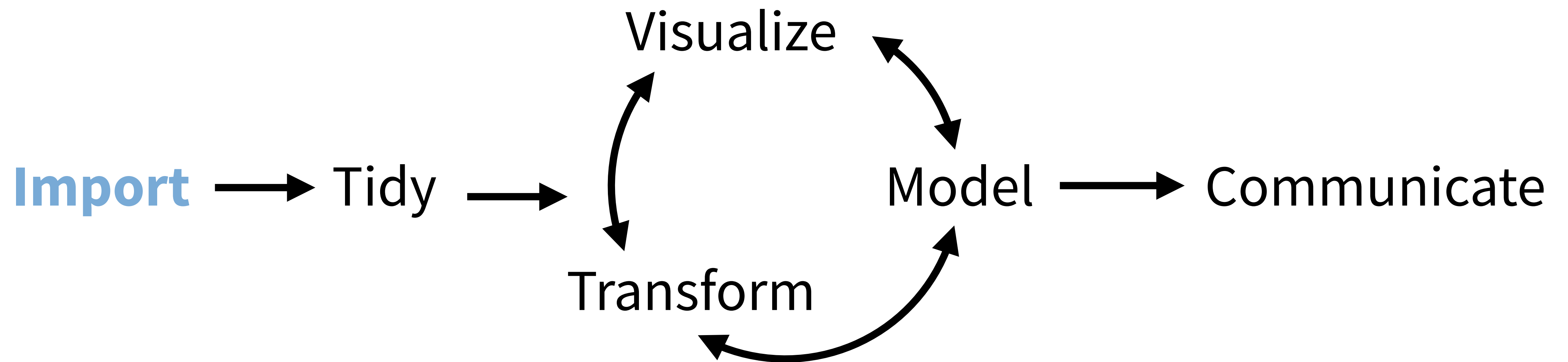


Import Data with



Open **04-Import-Data.Rmd**

(Applied) Data Science



Program



Importing Data



readr



Simple, consistent functions for working with strings.

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



Compared to `read.table` and its derivatives,
`readr` functions are:

1. ~ 10 times faster
2. Return tibbles
3. Have more intuitive defaults. No row names, no strings as factors.



readr functions

function	reads
<code>read_csv()</code>	Comma separated values
<code>read_csv2()</code>	Semi-colon separated values
<code>read_delim()</code>	General delimited files
<code>read_fwf()</code>	Fixed width files
<code>read_log()</code>	Apache log files
<code>read_table()</code>	Space separated
<code>read_tsv()</code>	Tab delimited values

readr functions

function	reads
read_csv()	Comma separated values
read_csv2()	Semi-colon separated values
read_delim()	General delimited files
read_fwf()	Fixed width files
read_log()	Apache log files
read_table()	Space separated
read_tsv()	Tab delimited values

nimbus.csv

```
date,longitude,latitude,ozone
1985-10-01T00:00:00Z,-179.375,-87.5,.
1985-10-01T00:00:00Z,-178.125,-87.5,.
1985-10-01T00:00:00Z,-176.875,-87.5,.
1985-10-01T00:00:00Z,-175.625,-87.5,.
1985-10-01T00:00:00Z,-174.375,-87.5,.
1985-10-01T00:00:00Z,-173.125,-87.5,.
1985-10-01T00:00:00Z,-171.875,-87.5,.
1985-10-01T00:00:00Z,-170.625,-87.5,.
1985-10-01T00:00:00Z,-169.375,-87.5..
```



nimbus.csv

```
date,longitude,latitude,ozone
1985-10-01T00:00:00Z,-179.375,-87.5,.
1985-10-01T00:00:00Z,-178.125,-87.5,.
1985-10-01T00:00:00Z,-176.875,-87.5,.
1985-10-01T00:00:00Z,-175.625,-87.5,.
1985-10-01T00:00:00Z,-174.375,-87.5,.
1985-10-01T00:00:00Z,-173.125,-87.5,.
1985-10-01T00:00:00Z,-171.875,-87.5,.
1985-10-01T00:00:00Z,-170.625,-87.5,.
1985-10-01T00:00:00Z,-169.375,-87.5,.
```





read_csv()

readr functions share a common syntax

```
df <- read_csv("path/to/file.csv", ...)
```

**object to save
output into**

**path from working
directory to file**



Your Turn 1

Find **nimbus.csv** on your server or computer. Then read it into an object. Then view the results.

