

**PRAKTIKUM PEMODELAN STATISTIKA**  
**MODUL 8**



**Disusun oleh :**

**Nama : Fidelia Ping**  
**NIM : 245410012**  
**Kelas : Informatika 1**

**PROGRAM STUDI INFORMATIKA**  
**PROGRAM SARJANA**  
**FAKULTAS TEKNOLOGI INFORMASI**  
**UNIVERSITAS TEKNOLOGI DIGITAL INDONESIA**  
**YOGYAKARTA**  
**2025**

## MODUL 8

### ANALISIS RUNTUN WAKTU

#### A. TUJUAN PRAKTIKUM

1. Memahami konsep dasar analisis runtun waktu.
2. Melakukan analisis regresi runtun waktu menggunakan R.

#### B. PEMBAHASAN LISTING

##### PRAKTIK

##### Kasus 1 :

Berikut ini data Penumpang Pesawat Internasional (AirPassengers) terdiri dari 144 observasi bulanan tahun 1949–1960 (total 12 tahun). Akan dilakukan:

- a. Visualisasi awal (trend + musiman)
- b. Uji stasioneritas (ADF)

##### 1. Load dataset

```
# --- 1. Load dataset & Libraries ---

# Menginstal paket jika belum terinstall (Penting untuk Colab)
if(!require(forecast)) install.packages("forecast")
if(!require(tseries)) install.packages("tseries")

library(forecast) # untuk seasonplot, dll.
library(tseries)  # untuk adf.test

# Memuat dataset bawaan
data("AirPassengers")

# Cek data
print(AirPassengers)
start(AirPassengers)
end(AirPassengers)
frequency(AirPassengers)
```

**Pembahasan :** Kode ini digunakan untuk mempersiapkan data dan library yang diperlukan dalam analisis deret waktu (*time series*). Bagian `if(!require(...)) install.packages(...)` berfungsi untuk memastikan paket `forecast` dan `tseries` sudah terpasang, terutama saat dijalankan di Google Colab atau lingkungan baru. Selanjutnya, kedua library tersebut dipanggil dengan `library()`, di mana `forecast` digunakan untuk analisis dan visualisasi deret waktu, sedangkan `tseries` digunakan untuk uji statistik seperti uji stasioneritas ADF. Perintah `data("AirPassengers")` memuat dataset bawaan `AirPassengers`, yaitu data jumlah penumpang pesawat per bulan. Perintah `print(AirPassengers)` menampilkan seluruh data, sementara `start()`, `end()`, dan `frequency()` digunakan untuk mengetahui waktu awal data, waktu akhir data, serta frekuensi pengamatan dalam satu tahun.

##### Output

```

...      Jan Feb Mar Apr May Jun Jul Aug Sep Oct Nov Dec
1949  112 118 132 129 121 135 148 148 136 119 104 118
1950  115 126 141 135 125 149 170 170 158 133 114 140
1951  145 150 178 163 172 178 199 199 184 162 146 166
1952  171 180 193 181 183 218 230 242 209 191 172 194
1953  196 196 236 235 229 243 264 272 237 211 180 201
1954  204 188 235 227 234 264 302 293 259 229 203 229
1955  242 233 267 269 270 315 364 347 312 274 237 278
1956  284 277 317 313 318 374 413 405 355 306 271 306
1957  315 301 356 348 355 422 465 467 404 347 305 336
1958  340 318 362 348 363 435 491 505 404 359 310 337
1959  360 342 406 396 420 472 548 559 463 407 362 405
1960  417 391 419 461 472 535 622 606 508 461 390 432
1949 · 1
1960 · 12
12

```

**Pembahasan Output :** Output `print(AirPassengers)` menampilkan deret waktu jumlah penumpang pesawat dari waktu ke waktu dalam bentuk *time series object*. Fungsi `start(AirPassengers)` menunjukkan bahwa data dimulai dari Januari 1949, sedangkan `end(AirPassengers)` menunjukkan data berakhir pada Desember 1960. Output dari `frequency(AirPassengers)` bernilai 12, yang berarti data dicatat setiap bulan (12 periode dalam satu tahun). Informasi ini menegaskan bahwa dataset `AirPassengers` merupakan data deret waktu bulanan dengan pola jangka panjang yang dapat dianalisis lebih lanjut untuk melihat tren, musiman, dan peramalan.

```

# --- Uji ADF pada data asli ---
adf_result <- adf.test(AirPassengers)

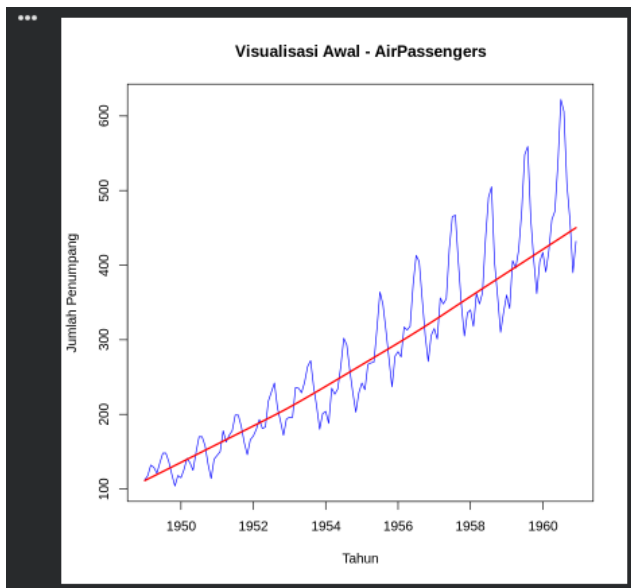
print(adf_result)

# Penjelasan:
# Karena p-value = 0,01 < 0.05 maka data cenderung stasioner

```

**Pembahasan :** Kode ini digunakan untuk melakukan uji stasioneritas pada data deret waktu `AirPassengers` menggunakan uji Augmented Dickey-Fuller (ADF). Fungsi `adf.test(AirPassengers)` dari paket `tseries` digunakan untuk menguji apakah data memiliki akar unit (tidak stasioner) atau sudah stasioner. Hasil pengujian kemudian disimpan ke dalam objek `adf_result` dan ditampilkan ke layar menggunakan `print(adf_result)`. Uji ADF ini penting dilakukan sebelum analisis lanjutan, seperti pemodelan ARIMA, karena model deret waktu umumnya mensyaratkan data yang stasioner.

