Script1: Gene regulation

Sara Beier & Angel Rain

08/05/2021

Contents

1 Gene regulation

1.1 Packages

```
rm(list = ls())
library(readxl)
library(ggplot2)
library(dplyr)
library(tidyr)
library(DESeq2)
library(kableExtra)
# Load color palette
cbbPalette <- c("#000000", "#E69F00", "#56B4E9", "#009E73", "#F0E442", "#0072B2",
    "#D55E00", "#CC79A7")
```

1.2 Calculate transcripts per cell (Satinsky et al. 2013)

```
# Load input data (from Table S6)
vec<-data.frame(read_excel("../data_transc/TableS6.xlsx",sheet ="TableS6"))
row.names(vec) <- paste(vec$Strain.ID, vec$Salinity.level.replicates, sep='_')
mRNA.length <- 1000 # Estimated average length of mRNA transcripts (bases)
vec.length <- 970  # Length of added vector (bases)
vec.MW <- 5.20622384589837E-10 # Molecular weight of vector (ng)
vec.ng <- 5 # Amount of vector added to RNA extract (ng)
vec.added<- vec.ng/vec.MW # Vector molecules added to RNA extract

# Calculate transcripts per cell
vec$n.cells <-                                #
  vec$Cell.per.mL * vec$Volume.medium..ml. # Total number of cells used for RNA extract
vec$mRNA.norm <-
  vec$Mapped.reads/mRNA.length # Normalize transcripts count data by transcript length
vec$vector.normvector.norm <-
  vec$Counts.vectorF/vec.length # Normalize vector count data by vector length
vec$mRNAmol.cell <-
  vec$mRNA.norm*vec.added/vec$vector.norm/vec$n.cells # Transcripts per cell
```

```
## pdf
##   2
```

1.3 Calculate transcripts per cell (DEseq)

```
# Calculate vector input variable for DESeq2 normalization
vec$vec.DeSeq <- round(vec$Counts.vectorF/vec.ng * vec$n.cells/1e+06, 0)
```

Count data are multiplied with cell number and divided by the amount of added standard to create DeSeq2 vector input variable. Too large input data as well as non integer values produce an error if using the DESeq2 ControlGenes option. Therefor all values are divided by 1000000 and rounded to an integer

```
# Transform corrected counts to wide format
vector.DeSeq <- reshape(vec[, c(2, 4, 13)], idvar = "Strain.ID", timevar = "Salinity.level.replicates",
    direction = "wide")
colnames(vector.DeSeq) <- c("Strain.ID", "S1.1", "S1.2", "S2.1", "S2.2", "S3.1",
    "S3.2")
vector.DeSeq$vector <- c("vectorF")
as_tibble(vector.DeSeq)
```

```
## # A tibble: 11 x 8
##    Strain.ID     S1.1     S1.2     S2.1     S2.2     S3.1     S3.2 vector
##    <chr>        <dbl>    <dbl>    <dbl>    <dbl>    <dbl>    <dbl> <chr>
##  1 S337       3262532  3679389  1034770  1420780   777538   698844 vectorF
##  2 S432       1377344  2151450   329089   812866   213006   184770 vectorF
##  3 S599       7723623  7637819  6169766  5740193  1598089  1643929 vectorF
##  4 S366       2041598  1321208  1031747  3579108  4552154  1952012 vectorF
##  5 S490        408428   165038   405391   533743   611605   301917 vectorF
##  6 S618       1931828  6526905  1598807  2521218  5827300  5522755 vectorF
##  7 S331       1004965  1088984  1583752  1674223   665571   665164 vectorF
```

