

# Data Wrangling in R

*Federico Lopez*

*November 13, 2015*

## Contents

<b>An introduction to manipulating data with <code>tidyr</code> and <code>dplyr</code></b>	<b>1</b>
<b>Reshaping data</b>	<b>2</b>
Gather columns into rows . . . . .	2
Spread rows into columns . . . . .	4
Order rows by values . . . . .	4
Rename the columns of a data frame . . . . .	5
<b>Manipulating data (filter, select, mutate, bind etc.)</b>	<b>5</b>
Subset observations (rows) . . . . .	5
Subset variables (columns) . . . . .	10
Summarize data . . . . .	13
Group data . . . . .	13
Make new variables . . . . .	14
Combine data sets . . . . .	16
<b>Using <code>magrittr</code> to create pipelines</b>	<b>19</b>
<b>Pattern matching and replacement</b>	<b>23</b>
Position of pattern within the string . . . . .	23
Operators . . . . .	24

## An introduction to manipulating data with `tidyr` and `dplyr`

`tidyr` is designed specifically for data tidying (not general reshaping or aggregating) and works well with ‘`dplyr`’ data pipelines. Tidy data ensures that values of different variables from the same observation are always paired. In tidy data:

- Each variable forms a column
- Each observation forms a row
- Each type of observational unit forms a table

For further detail see <https://github.com/hadley/tidyr> and <https://ramnathv.github.io/pycon2014-r/explore/tidy.html>

`dplyr` provides several functions for manipulating data frames. It is a new iteration of the `plyr` package, which implements the “split-apply-combine” strategy for data analysis. Here we will go over a few examples of data manipulation using `dplyr`.

# Reshaping data

## Gather columns into rows

```
# Install and load tidyr and dplyr
#install.packages("tidyr")
#install.packages("dplyr")
library(tidyr)
library(dplyr)

# Load the Iris data
# This data frame includes 150 obs. of 5 variables
str(iris)
```

```
## 'data.frame': 150 obs. of 5 variables:
## $ Sepal.Length: num 5.1 4.9 4.7 4.6 5 5.4 4.6 5 4.4 4.9 ...
## $ Sepal.Width : num 3.5 3 3.2 3.1 3.6 3.9 3.4 3.4 2.9 3.1 ...
## $ Petal.Length: num 1.4 1.4 1.3 1.5 1.4 1.7 1.4 1.5 1.4 1.5 ...
## $ Petal.Width : num 0.2 0.2 0.2 0.2 0.2 0.4 0.3 0.2 0.2 0.1 ...
## $ Species : Factor w/ 3 levels "setosa","versicolor",...: 1 1 1 1 1 1 1 1 1 1 ...
```

```
head(iris)
```

```
## Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## 1 5.1 3.5 1.4 0.2 setosa
## 2 4.9 3.0 1.4 0.2 setosa
## 3 4.7 3.2 1.3 0.2 setosa
## 4 4.6 3.1 1.5 0.2 setosa
## 5 5.0 3.6 1.4 0.2 setosa
## 6 5.4 3.9 1.7 0.4 setosa
```

```
# Use the tbl_df from dplyr to transform a regular data frame to a tbl object
# "tbl objects only print a few rows and all the columns that fit on one screen"
# (transforming data frames into tbl_dfs is not a requirement to use dplyr)
iris <- tbl_df(iris)
iris
```

```
## Source: local data frame [150 x 5]
##
## Sepal.Length Sepal.Width Petal.Length Petal.Width Species
## (dbl) (dbl) (dbl) (dbl) (fctr)
## 1 5.1 3.5 1.4 0.2 setosa
## 2 4.9 3.0 1.4 0.2 setosa
## 3 4.7 3.2 1.3 0.2 setosa
## 4 4.6 3.1 1.5 0.2 setosa
## 5 5.0 3.6 1.4 0.2 setosa
## 6 5.4 3.9 1.7 0.4 setosa
## 7 4.6 3.4 1.4 0.3 setosa
## 8 5.0 3.4 1.5 0.2 setosa
## 9 4.4 2.9 1.4 0.2 setosa
## 10 4.9 3.1 1.5 0.1 setosa
## .. ... .. ... .. ...
```

```
# Summarize tbl data
glimpse(iris)
```

```
## Observations: 150
## Variables: 5
## $ Sepal.Length (dbl) 5.1, 4.9, 4.7, 4.6, 5.0, 5.4, 4.6, 5.0, 4.4, 4.9,...
## $ Sepal.Width (dbl) 3.5, 3.0, 3.2, 3.1, 3.6, 3.9, 3.4, 3.4, 2.9, 3.1,...
## $ Petal.Length (dbl) 1.4, 1.4, 1.3, 1.5, 1.4, 1.7, 1.4, 1.5, 1.4, 1.5,...
## $ Petal.Width (dbl) 0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1,...
## $ Species (fctr) setosa, setosa, setosa, setosa, setosa, setosa, setosa, ...
```

```
# Get first observation for each Species in iris data
mini.iris <- iris[c(1, 51, 101), ]
mini.iris
```

```
## Source: local data frame [3 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl)        (fctr)
## 1         5.1         3.5         1.4         0.2      setosa
## 2         7.0         3.2         4.7         1.4 versicolor
## 3         6.3         3.3         6.0         2.5  virginica
```

```
# Gather columns into rows (converting data from wide to long)
# Usage: gather(data, key, value, ...)
# data: A data frame.
# key, value: Names of key and value columns to create in output
# ...: Specification of columns to gather
# -Species is used to drop the Species column and gather the remaining
# columns, that is, Sepal.Length, Sepal.Width, Petal.Length, and Petal.Width
mini.iris1 <- gather(mini.iris, key=flower_trait, value=measurement, -Species)
mini.iris1
```

```
## Source: local data frame [12 x 3]
##
##   Species flower_trait measurement
##   (fctr)      (fctr)      (dbl)
## 1   setosa Sepal.Length      5.1
## 2 versicolor Sepal.Length      7.0
## 3  virginica Sepal.Length      6.3
## 4   setosa Sepal.Width       3.5
## 5 versicolor Sepal.Width       3.2
## 6  virginica Sepal.Width       3.3
## 7   setosa Petal.Length       1.4
## 8 versicolor Petal.Length       4.7
## 9  virginica Petal.Length       6.0
## 10  setosa Petal.Width        0.2
## 11 versicolor Petal.Width       1.4
## 12 virginica Petal.Width       2.5
```

