Stat 565

Graphics and EDA

Jan 7 2016

Charlotte Wickham

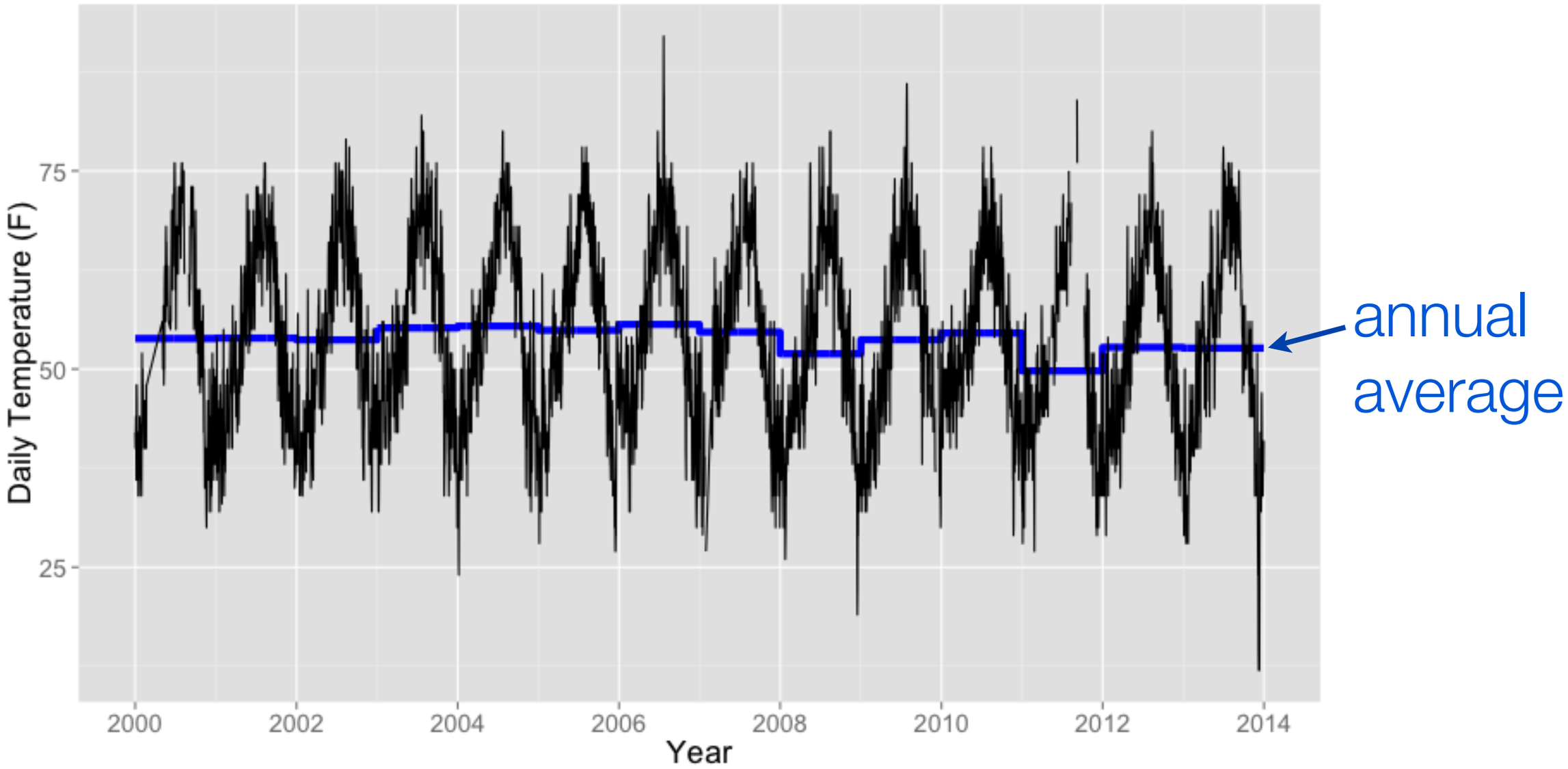
stat565.cwick.co.nz

Your turn

1. Get your R/RStudio up and running.
2. Grab today's code from the class webpage.
3. Run up to: `=== HERE! ===`

OR make a friend and sit next to someone with a laptop

Daily average temperature in Corvallis



Our goal today is to go from the raw data, to this plot

Today

Getting on the same page with R

Data structure and dealing with dates in R

Graphical exploration of time series using ggplot2

Aggregation and averaging using dplyr

There is no requirement you use the packages I use
I'm just showing you the way I do things

This also won't be very thorough introduction to any of these packages.
My goal is to give you some examples of how useful they are,
some experience using them,
and pointers on where to learn more.

You can always ask me for more info!

