

# Larval performance and substrate pH

Stijn Schreven

4 March 2021

## Contents

<b>Load packages</b>	<b>1</b>
<b>Input files</b>	<b>2</b>
<b>Larval performance parameters</b>	<b>2</b>
1. Prepare data . . . . .	2
2. Plot presets . . . . .	2
3. Chicken feed . . . . .	3
4. Chicken manure . . . . .	7
5. Plots . . . . .	10
6. Export plots . . . . .	15
<b>Substrate pH</b>	<b>15</b>
1. Prepare data . . . . .	16
2. GLMM regression . . . . .	16
2.3. Export ANOVA tables . . . . .	18
3. Errorbar plot . . . . .	18

## Load packages

```
library(plyr)
library(nlme)
library(lme4)
library(sciplot)
library(emmeans)
library(purrr)
library(ggplot2)
library(viridis)
library(car)
```

## Input files

```
perf <- read.delim("./input_data/Schreven_Ch4_data_performance.txt", header = T)
```

## Larval performance parameters

### 1. Prepare data

```
# add column Day for batch effects
## day 1 is rep1+2, 2 = 3+4, 3 = 5+6, for diets separate!
perf$Rep <- as.factor(perf$Rep)
perf$Day <- ifelse(perf$Rep %in% c(1,2), "1", ifelse(perf$Rep %in% c(3,4), "2", "3"))
perf$Day <- as.factor(perf$Day)

# exclude CF containers 6 and 7 because they were contaminated by fungal overgrowth
perf <- subset(perf, !Container %in% c(6,7))

# subset columns
harvest <- perf[, c("Diet", "Day", "Treatment", "Survival", "pPrepupae",
                   "dmLarvInd", "dmLarvTot", "pDMRes")]

# rescale units of some variables for convenient interpretation
harvest$Survival <- 100 * harvest$Survival # in %
harvest$pPrepupae <- 100 * harvest$pPrepupae # in %
harvest$pDMRes <- 100 * harvest$pDMRes # in %
harvest$pMoist <- 100 - harvest$pDMRes # in %

# separate per diet
harv.cf <- subset(harvest, Diet == "CF")
harv.cf <- subset(harv.cf, Treatment != "Ss/E") # exclude: harvest at t = 22 (not t = 15)
harv.cf$Treatment <- droplevels(harv.cf$Treatment)
harv.cf$Day <- droplevels(harv.cf$Day)
harv.cm <- subset(harvest, Diet == "CM")

# melt
harv.m <- reshape2::melt(harvest)

## Using Diet, Day, Treatment as id variables

# summarize
harv.sum <- ddply(harv.m, ~ Diet + Treatment + variable, summarise,
                  mean = mean(value), median = median(value),
                  sd = sd(value), se = se(value))
```

### 2. Plot presets

```

theme_perf <- theme_classic() +
  theme(panel.grid.major = element_line(colour = "grey80"),
        panel.grid.major.x = element_blank(),
        panel.spacing = unit(.5, "lines"),
        panel.border = element_rect(color = "black", fill = NA, size = .5),
        strip.background = element_blank(),
        strip.placement = "outside",
        legend.position = "none",
        text = element_text(size = 20))

theme_ph <- theme_classic() + theme(text = element_text(size = 20),
  panel.grid.major = element_line(colour = "grey80"),
  panel.grid.major.x = element_blank(),
  panel.spacing = unit(.5, "lines"),
  panel.border = element_rect(color = "black", fill = NA, size = .5),
  strip.background = element_blank(),
  strip.placement = "outside",
  axis.title.y = element_text(angle = 0, vjust = .5))

labs_perf <- as_labeller(c(CM = "chicken manure", CF = "chicken feed"))

```

### 3. Chicken feed

Linear regression. Fixed term: Treatment. Analyse diets separately: performed in different batches (Day over time, different sample sizes ( $n = 4$  for CF,  $n = 6$  for CM, per treatment)).

Mixed model selection:

- random effect for Day (batch);
- variance structure.

Use GLS/ LMM models (Zuur *et al.* 2009). If residuals are not normal, use GLMM (Gamma), if that does not improve the diagnostics, use non-parametric tests.

For chicken feed: excluded Ss/E samples as they are from day 22 (incomparable). Only included in figures, not in statistics.

#### 3.0. Model validation

Customize this code chunk for the model and data.frame in question. For each parametric model, check the residuals (“normalized” or “pearson”) using the following code. In the subsequent code chunks, I have excluded the validation part, as well as the linear model selection if a non-parametric test was chosen eventually because of non-normal residual distribution.

```

# mod = model
# data = data.frame with factors

# LM, GLS, LMM:

# extract model residuals and fitted values
mres <- resid(mod, type = "normalized")
mfit <- fitted(mod)

```

```

# check normality of residuals
qqnorm(mres); abline(0,1)
hist(mres)

# check normality of random term (LMM)
qqnorm(mod, ~ ranef (..))

# check homoskedasticity
# (plot residuals against fitted values or fixed/random factors)
plot(mod)
plot(mres ~ data$Treatment); abline(0,0)
plot(mres ~ data$Day); abline(0,0)
plot(mres ~ data$Timepoint); abline(0,0)

# GLM, GLMM:

# check normality of residuals
Mod.res <- data.frame(data[,c(1:3)], # factors to plot against
                      res = residuals(mod, type = "pearson"),
                      fit = fitted(mod))
ggplot(Mod.res, aes(sample = res)) + stat_qq() + stat_qq_line()
hist(Mod.res$res)

# check homoskedasticity
# (plot residuals against fitted values or fixed/random factors)
plot(res ~ fit, Mod.res); abline(0,0)
plot(res ~ Treatment, Mod.res); abline(0,0)
plot(res ~ Day, Mod.res); abline(0,0)

```

### 3.1. Survival rate

```

# Variance structure selection
S.m0 <- gls(Survival ~ Treatment, data = harv.cf, method = "REML")
S.m1 <- lme(Survival ~ Treatment, data = harv.cf, method = "REML", random = ~1|Day)
S.m2 <- update(S.m0, weights = varIdent(form = ~1|Treatment))
S.m3 <- update(S.m1, weights = varIdent(form = ~1|Treatment))
AIC(S.m0, S.m1, S.m2, S.m3)

```

```

##      df      AIC
## S.m0  4 72.51661
## S.m1  5 74.51661
## S.m2  6 70.98583
## S.m3  7 72.66916

```

```

# model m2 is best: variance structure for Treatment.

