Analyzing and visualizing data from manipulative experiments

Shannon Carter

February 25, 2019

Last summer, I performed an experiment where I created 60 mini experimental ponds in black tanks (pictured below), filled the tanks with tadpoles, and monitored the tanks daily to collect freshly emerged frogs (also picutred). The treatments imposed on the tanks were designed to mimic some biological changes we're expected to see with climate change.

The frogs I collected represented the data for this project. The number that came from each tank and each individual frog's weight and date of emergence were the main response variables for this experiment. Without getting too much into the details of the biology or motivation behind this project, here is an overview of the data processing, visuals, and analysis.

Each black tank is a mini experimental pond filled with tadpoles

Newly emerged gray tree frog

View raw data

First, a look at my raw data. Each row represents one frog. Tank refers to which tank number the frog came from and "block", "trt", "order" and "sync" are four treatment identifiers for each tank. "timetoemerge" is simply the number of days between when the tadpoles were added to the tank and when that particular frog emerged. This snippet shows the first and last 5 rows of the dataset. I checked on these tanks every day for 152 days and collected and weighed 1400 frogs. It was a lot of work!

```r
ind_results[c(1:5, 1383:1388),]
```

```
##         no       date tank block       trt order sync mass_mg timetoemerge
## 1        1 2019-05-07    3     1 early_low early  low   148.9           22
## 2        2 2019-05-07    3     1 early_low early  low   143.1           22
## 3        3 2019-05-07    3     1 early_low early  low      NA           22
## 4        4 2019-05-07    5     1  same_med  same  med   155.7           22
## 5       5 2019-05-07   12     2  same_low  same  low   142.2           22
## 1383 1399 2019-09-10   36     4 late_high  late high   218.3          148
## 1384 1400 2019-09-11   34     4 same_high  same high   245.0          149
## 1385 1401 2019-09-11   54     6 same_high  same high   179.4          149
## 1386 1402 2019-09-12    8     1  late_med  late  med   192.1          150
## 1387 1403 2019-09-12   49     5  late_med  late  med   198.2          150
## 1388 1404 2019-09-14   28     3 late_high  late high   197.1          152
```

Data processing

First thing to do is summarize results by tank. Here, I count the number of frogs that comes from each tank and calculate five response variables: proportion survival, cumulative biomass, mean per capita mass, mean per capita time to emergence, and variation in time to emergence.

```
## Add a column of ones to sum as a counting mechanism
ind_results$ones <- 1

## Biomass is the sum weight of all frogs that come out of a particular tank
## This function replaces NA masses (cases where the frog died, RIP) with global mean mass
biomassxNA <- function(x) {
  x[which(is.na(x))] = mean(ind_results$mass_mg, na.rm = T)
  return(cumsum(x))
}

## For each tank, add a cumulative frog number and biomass
meta_accum <- ind_results %>%
  group_by(tank) %>%
  mutate(cumulative_metas = cumsum(ones),
         cumulative_biom  = biomassxNA(mass_mg)) # using the function written above

## This data frame still has one row per frog, but now has a running cumulative number and mass
meta_accum <- dplyr::select(meta_accum, date, tank, block, trt, order, sync, ones,
                    mass_mg, timetoemerge, cumulative_metas, cumulative_biom)

## Now, I summarize to one row per tank
tank_results <- meta_accum %>%
  group_by(tank) %>%
  summarize(surv = max(cumulative_metas)/45,  # proportion of frogs that made it to emergence
            biom = max(cumulative_biom),      # final cumulative mass of all frogs
            mass = mean(mass_mg, na.rm = T),  # mean individual mass of frogs
            emer = mean(timetoemerge),        # mean time to emergence
            emsd = sd(timetoemerge))          # variation in time to emergence

## Add tank treatment info back in
tank_results <- inner_join(tank_results, treatments, by = 'tank')

## Select relevant columns and make 1 row per tank
tank_results <- as.data.frame(unique(dplyr::select(tank_results, tank, block, trt, order,
                                                   sync, surv, biom, mass, emer, emsd)))
```

