Introduction to Data Visualization

Data Analysis with R and Python

Deepayan Sarkar

Data Visualization

- Important component of data analysis

Data Visualization

- Important component of data analysis
- Main purposes
  - Exploration
  - Presentation

Data Visualization

- Important component of data analysis
- Main purposes
  - Exploration
  - Presentation
- Learning objectives
  - What kind of visualization to use
  - How to create them

Example datasets: `airquality` (size: small)

```
str(airquality) # built-in dataset
```

```
'data.frame':    153 obs. of  6 variables:
 $ Ozone  : int  41 36 12 18 NA 28 23 19 8 NA ...
 $ Solar.R: int  190 118 149 313 NA NA 299 99 19 194 ...
 $ Wind   : num  7.4 8 12.6 11.5 14.3 14.9 8.6 13.8 20.1 8.6 ...
 $ Temp   : int  67 72 74 62 56 66 65 59 61 69 ...
 $ Month  : int  5 5 5 5 5 5 5 5 5 5 ...
 $ Day    : int  1 2 3 4 5 6 7 8 9 10 ...
```

Example datasets: `airquality` (size: small)

```
head(airquality, 15)
```

```
   Ozone Solar.R Wind Temp Month Day
1     41     190  7.4   67     5   1
2     36     118  8.0   72     5   2
3     12     149 12.6   74     5   3
4     18     313 11.5   62     5   4
5     NA      NA 14.3   56     5   5
6     28      NA 14.9   66     5   6
7     23     299  8.6   65     5   7
8     19      99 13.8   59     5   8
9      8      19 20.1   61     5   9
10    NA     194  8.6   69     5  10
11     7      NA  6.9   74     5  11
12    16     256  9.7   69     5  12
13    11     290  9.2   66     5  13
14    14     274 10.9   68     5  14
15    18      65 13.2   58     5  15
```

Example datasets: gapminder (size: moderate)

```
gapminder <- read.table("data/gapminder.tsv", sep = "\t", header = TRUE)
str(gapminder)
```

```
'data.frame':    1698 obs. of  6 variables:
 $ country  : chr  "Afghanistan" "Afghanistan" "Afghanistan" "Afghanistan" ...
 $ continent: chr  "Asia" "Asia" "Asia" "Asia" ...
 $ year     : int  1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
 $ lifeExp  : num  28.8 30.3 32 34 36.1 ...
 $ pop      : int  8425333 9240934 10267083 11537966 13079460 14880372 12881816 13867957 16317921 22227415 ...
 $ gdpPercap: num  779 821 853 836 740 ...
```

Example datasets: gapminder (size: moderate)

```
subset(gapminder, country == "Australia")
```

```
     country continent year lifeExp      pop gdpPercap
61 Australia   Oceania 1952  69.120  8691212  10039.60
62 Australia   Oceania 1957  70.330  9712569  10949.65
63 Australia   Oceania 1962  70.930 10794968  12217.23
64 Australia   Oceania 1967  71.100 11872264  14526.12
65 Australia   Oceania 1972  71.930 13177000  16788.63
66 Australia   Oceania 1977  73.490 14074100  18334.20
67 Australia   Oceania 1982  74.740 15184200  19477.01
68 Australia   Oceania 1987  76.320 16257249  21888.89
69 Australia   Oceania 1992  77.560 17481977  23424.77
70 Australia   Oceania 1997  78.830 18565243  26997.94
71 Australia   Oceania 2002  80.370 19546792  30687.75
72 Australia   Oceania 2007  81.235 20434176  34435.37
```

Example datasets: NHANES (size: somewhat large)

```
library(package = "NHANES")
str(NHANES)
```

```
Classes 'tbl_df', 'tbl' and 'data.frame':    10000 obs. of  76 variables:
 $ ID              : int  51624 51624 51624 51625 51630 51638 51646 51647 51647 51647 ...
 $ SurveyYr        : Factor w/ 2 levels "2009_10","2011_12": 1 1 1 1 1 1 1 1 1 1 ...
 $ Gender          : Factor w/ 2 levels "female","male": 2 2 2 2 1 2 2 1 1 1 ...
 $ Age             : int  34 34 34 4 49 9 8 45 45 45 ...
 $ AgeDecade       : Factor w/ 8 levels " 0-9"," 10-19",..: 4 4 4 1 5 1 1 5 5 5 ...
 $ AgeMonths       : int  409 409 409 49 596 115 101 541 541 541 ...
 $ Race1           : Factor w/ 5 levels "Black","Hispanic",..: 4 4 4 5 4 4 4 4 4 4 ...
 $ Race3           : Factor w/ 6 levels "Asian","Black",..: NA NA NA NA NA NA NA NA NA NA ...
 $ Education       : Factor w/ 5 levels "8th Grade","9 - 11th Grade",..: 3 3 3 NA 4 NA NA 5 5 5 ...
 $ MaritalStatus   : Factor w/ 6 levels "Divorced","LivePartner",..: 3 3 3 NA 2 NA NA 3 3 3 ...
 $ HHIncome        : Factor w/ 12 levels " 0-4999"," 5000-9999",..: 6 6 6 5 7 11 9 11 11 11 ...
 $ HHIncomeMid     : int  30000 30000 30000 22500 40000 87500 60000 87500 87500 87500 ...
 $ Poverty         : num  1.36 1.36 1.36 1.07 1.91 1.84 2.33 5 5 5 ...
 $ HomeRooms       : int  6 6 6 9 5 6 7 6 6 6 ...
 $ HomeOwn         : Factor w/ 3 levels "Own","Rent","Other": 1 1 1 1 2 2 1 1 1 1 ...
 $ Work            : Factor w/ 3 levels "Looking","NotWorking",..: 2 2 2 NA 2 NA NA 3 3 3 ...
 $ Weight          : num  87.4 87.4 87.4 17 86.7 29.8 35.2 75.7 75.7 75.7 ...
```

The goal of data visualization

- Visualizations help us study relationships
- This is enabled by comparison

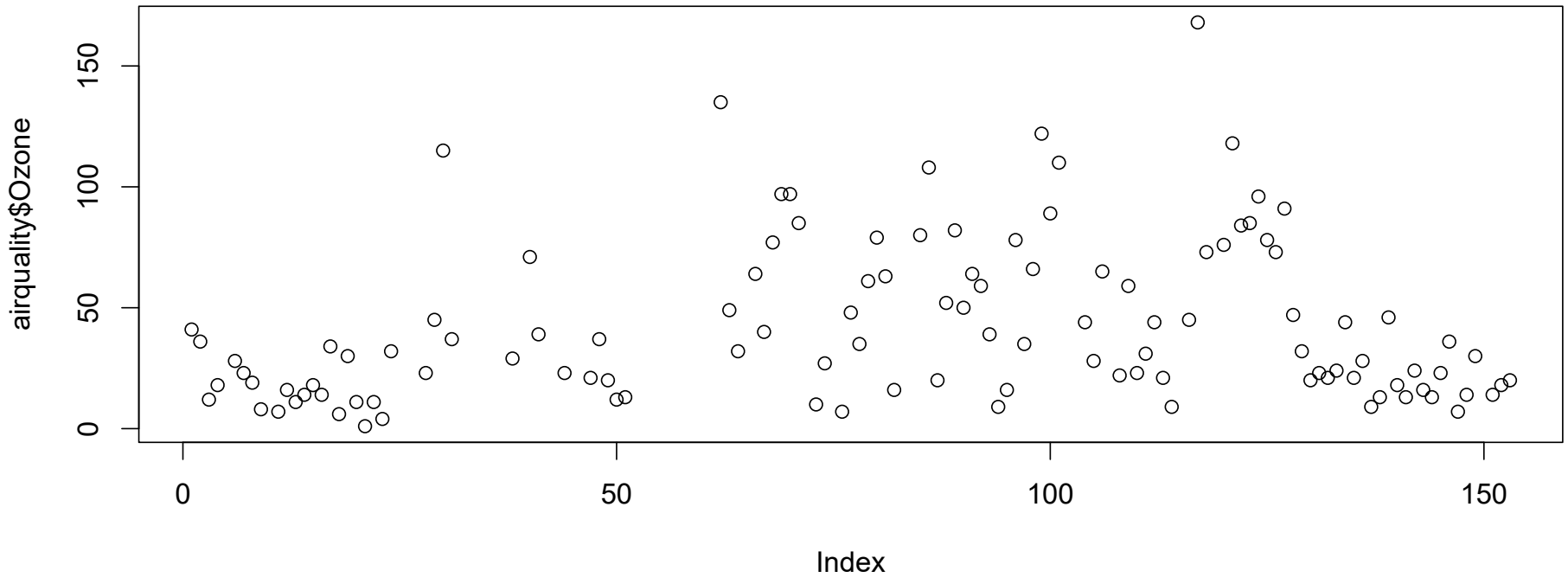
What do we study using visualization?

- Univariate distributions
- Bivariate and trivariate (generally multivariate) relationships
- Special case: Relationship with time (time-series) or space (spatial)

Univariate Data

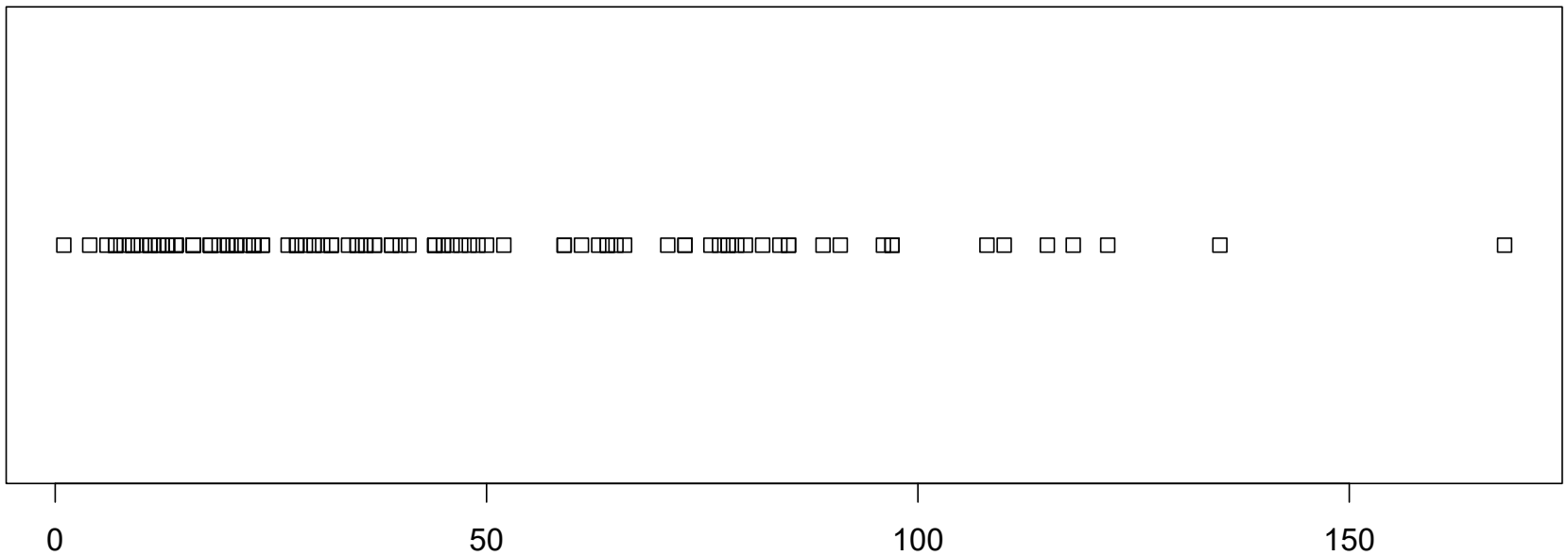
The `plot()` function

```
plot(airquality$Ozone)
```



