

# R

## Notes for Professionals

### Chapter 20: Reading and writing tabular data in plain-text files (CSV, TSV, etc.)

**Parameter**

- file
- header
- sep
- quote
- dec
- fill
- comment.char
- char
- extra arguments to be passed to `read.table`

**Details**

name of the CSV file to read  
logical: does the CSV file contain a header row with column names?  
character: symbol that separates the cells on each row  
character: symbol used to quote character strings  
character: symbol used as decimal separator  
logical: when TRUE, rows with unequal length are filled with blank fields  
comment.char: character used as comment in the CSV file. Lines preceded by this character are ignored  
extra arguments to be passed to `read.table`

#### Section 20.1: Importing .csv files

##### Importing using base R

Comma separated value files (CSVs) can be imported using `read.csv`, which wraps `read.table` to set the delimiter to a comma.

```
# get the file path of a CSV included in R's utils package
csv_path <- system.file("misc", "adult.csv", package = "utils")
```

# you will have to install location
`csv_path` <- c("C:/Program Files/R/R-3.6.0/library/rmsmodels/R/framework/Resources/library/csvs/adult.csv")

```
df <- read.csv(csv_path)
```

Or

as

as V1 V2 V3

as Y 1 2 3 4 5

as Z 1.14 A

as 3 10.00 X

as 4 -7.00 A

As an friendly option, `file.choose`, allows to browse through the directories

df <- read.csv(file.choose())

**Notes**

• Unlike `read.table`, `read.csv` defaults to `header = TRUE`, and uses 0 as the default for `na.strings`.

• All three functions will convert strings to `factor` class by default unless `stringsAsFactors = FALSE`.

• The `read.csv2` variant defaults to `sep = ","` and `dec = ","`, for US commas is used as a decimal point and the semicolon as a field separator.

**Importing using packages**

The `readr` package's `read_csv` function offers much faster performance, a progress bar for large files, and more popular default options than standard `read.csv`, including `stringAsFactors = FALSE`.

[Read more for Professionals](#)

### Chapter 92: I/O for foreign tables (Excel, SAS, SPSS, Stata)

#### Section 92.1: Importing data with rio

A very simple way to import data from many common file formats is with `rio`. This package provides a function `import()` that wraps many commonly used data import functions, thereby providing a standard interface. It works simply by passing a file name or URL to `import()`:

```
library("rio")
import("example.csv") # comma-separated values
import("example.rda") # RDA-separated values
import("example.dta") # Stata
import("example.mva") # SPSS
import("example.xls?biff=1") # SAS
import("example.xlsx") # Excel
```

`import()` can also read from compressed directories, URLs (HTTP or HTTPS), and the clipboard. A comprehensive list of all supported file formats is available on the [rio package GitHub repository](#).

It is even possible to specify some further parameters related to the specific file format you are trying to read: passing them directly within the `import()` function:

```
import("example.csv", format = ",") # for csv file where comma is used as separator
import("example.csv", format = ".") # for csv file where period is used as separator
```

#### Section 92.2: Read and write Stata, SPSS and SAS files

The packages `foreign` and `haven` can be used to import and export files from a variety of other statistical packages like Stata, SPSS and SAS and related software. There is a `read` function for each of the supported data types to import the files.

```
# loading the packages
library(foreign)
library(haven)
library(readr)
library(mosaic)
```

Some examples for the most common data types:

```
# reading Stata files with `foreign`
read_dta("path/to/your/dta")
# reading Stata files with `haven`
read_dta("path/to/your/dta")
```

The `foreign` package can read in `stata` (`dta`) files for versions of Stata 7-12. According to the development plan, `read_dta` is more or less frozen and will not be updated for reading in versions 13+. For more recent versions of Stata, you can use either the `readstata13` package or `haven`. For `readstata13`, the files are

```
# reading recent Stata (13+) files with `readstata13`
read_dta("path/to/your/dta")
```

For reading in SPSS and SAS files

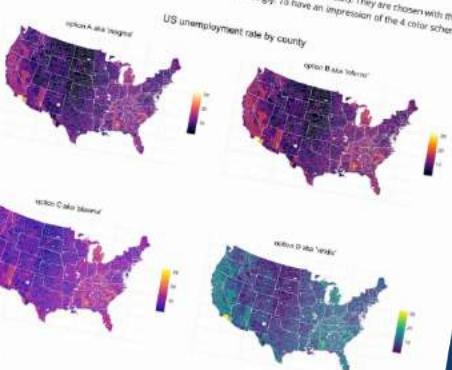
[Read more for Professionals](#)

### Chapter 106: Color schemes for graphics

#### Section 106.1: viridis - print and colorblind friendly palettes

`viridis` (named after the [classic viridis colormap](#)) is a recently developed color scheme for the Python library `matplotlib`. The video presentation by the link explains how the color scheme was developed and what are its main advantages. It is seamlessly ported to R.