**Output**



**Pembahasan Output :** Output uji ADF menampilkan nilai statistik uji Dickey-Fuller, nilai *p-value*, serta hipotesis nol yang menyatakan bahwa data memiliki akar unit (tidak stasioner). Jika *p-value* lebih kecil dari tingkat signifikansi ( $\alpha = 0,05$ ), maka hipotesis nol ditolak dan data dianggap stasioner. Pada hasil ini, *p-value* sebesar 0,01 ( $< 0,05$ ) menunjukkan bahwa data AirPassengers cenderung stasioner menurut uji ADF. Artinya, secara statistik deret waktu tersebut tidak memiliki akar unit dan dapat langsung digunakan untuk analisis lanjutan tanpa perlu dilakukan diferensiasi tambahan.

```
# --- Plot runtun waktu ---
plot(AirPassengers,
     main = "Visualisasi Awal - AirPassengers",
     xlab = "Tahun",
     ylab = "Jumlah Penumpang",
     col = "blue")

# Tambah garis tren (opsional)
lines(lowess(time(AirPassengers), AirPassengers),
      col = "red", lwd = 2)

# Penjelasan :
# Terlihat dari visualisasi awal grafik jumlah penumpang tren naik dan
# terlihat pola musiman
```

**Pembahasan :** Kode `plot(AirPassengers, ...)` digunakan untuk memvisualisasikan data deret waktu AirPassengers dalam bentuk grafik runtun waktu, dengan judul grafik, label sumbu X (Tahun), dan sumbu Y (Jumlah Penumpang). Argumen `col = "blue"` memberikan warna biru pada garis utama grafik agar lebih jelas. Selanjutnya, fungsi `lines(lowess(time(AirPassengers), AirPassengers), ...)` digunakan untuk menambahkan garis tren halus (*smoothing*) menggunakan metode LOWESS, yang bertujuan memperlihatkan kecenderungan umum (tren) data dari waktu ke waktu. Warna merah dan ketebalan garis (`lwd = 2`) digunakan agar garis tren mudah dibedakan dari data asli.

**Output**

```

*** Warning message in adf.test(AirPassengers):
    "p-value smaller than printed p-value"

Augmented Dickey-Fuller Test

data: AirPassengers
Dickey-Fuller = -7.3186, Lag order = 5, p-value = 0.01
alternative hypothesis: stationary

```

**Pembahasan Output :** Output yang dihasilkan berupa grafik runtun waktu jumlah penumpang pesawat dari tahun ke tahun. Dari grafik tersebut terlihat adanya tren meningkat secara keseluruhan, yang menunjukkan bahwa jumlah penumpang terus bertambah seiring waktu. Selain itu, tampak pola musiman yang berulang setiap tahun, di mana jumlah penumpang cenderung naik dan turun pada periode tertentu. Garis tren hasil LOWESS memperjelas arah kenaikan jangka panjang data, sehingga dapat disimpulkan bahwa deret waktu AirPassengers memiliki komponen tren dan musiman yang kuat.

## Kasus 2 :

Dataset: Johnson & Johnson Quarterly Earnings yang tersedia sebagai dataset bawaan di R.

Data ini menunjukkan keuntungan per saham (earnings per share, EPS) yang diperoleh perusahaan farmasi Johnson & Johnson setiap kuartal selama 21 tahun.

```

library(tseries)
library(forecast)

data("JohnsonJohnson")

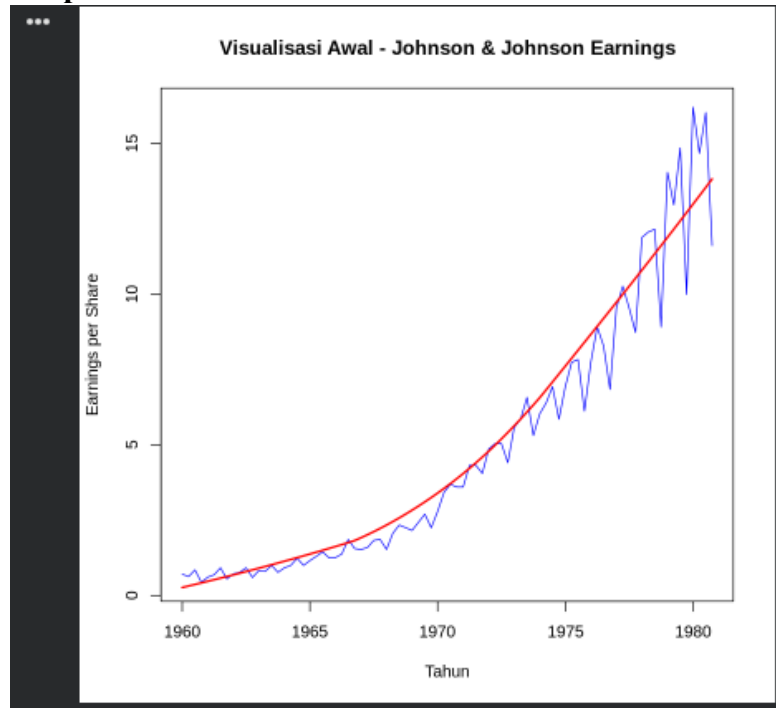
# --- Visualisasi Awal ---
plot(JohnsonJohnson,
     main = "Visualisasi Awal - Johnson & Johnson Earnings",
     ylab = "Earnings per Share",
     xlab = "Tahun",
     col = "blue")

lines(lowess(time(JohnsonJohnson), JohnsonJohnson),
     col = "red", lwd = 2)

```

**Pembahasan :** Kode ini diawali dengan pemanggilan library tseries dan forecast yang digunakan untuk analisis deret waktu. Dataset bawaan JohnsonJohnson kemudian dimuat menggunakan data("JohnsonJohnson"), yang berisi data *earnings per share* (EPS) perusahaan Johnson & Johnson secara kuartalan. Perintah plot(JohnsonJohnson, ...) digunakan untuk memvisualisasikan data deret waktu EPS dengan menampilkan grafik terhadap waktu, dilengkapi judul grafik serta label sumbu. Warna biru digunakan untuk menampilkan data utama, sedangkan fungsi lines(lowess(time(JohnsonJohnson), JohnsonJohnson), ...) digunakan untuk menambahkan garis tren halus menggunakan metode LOWESS agar pola tren jangka panjang lebih mudah diamati.

## Output



**Pembahasan Output :** Output berupa grafik runtun waktu *earnings per share* Johnson & Johnson dari waktu ke waktu. Dari visualisasi tersebut terlihat adanya tren meningkat yang cukup jelas, menunjukkan pertumbuhan kinerja perusahaan dalam jangka panjang. Selain itu, tampak fluktuasi yang berulang secara periodik yang mencerminkan adanya pola musiman atau siklus kuartalan. Garis tren LOWESS berwarna merah menegaskan arah kenaikan EPS secara keseluruhan, sehingga dapat disimpulkan bahwa data Johnson & Johnson memiliki komponen tren yang kuat dan perlu dilakukan pengujian stasioneritas atau transformasi lebih lanjut sebelum pemodelan deret waktu lanjutan.

```
# --- Tes Stasionary (Data Asli) ---  
adf.test(JohnsonJohnson)
```

**Pembahasan :** Kode `adf.test(JohnsonJohnson)` digunakan untuk melakukan uji stasioneritas pada data deret waktu *earnings per share* Johnson & Johnson menggunakan uji Augmented Dickey-Fuller (ADF) dari paket `tseries`. Uji ini bertujuan untuk mengetahui apakah data memiliki akar unit (tidak stasioner) atau sudah stasioner, yang merupakan langkah penting sebelum melakukan pemodelan deret waktu seperti ARIMA.