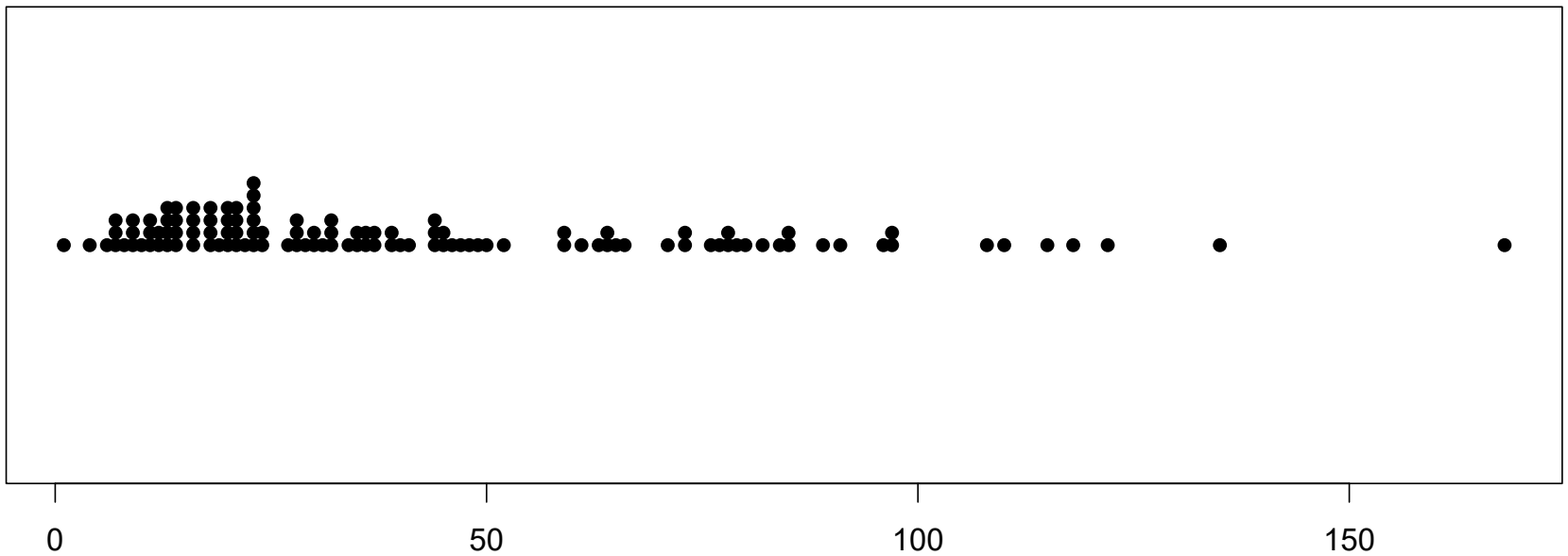
Univariate distributions: strip charts or dot plots

```
stripchart(airquality$Ozone)
```



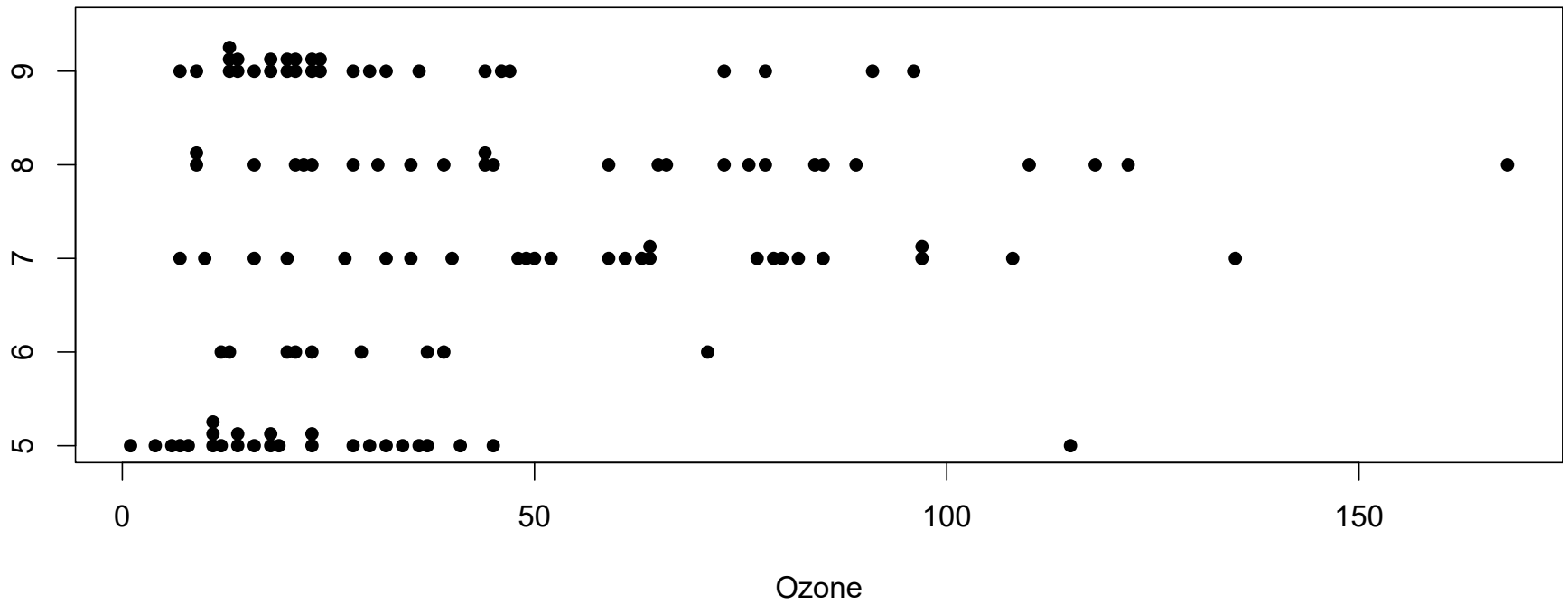
Univariate distributions: strip charts or dot plots

```
stripchart(airquality$Ozone, method = "stack", pch = 16)
```



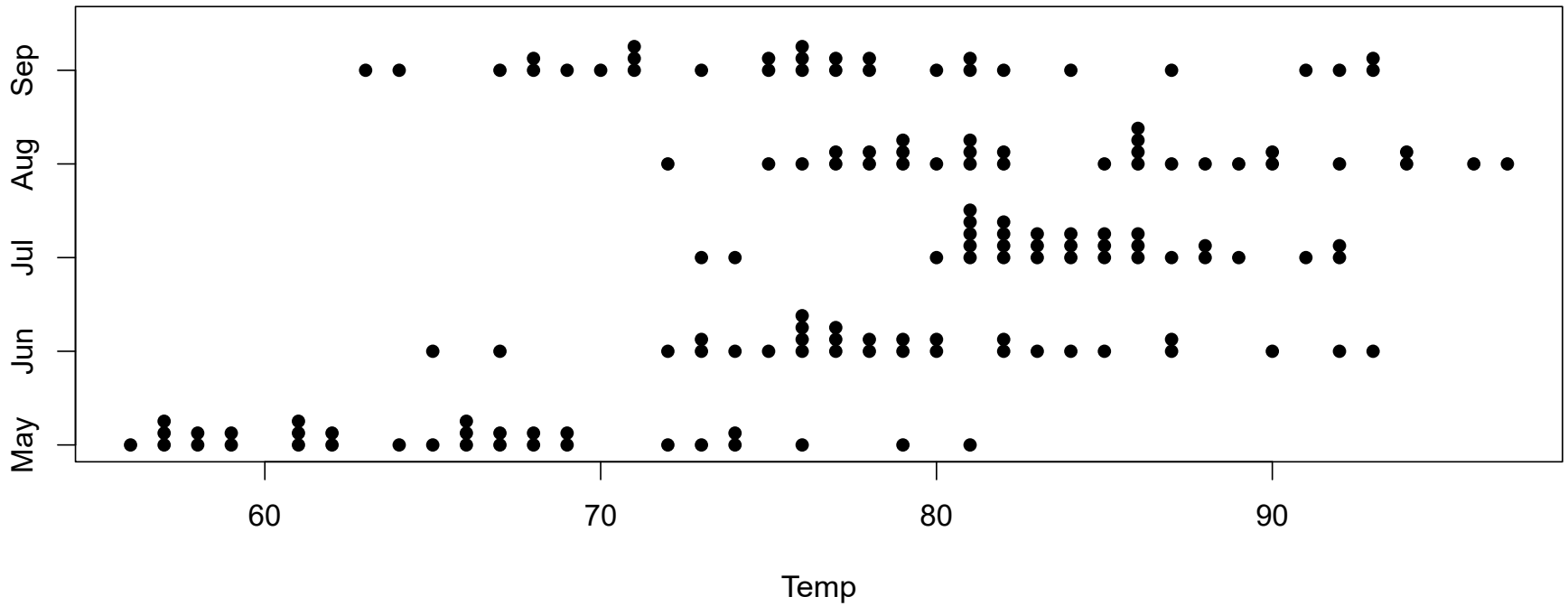
Univariate distributions: comparative strip charts

```
stripchart(Ozone ~ factor(Month), data = airquality,
           method = "stack", pch = 16)
```



