicted to encode domains containing "transposase", "THAP", "DDE_Tnp", "_Tnp" or "*tnp*".

non-coding RNA (ncRNA) Annotation

We considered sequences non-coding RNAs if they were lacking TE annotation, Swissprot hit, nr hit, known PFAM domain, and an ORF equal to or greater than 100 amino acids. ORFs were predicted using Transdecoder using command TransDecoder.LongOrfs -S -t.

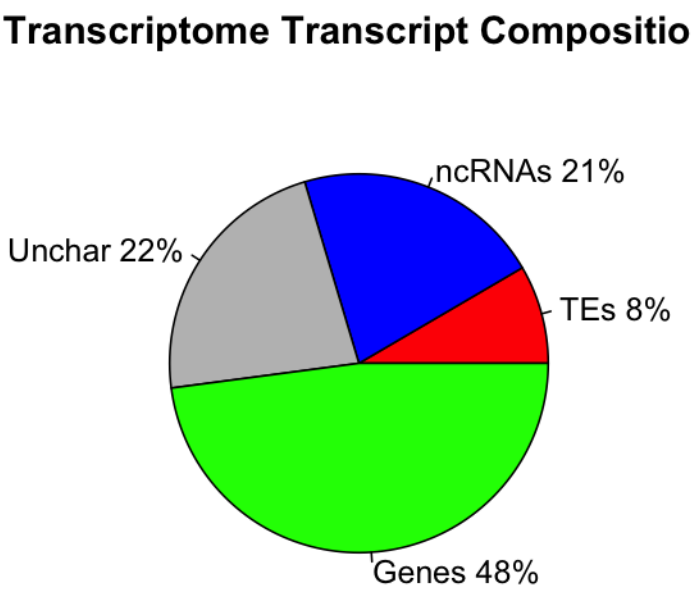
Taxonomically Restricted Genes (TRGs)/Uncharacterized Genes

Uncharacterized Genes were defined as transcripts predicted to contain an ORF equal to or greater than 100 amino acids without a Swissprot Hit, nr hit, known domain, or TE annotation. Nr hits termed "uncharacterized protein" were also considered in this category.

Gene Annotation

Genes were defined as transcripts with a Swissprot hit, nr hit, or domain annotation, and that were not classified as TEs by our annotation.

```
#Classify transcripts

Transcript_Characterization$Transcript_Class <- ifelse((

  !is.na(Transcript_Characterization$Repbase_Hit) | grepl(("transpos"), Transcript_Characterization$Uni
    grepl(("mobile element"), Transcript_Characterization$Uniprot_Description, ignore.case = TRUE) |
    grepl(("jerky"), Transcript_Characterization$Uniprot_Description, ignore.case = TRUE) |
    grepl(("transpos"), Transcript_Characterization$nr_Hit, ignore.case = TRUE) |
    grepl(("mobile element"), Transcript_Characterization$nr_Hit, ignore.case = TRUE) |
    grepl(("jerky"), Transcript_Characterization$nr_Hit, ignore.case = TRUE) |
    grepl(("Transposase"), Transcript_Characterization$PFAM_Annotation, ignore.case = TRUE) |
    grepl(("THAP"), Transcript_Characterization$PFAM_Annotation, ignore.case = TRUE) |
    grepl(("_Tnp_"), Transcript_Characterization$PFAM_Annotation, ignore.case = TRUE) |
    grepl(("DDE_Tnp"), Transcript_Characterization$PFAM_Annotation, ignore.case = TRUE) |
    grepl(("_Tnp"), Transcript_Characterization$PFAM_Annotation, ignore.case = TRUE)),"TE",

    ifelse(is.na(Transcript_Characterization$ORF) & is.na(Transcript_Characterization$Uniprot_Descriptio

    ifelse((!is.na(Transcript_Characterization$ORF) & is.na(Transcript_Characterization$Uniprot_Descrip

#Visualize Transcriptome Breakdown

transcriptome_annotation_whole <- c(sum(Transcript_Characterization$Transcript_Class == "TE"),sum(Transc

lbls <- c("TEs", "ncRNAs", "Unchar", "Genes")
colors = c("red", "blue", "gray", "green")
pct <- round(transcriptome_annotation_whole/sum(transcriptome_annotation_whole)*100)
lbls <- paste(lbls, pct) # add percents to labels
lbls <- paste(lbls,"%",sep="") # ad % to labels
pie(transcriptome_annotation_whole,labels = lbls, col=colors,
    main="Transcriptome Transcript Composition")
```



