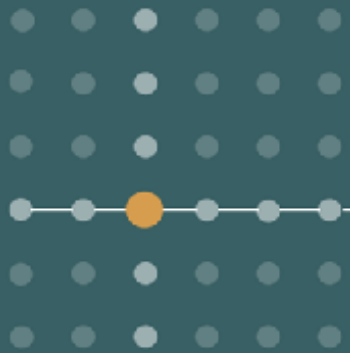


Analyx®

making sense of data.

Introduction to ggplot2

Adolfo Álvarez

Polski Akademicki Zlot Uzytkowników R

Outline

Why are we here?

- Graphs allow for a quick inspection of the data, e.g., results of a recently-completed simulation study
- For business: graphs appear in internal reports
- and they are crucial to show results to clients
- for Academia, integrate graphics within the analysis software is key for reproducibility
- and papers need plots too!

What do we want?

We are interested in producing high-quality graphs, often multipanel ones, with just few lines of code.

A. Jach, 2011

What is ggplot2

> *ggplot2 is an R package for producing statistical, or data, graphics. Provides beautiful, hassle-free plots, that take care of fiddly details like drawing legends. The plots can be built up iteratively and edited later.*
>
> *H. Wickham, "ggplot2"*

- Is unlike most other graphics packages because it has a deep underlying grammar
- This grammar is composed of a set of independent components that can be also composed in many different ways.
- You are not limited to a set of pre-specified graphics, but you can create new graphics that are precisely tailored for your problem.
- Instead of spending time making your graph look pretty, you can focus on creating a graph that best reveals the messages in your data.

What is ggplot2

> *ggplot2 is an R package for producing statistical, or data, graphics. Provides beautiful, hassle-free plots, that take care of fiddly details like drawing legends. The plots can be built up iteratively and edited later.*
>
> *H. Wickham, "ggplot2"*

- Is unlike most other graphics packages because it has a deep underlying grammar
- This grammar is composed of a set of independent components that can be also composed in many different ways.
- You are not limited to a set of pre-specified graphics, but you can create new graphics that are precisely tailored for your problem.
- Instead of spending time making your graph look pretty, you can focus on creating a graph that best reveals the messages in your data.

What is ggplot2

> *ggplot2 is an R package for producing statistical, or data, graphics. Provides beautiful, hassle-free plots, that take care of fiddly details like drawing legends. The plots can be built up iteratively and edited later.*
>
> *H. Wickham, "ggplot2"*

- Is unlike most other graphics packages because it has a deep underlying grammar
- This grammar is composed of a set of independent components that can be also composed in many different ways.
- You are not limited to a set of pre-specified graphics, but you can create new graphics that are precisely tailored for your problem.
- Instead of spending time making your graph look pretty, you can focus on creating a graph that best reveals the messages in your data.

What is ggplot2

> *ggplot2 is an R package for producing statistical, or data, graphics. Provides beautiful, hassle-free plots, that take care of fiddly details like drawing legends. The plots can be built up iteratively and edited later.*
>
> *H. Wickham, "ggplot2"*

- Is unlike most other graphics packages because it has a deep underlying grammar
- This grammar is composed of a set of independent components that can be also composed in many different ways.
- You are not limited to a set of pre-specified graphics, but you can create new graphics that are precisely tailored for your problem.
- Instead of spending time making your graph look pretty, you can focus on creating a graph that best reveals the messages in your data.

What is ggplot2

> *ggplot2 is an R package for producing statistical, or data, graphics. Provides beautiful, hassle-free plots, that take care of fiddly details like drawing legends. The plots can be built up iteratively and edited later.*
>
> *H. Wickham, "ggplot2"*

- Is unlike most other graphics packages because it has a deep underlying grammar
- This grammar is composed of a set of independent components that can be also composed in many different ways.
- You are not limited to a set of pre-specified graphics, but you can create new graphics that are precisely tailored for your problem.
- Instead of spending time making your graph look pretty, you can focus on creating a graph that best reveals the messages in your data.

How to install it

ggplot2 is an R package, so is installed in the usual way:

- `install.packages("ggplot2")`

And loading it with:

- `library(ggplot2)`

The ggplot2 grammar

- The **data** is what we want to visualize. It consists of variables, which are stored as columns in a data frame.
- **Geoms** are the geometric objects that are drawn to represent the data, such as bars, lines, and points.
- Aesthetic attributes, or **aesthetics**, are visual properties of geoms, such as x and y position, line color, point shapes, etc.
- Statistical transformations, **stats** for short, summarise data in many useful ways.
- There are **mappings** from data values to aesthetics.
- **Scales** control the mapping from the values in the data space to values in the aesthetic space.
- **Guides** show the viewer how to map the visual properties back to the data space.

The ggplot2 grammar

You have **data**, which is a set of numerical or categorical values. You have **geoms** to represent each observation. You have an **aesthetic**, such as y (vertical) position. And you have a **scale**, which defines the mapping from the data space (numeric values) to the aesthetic space (vertical position). A typical linear y-scale might map the value 0 to the baseline of the graph, 5 to the middle, and 10 to the top. A logarithmic y scale would place them differently.

Outline

Preparing the dataset

To introduce the qplot function we will use the diamonds dataset consisting of prices and quality information about 54,000 diamonds, and is included in the ggplot2 package. The data contains the four C´s of diamond quality, carat, cut, colour and clarity; and five physical measurements, depth, table, x, y and z.

```
## Loading required package:  methods
```

```
##   carat       cut color clarity depth table price    x    y
## 1  0.23     Ideal     E     SI2  61.5    55   326 3.95 3.98 2.4
## 2  0.21   Premium     E     SI1  59.8    61   326 3.89 3.84 2.3
## 3  0.23      Good     E     VS1  56.9    65   327 4.05 4.07 2.3
## 4  0.29   Premium     I     VS2  62.4    58   334 4.20 4.23 2.6
## 5  0.31      Good     J     SI2  63.3    58   335 4.34 4.35 2.7
## 6  0.24 Very Good     J    VVS2  62.8    57   336 3.94 3.96 2.4
```

