Week 7 Solutions

Isabelle Cretton

2024-11-01

Problem 1: Non-parametric regression

Data Loading and Initial Setup

```
# Read the data
medflies <- read.table("data/medflies.txt", header = TRUE)
head(medflies)
```

```
##   day  living mort.rate
## 1   0 1203646         0
## 2   1 1203646    0.0014
## 3   2 1201913    0.0040
## 4   3 1197098    0.0051
## 5   4 1191020    0.0064
## 6   5 1183419    0.0075
```

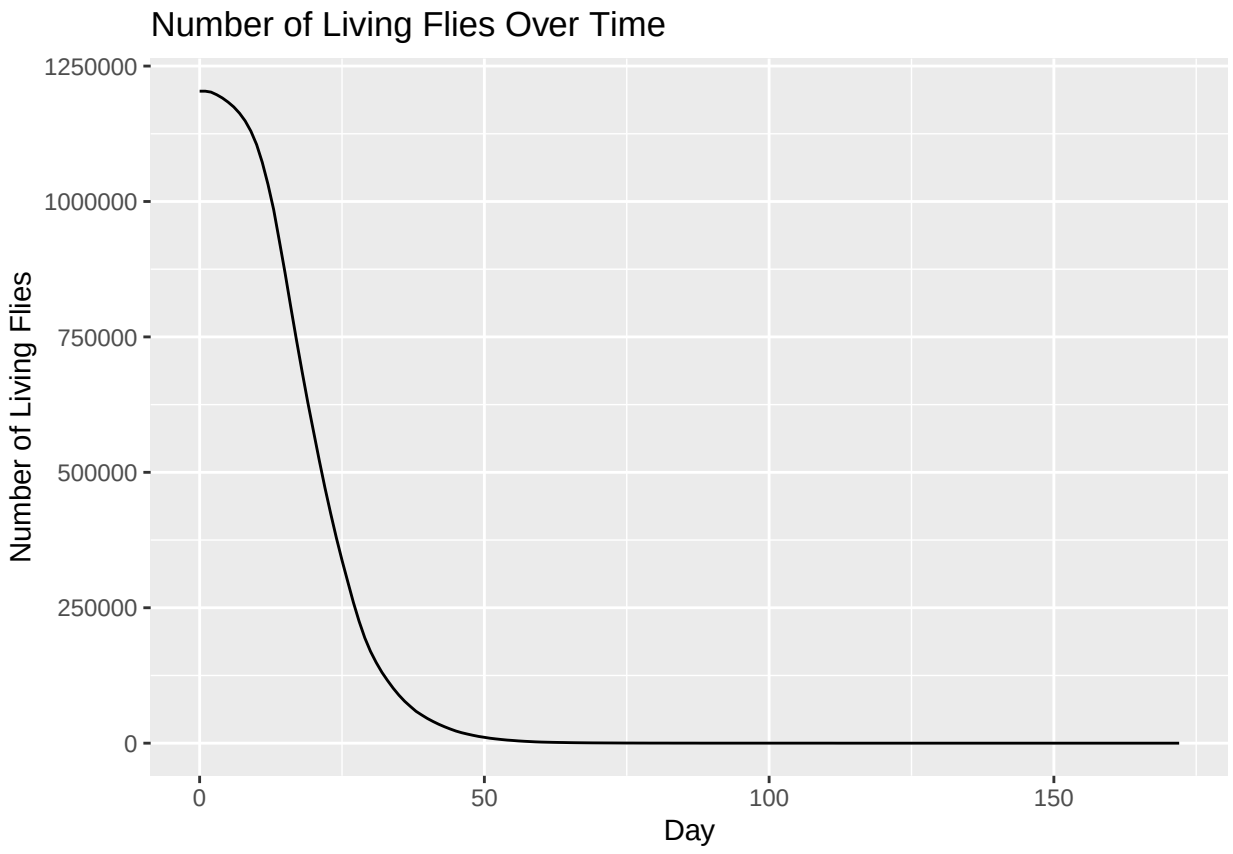
(a) Data Exploration

```
# Summary statistics
summary(medflies)
```

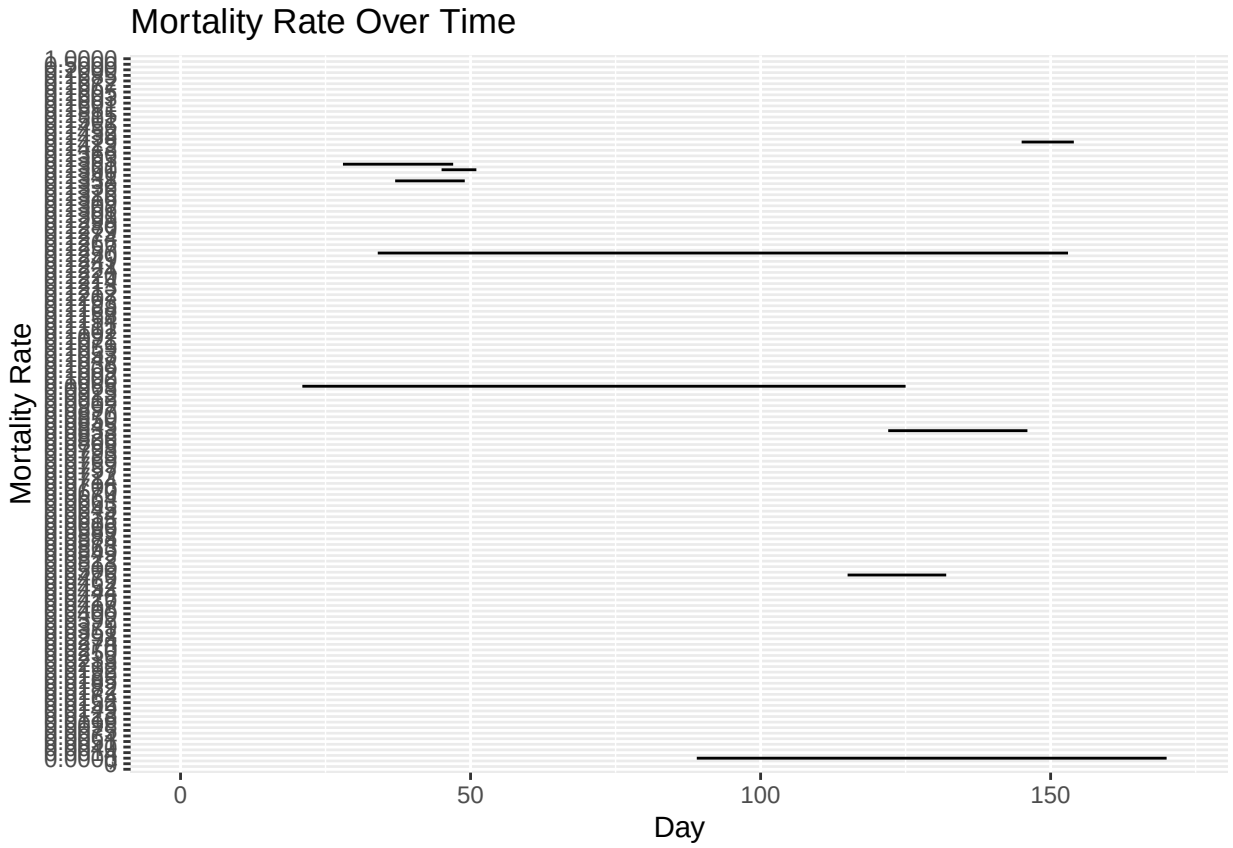
```
##       day          living         mort.rate
##  Min.   :  0   Min.   :      0   Length:173
##  1st Qu.: 43   1st Qu.:     23   Class :character
##  Median : 86   Median :    115   Mode  :character
##  Mean   : 86   Mean   : 148501
##  3rd Qu.:129   3rd Qu.:  30360
##  Max.   :172   Max.   :1203646
```

```
# Create plots to visualize the data
p1 <- ggplot(medflies, aes(x = day, y = living)) +
  geom_line() +
  labs(title = "Number of Living Flies Over Time",
       x = "Day",
       y = "Number of Living Flies")

p2 <- ggplot(medflies, aes(x = day, y = mort.rate)) +
```

```
  geom_line() +
  labs(title = "Mortality Rate Over Time",
       x = "Day",
       y = "Mortality Rate")

print(p1)
```




```
print(p2)
```