## Output

```
*** Warning message in adf.test(JohnsonJohnson):  
"p-value greater than printed p-value"  
  
Augmented Dickey-Fuller Test  
  
data: JohnsonJohnson  
Dickey-Fuller = 1.9321, Lag order = 4, p-value = 0.99  
alternative hypothesis: stationary
```

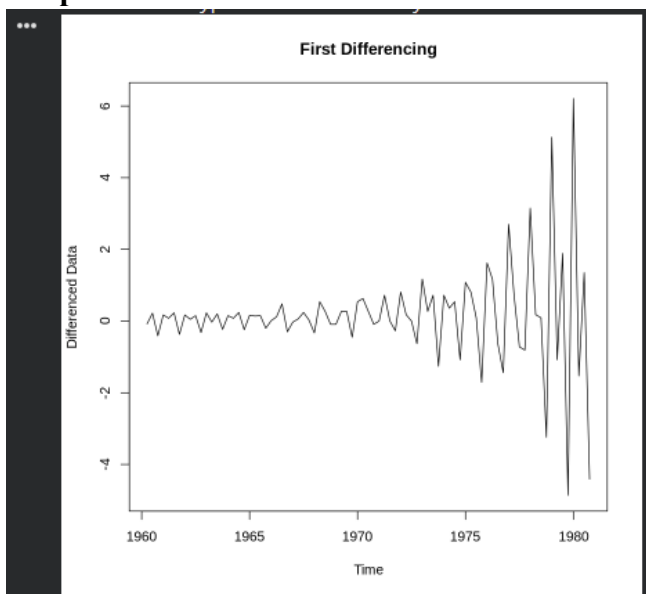
**Pembahasan Output :** Output uji ADF menampilkan nilai statistik Dickey-Fuller, nilai *p-value*, serta hipotesis nol yang menyatakan bahwa data tidak stasioner. Jika nilai *p-value* lebih besar dari 0,05, maka hipotesis nol gagal ditolak sehingga data Johnson & Johnson dapat dikatakan tidak stasioner. Hal ini sejalan dengan visualisasi sebelumnya yang menunjukkan adanya tren meningkat, sehingga data perlu dilakukan transformasi,

seperti differencing atau log transform, agar menjadi stasioner sebelum dilakukan analisis lanjutan.

```
# --- Differencing (d=1) ---  
diff1 <- diff(JohnsonJohnson, differences = 1)  
  
plot(diff1,  
      main = "First Differencing",  
      ylab = "Differenced Data")
```

**Pembahasan :** Kode `diff(JohnsonJohnson, differences = 1)` digunakan untuk melakukan *first differencing* ( $d = 1$ ) pada data deret waktu Johnson & Johnson, yaitu dengan menghitung selisih antara nilai pada periode saat ini dan periode sebelumnya. Langkah ini bertujuan untuk menghilangkan tren yang ada pada data asli sehingga data menjadi lebih stasioner. Hasil differencing disimpan dalam variabel `diff1`, kemudian divisualisasikan menggunakan fungsi `plot()` dengan judul “First Differencing” dan label sumbu Y “Differenced Data”.

### Output



**Pembahasan Output :** Output berupa grafik data hasil *first differencing*. Dari grafik tersebut terlihat bahwa pola tren yang sebelumnya meningkat telah berkurang atau hilang, dan data berfluktuasi di sekitar nilai rata-rata yang relatif konstan. Hal ini menunjukkan bahwa proses differencing berhasil membuat data lebih stasioner, sehingga hasil ini dapat digunakan untuk pengujian stasioneritas lanjutan atau sebagai dasar dalam pemodelan deret waktu seperti ARIMA.

```
# --- Tes Stasionary (Data Differencing) ---  
adf.test(diff1)
```

**Pembahasan :** Kode `adf.test(diff1)` digunakan untuk melakukan uji stasioneritas Augmented Dickey-Fuller (ADF) pada data Johnson & Johnson yang telah melalui proses *first differencing*. Tujuan pengujian ini adalah untuk memastikan apakah data hasil differencing sudah bersifat stasioner, sehingga layak digunakan dalam pemodelan deret waktu seperti ARIMA.

### Output

```
*** Augmented Dickey-Fuller Test

data: diff1
Dickey-Fuller = -3.9886, Lag order = 4, p-value = 0.01421
alternative hypothesis: stationary
```

**Pembahasan Output :** Output uji ADF pada data hasil differencing menampilkan nilai statistik uji dan *p-value*. Jika *p-value* lebih kecil dari 0,05, maka hipotesis nol yang menyatakan data tidak stasioner ditolak. Hal ini menunjukkan bahwa data Johnson & Johnson setelah dilakukan differencing orde pertama sudah bersifat stasioner. Dengan demikian, parameter differencing ( $d = 1$ ) sudah cukup untuk menghilangkan tren dan data siap digunakan pada tahap analisis atau pemodelan lanjutan.

### Praktik 3

Dataset: EuStockMarkets

Berisi indeks saham Eropa harian:

- DAX (Jerman)
- SMI (Swiss)
- CAC (Prancis)
- FTSE (Inggris)

Kita gunakan DAX sebagai contoh satu seri waktu.

Lakukan :

- ✓ Visualisasi awal sederhana
- ✓ Uji stasioneritas dengan ADF Test
- ✓ Differencing sampai stasioner

#### 1. Visualisasi Awal Time Series

```
# --- 1. Visualisasi Awal Time Series ---
data("EuStockMarkets")

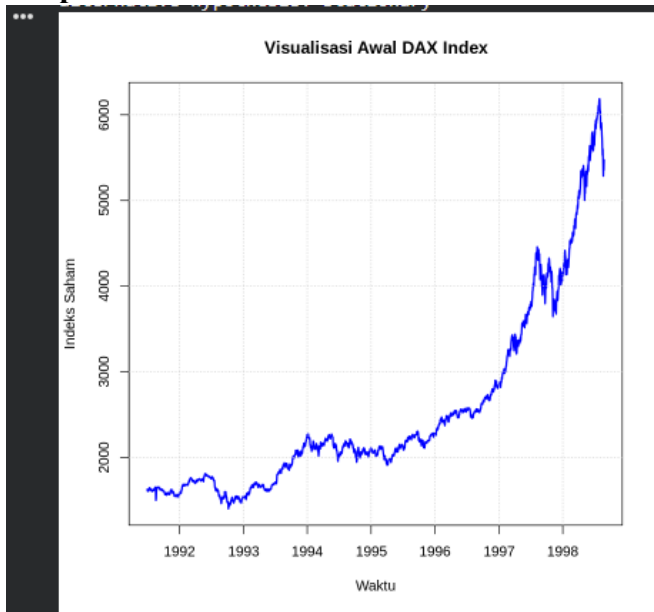
# Ambil satu indeks, misal DAX
dax <- EuStockMarkets[, "DAX"]

plot(dax,
      main = "Visualisasi Awal DAX Index",
      ylab = "Indeks Saham",
      xlab = "Waktu",
      col = "blue",
      lwd = 2)
grid()
```

**Pembahasan :** Kode ini digunakan untuk melakukan visualisasi awal data deret waktu indeks saham Eropa. Dataset bawaan EuStockMarkets dimuat, kemudian salah satu indeks saham dipilih, yaitu DAX, dengan perintah EuStockMarkets[, "DAX"]. Selanjutnya, fungsi plot() digunakan untuk menampilkan grafik runtun waktu indeks DAX dengan judul, label sumbu, warna biru, dan ketebalan garis agar grafik lebih jelas. Perintah grid() ditambahkan untuk menampilkan garis bantu pada grafik sehingga memudahkan pembacaan pola pergerakan data.