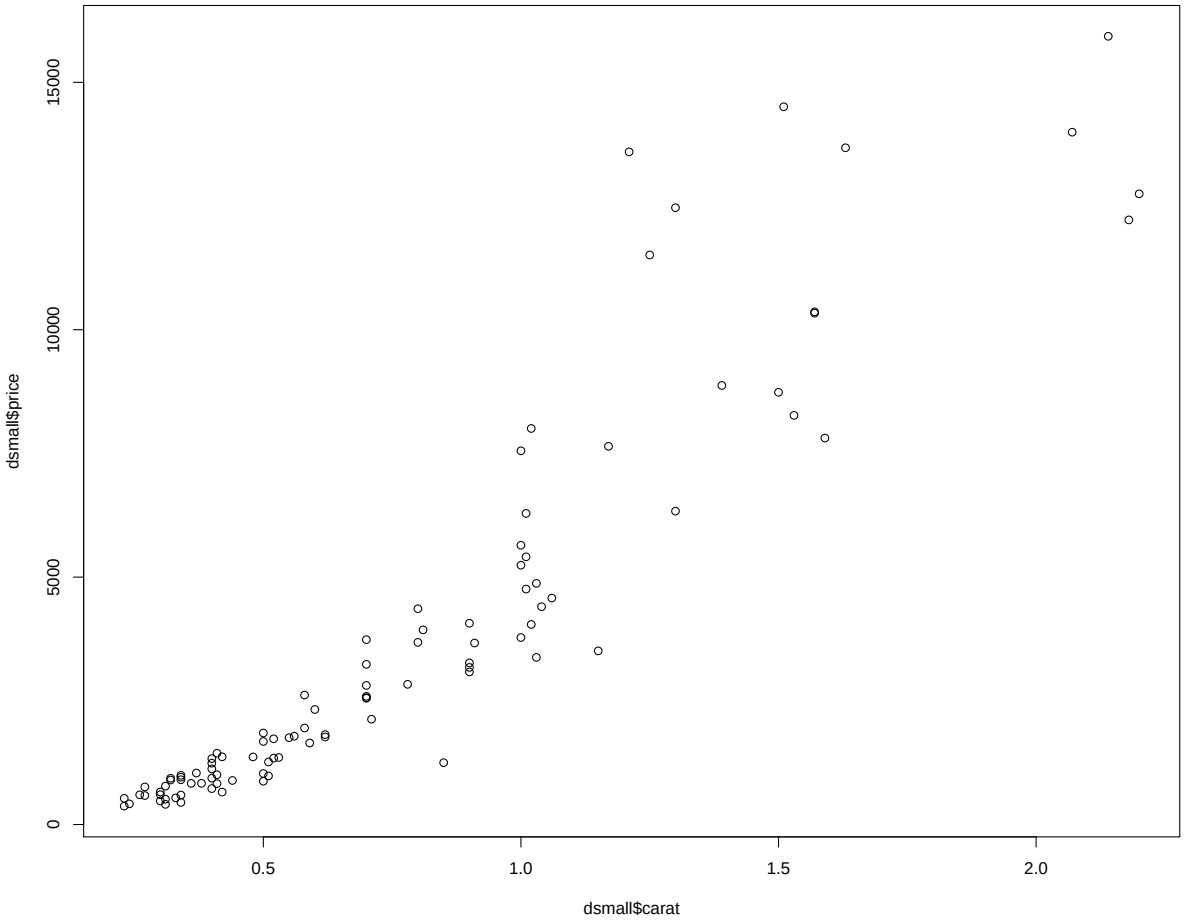
Preparing the dataset

We will use a small sample of the `diamonds` dataset:

```
dsmall <- diamonds[sample(nrow(diamonds), 100), ]
```

Scatterplots

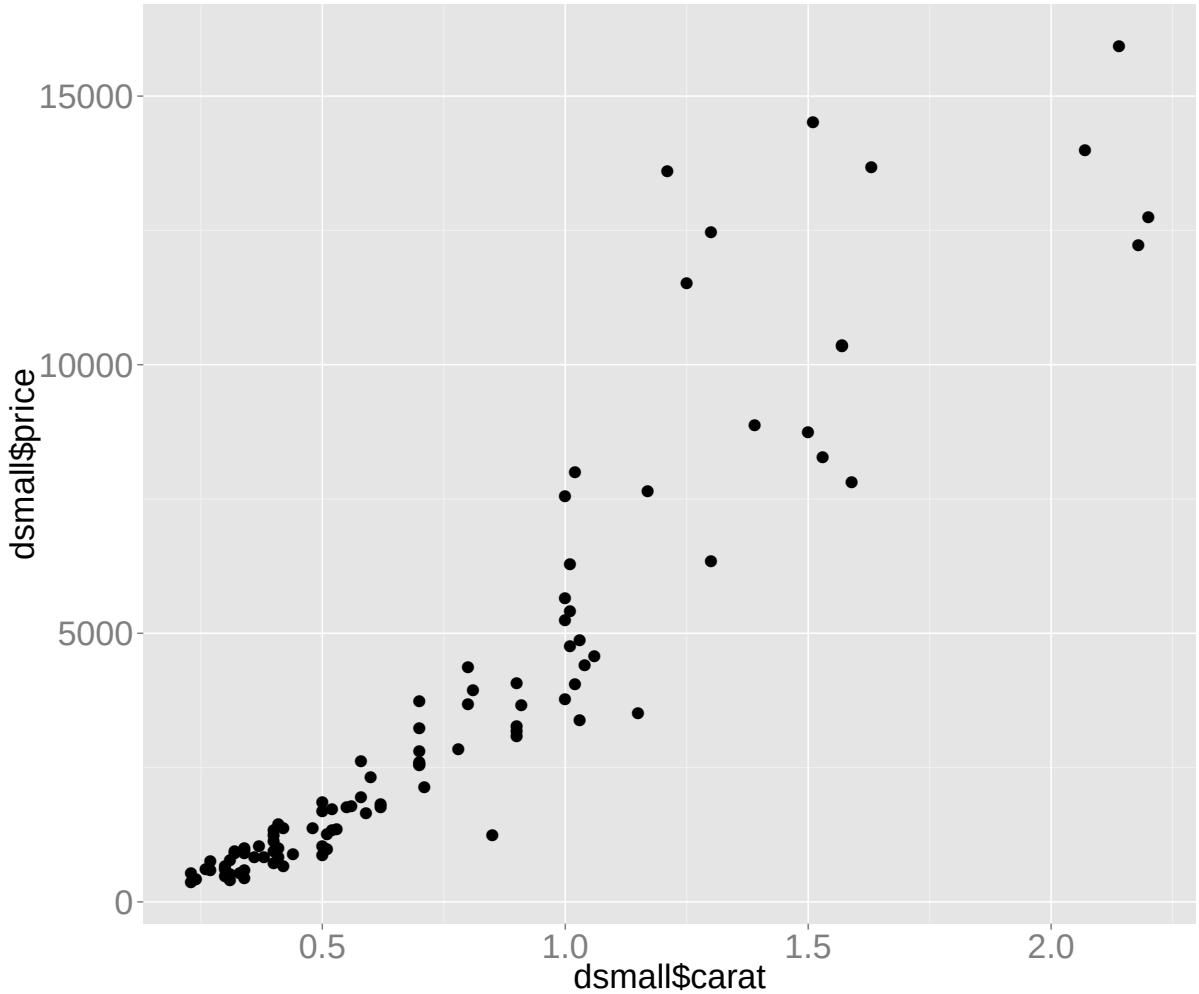
```
plot(dsmall$carat, dsmall$price)
```



A scatterplot from the "base" package.

Scatterplots

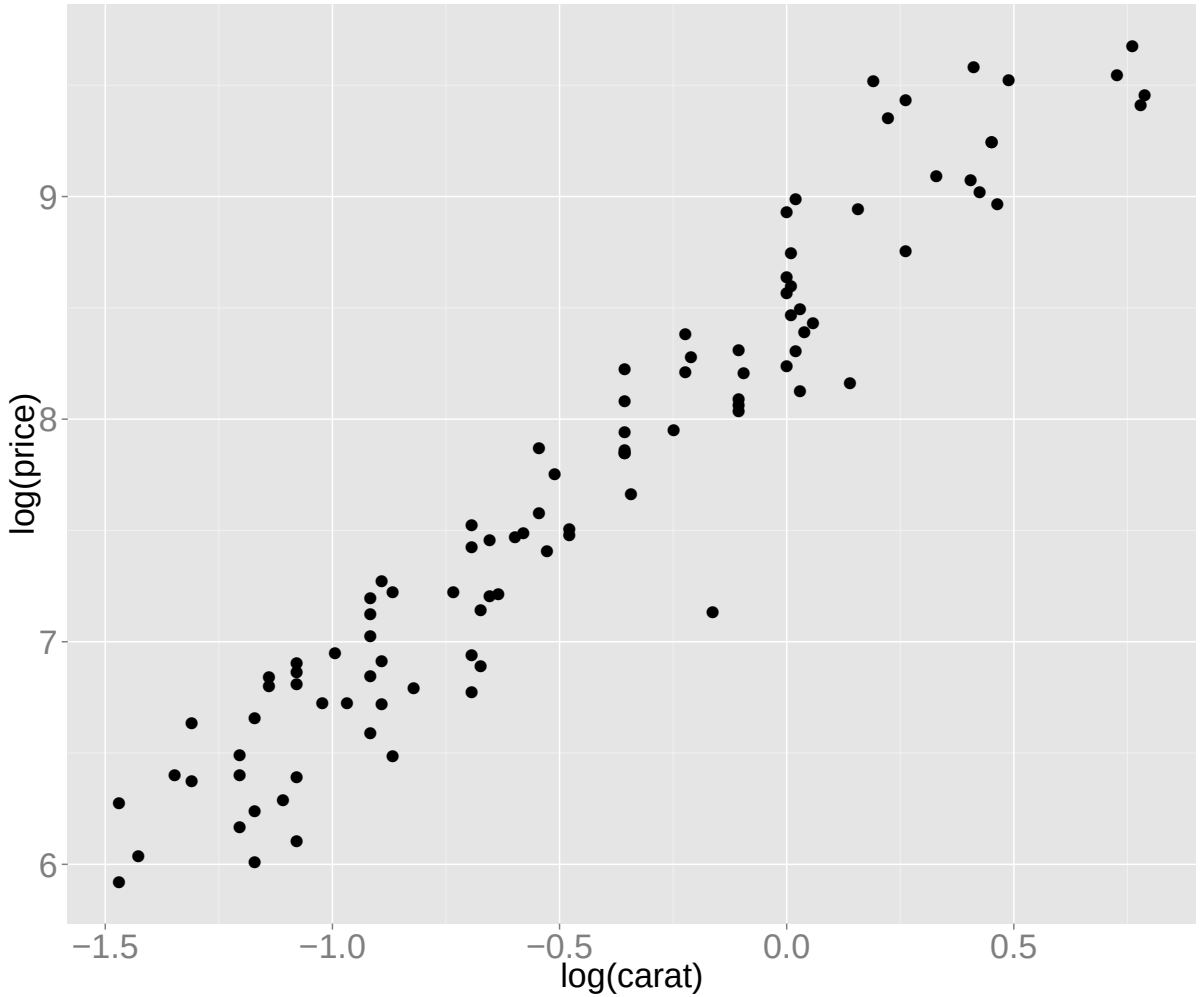
```
qplot(dsmall$carat, dsmall$price)
```