02:00

Your Turn 1

Find **nimbus.csv** on your server or computer. Then read it into an object. Then view the results.

```
nimbus <- read_csv("nimbus.csv")
```

```
nimbus
```


tibbles

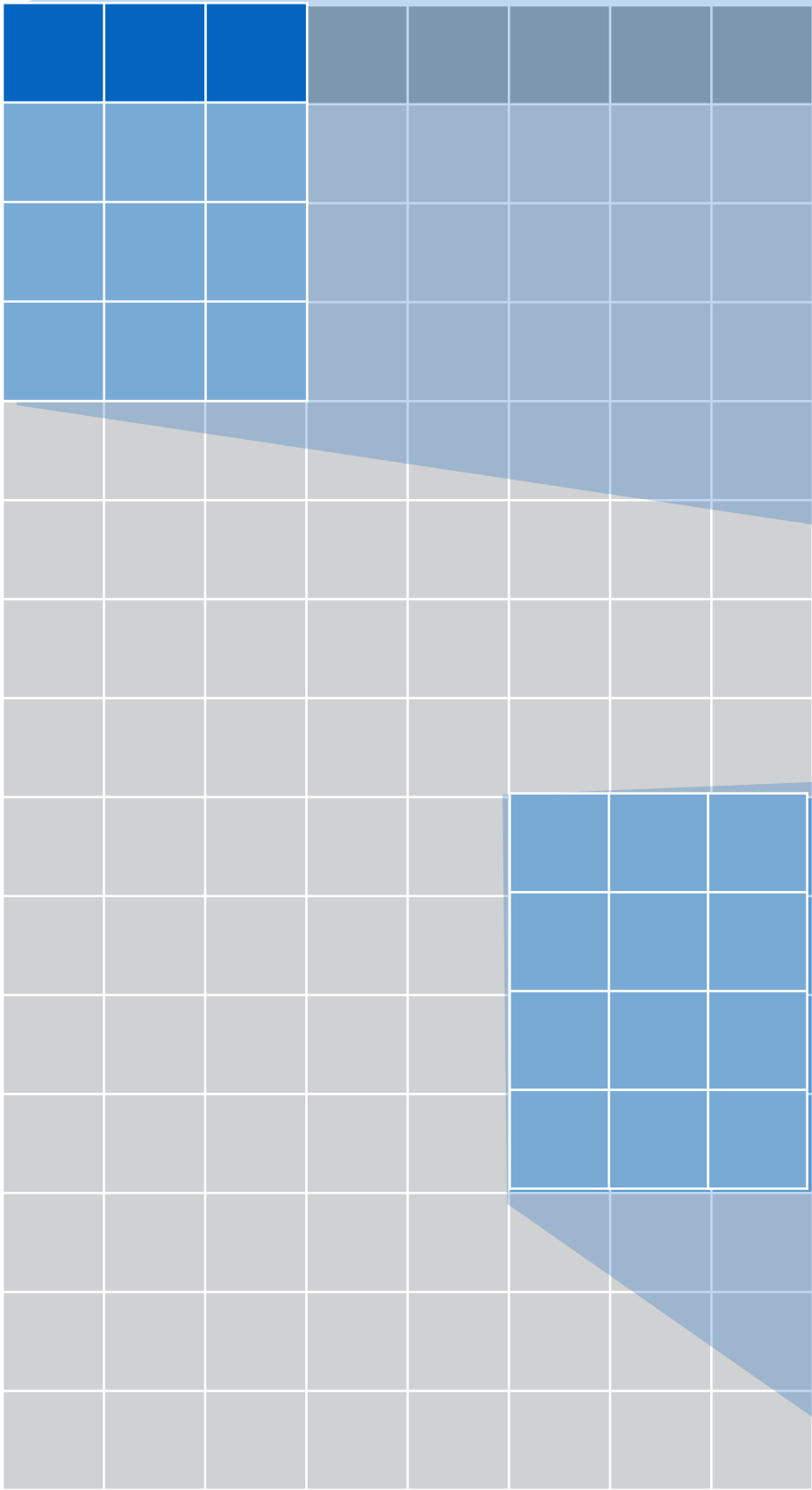


read.csv() vs. read_csv()

```
Console ~/Dropbox (RStudio)/RStudio/training/U-Master-the-tidyverse/0-course-developm
217 1985-10-01 -144.375 -86.5 .
218 1985-10-01 -143.125 -86.5 .
219 1985-10-01 -141.875 -86.5 .
220 1985-10-01 -140.625 -86.5 .
221 1985-10-01 -139.375 -86.5 .
222 1985-10-01 -138.125 -86.5 .
223 1985-10-01 -136.875 -86.5 .
224 1985-10-01 -135.625 -86.5 .
225 1985-10-01 -134.375 -86.5 .
226 1985-10-01 -133.125 -86.5 .
227 1985-10-01 -131.875 -86.5 .
228 1985-10-01 -130.625 -86.5 .
229 1985-10-01 -129.375 -86.5 .
230 1985-10-01 -128.125 -86.5 .
231 1985-10-01 -126.875 -86.5 .
232 1985-10-01 -125.625 -86.5 .
233 1985-10-01 -124.375 -86.5 .
234 1985-10-01 -123.125 -86.5 .
235 1985-10-01 -121.875 -86.5 .
236 1985-10-01 -120.625 -86.5 .
237 1985-10-01 -119.375 -86.5 .