The resulting data frame (snippet below) has one row per tank. “block”, “trt”, “order”, and “sync” give information about the treatment assigned to the tank. “surv”, “biom”, “mass”, “emer” and “emsd” are response variables I measured/calculated.

```r
head(tank_results)
```

```
##   tank block        trt order sync      surv     biom     mass      emer
## 1    1     1 early_high early high 0.6666667 5906.654 197.4333  85.66667
## 2    2     1  early_med early  med 0.8222222 6480.323 172.5125  67.08108
## 3    3     1  early_low early  low 0.6444444 5636.585 194.4500  44.62069
## 4    4     1  same_high  same high 0.2444444 2224.869 204.5444 120.27273
## 5    5     1   same_med  same  med 0.6888889 5685.685 183.1233  83.83871
## 6    6     1   same_low  same  low 0.4222222 3223.700 169.6684  68.47368
##       emsd
## 1 26.75861
## 2 29.72688
## 3 15.14788
## 4 22.79952
## 5 32.89285
## 6 26.16055
```

I then group the tanks by treatment (‘trt’) and calculate means and variance measures (standard deviation, standard error, and 95% confidence intervals) for each of the 5 response variables. This part isn’t shown, but a snippet of the resulting dataframe is below.

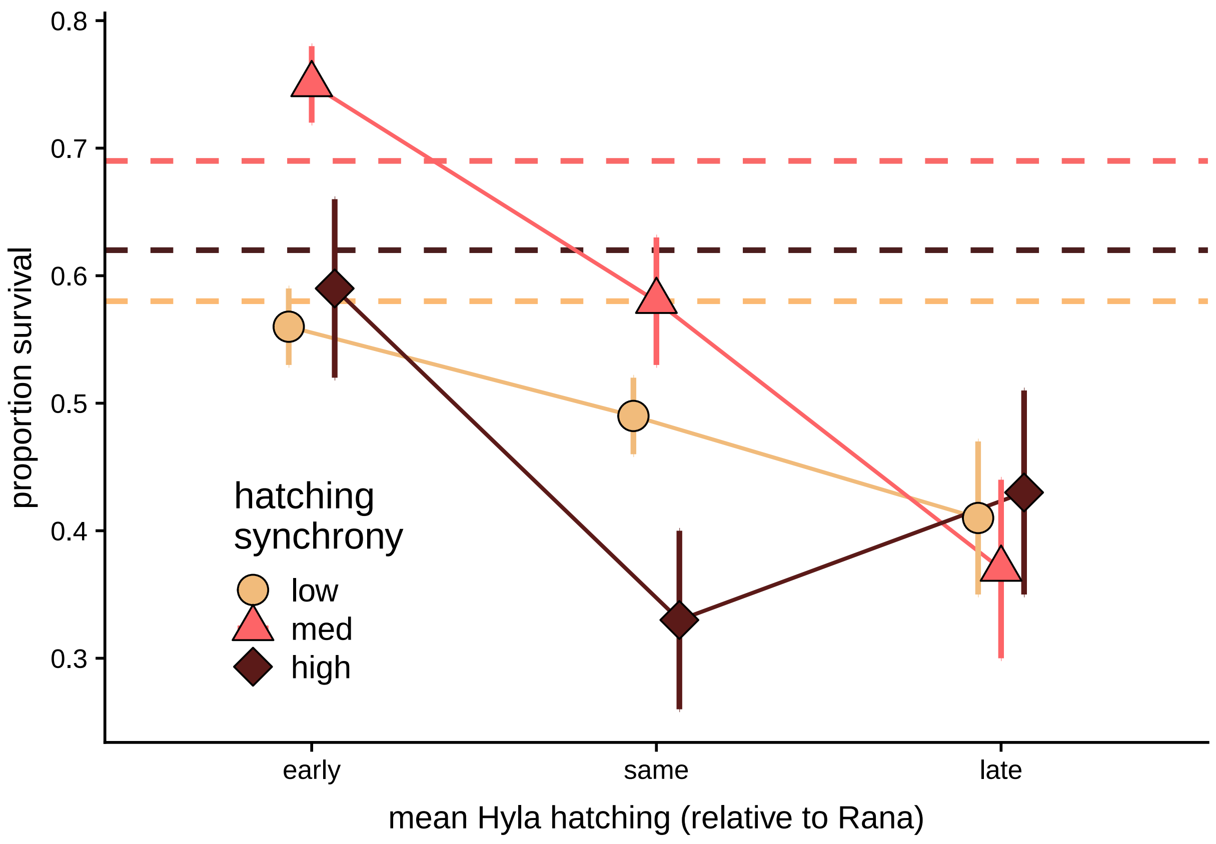
```r
head(trt_means1)
```

```
##          trt order sync variable   mean     sd     se      ci
## 1  cont_high  cont high     biom 5035.4 1678.3 1186.7 15078.9
## 2   cont_low  cont  low     biom 6127.3  775.1  548.1  6964.2
## 3   cont_med  cont  med     biom 6102.8  497.2  351.6  4467.5
## 4 early_high early high     biom 5025.5 1594.4  650.9  1673.2
## 5  early_low early  low     biom 4961.3  549.2  224.2   576.3
## 6  early_med early  med     biom 6308.2  600.7  245.2   630.4
```