The qplot ("quick plot") function allows to use the power of ggplot using a similar grammar as the base command `plot`

Scatterplots

```
qplot(log(carat), log(price), data = dsmall)
```

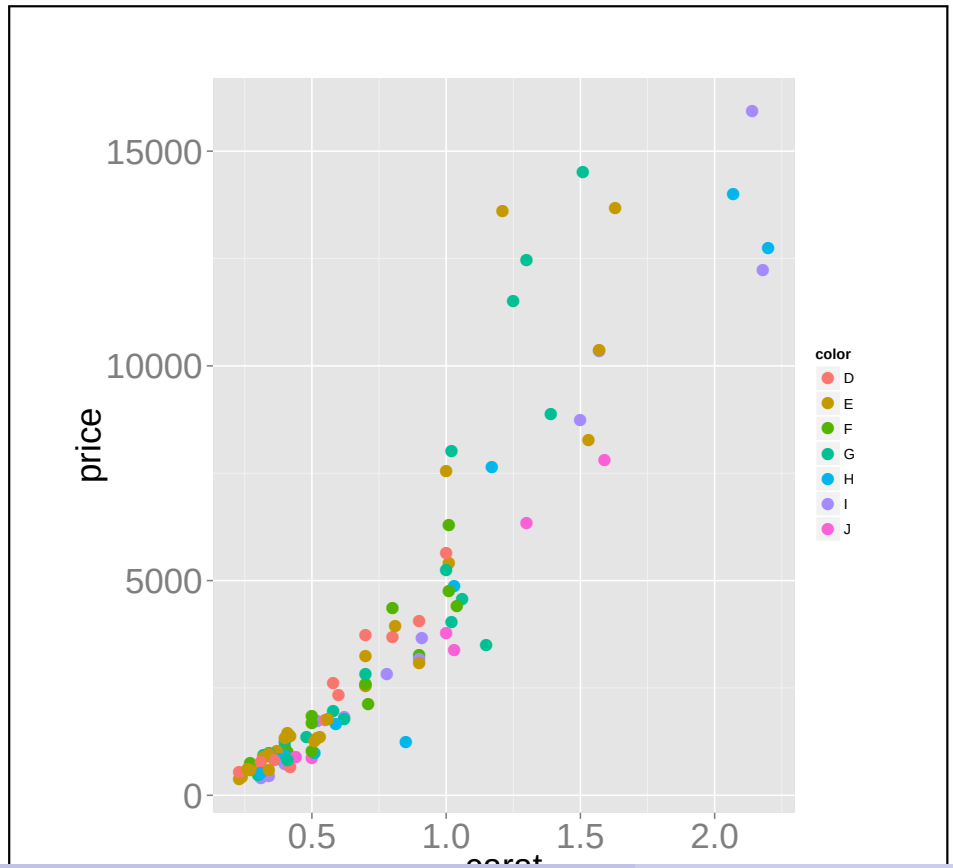


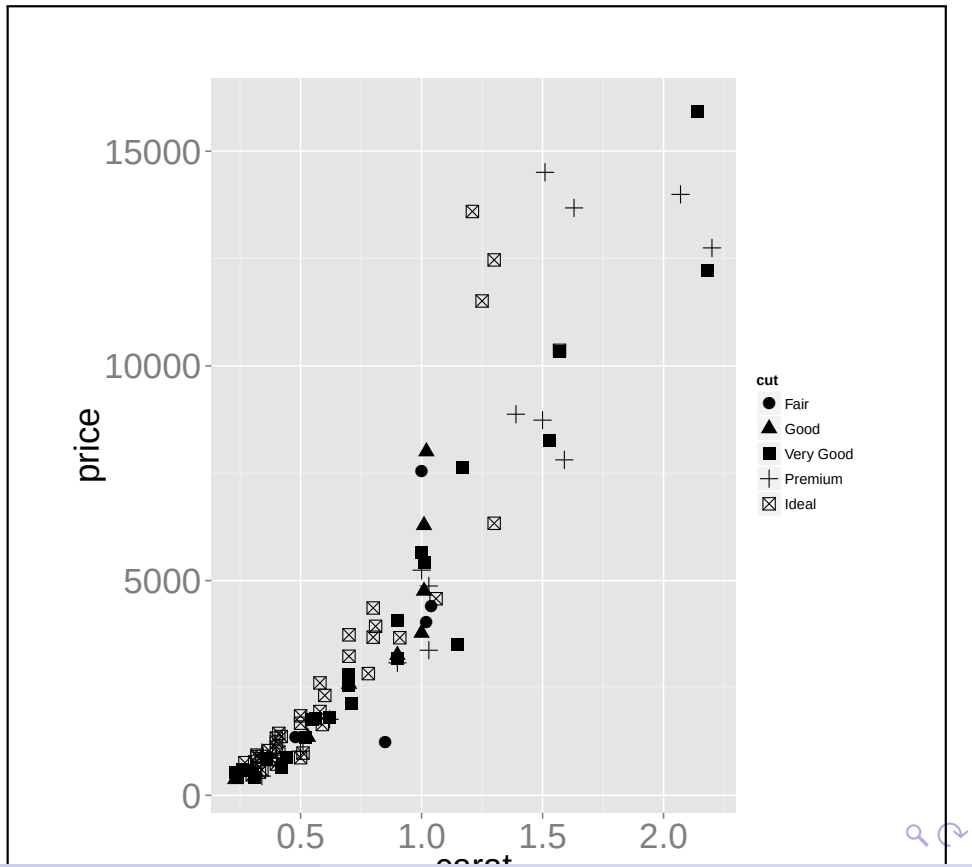
It also allows to define the data set where to extract the variables from, and the transformation of the data inside the function call.