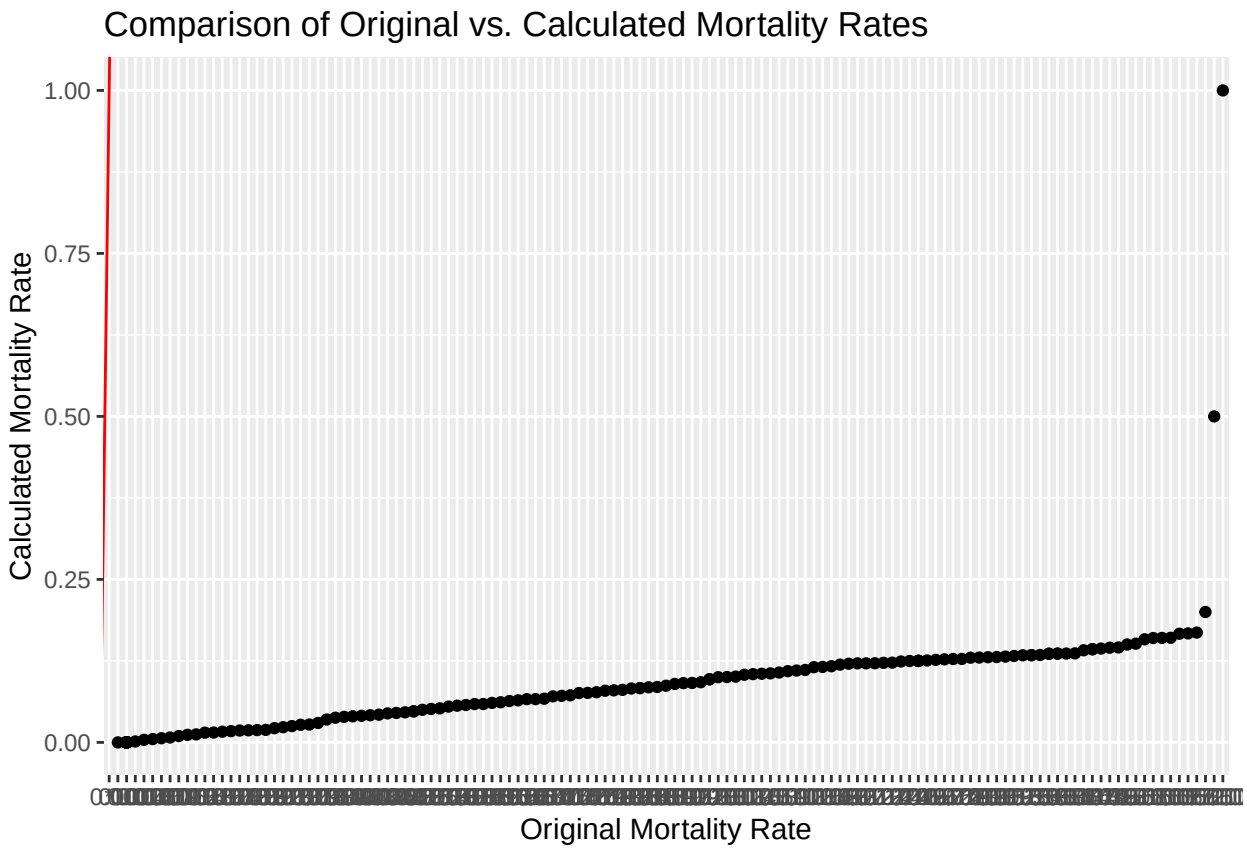
(b) Mortality Rate Verification

```
# Calculate mortality rate from the 'living' variable
# Added error handling for division by zero
medflies$calc_mort_rate <- c(-diff(medflies$living) / pmax(medflies$living[-nrow(medflies)], 1), NA)

# Compare calculated vs. provided mortality rates
head(data.frame(
  day = medflies$day,
  original_rate = medflies$mort.rate,
  calculated_rate = medflies$calc_mort_rate
))
```

```
##   day original_rate calculated_rate
## 1   0             0     0.000000000
## 2   1        0.0014     0.001439792
## 3   2        0.0040     0.004006114
## 4   3        0.0051     0.005077279
## 5   4        0.0064     0.006381925
## 6   5        0.0075     0.007534947
```

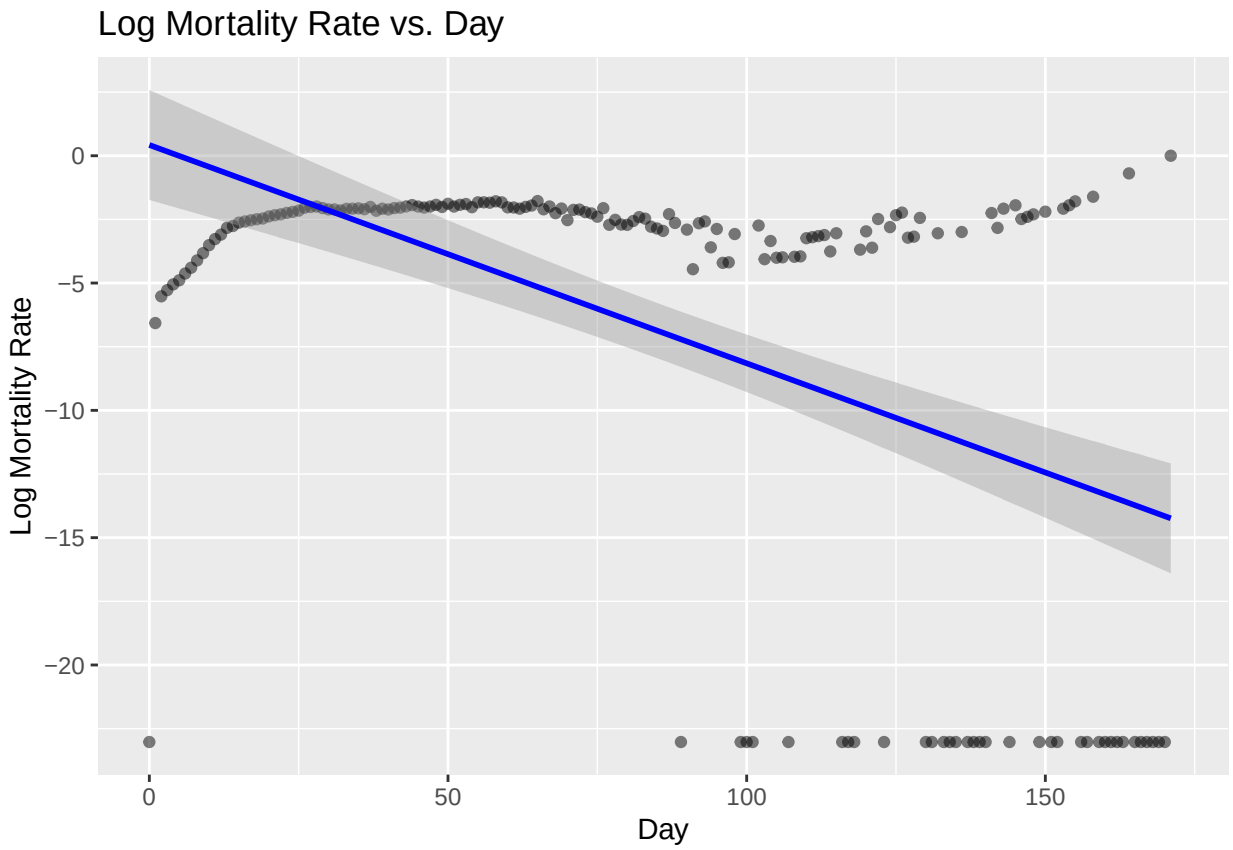
```
# Plot comparison
ggplot(medflies, aes(x = mort.rate, y = calc_mort_rate)) +
  geom_point() +
```

```
  geom_abline(intercept = 0, slope = 1, color = "red") +
  labs(title = "Comparison of Original vs. Calculated Mortality Rates",
       x = "Original Mortality Rate",
       y = "Calculated Mortality Rate")
```