Figures

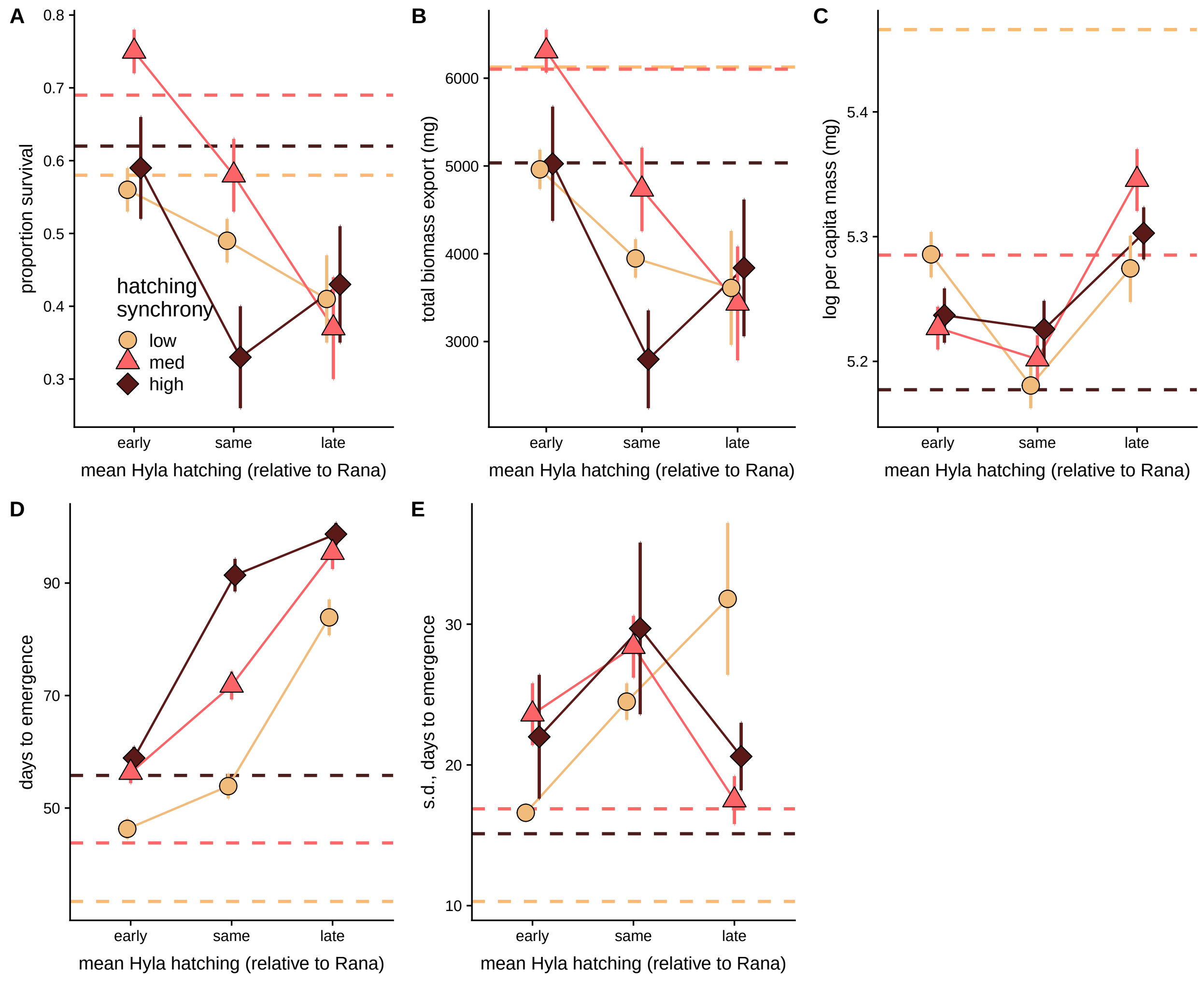
Now we’re ready to take a look at the results! For each of the five response variables, I make a figure that looks like this. This one shows proportion survival. Here, the x-axis and colors represent the two aspects of my treatments (‘order’ and ‘sync’). The dashed lines represent control values which correspond to each level of ‘sync’.

```r
surv_means <- ggplot(subset(trt_means, subset = (variable == 'surv' & order != 'cont')),
                     # synchrony as color, fill, and shape so color of lines match points and it's b/w
                     aes(x = order, y = mean, group = sync, fill = sync, color = sync, shape = sync)) +

  # 3 horizontal lines to show control baselines- colors matching synchrony level
  geom_hline(yintercept = 0.58, size = 1, linetype = 'dashed', color = "#fbb973") +
  geom_hline(yintercept = 0.69, size = 1, linetype = 'dashed', color = "#f96968") +
  geom_hline(yintercept = 0.62, size = 1, linetype = 'dashed', color = "#4b1d1d") +

  # lines connecting points
  geom_line(size = 0.7, position = position_dodge(width = 0.2)) +

  # error bars on points
  geom_errorbar(size = 1, position = position_dodge(width = 0.2),
              aes(ymin = mean - se, ymax = mean + se, width = 0)) +

  # points- color = 'black' only makes a black outline on points
  geom_point(size = 5, color = 'black', position = position_dodge(width = 0.2)) +

  # axis and legend labels
  labs(shape = "hatching\nsynchrony",
       fill = 'hatching\nsynchrony',
       color = "hatching\nsynchrony",
       x = "mean Hyla hatching (relative to Rana)",
       y = "proportion survival") +

  # style elements
  theme(legend.position = c(0.1, 0.22),
        axis.title = element_text(size = 12),
        axis.text  = element_text(size = 10)) +

  # adding fun Wes Anderson color scheme
  scale_fill_manual(values = wes_palette(n = 3, name = "GrandBudapest1")) +
  scale_color_manual(values = wes_palette(n = 3, name = "GrandBudapest1")) +
  scale_shape_manual(values = c(21, 24, 23))
surv_means
```

