Intoduction to data analysis with R

Mark Johnson

Macquarie University
Sydney, Australia

September 17, 2014

Outline

- Goals for today:
  - calculate summary statistics for data
  - construct several kinds of graphs
- Tools: *R statistical software*
  - many good statistical analysis packages
  - very good 2d graphics
  - complete programming language
  - Pandas, Matplotlib and Numpy/SciPy may offer comparable functionality in Python

Recommended readings

- Springer (a German publisher) is the main publisher of books about R (and publishes many statistics books)
- Our library lets us download many of them as pdfs.
- Recommended readings:
  - Dalgard, P. 2008 *Introductory statistics with R*: general introduction to statistics and R
  - Wickham, H. 2009 *ggplot2: Elegant graphics for data analysis*: describes the ggplot2 R graphics package
  - Allerhand, M. 2011 *A Tiny handbook of R*: introduces the R programming language
  - Hastie, Tibshirani and Friedman 2008 *The elements of statistical learning*: excellent textbook on statistics and machine learning
  - Wasserman, L. 2004 *All of statistics*: good general introduction to statistics
  - Wasserman, L. 2006 *All of non-parametric statistics*: follows above book

R as a calculator

```
1+1
```

```
## [1] 2
```

```
1+1;
```

```
## [1] 2
```

```
x <- 2
y <- 5
z <- x*y
print(z)
```

```
## [1] 10
```

```
z
```

```
## [1] 10
```

```
x <- c(1,2,3)
x
```

```
## [1] 1 2 3
```

```
x+1
```

```
## [1] 2 3 4
```

```
y <- c(4,5,6)
x*y
```

```
## [1]  4 10 18
```

```
sum(x)
```

```
## [1] 6
```

MACQUARIE UNIVERSITY

Getting help

```
x <- c(2,4,7)
mean(x)

## [1] 4.333
```

```
help(mean)
```

```
mean                    package:base                    R Documentation

Arithmetic Mean

Description:

     Generic function for the (trimmed) arithmetic mean.

Usage:

     mean(x, ...)

     ## Default S3 method:
     mean(x, trim = 0, na.rm = FALSE, ...)

Arguments:
```

Outline

MACQUARIE UNIVERSITY

CSV files

- Many programs read and write *CSV (comma-separated values)* files.
  - you can prepare these in a spreadsheet and export the file
- `mpg-data.csv` (available from iLearn) begins:

```
"manufacturer","model","displ","year","cyl","trans","drv","cty","hwy","fl","class"
"audi","a4",1.8,1999,4,"auto(l5)","f",18,29,"p","compact"
"audi","a4",1.8,1999,4,"manual(m5)","f",21,29,"p","compact"
"audi","a4",2,2008,4,"manual(m6)","f",20,31,"p","compact"
"audi","a4",2,2008,4,"auto(av)","f",21,30,"p","compact"
"audi","a4",2.8,1999,6,"auto(l5)","f",16,26,"p","compact"
"audi","a4",2.8,1999,6,"manual(m5)","f",18,26,"p","compact"
"audi","a4",3.1,2008,6,"auto(av)","f",18,27,"p","compact"
"audi","a4 quattro",1.8,1999,4,"manual(m5)","4",18,26,"p","compact"
"audi","a4 quattro",1.8,1999,4,"auto(l5)","4",16,25,"p","compact"
```

- The file begins with a header line
- Each line defines a single record

Reading CSV files into R

The `read.csv` command reads a CVS data file into a *data frame*:

```
d <- read.csv("mpg-data.csv", header=TRUE)
```

If you don't have this data file, it's available as part of the `ggplot2` package and can be downloaded as follows:

```
install.packages("ggplot2")
library(ggplot2)
d <- mpg
```

Changing the working directory

- Finding the current working directory:
  - open the CSV file in e.g., Excel
  - read the working directory from the file properties list
    e.g., `\\claudius.science.mq.edu.au\30267935\.Desktop")`
- Use the `setwd` command in R to the current working directory

```
setwd("\\\\claudius.science.mq.edu.au\\30267935\\.Desktop")
```

Viewing the head of a data frame

- After running these commands, `d` is a *data frame* containing the information in the CSV file.
- You can see the first few lines in `d` by running:

```
head(d)

##   manufacturer model displ year cyl      trans drv cty hwy
## 1         audi    a4   1.8 1999   4   auto(l5)   f  18  29
## 2         audi    a4   1.8 1999   4 manual(m5)   f  21  29
## 3         audi    a4   2.0 2008   4 manual(m6)   f  20  31
## 4         audi    a4   2.0 2008   4   auto(av)   f  21  30
## 5         audi    a4   2.8 1999   6   auto(l5)   f  16  26
## 6         audi    a4   2.8 1999   6 manual(m5)   f  18  26
##   fl   class
## 1  p compact
## 2  p compact
## 3  p compact
## 4  p compact
## 5  p compact
## 6  p compact
```

Viewing the size of a data frame

- A *data frame* can be viewed as a kind of table.
- `nrow` and `ncol` return the number of rows or columns in a data frame.

```
nrow(d)

## [1] 234
```

```
ncol(d)

## [1] 11
```

- `dim` returns the (row,column) dimensions of a data frame

```
dim(d)

## [1] 234  11
```

Summarising a data frame

```
summary(d)
```

```
##      manufacturer                    model         displ
##  dodge     :37    caravan 2wd          : 11   Min.   :1.60
##  toyota    :34    ram 1500 pickup 4wd: 10   1st Qu.:2.40
##  volkswagen:27    civic                :  9   Median :3.30
##  ford      :25    dakota pickup 4wd    :  9   Mean   :3.47
##  chevrolet :19    jetta                :  9   3rd Qu.:4.60
##  audi      :18    mustang              :  9   Max.   :7.00
##  (Other)   :74    (Other)              :177
##       year           cyl                trans    drv
##  Min.   :1999   Min.   :4.00   auto(l4)  :83   4:103
##  1st Qu.:1999   1st Qu.:4.00   manual(m5):58   f:106
##  Median :2004   Median :6.00   auto(l5)  :39   r: 25
##  Mean   :2004   Mean   :5.89   manual(m6):19
##  3rd Qu.:2008   3rd Qu.:8.00   auto(s6)  :16
##  Max.   :2008   Max.   :8.00   auto(l6)  : 6
##                                (Other)   :13
##       cty             hwy        fl               class
```

The internal structure of a data frame

- A data frame is *a list of vectors or factors*, one for each column
- Each of the vectors or factors in a data frame has the same length
- You can access a component of a data frame using the $ notation:

```
d$manufacturer
```

```
##   [1] audi       audi       audi       audi       audi
##   [6] audi       audi       audi       audi       audi
##  [11] audi       audi       audi       audi       audi
##  [16] audi       audi       audi       chevrolet  chevrolet
##  [21] chevrolet  chevrolet  chevrolet  chevrolet  chevrolet
##  [26] chevrolet  chevrolet  chevrolet  chevrolet  chevrolet
##  [31] chevrolet  chevrolet  chevrolet  chevrolet  chevrolet
##  [36] chevrolet  chevrolet  dodge      dodge      dodge
##  [41] dodge      dodge      dodge      dodge      dodge
##  [46] dodge      dodge      dodge      dodge      dodge
##  [51] dodge      dodge      dodge      dodge      dodge
##  [56] dodge      dodge      dodge      dodge      dodge
##  [61] dodge      dodge      dodge      dodge      dodge
##  [66] dodge      dodge      dodge      dodge      dodge
```

Vectors and factors in data frames

- *Numerical columns* in a data frame are represented by *vectors* (as in most programming languages)
- *Categorical values* in a data frame are represented by *factors*, which consist of an integer-valued vector and a list of *levels* that these integers index

```
levels(d$manufacturer)
```

```
##  [1] "audi"       "chevrolet"  "dodge"      "ford"
##  [5] "honda"      "hyundai"    "jeep"       "land rover"
##  [9] "lincoln"    "mercury"    "nissan"     "pontiac"
## [13] "subaru"     "toyota"     "volkswagen"
```

```
as.numeric(d$manufacturer)
```

```
##   [1]  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
##  [19]  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
##  [37]  2  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3
##  [55]  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3  3
##  [73]  3  3  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4  4
##  [91]  4  4  4  4  4  4  4  4  4  5  5  5  5  5  5  5  5  5
```

Summary statistics of vectors

```
summary(d$disp)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.60    2.40    3.30    3.47    4.60    7.00
```

```
length(d$disp)

## [1] 234
```

```
var(d$disp)

## [1] 1.669
```

```
mean(d$disp)

## [1] 3.472
```

```
sd(d$disp)

## [1] 1.292
```

```
median(d$disp)

## [1] 3.3
```

See also `quantile`

Subsetting rows in a data frame

```
dd <- subset(d, manufacturer=="volkswagen")
dim(dd)

## [1] 27 11

summary(dd)

##      manufacturer          model       displ
##  volkswagen:27    jetta       :9   Min.   :1.80
##  audi       : 0    passat      :7   1st Qu.:2.00
##  chevrolet  : 0    new beetle  :6   Median :2.00
##  dodge      : 0    gti         :5   Mean   :2.26
##  ford       : 0    4runner 4wd:0   3rd Qu.:2.50
##  honda      : 0    a4          :0   Max.   :3.60
##  (Other)    : 0    (Other)     :0
##       year            cyl                trans    drv
##  Min.   :1999   Min.   :4.00   manual(m5):11   4: 0
##  1st Qu.:1999   1st Qu.:4.00   auto(s6)  : 6   f:27
##  Median :1999   Median :4.00   auto(l4)  : 5   r: 0
##  Mean   :2003   Mean   :4.59   manual(m6): 3
```

Subsetting columns in a data frame

```
ddd <- subset(d, select=c("model", "year", "displ"))
dim(ddd)

## [1] 234   3

summary(ddd)

##                  model          year          displ
##  caravan 2wd         : 11   Min.   :1999   Min.   :1.60
##  ram 1500 pickup 4wd: 10   1st Qu.:1999   1st Qu.:2.40
##  civic               :  9   Median :2004   Median :3.30
##  dakota pickup 4wd   :  9   Mean   :2004   Mean   :3.47
##  jetta               :  9   3rd Qu.:2008   3rd Qu.:4.60
##  mustang             :  9   Max.   :2008   Max.   :7.00
##  (Other)             :177
```

Removing outliers by subsetting

- Here we remove all the data outside of 2 standard deviation from the mean (probably not a good idea here!)

```
cld <- subset(d,
              displ>mean(d$displ)-2*sd(d$displ) &
              displ<mean(d$displ)+2*sd(d$displ))
dim(cld)

## [1] 229  11

summary(cld)

##      manufacturer                    model         displ
##  dodge     :37    caravan 2wd          : 11   Min.   :1.60
##  toyota    :34    ram 1500 pickup 4wd: 10   1st Qu.:2.40
##  volkswagen:27    civic                :  9   Median :3.30
##  ford      :25    dakota pickup 4wd    :  9   Mean   :3.41
##  audi      :18    jetta                :  9   3rd Qu.:4.60
##  chevrolet :15    mustang              :  9   Max.   :6.00
##  (Other)   :73    (Other)              :172
##       year            cyl                trans       drv
```

Transforming a data frame

```
dd <- transform(d, log.displ=log(displ))
head(dd)

##   manufacturer model displ year cyl      trans drv cty hwy
## 1         audi    a4   1.8 1999   4   auto(l5)   f  18  29
## 2         audi    a4   1.8 1999   4 manual(m5)   f  21  29
## 3         audi    a4   2.0 2008   4 manual(m6)   f  20  31
## 4         audi    a4   2.0 2008   4   auto(av)   f  21  30
## 5         audi    a4   2.8 1999   6   auto(l5)   f  16  26
## 6         audi    a4   2.8 1999   6 manual(m5)   f  18  26
##   fl   class log.displ
## 1  p compact    0.5878
## 2  p compact    0.5878
## 3  p compact    0.6931
## 4  p compact    0.6931
## 5  p compact    1.0296
## 6  p compact    1.0296
```

Outline

Tables for one categorical variable

```
table(d$manufacturer)

##
##       audi  chevrolet      dodge       ford      honda
##         18         19         37         25          9
##    hyundai       jeep land rover    lincoln    mercury
##         14          8          4          3          4
##     nissan    pontiac     subaru     toyota volkswagen
##         13          5         14         34         27
```

Tables for two categorical variables

```
table(d$manufacturer, d$class)
```

```
##
##                2seater compact midsize minivan pickup
##   audi               0      15       3       0      0
##   chevrolet          5       0       5       0      0
##   dodge              0       0       0      11     19
##   ford               0       0       0       0      7
##   honda              0       0       0       0      0
##   hyundai            0       0       7       0      0
##   jeep               0       0       0       0      0
##   land rover         0       0       0       0      0
##   lincoln            0       0       0       0      0
##   mercury            0       0       0       0      0
##   nissan             0       2       7       0      0
##   pontiac            0       0       5       0      0
##   subaru             0       4       0       0      0
##   toyota             0      12       7       0      7
##   volkswagen         0      14       7       0      0
##
```

Pretty printing tables for LaTeX(1)

- The `xtable` R library package formats a table in LaTeXformat
- You'll need to download it before using it
- This only has to be done once.

```
install.packages("xtable")
```

- Once downloaded, packages are loaded into R with the `library` function.

```
library(xtable)
```

Pretty printing tables for LaTeX(2)

```
t <- table(d$manufacturer, d$class)
xtable(t)

## % latex table generated in R 3.0.2 by xtable 1.7-3 package
## % Wed Sep 17 08:18:30 2014
## \begin{table}[ht]
## \centering
## \begin{tabular}{rrrrrrrr}
##   \hline
##  & 2seater & compact & midsize & minivan & pickup & subcompact & suv \\
##   \hline
## audi &   0 &  15 &   3 &   0 &   0 &   0 &   0 \\
##   chevrolet &   5 &   0 &   5 &   0 &   0 &   0 &   9 \\
##   dodge &   0 &   0 &   0 &  11 &  19 &   0 &   7 \\
##   ford &   0 &   0 &   0 &   0 &   7 &   9 &   9 \\
##   honda &   0 &   0 &   0 &   0 &   0 &   9 &   0 \\
##   hyundai &   0 &   0 &   7 &   0 &   0 &   7 &   0 \\
##   jeep &   0 &   0 &   0 &   0 &   0 &   0 &   8 \\
##   land rover &   0 &   0 &   0 &   0 &   0 &   0 &   4 \\
##   lincoln &   0 &   0 &   0 &   0 &   0 &   0 &   3 \\
##   mercury &   0 &   0 &   0 &   0 &   0 &   0 &   4 \\
##   nissan &   0 &   2 &   7 &   0 &   0 &   0 &   4 \\
##   pontiac &   0 &   0 &   5 &   0 &   0 &   0 &   0 \\
##   subaru &   0 &   4 &   0 &   0 &   0 &   4 &   6 \\
```

Pretty printing tables for LaTeX(3)

```
t <- table(d$manufacturer, d$class)
xtable(t)
```

	2seater	compact	midsize	minivan	pickup	subcompact	suv
audi	0	15	3	0	0	0	0
chevrolet	5	0	5	0	0	0	9
dodge	0	0	0	11	19	0	7
ford	0	0	0	0	7	9	9
honda	0	0	0	0	0	9	0
hyundai	0	0	7	0	0	7	0
jeep	0	0	0	0	0	0	8
land rover	0	0	0	0	0	0	4
lincoln	0	0	0	0	0	0	3
mercury	0	0	0	0	0	0	4
nissan	0	2	7	0	0	0	4
pontiac	0	0	5	0	0	0	0
subaru	0	4	0	0	0	4	6
toyota	0	12	7	0	7	0	8
volkswagen	0	14	7	0	0	6	0

Relative proportion tables (1)

- The sum of the relative frequencies is 1

```
t <- table(d$manufacturer, d$class)
xtable(t/sum(t))
```

	2seater	compact	midsize	minivan	pickup	subcompact	suv
audi	0.00	0.06	0.01	0.00	0.00	0.00	0.00
chevrolet	0.02	0.00	0.02	0.00	0.00	0.00	0.04
dodge	0.00	0.00	0.00	0.05	0.08	0.00	0.03
ford	0.00	0.00	0.00	0.00	0.03	0.04	0.04
honda	0.00	0.00	0.00	0.00	0.00	0.04	0.00
hyundai	0.00	0.00	0.03	0.00	0.00	0.03	0.00
jeep	0.00	0.00	0.00	0.00	0.00	0.00	0.03
land rover	0.00	0.00	0.00	0.00	0.00	0.00	0.02
lincoln	0.00	0.00	0.00	0.00	0.00	0.00	0.01
mercury	0.00	0.00	0.00	0.00	0.00	0.00	0.02
nissan	0.00	0.01	0.03	0.00	0.00	0.00	0.02
pontiac	0.00	0.00	0.02	0.00	0.00	0.00	0.00
subaru	0.00	0.02	0.00	0.00	0.00	0.02	0.03
toyota	0.00	0.05	0.03	0.00	0.03	0.00	0.03
volkswagen	0.00	0.06	0.03	0.00	0.00	0.03	0.00

