CCFRP Central California Fork Length and Total Length Comparison

Rachel Brooks

2023-03-17

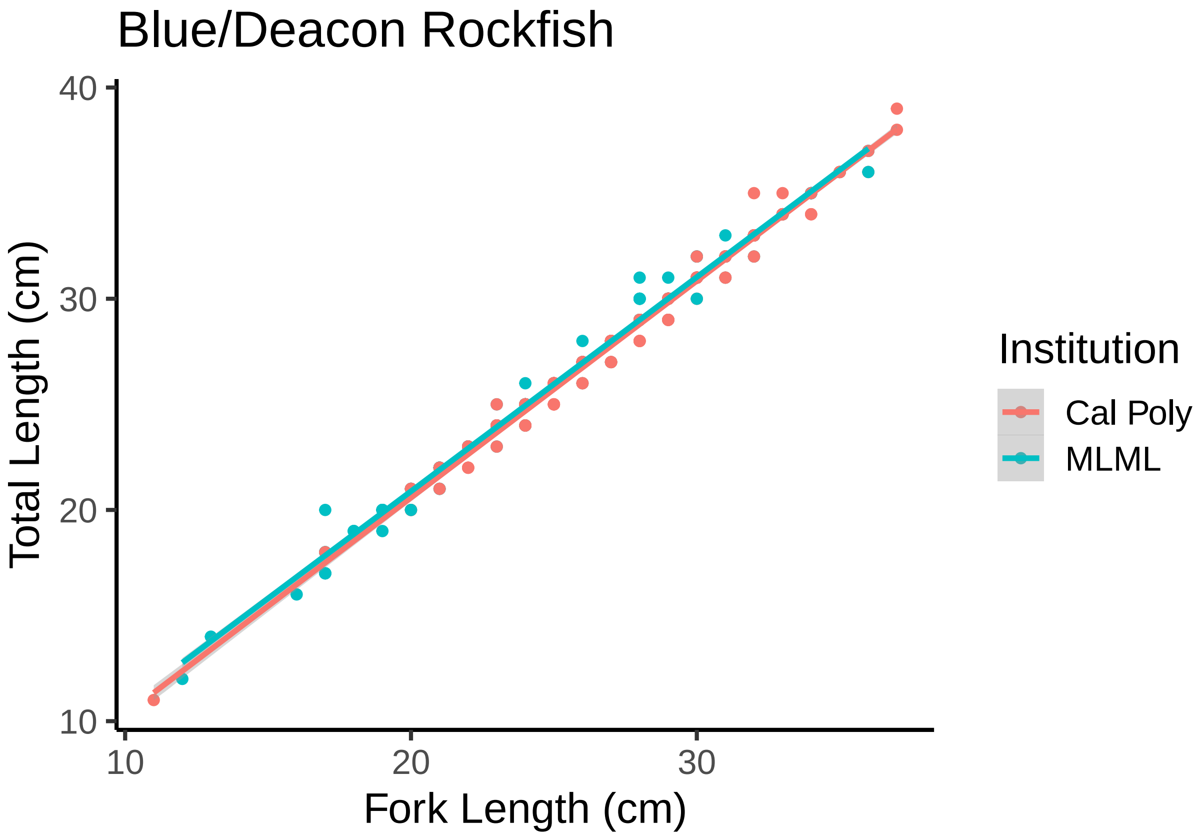
Load Data

```
length.data <-read.csv(here("Data", 'Fork-And-Total-Length-Data.csv'))
```

Blue/Deacon Rockfish

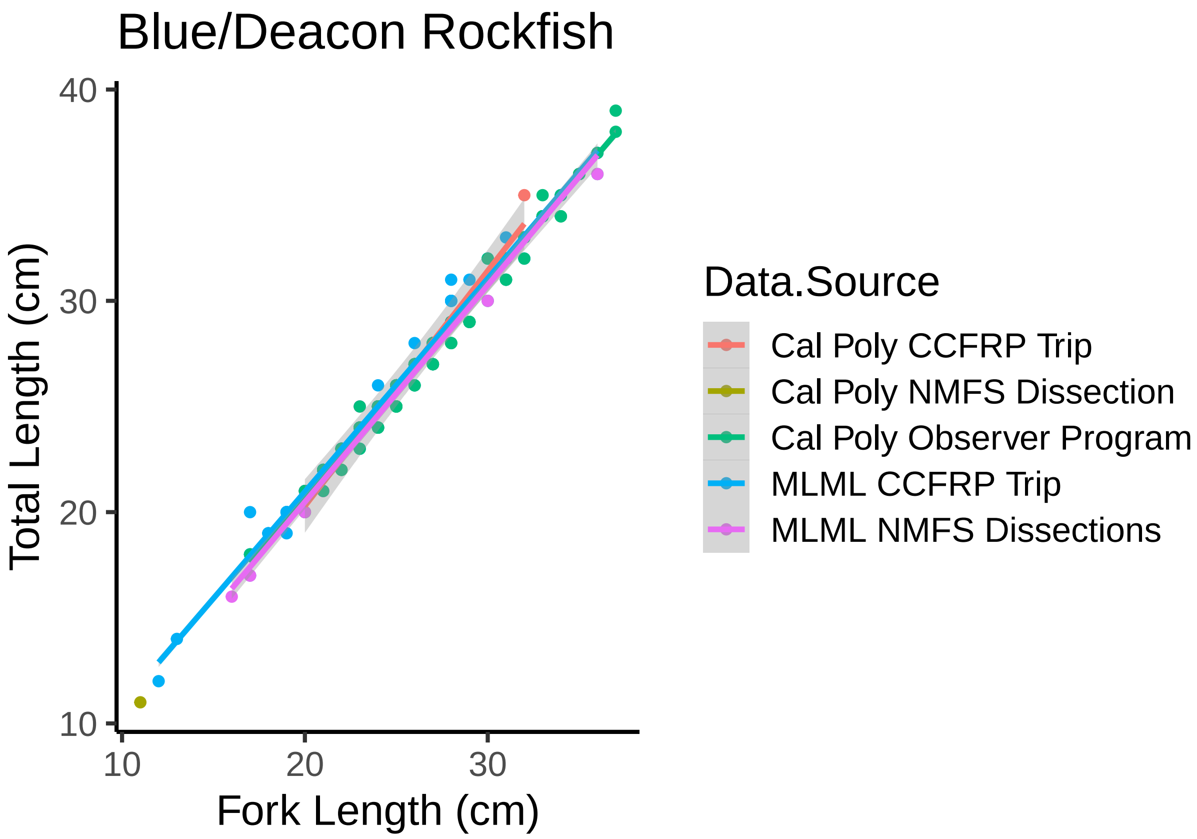
```
blue.deacon<-length.data%>%
  group_by(Institution, Species, Data.Source)%>%
  filter(Species == "Blue/Deacon Rockfish")

ggplot(blue.deacon, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Institution))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Blue/Deacon Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



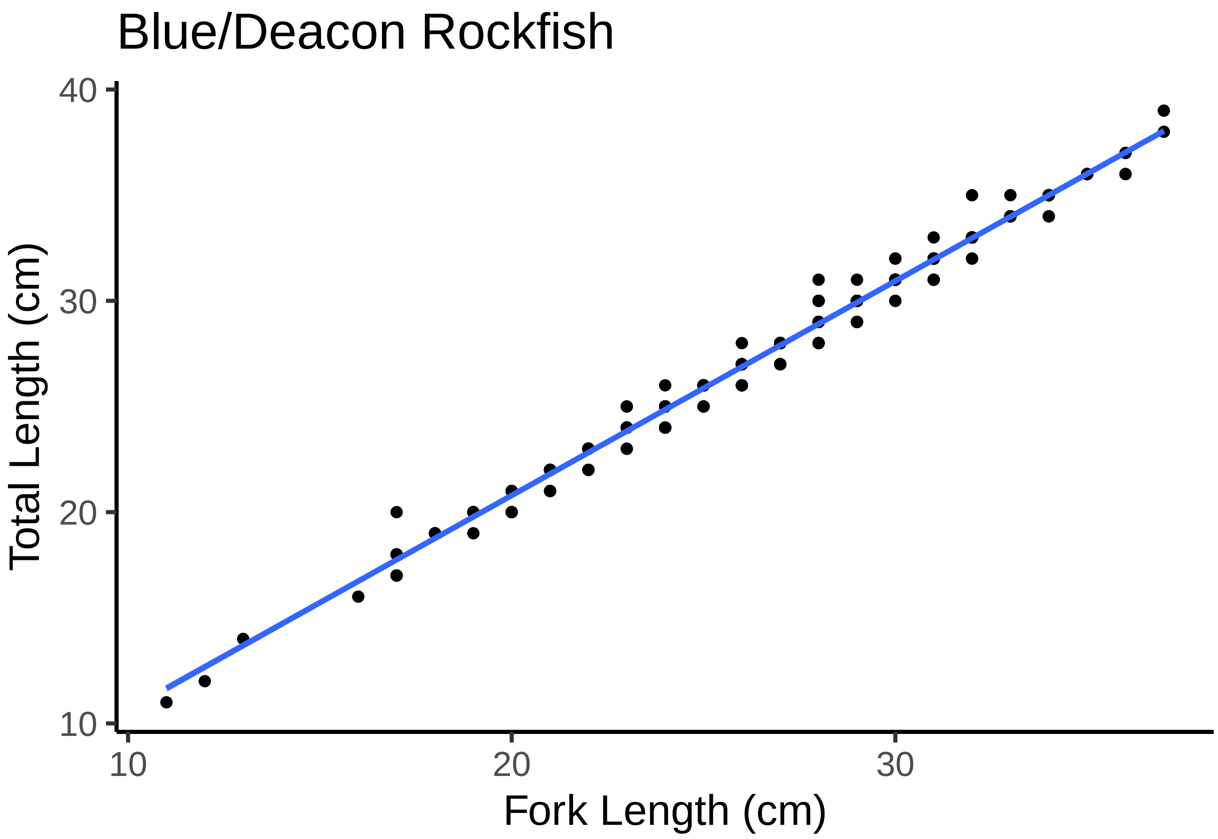
```r
ggplot(blue.deacon, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Data.Source))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Blue/Deacon Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```r
ggplot(blue.deacon, aes(x=Fork.Length..cm., y=Total.Length..cm.))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Blue/Deacon Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

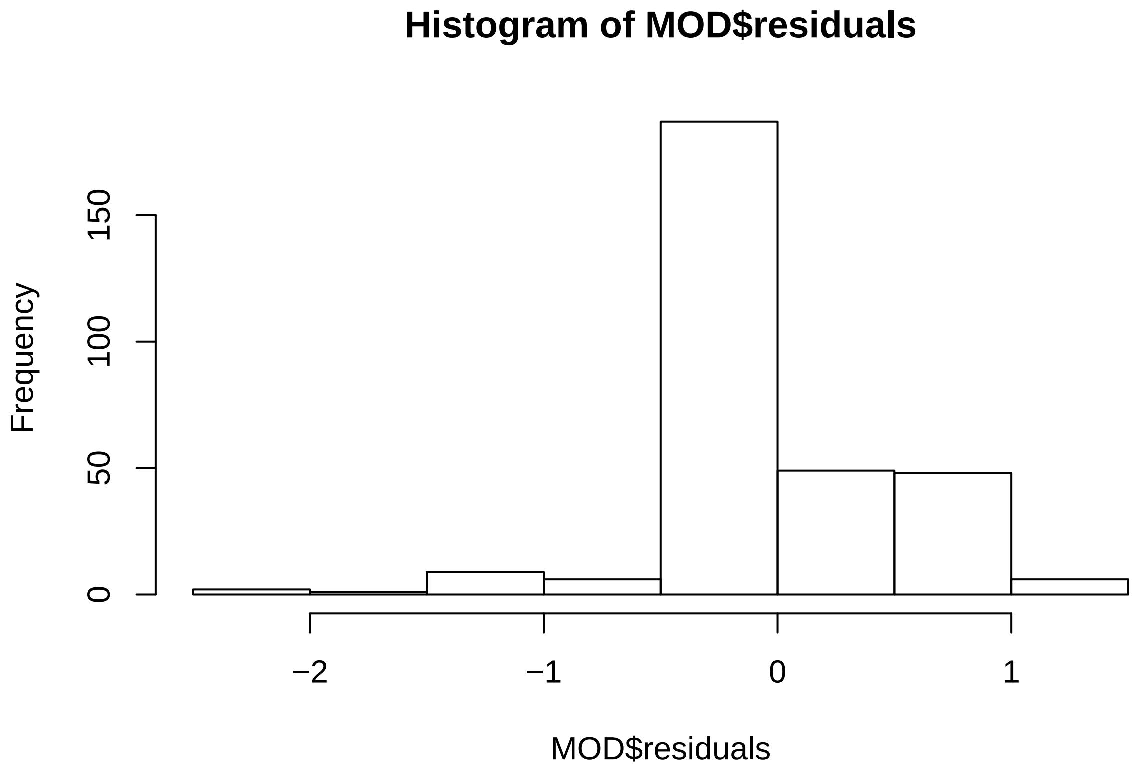
```
## `geom_smooth()` using formula 'y ~ x'
```



```
MOD <- lm(Fork.Length..cm.~Total.Length..cm., data = blue.deacon)
  summary(MOD)
```

```
##
## Call:
## lm(formula = Fork.Length..cm. ~ Total.Length..cm., data = blue.deacon)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.30306 -0.17545 -0.07336  0.03510  1.10529
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       -0.186500   0.165489  -1.127    0.261
## Total.Length..cm.  0.974478   0.005947 163.858   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4935 on 306 degrees of freedom
## Multiple R-squared:  0.9887, Adjusted R-squared:  0.9887
## F-statistic: 2.685e+04 on 1 and 306 DF,  p-value: < 2.2e-16
```