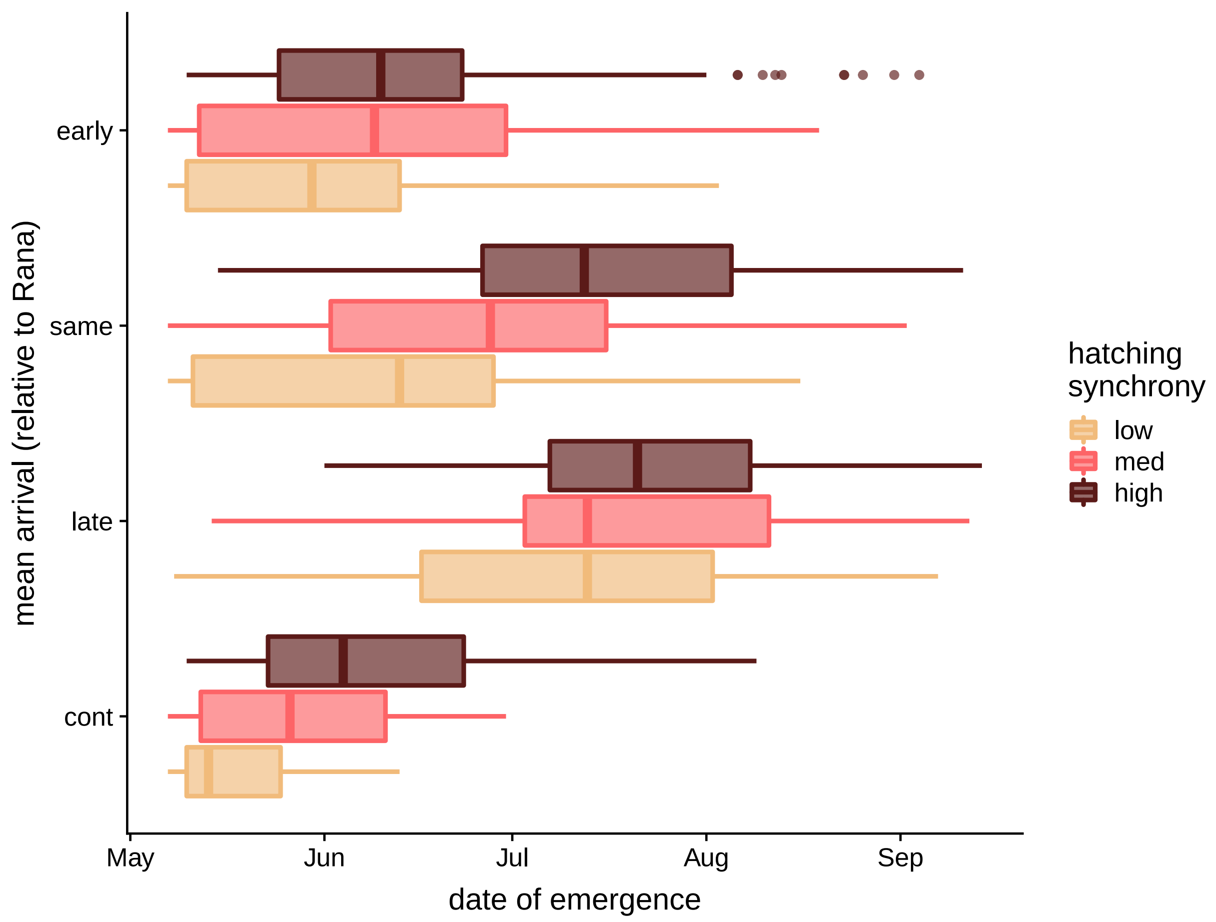


All response variables together

Here, I've reproduced the above figure for each of the 5 response variables and stitched them together. These results are very exciting! But you'll just have to take my word for it, because I'm not not going to get into the nitty gritty biology of it here.



I really like this figure because it reminds me of ice cream. Here, we see the mean and variation in how long individual frogs took to emerge broken down by treatment.