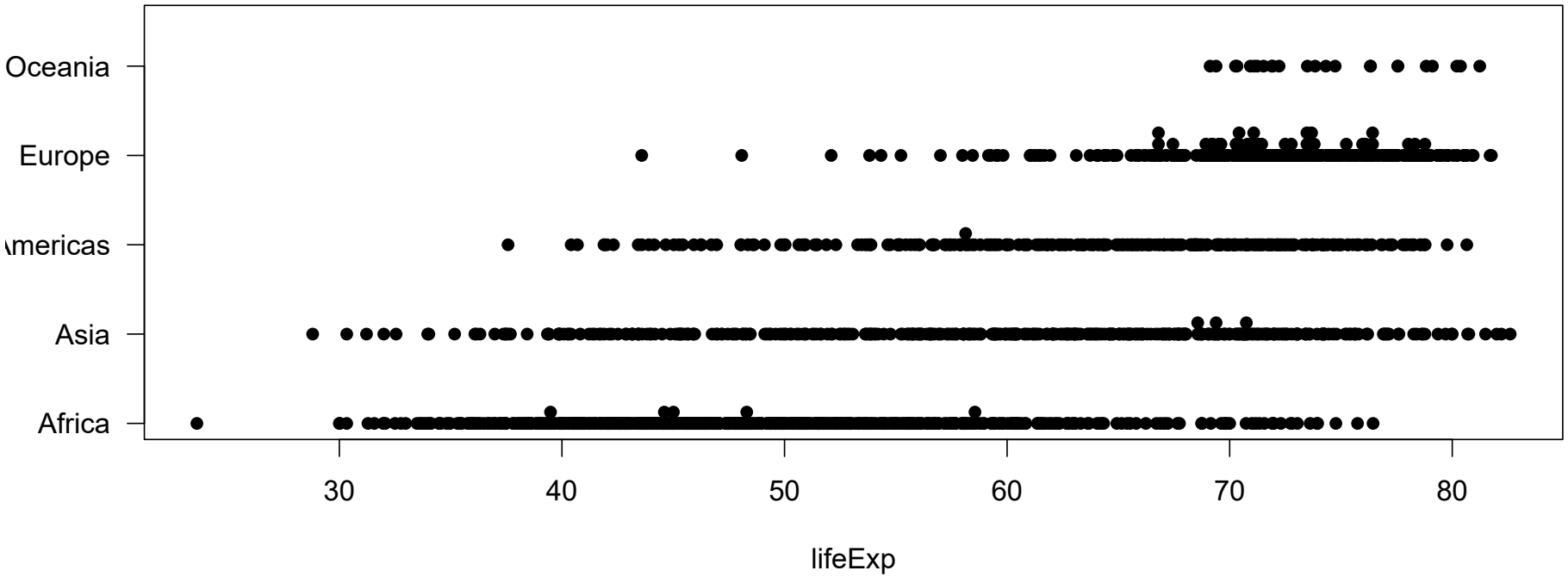
Univariate distributions: comparative strip charts

```
airquality$Month <- factor(airquality$Month, labels = month.abb[5:9])
stripchart(Temp ~ factor(Month, labels = month.abb[5:9]),
           data = airquality, method = "stack", pch = 16)
```



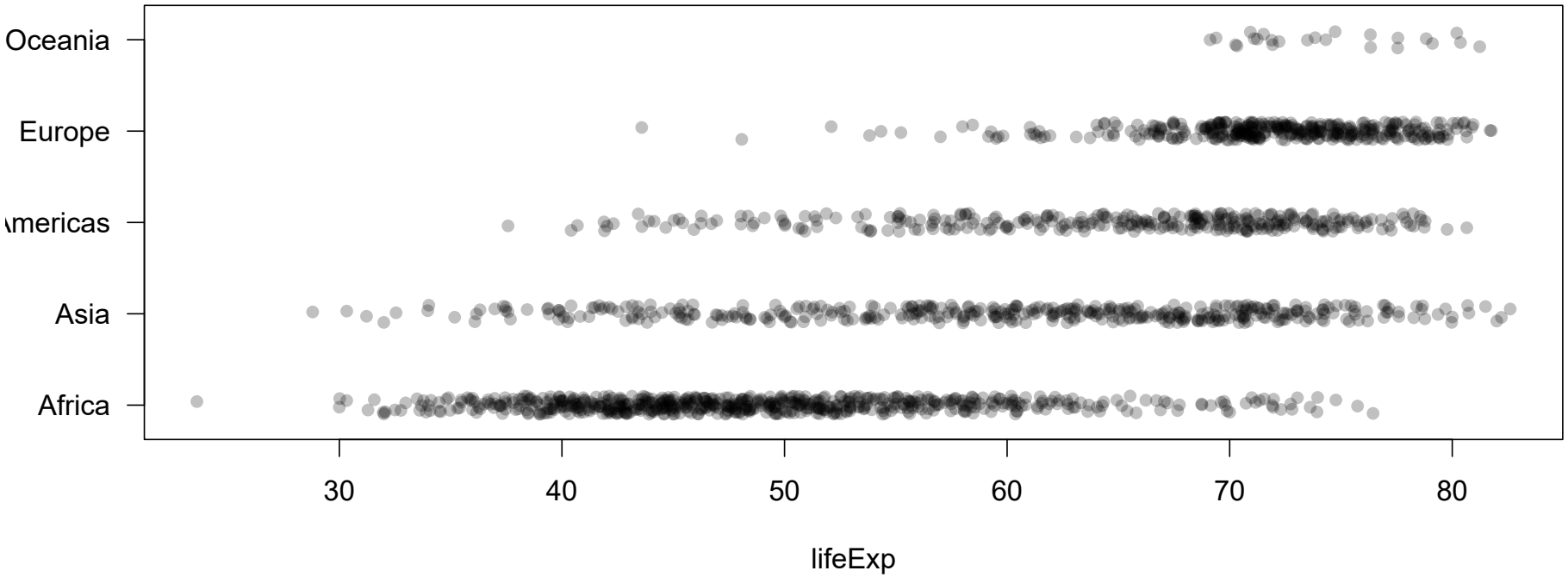
Univariate distributions: comparative strip charts

```
stripchart(lifeExp ~ reorder(continent, lifeExp), data = gapminder,
           method = "stack", pch = 16, las = 1)
```



Univariate distributions: comparative strip charts

```
stripchart(lifeExp ~ reorder(continent, lifeExp), data = gapminder,
           method = "jitter", pch = 16, las = 1,
           col = rgb(0, 0, 0, alpha = 0.25))
```

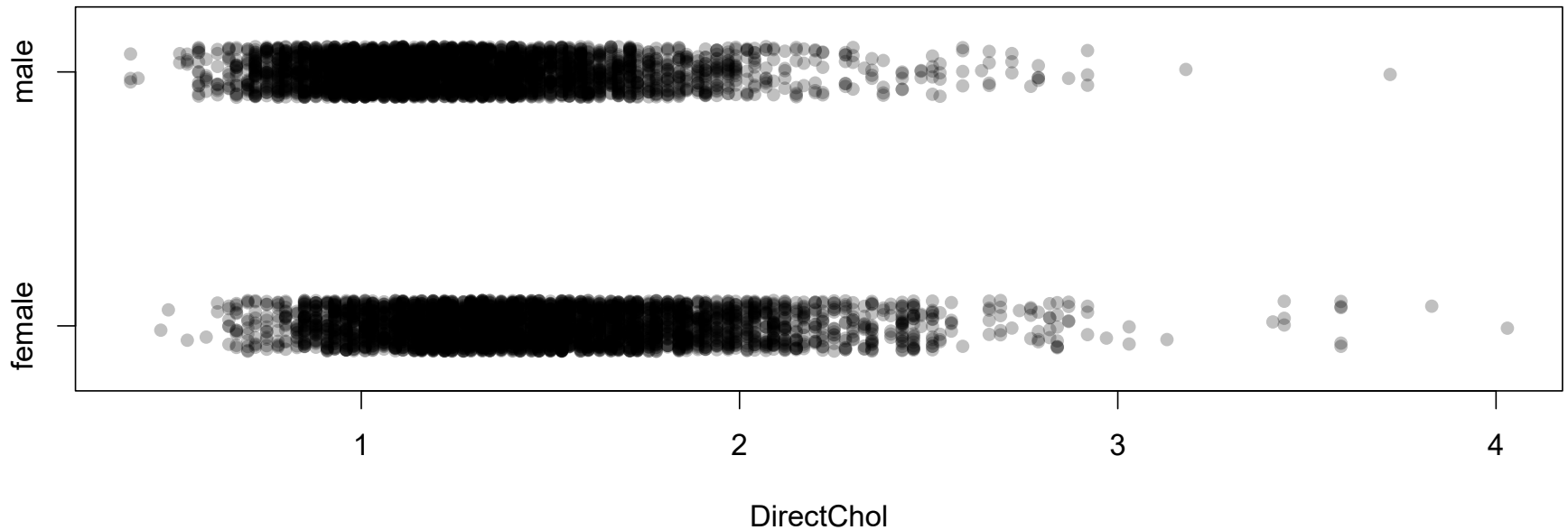


Univariate distributions: comparative strip charts

```
rgb(0, 0, 0, alpha = 0.25)
```

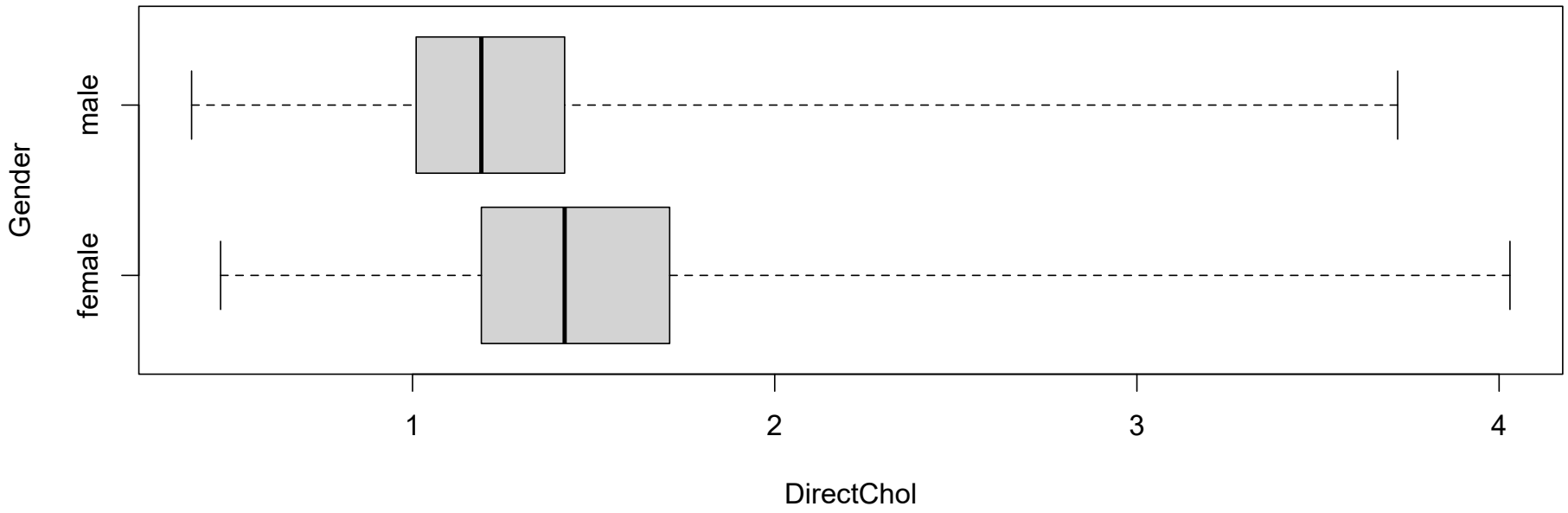
```
[1] "#00000040"
```

```
stripchart(DirectChol ~ Gender, data = NHANES, method = "jitter",
           pch = 16, col = "#00000040")
```