```

```

# final model parameter estimation: REML
S.mDef <- gls(Survival ~ Treatment, data = harv.cf, method = "REML",
              weights = varIdent(form = ~1|Treatment))
anova(S.mDef)

```

```
## Denom. DF: 9
##          numDF  F-value p-value
## (Intercept)    1 8386.635 <.0001
## Treatment      2  26.454  2e-04

# multiple comparisons emmeans
CLD(emmeans(S.mDef, ~ Treatment), method = "tukey", Letters = letters)

## Treatment emmean    SE   df lower.CL upper.CL .group
## S/E        51.2 4.404 3.03    37.3    65.2    a
## Si/E        83.8 0.946 3.00    80.7    86.8    b
## Si/Es       86.1 3.951 3.03    73.6    98.6    b
##
## Degrees-of-freedom method: satterthwaite
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 3 estimates
## significance level used: alpha = 0.05
```

### 3.2. Prepupae

```
# non-parametric test: Kruskal-Wallis
kruskal.test(pPrepupae ~ Treatment, data = harv.cf)

##
## Kruskal-Wallis rank sum test
##
## data:  pPrepupae by Treatment
## Kruskal-Wallis chi-squared = 5.3087, df = 2, p-value = 0.07034

# conclusion: no differences.
```

### 3.3. Individual larval weight

```
# non-parametric test: Kruskal-Wallis
kruskal.test(dmLarvInd ~ Treatment, data = harv.cf)

##
## Kruskal-Wallis rank sum test
##
## data:  dmLarvInd by Treatment
## Kruskal-Wallis chi-squared = 6.5769, df = 2, p-value = 0.03731

# multiple comparisons (pairwise Wilcoxon tests)
cf.combins <- combn(levels(harv.cf$Treatment), 2)
cf.params <- split(as.vector(cf.combins), rep(1:ncol(cf.combins), each = nrow(cf.combins)))

l.cf.wx.map <- map(.x = cf.params,
                  .f = ~ wilcox.test(formula = dmLarvInd ~ Treatment,
```

```

data = subset(harv.cf, Treatment %in% .x)))
I.cf.wx.pval <- t(data.frame(map(.x = I.cf.wx.map, .f = "p.value")))
I.cf.wx.pval <- as.data.frame(p.adjust(I.cf.wx.pval, method="fdr"))
row.names(I.cf.wx.pval) <- unlist(map(.x = cf.params, .f = ~ paste0(.x, collapse = " vs. ")))
colnames(I.cf.wx.pval) <- "P"
print(I.cf.wx.pval)

```

```

##
##
## S/E vs. Si/E 0.08571429
## S/E vs. Si/Es 0.08571429
## Si/E vs. Si/Es 0.68571429

```

```

# no significant pairwise differences.

```

### 3.4. Total larval biomass

```

# non-parametric test: Kruskal-Wallis
kruskal.test(dmLarvTot ~ Treatment, data = harv.cf)

```

```

##
## Kruskal-Wallis rank sum test
##
## data: dmLarvTot by Treatment
## Kruskal-Wallis chi-squared = 2.9231, df = 2, p-value = 0.2319

```

```

# no significant differences.

```

### 3.5. Moisture content

```

# non-parametric test: Kruskal-Wallis
kruskal.test(pMoist ~ Treatment, data = harv.cf)

```

```

##
## Kruskal-Wallis rank sum test
##
## data: pMoist by Treatment
## Kruskal-Wallis chi-squared = 7.7308, df = 2, p-value = 0.02095

```

```

# significant differences.

```

```

# pairwise Wilcoxon tests
M.combins <- combn(levels(harv.cf$Treatment), 2)
M.params <- split(as.vector(M.combins), rep(1:ncol(M.combins), each = nrow(M.combins)))

M.wx.map <- map(.x = M.params,
               .f = ~ wilcox.test(formula = pMoist ~ Treatment,
                                data = subset(harv.cf, Treatment %in% .x)))

```

```
M.wx.pval <- t(data.frame(map(.x = M.wx.map, .f = "p.value")))
M.wx.pval <- as.data.frame(p.adjust(M.wx.pval, method = "fdr"))

row.names(M.wx.pval) <- unlist(map(.x = M.params, .f = ~ paste0(.x, collapse = " vs. ")))
colnames(M.wx.pval) <- "P"
print(M.wx.pval)
```

```
##                P
## S/E vs. Si/E    0.04285714
## S/E vs. Si/Es   0.04285714
## Si/E vs. Si/Es 0.48571429
```

```
# conclusion: S/E has signif. higher moisture content than inoculated diets.
```

## 4. Chicken manure

### 4.1. Survival

```
# Variance structure selection
S.cm0 <- gls(Survival ~ Treatment, data = harv.cm, method = "REML")
S.cm1 <- lme(Survival ~ Treatment, data = harv.cm, method = "REML",
            random = ~1|Day)
S.cm2 <- update(S.cm0, weights = varIdent(form = ~1|Treatment))
S.cm3 <- update(S.cm1, weights = varIdent(form = ~1|Treatment))
AIC(S.cm0, S.cm1, S.cm2, S.cm3)
```

```
##      df      AIC
## S.cm0  5 170.3193
## S.cm1  6 172.2975
## S.cm2  8 172.6958
## S.cm3  9 174.6958
```

```
## null model is best.
```

```
# final model parameter estimation: LM since no variance structure
S.cmDef <- lm(Survival ~ Treatment, data = harv.cm)
anova(S.cmDef)
```

```
## Analysis of Variance Table
##
## Response: Survival
##           Df Sum Sq Mean Sq F value Pr(>F)
## Treatment  3  850.86   283.62   2.2885 0.1095
## Residuals 20 2478.64   123.93
```

```
# multiple comparisons emmeans
CLD(emmeans(S.cmDef, ~Treatment))
```

```
## Treatment emmean SE df lower.CL upper.CL .group
## Si/Es 71.4 4.54 20 61.9 80.8 1
## S/E 78.8 4.54 20 69.4 88.3 1
## Si/E 79.0 4.54 20 69.5 88.5 1
## Ss/E 88.2 4.54 20 78.7 97.6 1
##
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 4 estimates
## significance level used: alpha = 0.05
```

