Analysing Social Science Data Using R

Class 8

Advanced visualization: built-in methods and custom plots

Michał Bojanowski
mbojan@icm.edu.pl

ICM, University of Warsaw

April 23, 2012
SNS, Warsaw

Outline

Types of visualization methods

Criteria

Data visualization methods can be classified according to the following criteria:

Visualization goal Visualize data to **analyze** it (analytical graphics) or to **present** specific aspects of data, i.e., important results (presentational graphics).

Number of variables How many variables at a time we want to show: one, two, three, or more variables.

Use of time Is the visualization static or dynamic (e.g., movie)?

Interaction Can the user interact with the visualization? Interactive and non-interactive visualizations.

Medium What are the constraints of the medium: size, resolution, color availability, line thickness, interactiveness. E.g. computer screen or paper?

Some examples, by number of variables

Univariate	histogram, stripchart, dotplot, density estimator, piechart, barplot
Bivariate	scatterplot, box-and-whisker, barplot (stacked or not), conditional density plot,
Trivariate	3d extensions of bivariate plots
Multivariate	conditioning plot, scatterplot matrix

Advanced visualization functions

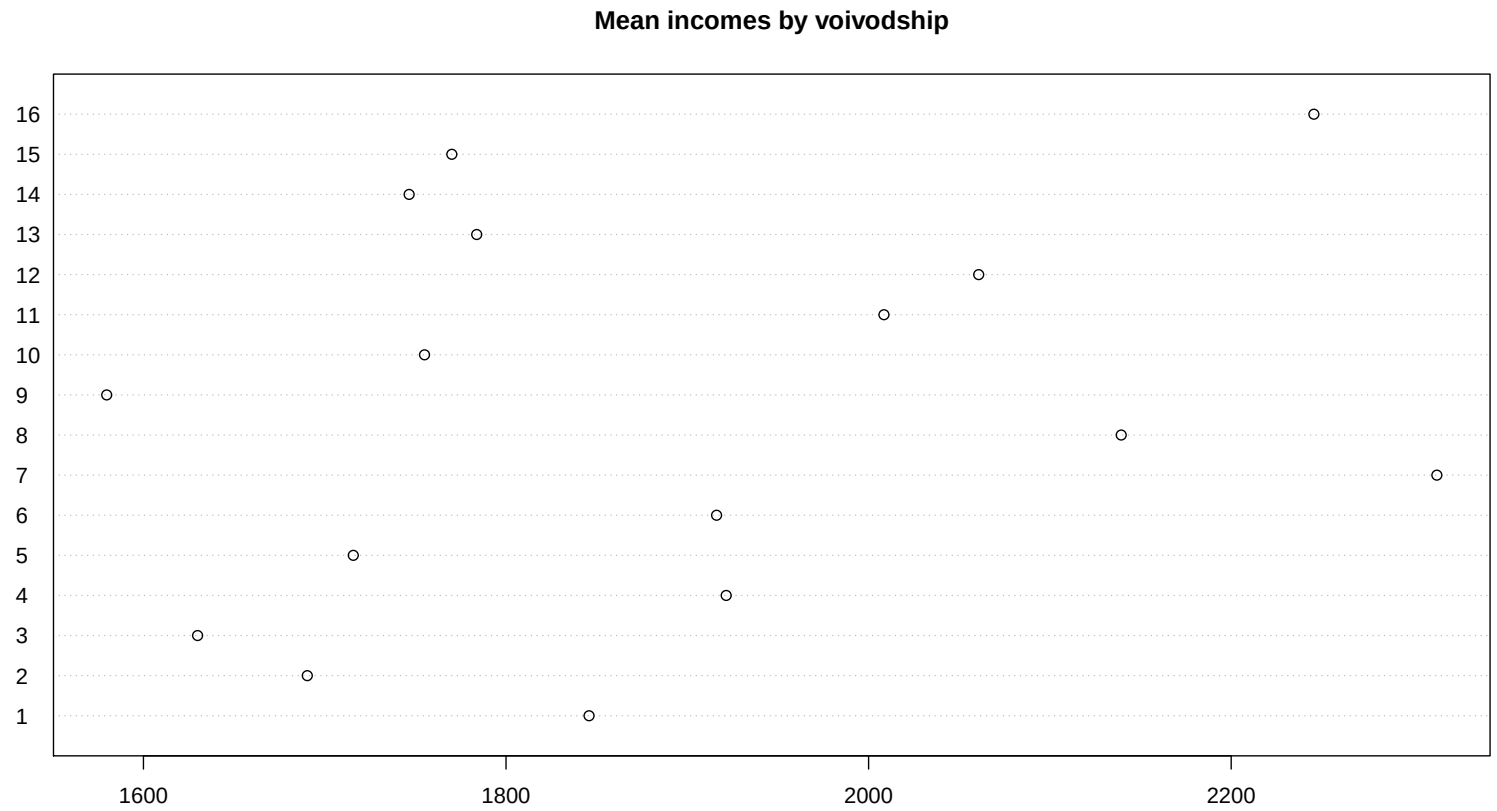
Advanced visualization functions

Univariate

1. Dotplot with `dotchart`

Dotplot

Dotplot is an alternative to barplot



Using dotchart

```
> x <- with(pgss, tapply(hincome, voiev16, mean, na.rm=TRUE))
> dotchart(x, main="Mean incomes by voivodship")
> x
       1        2        3        4        5        6
1845.806 1690.378 1629.866 1921.450 1715.754 1916.138
       7        8        9       10       11       12
2313.450 2139.333 1579.735 1755.047 2008.467 2060.717
      13       14       15       16
1783.827 1746.547 1770.097 2245.617
```

Using dotchart on matrices

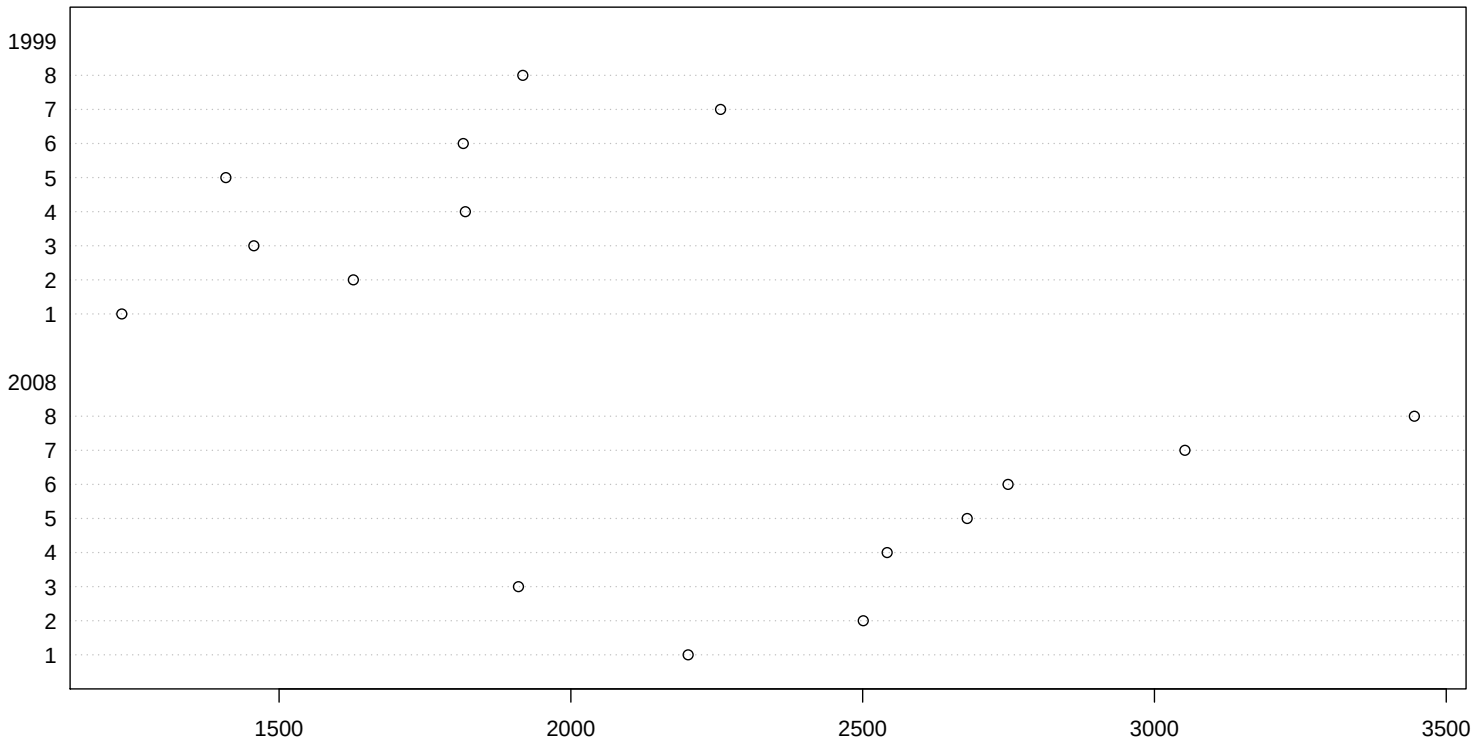
Argument x can be a vector or matrix.

```
> x <- with(pgss, tapply(hincome, list(size=size, year=pgssyear)
+                        mean, na.rm=TRUE))
> x
```

```
    year
size     1999     2008
   1 1230.477 2200.916
   2 1627.259 2501.007
   3 1456.779 1910.181
   4 1819.200 2541.961
   5 1408.791 2679.110
   6 1815.618 2749.149
   7 2256.515 3052.221
   8 1917.618 3445.373
```