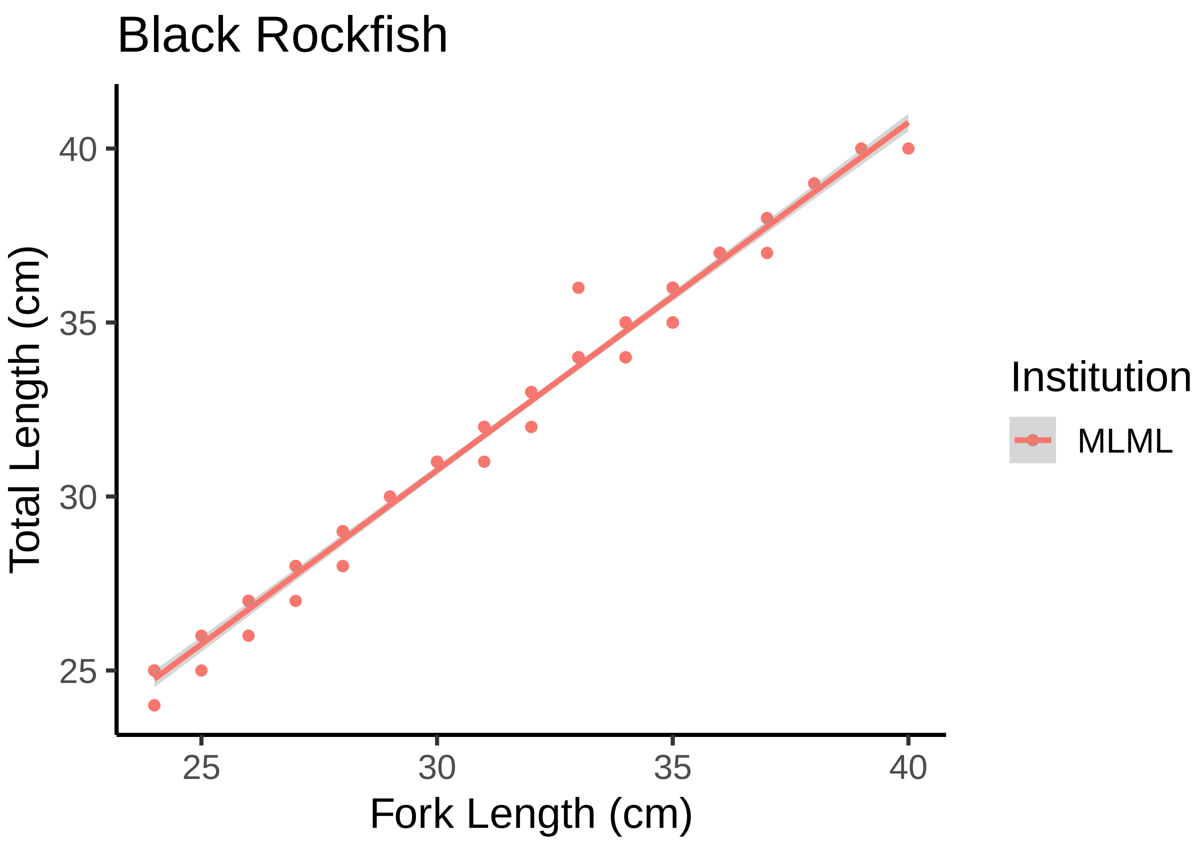
```
  hist(MOD$residuals)
```



Black Rockfish

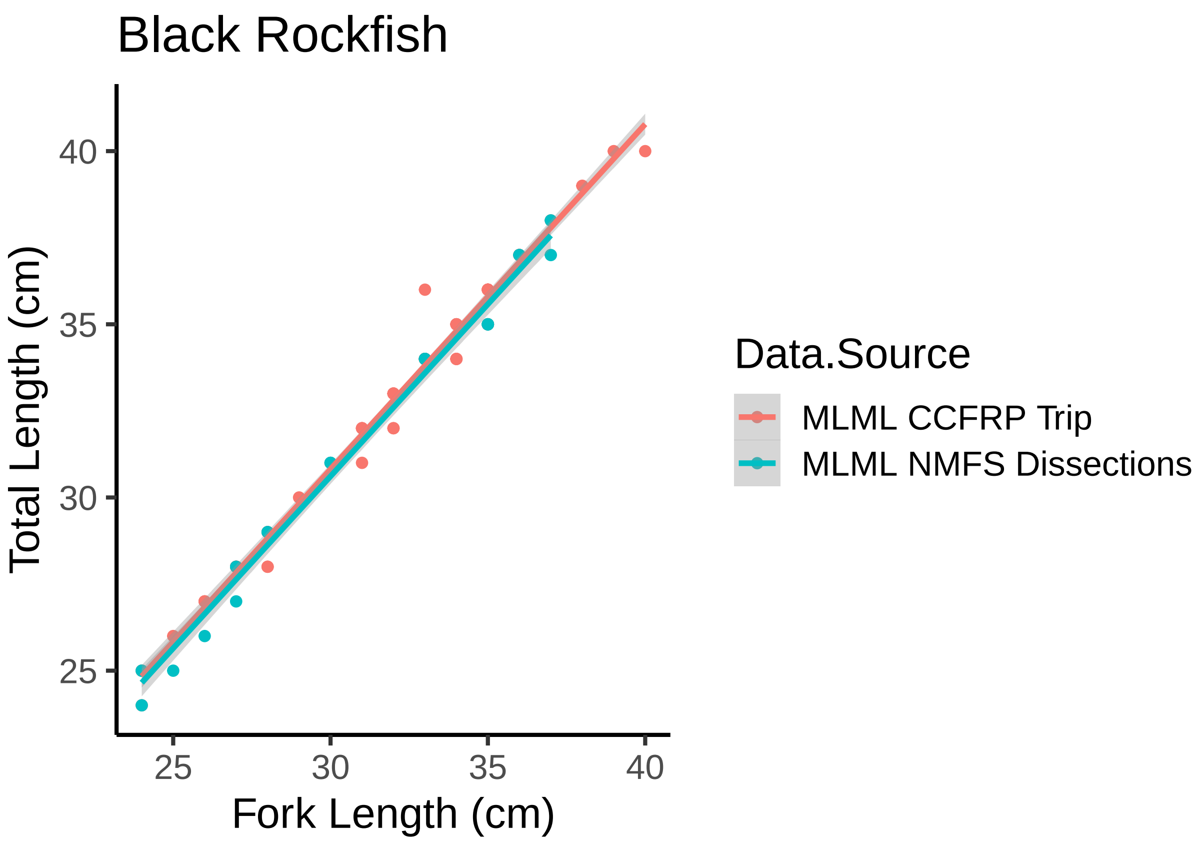
```
black<-length.data%>%
  group_by(Institution, Species, Data.Source)%>%
  filter(Species == "Black Rockfish")

ggplot(black, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Institution))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Black Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



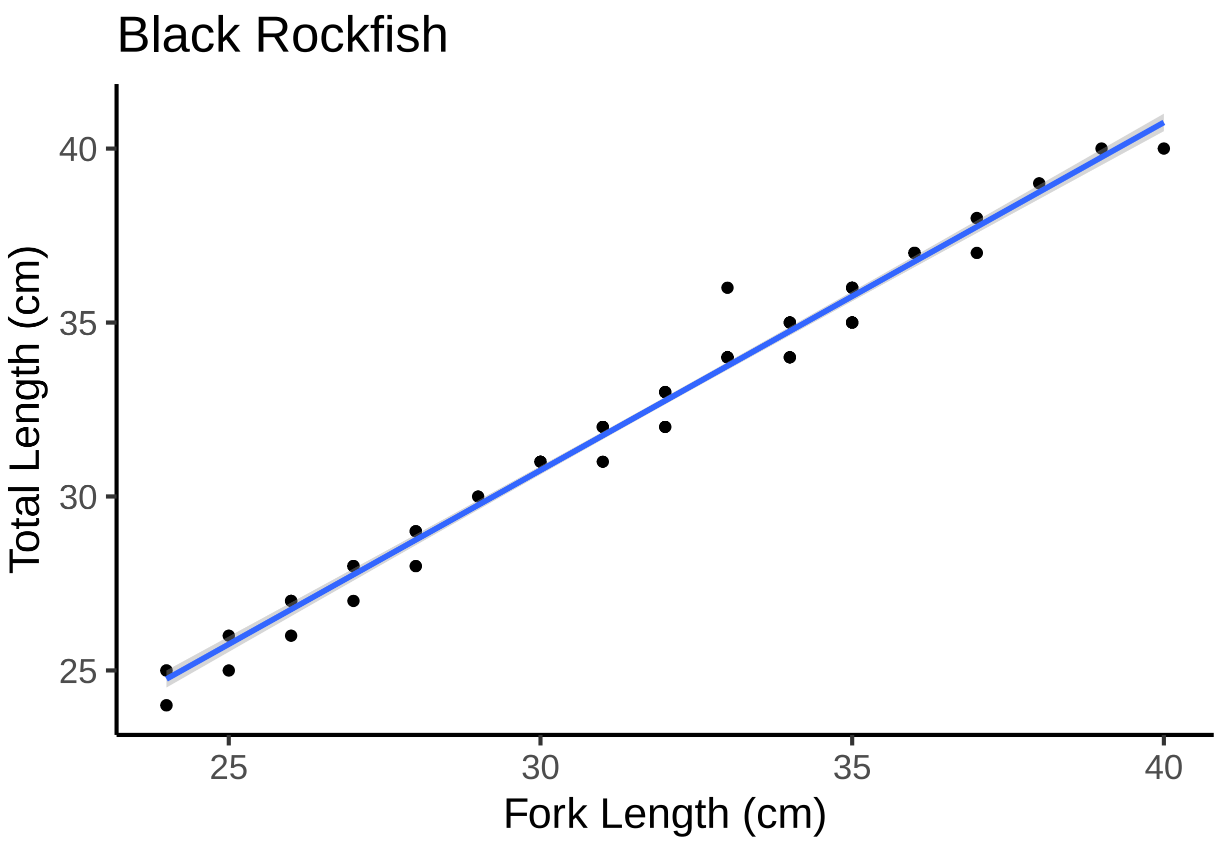
```
ggplot(black, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Data.Source))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Black Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
ggplot(black, aes(x=Fork.Length..cm., y=Total.Length..cm.))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Black Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

