Algal Growth Survival Analysis & Event Modeling: A Detailed Handout

Marc Los Huertos

November 19, 2025

Contents

1 Goal and Justification

The purpose of this handout is to compare the time it takes for algal cultures under three different treatments to reach a specific growth milestone (an absorbance threshold). Survival analysis is used because it correctly handles cultures that never reach the threshold (right-censoring) and provides robust, interpretable comparisons of growth rates over time.

1.1 Why Survival Analysis is an important way to go!

In simple terms, survival analysis focuses on speed. Instead of just looking at OD at a single time point, we look at the time-to-event.

- The Event: Our chosen growth milestone (e.g., OD ≥ 0.6).
- The Time: The hours it took to hit that milestone.
- The Magic (Censoring): If a culture is too slow and never reaches the threshold by the end of the 72-hour experiment, we don't throw it out. We simply record that the event did not occur (status = 0) by the time of the last observation. This avoids bias, giving us an honest assessment of growth rates.

Survival Term	Algae Equivalent	Statistical Role
Event	Reaching OD threshold (e.g., OD = 0.6)	Defines the failure point we are tracking
Time	Hours until OD $\geq$ threshold	The variable we are modeling
Censoring	Not reaching threshold by experiment end	Allows us to use incomplete data
Hazard Ratio	Relative rate of reaching threshold	HR $>$ 1 means faster growth (speed)

2 Packages and Data Setup

We first load the required packages. These provide tools for data manipulation (`tidyverse`), survival analysis (`survival`), visualization (`survminer`), and tidying model output (`broom`).

```
library(tidyverse)

## -- Attaching core tidyverse packages ------------------------
tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts()
--
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>)
to force all conflicts to become errors
```

```
library(survival)
library(survminer)
```

```
## Loading required package:  ggpubr
##
## Attaching package:  'survminer'
##
## The following object is masked from 'package:survival':
##
##     myeloma
```

```
library(broom)
```

2.1 Data Simulation

We simulate growth data for three treatments. The function `make_growth` generates noisy logistic growth curves.

