Regresi SPL

Zilda Ainun Tazkia

2025-10-06

AUTOKORELASI

Memuat Paket dan Data

```
library(lmtest)
library(car)
library(nlme)
library(readxl)
```

Membangkitkan Data

```
set.seed(123)
n <- 100

# Bangkitkan variabel independen
x1 <- runif(n, 50, 150)  #  iklan TV
x2 <- runif(n, 30, 100)  #  iklan Radio
x3 <- runif(n, 10, 50)   #  variabel lain

# Buat autokorelasi AR(1) untuk error
rho <- 0.7  # koefisien autokorelasi positif
e <- numeric(n)
e[1] <- rnorm(1, 0, 5)  # error pertama
for(t in 2:n){
  e[t] <- rho*e[t-1] + rnorm(1, 0, 5)  # AR(1)
}

# Bangkitkan Y
beta0 <- 10
beta1 <- 0.5
beta2 <- 0.3
beta3 <- 0.2

y <- beta0 + beta1*x1 + beta2*x2 + beta3*x3 + e

# Gabungkan jadi data frame
data_autokorelasi <- data.frame(
```

```
  time = 1:n,
  x1 = x1,
  x2 = x2,
  x3 = x3,
  y = y
)
```

Menampilkan Data

```
head(data_autokorelasi)
```

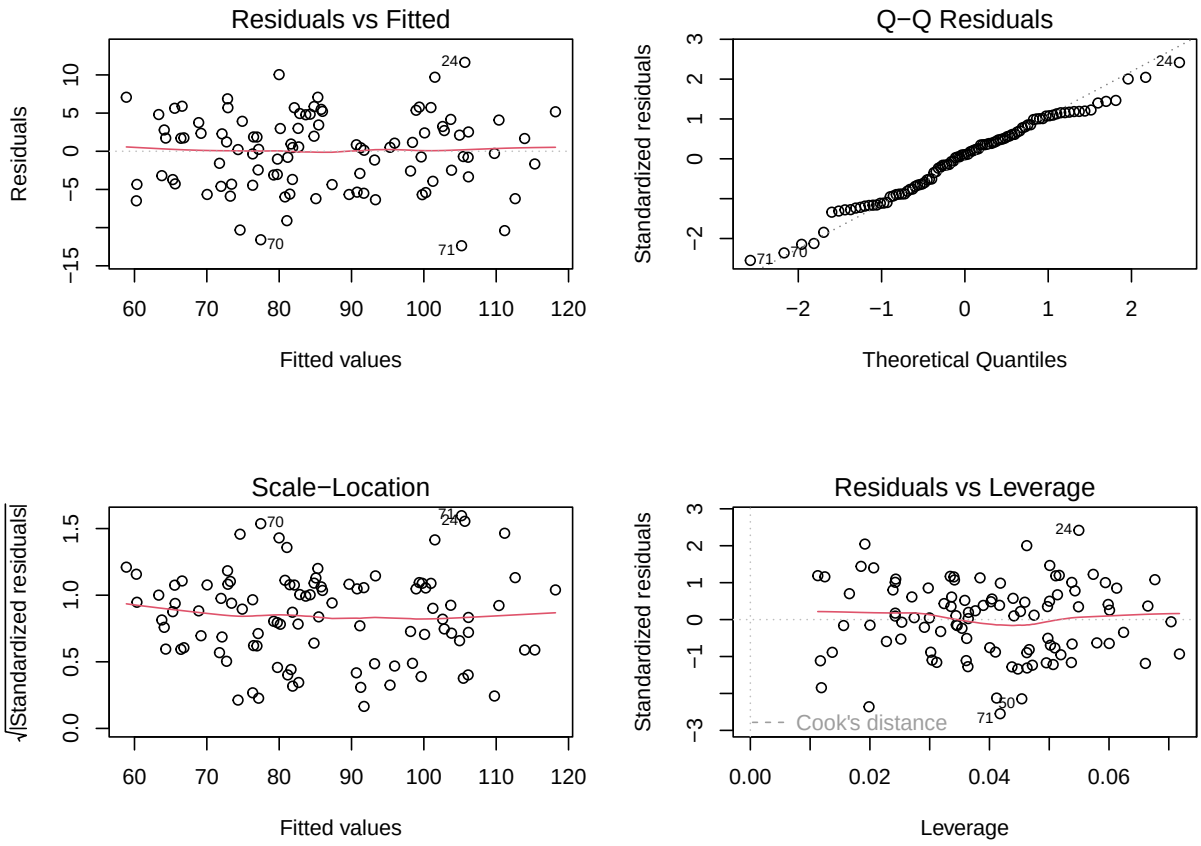
```
##   time        x1       x2       x3         y
## 1    1  78.75775 71.99923 19.54904  78.82715
## 2    2 128.83051 53.29765 48.49436 106.70572
## 3    3  90.89769 64.20291 34.05463  87.80327
## 4    4 138.30174 96.81317 30.60119 113.67101
## 5    5 144.04673 63.80317 26.10293 105.33680
## 6    6  54.55565 92.32452 45.20986  71.88151
```