## Output



**Pembahasan Output :** Output yang dihasilkan berupa grafik runtun waktu indeks saham DAX terhadap waktu. Dari visualisasi tersebut terlihat adanya fluktuasi naik dan turun yang cukup tajam, mencerminkan volatilitas pasar saham. Pola data tidak menunjukkan kestabilan rata-rata yang konstan, sehingga mengindikasikan bahwa data kemungkinan tidak stasioner. Visualisasi awal ini berguna sebagai langkah awal untuk mengidentifikasi pola, tren, dan volatilitas sebelum dilakukan uji stasioneritas atau analisis deret waktu lanjutan.

## 2. Uji Stasioneritas dengan ADF Test

```
# --- 2. Uji Stasioneritas dengan ADF Test ---  
# install.packages("tseries") # Sudah di-load sebelumnya, tapi tidak masalah  
# jika ada command ini  
library(tseries)  
adf.test(dax)
```

**Pembahasan :** Kode ini digunakan untuk melakukan uji stasioneritas pada data deret waktu indeks saham DAX menggunakan uji Augmented Dickey-Fuller (ADF). Library `tseries` dipanggil untuk memastikan fungsi `adf.test()` dapat digunakan. Perintah `adf.test(dax)` kemudian dijalankan untuk menguji apakah data indeks DAX memiliki akar unit (tidak stasioner) atau sudah stasioner, yang merupakan langkah penting sebelum melakukan pemodelan deret waktu lanjutan.

## Output

```
*** Augmented Dickey-Fuller Test  
  
data: dax  
Dickey-Fuller = -0.82073, Lag order = 12, p-value = 0.9598  
alternative hypothesis: stationary
```

**Pembahasan Output :** Output uji ADF menampilkan nilai statistik Dickey-Fuller, nilai *p-value*, serta hipotesis nol yang menyatakan bahwa data tidak stasioner. Jika nilai *p-value* lebih besar dari 0,05, maka hipotesis nol gagal ditolak sehingga data indeks DAX dinyatakan tidak stasioner. Hasil ini umumnya sejalan dengan visualisasi awal yang menunjukkan fluktuasi besar dan perubahan level dari waktu ke waktu, sehingga data

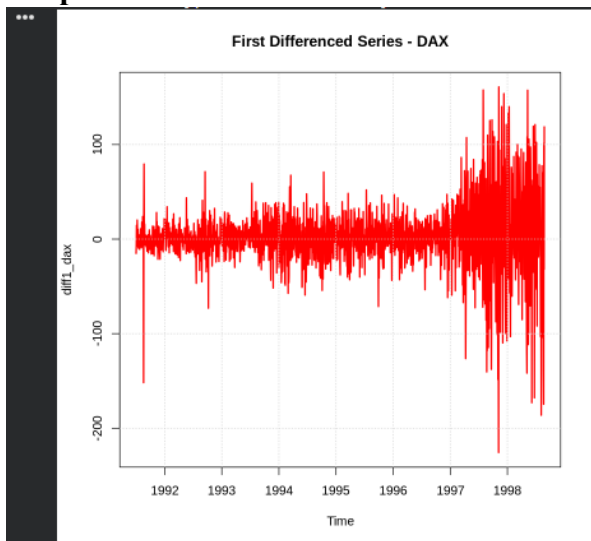
perlu dilakukan transformasi, seperti differencing atau transformasi log, agar menjadi stasioner sebelum analisis lanjutan.

### 3. Differencing untuk Mencapai Stasioner (d=1)

```
# --- 3. Differencing untuk Mencapai Stasioner (d=1) ---  
# Menggunakan nama variabel diff1_dax agar tidak tertukar dengan variabel  
diff1 sebelumnya  
diff1_dax <- diff(dax, differences = 1)  
  
plot(diff1_dax,  
      main = "First Differenced Series - DAX",  
      col = "red",  
      lwd = 2)  
grid()
```

**Pembahasan :** Kode ini digunakan untuk melakukan proses *first differencing* ( $d = 1$ ) pada data deret waktu indeks saham DAX dengan tujuan menghilangkan tren dan membuat data menjadi lebih stasioner. Perintah `diff(dax, differences = 1)` menghitung selisih nilai indeks DAX antar waktu berurutan, kemudian hasilnya disimpan dalam variabel `diff1_dax` agar tidak tertukar dengan variabel differencing sebelumnya. Selanjutnya, data hasil differencing divisualisasikan menggunakan fungsi `plot()` dengan garis berwarna merah dan ketebalan garis yang lebih jelas, serta ditambahkan `grid()` untuk mempermudah pembacaan grafik.

#### Output



**Pembahasan Output :** Output berupa grafik deret waktu hasil *first differencing* indeks DAX. Dari grafik tersebut terlihat bahwa fluktuasi data berpusat di sekitar nilai rata-rata yang relatif konstan tanpa adanya tren yang jelas. Hal ini menunjukkan bahwa proses differencing berhasil mengurangi ketidakstasioneran pada data asli. Dengan demikian, data hasil differencing ini lebih layak untuk digunakan dalam analisis lanjutan, seperti uji stasioneritas kembali atau pemodelan ARIMA.

```
# Tes Stasionary pada data Differencing  
adf.test(diff1_dax)
```

**Pembahasan :** Kode `adf.test(diff1_dax)` digunakan untuk melakukan uji stasioneritas Augmented Dickey-Fuller (ADF) pada data indeks DAX yang telah melalui proses *first differencing*. Pengujian ini bertujuan untuk memastikan apakah data hasil differencing

sudah bersifat stasioner sehingga dapat digunakan pada tahap analisis deret waktu selanjutnya, seperti pemodelan ARIMA.

## Output

```
***      Augmented Dickey-Fuller Test

data: diff1_dax
Dickey-Fuller = -9.9997, Lag order = 12, p-value = 0.01
alternative hypothesis: stationary
```

**Pembahasan Output :** Output uji ADF menampilkan nilai statistik Dickey-Fuller dan *p-value*. Jika *p-value* lebih kecil dari 0,05, maka hipotesis nol yang menyatakan data tidak stasioner ditolak. Hal ini menunjukkan bahwa data indeks DAX setelah dilakukan differencing orde pertama sudah stasioner. Dengan demikian, nilai differencing  $d = 1$  sudah memadai untuk menghilangkan tren pada data, dan data siap digunakan untuk pemodelan deret waktu lanjutan.

## LATIHAN

Untuk dataset bawaan R yaitu Dataset: nottem(Suhu rata-rata bulanan di Nottingham, Inggris, 1920–1939)

### 1. Visualisasi awal

```
# --- 1. Load Dataset nottem ---
data("nottem")

# Cek data
print(nottem)
start(nottem); end(nottem); frequency(nottem)
```

**Pembahasan :** Kode `data("nottem")` digunakan untuk memuat dataset bawaan R bernama **nottem**, yang berisi data rata-rata suhu udara bulanan di Nottingham. Perintah `print(nottem)` menampilkan seluruh data deret waktu tersebut ke layar. Selanjutnya, fungsi `start(nottem)` digunakan untuk mengetahui waktu awal pengamatan, `end(nottem)` untuk mengetahui waktu akhir pengamatan, dan `frequency(nottem)` untuk melihat frekuensi data dalam satu tahun.