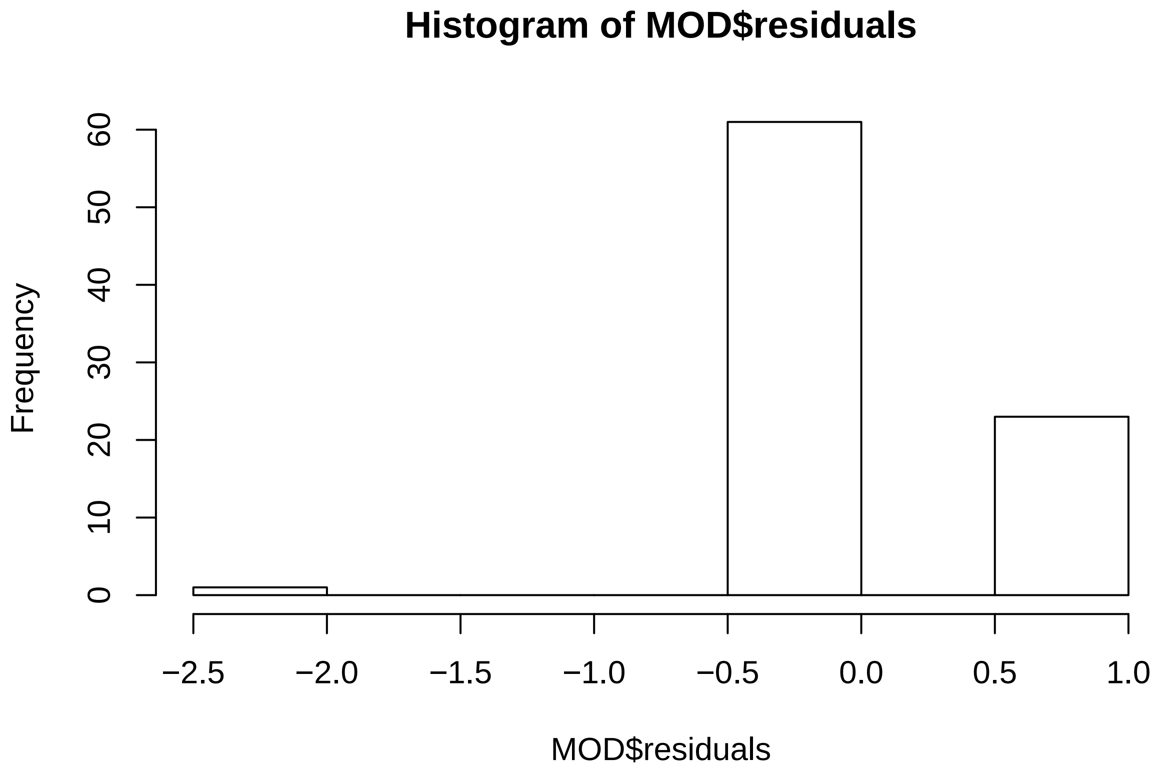
```
## `geom_smooth()` using formula 'y ~ x'
```



```
MOD <- lm(Fork.Length..cm.~Total.Length..cm., data = black)
  summary(MOD)
```

```
##
## Call:
## lm(formula = Fork.Length..cm. ~ Total.Length..cm., data = black)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.1942 -0.2570 -0.2099  0.6174  0.8685
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        -0.2407     0.4536  -0.531    0.597
## Total.Length..cm.   0.9843     0.0138  71.352   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5087 on 83 degrees of freedom
## Multiple R-squared:  0.984,   Adjusted R-squared:  0.9838
## F-statistic:  5091 on 1 and 83 DF,  p-value: < 2.2e-16
```

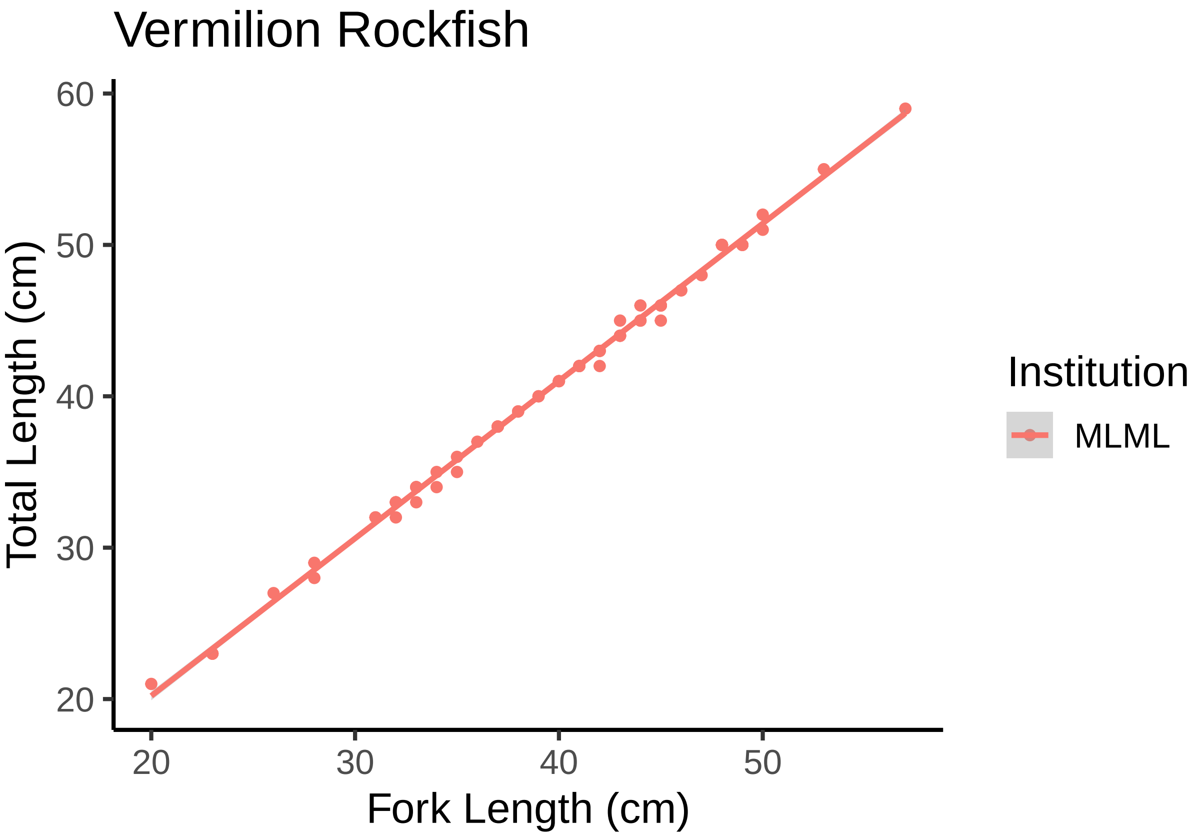
```
  hist(MOD$residuals)
```



Vermilion Rockfish

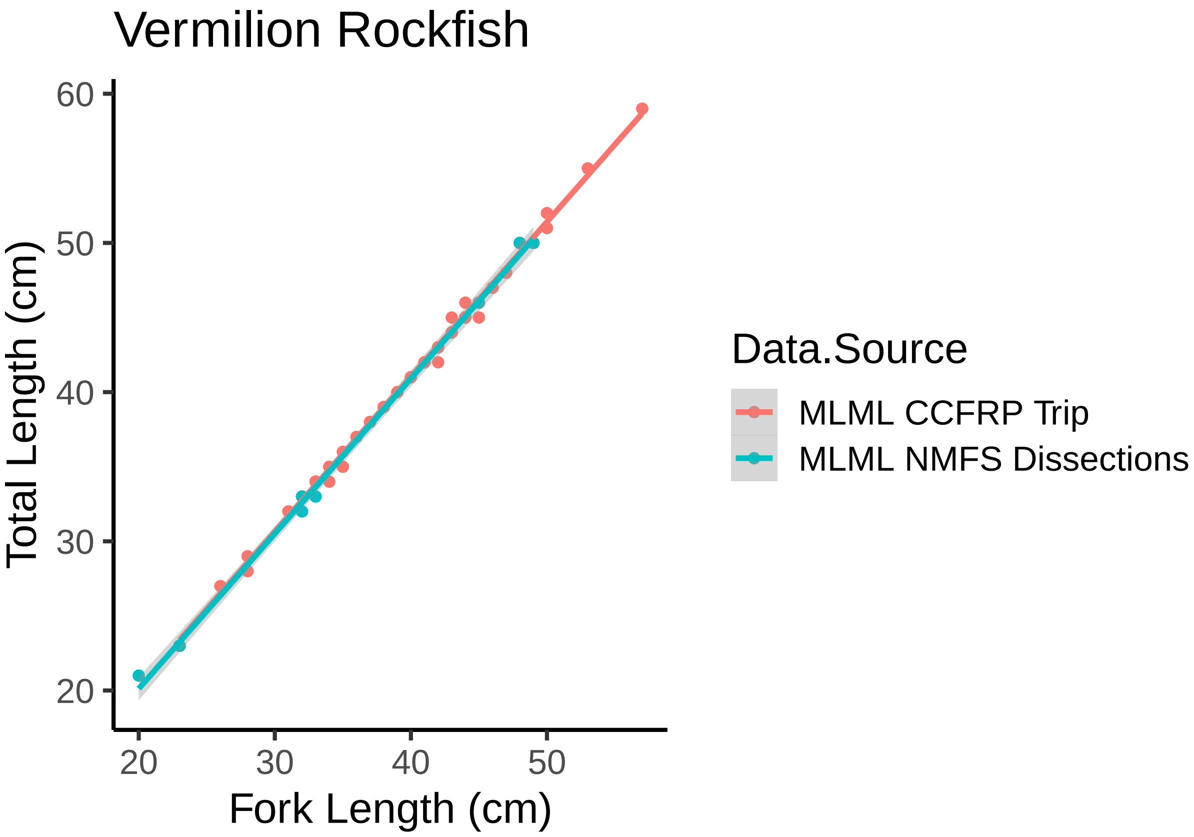
```
vermilion<-length.data%>%
  group_by(Institution, Species, Data.Source)%>%
  filter(Species == "Vermilion Rockfish")

ggplot(vermilion, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Institution))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Vermilion Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



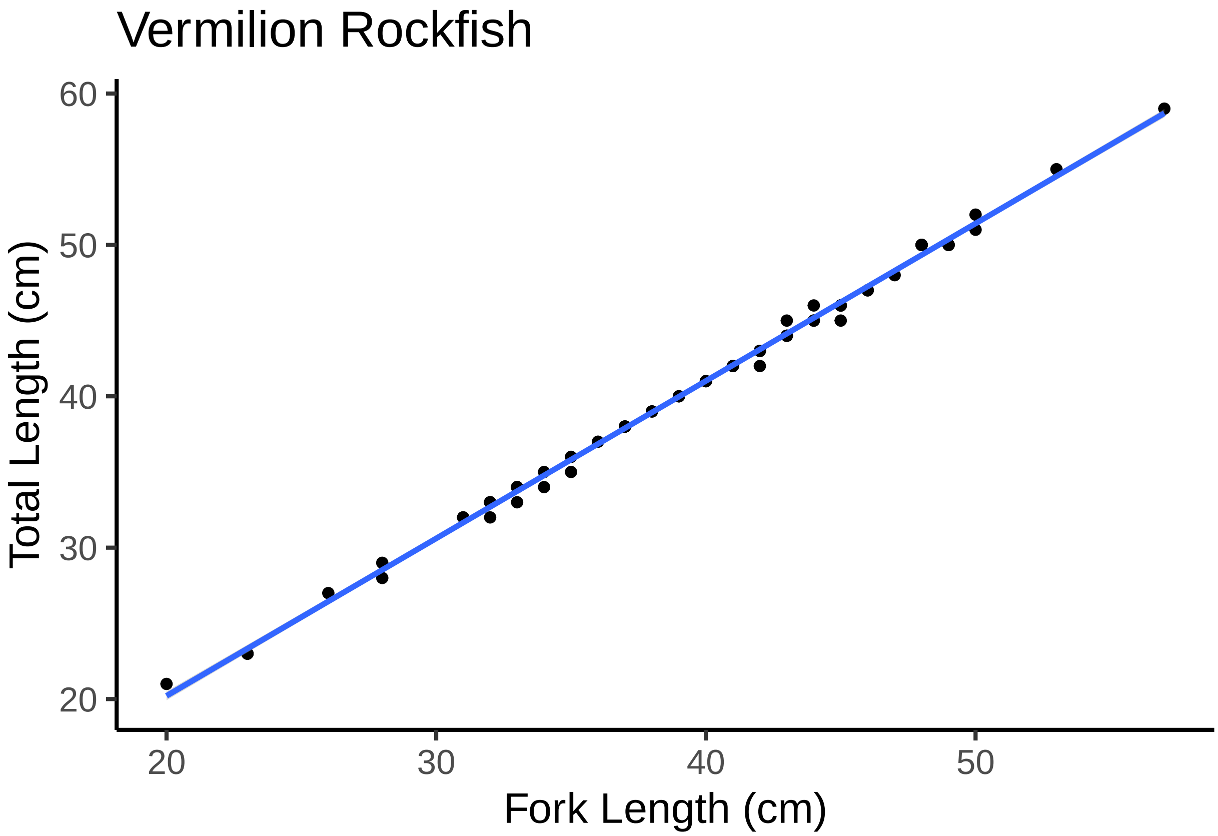
```
ggplot(vermilion, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Data.Source))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Vermilion Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
ggplot(vermilion, aes(x=Fork.Length..cm., y=Total.Length..cm.))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Vermilion Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