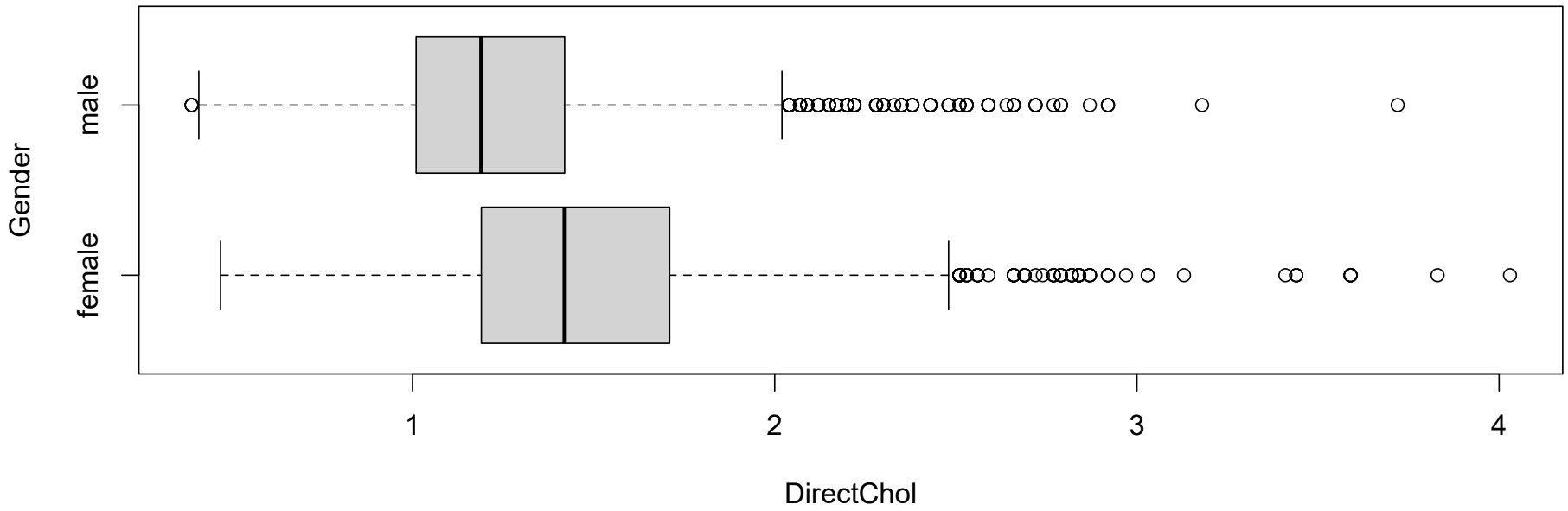
Univariate distributions: comparative box and whisker plots

```
boxplot(DirectChol ~ Gender, data = NHANES, horizontal = TRUE, range = 0)
```



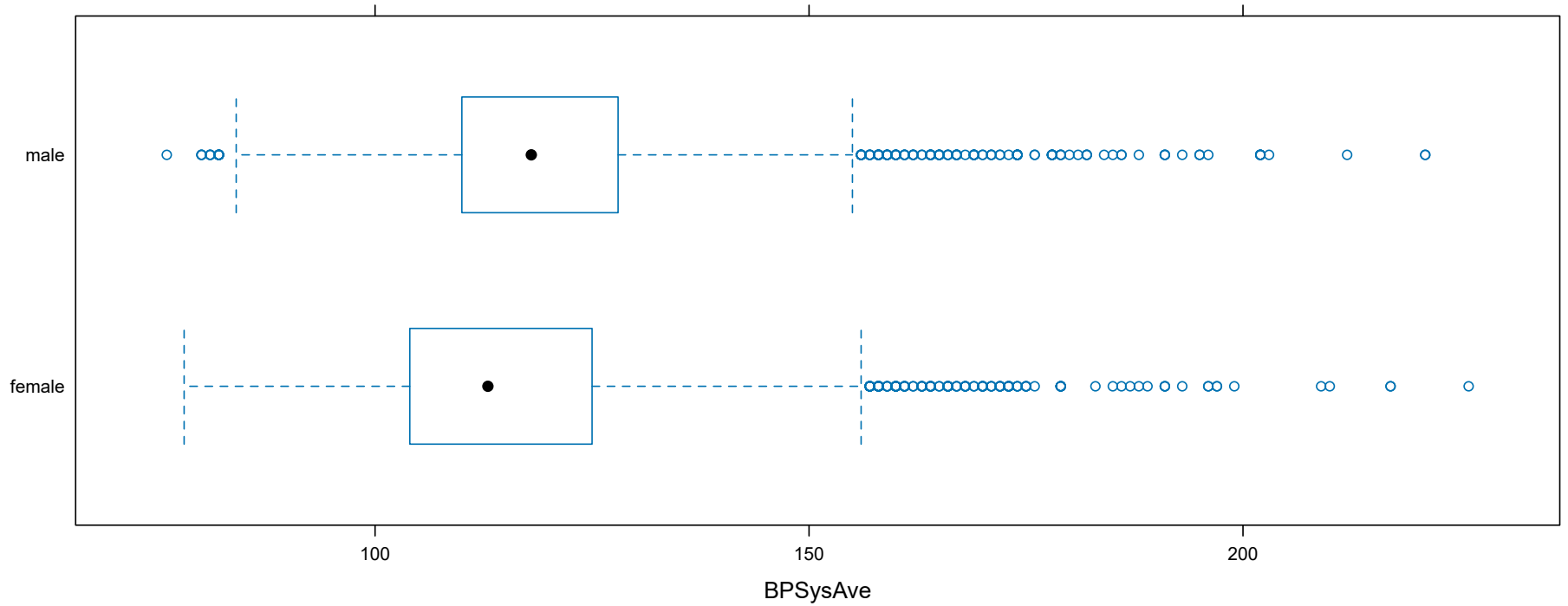
Univariate distributions: comparative box and whisker plots

```
boxplot(DirectChol ~ Gender, data = NHANES, horizontal = TRUE)
```



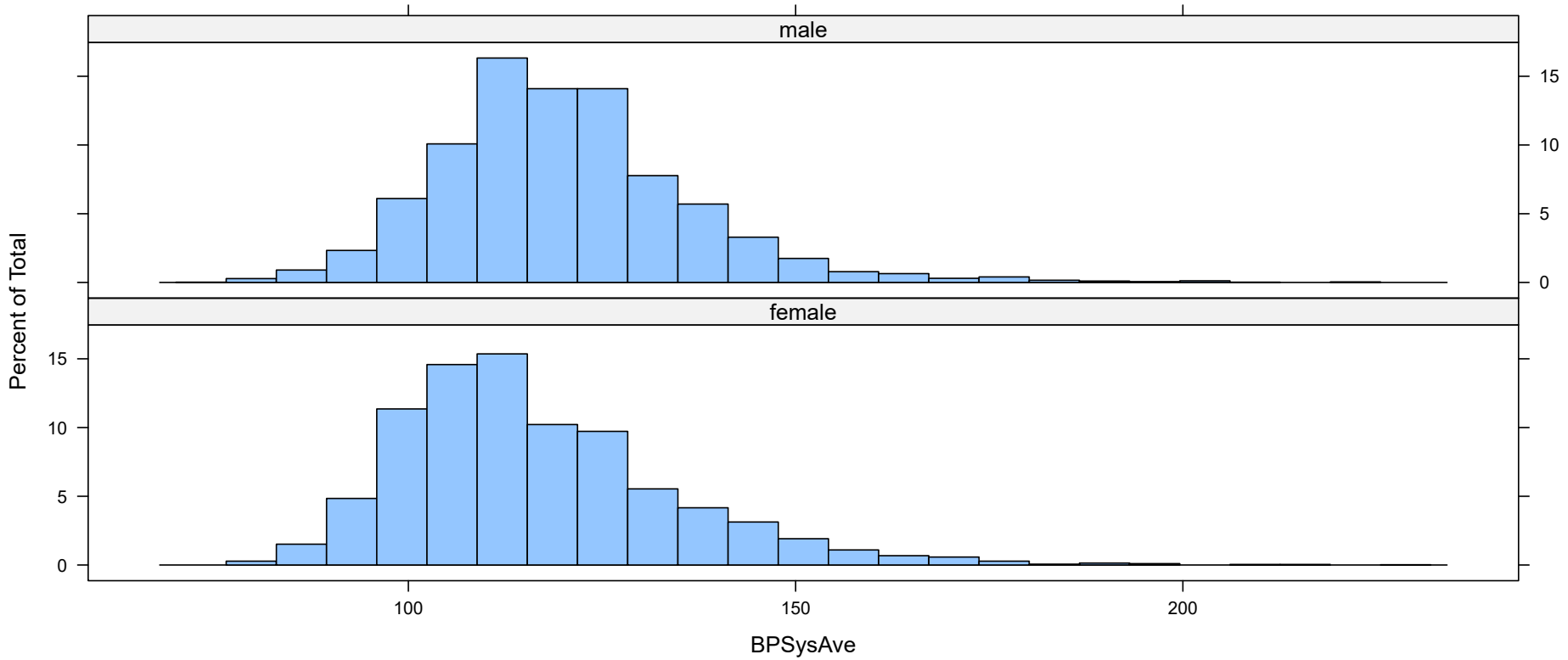
Univariate distributions: comparative box and whisker plots

```
library(package = "lattice")
bwplot(Gender ~ BPSysAve, data = NHANES)
```