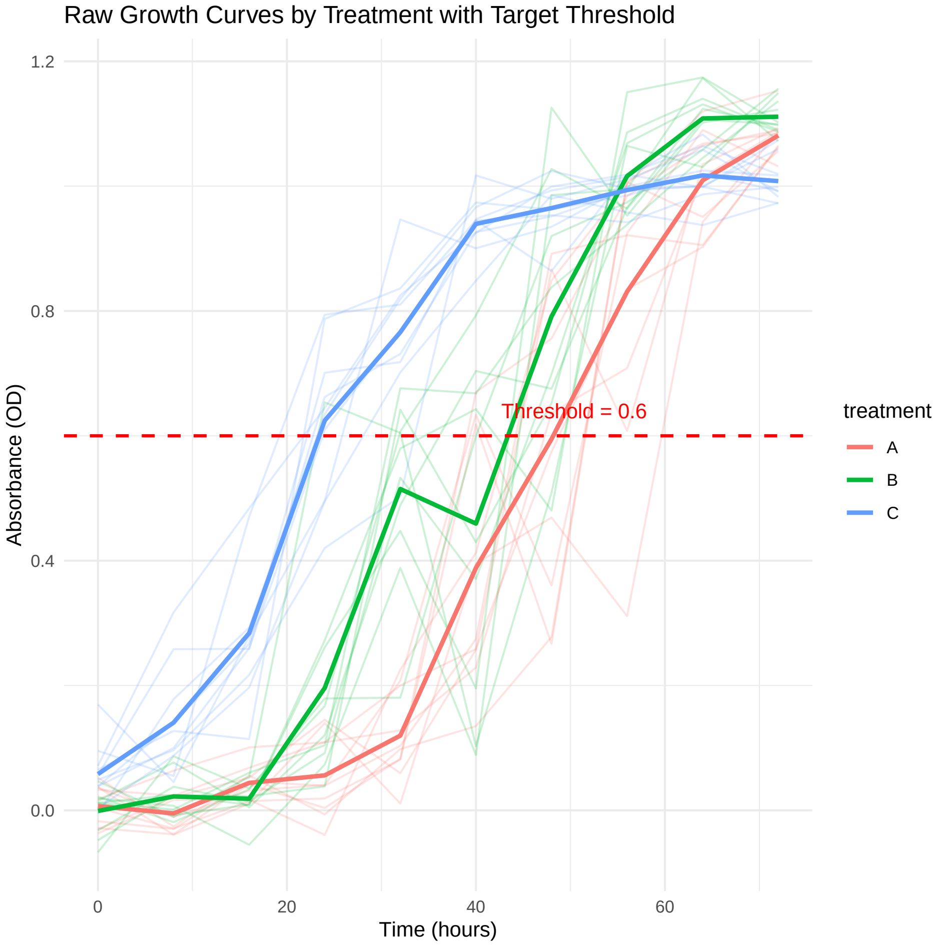
```
set.seed(2025)
n_rep <- 10
times <- seq(0, 72, by = 8)

make_growth <- function(n, treatment_label, mu_time_to_mid = 30, sd_time = 6, maxOD =
    tibble(sample = paste0(treatment_label, "_", seq_len(n))) %>%
    rowwise() %>%
    mutate(
      t_mid = rnorm(1, mu_time_to_mid, sd_time) |> pmax(6),
      slope = (maxOD) / (t_mid + 0.1),
      never = runif(1) < prop_no_reach
    ) %>%
    ungroup() %>%
    expand_grid(time = times) %>%
    rowwise() %>%
    mutate(
      mu = plogis((time - rnorm(1, mu_time_to_mid, sd_time))/7) * maxOD,
      absorbance = mu + rnorm(1, 0, 0.03)
    ) %>%
    ungroup() %>%
    mutate(treatment = treatment_label)
}

# --- ENSURE THESE ADJUSTED VALUES ARE USED ---
# Slower growth for A and B to create failures
df_A <- make_growth(n_rep, "A", mu_time_to_mid = 45, sd_time = 6, maxOD = 1.1, prop_n
df_B <- make_growth(n_rep, "B", mu_time_to_mid = 40, sd_time = 7, maxOD = 1.15, prop_
df_C <- make_growth(n_rep, "C", mu_time_to_mid = 22, sd_time = 5, maxOD = 1.0, prop_n

df_raw <- bind_rows(df_A, df_B, df_C) %>%
    mutate(treatment = factor(treatment))
```

3 Visualization and Event Definition

3.1 Raw Growth Curves and Threshold

We plot individual growth curves and treatment means. This shows variability and average trends. The dashed red line marks our target OD = 0.6 threshold.

```
df_means <- df_raw %>%
    group_by(treatment, time) %>%
    summarise(mean_abs = mean(absorbance), .groups = "drop")

threshold <- 0.6

p_raw <- ggplot() +
    geom_line(data = df_raw,
              aes(time, absorbance, group = sample, color = treatment),
              alpha = 0.2, show.legend = FALSE) +
    geom_line(data = df_means,
              aes(time, mean_abs, color = treatment),
              linewidth = 1.1) +
    geom_hline(yintercept = threshold, linetype = "dashed", color = "red", linewidth
    annotate("text", x = max(df_raw$time)*0.7, y = threshold + 0.04, label = paste0("
    labs(x = "Time (hours)",
         y = "Absorbance (OD)",
         title = "Raw Growth Curves by Treatment with Target Threshold") +
    theme_minimal()
p_raw
```