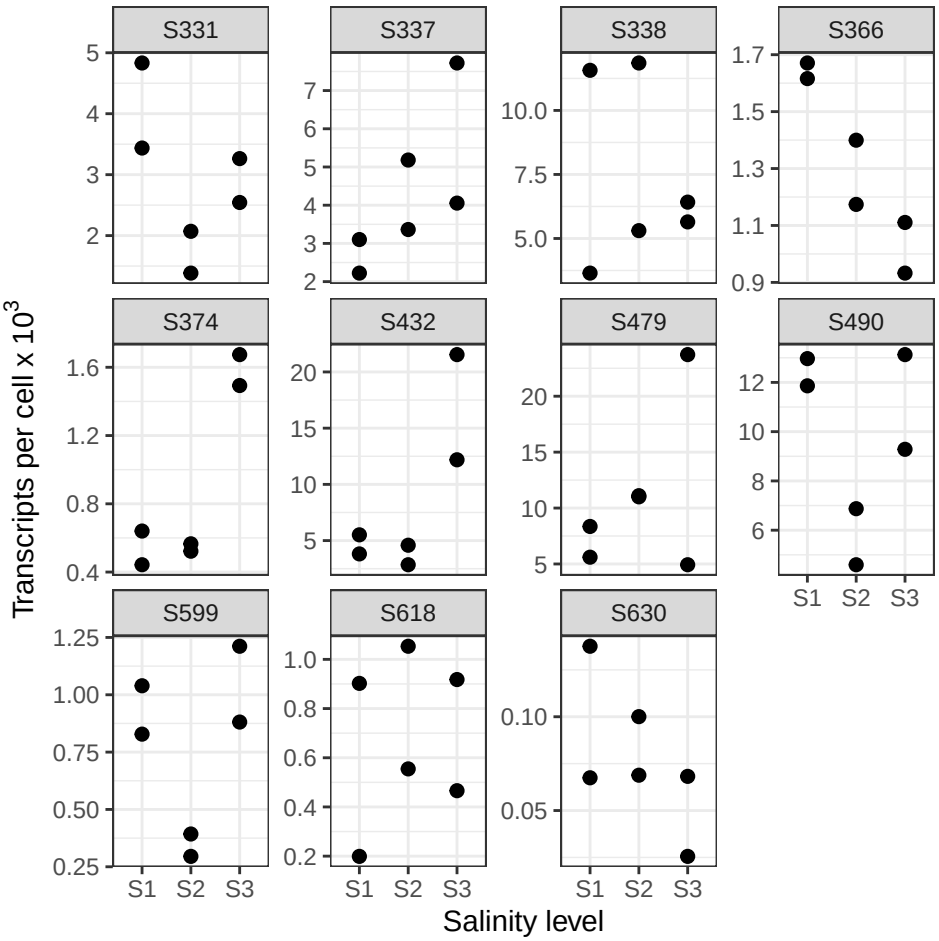

Figure 1: (Figure S3). Transcripts per cell in each strain and salinity level. Total transcript per cell by strain at the three sampling options from the transcriptional regulation experiment. Sampling points are indicated as S1,S2 and S3

```
##  8 S338        805316   238643   715762   227039   458983   408034 vectorF
##  9 S374       5061890  2987626  3503879  3412819  1229403   870391 vectorF
## 10 S479        195403   127434   145157   135395   136963   360126 vectorF
## 11 S630      15922628 10578708 10508689 19145422 39726656 46862468 vectorF
```