Controlling shape and color

Is also possible to control the shape and color in the following way:

```
qplot(carat, price, data = dsmall, colour = color)
qplot(carat, price, data = dsmall, shape = cut)
```





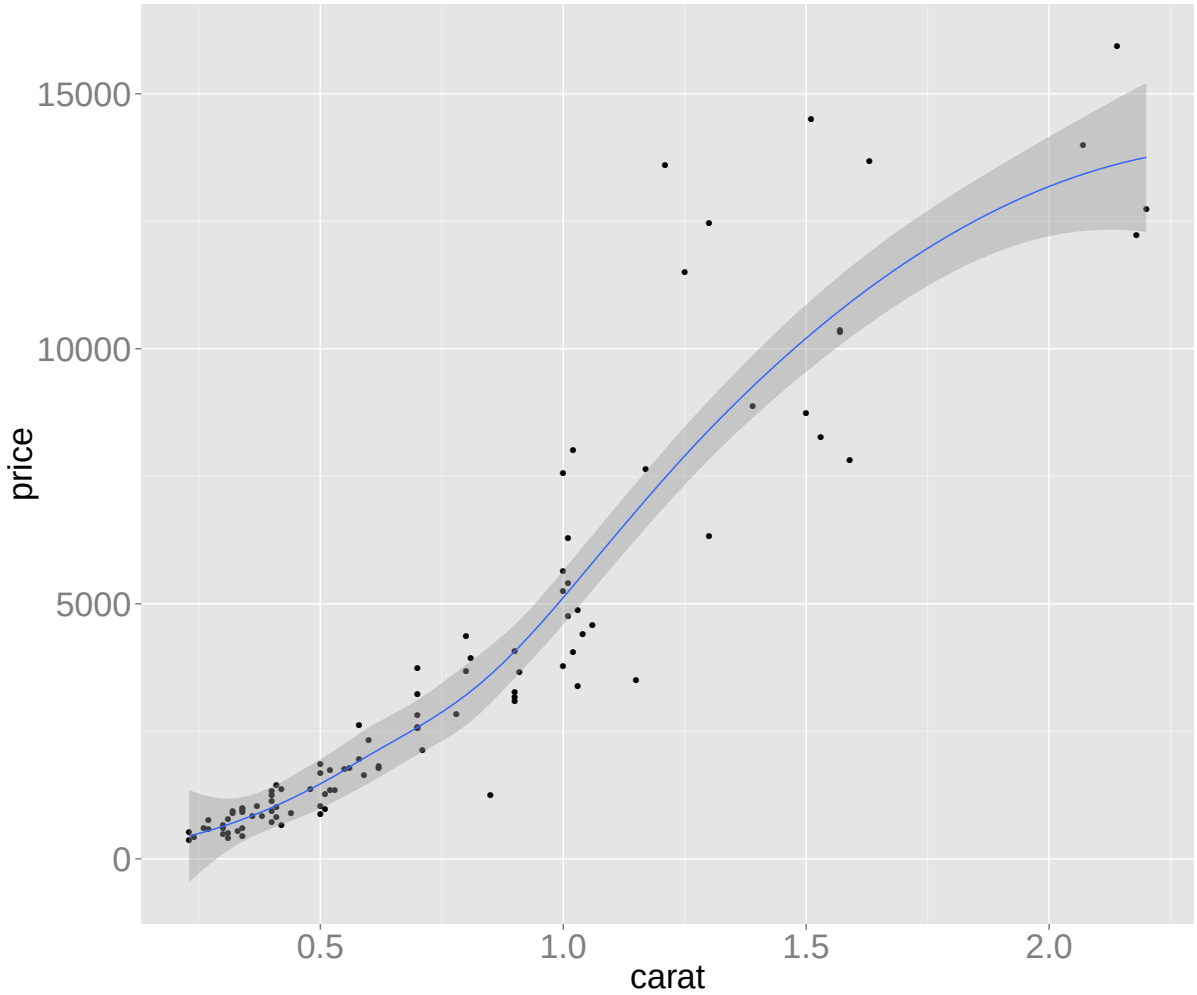
Geometric objects

The geometric objects are controlled in ggplot2 via the “geom = ...” parameter. The most typical cases are:

- "point" draws points to produce a scatterplot. (default)
- "smooth" fits a smoother to the data and displays the smooth and its standard error
- "boxplot" produces a box-and-whisker plot to summarise the distribution of a set of points.
- "path" and "line" draw lines between the data points. A line plot is constrained to produce lines that travel from left to right, while paths can go in any direction.
- "histogram" draws a histogram
- "freqpoly" produces a frequency polygon
- "density" creates a density plot
- For discrete variables, "bar" makes a bar chart.

Geometric objects

```
qplot(carat, price, data = dsmall,
geom = c("point", "smooth"))
```



Example of the "smooth" interpolation. The default option is "loess", for linear model use: `qplot(..., geom=c("point", "smooth"), method="lm").`

Geometric objects

Example of the use of different geometric objects:

- `qplot(color, price / carat, data = diamonds, geom = "boxplot")`
- `qplot(carat, data = diamonds, geom = "histogram")`
- `qplot(carat, data = diamonds, geom = "density")`
- `qplot(color, data = diamonds, geom = "bar")`
- `qplot(date, uempmed, data = economics, geom = "line")` date

Other options for qplot

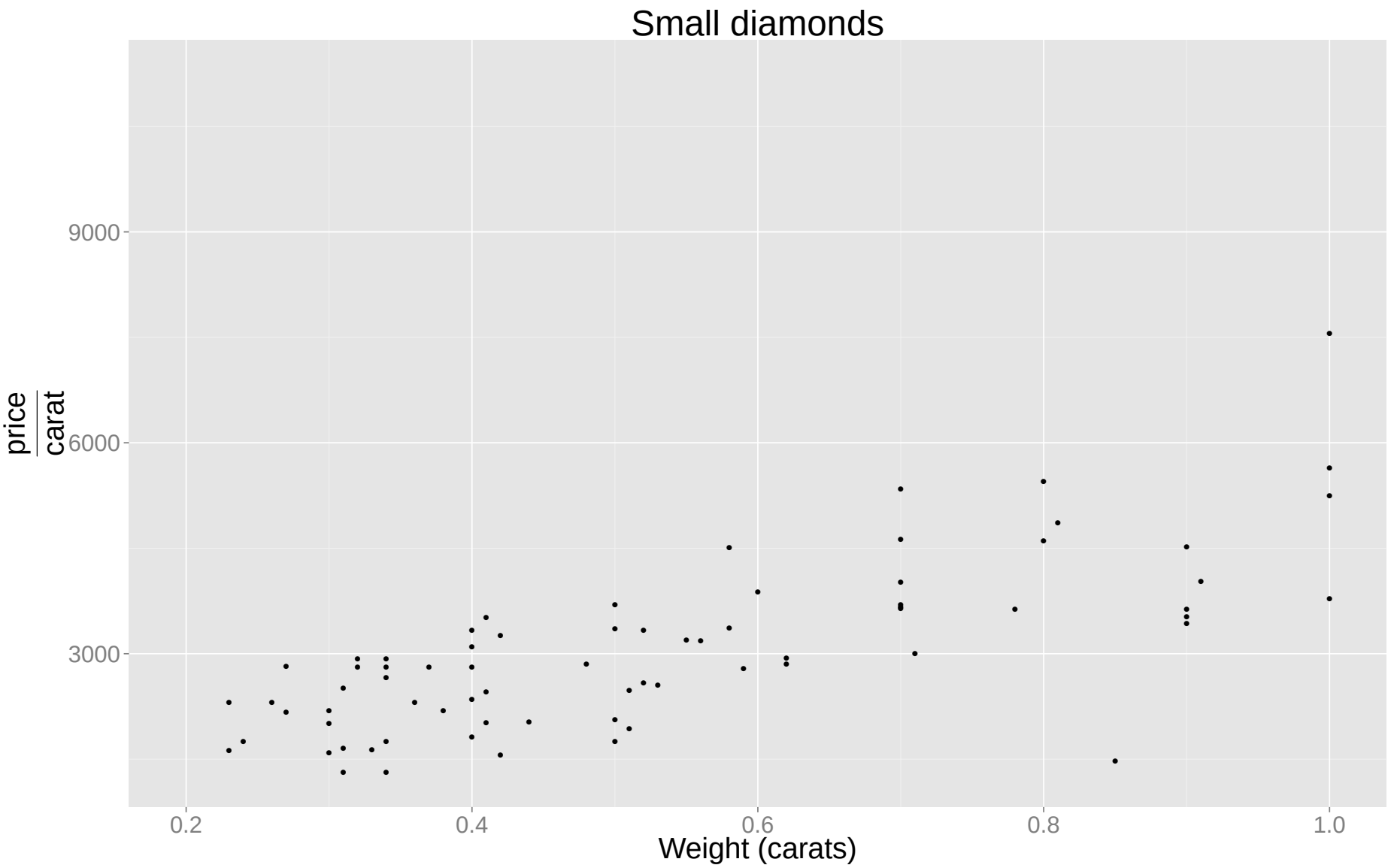
There exists equivalent to base plot function options to set plot parameters in qplot:

- **xlim, ylim**: set limits for the x- and y-axes, each a numeric vector of length two, e.g., xlim=c(0, 20) or ylim=c(-0.9, -0.5).
- **log**: a character vector indicating which (if any) axes should be logged. For example, log="x" will log the x-axis, log="xy" will log both.
- **main**: main title for the plot, centered in large text at the top of the plot. This can be a string (e.g., main="plot title") or an expression (e.g., main = expression(beta[1] == 1)). See ?plotmath for more examples of using mathematical formulae.
- **xlab, ylab**: labels for the x- and y-axes. As with the plot title, these can be character strings or mathematical expressions.

Example

```
qplot(
carat, price/carat, data = dsmall,
ylab = expression(frac(price,carat)),
xlab = "Weight (carats)",
main="Small diamonds",
xlim = c(.2,1)
)
```

Example


Small diamonds
price / carat
9000
6000
3000
0.2
0.4
0.6
0.8
1.0
Weight (carats)

Outline

Preparing the dataset

To introduce the ggplot function we will use the mpg dataset (among others) check `?mpg` for details...

```
str(mpg)

## 'data.frame': 234 obs. of  11 variables:
##  $ manufacturer: Factor w/ 15 levels "audi","chevrolet",..: 1 1
##  $ model       : Factor w/ 38 levels "4runner 4wd",..: 2 2 2 2
##  $ displ       : num  1.8 1.8 2 2 2.8 2.8 3.1 1.8 1.8 2 ...
##  $ year        : int  1999 1999 2008 2008 1999 1999 2008 1999 1
##  $ cyl         : int  4 4 4 4 6 6 6 4 4 4 ...
##  $ trans       : Factor w/ 10 levels "auto(av)","auto(l3)",..:
##  $ drv         : Factor w/ 3 levels "4","f","r": 2 2 2 2 2 2 2
##  $ cty         : int  18 21 20 21 16 18 18 18 16 20 ...
##  $ hwy         : int  29 29 31 30 26 26 27 26 25 28 ...
##  $ fl          : Factor w/ 5 levels "c","d","e","p",..: 4 4 4 4
##  $ class       : Factor w/ 7 levels "2seater","compact",..: 2 2
```

The ggplot function

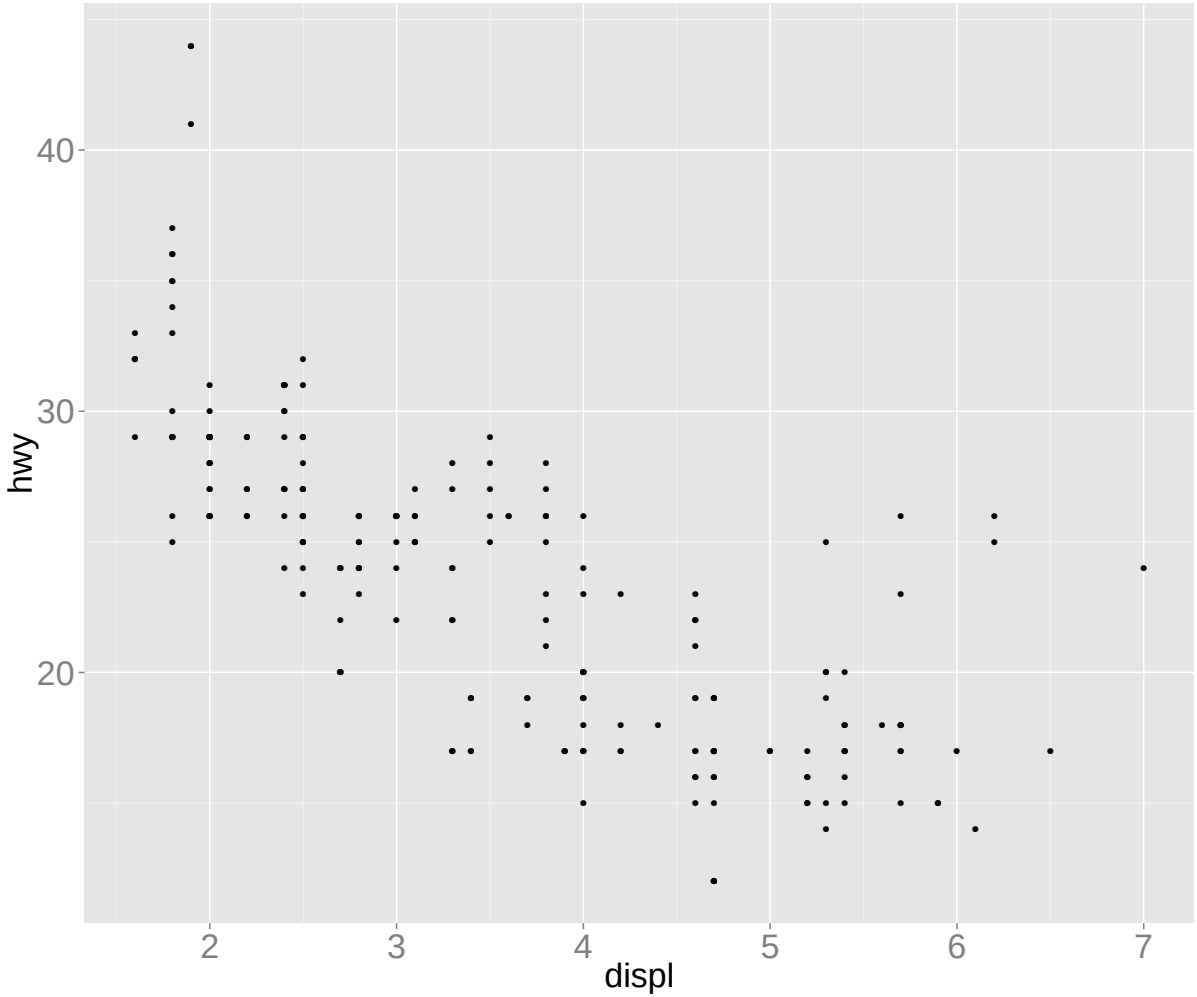
Despite qplot allows to get the power of ggplot in a "quick" way, sometimes we need to control more details of the plot. The ggplot function allows us to built our graphs in a step-by-step way.
All ggplot2 plots begin with the function ggplot(). ggplot() takes two primary arguments:

- **data** The data frame containing the data to be plotted
- **aes()**The aesthetic mappings to pass on to the plot elements

```
p <- ggplot(mpg, aes(displ, hwy))
```

The ggplot function

The plot is still not ready, we only told to ggplot the dataset and the aesthetic, but to produce a result, we need to add a geometry.

