Quantifying model fit

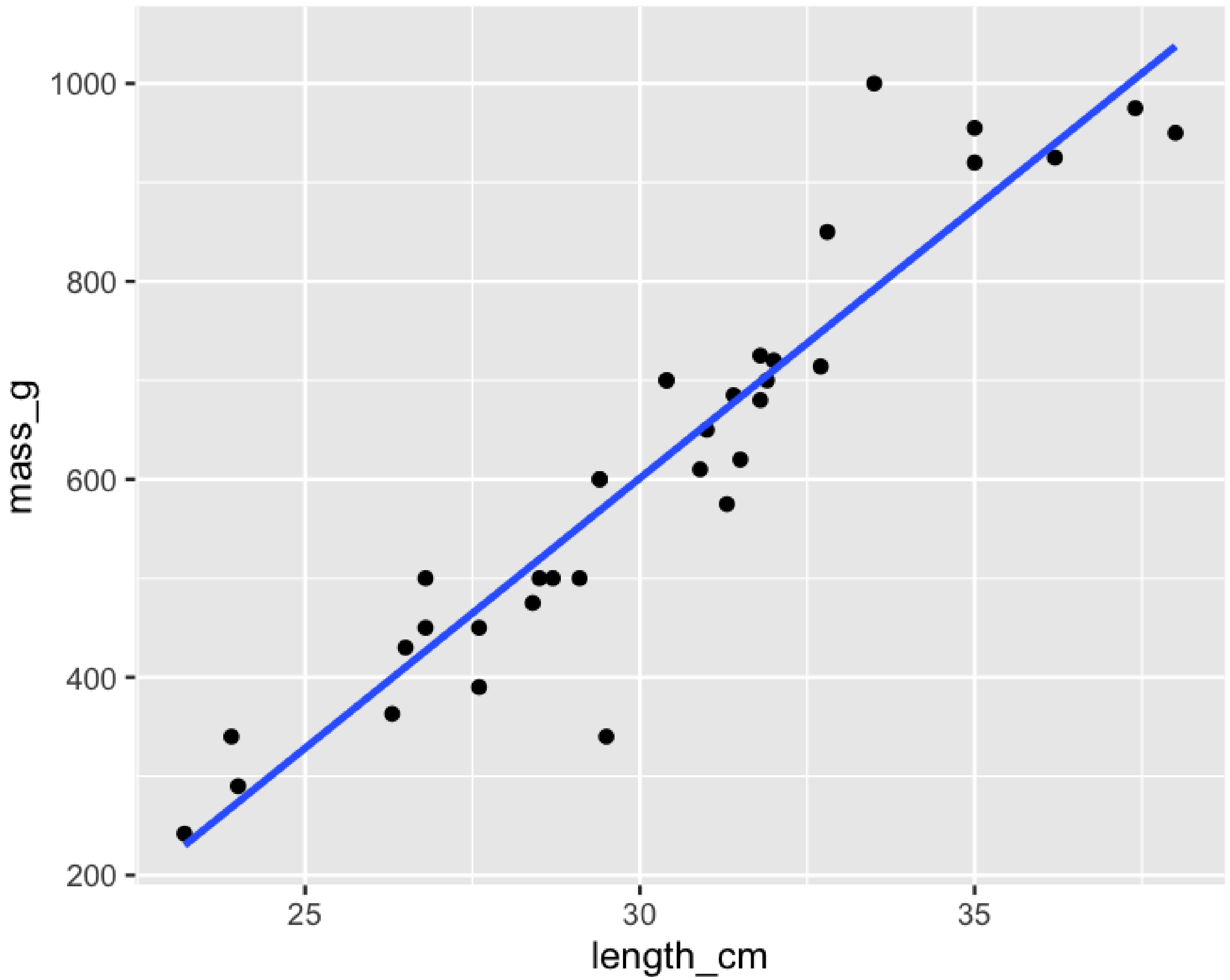
INTRODUCTION TO REGRESSION IN R

Richie Cotton

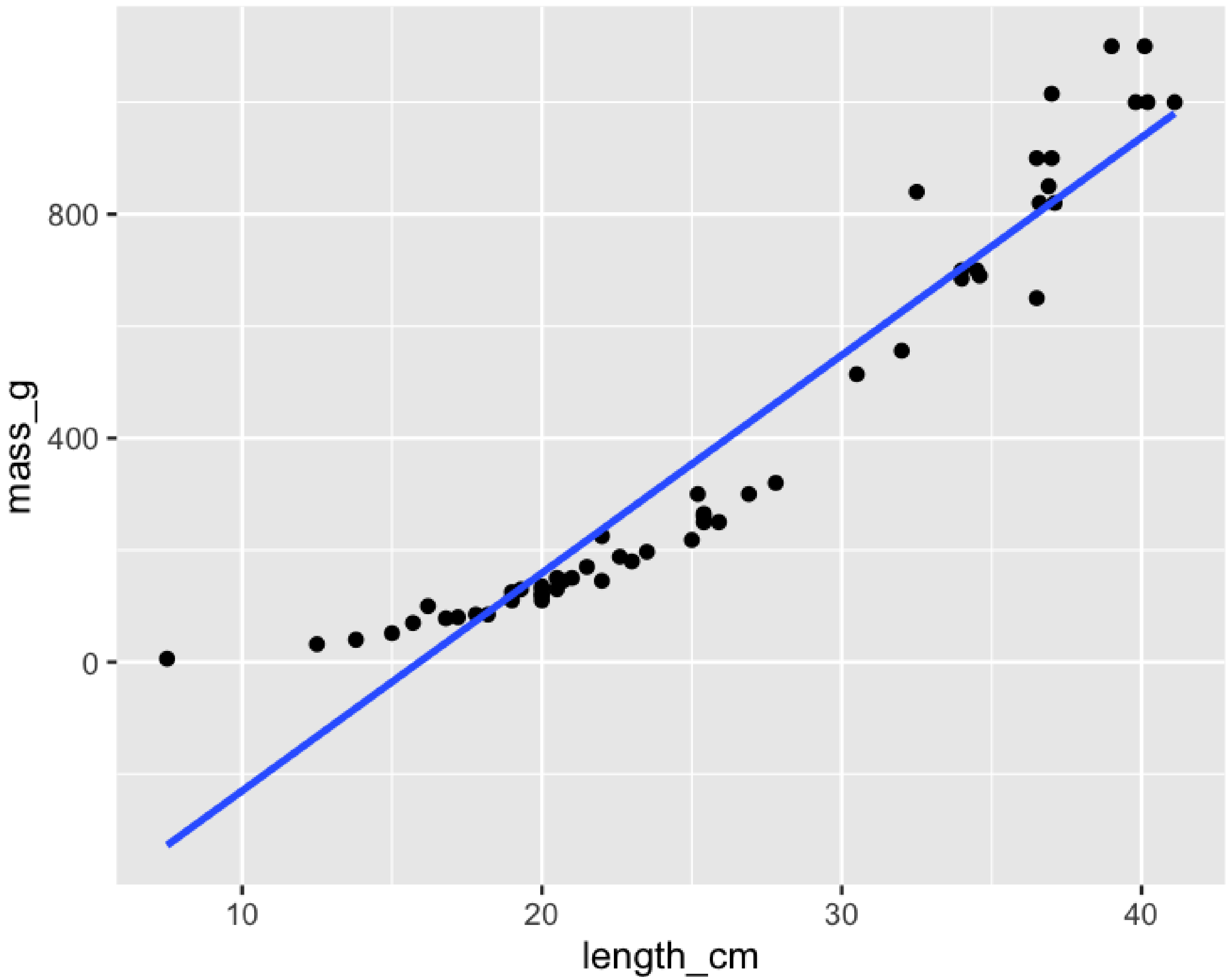
Learning Solutions Architect at DataCamp

Bream and perch models

Bream



Perch



Coefficient of determination

Sometimes called "r-squared" or "R-squared".