Using dotchart on matrices contd.

```
> dotchart(x)
```



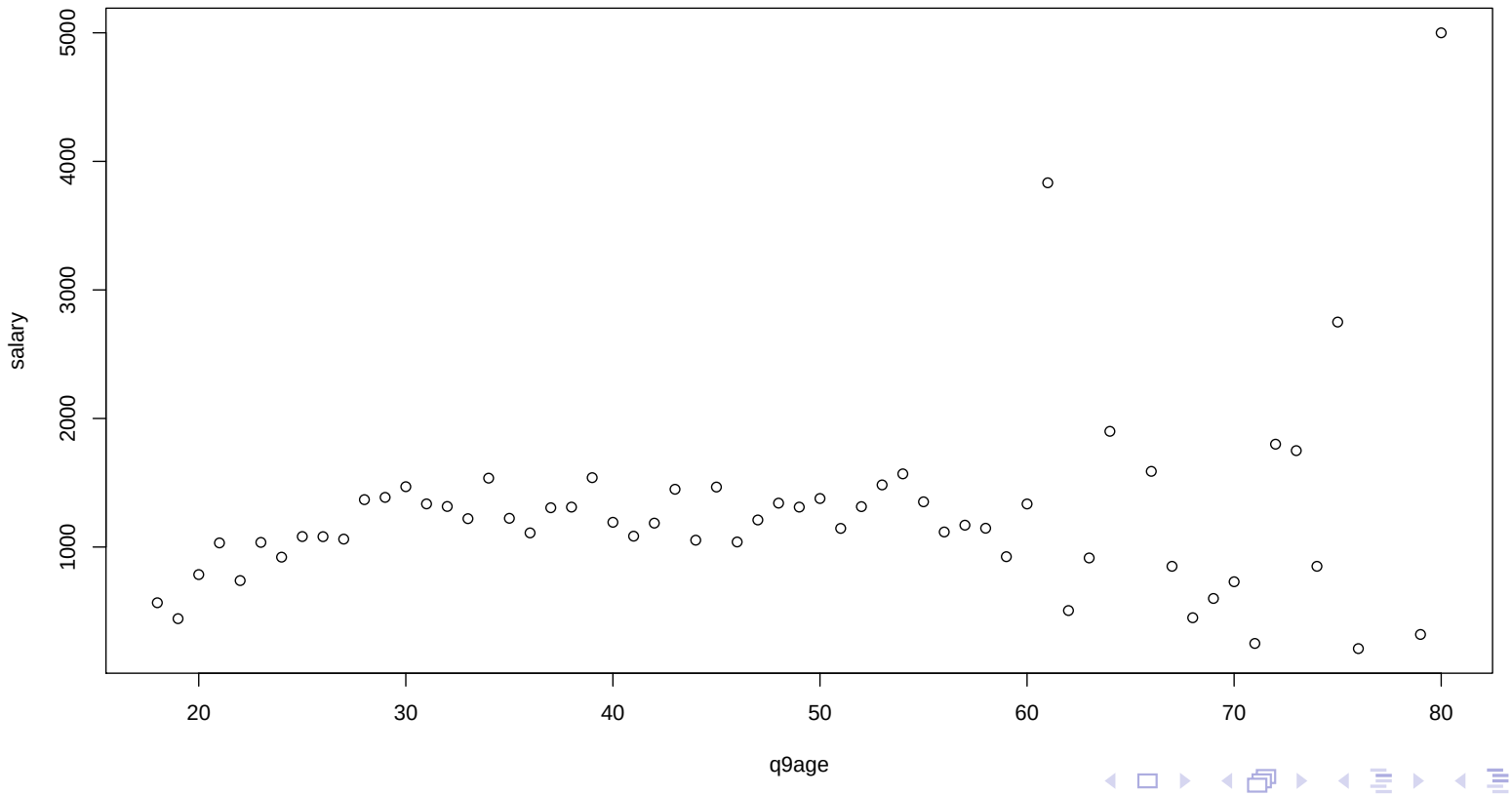
Advanced visualization functions

Bivariate

1. Scatterplot
2. Conditional density plot

Scatterplot

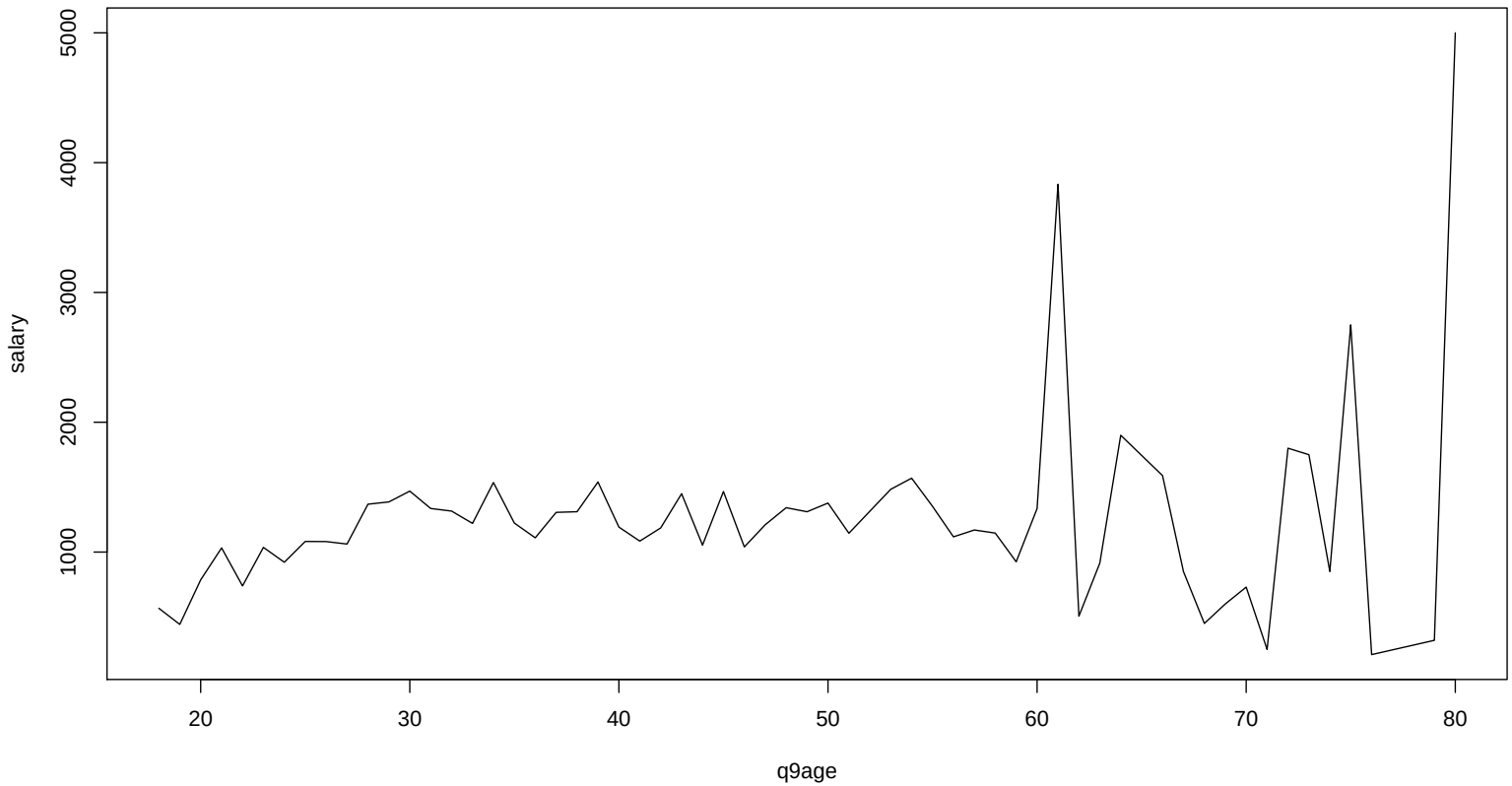
Points in 2d



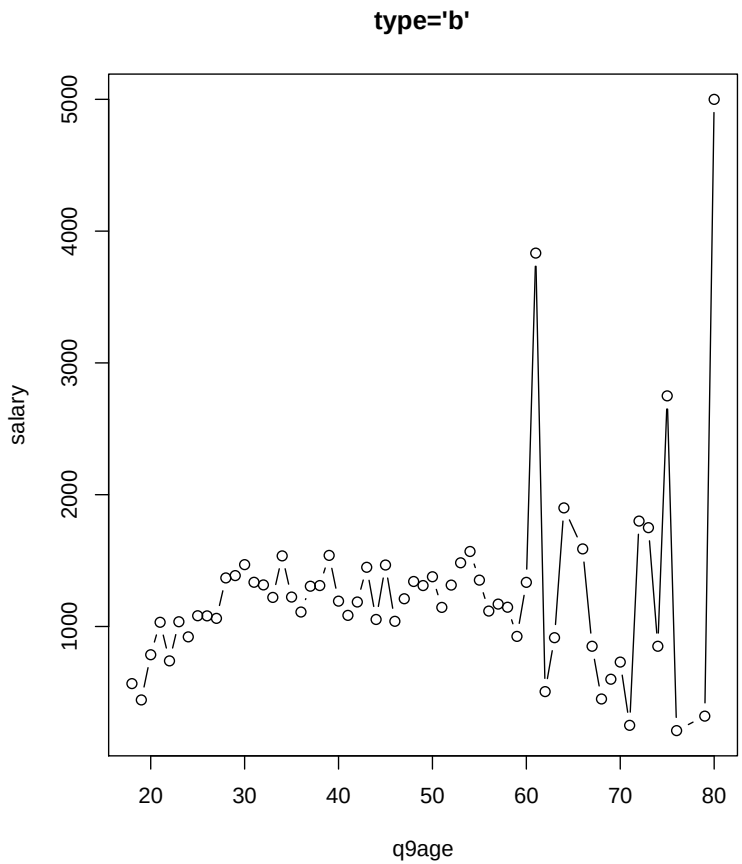