## Spread rows into columns

```
# Spread rows into columns (converting data from long to wide)
mini.irisw <- spread(mini.irisl, flower_trait, measurement)
mini.irisw
```

```
## Source: local data frame [3 x 5]
##
##      Species Sepal.Length Sepal.Width Petal.Length Petal.Width
##      (fctr)      (dbl)      (dbl)      (dbl)      (dbl)
## 1    setosa        5.1        3.5        1.4        0.2
## 2 versicolor       7.0        3.2        4.7        1.4
## 3  virginica       6.3        3.3        6.0        2.5
```

## Order rows by values

```
# low to high
arrange(iris, Sepal.Length)
```

```
## Source: local data frame [150 x 5]
##
##      Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##      (dbl)      (dbl)      (dbl)      (dbl)  (fctr)
## 1          4.3        3.0        1.1        0.1  setosa
## 2          4.4        2.9        1.4        0.2  setosa
## 3          4.4        3.0        1.3        0.2  setosa
## 4          4.4        3.2        1.3        0.2  setosa
## 5          4.5        2.3        1.3        0.3  setosa
## 6          4.6        3.1        1.5        0.2  setosa
## 7          4.6        3.4        1.4        0.3  setosa
## 8          4.6        3.6        1.0        0.2  setosa
## 9          4.6        3.2        1.4        0.2  setosa
## 10         4.7        3.2        1.3        0.2  setosa
## ..          ...        ...        ...        ...    ...
```

```
# high to low
arrange(iris, desc(Sepal.Length))
```

```
## Source: local data frame [150 x 5]
##
##      Sepal.Length Sepal.Width Petal.Length Petal.Width  Species
##      (dbl)      (dbl)      (dbl)      (dbl)  (fctr)
## 1          7.9        3.8        6.4        2.0 virginica
## 2          7.7        3.8        6.7        2.2 virginica
## 3          7.7        2.6        6.9        2.3 virginica
## 4          7.7        2.8        6.7        2.0 virginica
## 5          7.7        3.0        6.1        2.3 virginica
## 6          7.6        3.0        6.6        2.1 virginica
## 7          7.4        2.8        6.1        1.9 virginica
```

```
## 8      7.3      2.9      6.3      1.8 virginica
## 9      7.2      3.6      6.1      2.5 virginica
## 10     7.2      3.2      6.0      1.8 virginica
## ..      ...      ...      ...      ...      ...
```

## Rename the columns of a data frame

```
rename(iris, sp=Species)
```

```
## Source: local data frame [150 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width   sp
##   (dbl)        (dbl)        (dbl)        (dbl) (fctr)
## 1      5.1      3.5      1.4      0.2 setosa
## 2      4.9      3.0      1.4      0.2 setosa
## 3      4.7      3.2      1.3      0.2 setosa
## 4      4.6      3.1      1.5      0.2 setosa
## 5      5.0      3.6      1.4      0.2 setosa
## 6      5.4      3.9      1.7      0.4 setosa
## 7      4.6      3.4      1.4      0.3 setosa
## 8      5.0      3.4      1.5      0.2 setosa
## 9      4.4      2.9      1.4      0.2 setosa
## 10     4.9      3.1      1.5      0.1 setosa
## ..      ...      ...      ...      ...      ...
```

## Manipulating data (filter, select, mutate, bind etc.)

dplyr provides a function for each basic verb of data manipulation:

- `filter()` (and `slice()`)
- `arrange()`
- `select()` (and `rename()`)
- `distinct()`
- `mutate()` (and `transmute()`)
- `summarise()`
- `sample_n()` and `sample_frac()`

The data is always the first argument of the verb functions.

