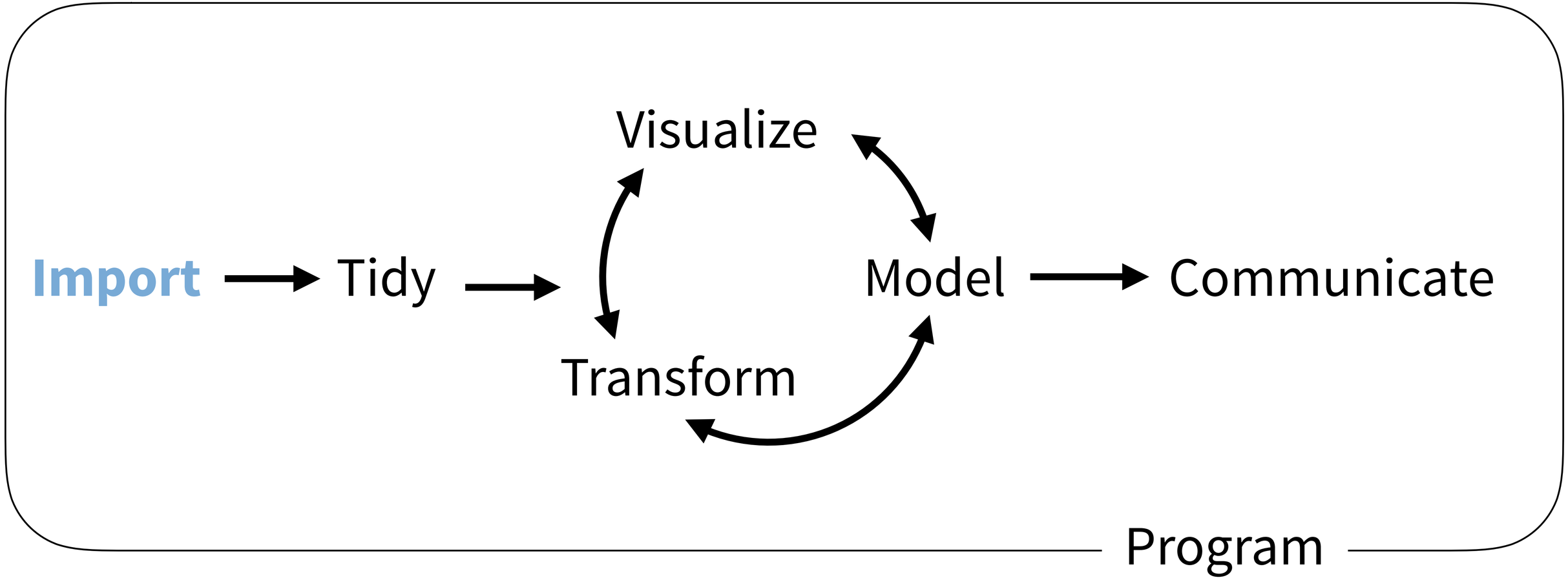
Import Data with

(Applied) **Data Science**



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
read_csv()	Comma separated values
read_csv2()	Semi-comma 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

readr functions

function	reads
read_csv()	**Comma separated values**
read_csv2()	Semi-comma 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

ozone.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,.
```

ozone.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

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

Your Turn

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

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

```
View(ozone)
```

. = NA

```
View(ozone)
```

	date	longitude	latitude	ozone
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	.

read_csv()

readr functions share a common syntax

```
df<- read_csv("file.csv", na = c("", "NA"))
```

object to save output into

path from working directory to file

Values to convert to NA

Your Turn

Reread in ozone.csv. But this time convert the "."'s to NA's. How many NA's are in the ozone column?

Your Turn

Reread in ozone.csv. But this time convert the "."'s to NA's. How many NA's are in the ozone column?