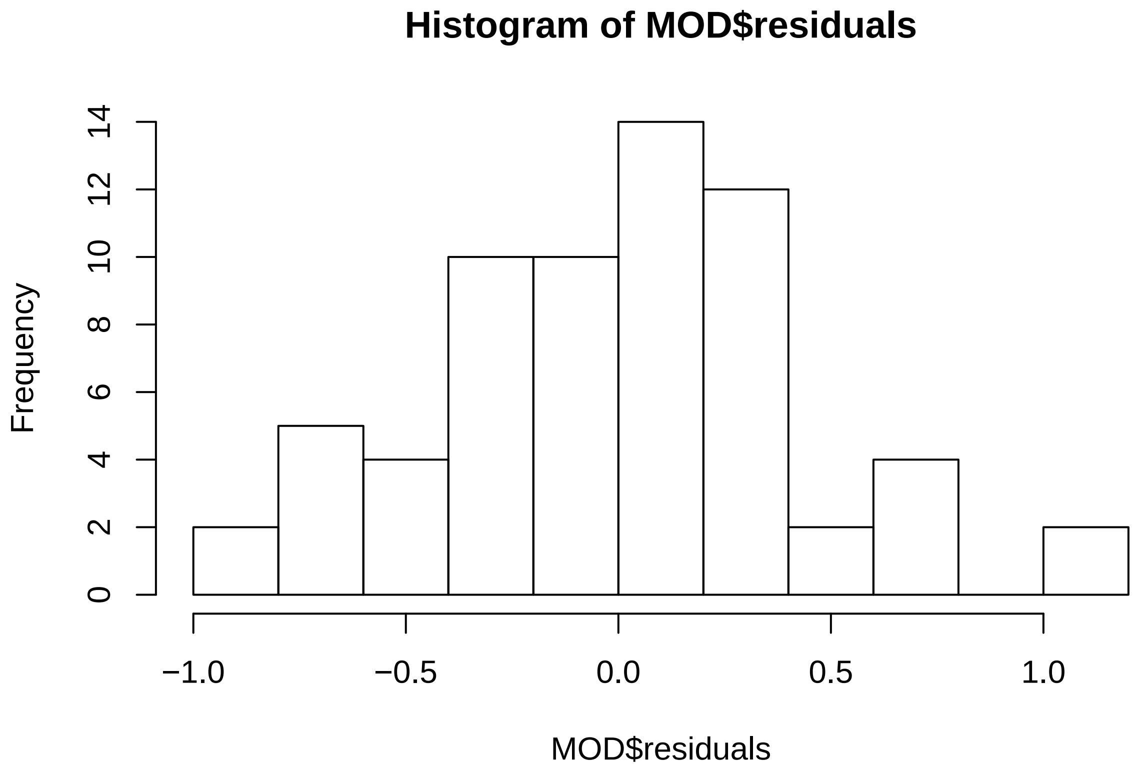
```
## `geom_smooth()` using formula 'y ~ x'
```



```
MOD <- lm(Fork.Length..cm.~Total.Length..cm., data = vermilion)
  summary(MOD)
```

```
##
## Call:
## lm(formula = Fork.Length..cm. ~ Total.Length..cm., data = vermilion)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.81878 -0.27371  0.01579  0.26394  1.18122
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.67987    0.26734   2.543   0.0135 *
## Total.Length..cm.  0.95864    0.00651 147.259   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4286 on 63 degrees of freedom
## Multiple R-squared:  0.9971, Adjusted R-squared:  0.9971
## F-statistic: 2.169e+04 on 1 and 63 DF,  p-value: < 2.2e-16
```

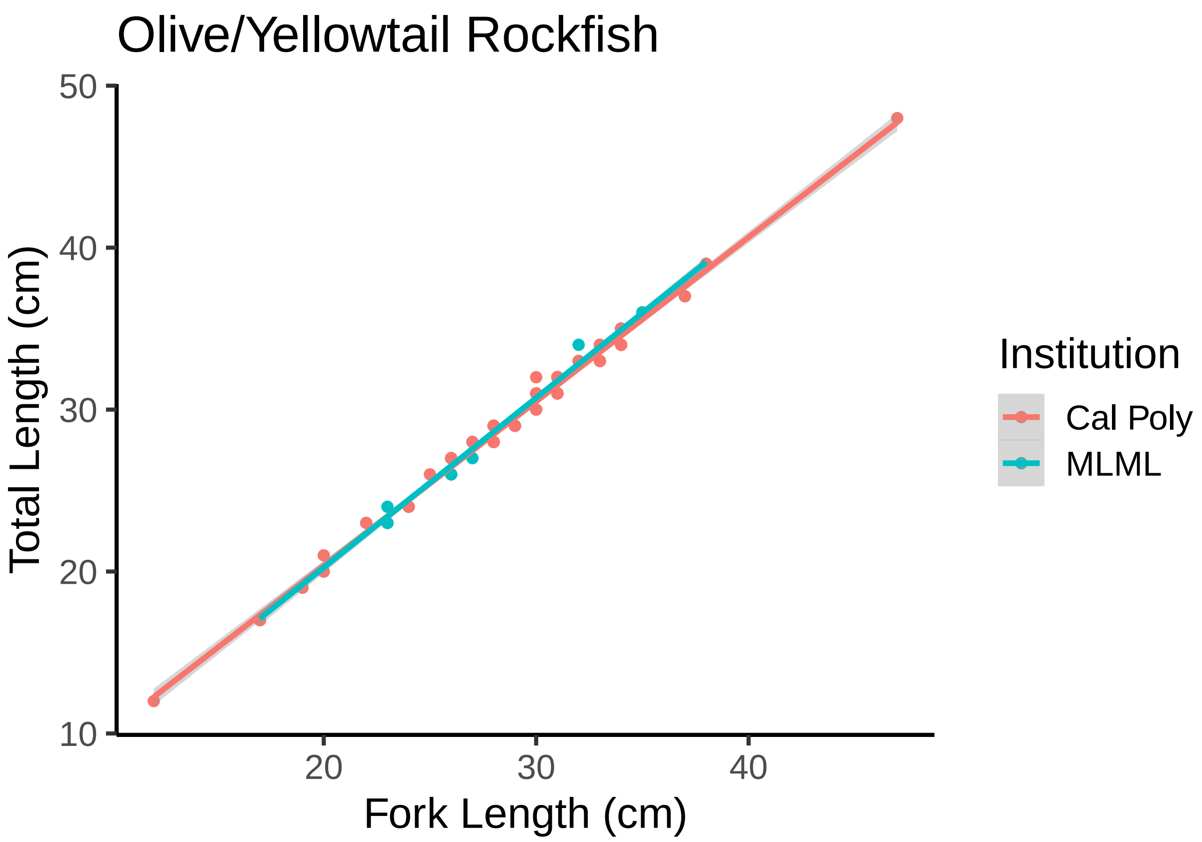
```
  hist(MOD$residuals)
```



Olive/Yellowtail Rockfish

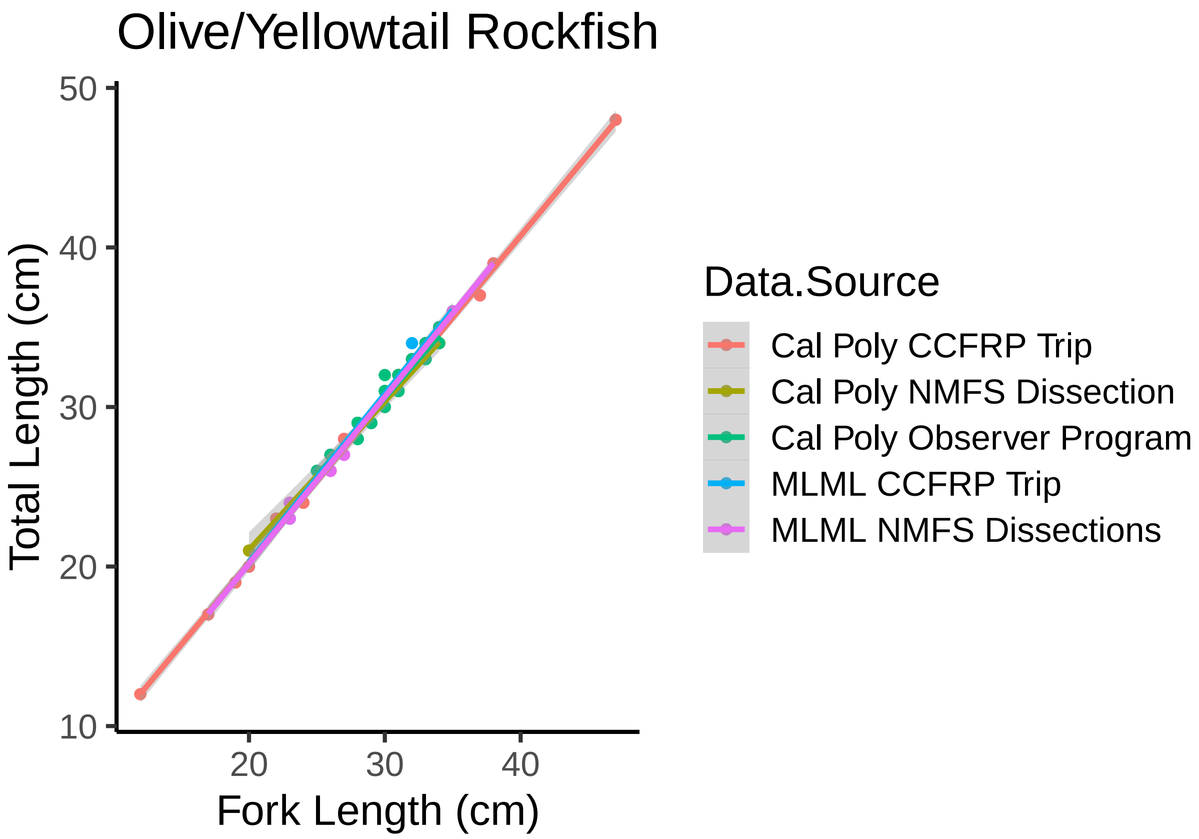
```
olive.yellowtail<-length.data%>%
  group_by(Institution, Species, Data.Source)%>%
  filter(Species == "Olive/Yellowtail Rockfish")

ggplot(olive.yellowtail, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Institution))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Olive/Yellowtail Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



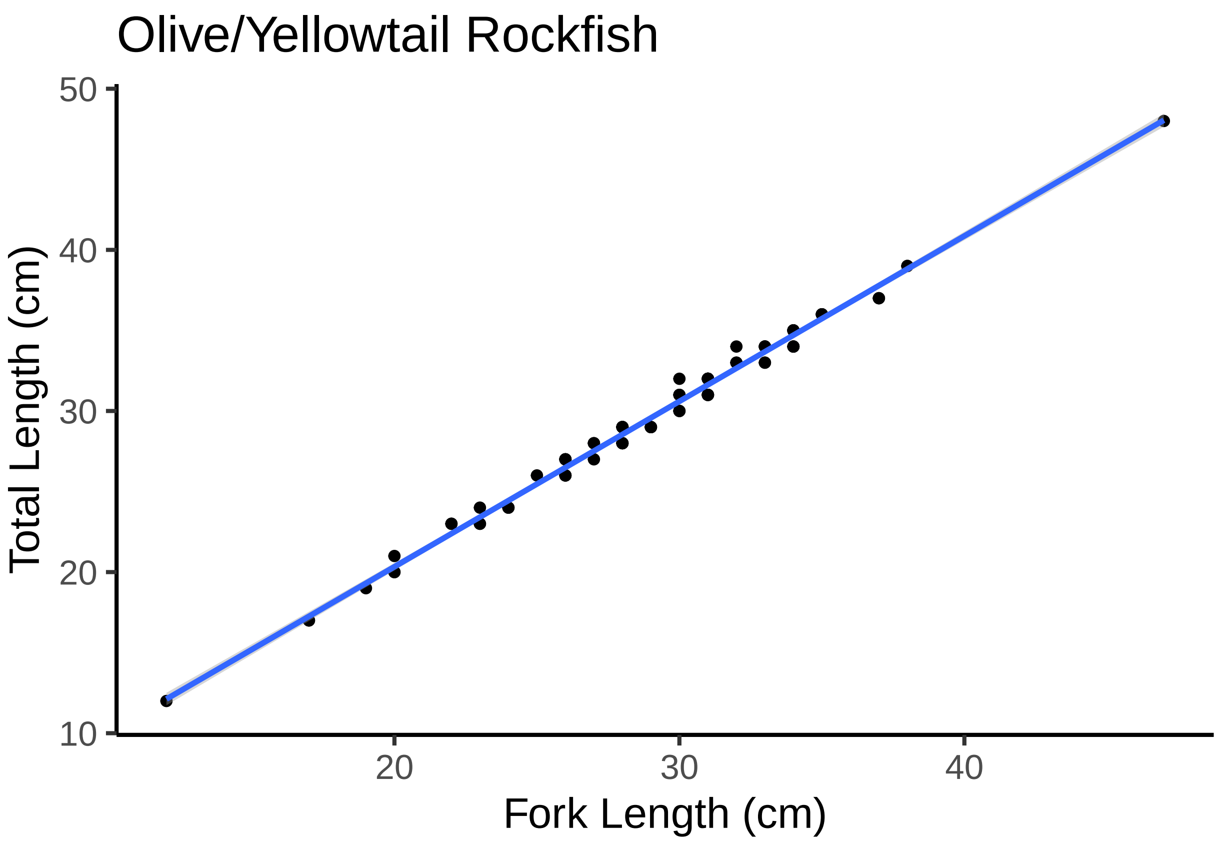
```
ggplot(olive.yellowtail, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Data.Source))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Olive/Yellowtail Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```
ggplot(olive.yellowtail, aes(x=Fork.Length..cm., y=Total.Length..cm.))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Olive/Yellowtail Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