### Subset observations (rows)

#### Filter

```
# Extract rows that meet logical criteria
filter(iris, Sepal.Length > 6)
```

```
## Source: local data frame [61 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width  Species
##           (dbl)         (dbl)         (dbl)         (dbl)    (fctr)
## 1           7.0           3.2           4.7           1.4 versicolor
## 2           6.4           3.2           4.5           1.5 versicolor
## 3           6.9           3.1           4.9           1.5 versicolor
## 4           6.5           2.8           4.6           1.5 versicolor
## 5           6.3           3.3           4.7           1.6 versicolor
## 6           6.6           2.9           4.6           1.3 versicolor
## 7           6.1           2.9           4.7           1.4 versicolor
## 8           6.7           3.1           4.4           1.4 versicolor
## 9           6.2           2.2           4.5           1.5 versicolor
## 10          6.1           2.8           4.0           1.3 versicolor
## ..          ...           ...           ...           ...      ...
```

```
iris[iris$Sepal.Length > 6, ] # Using base R
```

```
## Source: local data frame [61 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width  Species
##           (dbl)         (dbl)         (dbl)         (dbl)    (fctr)
## 1           7.0           3.2           4.7           1.4 versicolor
## 2           6.4           3.2           4.5           1.5 versicolor
## 3           6.9           3.1           4.9           1.5 versicolor
## 4           6.5           2.8           4.6           1.5 versicolor
## 5           6.3           3.3           4.7           1.6 versicolor
## 6           6.6           2.9           4.6           1.3 versicolor
## 7           6.1           2.9           4.7           1.4 versicolor
## 8           6.7           3.1           4.4           1.4 versicolor
## 9           6.2           2.2           4.5           1.5 versicolor
## 10          6.1           2.8           4.0           1.3 versicolor
## ..          ...           ...           ...           ...      ...
```

```
# Extract rows according to a factor
filter(iris, Species == "setosa")
```

```
## Source: local data frame [50 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           (dbl)         (dbl)         (dbl)         (dbl)    (fctr)
## 1           5.1           3.5           1.4           0.2 setosa
## 2           4.9           3.0           1.4           0.2 setosa
## 3           4.7           3.2           1.3           0.2 setosa
## 4           4.6           3.1           1.5           0.2 setosa
## 5           5.0           3.6           1.4           0.2 setosa
## 6           5.4           3.9           1.7           0.4 setosa
## 7           4.6           3.4           1.4           0.3 setosa
## 8           5.0           3.4           1.5           0.2 setosa
## 9           4.4           2.9           1.4           0.2 setosa
## 10          4.9           3.1           1.5           0.1 setosa
## ..          ...           ...           ...           ...      ...
```

```
subset(iris, Species == "setosa") # Using base R
```

```
## Source: local data frame [50 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl)    (fctr)
## 1         5.1         3.5         1.4         0.2    setosa
## 2         4.9         3.0         1.4         0.2    setosa
## 3         4.7         3.2         1.3         0.2    setosa
## 4         4.6         3.1         1.5         0.2    setosa
## 5         5.0         3.6         1.4         0.2    setosa
## 6         5.4         3.9         1.7         0.4    setosa
## 7         4.6         3.4         1.4         0.3    setosa
## 8         5.0         3.4         1.5         0.2    setosa
## 9         4.4         2.9         1.4         0.2    setosa
## 10        4.9         3.1         1.5         0.1    setosa
## ..          ...          ...          ...          ...
```

```
# Provide any number of filtering conditions,
# which are joined together with & or |
filter(iris, Species == "setosa" & Sepal.Length > 5)
```

```
## Source: local data frame [22 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl)    (fctr)
## 1         5.1         3.5         1.4         0.2    setosa
## 2         5.4         3.9         1.7         0.4    setosa
## 3         5.4         3.7         1.5         0.2    setosa
## 4         5.8         4.0         1.2         0.2    setosa
## 5         5.7         4.4         1.5         0.4    setosa
## 6         5.4         3.9         1.3         0.4    setosa
## 7         5.1         3.5         1.4         0.3    setosa
## 8         5.7         3.8         1.7         0.3    setosa
## 9         5.1         3.8         1.5         0.3    setosa
## 10        5.4         3.4         1.7         0.2    setosa
## ..          ...          ...          ...          ...
```

```
filter(iris, Species == "setosa" | Sepal.Length > median(Sepal.Length))
```

```
## Source: local data frame [120 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl)    (fctr)
## 1         5.1         3.5         1.4         0.2    setosa
## 2         4.9         3.0         1.4         0.2    setosa
## 3         4.7         3.2         1.3         0.2    setosa
## 4         4.6         3.1         1.5         0.2    setosa
## 5         5.0         3.6         1.4         0.2    setosa
## 6         5.4         3.9         1.7         0.4    setosa
## 7         4.6         3.4         1.4         0.3    setosa
## 8         5.0         3.4         1.5         0.2    setosa
```

```
## 9          4.4          2.9          1.4          0.2 setosa
## 10         4.9          3.1          1.5          0.1 setosa
## ..          ...          ...          ...          ...    ...
```

*# Using base R code*

```
iris[iris$Species == "setosa" | iris$Sepal.Length > median(iris$Sepal.Length), ]
```

```
## Source: local data frame [120 x 5]
```

```
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           (dbl)         (dbl)         (dbl)         (dbl)   (fctr)
## 1           5.1           3.5           1.4           0.2   setosa
## 2           4.9           3.0           1.4           0.2   setosa
## 3           4.7           3.2           1.3           0.2   setosa
## 4           4.6           3.1           1.5           0.2   setosa
## 5           5.0           3.6           1.4           0.2   setosa
## 6           5.4           3.9           1.7           0.4   setosa
## 7           4.6           3.4           1.4           0.3   setosa
## 8           5.0           3.4           1.5           0.2   setosa
## 9           4.4           2.9           1.4           0.2   setosa
## 10          4.9           3.1           1.5           0.1   setosa
## ..          ...          ...          ...          ...    ...
```

## Remove duplicate rows

```
# Remove duplicate rows
# Show duplicated rows: duplicated(iris)
distinct(iris)
```

```
## Source: local data frame [149 x 5]
```

```
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           (dbl)         (dbl)         (dbl)         (dbl)   (fctr)
## 1           5.1           3.5           1.4           0.2   setosa
## 2           4.9           3.0           1.4           0.2   setosa
## 3           4.7           3.2           1.3           0.2   setosa
## 4           4.6           3.1           1.5           0.2   setosa
## 5           5.0           3.6           1.4           0.2   setosa
## 6           5.4           3.9           1.7           0.4   setosa
## 7           4.6           3.4           1.4           0.3   setosa
## 8           5.0           3.4           1.5           0.2   setosa
## 9           4.4           2.9           1.4           0.2   setosa
## 10          4.9           3.1           1.5           0.1   setosa
## ..          ...          ...          ...          ...    ...
```