4 Analysis 1: Threshold Crossing (Speed Analysis)

4.1 Defining the Event (Survival Data Setup)

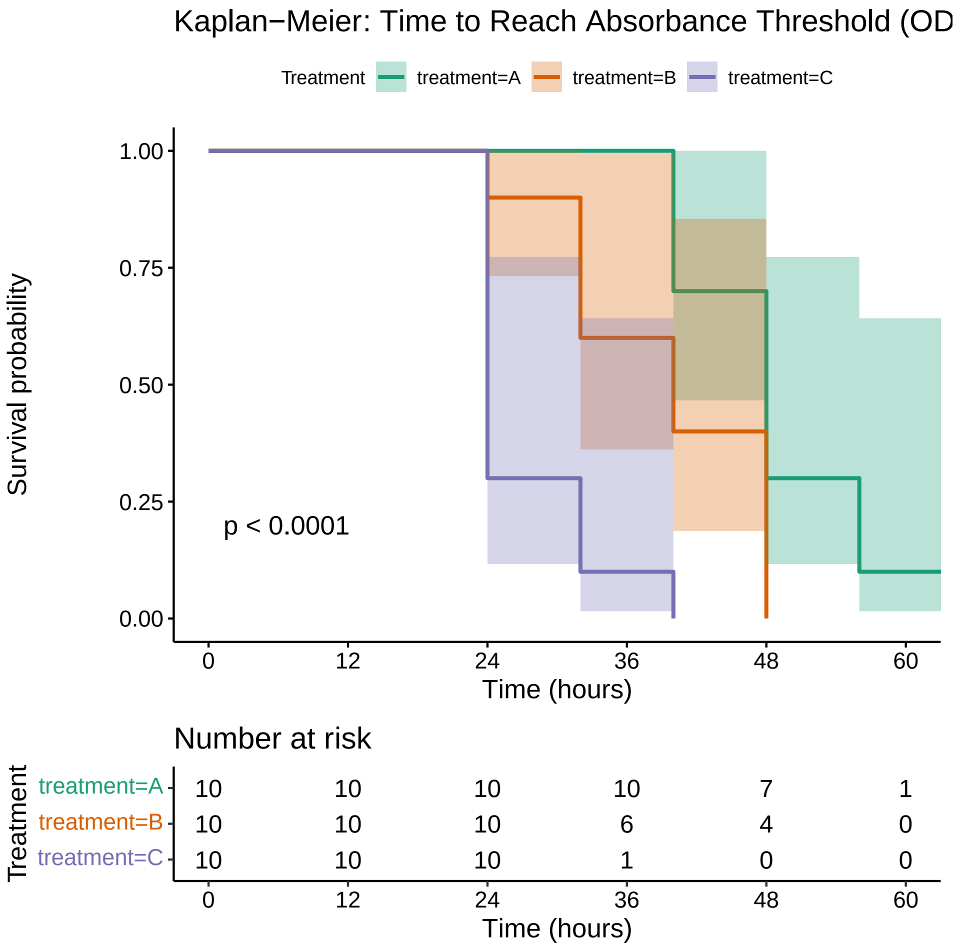
We define the event as the first time a culture reaches OD ≥ 0.6. If a culture never reaches this threshold by the end of the growth experiment (72 hours), it is censored at $t = 72$.

```
time_to_event <- df_raw %>%
    group_by(sample, treatment) %>%
    arrange(time) %>%
    summarise(
      event_time = {
        hit_rows <- which(absorbance >= threshold)
        if(length(hit_rows) == 0) NA_real_ else time[min(hit_rows)]
      },
      last_time = max(time),
      .groups = "drop"
    ) %>%
    mutate(
      status = if_else(is.na(event_time), 0L, 1L), # 1=Event Occurred, 0=Censored
      time = if_else(is.na(event_time), last_time, event_time)
    )

cat("Censoring Status (0 = Censored, 1 = Event Occurred):\n")

## Censoring Status (0 = Censored, 1 = Event Occurred):

print(time_to_event %>% count(treatment, status))

## # A tibble: 3 x 3
##   treatment status     n
##   <fct>      <int> <int>
## 1 A              1    10
## 2 B              1    10
## 3 C              1    10
```

4.2 Kaplan-Meier Curves: Visualizing Speed

Kaplan-Meier curves estimate the probability of not yet reaching the threshold over time. Steeper, earlier drops mean faster growth. We use ‘surv.median.line = “none” to avoid the interpolation error encountered earlier.

```
km_fit <- survfit(Surv(time, status) ~ treatment, data = time_to_event)

ggsurvplot(km_fit, data = time_to_event, risk.table = TRUE, pval = TRUE,
           conf.int = TRUE, palette = "Dark2",
           surv.median.line = "none",
           title = "Kaplan-Meier: Time to Reach Absorbance Threshold (OD 0.6)",
           xlab = "Time (hours)", legend.title = "Treatment",
           break.time.by = 12)
```



Treatment	0	12	24	36	48	60
treatment=A	10	10	10	10	7	1
treatment=B	10	10	10	6	4	0
treatment=C	10	10	10	1	0	0

4.3 Cox Proportional Hazards Model: Quantifying Speed

The Cox model estimates Hazard Ratios (HR), which quantify the relative speed of hitting the threshold. We use Treatment A as the reference group.