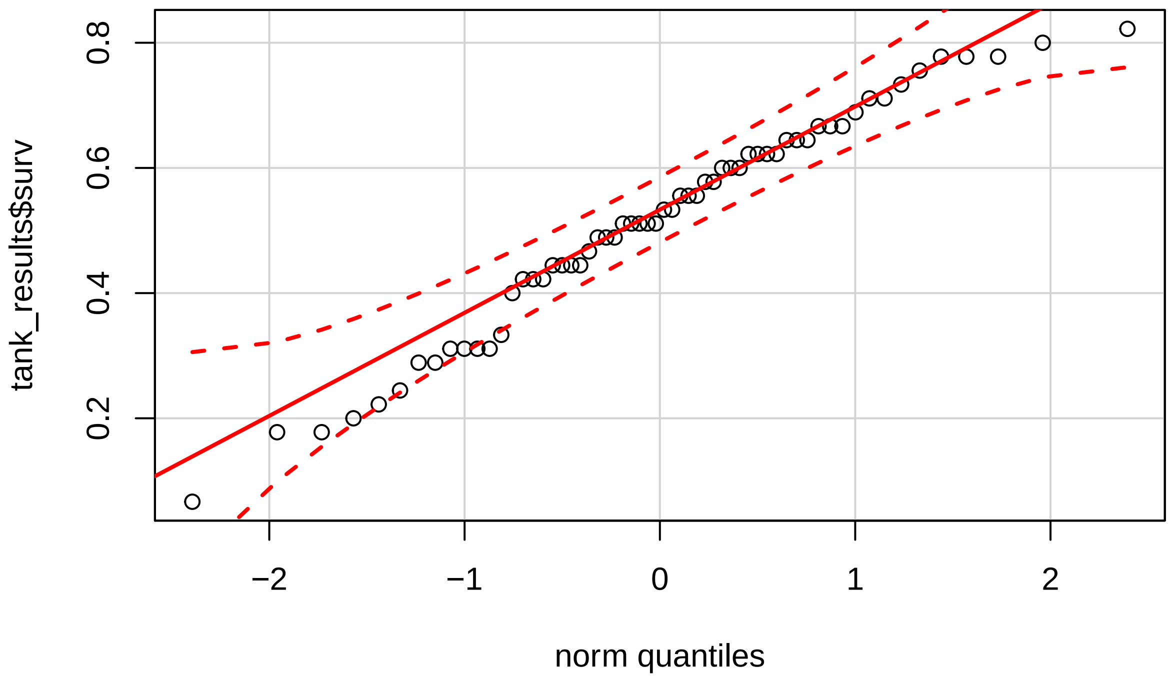
Analysis

I spent at least 40 hours on the statistical analysis for this project. Testing different error structures, validating model assumptions, comparing AICs, adding and dropping interactive terms and random effects, etc. Here, I'll give an overview of what that process looked like for one of the five response variables.