## 4.2. Prepupae

```
# Non-parametric test: Kruskal-Wallis
kruskal.test(pPrepupae ~ Treatment, data = harv.cm)

##
## Kruskal-Wallis rank sum test
##
## data: pPrepupae by Treatment
## Kruskal-Wallis chi-squared = 4.257, df = 3, p-value = 0.235

# no significant differences.
```

## 4.3. Individual larval weight

```
# non-parametric test: Kruskal-Wallis
kruskal.test(dmLarvInd ~ Treatment, data = harv.cm)

##
## Kruskal-Wallis rank sum test
##
## data: dmLarvInd by Treatment
## Kruskal-Wallis chi-squared = 12.716, df = 3, p-value = 0.005293

# pairwise Wilcoxon tests
I.combins <- combn(levels(harv.cm$Treatment), 2)
I.params <- split(as.vector(I.combins), rep(1:ncol(I.combins), each = nrow(I.combins)))

I.wx.map <- map(.x = I.params,
               .f = ~ wilcox.test(formula = dmLarvInd ~ Treatment,
                                   data = subset(harv.cm, Treatment %in% .x)))
I.wx.pval <- t(data.frame(map(.x = I.wx.map, .f = "p.value")))
I.wx.pval <- as.data.frame(p.adjust(I.wx.pval, method="fdr"))
row.names(I.wx.pval) <- unlist(map(.x = I.params, .f = ~ paste0(.x, collapse = " vs. ")))
colnames(I.wx.pval) <- "P"
print(I.wx.pval)

## P
```



```
## S/E vs. Si/E 0.25909924
## S/E vs. Si/Es 0.80985488
## S/E vs. Ss/E 0.01298701
## Si/E vs. Si/Es 0.80985488
## Si/E vs. Ss/E 0.01499437
## Si/Es vs. Ss/E 0.03224483
```

#### 4.4. Total larval biomass

```
# non-parametric test: Kruskal-Wallis
kruskal.test(dmLarvTot ~ Treatment, data = harv.cm)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: dmLarvTot by Treatment
## Kruskal-Wallis chi-squared = 13.947, df = 3, p-value = 0.002979
```

```
# pairwise Wilcoxon tests
Y.combins <- combn(levels(harv.cm$Treatment), 2)
Y.params <- split(as.vector(Y.combins), rep(1:ncol(Y.combins), each = nrow(Y.combins)))
Y.wx.map <- map(.x = Y.params,
               .f = ~ wilcox.test(formula = dmLarvTot ~ Treatment,
                                   data = subset(harv.cm, Treatment %in% .x)))
Y.wx.pval <- t(data.frame(map(.x = Y.wx.map, .f = "p.value")))
Y.wx.pval <- as.data.frame(p.adjust(Y.wx.pval, method="fdr"))
row.names(Y.wx.pval) <- unlist(map(.x = Y.params, .f = ~ paste0(.x, collapse = " vs. ")))
colnames(Y.wx.pval) <- "P"
print(Y.wx.pval)
```

```
##
## P
## S/E vs. Si/E 0.360389610
## S/E vs. Si/Es 0.472727273
## S/E vs. Ss/E 0.004329004
## Si/E vs. Si/Es 0.818181818
## Si/E vs. Ss/E 0.004329004
## Si/Es vs. Ss/E 0.004329004
```

```
# conclusion: significant effect of treatment on biomass, Ss/E is different from rest.
```

#### 4.5. Moisture content

```
# non-parametric test: Kruskal-Wallis
kruskal.test(pMoist ~ Treatment, data = harv.cm)
```

```
##
## Kruskal-Wallis rank sum test
##
## data: pMoist by Treatment
## Kruskal-Wallis chi-squared = 13.287, df = 3, p-value = 0.004056
```

```

# pairwise Wilcoxon tests
M.combins <- combn(levels(harv.cm$Treatment), 2)
M.params <- split(as.vector(M.combins), rep(1:ncol(M.combins), each = nrow(M.combins)))
M.wx.map <- map(.x = M.params,
               .f = ~ wilcox.test(formula = pMoist ~ Treatment,
                                data = subset(harv.cm, Treatment %in% .x)))
M.wx.pval <- t(data.frame(map(.x = M.wx.map, .f = "p.value")))
M.wx.pval <- as.data.frame(p.adjust(M.wx.pval, method="fdr"))

row.names(M.wx.pval) <- unlist(map(.x = M.params,
                                .f = ~ paste0(.x, collapse = " vs. ")))
colnames(M.wx.pval) <- "P"
print(M.wx.pval)

```

```

##                                P
## S/E vs. Si/E    1.000000000
## S/E vs. Si/Es   0.838961039
## S/E vs. Ss/E    0.004329004
## Si/E vs. Si/Es  0.727272727
## Si/E vs. Ss/E   0.004329004
## Si/Es vs. Ss/E  0.004329004

```

```

# conclusion: Ss/E has lower moisture content than rest.