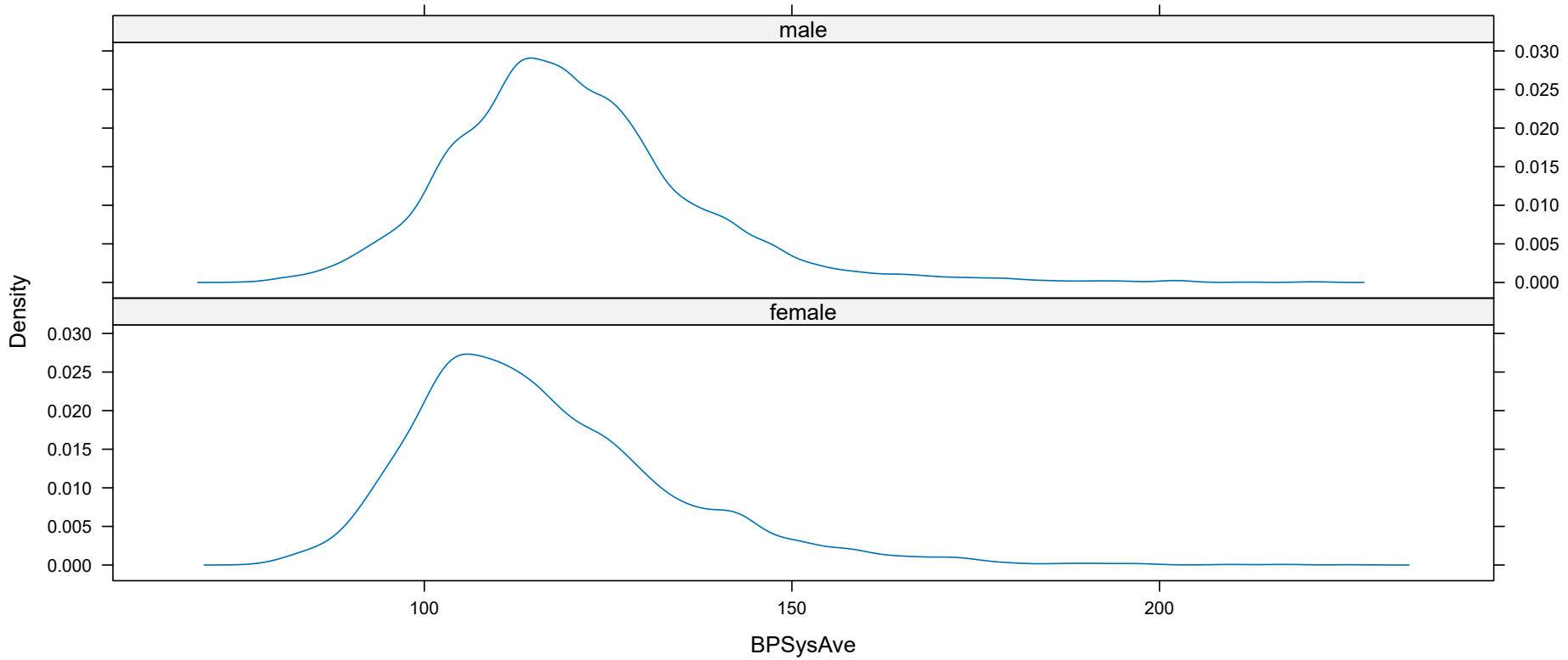
Univariate distributions: comparative histograms

```
histogram( ~ BPSysAve | Gender, data = NHANES,
          layout = c(1, 2), nint = 25)
```



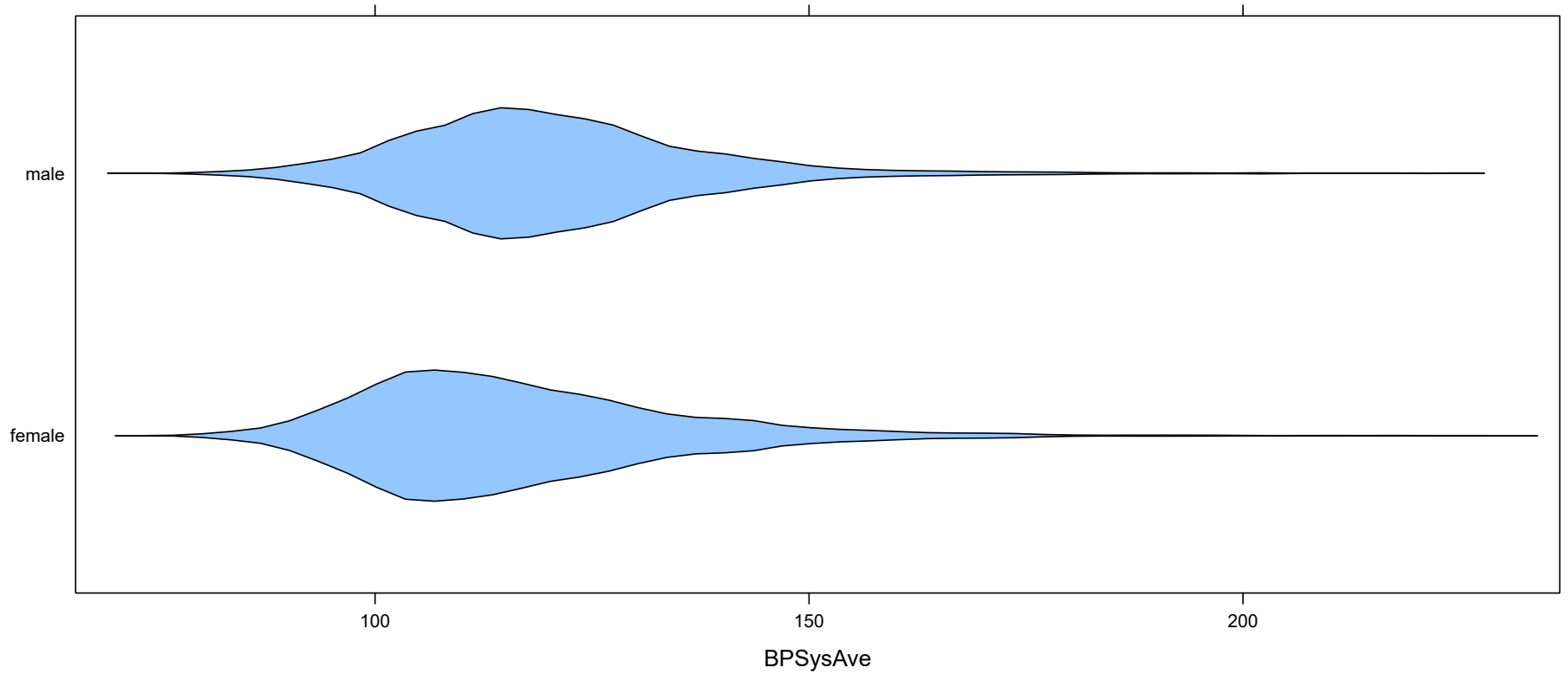
Univariate distributions: kernel density estimates

```
densityplot( ~ BPSysAve | Gender, data = NHANES,
             layout = c(1, 2), plot.points = FALSE)
```



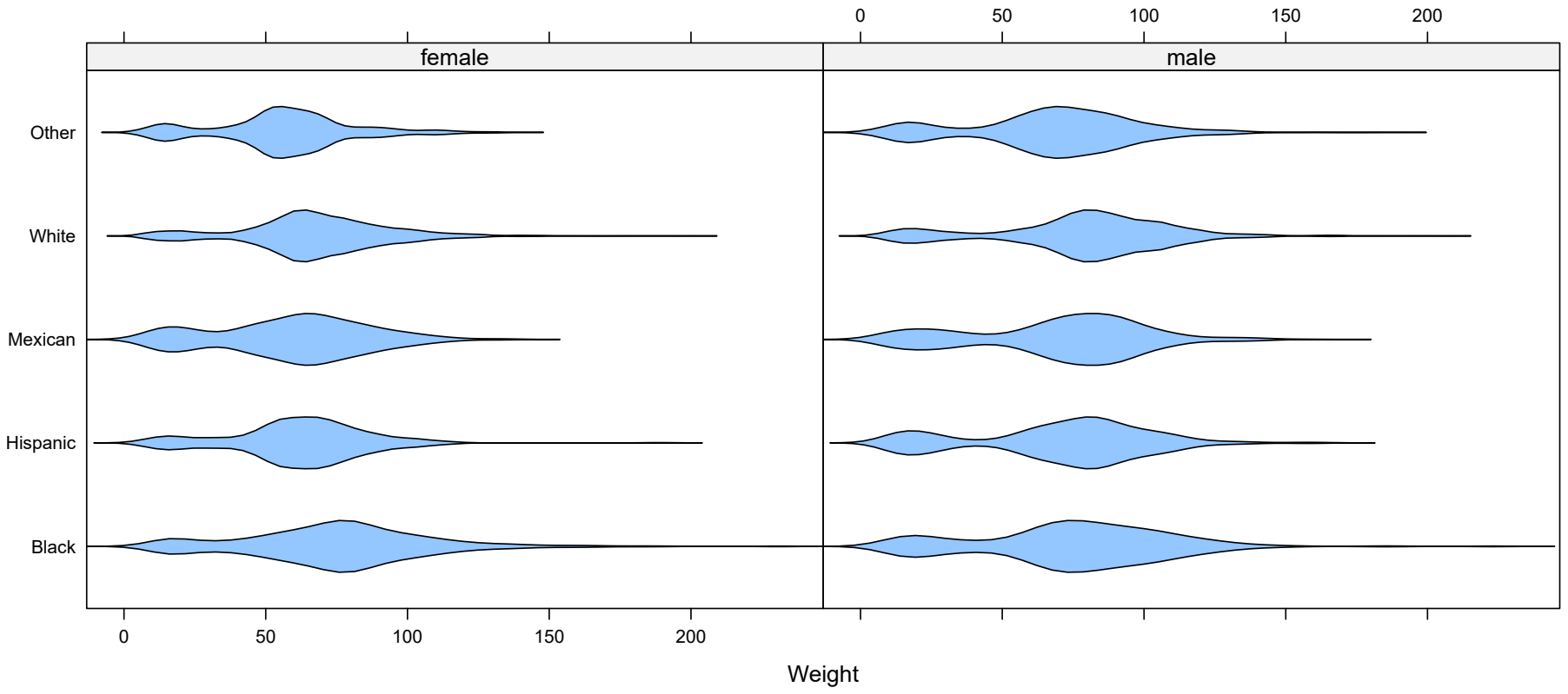
Univariate distributions: comparative violin plots

```
bwplot(Gender ~ BPSysAve, data = NHANES,
       panel = panel.violin)
```



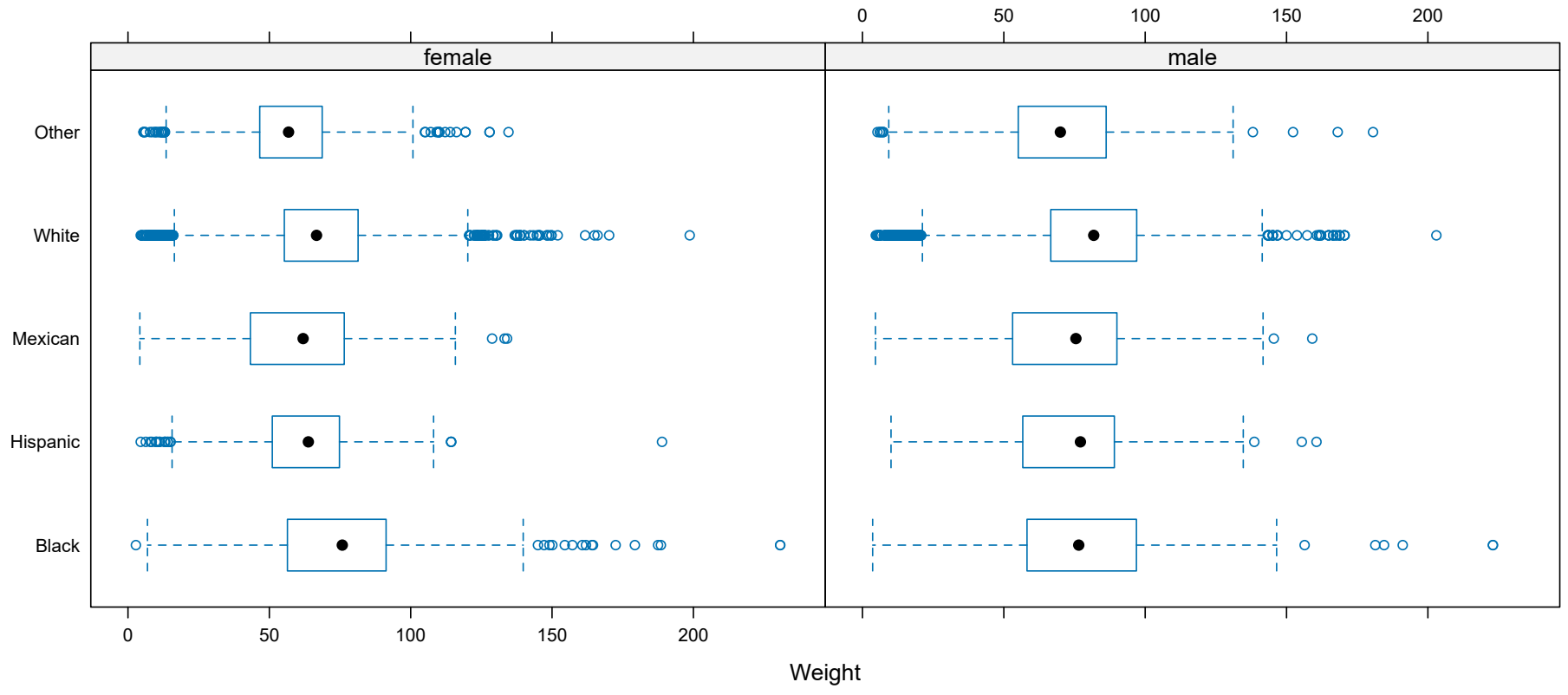
Univariate distributions: comparative violin plots

```
bwplot(Race1 ~ Weight | Gender, data = NHANES, panel = panel.violin)
```