Relative proportion tables (2)

- Sometimes we want the proportions along a row to sum to 1

```
t <- table(d$manufacturer, d$class)
xtable(prop.table(t, 1))
```

	2seater	compact	midsize	minivan	pickup	subcompact	suv
audi	0.00	0.83	0.17	0.00	0.00	0.00	0.00
chevrolet	0.26	0.00	0.26	0.00	0.00	0.00	0.47
dodge	0.00	0.00	0.00	0.30	0.51	0.00	0.19
ford	0.00	0.00	0.00	0.00	0.28	0.36	0.36
honda	0.00	0.00	0.00	0.00	0.00	1.00	0.00
hyundai	0.00	0.00	0.50	0.00	0.00	0.50	0.00
jeep	0.00	0.00	0.00	0.00	0.00	0.00	1.00
land rover	0.00	0.00	0.00	0.00	0.00	0.00	1.00
lincoln	0.00	0.00	0.00	0.00	0.00	0.00	1.00
mercury	0.00	0.00	0.00	0.00	0.00	0.00	1.00
nissan	0.00	0.15	0.54	0.00	0.00	0.00	0.31
pontiac	0.00	0.00	1.00	0.00	0.00	0.00	0.00
subaru	0.00	0.29	0.00	0.00	0.00	0.29	0.43
toyota	0.00	0.35	0.21	0.00	0.21	0.00	0.24
volkswagen	0.00	0.52	0.26	0.00	0.00	0.22	0.00

Finding the maximum value

```
i <- which.max(d$hwy)
i

## [1] 213
```

```
xtable(d[i,])
```

	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl	cla
213	volkswagen	jetta	1.90	1999	4	manual(m5)	f	33	44	d	cor

```
xtable(subset(d, hwy==max(hwy)))
```

	manufacturer	model	displ	year	cyl	trans	drv	cty	hwy	fl
213	volkswagen	jetta	1.90	1999	4	manual(m5)	f	33	44	d
222	volkswagen	new beetle	1.90	1999	4	manual(m5)	f	35	44	d

plyR for data restructuring

- The plyR package can do more advanced data restructuring
- See chapter 9 of Wickham (2009) for details

```
library(plyr)
xtable(ddply(mpg, .(manufacturer), function(df) data.frame(n = nrow(df),
    mean.disp = mean(df$displ), std.dev.disp = sd(df$disp))))
```

	manufacturer	n	mean.disp	std.dev.disp
1	audi	18	2.54	0.67
2	chevrolet	19	5.06	1.37
3	dodge	37	4.38	0.87
4	ford	25	4.54	0.54
5	honda	9	1.71	0.15
6	hyundai	14	2.43	0.36
7	jeep	8	4.58	1.02
8	land rover	4	4.30	0.26
9	lincoln	3	5.40	0.00
10	mercury	4	4.40	0.49
11	nissan	13	3.27	0.86
12	pontiac	5	3.96	0.81
13	subaru	14	2.46	0.11
14	toyota	34	2.95	0.93
15	volkswagen	27	2.26	0.44

Outline

MACQUARIE UNIVERSITY

Downloading the ggplot2 package

- You'll have to download the ggplot2 graphics package.
- This only has to be done once.

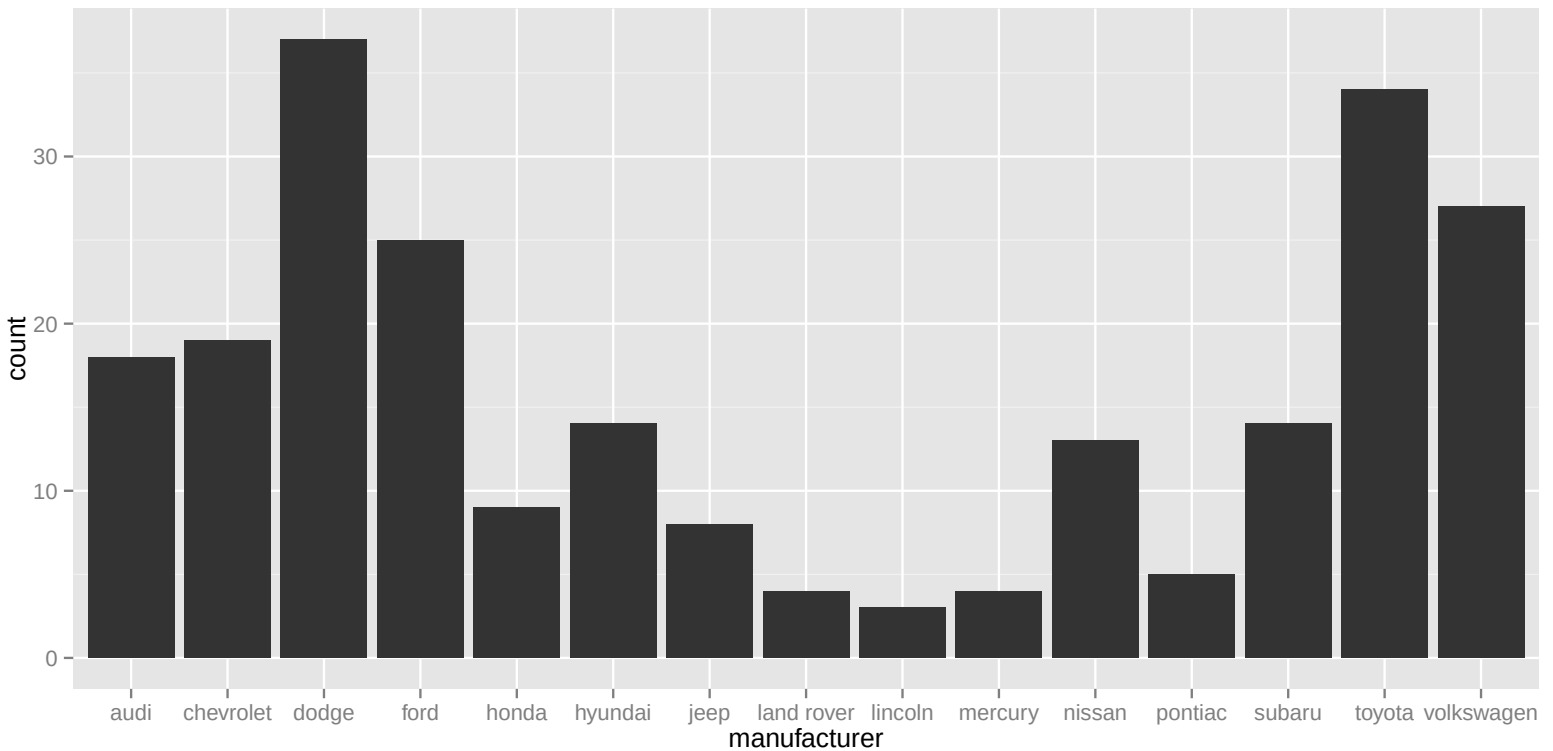
```
install.packages("ggplot2")
```

- Packages are loaded into R with the library function.

```
library(ggplot2)
```

Bar charts of categorical data (1)

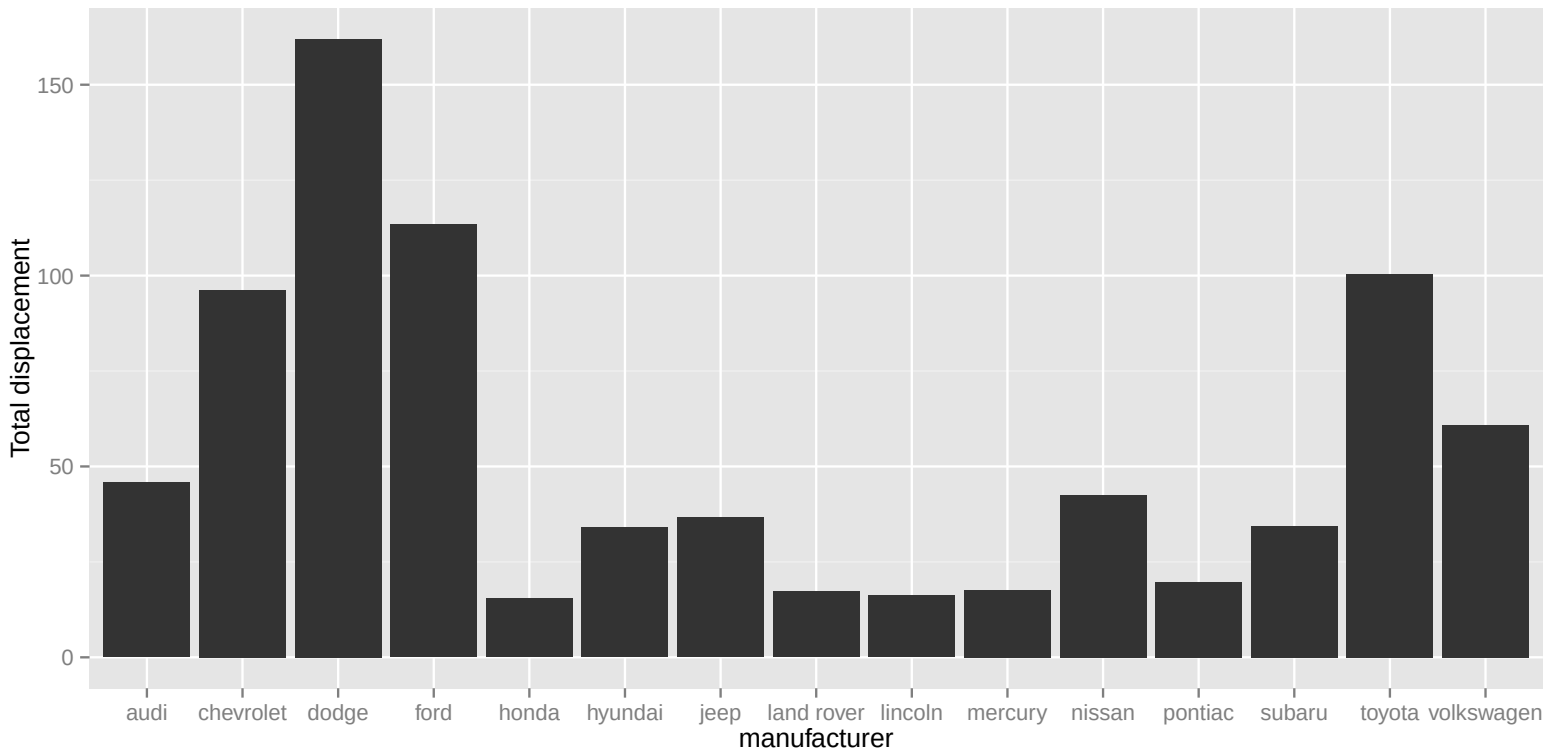
```
qplot(manufacturer, data=d, geom="bar")
```



Bar charts of categorical data (2)

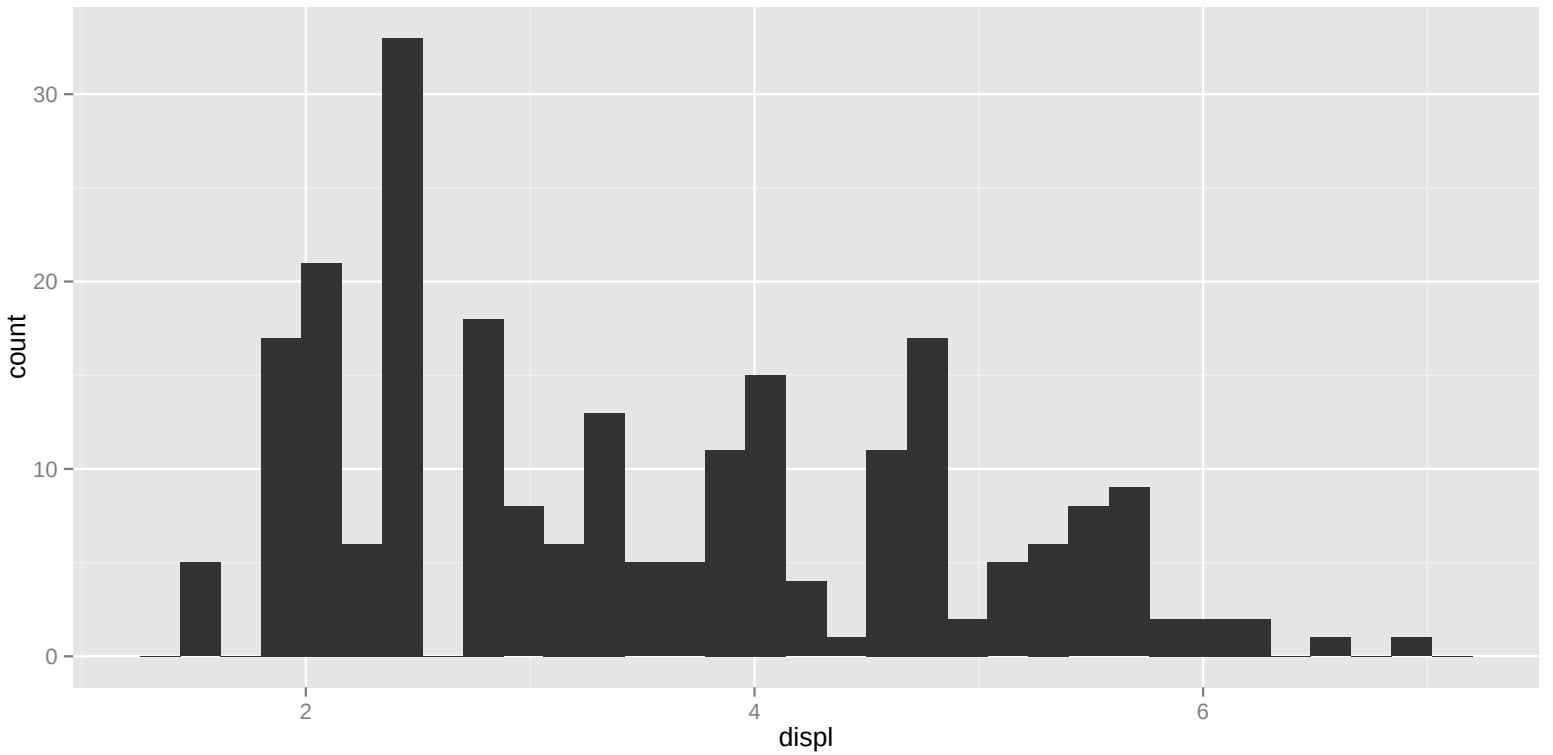
- Sometimes we want to display the value of some numerical attribute

```
qplot(manufacturer, data = d, geom = "bar", weight = displ) +
    scale_y_continuous("Total displacement")
```