Data goes in data.frames

Original data

```
  PST      temp
1 2000-1-1   42
2 2000-1-2   40
3 2000-1-3   42
4 2000-1-4   44
5 2000-1-5   42
6 2000-1-6   36
```

character

We'll make these

```
        date year month yday
1 2000-01-01 2000     1    1
2 2000-01-02 2000     1    2
3 2000-01-03 2000     1    3
4 2000-01-04 2000     1    4
5 2000-01-05 2000     1    5
6 2000-01-06 2000     1    6
```

Times & Dates in R

R won't know something is a date unless you tell it:

```
01/02/2014 - 01/01/2014
"01/02/2014" - "01/01/2014"
as.Date("01/02/2014") - as.Date("01/01/2014")
```

Same with times

See ?DateTimeClasses

(and all the suggestions in the See Also section)

lubridate

An R package that makes converting and working with dates and times a little easier.

```
library(lubridate)

# ymd, dmy, mdy,...for converting strings that contain dates
ymd("2010-12-01")
mdy("1/01/10")
# More generally use parse_date_time
parse_date_time("1/01/10", "mdy")

# year, month, day, yday, wday for pulling out parts of a date
today()
class(today())
month(today())
year(today())
yday(today())
wday(today())
```

?lubridate

Convert PST to date

```
corv$date <- ymd(corv$PST)

corv$year <- year(corv$date)
corv$month <- month(corv$date)
corv$yday <- yday(corv$date)
```

Your turn

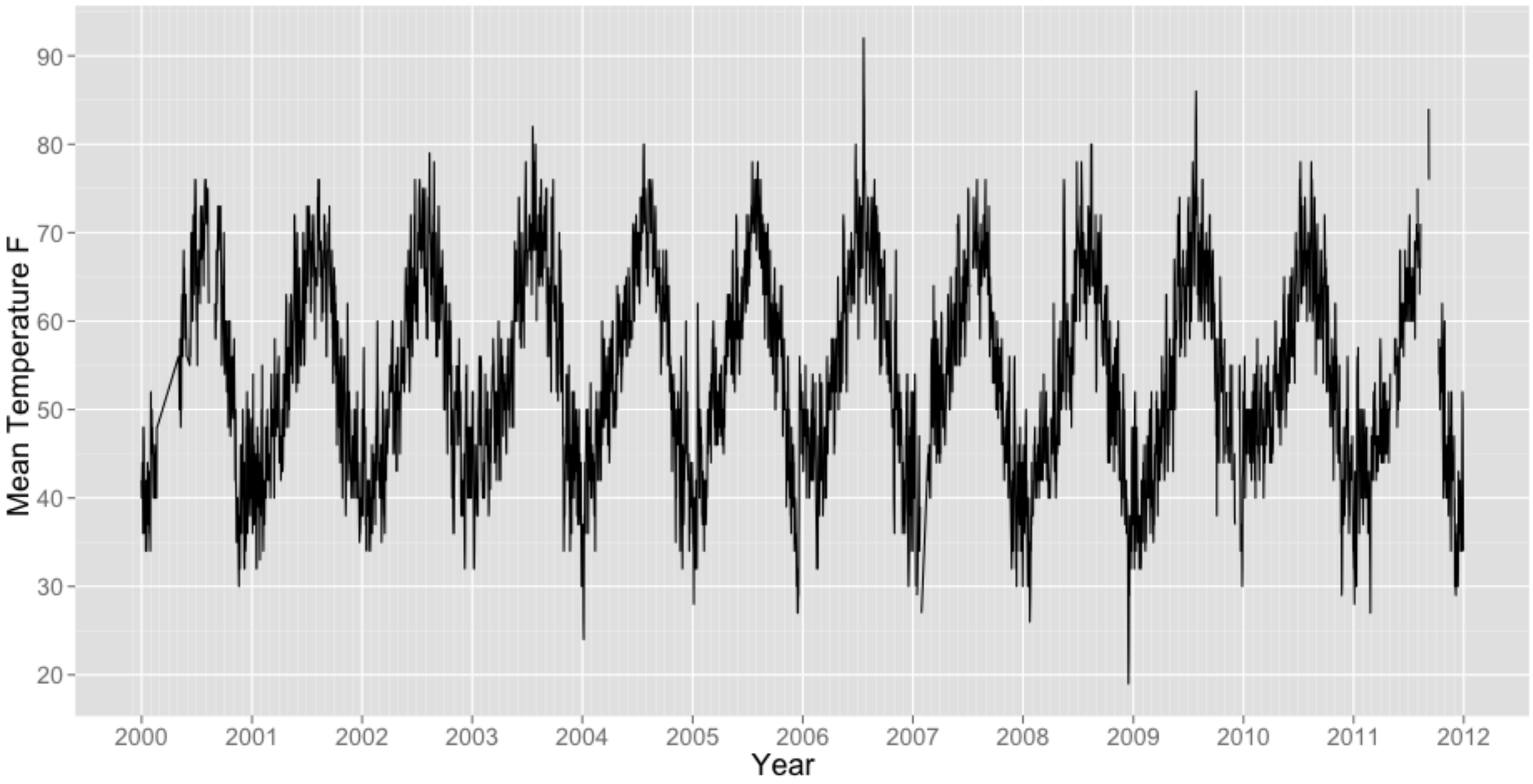
Scan the Details section in the help for parse_date_time to find out how formats are specified.