Generating piRNA and Degradome counts

Since piRNAs are complementary to RNA transcript targets (antisense to the target) or derived directly from targets (sense to the target), we used piRNA mapping to identify these targets in the *Hydra* transcriptome.

Adapters were trimmed from the WT and Colchicine raw reads using the script trimbiooadapter.sh.

piRNAs were mapped to the *Hydra* transcriptome using RSEM functions rsem-calculate-expression and rsem-generate-data-matrix was used to generate the count matrix. Three mismatches were allowed in the antisense orientation and no mismatches were allowed in the sense orientation. Degradome reads were mapped in the sense orientation with no mismatches since degradome reads are transcript fragments.

piRNA_Deg_rsem_mapping.sh was used for piRNA and Degradome Mapping.

Results are summarized in the table, "piRNA_Counts_Matrix.txt" and "Degradome_Counts_Matrix.txt", which can be found in the GEO repository (GSE135440).

Load and Merge the piRNA and Degradome Count Files

```
piRNA_counts <- read.table("objects/piRNA_Counts_Matrix.txt", sep = "\t", header = T)

Deg_counts <- read.table("objects/Degradome_Counts_Matrix.txt", sep = "\t", header = T)

piRNA_Deg_counts <- merge(piRNA_counts, Deg_counts, by = "ID")
```

```
# Merge with Count Files

piRNA_Deg_counts <- merge(Transcript_Characterization, piRNA_Deg_counts, by = "ID")
```

Generate Normalized piRNA Counts

PIWI targets should have a high density of piRNA counts. We normalize piRNA counts by transcript length to determine piRNA count density.

