Rscript3: Mixed linear models for transcriptional regulation patterns

Angel Rain & Sara Beier

11/02/2021

Contents

1 Packages

```
rm(list = ls())
# Mixed linear model
library(nlme)
library(lmerTest)
library(olsrr)
library(MuMIn)
library(stringr)
library(ggplot2)
library(kableExtra)
library(readxl)
library(dplyr)
library(ggstatsplot)
# Colorblind pallete
cbbPalette <- c("#000000", "#E69F00", "#56B4E9", "#009E73", "#F0E442",
    "#0072B2", "#D55E00", "#CC79A7")
```

2 Load and format input data

```
# Load KEGG onthology and transporter data
descriptionK <- read.table("../ko_description_May16.tab", sep = "\t", quote = "",
    stringsAsFactors = F, row.names = 1, header = T)
descriptionK$ko <- rownames(descriptionK)
as_tibble(descriptionK)
```

```
## # A tibble: 19,002 x 2
##    description                                                          ko
##    <chr>                                                                <chr>
##  1 E1.1.1.1, adh; alcohol dehydrogenase [EC:1.1.1.1]                    K000~
##  2 AKR1A1, adh; alcohol dehydrogenase (NADP+) [EC:1.1.1.2]              K000~
##  3 E1.1.1.3; homoserine dehydrogenase [EC:1.1.1.3]                      K000~
##  4 BDH, butB; (R,R)-butanediol dehydrogenase / meso-butanediol dehydrogen~ K000~
##  5 gldA; glycerol dehydrogenase [EC:1.1.1.6]                            K000~
##  6 GPD1; glycerol-3-phosphate dehydrogenase (NAD+) [EC:1.1.1.8]         K000~
##  7 dalD; D-arabinitol 4-dehydrogenase [EC:1.1.1.11]                     K000~
##  8 SORD, gutB; L-iditol 2-dehydrogenase [EC:1.1.1.14]                   K000~
##  9 mtlD; mannitol-1-phosphate 5-dehydrogenase [EC:1.1.1.17]             K000~
## 10 iolG; myo-inositol 2-dehydrogenase / D-chiro-inositol 1-dehydrogenase ~ K000~
## # ... with 18,992 more rows
```

```
# Gene description for each KEGG entry
ok_path = read.table("../ko_pathway.mod.list", header = F)
names(ok_path) = c("ko", "path")
as_tibble(ok_path)  # Path information for each KEGG entry
```

```
## # A tibble: 31,051 x 2
##    ko     path
##    <fct>  <fct>
##  1 K00001 ko00010
##  2 K00002 ko00010
```

```
##  3 K00016 ko00010
##  4 K00114 ko00010
##  5 K00121 ko00010
##  6 K00128 ko00010
##  7 K00129 ko00010
##  8 K00131 ko00010
##  9 K00134 ko00010
## 10 K00138 ko00010
## # ... with 31,041 more rows
```

```
# Load Table S4
trans_op <- read_xlsx("../Table S4.xlsx", sheet = "Table S4")
as_tibble(trans_op)  # List for genes encoding the transport of osmoprotectants
```

```
## # A tibble: 402 x 4
##    KEGG.ID Gene.description                     KEGG.path.ID Potential.osmolyte~
##    <chr>   <chr>                                <chr>        <lgl>
##  1 K09969  aapJ, bztA; general L-amino acid tr~ ko02010      TRUE
##  2 K09970  aapQ, bztB; general L-amino acid tr~ ko02010      TRUE
##  3 K09971  aapM, bztC; general L-amino acid tr~ ko02010      TRUE
##  4 K09972  aapP, bztD; general L-amino acid tr~ ko02010      TRUE
##  5 K02025  ABC.MS.P; multiple sugar transport ~ NA           TRUE
##  6 K02026  ABC.MS.P1; multiple sugar transport~ NA           TRUE
##  7 K02027  ABC.MS.S; multiple sugar transport ~ NA           TRUE
##  8 K02028  ABC.PA.A; polar amino acid transpor~ NA           TRUE
##  9 K02029  ABC.PA.P; polar amino acid transpor~ NA           TRUE
## 10 K02030  ABC.PA.S; polar amino acid transpor~ NA           TRUE
## # ... with 392 more rows
```

```
# Load and format gene regulation data obtained from DESeq2
# (Rscript1_Gene_regulation)
res.DeSeq.K <- read.table("../data_transc/res.DeSeq.K2016new.tab", sep = "\t",
    header = TRUE)
# aggregate DESeq output for gene variants annotated to the same KEGG
# entry (mean)
res.DeSeq.K <- aggregate(. ~ strain.ID + direction + ko, data = res.DeSeq.K[,
    c(1:3, 6)], FUN = mean)
res.DeSeq.K[, 1:3] <- lapply(res.DeSeq.K[, 1:3], factor)  # Assign factor to column 1 to 3

# Load data for total transcripts per cell estimations
# (Rscript1_Gene_regulation)
DESeq.cell <- read.table("../DESeq.cell.txt", header = T)
as_tibble(DESeq.cell)
```

```
## # A tibble: 66 x 15
##    Row.names Library.name Strain.ID Salinity.level Salinity.level.replicates
##    <fct>     <fct>        <fct>     <fct>          <fct>
##  1 S331_S1.1 M3L1         S331      S1             S1.1
##  2 S331_S1.2 M3L2         S331      S1             S1.2
##  3 S331_S2.1 M3M1         S331      S2             S2.1
##  4 S331_S2.2 M3M2         S331      S2             S2.2
##  5 S331_S3.1 M3H1         S331      S3             S3.1
##  6 S331_S3.2 M3H2         S331      S3             S3.2
```

```
##  7 S337_S1.1 M1L1         S337      S1             S1.1
##  8 S337_S1.2 M1L2         S337      S1             S1.2
##  9 S337_S2.1 M1M1         S337      S2             S2.1
## 10 S337_S2.2 M1M2         S337      S2             S2.2
## # ... with 56 more rows, and 10 more variables: Counts.vectorF <dbl>,
## #   Cell.per.mL <dbl>, Volume.medium..ml. <int>, Mapped.reads <int>,
## #   n.cells <dbl>, mRNA.norm <dbl>, vector.normvector.norm <dbl>,
## #   mRNAmol.cell <dbl>, vec.DeSeq <int>, DESeq.cell <dbl>
```

```