```

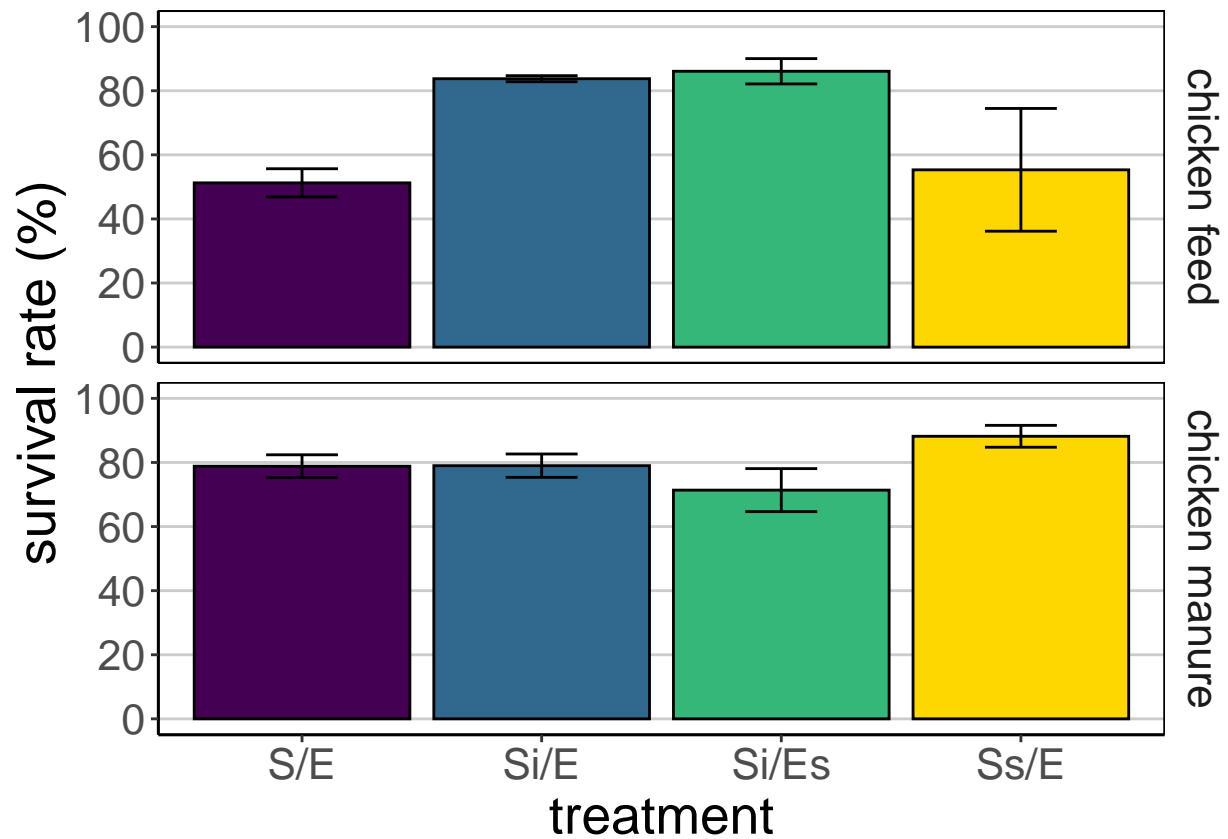
## 5. Plots

Figure 2 and Supplementary Figures S1 and S3.

