Interactive Data Visualizations in R with Shiny and ggplot2

ZevRoss
Know Your Data

Your "TA" for today is Shiny's creator

Joe Cheng

A training with three acts

- Data visualization with `ggplot2`
- Making your visualizations interactive with `Shiny`
- Graphic interactivity with `htmlwidgets`

Notes about software and exercises

- We're using RStudio Server so you all have the same software setup
- There is a worshop package -- load with `library(shinyRworkshop)`
- Open an exercise with `exercise(1)`
- Solution to an exercise with `solution(1)`

Plotting with ggplot2

Some introductory notes

Hadley Wickham

Developed by Hadley Wickham who has also developed other great R packages (**dplyr**, **reshape2** etc) and now works for RStudio.

Documentation

You can find documentation at:

docs.ggplot2.org/current/

Spelling

You can use American or British spelling in function names (e.g, color and colour both work).

Getting started

ggplot2 components

- Data and aesthetic mappings (`aes`)
- Layers: Geometric objects (`geom`)
- Layers: Statistical transformation (`stat`)
- Faceting for multipanel plots (`facet`)
- Scales (`scale`, not discussed today)
- Coordinate system (`coord`, not discussed today)

Creating a plot object

```
ggplot(data=Melanoma, aes(x=thickness, y=time))
```

This sets up the **plot object** but cannot be plotted because we have not added any layers.

ggplot requires a data frame

Unlike `base` you must use a data frame.

Layers

Layers

Two types of layers:

- Geometric objects (**geom**)
- Statistical transformations (**stat**)

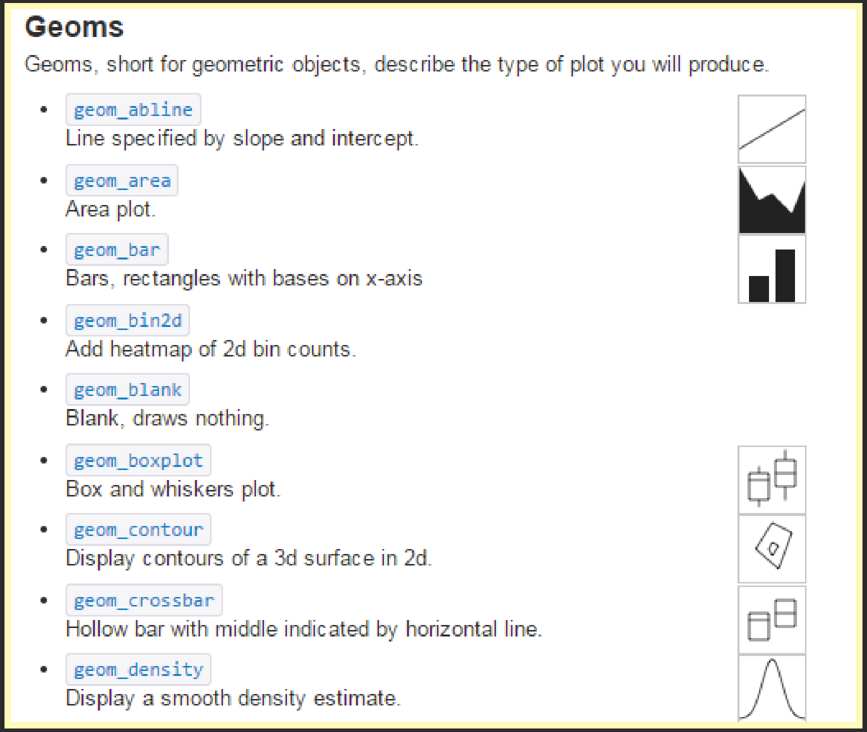
Layers: geoms

Examples:

- geom_point()
- geom_boxplot()
- geom_histogram()
- geom_text()

For more geoms see the ggplot2 docs

docs.ggplot2.org/current/



Geoms

Geoms, short for geometric objects, describe the type of plot you will produce.

- `geom_abline`
  Line specified by slope and intercept.
- `geom_area`
  Area plot.
- `geom_bar`
  Bars, rectangles with bases on x-axis
- `geom_bin2d`
  Add heatmap of 2d bin counts.
- `geom_blank`
  Blank, draws nothing.
- `geom_boxplot`
  Box and whiskers plot.
- `geom_contour`
  Display contours of a 3d surface in 2d.
- `geom_crossbar`
  Hollow bar with middle indicated by horizontal line.
- `geom_density`
  Display a smooth density estimate.

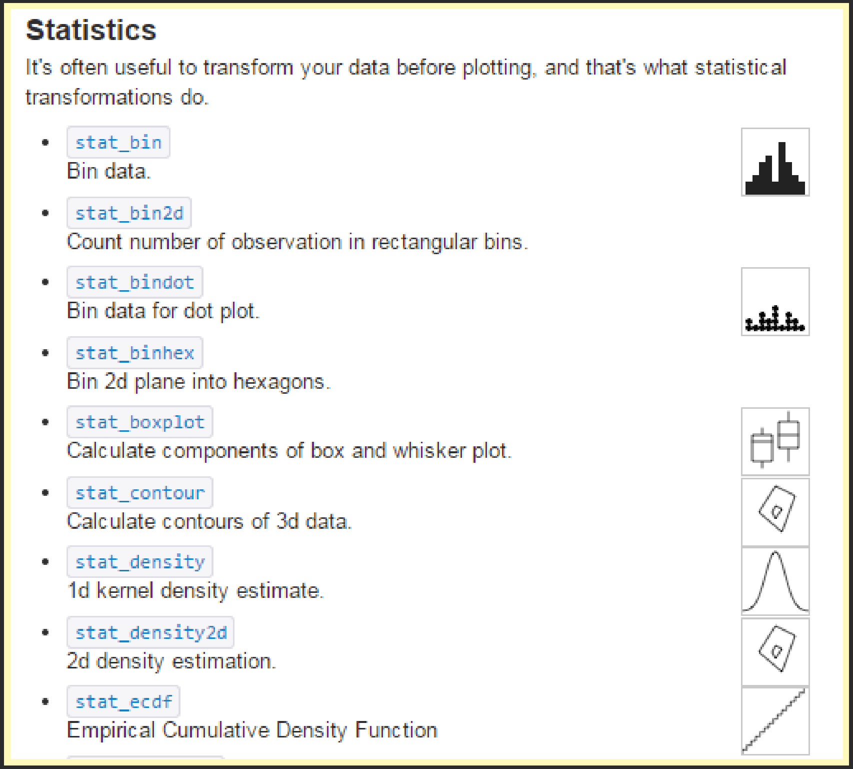
Layers: stats

Examples:

- stat_smooth()
- stat_density()
- stat_step()

For more stats see the ggplot2 docs

docs.ggplot2.org/current/



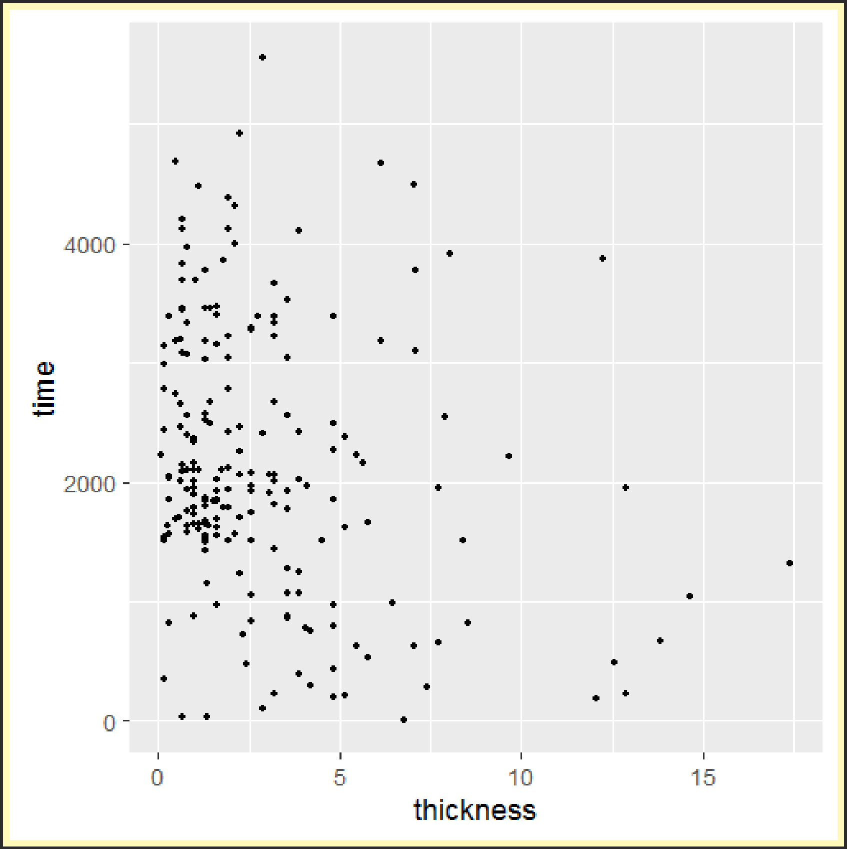
Statistics

It's often useful to transform your data before plotting, and that's what statistical transformations do.