# Mean value from replicates
D = aggregate(DESeq.cell ~ Strain.ID + Salinity.level, data = DESeq.cell,
    FUN = mean)

# Log2fold changes for total transcripts
dT1 = log2(D$DESeq.cell[D$Salinity.level == "S2"]/D$DESeq.cell[D$Salinity.level ==
    "S1"])
dT2 = log2(D$DESeq.cell[D$Salinity.level == "S2"]/D$DESeq.cell[D$Salinity.level ==
    "S3"])
dT3 = log2(D$DESeq.cell[D$Salinity.level == "S1"]/D$DESeq.cell[D$Salinity.level ==
    "S3"])

# Add Log2fold changes for total transcripts to DESeq output
tot1 <- data.frame(strain.ID = levels(res.DeSeq.K$strain.ID), direction = c("S2:S1"),
    ko = c("total"), log2FoldChange = dT1)
tot2 <- data.frame(strain.ID = levels(res.DeSeq.K$strain.ID), direction = c("S2:S3"),
    ko = c("total"), log2FoldChange = dT2)
tot3 <- data.frame(strain.ID = levels(res.DeSeq.K$strain.ID), direction = c("S1:S3"),
    ko = c("total"), log2FoldChange = dT3)
res.DeSeq.K <- rbind(res.DeSeq.K, tot1, tot2, tot3)

# Creating index column
res.DeSeq.K$strain.dir <- paste(res.DeSeq.K$strain, res.DeSeq.K$direction,
    sep = ".")
tail(res.DeSeq.K)
```

```
##       strain.ID direction    ko log2FoldChange strain.dir
## 66739      S432     S1:S3 total     -1.8546800 S432.S1:S3
## 66740      S479     S1:S3 total     -1.0373003 S479.S1:S3
## 66741      S490     S1:S3 total      0.1476862 S490.S1:S3
## 66742      S599     S1:S3 total     -0.1640665 S599.S1:S3
## 66743      S618     S1:S3 total     -0.3294566 S618.S1:S3
## 66744      S630     S1:S3 total      1.1287968 S630.S1:S3
```

```
# Load and format NB /fitness data (Rscript2_Niche_breadth)
niches <- read.table("../niche-performance.txt", sep = "\t", header = TRUE)

# Creating index column
niches$strain.dir <- paste(niches$strains.ID, niches$direction, sep = ".")
as_tibble(niches)
```

```
## # A tibble: 33 x 7
##    strains.ID direction n_breadth n_breadth_filtered n_breadth_weighted
##    <fct>      <fct>         <int>              <int>              <dbl>
```

```
##  1 S331       S2:S1            45                 98              98
##  2 S337       S2:S1            17                 17              17
##  3 S338       S2:S1            17                 NA              25.9
##  4 S366       S2:S1            18                 NA              31.3
##  5 S374       S2:S1            18                 18              18
##  6 S432       S2:S1            18                 18              18
##  7 S479       S2:S1            29                 29              29
##  8 S490       S2:S1            18                 18              21
##  9 S599       S2:S1            12                 12              12
## 10 S618       S2:S1            14                 53              53
## # ... with 23 more rows, and 2 more variables: delta_fitness <dbl>,
## #   strain.dir <chr>
```

```r
# Merge gene regulation data with fitness information
dat <- merge(res.DeSeq.K, niches[, -c(1, 2)], by.x = "strain.dir", by.y = "strain.dir")

# Log transform NB data
dat$n_breadth_filtered.log <- log(dat$n_breadth_filtered)  # log-transformation of NB

# Create correction factor negative value for comparisons with positive
# fitness
dat$corr.factor <- ifelse(dat$delta_fitness > 0, -1, 1)

# Turn all values for fitness in negative values to always have the
# perspective of fitness decrease (stress)
dat$delta_fitness.corr <- dat$delta_fitness * dat$corr.factor

# Turn logLFC if delta fitness is positive: now all values indicate
# direction of fold change regulation for fitness decrease (stress)
dat$log2FoldChange.corr <- dat$log2FoldChange * dat$corr.factor

## Assign stress variable (negative value of delta_fitness.corr)
dat$stress <- -dat$delta_fitness.corr

# Filter stress values for correlation with filtered NBs
dat$stress_filtered <- ifelse(dat$n_breadth_filtered == "NA", "NA", dat$stress)
as_tibble(dat)
```

```
## # A tibble: 66,744 x 15
##    strain.dir strain.ID direction ko     log2FoldChange n_breadth
##    <chr>      <fct>     <fct>     <fct>           <dbl>     <int>
##  1 S331.S1:S3 S331      S1:S3     K00001         0.695        143
##  2 S331.S1:S3 S331      S1:S3     K13288         0.470        143
##  3 S331.S1:S3 S331      S1:S3     K00108         0.0872       143
##  4 S331.S1:S3 S331      S1:S3     K01505         0.699        143
##  5 S331.S1:S3 S331      S1:S3     K02051         1.63         143
##  6 S331.S1:S3 S331      S1:S3     K00390         0.837        143
##  7 S331.S1:S3 S331      S1:S3     K15984         1.36         143
##  8 S331.S1:S3 S331      S1:S3     K11749         0.553        143
##  9 S331.S1:S3 S331      S1:S3     K02379         0.495        143
## 10 S331.S1:S3 S331      S1:S3     K00616         0.627        143
## # ... with 66,734 more rows, and 9 more variables: n_breadth_filtered <int>,
## #   n_breadth_weighted <dbl>, delta_fitness <dbl>,
## #   n_breadth_filtered.log <dbl>, corr.factor <dbl>, delta_fitness.corr <dbl>,
```

```
## #   log2FoldChange.corr <dbl>, stress <dbl>, stress_filtered <dbl>
```

3 Total transcripts regulation

We used the total transcript per cell (estimated from DESeq2) to test the overall correlations between total per cell transcription levels against NB and stress.