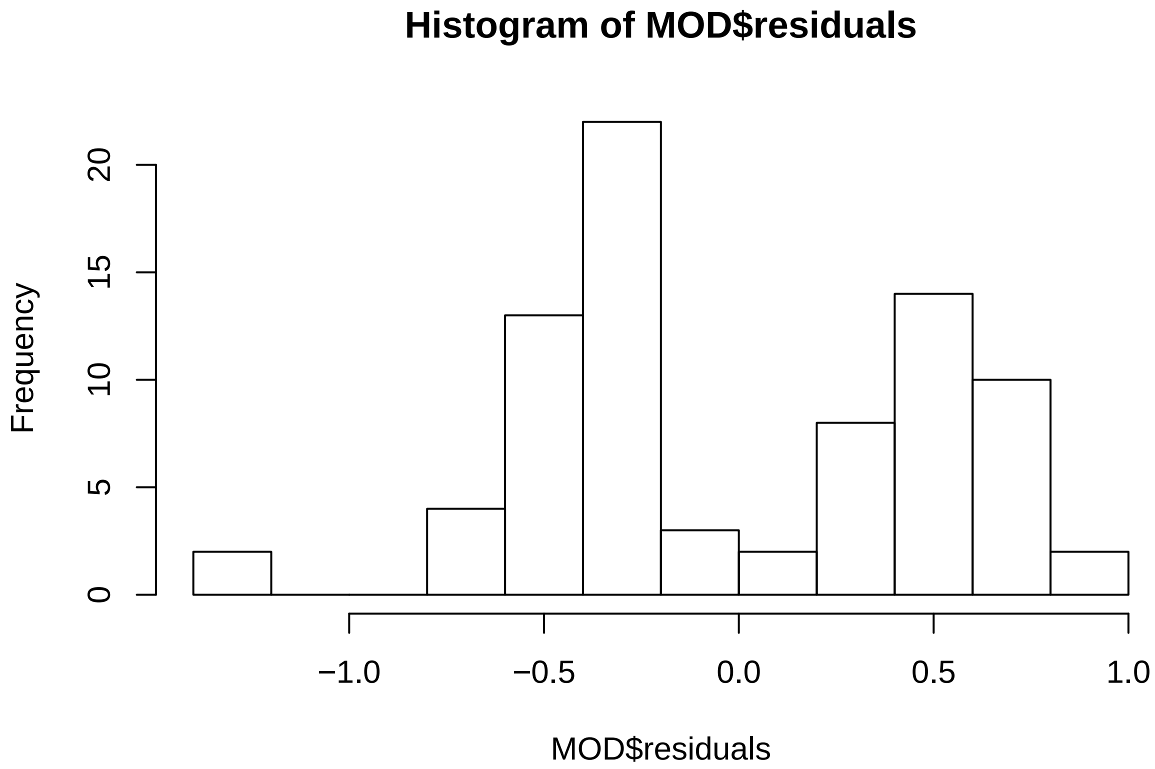
```
## `geom_smooth()` using formula 'y ~ x'
```



```
MOD <- lm(Fork.Length..cm.~Total.Length..cm., data = olive.yellowtail)
#MOD2 <- lm(Total.Length..cm.~Fork.Length..cm., data = olive.yellowtail)
  summary(MOD)
```

```
##
## Call:
## lm(formula = Fork.Length..cm. ~ Total.Length..cm., data = olive.yellowtail)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.3457 -0.3800 -0.2086  0.5171  0.8257
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)        0.44289    0.30927   1.432    0.156
## Total.Length..cm.  0.96571    0.01024  94.344   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5138 on 78 degrees of freedom
## Multiple R-squared:  0.9913, Adjusted R-squared:  0.9912
## F-statistic:  8901 on 1 and 78 DF,  p-value: < 2.2e-16
```

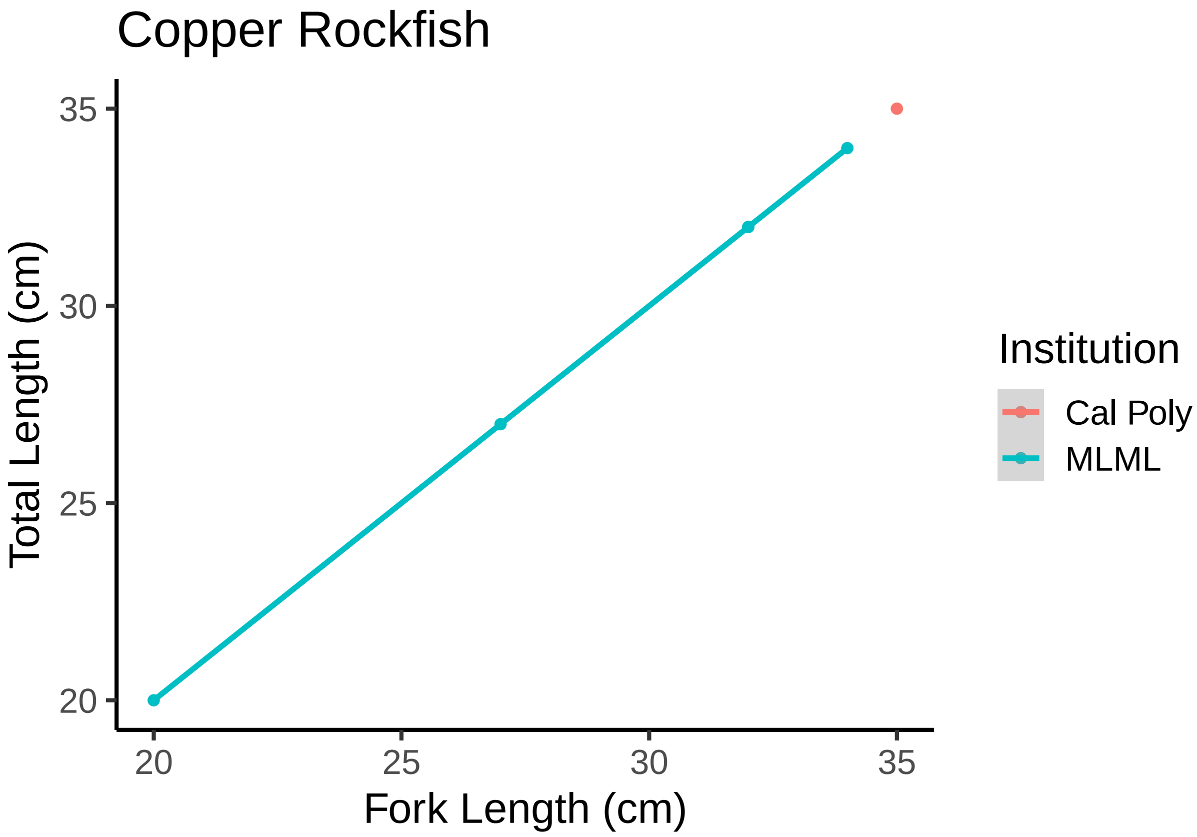
```
  #summary(MOD2)
  hist(MOD$residuals)
```



Copper Rockfish

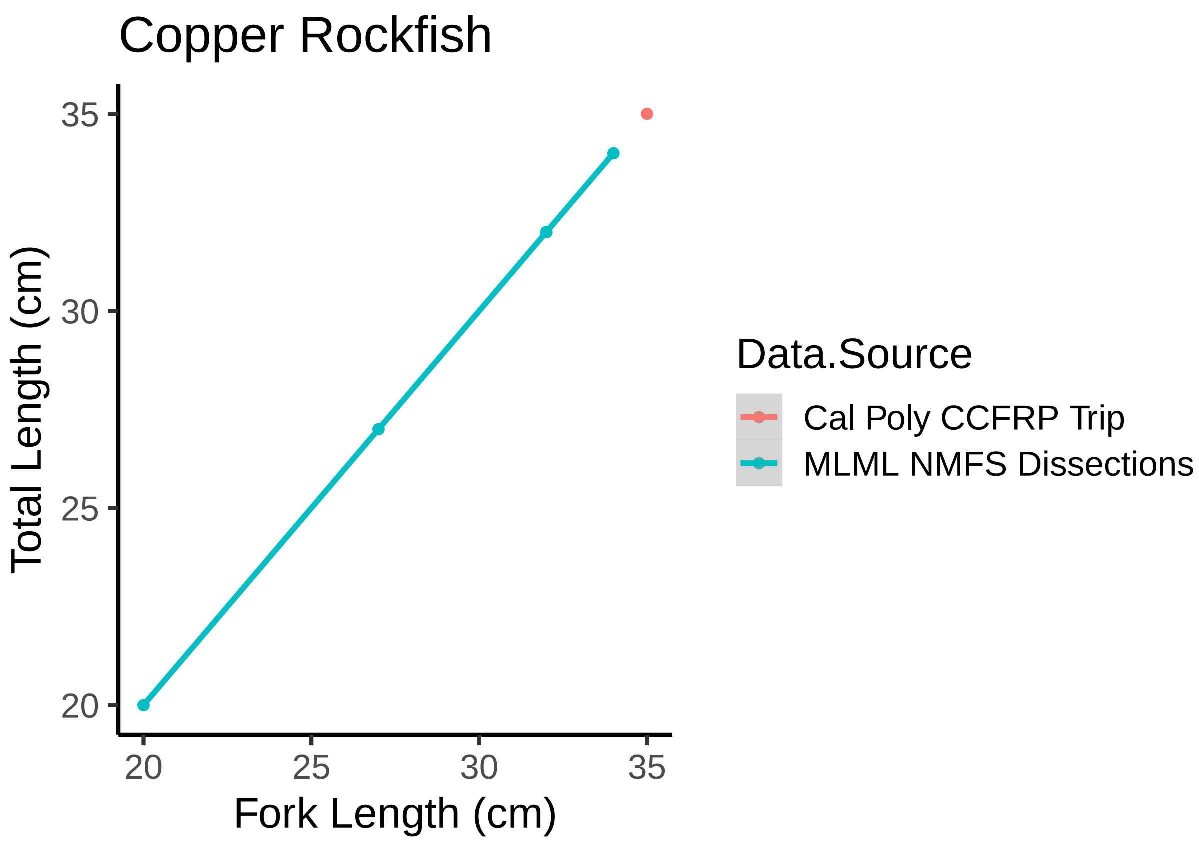
```
copper<-length.data%>%
  group_by(Institution, Species, Data.Source)%>%
  filter(Species == "Copper Rockfish")

ggplot(copper, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Institution))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Copper Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



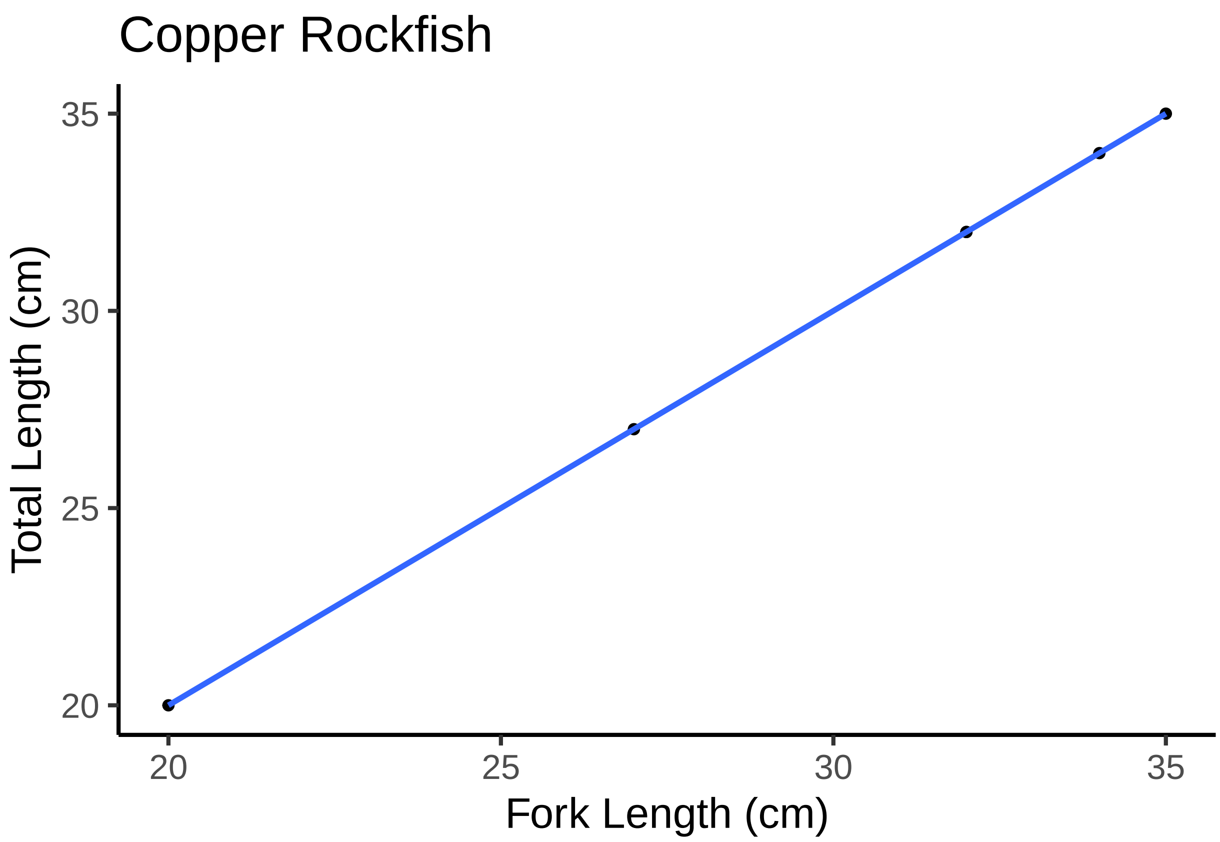
```
ggplot(copper, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Data.Source))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Copper Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```r
ggplot(copper, aes(x=Fork.Length..cm., y=Total.Length..cm.))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Copper Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```