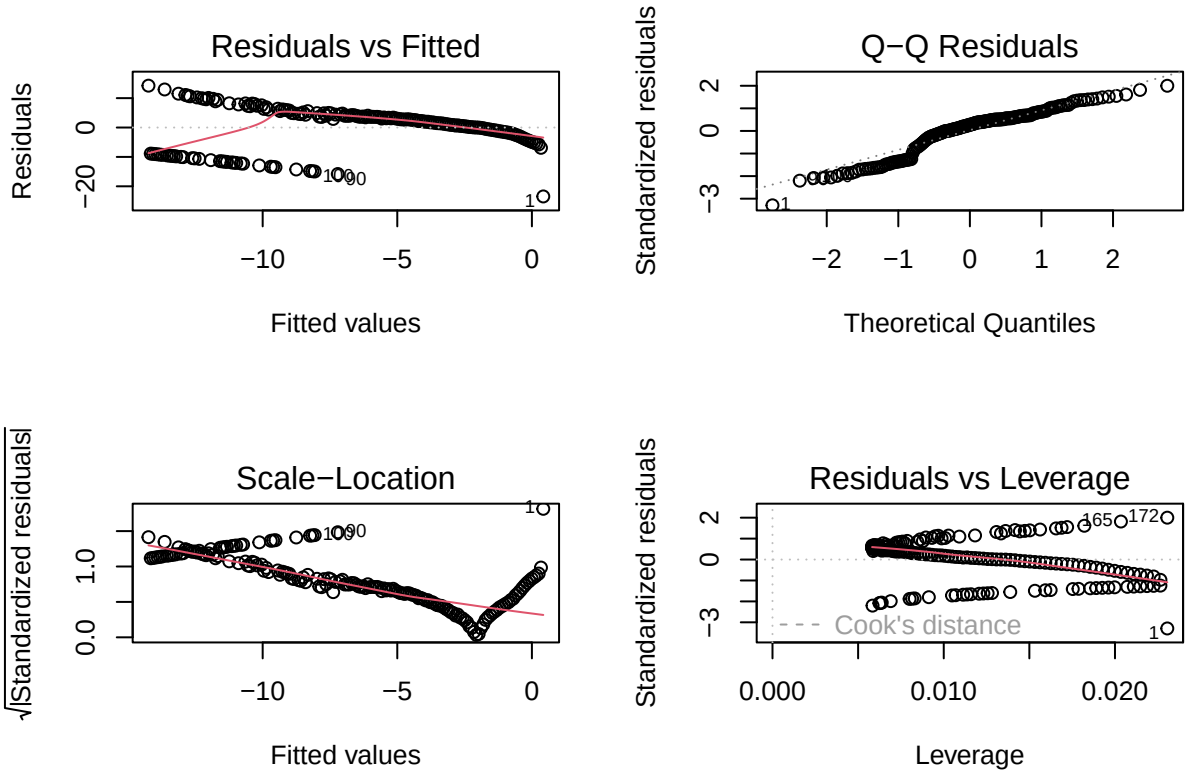


(c) Verification of Gompertz's Theory

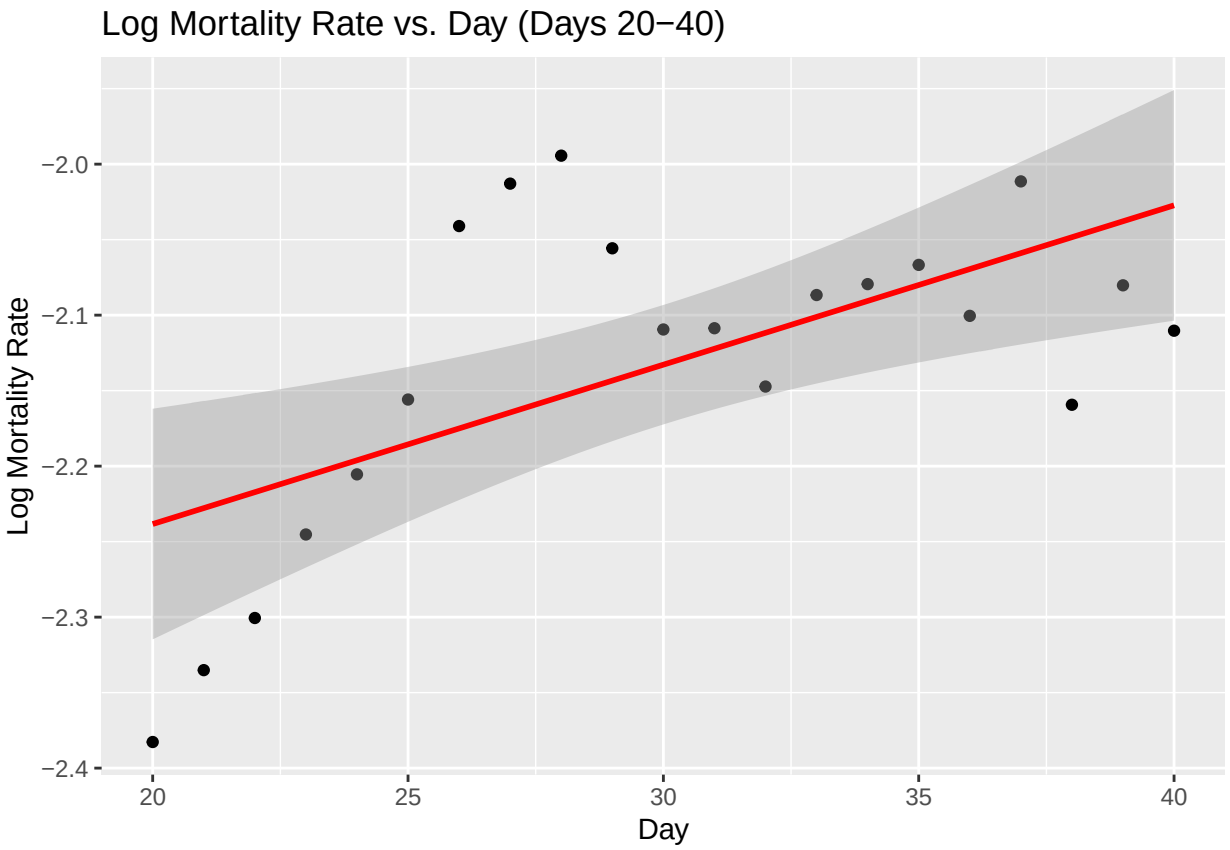
```
# Ensure mort.rate is numeric and handle log transformation properly
medflies$mort.rate <- as.numeric(medflies$mort.rate)
# Add small constant to avoid log(0)
medflies$log_mort_rate <- log(pmax(medflies$mort.rate, 1e-10))

# Fit linear regression on log-transformed data
model_full <- lm(log_mort_rate ~ day, data = medflies)

# Plot
ggplot(medflies, aes(x = day, y = log_mort_rate)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", color = "blue") +
  labs(title = "Log Mortality Rate vs. Day",
       x = "Day",
       y = "Log Mortality Rate")
```




```
# Diagnostic plots
par(mfrow = c(2,2))
plot(model_full)
```




```r
# Try finding a subset where theory holds better
# Let's look at days 20-40
subset_data <- subset(medflies, day >= 20 & day <= 40)
model_subset <- lm(log_mort_rate ~ day, data = subset_data)

ggplot(subset_data, aes(x = day, y = log_mort_rate)) +
  geom_point() +
  geom_smooth(method = "lm", color = "red") +
  labs(title = "Log Mortality Rate vs. Day (Days 20-40)",
       x = "Day",
       y = "Log Mortality Rate")
```