3.1 MLMs for total transcripts

```
# Subset dataframe for total transcripts per cell
dat.tot <- dat[dat$ko %in% c("total"), ]

# Model fitting
t.fit1 <- lme(abs(log2FoldChange.corr) ~ n_breadth_filtered.log, random = ~1 |
    direction, data = dat.tot, na.action = na.omit)
t.fit2 <- lme(abs(log2FoldChange.corr) ~ stress, random = ~1 | direction,
    data = dat.tot, na.action = na.omit)
t.fit3 <- lme((log2FoldChange.corr) ~ stress, random = ~1 | direction,
    data = dat.tot, na.action = na.omit)
t.fit4 <- lme(abs(log2FoldChange.corr) ~ stress_filtered, random = ~1 |
    direction, data = dat.tot, na.action = na.omit)
t.fit5 <- lme((log2FoldChange.corr) ~ stress_filtered, random = ~1 | direction,
    data = dat.tot, na.action = na.omit)

# Kolmogorv smirnov test
res.normality <- c(round(as.numeric(as.vector(ols_test_normality(resid(t.fit1)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(t.fit2)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(t.fit3)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(t.fit4)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(t.fit5)))[[1]][2]),
    2) >= 0.05)
names(res.normality) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")

kbl(res.normality, booktabs = TRUE)
```

	x
|L2FC|vsNB	TRUE
a(|L2FC|vsStress)	TRUE
a(L2FCvsStress)	TRUE
f(|L2FC|vsStress)	TRUE
f(L2FCvsStress)	TRUE

Inspection of res.normality verifies that all MLMs have normal distributed residuals

```
# Function to extract n, Slope, peusdo R2, and Pvalue from the fitted
# models
MLM.stats <- function(i) {
    c(n = as.numeric(dim(summary(get(fit[i]))$groups)[2]), slope = summary(get(fit[i]))$tTable[2,
```

```
        1], Peusdo.R2 = r.squaredGLMM(get(fit[i]))[2], Pvalue = summary(get(fit[i]))$tTable[2,
        5])
}

# Create vector with the names of the fitted models
fit <- paste0("t.fit", 1:5)
mlm.tot <- as.data.frame.matrix(sapply(1:5, MLM.stats))
```

```
## Warning: 'r.squaredGLMM' now calculates a revised statistic. See the help page.
```

```
colnames(mlm.tot) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")
mlm.tot$category = c("total transcripts")

# Summary MLM statistic for total transcripts
kbl(mlm.tot, booktabs = TRUE) %>% kable_styling(latex_options = "scale_down")
```

	|L2FC|vsNB	a(|L2FC|vsStress)	a(L2FCvsStress)	f(|L2FC|vsStress)	f(L2FCvsStress)	category
n	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	total transcripts
slope	-0.0629220	0.3779362	0.2825325	0.0403286	0.8329814	total transcripts
Peusdo.R2	0.0056232	0.0175577	0.0645325	0.0002387	0.1314727	total transcripts
Pvalue	0.7679877	0.4575296	0.7489949	0.9515911	0.4817701	total transcripts

4 Transcriptional regulation of fitness-related genes

To test if genes with a direct link to an organism's fitness are stronger regulated in strains with narrow NB, we created 3 categories of putative fitness-related encoding genes from genes shared across the 11 strains: 1) DNA polymerases enzymes are essential for DNA replication and cell proliferation. This category considered all genes containing the text string "DNA polymerase" in the gene description. 2) RNA polymerases transcribe genes into mRNA and are directly related to growth via protein synthesis (KEGG pathway ko03020). 3) Ribosomal proteins are ribosomes units that catalyze the translation of mRNA into proteins and play an essential role in cellular biomass production via protein synthesis (KEGG pathway ko03010). We run the MLMs for gene regulation against the NB, and the stress estimations for the 3 categories mentioned above, considering genes shared among all 11 strains.

4.1 Identification of genes shared among all strains

```
# Extract shared genes in all strains
ko_shared <- Reduce(intersect, list(dat$ko[dat$strain.ID == levels(dat$strain.ID)[1]],
    dat$ko[dat$strain.ID == levels(dat$strain.ID)[2]], dat$ko[dat$strain.ID ==
        levels(dat$strain.ID)[3]], dat$ko[dat$strain.ID == levels(dat$strain.ID)[4]],
    dat$ko[dat$strain.ID == levels(dat$strain.ID)[5]], dat$ko[dat$strain.ID ==
        levels(dat$strain.ID)[6]], dat$ko[dat$strain.ID == levels(dat$strain.ID)[7]],
    dat$ko[dat$strain.ID == levels(dat$strain.ID)[8]], dat$ko[dat$strain.ID ==
        levels(dat$strain.ID)[9]], dat$ko[dat$strain.ID == levels(dat$strain.ID)[10]],
    dat$ko[dat$strain.ID == levels(dat$strain.ID)[11]]))
ko_shared <- ko_shared[grep("K", ko_shared)]  #remove 'total' from the list of shred genes
```

```r
# Select intersect genes from DeSeq ouput
dat_shared <- dat[dat$ko %in% ko_shared, ]
dat_shared$ko <- factor(dat_shared$ko)

paste("Number of shared genes=", length(levels(dat_shared$ko)))
```

```
## [1] "Number of shared genes= 253"
```