Membangun Model OLS Awal (Untuk Diagnosis)

```
# Model OLS sama seperti file sebelumnya
model_ols <- lm(y ~ x1 + x2 + x3, data = data_autokorelasi)
```

Diagnosis Autokorelasi

```
par(mfrow = c(2, 2))
plot(model_ols)
```




```
par(mfrow = c(1, 1))
```

Uji Autokorelasi (Durbin-Watson)

Tujuan: Menguji apakah ada korelasi antar sisaan pada observasi yang berdekatan. Autokorelasi umumnya menjadi masalah pada data deret waktu (time series).

Hipotesis:

H_0: Tidak ada autokorelasi (koefisien autokorelasi = 0).

H_1: Terdapat autokorelasi.

Kriteria Keputusan: Tolak H_0 jika p-value < 0.05. Asumsi independensi sisaan terpenuhi jika kita gagal menolak H_0 (p-value > 0.05).

```
print(dwtest(model_ols))
```

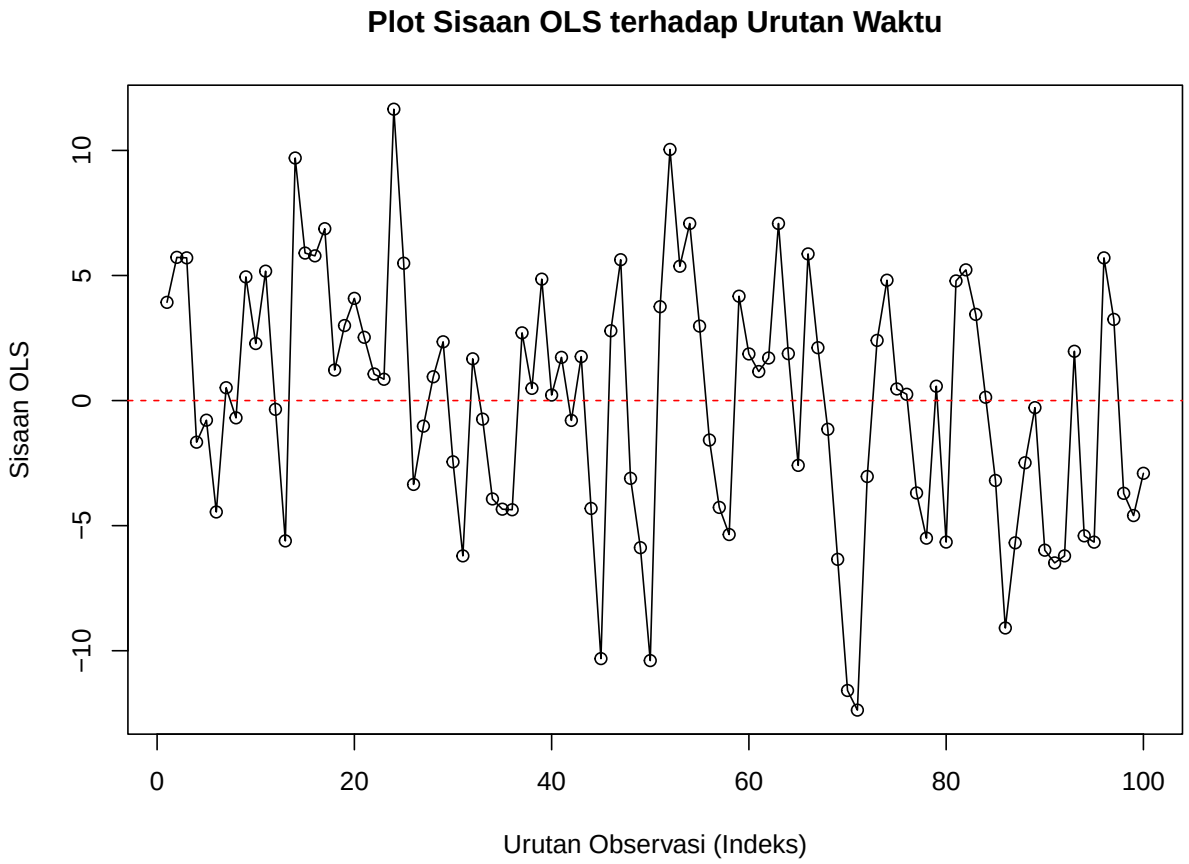
```
##
##  Durbin-Watson test
##
## data:  model_ols
## DW = 1.1565, p-value = 8.817e-06
## alternative hypothesis: true autocorrelation is greater than 0
```