- `stat_bin`
  Bin data.
- `stat_bin2d`
  Count number of observation in rectangular bins.
- `stat_bindot`
  Bin data for dot plot.
- `stat_binhex`
  Bin 2d plane into hexagons.
- `stat_boxplot`
  Calculate components of box and whisker plot.
- `stat_contour`
  Calculate contours of 3d data.
- `stat_density`
  1d kernel density estimate.
- `stat_density2d`
  2d density estimation.
- `stat_ecdf`
  Empirical Cumulative Density Function

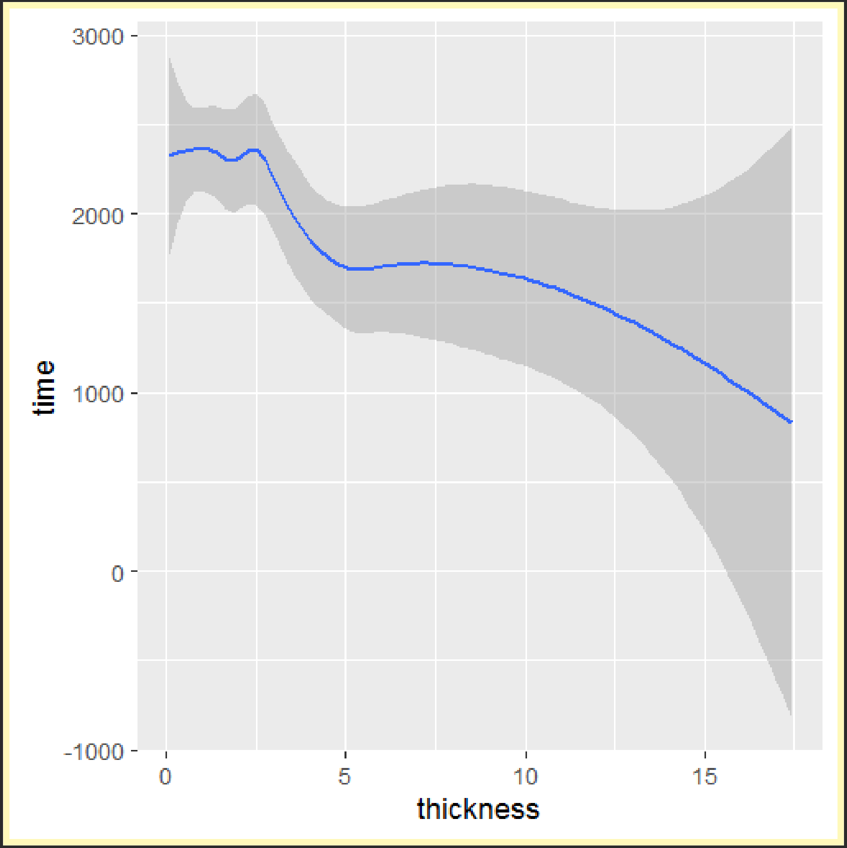
Adding a geom layer is easy

```
> ggplot(Melanoma, aes(x=thickness, y=time)) +
geom_point()
```



Adding a stat layer is easy

```
> ggplot(Melanoma, aes(x=thickness, y=time)) +
stat_smooth()
```

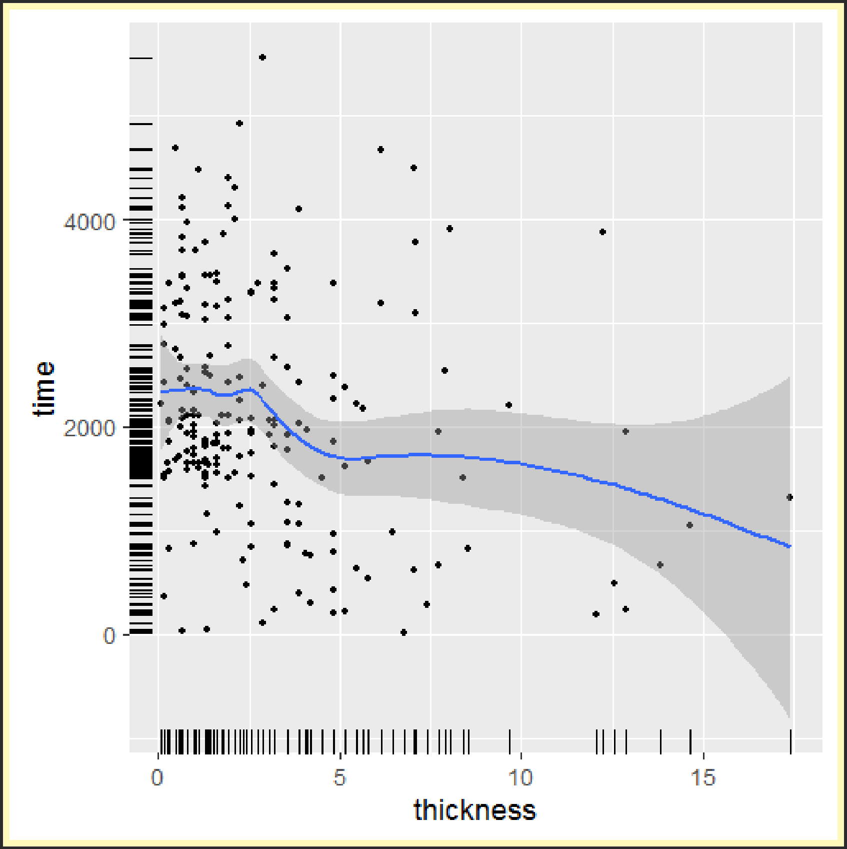


Adding more than one layer is easy

What if you want the smooth **and** the points **and** even a rug plot?

Adding multiple layers is easy

```
> ggplot(Melanoma, aes(x=thickness, y=time)) +
geom_point() + stat_smooth() + geom_rug()
```



With multiple line plots keep + on first line

```r
# This will cause an ERROR
ggplot(Melanoma, aes(x=thickness, y=time))
  + geom_point()
  + stat_smooth()
```

```r
# This is OK
ggplot(Melanoma, aes(x=thickness, y=time)) +
  geom_point() +
  stat_smooth()
```

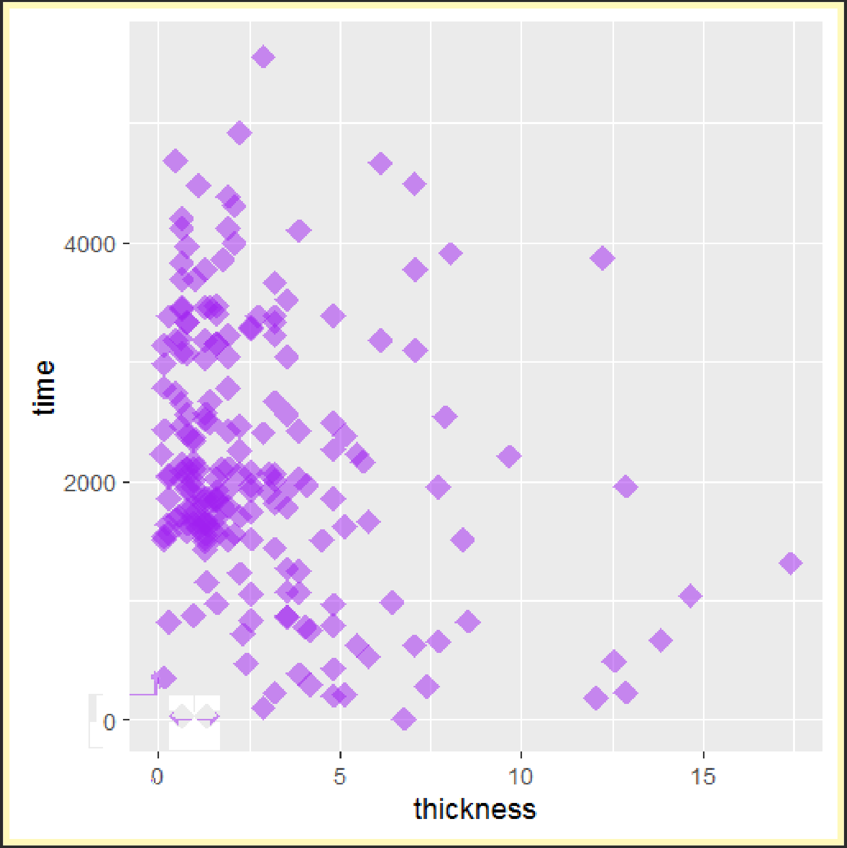
Setting the appearance of layers

Like `base` use plotting arguments:

- `color`
- `fill`
- `shape`
- `size`
- `alpha`

Setting the appearance of layers

```
> ggplot(Melanoma, aes(x=thickness, y=time)) +
geom_point(color="purple", shape=18, size=7, alpha=0.5)
```



With aes you are mapping not setting

The `aes` function is used to *map* variables to aesthetics.

Initial prep for an example

Our sex variable is numeric (0,1) but needs to be a factor or character.

```
Melanoma$sex<-factor(Melanoma$sex, levels=c(0,1),
labels=c("Female", "Male"))
```

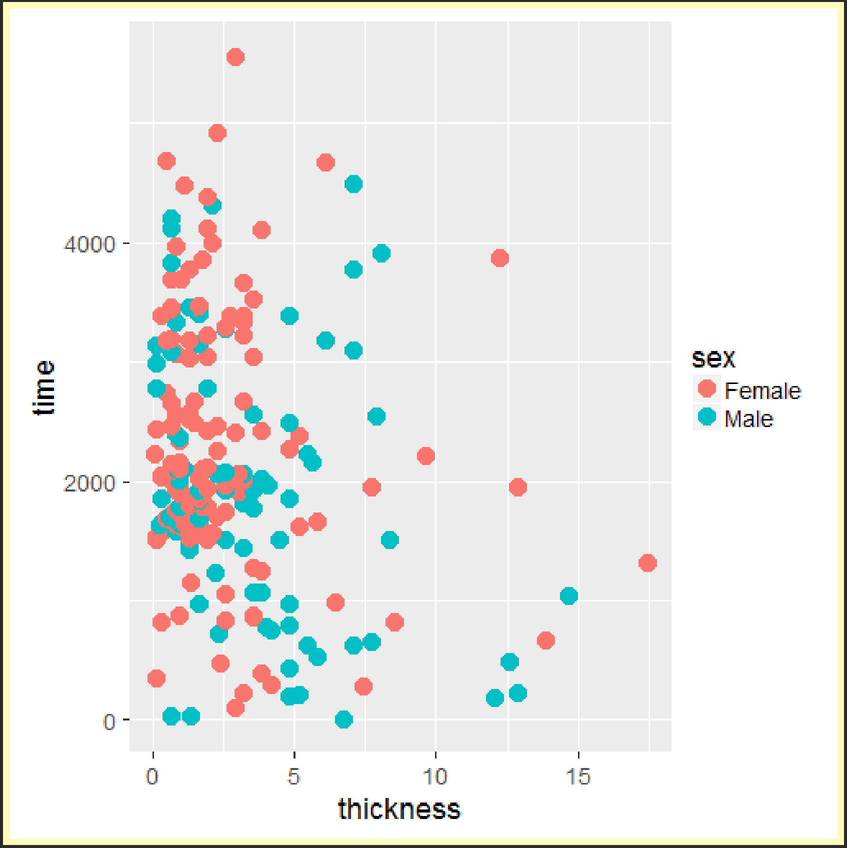
Using aes to map color

```
ggplot(Melanoma, aes(x=thickness, y=time, color=sex))
```

Using aes to map color

Here we *map* the values in the variable sex to colors

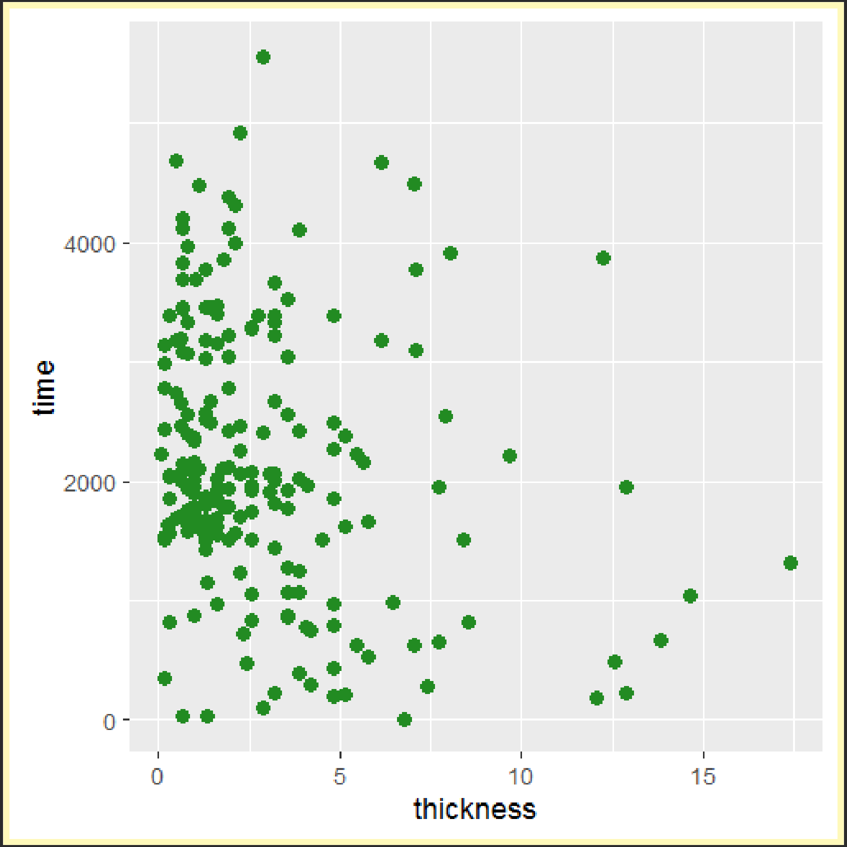
```
> ggplot(Melanoma, aes(x=thickness, y=time, color=sex)) +
geom_point(size=5)
```



Setting appearance

To simply change color don't use aes and set the color in the layer:

```
> ggplot(Melanoma, aes(x=thickness, y=time)) +
geom_point(color="forestgreen", size=4)
```



Strange things happen if you confuse mapping and setting

If you try to `map` to a non-variable

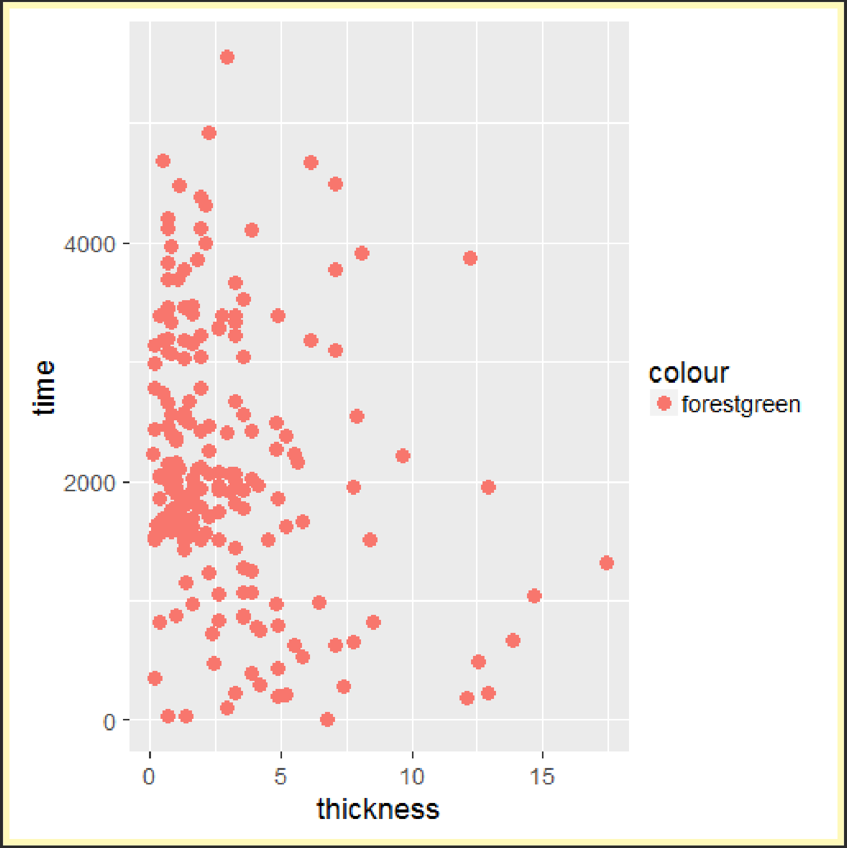
```r
ggplot(Melanoma, aes(x=thickness, y=time,
color="forestgreen")) + geom_point(size=4)
```