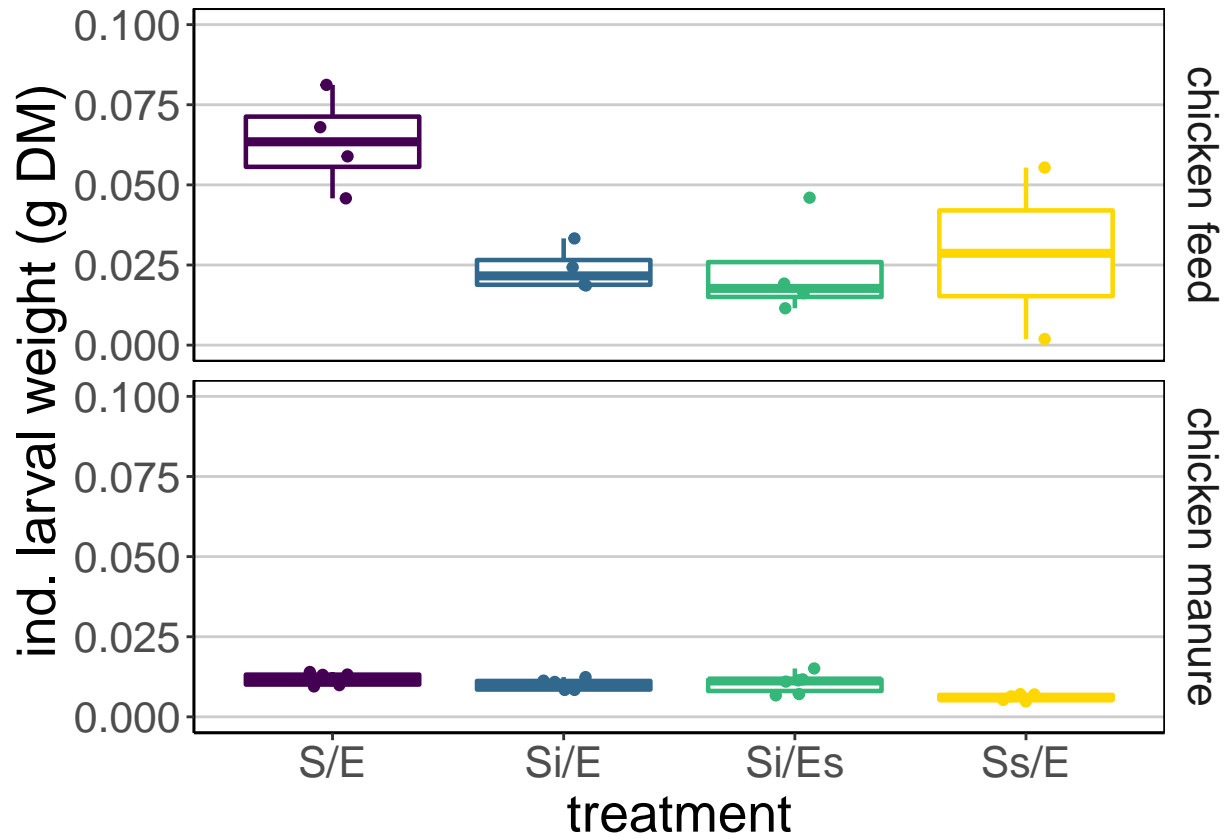
```

# survival rate: Figure 2A
pS1 <- ggplot(harv.sum[harv.sum$variable == "Survival",],
             aes(x = Treatment, y = mean, group = Treatment)) +
  geom_col(aes(fill = Treatment), position = position_dodge(),
           colour = "black") +
  geom_errorbar(aes(ymin = mean - se, ymax = mean + se), width = .3,
               position = position_dodge(0.9)) +
  scale_fill_manual(values = c("#440154FF", "#31688EFF", "#35B779FF", "gold")) +
  labs(y = "survival rate (%)", x = "treatment") +
  scale_y_continuous(limits = c(0, 100), n.breaks = 6) +
  facet_grid(Diet ~ ., labeller = labs_perf) +
  theme_perf
pS1

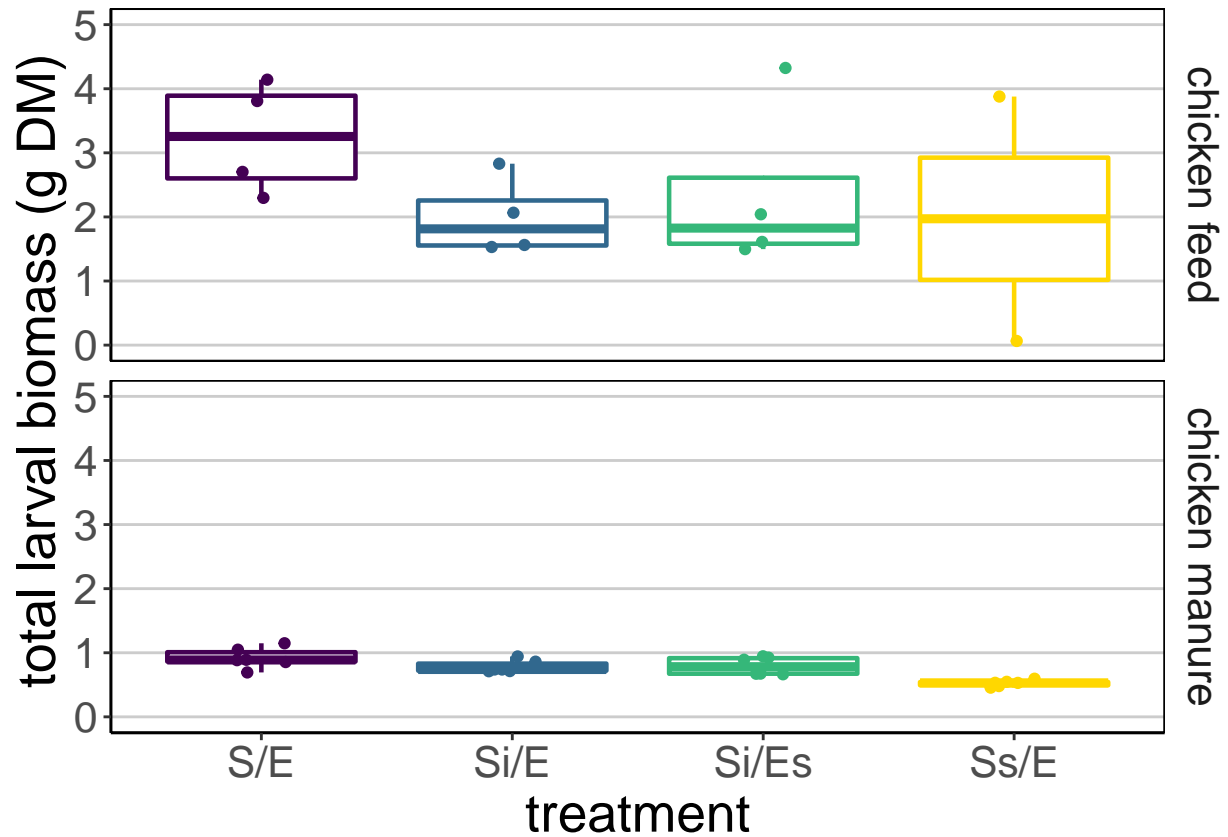
```