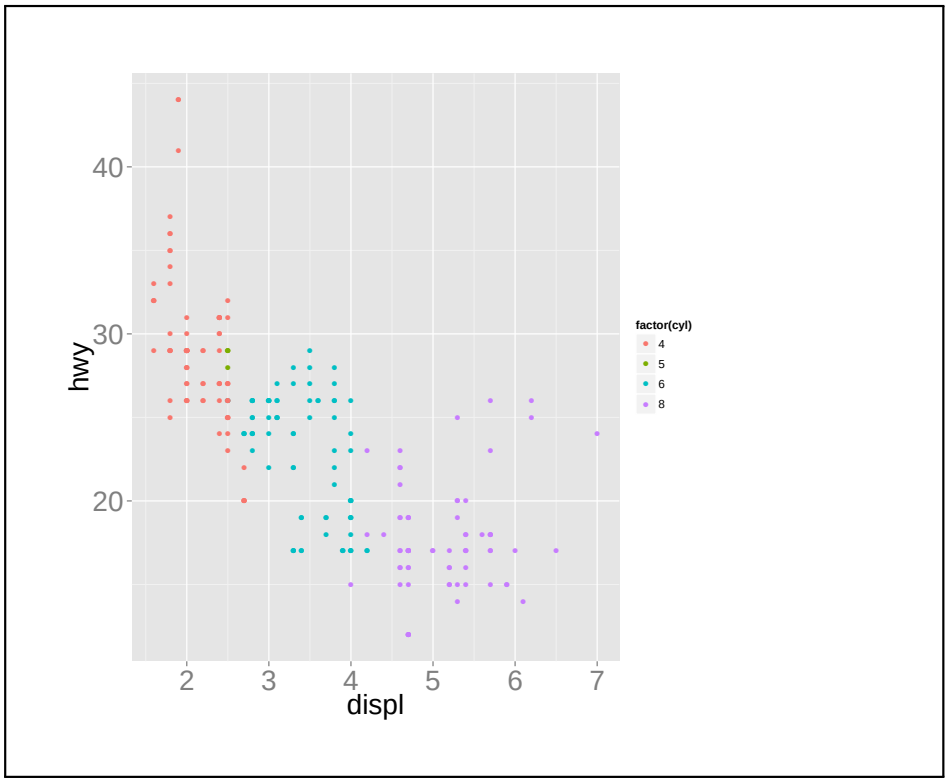


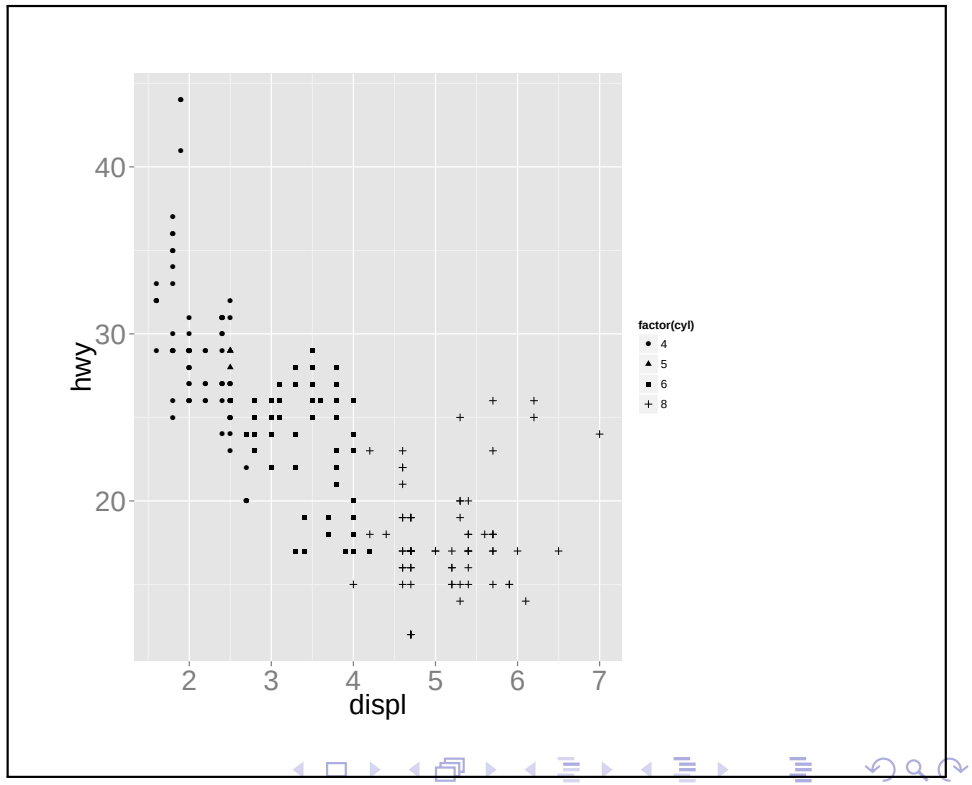

```
p + geom_point()
```

The ggplot function

We can also include different shapes or color for the points, depending on our variables, inside the `aes()` function

```
ggplot(mpg, aes(displ, hwy, color=factor(cyl)))+ geom_point()
ggplot(mpg, aes(displ, hwy))+ geom_point(aes(shape=factor(cyl)))
```





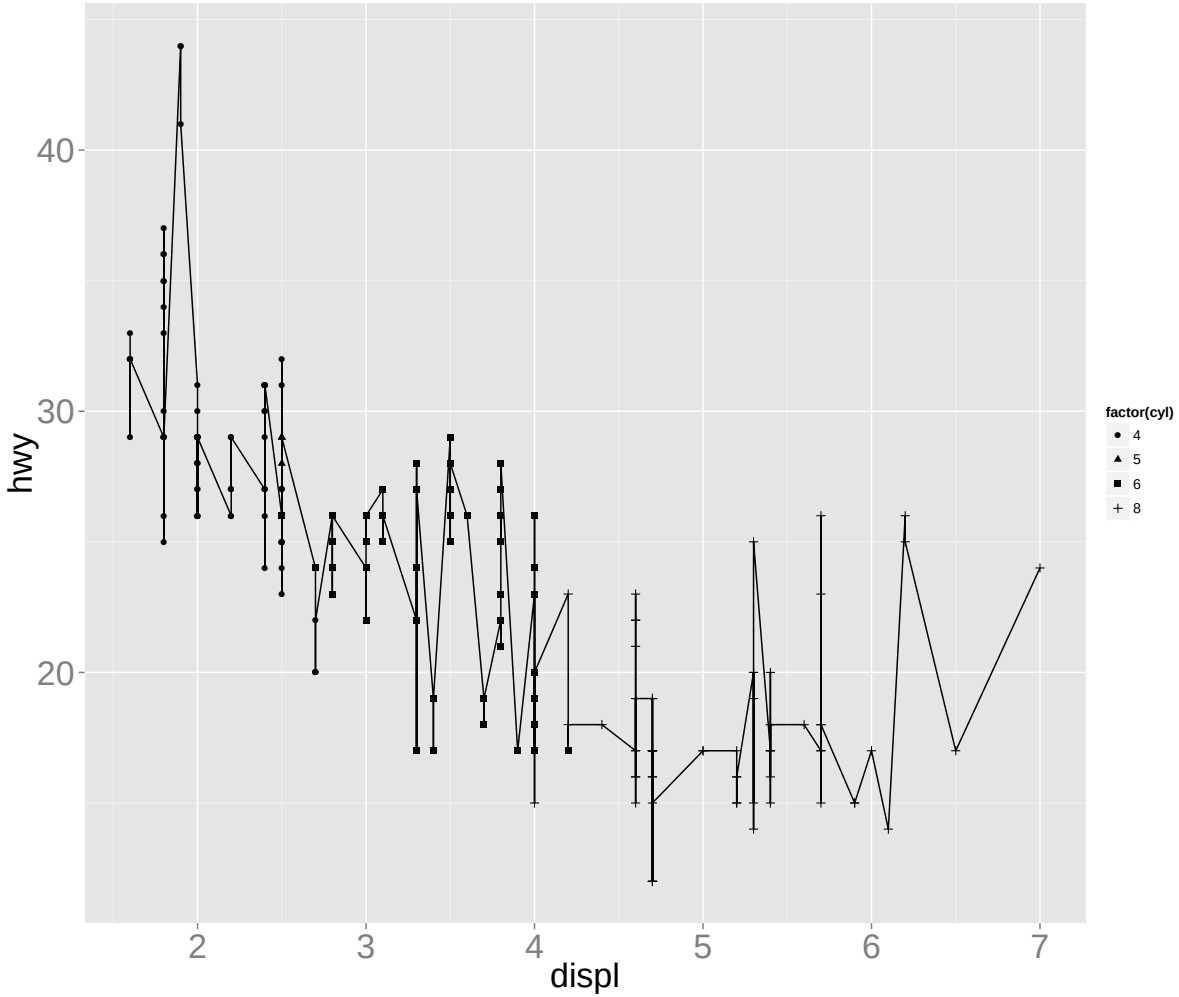
Geometric objects in ggplot function

A geometric object "x" can be assigned using the structure "`geom_x`". Remembering from the qplot function slides, the most common geometries available are :