Visualisasi Autokorelasi

Pola non-acak pada plot sisaan mengindikasikan adanya autokorelasi.

```
# Simpan sisaan ke dalam variabel baru
sisaan_ols <- residuals(model_ols)

# Plot sisaan terhadap indeks yang panjangnya dijamin sama.
# Ini akan mencegah error jika ada data NA yang dihilangkan oleh lm().
plot(1:length(sisaan_ols), sisaan_ols, type = 'o',
     xlab = "Urutan Observasi (Indeks)",
     ylab = "Sisaan OLS",
     main = "Plot Sisaan OLS terhadap Urutan Waktu")
abline(h = 0, col = "red", lty = 2)
```



ACF (Autocorrelation Function): Menunjukkan korelasi antara sisaan dengan nilai-nilai lag-nya.

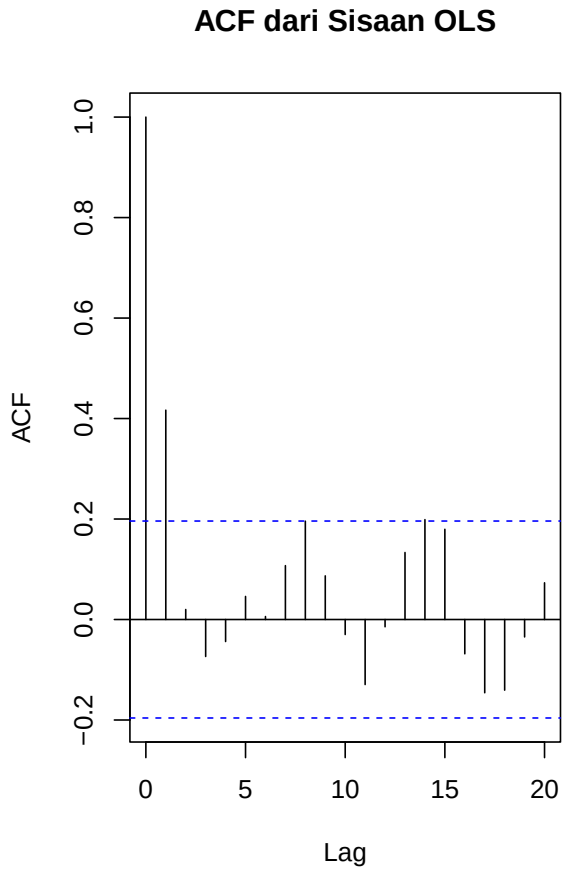
PACF (Partial Autocorrelation Function): Menunjukkan korelasi antara sisaan dengan nilai lag-nya setelah menghilangkan efek dari lag-lag perantara.