Checking model assumptions

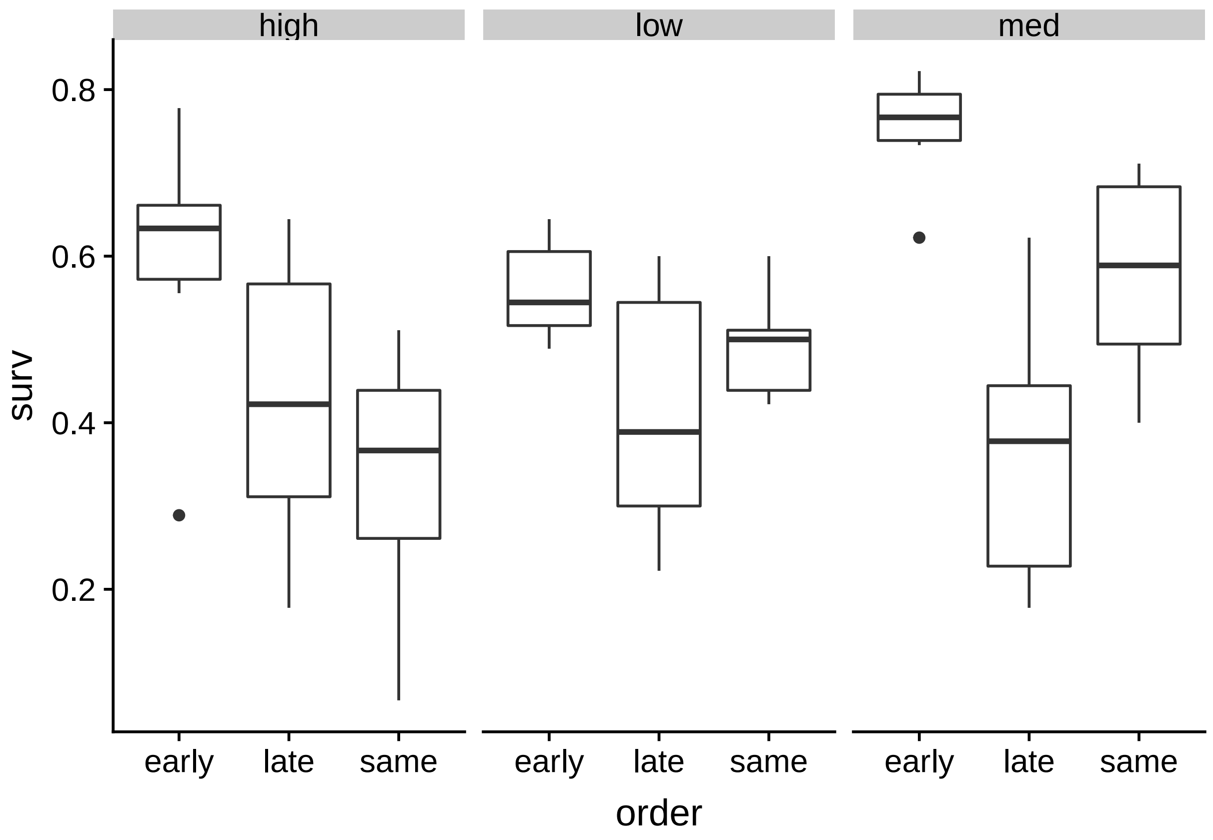
Is data normally distributed? Yes.

```r
qqp(tank_results$surv, 'norm')
```



Are variances equal across treatments? Yes.

```
ggplot(tank_results2, aes(x = order, y = surv)) +
  geom_boxplot() +
  facet_grid(~ sync)
```