To determine if the piRNA count density values were significantly different between classes of transcripts, we performed Tukey's Honest Significant Difference test to compare mean piRNA count density between each transcript type (i.e. TE, ncRNA, Unchar., Gene) for each piRNA class (i.e. Hywi Antisense-mapped, Hyli Sense-mapped).

```
# Determine count density by dividing counts by transcrpt length in kilobases

norm <- (piRNA_Deg_counts$Length/1000)

piRNA_Deg_counts$WT_Hyli_AS_kb <- piRNA_Deg_counts$WT_Hyli_AS/norm
piRNA_Deg_counts$WT_Hyli_S_kb <- piRNA_Deg_counts$WT_Hyli_S/norm
piRNA_Deg_counts$WT_Hywi_AS_kb <- piRNA_Deg_counts$WT_Hywi_AS/norm
piRNA_Deg_counts$WT_Hywi_S_kb <- piRNA_Deg_counts$WT_Hywi_S/norm
piRNA_Deg_counts$WT_Deg_S_kb <- piRNA_Deg_counts$WT_Deg_S/norm
piRNA_Deg_counts$Colch_Hyli_AS_kb <- piRNA_Deg_counts$Colch_Hyli_AS/norm
piRNA_Deg_counts$Colch_Hyli_S_kb <- piRNA_Deg_counts$Colch_Hyli_S/norm
piRNA_Deg_counts$Colch_Hywi_AS_kb <- piRNA_Deg_counts$Colch_Hywi_AS/norm
piRNA_Deg_counts$Colch_Hywi_S_kb <- piRNA_Deg_counts$Colch_Hywi_S/norm
piRNA_Deg_counts$Colch_Deg_S_kb <- piRNA_Deg_counts$Colch_Deg_S/norm

# Perform Tukey's Honest Significant Difference test

# Group normalized mapping counts
Normalized_Mapping_Counts_Matrix <- piRNA_Deg_counts[, c(22:25, 27:30, 11)]
Normalized_Mapping_Counts_Matrix_Formatted <- melt(Normalized_Mapping_Counts_Matrix,
    id.var = "Transcript_Class")

# Subset count density based on piRNA origin
WT_Hyli_AS_Stats <- subset(Normalized_Mapping_Counts_Matrix_Formatted, variable ==
    "WT_Hyli_AS_kb")
WT_Hyli_S_Stats <- subset(Normalized_Mapping_Counts_Matrix_Formatted, variable ==
    "WT_Hyli_S_kb")
WT_Hywi_AS_Stats <- subset(Normalized_Mapping_Counts_Matrix_Formatted, variable ==
    "WT_Hywi_AS_kb")
WT_Hywi_S_Stats <- subset(Normalized_Mapping_Counts_Matrix_Formatted, variable ==
    "WT_Hywi_S_kb")

Colch_Hyli_AS_Stats <- subset(Normalized_Mapping_Counts_Matrix_Formatted, variable ==
    "Colch_Hyli_AS_kb")
Colch_Hyli_S_Stats <- subset(Normalized_Mapping_Counts_Matrix_Formatted, variable ==
    "Colch_Hyli_S_kb")
Colch_Hywi_AS_Stats <- subset(Normalized_Mapping_Counts_Matrix_Formatted, variable ==
    "Colch_Hywi_AS_kb")
```

```r
Colch_Hywi_S_Stats <- subset(Normalized_Mapping_Counts_Matrix_Formatted, variable ==
    "Colch_Hywi_S_kb")

# Develop Tukey Test Function (ANOVA post hoc test)

Tukey_Test <- function(x) {
    res.aov <- aov(value ~ Transcript_Class, data = x)
    return(TukeyHSD(res.aov))
}

# Run Tukey Test

Tukey_Test(WT_Hyli_AS_Stats)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = value ~ Transcript_Class, data = x)
##
## $Transcript_Class
##                     diff         lwr        upr     p adj
## ncRNA-Gene     380.1172   127.98864   632.2457 0.0006214
## TE-Gene       2003.2537  1641.98790  2364.5194 0.0000000
## Unchar-Gene    264.9371    17.73571   512.1385 0.0300726
## TE-ncRNA      1623.1365  1229.34925  2016.9237 0.0000000
## Unchar-ncRNA  -115.1801  -407.86415   177.5040 0.7429976
## Unchar-TE    -1738.3166 -2128.96747 -1347.6657 0.0000000
```

```r
Tukey_Test(WT_Hyli_S_Stats)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = value ~ Transcript_Class, data = x)