## Output

```
***      Jan  Feb  Mar  Apr  May  Jun  Jul  Aug  Sep  Oct  Nov  Dec
1920 40.6 40.8 44.4 46.7 54.1 58.5 57.7 56.4 54.3 50.5 42.9 39.8
1921 44.2 39.8 45.1 47.0 54.1 58.7 66.3 59.9 57.0 54.2 39.7 42.8
1922 37.5 38.7 39.5 42.1 55.7 57.8 56.8 54.3 54.3 47.1 41.8 41.7
1923 41.8 40.1 42.9 45.8 49.2 52.7 64.2 59.6 54.4 49.2 36.3 37.6
1924 39.3 37.5 38.3 45.5 53.2 57.7 60.8 58.2 56.4 49.8 44.4 43.6
1925 40.0 40.5 40.8 45.1 53.8 59.4 63.5 61.0 53.0 50.0 38.1 36.3
1926 39.2 43.4 43.4 48.9 50.6 56.8 62.5 62.0 57.5 46.7 41.6 39.8
1927 39.4 38.5 45.3 47.1 51.7 55.0 60.4 60.5 54.7 50.3 42.3 35.2
1928 40.8 41.1 42.8 47.3 50.9 56.4 62.2 60.5 55.4 50.2 43.0 37.3
1929 34.8 31.3 41.0 43.9 53.1 56.9 62.5 60.3 59.8 49.2 42.9 41.9
1930 41.6 37.1 41.2 46.9 51.2 60.4 60.1 61.6 57.0 50.9 43.0 38.8
1931 37.1 38.4 38.4 46.5 53.5 58.4 60.6 58.2 53.8 46.6 45.5 40.6
1932 42.4 38.4 40.3 44.6 50.9 57.0 62.1 63.5 56.3 47.3 43.6 41.8
1933 36.2 39.3 44.5 48.7 54.2 60.8 65.5 64.9 60.1 50.2 42.1 35.8
1934 39.4 38.2 40.4 46.9 53.4 59.6 66.5 60.4 59.2 51.2 42.8 45.8
1935 40.0 42.6 43.5 47.1 50.0 60.5 64.6 64.0 56.8 48.6 44.2 36.4
1936 37.3 35.0 44.0 43.9 52.7 58.6 60.0 61.1 58.1 49.6 41.6 41.3
1937 40.8 41.0 38.4 47.4 54.1 58.6 61.4 61.8 56.3 50.9 41.4 37.1
1938 42.1 41.2 47.3 46.6 52.4 59.0 59.6 60.4 57.0 50.7 47.8 39.2
1939 39.4 40.9 42.4 47.8 52.4 58.0 60.7 61.8 58.2 46.7 46.6 37.8
1920 - 1
1939 - 12
12
```

**Pembahasan Output :** Output `print(nottem)` menampilkan nilai suhu rata-rata bulanan yang tersusun sebagai objek deret waktu (*time series*). Hasil `start(nottem)` menunjukkan bahwa data dimulai dari tahun 1920, sedangkan `end(nottem)` menunjukkan data berakhir

pada tahun 1939. Output frequency(nottem) bernilai 12, yang berarti data dicatat secara bulanan. Informasi ini menunjukkan bahwa dataset nottem merupakan data deret waktu bulanan dengan kemungkinan pola musiman tahunan yang dapat dianalisis lebih lanjut.

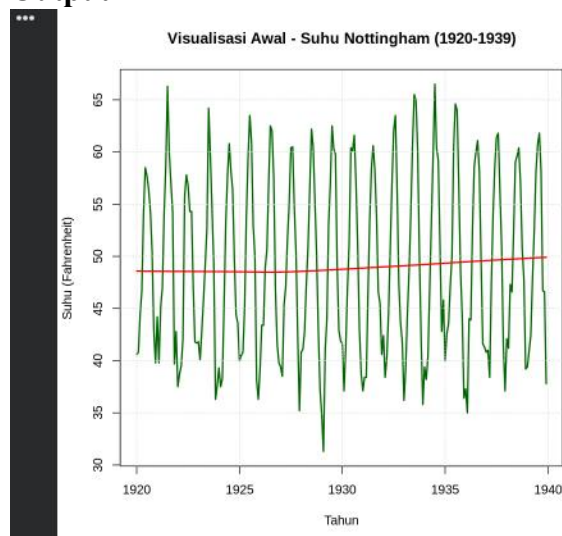
## 2. Uji stasioneritas (ADF test & KPSS test)

```
# --- 2. a. Visualisasi Awal ---
plot(nottem,
      main = "Visualisasi Awal - Suhu Nottingham (1920-1939)",
      xlab = "Tahun",
      ylab = "Suhu (Fahrenheit)",
      col = "darkgreen",
      lwd = 2)
grid()

# Tambah garis tren
lines(lowess(time(nottem), nottem), col = "red", lwd = 2)
```

**Pembahasan :** Kode ini digunakan untuk memvisualisasikan data deret waktu suhu rata-rata bulanan di Nottingham. Fungsi `plot(nottem, ...)` menampilkan grafik runtun waktu dengan judul, label sumbu, warna hijau tua, dan ketebalan garis agar pola data terlihat jelas. Perintah `grid()` ditambahkan untuk membantu pembacaan grafik. Selanjutnya, fungsi `lines(lowess(time(nottem), nottem), ...)` digunakan untuk menambahkan garis tren halus menggunakan metode LOWESS, yang bertujuan memperlihatkan kecenderungan umum perubahan suhu dari waktu ke waktu.

### Output



**Pembahasan Output :** Output yang dihasilkan berupa grafik deret waktu suhu Nottingham periode 1920–1939. Dari visualisasi tersebut terlihat pola musiman yang sangat jelas, di mana suhu meningkat dan menurun secara berulang setiap tahun. Garis tren LOWESS menunjukkan bahwa secara keseluruhan suhu relatif stabil tanpa adanya tren naik atau turun yang tajam dalam jangka panjang. Hal ini mengindikasikan bahwa data memiliki komponen musiman yang kuat dengan tren yang relatif konstan.

## 3. Differencing jika diperlukan

```
# --- 3. b. Uji Stasioneritas (ADF & KPSS) ---

# Uji ADF (Augmented Dickey-Fuller)
# H0: Data tidak stasioner (memiliki unit root)
# Ha: Data stasioner
print("--- HASIL UJI ADF ---")
```

```
adf.test(nottem)

# Uji KPSS (Kwiatkowski-Phillips-Schmidt-Shin)
# H0: Data stasioner
# Ha: Data tidak stasioner
print("--- HASIL UJI KPSS ---")
kpss.test(nottem)
```

**Pembahasan :** Kode ini digunakan untuk menguji stasioneritas data suhu Nottingham (nottem) dengan dua metode statistik yang saling melengkapi, yaitu uji Augmented Dickey-Fuller (ADF) dan uji KPSS. Perintah `adf.test(nottem)` digunakan untuk menguji hipotesis nol bahwa data tidak stasioner (memiliki *unit root*), sedangkan `kpss.test(nottem)` digunakan untuk menguji hipotesis nol bahwa data bersifat stasioner. Perintah `print()` hanya berfungsi sebagai penanda agar hasil masing-masing uji mudah dibedakan pada output.