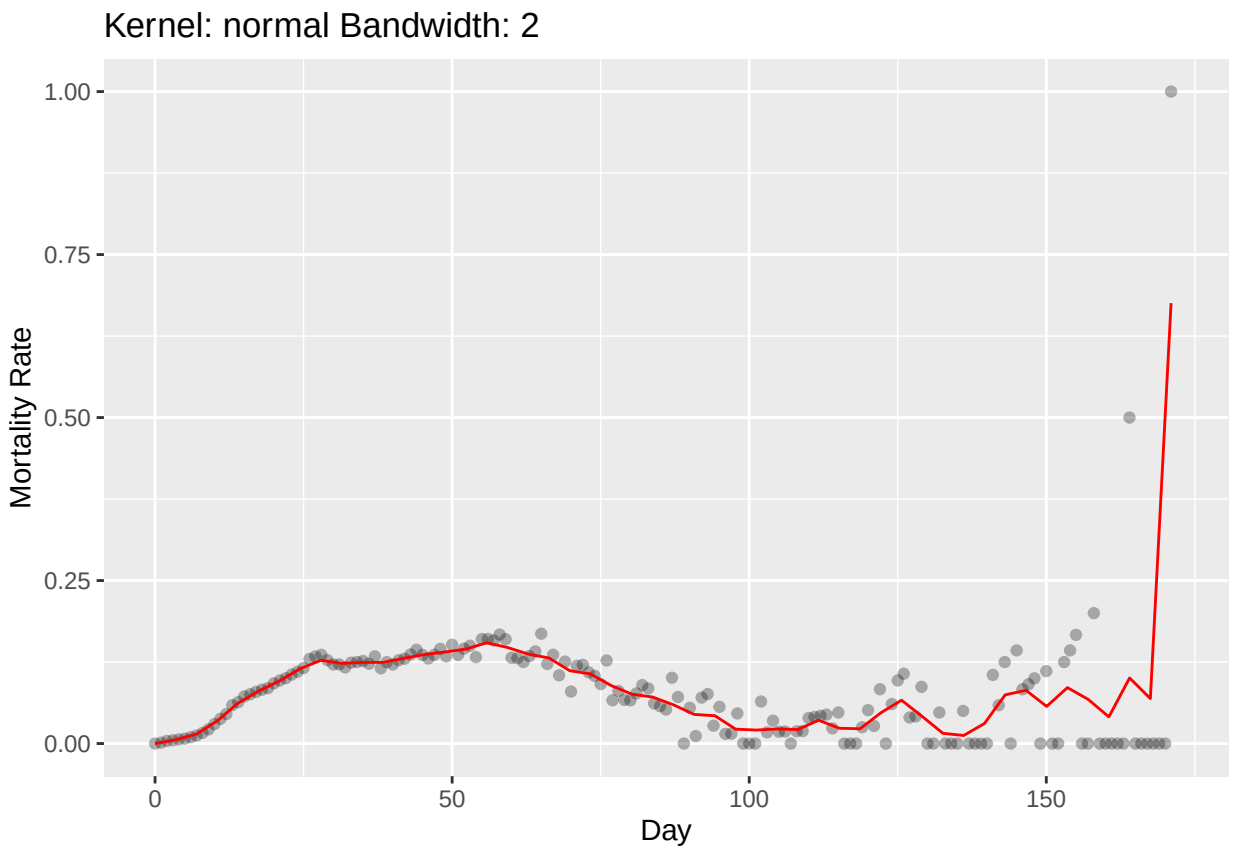


(d) Kernel Regression

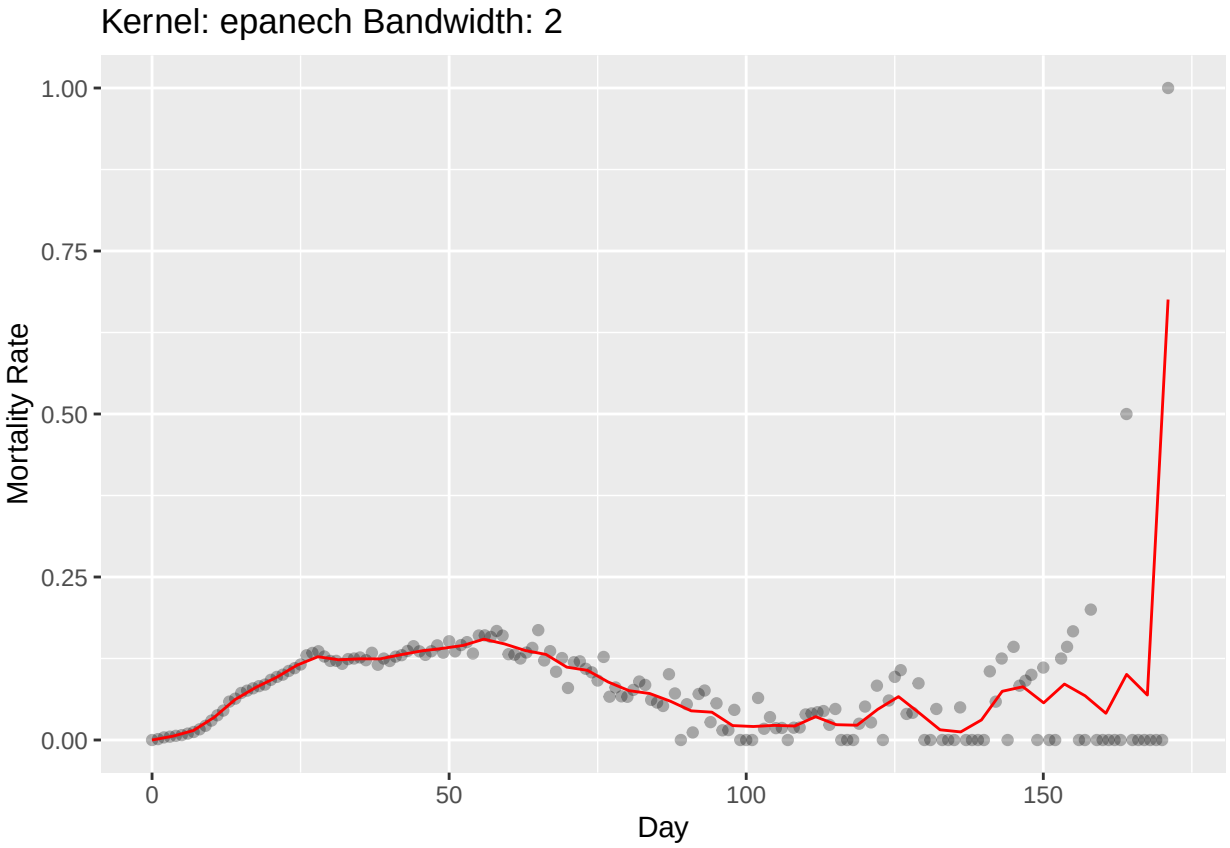
```r
# Try different kernels and bandwidths
h_values <- c(2, 5, 10)
kernels <- c("normal", "epanech", "triangular")

# Function to plot kernel regression with different parameters
plot_kernel <- function(h, kernel) {
  sm.regression(medflies$day,
                medflies$mort.rate,
                h = h,
                kernel = kernel,
                display = "none") -> fit

  data.frame(x = fit$eval.points,
             y = fit$estimate) -> smooth_data

  ggplot() +
    geom_point(data = medflies, aes(x = day, y = mort.rate), alpha = 0.3) +
    geom_line(data = smooth_data, aes(x = x, y = y), color = "red") +
    labs(title = paste("Kernel:", kernel, "Bandwidth:", h),
         x = "Day",
         y = "Mortality Rate")
}
```

```
# Plot different combinations
for(h in h_values) {
  for(k in kernels) {
    print(plot_kernel(h, k))
  }
}
```