##
## $Transcript_Class
##                     diff         lwr        upr     p adj
## ncRNA-Gene     611.2690   293.53321   929.0049 0.0000046
## TE-Gene       2817.3087  2362.03660  3272.5808 0.0000000
## Unchar-Gene    991.6493   680.12272  1303.1759 0.0000000
## TE-ncRNA      2206.0396  1709.78354  2702.2957 0.0000000
## Unchar-ncRNA   380.3803    11.53577   749.2247 0.0402490
## Unchar-TE    -1825.6594 -2317.96301 -1333.3558 0.0000000
```

```r
Tukey_Test(WT_Hywi_AS_Stats)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = value ~ Transcript_Class, data = x)
##
## $Transcript_Class
##                     diff         lwr        upr     p adj
## ncRNA-Gene     438.5738   287.65972   589.4878 0.0000000
## TE-Gene       2380.8399  2164.60063  2597.0791 0.0000000
```

```
## Unchar-Gene    655.0570   507.09215   803.0219 0.0000000
## TE-ncRNA      1942.2661  1706.56080  2177.9714 0.0000000
## Unchar-ncRNA   216.4832    41.29426   391.6722 0.0081737
## Unchar-TE    -1725.7829 -1959.61086 -1491.9548 0.0000000
```

```
Tukey_Test(WT_Hywi_S_Stats)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = value ~ Transcript_Class, data = x)
##
## $Transcript_Class
##                     diff         lwr        upr     p adj
## ncRNA-Gene     25.60785  -21.999540   73.21524 0.5107346
## TE-Gene       429.48593  361.271034  497.70082 0.0000000
## Unchar-Gene    48.24260    1.565567   94.91964 0.0396104
## TE-ncRNA      403.87808  329.522416  478.23374 0.0000000
## Unchar-ncRNA   22.63475  -32.630411   77.89992 0.7186042
## Unchar-TE    -381.24332 -455.006771 -307.47987 0.0000000
```

```
Tukey_Test(Colch_Hyli_AS_Stats)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = value ~ Transcript_Class, data = x)
##
## $Transcript_Class
##                      diff         lwr       upr     p adj
## ncRNA-Gene     215.40525    68.89178  361.9187 0.0009129
## TE-Gene       1238.53260  1028.59878 1448.4664 0.0000000
## Unchar-Gene    169.34452    25.69424  312.9948 0.0131238
## TE-ncRNA      1023.12734   794.29510 1251.9596 0.0000000
## Unchar-ncRNA   -46.06073  -216.14128  124.0198 0.8987613
## Unchar-TE    -1069.18807 -1296.19777 -842.1784 0.0000000
```

```
Tukey_Test(Colch_Hyli_S_Stats)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = value ~ Transcript_Class, data = x)
##
## $Transcript_Class
##                     diff        lwr       upr     p adj
## ncRNA-Gene     16.62708  -86.87629  120.1305 0.9763053
## TE-Gene       762.39902  614.09282  910.7052 0.0000000
## Unchar-Gene   213.11326  111.63257  314.5939 0.0000004
## TE-ncRNA      745.77194  584.11507  907.4288 0.0000000
## Unchar-ncRNA  196.48617   76.33401  316.6383 0.0001558
## Unchar-TE    -549.28577 -709.65511 -388.9164 0.0000000
```

```
Tukey_Test(Colch_Hywi_AS_Stats)
```

```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
```

```
##
## Fit: aov(formula = value ~ Transcript_Class, data = x)
##
## $Transcript_Class
##                      diff         lwr        upr     p adj
## ncRNA-Gene     312.40992    54.84956   569.9703 0.0099067
## TE-Gene       3260.65993  2891.61109  3629.7088 0.0000000
## Unchar-Gene    357.13604   104.60898   609.6631 0.0015942
## TE-ncRNA      2948.25001  2545.97905  3350.5210 0.0000000
## Unchar-ncRNA    44.72612  -254.26350   343.7157 0.9807061
## Unchar-TE    -2903.52388 -3302.59092 -2504.4568 0.0000000
```

```
Tukey_Test(Colch_Hywi_S_Stats)
```