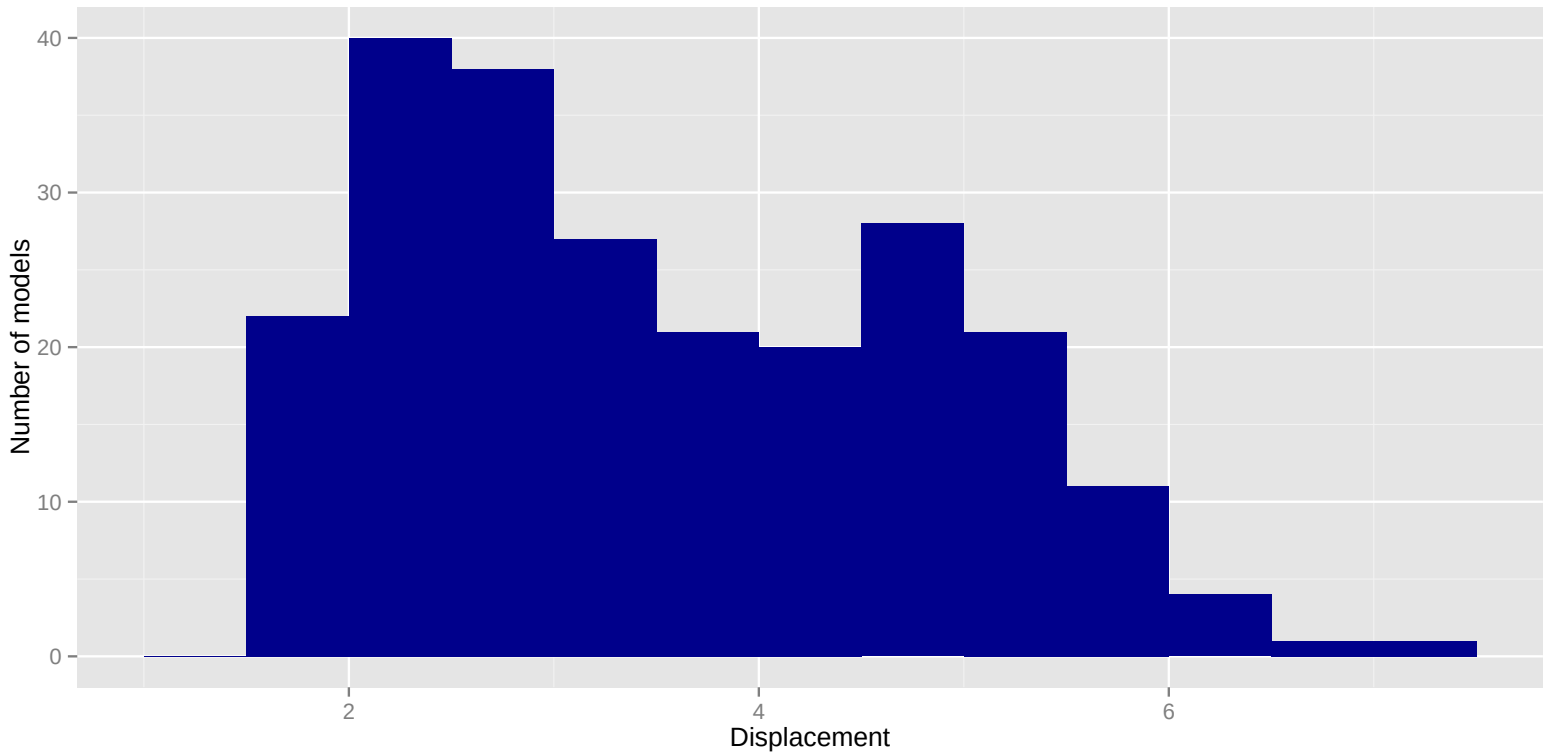
Histograms of continuous data

```
qplot(displ, data=d, geom="histogram")

## stat_bin:  binwidth defaulted to range/30.  Use 'binwidth = x'
to adjust this.
```



Histograms of continuous data

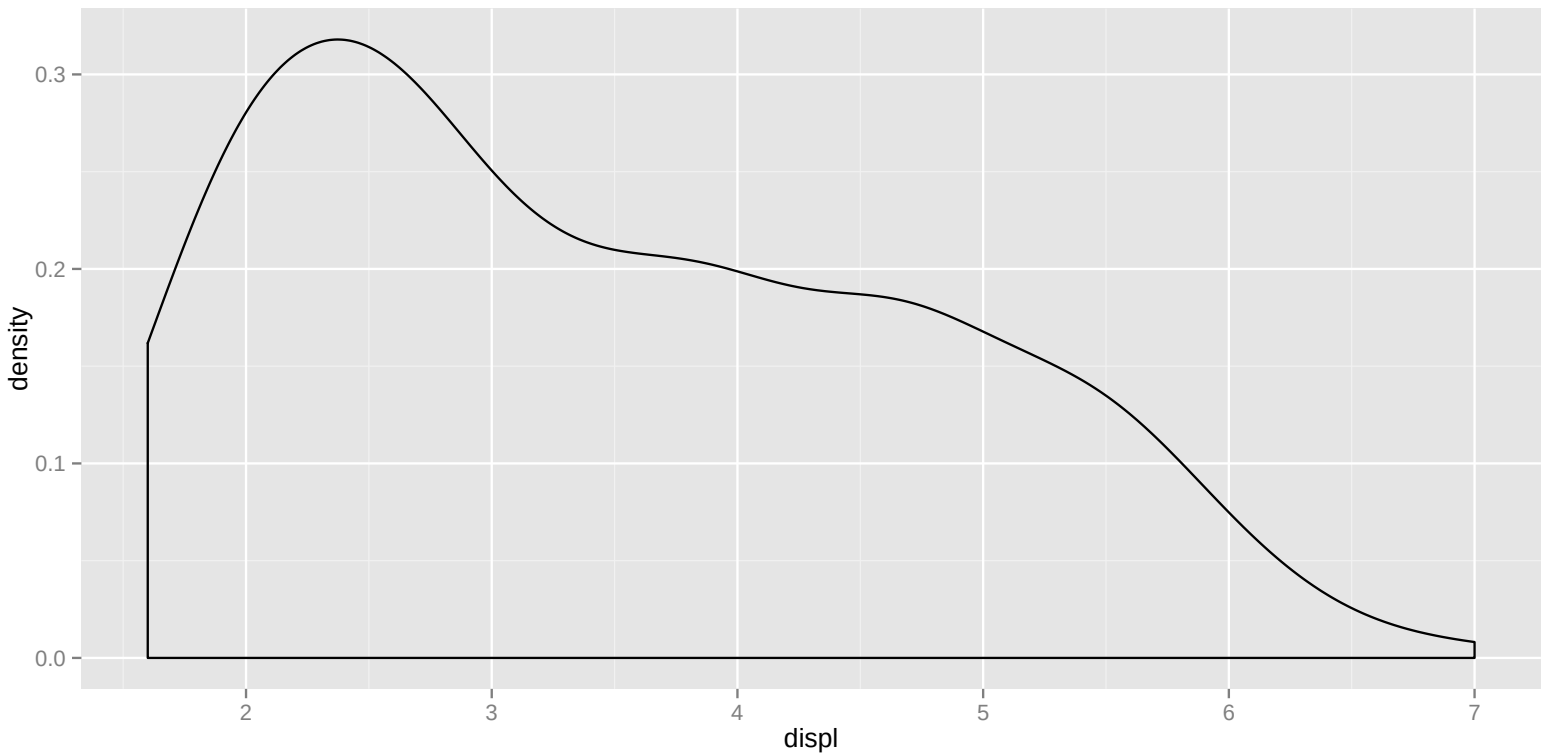
```
qplot(displ, data = d, geom = "histogram", binwidth = 0.5,
    fill = I("DarkBlue")) + scale_x_continuous("Displacement") +
    scale_y_continuous("Number of models")
```



MACQUARIE UNIVERSITY

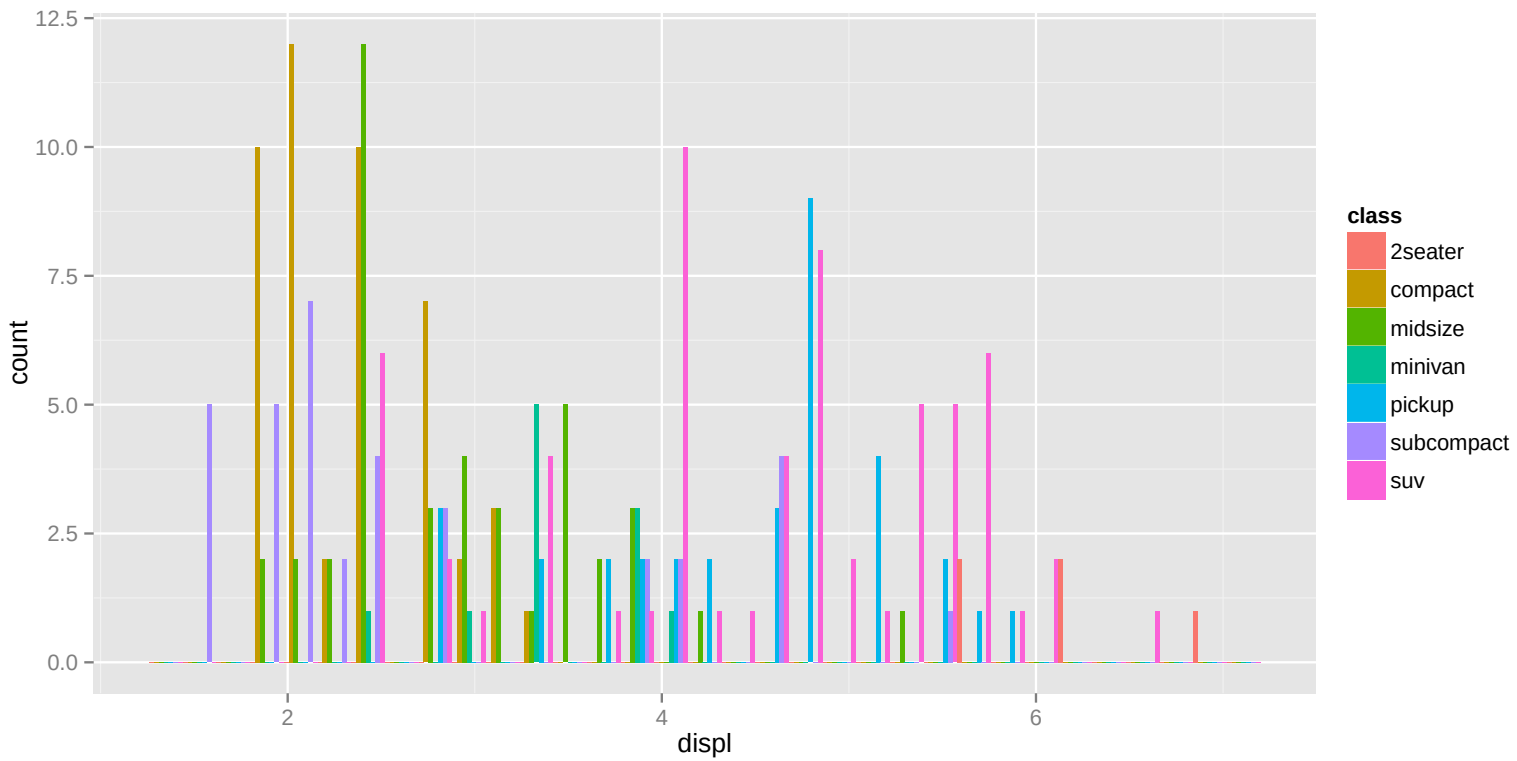
Density plots of continuous data

```
qplot(displ, data=d, geom="density")
```



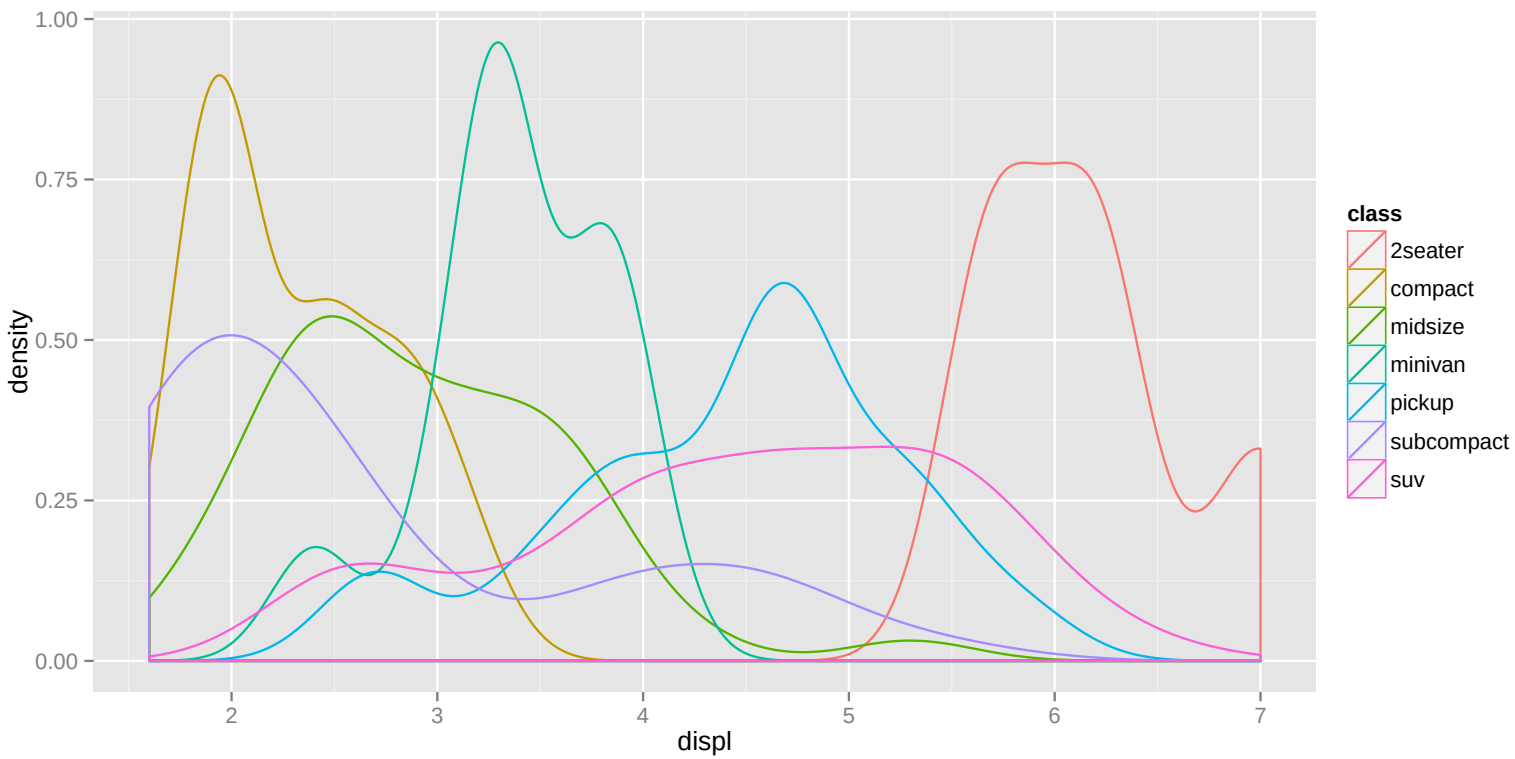
Using colour in a histogram

```
qplot(displ, data=d, geom="histogram", fill=class, position="dodge")

## stat_bin:  binwidth defaulted to range/30.  Use 'binwidth = x'
to adjust this.
```



Using colour in a density plot

```
qplot(displ, data=d, geom="density", colour=class)
```



Scatter plots (1)

- Here we use the `diamonds` data set that comes with `ggplot2`

```
dim(diamonds)
```

[1] 53940 10

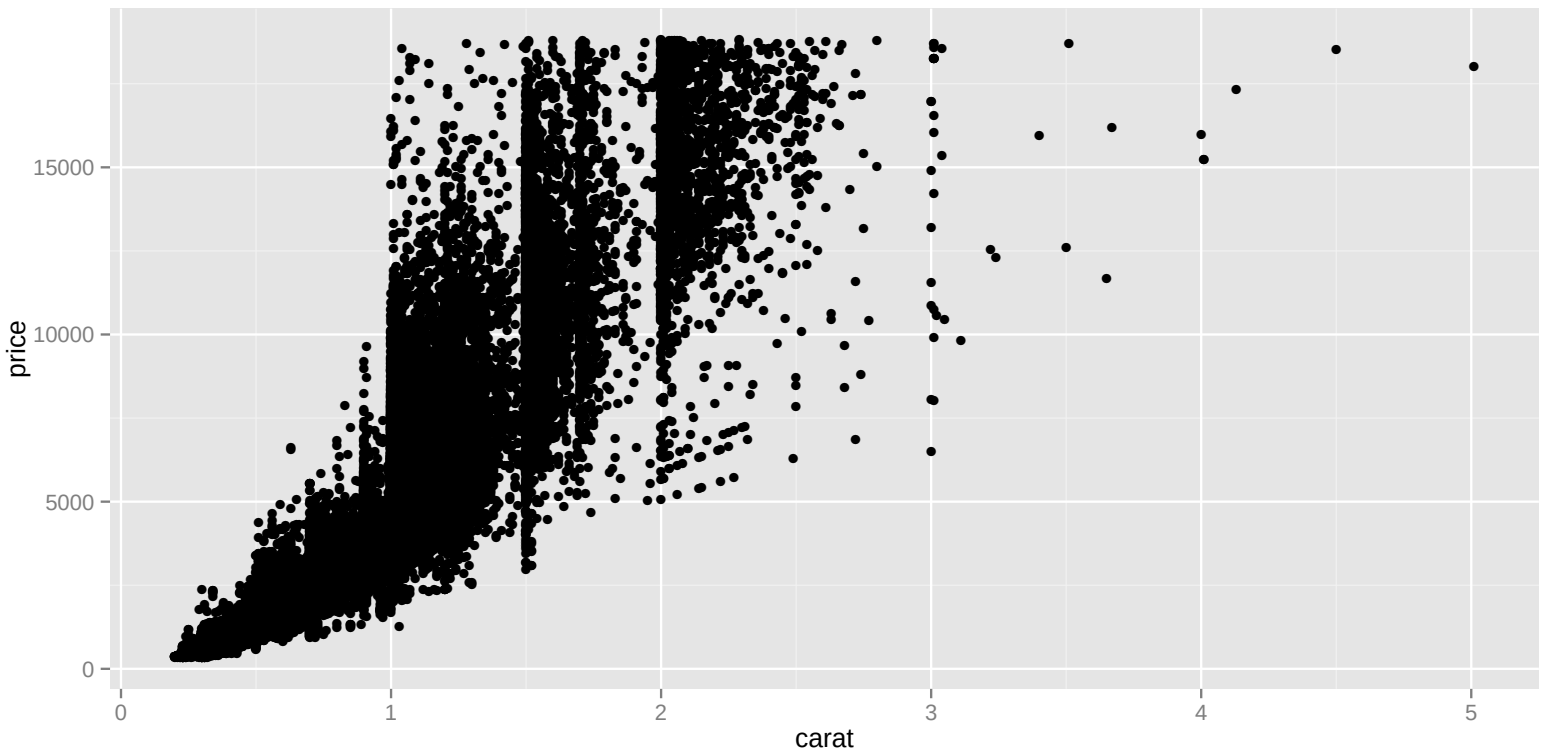
```
xtable(head(diamonds))
```

	carat	cut	color	clarity	depth	table	price	x	y	z
1	0.23	Ideal	E	SI2	61.50	55.00	326	3.95	3.98	2.43
2	0.21	Premium	E	SI1	59.80	61.00	326	3.89	3.84	2.31
3	0.23	Good	E	VS1	56.90	65.00	327	4.05	4.07	2.31
4	0.29	Premium	I	VS2	62.40	58.00	334	4.20	4.23	2.63
5	0.31	Good	J	SI2	63.30	58.00	335	4.34	4.35	2.75
6	0.24	Very Good	J	VVS2	62.80	57.00	336	3.94	3.96	2.48

From Wickham (2009), section 2.3.