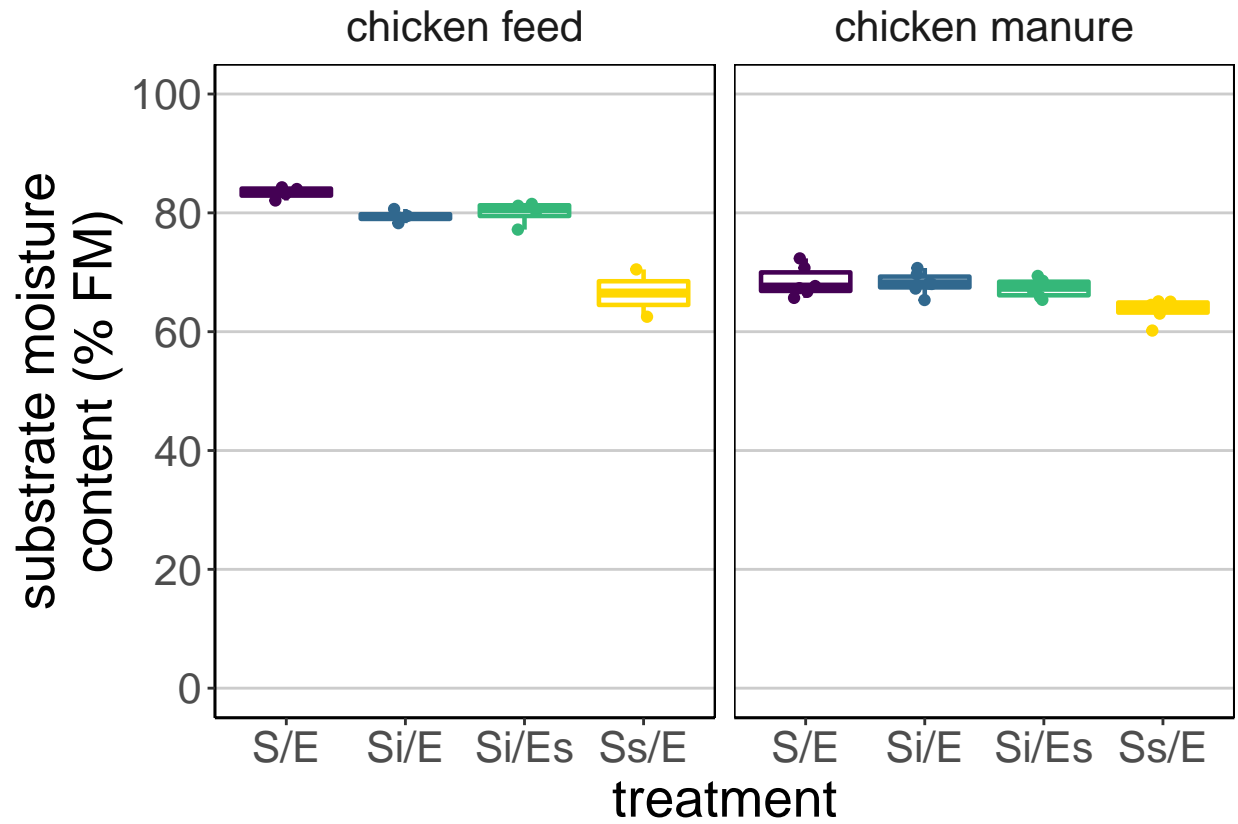
```
# individual larval weight: Figure 2B
pS2 <- ggplot(data = harv.m[harv.m$variable == "dmLarvInd",],
  aes(x = Treatment, y = value, colour = Treatment)) +
  geom_boxplot(size = .8, outlier.alpha = 0) +
  geom_point(position = position_jitter(width=0.1, height=0)) +
  scale_color_manual(values = c("#440154FF", "#31688EFF", "#35B779FF", "gold")) +
  labs(y = "ind. larval weight (g DM)", x = "treatment") +
  scale_y_continuous(limits = c(0, .1), n.breaks = 5) +
  facet_grid(Diet ~ ., labeller = labs_perf) +
  theme_perf
pS2
```



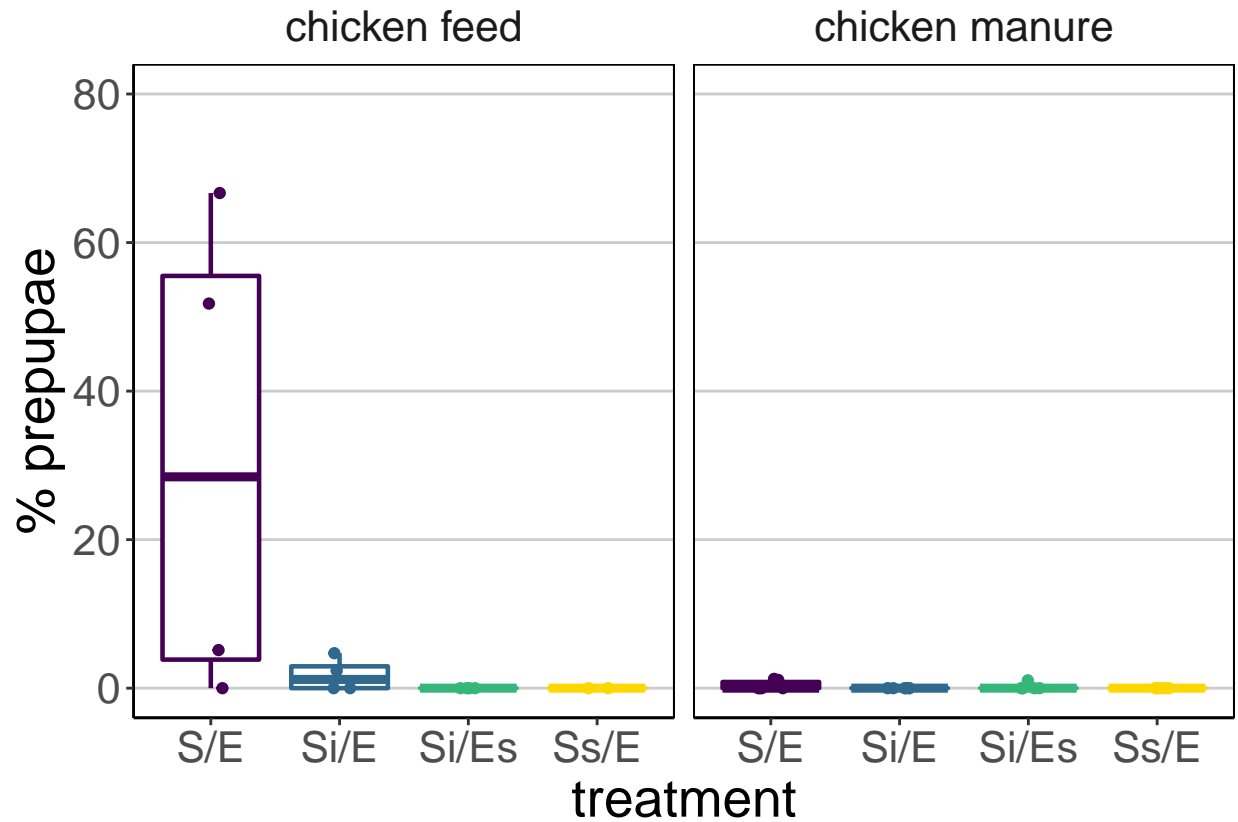
```
# total larval biomass: Figure 2C
pS3 <- ggplot(data = harv.m[harv.m$variable == "dmLarvTot",],
  aes(x = Treatment, y = value, colour = Treatment)) +
  geom_boxplot(size = .8, outlier.alpha = 0) +
  geom_point(position = position_jitter(width = 0.1, height = 0)) +
  scale_color_manual(values = c("#440154FF", "#31688EFF", "#35B779FF", "gold")) +
  labs(y = "total larval biomass (g DM)", x = "treatment") +
  scale_y_continuous(limits = c(0, 5), n.breaks = 6) +
  facet_grid(Diet ~ ., labeller = labs_perf) +
  theme_perf
pS3
```



```
# moisture content residue: Supplementary Figure S3
pS4 <- ggplot(data = harv.m[harv.m$variable == "pMoist",],
  aes(x = Treatment, y = value, colour = Treatment)) +
  geom_boxplot(size = .8, outlier.alpha = 0) +
  geom_point(position = position_jitter(width = 0.1, height = 0)) +
  scale_color_manual(values = c("#440154FF", "#31688EFF", "#35B779FF", "gold")) +
  labs(y = "substrate moisture\ncontent (% FM)", x = "treatment") +
  scale_y_continuous(limits = c(0, 100), n.breaks = 6) +
  facet_grid(. ~ Diet, labeller = labs_perf) +
  theme_perf
pS4
```



```
# percentage prepupae: Supplementary Figure S1
pS5 <- ggplot(data = harv.m[harv.m$variable == "pPrepupae",],
  aes(x = Treatment, y = value, colour = Treatment)) +
  geom_boxplot(size = .8, outlier.alpha = 0) +
  geom_point(position = position_jitter(width = 0.1, height = 0)) +
  scale_color_manual(values = c("#440154FF", "#31688EFF", "#35B779FF", "gold")) +
  labs(y = "% prepupae", x = "treatment") +
  scale_y_continuous(limits = c(0, 80), n.breaks = 5) +
  facet_grid(~ Diet, labeller = labs_perf) +
  theme_perf
pS5
```