Using plot

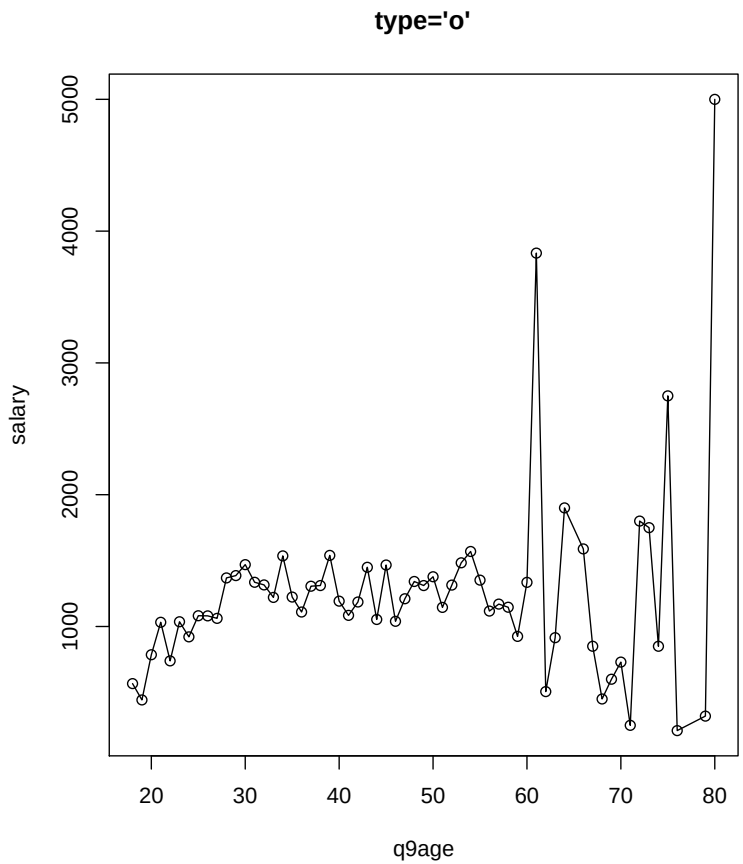
```
> ageg <- with(pgss, cut(q9age,
+                        c(-Inf, seq(20, 90, by=5), Inf),
+                        right=FALSE))
> a <- aggregate(salary ~ q9age, mean, na.rm=TRUE, data=pgss)
> with(a, plot(q9age, salary))
```