- `geom_point()`
- `geom_boxplot()`
- `geom_histogram()`
- `geom_density()`
- `geom_bar()`
- `geom_line()`

The ggplot function

The geometric objects, and other layers can be added sequentially using the "+" connector




```
p + geom_point() +
geom_line()
```

Other useful layers

We can control many other parameters of our plot using layers as :

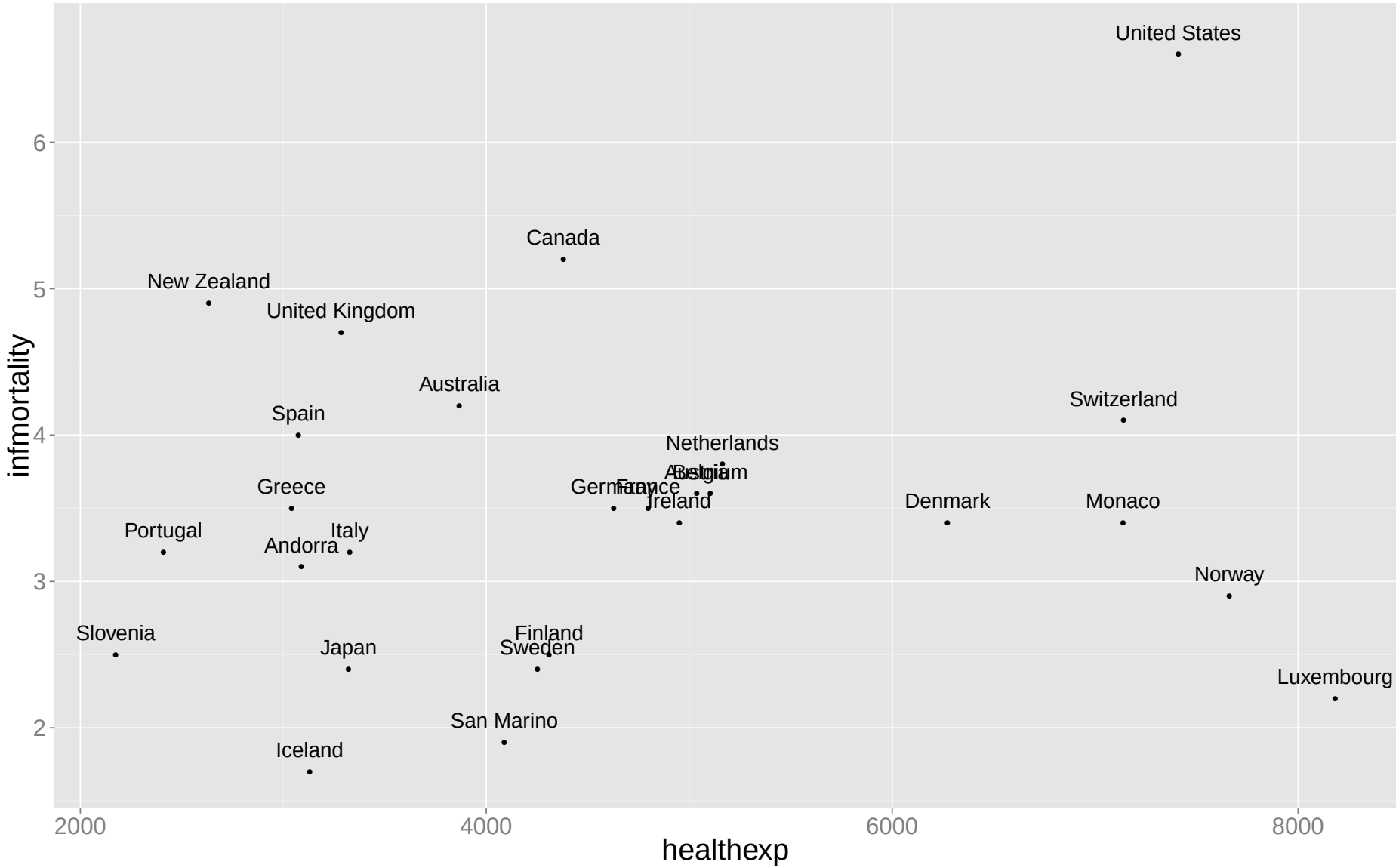
- `facet_grid()` To include more graphs, given a variable, in the same plot.
- `ggtitle("Main title")` To set the main title of the plot.
- `xlab("X label")`
- `ylab("Y label")`
- `theme()` To control more aspects of the legend, axis, grid and other from the plot. See help(theme) for details. Use `+ theme_bw()` for a white and black theme.
- `annotate("text", label="label", x=x, y=y)` To annotate any "label" in a given (x,y) position of the plot.

Outline

Example I

```
library(gcookbook) # For the data set
data= subset(countries, Year==2009 & healthexp>2000)
sp <- ggplot(data, aes(x=healthexp, y=infmortality))
sp <- sp + geom_point()
sp <- sp + geom_text(aes(y=infmortality+.1, label=Name),
size=6, vjust=0)
print(sp)
```