```
ozone <- read_csv("ozone.csv", na = ".")
View(ozone)
ozone %>%
  summarize(nas = sum(is.na(ozone)), n = n())
## # A tibble: 1 × 2
##     nas     n
##   <int> <int>
## 1  3092 25224
```

Quiz

What "type" of column is ozone?

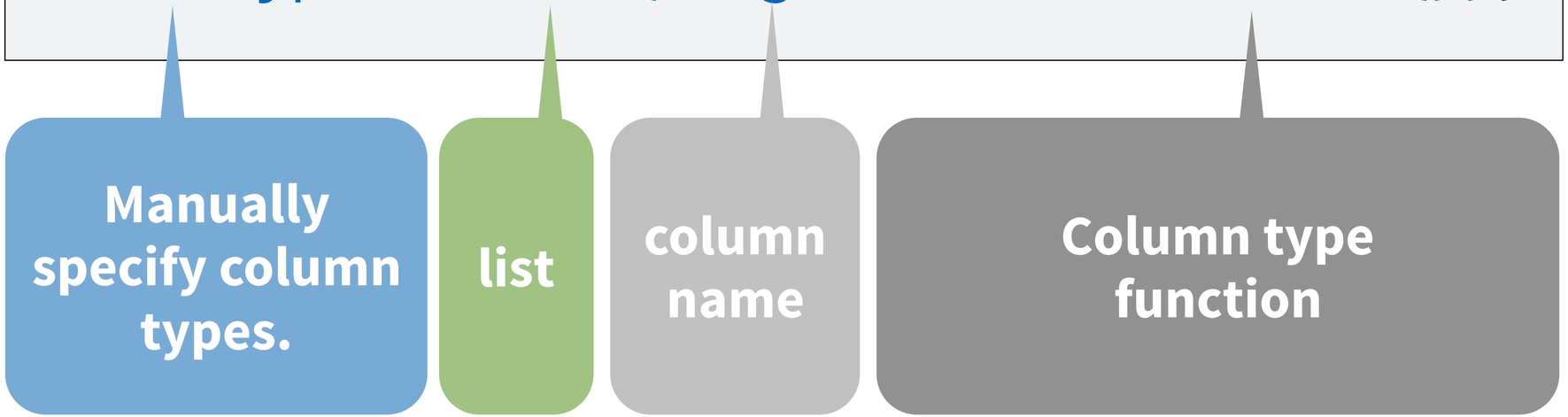
```
ozone
## # A tibble: 25,224 × 4
##          date longitude latitude ozone
##         <dttm>     <dbl>    <dbl> <chr>
## 1  1985-10-01  -179.375    -87.5  <NA>
## 2  1985-10-01  -178.125    -87.5  <NA>
## 3  1985-10-01  -176.875    -87.5  <NA>
## 4  1985-10-01  -175.625    -87.5  <NA>
## 5  1985-10-01  -174.375    -87.5  <NA>
## 6  1985-10-01  -173.125    -87.5  <NA>
## 7  1985-10-01  -171.875    -87.5  <NA>
## 8  1985-10-01  -170.625    -87.5  <NA>
## 9  1985-10-01  -169.375    -87.5  <NA>
## 10 1985-10-01  -168.125    -87.5  <NA>
### ... with 25,214 more rows
```

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

read_csv()

readr functions share a common syntax