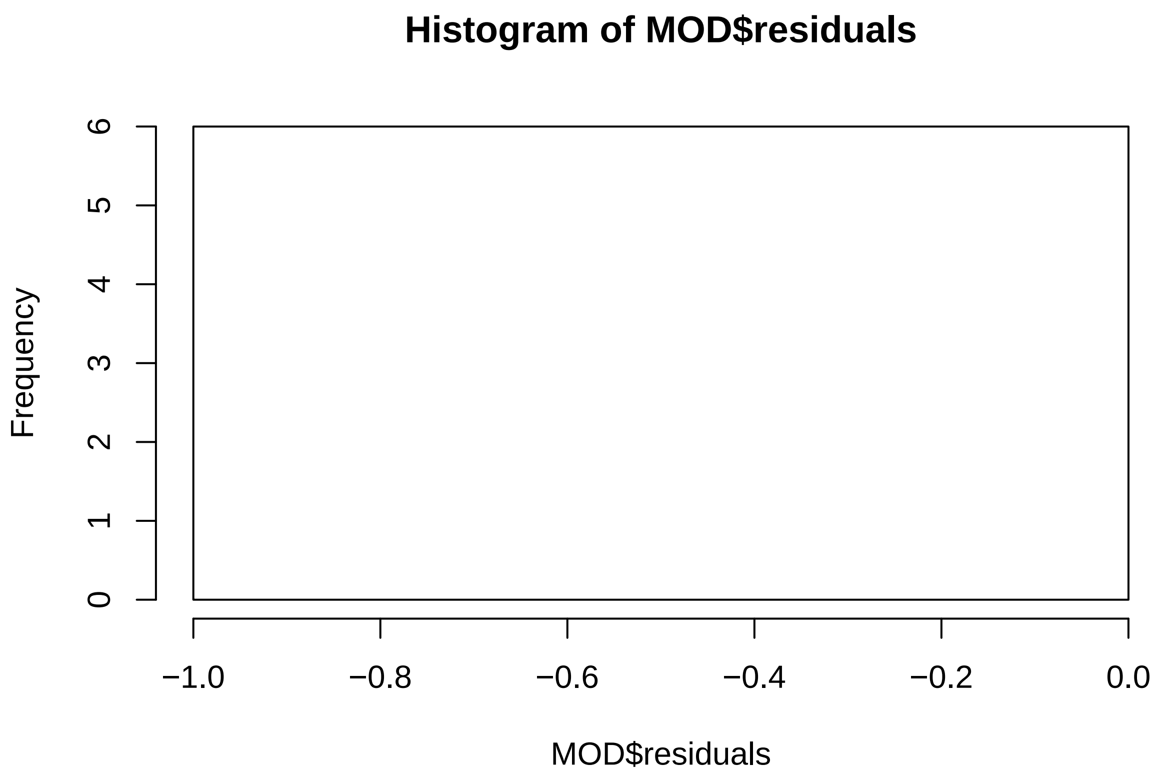


```
MOD <- lm(Fork.Length..cm.~Total.Length..cm., data = copper)
  summary(MOD)
```

```
## Warning in summary.lm(MOD): essentially perfect fit: summary may be unreliable
```

```
##
## Call:
## lm(formula = Fork.Length..cm. ~ Total.Length..cm., data = copper)
##
## Residuals:
## 1 2 3 4 5 6
## 0 0 0 0 0 0
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)              0          0      NA       NA
## Total.Length..cm.        1          0     Inf   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0 on 4 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic:   Inf on 1 and 4 DF,  p-value: < 2.2e-16
```

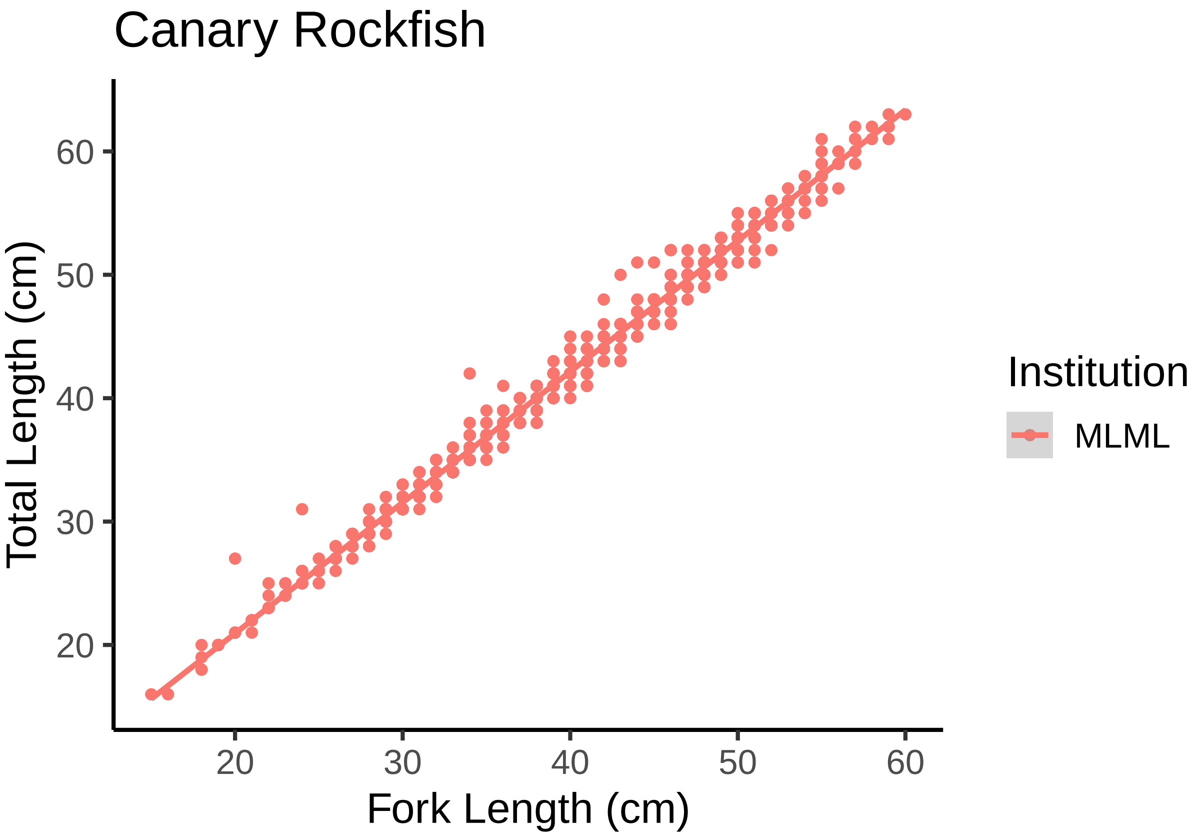
```
hist(MOD$residuals)
```



Canary Rockfish

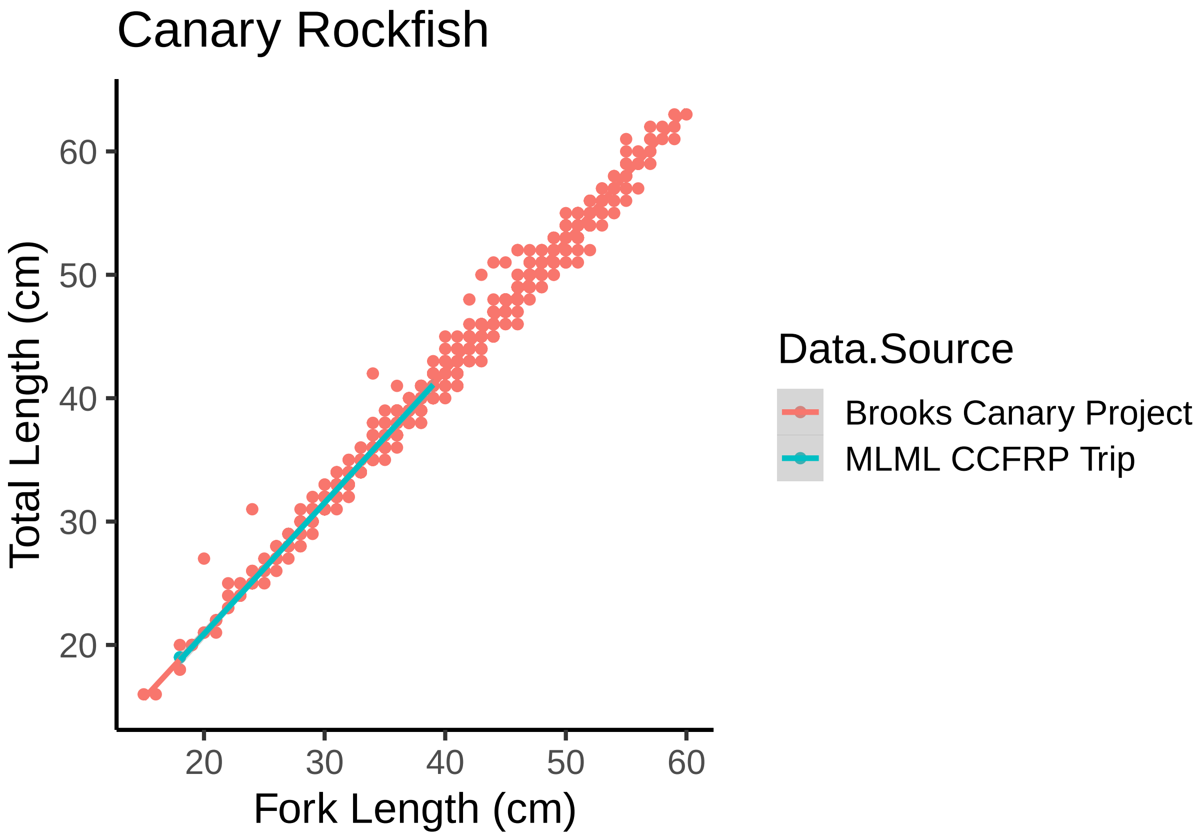
```
canary<-length.data%>%
  group_by(Institution, Species, Data.Source)%>%
  filter(Species == "Canary Rockfish")

ggplot(canary, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Institution))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Canary Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



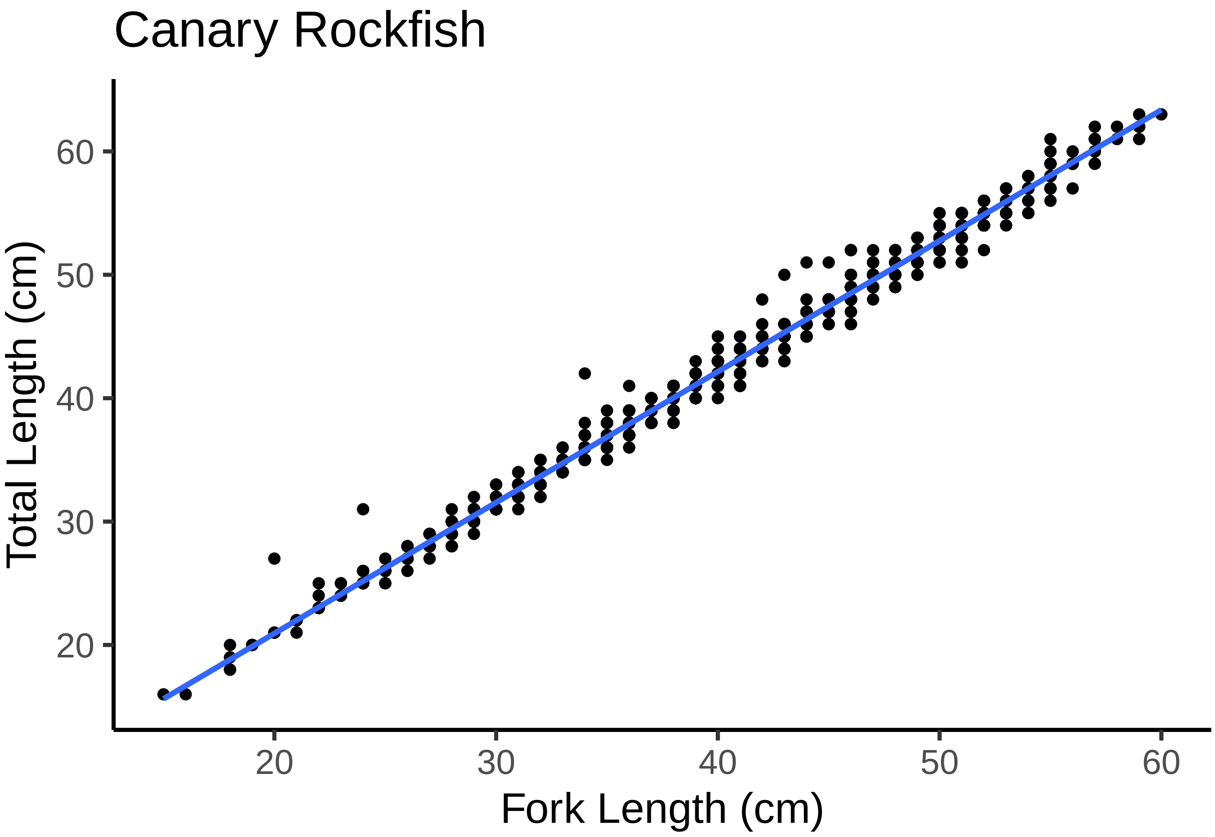
```
ggplot(canary, aes(x=Fork.Length..cm., y=Total.Length..cm., col=Data.Source))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Canary Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



```r
ggplot(canary, aes(x=Fork.Length..cm., y=Total.Length..cm.))+
  geom_point()+
  geom_smooth(method = "lm")+
  ggtitle("Canary Rockfish") +
  xlab("Fork Length (cm)") + ylab("Total Length (cm)")
```