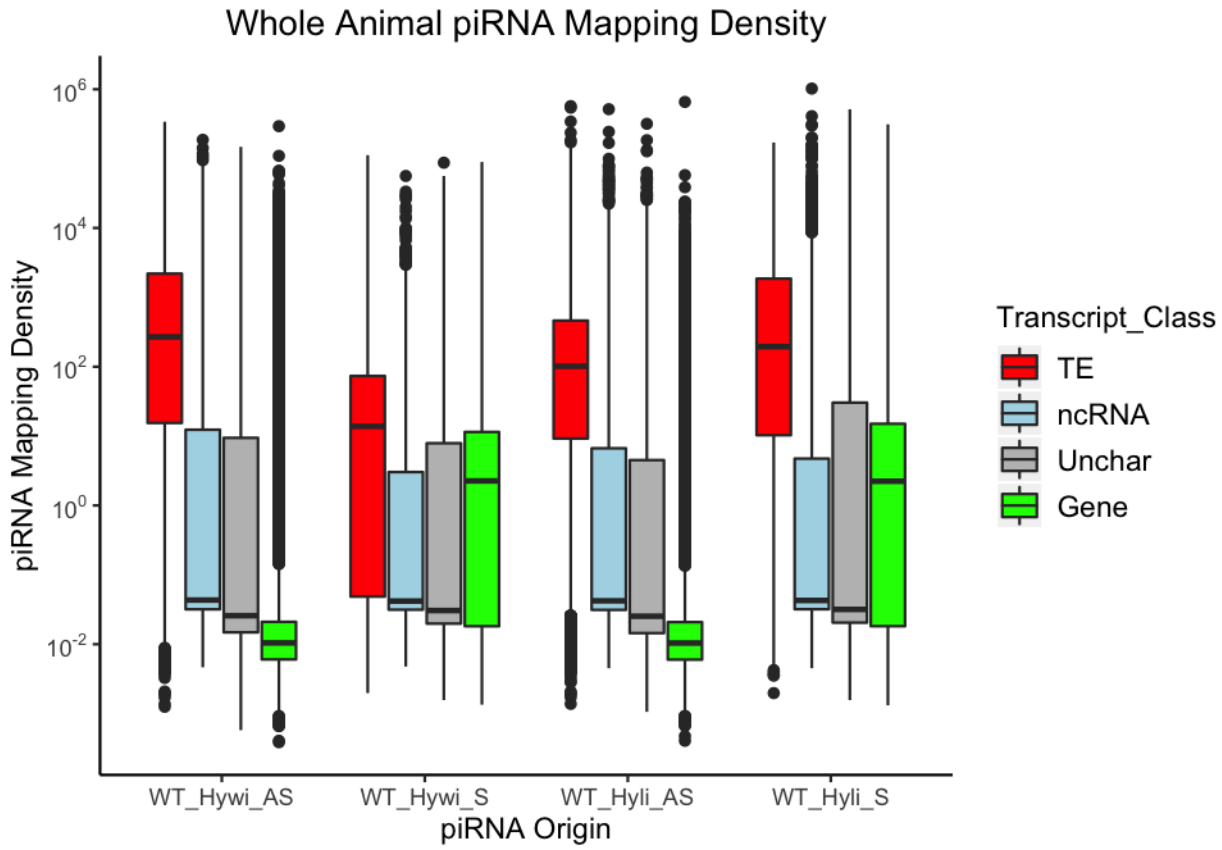
```
##   Tukey multiple comparisons of means
##     95% family-wise confidence level
##
## Fit: aov(formula = value ~ Transcript_Class, data = x)
##
## $Transcript_Class
##                      diff       lwr        upr     p adj
## ncRNA-Gene    -60.003234 -161.3535   41.34708 0.4248261
## TE-Gene       323.611889  178.3907  468.83306 0.0000001
## Unchar-Gene   -54.290610 -153.6603   45.07910 0.4970390
## TE-ncRNA      383.615123  225.3210  541.90923 0.0000000
## Unchar-ncRNA    5.712624 -111.9402  123.36540 0.9993069
## Unchar-TE    -377.902499 -534.9359 -220.86913 0.0000000
```