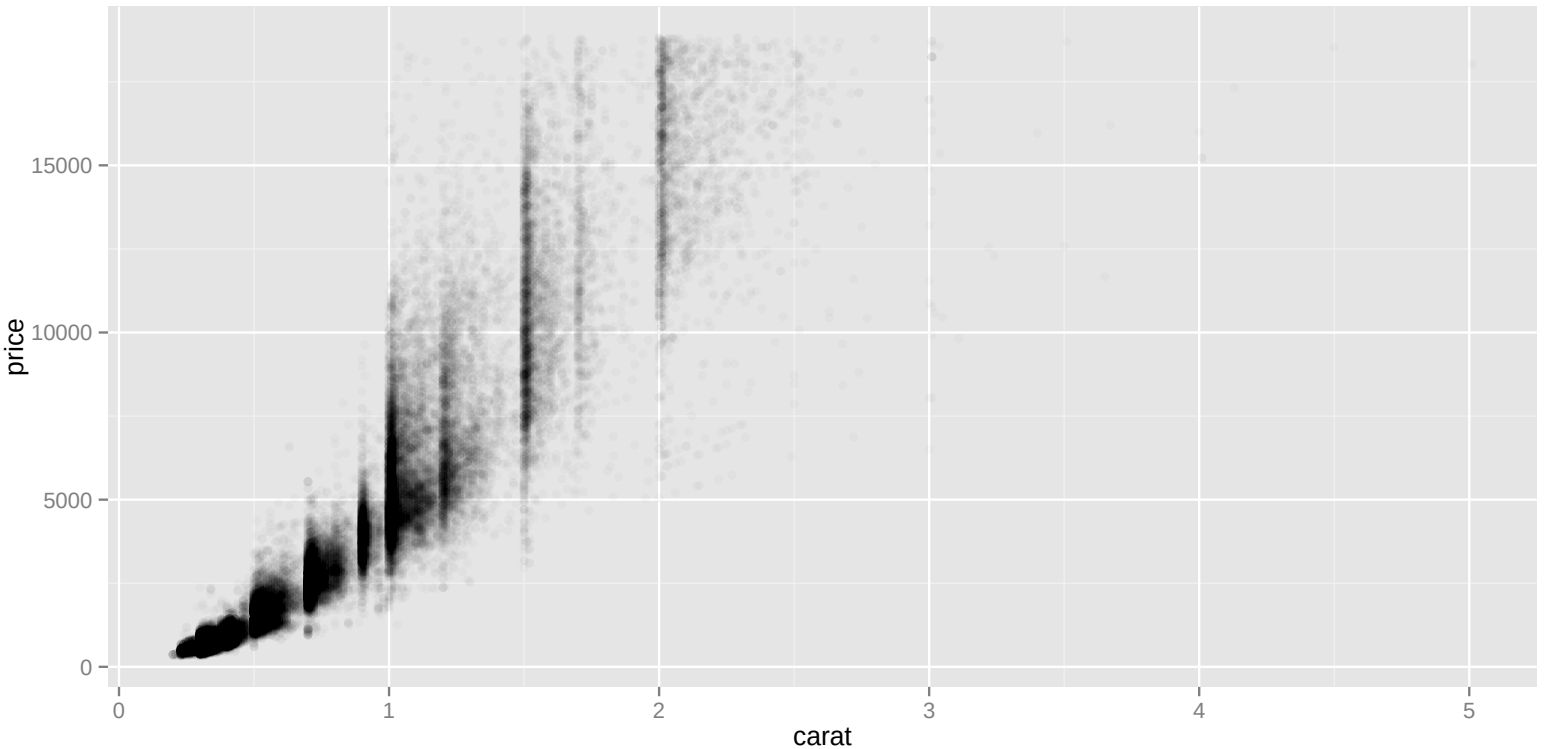
Scatter plots (2)

```
qplot(carat, price, data=diamonds)
```



Reducing overplotting with transparency

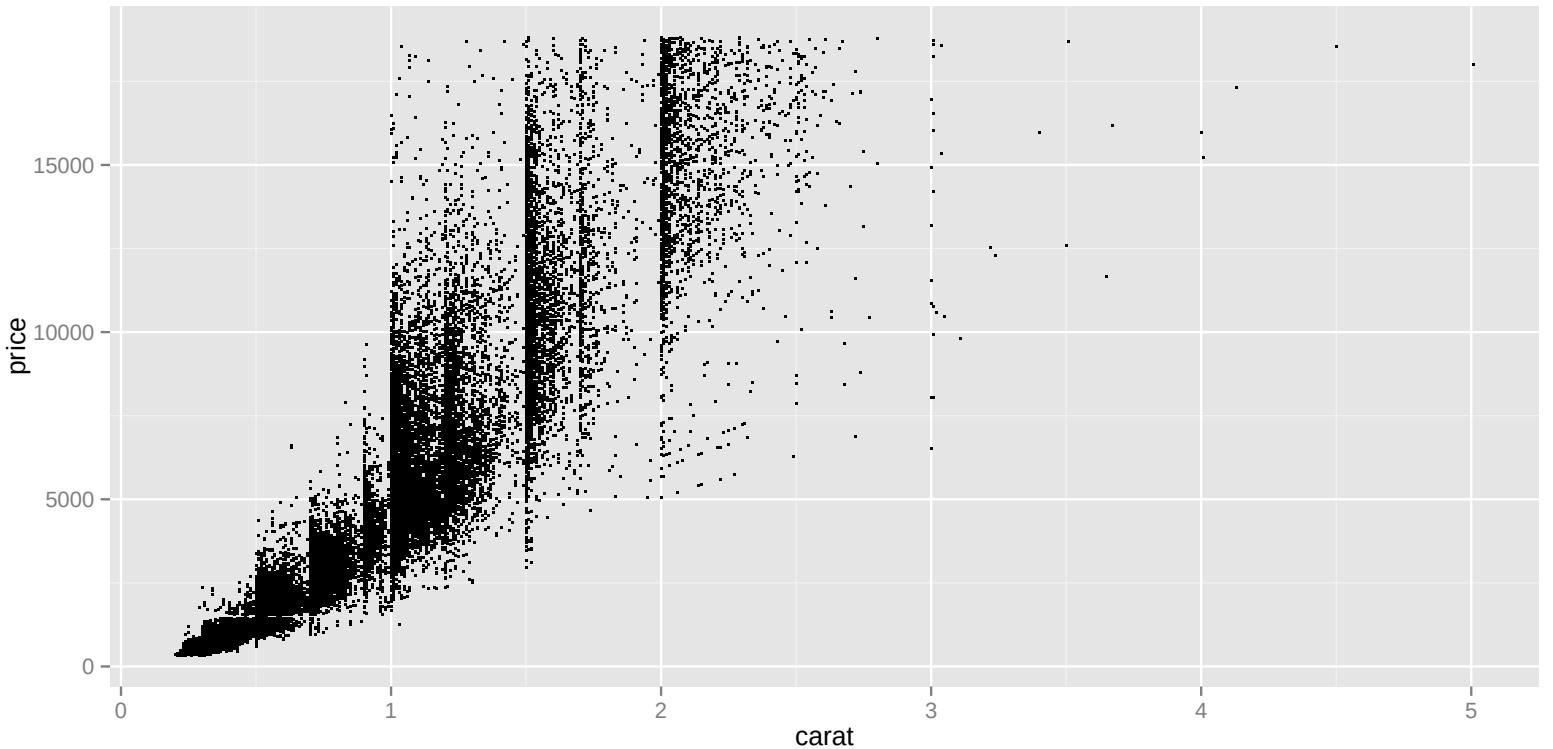
```
qplot(carat, price, data=diamonds, alpha=I(1/50))
```



MACQUARIE UNIVERSITY

Reducing overplotting with pixel-sized points

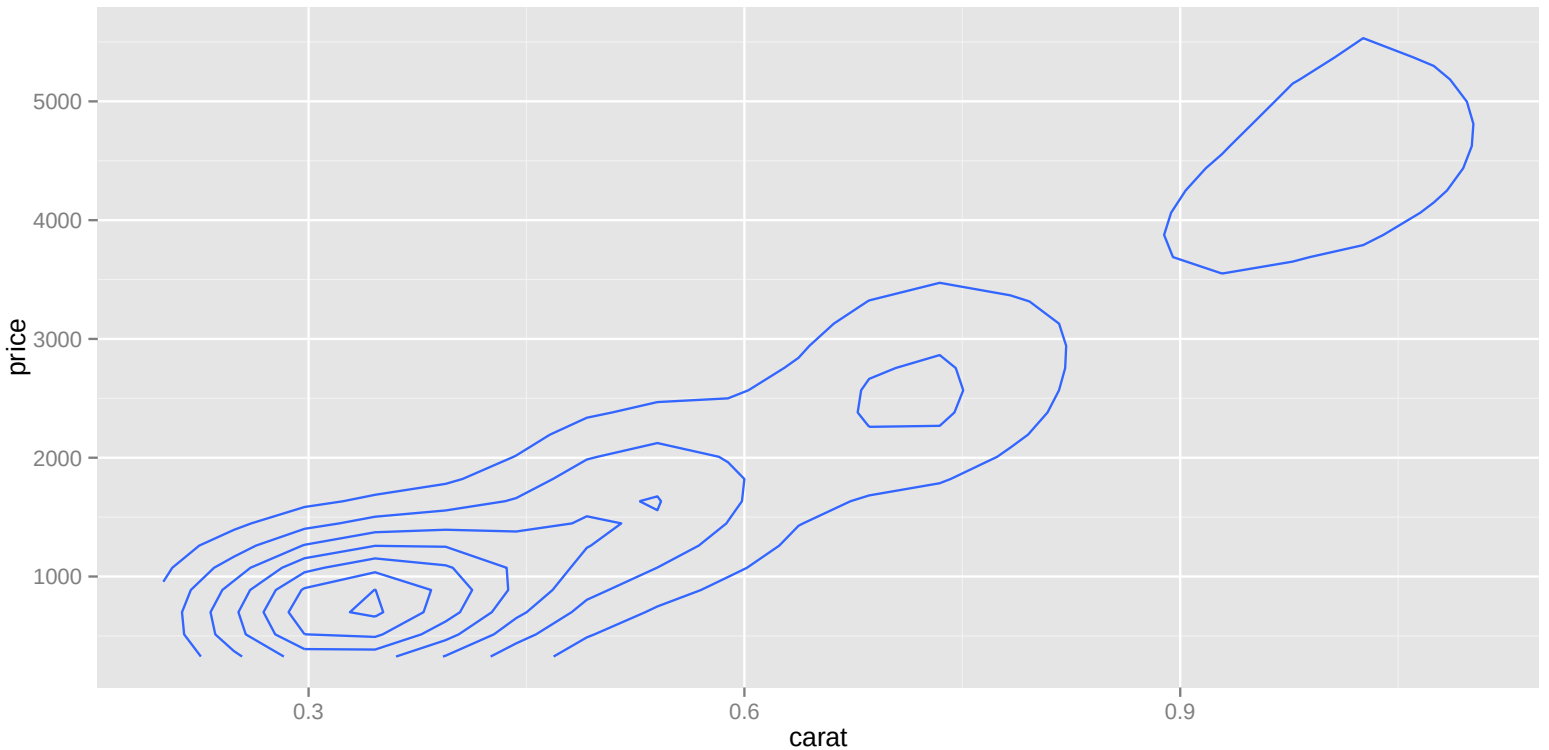
```
ggplot(diamonds, aes(carat, price)) + geom_point(shape=".")
```



See Wickham (2009), section 5.5

Density plots (1)

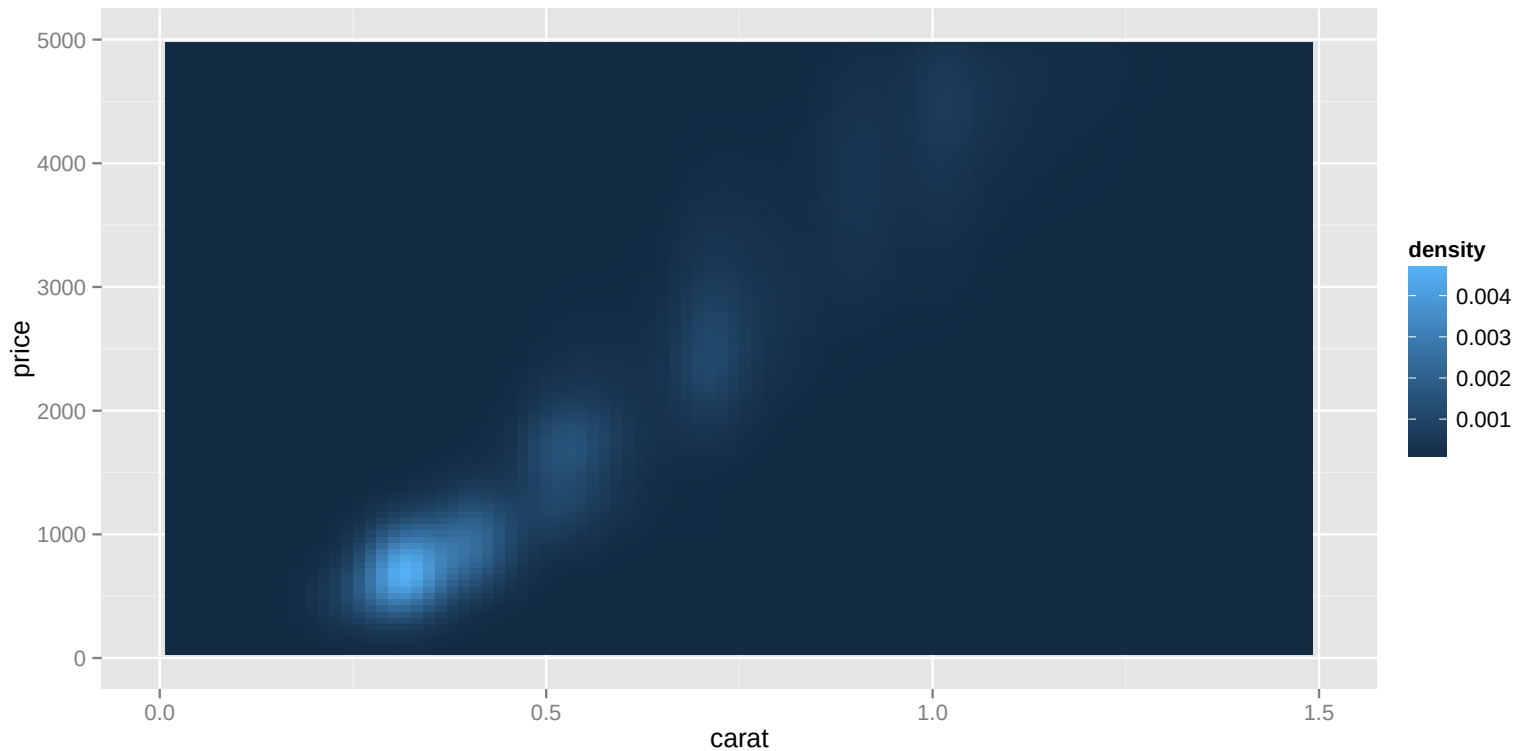
```
ggplot(diamonds, aes(carat, price)) + geom_density2d()
```