Then try to convert these strings to dates and times.

```
A <- "1-7-14"
B <- "Jan 7 14"
C <- "1:15 AM 2014-01-07"
D <- "3:25 PM"
E <- "Tues Jan 7 2014"
```

A "time series" plot in ggplot2

```
qplot(date, temp, data = corv, geom = "line")
```



```
qplot(x = date, y = temp, data = corv, geom = "line")
```

variable on the x-axis

variable on the y-axis

the data frame the variables are in

the geometric object used to represent the data:
point
boxplot
histogram
tile
...
http://docs.ggplot2.org

Your turn

```
qplot(date, temp, data = corv, geom = "line")
```

1. Plot precipitation against temperature using points.

2. Plot precipitation against month using points.

3. Try the previous plot with `geom = "jitter"` , what does it do?

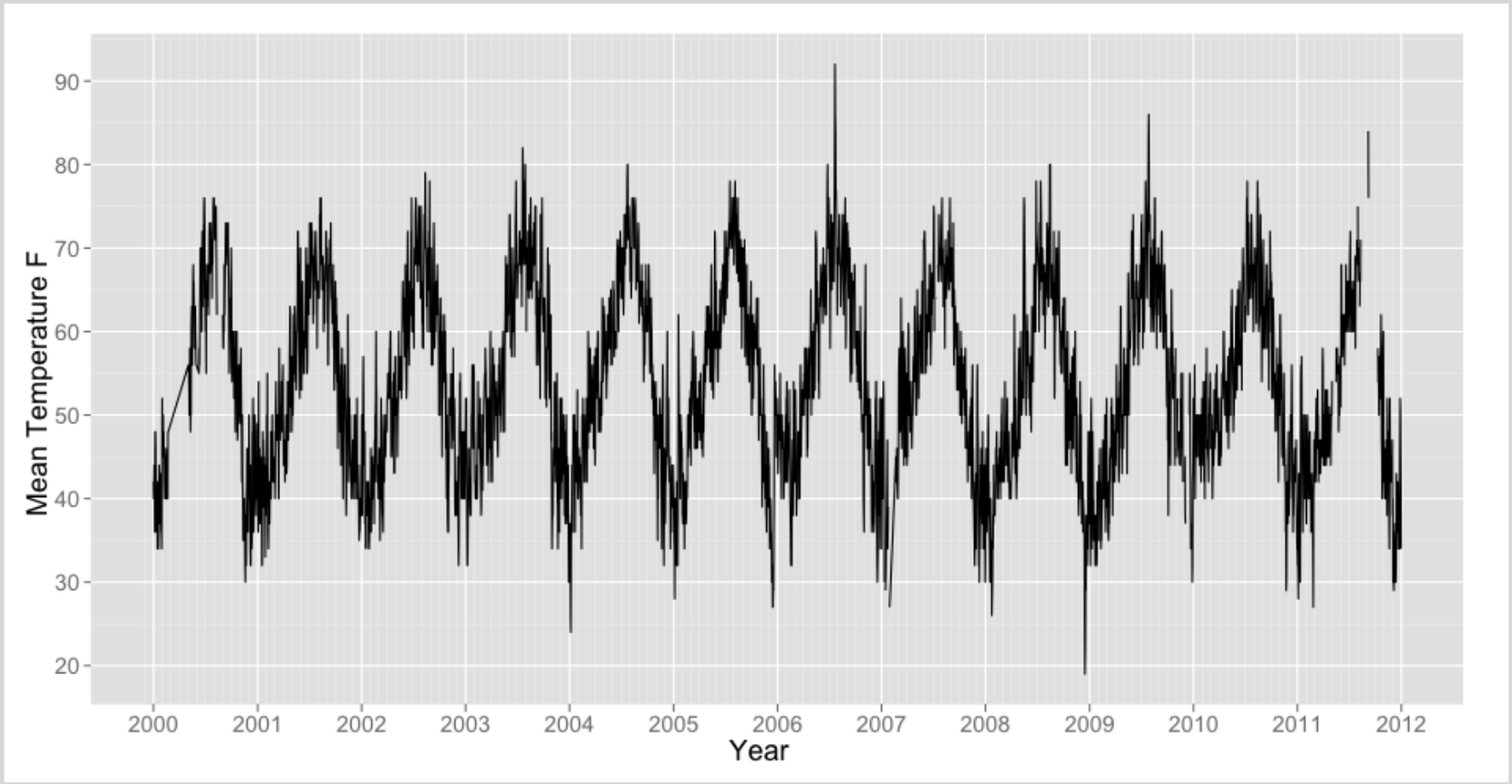
Map other aesthetics to other variables

depends on the geom: look at the help to find out what is available,
or at docs.ggplot2.org for examples