238 1985-10-01 -118.125 -86.5 .
239 1985-10-01 -116.875 -86.5 .
240 1985-10-01 -115.625 -86.5 .
241 1985-10-01 -114.375 -86.5 .
242 1985-10-01 -113.125 -86.5 .
243 1985-10-01 -111.875 -86.5 .
244 1985-10-01 -110.625 -86.5 .
245 1985-10-01 -109.375 -86.5 .
246 1985-10-01 -108.125 -86.5 .
247 1985-10-01 -106.875 -86.5 .
248 1985-10-01 -105.625 -86.5 .
249 1985-10-01 -104.375 -86.5 .
250 1985-10-01 -103.125 -86.5 .
[ reached getOption("max.print") -- omitted 24974 rows ]
> |
```

```
Console ~/Dropbox (RStudio)/RStudio/training/U-Master-the-tidyverse/0-course-developm
> nimbus
# A tibble: 25,224 x 4
   date longitude latitude ozone
   <dtm>      <dbl>    <dbl> <chr>
1 1985-10-01 -179.375  -87.5   .
2 1985-10-01 -178.125  -87.5   .
3 1985-10-01 -176.875  -87.5   .
4 1985-10-01 -175.625  -87.5   .
5 1985-10-01 -174.375  -87.5   .
6 1985-10-01 -173.125  -87.5   .
7 1985-10-01 -171.875  -87.5   .
8 1985-10-01 -170.625  -87.5   .
9 1985-10-01 -169.375  -87.5   .
10 1985-10-01 -168.125 -87.5   .
# ... with 25,214 more rows
> |
```