Using `plot`: argument `type="l"`



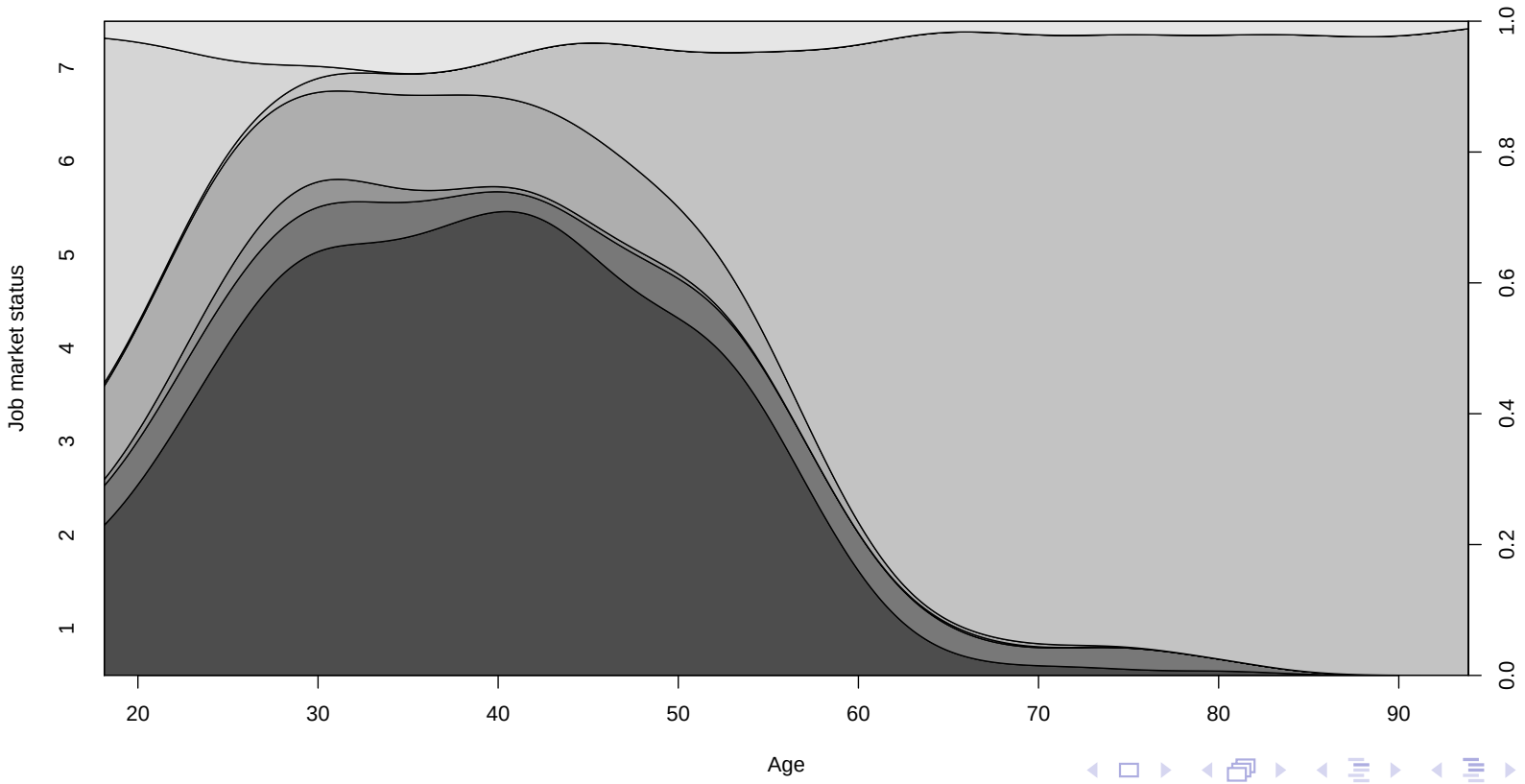
Using `plot`: argument `type="b"` lub `"o"`





Conditional density plot

How categorical variable depends on continuous variable.



Using cdplot

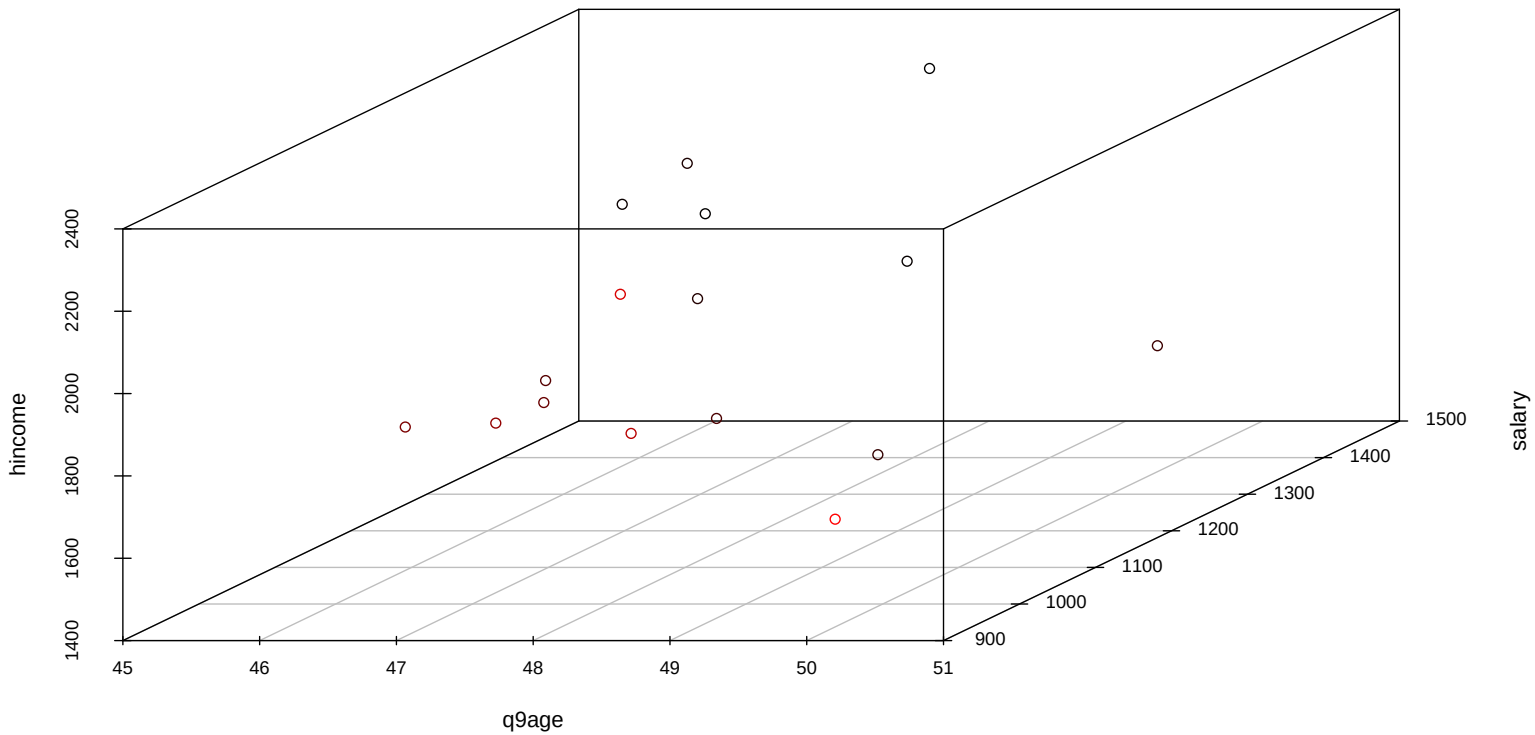
```
> cdplot( factor(q18st) ~ q9age, data=pgss,
+         xlab="Age", ylab="Job market status")
```