1.4 Estimate transcriptional regulation patterns using DESeq2

```
# Load input dunctional annotation data (BLAST output against the KEGG
# database 2016)
diamond_KEGG1 = read.table("../data_transc/crit.set1.KEGGko.tab", header = F)
diamond_KEGG2 = read.table("../data_transc/crit.set2.KEGGko.tab", header = F)
diamond_KEGG3 = read.table("../data_transc/crit.set3.KEGGko.tab", header = F)
diamond_KEGG = rbind(diamond_KEGG1, diamond_KEGG2, diamond_KEGG3)
rm(diamond_KEGG1, diamond_KEGG2, diamond_KEGG3)
as_tibble(diamond_KEGG)
```

```
## # A tibble: 33,060 x 13
##    V1     V2      V3       V4    V5    V6    V7    V8    V9   V10   V11      V12
##    <fct>  <fct>   <fct> <dbl> <int> <int> <int> <int> <int> <int> <int>    <dbl>
##  1 K01257 DNA432~ hsa:~ 100      33     0     1    33   385   417 9.80e-11
##  2 K09228 DNA432~ ptr:~  84.4    32     5     0     2    33  1151  1182 7.30e- 7
##  3 K13239 DNA337~ ggo:~  52.4    63    30     0    21    83    29    91 3.80e-13
##  4 K02870 DNA599~ cjc:~ 100      71     0     0     1    71   158   228 1.60e-33
##  5 K03349 DNA599~ mmu:~ 100      84     0     0     1    84   493   576 1.60e-40
##  6 K12319 DNA599~ mmu:~  30.3   155   107     1     1   155    82   235 1.70e-13
##  7 K13646 DNA432~ mmu:~  26.3   205   119     7    75   266   550   735 4.40e- 8
##  8 K00902 DNA432~ mmu:~  97.7    87     2     0     1    87   116   202 2.60e-38
##  9 K15414 DNA432~ mmu:~ 100      87     0     0     1    87   127   213 1.60e-43
## 10 K10385 DNA337~ rno:~  50.5   194    56     9    29   185    55   245 1.70e-39
## # ... with 33,050 more rows, and 1 more variable: V13 <dbl>
```

```
# Lad count data (output from mapping of reads against assembled genomes)
M1 = read.table("../data_transc/counts.M1.length.tab", header = T)
colnames(M1) = c("Chr_Geneid", "Length", "S3.1", "S3.2", "S1.1", "S1.2", "S2.1",
    "S2.2")
M2 = read.table("../data_transc/counts.M2.length.tab", header = T)
colnames(M2) = c("Chr_Geneid", "Length", "S3.1", "S3.2", "S1.1", "S1.2", "S2.1",
    "S2.2")
M3 = read.table("../data_transc/counts.M3.length.tab", header = T)
colnames(M3) = c("Chr_Geneid", "Length", "S3.1", "S3.2", "S1.1", "S1.2", "S2.1",
    "S2.2")
M.all = rbind(M1, M2, M3)
as_tibble(M.all)
```

```
## # A tibble: 46,916 x 8
##    Chr_Geneid                          Length  S3.1  S3.2  S1.1  S1.2  S2.1  S2.2
##    <fct>                                <int> <int> <int> <int> <int> <int> <int>
##  1 DNA599.NODE_1_length_601777_cov_1~    123     0     1     4     5     1     1
##  2 DNA599.NODE_1_length_601777_cov_1~    339    32    69   258   191    54    69
##  3 DNA599.NODE_1_length_601777_cov_1~    330    19    41   155    84    33    34
##  4 DNA599.NODE_1_length_601777_cov_1~   1662   166   257   891   685   231   226
```

```
##  5 DNA599.NODE_1_length_601777_cov_1~    480    51    60   194   109    48    54
##  6 DNA599.NODE_1_length_601777_cov_1~   1218   119   143   380   276   193   174
##  7 DNA599.NODE_1_length_601777_cov_1~    495    38    53   100    68    32    45
##  8 DNA599.NODE_1_length_601777_cov_1~   1299   250   415   947   755   249   353
##  9 DNA599.NODE_1_length_601777_cov_1~    762   246   365   776   654   256   306
## 10 DNA599.NODE_1_length_601777_cov_1~    600   120   157   729   451    86   118
## # ... with 46,906 more rows
```

```
# Data formating
li = strsplit(as.character(M.all[, 1]), "\\.")
li = do.call(rbind, li)
M.all$Strain.ID = gsub("DNA", "S", as.factor(li[, 1]))  #create column for Strain ID
M.all$Gene.ID = as.factor(paste0(li[, 2], ".", li[, 3]))  #create column for Gene ID
M.all = M.all[, c("Chr_Geneid", "Length", "S1.1", "S1.2", "S2.1", "S2.2", "S3.1",
    "S3.2", "Strain.ID", "Gene.ID")]
as_tibble(M.all)
```

```