Strange things happen if you confuse mapping and setting

- This Creates a new variable called `"forestgreen"`.
- All the values for the new variable are `forestgreen`.
- Then it **maps** that variable to color.

Strange things happen...

```
> ggplot(Melanoma, aes(x=thickness, y=time,
color="forestgreen")) + geom_point(size=4)
```



exercise 1 (1-7)

Layers inherit data and aesthetics

```r
ggplot(Melanoma, aes(x=thickness, y=time, color=sex)) +
  geom_point()
```

What if you want different aesthetics in different layers?

You can apply mapping directly to a layer

```
ggplot(Melanoma) +
  geom_point(aes(x=thickness, y=time, color=sex))
```

These are equivalent

```
ggplot(Melanoma, aes(x=thickness, y=time, color=sex)) +
  geom_point()
```

```
ggplot(Melanoma) +
  geom_point(aes(x=thickness, y=time, color=sex))
```

You can even split aes arguments

```
ggplot(Melanoma, aes(x=thickness, y=time)) +
  geom_point(aes(color=sex))
```

The data argument can also occur in the layers

```
ggplot() +
  geom_point(data=Melanoma,aes(x=thickness, y=time))
```

Beware: in layers you must include data=

```
# this is OK
ggplot(Melanoma) +
  geom_point(aes(x=thickness, y=time))
```

```
# this is NOT ok!!!
ggplot() +
  geom_point(Melanoma,aes(x=thickness, y=time))
#Error!!!!
```

```
# OK: must include data= in layers
ggplot() +
  geom_point(data=Melanoma,aes(x=thickness, y=time))
```

Why might it be useful to include data in the layers?

Multiple layers - different datasets

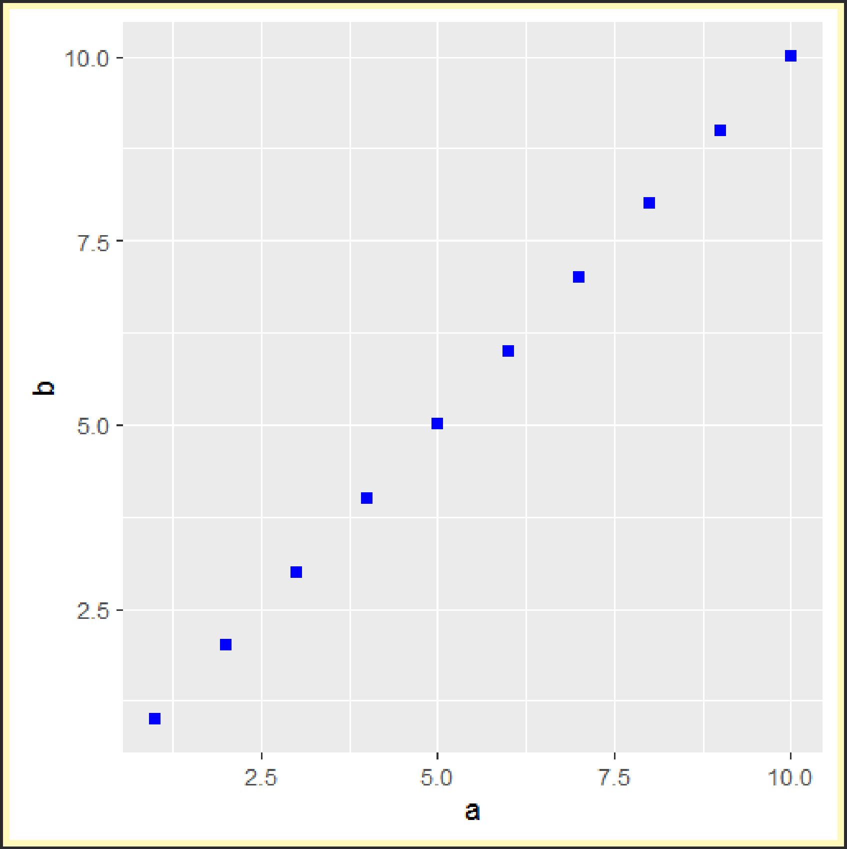
Dataset 1:

```
dat1<-data.frame(a=1:10, b=1:10)
```

Multiple layers - different datasets

Dataset 1:

```
> ggplot(dat1, aes(a, b)) +
+   geom_point(shape=15, size=3, color="blue")
```



Multiple layers - different datasets

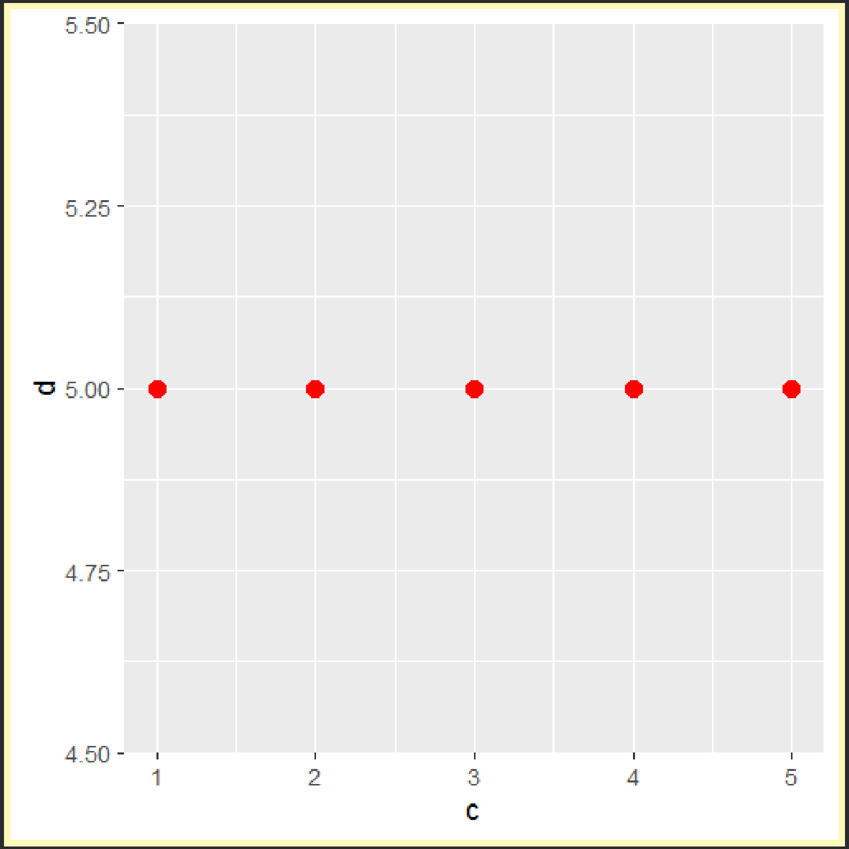
Dataset 2:

```r
dat2<-data.frame(c=1:5, d=rep(5,5))
```