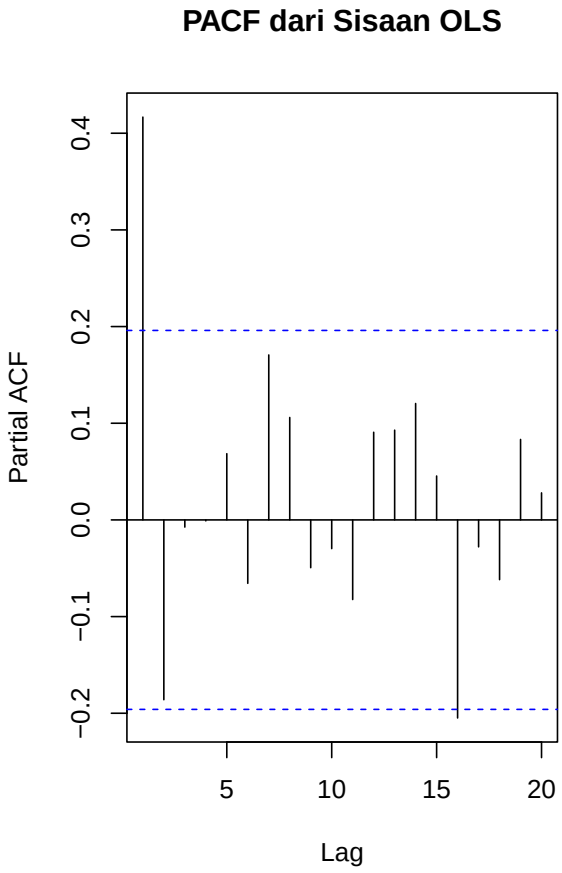
```
# Dapatkan sisaan dari model OLS awal
sisaan_ols <- residuals(model_ols)

# Atur layout plot menjadi 1 baris, 2 kolom
```

```r
par(mfrow = c(1, 2))

# Buat plot ACF dan PACF
acf(sisaan_ols, main = "ACF dari Sisaan OLS")
pacf(sisaan_ols, main = "PACF dari Sisaan OLS")
```






```r
# Kembalikan layout ke default
par(mfrow = c(1, 1))
```

Multikolineritas

Memuat Paket dan Data

```r
library(readxl)
library(car)
library(glmnet)
library(knitr)
library(reshape)
library(ggplot2)
```

Membangkitkan Data

```r
set.seed(123)  # supaya hasil bisa direproduksi
n <- 100

# Bangkitkan variabel independen
x1 <- runif(n, 5000000, 15000000)
x2 <- x1 + runif(n, 500000, 1000000)
x3 <- runif(n, 10, 50)

# Bangkitkan variabel dependen Y
beta0 <- 20
beta1 <- 0.5
beta2 <- 0.3
beta3 <- 0.2
y <- beta0 + beta1*x1 + beta2*x2 + beta3*x3 + rnorm(n, 0, 5)

# Gabungkan jadi data frame
data_multikolineritas <- data.frame(
  Iklan_Tv = x1,
  Iklan_Radio = x2,
  Endorse = x3,
  Penjualan = y
)
```

Menampilkan Data

```r
head(data_multikolineritas)
```

```
##   Iklan_Tv Iklan_Radio  Endorse Penjualan
## 1  7875775     8675770 19.54904   6540646
## 2 12883051    13549463 48.49436  10506398
## 3  9089769     9834076 34.05463   7495136
## 4 13830174    14807411 30.60119  11357331
## 5 14404673    15146124 26.10293  11746198
## 6  5455565     6400740 45.20986   4648032
```

Diagnosis Multikolinearitas

Mengukur seberapa besar korelasi linear antar prediktor dalam model regresi. Multikolinearitas tinggi dapat membuat koefisien regresi tidak stabil dan standar error meningkat, sehingga uji t/f menjadi tidak reliabel.

Hipotesis:

H_0: Tidak ada multikolinearitas (VIF < 10).

H_1: Terdapat multikolinearitas (VIF > 10).

Kriteria Keputusan:

Jika VIF $> 10 \rightarrow$ tolak H_0 $\rightarrow$ multikolinearitas parah.

Jika VIF $< 10 \rightarrow$ gagal menolak H_0 $\rightarrow$ multikolinearitas tidak serius.

Interpretasi:

VIF tinggi → variabel prediktor sangat berkorelasi dengan prediktor lain.

Langkah penanganan:

Menghapus salah satu variabel yang berkorelasi tinggi.