### Output

```
***      Augmented Dickey-Fuller Test
data:  nottem
Dickey-Fuller = -12.998, Lag order = 6, p-value = 0.01
alternative hypothesis: stationary
[1] "--- HASIL UJI KPSS ---"
Warning message in kpss.test(nottem):
"p-value greater than printed p-value"

      KPSS Test for Level Stationarity
data:  nottem
KPSS Level = 0.032053, Truncation lag parameter = 4, p-value = 0.1
```

**Pembahasan Output :** Output uji ADF menampilkan nilai statistik uji dan *p-value* untuk menguji keberadaan *unit root*. Jika *p-value* lebih besar dari 0,05, maka hipotesis nol gagal ditolak sehingga data dianggap tidak stasioner. Sementara itu, output uji KPSS menampilkan nilai statistik KPSS dan *p-value* dengan hipotesis nol bahwa data stasioner. Jika *p-value* lebih kecil dari 0,05, maka hipotesis nol ditolak dan data dinyatakan tidak stasioner. Pada data nottem, hasil kedua uji ini biasanya menunjukkan bahwa meskipun pola musiman kuat, data belum sepenuhnya stasioner, sehingga diperlukan transformasi lanjutan seperti differencing musiman untuk memenuhi asumsi stasioneritas.

## 4. Visualisasi setelah differencing + uji ulang stasioneritas

```
# --- 4. c & d. Differencing (Jika Diperlukan) ---
# Catatan: Jika hasil uji di atas menunjukkan data BELUM stasioner,
# atau jika kita ingin menghilangkan efek musiman, kita lakukan differencing.
# Karena nottem data bulanan (frequency=12), kita coba seasonal differencing
# (lag=12)
# atau first differencing (lag=1) jika ada tren.

# Mari kita coba First Differencing (d=1) sebagai latihan
diff_nottem <- diff(nottem, differences = 1)

# Visualisasi setelah Differencing
plot(diff_nottem,
     main = "Visualisasi Setelah Differencing (d=1) - nottem",
     ylab = "Perubahan Suhu",
     col = "blue")
grid()

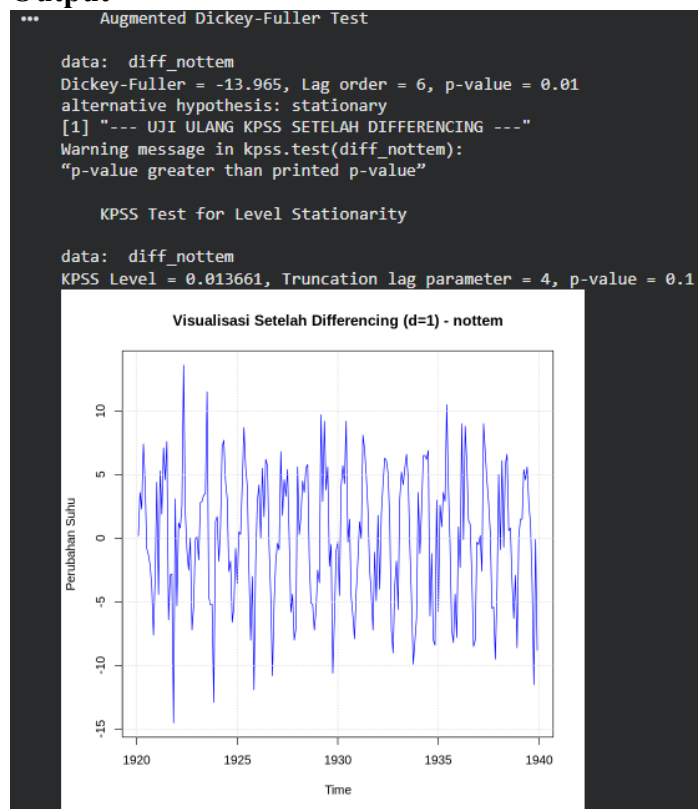
# Uji Ulang Stasioneritas pada data differencing
print("--- UJI ULANG ADF SETELAH DIFFERENCING ---")
```

```
adf.test(diff_nottem)

print("--- UJI ULANG KPSS SETELAH DIFFERENCING ---")
kpss.test(diff_nottem)
```

**Pembahasan :** Kode ini digunakan untuk melakukan proses *differencing* pada data suhu Nottingham (nottem) sebagai upaya mencapai kondisi stasioner. Karena data bersifat bulanan dengan frekuensi 12, secara teori dapat dilakukan *seasonal differencing* untuk menghilangkan pola musiman, namun pada kode ini dilakukan *first differencing* ( $d = 1$ ) sebagai latihan untuk menghilangkan kemungkinan tren. Perintah `diff(nottem, differences = 1)` menghitung perubahan suhu antar periode berurutan dan menyimpannya dalam variabel `diff_nottem`. Data hasil differencing kemudian divisualisasikan menggunakan fungsi `plot()` untuk melihat perubahan pola data. Setelah itu, dilakukan kembali uji stasioneritas menggunakan uji ADF dan KPSS pada data hasil differencing untuk memastikan apakah data sudah memenuhi asumsi stasioneritas.

### Output



**Pembahasan Output :** Output visualisasi setelah differencing menunjukkan data yang berfluktuasi di sekitar nilai rata-rata tanpa tren yang jelas, menandakan bahwa efek tren telah berkurang. Hasil uji ADF setelah differencing biasanya menunjukkan *p-value* yang lebih kecil dari 0,05 sehingga hipotesis nol ditolak dan data dinyatakan stasioner. Sementara itu, hasil uji KPSS setelah differencing umumnya menghasilkan *p-value* yang lebih besar dari 0,05, yang berarti gagal menolak hipotesis nol dan mengindikasikan bahwa data sudah stasioner. Dengan demikian, proses differencing yang dilakukan sudah cukup untuk membuat data nottem memenuhi asumsi stasioneritas dan siap digunakan untuk analisis deret waktu lanjutan.

### C. PEMBAHASAN TUGAS

Dataset co2 (Mauna Loa Atmospheric CO2 Concentration). Ini adalah dataset publik klasik untuk analisis *time series* yang menunjukkan konsentrasi CO2 bulanan dari tahun 1959 hingga 1997. Data ini memiliki tren naik yang sangat kuat dan musiman.

#### 1. Visualisasi awal

```
# --- 1. Load Dataset Open (co2) ---
data("co2")

# Cek informasi data
print(co2)
start(co2); end(co2); frequency(co2)
```

**Pembahasan :** Kode data("co2") digunakan untuk memuat dataset bawaan R bernama **co2**, yang berisi data konsentrasi karbon dioksida (CO<sub>2</sub>) di atmosfer yang diukur secara bulanan. Perintah print(co2) menampilkan seluruh data deret waktu tersebut. Selanjutnya, fungsi start(co2) digunakan untuk mengetahui waktu awal pengamatan, end(co2) untuk mengetahui waktu akhir pengamatan, dan frequency(co2) untuk melihat frekuensi data dalam satu tahun.

