analysis for erosion

Yingmai Chen,Yan Wang

目录

```r
library(ggplot2)
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.1     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v lubridate 1.9.2     v tibble    3.2.1
## v purrr     1.0.2     v tidyr     1.3.0
## -- 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
```

```r
library(readr)
library(gridExtra)
```

```
##
## 载入程辑包: 'gridExtra'
##
## The following object is masked from 'package:dplyr':
##
##     combine
```

```r
library(car)
```

```
## 载入需要的程辑包：carData
##
## 载入程辑包：’car’
##
## The following object is masked from ’package:dplyr’:
##
##     recode
##
## The following object is masked from ’package:purrr’:
##
##     some
```

```r
library(grid)
```

```r
erosion<-read.csv("erosion.csv")
```

linear regression for all variables(MAX waveheight)

```r
stan <- function(x) {
  return((x - min(x)) / (max(x) - min(x)))
}
data <- read_csv("erosion.csv")
```

```
## Rows: 31 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Bluff
## dbl (7): Orientation (deg), RR (m/yr), Max Wave Height (m), Mud (%), BaseEl ...
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```r
names(data) <- gsub(" ", "_", names(data))
names(data) <- gsub("\\(", "", names(data))
names(data) <- gsub("\\)", "", names(data))
names(data) <- gsub("/", "_per_", names(data))
names(data) <- gsub("%", "percent", names(data))
if ("RR (m/yr)" %in% names(data)) {
  names(data)[names(data) == "RR (m/yr)"] <- "RR_m_per_yr"
}
numeric_columns <- sapply(data, is.numeric) & names(data) != "RR_m_per_yr"
data[numeric_columns] <- lapply(data[numeric_columns], stan)
predictors <- setdiff(names(data), c("Bluff", "RR_m_per_yr"))
target <- "RR_m_per_yr"
model_formula <- as.formula(paste(target, "~", paste(predictors, collapse = " + ")))
model1 <- lm(model_formula, data = data)
print(summary(model1))
```

```
## 
## Call:
## lm(formula = model_formula, data = data)
## 
## Residuals:
##      Min       1Q   Median       3Q      Max 
## -0.45124 -0.12697 -0.03711  0.09424  0.72960 
## 
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)   
## (Intercept)         0.22946    0.16714   1.373  0.18249   
## Orientation_deg     0.06428    0.16130   0.399  0.69376   
## Max_Wave_Height_m   0.79835    0.24809   3.218  0.00368 **
## Mud_percent        -0.21316    0.18391  -1.159  0.25785   
## BaseEl_m           -0.08859    0.31426  -0.282  0.78043   
## BluffEl_m          -0.48522    0.22160  -2.190  0.03851 * 
## Seawall            -0.05905    0.15774  -0.374  0.71143   
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2641 on 24 degrees of freedom
## Multiple R-squared:  0.478,  Adjusted R-squared:  0.3475
## F-statistic: 3.662 on 6 and 24 DF,  p-value: 0.01006
```

```
predictions <- predict(model1, data)
mse <- mean((data[[target]] - predictions)^2)
rsquared <- summary(model1)$r.squared
cat("(MSE):", mse, "\n")
```

```
## (MSE): 0.05400199
```

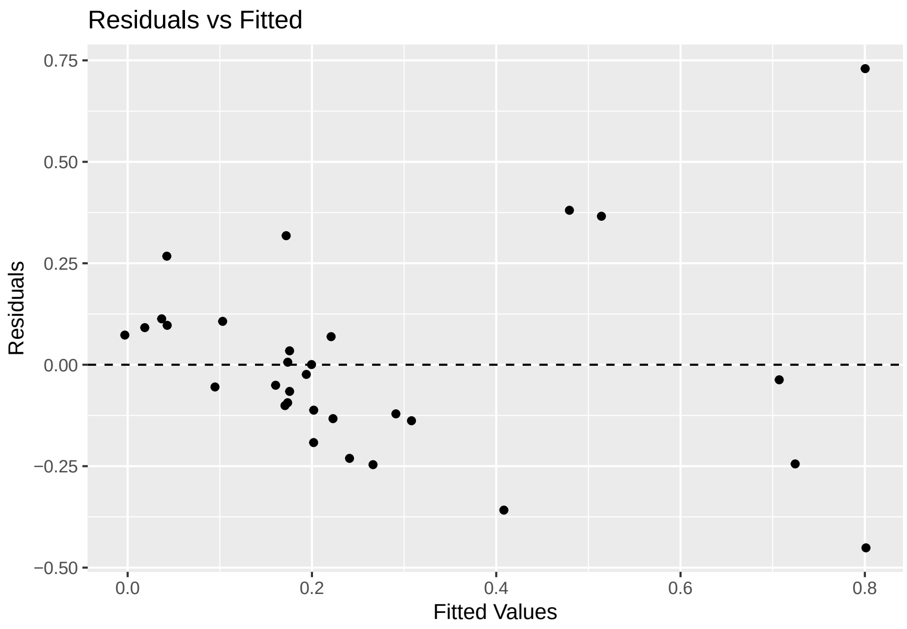
```
cat("R square:", rsquared, "\n")
```

```
## R square: 0.4779682
```