A large table to display

```
# A tibble: 234 × 6
  manufacturer      model displ
  <chr>            <chr> <dbl>
1      audi         a4    1.8
2      audi         a4    1.8
3      audi         a4    2.0
4      audi         a4    2.0
5      audi         a4    2.8
6      audi         a4    2.8
7      audi         a4    3.1
8      audi a4 quattro  1.8
9      audi a4 quattro  1.8
10     audi a4 quattro  2.0
# ... with 224 more rows, and 3
#   more variables: year <int>,
#   cyl <int>, trans <chr>
```

tibble display

```
156 1999      6  auto(l4)
157 1999      6  auto(l4)
158 2008      6  auto(l4)
159 2008      8  auto(s4)
160 1999      4 manual(m5)
161 1999      4  auto(l4)
162 2008      4 manual(m5)
163 2008      4 manual(m5)
164 2008      4  auto(l4)
165 2008      4  auto(l4)
166 1999      4  auto(l4)
[ reached getOption("max.print") --
omitted 68 rows ]
```

data frame display



tibbles

A type of data frame common throughout tidyverse packages.
Tibbles enhance data frames in three ways:

- 1. Subsetting** - `[` always returns a new tibble, `[[` and `$` always return a new vector
- 2. No partial matching** - You must use full column names when subsetting
- 3. Display** - When you print a tibble, R provides a concise view of the data that fits on one screen





tibble

A package with several helper functions for tibbles:

- **as_tibble()** - convert a data frame to a tibble
- **as.data.frame()** - convert a tibble to a data frame
- **tribble()** - make a tibble (transversed)

```
tribble(  
  ~x, ~y,  
  1, "a",  
  2, "b",  
  3, "c")
```

x	y
1	a
2	b
3	c



Tibbles - an enhanced data frame

The **tibble** package provides a new S3 class for storing tabular data, the tibble. Tibbles inherit the data frame class, but improve two behaviors:

- **Display** - When you print a tibble, R provides a concise view of the data that fits on one screen.
- **Subsetting** - `[` always returns a new tibble, `[[` and `$` always return a vector.
- **No partial matching** - You must use full column names when subsetting

```
# A tibble: 234 x 6
  manufacturer    model displ
    <chr>         <chr>   <dbl>
1      audi      a4         1.8
2      audi      a4         1.8
3      audi      a4         2.0
4      audi      a4         2.0
5      audi      a4         2.8
6      audi      a4         2.8
7      audi      a4         3.1
8      audi      a4 quattro 1.8
9      audi      a4 quattro 1.8
10     audi      a4 quattro 2.0
# ... with 224 more rows, and 3
# more variables: year <int>,
# cyl <int>, trans <chr>
```

tibble display

```
156 1999    6 auto(l4)
157 1999    6 auto(l4)
158 2008    6 auto(l4)
159 2008    8 auto(s4)
160 1999    4 manual(m5)
161 1999    4 auto(l4)
162 2008    4 manual(m5)
163 2008    4 manual(m5)
164 2008    4 auto(l4)
165 2008    4 auto(l4)
166 1999    4 auto(l4)
# reached getOption("max.print")
# omitted 68 rows
```

data frame display

A large table to display

- Control the default appearance with options:


```
options(tibble.print_max = n,
        tibble.print_min = m, tibble.width = Inf)
```
- View entire data set with **View(x, title)** or **glimpse(x, width = NULL, ...)**
- Revert to data frame with **as.data.frame()** (required for some older packages)

Construct a tibble in two ways

tibble(...)

Construct by columns.

```
tibble(x = 1:3,
       y = c("a", "b", "c"))
```

Both make this tibble

tribble(...)