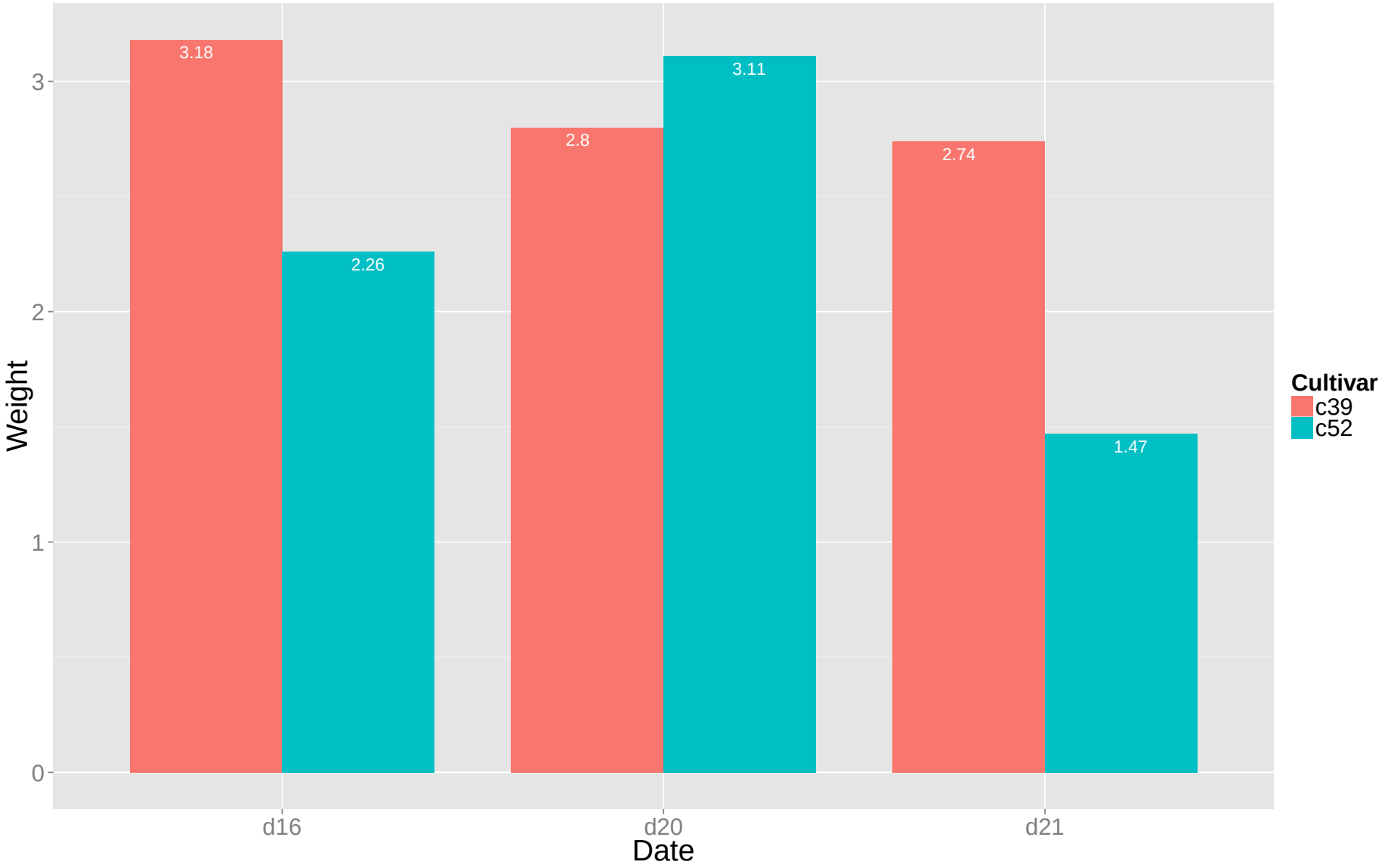
Example I


United States
Canada
New Zealand
United Kingdom
Australia
Switzerland
Spain
Netherlands
Greece
Denmark
Monaco
Portugal
Italy
Andorra
Norway
Slovenia
Finland
Japan
Sweden
Luxembourg
San Marino
Iceland
infmortality
healthexp
2000
4000
6000
8000
2
3
4
5
6

Example II

```
library(gcookbook)
p <- ggplot(cabbage_exp, aes(x=Date, y=Weight, fill=Cultivar))
p <- p + geom_bar(stat="identity", position="dodge", width=0.8)
p <- p + geom_text(aes(label=Weight), vjust=1.5, colour="white",
               position=position_dodge(.9), size=3)
print(p)
```

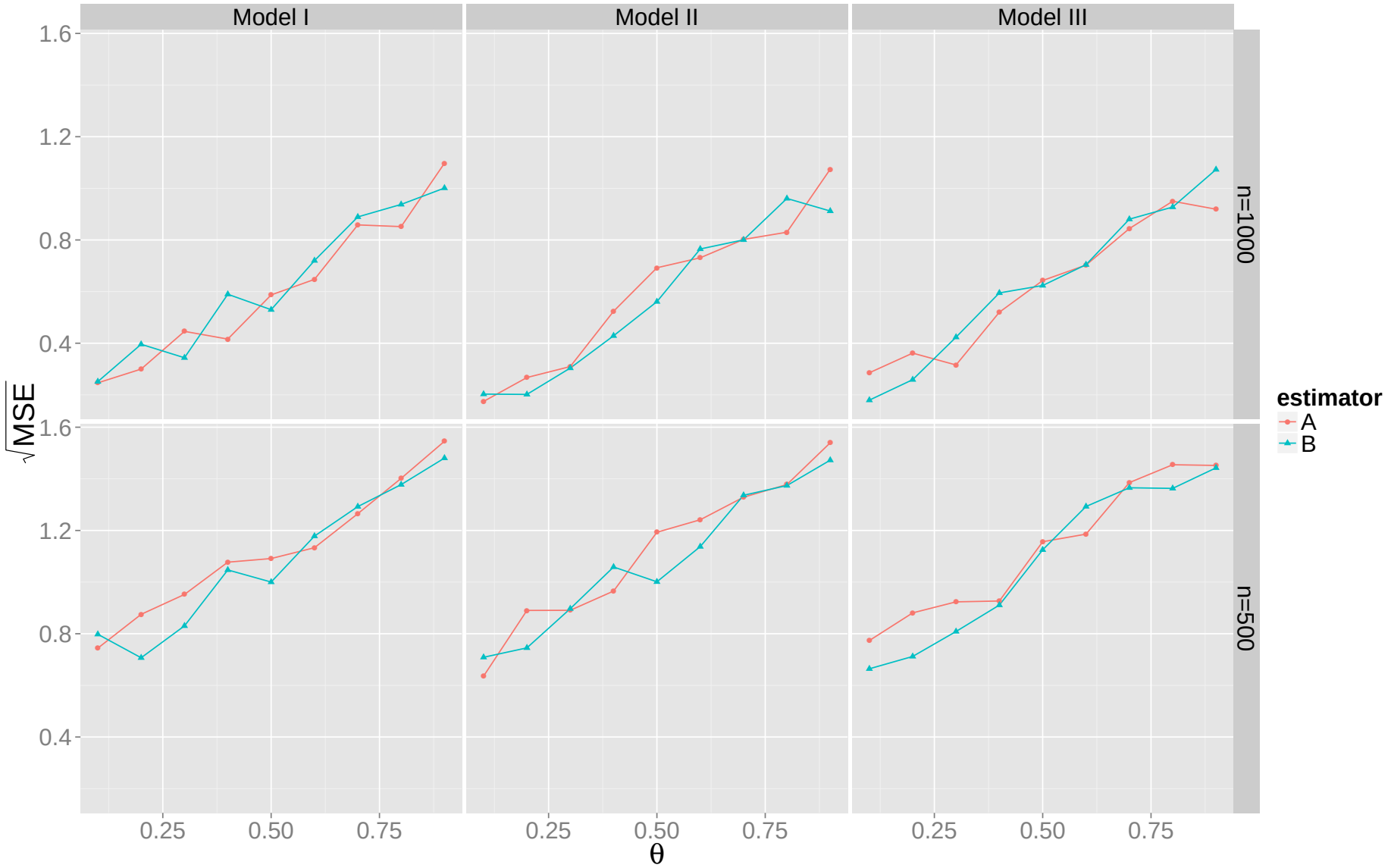
Example II


3.18
2.26
2.8
3.11
2.74
1.47
3
2
1
0
Weight
d16
d20
d21
Date
Cultivar
c39
c52

Example III

```
ex3=read.table("ex3.txt",head=T)
p<-ggplot(ex3, aes(x=theta,y=rmse,shape=estimator,col=estimator))
p<-p+facet_grid(samplesize~model)
p<-p +  geom_line() + geom_point()
p<-p + scale_x_continuous(expression(theta))
p<-p+scale_y_continuous(expression(sqrt(MSE)))
print(p)
```

Example III


Model I
Model II
Model III
n=1000
n=500
1.6
1.2
0.8
0.4
0.25
0.50
0.75
θ
√MSE
estimator
A
B

And after plotting...

- **print()**: Render it on screen. This happens automatically when running interactively, but inside a loop or function, you'll need to print()it yourself.
- **ggsave():** Render it to disk. Supports eps/ps, tex (pictex), pdf, jpeg, tiff, png, bmp, svg and wmf (windows only).
- **summary():** Brief description of the structure .
- **save():** Save a cached copy of it to disk.

ggplot2 extensions

ggplot2 can be extended with other R packages to plot different kind of data, use new themes, etc. Some of the extensions are:

- **xkcd**: Tries to imitate the style of the xkcd cartoon.
- **ggthemes** Other themes for ggplot, including classic ugly Excel, the economist magazine style, google docs, etc.
- **GGally** Includes network plots, and plot matrix.
- **ggmap** Visualization of spatial data.