Dasar *R**

Jarot Indarto[†]

Desember 2020

*Bahan utama bersumber dari "*R Workbook*, bahan pelatihan R, PERHEPI Indonesia - *the Institute of Statistical Mathematics (ISM)* Jepang, 21-23 Desember 2020".

[†]Bekerja di Kementerian Perencanaan Pembangunan Nasional/BAPPENAS; email: **indarto@bappenas.go.id** dan **j.indarto@gmail.com**; situs: **https://indarto.weebly.com**.

Daftar Isi

Catatan Pengantar

Bismillaahirrahmaanirrahiim. . .

Mendengar bahasa *R* baru dimulai pada tahun 2015, saat tugas belajar. Pihak kampus (laboratorium dan perpustakaan) sebenarnya menyediakan berbagai perangkat lunak berbayar untuk telaah data. Namun demikian, pembimbing laboratorium mendorong penggunaan perangkat lunak dan bahasa program *open source*, yang berkembang pesat dan banyak dimanfaatkan oleh akademisi. Dan, *R* adalah salah satu bahasa program yang direkomendasikan.

Seiring dengan perjalanan waktu, berbagai kesibukan mengurangi alokasi waktu untuk menelaah data, permodelan dan perangkat lunak telaah data, secara konsisten. Walaupun sesekali berusaha untuk meluangkan waktu untuk mengenal *R*, namun hal tersebut tidak berjalan sepenuh hati.

Minggu terakhir di penghujung tahun 2020 merupakan salah satu waktu yang layak untuk disyukuri. Selama 3 hari (21 - 23 Desember 2020), mempunyai kesempatan berharga untuk belajar dan menikmati *R* kembali[1]. Terutama dari bahan pelatihan ini *lah*, penulisan buku/modul ini bersumber.

"Dasar R" kiranya merupakan judul yang mewakili isi dan pendekatan buku/modul, karena memang baru mampu menyajikan substansi dasar dan sebatas memanfaatkan fungsi dasar yang sudah ada pada bahasa *R* (*R base functions*). Sehingga, seluruh proses latihan tidak menggunakan tambahan *modul* atau *package* tambahan. Selain itu, buku/modul ini hanya menggunakan satu *dataset* yang diterapkan dari awal sampai akhir untuk berbagai bahasan dan latihan telaah. Bahasan yang dicakup dalam buku ini, mulai dari proses instalasi, penyiapan *working directory*, peninjauan data, visualisasi data, sampai dengan latihan dasar regresi sederhana.

Proses telaah dan penulisan sangat dibantu oleh *R*, *RStudio* dan *R Markdown*. Dan, buku/modul dan data latihan disediakan secara bebas **di sini**.

Semoga buku/modul ini menjembatani pembaca dalam mengenal bahasa *R*, serta memotivasi pembaca untuk lebih memperdalam dan memanfaatkan *R* lebih lanjut.

Penghujung Desember 2020,

Jarot Indarto
j.indarto@gmail.com
https://indarto.weebly.com

[1]Terima kasih kepada Perhimpunan Ekonomi Pertanian Indonesia (PERHEPI) dan *the Institute of Statistical Mathematics (ISM)* Jepang yang memfasilitasi pelatihan ini.

1 Dasar Instalasi (untuk *Windows*)

Buku/modul ini diawali dengan bahasan instalasi dasar, yaitu instalasi *R*, dan juga instalasi *RStudio* jika membutuhkan *platform* tambahan untuk mempermudah proses telaah data. Selain itu, bab ini juga memberikan pilihan *platform* lain yang tersedia luas dalam menjalankan bahasa *R*.

1.1 Instalasi *R*

R harus dipasang (*install*) pada perangkat, dengan langkah sederhana berikut.

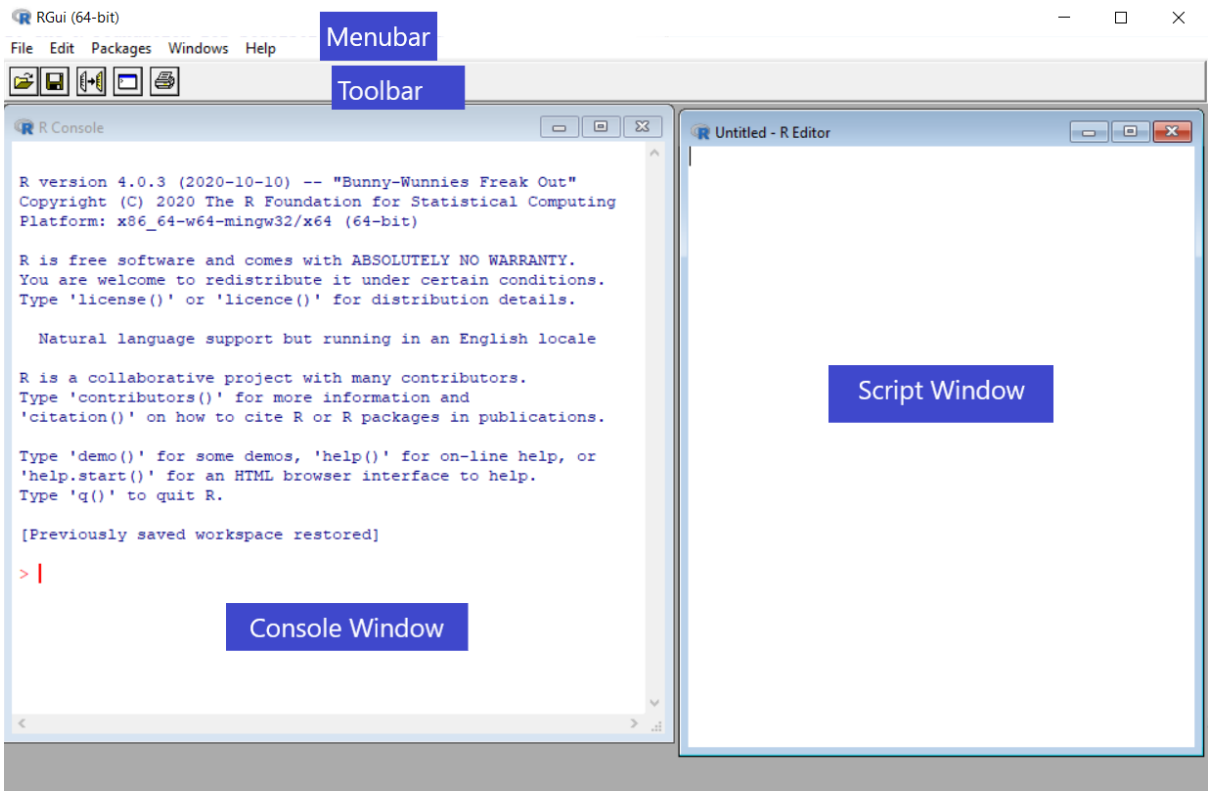
- kunjungi situs: https://cran.r-project.org/[1]
  - *mirror* Indonesia disediakan oleh BPTT di tautan berikut: https://repo.bppt.go.id/cran/.
- pilih sesuai sistem operasi (*operating system - OS*) dari perangkat:
  - *Windows* (selanjutnya akan fokus pada *OS Windows*),
  - *Linux*, atau
  - *Mac*.
- pilih *base* untuk *windows*: https://cran.r-project.org/bin/windows/base/
  - versi pada 26/12/2020: *R-4.0.3 for Windows (32/64 bit)*,
- unduh *file* instalasi.
- ikuti langkah instalasi.
- untuk sistem operasi lain, bisa disesuaikan.

R telah terpasang dalam perangkat/mesin, sehingga *R* telah siap untuk dijalankan.

> Mengapa disebut *R*? Penamaan ini sesuai dengan huruf awal dari nama pengembangnya (***R***oss Ihaka dan ***R***obert Gentlement).

[1] CRAN: *the Comprehensive R Archive Network*.

Berikut tampilan antarmuka *R*.



1.2 Instalasi *RStudio*

Pada tahap awal, pemula memerlukan wahana *platform* bantuan dalam menjalankan bahasa *R*. Salah satu pilihan *platform* bantuan adalah *RStudio*. Dan, di bawah ini adalah langkah-langkah sederhana untuk melakukan instalasi *RStudi*.

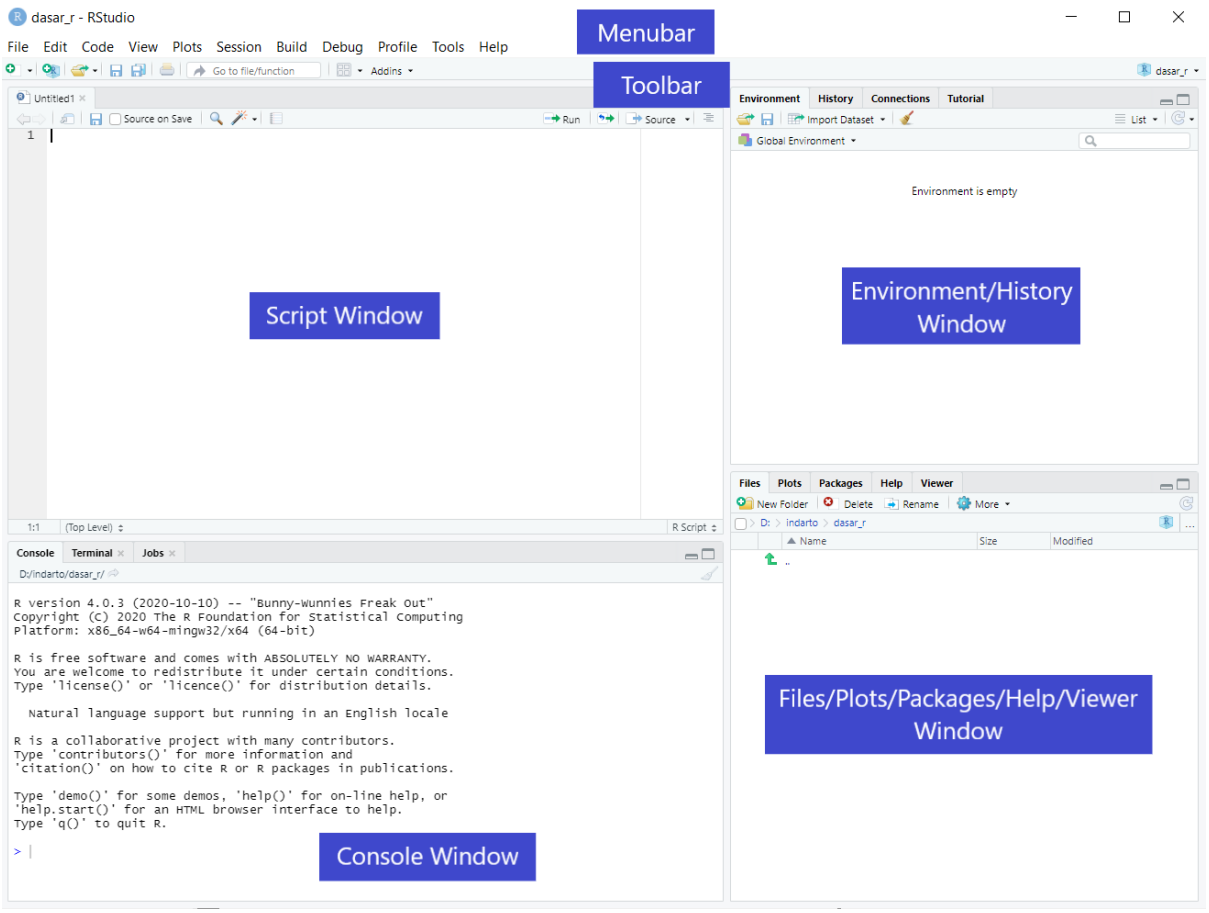
- kunjungi situs: https://www.rstudio.com.
- unduh *RStudio IDE*[2]: https://rstudio.com/products/rstudio/download/.
  - versi pada 26/12/2020: *RStudio Desktop 1.3.1093*,
  - versi ini mensyaratkan perlunya *R* versi *R3.0.1+* atau lebih.

RStudio telah terpasang dalam perangkat/mesin, serta siap membantu dan mempermudah penggunaan bahasa *R*.

> Apakah *RStudio* wajib dipasang? TIDAK, namun sebagaimana fungsinya (sebagai *IDE*), maka *RStudio* dapat mempermudah telaah data dan/atau proses pemrograman dengan *R*. Tetapi, *R* wajib terpasang dalam perangkat, agar dapat menjalankan *RStudio*.

[2] *IDE (Integrated Development Environment)*: perangkat lunak atau program sebagai wadah/lingkungan yang mempermudah pelaksanaan pemrograman.

Berikut tampilan antarmuka *RStudio*.



Bahasa *R* telah dapat dijalankan, baik langsung melalui aplikasi *R* maupun dengan bantuan *RStudio*.

1.3 Menjalankan *R* dan/atau *RStudio*

Setelah proses instalasi berhasil, saatnya membuka dan menjalankan aplikasi, baik melalui *R* langsung maupun melalui bantuan *Rstudio*.

Mari memulai perkenalan dan percakapan dengan bahasa *R*, melalui beberapa contoh sintaks sederhana di bawah ini.

- membuat (*assignment*) peubah atau *variable*.
  - *assignment* umumnya menggunakan tanda `"<-"` atau `"="`.
  - sintaks dasar membuat peubah.

```
nama_peubah <- nilai_peubah
```

- beberapa contoh pembuatan peubah.
  - membuat peubah bernama `"salam"`, yang berisi kata `"Assalaamuálaikum"`.

```
salam <- "Assalaamuálaikum"

print(salam)  # sintaks 'print()' untuk mencetak
```

```
## [1] "Assalaamuálaikum"
```

- contoh lain.

```
nama <- "Fulan"
usia <- 25
print(c(nama, "berumur", usia, "tahun."))
```

```
## [1] "Fulan"   "berumur" "25"      "tahun."
```

```
cat(nama, "berumur", usia, "tahun.")  # cat: concatenate
```

```
## Fulan berumur 25 tahun.
```

Kita telah mulai berkenalan dengan bahasa *R*. Juga, kita telah memastikan bahwa perangkat/mesin telah dapat menjalankan bahasa *R* dengan baik.

1.4 Pilihan Lain

Selain langsung melalui aplikasi *R* maupun bantuan *RStudio*, bahasa *R* juga dapat dijalankan melalui beberapa *notebook platform/environment*, tanpa melakukan proses instalasi dalam perangkat/mesin. Di bawah ini disajikan beberapa *notebook*[3], yang dapat dimanfaatkan untuk menjalankan bahasa *R*, baik yang berdiri-sendiri maupun yang berbasis *web/cloud*.

a. Jupyter Notebook

- *Jupyter notebook* dapat dipasang dan dijalankan tersendiri, dengan langkah-langkah yang tersedia di tautan berikut: https://jupyter.org/install.
- atau sebagai bagian dari *Anaconda toolkit*, yang dapat diunduh di sini: https://www.anaconda.com/products/individual.
- dapat dijalankan di luar jaringan (luring) untuk menjalankan *Jupyter*.