Construct by rows.

```
tribble(
  ~x, ~y,
  1, "a",
  2, "b",
  3, "c")
```

```
A tibble: 3 x 2
      x     y
  <int> <dbl>
1     1     a
2     2     b
3     3     c
```

as_tibble(x, ...) Convert data frame to tibble.

enframe(x, name = "name", value = "value")
Converts named vector to a tibble with a names column and a values column.

is_tibble(x) Test whether x is a tibble.

tibbles

Tibbles - an enhanced data frame

The tibble package provides a new S3 class for storing tabular data, the tibble. Tibbles inherit the data frame class, but improve two behaviors:

- **Display** - When you print a tibble, R provides a concise view of the data that fits on one screen.
- **Subsetting** - `[` always returns a new tibble, `[[` and `$` always return a vector.
- **No partial matching** - You must use full column names when subsetting

```
# A tibble: 234 x 6
  manufacturer    model displ
    <chr>         <chr>   <dbl>
1      audi      a4         1.8
2      audi      a4         1.8
3      audi      a4         2.0
4      audi      a4         2.0
5      audi      a4         2.8
6      audi      a4         2.8
7      audi      a4         3.1
8      audi      a4 quattro 1.8
9      audi      a4 quattro 1.8
10     audi      a4 quattro 2.0
# ... with 224 more rows, and 3
# more variables: year <int>,
# cyl <int>, trans <chr>
```

tibble display

```
156 1999    6 auto(l4)
157 1999    6 auto(l4)
158 2008    6 auto(l4)
159 2008    8 auto(s4)
160 1999    4 manual(m5)
161 1999    4 auto(l4)
162 2008    4 manual(m5)
163 2008    4 manual(m5)
164 2008    4 auto(l4)
165 2008    4 auto(l4)
166 1999    4 auto(l4)
# reached getOption("max.print")
# omitted 68 rows
```

data frame display

A large table to display

- Control the default appearance with options:


```
options(tibble.print_max = n,
        tibble.print_min = m, tibble.width = Inf)
```
- View entire data set with **View(x, title)** or **glimpse(x, width = NULL, ...)**
- Revert to data frame with **as.data.frame()** (required for some older packages)

Construct a tibble in two ways

tibble(...)
Construct by columns.

```
tibble(x = 1:3,
       y = c("a", "b", "c"))
```

tribble(...)
Construct by rows.

```
tribble(
  ~x, ~y,
  1, "a",
  2, "b",
  3, "c")
```

as_tibble(x, ...) Convert data frame to tibble.

enframe(x, name = "name", value = "value")
Converts named vector to a tibble with a names column and a values column.

is_tibble(x) Test whether x is a tibble.

Tidy Data with tidyr

Tidy data is a way to organize tabular data. It provides a consistent data structure across packages. A table is tidy if:

- Each variable is in its own column
- Each observation, or case, is in its own row
- Makes variables easy to access as vectors
- Preserves cases during vectorized operations

Reshape Data - change the layout of values in a table

Use **gather()** and **spread()** to reorganize the values of a table into a new layout. Each uses the idea of a key column: value column pair.

gather(data, key, value, ...) `na.rm = FALSE`, `convert = FALSE`, `factor_key = FALSE`

Gather moves column names into a key column, gathering the column values into a single value column.

spread(data, key, value, fill = NA, convert = FALSE, drop = TRUE, set = NULL)

Spread moves the unique values of a key column into the column names, spreading the values of a value column across the new columns that result.

Handle Missing Values

drop_na(data, ...)
Drop rows containing NA's in ... columns.

fill(data, ...) `direction = c("down", "up")`
Fill in NA's in ... columns with most recent non-NA values.

replace_na(data, replace = list(), ...)
Replace NA's by column.

Expand Tables - quickly create tables with combinations of values

complete(data, ...) `fill = list()`
Add to the data missing combinations of the values of the variables listed in ...

expand_grid(x, y, ...)
Create new tibble with all possible combinations of the values of the variables listed in ...

Split and Combine Cells

Use these functions to split or combine cells into individual, atomic values.

separate(data, col_name, sep = "[^a-zA-Z]", remove = TRUE, convert = FALSE, extra = "warn", fill = "warn")

Separate each cell in a column to make several columns.

separate_rows(data, ...) `sep = "[^a-zA-Z]"`, `convert = FALSE`

Separate each cell in a column to make several rows. Also **separate_rows()**.

unite(data, col, ...) `sep = ""`, `remove = TRUE`

Collapse cells across several columns to make a single column.