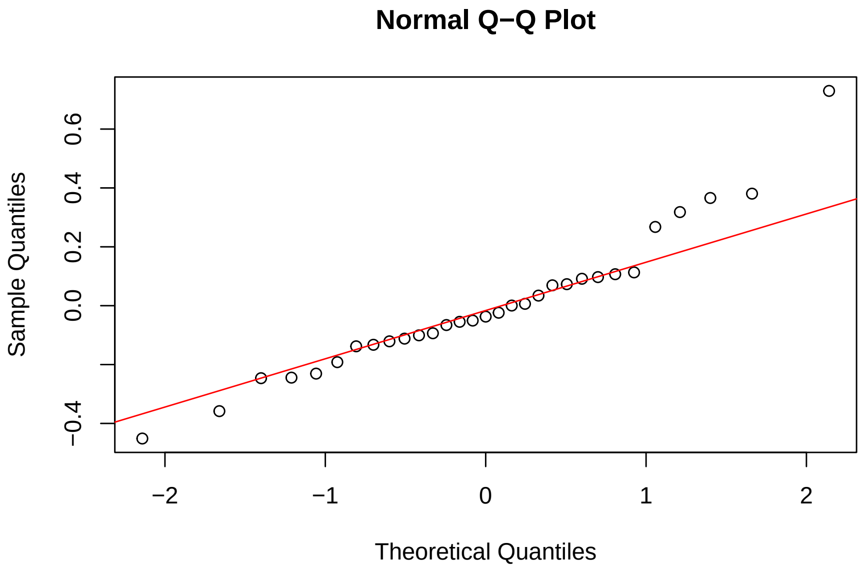
check

```
residuals <- residuals(model1)
fitted_values <- fitted(model1)

ggplot() +
  geom_point(aes(x = fitted_values, y = residuals)) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  xlab("Fitted Values") +
  ylab("Residuals") +
  ggtitle("Residuals vs Fitted")
```

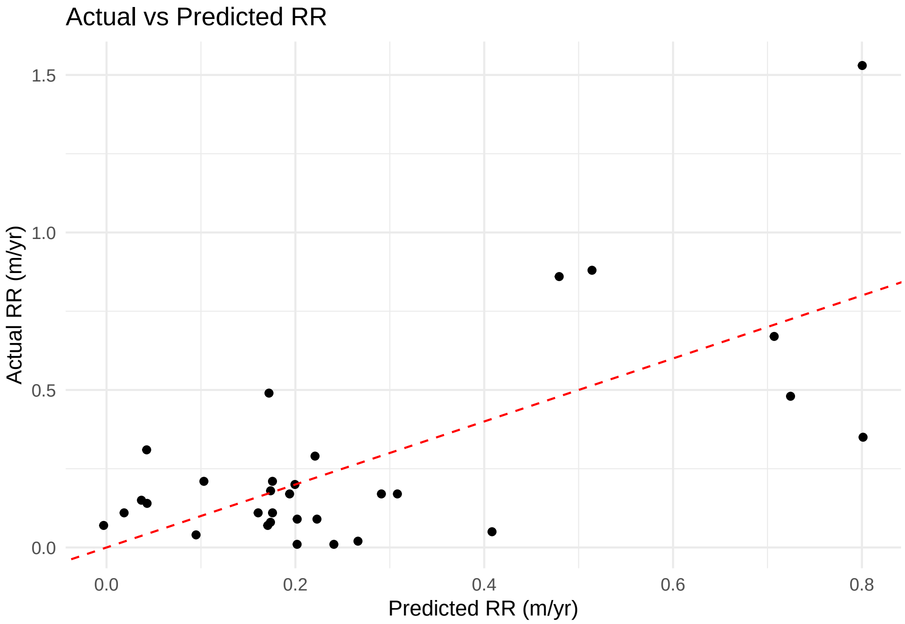


```r
qqnorm(residuals)
qqline(residuals, col = "red")
```



```r
data$Predicted_RR <- predictions


ggplot(data, aes(x = Predicted_RR, y = RR_m_per_yr)) +
  geom_point() +  # Add points
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +
  labs(x = "Predicted RR (m/yr)", y = "Actual RR (m/yr)", title = "Actual vs Predicted
  theme_minimal()
```