4.2 MLMs DNA polymerases

```r
# Subset dataframe (only DNA polymerases)
ko.dna.poly <- (c(rownames(descriptionK[grep("DNA polymerase", descriptionK$description),
    , drop = FALSE])))
dat.dna.poly <- dat_shared[dat_shared$ko %in% ko.dna.poly, ]

# Model fitting
d.fit1 <- lme(abs(log2FoldChange.corr) ~ n_breadth_filtered.log, random = ~1 |
    direction/ko, data = dat.dna.poly, na.action = na.omit)
d.fit2 <- lme(abs(log2FoldChange.corr) ~ stress, random = ~1 | direction/ko,
    data = dat.dna.poly, na.action = na.omit)
d.fit3 <- lme((log2FoldChange.corr) ~ stress, random = ~1 | direction/ko,
    data = dat.dna.poly, na.action = na.omit)
d.fit4 <- lme(abs(log2FoldChange.corr) ~ stress_filtered, random = ~1 |
    direction/ko, data = dat.dna.poly, na.action = na.omit)
d.fit5 <- lme((log2FoldChange.corr) ~ stress_filtered, random = ~1 | direction/ko,
    data = dat.dna.poly, na.action = na.omit)

# Kolmogorv smirnov test
res.normality <- c(round(as.numeric(as.vector(ols_test_normality(resid(d.fit1)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(d.fit2)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(d.fit3)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(d.fit4)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(d.fit5)))[[1]][2]),
    2) >= 0.05)
names(res.normality) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")

kbl(res.normality, booktabs = TRUE)
```

	x
|L2FC|vsNB	TRUE
a(|L2FC|vsStress)	TRUE
a(L2FCvsStress)	TRUE
f(|L2FC|vsStress)	TRUE
f(L2FCvsStress)	TRUE

Kolmogorv-Smirnov tests indicate normal distribution of the residuals in all models

```r
# Create vector with the names of the fitted models
fit <- paste0("d.fit", 1:5)
mlm.dna.poly <- as.data.frame.matrix(sapply(1:5, MLM.stats))
colnames(mlm.dna.poly) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")
mlm.dna.poly$category = c("dna polymerases")

# Summary MLM statistic for DNA polymerases
kbl(mlm.dna.poly, booktabs = TRUE) %>% kable_styling(latex_options = "scale_down")
```

	|L2FC|vsNB	a(|L2FC|vsStress)	a(L2FCvsStress)	f(|L2FC|vsStress)	f(L2FCvsStress)	category
n	2.0000000	2.0000000	2.0000000	2.0000000	2.0000000	dna polymerases
slope	-0.1867074	0.4163817	1.1875607	0.6683344	1.9280204	dna polymerases
Peusdo.R2	0.0218296	0.0109984	0.1276729	0.0289193	0.2869594	dna polymerases
Pvalue	0.3978371	0.3986485	0.1507068	0.3297187	0.0659388	dna polymerases

4.3 MLMs RNA polymerases

```r
# Subset dataframe (only RNA polymerases)
ko.rna.poly <- ok_path[ok_path$path == "ko03020", 1]
dat.rna.poly <- dat_shared[dat_shared$ko %in% ko.rna.poly, ]

# Model fitting
r.fit1 <- lme(abs(log2FoldChange.corr) ~ n_breadth_filtered.log, random = ~1 |
    direction/ko, data = dat.rna.poly, na.action = na.omit)
r.fit2 <- lme(abs(log2FoldChange.corr) ~ stress, random = ~1 | direction/ko,
    data = dat.rna.poly, na.action = na.omit)
r.fit3 <- lme((log2FoldChange.corr) ~ stress, random = ~1 | direction/ko,
    data = dat.rna.poly, na.action = na.omit)
r.fit4 <- lme(abs(log2FoldChange.corr) ~ stress_filtered, random = ~1 |
    direction/ko, data = dat.rna.poly, na.action = na.omit)
r.fit5 <- lme((log2FoldChange.corr) ~ stress_filtered, random = ~1 | direction/ko,
    data = dat.rna.poly, na.action = na.omit)

# Kolmogorv smirnov test
res.normality <- c(round(as.numeric(as.vector(ols_test_normality(resid(r.fit1)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(r.fit2)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(r.fit3)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(r.fit4)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(r.fit5)))[[1]][2]),
    2) >= 0.05)
names(res.normality) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")

kbl(res.normality, booktabs = TRUE)
```

	x
|L2FC|vsNB	TRUE
a(|L2FC|vsStress)	TRUE
a(L2FCvsStress)	TRUE
f(|L2FC|vsStress)	TRUE
f(L2FCvsStress)	TRUE

Kolmogorv-Smirnov tests indicate normal distribution of the residuals in all models

```
# Create vector with the names of the fitted models
fit <- paste0("r.fit", 1:5)
mlm.rna.poly <- as.data.frame.matrix(sapply(1:5, MLM.stats))
colnames(mlm.rna.poly) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")
mlm.rna.poly$category = c("rna polymerases")

# Summary MLM statistic for RNA polymerases
kbl(mlm.rna.poly, booktabs = TRUE) %>% kable_styling(latex_options = "scale_down")
```

	|L2FC|vsNB	a(|L2FC|vsStress)	a(L2FCvsStress)	f(|L2FC|vsStress)	f(L2FCvsStress)	category
n	2.0000000	2.0000000	2.0000000	2.0000000	2.0000000	rna polymerases
slope	-0.4648802	0.8667588	1.8422201	1.9845155	3.0614772	rna polymerases
Peusdo.R2	0.1217839	0.0382368	0.1359754	0.2334679	0.3826729	rna polymerases
Pvalue	0.0409670	0.1132736	0.0300618	0.0039487	0.0057940	rna polymerases