Parsing



. = NA

nimbus

date <S3: POSIXct>	longitude <dbl>	latitude <dbl>	ozone <chr>
1985-10-01	-179.375	-87.5	.
1985-10-01	-178.125	-87.5	.
1985-10-01	-176.875	-87.5	.
1985-10-01	-175.625	-87.5	.
1985-10-01	-174.375	-87.5	.
1985-10-01	-173.125	-87.5	.
1985-10-01	-171.875	-87.5	.
1985-10-01	-170.625	-87.5	.
1985-10-01	-169.375	-87.5	.



read_csv()

readr functions share a common syntax

```
nimbus <- read_csv("nimbus.csv", na = ".")
```

**object to save
output into**

**path from working
directory to file**

**Value(s) to
convert to NA**



```
nimbus <- read_csv("nimbus.csv", na = ".")
```

date <S3: POSIXct>	longitude <dbl>	latitude <dbl>	ozone <chr>
1985-10-01	-179.375	-87.5	NA
1985-10-01	-178.125	-87.5	NA
1985-10-01	-176.875	-87.5	NA
1985-10-01	-175.625	-87.5	NA
1985-10-01	-174.375	-87.5	NA
1985-10-01	-173.125	-87.5	NA
1985-10-01	-171.875	-87.5	NA
1985-10-01	-170.625	-87.5	NA
1985-10-01	-169.375	-87.5	NA
1985-10-01	-168.125	-87.5	NA

**<chr> stands for
character string
(not a number)**



read_csv()

readr functions share a common syntax

```
nimbus <- read_csv("nimbus.csv", na = "."),  
  col_types = list(ozone = col_double()))
```

**Manually
specify column
types.**

list

**column
name**

**Column type
function**



type function	data type
col_character()	character
col_date()	Date
col_datetime()	POSIXct (date-time)
col_double()	double (numeric)
col_factor()	factor
col_guess()	let readr guess (default)
col_integer()	integer
col_logical()	logical
col_number()	numbers mixed with non-number characters
col_numeric()	double or integer
col_skip()	do not read
col_time()	time