> the proportion of the variance in the response variable that is predictable from the explanatory variable

- `1` means a perfect fit
- `0` means the worst possible fit

summary()

Look at the value titled "Multiple R-Squared"

```
mdl_bream <- lm(mass_g ~ length_cm, data = bream)
```

```
summary(mdl_bream)
```

```
# Some lines of output omitted


Residual standard error: 74.15 on 33 degrees of freedom
Multiple R-squared:  0.8781,    Adjusted R-squared:  0.8744
F-statistic: 237.6 on 1 and 33 DF,  p-value: < 2.2e-16
```

glance()

```
library(broom)
library(dplyr)
mdl_bream %>%
  glance()
```

```
# A tibble: 1 x 11
  r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC deviance df.residual
      <dbl>         <dbl> <dbl>     <dbl>    <dbl> <int>  <dbl> <dbl> <dbl>    <dbl>       <int>
1     0.878         0.874  74.2      238. 1.22e-16     2  -199.  405.  409.  181452.          33
```

```
mdl_bream %>%
  glance() %>%
  pull(r.squared)
```

```
0.8780627
```

It's just correlation squared

```
bream %>%
  summarize(
    coeff_determination = cor(length_cm, mass_g) ^ 2
  )
```

```
  coeff_determination
1           0.8780627
```

Residual standard error (RSE)

> a "typical" difference between a prediction and an observed response

It has the same unit as the response variable.

summary() again

Look at the value titled "Residual standard error"

```
summary(mdl_bream)
```

```
# Some lines of output omitted


Residual standard error: 74.15 on 33 degrees of freedom
Multiple R-squared:  0.8781,    Adjusted R-squared:  0.8744
F-statistic: 237.6 on 1 and 33 DF,  p-value: < 2.2e-16
```

glance() again

```r
library(broom)
library(dplyr)
mdl_bream %>%
  glance()
```

```
# A tibble: 1 x 11
  r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC deviance df.residual
      <dbl>         <dbl> <dbl>     <dbl>    <dbl> <int>  <dbl> <dbl> <dbl>    <dbl>       <int>
1     0.878         0.874  74.2      238. 1.22e-16     2  -199.  405.  409.  181452.          33
```

```r
mdl_bream %>%
  glance() %>%
  pull(sigma)
```

```
74.15224
```

Calculating RSE: residuals squared

```
bream %>%
  mutate(
    residuals_sq = residuals(mdl_bream) ^ 2
  )
```

```
  species mass_g length_cm residuals_sq
1   Bream    242      23.2     138.9571
2   Bream    290      24.0     260.7586
3   Bream    340      23.9    5126.9926
4   Bream    363      26.3    1318.9197
5   Bream    430      26.5     390.9743
6   Bream    450      26.8     547.9380
...
```

Calculating RSE: sum of residuals squared

```
bream %>%
  mutate(
    residuals_sq = residuals(mdl_bream) ^ 2
  ) %>%
  summarize(
    resid_sum_of_sq = sum(residuals_sq)
  )
```

```
  resid_sum_of_sq
1        181452.3
```

Calculating RSE: degrees of freedom

Degrees of freedom equals the number of observations minus the number of model coefficients.

```
bream %>%
  mutate(
    residuals_sq = residuals(mdl_bream) ^ 2
  ) %>%
  summarize(
    resid_sum_of_sq = sum(residuals_sq),
    deg_freedom = n() - 2
  )
```

```
  resid_sum_of_sq deg_freedom
1        181452.3          33
```

Calculating RSE: square root of ratio

```r
bream %>%
  mutate(
    residuals_sq = residuals(mdl_bream) ^ 2
  ) %>%
  summarize(
    resid_sum_of_sq = sum(residuals_sq),
    deg_freedom = n() - 2,
    rse = sqrt(resid_sum_of_sq / deg_freedom)
  )
```

```
  resid_sum_of_sq deg_freedom      rse
1        181452.3          33 74.15224
```

Interpreting RSE

`mdl_bream` has an RSE of `74`.

> The difference between predicted bream masses and observed bream masses is typically about 74g.

Root-mean-square error (RMSE)

Residual standard error

```
bream %>%
  mutate(
    residuals_sq = residuals(mdl_bream) ^ 2
  ) %>%
  summarize(
    resid_sum_of_sq = sum(residuals_sq),
    deg_freedom = n() - 2,
    rse = sqrt(resid_sum_of_sq / deg_freedom)
  )
```

Root-mean-square error

```
bream %>%
  mutate(
    residuals_sq = residuals(mdl_bream) ^ 2
  ) %>%
  summarize(
    resid_sum_of_sq = sum(residuals_sq),
    n_obs = n(),
    rmse = sqrt(resid_sum_of_sq / n_obs)
  )
```

Let's practice!