  Dinamakan *Jupyter*, karena pada awalnya dikembangkan untuk mendukung penggunaan bahasa program ***Ju***lia - ***Pyt***hon - ***R***.

[3] *Notebook* atau *notebook document* merupakan *platform* yang mampu memadukan antara pengkodean (penulisan kode, menjalankan kode, dan hasil pengkodean, termasuk visualisasi data), dengan narasi teks, penulisan karakter matematika, dan media lain dalam satu dokumen.

b. *Google Colaboratory Notebook (Google Colab)*

- merupakan *platform* gratis berbasis *Jupyter notebook* yang dijalankan sepenuh melalui *cloud*.
- *Google Colab* sebenarnya dirancang untuk menjalankan bahasa *Python*.
- namun demikian, *Google Colab* dapat menjalankan bahasa *R*, dengan mengakses melalui tautan berikut:
  - https://colab.research.google.com/#create=true&language=r, atau
  - https://colab.to/r.
- perlu masuk dalam jaringan (daring) untuk menjalankan *Google Colab*.

c. *Kaggle Notebook*

- *Kaggle* menyediakan fasilitas pengkodean *R* ini, dengan mengakses melaui tautan ini: https://www.kaggle.com/notebooks.
- perlu masuk dalam jaringan (daring) untuk menjalankan *Kaggle Notebook*.

d. *Azure Notebooks*[4]

- *Microsoft* mengembangkan *Azure Notebooks* sebagai *platform* bahasa program berbasis *Jupyter Notebook*, yang dapat digunakan untuk menjalankan bahasa R.
- *Azure Notebooks* dapat diakses di sini: https://notebooks.azure.com/.
- perlu masuk dalam jaringan (daring) untuk menjalankan *Azure Notebook*.

e. *Amazon SageMaker*

- *Amazon* juga mengembangkan *cloud platform* sebagai *Amazon Web Services (AWS)*.
- salah satu fasilitas *AWS* adalah *Amazon SageMaker*, yang berbasis *Jupyter Notebook* dan dapat digunakan untuk menjalankan bahasa *R* melalui tautan: https://aws.amazon.com/sagemaker/.
- perlu masuk dalam jaringan (daring) untuk menjalankan *Amazon SageMaker*.

f. *IBM DataPlatform Notebooks*

- Pilihan lain untuk menjalankan *R* adalah *IBM DataPlatfrom Notebook* yang dikembangkan oleh *IBM Watson Data Platform and Data Science Experince (DSX)*.
- fasilitas ini dapat dinikmati melalui tautan berikut: https://dataplatform.cloud.ibm.com/.
- perlu masuk dalam jaringan (daring) untuk menjalankan *IBM Notebook*.

[4]Dalam situs resminya, yang kami akses pada 30-Desember-2020, diberikan pengumuman sebagai berikut: "*The Azure Notebooks preview will be retired on January 15th, 2021, and all user data will be destroyed. Please download your user data before then. To execute notebooks or create new notebook content, learn about our other notebooks experiences at Microsoft*". Saat ini, *Microsoft* mengembangkan beberapa *platform* layanan, antara lain: *Visual Code*, *Github Codespace*, *Azure Machine Learning*, maupun *Azure Lab*. Pilihan layanan tersebut dapat diakses di sini: https://notebooks.azure.com/Content/alternatives.html.

Dengan telah terpasangnya *R*, maka telaah data dan/atau pemrograman sudah dapat dilakukan dengan bahasa *R* dalam perangkat. *RStudio* juga dapat dijalankan untuk mempermudah proses tersebut. Selain itu, beberapa pilihan *notebooks platform*, baik yang berdiri-sendiri maupun *web-based/cloud*, terbuka luas untuk menjembatani kita dalam menjalankan bahasa *R*.

Selain itu, kita telah berkenalan dengan beberapa sintaks (*script*) dalam bahasa *R*. Bab-bab selanjutnya membahas telaah data dan *dataset* dengan bahasa *R*.

2 Dasar Penyiapan Direktori Kerja dan *Dataset*

Bab ini menyajikan beberapa persiapan awal sebelum melakukan telaah data. Persiapan ini ditujukan untuk mempermudah langkah-langkah selanjutnya. Bahasan meliputi penyiapan direktori kerja (*working directory - wd*), penyiapan *file* data yang akan ditelaah, penyiapan *dataset* agar dapat dibaca oleh bahasa *R*, serta pengenalan beberapa tipe data atomik bahasa *R*.

2.1 Penyiapan *Working Directory (wd)*

Sebelum menelaah data lebih lanjut, kita menyiapkan *folder* direktori kerja (*wd*) terlebih dahulu. *Folder* ini merupakan *working directory* yang dipergunakan untuk menyimpan seluruh file yang diperlukan dan akan dihasilkan dalam dalam proyek (*project*) kita.

- melihat posisi/jalur (*path*) *folder* saat ini.

```
getwd()  # membaca posisi folder dimana mesin bekerja saat ini
```

- mengatur *path* pada *folder wd* yang kita ingingkan.

```
setwd("D:/indarto/r_boekoe")  # contoh folder yang disiapkan
```

- membaca data apa saja yang berada pada *folder wd*.

```
dir()
```

2.2 Penyiapan *File* Data

Untuk mempermudah dan menyederhanakan pemahaman, buku/modul ini telah menyiapkan data jadi[1], untuk dipergunakan pada latihan-latihan pada Bab-bab selanjutnya. Data latihan ini adalah ***"tree.csv"*** dan berbentuk format ***.csv***[2], dengan tanda koma

[1] Topik tentang pembuatan data, konversi tipe dan format data, penggalian data (*data mining*), dan lain-lain, belum dibahas dalam buku/modul ini; namun, sangat disarankan untuk dipelajari lebih lanjut.

[2] *Comma Separated Values*, kadang disebut *Character Separated Values* atau *Comma Delimited File*, merupakan format *file* dalam bentuk teks dimana data disimpan dan dipisahkan dengan tanda tertentu (koma, titik koma, *tab*, dan lain-lain)

(,) sebagai *delimiter*. Data latihan ini dapat diunduh melalui tautan berikut: https://indarto.weebly.com/uploads/7/9/1/0/7910347/tree.csv.

- sekilas tentang isi data **"tree.csv"**.
  - data merupakan pengamatan dari 95 pohon.
    - terdiri dari 5 kelompok, yaitu kelompok 7, kelompok 12, kelompok 15, kelompok 31, dan kelompok 37.
    - kelompok ini berada pada kolom pertama "treeID".
  - *variable* yang diamati meliputi:
    - "Age": umur pohon.
    - "DBH": *diameter at breast height* (diameter pada ketinggian 4.5 kaki dari permukaan tanah).
    - "Height": tinggi pohon.
    - "Volume": volume kayu.
- silahkan mengunduh *file* data latihan tersebut di sini.
- simpan dalam *folder wd* yang telah kita siapkan di atas.
- periksa dan pastikan kembali bahwa *file* latihan tersebut telah siap di *folder wd*.

```
dir()
```

Kita telah menyiapkan *folder wd* dan memastikan bahwa data telah berada pada *folder wd* tersebut.

2.3 Penyiapan *Dataset* bagi *R*

Data latihan *"tree.csv"* telah siap dan berada pada *folder wd* yang diinginkan. Namun, data tersebut masih berada dalam format ekstensi *".csv"*, sehingga perlu di-*import* menjadi *dataframe* untuk bisa dibaca oleh bahasa *R*[3]. Dari sesi di atas, *file* ***tree.csv*** telah disimpan dalam *wd*.

- membuat (*assignment*) *dataset*, sintaks: `nama_dataset <- isi_dataset`.
  - membuat *dataset* bernama **"pohon"** yang berisi data dari *file* **"tree.csv"**.

```
pohon <- read.csv("tree.csv")
```

- R juga bisa langsung mengakses data langsung dari sumber situsnya (perlu terhubung ke internet).

```
pohon <- read.csv("https://indarto.weebly.com/uploads/7/9/1/0/7910347/tree.csv")
```

[3]Sebagai bahasa program, *R environment* mampu membaca, menuliskan, meng-*import*, atau meng-*export* data dari/ke berbagai tipe format *file* (antara lain: *.csv*, *.xlsx*, *.txt*, *.rds*, *.xml*, *.json*, maupun dari situs, dan lain-lain). Buku/modul ini tidak memberikan pembahasan khusus tentang ini, namun sangat disarankan untuk dapat dipelajari lebih lanjut.

- bisa menambahkan *parameter* "header=T" karena baris pertama pada data **"tree.csv"** adalah *header* (nama kolom).

```
pohon <- read.csv("tree.csv", header = T)
```

- memeriksa dan memastikan bahwa *dataset* **"pohon"** telah ada pada *folder wd*.

```
ls()
```

- memeriksa *class* dari data *pohon*.
  - sintaks: `class()`.

```
class(pohon)
```

```
## [1] "data.frame"
```

File data *tree.csv* yang awalnya berbentuk *csv*, sekarang berbentuk kelas *dataframe*[4] dan siap dijalankan oleh bahasa *R*.

2.4 Pengenalan Tipe Data

Dari peubah-peubah yang dibuat pada Bab sebelumnya, sesi Menjalankan *R* dan/atau *RStudio* , kita mengenal beberapa tipe data. Bahasa R mengenal tipe-tipe data, disebut sebagai ***atomic data types***, antara lain: ***character***, ***numeric***, ***logical***, *complex* dan *raw*. Tiga *atomic data types* diperkenalkan di bawah ini. Sintaks: `class()` atau `typeof()`.

a. Data Karakter (*character* atau *string*)

- peubah bukan angka.
- data karakter dituliskan di dalam tanda petik ("data_karakter").
  - contoh, peubah "salam" dan "nama" di atas merupakan data karakter.

```
class(salam)
```

```
## [1] "character"
```

```
class(nama)
```

```
## [1] "character"
```

- angka jika ditulis dalam tanda petik, maka merupakan data karakter.

[4] *Dataframe* merupakan salah satu bentuk *R-objects*. Beberapa jenis *R-objects*, antara lain: *vectors*, *lists*, *matrices*, *arrays*, *factors*, dan *dataframe*. Pilihan *R-objects* tersebut, memungkinkan pengguna dalam memilih kelas *dataset* yang sesuai untuk menyimpan datanya. Buku/modul ini tidak membahas jenis *R-objects* tersebut; namun, sangat disarankan untuk dipelajari lebih lanjut.

```r
class(usia)
```

```
## [1] "numeric"
```

```r
class("25")
```

```
## [1] "character"
```

b. Data Numerik (*numeric*)

- peubah angka.
- dapat berupa angka bulat (tanpa desimal).

```r
class(usia)
```

```
## [1] "numeric"
```

- dapat berupa angka desimal.

```r
ipk <- 3.45
class(ipk)
```

```
## [1] "numeric"
```

- data numerik memungkinkan untuk diolah dengan operasi aritmatika (penambahan, pengurangan, perkalian, pembagian, pangkat, dan lain-lain).

```r
next5years <- usia + 5  # usia lima tahun ke depan
print(next5years)
```

```
## [1] 30
```

```r
class(next5years)
```

```
## [1] "numeric"
```

c. Data Logika (*logical* atau *boolean*)

- bernilai *TRUE* (*T*) atau *FALSE* (*F*).

```r
lulus <- ipk >= 2.75
print(lulus)
```

```
## [1] TRUE
```

```
class(lulus)
```

```
## [1] "logical"
```

Data kategori (*categorical*) akan disajikan contohnya pada akhir sesi Tinjauan *Dataset*. Sedangkan, tipe data atomik lain (misalkan: *complex*, *integer*, *real*, atau *raw*) tidak dibahas dalam buku/modul ini; namun, disarankan untuk dapat dipelajari lebih lanjut.

3 Dasar Tinjauan Data

Kita telah menyiapkan direktori kerja dan *dataset* yang sudah dapat dibaca oleh bahasa *R*. Selanjutnya, Bab ini mengenalkan dasar-dasar untuk meninjau isi dari *dataset* kita. Bahasan meliputi tinjauan umum *dataset*, tinjauan baris dan kolom (atau dikenal sebagai proses *subsetting*, *slicing* atau *extracting*), serta modifikasi sederhana terhadap data (penambahan dan penghapusan dan kolom).

3.1 Tinjauan *Dataset*

Dataset berisi sekumpulan data, dalam bentuk baris dan kolom. *Dataset* dapat terdiri dari satu atau lebih baris dan satu atau lebih kolom. Dengan demikian, *dataset* paling tidak mempunyai satu baris dan satu kolom.

Dalam *dataset* "pohon" yang digunakan dalam latihan buku/modul ini, satu kolom mewakili satu *variable*, sehingga pada kolom yang sama bertipe data yang sama.

Selanjutnya, kita melakukan tinjauan singkat terhadap *dataset* kita.

- dimensi *dataset*, sintaks: `dim()`.

```
dim(pohon)  # dataset 'pohon' terdiri dari 95 baris dan 5 kolom
```

```
## [1] 95  5
```

- nama kolom *header* pada *dataset*, sintaks: `names()` atau `colnames()`.

```
names(pohon)  # memberikan hasil yang sama
```

```
## [1] "treeID" "Age"    "DBH"    "Height" "Volume"
```

```
colnames(pohon)
```

```
## [1] "treeID" "Age"    "DBH"    "Height" "Volume"
```

Terlihat bahwa *dataset* "pohon" mempunyai 5 kolom atau *header* atau *variable*, yaitu: "treeID": kolom 1, "Age": kolom 2, "DBH": kolom 3, "Height": kolom 4 dan "Volume": kolom 5.

- struktur *dataset*, sintaks: `str()`.

```
str(pohon)
```

```
## 'data.frame':    95 obs. of  5 variables:
##  $ treeID: int  7 7 7 7 7 7 7 7 7 7 ...
##  $ Age   : int  2 3 4 5 6 7 8 9 10 11 ...
##  $ DBH   : num  0 0.35 1.73 4.29 6.59 ...
##  $ Height: num  0.7 1.6 2.6 3.6 4.6 5.5 6.6 7.6 8.4 8.9 ...
##  $ Volume: num  0 0 0.001 0.004 0.01 0.018 0.027 0.035 0.045 0.051 ...
```

Sintaks `str()` menampilkan ringkasan struktur data, antara lain: nama kolom/*variable*, jumlah observasi, jumlah *variables* atau kolom, tipe data (*integer*, *numeric*, dll.) dan ringkasan nilai dari setiap *variable* atau kolom.