## # A tibble: 46,916 x 10
##    Chr_Geneid     Length  S1.1  S1.2  S2.1  S2.2  S3.1  S3.2 Strain.ID Gene.ID
##    <fct>           <int> <int> <int> <int> <int> <int> <int> <chr>     <fct>
##  1 DNA599.NODE_1~    123     4     5     1     1     0     1 S599      NODE_1_1~
##  2 DNA599.NODE_1~    339   258   191    54    69    32    69 S599      NODE_1_1~
##  3 DNA599.NODE_1~    330   155    84    33    34    19    41 S599      NODE_1_1~
##  4 DNA599.NODE_1~   1662   891   685   231   226   166   257 S599      NODE_1_1~
##  5 DNA599.NODE_1~    480   194   109    48    54    51    60 S599      NODE_1_1~
##  6 DNA599.NODE_1~   1218   380   276   193   174   119   143 S599      NODE_1_1~
##  7 DNA599.NODE_1~    495   100    68    32    45    38    53 S599      NODE_1_1~
##  8 DNA599.NODE_1~   1299   947   755   249   353   250   415 S599      NODE_1_1~
##  9 DNA599.NODE_1~    762   776   654   256   306   246   365 S599      NODE_1_1~
## 10 DNA599.NODE_1~    600   729   451    86   118   120   157 S599      NODE_1_1~
## # ... with 46,906 more rows
```

```
rm(li, M1, M2, M3)

# Reformat count data to wide format
M.all_wide <- gather(M.all[, -c(2, 10)], key, value, -Chr_Geneid, -Strain.ID) %>%
    unite(new.col, c(Strain.ID, key)) %>% spread(new.col, value)
M.all_wide[is.na(M.all_wide)] <- 0
rownames(M.all_wide) <- M.all_wide[, 1]
M.all_wide <- M.all_wide[, -1]
as_tibble(M.all_wide)
```

```
## # A tibble: 46,916 x 66
##    S331_S1.1 S331_S1.2 S331_S2.1 S331_S2.2 S331_S3.1 S331_S3.2 S337_S1.1
##        <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
##  1         0         0         0         0         0         0       363
##  2         0         0         0         0         0         0       433
##  3         0         0         0         0         0         0       566
##  4         0         0         0         0         0         0       438
##  5         0         0         0         0         0         0       149
##  6         0         0         0         0         0         0       606
##  7         0         0         0         0         0         0       184
##  8         0         0         0         0         0         0        90
```

```
##  9         0         0         0         0         0         0       215
## 10         0         0         0         0         0         0       187
## # ... with 46,906 more rows, and 59 more variables: S337_S1.2 <dbl>,
## #   S337_S2.1 <dbl>, S337_S2.2 <dbl>, S337_S3.1 <dbl>, S337_S3.2 <dbl>,
## #   S338_S1.1 <dbl>, S338_S1.2 <dbl>, S338_S2.1 <dbl>, S338_S2.2 <dbl>,
## #   S338_S3.1 <dbl>, S338_S3.2 <dbl>, S366_S1.1 <dbl>, S366_S1.2 <dbl>,
## #   S366_S2.1 <dbl>, S366_S2.2 <dbl>, S366_S3.1 <dbl>, S366_S3.2 <dbl>,
## #   S374_S1.1 <dbl>, S374_S1.2 <dbl>, S374_S2.1 <dbl>, S374_S2.2 <dbl>,
## #   S374_S3.1 <dbl>, S374_S3.2 <dbl>, S432_S1.1 <dbl>, S432_S1.2 <dbl>,
## #   S432_S2.1 <dbl>, S432_S2.2 <dbl>, S432_S3.1 <dbl>, S432_S3.2 <dbl>,
## #   S479_S1.1 <dbl>, S479_S1.2 <dbl>, S479_S2.1 <dbl>, S479_S2.2 <dbl>,
## #   S479_S3.1 <dbl>, S479_S3.2 <dbl>, S490_S1.1 <dbl>, S490_S1.2 <dbl>,
## #   S490_S2.1 <dbl>, S490_S2.2 <dbl>, S490_S3.1 <dbl>, S490_S3.2 <dbl>,
## #   S599_S1.1 <dbl>, S599_S1.2 <dbl>, S599_S2.1 <dbl>, S599_S2.2 <dbl>,
## #   S599_S3.1 <dbl>, S599_S3.2 <dbl>, S618_S1.1 <dbl>, S618_S1.2 <dbl>,
## #   S618_S2.1 <dbl>, S618_S2.2 <dbl>, S618_S3.1 <dbl>, S618_S3.2 <dbl>,
## #   S630_S1.1 <dbl>, S630_S1.2 <dbl>, S630_S2.1 <dbl>, S630_S2.2 <dbl>,
## #   S630_S3.1 <dbl>, S630_S3.2 <dbl>
```

```
# Reformat vector data to wide format
vectors_wide <- gather(vector.DeSeq, key, value, -vector, -Strain.ID) %>% unite(new.col,
    c(Strain.ID, key)) %>% spread(new.col, value)
rownames(vectors_wide) <- vectors_wide[, 1]
vectors_wide <- vectors_wide[, -1]
as_tibble(vectors_wide)
```

```