Visualizing model fit

INTRODUCTION TO REGRESSION IN R

Richie Cotton

Learning Solutions Architect at DataCamp

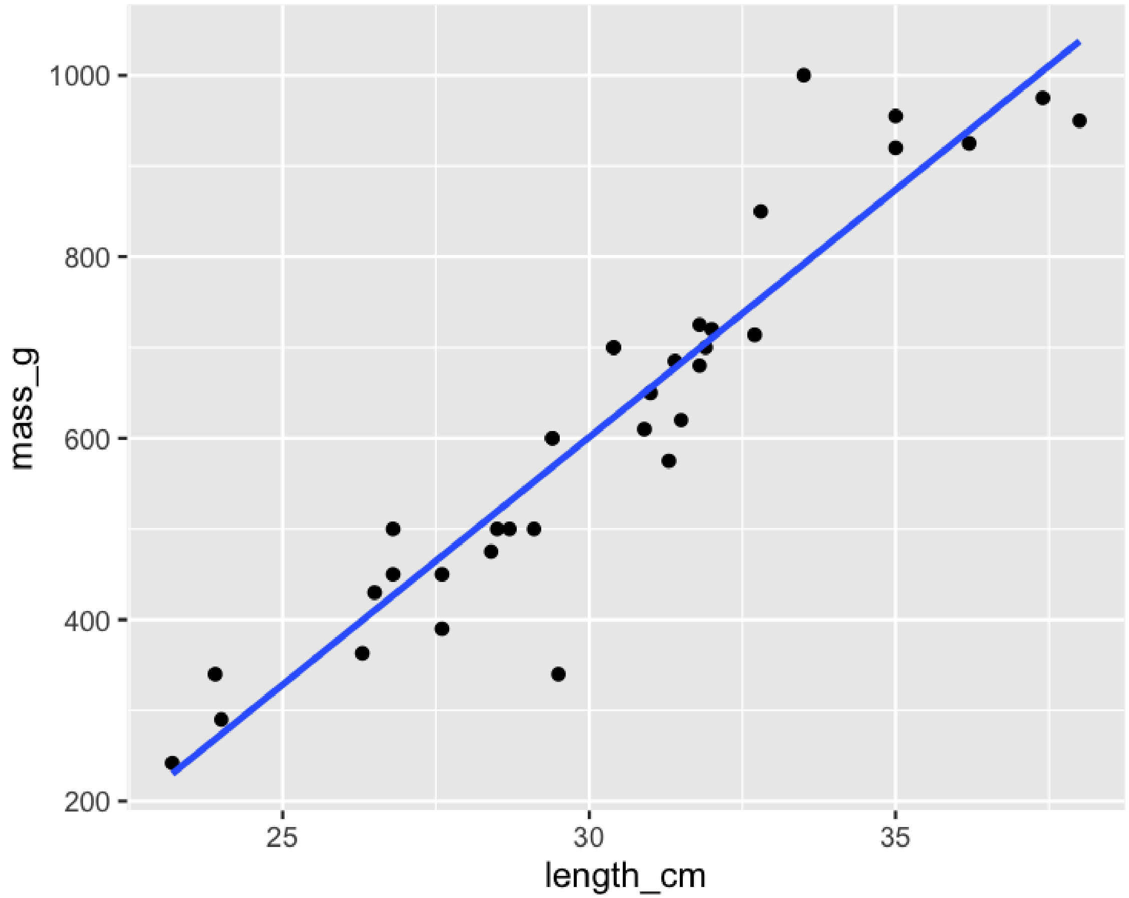
Hoped for properties of residuals

- Residuals are normally distributed
- The mean of the residuals is zero

Bream and perch again

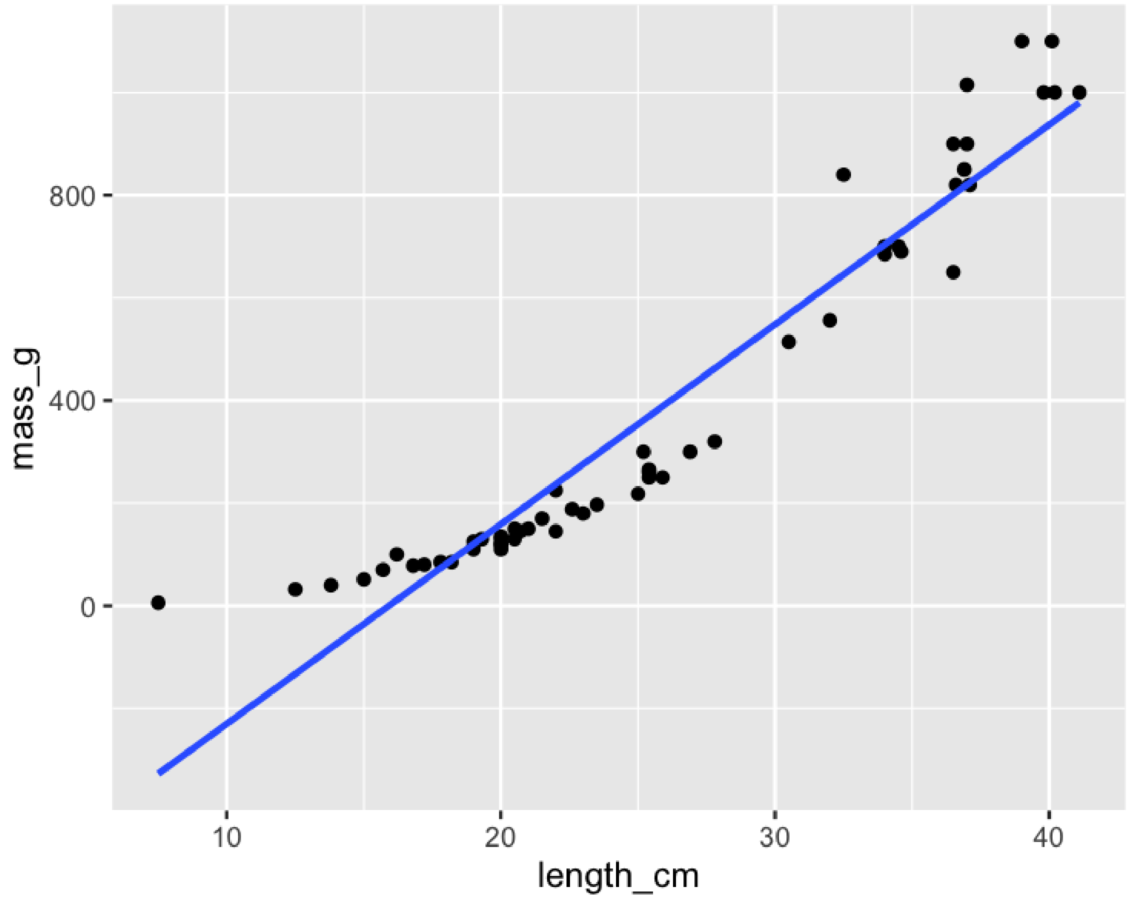
Bream: the "good" model

```
mdl_bream <- lm(mass_g ~ length_cm, data = bream)
```



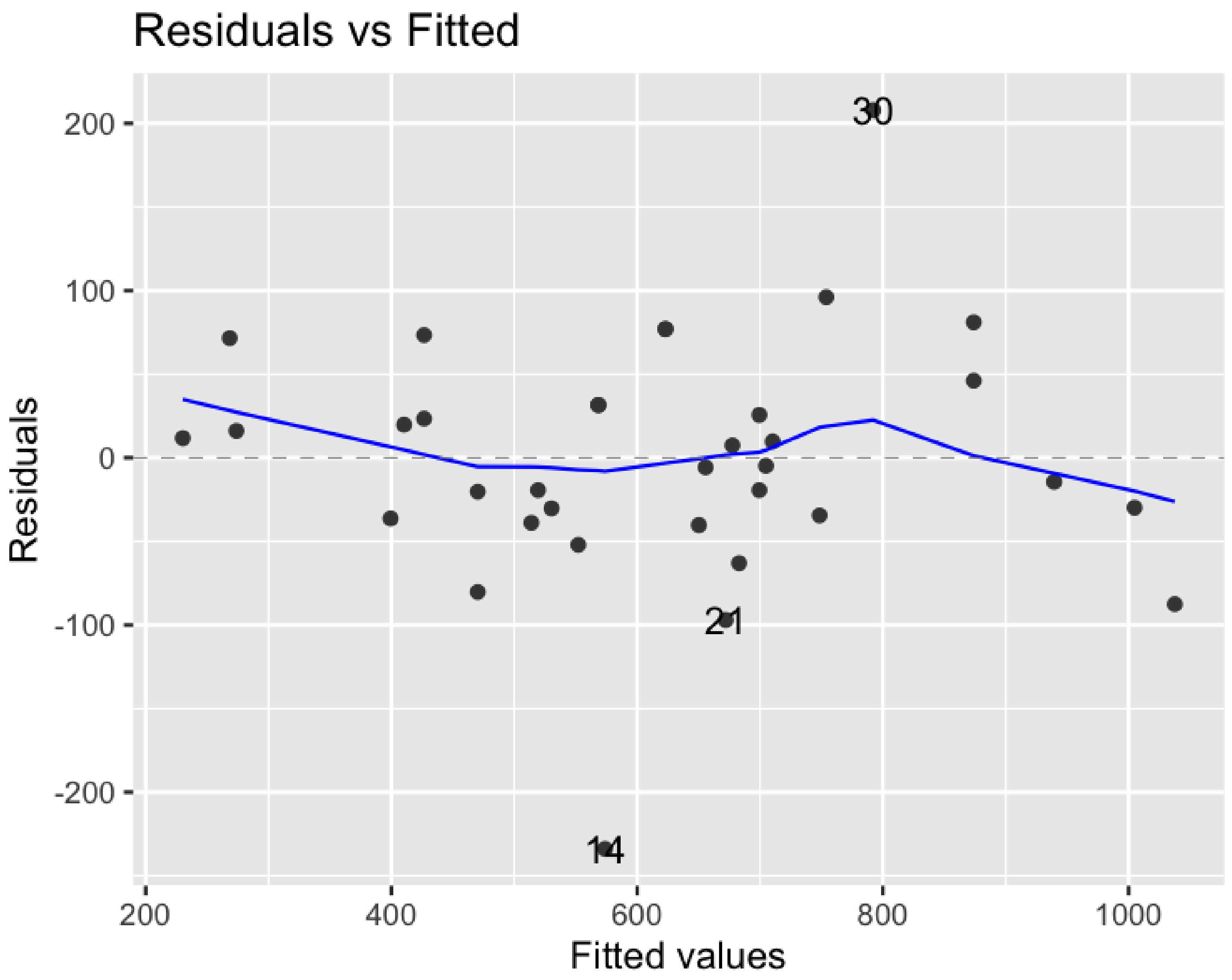
Perch: the "bad" model

```
mdl_perch <- lm(mass_g ~ length_cm, data = perch)
```