Advanced visualization functions

Trivariate

Three-dimensional scatterplot: `scatterplot3d` in package scatterplot3d

Scatterplot 3d



Using scatterplot3d

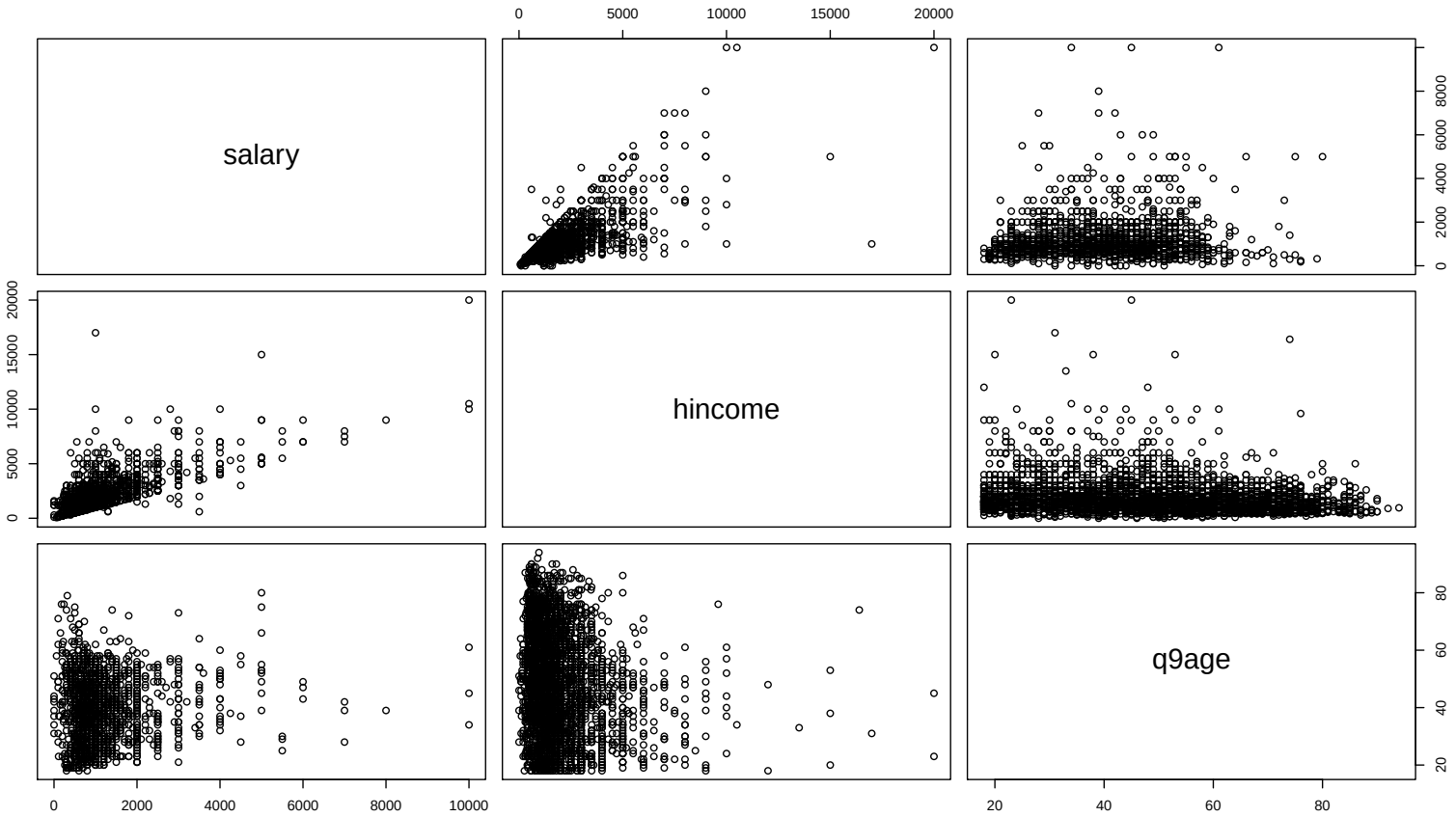
```
> # install.packages("scatterplot3d")
> library(scatterplot3d)
> m <- with(pgss, aggregate( cbind(q9age, salary, hincome),
+           list(voiev=voiev16), mean, na.rm=TRUE))
> scatterplot3d(m[,-1], highlight.3d=TRUE)
> head(m)
  voiev    q9age    salary  hincome
1     1 50.87313 1204.3111 1845.806
2     2 47.78261 1180.3896 1690.378
3     3 49.80102  973.2099 1629.866
4     4 47.26923 1247.8571 1921.450
5     5 45.79583 1128.4314 1715.754
6     6 48.20000 1356.0900 1916.138
```

Advanced visualization functions

Multivariate

1. Scatterplot matrix: `plotting` data frames
2. Conditioning plot: `coplot`

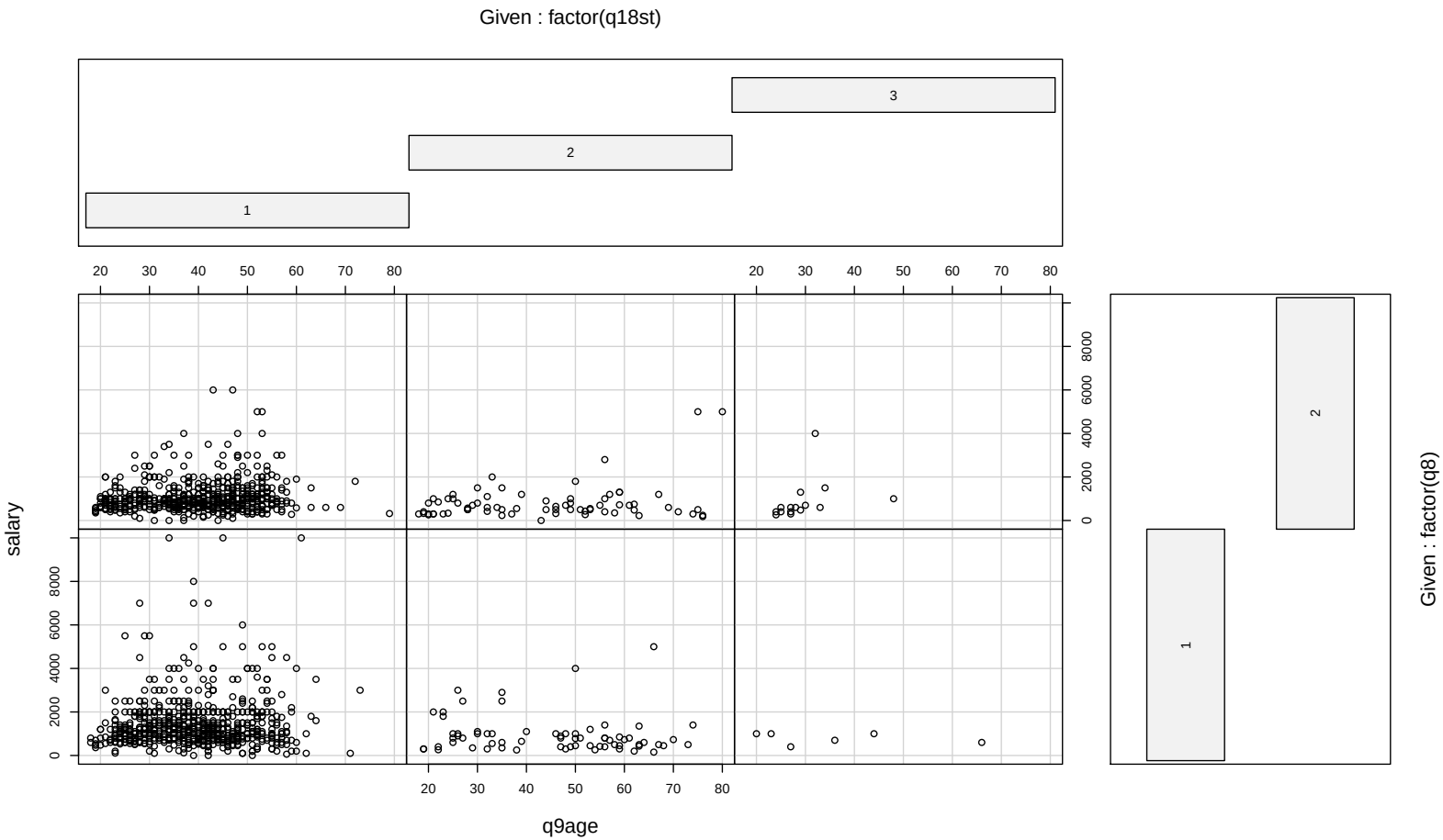
Scatterplot matrix



Creating scatterplot matrix

```
> d <- pgss[,c("salary", "hincome", "q9age")]
> plot(d)
```