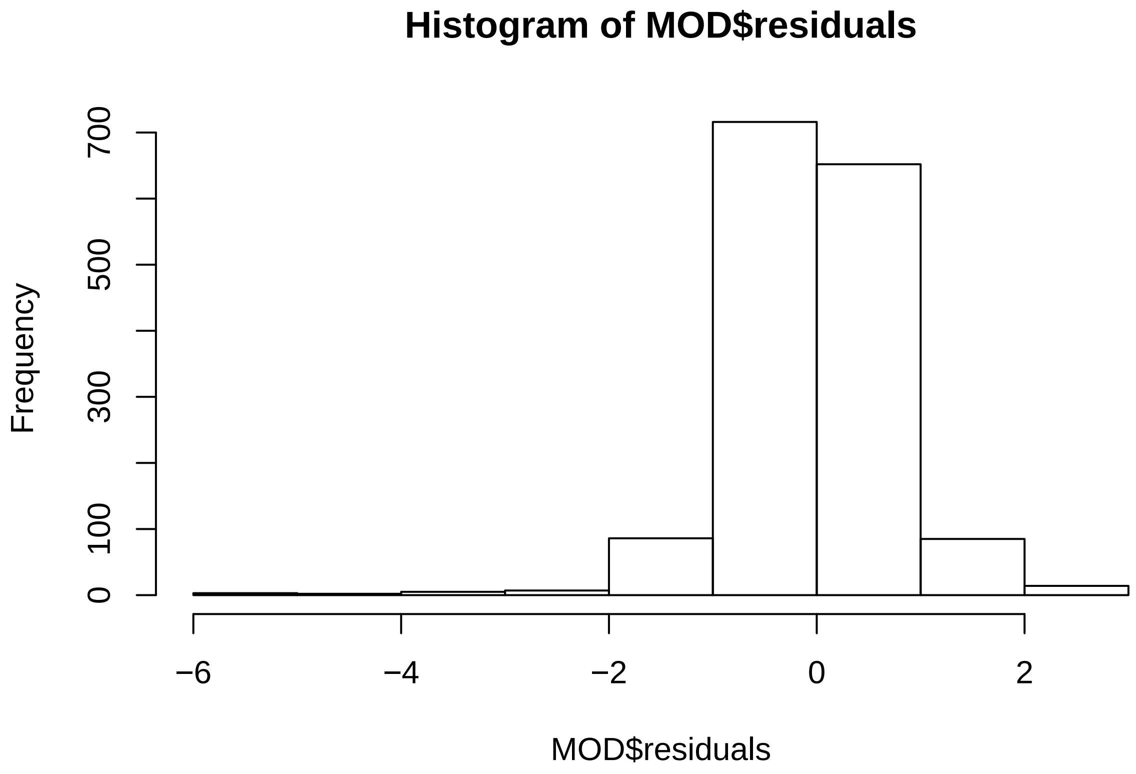
```
## `geom_smooth()` using formula 'y ~ x'
```



```
MOD <- lm(Fork.Length..cm.~Total.Length..cm., data = canary)
  summary(MOD)
```

```
##
## Call:
## lm(formula = Fork.Length..cm. ~ Total.Length..cm., data = canary)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -5.8555 -0.3691 -0.0208  0.4383  2.7866
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)       0.552121   0.086977   6.348 2.85e-10 ***
## Total.Length..cm. 0.935794   0.002048 456.853  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.7459 on 1568 degrees of freedom
## Multiple R-squared:  0.9925, Adjusted R-squared:  0.9925
## F-statistic: 2.087e+05 on 1 and 1568 DF,  p-value: < 2.2e-16
```

```
  hist(MOD$residuals)
```



Binning Addition

E. Johnston 2023-03-19

Same comparisons, but binned in 2 cm length bins, similar to what M. Monk says happens for stock assessment purposes.

```
length_dat <- read_csv(here("Data", "Fork-And-Total-Length-Data2.csv"))
```

```
## Parsed with column specification:
## cols(
##   Institution = col_character(),
##   Source = col_character(),
##   Species = col_character(),
##   TL = col_double(),
##   FL = col_double(),
##   Difference = col_double()
## )
```

```
bin_dat <- length_dat %>%
  mutate(TL_bin = cut(TL, breaks = c(0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,
                                     32,34,36,38,40,42,44,46,48,50,52,54,56,58,60,
                                     62,64)),
         FL_bin = cut(FL, breaks = c(0,2,4,6,8,10,12,14,16,18,20,22,24,26,28,30,
                                     32,34,36,38,40,42,44,46,48,50,52,54,56,58,60,
                                     62,64)),
         match = case_when(
           TL_bin == FL_bin ~ "YES",
           TL_bin != FL_bin ~ "NO"
         ))
```

```
## the annoying long bin conversion

bin_dat_num <- bin_dat %>%
  mutate(TL_int = case_when(
    TL %in% c(11,12) ~ 12,
    TL %in% c(13,14) ~ 14,
    TL %in% c(15,16) ~ 16,
    TL %in% c(17,18) ~ 18,
    TL %in% c(19,20) ~ 20,
    TL %in% c(21,22) ~ 22,
    TL %in% c(23,24) ~ 24,
    TL %in% c(25,26) ~ 26,
    TL %in% c(27,28) ~ 28,
    TL %in% c(29,30) ~ 30,
    TL %in% c(31,32) ~ 32,
    TL %in% c(33,34) ~ 34,
    TL %in% c(35,36) ~ 36,
    TL %in% c(37,38) ~ 38,
    TL %in% c(39,40) ~ 40,
    TL %in% c(41,42) ~ 42,
    TL %in% c(43,44) ~ 44,
    TL %in% c(45,46) ~ 46,
    TL %in% c(47,48) ~ 48,
    TL %in% c(49,50) ~ 50,
    TL %in% c(51,52) ~ 52,
    TL %in% c(53,54) ~ 54,
    TL %in% c(55,56) ~ 56,
    TL %in% c(57,58) ~ 58,
    TL %in% c(59,60) ~ 60,
    TL %in% c(61,62) ~ 62,
    TL %in% c(63,64) ~ 64),
    FL_int = case_when(
      FL %in% c(11,12) ~ 12,
      FL %in% c(13,14) ~ 14,
      FL %in% c(15,16) ~ 16,
      FL %in% c(17,18) ~ 18,
      FL %in% c(19,20) ~ 20,
      FL %in% c(21,22) ~ 22,
      FL %in% c(23,24) ~ 24,
      FL %in% c(25,26) ~ 26,
      FL %in% c(27,28) ~ 28,
      FL %in% c(29,30) ~ 30,
      FL %in% c(31,32) ~ 32,
      FL %in% c(33,34) ~ 34,
      FL %in% c(35,36) ~ 36,
      FL %in% c(37,38) ~ 38,
      FL %in% c(39,40) ~ 40,
      FL %in% c(41,42) ~ 42,
      FL %in% c(43,44) ~ 44,
      FL %in% c(45,46) ~ 46,
      FL %in% c(47,48) ~ 48,
      FL %in% c(49,50) ~ 50,
      FL %in% c(51,52) ~ 52,
      FL %in% c(53,54) ~ 54,
      FL %in% c(55,56) ~ 56,
      FL %in% c(57,58) ~ 58,
      FL %in% c(59,60) ~ 60,
      FL %in% c(61,62) ~ 62,
      FL %in% c(63,64) ~ 64),
    Diff_int = TL_int -FL_int)
```

```
## Blue/Deacon rockfish

BLU_DEA <- bin_dat_num %>%
  filter(Species == "Blue/Deacon Rockfish")

mod_BLU <- lm(TL_int ~ FL_int, data = BLU_DEA)
summary(mod_BLU)
```

```
##
## Call:
## lm(formula = TL_int ~ FL_int, data = BLU_DEA)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.8992 -0.8727 -0.8568  1.1273  3.1273
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.79841    0.34414    2.32    0.021 *
## FL_int       1.00265    0.01254   79.93   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.021 on 306 degrees of freedom
## Multiple R-squared:  0.9543, Adjusted R-squared:  0.9541
## F-statistic:  6388 on 1 and 306 DF,  p-value: < 2.2e-16
```

```
## Black rockfish

BLA <- bin_dat_num %>%
  filter(Species == "Black Rockfish")

mod_BLA <- lm(TL_int ~ FL_int, data = BLA)
summary(mod_BLA)
```

```
##
## Call:
```

```
## lm(formula = TL_int ~ FL_int, data = BLA)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.0885 -0.6610 -0.5755  1.0825  1.5100
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   2.1143     0.8296   2.549   0.0127 *
## FL_int        0.9573     0.0254  37.685   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.958 on 83 degrees of freedom
## Multiple R-squared:  0.9448, Adjusted R-squared:  0.9441
## F-statistic:  1420 on 1 and 83 DF,  p-value: < 2.2e-16
```

```
## Vermilion rockfish

VER <- bin_dat_num %>%
  filter(Species == "Vermilion Rockfish")

mod_VER <- lm(TL_int ~ FL_int, data = VER)
summary(mod_VER)
```

```
##
## Call:
## lm(formula = TL_int ~ FL_int, data = VER)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.1289 -0.9274 -0.6051  0.9920  1.4755
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.12161    0.63978     0.19     0.85
## FL_int       1.02015    0.01578    64.66   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9999 on 63 degrees of freedom
## Multiple R-squared:  0.9852, Adjusted R-squared:  0.9849
## F-statistic:  4181 on 1 and 63 DF,  p-value: < 2.2e-16
```

```
## Olive/Yellowtail rockfish

OYT <- bin_dat_num %>%
  filter(Species == "Olive/Yellowtail Rockfish")

mod_OYT <- lm(TL_int ~ FL_int, data = OYT)
summary(mod_OYT)
```

```
##
```

```
## Call:
## lm(formula = TL_int ~ FL_int, data = OYT)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.6636 -0.6512 -0.6466  1.3472  1.3565
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.67290    0.57627   1.168    0.246
## FL_int       0.99923    0.01915  52.173   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.9487 on 78 degrees of freedom
## Multiple R-squared:  0.9721, Adjusted R-squared:  0.9718
## F-statistic:  2722 on 1 and 78 DF,  p-value: < 2.2e-16
```

```
## Copper rockfish

CPR <- bin_dat_num %>%
  filter(Species == "Copper Rockfish")

mod_CPR <- lm(TL_int ~ FL_int, data = CPR)
summary(mod_CPR)
```

```
## Warning in summary.lm(mod_CPR): essentially perfect fit: summary may be
## unreliable
```

```
##
## Call:
## lm(formula = TL_int ~ FL_int, data = CPR)
##
## Residuals:
##          1          2          3          4          5          6
##  1.059e-15 -1.400e-15 -1.563e-17  3.410e-16  2.961e-16 -2.803e-16
##
## Coefficients:
##              Estimate Std. Error   t value Pr(>|t|)
## (Intercept) 0.000e+00  2.209e-15 0.000e+00        1
## FL_int      1.000e+00  7.177e-17 1.393e+16   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9.172e-16 on 4 degrees of freedom
## Multiple R-squared:      1,  Adjusted R-squared:      1
## F-statistic: 1.942e+32 on 1 and 4 DF,  p-value: < 2.2e-16
```

```
## Canary rockfish

CNY <- bin_dat_num %>%
  filter(Species == "Canary Rockfish")
```

```
mod_CNY <- lm(TL_int ~ FL_int, data = CNY)
summary(mod_CNY)
```

```
##
## Call:
## lm(formula = TL_int ~ FL_int, data = CNY)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0391 -0.5803 -0.0067  0.4522  7.0258
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.172980   0.127907  -1.352    0.176
## FL_int       1.057359   0.003137 337.019   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.076 on 1568 degrees of freedom
## Multiple R-squared:  0.9864, Adjusted R-squared:  0.9864
## F-statistic: 1.136e+05 on 1 and 1568 DF,  p-value: < 2.2e-16
```

```


dummy_dat <- data.frame(FL = seq(11,63, by = 1))

dummy_dat_rb <- dummy_dat %>%
  mutate(TL_blu = 1.014627*FL + 0.498233,
         TL_bla = 0.99965*FL + 0.76415,
         TL_ver = 1.040120*FL - 0.590558,
         TL_oyt = 1.02651*FL - 0.19673,
         TL_cpr = 1*FL + 0,
         TL_cny = 1.060643*FL - 0.276485)

dummy_dat_bin <- dummy_dat %>%
  mutate(TL_blu = 1.00265*FL + 0.79841,
         TL_bla = 0.9573*FL + 2.1143,
         TL_ver = 1.02015*FL + 0.12161,
         TL_oyt = 0.99923*FL + 0.67290,
         TL_cpr = 1*FL + 0,
         TL_cny = 1.057358*FL - 0.17298)
```