- ringkasan statistik *dataset*, sintaks: `summary()`.

```
summary(pohon)
```

```
##      treeID          Age          DBH            Height     
##  Min.   : 7.0   Min.   : 2   Min.   : 0.000   Min.   : 0.500  
##  1st Qu.:12.0   1st Qu.: 6   1st Qu.: 6.245   1st Qu.: 4.300  
##  Median :15.0   Median :11   Median :11.240   Median : 8.600  
##  Mean   :20.4   Mean   :11   Mean   : 9.656   Mean   : 8.058  
##  3rd Qu.:31.0   3rd Qu.:16   3rd Qu.:13.715   3rd Qu.:11.550  
##  Max.   :37.0   Max.   :20   Max.   :15.790   Max.   :15.400  
##      Volume       
##  Min.   :0.00000  
##  1st Qu.:0.00900  
##  Median :0.04500  
##  Mean   :0.05101  
##  3rd Qu.:0.08600  
##  Max.   :0.14800
```

Sintaks `summary()` menampilkan ringkasan statistik dari setiap *variable* yang mengandung tipe data *numeric* atau *non-character*. Ringkasan meliputi: nama kolom/*variable*, *minimum*, *maximum*, *median*, *mean*, *1st quantile*, dan *3rd quantile*.

- melihat isi atau nilai data, sintaks: `head()` atau `tail()`.

```
head(pohon)  # menampilkan isi 5 baris pertama
```

```
##   treeID Age  DBH Height Volume
## 1      7   2 0.00    0.7  0.000
## 2      7   3 0.35    1.6  0.000
## 3      7   4 1.73    2.6  0.001
## 4      7   5 4.29    3.6  0.004
## 5      7   6 6.59    4.6  0.010
## 6      7   7 8.38    5.5  0.018
```

```
tail(pohon)  # menampilkan 5 baris terakhir
```

```
##    treeID Age   DBH Height Volume
## 90     37  15 14.16   10.9  0.075
## 91     37  16 14.53   11.6  0.083
## 92     37  17 14.86   12.3  0.090
## 93     37  18 15.13   12.9  0.097
## 94     37  19 15.36   13.7  0.104
## 95     37  20 15.55   14.4  0.111
```

Kita juga dapat mengatur jumlah baris yang akan ditampilkan, dengan menambahkan angka sebagai *parameter* pada sintaks `head()` atau `tail()` di atas.

```
head(pohon, 7)  # menampilkan 7 baris pertama
```

```
##   treeID Age  DBH Height Volume
## 1      7   2 0.00    0.7  0.000
## 2      7   3 0.35    1.6  0.000
## 3      7   4 1.73    2.6  0.001
## 4      7   5 4.29    3.6  0.004
## 5      7   6 6.59    4.6  0.010
## 6      7   7 8.38    5.5  0.018
## 7      7   8 9.92    6.6  0.027
```

```
tail(pohon, 3)  # menampilkan 3 baris terakhir
```

```
##    treeID Age   DBH Height Volume
## 93     37  18 15.13   12.9  0.097
## 94     37  19 15.36   13.7  0.104
## 95     37  20 15.55   14.4  0.111
```

Data Kategori (*categorical variable*)

Pada sesi Mengenal Tipe Data, kita telah mengenal tiga tipe data atomik, yaitu *character*, *numeric*, dan *logical*. Dari *dataset* "pohon" di atas, kita menambah pengenalan tentang tipe data kategori. Dengan mencermati hasil sintaks `head(pohon)` dan `tail(pohon)` di atas, terlihat bahwa *variable* "treeID" terkategorikan dalam beberapa kelompok.

- sintaks untuk memerika nilai unik dari variable tertentu: `unique()`.
- memeriksa struktur *variable* "treeID".

```
str(pohon$treeID)  # 'treeID' berjumlah 95 observasi.
```

```
##  int [1:95] 7 7 7 7 7 7 7 7 7 7 ...
```

- memeriksa nilai unik atau kelompok kategori "treeID".

```
unique(pohon$treeID)
```

```
## [1]  7 12 15 31 37
```

Walaupun "treeID" mempunyai 95 obervasi, *variable* ini mempunyai 5 kategori/kelompok, yaitu: 7, 12, 15, 31, 37.

3.2 Tinjauan Baris dan Kolom

Kita telah melakukan tinjauan singkat terhadap *dataset* "pohon" (struktur, ringkasan statistik, dan isi). Kita akan melakukan tinjauan lebih lanjut terhadap baris atau kolom tertentu, termasuk ekstraksi (*extracting/slicing*) terhadap *dataset* kita.

- sintaks dasar untuk melakukan *slicing* baris/kolom tertentu dari *dataset*: `nama_*dataset*[baris ke-, kolom ke-]`.
- ingat kembali bahwa *dataset* "pohon" terdiri dari 95 baris dan 5 kolom.

```
dim(pohon)
```

```
## [1] 95  5
```

a. Ekstrak/iris baris

- sintaks: `dataset[baris ke-, ]`.
- mengekstrak baris ke-3, untuk seluruh kolom.

```
pohon[3, ]
```

```
##   treeID Age  DBH Height Volume
## 3      7   4 1.73    2.6  0.001
```

- mengesktrak baris ke 1-5 untuk seluruh kolom.

```
pohon[1:5, ]  # hasilnya sama dengan sintaks 'head()'
```

```
##   treeID Age  DBH Height Volume
## 1      7   2 0.00    0.7  0.000
## 2      7   3 0.35    1.6  0.000
## 3      7   4 1.73    2.6  0.001
## 4      7   5 4.29    3.6  0.004
## 5      7   6 6.59    4.6  0.010
```

b. Ekstrak/iris kolom

- sintaks: `dataset[ , kolom ke-]`.
- ingat kembali bahwa *dataset* "pohon" mempunyai 5 kolom atau *variables*, yaitu: kolom ke-1 = "treeID", kolom ke-2 = "Age", kolom ke-3 = "DBH", kolom ke-4 = "Height", dan kolom ke-5 = "Volume".

```
colnames(pohon)
```

```
## [1] "treeID" "Age"    "DBH"    "Height" "Volume"
```

- mengekstrak kolom ke-3, untuk seluruh baris.

```
options(width = 70)  # sintaks ini tidak harus ada
pohon[, 3]
```

```
##  [1]  0.00  0.35  1.73  4.29  6.59  8.38  9.92 10.86 11.66 11.97 12.40
## [12] 12.97 13.57 13.88 14.12 14.57 14.94 15.41 15.79  0.00  0.00  1.36
## [23]  3.23  5.21  6.71  8.08  8.99  9.77 10.19 10.81 11.41 12.05 12.40
## [34] 12.74 13.06 13.38 13.69 13.94  0.00  0.05  1.60  3.51  5.90  7.61
## [45]  9.02  9.80 10.67 11.14 11.93 12.60 13.30 13.66 14.00 14.41 14.86
## [56] 15.22 15.57  0.00  0.51  1.25  3.16  5.60  7.41  8.82  9.82 10.82
## [67] 11.24 11.87 12.60 13.29 13.54 13.83 14.28 14.65 15.06 15.35  0.00
## [78]  0.00  1.42  3.33  5.71  7.51  9.03 10.09 10.96 11.80 12.57 13.21
## [89] 13.74 14.16 14.53 14.86 15.13 15.36 15.55
```

- dapat juga mengekstrak kolom berdasarkan nama kolomnya; misal, mengekstrak kolom "Age", untuk seluruh baris.

```
options(width = 70)  # tidak wajib ada
pohon$Age  # hasilnya sama dengan sintaks 'pohon[,3]' di atas
```

```
##  [1]  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20  2  3  4
## [23]  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20  2  3  4  5  6  7
## [45]  8  9 10 11 12 13 14 15 16 17 18 19 20  2  3  4  5  6  7  8  9 10
## [67] 11 12 13 14 15 16 17 18 19 20  2  3  4  5  6  7  8  9 10 11 12 13
## [89] 14 15 16 17 18 19 20
```

- mengekstrak kolom ke-2 sampai dengan kolom ke-5, untuk seluruh baris.

```
pohon[, 2:5]  # hasil tidak ditampilkan karena terlalu panjang
```

Menambah kolom baru

- sintaks: `dataset$namakolombaru <- isi_kolombaru`.
- misalkan, kita menambahkan kolom baru (diberi nama "rasio_tinggi"), yang diisi sebagai hasil pembagian dari "Volume" dengan "Height".

```
pohon$rasio_tinggi <- pohon$Volume/pohon$Height
```

- memastikan bahwa tambahan kolom baru tersebut sudah berhasil.

```
colnames(pohon)  # periksa nama kolom
```

```
## [1] "treeID"       "Age"          "DBH"          "Height"
## [5] "Volume"       "rasio_tinggi"
```

```
head(pohon)  # periksa 5 baris pertama
```

```
##   treeID Age  DBH Height Volume rasio_tinggi
## 1      7   2 0.00    0.7  0.000 0.0000000000
## 2      7   3 0.35    1.6  0.000 0.0000000000
## 3      7   4 1.73    2.6  0.001 0.0003846154
## 4      7   5 4.29    3.6  0.004 0.0011111111
## 5      7   6 6.59    4.6  0.010 0.0021739130
## 6      7   7 8.38    5.5  0.018 0.0032727273
```

Menghapus kolom

- misalkan, kita ingin menghapus kolom "rasio_tinggi", yang baru saja kita buat.

```
pohon$rasio_tinggi <- NULL
```

- untuk memastikan penghapusan tersebut, kita periksa kembali *dataset* kita.

```
colnames(pohon)
```

```
## [1] "treeID" "Age"    "DBH"    "Height" "Volume"
```

```
head(pohon, 3)
```

```
##   treeID Age  DBH Height Volume
## 1      7   2 0.00    0.7  0.000
## 2      7   3 0.35    1.6  0.000
## 3      7   4 1.73    2.6  0.001
```

c. Ekstrak/iris baris dan kolom

- sintaks: `dataset[baris ke- , kolom ke-]`.
- mengekstrak baris ke-3 dan kolom ke-4.

```
pohon[3, 4]
```

```
## [1] 2.6
```

- mengekstrak baris ke 1-3 dan kolom 2-5.

```
pohon[1:5, 2:5]
```

```
##   Age  DBH Height Volume
## 1   2 0.00    0.7  0.000
## 2   3 0.35    1.6  0.000
## 3   4 1.73    2.6  0.001
## 4   5 4.29    3.6  0.004
## 5   6 6.59    4.6  0.010
```

d. Membuat *dataset* baru

Dari hasil ekstraksi di atas, kita dapat membuat dan menyimpannya dalam *dataset* baru.

- sintaks, sama dengan sintaks *assignment*: `dataset_baru <- isi_dataset_baru`.
- dari proses ekstraksi (*slicing*) data, kita dapat meng-*assign* atau membuat *dataset* baru, yang isinya hasil ekstraksi/iris di atas.
- misalkan, kita membuat *dataset* baru bernama "pohon_umur", yang berisi hanya kolom "treeID" dan kolom "Age".

```
pohon_umur <- pohon[, 1:2]
head(pohon_umur)  # periksa 5 baris pertama
```

```
##   treeID Age
## 1      7   2
## 2      7   3
## 3      7   4
## 4      7   5
## 5      7   6
## 6      7   7
```

Kita telah melakukan tinjauan terhadap *dataset* dan memahami isi *dataset* tersebut. Bab berikut membahas dasar visualisasi data, untuk lebih mendalami isi data kita.

4 Dasar Visualisasi Data

Visualisasi data merupakan bagian penting dalam telaah data. Bab ini memberikan langkah-langkah dasar dalam menampilkan visualisasi data, terutama *histogram*, *boxplot*, *scatterplot*, termasuk *line*.

4.1 *Histogram*

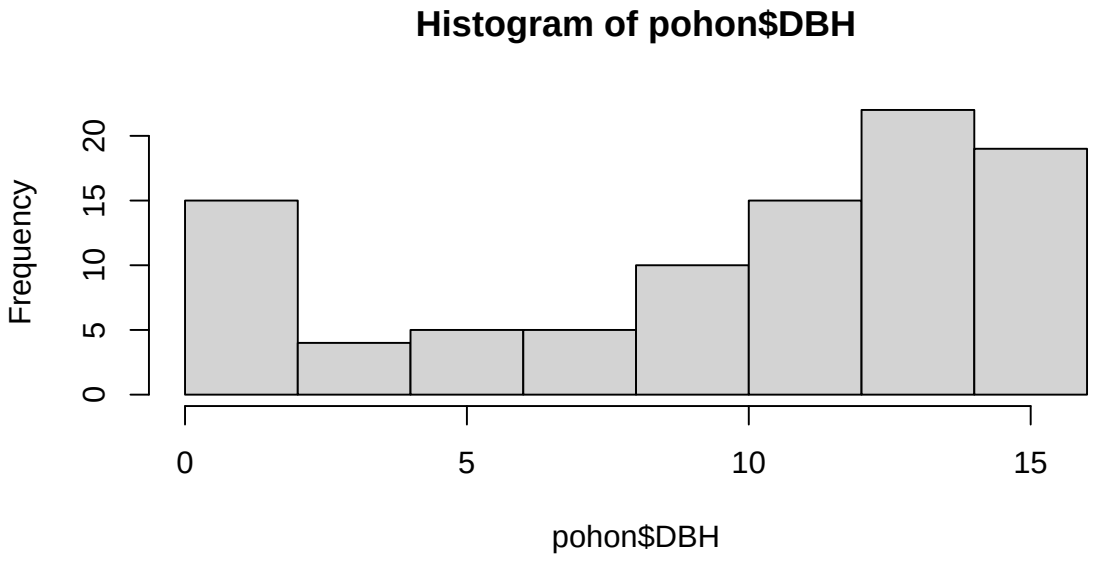
Histogram menampilkan distribusi probabilitas (*frequency*) dan densitas (*density*) dari data (*continous variabel*).