Multiple layers - different datasets

Dataset 2:

```
> ggplot(dat2, aes(c, d)) + geom_point(color="red",
size=5)
```

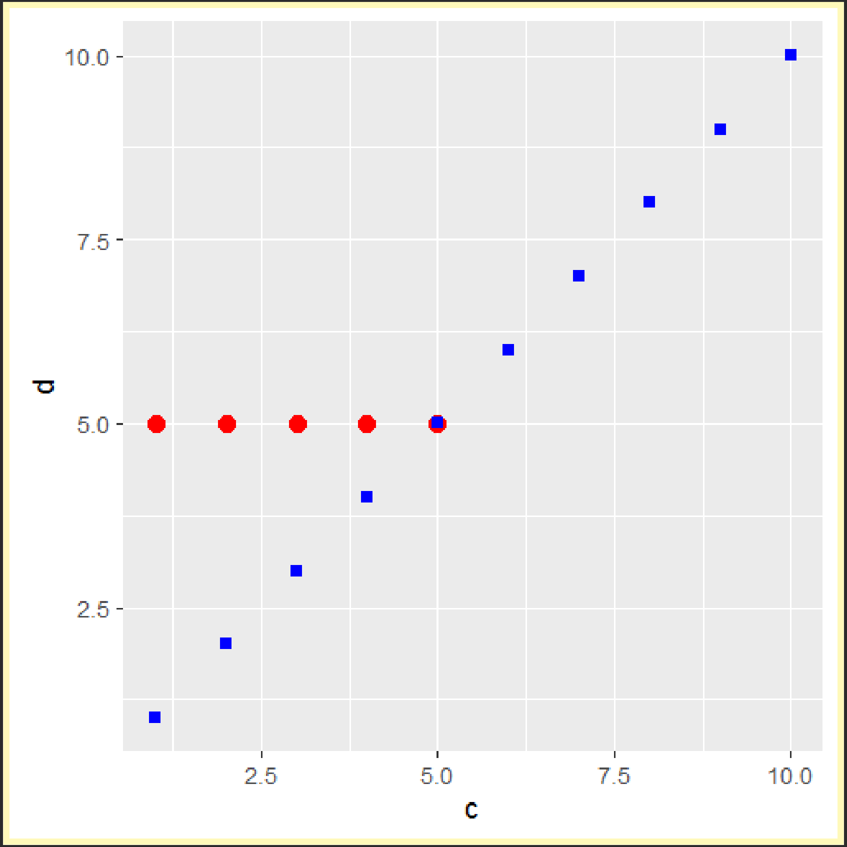


Multiple layers - different datasets

We have two datasets. Layers inherit from `ggplot()` so we want to include data in the layers.

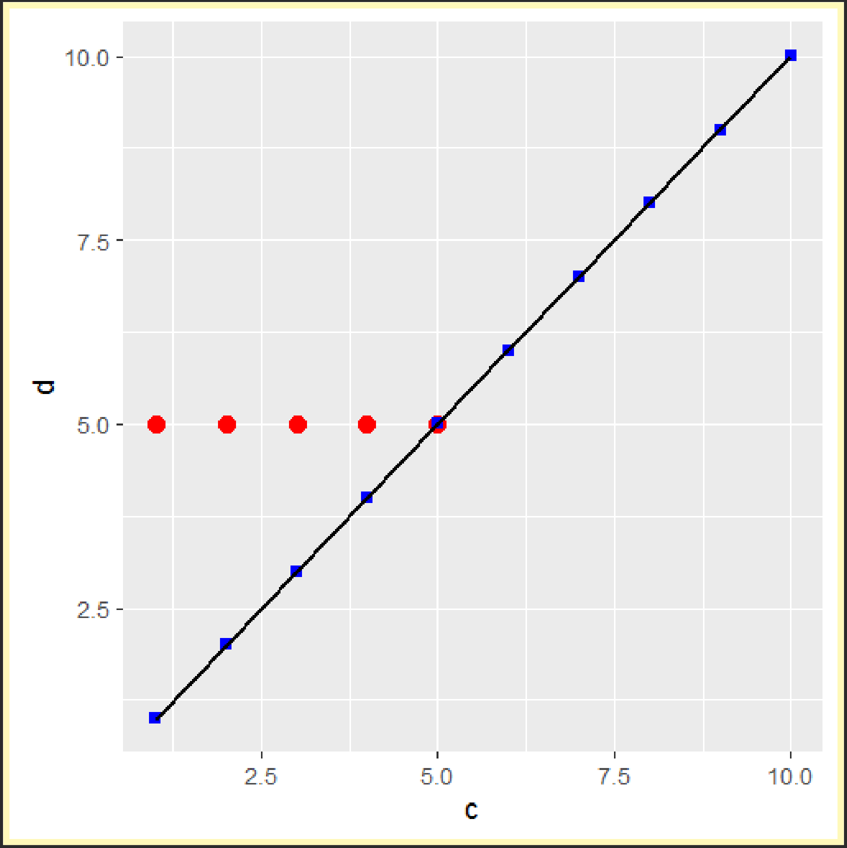
Multiple layers - different datasets

```
> ggplot() + geom_point(data=dat2, aes(c, d), color="red",
size=5) + geom_point(dat=dat1, aes(a, b), shape=15,
size=3, color="blue")
```



Multiple layers - different datasets

```
> ggplot() + geom_point(data=dat2, aes(c, d), color="red",
size=5) + geom_point(dat=dat1, aes(a, b), shape=15,
size=3, color="blue") + geom_line(dat=dat1, aes(a, b),
size=1)
```

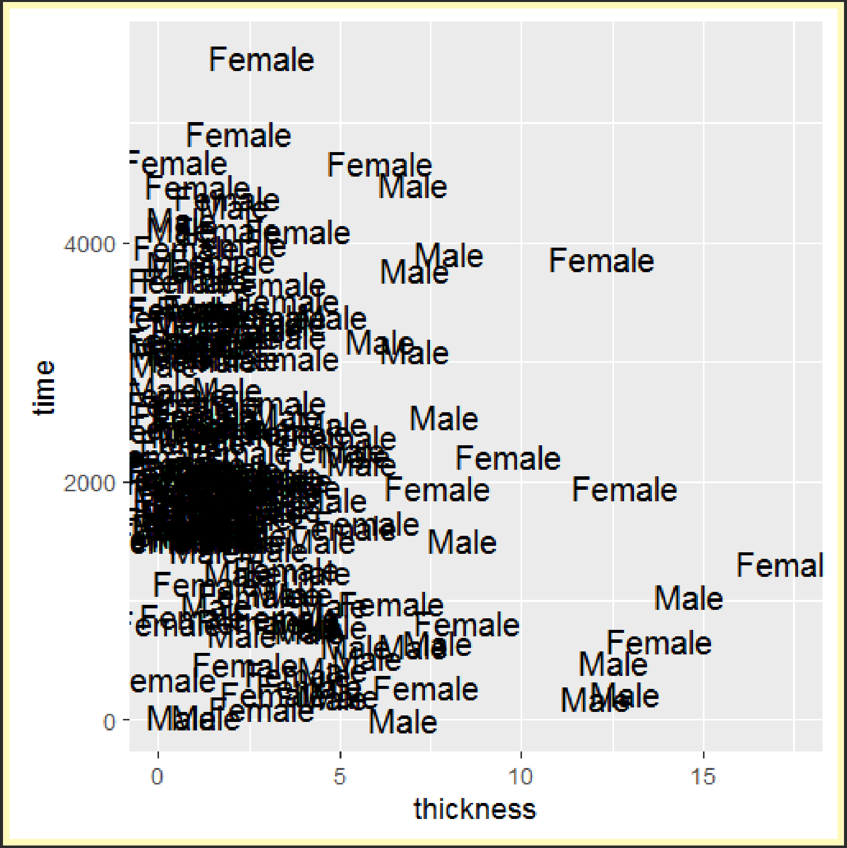


Aesthetic mapping can also be applied, for example, to:

- Labels
- Size
- Groups

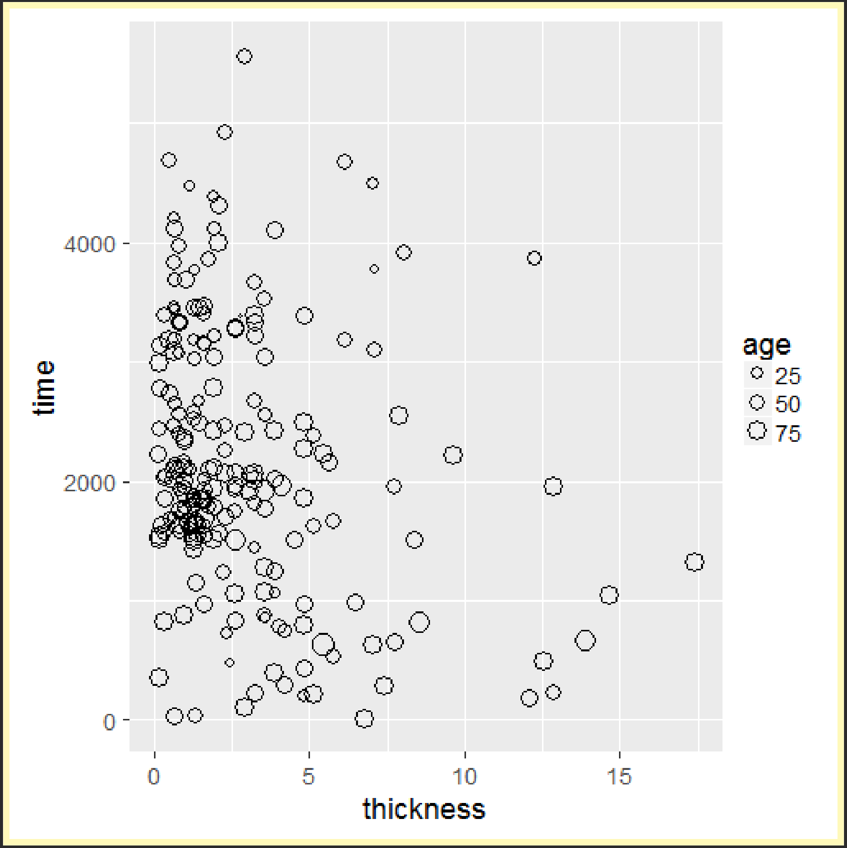
Aesthetic mapping: Labels

```
> ggplot(Melanoma, aes(x=thickness, y=time, label=sex)) +
geom_text(size=7)
```



Aesthetic mapping: Size

```
> ggplot(Melanoma, aes(x=thickness, y=time, size=age)) +
geom_point(shape=21)
```



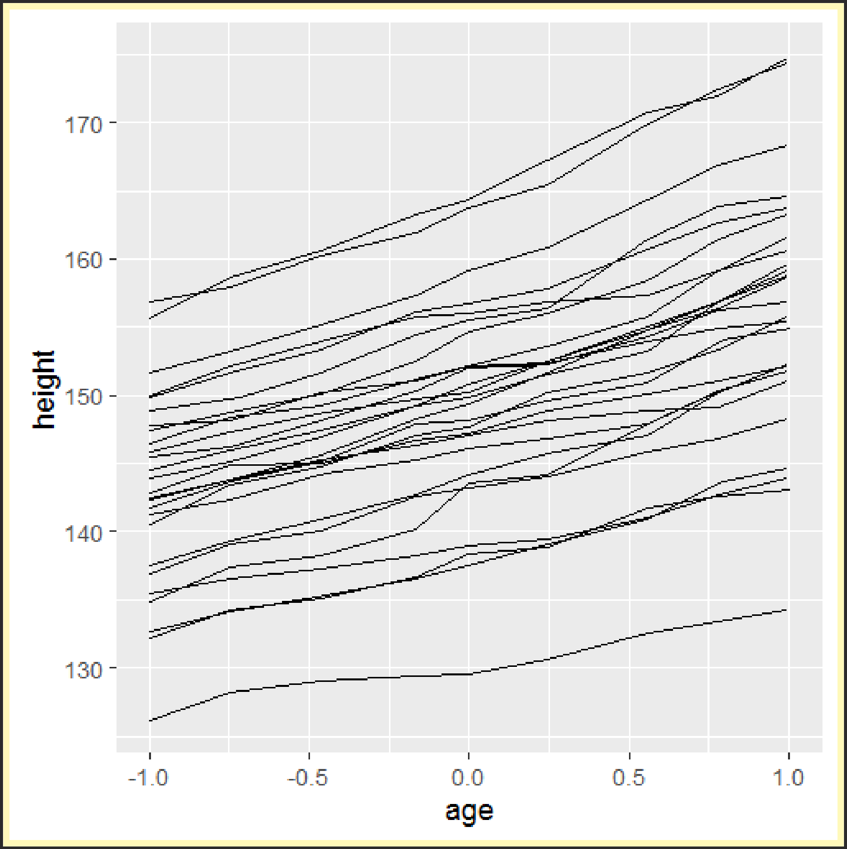
Aesthetic mapping: Groups

`group` works the same as `color` except the groups are not uniquely colored/identified in legend. An example:

```
library(nlme)
head(data.frame(Oxboys))
##   Subject     age height Occasion
## 1       1 -1.0000  140.5        1
## 2       1 -0.7479  143.4        2
## 3       1 -0.4630  144.8        3
## 4       1 -0.1643  147.1        4
## 5       1 -0.0027  147.7        5
## 6       1  0.2466  150.2        6
```

Aesthetic mapping: Groups

```
> ggplot(Oxboys, aes(age, height, group=Subject)) +
geom_line()
```

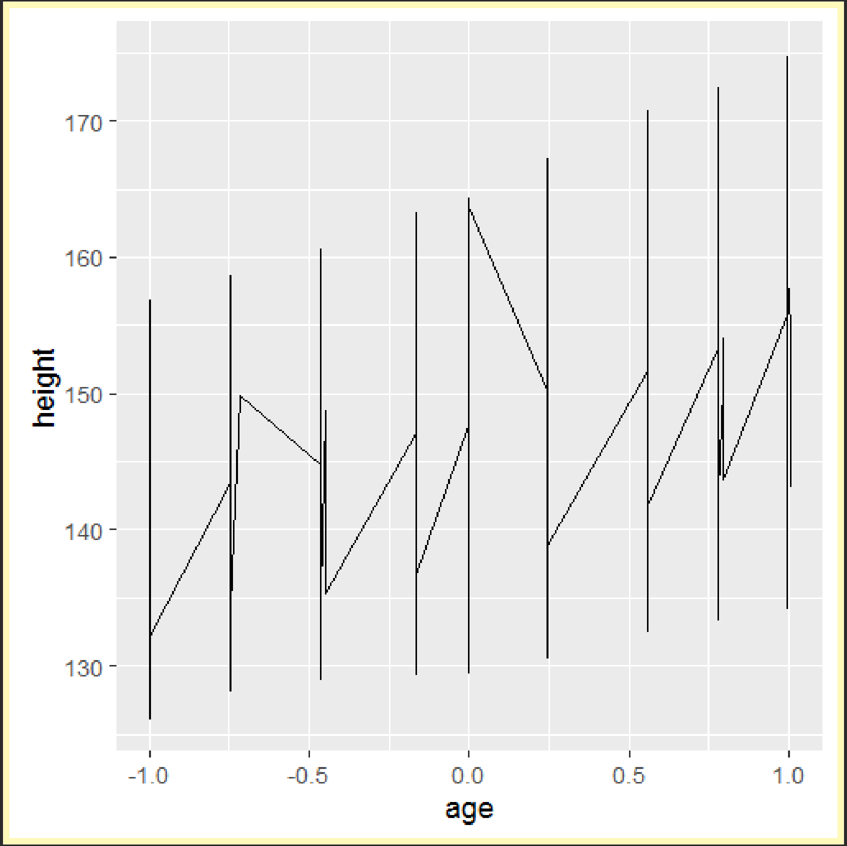


Aesthetic mapping: Groups

In this case what happens if you leave off the grouping variable?