## Randomly select rows

```
# Randomly select fraction of rows
sample_frac(iris, 0.5, replace=TRUE)
```



```
## Source: local data frame [75 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl)        (fctr)
## 1         4.8         3.0         1.4         0.1      setosa
## 2         7.2         3.0         5.8         1.6  virginica
## 3         6.5         3.0         5.8         2.2  virginica
## 4         5.0         3.0         1.6         0.2      setosa
## 5         4.9         3.0         1.4         0.2      setosa
## 6         4.4         3.0         1.3         0.2      setosa
## 7         5.6         2.9         3.6         1.3  versicolor
## 8         5.5         4.2         1.4         0.2      setosa
## 9         5.5         2.4         3.8         1.1  versicolor
## 10        7.2         3.6         6.1         2.5  virginica
## ..          ...          ...          ...          ...
```

```
# Randomly select n rows
sample_n(iris, 10, replace=TRUE)
```

```
## Source: local data frame [10 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl)        (fctr)
## 1         6.4         2.8         5.6         2.1  virginica
## 2         4.8         3.1         1.6         0.2      setosa
## 3         5.8         2.7         5.1         1.9  virginica
## 4         4.4         3.0         1.3         0.2      setosa
## 5         5.7         2.5         5.0         2.0  virginica
## 6         7.6         3.0         6.6         2.1  virginica
## 7         5.4         3.9         1.3         0.4      setosa
## 8         6.2         3.4         5.4         2.3  virginica
## 9         6.0         3.4         4.5         1.6  versicolor
## 10        6.0         3.0         4.8         1.8  virginica
```

```
# Select rows by position
slice(iris, 10:15)
```

```
## Source: local data frame [6 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl)        (fctr)
## 1         4.9         3.1         1.5         0.1      setosa
## 2         5.4         3.7         1.5         0.2      setosa
## 3         4.8         3.4         1.6         0.2      setosa
## 4         4.8         3.0         1.4         0.1      setosa
## 5         4.3         3.0         1.1         0.1      setosa
## 6         5.8         4.0         1.2         0.2      setosa
```

## Subset variables (columns)

Select columns by complete or partial name, drop variables

```
# Select columns by name  
select(iris, Sepal.Width, Petal.Length, Species)
```

```
## Source: local data frame [150 x 3]  
##  
##   Sepal.Width Petal.Length Species  
##   (dbl)       (dbl)   (fctr)  
## 1      3.5      1.4   setosa  
## 2      3.0      1.4   setosa  
## 3      3.2      1.3   setosa  
## 4      3.1      1.5   setosa  
## 5      3.6      1.4   setosa  
## 6      3.9      1.7   setosa  
## 7      3.4      1.4   setosa  
## 8      3.4      1.5   setosa  
## 9      2.9      1.4   setosa  
## 10     3.1      1.5   setosa  
## ..      ...      ...     ...
```

```
# Select columns whose name starts with a character string  
select(iris, starts_with("Petal"))
```

```
## Source: local data frame [150 x 2]  
##  
##   Petal.Length Petal.Width  
##   (dbl)       (dbl)  
## 1      1.4      0.2  
## 2      1.4      0.2  
## 3      1.3      0.2  
## 4      1.5      0.2  
## 5      1.4      0.2  
## 6      1.7      0.4  
## 7      1.4      0.3  
## 8      1.5      0.2  
## 9      1.4      0.2  
## 10     1.5      0.1  
## ..      ...      ...
```

```
# Select columns whose name ends with a character string  
select(iris, ends_with("Length"))
```

```
## Source: local data frame [150 x 2]  
##  
##   Sepal.Length Petal.Length  
##   (dbl)       (dbl)  
## 1      5.1      1.4  
## 2      4.9      1.4
```

```
## 3      4.7      1.3
## 4      4.6      1.5
## 5      5.0      1.4
## 6      5.4      1.7
## 7      4.6      1.4
## 8      5.0      1.5
## 9      4.4      1.4
## 10     4.9      1.5
## ..      ...      ...
```

```
# Select columns whose names are in a group of names
select(iris, one_of("Species"))
```

```
## Source: local data frame [150 x 1]
##
##   Species
##   (fctr)
## 1  setosa
## 2  setosa
## 3  setosa
## 4  setosa
## 5  setosa
## 6  setosa
## 7  setosa
## 8  setosa
## 9  setosa
## 10 setosa
## ..      ...
```

```
# Drop variables
select(iris, -starts_with("Petal"))
```

```
## Source: local data frame [150 x 3]
##
##   Sepal.Length Sepal.Width Species
##   (dbl)        (dbl)    (fctr)
## 1      5.1      3.5    setosa
## 2      4.9      3.0    setosa
## 3      4.7      3.2    setosa
## 4      4.6      3.1    setosa
## 5      5.0      3.6    setosa
## 6      5.4      3.9    setosa
## 7      4.6      3.4    setosa
## 8      5.0      3.4    setosa
## 9      4.4      2.9    setosa
## 10     4.9      3.1    setosa
## ..      ...      ...      ...
```

```
select(iris, -contains("etal"))
```

```
## Source: local data frame [150 x 3]
##
```

```
##      Sepal.Length Sepal.Width Species
##              (dbl)         (dbl)  (fctr)
## 1           5.1           3.5   setosa
## 2           4.9           3.0   setosa
## 3           4.7           3.2   setosa
## 4           4.6           3.1   setosa
## 5           5.0           3.6   setosa
## 6           5.4           3.9   setosa
## 7           4.6           3.4   setosa
## 8           5.0           3.4   setosa
## 9           4.4           2.9   setosa
## 10          4.9           3.1   setosa
## ..          ...           ...     ...
```

```
select(iris, -Petal.Length, -Petal.Width)
```

```
## Source: local data frame [150 x 3]
##
##      Sepal.Length Sepal.Width Species
##              (dbl)         (dbl)  (fctr)
## 1           5.1           3.5   setosa
## 2           4.9           3.0   setosa
## 3           4.7           3.2   setosa
## 4           4.6           3.1   setosa
## 5           5.0           3.6   setosa
## 6           5.4           3.9   setosa
## 7           4.6           3.4   setosa
## 8           5.0           3.4   setosa
## 9           4.4           2.9   setosa
## 10          4.9           3.1   setosa
## ..          ...           ...     ...
```

```
select(iris, -(Sepal.Length:Petal.Length))
```

```
## Source: local data frame [150 x 2]
##
##      Petal.Width Species
##              (dbl)  (fctr)
## 1           0.2   setosa
## 2           0.2   setosa
## 3           0.2   setosa
## 4           0.2   setosa
## 5           0.2   setosa
## 6           0.4   setosa
## 7           0.3   setosa
## 8           0.2   setosa
## 9           0.2   setosa
## 10          0.1   setosa
## ..          ...     ...
```

```
# Using base R
# iris$Species <- NULL
```

```
# iris[["Species"]] <- NULL
# iris[, "Species"] <- NULL
# iris[[5]] <- NULL
# iris[,5] <- NULL
# iris <- subset(iris, select=-Species)
```