Univariate distributions: comparative violin plots

```
bwplot(Race1 ~ Weight | Gender, data = NHANES)
```



Summary: Univariate distributions

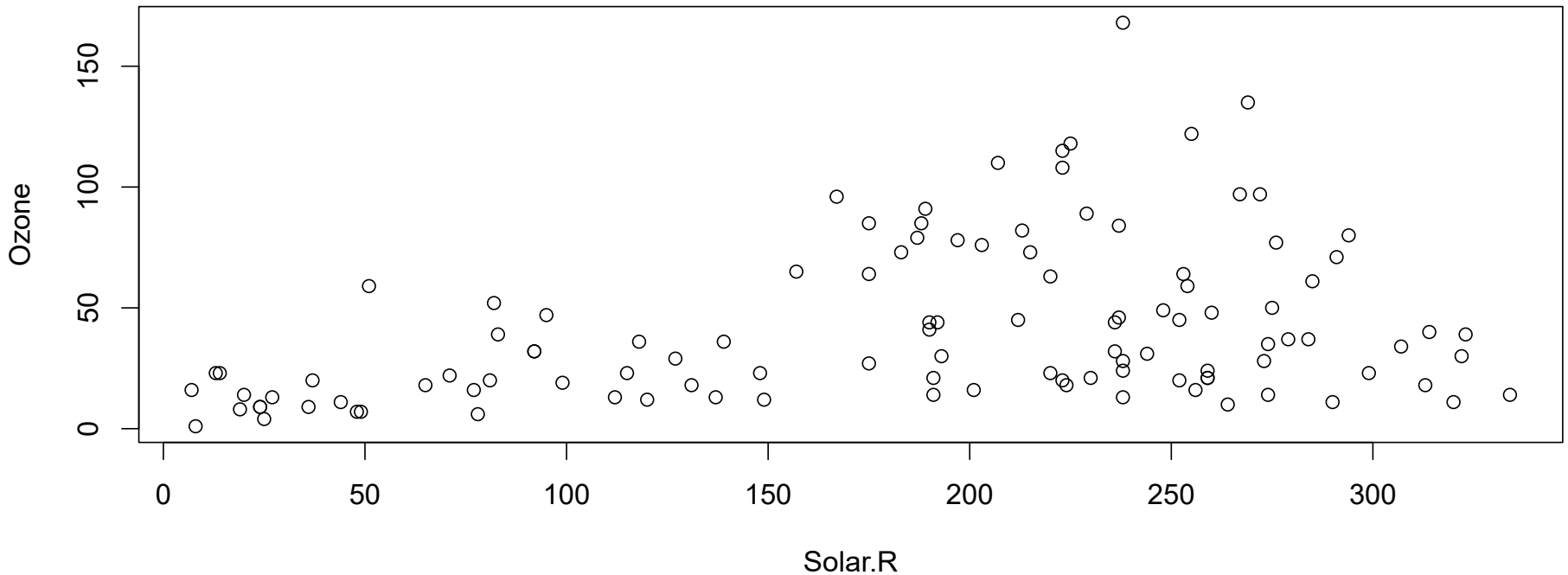
- Basic design: Strip charts
- Generalizations: box and whisker plots, density estimates

Bivariate Data

Bivariate distributions: scatter plot

- Encodes two variables as x- and y-coordinates

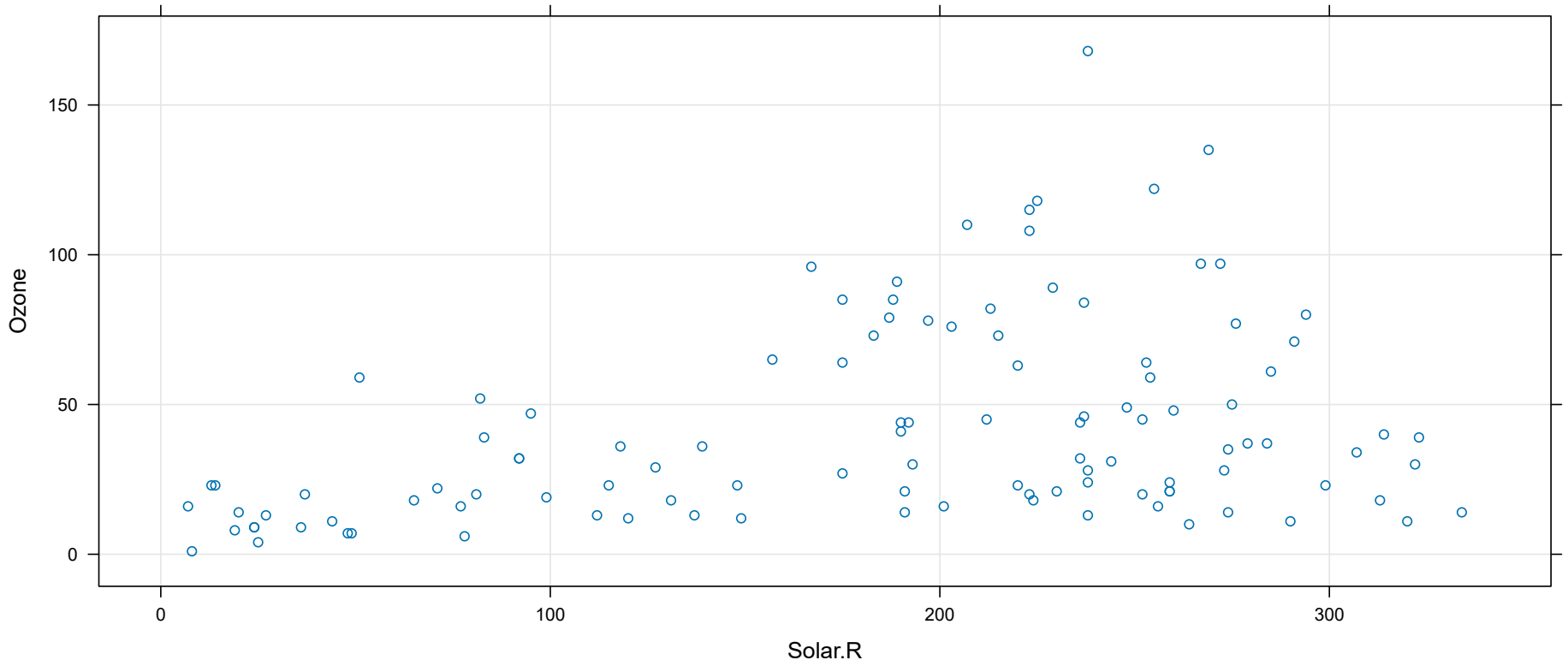
```
plot(Ozone ~ Solar.R, data = airquality)
```



Bivariate distributions: scatter plot

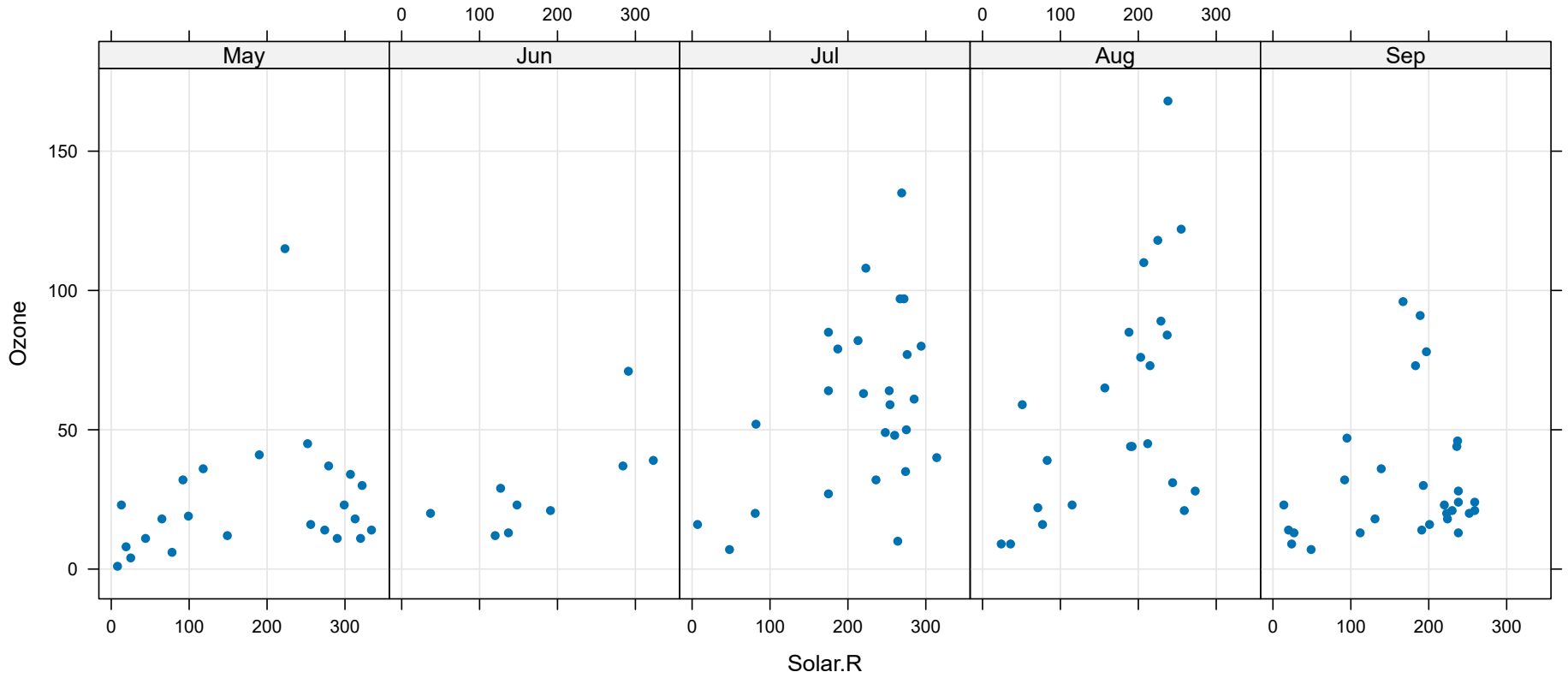
- Encodes two variables as x- and y-coordinates

```
xyplot(Ozone ~ Solar.R, data = airquality, grid = TRUE)
```