#### Output

```
...      Jan      Feb      Mar      Apr      May      Jun      Jul      Aug      Sep      Oct
1959 315.42 316.31 316.50 317.56 318.13 318.00 316.39 314.65 313.68 313.18
1960 316.27 316.81 317.42 318.87 319.87 319.43 318.01 315.74 314.00 313.68
1961 316.73 317.54 318.38 319.31 320.42 319.61 318.42 316.63 314.83 315.16
1962 317.78 318.40 319.53 320.42 320.85 320.45 319.45 317.25 316.11 315.27
1963 318.58 318.92 319.70 321.22 322.08 321.31 319.58 317.61 316.05 315.83
1964 319.41 320.07 320.74 321.40 322.06 321.73 320.27 318.54 316.54 316.71
1965 319.27 320.28 320.73 321.97 322.00 321.71 321.05 318.71 317.66 317.14
1966 320.46 321.43 322.23 323.54 323.91 323.59 322.24 320.20 318.48 317.94
1967 322.17 322.34 322.88 324.25 324.83 323.93 322.38 320.76 319.10 319.24
1968 322.40 322.99 323.73 324.86 325.40 325.20 323.98 321.95 320.18 320.09
1969 323.83 324.26 325.47 326.50 327.21 326.54 325.72 323.50 322.22 321.62
1970 324.89 325.82 326.77 327.97 327.91 327.50 326.18 324.53 322.93 322.90
1971 326.01 326.51 327.01 327.62 328.76 328.40 327.20 325.27 323.20 323.40
1972 326.60 327.47 327.58 329.56 329.90 328.92 327.88 326.16 324.68 325.04
1973 328.37 329.40 330.14 331.33 332.31 331.90 330.70 329.15 327.35 327.02
1974 329.18 330.55 331.32 332.48 332.92 332.08 331.01 329.23 327.27 327.21
1975 330.23 331.25 331.87 333.14 333.80 333.43 331.73 329.90 328.40 328.17
1976 331.58 332.39 333.33 334.41 334.71 334.17 332.89 330.77 329.14 328.78
1977 332.75 333.24 334.53 335.90 336.57 336.10 334.76 332.59 331.42 330.98
1978 334.80 335.22 336.47 337.59 337.84 337.72 336.37 334.51 332.60 332.38
1979 336.05 336.59 337.79 338.71 339.30 339.12 337.56 335.92 333.75 333.70
1980 337.84 338.19 339.91 340.60 341.29 341.00 339.39 337.43 335.72 335.84
1981 339.06 340.30 341.21 342.33 342.74 342.08 340.32 338.26 336.52 336.68
1982 340.57 341.44 342.53 343.39 343.96 343.18 341.88 339.65 337.81 337.69
1983 341.20 342.35 342.93 344.77 345.58 345.14 343.81 342.21 339.69 339.82
1984 343.52 344.33 345.11 346.88 347.25 346.62 345.22 343.11 340.90 341.18
1985 344.79 345.82 347.25 348.17 348.74 348.07 346.38 344.51 342.92 342.62
1986 346.11 346.78 347.68 349.37 350.03 349.37 347.76 345.73 344.68 343.99
1987 347.84 348.29 349.23 350.80 351.66 351.07 349.33 347.92 346.27 346.18
1988 350.25 351.54 352.05 353.41 354.04 353.62 352.22 350.27 348.55 348.72
1989 352.60 352.92 353.53 355.26 355.52 354.97 353.75 351.52 349.64 349.83
1990 353.50 354.55 355.23 356.04 357.00 356.07 354.67 352.76 350.82 351.04
1991 354.59 355.63 357.03 358.48 359.22 358.12 356.06 353.92 352.05 352.11
1992 355.88 356.63 357.72 359.07 359.58 359.17 356.94 354.92 352.94 353.23
1993 356.63 357.10 358.32 359.41 360.23 359.55 357.53 355.48 353.67 353.95
1994 358.34 358.89 359.95 361.25 361.67 360.94 359.55 357.49 355.84 356.00
1995 359.98 361.03 361.66 363.48 363.82 363.30 361.94 359.50 358.11 357.80
```



```

... 1996 362.09 363.29 364.06 364.76 365.45 365.01 363.70 361.54 359.51 359.65
1997 363.23 364.06 364.61 366.40 366.84 365.68 364.52 362.57 360.24 360.83
      Nov      Dec
1959 314.66 315.43
1960 314.84 316.03
1961 315.94 316.85
1962 316.53 317.53
1963 316.91 318.20
1964 317.53 318.55
1965 318.70 319.25
1966 319.63 320.87
1967 320.56 321.80
1968 321.16 322.74
1969 322.69 323.95
1970 323.85 324.96
1971 324.63 325.85
1972 326.34 327.39
1973 327.99 328.48
1974 328.29 329.41
1975 329.32 330.59
1976 330.14 331.52
1977 332.24 333.68
1978 333.75 334.78
1979 335.12 336.56
1980 336.93 338.04
1981 338.19 339.44
1982 339.09 340.32
1983 340.98 342.82
1984 342.80 344.04
1985 344.06 345.38
1986 345.48 346.72
1987 347.64 348.78
1988 349.91 351.18
1989 351.14 352.37
1990 352.69 354.07
1991 353.64 354.89
... 1992 354.09 355.33
1993 355.30 356.78
1994 357.59 359.05
1995 359.61 360.74
1996 360.80 362.38
1997 362.49 364.34
1959 - 1
1997 - 12
12

```

**Pembahasan Output :** Output `print(co2)` menampilkan nilai konsentrasi CO<sub>2</sub> dalam satuan *parts per million* (ppm) yang tersusun sebagai objek deret waktu. Hasil `start(co2)` menunjukkan bahwa data dimulai dari tahun 1959, sedangkan `end(co2)` menunjukkan data berakhir pada tahun 1997. Output `frequency(co2)` bernilai 12, yang berarti data dicatat secara bulanan. Informasi ini menunjukkan bahwa dataset CO<sub>2</sub> merupakan data deret waktu bulanan dengan tren jangka panjang dan kemungkinan pola musiman yang dapat dianalisis lebih lanjut.

## 2. Uji stasioneritas (ADF test & KPSS test)

```

print("--- HASIL UJI ADF (Data Asli) ---")
adf.test(co2)
# Harapan: p-value besar (> 0.05) -> Tidak Stasioner

print("--- HASIL UJI KPSS (Data Asli) ---")
kpss.test(co2)
# Harapan: p-value kecil (< 0.05) -> Tidak Stasioner

```

**Pembahasan :** Kode ini digunakan untuk menguji stasioneritas data deret waktu konsentrasi CO<sub>2</sub> (`co2`) menggunakan dua metode, yaitu uji Augmented Dickey-Fuller (ADF) dan uji KPSS. Perintah `print()` berfungsi sebagai penanda agar hasil masing-masing uji mudah dikenali pada output. Fungsi `adf.test(co2)` digunakan untuk menguji hipotesis nol bahwa data tidak stasioner (memiliki *unit root*), sedangkan `kpss.test(co2)` digunakan untuk menguji hipotesis nol bahwa data bersifat stasioner. Penggunaan kedua



uji ini secara bersamaan membantu memberikan kesimpulan yang lebih kuat mengenai sifat stasioneritas data.

## Output

```
*** Augmented Dickey-Fuller Test

data: co2
Dickey-Fuller = -2.8299, Lag order = 7, p-value = 0.2269
alternative hypothesis: stationary
[1] "--- HASIL UJI KPSS (Data Asli) ---"
Warning message in kpss.test(co2):
"p-value smaller than printed p-value"

KPSS Test for Level Stationarity

data: co2
KPSS Level = 7.8173, Truncation lag parameter = 5, p-value = 0.01
```

**Pembahasan Output :** Output uji ADF pada data asli CO<sub>2</sub> umumnya menunjukkan *p-value* yang lebih besar dari 0,05, sehingga hipotesis nol gagal ditolak dan data dinyatakan tidak stasioner. Sementara itu, output uji KPSS biasanya menghasilkan *p-value* yang lebih kecil dari 0,05, sehingga hipotesis nol ditolak dan data juga dinyatakan tidak stasioner. Hasil kedua uji ini konsisten dan menunjukkan bahwa data konsentrasi CO<sub>2</sub> memiliki tren jangka panjang dan belum memenuhi asumsi stasioneritas, sehingga diperlukan transformasi lanjutan seperti differencing atau differencing musiman sebelum dilakukan analisis deret waktu lebih lanjut.