See Wickham (2009), section 5.5

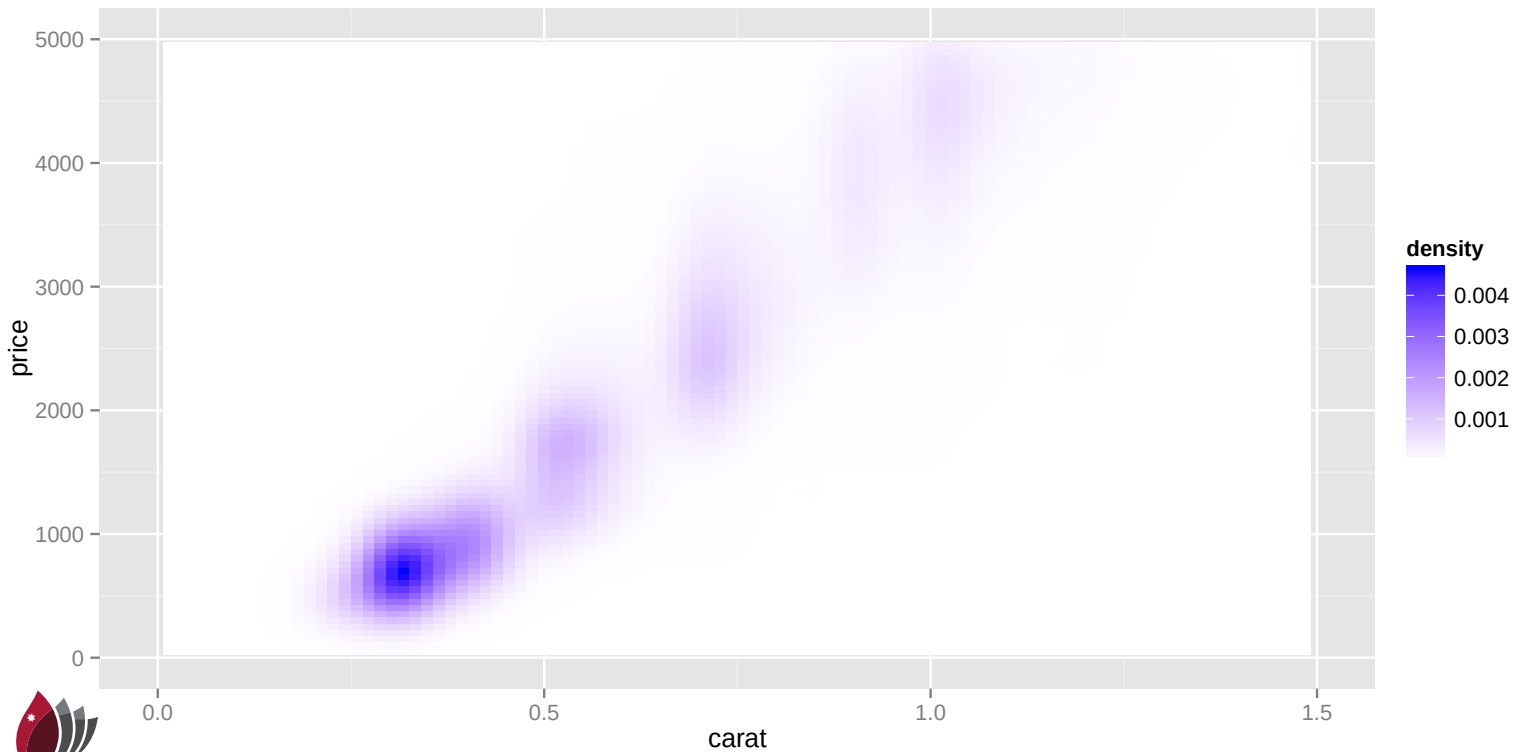
Density plots (2)

```
ggplot(diamonds, aes(carat, price)) + xlim(0,
    1.5) + ylim(0, 5000) + stat_density2d(geom = "tile",
    aes(fill = ..density..), contour = FALSE)

## Warning:  Removed 14736 rows containing non-finite values
(stat_density2d).
```



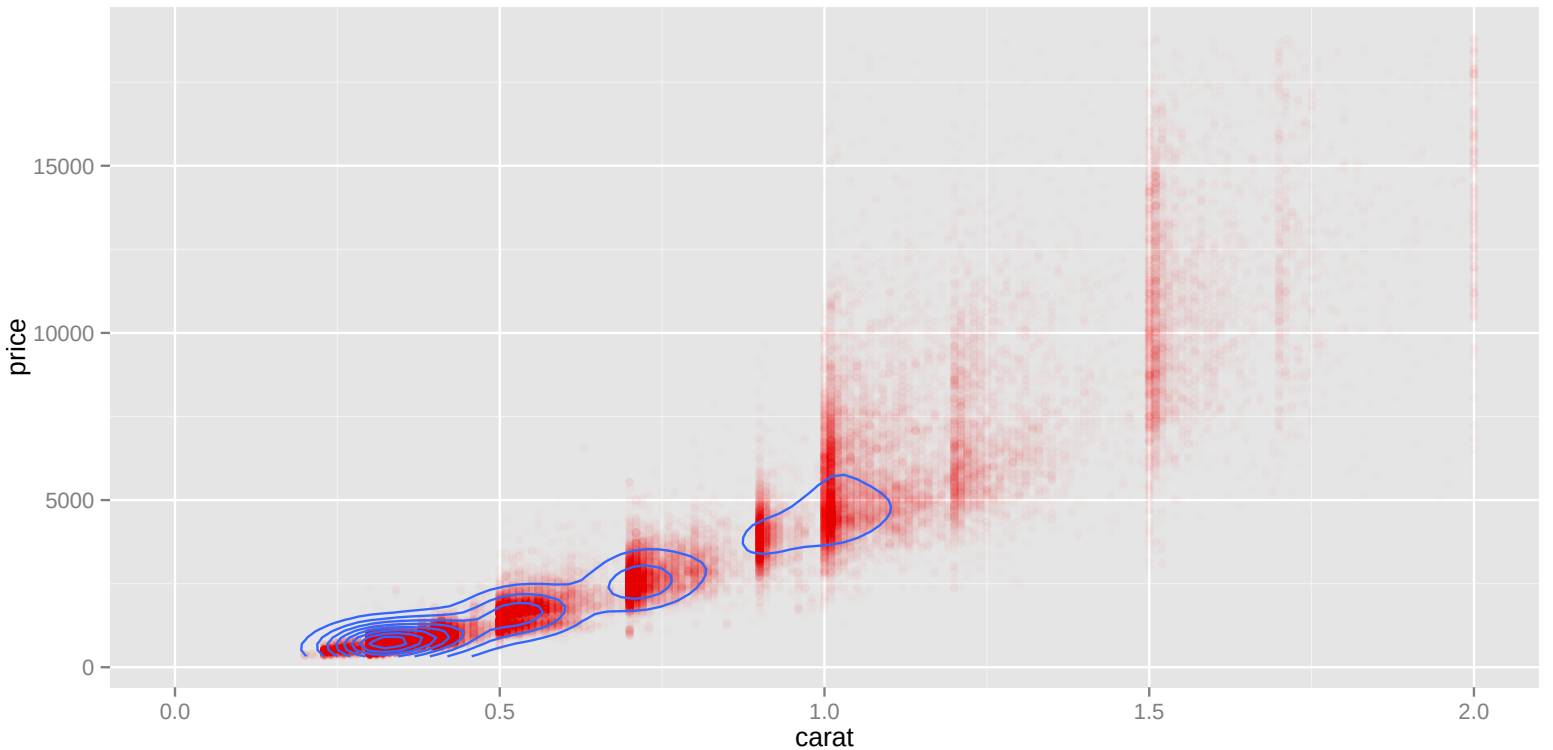
Density plots (3)

```
ggplot(diamonds, aes(carat, price)) + xlim(0,
    1.5) + ylim(0, 5000) + stat_density2d(geom = "tile",
    aes(fill = ..density..), contour = FALSE) +
    scale_fill_gradient(low = "white", high = "blue")

## Warning:  Removed 14736 rows containing non-finite values
(stat_density2d).
```



Density plots (4)

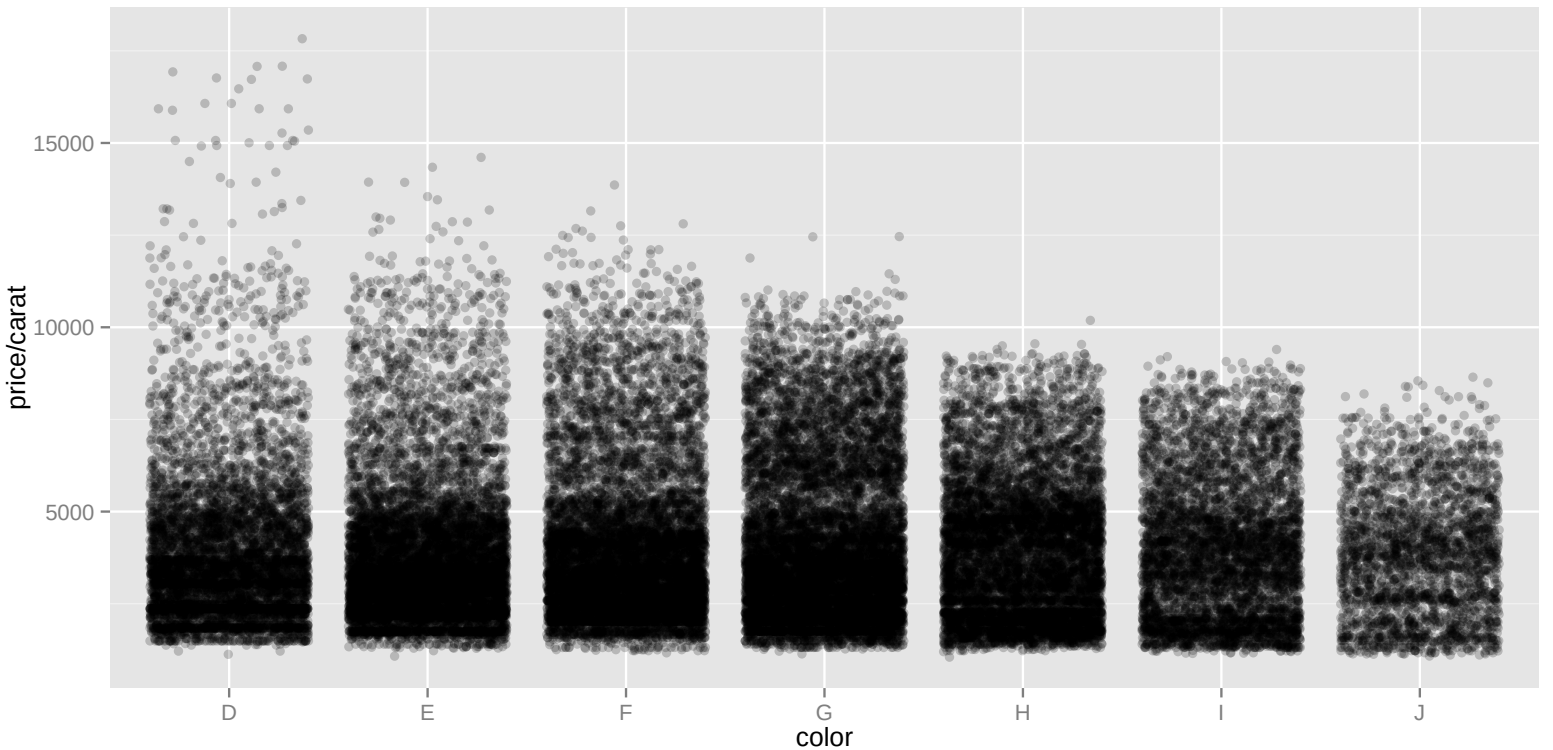
```
ggplot(diamonds, aes(carat, price)) + xlim(0,
    2) + geom_point(colour = "red", alpha = 0.01) +
    geom_density2d()
```



MACQUARIE UNIVERSITY

Reducing overplotting with jittering

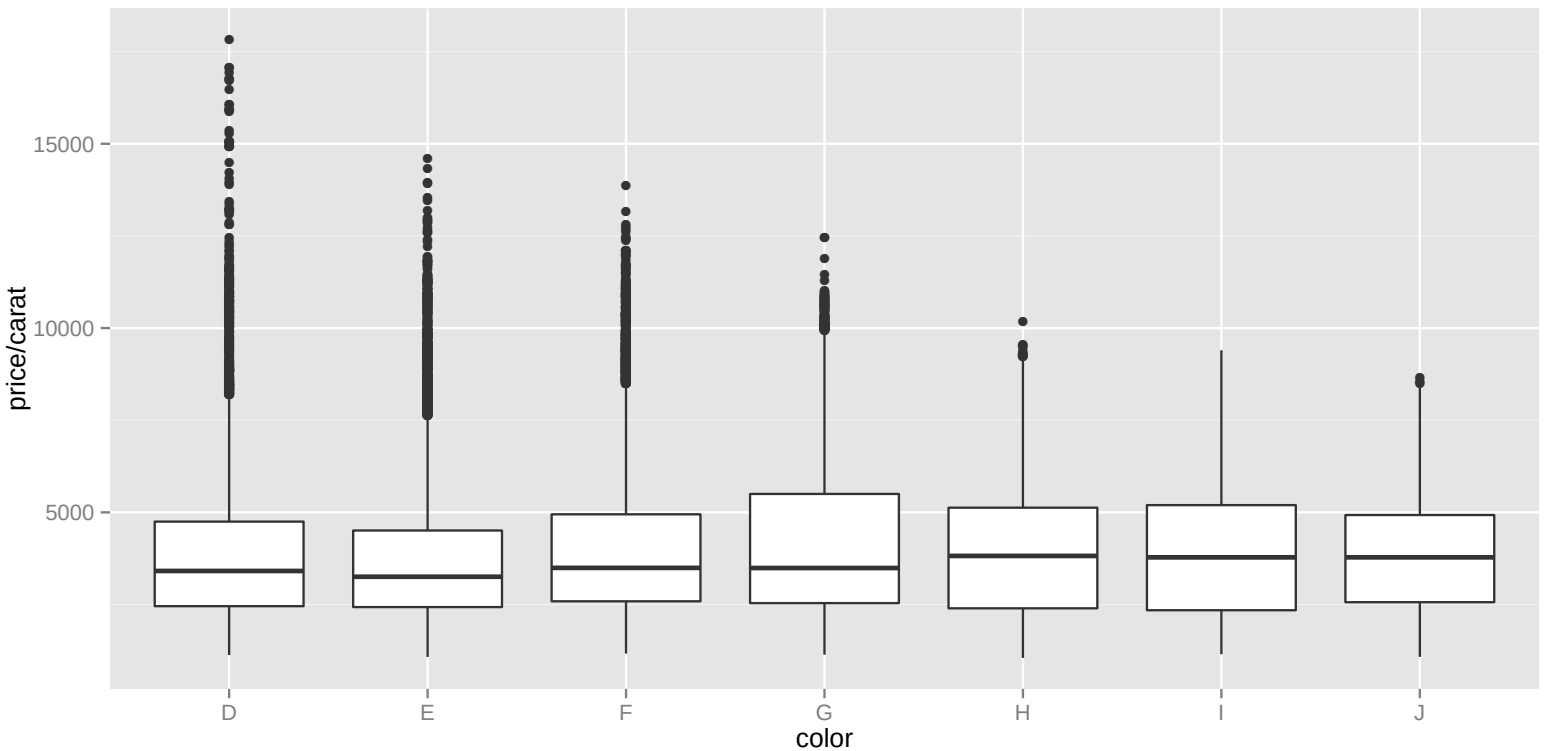
```
qplot(color, price/carat, data = diamonds,
    geom = "jitter", alpha = I(1/5))
```