Conditioning plot



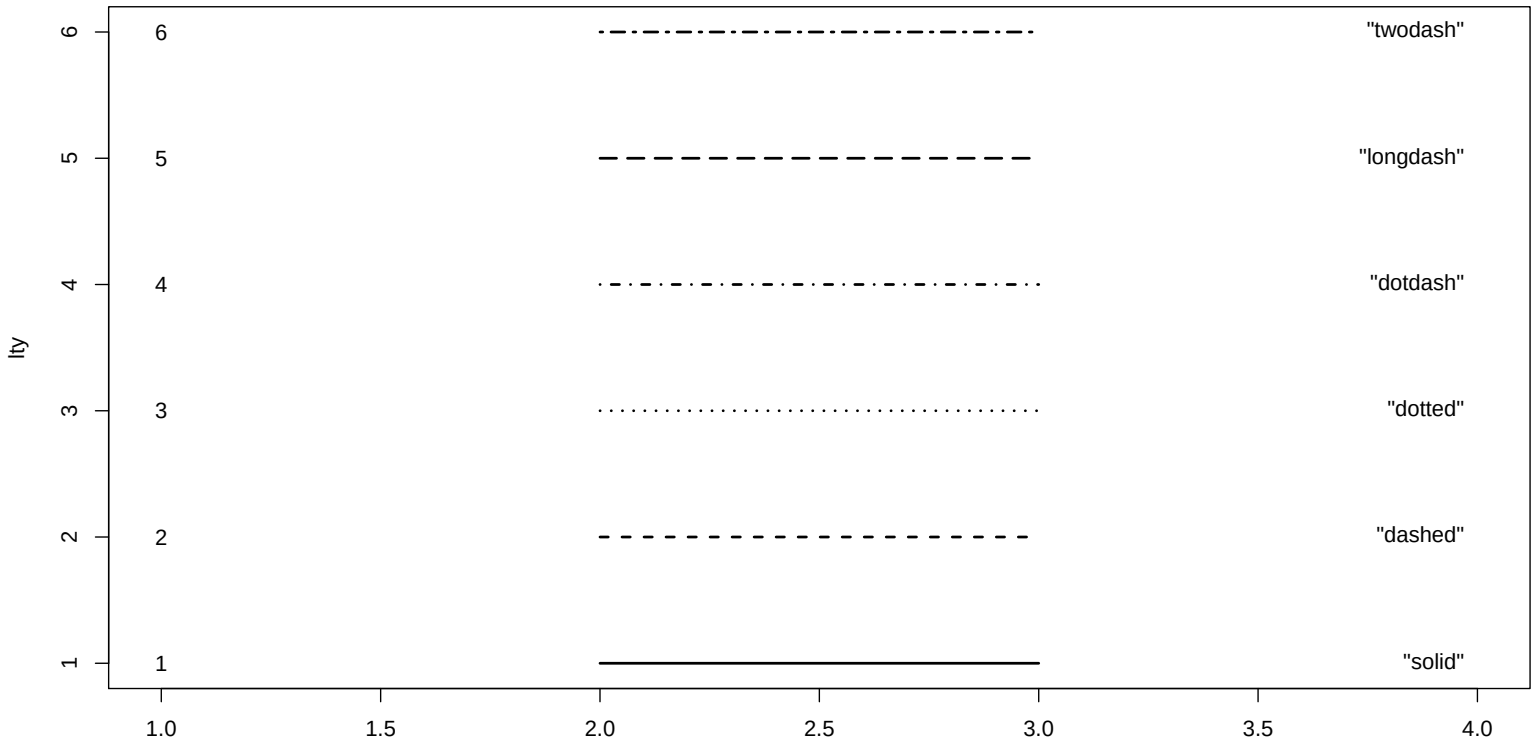
Using coplot

```
> d <- pgss[,c("salary", "q9age", "q8", "q18st")]
> d <- d[complete.cases(d),]
> coplot( salary ~ q9age | factor(q18st) * factor(q8), data=d)
```

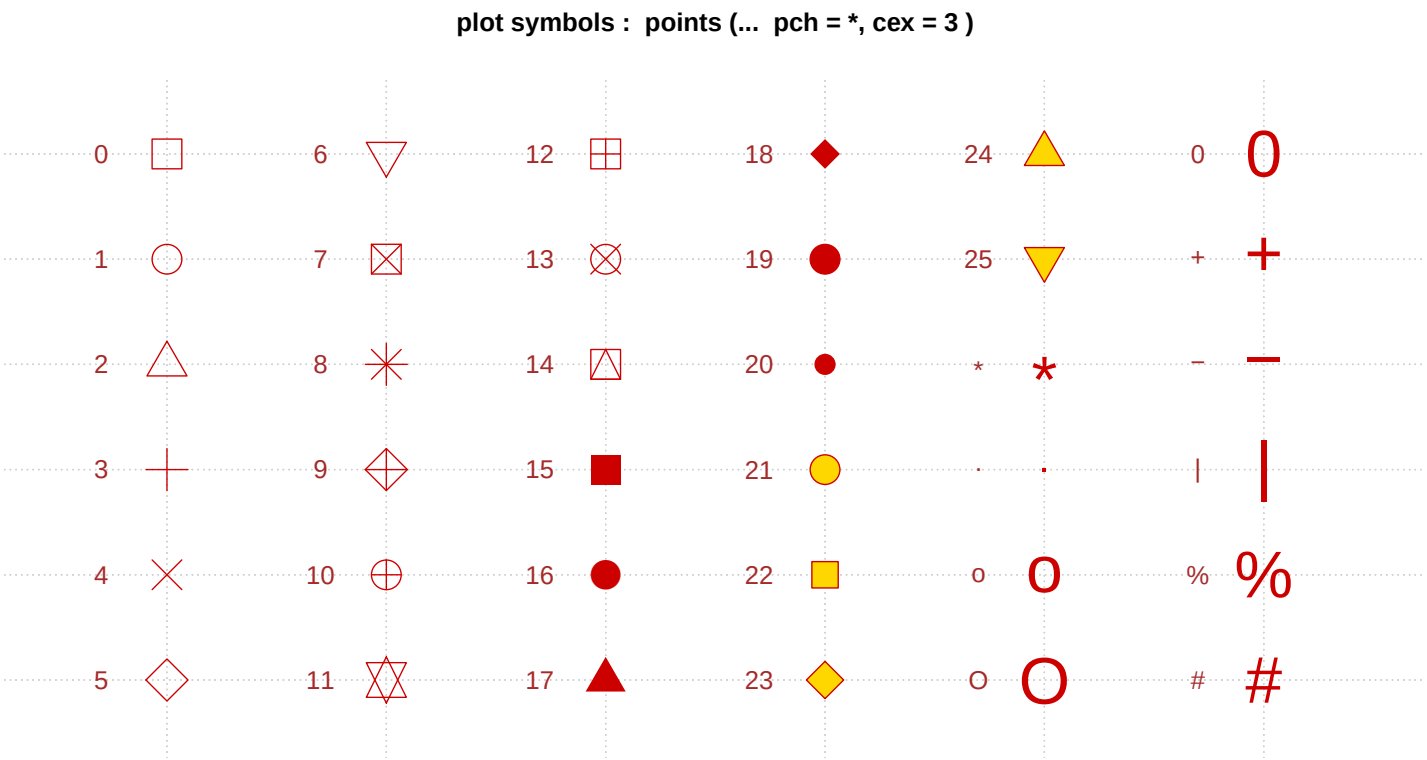
Customizing plots

Line types: argument `lty`

Specifying line type by name or number to argument `lty`



Point types: argument pch



Colors: arguments `col`, `bg` etc.