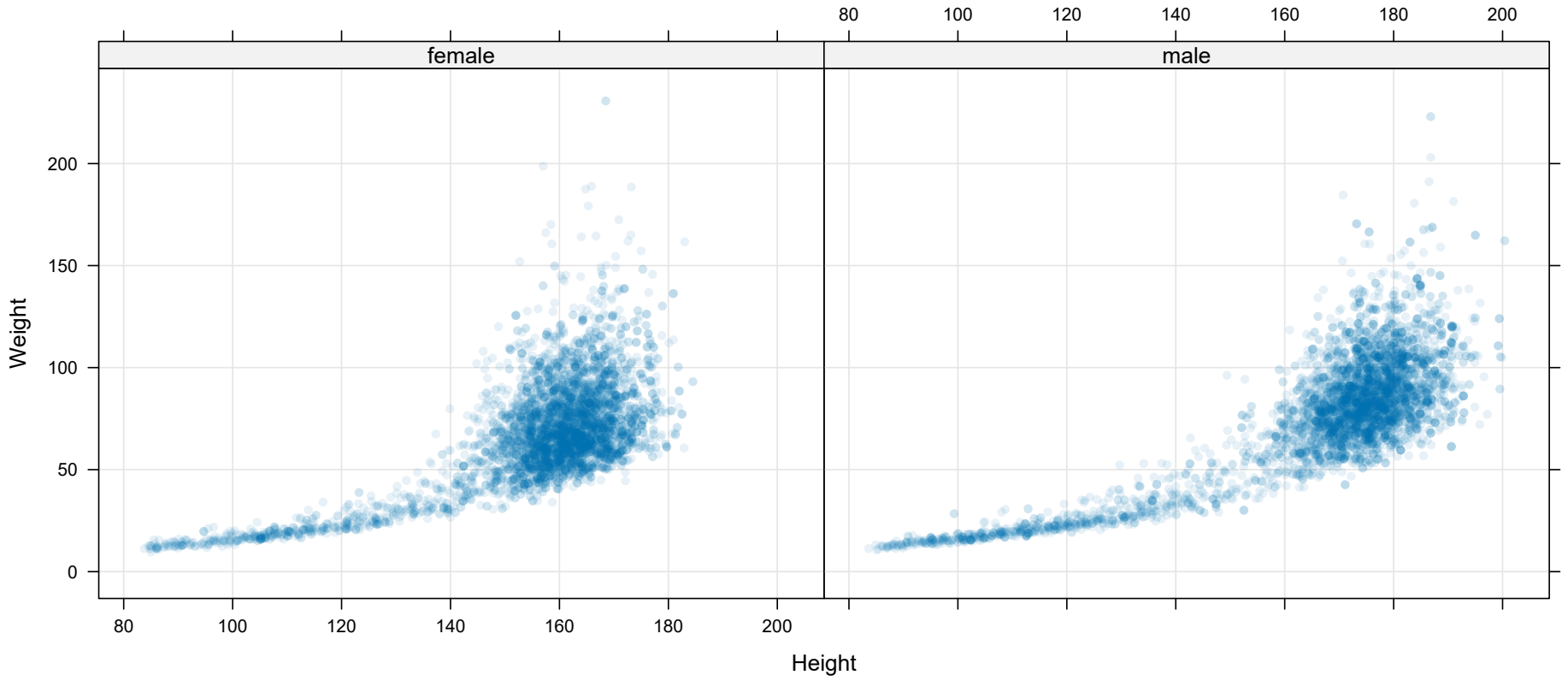
Bivariate distributions: comparative scatter plots

```
xyplot(Ozone ~ Solar.R | Month, data = airquality,
       grid = TRUE, layout = c(5, 1), pch = 16)
```



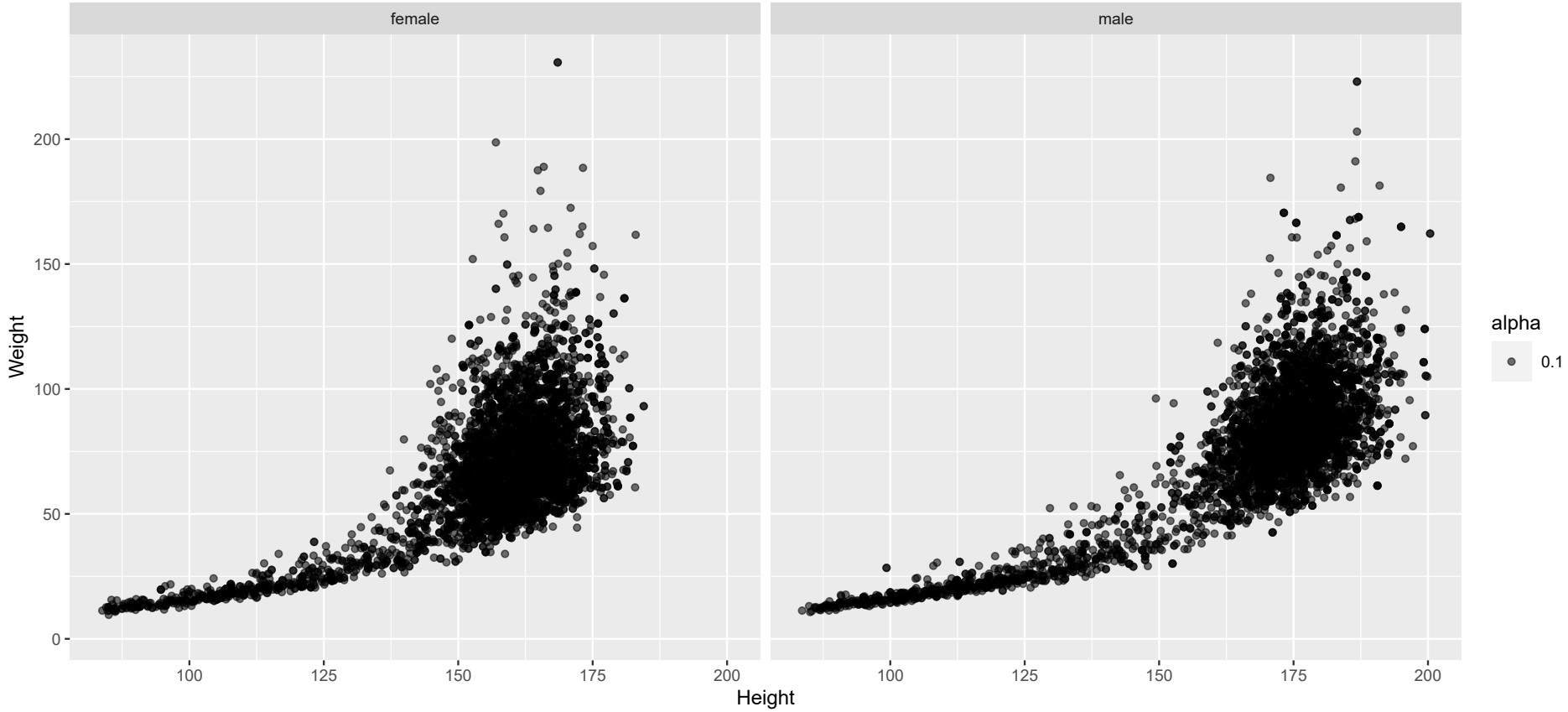
Bivariate distributions: semi-transparent colors

```
xyplot(Weight ~ Height | Gender, data = NHANES,
       grid = TRUE, pch = 16, alpha = 0.1)
```



Bivariate distributions: ggplot2

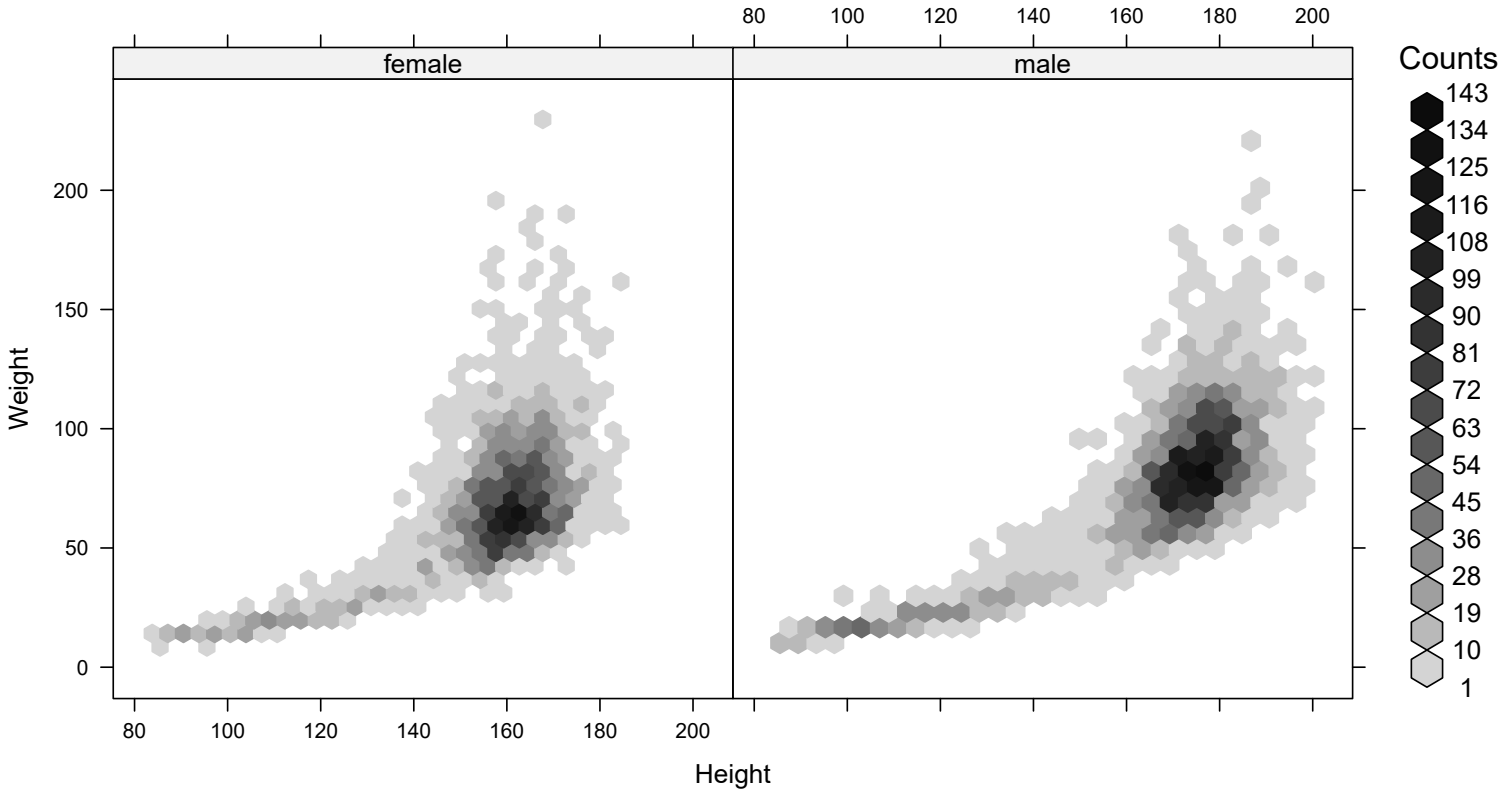
```
library(ggplot2)
ggplot(data = NHANES) + facet_grid(~ Gender) +
    geom_point(mapping = aes(x = Height, y = Weight, alpha = 0.1))
```