MACQUARIE UNIVERSITY

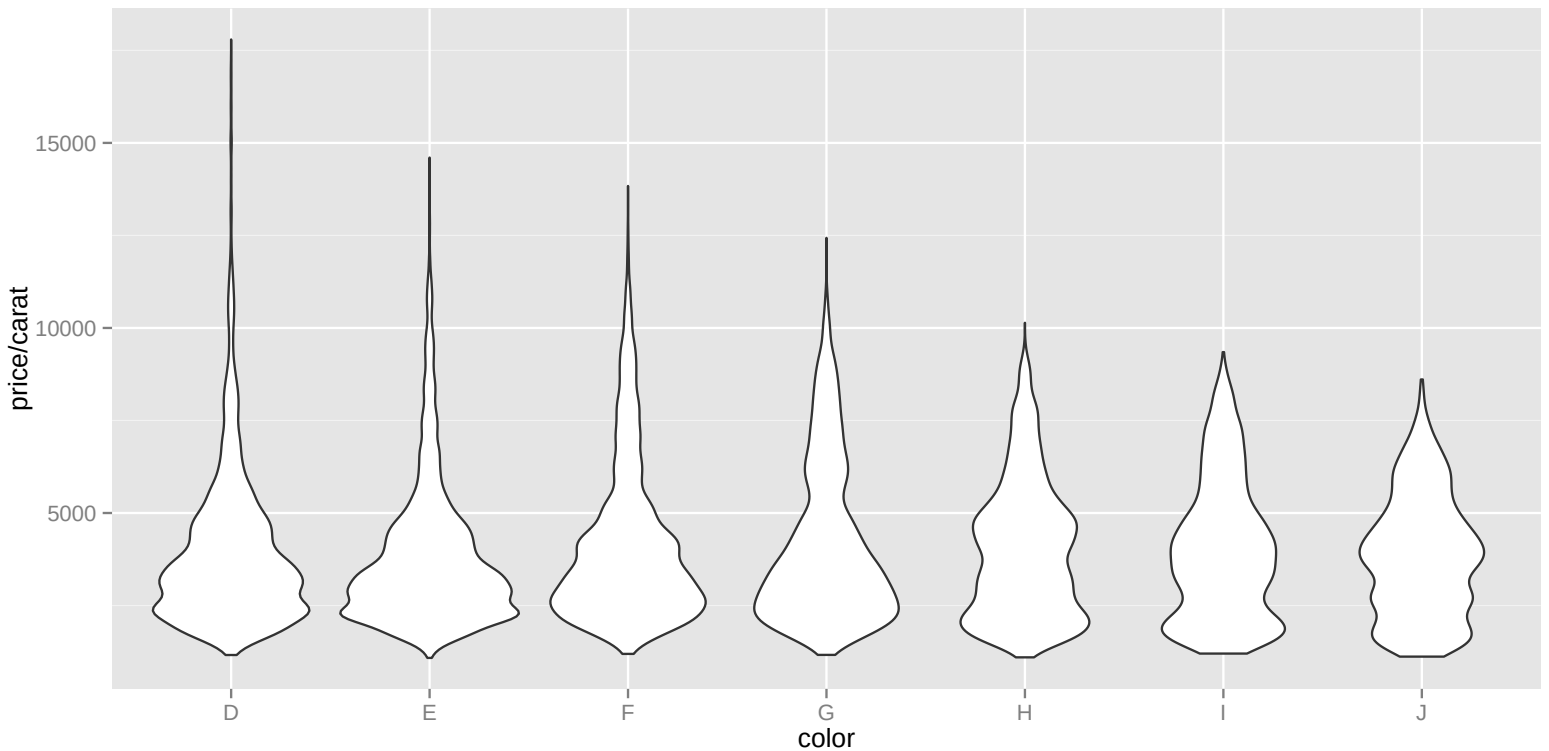
Box and whisker plots

```
qplot(color, price/carat, data=diamonds, geom="boxplot")
```



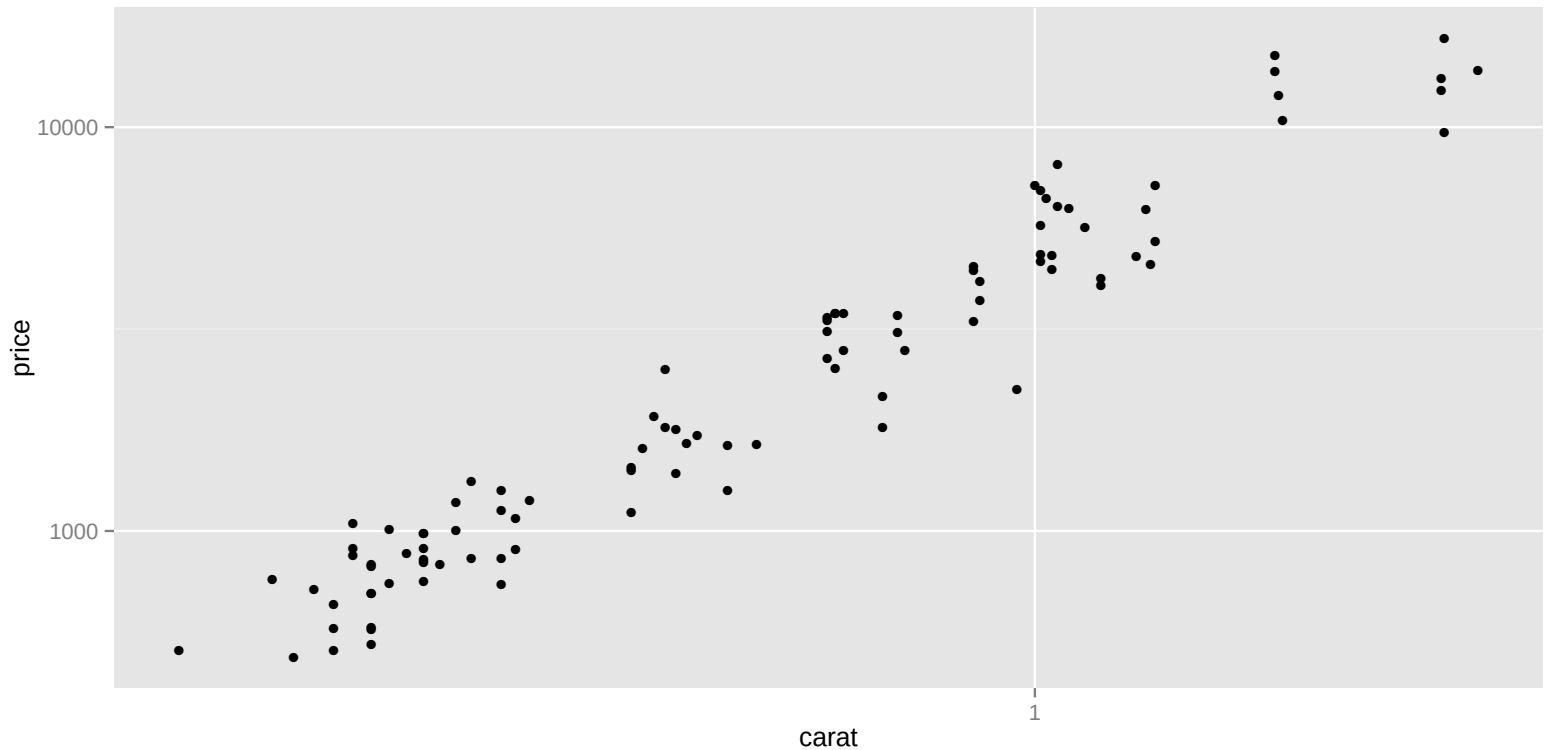
Violin plots

```
ggplot(diamonds, aes(color,price/carat)) + geom_violin()
```



Logarithmic scales

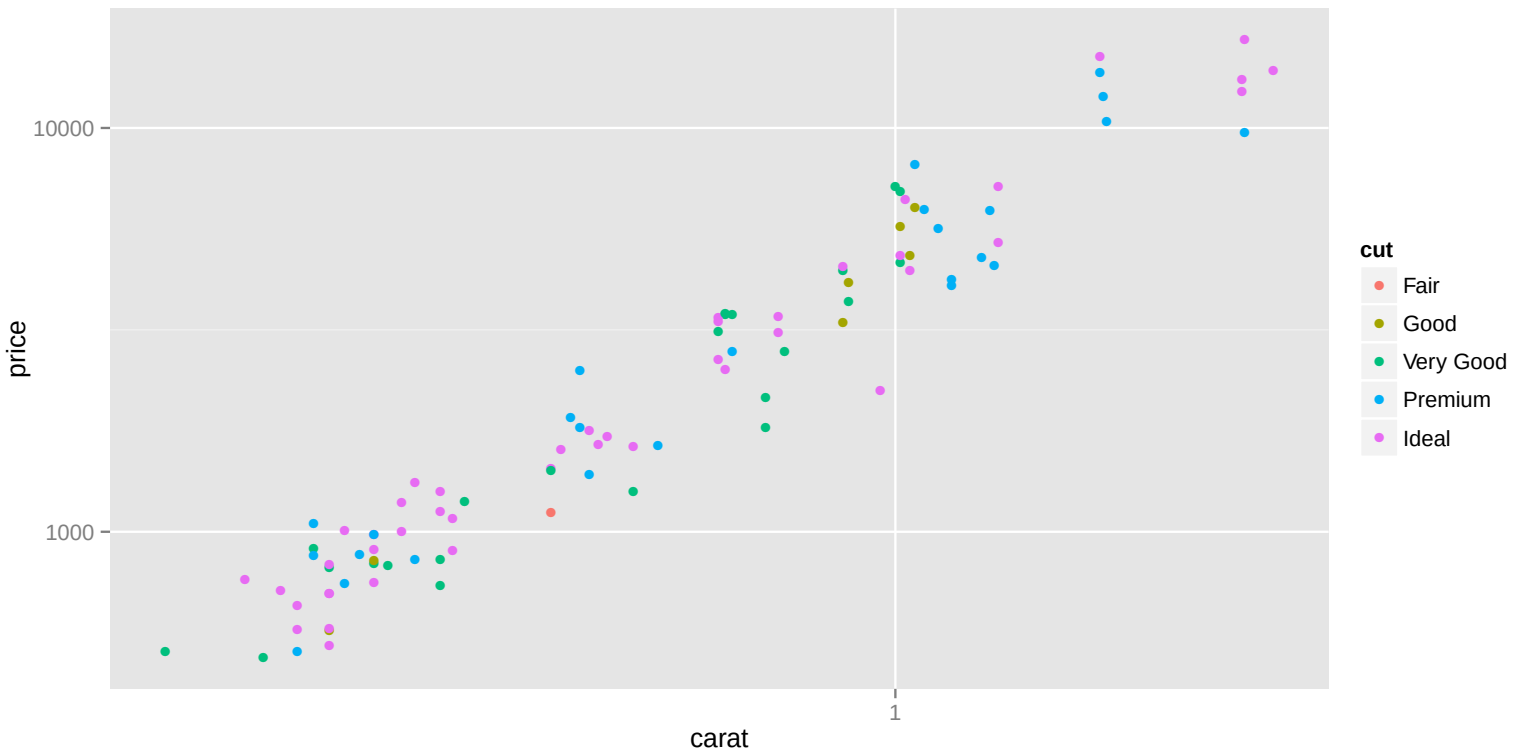
```
sdiamonds <- diamonds[sample(nrow(diamonds), 100), ]
qplot(carat, price, data=sdiamonds, log="xy")
```



MACQUARIE UNIVERSITY

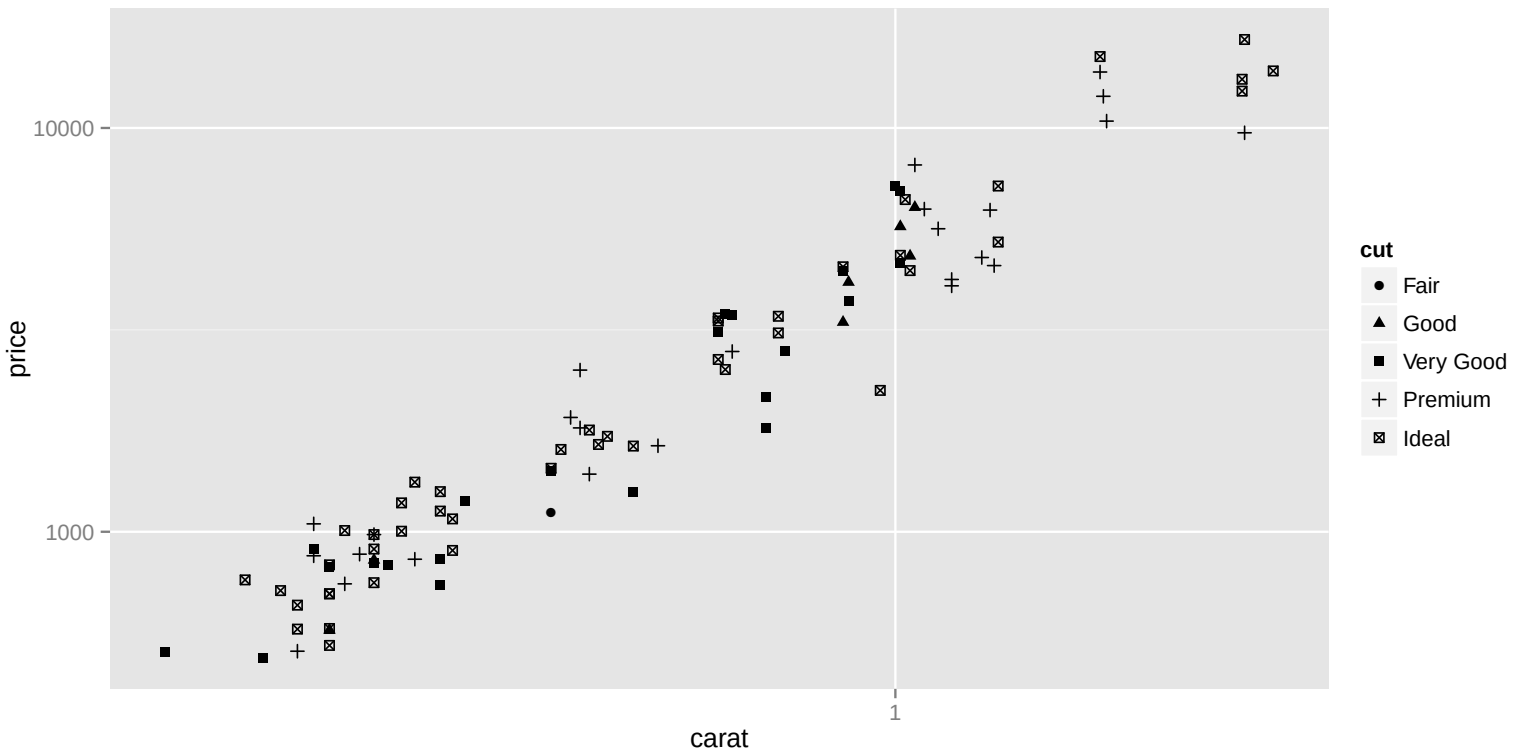
Using colour to indicate a categorical variable

```
qplot(carat, price, data=sdiamonds, log="xy", colour=cut)
```