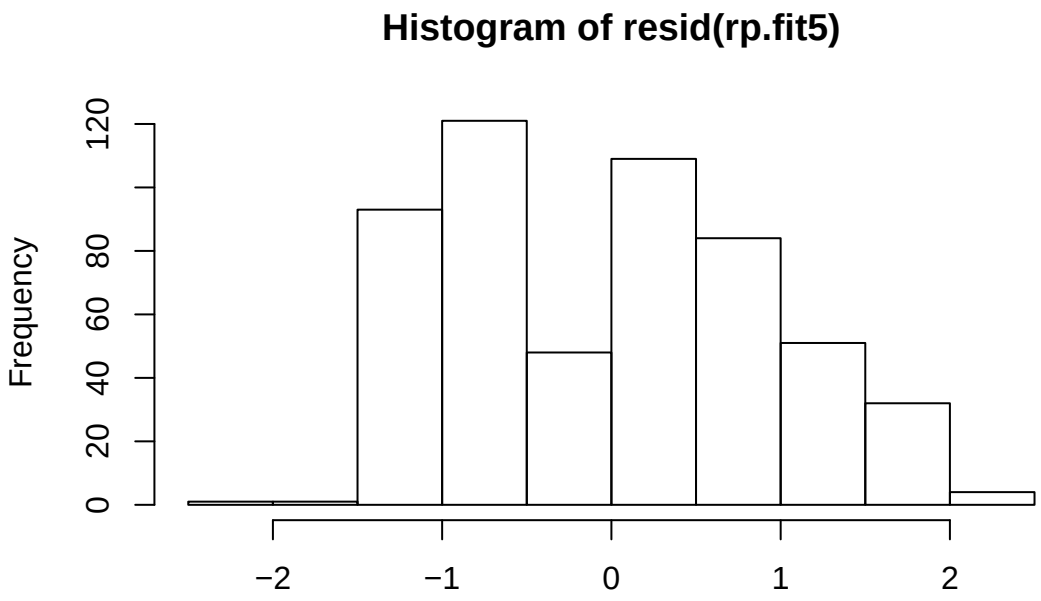
4.4 MLMs Ribosomal proteins

```
# Subset dataframe (only ribosomal proteins)
ko.rib <- ok_path[ok_path$path == "ko03010", 1]
dat.rib <- dat_shared[dat_shared$ko %in% ko.rib, ]

# Model fitting
rp.fit1 <- lme(abs(log2FoldChange.corr) ~ n_breadth_filtered.log, random = ~1 |
    direction/ko, data = dat.rib, na.action = na.omit)
rp.fit2 <- lme(abs(log2FoldChange.corr) ~ stress, random = ~1 | direction/ko,
    data = dat.rib, na.action = na.omit)
rp.fit3 <- lme((log2FoldChange.corr) ~ stress, random = ~1 | direction/ko,
    data = dat.rib, na.action = na.omit)
rp.fit4 <- lme(abs(log2FoldChange.corr) ~ stress_filtered, random = ~1 |
    direction/ko, data = dat.rib, na.action = na.omit)
rp.fit5 <- lme((log2FoldChange.corr) ~ stress_filtered, random = ~1 | direction/ko,
    data = dat.rib, na.action = na.omit)

# Model fitting after sqrt transformation
rp.fit1.sqrt <- lme(sqrt(abs(log2FoldChange.corr)) ~ n_breadth_filtered.log,
    random = ~1 | direction/ko, data = dat.rib, na.action = na.omit)
rp.fit2.sqrt <- lme(sqrt(abs(log2FoldChange.corr)) ~ stress, random = ~1 |
    direction/ko, data = dat.rib, na.action = na.omit)
rp.fit4.sqrt <- lme(sqrt(abs(log2FoldChange.corr)) ~ stress_filtered, random = ~1 |
    direction/ko, data = dat.rib, na.action = na.omit)
```

```
# Kolmogorv smirnov test
res.normality <- c(round(as.numeric(as.vector(ols_test_normality(resid(rp.fit1.sqrt)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(rp.fit2.sqrt)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(rp.fit3)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(rp.fit4.sqrt)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(rp.fit5)))[[1]][2]),
    2) >= 0.05)
names(res.normality) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")

kbl(res.normality, booktabs = TRUE)
```

	x
|L2FC|vsNB	TRUE
a(|L2FC|vsStress)	TRUE
a(L2FCvsStress)	TRUE
f(|L2FC|vsStress)	TRUE
f(L2FCvsStress)	FALSE

Residuals in the models rp.fit1,rp.fit2,rp.fit4 and rp.fit5 were accordingy to Kolmogorv-Smirnov tests not normally distributed. We assume that this was partly due to the large number of residuals obtained from the ribosomal protein models, as large numbers of input variables generally cause Kolmogorv-Smirnov tests to become very stringent. We applied sqrt transformations for data in the models rp.fit1,rp.fit2 and rp.fit4, which resulted in normally distributed residuals. For the model rp.fit5 none of several tested transformation resulted in a better fit of the residual to the normal distribution and no modifications were applied in this case. The distribution of residuals of not sqrt transformed data for rp.fit5 is displayed below. Statistic parameters were extracted from the models rp.fit1.sqrt, rp.fit2.sqrt, rp.fit3, rp.fit4.sqrt and rp.fit5. However, for displaying regression in Fig. 5g,h and i we used the not sqrt tranformed data.




```
# Create vector with the names of the fitted models
fit <- c("rp.fit1.sqrt", "rp.fit2.sqrt", "rp.fit3", "rp.fit4.sqrt", "rp.fit1")
mlm.rib <- as.data.frame.matrix(sapply(1:5, MLM.stats))
colnames(mlm.rib) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")
mlm.rib$category = c("ribosomal proteins")

# Summary MLM statistic for Ribosomal proteins
kbl(mlm.rib, booktabs = TRUE) %>% kable_styling(latex_options = "scale_down")
```

	|L2FC|vsNB	a(|L2FC|vsStress)	a(L2FCvsStress)	f(|L2FC|vsStress)	f(L2FCvsStress)	category
n	2.0000000	2.0000000	2.0000000	2.0000000	2.0000000	ribosomal proteins
slope	-0.0771234	0.2959684	1.1851025	0.2718107	-0.1510450	ribosomal proteins
Peusdo.R2	0.0908456	0.0205720	0.1326065	0.0816138	0.0551034	ribosomal proteins
Pvalue	0.0032134	0.0000033	0.0000000	0.0013790	0.0016415	ribosomal proteins

5 Transcriptional regulation of adaptation-related genes

To test if genes encoding for cell adaption against salt stress are stronger regulated in strains with broader NB, we defined 2 categories for potential adaptation-related genes: 1) Genes encoding the transport of osmoprotective compounds. We selected from all transporter genes that were expressed in at least one of the model strains a list of potential osmoprotectant transporters (Table S4 manuscript). 2) Heat-shock proteins (HSP). These proteins catalyze the folding and unfolding of macromolecules and are therefore involved in the repair of damaged macromolecules, commonly descibed an stress response mechanism. This category included all genes containing the text strings “heat-shock” or “chaperone” in the gene description. Because adaption mechanisms against salinity changes are highly species-specific we selected the genes in

the category not from the subset of shred genes, but the full data set. We assumed that the individual genes in the categories of the adaptation-related proteins act independent from each other and their reponse effect is additive. For this reason, we used the additive transcriptional response by summing up the fold changes of the individual genes for downstream statistical analyses.