```
df<- read_csv("file.csv", na = c("", "NA"),
  col_types = list(longitude = col_double()))
```



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

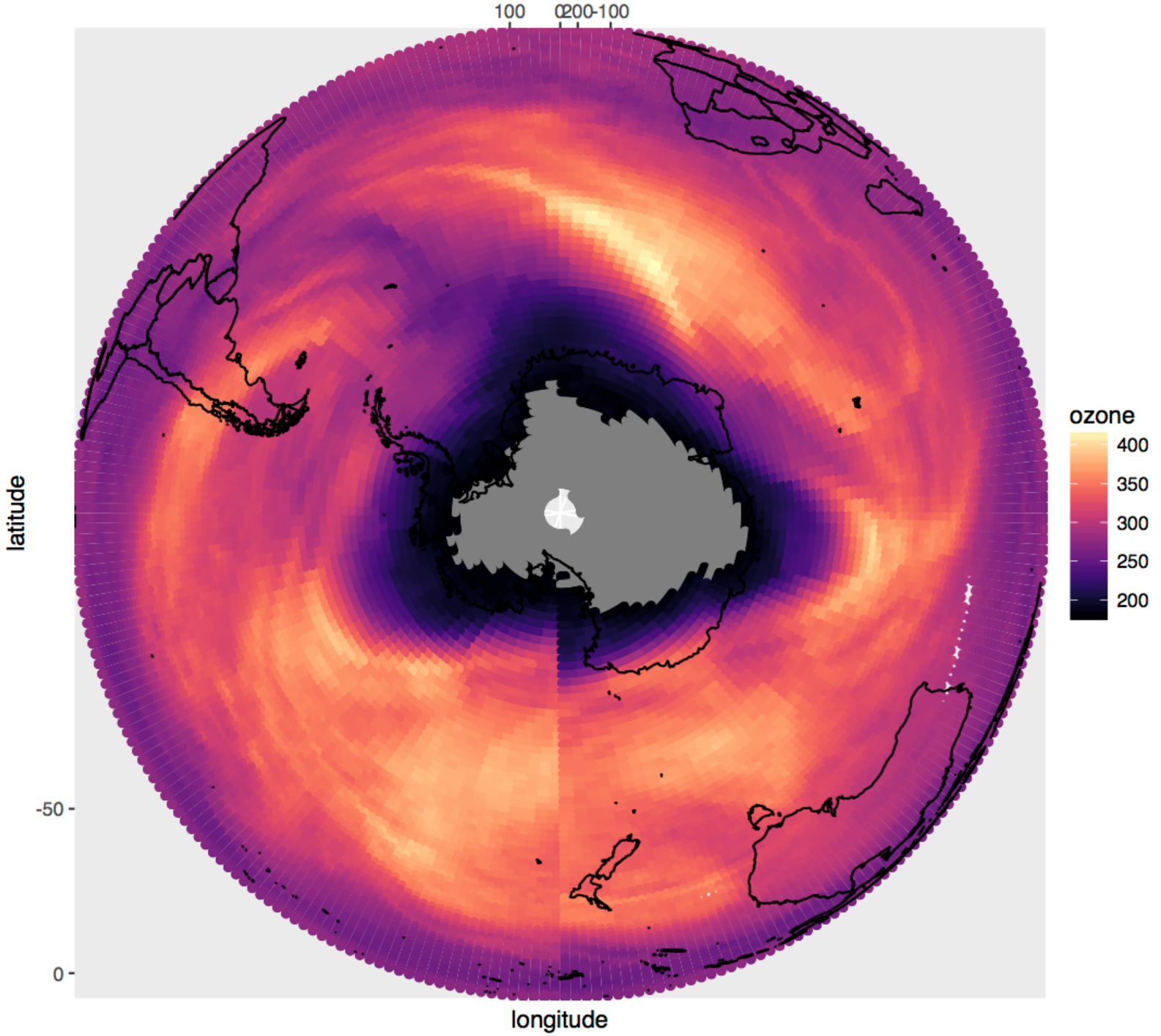
Your Turn

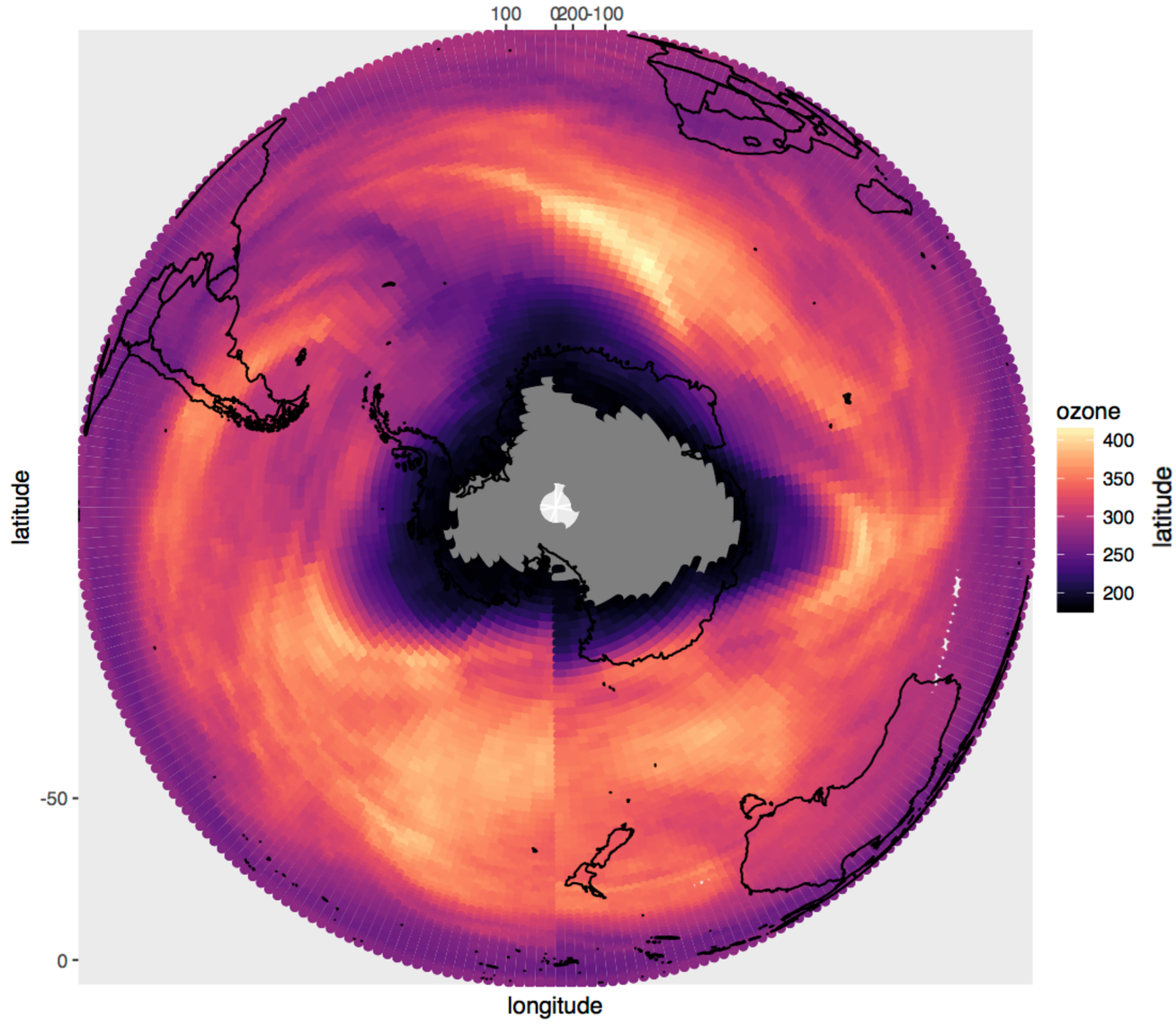
Read in ozone.csv. accounting for NA's and setting the col_type of ozone to a double. Then make this plot. What do you see?