Aesthetic mapping: Groups

```
> ggplot(Oxboys, aes(age, height)) + geom_line()
```

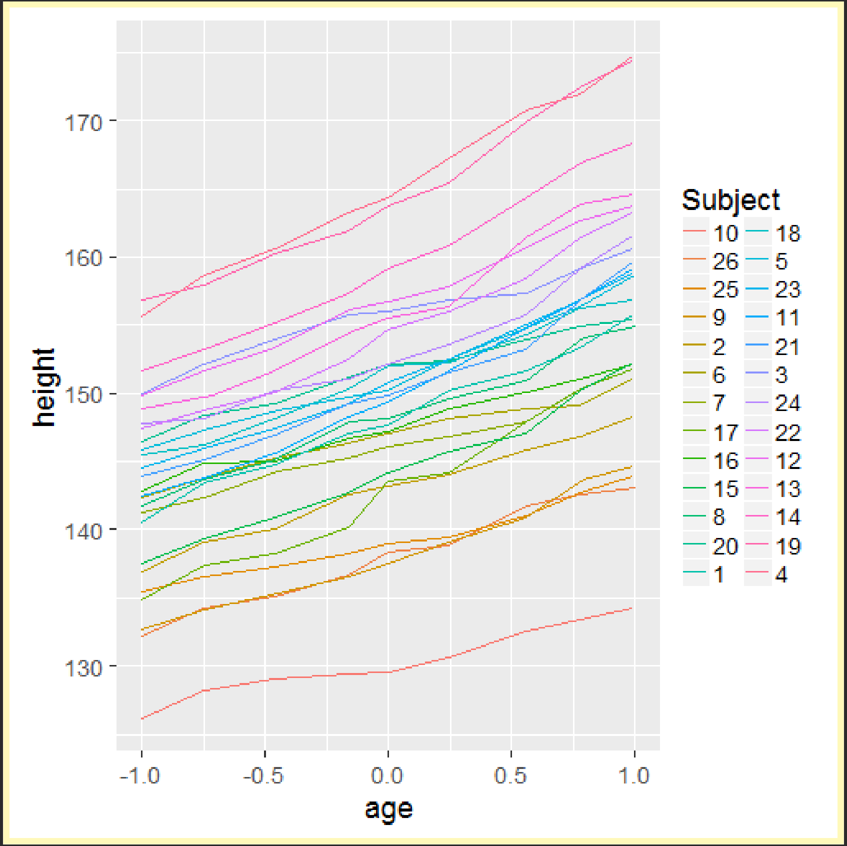


Aesthetic mapping: Groups

What about using `color` instead of `group`?

Aesthetic mapping: color instead of group

```
> ggplot(Oxboys, aes(age, height, color=Subject)) +
geom_line()
```



Viewing your ggplot

You can view the plot right away or save your plot as an object.

View plot right away

```
#create and display the plot
ggplot(Melanoma, aes(x=thickness, y=time)) + geom_point()
```

Save your plot as an object

```r
# create and save the plot (not displayed)
p <- ggplot(Melanoma, aes(x=thickness, y=time)) +
  geom_point()
```

```r
p # display plot
```

Add to a saved plot

```r
# create and save the plot (not displayed)
p <- ggplot(Melanoma, aes(x=thickness, y=time))
```

```r
p+ geom_point() # display plot
```

Faceting

Facet grid vs facet wrap

Faceting is the automatic layout of multiple plots and there are two types:

- facet_grid -- 2D grid
- facet_wrap -- 1D ribbon wrapped to 2D

Facet grid syntax

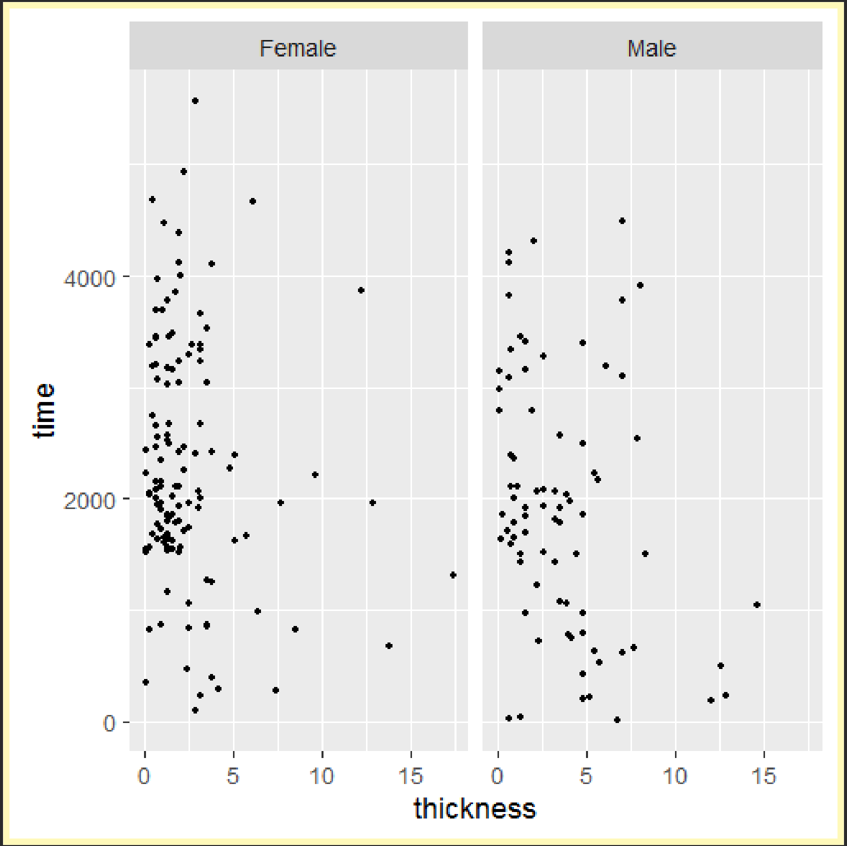
```
YOURPLOT + facet_grid(ROW_VARIABLE ~ COLUMN_VARIABLE)
```

If you want to leave row or column blank use a period:
`facet_grid(.~ COLUMN)`

Facet grid

An example of facet grid:

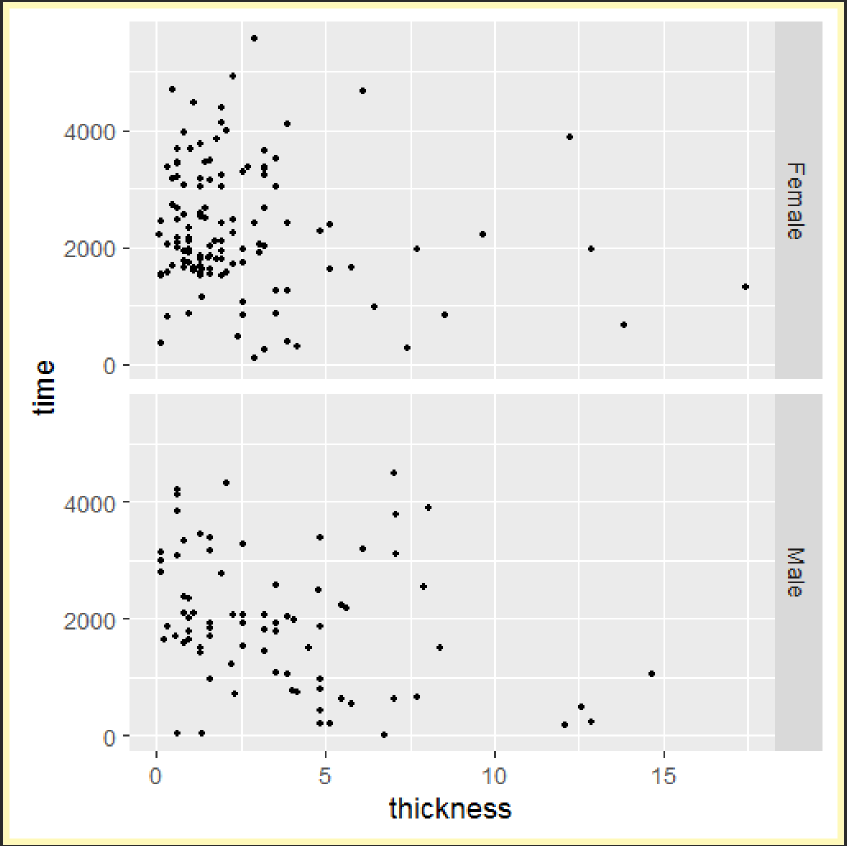
```
> ggplot(Melanoma, aes(x=thickness, y=time)) +
geom_point() + facet_grid(.~sex)
```



An example of facet grid:

An example of facet grid:

```
> ggplot(Melanoma, aes(x=thickness, y=time)) +
geom_point() + facet_grid(sex~.)
```