linear regression for all variables(MAX waveheight nne 15m/s)

```r
stan <- function(x) {
  return((x - min(x)) / (max(x) - min(x)))
}
data <- read_csv("erosionnne15.csv")
```

```
## Rows: 31 Columns: 8
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (1): Bluff
## dbl (7): Orientation (deg), RR (m/yr), Wave Height for NNE wind 15 m/s (m), ...
##
## i Use ‘spec()‘ to retrieve the full column specification for this data.
## i Specify the column types or set ‘show_col_types = FALSE‘ to quiet this message.
```

```
names(data) <- gsub(" ", "_", names(data))
names(data) <- gsub("\\(", "", names(data))
names(data) <- gsub("\\)", "", names(data))
names(data) <- gsub("/", "_per_", names(data))
names(data) <- gsub("%", "percent", names(data))
if ("RR (m/yr)" %in% names(data)) {
  names(data)[names(data) == "RR (m/yr)"] <- "RR_m_per_yr"
}
numeric_columns <- sapply(data, is.numeric) & names(data) != "RR_m_per_yr"
data[numeric_columns] <- lapply(data[numeric_columns], stan)
predictors <- setdiff(names(data), c("Bluff", "RR_m_per_yr"))
target <- "RR_m_per_yr"
model_formula <- as.formula(paste(target, "~", paste(predictors, collapse = " + ")))
model2 <- lm(model_formula, data = data)
print(summary(model2))
```

```
##
## Call:
## lm(formula = model_formula, data = data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.46536 -0.10867 -0.05099  0.06523  0.56911
##
## Coefficients:
##                                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)                            0.10887    0.15424   0.706 0.487101
## Orientation_deg                        0.05753    0.14458   0.398 0.694197
## Wave_Height_for_NNE_wind_15_m_per_s_m  1.10008    0.25469   4.319 0.000234 ***
## Mud_percent                           -0.10007    0.16959  -0.590 0.560675
## BaseEl_m                              -0.15821    0.28083  -0.563 0.578423
## BluffEl_m                             -0.60463    0.20486  -2.951 0.006962 **
## Seawall                               -0.09229    0.14080  -0.655 0.518401
## ---
## Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
##
## Residual standard error: 0.237 on 24 degrees of freedom
## Multiple R-squared:  0.5796, Adjusted R-squared:  0.4744
## F-statistic: 5.514 on 6 and 24 DF,  p-value: 0.001042
```

```
predictions <- predict(model2, data)
mse <- mean((data[[target]] - predictions)^2)
rsquared <- summary(model2)$r.squared
cat("(MSE):", mse, "\n")
```

```
## (MSE): 0.04349345
```

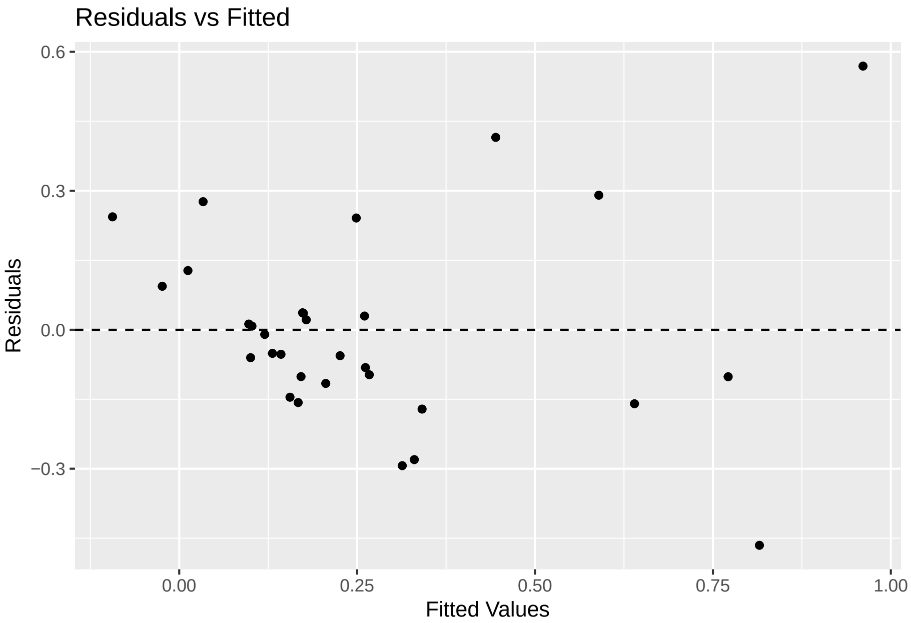
```
cat("R square:", rsquared, "\n")
```

```
## R square: 0.5795532
```