There are 4 variants of color schemes: `viridis`, `plasma`, `inferno`, and `viridis_r` (default). They are chosen with the `option` parameter and are coded as A, B, C, and D, correspondingly. To have an impression of the 4 color schemes, look at the maps:



##### Image usage

The package can be installed from [CRAN](#) or [GitHub](#).

The name for `viridis` package is just brilliant.

Nice feature of the `viridis` color scheme is integration with `ggplot2`. Within the package two `ggplot2`-specific functions are defined: `scale_color_viridis()` and `scale_fill_viridis()`. See the example below:

```
library(viridis)
library(ggplot2)
gg1 <- ggplot(mtcars)
```

[Read more for Professionals](#)

**400+ pages**  
of professional hints and tricks

# Contents

<a href="#">About</a>	1
<b><a href="#">Chapter 1: Getting started with R Language</a></b>	2
<a href="#">Section 1.1: Installing R</a>	2
<a href="#">Section 1.2: Hello World!</a>	3
<a href="#">Section 1.3: Getting Help</a>	3
<a href="#">Section 1.4: Interactive mode and R scripts</a>	3
<b><a href="#">Chapter 2: Variables</a></b>	7
<a href="#">Section 2.1: Variables, data structures and basic Operations</a>	7
<b><a href="#">Chapter 3: Arithmetic Operators</a></b>	10
<a href="#">Section 3.1: Range and addition</a>	10
<a href="#">Section 3.2: Addition and subtraction</a>	10
<b><a href="#">Chapter 4: Matrices</a></b>	13
<a href="#">Section 4.1: Creating matrices</a>	13
<b><a href="#">Chapter 5: Formula</a></b>	15
<a href="#">Section 5.1: The basics of formula</a>	15
<b><a href="#">Chapter 6: Reading and writing strings</a></b>	17
<a href="#">Section 6.1: Printing and displaying strings</a>	17
<a href="#">Section 6.2: Capture output of operating system command</a>	18
<a href="#">Section 6.3: Reading from or writing to a file connection</a>	19
<b><a href="#">Chapter 7: String manipulation with stringi package</a></b>	21
<a href="#">Section 7.1: Count pattern inside string</a>	21
<a href="#">Section 7.2: Duplicating strings</a>	21
<a href="#">Section 7.3: Paste vectors</a>	22
<a href="#">Section 7.4: Splitting text by some fixed pattern</a>	22
<b><a href="#">Chapter 8: Classes</a></b>	23
<a href="#">Section 8.1: Inspect classes</a>	23
<a href="#">Section 8.2: Vectors and lists</a>	23
<a href="#">Section 8.3: Vectors</a>	24
<b><a href="#">Chapter 9: Lists</a></b>	25
<a href="#">Section 9.1: Introduction to lists</a>	25
<a href="#">Section 9.2: Quick Introduction to Lists</a>	25
<a href="#">Section 9.3: Serialization: using lists to pass information</a>	27
<b><a href="#">Chapter 10: Hashmaps</a></b>	29
<a href="#">Section 10.1: Environments as hash maps</a>	29
<a href="#">Section 10.2: package:hash</a>	32
<a href="#">Section 10.3: package:listenv</a>	33
<b><a href="#">Chapter 11: Creating vectors</a></b>	35
<a href="#">Section 11.1: Vectors from build in constants: Sequences of letters &amp; month names</a>	35
<a href="#">Section 11.2: Creating named vectors</a>	35
<a href="#">Section 11.3: Sequence of numbers</a>	37
<a href="#">Section 11.4: seq()</a>	37
<a href="#">Section 11.5: Vectors</a>	38
<a href="#">Section 11.6: Expanding a vector with the rep() function</a>	39
<b><a href="#">Chapter 12: Date and Time</a></b>	41
<a href="#">Section 12.1: Current Date and Time</a>	41