- Sintaks *histogram*: `hist(variable)`.

a. *Histogram* Frekuensi

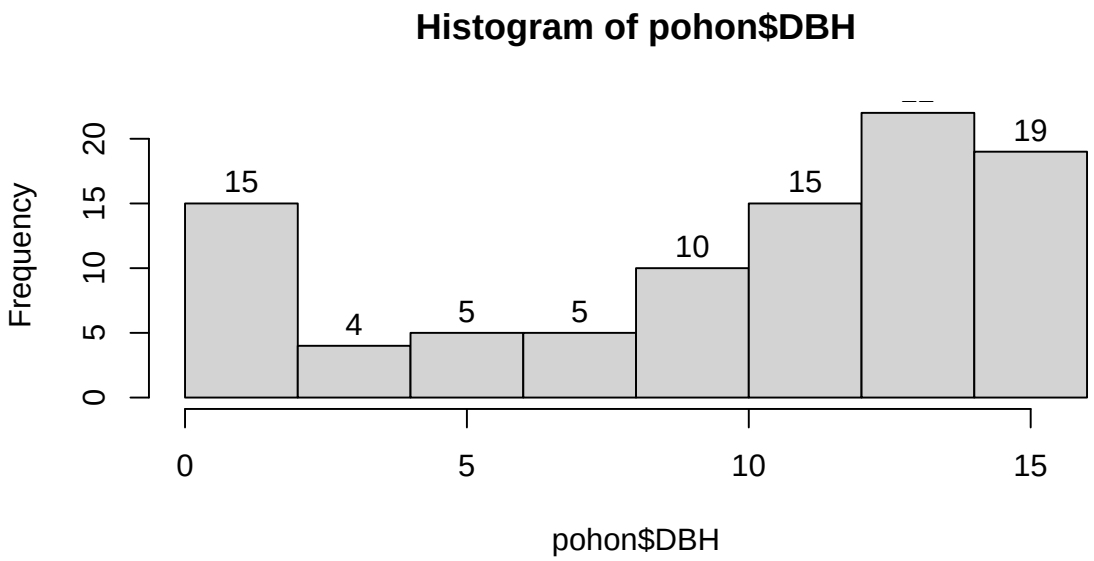
- *histogram* dari kolom atau *variable* "DBH".

```
hist(pohon$DBH)
```



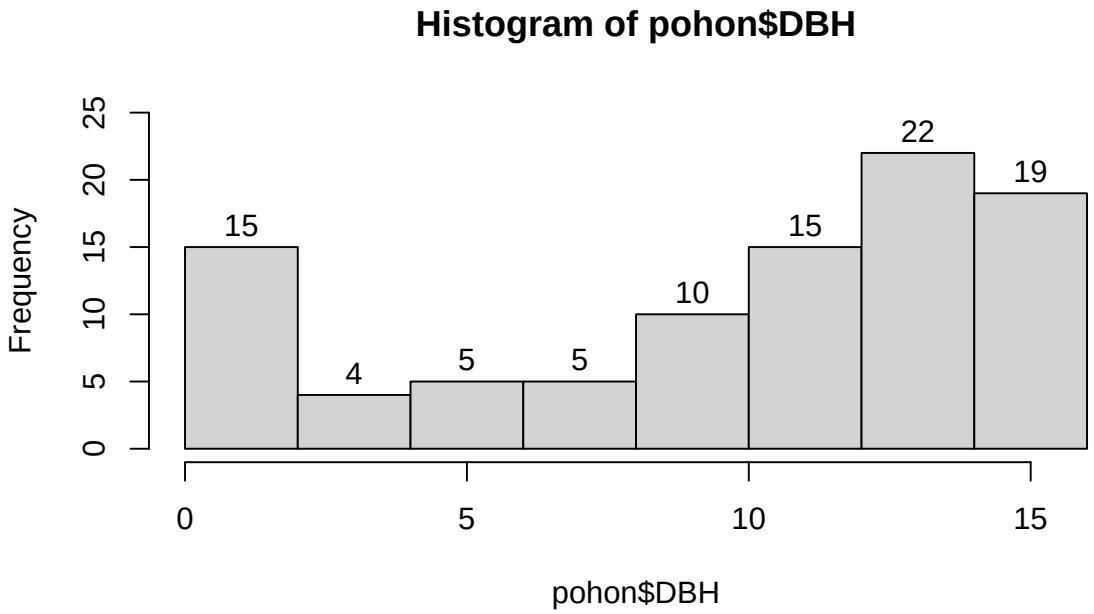
- memberikan label nilai pada *histogram* di atas, tambahan *parameter* `labels=T`.

```
hist(pohon$DBH, labels = T)
```



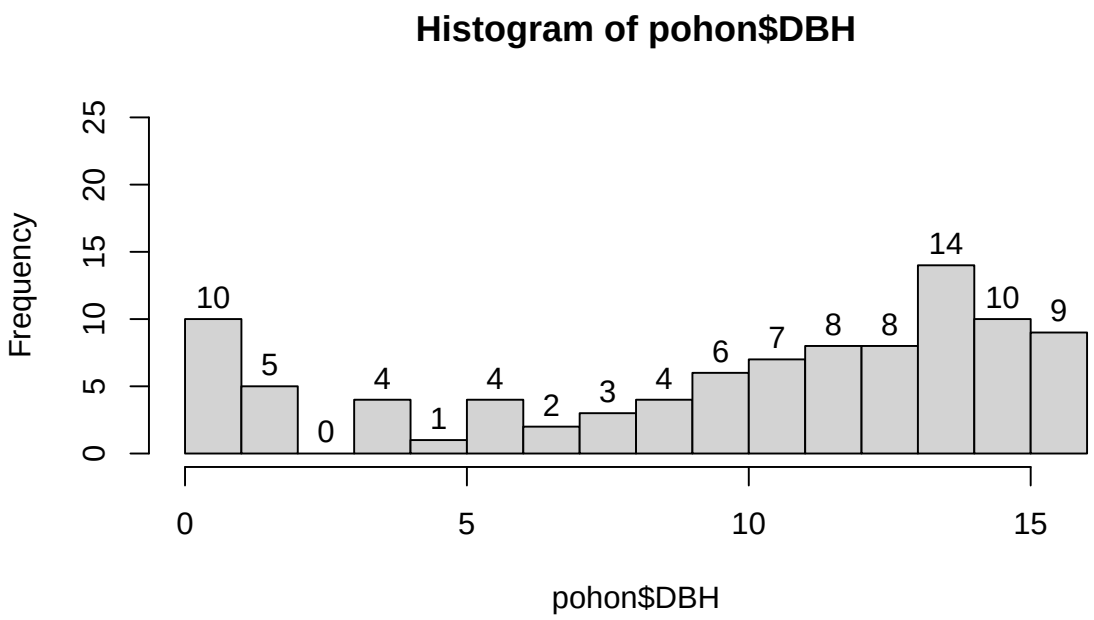
- mengatur batas sumbu y (*y-axis*), tambahan *parameter* `ylim=c(batas bawah, batas atas)`.

```
hist(pohon$DBH, labels = T, ylim = c(0, 25))
```



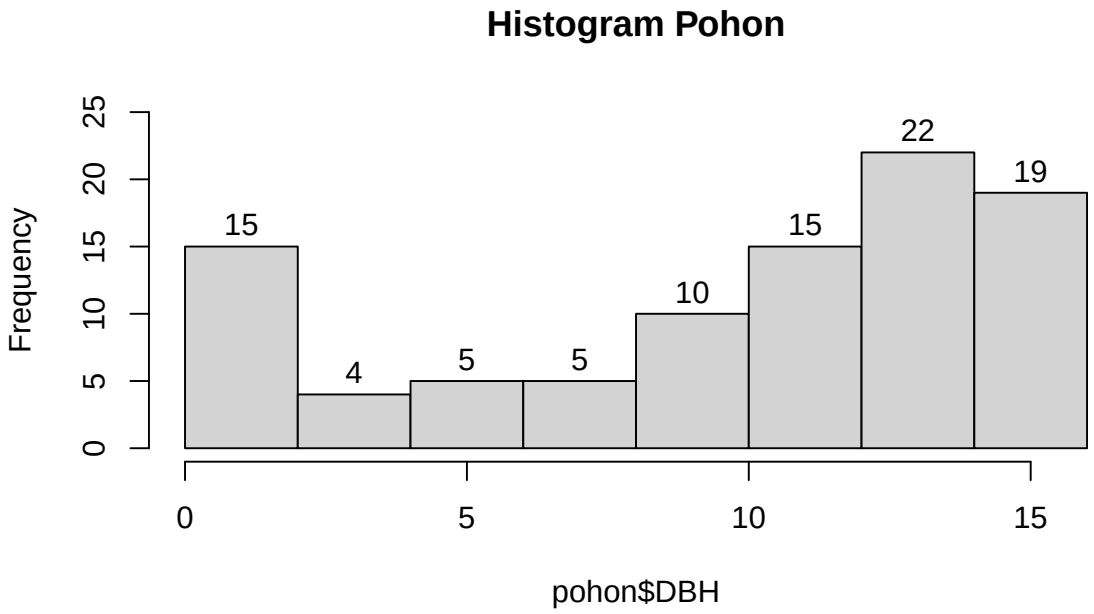
- mengatur jumlah kelompok sumbu X, tambahan parameter `breaks=jumlah_breaks`.

```
hist(pohon$DBH, labels = T, ylim = c(0, 25), breaks = 15)
```



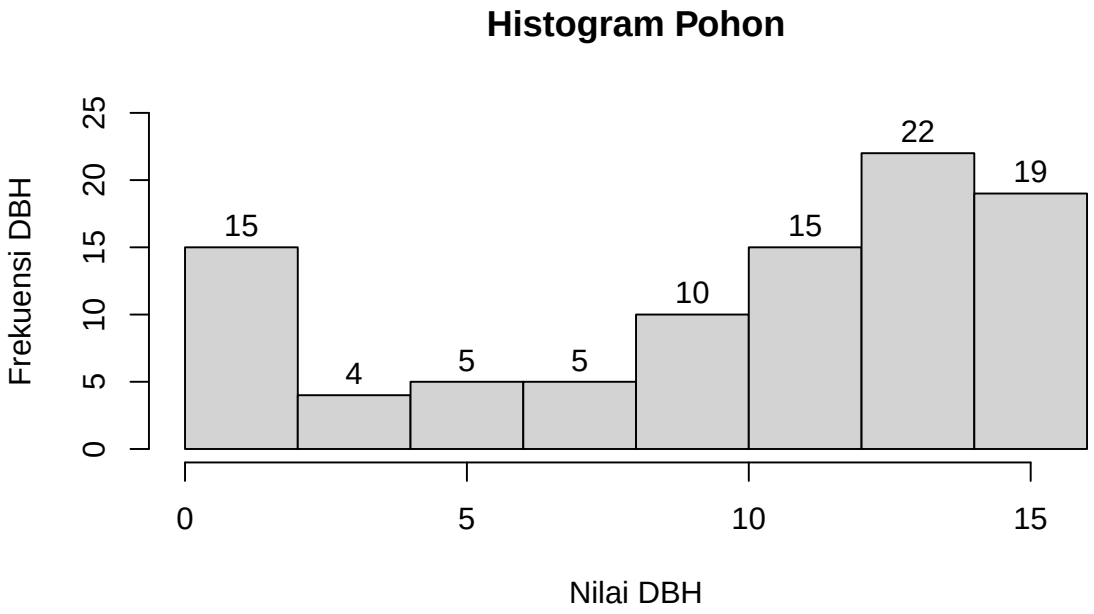
- menambahkan judul *histogram*, tambahan *parameter* `main="judul_histogram"`.

```
hist(pohon$DBH, labels = T, ylim = c(0, 25), main = "Histogram Pohon")
```



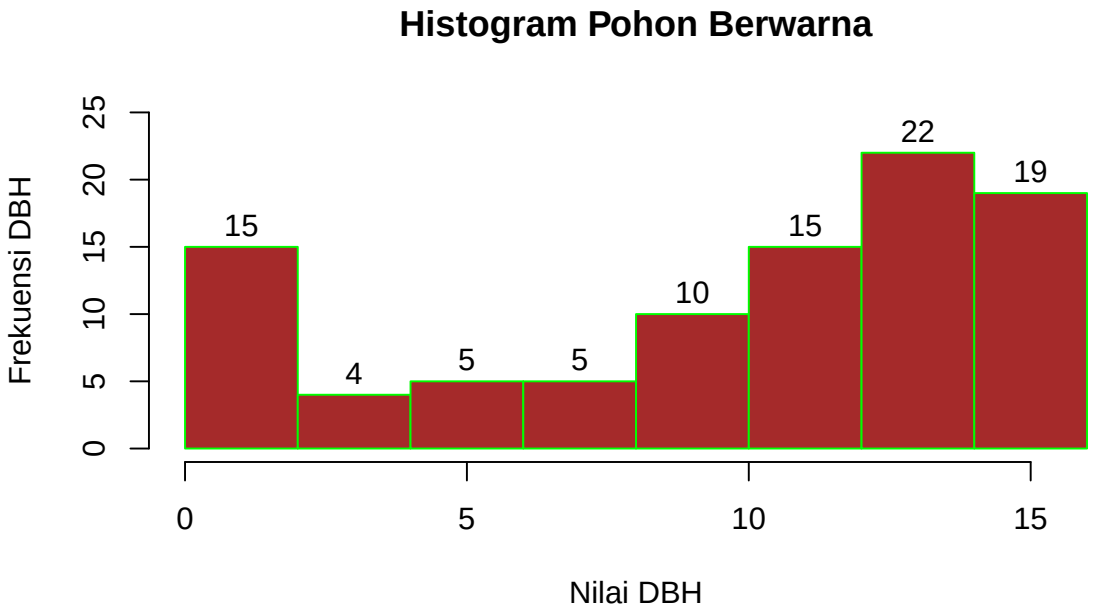
- mengubah nama sumbu X (`xlab="nama_sumbu_x"`) dan sumbu Y (`ylab="nama_sumbu_y"`).

```
hist(pohon$DBH, labels = T, ylim = c(0, 25), main = "Histogram Pohon",
    xlab = "Nilai DBH", ylab = "Frekuensi DBH")
```



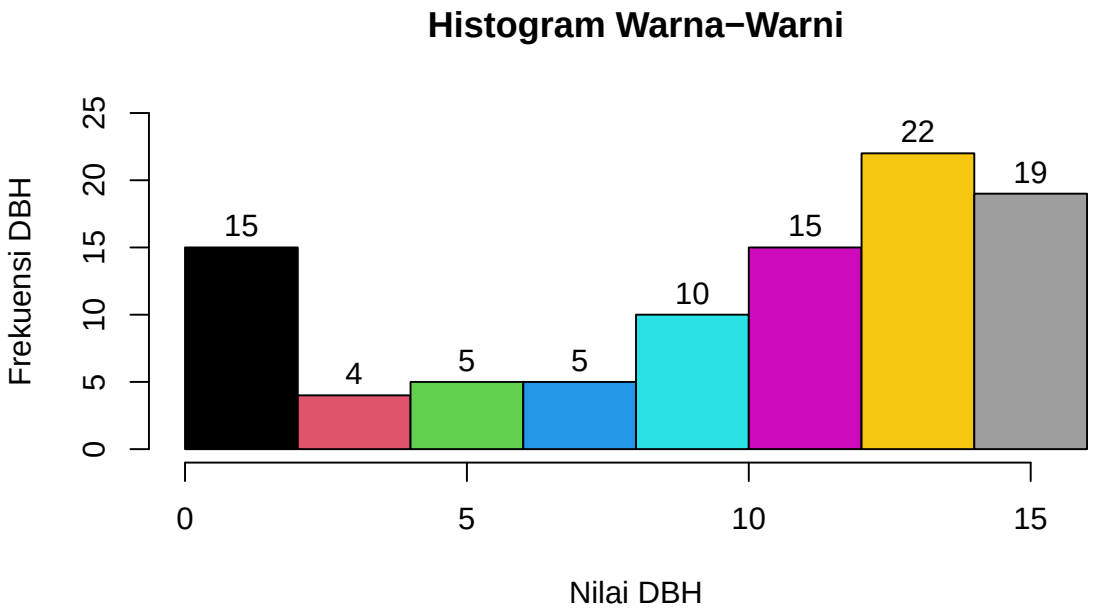
- mewarnai *bar histogram* (`col="warna"`) dan batas *bar histogram* (`border="warna"`).

```
hist(pohon$DBH, labels = T, ylim = c(0, 25), main = "Histogram Pohon Berwarna",
    xlab = "Nilai DBH", ylab = "Frekuensi DBH", col = "brown",
    border = "green")
```



- pilihan lain: memberikan warna-warni yang berbeda pada setiap *bar histogram* (perhatikan tambahan *parameter* `col=c(1:8)`.