Menggunakan teknik regularisasi seperti LASSO atau Ridge Regression.

```
# Membuat model OLS pada data asli untuk menghitung VIF
model_ols <- lm(Penjualan ~ Iklan_Tv + Iklan_Radio + Endorse, data = data_multikolineritas)
vif_values <- vif(model_ols)

print("--- Hasil Uji VIF untuk Multikolinearitas ---")
```

```
## [1] "--- Hasil Uji VIF untuk Multikolinearitas ---"
```

```
print(vif_values)
```

```
##    Iklan_Tv Iklan_Radio     Endorse
##  473.243337  473.583713    1.017238
```

```
# Analisis hasil VIF
if (any(vif_values > 10)) {
  print("MULTIKOLINEARITAS TERDETEKSI!")
  print("KESIMPULAN: Ada multikolinearitas parah.")
  high_vif_vars <- names(vif_values[vif_values > 10])
  print(paste("Variabel dengan VIF > 10:", paste(high_vif_vars, collapse = ", ")))
} else {
  print("Tidak ada multikolinearitas yang serius (semua VIF < 10)")
}
```

```
## [1] "MULTIKOLINEARITAS TERDETEKSI!"
## [1] "KESIMPULAN: Ada multikolinearitas parah."
## [1] "Variabel dengan VIF > 10: Iklan_Tv, Iklan_Radio"
```

```
# Matriks korelasi untuk memahami hubungan antar variabel
cat("\n--- Matriks Korelasi Antar Prediktor ---\n")
```

```
##
## --- Matriks Korelasi Antar Prediktor ---
```

```
cor_matrix <- cor(data_multikolineritas[, c("Iklan_Tv", "Iklan_Radio", "Endorse")])
kable(cor_matrix, caption = "Matriks Korelasi Antar Variabel Prediktor", digits = 3)
```

Table 1: Matriks Korelasi Antar Variabel Prediktor

	Iklan_Tv	Iklan_Radio	Endorse
Iklan_Tv	1.000	0.999	-0.068
Iklan_Radio	0.999	1.000	-0.073

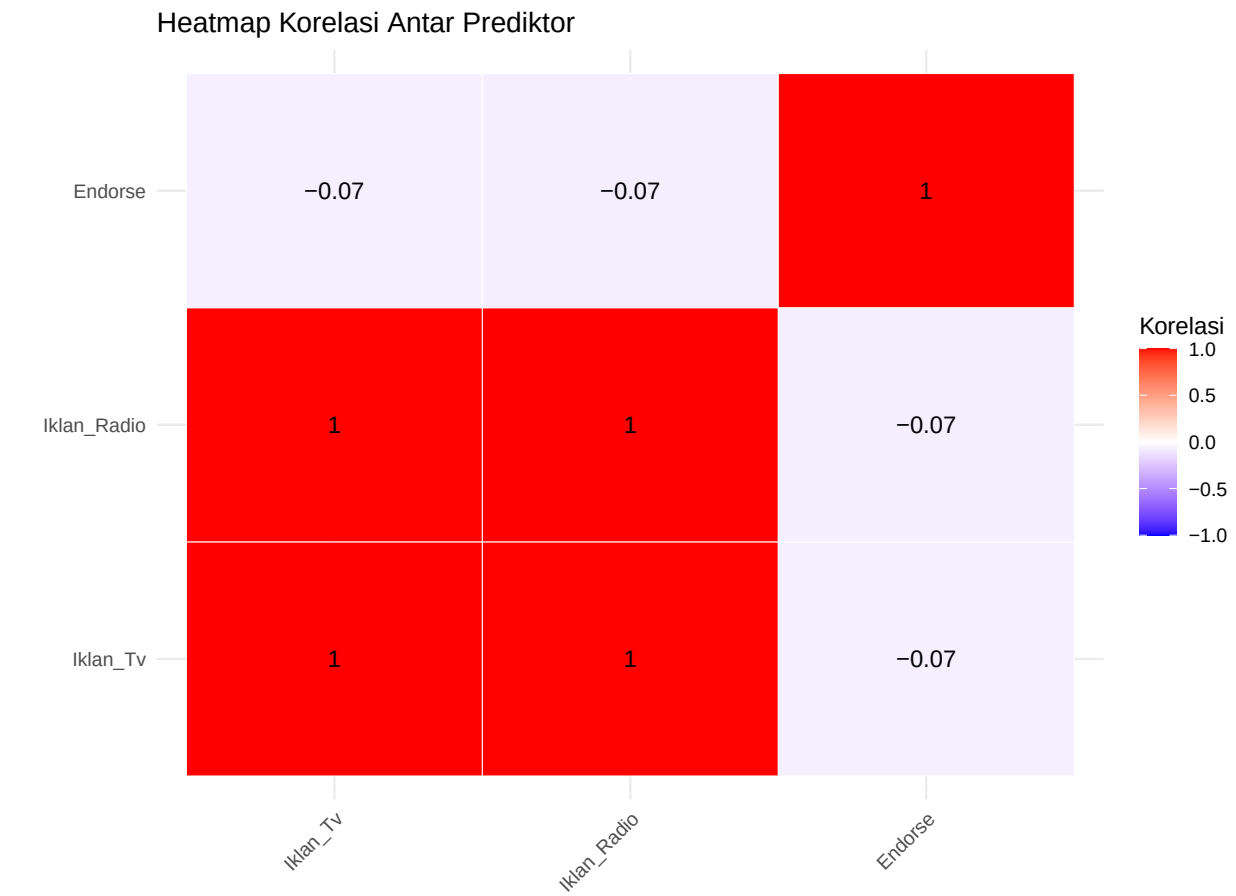
	Iklan_Tv	Iklan_Radio	Endorse
Endorse	-0.068	-0.073	1.000