## Summarize data

```
summarise(group_by(iris, Species), mean(Sepal.Length))
```

```
## Source: local data frame [3 x 2]
##
##      Species mean(Sepal.Length)
##      (fctr)          (dbl)
## 1    setosa          5.006
## 2 versicolor        5.936
## 3 virginica         6.588
```

```
# Apply summary function to each column
summarise_each(iris, funs(mean))
```

```
## Source: local data frame [1 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl)    (dbl)
## 1    5.843333    3.057333    3.758        1.199333    NA
```

```
iris %>%
  select(-Species) %>%
  summarise_each(funs(mean))
```

```
## Source: local data frame [1 x 4]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width
##   (dbl)        (dbl)        (dbl)        (dbl)
## 1    5.843333    3.057333    3.758        1.199333
```

## Group data

```
# Group data into rows with the same value of Species
group_by(iris, Species)
```

```
## Source: local data frame [150 x 5]
## Groups: Species [3]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl)    (fctr)
```

```
## 1      5.1      3.5      1.4      0.2 setosa
## 2      4.9      3.0      1.4      0.2 setosa
## 3      4.7      3.2      1.3      0.2 setosa
## 4      4.6      3.1      1.5      0.2 setosa
## 5      5.0      3.6      1.4      0.2 setosa
## 6      5.4      3.9      1.7      0.4 setosa
## 7      4.6      3.4      1.4      0.3 setosa
## 8      5.0      3.4      1.5      0.2 setosa
## 9      4.4      2.9      1.4      0.2 setosa
## 10     4.9      3.1      1.5      0.1 setosa
## ..      ...      ...      ...      ...      ...
```

```
# Remove grouping information from data frame
ungroup(iris)
```

```
## Source: local data frame [150 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl)    (fctr)
## 1      5.1      3.5      1.4      0.2 setosa
## 2      4.9      3.0      1.4      0.2 setosa
## 3      4.7      3.2      1.3      0.2 setosa
## 4      4.6      3.1      1.5      0.2 setosa
## 5      5.0      3.6      1.4      0.2 setosa
## 6      5.4      3.9      1.7      0.4 setosa
## 7      4.6      3.4      1.4      0.3 setosa
## 8      5.0      3.4      1.5      0.2 setosa
## 9      4.4      2.9      1.4      0.2 setosa
## 10     4.9      3.1      1.5      0.1 setosa
## ..      ...      ...      ...      ...      ...
```