## # A tibble: 1 x 66
##   S331_S1.1 S331_S1.2 S331_S2.1 S331_S2.2 S331_S3.1 S331_S3.2 S337_S1.1
##       <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
## 1   1004965   1088984   1583752   1674223    665571    665164   3262532
## # ... with 59 more variables: S337_S1.2 <dbl>, S337_S2.1 <dbl>,
## #   S337_S2.2 <dbl>, S337_S3.1 <dbl>, S337_S3.2 <dbl>, S338_S1.1 <dbl>,
## #   S338_S1.2 <dbl>, S338_S2.1 <dbl>, S338_S2.2 <dbl>, S338_S3.1 <dbl>,
## #   S338_S3.2 <dbl>, S366_S1.1 <dbl>, S366_S1.2 <dbl>, S366_S2.1 <dbl>,
## #   S366_S2.2 <dbl>, S366_S3.1 <dbl>, S366_S3.2 <dbl>, S374_S1.1 <dbl>,
## #   S374_S1.2 <dbl>, S374_S2.1 <dbl>, S374_S2.2 <dbl>, S374_S3.1 <dbl>,
## #   S374_S3.2 <dbl>, S432_S1.1 <dbl>, S432_S1.2 <dbl>, S432_S2.1 <dbl>,
## #   S432_S2.2 <dbl>, S432_S3.1 <dbl>, S432_S3.2 <dbl>, S479_S1.1 <dbl>,
## #   S479_S1.2 <dbl>, S479_S2.1 <dbl>, S479_S2.2 <dbl>, S479_S3.1 <dbl>,
## #   S479_S3.2 <dbl>, S490_S1.1 <dbl>, S490_S1.2 <dbl>, S490_S2.1 <dbl>,
## #   S490_S2.2 <dbl>, S490_S3.1 <dbl>, S490_S3.2 <dbl>, S599_S1.1 <dbl>,
## #   S599_S1.2 <dbl>, S599_S2.1 <dbl>, S599_S2.2 <dbl>, S599_S3.1 <dbl>,
## #   S599_S3.2 <dbl>, S618_S1.1 <dbl>, S618_S1.2 <dbl>, S618_S2.1 <dbl>,
## #   S618_S2.2 <dbl>, S618_S3.1 <dbl>, S618_S3.2 <dbl>, S630_S1.1 <dbl>,
## #   S630_S1.2 <dbl>, S630_S2.1 <dbl>, S630_S2.2 <dbl>, S630_S3.1 <dbl>,
## #   S630_S3.2 <dbl>
```

```
# Join vector DESeq2 variable with count data
M.all.vector <- rbind(vectors_wide, M.all_wide)
as_tibble(M.all.vector)
```

```
## # A tibble: 46,917 x 66
##    S331_S1.1 S331_S1.2 S331_S2.1 S331_S2.2 S331_S3.1 S331_S3.2 S337_S1.1
```

```
##        <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
##  1   1004965   1088984   1583752   1674223    665571    665164   3262532
##  2         0         0         0         0         0         0       363
##  3         0         0         0         0         0         0       433
##  4         0         0         0         0         0         0       566
##  5         0         0         0         0         0         0       438
##  6         0         0         0         0         0         0       149
##  7         0         0         0         0         0         0       606
##  8         0         0         0         0         0         0       184
##  9         0         0         0         0         0         0        90
## 10         0         0         0         0         0         0       215
## # ... with 46,907 more rows, and 59 more variables: S337_S1.2 <dbl>,
## #   S337_S2.1 <dbl>, S337_S2.2 <dbl>, S337_S3.1 <dbl>, S337_S3.2 <dbl>,
## #   S338_S1.1 <dbl>, S338_S1.2 <dbl>, S338_S2.1 <dbl>, S338_S2.2 <dbl>,
## #   S338_S3.1 <dbl>, S338_S3.2 <dbl>, S366_S1.1 <dbl>, S366_S1.2 <dbl>,
## #   S366_S2.1 <dbl>, S366_S2.2 <dbl>, S366_S3.1 <dbl>, S366_S3.2 <dbl>,
## #   S374_S1.1 <dbl>, S374_S1.2 <dbl>, S374_S2.1 <dbl>, S374_S2.2 <dbl>,
## #   S374_S3.1 <dbl>, S374_S3.2 <dbl>, S432_S1.1 <dbl>, S432_S1.2 <dbl>,
## #   S432_S2.1 <dbl>, S432_S2.2 <dbl>, S432_S3.1 <dbl>, S432_S3.2 <dbl>,
## #   S479_S1.1 <dbl>, S479_S1.2 <dbl>, S479_S2.1 <dbl>, S479_S2.2 <dbl>,
## #   S479_S3.1 <dbl>, S479_S3.2 <dbl>, S490_S1.1 <dbl>, S490_S1.2 <dbl>,
## #   S490_S2.1 <dbl>, S490_S2.2 <dbl>, S490_S3.1 <dbl>, S490_S3.2 <dbl>,
## #   S599_S1.1 <dbl>, S599_S1.2 <dbl>, S599_S2.1 <dbl>, S599_S2.2 <dbl>,
## #   S599_S3.1 <dbl>, S599_S3.2 <dbl>, S618_S1.1 <dbl>, S618_S1.2 <dbl>,
## #   S618_S2.1 <dbl>, S618_S2.2 <dbl>, S618_S3.1 <dbl>, S618_S3.2 <dbl>,
## #   S630_S1.1 <dbl>, S630_S1.2 <dbl>, S630_S2.1 <dbl>, S630_S2.2 <dbl>,
## #   S630_S3.1 <dbl>, S630_S3.2 <dbl>
```

```
# Prepare metadata and the test design for DESeq2 analyses
coldata = data.frame(level = factor(substring(colnames(M.all.vector), 1, 7)),
    type = factor(rep("paired-end", 66)), incubation = factor(colnames(M.all.vector)))

dds <- DESeqDataSetFromMatrix(countData = M.all.vector, colData = coldata, design = ~level)
```

```
## converting counts to integer mode
```