Theoretical Datasets This table has FL 11-63 cm (the actual size ranges in our data) and the theoretical TL calculated based on linear regression output for real size measurements rounded to nearest whole cm, as is procedure for CCFRP.

```
dummy_dat_rb
```

```
##    FL   TL_blu   TL_bla   TL_ver   TL_oyt TL_cpr   TL_cny
## 1  11 11.65913 11.76030 10.85076 11.09488     11 11.39059
## 2  12 12.67376 12.75995 11.89088 12.12139     12 12.45123
## 3  13 13.68838 13.75960 12.93100 13.14790     13 13.51187
## 4  14 14.70301 14.75925 13.97112 14.17441     14 14.57252
## 5  15 15.71764 15.75890 15.01124 15.20092     15 15.63316
## 6  16 16.73226 16.75855 16.05136 16.22743     16 16.69380
## 7  17 17.74689 17.75820 17.09148 17.25394     17 17.75445
## 8  18 18.76152 18.75785 18.13160 18.28045     18 18.81509
## 9  19 19.77615 19.75750 19.17172 19.30696     19 19.87573
## 10 20 20.79077 20.75715 20.21184 20.33347     20 20.93637
## 11 21 21.80540 21.75680 21.25196 21.35998     21 21.99702
## 12 22 22.82003 22.75645 22.29208 22.38649     22 23.05766
## 13 23 23.83465 23.75610 23.33220 23.41300     23 24.11830
## 14 24 24.84928 24.75575 24.37232 24.43951     24 25.17895
## 15 25 25.86391 25.75540 25.41244 25.46602     25 26.23959
## 16 26 26.87853 26.75505 26.45256 26.49253     26 27.30023
## 17 27 27.89316 27.75470 27.49268 27.51904     27 28.36088
## 18 28 28.90779 28.75435 28.53280 28.54555     28 29.42152
## 19 29 29.92242 29.75400 29.57292 29.57206     29 30.48216
## 20 30 30.93704 30.75365 30.61304 30.59857     30 31.54280
## 21 31 31.95167 31.75330 31.65316 31.62508     31 32.60345
## 22 32 32.96630 32.75295 32.69328 32.65159     32 33.66409
## 23 33 33.98092 33.75260 33.73340 33.67810     33 34.72473
## 24 34 34.99555 34.75225 34.77352 34.70461     34 35.78538
## 25 35 36.01018 35.75190 35.81364 35.73112     35 36.84602
## 26 36 37.02481 36.75155 36.85376 36.75763     36 37.90666
## 27 37 38.03943 37.75120 37.89388 37.78414     37 38.96731
## 28 38 39.05406 38.75085 38.93400 38.81065     38 40.02795
## 29 39 40.06869 39.75050 39.97412 39.83716     39 41.08859
## 30 40 41.08331 40.75015 41.01424 40.86367     40 42.14923
## 31 41 42.09794 41.74980 42.05436 41.89018     41 43.20988
## 32 42 43.11257 42.74945 43.09448 42.91669     42 44.27052
## 33 43 44.12719 43.74910 44.13460 43.94320     43 45.33116
## 34 44 45.14182 44.74875 45.17472 44.96971     44 46.39181
## 35 45 46.15645 45.74840 46.21484 45.99622     45 47.45245
## 36 46 47.17107 46.74805 47.25496 47.02273     46 48.51309
## 37 47 48.18570 47.74770 48.29508 48.04924     47 49.57374
## 38 48 49.20033 48.74735 49.33520 49.07575     48 50.63438
## 39 49 50.21496 49.74700 50.37532 50.10226     49 51.69502
## 40 50 51.22958 50.74665 51.41544 51.12877     50 52.75567
## 41 51 52.24421 51.74630 52.45556 52.15528     51 53.81631
## 42 52 53.25884 52.74595 53.49568 53.18179     52 54.87695
## 43 53 54.27346 53.74560 54.53580 54.20830     53 55.93759
## 44 54 55.28809 54.74525 55.57592 55.23481     54 56.99824
## 45 55 56.30272 55.74490 56.61604 56.26132     55 58.05888
## 46 56 57.31734 56.74455 57.65616 57.28783     56 59.11952
## 47 57 58.33197 57.74420 58.69628 58.31434     57 60.18017
## 48 58 59.34660 58.74385 59.73640 59.34085     58 61.24081
## 49 59 60.36123 59.74350 60.77652 60.36736     59 62.30145
## 50 60 61.37585 60.74315 61.81664 61.39387     60 63.36209
## 51 61 62.39048 61.74280 62.85676 62.42038     61 64.42274
## 52 62 63.40511 62.74245 63.89688 63.44689     62 65.48338
## 53 63 64.41973 63.74210 64.93700 64.47340     63 66.54402
```

This table has FL 11-63 cm and the theoretical TL calculated based on linear regression output for 2cm size bins, as they do for stock assessment purposes.

```
dummy_dat_bin
```

```
##    FL   TL_blu  TL_bla   TL_ver   TL_oyt TL_cpr   TL_cny
## 1  11 11.82756 12.6446 11.34326 11.66443     11 11.45796
## 2  12 12.83021 13.6019 12.36341 12.66366     12 12.51532
## 3  13 13.83286 14.5592 13.38356 13.66289     13 13.57267
## 4  14 14.83551 15.5165 14.40371 14.66212     14 14.63003
## 5  15 15.83816 16.4738 15.42386 15.66135     15 15.68739
## 6  16 16.84081 17.4311 16.44401 16.66058     16 16.74475
## 7  17 17.84346 18.3884 17.46416 17.65981     17 17.80211
## 8  18 18.84611 19.3457 18.48431 18.65904     18 18.85946
## 9  19 19.84876 20.3030 19.50446 19.65827     19 19.91682
## 10 20 20.85141 21.2603 20.52461 20.65750     20 20.97418
## 11 21 21.85406 22.2176 21.54476 21.65673     21 22.03154
## 12 22 22.85671 23.1749 22.56491 22.65596     22 23.08890
## 13 23 23.85936 24.1322 23.58506 23.65519     23 24.14625
## 14 24 24.86201 25.0895 24.60521 24.65442     24 25.20361
## 15 25 25.86466 26.0468 25.62536 25.65365     25 26.26097
## 16 26 26.86731 27.0041 26.64551 26.65288     26 27.31833
## 17 27 27.86996 27.9614 27.66566 27.65211     27 28.37569
## 18 28 28.87261 28.9187 28.68581 28.65134     28 29.43304
## 19 29 29.87526 29.8760 29.70596 29.65057     29 30.49040
## 20 30 30.87791 30.8333 30.72611 30.64980     30 31.54776
## 21 31 31.88056 31.7906 31.74626 31.64903     31 32.60512
## 22 32 32.88321 32.7479 32.76641 32.64826     32 33.66248
## 23 33 33.88586 33.7052 33.78656 33.64749     33 34.71983
## 24 34 34.88851 34.6625 34.80671 34.64672     34 35.77719
## 25 35 35.89116 35.6198 35.82686 35.64595     35 36.83455
## 26 36 36.89381 36.5771 36.84701 36.64518     36 37.89191
## 27 37 37.89646 37.5344 37.86716 37.64441     37 38.94927
## 28 38 38.89911 38.4917 38.88731 38.64364     38 40.00662
## 29 39 39.90176 39.4490 39.90746 39.64287     39 41.06398
## 30 40 40.90441 40.4063 40.92761 40.64210     40 42.12134
## 31 41 41.90706 41.3636 41.94776 41.64133     41 43.17870
## 32 42 42.90971 42.3209 42.96791 42.64056     42 44.23606
## 33 43 43.91236 43.2782 43.98806 43.63979     43 45.29341
## 34 44 44.91501 44.2355 45.00821 44.63902     44 46.35077
## 35 45 45.91766 45.1928 46.02836 45.63825     45 47.40813
## 36 46 46.92031 46.1501 47.04851 46.63748     46 48.46549
## 37 47 47.92296 47.1074 48.06866 47.63671     47 49.52285
## 38 48 48.92561 48.0647 49.08881 48.63594     48 50.58020
## 39 49 49.92826 49.0220 50.10896 49.63517     49 51.63756
## 40 50 50.93091 49.9793 51.12911 50.63440     50 52.69492
## 41 51 51.93356 50.9366 52.14926 51.63363     51 53.75228
## 42 52 52.93621 51.8939 53.16941 52.63286     52 54.80964
## 43 53 53.93886 52.8512 54.18956 53.63209     53 55.86699
## 44 54 54.94151 53.8085 55.20971 54.63132     54 56.92435
## 45 55 55.94416 54.7658 56.22986 55.63055     55 57.98171
## 46 56 56.94681 55.7231 57.25001 56.62978     56 59.03907
## 47 57 57.94946 56.6804 58.27016 57.62901     57 60.09643
## 48 58 58.95211 57.6377 59.29031 58.62824     58 61.15378
## 49 59 59.95476 58.5950 60.31046 59.62747     59 62.21114
## 50 60 60.95741 59.5523 61.33061 60.62670     60 63.26850
## 51 61 61.96006 60.5096 62.35076 61.62593     61 64.32586
## 52 62 62.96271 61.4669 63.37091 62.62516     62 65.38322
## 53 63 63.96536 62.4242 64.39106 63.62439     63 66.44057
```