5.1 Transport of osmoprotectants

```
# Subset dataframe (putative osmoprotectant transporters)
ko.op = trans_op$KEGG.ID[trans_op$Potential.osmolyte.transport == "TRUE"]
dat.op <- dat[dat$ko %in% ko.op, ]

# Create data frame with summed log2FoldChanges of putative
# osmoprotectant transporters Function to sum up logtransformed data
sumlog <- function(x) {
    # re-transform log of positive values
    pos <- ifelse(x > 0, (2^x), 0)
    # re-transform log of negative values after multiplication of -1, and
    # multiply result by -1
    neg <- ifelse(x < 0, -(2^-x), 0)
    # assign 1 to values between -1 and 1
    posneg <- ifelse((sum(pos, neg) > -1 & sum(pos, neg) < 1), 1, sum(pos,
        neg))
    # sum up re-transformed value, if sum is negative multiply by -1 and
    # take inverse value
    posneg <- ifelse(posneg > 0, posneg, -1/posneg)
    # log-transform the summed values
    log2(posneg)
}


dat.op.sum <- aggregate(log2FoldChange.corr ~ strain.ID + direction, data = dat.op,
    FUN = function(x) sumlog(x))
dat.op.sum$abs.log2FoldChange.corr = aggregate(log2FoldChange.corr ~ strain.ID +
    direction, data = dat.op, FUN = function(x) sumlog(abs(x)))[, 3]
dat.op.sum$stress = aggregate(stress ~ strain.ID + direction, data = dat.op,
    FUN = mean)[, 3]
dat.op.sum$n = aggregate(log2FoldChange.corr ~ strain.ID + direction, data = dat.op,
    FUN = length)[, 3]
dat.op.sum$n_breadth_filtered.log = aggregate(n_breadth_filtered.log ~
    strain.ID + direction, data = dat.op, FUN = mean, na.action = na.pass)[,
    3]
dat.op.sum$stress_filtered = aggregate(stress_filtered ~ strain.ID + direction,
    data = dat.op, FUN = mean, na.action = na.pass)[, 3]


# Model fitting
op.fit1 <- lme(abs.log2FoldChange.corr ~ n_breadth_filtered.log, random = ~1 |
    direction, data = dat.op.sum, na.action = na.omit)
op.fit2 <- lme(abs.log2FoldChange.corr ~ stress, random = ~1 | direction,
    data = dat.op.sum, na.action = na.omit)
op.fit3 <- lme(log2FoldChange.corr ~ stress, random = ~1 | direction, data = dat.op.sum,
    na.action = na.omit)
```

```
op.fit4 <- lme(abs.log2FoldChange.corr ~ stress_filtered, random = ~1 |
    direction, data = dat.op.sum, na.action = na.omit)
op.fit5 <- lme(log2FoldChange.corr ~ stress_filtered, random = ~1 | direction,
    data = dat.op.sum, na.action = na.omit)

op.fit3.inv <- lme(1/(log2FoldChange.corr + 1) ~ stress, random = ~1 |
    direction, data = dat.op.sum, na.action = na.omit)

# Kolmogorv smirnov test
res.normality <- c(round(as.numeric(as.vector(ols_test_normality(resid(op.fit1)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(op.fit2)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(op.fit3.inv)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(op.fit4)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(op.fit5)))[[1]][2]),
    2) >= 0.05)
names(res.normality) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")

kbl(res.normality, booktabs = TRUE)
```

	x
|L2FC|vsNB	TRUE
a(|L2FC|vsStress)	TRUE
a(L2FCvsStress)	TRUE
f(|L2FC|vsStress)	TRUE
f(L2FCvsStress)	TRUE

Residuals in the models op.fit3 was according to Kolmogorv-Smirnov tests not normally distributes. We applied inverse transformations for data in the models op.fit3.inv, which resulted in normally distributed residuals. Statistic parameters were extracted from the models op.fit, op.fit2, op.fit3.inv, op.fit4 and op.fit5. However, for displaying regression in Fig. 5 we used the non-transformed data from op.fit3.

```
# Create vector with the names of the fitted models
fit <- c("op.fit1", "op.fit2", "op.fit3", "op.fit4", "op.fit5")
mlm.op <- as.data.frame.matrix(sapply(1:5, MLM.stats))
colnames(mlm.op) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")
mlm.op$category = c("osmoprotectant transporter")

# Summary MLM statistic for osmoprotectant transporter
kbl(mlm.op, booktabs = TRUE) %>% kable_styling(latex_options = "scale_down")
```

	|L2FC|vsNB	a(|L2FC|vsStress)	a(L2FCvsStress)	f(|L2FC|vsStress)	f(L2FCvsStress)	category
n	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	osmoprotectant transporter
slope	0.5222646	-0.2277750	-0.4424350	-1.0400866	-5.8050515	osmoprotectant transporter
Peusdo.R2	0.1864707	0.0027907	0.0220530	0.0758886	0.0297980	osmoprotectant transporter
Pvalue	0.0761437	0.7668802	0.9397963	0.2708994	0.4947858	osmoprotectant transporter