Colors can be specified in three basic ways

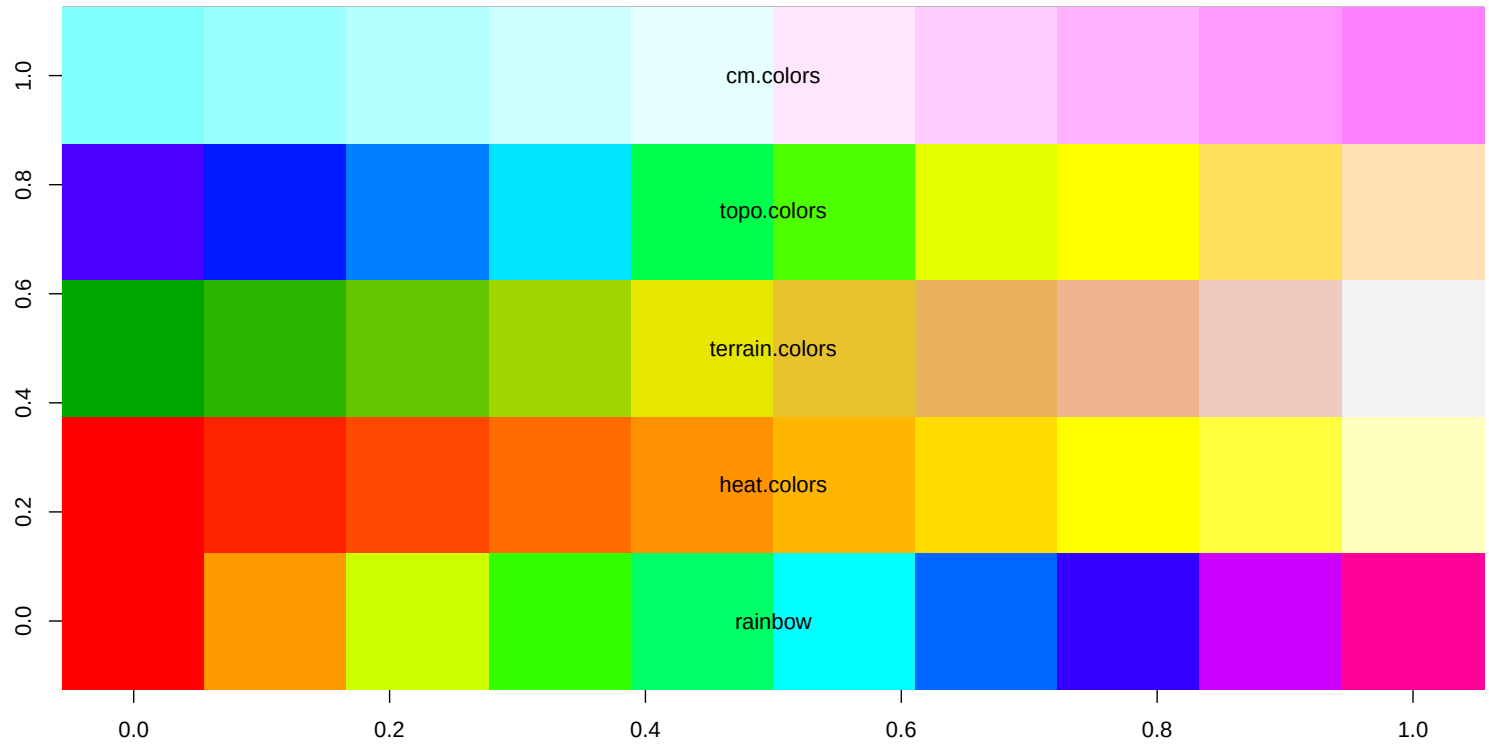
by name Using built-in color names. See the output of `colors()` for available color names.

by number Indexing default palette of colors. See output of `palette()`. Palette can be changed.

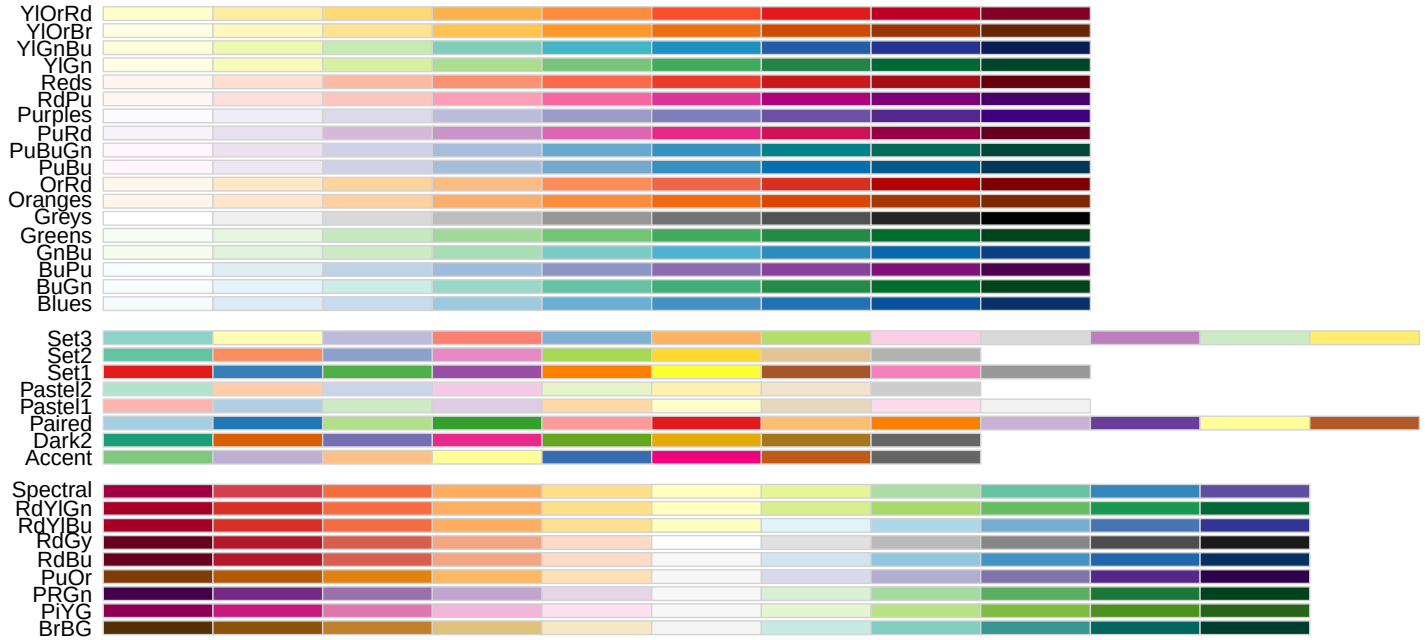
by hexcode For example, #000000 is black, `#ffffff` is white.

Color ramps

There are functions to create sequences of colors (vectors)



Palettes in package RColorBrewer



Creating custom plots

Steps in creating custom plots

1. Setting-up coordinate system
2. Plotting data with `points`, `lines`, `text`, etc.
3. Adding legends
4. Adding titles

Some data

Let's plot age profiles for salaries for men and women separately

```
> m <- with(pgss, tapply(salary, list(age=q9age, gender=q8),
+                        mean, na.rm=TRUE))
> head(m)
```

```
    gender
age          1        2
  18  700.0000 300.0000
  19  445.0000 440.7143
  20  935.0000 711.0000
  21 1580.0000 887.7895
  22  652.3333 782.5000
  23 1173.3333 871.3333
```