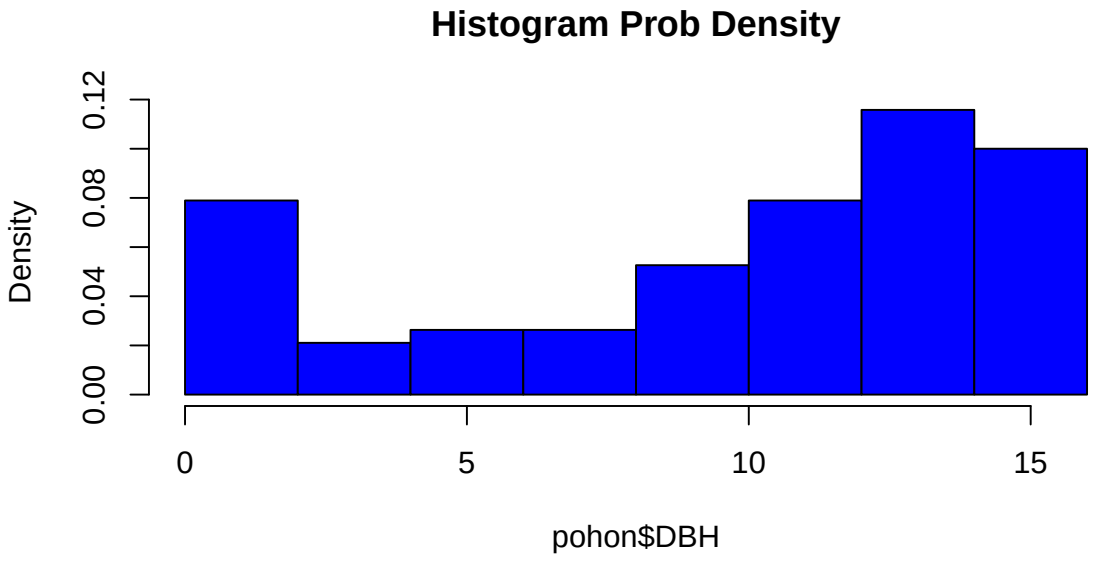
```
hist(pohon$DBH, labels = T, ylim = c(0, 25), main = "Histogram Warna-Warni",
    xlab = "Nilai DBH", ylab = "Frekuensi DBH", col = c(1:8))
```



b. *Histogram Probability Density*

- menambahkan *parameter* `prob=T` pada *histogram* frekuensi.
  - *histogram probability density* dari *variable* "DBH".

```
hist(pohon$DBH, prob = T, col = "blue", main = "Histogram Prob Density")
```

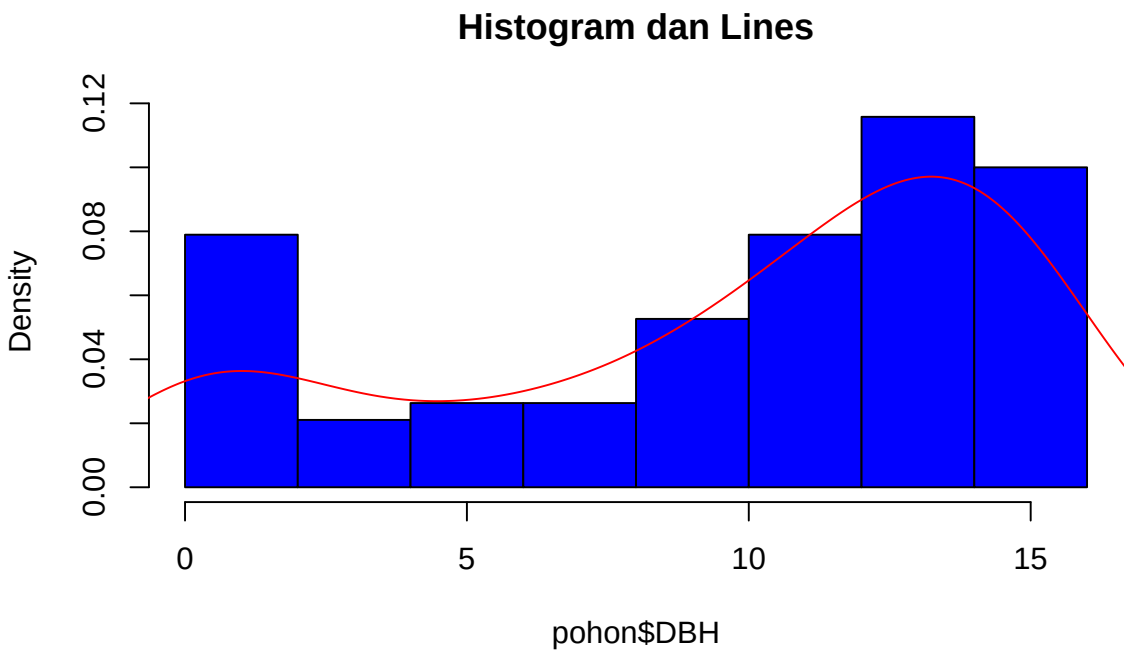


- atau, tambahan *parameter* `freq=F`.

```
hist(pohon$DBH, freq = F)
```

- menambahkan garis *probability density*-nya, dengan sintaks `lines(density())`.

```
hist(pohon$DBH, prob = T, col = "blue", main = "Histogram dan Lines")
lines(density(pohon$DBH), col = "red")
```

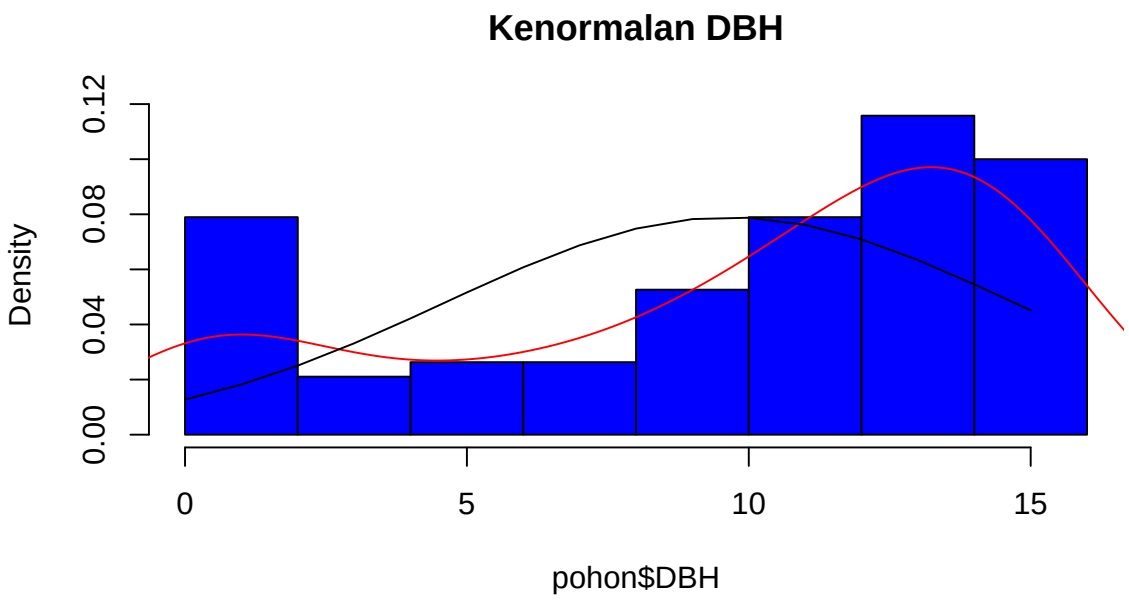


c. *Normality Checking*

Salah satu penerapan *histogram* dan *lines* adalah untuk memeriksa kenormalan data. Misal, kita ingin memeriksa kenormalan sebaran data *variable* "DBH".

- membuat *histogram probability density* "DBH".
- menambahkan *lines probability density* "DBH" (diberi warna "red").
- membuat dataset (diberi nama "x") berisi nilai angka dari nilai minimal "DBH" sampai dengan nilai maksimal "DBH".
- membuat *dataset* (diberi nama "normal") berisi distribusi normal dari "DBH".
- menambahkan *lines* antara *variable* "x" dan "normal".

```
hist(pohon$DBH, prob = T, col = "blue", main = "Kenormalan DBH")
lines(density(pohon$DBH), col = "red")
x <- seq(min(pohon$DBH), max(pohon$DBH))
normal <- dnorm(x, mean = mean(pohon$DBH), sd = sd(pohon$DBH))  # membuat
# *dataset* baru berupa data distribusi normal dari 'DBH'
lines(x, normal)  # me
```



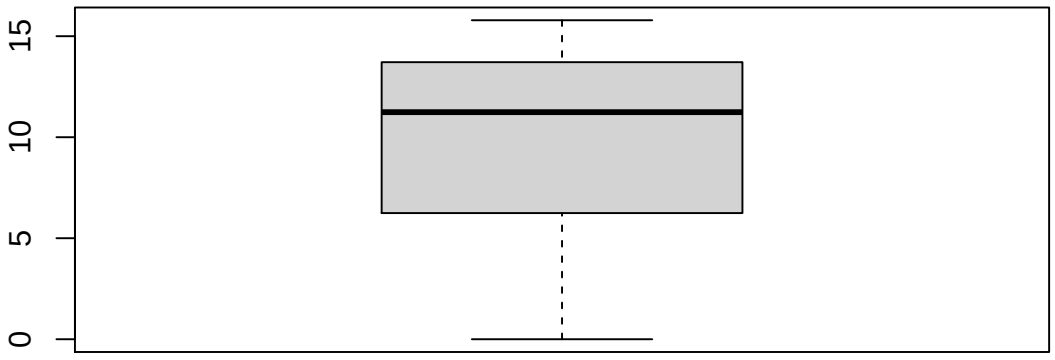
4.2 *Boxplot*

Boxplot menampilkan sebaran dari data (median, kuartil, *whiskers*, dan *outlier*). Sintaks *boxplot*: `boxplot(variables)`.

a. *Boxplot* Vertikal

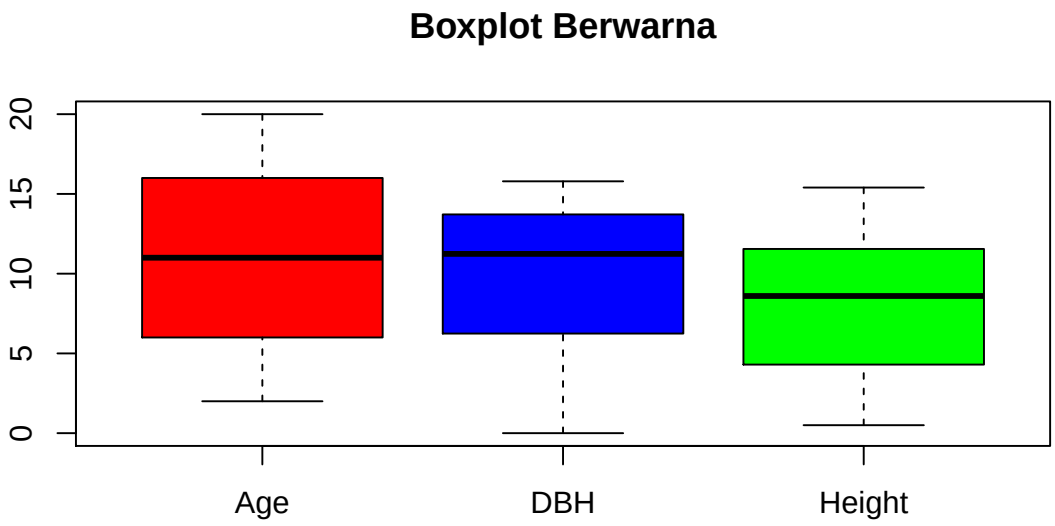
- membuat *boxplot variable* "DBH".

```
boxplot(pohon$DBH)
```



- membuat *boxplot* dari *variable*/kolom ke-2 sampai dengan kolom ke-4, dengan tambahan judul dan warna.

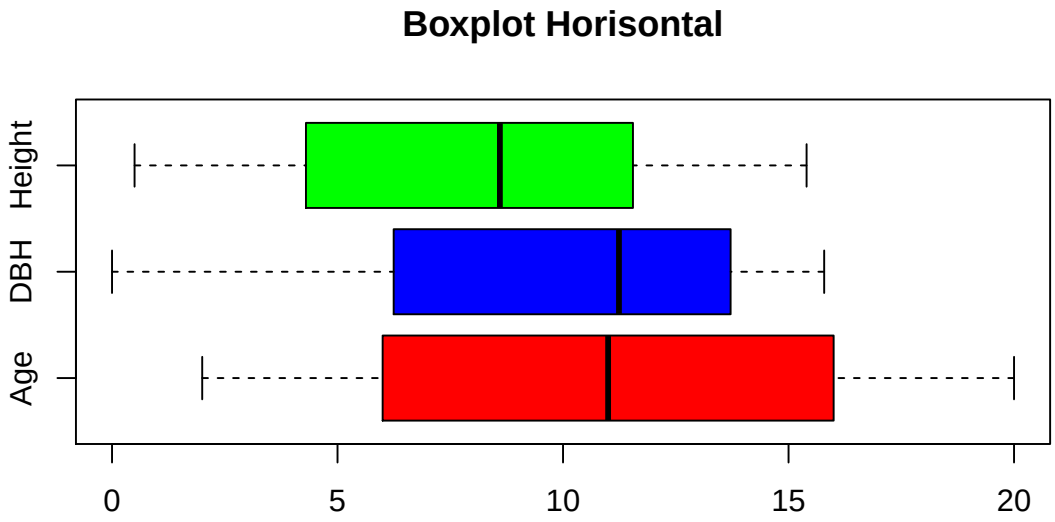
```
boxplot(pohon[, 2:4], main = "Boxplot Berwarna", col = c("red",
    "blue", "green"))
```



b. *Boxplot* Horisontal

- menambah parameter `horizontal=T`.

```
boxplot(pohon[, 2:4], main = "Boxplot Horisontal", col = c("red",
    "blue", "green"), horizontal = T)
```

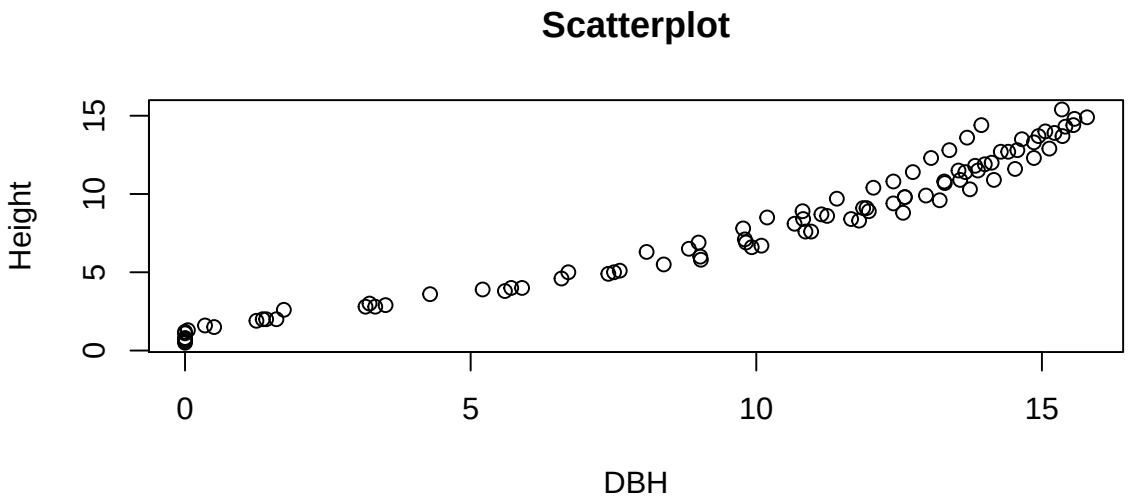


4.3 *Scatterplot*

Scatterplot menampilkan koleksi titik sebagai hubungan antara dua *variable*. Sintaks *scatterplot*: `plot()`.