<a href="#">Section 12.2: Go to the End of the Month</a>	41
<a href="#">Section 12.3: Go to First Day of the Month</a>	42
<a href="#">Section 12.4: Move a date a number of months consistently by months</a>	42
<b>Chapter 13: The Date class</b>	44
<a href="#">Section 13.1: Formatting Dates</a>	44
<a href="#">Section 13.2: Parsing Strings into Date Objects</a>	44
<a href="#">Section 13.3: Dates</a>	45
<b>Chapter 14: Date-time classes (POSIXct and POSIXlt)</b>	47
<a href="#">Section 14.1: Formatting and printing date-time objects</a>	47
<a href="#">Section 14.2: Date-time arithmetic</a>	47
<a href="#">Section 14.3: Parsing strings into date-time objects</a>	48
<b>Chapter 15: The character class</b>	50
<a href="#">Section 15.1: Coercion</a>	50
<b>Chapter 16: Numeric classes and storage modes</b>	51
<a href="#">Section 16.1: Numeric</a>	51
<b>Chapter 17: The logical class</b>	53
<a href="#">Section 17.1: Logical operators</a>	53
<a href="#">Section 17.2: Coercion</a>	53
<a href="#">Section 17.3: Interpretation of NAs</a>	53
<b>Chapter 18: Data frames</b>	55
<a href="#">Section 18.1: Create an empty data.frame</a>	55
<a href="#">Section 18.2: Subsetting rows and columns from a data frame</a>	56
<a href="#">Section 18.3: Convenience functions to manipulate data.frames</a>	59
<a href="#">Section 18.4: Introduction</a>	60
<a href="#">Section 18.5: Convert all columns of a data.frame to character class</a>	61
<b>Chapter 19: Split function</b>	63
<a href="#">Section 19.1: Using split in the split-apply-combine paradigm</a>	63
<a href="#">Section 19.2: Basic usage of split</a>	64
<b>Chapter 20: Reading and writing tabular data in plain-text files (CSV, TSV, etc.)</b>	67
<a href="#">Section 20.1: Importing .csv files</a>	67
<a href="#">Section 20.2: Importing with data.table</a>	68
<a href="#">Section 20.3: Exporting .csv files</a>	69
<a href="#">Section 20.4: Import multiple csv files</a>	69
<a href="#">Section 20.5: Importing fixed-width files</a>	69
<b>Chapter 21: Pipe operators (%&gt;% and others)</b>	71
<a href="#">Section 21.1: Basic use and chaining</a>	71
<a href="#">Section 21.2: Functional sequences</a>	72
<a href="#">Section 21.3: Assignment with %&lt;&gt;%</a>	73
<a href="#">Section 21.4: Exposing contents with %\$%</a>	73
<a href="#">Section 21.5: Creating side effects with %T&gt;%</a>	74
<a href="#">Section 21.6: Using the pipe with dplyr and ggplot2</a>	75
<b>Chapter 22: Linear Models (Regression)</b>	76
<a href="#">Section 22.1: Linear regression on the mtcars dataset</a>	76
<a href="#">Section 22.2: Using the 'predict' function</a>	78
<a href="#">Section 22.3: Weighting</a>	79
<a href="#">Section 22.4: Checking for nonlinearity with polynomial regression</a>	81
<a href="#">Section 22.5: Plotting The Regression (base)</a>	83
<a href="#">Section 22.6: Quality assessment</a>	85
<b>Chapter 23: data.table</b>	87

<a href="#">Section 23.1: Creating a data.table</a>	87
<a href="#">Section 23.2: Special symbols in data.table</a>	88
<a href="#">Section 23.3: Adding and modifying columns</a>	89
<a href="#">Section 23.4: Writing code compatible with both data.frame and data.table</a>	91
<a href="#">Section 23.5: Setting keys in data.table</a>	93
<b>Chapter 24: Pivot and unpivot with data.table</b>	95
<a href="#">Section 24.1: Pivot and unpivot tabular data with data.table - I</a>	95
<a href="#">Section 24.2: Pivot and unpivot tabular data with data.table - II</a>	96
<b>Chapter 25: Bar Chart</b>	98
<a href="#">Section 25.1: barplot() function</a>	98
<b>Chapter 26: Base Plotting</b>	104
<a href="#">Section 26.1: Density plot</a>	104
<a href="#">Section 26.2: Combining Plots</a>	105
<a href="#">Section 26.3: Getting Started with R_Plots</a>	107
<a href="#">Section 26.4: Basic Plot</a>	108
<a href="#">Section 26.5: Histograms</a>	111
<a href="#">Section 26.6: Matplot</a>	113
<a href="#">Section 26.7: Empirical Cumulative Distribution Function</a>	119
<b>Chapter 27: boxplot</b>	121
<a href="#">Section 27.1: Create a box-and-whisker plot with boxplot() {graphics}</a>	121
<a href="#">Section 27.2: Additional boxplot style parameters</a>	125
<b>Chapter 28: ggplot2</b>	128
<a href="#">Section 28.1: Displaying multiple plots</a>	128
<a href="#">Section 28.2: Prepare your data for plotting</a>	131
<a href="#">Section 28.3: Add horizontal and vertical lines to plot</a>	133
<a href="#">Section 28.4: Scatter Plots</a>	136
<a href="#">Section 28.5: Produce basic plots with qplot</a>	136
<a href="#">Section 28.6: Vertical and Horizontal Bar Chart</a>	138
<a href="#">Section 28.7: Violin plot</a>	140
<b>Chapter 29: Factors</b>	143
<a href="#">Section 29.1: Consolidating Factor Levels with a List</a>	143
<a href="#">Section 29.2: Basic creation of factors</a>	144
<a href="#">Section 29.3: Changing and reordering factors</a>	145
<a href="#">Section 29.4: Rebuilding factors from zero</a>	150
<b>Chapter 30: Pattern Matching and Replacement</b>	152
<a href="#">Section 30.1: Finding Matches</a>	152
<a href="#">Section 30.2: Single and Global match</a>	153
<a href="#">Section 30.3: Making substitutions</a>	154
<a href="#">Section 30.4: Find matches in big data sets</a>	154
<b>Chapter 31: Run-length encoding</b>	156
<a href="#">Section 31.1: Run-length Encoding with `rle`</a>	156
<a href="#">Section 31.2: Identifying and grouping by runs in base R</a>	156
<a href="#">Section 31.3: Run-length encoding to compress and decompress vectors</a>	157
<a href="#">Section 31.4: Identifying and grouping by runs in data.table</a>	158
<b>Chapter 32: Speeding up tough-to-vectorize code</b>	159
<a href="#">Section 32.1: Speeding tough-to-vectorize for loops with Rcpp</a>	159
<a href="#">Section 32.2: Speeding tough-to-vectorize for loops by byte compiling</a>	159
<b>Chapter 33: Introduction to Geographical Maps</b>	161
<a href="#">Section 33.1: Basic map-making with map() from the package maps</a>	161