Coarse models

Here, I change big things, like adding/dropping explanatory (treatment) variables, random effects for replicates, and interaction terms

```
# only one explanatory variable (order). bad fit
m1 <- lmer(data = tank_results2, surv ~ order + (1|block), REML = F)

# only one explanatory variable (sync). bad fit
m2 <- lmer(data = tank_results2, surv ~ sync + (1|block), REML = F)

# order and sync but no interaction, block as random effect
m3 <- lmer(data = tank_results2, surv ~ sync + order + (1|block), REML = F)

# order and sync but no interaction, block as fixed effect
m4 <- lm(data = tank_results2, surv ~ sync + order + block)

# interaction between order and sync, block as random effect
m5 <- lmer(data = tank_results2, surv ~ sync * order + (1|block), REML = F)

# interaction between order and sync, block as fixed effect
m6 <- lm(data = tank_results2, surv ~ sync * order + block)
```

Model selection

I use AIC values to compare the different models. m5 has the lowest AIC, so is the best fitting model of these six.

```r
# comparing m1-6
anova(m1, m2, m3, m4, m5, m6)
```

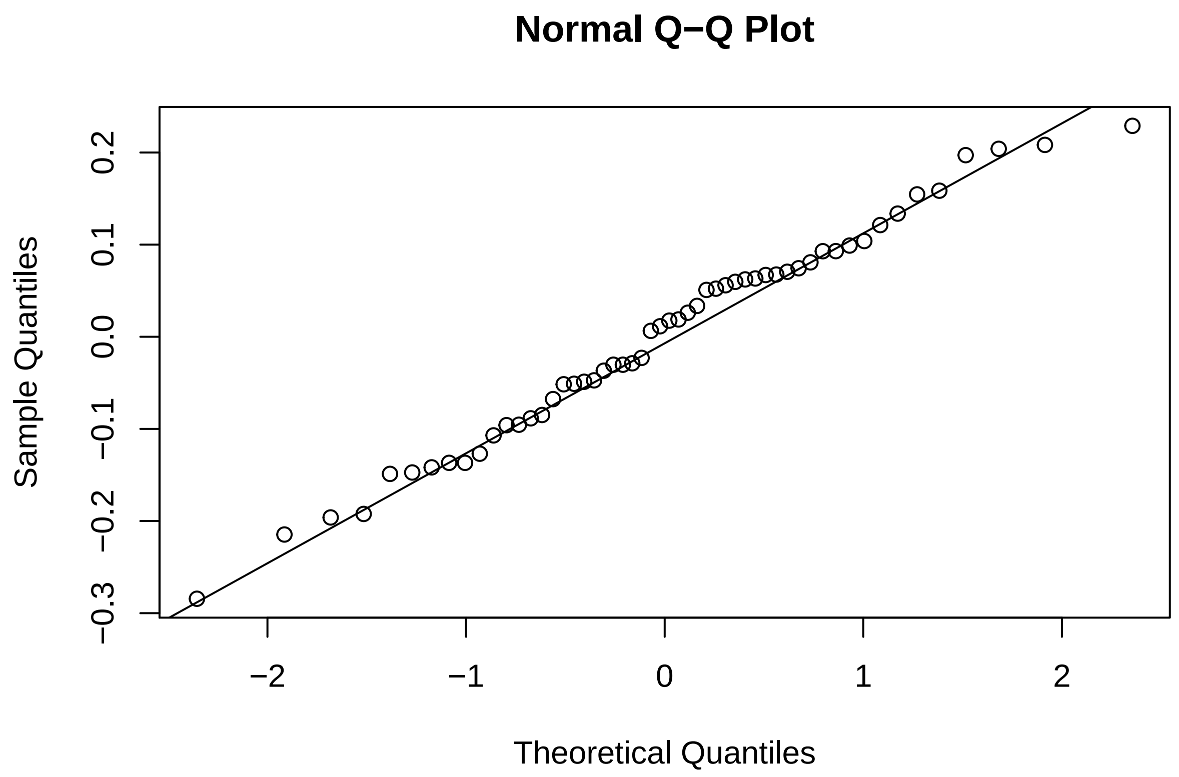
```
## Data: tank_results2
## Models:
## m1: surv ~ order + (1 | block)
## m2: surv ~ sync + (1 | block)
## m3: surv ~ sync + order + (1 | block)
## m4: surv ~ sync + order + block
## m5: surv ~ sync * order + (1 | block)
## m6: surv ~ sync * order + block
##    Df     AIC     BIC