## Make new variables

```
# Compute and append one or more new columns
mutate(iris, sepal=Sepal.Length + Sepal.Width)
```

```
## Source: local data frame [150 x 6]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species sepal
##   (dbl)        (dbl)        (dbl)        (dbl)    (fctr) (dbl)
## 1      5.1      3.5      1.4      0.2 setosa  8.6
## 2      4.9      3.0      1.4      0.2 setosa  7.9
## 3      4.7      3.2      1.3      0.2 setosa  7.9
## 4      4.6      3.1      1.5      0.2 setosa  7.7
## 5      5.0      3.6      1.4      0.2 setosa  8.6
## 6      5.4      3.9      1.7      0.4 setosa  9.3
## 7      4.6      3.4      1.4      0.3 setosa  8.0
## 8      5.0      3.4      1.5      0.2 setosa  8.4
## 9      4.4      2.9      1.4      0.2 setosa  7.3
## 10     4.9      3.1      1.5      0.1 setosa  8.0
## ..      ...      ...      ...      ...      ...      ...
```

```
mutate(iris, logSepal.Length=log10(Sepal.Length))
```

```
## Source: local data frame [150 x 6]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##   (dbl)        (dbl)        (dbl)        (dbl) (fctr)
## 1         5.1         3.5         1.4         0.2  setosa
## 2         4.9         3.0         1.4         0.2  setosa
## 3         4.7         3.2         1.3         0.2  setosa
## 4         4.6         3.1         1.5         0.2  setosa
## 5         5.0         3.6         1.4         0.2  setosa
## 6         5.4         3.9         1.7         0.4  setosa
## 7         4.6         3.4         1.4         0.3  setosa
## 8         5.0         3.4         1.5         0.2  setosa
## 9         4.4         2.9         1.4         0.2  setosa
## 10        4.9         3.1         1.5         0.1  setosa
## ..          ...          ...          ...      ...
## Variables not shown: logSepal.Length (dbl)
```

```
# Using base R
# iris[["logSepal.Length"]] <- log10(iris$Sepal.Length)
# iris[, "logSepal.Length"] <- log10(iris$Sepal.Length)
# iris$logSepal.Length <- log10(iris$Sepal.Length)
```

```
# Drop existing variables
transmute(iris, logSepal.Length=log10(Sepal.Length))
```

```
## Source: local data frame [150 x 1]
##
##   logSepal.Length
##   (dbl)
## 1      0.7075702
## 2      0.6901961
## 3      0.6720979
## 4      0.6627578
## 5      0.6989700
## 6      0.7323938
## 7      0.6627578
## 8      0.6989700
## 9      0.6434527
## 10     0.6901961
## ..          ...
```

```
# Compute one or more new columns. Drop original columns
transmute(iris, sepal=Sepal.Length + Sepal.Width)
```

```
## Source: local data frame [150 x 1]
##
##   sepal
##   (dbl)
## 1    8.6
## 2    7.9
```

```
## 3    7.9
## 4    7.7
## 5    8.6
## 6    9.3
## 7    8.0
## 8    8.4
## 9    7.3
## 10   8.0
## ..   ...
```

## Combine data sets

```
a <- data_frame(x1=c("A", "B", "C"), x2=c(1, 2, 3))
b <- data_frame(x1=c("A", "B", "D"), x3=c("T", "F", "T"))

# Join matching rows from b to a
left_join(a, b, by="x1")
```

```
## Source: local data frame [3 x 3]
##
##      x1      x2      x3
##   (chr) (dbl) (chr)
## 1     A      1      T
## 2     B      2      F
## 3     C      3     NA
```

```
# Join matching rows from a to b
right_join(a, b, by="x1")
```

```
## Source: local data frame [3 x 3]
##
##      x1      x2      x3
##   (chr) (dbl) (chr)
## 1     A      1      T
## 2     B      2      F
## 3     D     NA      T
```

```
# Join data. Retain only rows in both sets
inner_join(a, b, by="x1")
```

```
## Source: local data frame [2 x 3]
##
##      x1      x2      x3
##   (chr) (dbl) (chr)
## 1     A      1      T
## 2     B      2      F
```

```
# Join data. Retain all values, all rows
full_join(a, b, by="x1")
```



```
## Source: local data frame [4 x 3]
##
##      x1      x2      x3
##   (chr) (dbl) (chr)
## 1     A      1      T
## 2     B      2      F
## 3     C      3     NA
## 4     D     NA      T
```

```
# All rows in a that have a match in b
semi_join(a, b, by="x1")
```

```
## Source: local data frame [2 x 2]
##
##      x1      x2
##   (chr) (dbl)
## 1     A      1
## 2     B      2
```

```
# All rows in a that do not have a match in b
anti_join(a, b, by="x1")
```

```
## Source: local data frame [1 x 2]
##
##      x1      x2
##   (chr) (dbl)
## 1     C      3
```

## Set operations

These expect the x and y inputs to have the same variables, and treat the observations like sets.

```
y <- data_frame(x1=c("A", "B", "C"), x2=c(1, 2, 3))
z <- data_frame(x1=c("B", "C", "D"), x2=c(2, 3, 4))
str(y)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':   3 obs. of  2 variables:
## $ x1: chr  "A" "B" "C"
## $ x2: num  1 2 3
```

```
glimpse(y)
```

```
## Observations: 3
## Variables: 2
## $ x1 (chr) "A", "B", "C"
## $ x2 (dbl) 1, 2, 3
```

```
# Rows that appear in both y and z
intersect(y, z)
```

```
## Source: local data frame [2 x 2]
##
##      x1    x2
##   (chr) (dbl)
## 1     B     2
## 2     C     3
```

```
# Rows that appear in either or both y and z
union(y, z)
```

```
## Source: local data frame [4 x 2]
##
##      x1    x2
##   (chr) (dbl)
## 1     D     4
## 2     C     3
## 3     B     2
## 4     A     1
```

```
# Rows that appear in y but not z
setdiff(y, z)
```

```
## Source: local data frame [1 x 2]
##
##      x1    x2
##   (chr) (dbl)
## 1     A     1
```

## Binding

```
# Append z to y as new rows
bind_rows(y, z)
```

```
## Source: local data frame [6 x 2]
##
##      x1    x2
##   (chr) (dbl)
## 1     A     1
## 2     B     2
## 3     C     3
## 4     B     2
## 5     C     3
## 6     D     4
```

```
# Append z to y as new columns. Caution: matches rows by position
bind_cols(y, z)
```

```
## Source: local data frame [3 x 4]
##
##      x1    x2    x1    x2
```

```
##   (chr) (dbl) (chr) (dbl)
## 1    A     1    B     2
## 2    B     2    C     3
## 3    C     3    D     4
```

For further information on `dplyr`:

<https://cran.r-project.org/web/packages/dplyr/index.html>

<http://blog.rstudio.org/2014/01/17/introducing-dplyr/>

<http://www.dataschool.io/dplyr-tutorial-for-faster-data-manipulation-in-r/>

## Using `magrittr` to create pipelines

According to its documentation (<https://github.com/smbache/magrittr>), the `magrittr` package offers a set of operators which promote semantics that will improve your code by:

- structuring sequences of data operations left-to-right (as opposed to from the inside and out),
- avoiding nested function calls,
- minimizing the need for local variables and function definitions, and
- making it easy to add steps anywhere in the sequence of operations.