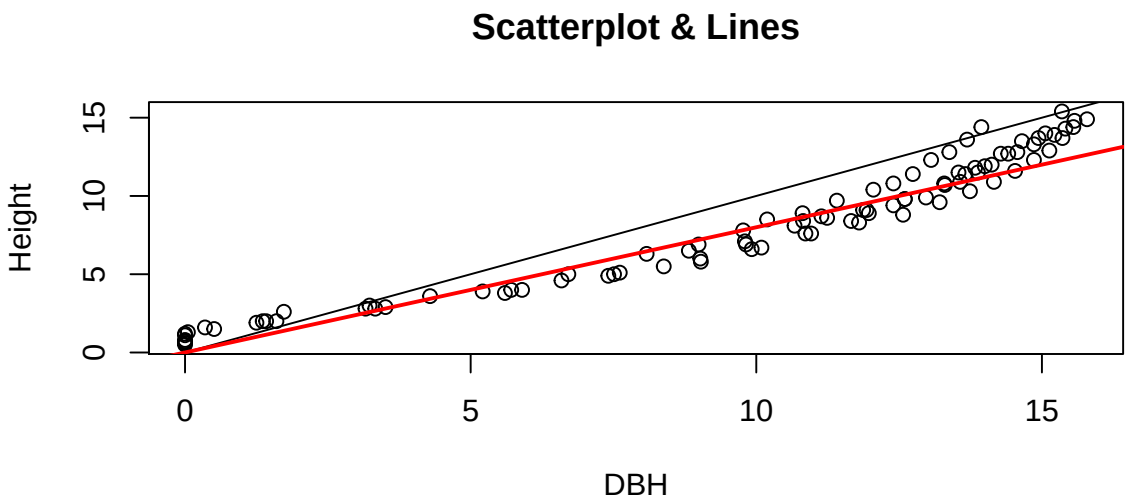
- membuat *scatterplot* antara "Height" dan "DBH".

```
plot(Height ~ DBH, pohon, main = "Scatterplot")
```



- menambahkan *lines*, dengan sintaks `abline(a=, b=)`.
  - *lines* dengan *intercept* a=0 dan *slope* b=1.
  - dan, *lines* dengan *intercept* a=0 dan *slope* b= 0.8.

```
plot(Height ~ DBH, pohon, main = "Scatterplot & Lines")
abline(a = 0, b = 1)
abline(a = 0, b = 0.8, col = "red", lwd = 2)
```



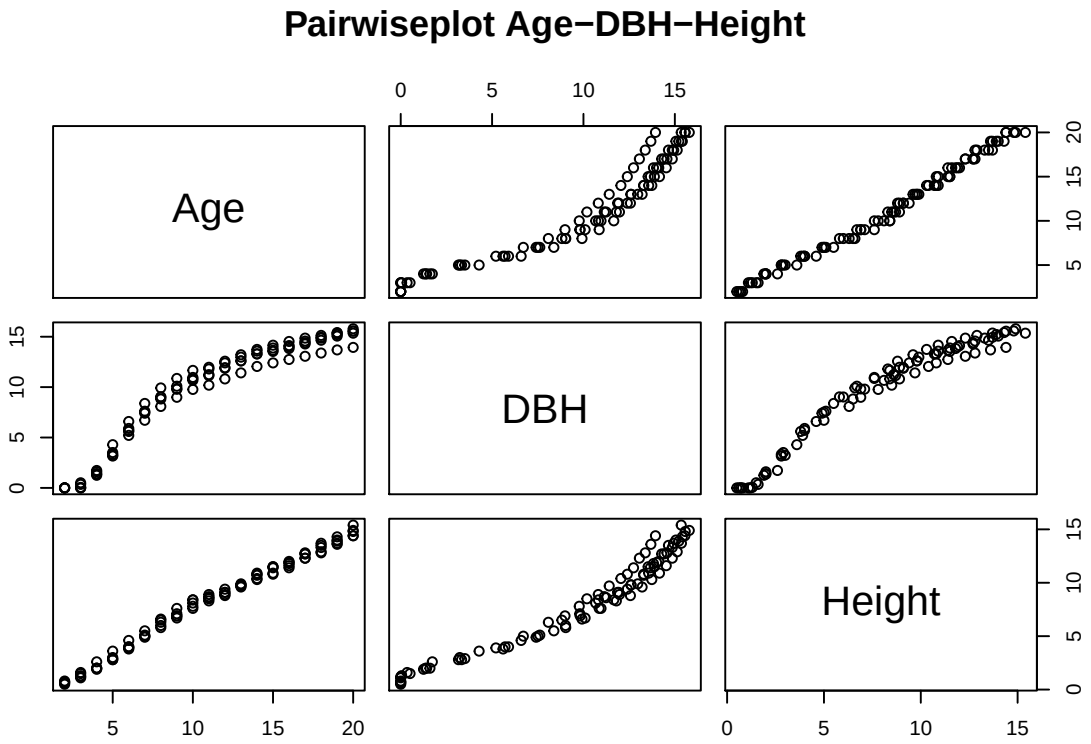
4.4 *Pairwiseplot*

Pairwiseplot semacam matriks dari beberapa grafik *scatterplot*.

Sintaks *pairwiseplot*: `pairs(dataset)`.

- menampilkan *pairwiseplot* dari *dataset* "pohon" untuk kolom ke-2 sampai dengan kolom ke-4.

```
pairs(pohon[, 2:4], main = "Pairwiseplot Age-DBH-Height")
```



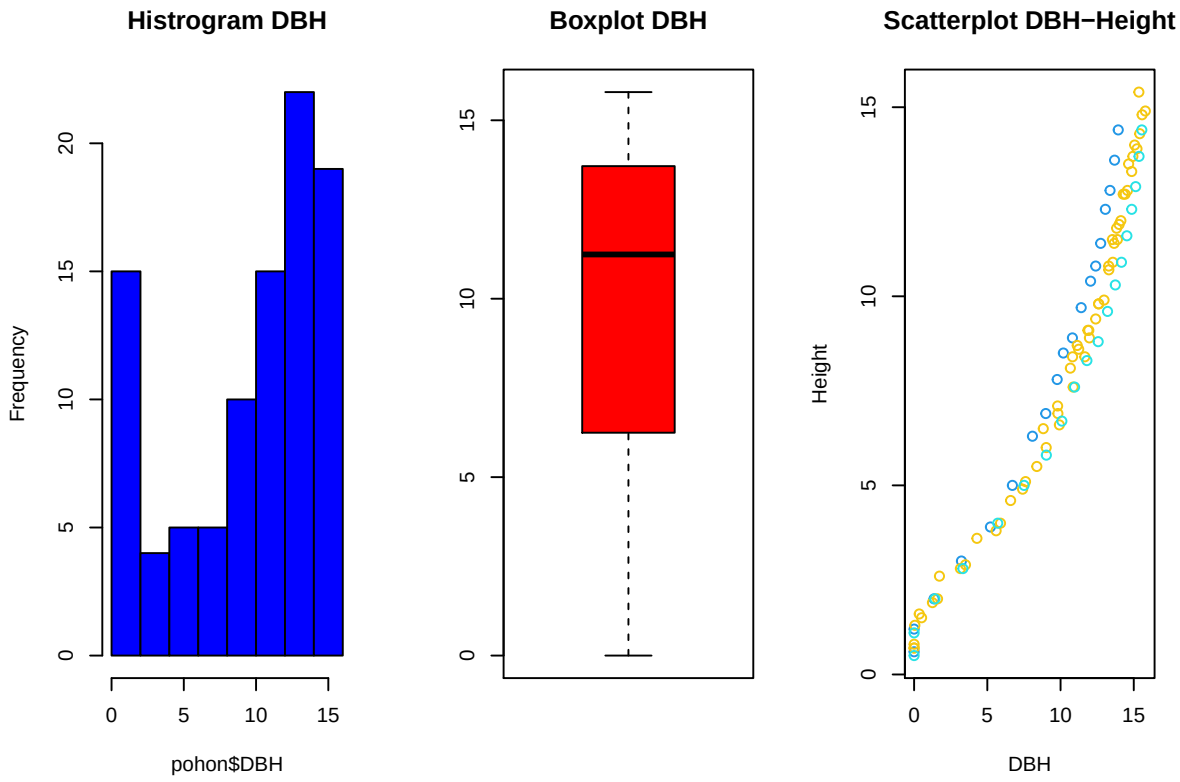
4.5 Gabungan Beberapa Visualisasi Data dalam Satu Kerangka Gambar

Beberapa jenis grafik yang berbeda-beda dapat ditampilkan dalam satu kerangka (*frame*) gambar. Misal, kita ingin menampilkan 3 grafik (*histogram*, *boxplot*, dan *scatterplot*) dalam satu kerangka gambar (1 baris dengan 3 kolom).

Sintaks menyiapkan kerangka gambar: `par(mfrow=c(jumlah_baris, jumlah_kolom))`.

- baris 1 kolom 1 diisi *histogram* "DBH" (warna biru).
- baris 1 kolom 2 diisi *boxplot* "DBH" (warna merah).
- baris 1 kolom 3 diisi *scatterplot* "DBH" dan "Height" (warna hijau).

```
par(mfrow = c(1, 3))
hist(pohon$DBH, col = "blue", main = "Histrogram DBH")
boxplot(pohon$DBH, col = "red", main = "Boxplot DBH")
plot(Height ~ DBH, pohon, main = "Scatterplot DBH-Height", col = c(treeID))
```



Kita telah mengenal pemahaman dasar dalam menampilkan beberapa pilihan visualisasi data, terutama *histogram*, *boxplot*, *scatterplot* dan *lines*.

5 Dasar Regresi Sederhana

Visualisasi grafik dari data sangat berguna dalam menampilkan isi dan perilaku data kita. Visualisasi juga mempermudah mengkomunikasikan data dengan pihak yang kita inginkan. Guna mendukung telaah data dengan bukti kuantitatif, Bab ini memberikan pemahaman dasar dalam melakukan regresi data. Bahasan meliputi hal-hal dasar tentang statistik, regresi linear dan regresi non-linear.

5.1 Dasar Statistik Deskriptif

Sesi ini sekedar untuk mengingatkan kembali tentang cara memperoleh beberapa nilai statistik deskriptif dari data.

- menghitung nilai minimal `min()` dan maksimal `max()`.

```
min(pohon$Height)  # nilai minimum dari 'Height'
```

```
## [1] 0.5
```

```
max(pohon$Height)  # nilai maksimum dari 'Height'
```

```
## [1] 15.4
```

- pada sesi sebelumnya, sintaks `summary()` telah dikenal, untuk menampilkan ringkasan statistik.

```
summary(pohon$Height)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.500   4.300   8.600   8.058  11.550  15.400
```

- menghitung penjumlahan (*summation*) data.

```
sum(pohon$Height)  # penjumlahan dari 'Height'
```

```
## [1] 765.5
```

- menghitung panjang atau jumlah observasi data.

```
length(pohon$Height)  # jumlah observasi dari 'Height'
```

```
## [1] 95
```

- menghitung *variance*, *standard deviation*, *covariance*, dan *correlation*.

```
var(pohon$Height)  # variance
```

```
## [1] 18.71395
```

```
sd(pohon$Height)  # standard deviation
```

```
## [1] 4.325963
```

```
cov(pohon$Height, pohon$DBH)  # covariance
```

```
## [1] 21.18354
```

```
cor(pohon$Height, pohon$DBH)  # correlation
```

```
## [1] 0.9687732
```

Sesi selanjutnya menyajikan pembahasan model regresi sederhana, baik regresi *linear* maupun regresi *non-linear*.

5.2 *Linear Model (lm)*

Model linear dapat terdiri dari *independent variable* tunggal (*simple linear model*) atau *independent variable* yang lebih dari satu (*multiple linear model*).

5.2.1 *Simple Linear Model*

Model teoretis (*theoretical model*) dari model *linear* tunggal seperti di bawah ini.

$$Y = \alpha + \beta X$$

Pada sesi visualisasi, kita telah mengenal sintaks `abline(a=, b=)`, dimana nilai "a" adalah *intercept*, dan nilai "b" adalah *slope*. Dalam fungsi *linear* tunggal di atas, α = "a" = *intercept*; dan β = "b" = *slope*.

Latihan kasus di bawah ini ingin membangun model *linear* tunggal, untuk melihat pengaruh *variable* "DBH" terhadap *variable* "Height". Dengan demikian, model empiris (*empirical model*) *linear* tunggalnya dituliskan sebagai berikut.

$$Height = \alpha + \beta DBH$$

Data yang digunakan bersumber dari *dataset* "pohon". Namun demikian, kita hanya membatasi pada kelompok pohon ke-7 ("treeID=7").