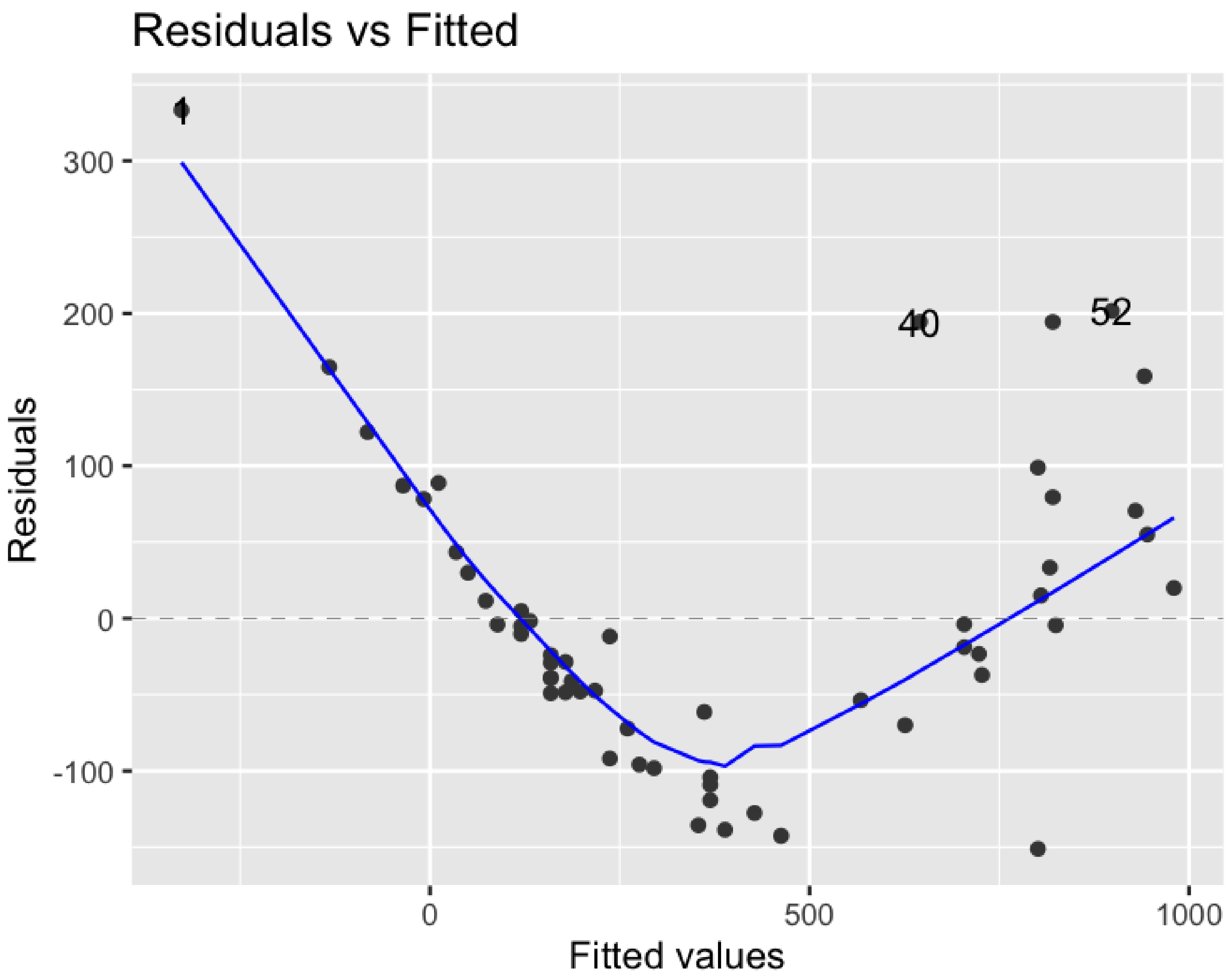


Residuals vs. fitted values

Bream

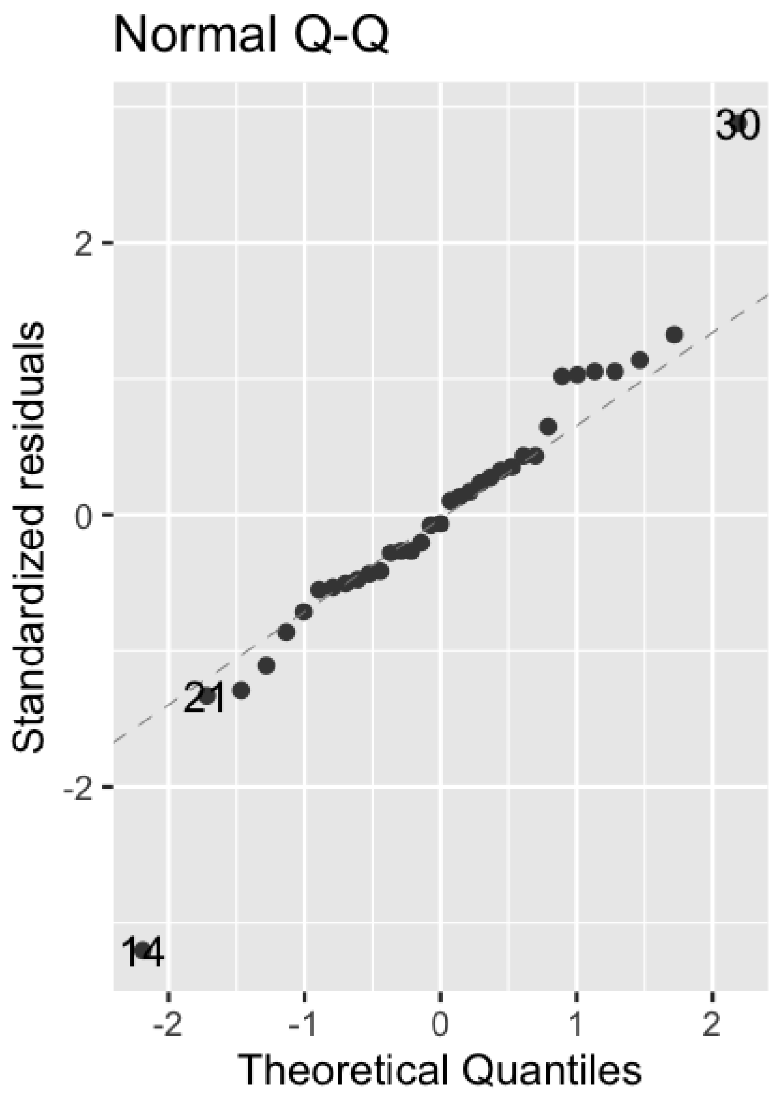


Perch

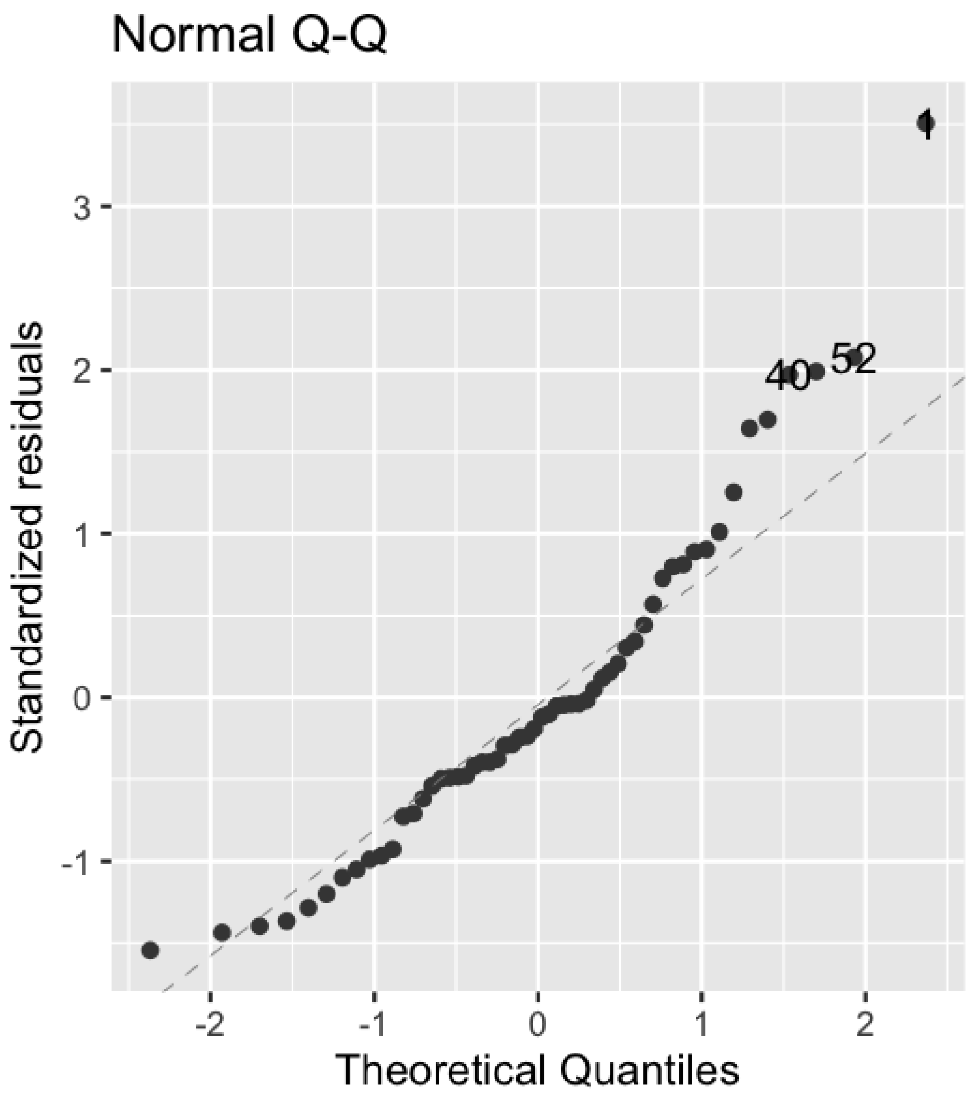


Q-Q plot

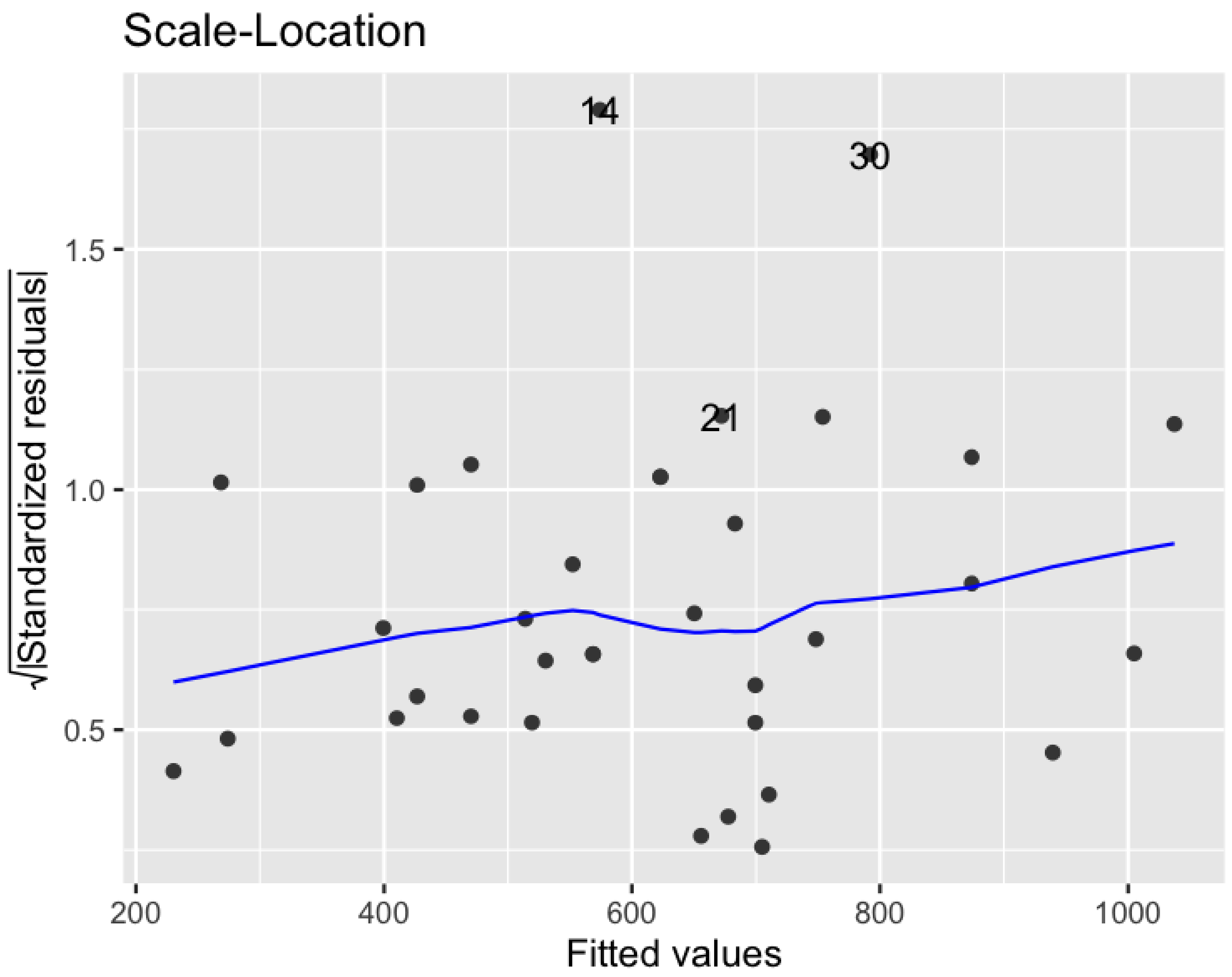