logLik deviance   Chisq Chi Df Pr(>Chisq)
## m1  5 -43.666 -33.721 26.833  -53.666
## m2  5 -27.738 -17.793 18.869  -37.738  0.0000      0    1.00000
## m3  7 -45.948 -32.025 29.974  -59.948 22.2100      2  1.504e-05 ***
## m4  7 -46.093 -32.170 30.047  -60.093  0.1452      0  < 2.2e-16 ***
## m5 11 -50.236 -28.357 36.118  -72.236 12.1427      4    0.01632 *
## m6 11 -49.617 -27.739 35.809  -71.617  0.0000      0    1.00000
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

At this point, I made several new versions of the chosen m5 with finetune changes and compared the model fits again. This includes things like trying different data structures, coding explanatory variables as numeric vs. factors vs. ordered factors, etc. But we'll skip that part for now and move on to model diagnostics.

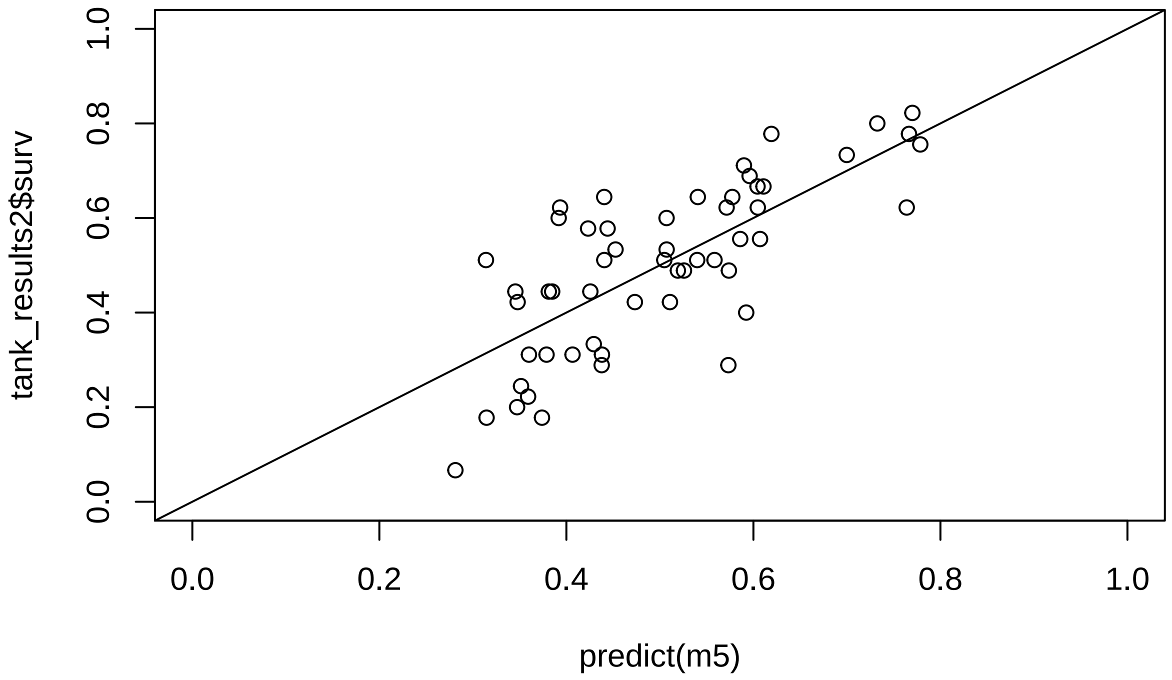
Model diagnostics

Lastly, I take a closer look at the chosen model. I normally check out these and other diagnostics for all candidate models, but here are just the highlights.

```r
# the tightness of the points to the line here validates that our data was normally distributed
res_m5 <- residuals(m5)
qqnorm(res_m5)
qqline(res_m5)
```



```r
# this plots our model predicted values against actual values-- looks pretty good!
plot(predict(m5), tank_results2$surv, xlim = c(0, 1), ylim = c(0, 1))
abline(a = 0, b = 1)
```


tank_results2$surv
predict(m5)
0.0
0.2
0.4
0.6
0.8
1.0