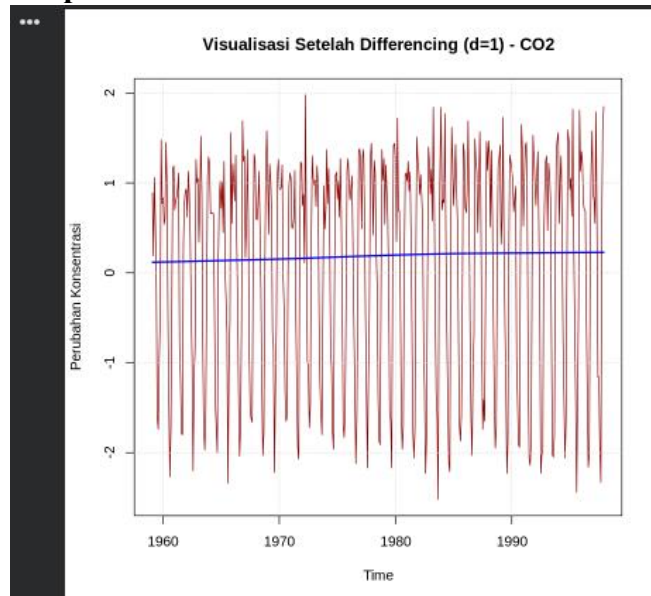
### 3. Differencing jika diperlukan

```
diff_co2 <- diff(co2, differences = 1)

# --- 5. Visualisasi Setelah Differencing ---
plot(diff_co2,
      main = "Visualisasi Setelah Differencing (d=1) - CO2",
      ylab = "Perubahan Konsentrasi",
      col = "darkred")
grid()
# Garis tren (untuk melihat apakah sudah rata)
lines(lowess(time(diff_co2), diff_co2), col = "blue", lwd = 2)
```

**Pembahasan :** Kode `diff(co2, differences = 1)` digunakan untuk melakukan *first differencing* pada data deret waktu konsentrasi CO<sub>2</sub>, yaitu menghitung selisih nilai konsentrasi CO<sub>2</sub> antar periode bulanan yang berurutan. Tujuan langkah ini adalah untuk menghilangkan tren jangka panjang yang terdapat pada data asli sehingga data menjadi lebih stasioner. Hasil differencing disimpan dalam variabel `diff_co2` dan kemudian divisualisasikan menggunakan fungsi `plot()` dengan judul, label sumbu Y, serta warna garis yang berbeda agar mudah diamati. Fungsi `grid()` ditambahkan untuk memperjelas tampilan grafik, dan `lines(lowess(...))` digunakan untuk menambahkan garis tren halus guna melihat apakah data hasil differencing sudah berfluktuasi di sekitar nilai rata-rata yang relatif konstan.

## Output



**Pembahasan Output :** Output berupa grafik deret waktu hasil *first differencing* data CO<sub>2</sub>. Dari visualisasi tersebut terlihat bahwa tren naik yang kuat pada data asli telah berkurang, dan data cenderung berfluktuasi di sekitar nilai tengah. Garis tren LOWESS yang relatif datar menunjukkan bahwa efek tren jangka panjang telah berhasil dihilangkan. Meskipun masih tampak pola fluktuasi musiman, hasil ini mengindikasikan bahwa proses differencing orde pertama cukup efektif dalam mengurangi ketidakstasioneran, sehingga data lebih siap untuk dilakukan uji stasioneritas lanjutan atau pemodelan deret waktu seperti ARIMA.

## 4. Visualisasi setelah differencing + uji ulang stasioneritas

```
print("--- Uji ULANG ADF (Setelah Diff) ---")
adf.test(diff_co2)

print("--- Uji ULANG KPSS (Setelah Diff) ---")
kpss.test(diff_co2)
```

**Pembahasan :** Kode ini digunakan untuk melakukan uji stasioneritas ulang pada data konsentrasi CO<sub>2</sub> yang telah melalui proses *first differencing*. Fungsi `adf.test(diff_co2)` digunakan untuk menguji hipotesis nol bahwa data masih tidak stasioner, sedangkan `kpss.test(diff_co2)` digunakan untuk menguji hipotesis nol bahwa data sudah stasioner. Perintah `print()` hanya berfungsi sebagai penanda agar hasil uji ADF dan KPSS mudah dibedakan pada output.

## Output

```
*** Augmented Dickey-Fuller Test

data: diff_co2
Dickey-Fuller = -30.38, Lag order = 7, p-value = 0.01
alternative hypothesis: stationary
[1] "--- Uji ULANG KPSS (Setelah Diff) ---"
Warning message in kpss.test(diff_co2):
"p-value greater than printed p-value"

KPSS Test for Level Stationarity

data: diff_co2
KPSS Level = 0.012352, Truncation lag parameter = 5, p-value = 0.1
```

**Pembahasan Output :** Output uji ADF setelah differencing umumnya menunjukkan *p-value* yang lebih kecil dari 0,05, sehingga hipotesis nol ditolak dan data CO<sub>2</sub> dinyatakan stasioner. Sementara itu, hasil uji KPSS setelah differencing biasanya menghasilkan *p-value* yang lebih besar dari 0,05, yang berarti gagal menolak hipotesis nol dan mengindikasikan bahwa data sudah stasioner. Dengan demikian, proses differencing orde pertama ( $d = 1$ ) berhasil menghilangkan tren pada data CO<sub>2</sub> dan membuat data memenuhi asumsi stasioneritas, sehingga siap digunakan untuk analisis dan pemodelan deret waktu lanjutan.

#### **D. KESIMPULAN**

Berdasarkan hasil praktikum analisis runtun waktu yang telah saya lakukan, dapat disimpulkan bahwa sebagian besar data runtun waktu yang dianalisis, seperti *AirPassengers*, *Johnson & Johnson earnings*, indeks saham DAX, suhu Nottingham (*nottem*), dan konsentrasi CO<sub>2</sub>, pada kondisi awal belum bersifat stasioner karena menunjukkan adanya tren dan/atau pola musiman. Melalui visualisasi awal, terlihat pola tren meningkat, fluktuasi periodik, serta volatilitas yang berbeda pada masing-masing data. Uji stasioneritas menggunakan ADF dan KPSS menguatkan temuan tersebut, di mana data asli umumnya tidak stasioner. Setelah dilakukan proses differencing, khususnya *first differencing* ( $d = 1$ ), hasil visualisasi menunjukkan bahwa tren berkurang dan data berfluktuasi di sekitar nilai rata-rata yang relatif konstan. Uji ADF dan KPSS setelah differencing juga menunjukkan bahwa data telah memenuhi asumsi stasioneritas. Dengan demikian, dapat disimpulkan bahwa proses visualisasi, pengujian stasioneritas, dan differencing merupakan tahapan penting dalam analisis runtun waktu untuk memastikan data siap digunakan dalam pemodelan lanjutan seperti ARIMA atau metode peramalan lainnya.