Bream

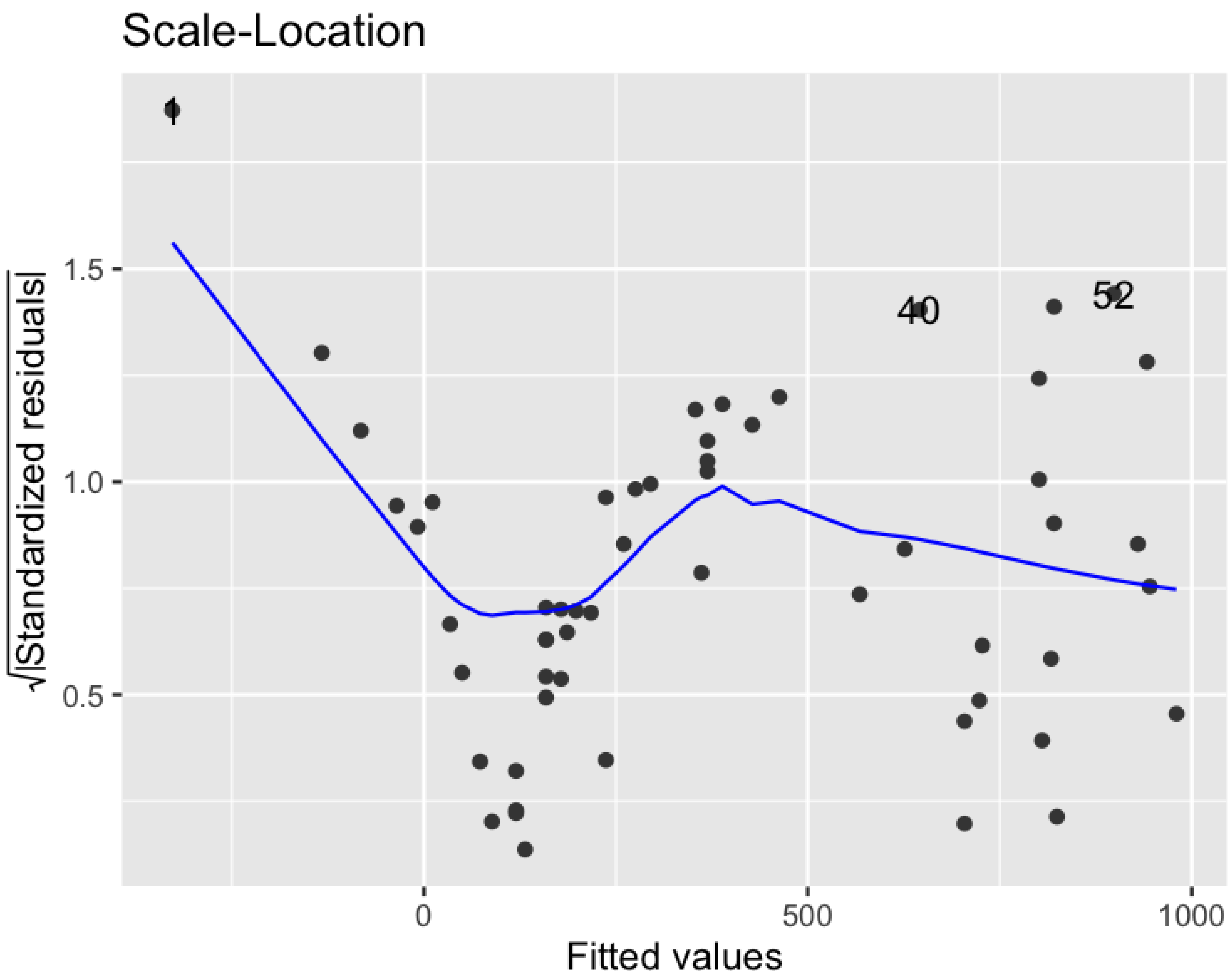


Perch



Scale-location





autoplot()

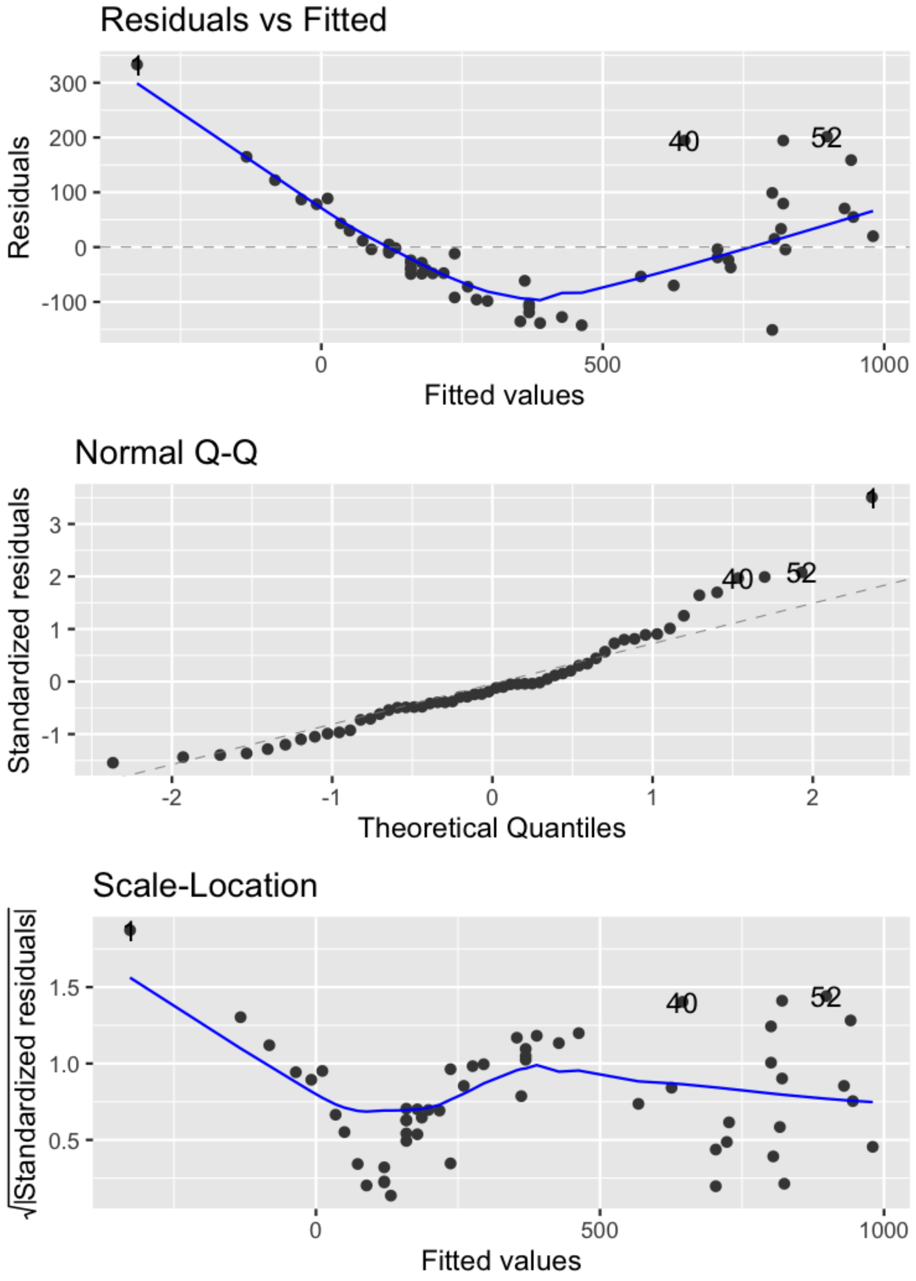
```
library(ggplot2)
library(ggfortify)

autoplot(model_object, which = ???)
```