Visualisasi Multikolineritas

```
# Pastikan cor_matrix adalah matriks
cor_matrix <- as.matrix(cor(data_multikolineritas[, c("Iklan_Tv", "Iklan_Radio", "Endorse")]))

# Ubah menjadi format long
cor_long <- melt(cor_matrix, varnames = c("Var1", "Var2"))

# Heatmap
ggplot(cor_long, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile(color = "white") +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1,1), space = "Lab",
                       name="Korelasi") +
  geom_text(aes(label = round(value, 2)), color = "black", size = 4) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, hjust=1)) +
  labs(title = "Heatmap Korelasi Antar Prediktor", x = "", y = "")
```



Sisaan Normal Baku

Membangkitkan Data

```
set.seed(123)
n <- 100

# Bangkitkan 3 variabel independen
x1 <- rnorm(n, mean = 0, sd = 1)
x2 <- rnorm(n, mean = 0, sd = 1)
x3 <- rnorm(n, mean = 0, sd = 1)

# Bangkitkan variabel dependen (linear combination + noise)
beta0 <- 0
beta1 <- 0.5
beta2 <- 0.3
beta3 <- 0.2
y <- beta0 + beta1*x1 + beta2*x2 + beta3*x3 + rnorm(n, 0, 1)

# Gabungkan ke data frame
data_normalitas <- data.frame(
  x1 = x1,
  x2 = x2,
  x3 = x3,
  y = y
)
```

Menampilkan Data

```
head(data_normalitas)
```

```
##            x1          x2         x3          y
## 1 -0.56047565 -0.71040656  2.1988103 -0.7688399
## 2 -0.23017749  0.25688371  1.3124130 -0.5282300
## 3  1.55870831 -0.24669188 -0.2651451 -0.2862211
## 4  0.07050839 -0.34754260  0.5431941 -1.0128831
## 5  0.12928774 -0.95161857 -0.4143399 -0.7408692
## 6  1.71506499 -0.04502772 -0.4762469  1.0799540
```

Membangun Model OLS Awal (Untuk Diagnosis)

```
# Model OLS sama seperti file sebelumnya
model_ols <- lm(y ~ x1 + x2 + x3, data = data_normalitas)
```

Diagnosis Normalitas

Uji Normalitas (Shapiro-Wilk)

Tujuan: Menguji apakah sisaan model berdistribusi normal.

Hipotesis:

H_0: Sisaan berdistribusi normal.

H_1: Sisaan tidak berdistribusi normal.

Kriteria Keputusan: Tolak H_0 jika p-value < 0.05. Asumsi normalitas terpenuhi jika kita gagal menolak H_0 (p-value > 0.05).