Visualizing piRNA Mapping

Since the range of observed count density values was large, we used a log scale to visualize piRNA count density. For boxplot visualization, we added a pseudocount to the raw piRNA counts to remove any 0 count density values that would return infinite values on a log scale. The pseudocount we chose was 0.01 since that was the lowest fractional count used in our counting strategy. We explored piRNA count density for 1) Whole animals and 2) Epithelial Animals.

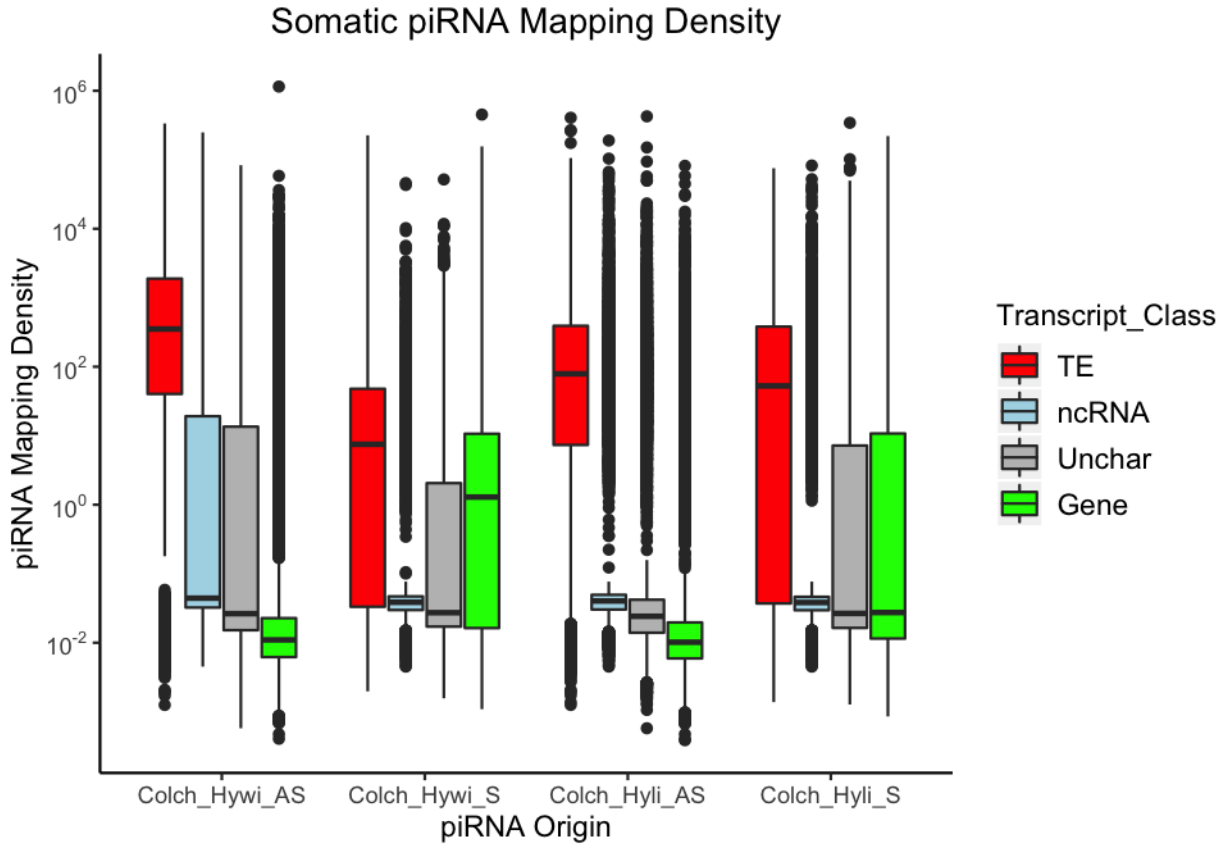
```
# Set pseudocount

pseudocount <- 0.01

# Add pseudocount to raw piRNA counts then generate piRNA count density values

boxplot_matrix <- piRNA_Deg_counts[, c(12:19, 3, 11)]
boxplot_matrix[, c(1:8)] <- boxplot_matrix[, c(1:8)] + pseudocount
boxplot_matrix[, c(1:8)] <- boxplot_matrix[, c(1:8)]/(boxplot_matrix$Length/1000)

# Plot whole animal piRNA count density values

wt_matrix_data <- boxplot_matrix[, c(10, 1:4)]
wt_matrix_data_formatted <- melt(wt_matrix_data, id.var = "Transcript_Class")
colnames(wt_matrix_data_formatted) <- c("Transcript_Class", "piRNA_Origin", "piRNA_Mapping_Density")
wt_matrix_data_formatted$Transcript_Class <- factor(wt_matrix_data_formatted$Transcript_Class,
    levels = c("TE", "ncRNA", "Unchar", "Gene"))
```

```
WT_level_order <- c("WT_Hywi_AS", "WT_Hywi_S", "WT_Hyli_AS", "WT_Hyli_S")

WT_boxplot <- ggplot(data = wt_matrix_data_formatted, aes(x = factor(piRNA_Origin,
    level = WT_level_order), y = piRNA_Mapping_Density), log = "y") + geom_boxplot(aes(fill = Transcript
    scale_y_log10(breaks = scales::trans_breaks("log10", function(x) 10^x), labels = scales::trans_forma
        scales::math_format(10^.x)))

WT_boxplot + scale_fill_manual(values = c("red", "light blue", "grey", "green")) +
    theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(), axis.line = element_line(colour = "black")) +
    theme(legend.text = element_text(size = rel(1))) + ggtitle("Whole Animal piRNA Mapping Density") +
    theme(plot.title = element_text(hjust = 0.5)) + xlab("piRNA Origin") + ylab("piRNA Mapping Density")
```




```
# Plot epithelial animal piRNA count density values

colch_matrix_data <- boxplot_matrix[, c(10, 5:8)]
colch_matrix_data_formatted <- melt(colch_matrix_data, id.var = "Transcript_Class")
colnames(colch_matrix_data_formatted) <- c("Transcript_Class", "piRNA_Origin", "piRNA_Mapping_Density")
colch_matrix_data_formatted$Transcript_Class <- factor(colch_matrix_data_formatted$Transcript_Class,
    levels = c("TE", "ncRNA", "Unchar", "Gene"))

colch_level_order <- c("Colch_Hywi_AS", "Colch_Hywi_S", "Colch_Hyli_AS", "Colch_Hyli_S")

colch_boxplot <- ggplot(data = colch_matrix_data_formatted, aes(x = factor(piRNA_Origin,
    level = colch_level_order), y = piRNA_Mapping_Density), log = "y") + geom_boxplot(aes(fill = Transc
    scale_y_log10(breaks = scales::trans_breaks("log10", function(x) 10^x), labels = scales::trans_forma
```

```
        scales::math_format(10^.x)))

colch_boxplot + scale_fill_manual(values = c("red", "light blue", "grey", "green")) +
    theme(panel.grid.major = element_blank(), panel.grid.minor = element_blank(),
        panel.background = element_blank(), axis.line = element_line(colour = "black")) +
    theme(legend.text = element_text(size = rel(1))) + ggtitle("Somatic piRNA Mapping Density") +
    theme(plot.title = element_text(hjust = 0.5)) + xlab("piRNA Origin") + ylab("piRNA Mapping Density")
```




```
# write table that summarizes results write.table(piRNA_Deg_counts, file =
# 'Annotated_piRNA_Degradome_Count_Matrix.txt')
```