Values for `which`

- `1` residuals vs. fitted values
- `2` Q-Q plot
- `3` scale-location

autoplot() with the perch model

```
autoplot(
  mdl_perch,
  which = 1:3,
  nrow = 3,
  ncol = 1
)
```



'Autoplots, roll out!' - Plotimus Prime

INTRODUCTION TO REGRESSION IN R

Outliers, leverage, and influence

INTRODUCTION TO REGRESSION IN R

Richie Cotton

Learning Solutions Architect at DataCamp

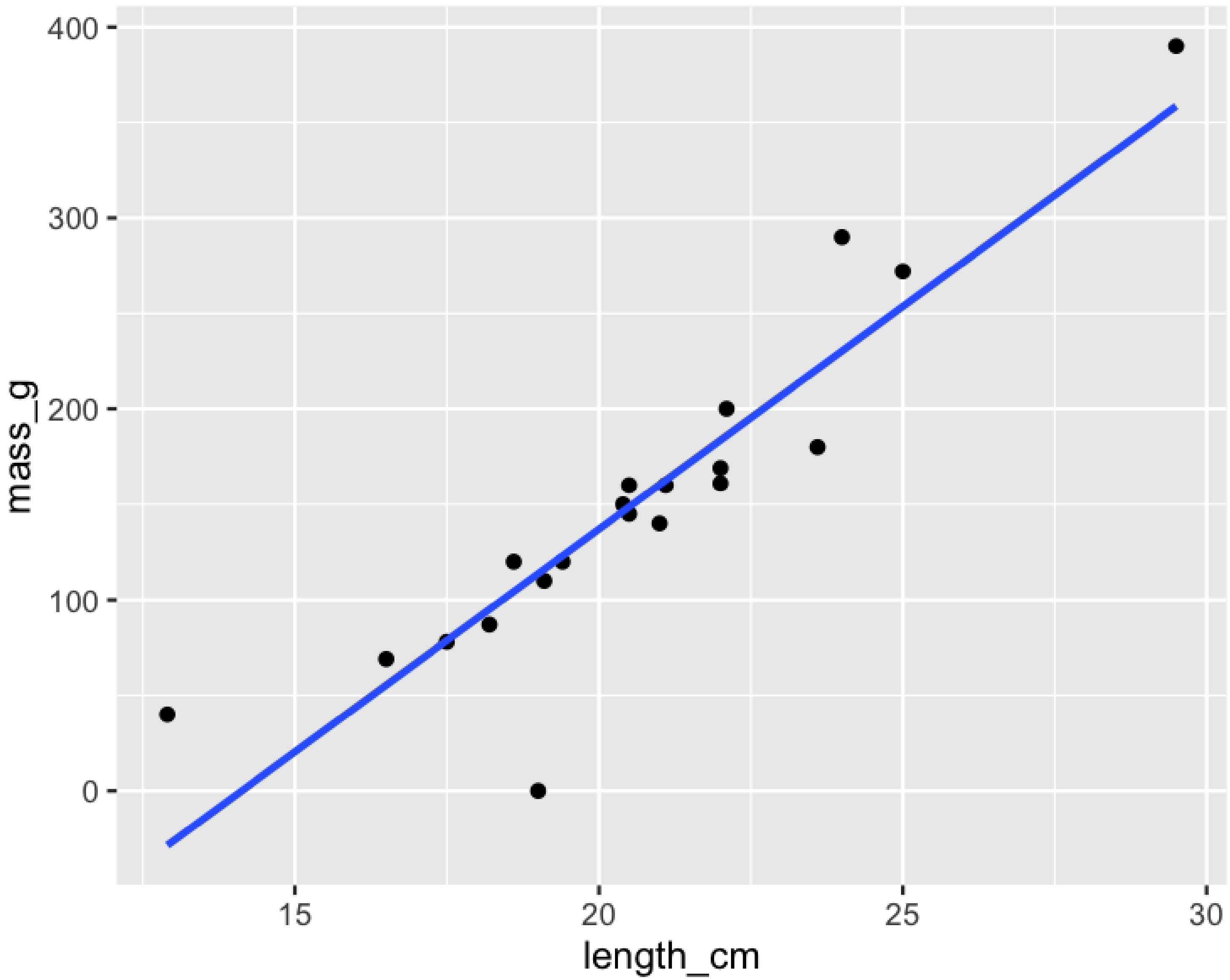
Roach dataset

```
roach <- fish %>%
  filter(species == "Roach")
```

species	length_cm	mass_g
Roach	12.9	40
Roach	16.5	69
Roach	17.5	78
Roach	18.2	87
Roach	18.6	120
...	...	...