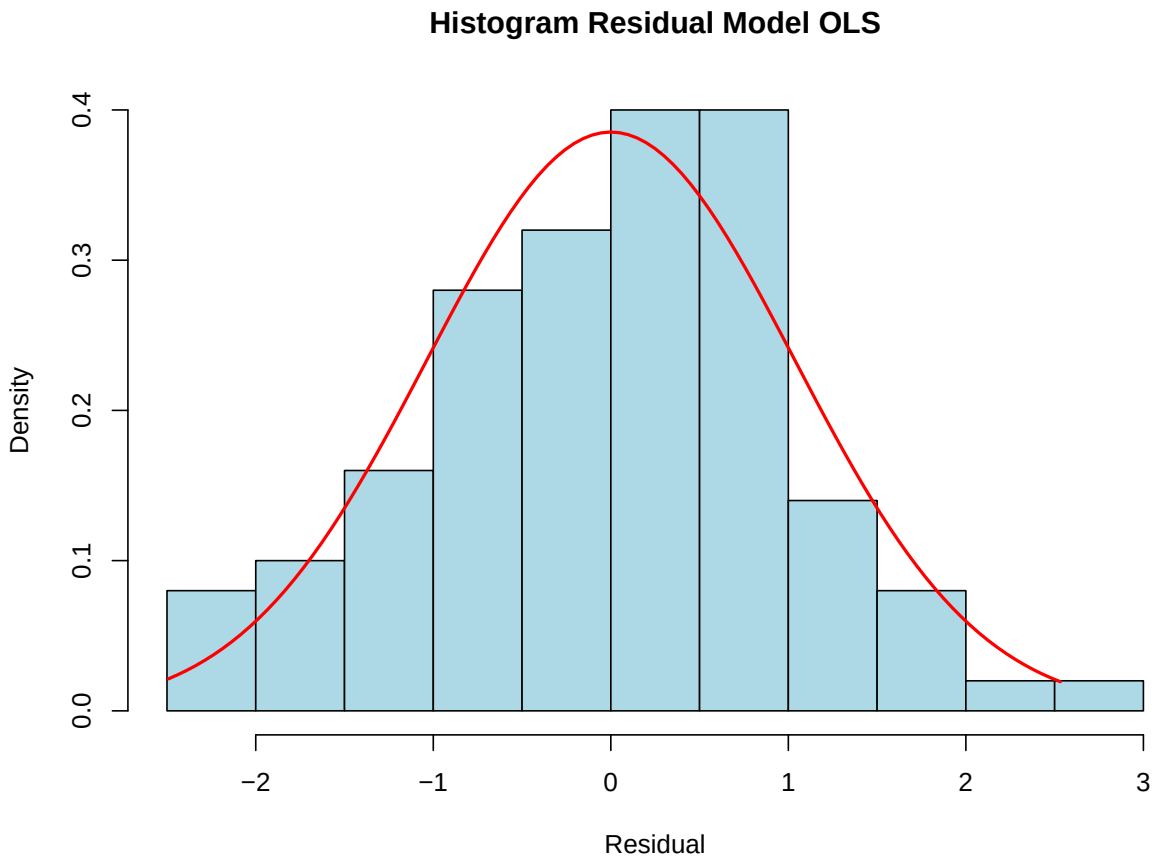
```
print(shapiro.test(residuals(model_ols)))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(model_ols)
## W = 0.99399, p-value = 0.9398
```

Visualisasi Normalitas

```
# Ambil residual dari model OLS
residuals_ols <- residuals(model_ols)

# 1. Histogram dengan kurva normal
hist(residuals_ols, breaks = 15, probability = TRUE,
     main = "Histogram Residual Model OLS",
     xlab = "Residual",
     col = "lightblue", border = "black")
# Tambahkan kurva distribusi normal
xfit <- seq(min(residuals_ols), max(residuals_ols), length = 100)
yfit <- dnorm(xfit, mean = mean(residuals_ols), sd = sd(residuals_ols))
lines(xfit, yfit, col = "red", lwd = 2)
```



Heteroskedastisitas

Membangkitkan Data

```
set.seed(123)
n <- 100

# Variabel independen
x1 <- runif(n, 1, 10)
x2 <- runif(n, 5, 15)
x3 <- runif(n, 10, 20)

error <- rnorm(n, mean = 0, sd = 0.5 * x1)

# Variabel dependen
beta0 <- 5
beta1 <- 2
beta2 <- 1.5
beta3 <- 1
y <- beta0 + beta1*x1 + beta2*x2 + beta3*x3 + error
```

```r
# Gabungkan menjadi data frame
data_hetero <- data.frame(x1 = x1, x2 = x2, x3 = x3, y = y)
```

Menampilkan Data

```r
head(data_hetero)
```

```
##         x1        x2       x3        y
## 1 3.588198 10.999890 12.38726 42.47677
## 2 8.094746  8.328235 19.62359 56.41804
## 3 4.680792  9.886130 16.01366 45.98192
## 4 8.947157 14.544738 15.15030 55.35067
## 5 9.464206  9.829024 14.02573 52.13242
## 6 1.410008 13.903502 18.80247 47.28006
```

Membangun Model OLS Awal (Untuk Diagnosis)

```r
# Model OLS sama seperti file sebelumnya
model_ols <- lm(y ~ x1 + x2 + x3, data = data_hetero)
```

Diagnosis Heteroskedastisitas

Uji Homoskedastisitas (Breusch-Pagan)

Tujuan: Menguji apakah varians dari sisaan konstan (homoskedastisitas) atau tidak (heteroskedastisitas).