A simple example:

```
#install.packages("magrittr")
library(magrittr)
iris %>% head(4)
```

```
## Source: local data frame [4 x 5]
##
##   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
##           (dbl)         (dbl)         (dbl)         (dbl)   (fctr)
## 1           5.1           3.5           1.4           0.2   setosa
## 2           4.9           3.0           1.4           0.2   setosa
## 3           4.7           3.2           1.3           0.2   setosa
## 4           4.6           3.1           1.5           0.2   setosa
```

The pipe operator `%>%` passes the object on left hand side as first argument of function on righthand side. `dplyr` imports the `%>%` operator from `magrittr`.

```
iris %>%
  filter(Species == "setosa") %>%
  select(Sepal.Length, Sepal.Width) %>%
  head(10)
```

```
## Source: local data frame [10 x 2]
##
##   Sepal.Length Sepal.Width
##           (dbl)         (dbl)
## 1           5.1           3.5
## 2           4.9           3.0
## 3           4.7           3.2
```

```
## 4          4.6          3.1
## 5          5.0          3.6
## 6          5.4          3.9
## 7          4.6          3.4
## 8          5.0          3.4
## 9          4.4          2.9
## 10         4.9          3.1
```

```
iris %>%
  group_by(Species) %>%
  summarise(avg=mean(Sepal.Width)) %>%
  arrange(avg)
```

```
## Source: local data frame [3 x 2]
##
##      Species    avg
##      (fctr) (dbl)
## 1 versicolor 2.770
## 2 virginica  2.974
## 3 setosa     3.428
```

There are also functions that do not have a data argument, for which it is useful to expose the variables in the data. This is done with the `%%` operator:

```
iris %>%
  filter(Sepal.Length > mean(Sepal.Length)) %%%
  cor(Sepal.Length, Sepal.Width)
```

```
## [1] 0.3361992
```

Which country experienced the sharpest 5-year drop in life expectancy?

```
#install.packages("gapminder")
library(gapminder)
```

```
## Warning: package 'gapminder' was built under R version 3.1.3
```

```
gtbl <- tbl_df(gapminder)
glimpse(gtbl)
```

```
## Observations: 1,704
## Variables: 6
## $ country   (fctr) Afghanistan, Afghanistan, Afghanistan, Afghanistan,...
## $ continent (fctr) Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asia, Asi...
## $ year      (dbl) 1952, 1957, 1962, 1967, 1972, 1977, 1982, 1987, 1992...
## $ lifeExp   (dbl) 28.801, 30.332, 31.997, 34.020, 36.088, 38.438, 39.8...
## $ pop       (dbl) 8425333, 9240934, 10267083, 11537966, 13079460, 1488...
## $ gdpPercap (dbl) 779.4453, 820.8530, 853.1007, 836.1971, 739.9811, 78...
```

```
#worstle <-
gtbl %>%
  group_by(continent, country) %>%
  select(country, year, continent, lifeExp) %>%
  # lag(): copy with values lagged by 1
  mutate(le.delta=lifeExp - lag(lifeExp)) %>%
  summarize(worstle.delta=min(le.delta, na.rm=TRUE)) %>%
  filter(min_rank(worstle.delta) < 2) %>%
  arrange(worstle.delta)
```

```
## Source: local data frame [5 x 3]
## Groups: continent [5]
##
##   continent      country worstle.delta
##   (fctr)        (fctr)      (dbl)
## 1   Africa      Rwanda      -20.421
## 2 Americas El Salvador    -1.511
## 3    Asia      Cambodia    -9.097
## 4   Europe  Montenegro    -1.464
## 5  Oceania  Australia      0.170
```

```
#worstle
```

What is the correlation between life expectancy and year within each country?

```
# Fit a linear regression within country
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.1.3
```

```
str(gapminder)
```

```
## 'data.frame': 1704 obs. of 6 variables:
## $ country : Factor w/ 142 levels "Afghanistan",...: 1 1 1 1 1 1 1 1 1 ...
## $ continent: Factor w/ 5 levels "Africa","Americas",...: 3 3 3 3 3 3 3 3 3 ...
## $ year : num 1952 1957 1962 1967 1972 ...
## $ lifeExp : num 28.8 30.3 32 34 36.1 ...
## $ pop : num 8425333 9240934 10267083 11537966 13079460 ...
## $ gdpPercap: num 779 821 853 836 740 ...
```

```
ggplot(gapminder, aes(x=year, y=lifeExp, colour=continent)) +
  geom_jitter() +
  geom_smooth(lwd=1.5, method="lm")
```



```
# Calculate overall correlation between year and life expectancy
(ov.cor <- gapminder %$%
  cor(year, lifeExp))
```