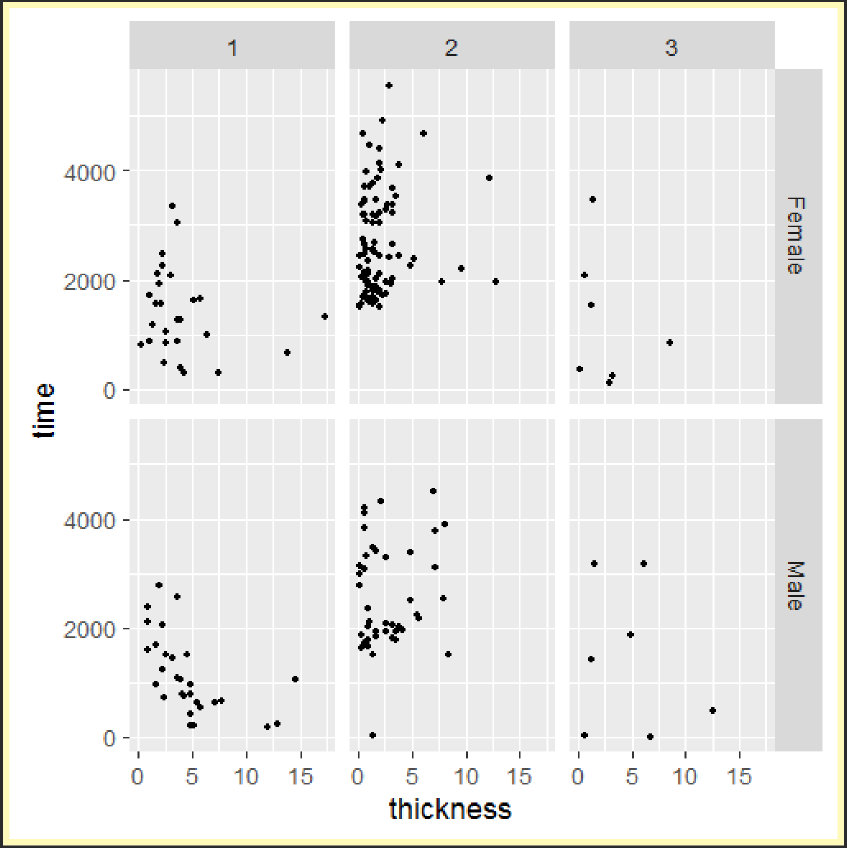
Facet grid

What about grid with sex as rows and status as columns?

What a grid with sex as rows and status as columns?

An example of facet grid:

```
> ggplot(Melanoma, aes(x=thickness, y=time)) +
geom_point() + facet_grid(sex~status)
```



Facet wrap

When you hear "facet_wrap" think "ribbon".

Facet wrap syntax

```
YOURPLOT + facet_wrap(~ WRAP_VARIABLE)
```

If you want to wrap by multiple variables using a `+`:
`facet_wrap(~ VAR1 + VAR2)`

Facet wrap

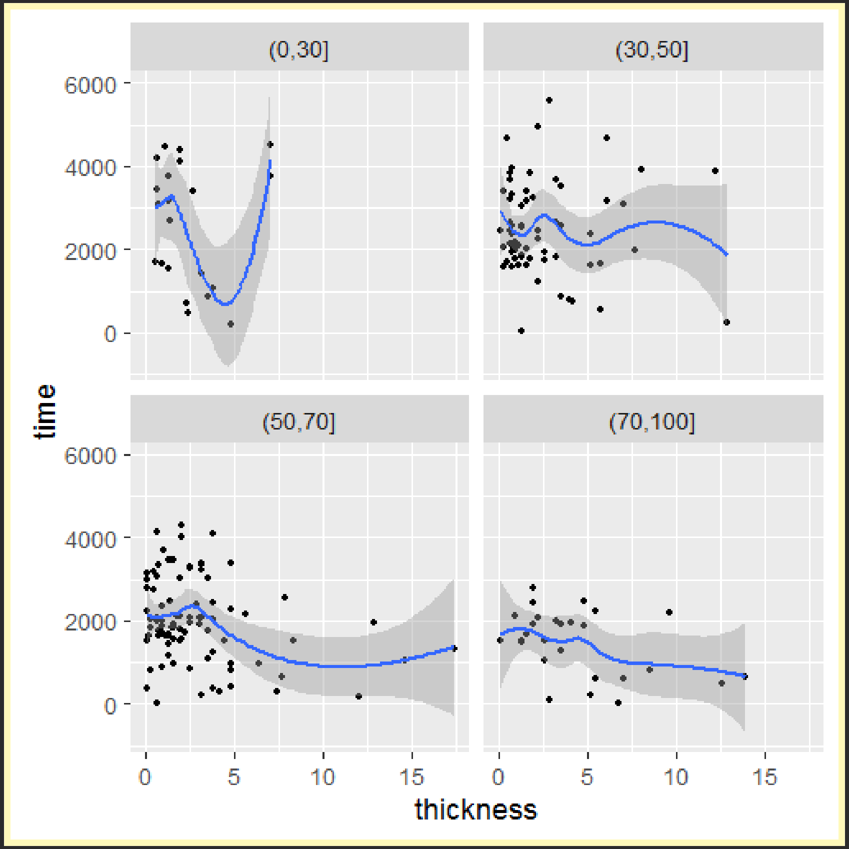
To demonstrate facet_wrap I'm going to create a new discrete variable based on age:

```
Melanoma$agecat<-cut(Melanoma$age, breaks=c(0, 30, 50, 70,
100))
head(Melanoma)
##   time status    sex age year thickness ulcer   agecat
## 1   10      3   Male  76 1972      6.76     1 (70,100]
## 2   30      3   Male  56 1968      0.65     0  (50,70]
## 3   35      2   Male  41 1977      1.34     0  (30,50]
## 4   99      3 Female  71 1968      2.90     0 (70,100]
## 5  185      1   Male  52 1965     12.08     1  (50,70]
## 6  204      1   Male  28 1971      4.84     1   (0,30]
```

Facet wrap

An example of facet wrap:

```
> ggplot(Melanoma, aes(x=thickness, y=time)) +
geom_point() + stat_smooth() + facet_wrap(~agecat)
```



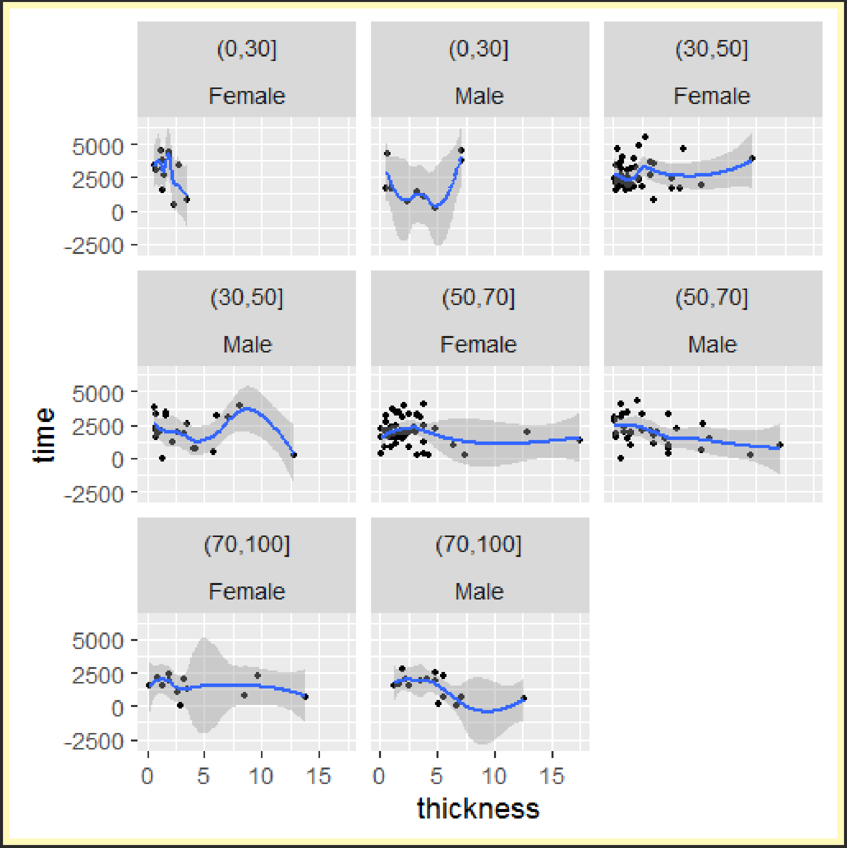
Facet wrap

What if you want to use `facet_wrap` on a ribbon based on two variables?

Facet wrap

An example of facet wrap with two variables:

```
> ggplot(Melanoma, aes(x=thickness, y=time)) +
geom_point() + stat_smooth() + facet_wrap(~agecat+sex)
```



exercise 1 (8-end)