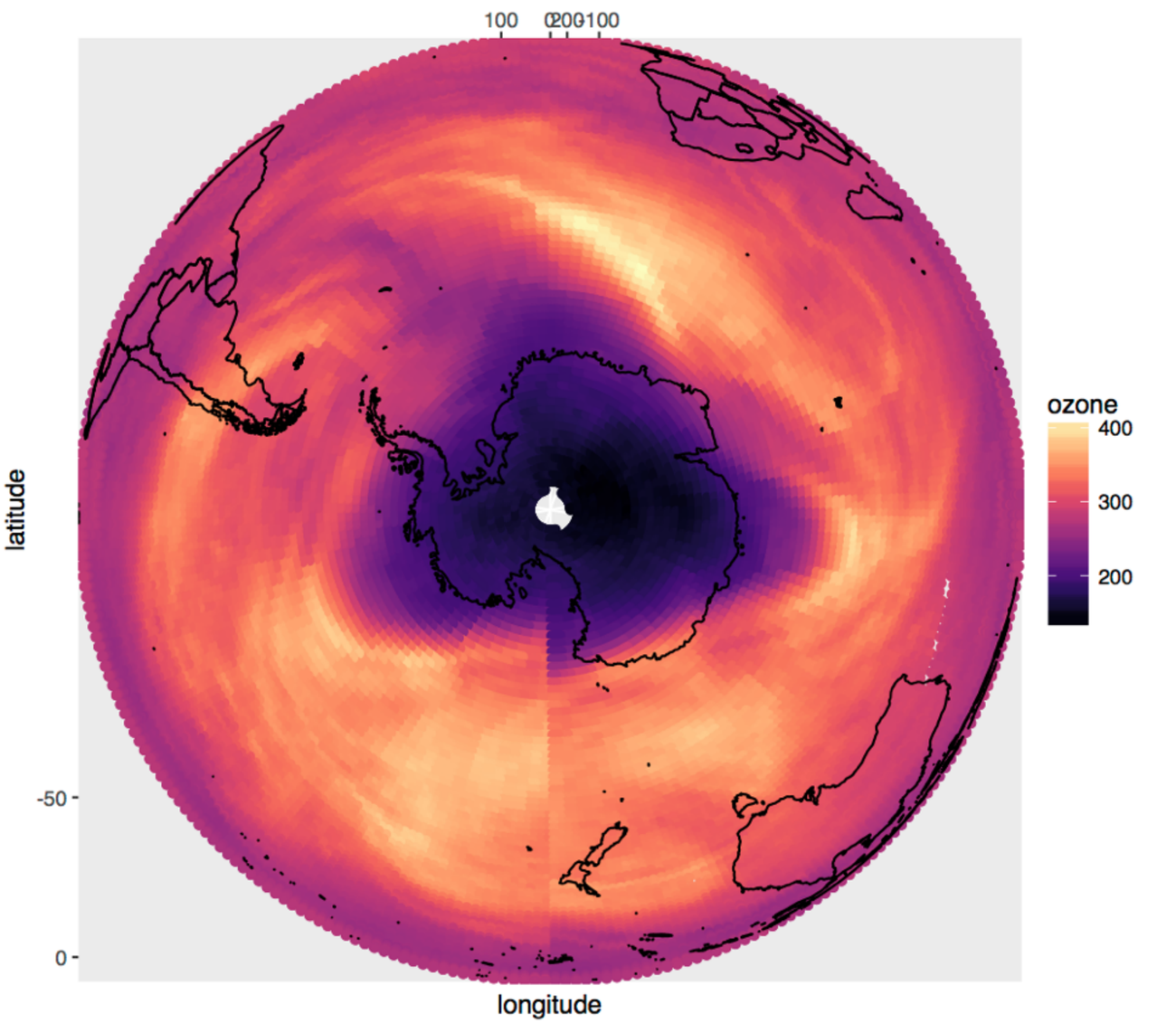
```
library(viridis)
world <- map_data(map = "world")
ozone %>%
  ggplot() +
    geom_point(aes(longitude, latitude, color = ozone)) +
    geom_path(aes(long, lat, group = group), data = world) +
    coord_map("ortho", orientation=c(-90, 0, 0)) +
    scale_color_viridis(option = "A")
```

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


library(viridis)
world <- map_data(map = "world")
ozone %>%
  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")
```


100
0
200
-100
latitude
-50
0
longitude
ozone
400
350
300
250
200


100
0
200
-100
latitude
-50
0
longitude
ozone
400
350
300
250
200


100
0
200
100
latitude
-50
0
longitude
ozone
400
300
200

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