Software versions

This document was computed on Fri Aug 09 19:21:20 2019 with the following R package versions.

```
R version 3.5.3 (2019-03-11)
Platform: x86_64-apple-darwin15.6.0 (64-bit)
Running under: macOS Mojave 10.14.5

Matrix products: default
BLAS: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRblas.0.dylib
LAPACK: /Library/Frameworks/R.framework/Versions/3.5/Resources/lib/libRlapack.dylib

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
```

```
attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] ggpubr_0.2     magrittr_1.5   ggplot2_3.2.0  reshape2_1.4.3
[5] dplyr_0.8.3    knitr_1.22

loaded via a namespace (and not attached):
 [1] Rcpp_1.0.1       munsell_0.5.0    tidyselect_0.2.5 colorspace_1.4-1
 [5] R6_2.4.0         rlang_0.4.0      stringr_1.4.0    plyr_1.8.4
 [9] tools_3.5.3      grid_3.5.3       gtable_0.3.0     xfun_0.5
[13] withr_2.1.2      htmltools_0.3.6  lazyeval_0.2.2   yaml_2.2.0
[17] assertthat_0.2.1 digest_0.6.20    tibble_2.1.3     crayon_1.3.4
[21] purrr_0.3.2      formatR_1.7      glue_1.3.1       evaluate_0.13
[25] rmarkdown_1.12   stringi_1.4.3    compiler_3.5.3   pillar_1.4.2
[29] scales_1.0.0     pkgconfig_2.0.2
```