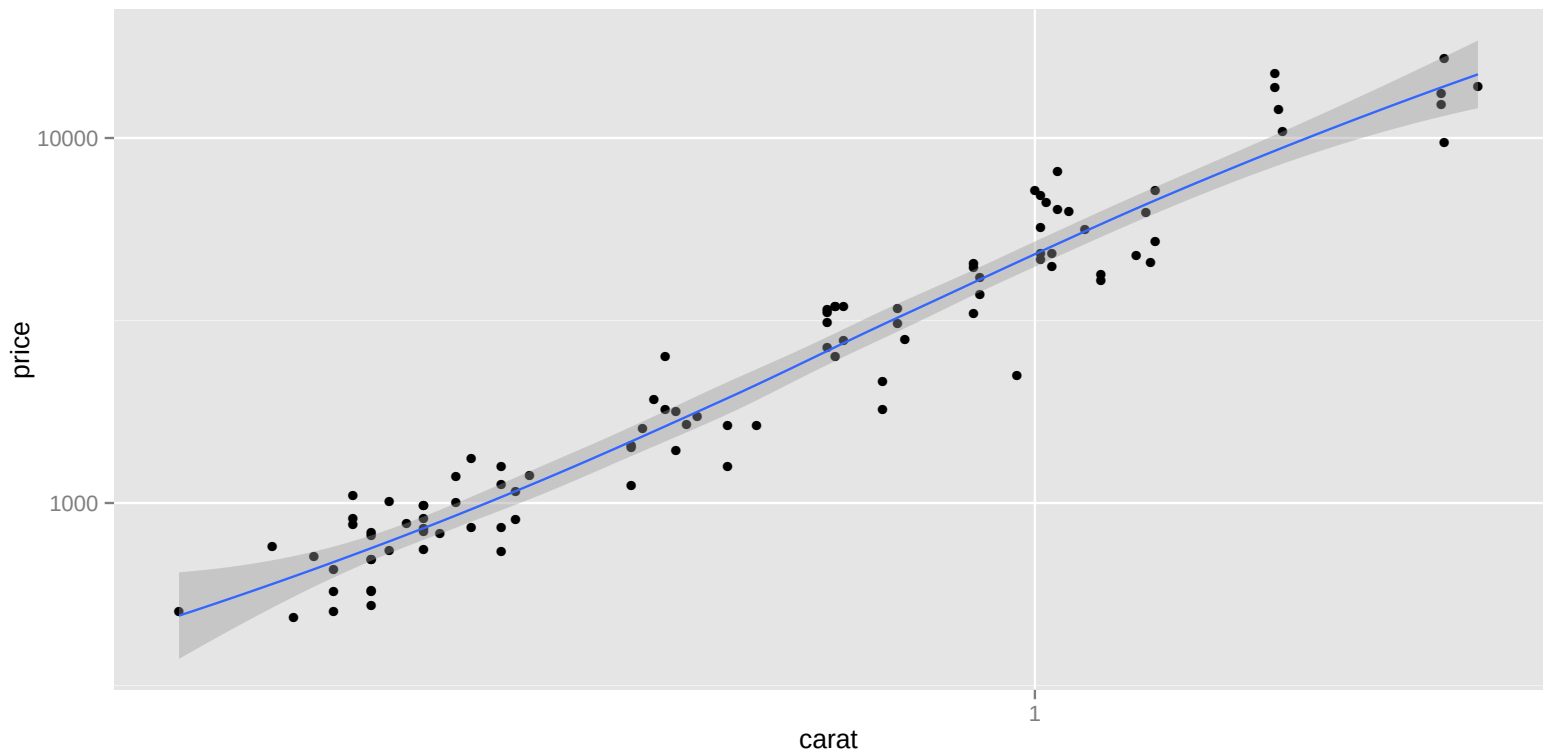
Using shape to indicate a categorical variable

```
qplot(carat, price, data=sdiamonds, log="xy", shape=cut)
```



Adding a smoother

```
qplot(carat, price, data = sdiamonds, log = "xy",
    geom = c("point", "smooth"))

## geom_smooth:  method="auto" and size of largest group is <1000,
so using loess.  Use 'method = x' to change the smoothing method.
```



Line plots (1)

- Here we use the `economics` data set that comes with `ggplot2`

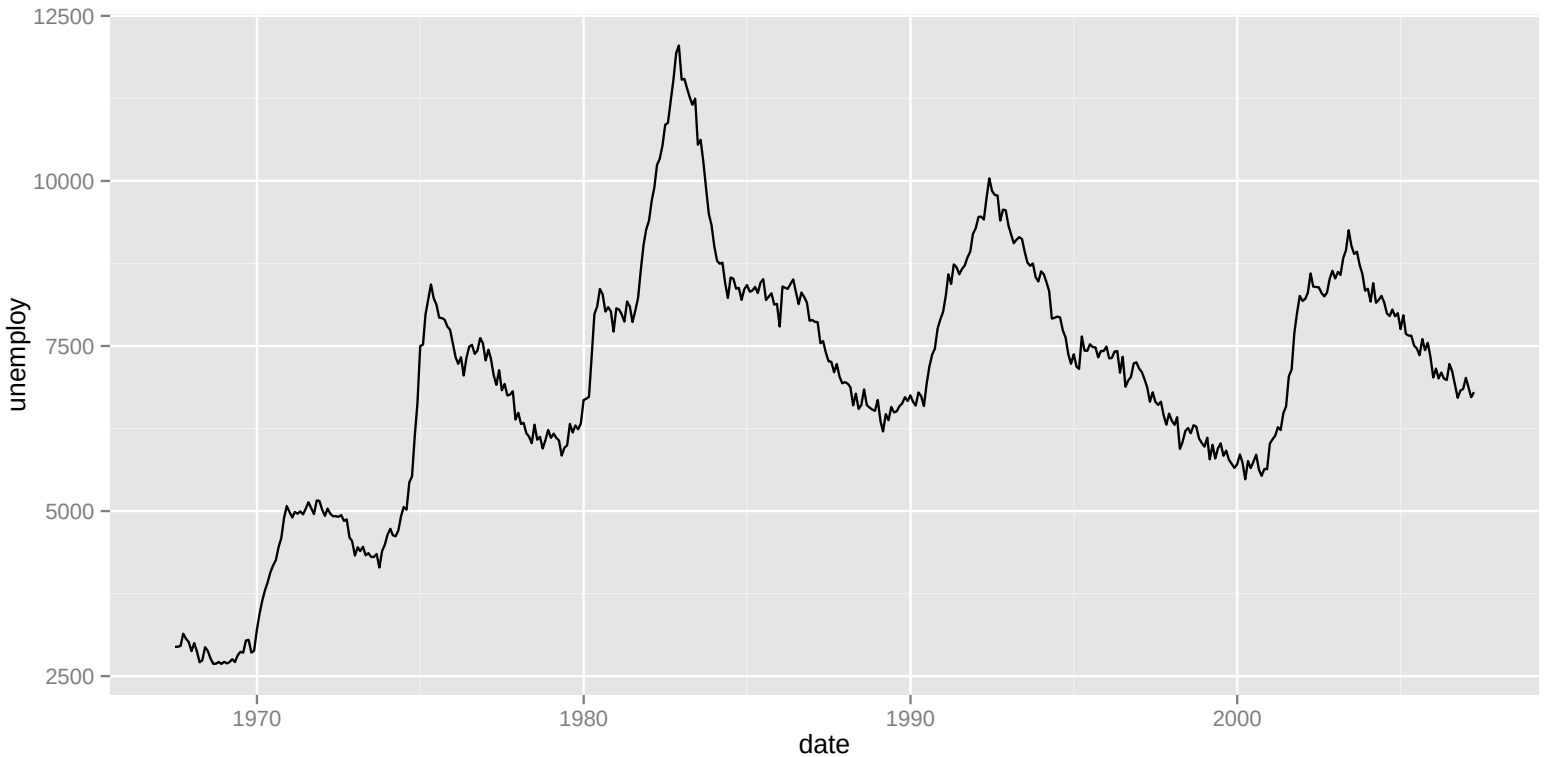
```
xtable(summary(economics))
```

	date	pce	pop	psavert	uem
1	Min. :1967-06-30	Min. : 508	Min. :198712	Min. :-3.00	Min
2	1st Qu.:1977-06-07	1st Qu.:1272	1st Qu.:220094	1st Qu.: 4.00	1st
3	Median :1987-05-15	Median :3082	Median :242516	Median : 7.60	Med
4	Mean :1987-05-16	Mean :3654	Mean :246349	Mean : 6.72	Mea
5	3rd Qu.:1997-04-22	3rd Qu.:5474	3rd Qu.:272277	3rd Qu.: 9.50	3rd
6	Max. :2007-03-31	Max. :9705	Max. :301913	Max. :14.60	Max

See Wickham (2009), section 2.5.5.

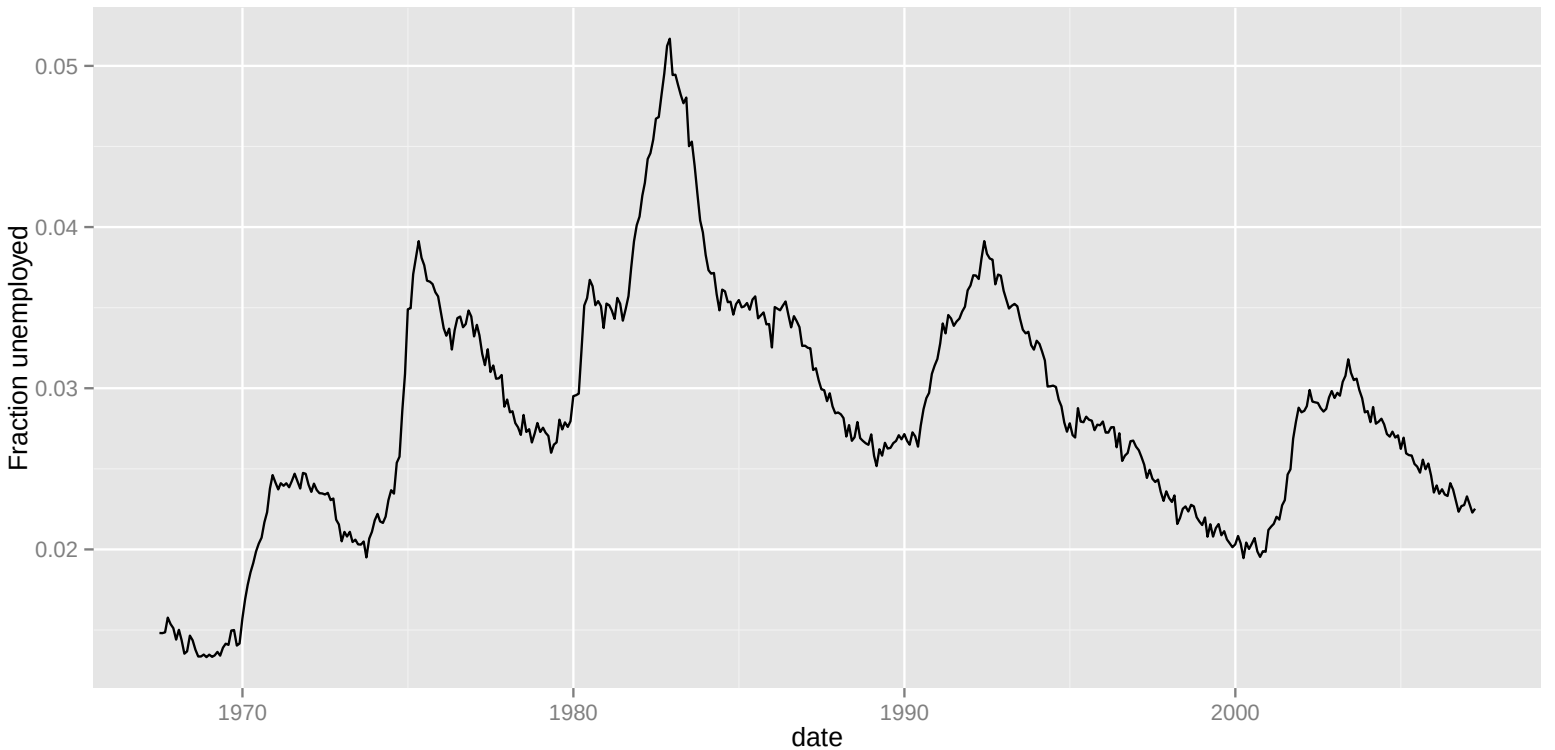
Line plots (2)

```
qplot(date, unemploy, data=economics, geom="line")
```



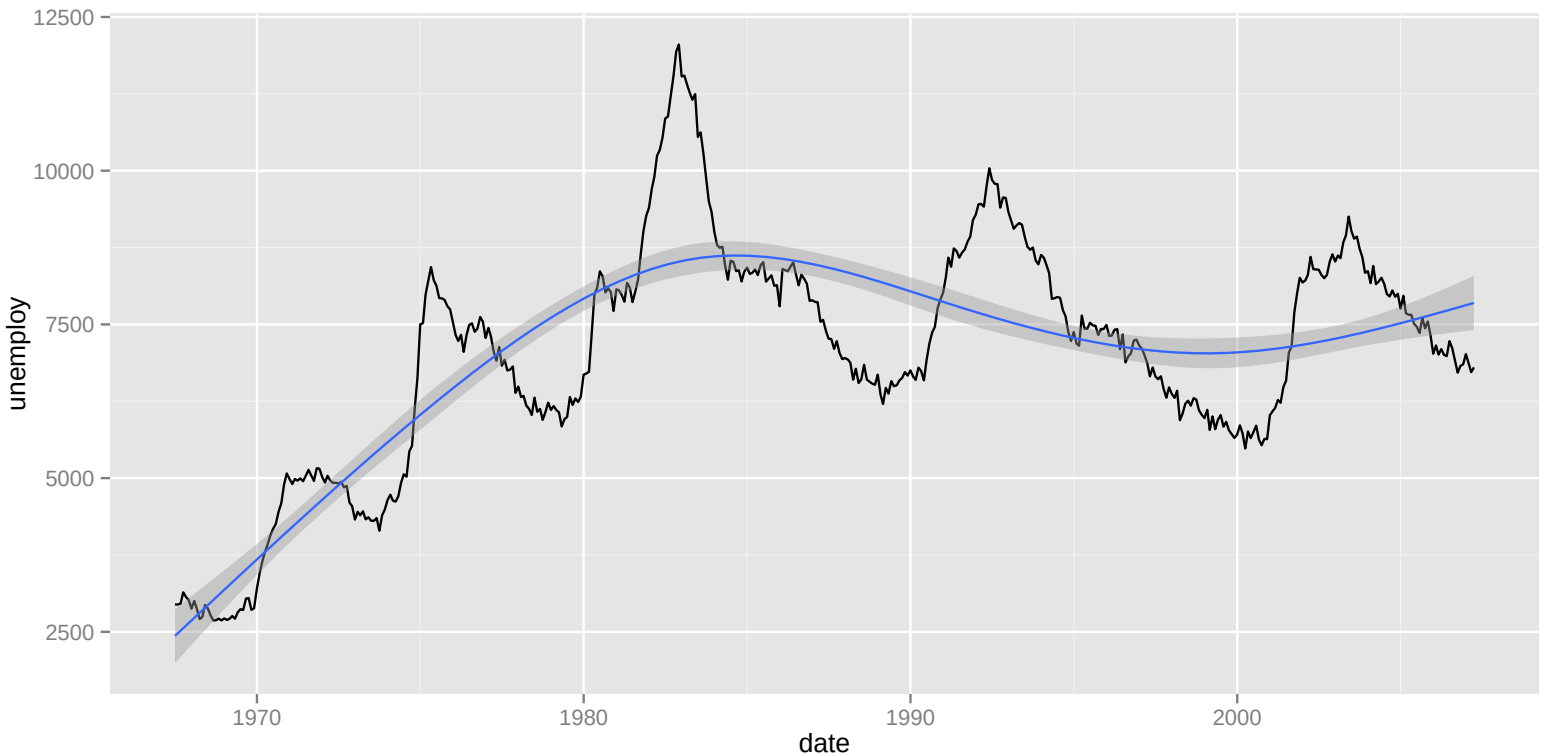
Line plots (3)

```
qplot(date, unemploy/pop, data = economics, geom = "line") +
    scale_y_continuous("Fraction unemployed")
```



Line plots (4)

```
library(splines)
qplot(date, unemploy, data = economics, geom = c("line",
    "smooth"), method = "lm", formula = y ~ ns(x,
    5))
```