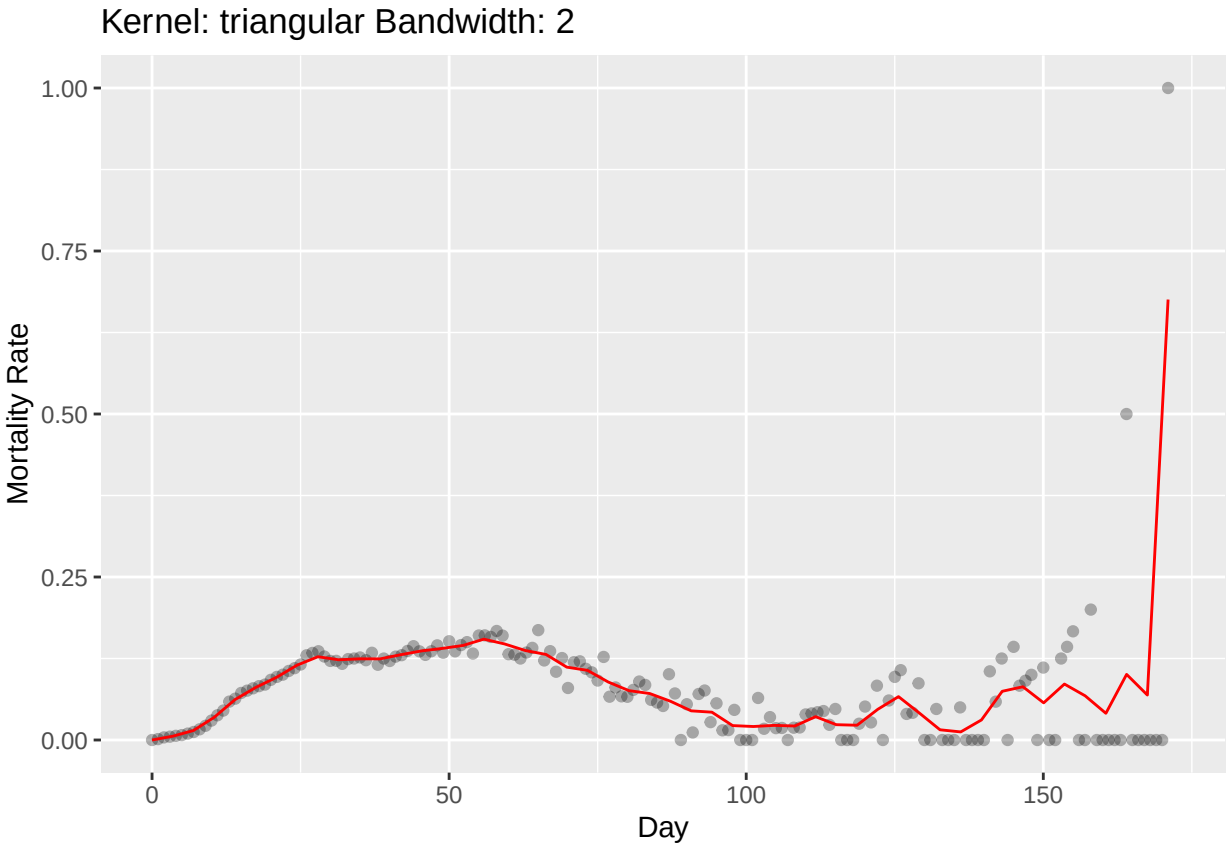
```
## missing data are removed
```



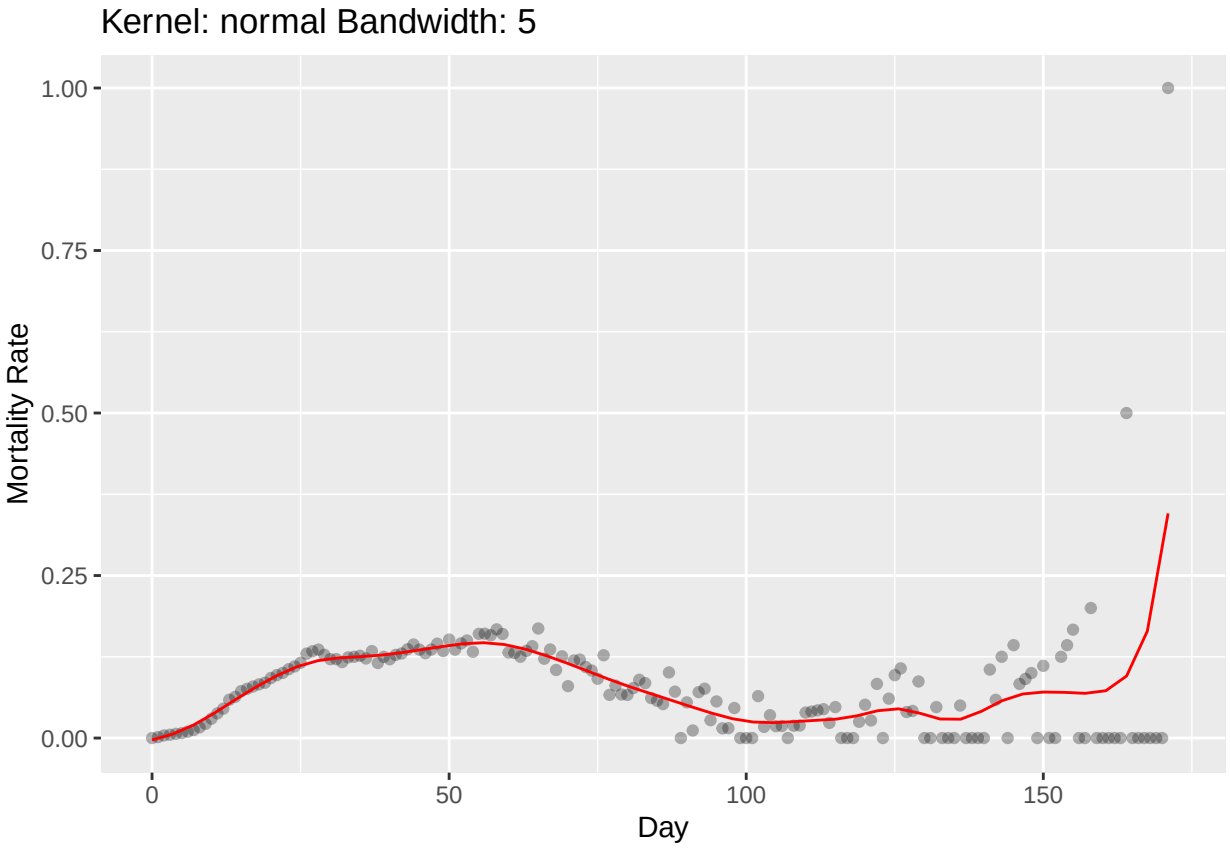

```
## missing data are removed
```



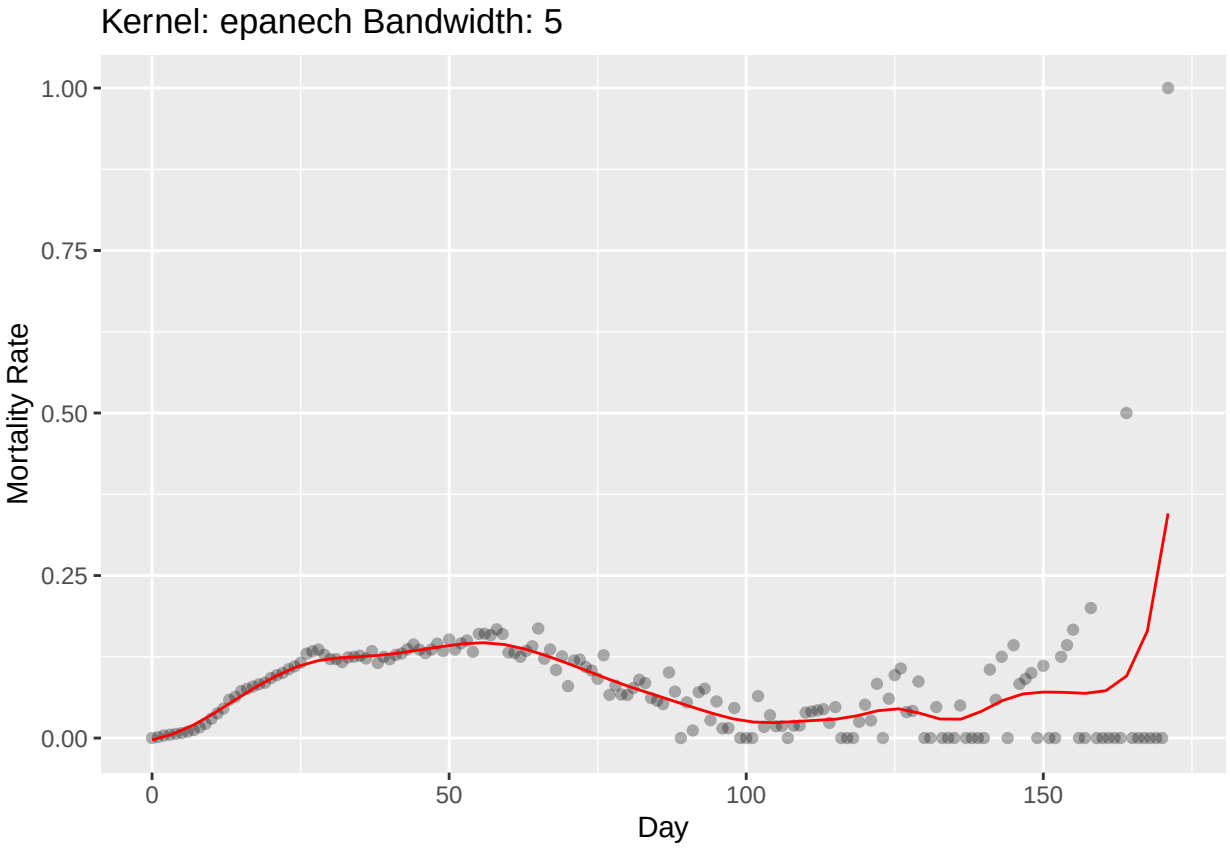