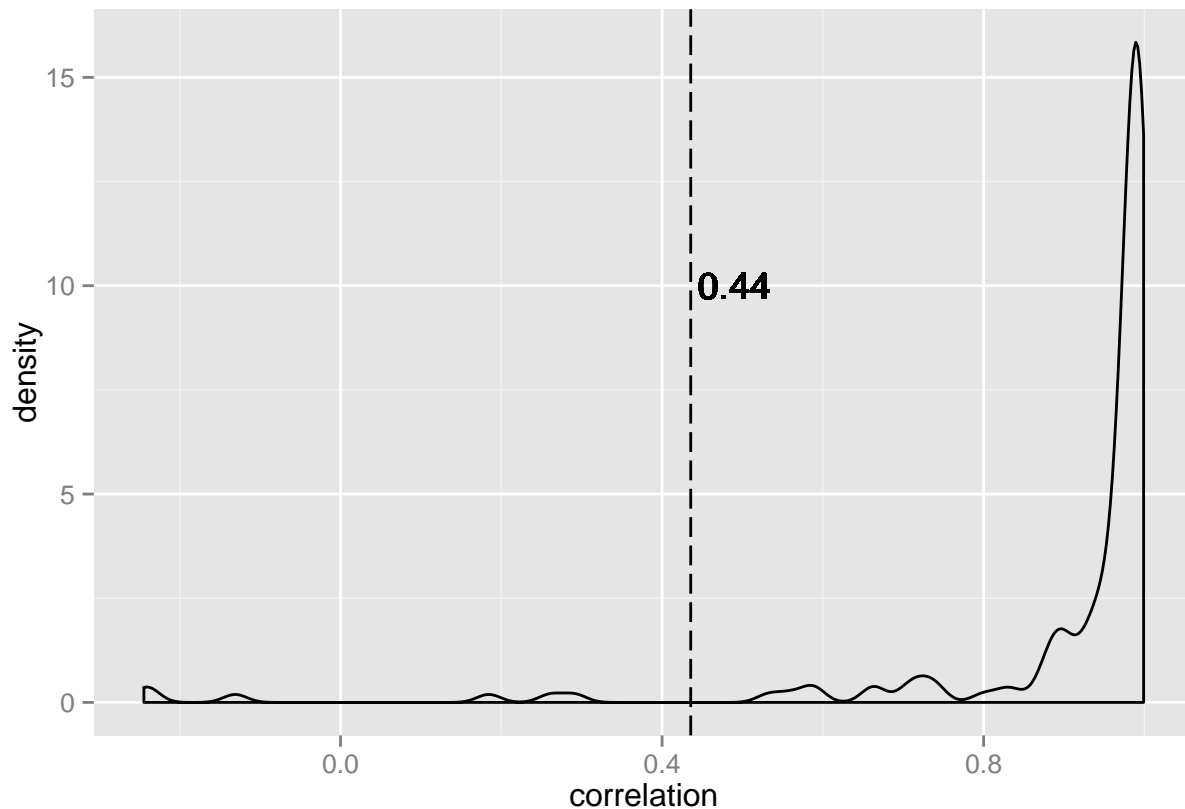
```
## [1] 0.4356112
```

```
#> [1] 0.4356112
# Calculate correlation within each country
(gcor <- gapminder %>%
  group_by(country) %>%
  summarize(correlation=cor(year, lifeExp)))
```

```
## Source: local data frame [142 x 2]
##
##   country correlation
##   (fctr)      (dbl)
## 1 Afghanistan  0.9735051
## 2 Albania      0.9542420
## 3 Algeria      0.9925307
## 4 Angola       0.9422392
## 5 Argentina    0.9977816
## 6 Australia    0.9897716
## 7 Austria      0.9960592
## 8 Bahrain      0.9832293
## 9 Bangladesh   0.9946662
```

```
## 10      Belgium    0.9972665
## ..      ...      ...
```

```
# The correlation between life expectancy and year is
# much higher within countries
ggplot(gcor, aes(x=correlation)) +
  geom_density() +
  geom_vline(xintercept=ov.cor, linetype="longdash") +
  geom_text(data=NULL, x=ov.cor, y=10, label=round(ov.cor, 2),
            hjust=-0.1)
```



## Pattern matching and replacement

Some content included here taken from <https://github.com/STAT545-UBC>

### Position of pattern within the string

- `^`: start of the string.
- `$`: end of the string.
- `\b`: empty string at either edge of a word. Don't confuse it with `^ $` which marks the edge of a string.
- `\B`: empty string provided it is not at an edge of a word.

## Operators

- `.`: matches any single character, as shown in the first example.
- `[...]`: a character list, matches any one of the characters inside the square brackets. We can also use `-` inside the brackets to specify a range of characters.
- `[^...]`: an inverted character list, similar to `[...]`, but matches any characters except those inside the square brackets.
- `\`: suppress the special meaning of metacharacters in regular expression, i.e. `$ * + . ? [ ] ^ { }` | `( ) \`, similar to its usage in escape sequences. Since `\` itself needs to be escaped in R, we need to escape these metacharacters with double backslash like `\\$`.
- `|`: an “or” operator, matches patterns on either side of the `|`.
- `(...)`: grouping in regular expressions. This allows you to retrieve the bits that matched various parts of your regular expression so you can alter them or use them for building up a new string. Each group can then be referred using `\\N`, with N being the No. of `(...)` used. This is called backreference.

```
?grep
bee.spp <- c("Apis cerana", "Apis koschevnikovi", "Apis mellifera",
  "Apis nigrocincta", "Bombus atratus", "Bombus dahlbomii",
  "Bombus fervidus", "Bombus lapidarius", "Bombus ruderatus",
  "Bombus rupestris")

i <- grep("Bombus", bee.spp)
cat("'Bombus' appears", length(bee.spp[i]), "times")
```

```
## 'Bombus' appears 6 times
```

```
i
```

```
## [1] 5 6 7 8 9 10
```

```
bee.spp[i]
```

```
## [1] "Bombus atratus"      "Bombus dahlbomii"   "Bombus fervidus"
## [4] "Bombus lapidarius"   "Bombus ruderatus"   "Bombus rupestris"
```