- HR > 1: Treatment reaches the threshold faster than A.
- HR < 1: Treatment reaches the threshold slower than A.

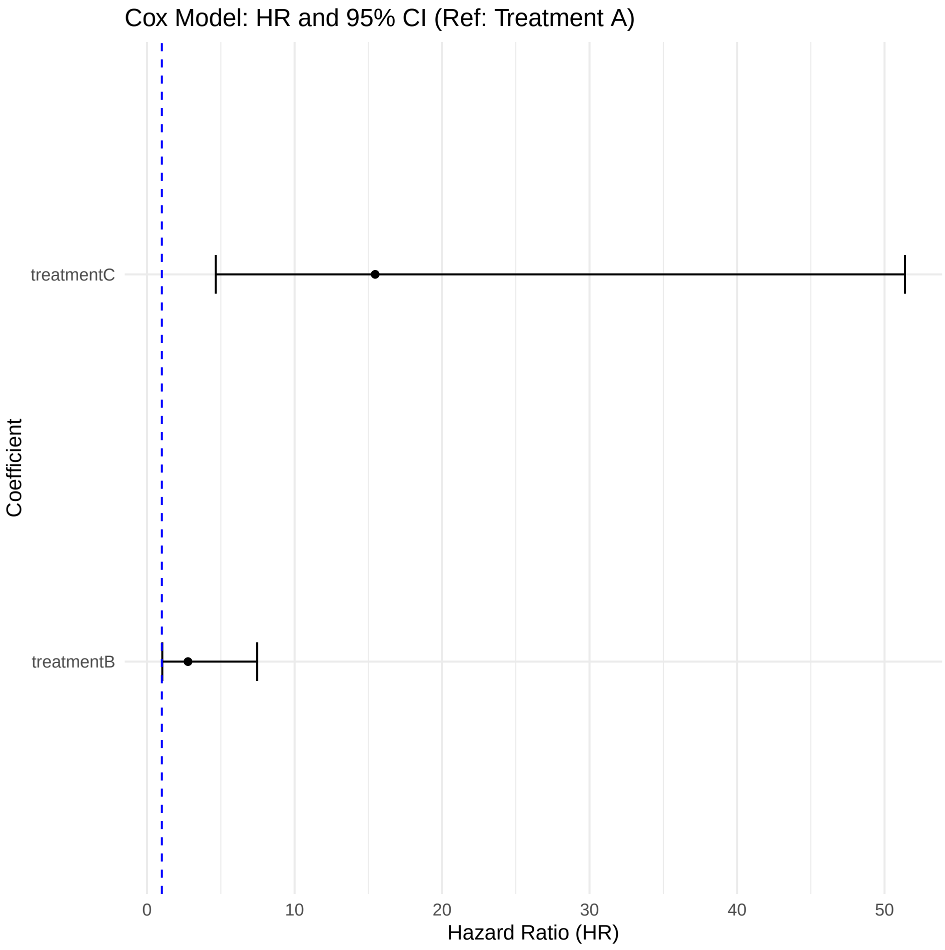
```
time_to_event <- time_to_event %>% mutate(treatment = relevel(treatment, ref = "A"))
cox1 <- coxph(Surv(time, status) ~ treatment, data = time_to_event)
```

```
summary(cox1)

## Call:
## coxph(formula = Surv(time, status) ~ treatment, data = time_to_event)
##
##   n= 30, number of events= 30
##
##                coef exp(coef) se(coef)     z Pr(>|z|)
## treatmentB   1.0199    2.7729   0.5053 2.019   0.0435 *
## treatmentC   2.7386   15.4646   0.6127 4.470 7.83e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##            exp(coef) exp(-coef) lower .95 upper .95
## treatmentB     2.773    0.36063     1.030     7.465
## treatmentC    15.465    0.06466     4.654    51.387
##
## Concordance= 0.817  (se = 0.032 )
## Likelihood ratio test= 21.02  on 2 df,   p=3e-05
## Wald test            = 20.34  on 2 df,   p=4e-05
## Score (logrank) test = 26.43  on 2 df,   p=2e-06
```