## 6. Export plots

```
ggsave(plot = pS1, "./figures/Fig_2A_survival.png", w = 4, h = 8)
ggsave(plot = pS1, "./figures/Fig_2A_survival.pdf", w = 120, h = 200, u = "mm")

ggsave(plot = pS2, "./figures/Fig_2B_indWeight.png", w = 4, h = 8)
ggsave(plot = pS2, "./figures/Fig_2B_indWeight.pdf", w = 120, h = 200, u = "mm")

ggsave(plot = pS3, "./figures/Fig_2C_totWeight.png", w = 4, h = 8)
ggsave(plot = pS3, "./figures/Fig_2C_totWeight.pdf", w = 120, h = 200, u = "mm")

ggsave(plot = pS3, "./figures/Fig_S3_moisture.png", w = 8, h = 5)
ggsave(plot = pS3, "./figures/Fig_S3_moisture.pdf", w = 200, h = 120, u = "mm")

ggsave(plot = pS3, "./figures/Fig_S1_prepupae.png", w = 8, h = 5)
ggsave(plot = pS3, "./figures/Fig_S1_prepupae.pdf", w = 200, h = 120, u = "mm")
```

## Substrate pH

## 1. Prepare data

```
# subset pH data
pH <- perf[,c("Container", "Diet", "Treatment", "Rep", "pH0", "pH15", "Day")]
pH$Container <- as.factor(pH$Container)
pH.m <- reshape2::melt(pH)

## Using Container, Diet, Treatment, Rep, Day as id variables

colnames(pH.m)[7] <- "pH"
colnames(pH.m)[6] <- "Timepoint"
pH.m$Timepoint <- gsub(pattern = "pH", x = pH.m$Timepoint, replacement="",)
pH.m$Timepoint <- as.factor(pH.m$Timepoint)

# subset per diet
pH.cf <- pH.m[pH.m$Diet == "CF",]
pH.cm <- pH.m[pH.m$Diet == "CM",]

# subset per timepoint
pH.cf0 <- pH.cf[pH.cf$Timepoint == 0,]
pH.cf15 <- pH.cf[pH.cf$Timepoint == 15,]
pH.cm0 <- pH.cm[pH.cm$Timepoint == 0,]
pH.cm15 <- pH.cm[pH.cm$Timepoint == 15,]

# subset CF Ss/E all timepoints
pH.cf.ds <- pH.m[pH.m$Diet == "CF" & pH.m$Treatment == "Ss/E",]
```

## 2. GLMM regression

Separate statistics per Diet. Repeated measures for Container (2 timepoints); in that case, no random term for Day needed, because Containers are distributed over different days. GLMM Gamma was used because LMM residuals did not meet the model assumptions.

### 2.1. Chicken feed

```
ph.gm0 <- glmer(pH ~ Treatment * Timepoint + (1|Container),
               data = pH.cf, family = Gamma, nAGQ = 25)

# model output
car::Anova(ph.gm0)

## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: pH
##               Chisq Df Pr(>Chisq)
## Treatment      60.202  3  5.321e-13 ***
## Timepoint      609.555  1  < 2.2e-16 ***
## Treatment:Timepoint 75.474  3  2.867e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```



```
CLD(emmeans(ref_grid(ph.gm0, transform = "response"), ~ Treatment + Timepoint),
     Letters = letters, method = "tukey")
```

```
## Treatment Timepoint response SE df asymp.LCL asymp.UCL .group
## Ss/E 0 5.54 0.140 Inf 5.27 5.82 a
## Ss/E 15 5.61 0.142 Inf 5.33 5.89 a
## Si/Es 0 5.62 0.101 Inf 5.42 5.82 a
## S/E 0 5.65 0.102 Inf 5.45 5.85 a
## Si/E 0 5.66 0.102 Inf 5.46 5.86 a
## Si/E 15 7.52 0.148 Inf 7.23 7.81 b
## S/E 15 7.91 0.159 Inf 7.60 8.22 bc
## Si/Es 15 8.24 0.169 Inf 7.91 8.57 c
##
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 8 estimates
## significance level used: alpha = 0.05
```

## 2.2. Chicken manure