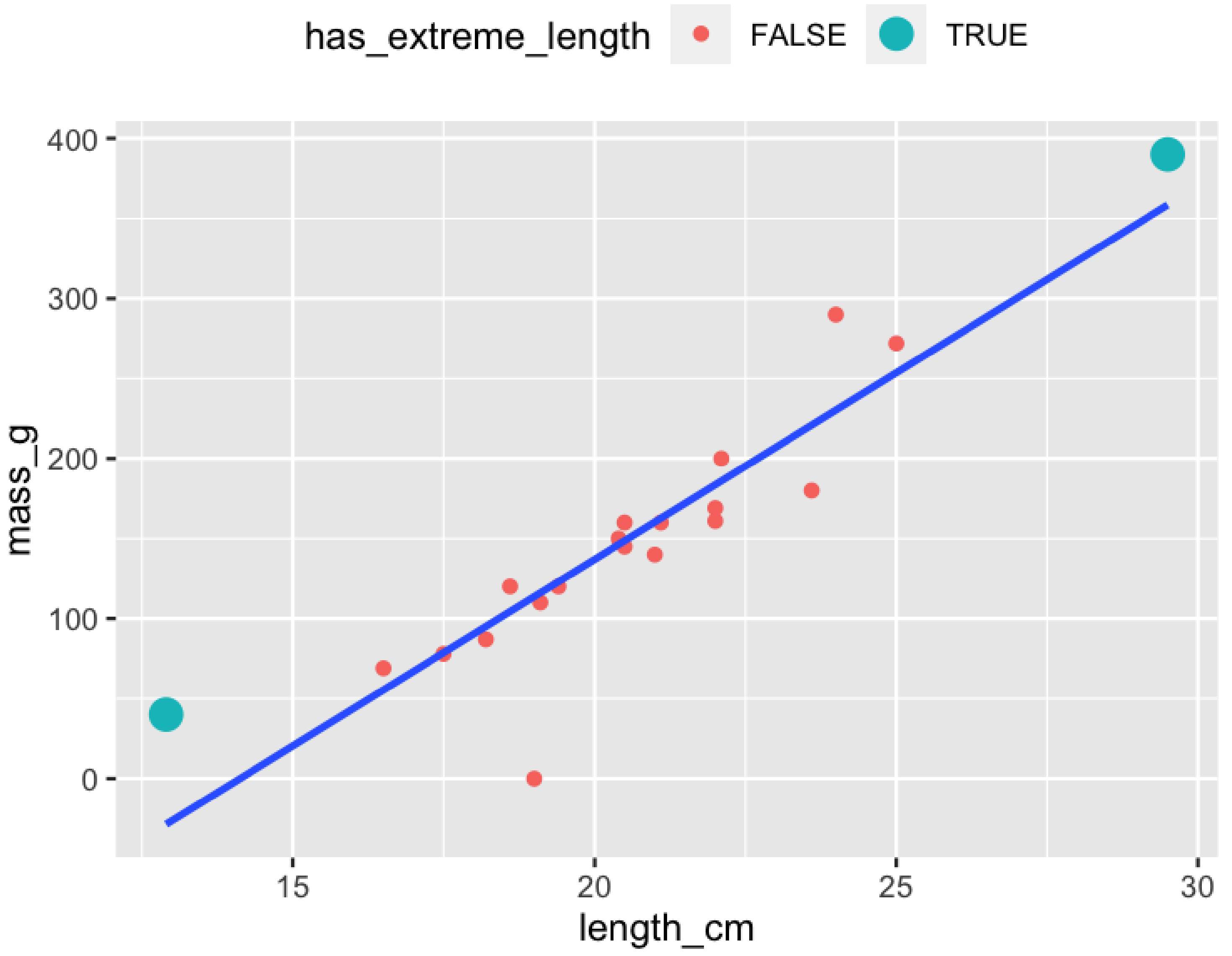
Which points are outliers?

```
ggplot(roach, aes(length_cm, mass_g)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```



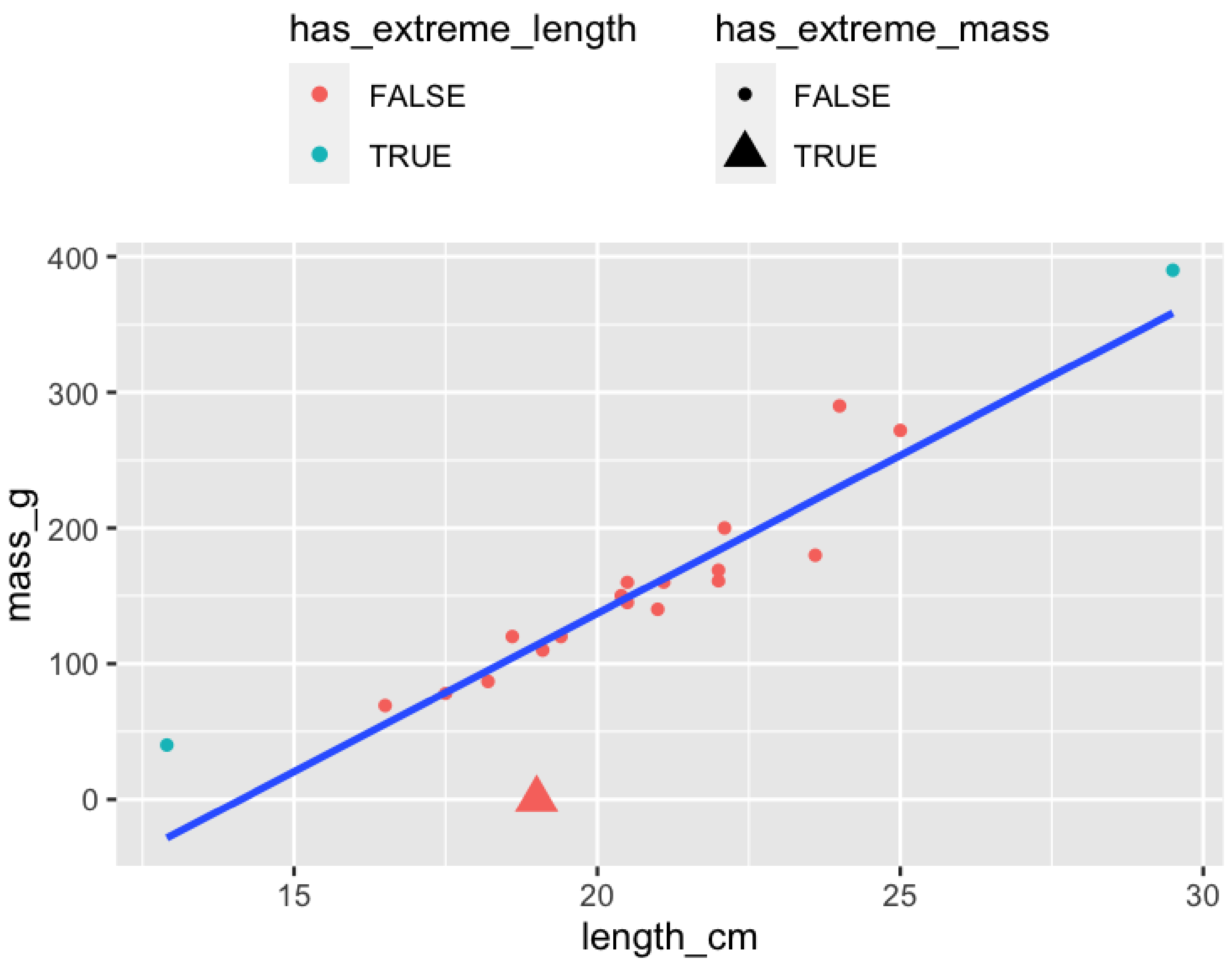
Extreme explanatory values

```
roach %>%
  mutate(
    has_extreme_length = length_cm < 15 | length_cm > 26
  ) %>%
  ggplot(aes(length_cm, mass_g)) +
  geom_point(aes(color = has_extreme_length)) +
  geom_smooth(method = "lm", se = FALSE)
```



Response values away from the regression line

```
roach %>%
  mutate(
    has_extreme_length = length_cm < 15 | length_cm > 26,
    has_extreme_mass = mass_g < 1
  ) %>%
  ggplot(aes(length_cm, mass_g)) +
  geom_point(
    aes(
      color = has_extreme_length,
      shape = has_extreme_mass
    )
  ) +
  geom_smooth(method = "lm", se = FALSE)
```



Leverage

Leverage is a measure of how extreme the explanatory variable values are.

```r
mdl_roach <- lm(mass_g ~ length_cm, data = roach)

hatvalues(mdl_roach)
```

```
     1      2      3      4      5      6      7
0.3137 0.1255 0.0935 0.0763 0.0684 0.0619 0.0605
     8      9     10     11     12     13     14
0.0568 0.0503 0.0501 0.0501 0.0506 0.0509 0.0581
    15     16     17     18     19     20
0.0581 0.0593 0.0884 0.0995 0.1334 0.3947
```

The .hat column

```r
library(broom)
augment(mdl_roach)
```

```
# A tibble: 20 x 9
   mass_g length_cm .fitted .se.fit   .resid   .hat .sigma   .cooksd .std.resid
    <dbl>     <dbl>   <dbl>   <dbl>    <dbl>  <dbl>  <dbl>     <dbl>      <dbl>
 1     40      12.9   -28.6   21.4    68.6   0.314    33.8 1.07          2.17
 2     69      16.5    55.4   13.5    13.6   0.126    39.1 0.0104        0.381
 3     78      17.5    78.7   11.7    -0.711 0.0935   39.3 0.0000197    -0.0196
 4     87      18.2    95.0   10.5    -8.03  0.0763   39.2 0.00198      -0.219
 5    120      18.6   104.     9.98   15.6   0.0684   39.1 0.00661       0.424
...
```

Highly leveraged roaches

```
mdl_roach %>%
  augment() %>%
  select(mass_g, length_cm, leverage = .hat) %>%
  arrange(desc(leverage)) %>%
  head()
```

```
# A tibble: 6 x 3
  mass_g length_cm leverage
   <dbl>     <dbl>    <dbl>
1    390      29.5   0.395  # really long roach
2     40      12.9   0.314  # really short roach
3    272      25     0.133
4     69      16.5   0.126
5    290      24     0.0995
6     78      17.5   0.0935
```

Influence

Influence measures how much the model would change if you left the observation out of the dataset when modeling.

Cook's distance

Cook's distance is the most common measure of influence.