4.4 Hazard Ratio Forest Plot

This plot visualizes the HR estimates and confidence intervals (CI). If the CI for an HR does not cross 1 (the blue dashed line), the difference in speed is statistically significant.

```
hr <- tidy(cox1, exponentiate = TRUE, conf.int = TRUE)

ggplot(hr, aes(x = term, y = estimate)) +
    geom_point() +
    geom_errorbar(aes(ymin = conf.low, ymax = conf.high), width = 0.1) +
    geom_hline(yintercept = 1, linetype = "dashed", color = "blue") +
    coord_flip() +
    labs(y = "Hazard Ratio (HR)", x = "Coefficient", title = "Cox Model: HR and 95% C
    theme_minimal()
```



4.5 Assumptions Check: Proportional Hazards

The Cox model assumes the HR is constant over time. A non-significant p-value ($p > 0.05$) from the Schoenfeld residuals test indicates the assumption holds.

```
ph_test <- cox.zph(cox1)
ph_test

##            chisq df    p
## treatment  1.74  2 0.42
## GLOBAL     1.74  2 0.42
```

—

5 Analysis 2: End of Event (Success Probability Analysis)

In this alternative analysis, we ignore when the event happened and only ask: Did the culture succeed in reaching OD ≥ 0.6 by the time the experiment ended (72 hours)?

This shifts our focus from speed to final probability of success. Since the outcome is binary (Success=1 or Failure=0), we use Logistic Regression.

5.1 Event Definition (Logistic Data Setup)

We filter the data to the final time point and create a binary success variable.

```
final_data <- df_raw %>%
    filter(time == max(time)) %>%
    mutate(
        success = if_else(absorbance >= threshold, 1, 0),
        treatment = relevel(treatment, ref = "A")
    )

cat("Success/Failure Status at 72 Hours:\n")

## Success/Failure Status at 72 Hours:

print(final_data %>% count(treatment, success))

## # A tibble: 3 x 3
##   treatment success     n
##   <fct>       <dbl> <int>
## 1 A               1    10
## 2 B               1    10
## 3 C               1    10
```

5.2 Logistic Regression Model

The model estimates Odds Ratios (OR), which quantify the relative odds of a culture achieving OD ≥ 0.6 at the end of the experiment compared to the reference group (Treatment A).

OR > 1: Treatment has higher odds of success than A. OR < 1: Treatment has lower odds of success than A.

```
logis_model <- glm(success ~ treatment, data = final_data, family = "binomial")
summary(logis_model)

##
## Call:
## glm(formula = success ~ treatment, family = "binomial", data = final_data)
##
## Deviance Residuals:
##        Min          1Q      Median          3Q         Max
## 3.971e-06  3.971e-06  3.971e-06  3.971e-06  3.971e-06
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)  2.557e+01  6.831e+04       0        1
## treatmentB  -1.759e-09  9.660e+04       0        1
## treatmentC  -1.759e-09  9.660e+04       0        1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 0.000e+00  on 29  degrees of freedom
## Residual deviance: 4.731e-10  on 27  degrees of freedom
## AIC: 6
##
## Number of Fisher Scoring iterations: 24
```

5.3 Odds Ratio Plot

We visualize the OR estimates and their confidence intervals. If the CI does not cross 1, the difference in the odds of success is statistically significant.

```
or_results <- tidy(logis_model, exponentiate = TRUE, conf.int = TRUE) %>%
    filter(term != "(Intercept)") # Remove the intercept for the plot

## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning:  glm.fit:  fitted probabilities numerically 0 or 1 occurred
## Warning in regularize.values(x, y, ties, missing(ties), na.rm
= na.rm):  collapsing to unique ’x’ values
## Error in approx(sp$y, sp$x, xout = cutoff):  need at least two
non-NA values to interpolate
```

```
ggplot(or_results, aes(x = term, y = estimate)) +
    geom_point() +
    geom_errorbar(aes(ymin = conf.low, ymax = conf.high), width = 0.1) +
    geom_hline(yintercept = 1, linetype = "dashed", color = "blue") +
    coord_flip() +
    labs(y = "Odds Ratio (OR)", x = "Coefficient", title = "Logistic Model: Odds of S
    theme_minimal()
```

```
## Error in ggplot(or_results, aes(x = term, y = estimate)):  object
’or_results’ not found
```