5.2 Heat-Shock proteins

```
# Subset dataframe (heat shock proteins)
ko.HP <- c(rownames(descriptionK[grep("heat shock protein", descriptionK$description),
    , drop = FALSE]), rownames(descriptionK[grep("chaperon", descriptionK$description),
    , drop = FALSE]))
dat.HP <- dat[dat$ko %in% ko.HP, ]

# Create data frame with summed log2FoldChanges of putative
# osmoprotectant transporters
dat.HP.sum <- aggregate(log2FoldChange.corr ~ strain.ID + direction, data = dat.HP,
    FUN = function(x) sumlog(x))
dat.HP.sum$abs.log2FoldChange.corr = aggregate(log2FoldChange.corr ~ strain.ID +
    direction, data = dat.HP, FUN = function(x) sumlog(abs(x)))[, 3]
dat.HP.sum$stress = aggregate(stress ~ strain.ID + direction, data = dat.HP,
    FUN = mean)[, 3]
dat.HP.sum$n = aggregate(log2FoldChange.corr ~ strain.ID + direction, data = dat.HP,
    FUN = length)[, 3]
dat.HP.sum$n_breadth_filtered.log = aggregate(n_breadth_filtered.log ~
    strain.ID + direction, data = dat.HP, FUN = mean, na.action = na.pass)[,
    3]
dat.HP.sum$stress_filtered = aggregate(stress_filtered ~ strain.ID + direction,
    data = dat.HP, FUN = mean, na.action = na.pass)[, 3]

# Model fitting
HP.fit1 <- lme(abs.log2FoldChange.corr ~ n_breadth_filtered.log, random = ~1 |
    direction, data = dat.HP.sum, na.action = na.omit)
HP.fit2 <- lme(abs.log2FoldChange.corr ~ stress, random = ~1 | direction,
    data = dat.HP.sum, na.action = na.omit)
HP.fit3 <- lme((log2FoldChange.corr) ~ stress, random = ~1 | direction,
    data = dat.HP.sum, na.action = na.omit)
HP.fit4 <- lme(abs.log2FoldChange.corr ~ stress_filtered, random = ~1 |
    direction, data = dat.HP.sum, na.action = na.omit)
HP.fit5 <- lme(log2FoldChange.corr ~ stress_filtered, random = ~1 | direction,
    data = dat.HP.sum, na.action = na.omit)

# Kolmogorv Smirnov tests
res.normality <- c(round(as.numeric(as.vector(ols_test_normality(resid(HP.fit1)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(HP.fit2)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(HP.fit3)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(HP.fit4)))[[1]][2]),
    2) >= 0.05, round(as.numeric(as.vector(ols_test_normality(resid(HP.fit5)))[[1]][2]),
    2) >= 0.05)
names(res.normality) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")

kbl(res.normality, booktabs = TRUE)
```

	x
|L2FC|vsNB	TRUE
a(|L2FC|vsStress)	TRUE
a(L2FCvsStress)	TRUE
f(|L2FC|vsStress)	TRUE
f(L2FCvsStress)	TRUE

Kolmogorv-Smirnov tests indicate normal distribution of the residuals in all models

```
# Create vector with the names of the fitted models
fit <- paste0("HP.fit", 1:5)
mlm.HP <- as.data.frame.matrix(sapply(1:5, MLM.stats))
colnames(mlm.HP) = c("|L2FC|vsNB", "a(|L2FC|vsStress)", "a(L2FCvsStress)",
    "f(|L2FC|vsStress)", "f(L2FCvsStress)")
mlm.HP$category = c("osmoprotectant transporter")

# Summary MLM statistic for heat-shock proteins
kbl(mlm.HP, booktabs = TRUE) %>% kable_styling(latex_options = "scale_down")
```

	|L2FC|vsNB	a(|L2FC|vsStress)	a(L2FCvsStress)	f(|L2FC|vsStress)	f(L2FCvsStress)	category
n	1.0000000	1.0000000	1.0000000	1.0000000	1.0000000	osmoprotectant transporter
slope	-0.2369684	1.2605298	2.0645159	1.0910924	0.9078842	osmoprotectant transporter
Peusdo.R2	0.0593813	0.1406214	0.1684146	0.1307456	0.0018572	osmoprotectant transporter
Pvalue	0.3319392	0.0295973	0.5704785	0.1431331	0.8654826	osmoprotectant transporter

6 Candidate stress marker genes

We applied MLM on each of the 253 genes shared by the 11 strains. We used stress as a fixed factor and the three replicate stress pairwise comparison as random factors (S1:S2,S1:S3,S2:S3). The MLMs performed considered the direction of the gene regulation under stress conditions (upregulation or downregulation). The resulting P-values were adjusted to account for false discovery rates by multiple comparisons (BH correction).

6.1 Loop for testing normality of residuals form single gene regulation mixed linear model (MLM)

```
# Creating empty objects for the loop
mod = list()
tmp.res = data.frame(Model = NA, Row.names = NA, Estimate = NA, Std..Error = NA,
    t.value = NA, P.value = NA)
mod.res = data.frame(Model = NA, Row.names = NA, Estimate = NA, Std..Error = NA,
    t.value = NA, P.value = NA)

# Residual diagnosis
t = NA
# Normality test for each single regression
for (i in 1:length(levels(dat_shared$ko))) {
    t[i] = as.numeric(as.vector(ols_test_normality(resid(lmer((log2FoldChange.corr) ~
        stress + (1 | direction), data = dat_shared[dat_shared$ko == levels(dat_shared$ko)[i],
        ]))))[[1]])[2])
```

```
}

# Residual diagnosis
t = NA
# Normality test for each single regression
for (i in 1:length(levels(dat_shared$ko))) {
    t[i] = as.numeric(as.vector(ols_test_normality(resid(lme((log2FoldChange.corr) ~
        stress, random = ~1 | direction, na.action = na.omit, data = dat_shared[dat_shared$ko ==
        levels(dat_shared$ko)[i], ]))[[1]])[2])
}
print(paste("number of models that did not pass the normality test: ",
    sum(t <= 0.05)))
```

```
## [1] "number of models that did not pass the normality test:  0"
```