```
ph.gm1 <- glmer(pH ~ Treatment * Timepoint + (1|Container), data = pH.cm,
                family = Gamma, nAGQ = 25)
```

```
# model output
car::Anova(ph.gm1)
```

```
## Analysis of Deviance Table (Type II Wald chisquare tests)
##
## Response: pH
##              Chisq Df Pr(>Chisq)
## Treatment      9.6487 3  0.021801 *
## Timepoint     73.3169 1 < 2.2e-16 ***
## Treatment:Timepoint 13.3248 3  0.003984 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
CLD(emmeans(ref_grid(ph.gm1, transform = "response"), ~ Treatment + Timepoint),
     Letters = letters, method = "tukey")
```

```
## Treatment Timepoint response SE df asymp.LCL asymp.UCL .group
## Ss/E 0 7.52 0.170 Inf 7.18 7.85 a
## Si/E 0 7.95 0.181 Inf 7.59 8.30 ab
## Si/Es 0 8.05 0.184 Inf 7.69 8.41 ab
## S/E 0 8.71 0.202 Inf 8.31 9.11 bc
## Ss/E 15 8.97 0.209 Inf 8.56 9.38 c
## S/E 15 9.10 0.213 Inf 8.68 9.52 c
## Si/Es 15 9.26 0.218 Inf 8.84 9.69 c
## Si/E 15 9.26 0.218 Inf 8.84 9.69 c
##
## Confidence level used: 0.95
## P value adjustment: tukey method for comparing a family of 8 estimates
## significance level used: alpha = 0.05
```

## 2.3. Export ANOVA tables

Supplementary Tables 2 and 3.

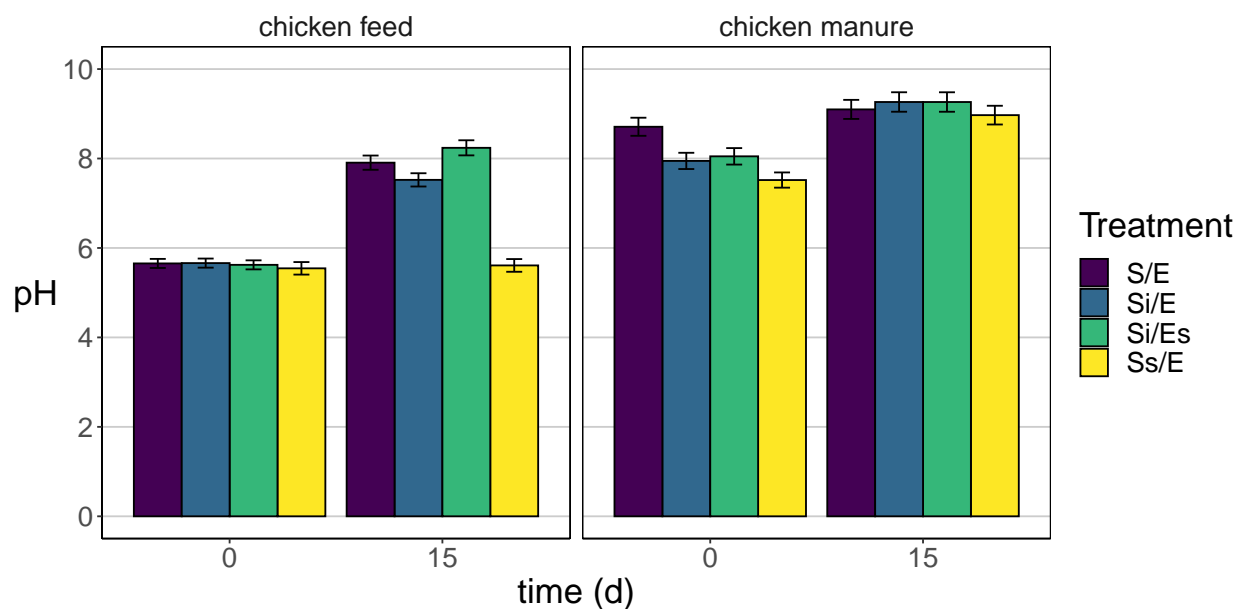
```
write.csv(data.frame(car::Anova(ph.gm0)), "./tables/Supplementary_Table_S2.csv")
write.csv(data.frame(car::Anova(ph.gm1)), "./tables/Supplementary_Table_S3.csv")
```

## 3. Errorbar plot

Supplementary Figure S2.

```
# collect EMM + SE
ph.cf.emm <- CLD(emmeans(ref_grid(ph.gm0, transform = "response"),
                           pairwise ~ Treatment + Timepoint))
ph.cf.emm$Diet <- "CF"
ph.cm.emm <- CLD(emmeans(ref_grid(ph.gm1, transform = "response"),
                           pairwise ~ Treatment + Timepoint))
ph.cm.emm$Diet <- "CM"
ph.emm <- rbind(ph.cf.emm, ph.cm.emm)

# plot mean +/- SE
ppH <- ggplot(ph.emm, aes(x = Timepoint, y = response,
                          group = interaction(Treatment, Timepoint)))
ppH <- ppH + geom_col(aes(fill = Treatment), colour = "black",
                      position = position_dodge()) +
  scale_y_continuous(limits = c(0, 10), n.breaks = 6) +
  scale_fill_viridis(name = "Treatment", discrete = T, option = "D") +
  geom_errorbar(aes(ymin = response - SE, ymax = response + SE),
                width = .3, position = position_dodge(width = 0.9)) +
  labs(y = "pH", x = "time (d)") +
  facet_wrap(~Diet, nrow = 1, scales = "fixed", labeller = labs_perf) + theme_ph
ppH
```



```
# export plot  
ggsave(plot = ppH, "./figures/Fig_S2_pH.png", w = 10, h = 5)  
ggsave(plot = ppH, "./figures/Fig_S2_pH.pdf", w = 300, h = 150, u = "mm")
```