<a href="#">Section 33.2: 50 State Maps and Advanced Choropleths with Google Viz</a>	164
<a href="#">Section 33.3: Interactive plotly maps</a>	165
<a href="#">Section 33.4: Making Dynamic HTML Maps with Leaflet</a>	167
<a href="#">Section 33.5: Dynamic Leaflet maps in Shiny applications</a>	168
<b>Chapter 34: Set operations</b>	171
<a href="#">Section 34.1: Set operators for pairs of vectors</a>	171
<a href="#">Section 34.2: Cartesian or "cross" products of vectors</a>	171
<a href="#">Section 34.3: Set membership for vectors</a>	172
<a href="#">Section 34.4: Make unique / drop duplicates / select distinct elements from a vector</a>	172
<a href="#">Section 34.5: Measuring set overlaps / Venn diagrams for vectors</a>	173
<b>Chapter 35: tidyverse</b>	174
<a href="#">Section 35.1: tidyverse: an overview</a>	174
<a href="#">Section 35.2: Creating <code>tbl_df</code>'s</a>	175
<b>Chapter 36: Rcpp</b>	176
<a href="#">Section 36.1: Extending Rcpp with Plugins</a>	176
<a href="#">Section 36.2: Inline Code Compile</a>	176
<a href="#">Section 36.3: Rcpp Attributes</a>	177
<a href="#">Section 36.4: Specifying Additional Build Dependencies</a>	178
<b>Chapter 37: Random Numbers Generator</b>	179
<a href="#">Section 37.1: Random permutations</a>	179
<a href="#">Section 37.2: Generating random numbers using various density functions</a>	179
<a href="#">Section 37.3: Random number generator's reproducibility</a>	181
<b>Chapter 38: Parallel processing</b>	182
<a href="#">Section 38.1: Parallel processing with <code>parallel</code> package</a>	182
<a href="#">Section 38.2: Parallel processing with <code>foreach</code> package</a>	183
<a href="#">Section 38.3: Random Number Generation</a>	184
<a href="#">Section 38.4: <code>mcpallelDo</code></a>	184
<b>Chapter 39: Subsetting</b>	186
<a href="#">Section 39.1: Data frames</a>	186
<a href="#">Section 39.2: Atomic vectors</a>	187
<a href="#">Section 39.3: Matrices</a>	188
<a href="#">Section 39.4: Lists</a>	190
<a href="#">Section 39.5: Vector indexing</a>	191
<a href="#">Section 39.6: Other objects</a>	192
<a href="#">Section 39.7: Elementwise Matrix Operations</a>	192
<b>Chapter 40: Debugging</b>	194
<a href="#">Section 40.1: Using <code>debug</code></a>	194
<a href="#">Section 40.2: Using <code>browser</code></a>	194
<b>Chapter 41: Installing packages</b>	196
<a href="#">Section 41.1: Install packages from GitHub</a>	196
<a href="#">Section 41.2: Download and install packages from repositories</a>	197
<a href="#">Section 41.3: Install package from local source</a>	198
<a href="#">Section 41.4: Install local development version of a package</a>	198
<a href="#">Section 41.5: Using a CLI package manager -- basic <code>pacman</code> usage</a>	199
<b>Chapter 42: Inspecting packages</b>	200
<a href="#">Section 42.1: View Package Version</a>	200
<a href="#">Section 42.2: View Loaded packages in Current Session</a>	200
<a href="#">Section 42.3: View package information</a>	200
<a href="#">Section 42.4: View package's built-in data sets</a>	200

[Click here to download full PDF material](#)