```
## missing data are removed
```



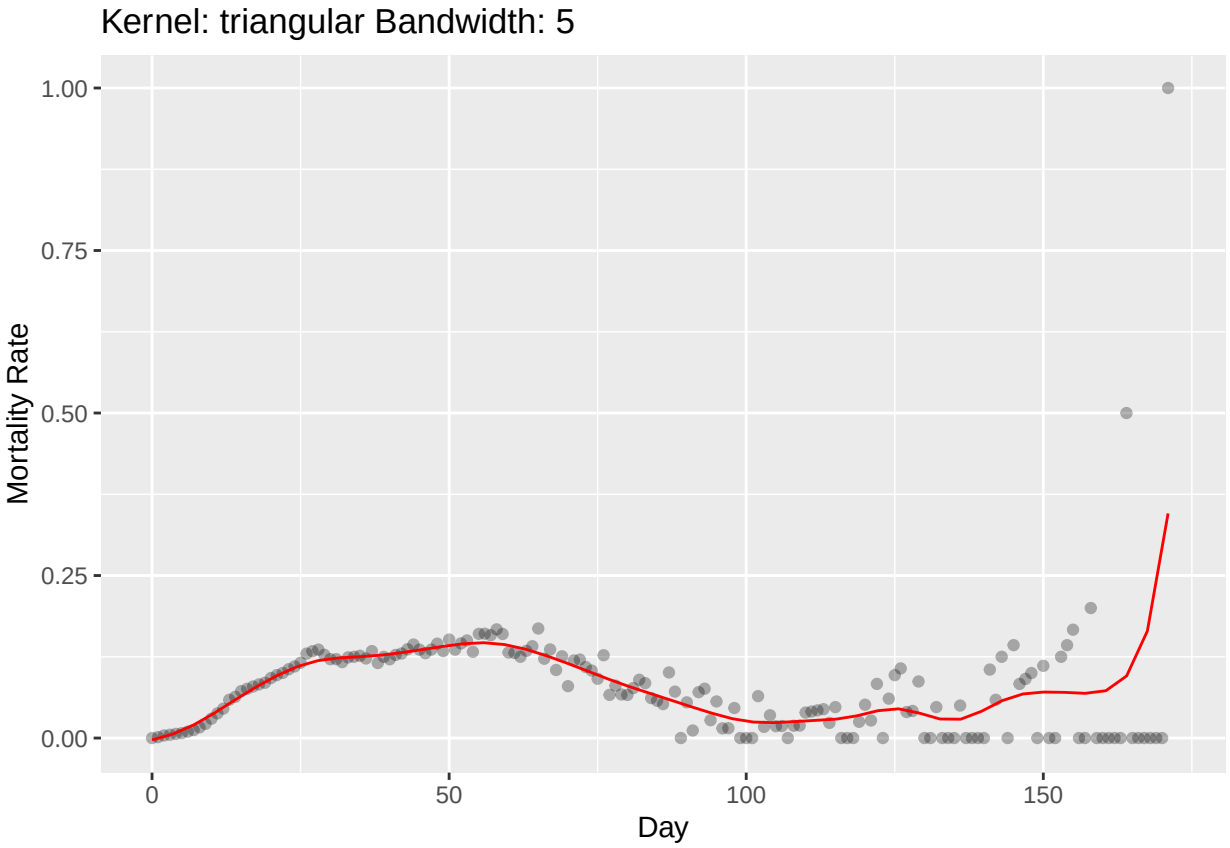

```
## missing data are removed
```



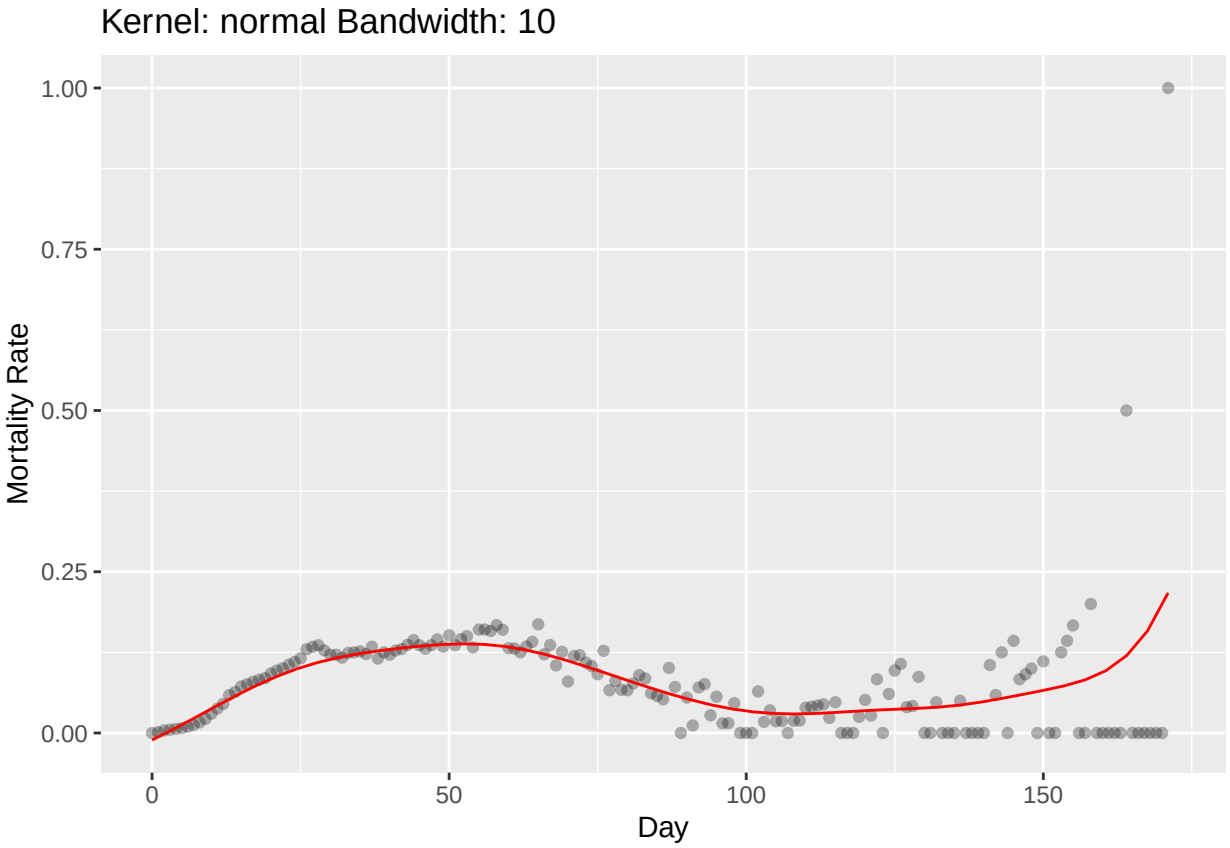

```
## missing data are removed
```