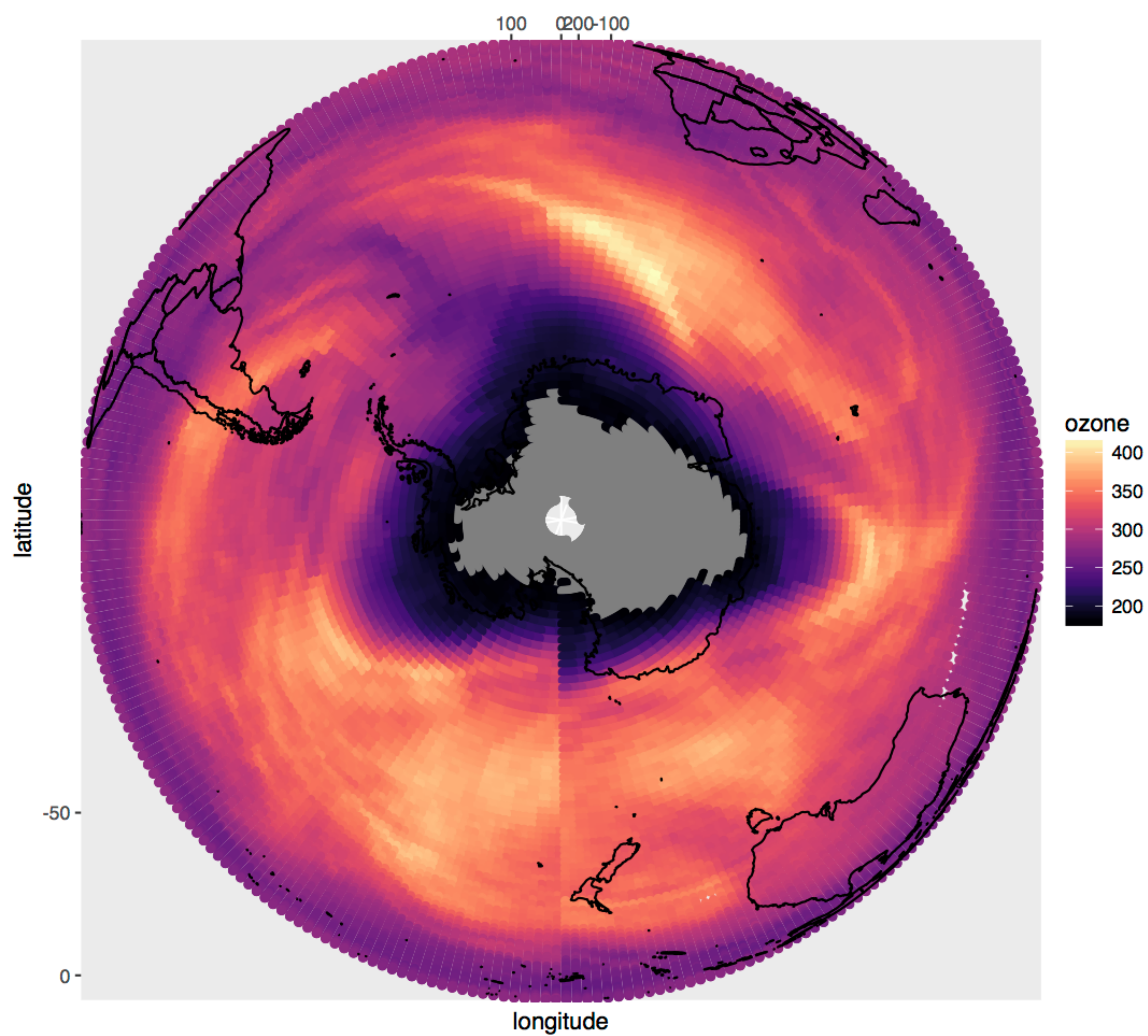


type function	data type
col_character()	character
col_date()	Date
col_datetime()	POSIXct (date-time)
col_double()	double (numeric)
col_factor()	factor
col_guess()	let readr guess (default)
col_integer()	integer
col_logical()	logical
col_number()	numbers mixed with non-number characters
col_numeric()	double or integer
col_skip()	do not read
col_time()	time



```
nimbus <- read_csv("nimbus.csv", na = ".",  
  col_types = list(ozone = col_double()))  
  
library(viridis)  
world <- map_data(map = "world")  
nimbus %>%  
  ggplot() +  
    geom_point(aes(longitude, latitude, color = ozone)) +  
    geom_path(aes(long, lat, group = group), data = world) +  
    coord_map("ortho", orientation=c(-90, 0, 0)) +  
    scale_color_viridis(option = "A")
```





Writing



readr functions

function	writes
<code>write_csv()</code>	Comma separated values
<code>write_excel_csv()</code>	CSV intended for opening in Excel
<code>write_delim()</code>	General delimited files
<code>write_file()</code>	Single string, written as is
<code>write_lines()</code>	Vector of strings, one element per line
<code>write_tsv()</code>	Tab delimited values



write_csv()

Saves data set as a csv on your computer.

```
write_csv(nimbus, file = "nimbus2.csv")
```

Table to save

**file
path to save at**



Other types of data

package	accesses
haven	SPSS, Stata, and SAS files
readxl	excel files (.xls, .xlsx)
jsonlite	json
xml2	xml
httr	web API's
rvest	web pages (web scraping)
DBI	databases
sparklyr	data loaded into spark

Import Data with