See Wickham (2009) sections 2.5.1 and 5.9

MACQUARIE UNIVERSITY

Rescaling values before plotting

```
xtable(head(economics))
```

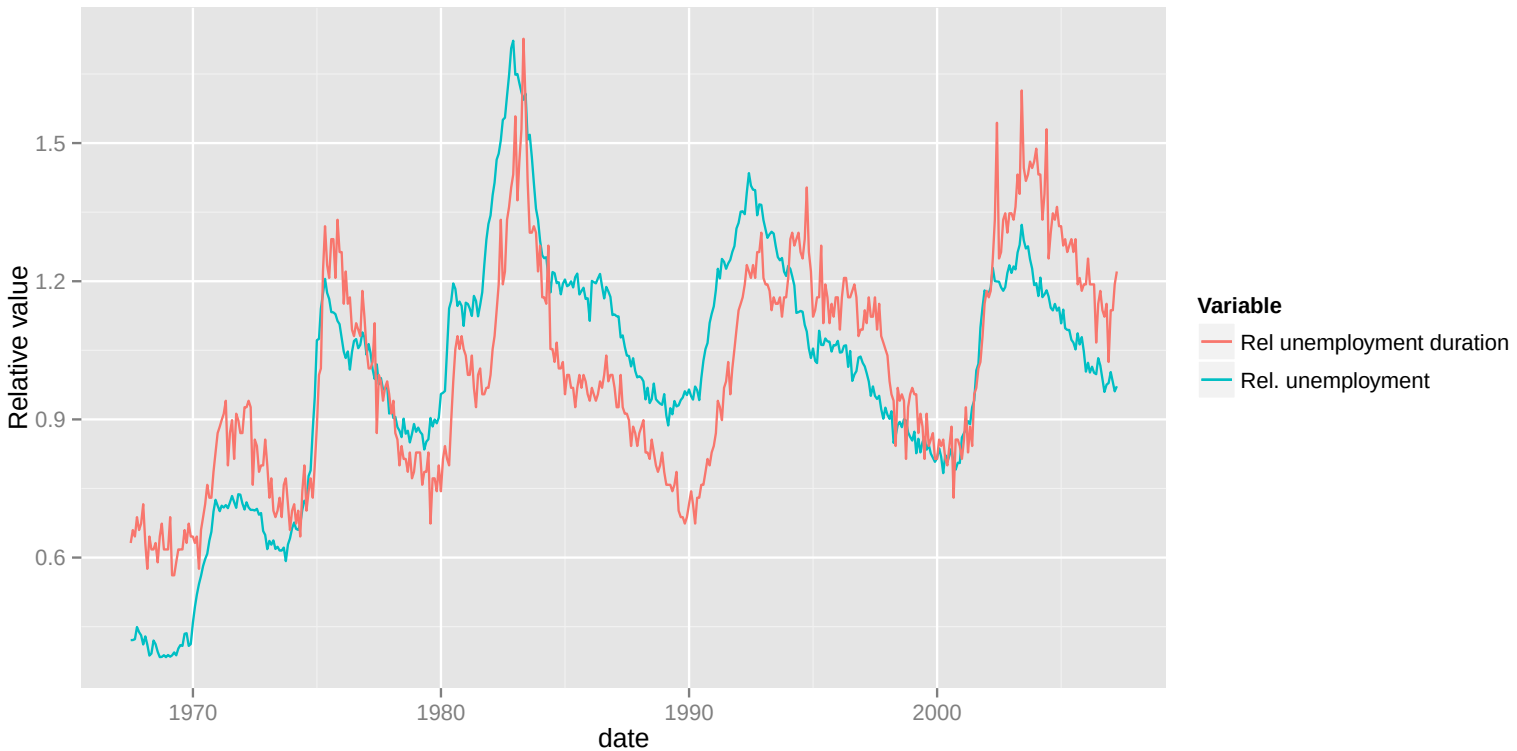
	date	pce	pop	psavert	uempmed	unemploy
1	-916.00	507.80	198712	9.80	4.50	2944
2	-885.00	510.90	198911	9.80	4.70	2945
3	-854.00	516.70	199113	9.00	4.60	2958
4	-824.00	513.30	199311	9.80	4.90	3143
5	-793.00	518.50	199498	9.70	4.70	3066
6	-763.00	526.20	199657	9.40	4.80	3018

```
e <- with(economics, data.frame(date = date,
    rel.unemploy = unemploy/mean(unemploy),
    rel.uempmed = uempmed/mean(uempmed)))
xtable(head(e))
```

	date	rel.unemploy	rel.uempmed
1	-916.00	0.42	0.63
2	-885.00	0.42	0.66
3	-854.00	0.42	0.65
4	-824.00	0.45	0.69
5	-793.00	0.44	0.66
6	-763.00	0.43	0.67

Multiple lines on a single plot

```
ggplot(e, aes(date)) + geom_line(aes(y = rel.unemploy,
    colour = "Rel. unemployment")) + geom_line(aes(y = rel.uempmed,
    colour = "Rel unemployment duration")) +
    scale_colour_hue("Variable") + scale_y_continuous("Relative value")
```



See Wickham (2009) section 9.2.1

"Wide" and "narrow" data

```
xtable(head(e))
```

	date	rel.unemploy	rel.uempmed
1	-916.00	0.42	0.63
2	-885.00	0.42	0.66
3	-854.00	0.42	0.65
4	-824.00	0.45	0.69
5	-793.00	0.44	0.66
6	-763.00	0.43	0.67

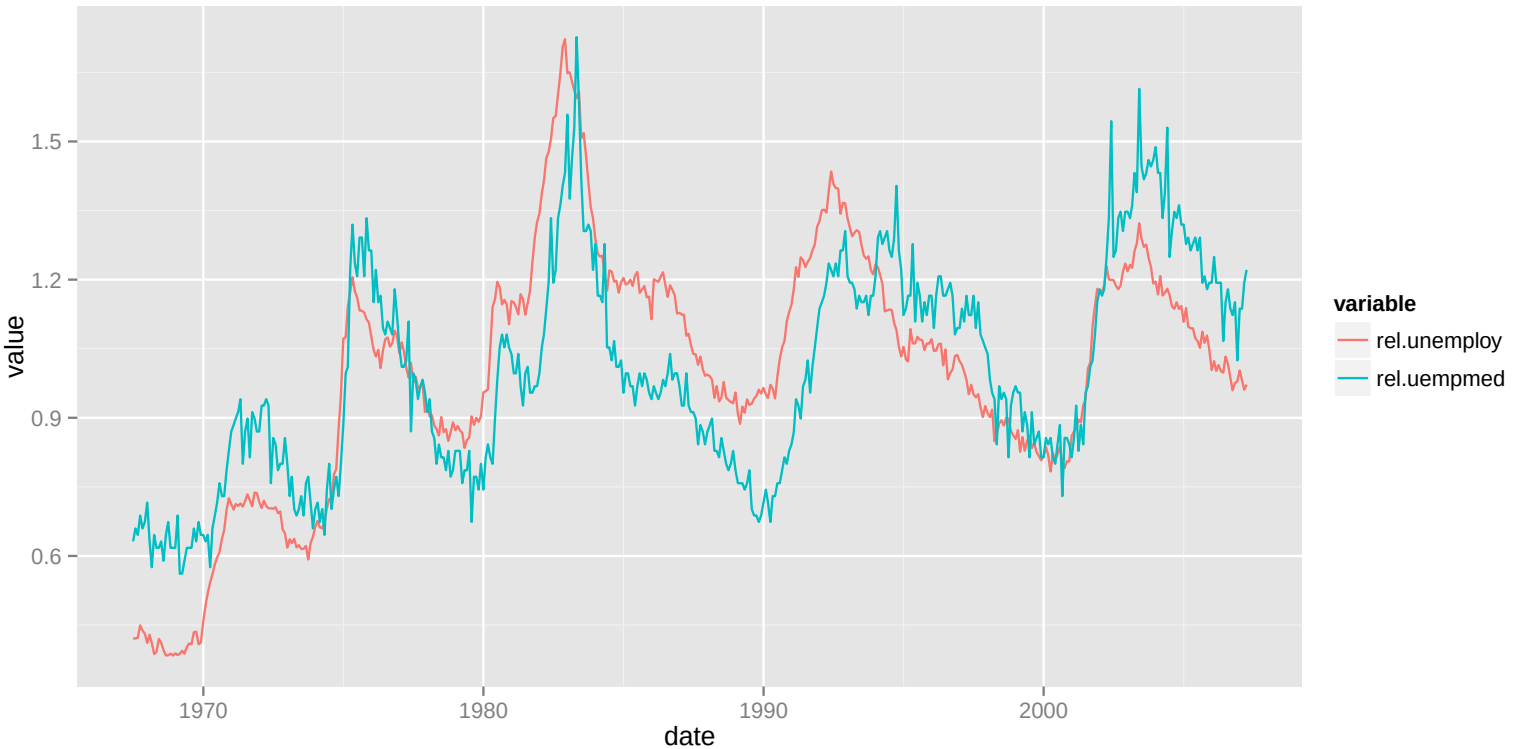
```
library(reshape)
ne <- melt(e, id="date", measure=c("rel.unemploy","rel.uempmed"))
xtable(head(ne))
```

	date	variable	value
1	-916.00	rel.unemploy	0.42
2	-885.00	rel.unemploy	0.42
3	-854.00	rel.unemploy	0.42
4	-824.00	rel.unemploy	0.45
5	-793.00	rel.unemploy	0.44
6	-763.00	rel.unemploy	0.43

See Wickham (2009) section 9.2

"Narrow" data is easier to use with ggplot

```
qplot(date, value, data = ne, geom = "line",
    colour = variable)
```



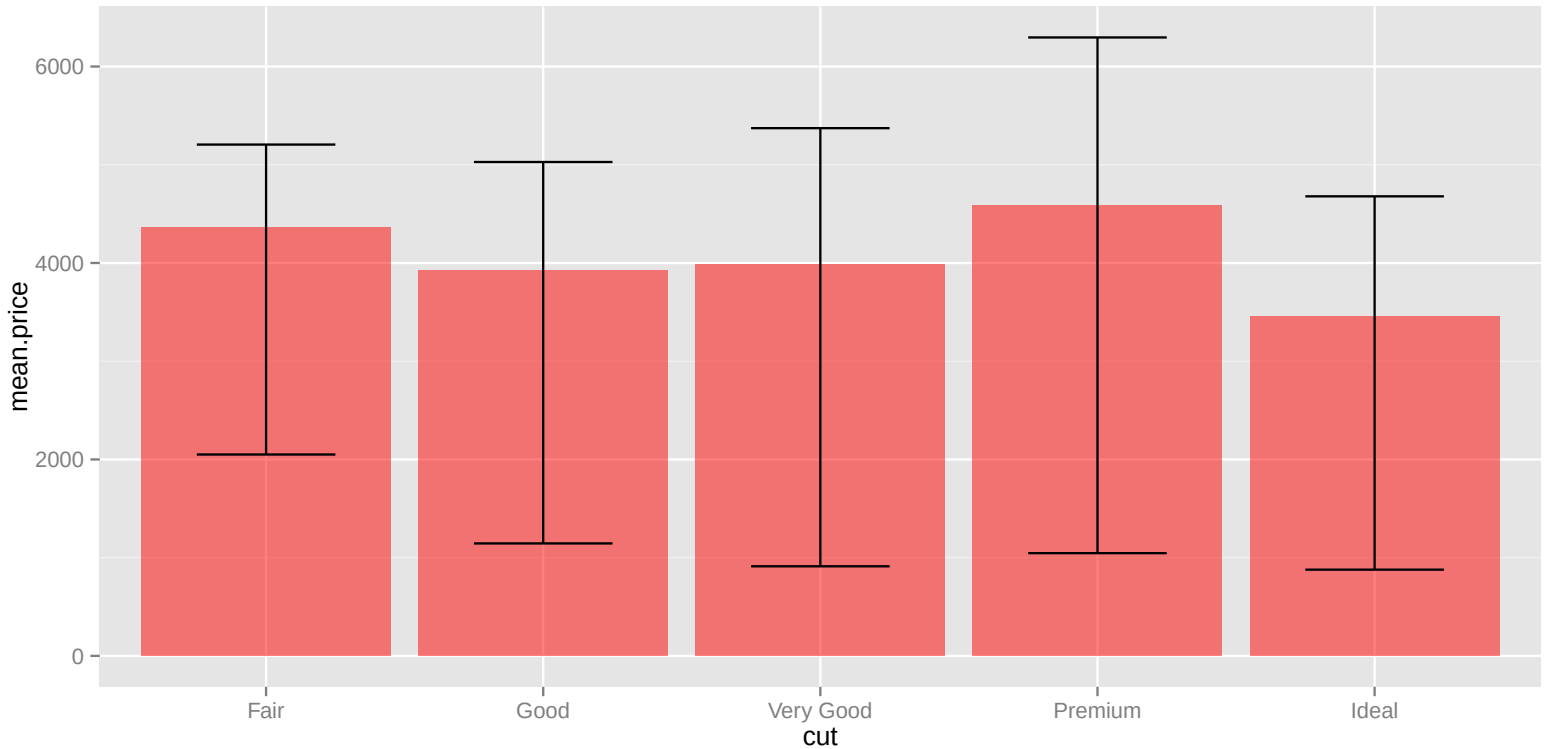
Computing statistical summary tables

```
my_fun <- function(df) {
    data.frame(mean.price = mean(df$price),
        std.dev.price = sd(df$price), std.err.price = sd(df$price)/nrow(df),
        lower.25 = quantile(df$price, c(0.25)),
        upper.75 = quantile(df$price, c(0.75)))
}
dsummary <- ddply(diamonds, .(cut), my_fun)
xtable(dsummary)
```

	cut	mean.price	std.dev.price	std.err.price	lower.25	upper.75
1	Fair	4358.76	3560.39	2.21	2050.25	5205.50
2	Good	3928.86	3681.59	0.75	1145.00	5028.00
3	Very Good	3981.76	3935.86	0.33	912.00	5372.75
4	Premium	4584.26	4349.20	0.32	1046.00	6296.00
5	Ideal	3457.54	3808.40	0.18	878.00	4678.50

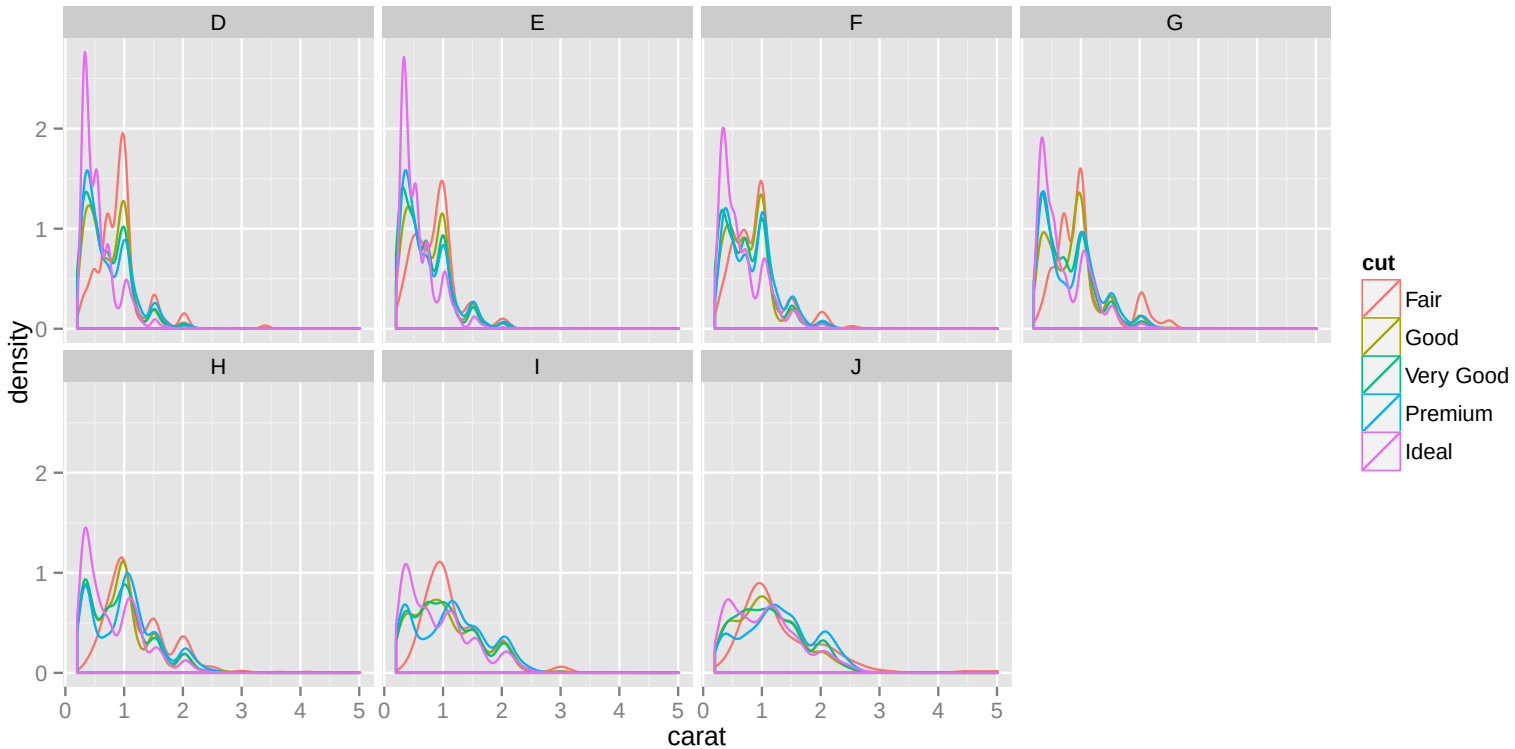
Bar plot with error bars

```
qplot(cut, mean.price, data = dsummary, geom = "bar",
    stat = "identity", fill = I("red"), alpha = I(0.5)) +
    geom_errorbar(aes(ymin = lower.25, ymax = upper.75),
        width = 0.5)
```



Faceting

```
qplot(carat, data = diamonds, geom = "density",
    colour = cut) + facet_wrap(~color, ncol = 4)
```



See Wickam (2009) sections 2.6 and 7.2

Outline

Exporting plots as PDFs

- Chapter 8 of Wickham (2009) has full details
- For interactive use, `ggsave` saves the current plot into a pdf file

```
ggsave(file="output.pdf")
```

- For batch scripts, use the following:

```
pdf(file="output.pdf", width=7, height=7)
qplot(displ, data=d, geom="histogram")
dev.off()
```