Kolmogorv-Smirnov tests indicate normal distribution of the residuals in all models

6.2 MLMs to test correlation between gene regulation of individual genes and stress

```
# Linear mixed models for single gene regulation
for (i in 1:length(levels(dat_shared$ko))) {
    mod[[i]] = lme((log2FoldChange.corr) ~ stress, random = ~1 | direction,
        na.action = na.omit, data = dat_shared[dat_shared$ko == levels(dat_shared$ko)[i],
            ])

    tmp.res$Model = "LMM"
    # extractiong values from the model output
    tmp.res$Row.names = levels(dat_shared$ko)[i]
    tmp.res$Estimate = coef(summary(mod[[i]]))[2]
    tmp.res$Std..Error = coef(summary(mod[[i]]))[4]
    tmp.res$t.value = coef(summary(mod[[i]]))[8]
    tmp.res$P.value = coef(summary(mod[[i]]))[10]
    mod.res = rbind(tmp.res, mod.res)
}
mod.res = mod.res[complete.cases(mod.res), ]  # Remove row with NA
as_tibble(mod.res)
```

```
## # A tibble: 253 x 6
##    Model Row.names Estimate Std..Error t.value P.value
##    <chr> <chr>        <dbl>      <dbl>   <dbl>   <dbl>
##  1 LMM   K19577       0.541      0.782   0.692  0.494
##  2 LMM   K16137       0.600      0.813   0.739  0.466
##  3 LMM   K16012      -1.52       1.09   -1.39   0.175
##  4 LMM   K15268       2.17       0.919   2.36   0.0253
##  5 LMM   K14742       0.390      1.01    0.384  0.704
##  6 LMM   K13953      -3.59       1.65   -2.18   0.0376
##  7 LMM   K13566       0.502      0.967   0.519  0.608
##  8 LMM   K12132       0.127      1.01    0.126  0.901
##  9 LMM   K11754       1.32       1.15    1.14   0.262
## 10 LMM   K11753       1.25       1.07    1.17   0.253
## # ... with 243 more rows
```

6.3 Formating output

```
# Sort by P-value
mod.res <- mod.res[with(mod.res, order(P.value)), ]

# P-values adjusted for multiple sampling (Bonferroni)
mod.res$lm.pad = p.adjust(mod.res$P.value, "BH")

# Pseudo-R2
for (i in 1:dim(mod.res)[1]) {
    mod.res$r2var[i] = r.squaredGLMM(mod[[i]])[2]
}
# Merge with gene descriptions
res.lm1 <- merge(mod.res, descriptionK, by.x = "Row.names", by.y = 0, all.x = T)[,
    -10]
```

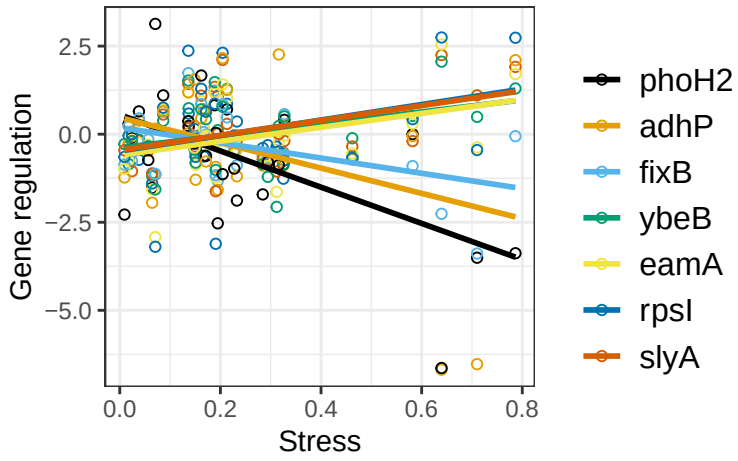
6.4 Filtering genes (KO number) by P-value <0.05

We report 7 genes with significant regression of transcription levels against (P<0.05) as candidate stress marker genes.

```
# Select candidate stress marker gene (MLM Pvalue<0.05)
dat.sm = dat_shared[dat_shared$ko %in% res.lm1$Row.names[res.lm1$P.value <
    0.05], ]
dat.sm = droplevels(dat.sm)

# Statistical results LMM
res.lm2 = res.lm1[res.lm1$P.value < 0.05, ]
dat.sm$ko <- factor(dat.sm$ko, levels = res.lm2$Row.names[order(res.lm2$Estimate)])
# Formating gene descriptions
levels(dat.sm$ko) = str_split(res.lm2$description[order(res.lm2$Estimate)],
    ";", simplify = TRUE)[, 1]
# Rounding values
res.lm2[, 3:8] = round(res.lm2[, 3:8], 3)
```

6.5 Results for candidate stress marker genes

This Figure displays the regression of candidate stress marker gene regulation against stress exposure levels. This graphic displays the regression of 7 candidate stress marker genes that were selected because of their significant correlation (P<0.05, no adjustments for multiple testing) against the exposure of the 11 model strain to osmotic stress (Fig. 7).



	Row.names	Estimate	P.value	lm.pad	r2var	description
226	K07175	-5.141	0.000	0.088	0.036	phoH2; PhoH-like ATPase
248	K13953	-3.586	0.038	0.982	0.089	adhP; alcohol dehydrogenase, propanol-preferring [EC:1.1.1.1]
175	K03522	-2.098	0.015	0.982	0.154	fixB, etfA; electron transfer flavoprotein alpha subunit
238	K09710	2.055	0.018	0.982	0.041	ybeB; ribosome-associated protein
250	K15268	2.168	0.025	0.982	0.058	eamA; O-acetylserine/cysteine efflux transporter
151	K02996	2.478	0.045	0.982	0.080	RP-S9, MRPS9, rpsI; small subunit ribosomal protein S9
215	K06075	2.496	0.004	0.450	0.077	slyA; MarR family transcriptional regulator, transcriptional regulator for hemolysin