depends on the variable: some aesthetics only work with discrete variables,
some have different behaviour for discrete variables

```
?geom_point
qplot(date, temp, data = corv)
qplot(date, temp, data = corv,
      colour = month)
qplot(date, temp, data = corv,
      colour = factor(month))
qplot(date, temp, data = corv,
      size = month)
qplot(date, temp, data = corv,
      shape = month)
```

Your turn



How could I display the data to examine the average annual seasonal pattern?

Sketch your ideas, or try to make them in R

Looking for a trend

A simple exploratory approach might be to aggregate our daily data to annual data.

For example, let's just find the average temperature for each year and plot it against years.

There are lots of ways to do this, but it's an ideal task to introduce `dplyr`

"dplyr provides a flexible grammar of data manipulation. It's the next iteration of plyr, focused on tools for working with data frames (hence the *d* in the name)."

dplyr

A set of data (frame) manipulation verbs:

`filter` return certain rows

`select` return certain columns

`arrange` arrange rows (i.e. sorted)

`summarize` collapse to a single row

`mutate` add new columns

+ grouped operations with group_by

mutate and summarise

```
last_year <- filter(corv, year == 2013)

mutate(last_year,
  avg_temp = mean(temp, na.rm = TRUE),
  n_temp = sum(!is.na(temp))
)

summarise(last_year,
  avg_temp = mean(temp, na.rm = TRUE),
  n_temp = sum(!is.na(temp))
)
```

What's happening? What is being returned?

Chaining operations %>%

`x %>% f(y)` is equivalent to `f(x, y)`

read %>% as “then”

```
last_year %>%
  mutate(
    avg_temp = mean(temp, na.rm = TRUE),
    n_temp = sum(!is.na(temp))
  )

last_year %>%
  summarize(
    avg_temp = mean(temp, na.rm = TRUE),
    n_temp = sum(!is.na(temp))
  )
```

mutate and summarize

```
mutate(.data, ...)
```

make **new columns** in the data.frame as defined by the expressions

data.frame

expressions that calculate variables

```
summarize(.data, ...)
```

make a **new data.frame** with columns defined by these expressions

group_by

Two very common use cases:

```
corv %>% group_by(year) %>%
  summarise(avg_temp = mean(temp, na.rm = TRUE))
```

```
corv %>% group_by(year) %>%
  mutate(avg_temp = mean(temp, na.rm = TRUE))
```

Can you figure out what is being returned?

data.frame

group the
data.frame by year

```
corv %>% group_by(year) %>%
    summarise(avg_temp = mean(temp, na.rm = TRUE))
```

take the grouped data.frame and
summarise by each group

Take the `corv` data.frame and group it by year. For each year (i.e. group) use the `summarise` function to create a new column called `avg_temp`, that contains the value `mean(temp,…)`

```
# summarise to annual level data
ann_temp <- corv %>% group_by(year) %>%
  summarise(
    avg_temp = mean(temp, na.rm = TRUE),
    n = sum(!is.na(temp)))

qplot(year, avg_temp,
  data = ann_temp, geom = "line")
# but we lose all sense of scale
```

```
# An alternative, rather than summarise, mutate
corv <-  corv %>%
  group_by(year) %>%
  mutate(
    avg_temp = mean(temp, na.rm = TRUE),
    n = sum(!is.na(temp)))

qplot(date, avg_temp,
  data = corv, geom = "line",
  size = I(2), colour = I("blue")) +
  geom_line(aes(y = temp))
```

Your turn

Think of another annual summary, calculate it and plot it over time.

Try doing a monthly summary, or a day of the year summary (i.e. find the average Jan 1 temperature).

Learning more

Google is your friend

http://stackoverflow.com/

http://www.cookbook-r.com/Graphs/

Work through a vignette:

dplyr: https://cran.rstudio.com/web/packages/dplyr/vignettes/introduction.html

lubridate: https://cran.rstudio.com/web/packages/lubridate/vignettes/lubridate.html

Read papers outlining the ideas behind the packages:

ggplot2: http://vita.had.co.nz/papers/layered-grammar.html

lubridate:http://www.jstatsoft.org/v40/i03/paper

Ask me!