- mengekstrak dan membentuk *dataset* baru ("pohon7"), yang berisi seluruh *variable* dari pohon kelompok ke-7.
  - sintaks untuk *slicing*: `dataset[baris ke-, kolom ke-]`.

```
pohon7 <- pohon[pohon$treeID == 7, ]
```

- meninjau ringkasan *dataset* "pohon7".

```
head(pohon7)
```

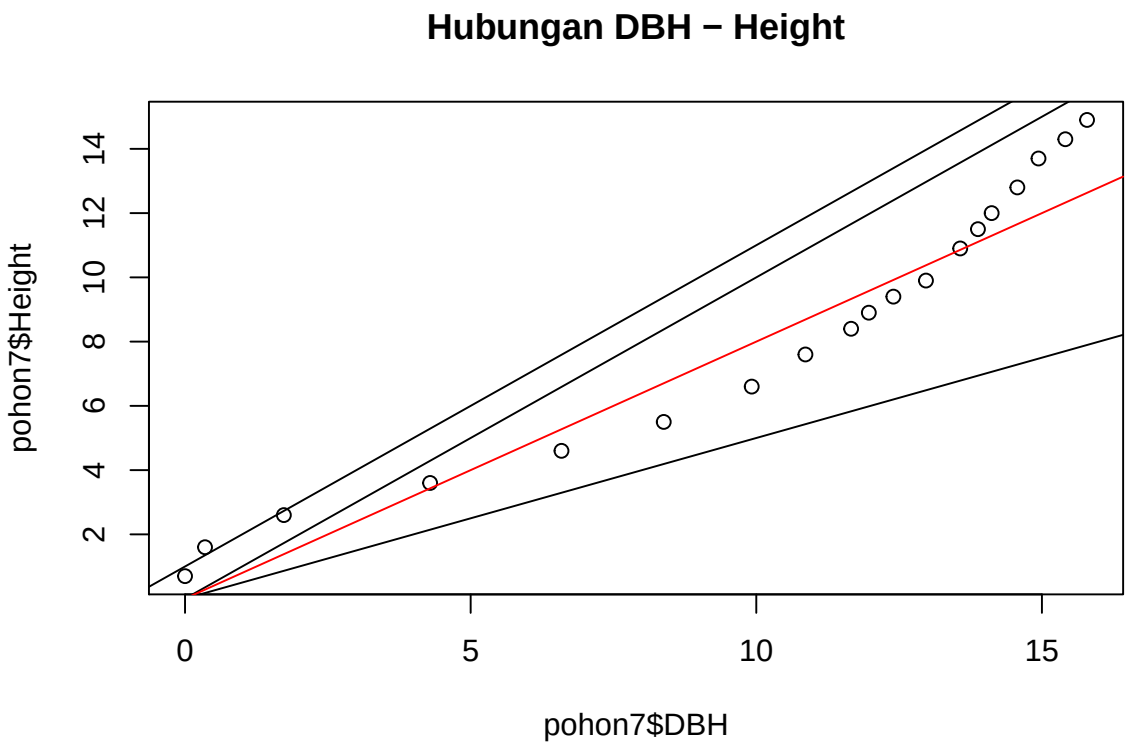
```
##   treeID Age  DBH Height Volume
## 1      7   2 0.00    0.7  0.000
## 2      7   3 0.35    1.6  0.000
## 3      7   4 1.73    2.6  0.001
## 4      7   5 4.29    3.6  0.004
## 5      7   6 6.59    4.6  0.010
## 6      7   7 8.38    5.5  0.018
```

```
tail(pohon7)
```

```
##    treeID Age   DBH Height Volume
## 14      7  15 13.88   11.5  0.090
## 15      7  16 14.12   12.0  0.098
## 16      7  17 14.57   12.8  0.110
## 17      7  18 14.94   13.7  0.122
## 18      7  19 15.41   14.3  0.135
## 19      7  20 15.79   14.9  0.148
```

- menampilkan visualisasi hubungan "DBH" dan "Height".

```
plot(pohon7$DBH, pohon7$Height, main = "Hubungan DBH - Height")
abline(a = 0, b = 1)
abline(a = 0, b = 0.5)
abline(a = 1, b = 1)
abline(a = 0, b = 0.8, col = "red")
```



- sintaks model *linear* tunggal sebagai berikut.
  `lm(Y~X, data="nama_dataset")`

- menjalankan model *lm* antara *variable* "Height" dan variable "DBH".
  - hasilnya, kita *assign* sebagai *variable* bernama "lm_pohon7".

```
lm_pohon7 <- lm(Height ~ DBH, data = pohon7)
lm_pohon7
```

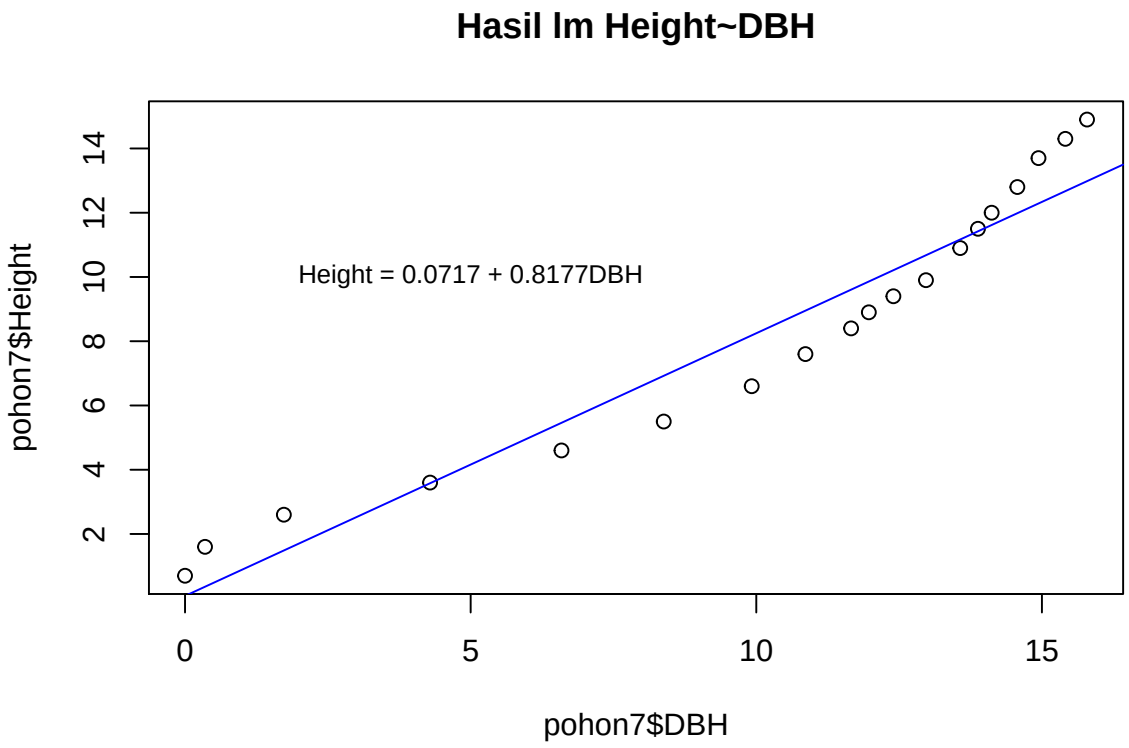
```
##
## Call:
## lm(formula = Height ~ DBH, data = pohon7)
##
## Coefficients:
## (Intercept)          DBH
##      0.0717       0.8177
```

- model di atas menghasilkan nilai di bawah ini.
  - α = *intercept* = a = 0.0717.
  - β = *slope* = b = 0.8177.
  - sehingga hasil model empirisnya adalah sebagai berikut.

$$Height = 0.0717 + 0.8177DBH$$

- menampilkan visualiasi hasil model.

```
plot(pohon7$DBH, pohon7$Height, main = "Hasil lm Height~DBH")
abline(lm_pohon7, col = "blue")
text(5, 10, "Height = 0.0717 + 0.8177DBH", cex = 0.8)
```



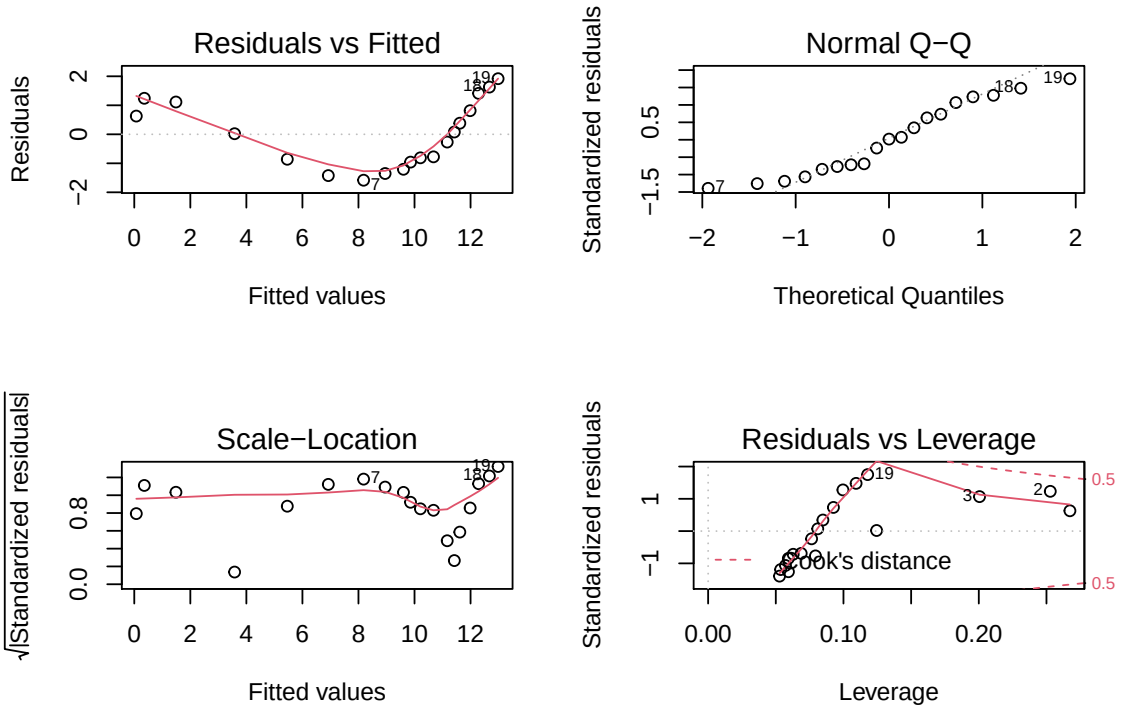
- menampilkan ringkasan dari hasil model.

```
summary(lm_pohon7)
```

```
##
## Call:
## lm(formula = Height ~ DBH, data = pohon7)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.58300 -0.90969  0.02049  0.96427  1.91727
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.07170    0.60300   0.119    0.907
## DBH          0.81767    0.05309  15.402 2.03e-11 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.166 on 17 degrees of freedom
## Multiple R-squared:  0.9331, Adjusted R-squared:  0.9292
## F-statistic: 237.2 on 1 and 17 DF,  p-value: 2.034e-11
```

- menampilkan visualisasi model.

```
par(mfrow = c(2, 2))  # menyiapkan kerangka grafik: 2 baris dan 2 kolom
plot(lm_pohon7)
```



5.2.2 *Multiple Linear Model*

Multiple linear model mempunyai *independent variable* lebih dari satu. Model teoretis (*theoretical model*) dari *multiple linear model* adalah sebagai berikut.

$$Y = \alpha + \beta_1 X_1 + \beta_2 X_2$$

Latihan di bawah ini menggunakan *dataset* "pohon", untuk menelaah *variables* yang mempengaruhi "Volume".

- dari sintaks `cor()`, kita bisa menampilkan keterhubungan (*correlation*) antar *variable*.

```
cor(pohon)
```

```
##             treeID       Age         DBH      Height      Volume
## treeID  1.00000000 0.0000000 0.01538271 -0.02549617 -0.04616717
## Age     0.00000000 1.0000000 0.94567643  0.99424875  0.97336552
## DBH     0.01538271 0.9456764 1.00000000  0.96877317  0.88701431
## Height -0.02549617 0.9942488 0.96877317  1.00000000  0.96358634
## Volume -0.04616717 0.9733655 0.88701431  0.96358634  1.00000000
```

Agar lebih sederhana, kita membatasi pada dua *independent variable* ("DBH" dan "Height") terhadap "Volume", dengan model empiris sebagai berikut.

$$Volume = \alpha + \beta_1 DBH + \beta_2 Height$$

- sintaks *multiple linear model* sebagai berikut.
  `lm(Y~X1+X2, data="nama_dataset")`
- menjalankan dan menyimpan model empiris *lm* tidak tunggal, dengan nama "lmm_pohon".

```
lmm_pohon <- lm(Volume ~ DBH + Height, data = pohon)
lmm_pohon
```

```
##
## Call:
## lm(formula = Volume ~ DBH + Height, data = pohon)
##
## Coefficients:
## (Intercept)          DBH       Height
##   -0.023080    -0.006463     0.016939
```

```
summary(lmm_pohon)
```

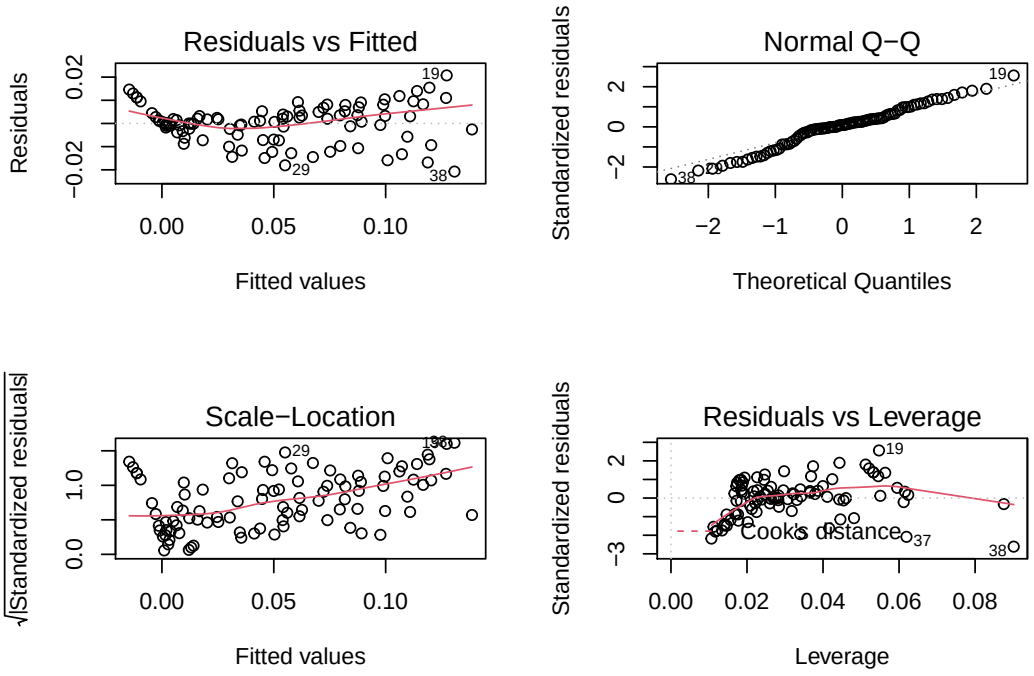
```
##
## Call:
## lm(formula = Volume ~ DBH + Height, data = pohon)
##
## Residuals:
##       Min         1Q     Median         3Q        Max
## -0.0207555 -0.0043590  0.0007435  0.0047002  0.0207307
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -0.0230801  0.0018503  -12.47  < 2e-16 ***
## DBH         -0.0064626  0.0006853   -9.43 3.61e-15 ***
## Height       0.0169392  0.0008007   21.15  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.008327 on 92 degrees of freedom
## Multiple R-squared:  0.9636, Adjusted R-squared:  0.9629
## F-statistic:  1219 on 2 and 92 DF,  p-value: < 2.2e-16
```

Dari hasil di atas, maka hasil model empirisnya adalah sebagai berikut.

$$Volume = -0.023080 - 0.006463DBH + 0.016939Height$$

- menampilkan visualisasi dari hasil model empiris.