1.5 Normalization of count data based on vector counts via the controlGenes option

```
# Absolute quantification
dds <- estimateSizeFactors(dds, controlGenes = c(1))

# Raw count data
as_tibble(counts(dds))
```

```
## # A tibble: 46,917 x 66
##    S331_S1.1 S331_S1.2 S331_S2.1 S331_S2.2 S331_S3.1 S331_S3.2 S337_S1.1
##        <int>     <int>     <int>     <int>     <int>     <int>     <int>
##  1   1004965   1088984   1583752   1674223    665571    665164   3262532
##  2         0         0         0         0         0         0       363
##  3         0         0         0         0         0         0       433
```

```
##  4         0         0         0         0         0         0       566
##  5         0         0         0         0         0         0       438
##  6         0         0         0         0         0         0       149
##  7         0         0         0         0         0         0       606
##  8         0         0         0         0         0         0       184
##  9         0         0         0         0         0         0        90
## 10         0         0         0         0         0         0       215
## # ... with 46,907 more rows, and 59 more variables: S337_S1.2 <int>,
## #   S337_S2.1 <int>, S337_S2.2 <int>, S337_S3.1 <int>, S337_S3.2 <int>,
## #   S338_S1.1 <int>, S338_S1.2 <int>, S338_S2.1 <int>, S338_S2.2 <int>,
## #   S338_S3.1 <int>, S338_S3.2 <int>, S366_S1.1 <int>, S366_S1.2 <int>,
## #   S366_S2.1 <int>, S366_S2.2 <int>, S366_S3.1 <int>, S366_S3.2 <int>,
## #   S374_S1.1 <int>, S374_S1.2 <int>, S374_S2.1 <int>, S374_S2.2 <int>,
## #   S374_S3.1 <int>, S374_S3.2 <int>, S432_S1.1 <int>, S432_S1.2 <int>,
## #   S432_S2.1 <int>, S432_S2.2 <int>, S432_S3.1 <int>, S432_S3.2 <int>,
## #   S479_S1.1 <int>, S479_S1.2 <int>, S479_S2.1 <int>, S479_S2.2 <int>,
## #   S479_S3.1 <int>, S479_S3.2 <int>, S490_S1.1 <int>, S490_S1.2 <int>,
## #   S490_S2.1 <int>, S490_S2.2 <int>, S490_S3.1 <int>, S490_S3.2 <int>,
## #   S599_S1.1 <int>, S599_S1.2 <int>, S599_S2.1 <int>, S599_S2.2 <int>,
## #   S599_S3.1 <int>, S599_S3.2 <int>, S618_S1.1 <int>, S618_S1.2 <int>,
## #   S618_S2.1 <int>, S618_S2.2 <int>, S618_S3.1 <int>, S618_S3.2 <int>,
## #   S630_S1.1 <int>, S630_S1.2 <int>, S630_S2.1 <int>, S630_S2.2 <int>,
## #   S630_S3.1 <int>, S630_S3.2 <int>
```

```
# Normalized count data
as_tibble(counts(dds, normalized = TRUE))
```

```
## # A tibble: 46,917 x 66
##    S331_S1.1 S331_S1.2 S331_S2.1 S331_S2.2 S331_S3.1 S331_S3.2 S337_S1.1
##        <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>     <dbl>
##  1 1376135.  1376135.  1376135.  1376135.  1376135.  1376135. 1376135.
##  2        0         0         0         0         0         0      153.
##  3        0         0         0         0         0         0      183.
##  4        0         0         0         0         0         0      239.
##  5        0         0         0         0         0         0      185.
##  6        0         0         0         0         0         0       62.8
##  7        0         0         0         0         0         0      256.
##  8        0         0         0         0         0         0       77.6
##  9        0         0         0         0         0         0       38.0
## 10        0         0         0         0         0         0       90.7
## # ... with 46,907 more rows, and 59 more variables: S337_S1.2 <dbl>,
## #   S337_S2.1 <dbl>, S337_S2.2 <dbl>, S337_S3.1 <dbl>, S337_S3.2 <dbl>,
## #   S338_S1.1 <dbl>, S338_S1.2 <dbl>, S338_S2.1 <dbl>, S338_S2.2 <dbl>,
## #   S338_S3.1 <dbl>, S338_S3.2 <dbl>, S366_S1.1 <dbl>, S366_S1.2 <dbl>,
## #   S366_S2.1 <dbl>, S366_S2.2 <dbl>, S366_S3.1 <dbl>, S366_S3.2 <dbl>,
## #   S374_S1.1 <dbl>, S374_S1.2 <dbl>, S374_S2.1 <dbl>, S374_S2.2 <dbl>,
## #   S374_S3.1 <dbl>, S374_S3.2 <dbl>, S432_S1.1 <dbl>, S432_S1.2 <dbl>,
## #   S432_S2.1 <dbl>, S432_S2.2 <dbl>, S432_S3.1 <dbl>, S432_S3.2 <dbl>,
## #   S479_S1.1 <dbl>, S479_S1.2 <dbl>, S479_S2.1 <dbl>, S479_S2.2 <dbl>,
## #   S479_S3.1 <dbl>, S479_S3.2 <dbl>, S490_S1.1 <dbl>, S490_S1.2 <dbl>,
## #   S490_S2.1 <dbl>, S490_S2.2 <dbl>, S490_S3.1 <dbl>, S490_S3.2 <dbl>,
## #   S599_S1.1 <dbl>, S599_S1.2 <dbl>, S599_S2.1 <dbl>, S599_S2.2 <dbl>,
## #   S599_S3.1 <dbl>, S599_S3.2 <dbl>, S618_S1.1 <dbl>, S618_S1.2 <dbl>,
## #   S618_S2.1 <dbl>, S618_S2.2 <dbl>, S618_S3.1 <dbl>, S618_S3.2 <dbl>,
```

```
## #   S630_S1.1 <dbl>, S630_S1.2 <dbl>, S630_S2.1 <dbl>, S630_S2.2 <dbl>,
## #   S630_S3.1 <dbl>, S630_S3.2 <dbl>
```

```
# Get DESeq variable for per cell transcription levels by summing up
# normalized count data
DESeq.cell <- data.frame(DESeq.cell = colSums(counts(dds, normalized = TRUE)[2:dim(counts(dds,
    normalized = TRUE))[1], ]))

# merge DESeq variable for per cell transcription levels into vec dataframe
vec <- merge(vec, DESeq.cell, by.x = 0, by.y = 0)
# write.table(file='../DESeq.cell.txt',vec)
```