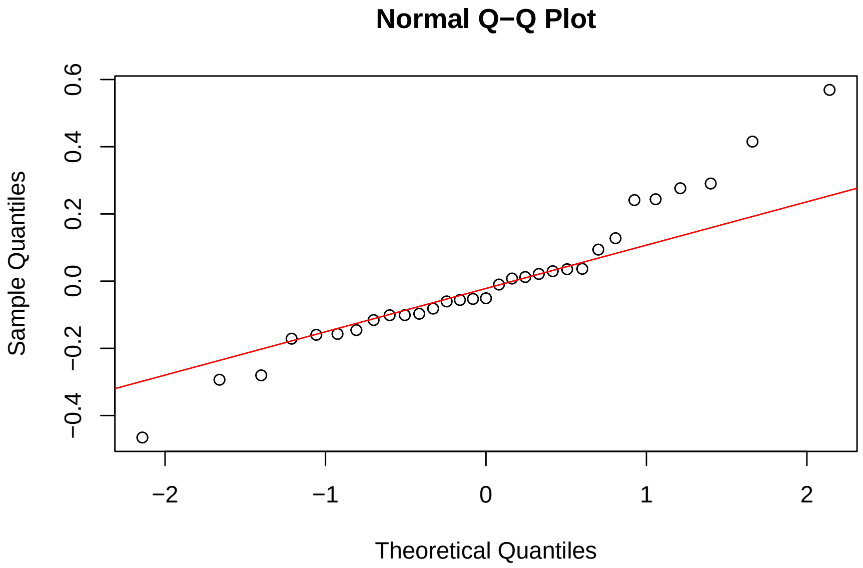
check

```
# 1.residual vs fitted
residuals <- residuals(model2)
fitted_values <- fitted(model2)

ggplot() +
  geom_point(aes(x = fitted_values, y = residuals)) +
  geom_hline(yintercept = 0, linetype = "dashed") +
  xlab("Fitted Values") +
  ylab("Residuals") +
  ggtitle("Residuals vs Fitted")
```

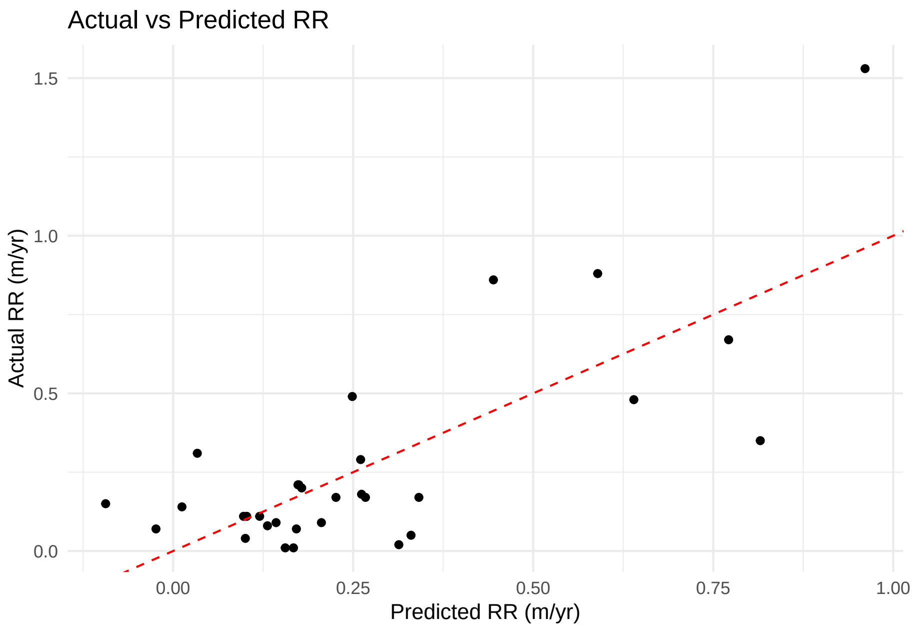


```
# 2. Q-Q Plot for Normal Distribution of Residuals
qqnorm(residuals)
qqline(residuals, col = "red")
```



```
# 3.actual with fitted
data$Predicted_RR <- predictions


ggplot(data, aes(x = Predicted_RR, y = RR_m_per_yr)) +
  geom_point() +  # Add points
  geom_abline(intercept = 0, slope = 1, color = "red", linetype = "dashed") +  # Add a
  labs(x = "Predicted RR (m/yr)", y = "Actual RR (m/yr)", title = "Actual vs Predicted
  theme_minimal()
```


Actual vs Predicted RR
Actual RR (m/yr)
1.5
1.0
0.5
0.0
0.00
0.25
0.50
0.75
1.00
Predicted RR (m/yr)