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



5.3 *Nonlinear Model - Nonlinear Least Square (nls)*

Pada beberapa kasus, hubungan antara *independent variable* X dan *dependent variable* Y tidak selalu bersifat *linear*. Sesi ini menyajikan dasar pemahaman terhadap *non-linear model*.

Latihan menggunakan kembali *dataset* "pohon7". Dan, telaah difokuskan pada keterhubungan antara "Age" (sebagai *independent variable*) dan "DBH" (sebagai *dependent variable*).

- menginspeksi *dataset* "pohon7".

```
head(pohon7)
```

```
##   treeID Age  DBH Height Volume
## 1      7   2 0.00    0.7  0.000
## 2      7   3 0.35    1.6  0.000
## 3      7   4 1.73    2.6  0.001
## 4      7   5 4.29    3.6  0.004
## 5      7   6 6.59    4.6  0.010
## 6      7   7 8.38    5.5  0.018
```

Terlihat bahwa baris ke-1 dan ke-2 mempunyai nilai "Volume" = 0, sehingga kita tidak ingin menggunakan data baris ke-1 dan ke-2.

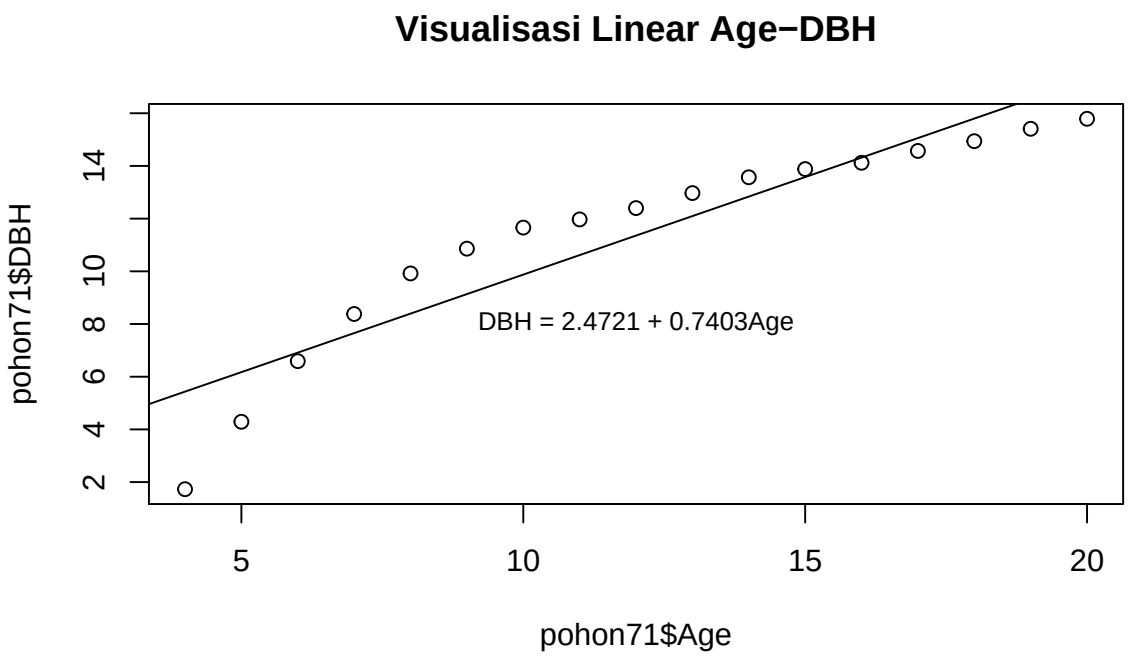
- mengekstrak dan membuat *dataset* baru (dinamakan "pohon71"), yang berisi seluruh dari dari *dataset* "pohon7" yang mempunyai "Volume">0.

```
pohon71 <- pohon7[pohon7$Volume > 0, ]  # mengekstrak dataset 'pohon7'
# yang 'Volume'>0
head(pohon71)
```

```
##   treeID Age   DBH Height Volume
## 3      7   4  1.73    2.6  0.001
## 4      7   5  4.29    3.6  0.004
## 5      7   6  6.59    4.6  0.010
## 6      7   7  8.38    5.5  0.018
## 7      7   8  9.92    6.6  0.027
## 8      7   9 10.86    7.6  0.035
```

- menampilkan visualisasi hubungan "Age" dan "DBH".

```
plot(pohon71$Age, pohon71$DBH, main = "Visualisasi Linear Age-DBH")
lm_pohon71 <- lm(DBH ~ Age, data = pohon71)
abline(lm_pohon71)
text(12, 8, "DBH = 2.4721 + 0.7403Age", cex = 0.8)
```



Terlihat bahwa sebaran observasi bersifat *sigmoid*, dan terlihat bahwa *linear model* kurang sesuai untuk memodelkan hubungan antara *Age* dan *DBH*.

- salah satu model *non-linear sigmoid curve* atau *growth analysis* adalah *Bertalanffy function*, dengan model teoretis sebagai berikut.

$$Y = \alpha(1 - e^{-\beta X})^3$$

Dengan demikian, model empiris yang dibangun adalah sebagai berikut.

$$DBH = \alpha(1 - e^{-\beta Age})^3$$

- menampilkan visualisasi *scatterplot* dari kedua *variables*.
- dan, melakukan *trial and error* untuk mencari nilai α (“a”) dan nilai β (“b”) yang paling mendekati.

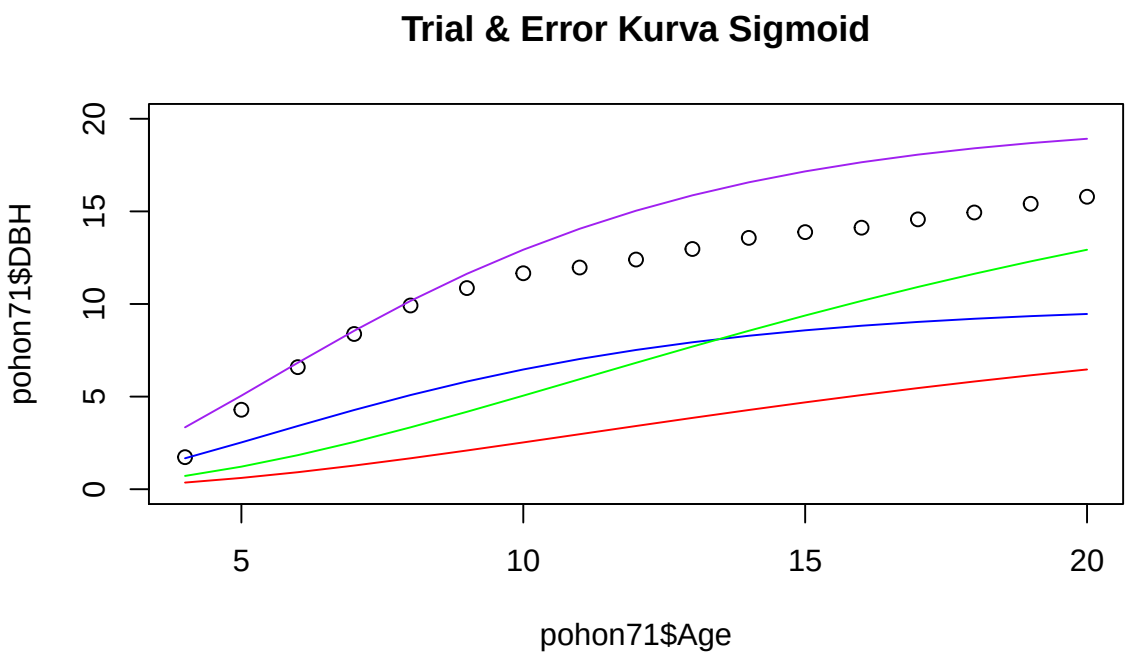
```
plot(pohon71$Age, pohon71$DBH, ylim = c(0, 20), main = "Trial & Error Kurva Sigmoid")

y1 <- 10 * (1 - exp(-0.1 * pohon71$Age))^3
lines(pohon71$Age, y1, col = "red")

y2 <- 10 * (1 - exp(-0.2 * pohon71$Age))^3
lines(pohon71$Age, y2, col = "blue")

y3 <- 20 * (1 - exp(-0.1 * pohon71$Age))^3
lines(pohon71$Age, y3, col = "green")

y4 <- 20 * (1 - exp(-0.2 * pohon71$Age))^3
lines(pohon71$Age, y4, col = "purple")
```



Terlihat bahwa *lines* yang paling mendekati observasi adalah *lines* dengan nilai a=20 dan b=0.2 (garis warna ungu).

- menjalankan model empiris dengan menggunakan nilai a dan b di atas.

```
nls_bert_pohon71 <- nls(pohon71$DBH ~ a * (1 - exp(-b * pohon71$Age))^3,
    data = pohon71, start = list(a = 20, b = 0.2))
nls_bert_pohon71
```

```
## Nonlinear regression model
##   model: pohon71$DBH ~ a * (1 - exp(-b * pohon71$Age))^3
##    data: pohon71
##       a       b
## 15.7731  0.2244
##  residual sum-of-squares: 5.138
##
## Number of iterations to convergence: 4
## Achieved convergence tolerance: 1.763e-06
```

```
summary(nls_bert_pohon71)
```

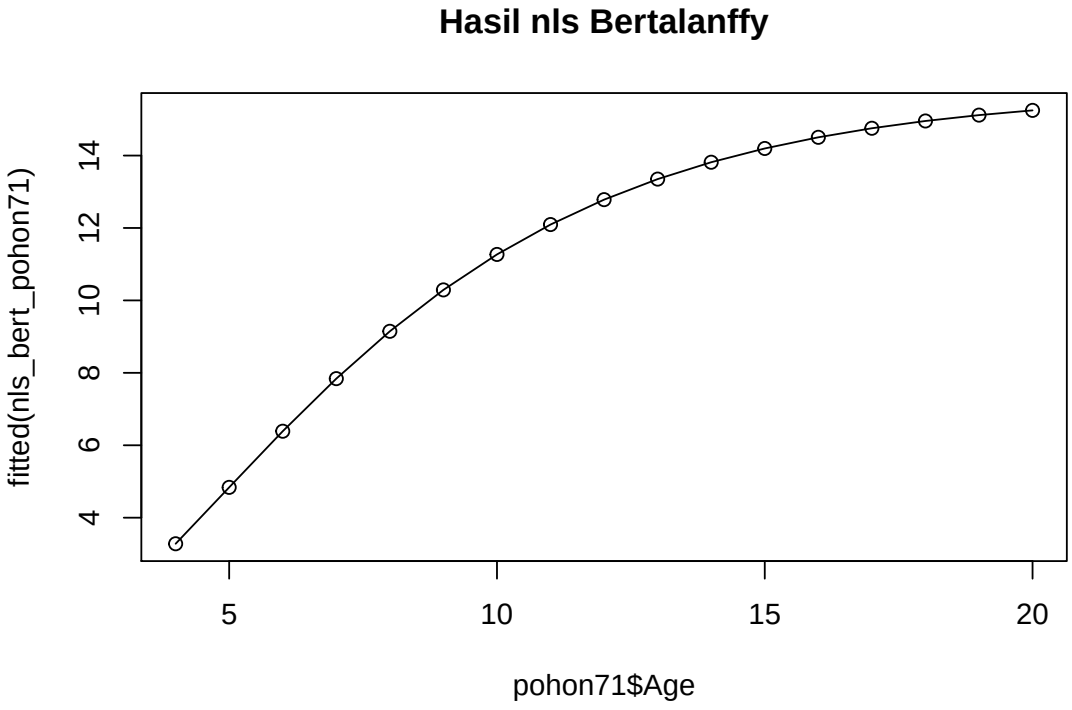
```
##
## Formula: pohon71$DBH ~ a * (1 - exp(-b * pohon71$Age))^3
##
## Parameters:
##    Estimate Std. Error t value Pr(>|t|)
## a 15.773137   0.324666   48.58  < 2e-16 ***
## b  0.224413   0.007923   28.32 1.94e-14 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5853 on 15 degrees of freedom
##
## Number of iterations to convergence: 4
## Achieved convergence tolerance: 1.763e-06
```

Dari hasil tersebut, maka hasil model empirisnya adalah sebagai berikut.

$$DBH = 15.7731(1 - e^{-0.2244Age})^3$$

- menampilkan visualisasi data observasi dan data hasil model empiris.

```
plot(pohon71$Age, fitted(nls_bert_pohon71), type = "o", main = "Hasil nls Bertalanffy
```



Bab ini menyajikan secara sederhana contoh penerapan bahasa *R* untuk statistik deskriptif maupun inferensia, baik *linear regression* (*lm*) maupun *nonlinear regression* (*nls*).

Catatan Penutup

Sebagaimana judulnya, ***Dasar R***, buku/modul ini menyajikan hal-hal dasar tentang bahasa *R*. Juga, materi yang disajikan relatif sederhana, dengan hanya memanfaatkan kapasitas *built-in* yang sudah ada di bahasa *R*, tanpa menambahkan suatu *modul* atau *package* tambahan. Selain itu, latihan-latihan juga hanya menggunakan satu *dataset* dari awal sampai akhir. Harapannya, buku/modul ini bisa memberikan pengenalan awal sehingga siapapun bisa menikmati percakapan dengan bahasa *R*.

Ke depan, sangat dimungkinkan untuk dilakukan pemutakhiran (*updating*) materi, baik berupa penambahan, koreksi, modifikasi atau bahkan pengurangan, untuk lebih menyesuaikan kebutuhan pengguna. Oleh karena itu, kritikan dan masukan sangat dinantikan melalui email: **j.indarto@gmail.com**. Buku/modul dan data latihan juga disediakan secara bebas di situs **https://indarto.weebly.com** ini.

Alhamdulillaahirrabbiláalamiin. . .