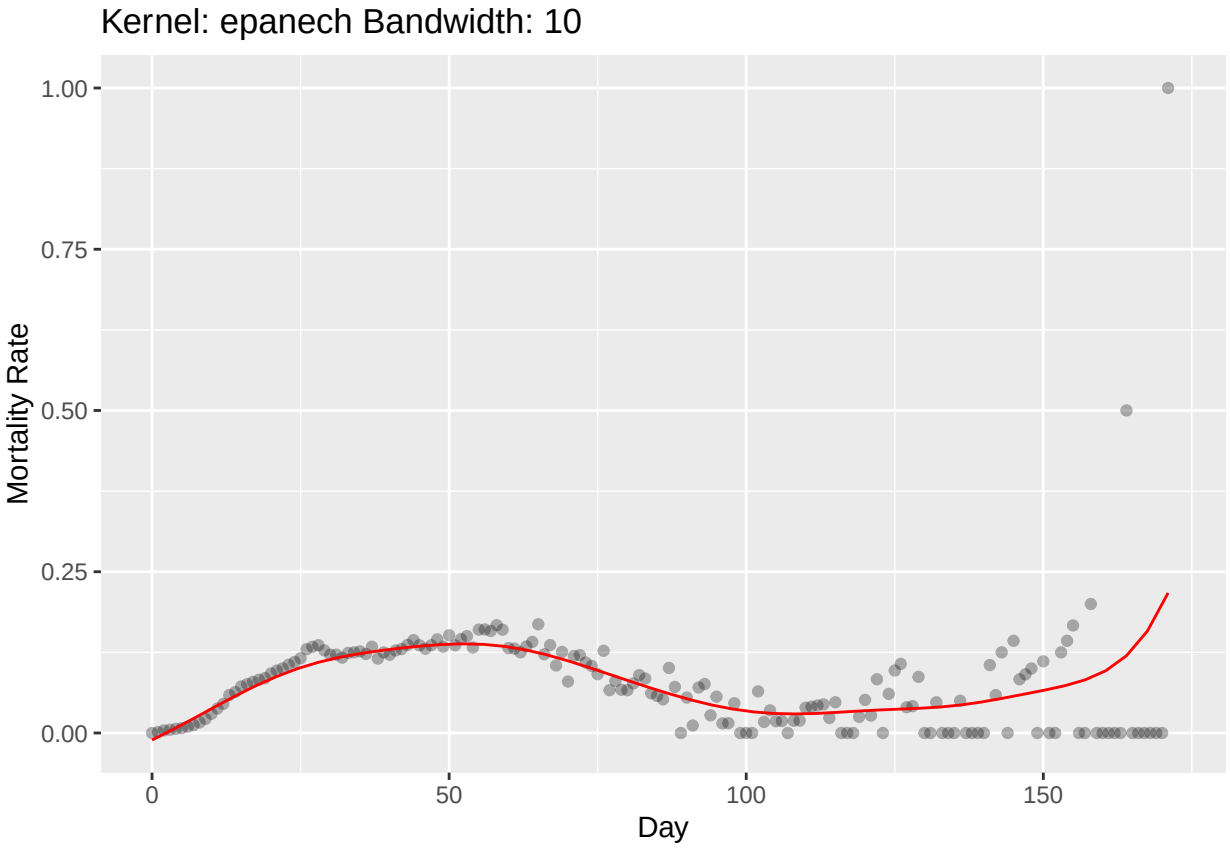

```
## missing data are removed
```



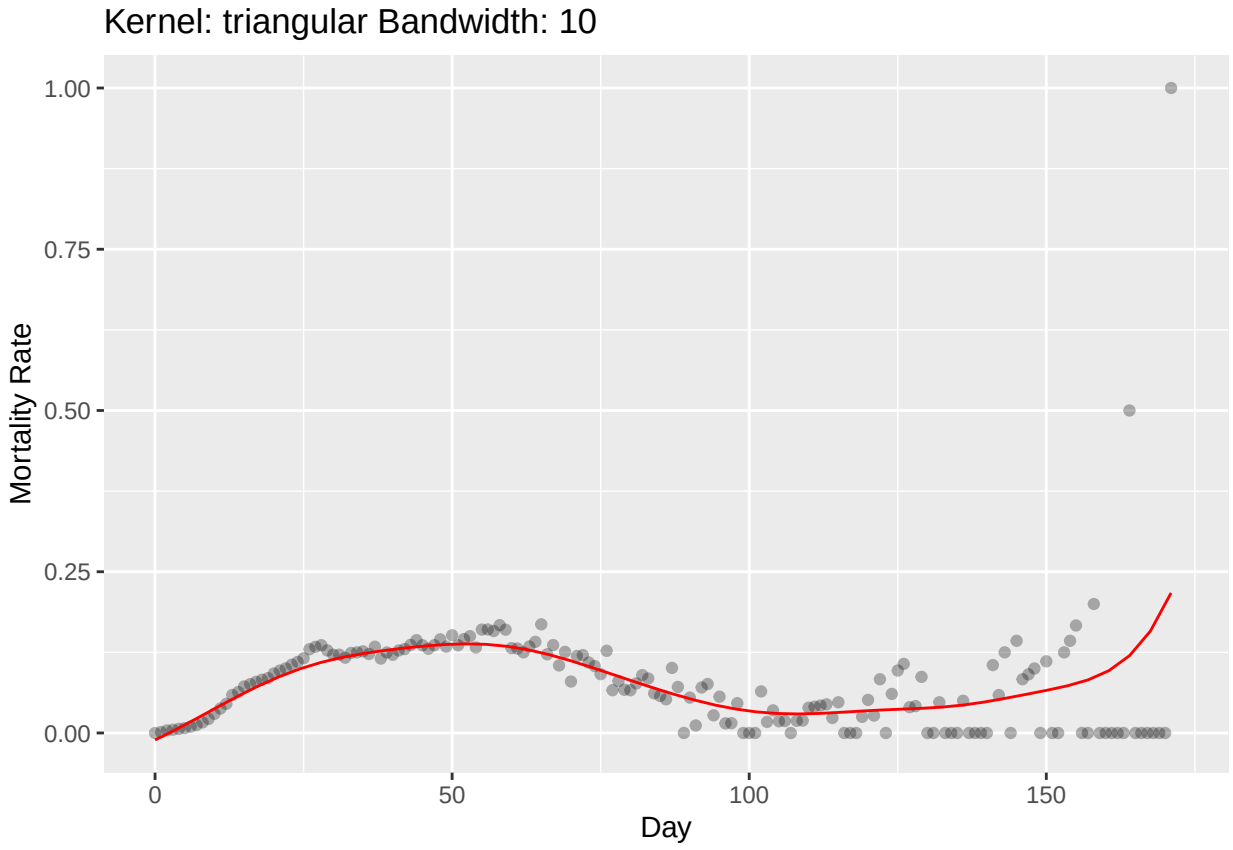

```
## missing data are removed
```




```
## missing data are removed
```