Setting-up coordinates

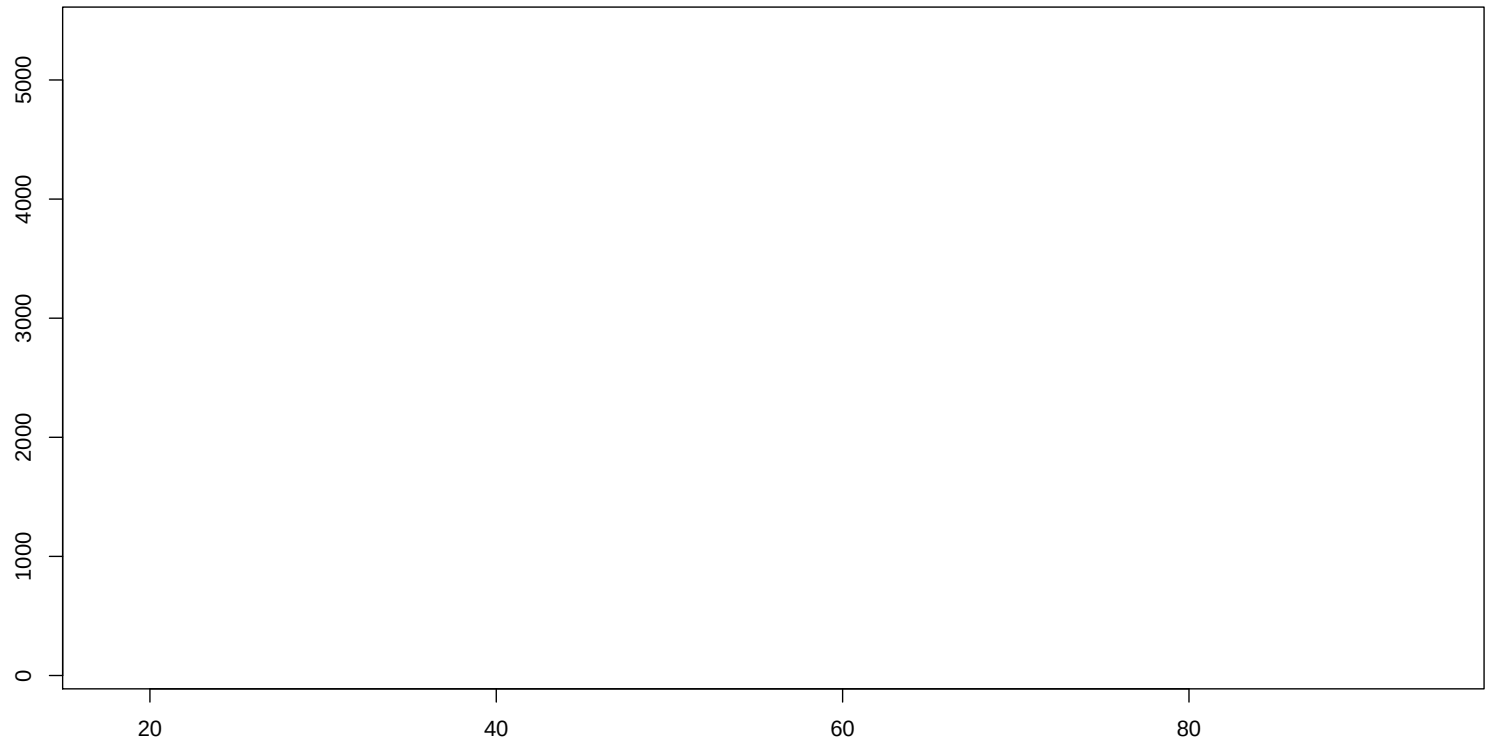
Using `xlim` and `ylim` arguments

```
> plot(NULL, xlim=c(1, 100), ylim=c(0, 1), ann=FALSE)
```

or based on data with `type="n"` (nothing) and no axis annotation

```
> plot( range(pgss$q9age, na.rm=TRUE), range(m, na.rm=TRUE),
+       type="n", ann=FALSE)
```

Setting-up coordinates



Plotting data

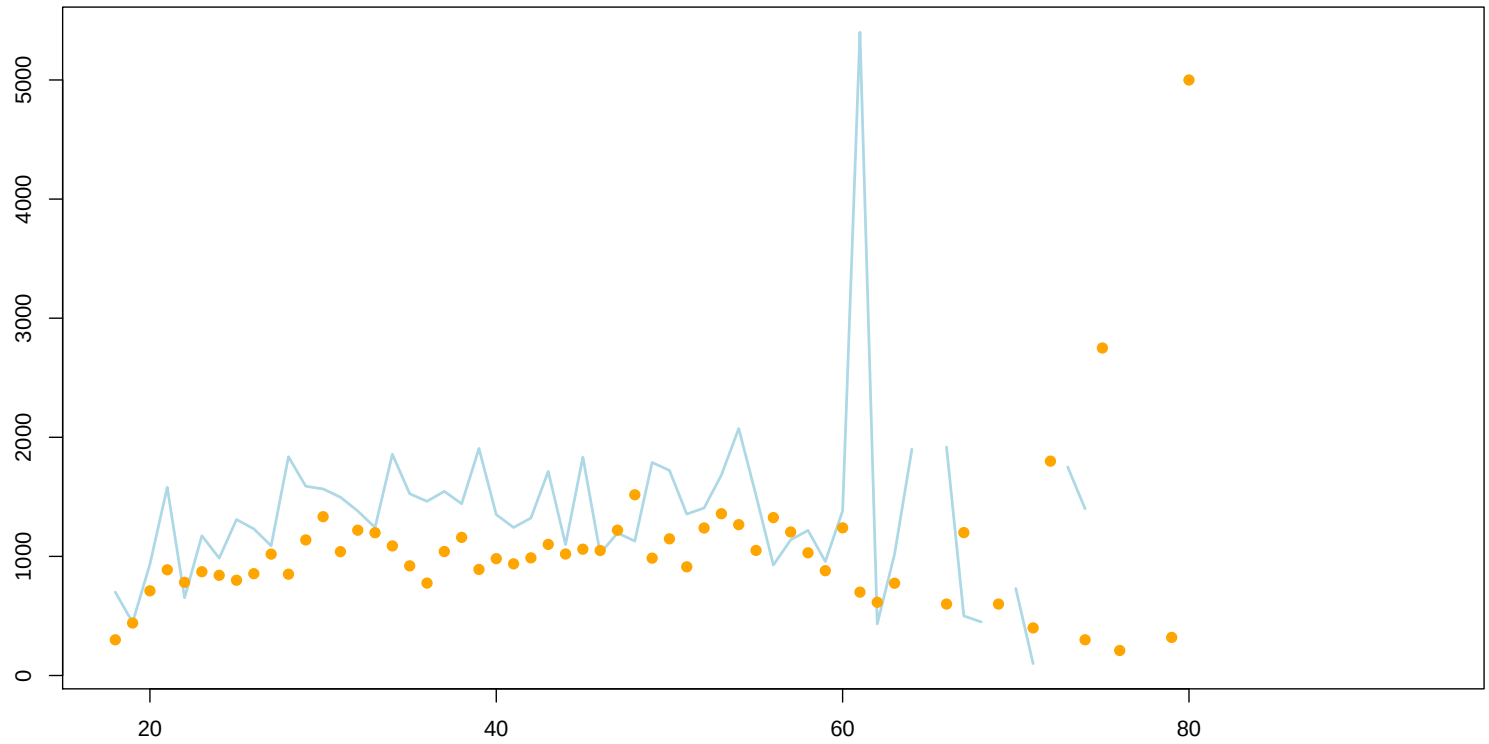
Function to add lines/points/text to existing plot

```
lines(x, y, ...)
points(x, y, ...)
text(x, y, labels, ...)
```

Add line for men and points for men.

```
> lines( as.numeric(rownames(m)), m[,1],
+        lty=1, lwd=2, col="lightblue")
> points( as.numeric(rownames(m)), m[,2],
+         pch=19, col="orange")
```

Plotting data



Adding legend and titles

Use `legend` and `title`:

```
> legend("topleft", lty=c(1, NA), pch=c(NA, 19),
+        col=c("lightblue", "orange"),
+        legend=c("Men", "Women"), bty="n", cex=2)
> title( main="Salary age profiles for men and women",
+       xlab="Age", ylab="Salary")
```

With legend and titles