1.6 Regression between transcription per cell following Satinsky et al 2013 and DESeq2

```
## pdf
##   2
```

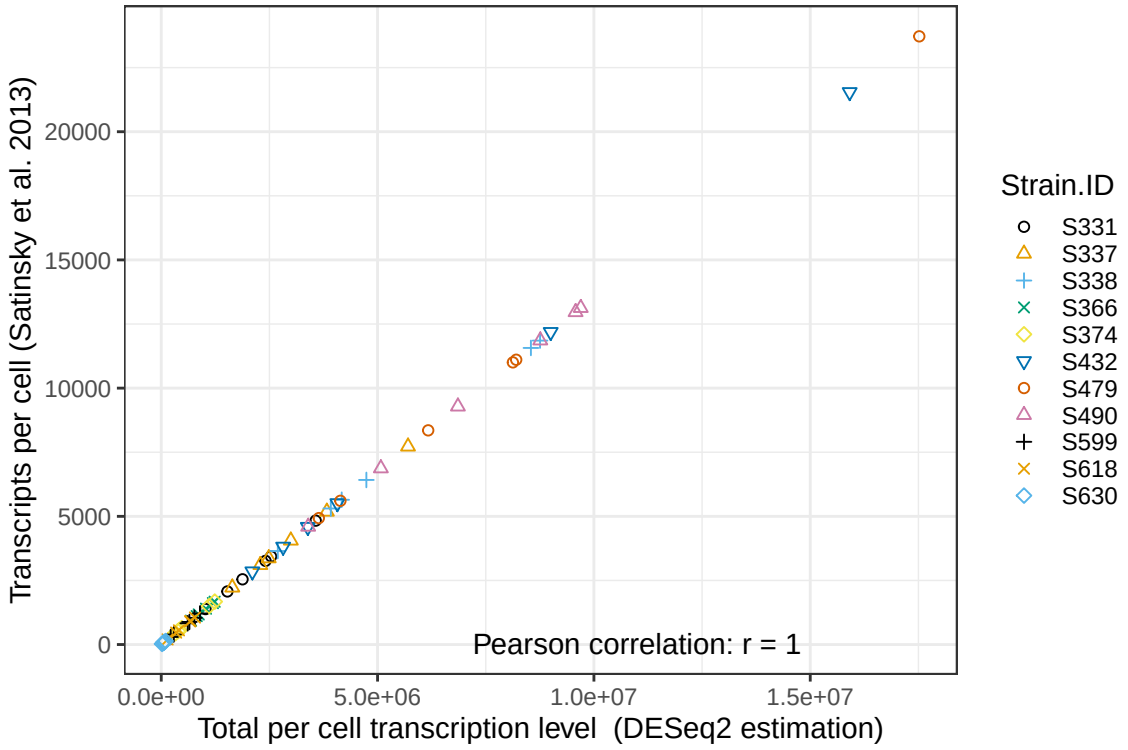

Figure 2: (Figure S2). Regression of the number transcripts per cell estimated as published elsewhere (Satinsky et al., 2013) against a variable for the total per cell transcription level estimated using the DESeq2 package (R2=1.00, P<0.001, Pearson correlation)

```
summary(lm(vec$DESeq.cell ~ vec$mRNAmol.cell))
```

```
##
## Call:
```

```
## lm(formula = vec$DESeq.cell ~ vec$mRNAmol.cell)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -17.2427  -0.5245   0.1086   0.3844  14.9064
##
## Coefficients:
##                    Estimate Std. Error    t value Pr(>|t|)
## (Intercept)      -4.579e-01  7.454e-01 -6.140e-01    0.541
## vec$mRNAmol.cell  7.386e+02  1.133e-04  6.518e+06   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.58 on 64 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 4.249e+13 on 1 and 64 DF,  p-value: < 2.2e-16
```