Hipotesis:

H_0: Varians sisaan konstan (homoskedastisitas).

H_1: Varians sisaan tidak konstan (terdapat heteroskedastisitas).

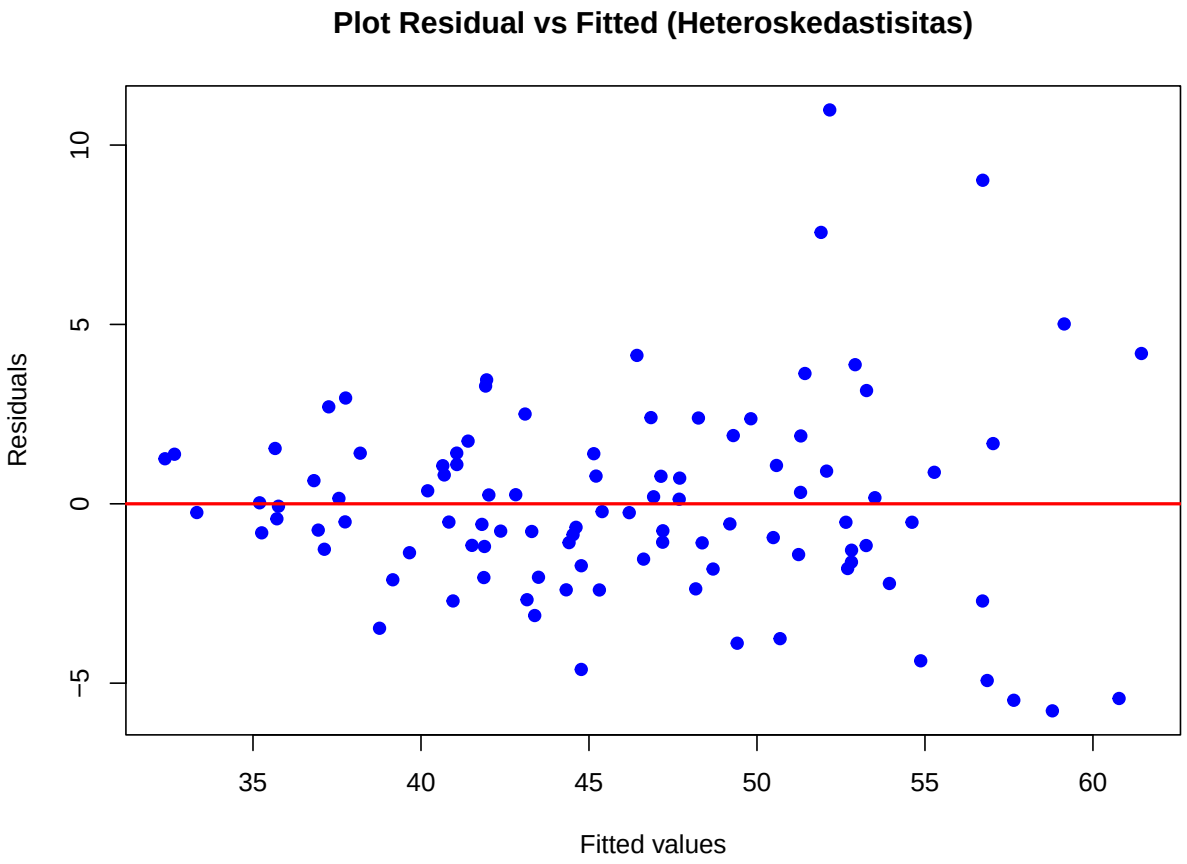
Kriteria Keputusan: Tolak H_0 jika p-value < 0.05. Asumsi homoskedastisitas terpenuhi jika kita gagal menolak H_0 (p-value > 0.05).

```r
print(bptest(model_ols))
```

```
##
##  studentized Breusch-Pagan test
##
## data:  model_ols
## BP = 17.21, df = 3, p-value = 0.0006397
```

Visualisasi heteroskedastisitas

```r
plot(model_ols$fitted.values, residuals(model_ols),
     xlab = "Fitted values", ylab = "Residuals",
     main = "Plot Residual vs Fitted (Heteroskedastisitas)",
     pch = 19, col = "blue")
abline(h = 0, col = "red", lwd = 2)
```


Plot Residual vs Fitted (Heteroskedastisitas)
Residuals
10
5
0
−5
35
40
45
50
55
60
Fitted values