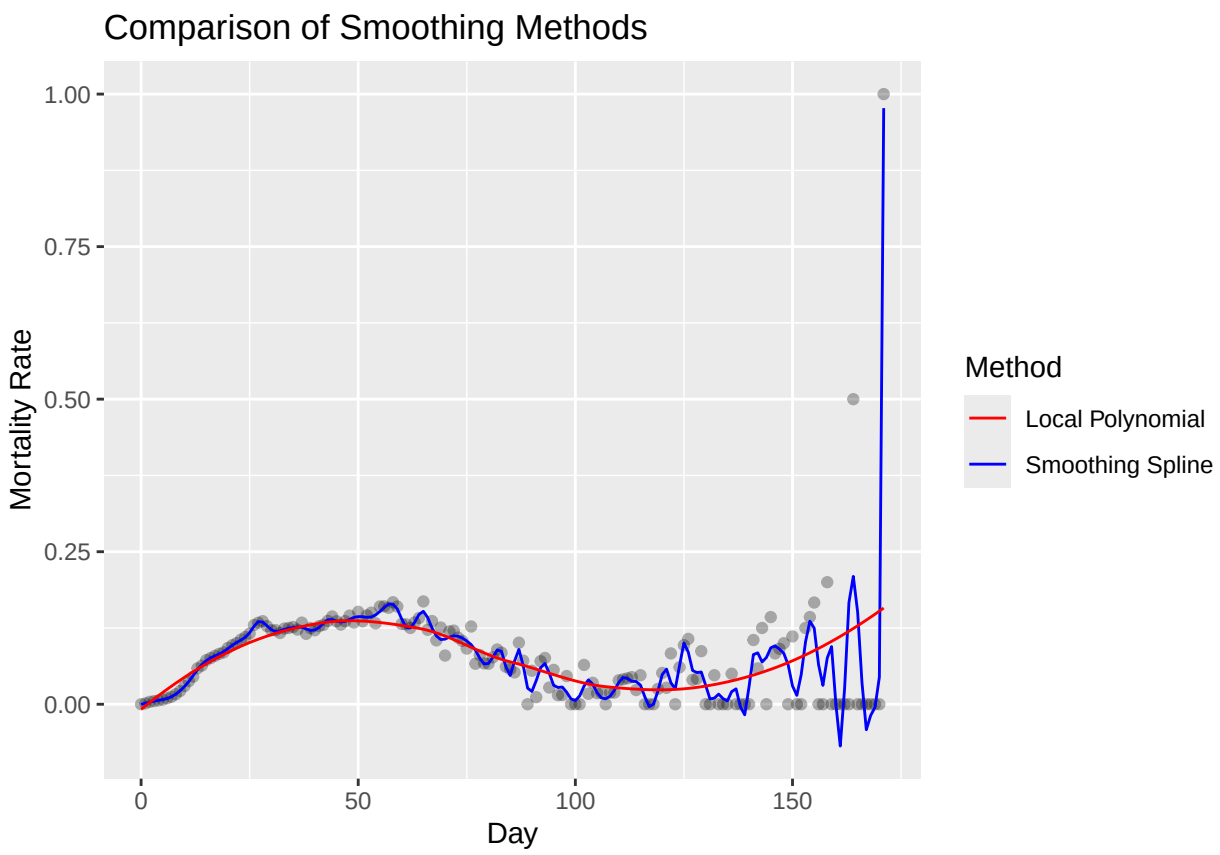

```
## missing data are removed
```



(e) Smoothing Splines and Local Polynomials

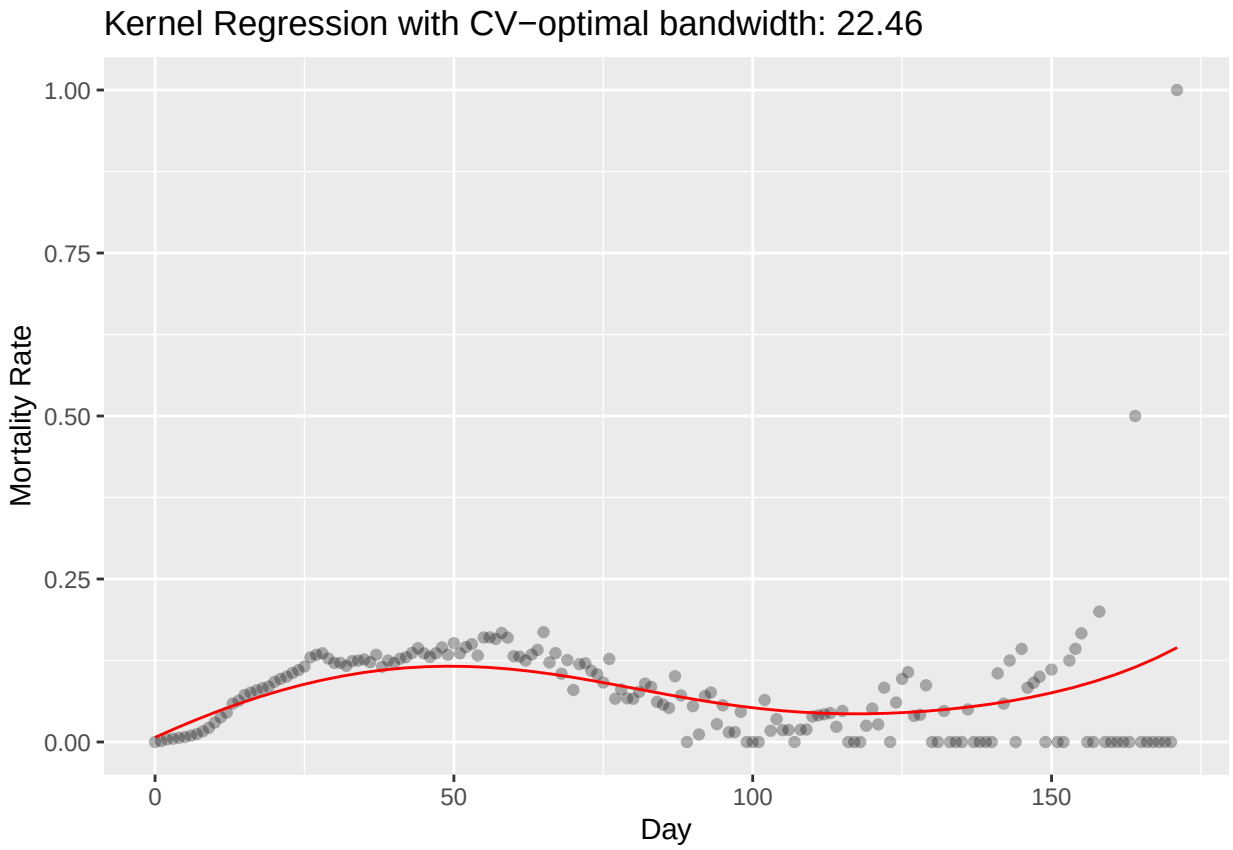
```r
# Clean data for smoothing splines
medflies_clean <- na.omit(medflies)

# (i) Smoothing splines with error handling
tryCatch({
  smooth_spline <- smooth.spline(medflies_clean$day, medflies_clean$mort.rate)
}, error = function(e) {
  message("Error in smooth.spline: ", e$message)
  return(NULL)
})

# (ii) Local polynomials (using loess)
local_poly <- loess(mort.rate ~ day, data = medflies_clean, span = 0.75)

# Plot both methods
ggplot(medflies_clean, aes(x = day, y = mort.rate)) +
  geom_point(alpha = 0.3) +
  {if(!is.null(smooth_spline)) geom_line(aes(y = predict(smooth_spline)$y, color = "Smoothing Spline"))}
  geom_line(aes(y = predict(local_poly), color = "Local Polynomial")) +
  scale_color_manual(values = c("red", "blue")) +
  labs(title = "Comparison of Smoothing Methods",
       x = "Day",
       y = "Mortality Rate",
```

```
       color = "Method")
```



(f) Cross-validation for Optimal Bandwidth

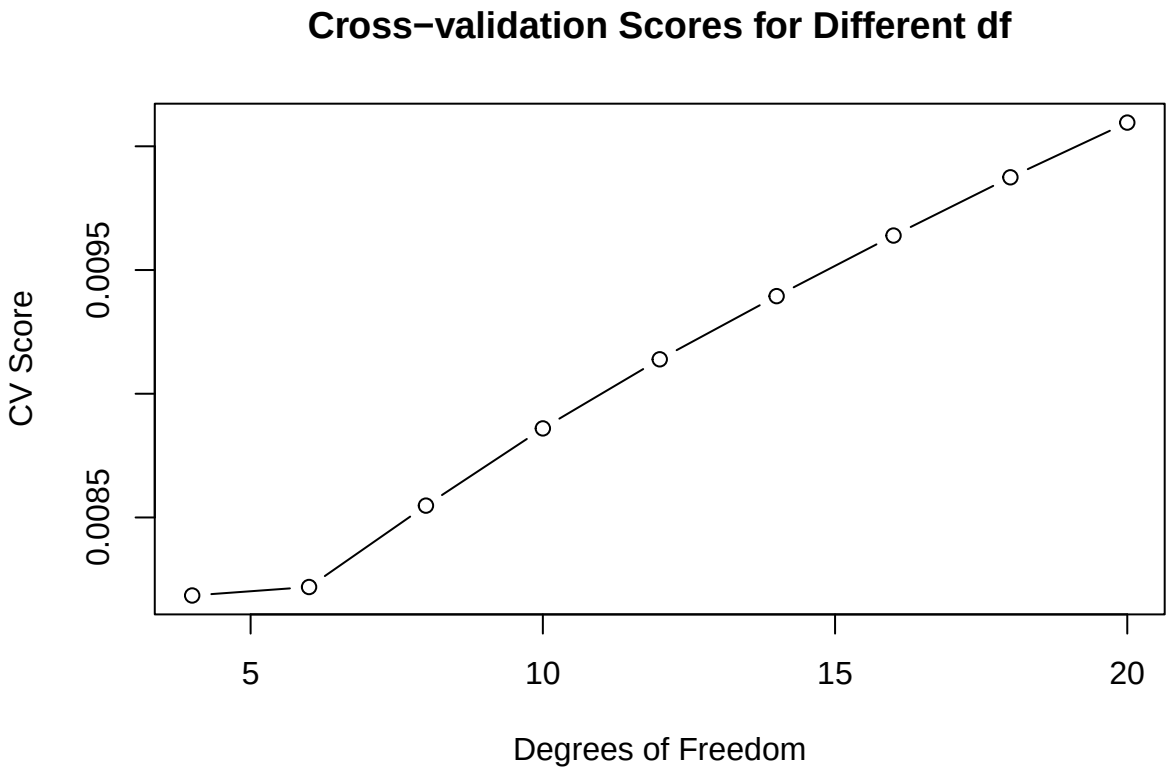
```
# Find optimal bandwidth using cross-validation
h.cv <- hcv(medflies_clean$day, medflies_clean$mort.rate)

# Fit model with optimal bandwidth
sm.regression(medflies_clean$day,
              medflies_clean$mort.rate,
              h = h.cv,
              display = "none") -> optimal_fit

# Plot result
ggplot() +
  geom_point(data = medflies_clean, aes(x = day, y = mort.rate), alpha = 0.3) +
  geom_line(data = data.frame(x = optimal_fit$eval.points,
                              y = optimal_fit$estimate),
            aes(x = x, y = y), color = "red") +
  labs(title = paste("Kernel Regression with CV-optimal bandwidth:", round(h.cv, 2)),
       x = "Day",
       y = "Mortality Rate")
```



(g) Manual Cross-validation for Smoothing Splines

```
# Function to compute CV score for smoothing splines
cv_spline <- function(df) {
  cv_scores <- numeric(length(medflies_clean$mort.rate))

  for(i in 1:length(medflies_clean$mort.rate)) {
    tryCatch({
      # Fit model without i-th observation
      fit <- smooth.spline(medflies_clean$day[-i],
                           medflies_clean$mort.rate[-i],
                           df = df)
      # Predict i-th observation
      pred <- predict(fit, medflies_clean$day[i])$y
      # Compute squared error
      cv_scores[i] <- (medflies_clean$mort.rate[i] - pred)^2
    }, error = function(e) {
      cv_scores[i] <- NA
    })
  }

  mean(cv_scores, na.rm = TRUE)
}

# Try different degrees of freedom
```

```r
df_values <- seq(4, 20, by = 2)
cv_results <- sapply(df_values, cv_spline)

# Plot CV scores
plot(df_values, cv_results, type = "b",
     xlab = "Degrees of Freedom",
     ylab = "CV Score",
     main = "Cross-validation Scores for Different df")
```




```r
# Optimal df
opt_df <- df_values[which.min(cv_results)]
```

(h) Scientific Questions

Non-parametric models are useful for: - Understanding the general pattern of mortality rates over time without assuming a specific functional form - Identifying periods of unusual mortality rate changes - Describing the day-to-day variation in mortality rates

Linear models (like in Gompertz's theory) are useful for: - Testing specific theories about exponential growth in mortality rates - Making predictions about mortality rates at specific ages - Quantifying the rate of increase in mortality over time

The choice between parametric and non-parametric models depends on the scientific question at hand and the assumptions we're willing to make about the underlying relationship between variables.