1.7 Run DESeq2

Log2FoldChanges for absolute per cell transcription levels are calculated for each strain applying a loop

```
dds <- DESeq(dds)
head(dds)
```

```
## class: DESeqDataSet
## dim: 6 66
## metadata(1): version
## assays(4): counts mu H cooks
## rownames(6): vectorF DNA337.NODE_1_length_671332_cov_119.298333_1 ...
##   DNA337.NODE_1_length_671332_cov_119.298333_101
##   DNA337.NODE_1_length_671332_cov_119.298333_102
## rowData names(146): baseMean baseVar ... deviance maxCooks
## colnames(66): S331_S1.1 S331_S1.2 ... S630_S3.1 S630_S3.2
## colData names(4): level type incubation sizeFactor
```

```
# Pairwise comparisons of fold change transcription between salinity levels
res.all = list()
for (i in 1:11) {
    strain = levels(factor(M.all$Strain.ID))[i]
    # Pairwise comparison S2 vs S1
    res.S2S1 <- as.data.frame(results(dds, contrast = c("level", paste0(strain,
        "_S2"), paste0(strain, "_S1"))))
    # Remove lines from comparisons with zero-counts (other strains)
    res.S2S1 <- res.S2S1[grepl(gsub("S", "DNA", strain), rownames(res.S2S1)),
        ]
    res.S2S1$strain.ID <- strain
    res.S2S1$direction <- c("S2:S1")
    res.S2S1$gene.ID <- rownames(res.S2S1)
    ## Pairwise comparison S2 vs S3
    res.S2S3 <- as.data.frame(results(dds, contrast = c("level", paste0(strain,
        "_S2"), paste0(strain, "_S3"))))
    # Remove lines from comparisons with zero-counts (other strains)
```

```
    res.S2S3 <- res.S2S3[grepl(gsub("S", "DNA", strain), rownames(res.S2S3)),
        ]
    res.S2S3$strain.ID <- strain
    res.S2S3$direction <- c("S2:S3")
    res.S2S3$gene.ID <- rownames(res.S2S3)
    # Pairwise comparison S1 vs S3
    res.S1S3 <- as.data.frame(results(dds, contrast = c("level", paste0(strain,
        "_S1"), paste0(strain, "_S3"))))
    # Remove lines from comparisons with zero-counts (other strains)
    res.S1S3 <- res.S1S3[grepl(gsub("S", "DNA", strain), rownames(res.S1S3)),
        ]
    res.S1S3$strain.ID <- strain
    res.S1S3$direction <- c("S1:S3")
    res.S1S3$gene.ID <- rownames(res.S1S3)

    res <- rbind(res.S2S1, res.S2S3, res.S1S3)
    res.all[[strain]] <- res
}

rm(list = c("res.S1S3", "res.S2S1", "res.S2S3", "coldata", "dds", "M.all_wide",
    "vectors_wide"))
res.DeSeq <- as.data.frame.matrix(do.call(rbind, res.all))
rownames(res.DeSeq) <- substring(rownames(res.DeSeq), 6, 200)
```

1.8 merge DESeq2 output with gene annotation data

```
res.DeSeq.K <- merge(res.DeSeq, diamond_KEGG[, 1:2], by.x = "gene.ID", by.y = "V2")
colnames(res.DeSeq.K)[10] <- "ko"
# Summary of the dataset including the gene name
res.DeSeq.Kegg <- aggregate(. ~ strain.ID + direction + ko + gene.ID, data = res.DeSeq.K[,
    c(1:3, 8:10)], FUN = mean)
as_tibble(res.DeSeq.Kegg)
```

```
## # A tibble: 95,655 x 6
##    strain.ID direction ko     gene.ID                    baseMean log2FoldChange
##    <chr>     <chr>     <fct>  <chr>                         <dbl>          <dbl>
##  1 S331      S1:S3     K00523 DNA331.NODE_1_length_1298~     4.73          2.24
##  2 S331      S2:S1     K00523 DNA331.NODE_1_length_1298~     4.73         -1.75
##  3 S331      S2:S3     K00523 DNA331.NODE_1_length_1298~     4.73          0.490
##  4 S331      S1:S3     K05560 DNA331.NODE_1_length_1298~     2.14          1.09
##  5 S331      S2:S1     K05560 DNA331.NODE_1_length_1298~     2.14         -2.28
##  6 S331      S2:S3     K05560 DNA331.NODE_1_length_1298~     2.14         -1.19
##  7 S331      S1:S3     K05559 DNA331.NODE_1_length_1298~    50.9           0.731
##  8 S331      S2:S1     K05559 DNA331.NODE_1_length_1298~    50.9          -1.32
##  9 S331      S2:S3     K05559 DNA331.NODE_1_length_1298~    50.9          -0.590
## 10 S331      S1:S3     K08714 DNA331.NODE_1_length_1298~   103.           0.470
## # ... with 95,645 more rows
```

```
write.table(res.DeSeq.Kegg, "../data_transc/res.DeSeq.K2016new.tab", sep = "\t",
    row.names = FALSE)
```