```
cooks.distance(mdl_roach)
```

```
       1        2        3        4        5        6 
1.07e+00 1.04e-02 1.97e-05 1.98e-03 6.61e-03 3.12e-01 
       7        8        9       10       11       12 
8.53e-04 1.99e-04 2.57e-04 2.56e-04 2.45e-03 7.95e-03 
      13       14       15       16       17       18 
1.37e-04 4.82e-03 1.15e-02 4.52e-03 6.12e-02 1.50e-01 
      19       20 
2.06e-02 3.66e-01 
```

The .cooksd column

```
library(broom)
augment(mdl_roach)
```

```
# A tibble: 20 x 9
   mass_g length_cm .fitted .se.fit   .resid   .hat .sigma   .cooksd .std.resid
    <dbl>     <dbl>   <dbl>   <dbl>    <dbl>  <dbl>  <dbl>     <dbl>      <dbl>
 1     40      12.9   -28.6   21.4   68.6   0.314    33.8 1.07          2.17
 2     69      16.5    55.4   13.5   13.6   0.126    39.1 0.0104        0.381
 3     78      17.5    78.7   11.7   -0.711 0.0935   39.3 0.0000197    -0.0196
 4     87      18.2    95.0   10.5   -8.03  0.0763   39.2 0.00198      -0.219
 5    120      18.6   104.     9.98   15.6   0.0684   39.1 0.00661       0.424
...
```

Most influential roaches

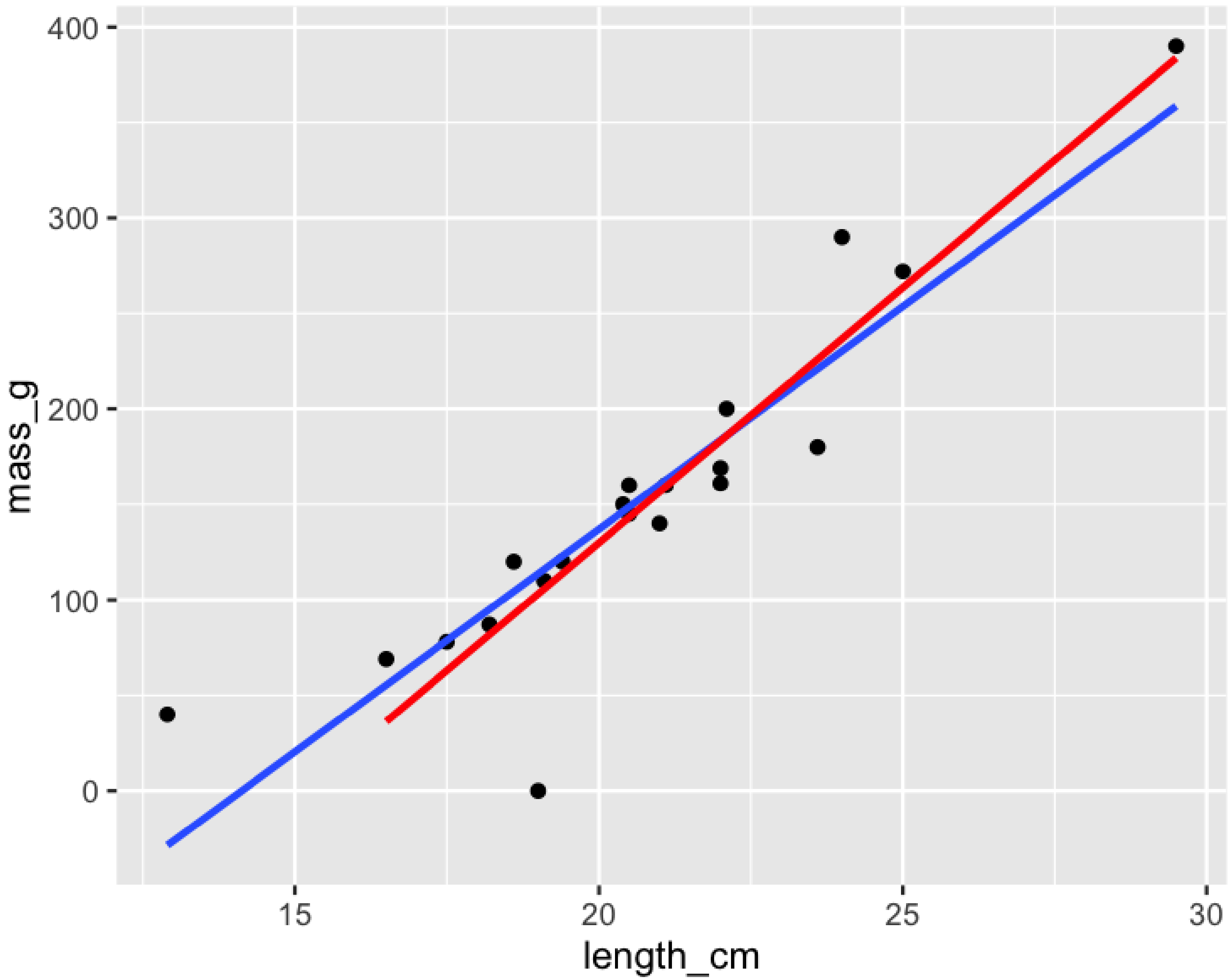
```
mdl_roach %>%
  augment() %>%
  select(mass_g, length_cm, cooks_dist = .cooksd) %>%
  arrange(desc(cooks_dist)) %>%
  head()
```

```
# A tibble: 6 x 3
  mass_g length_cm cooks_dist
   <dbl>     <dbl>      <dbl>
1     40      12.9     1.07   # really short roach
2    390      29.5     0.366  # really long roach
3      0      19       0.312  # zero mass roach
4    290      24       0.150
5    180      23.6     0.0612
6    272      25       0.0206
```

Removing the most influential roach

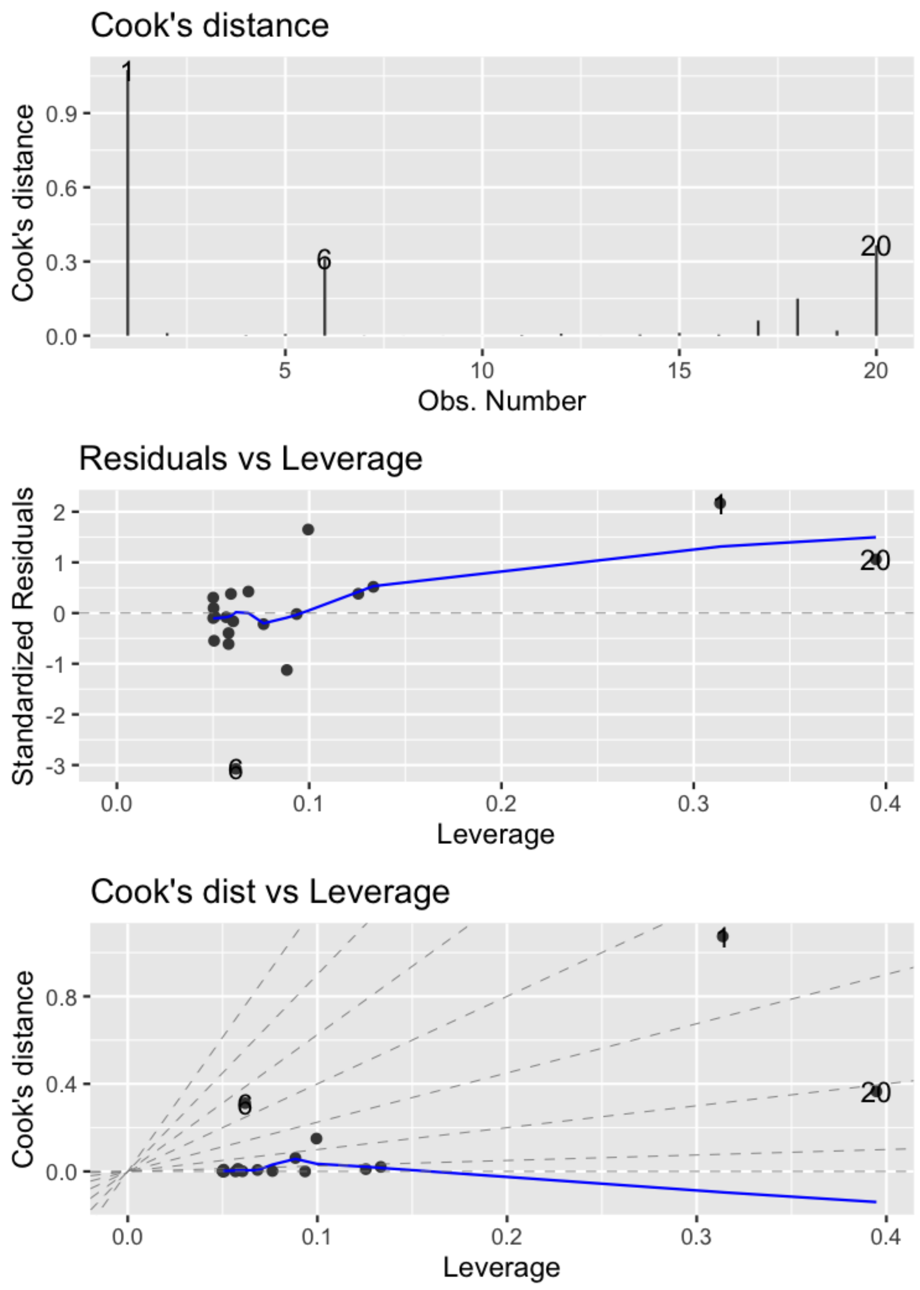
```
roach_not_short <- roach %>%
  filter(length != 12.9)
```

```
ggplot(roach, aes(length_cm, mass_g)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE) +
  geom_smooth(
    method = "lm", se = FALSE,
    data = roach_not_short, color = "red"
  )
```



autoplot()

```
autoplot(
  mdl_roach,
  which = 4:6,
  nrow = 3,
  ncol = 1
)
```



Let's practice!