Bivariate distributions: hexagonal binning

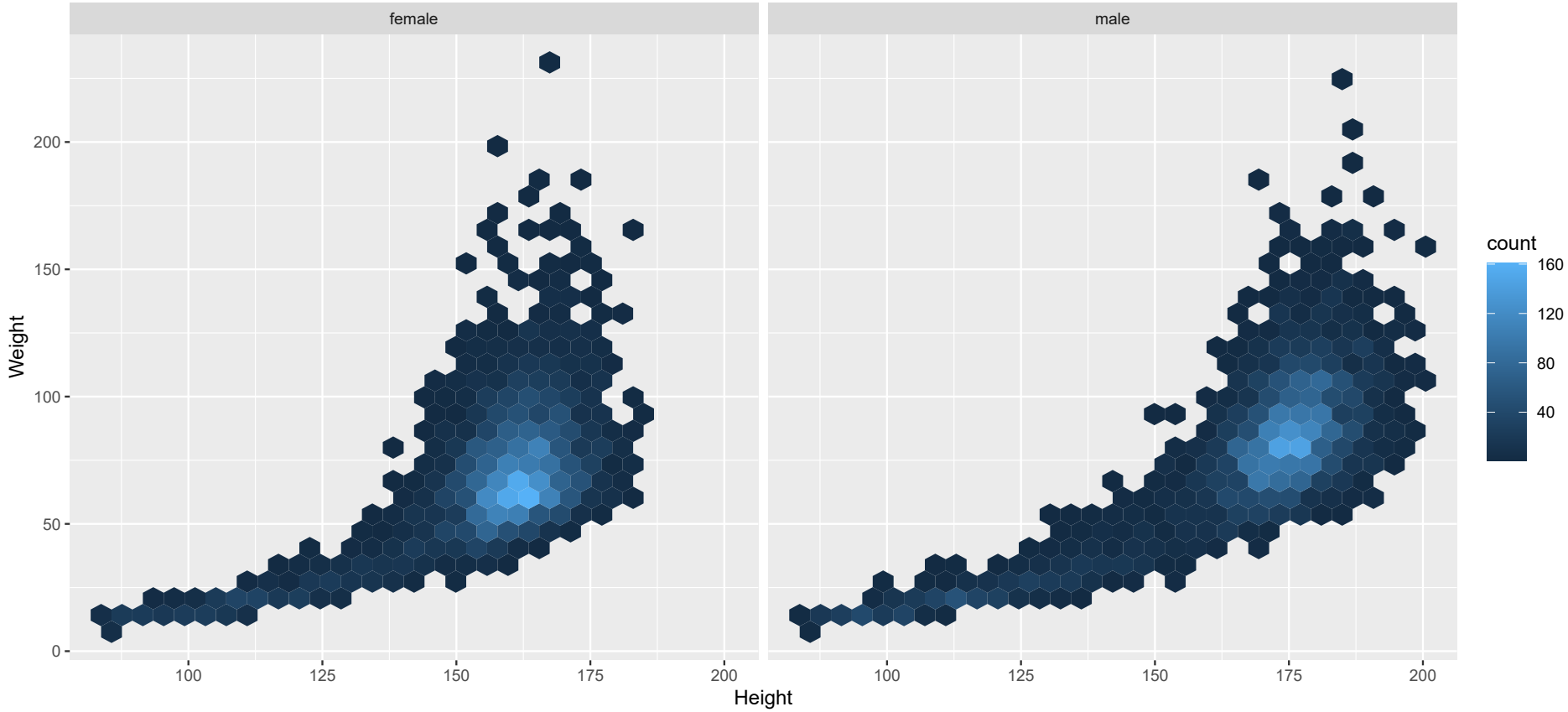
- Like histograms: but with hexagons instead of rectangles
- Bin counts are usually indicated by color

```
library(hexbin); hexbinplot(Weight ~ Height | Gender, data = NHANES, aspect = 1)
```



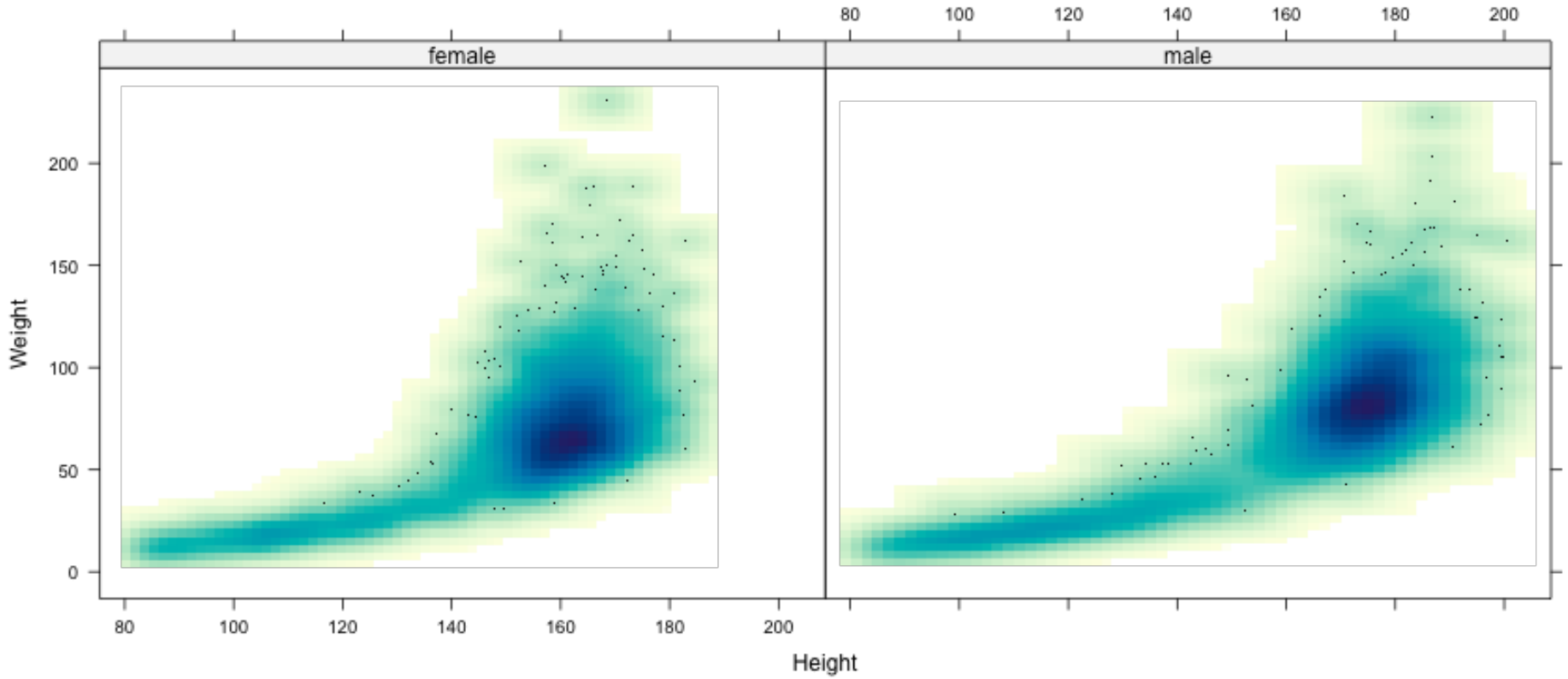
Bivariate distributions: hexagonal binning

```
ggplot(data = NHANES) + facet_grid(~ Gender) +
    geom_hex(mapping = aes(x = Height, y = Weight))
```



Bivariate distributions: kernel density estimates

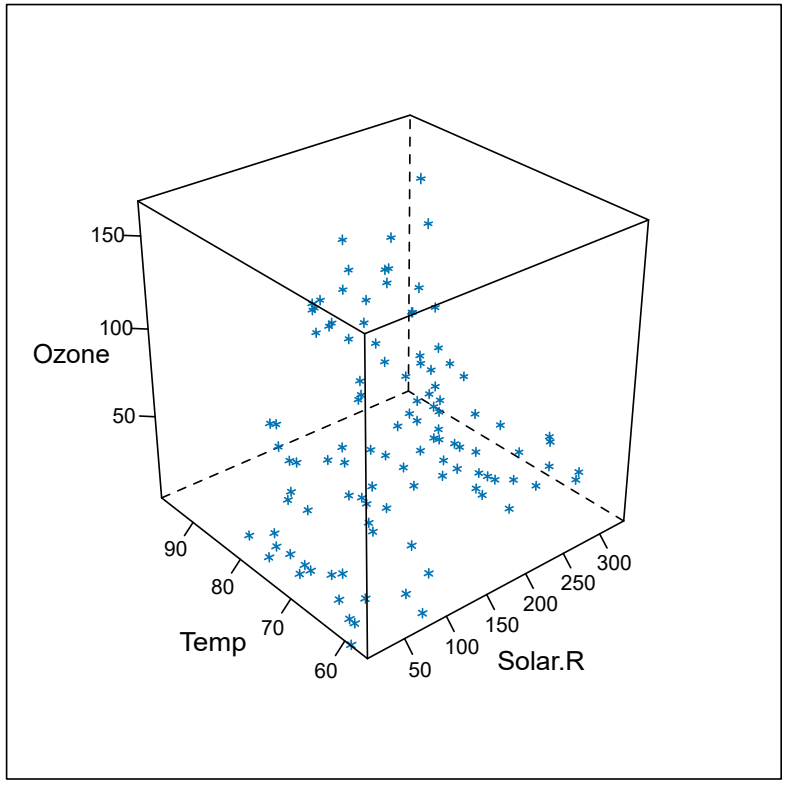
```
xyplot(Weight ~ Height | Gender, data = NHANES,
       grid = TRUE, panel = panel.smoothScatter)
```



Trivariate data: projection into two-dimensional space

- Up to three variables can be mapped to x, y, z-coordinates

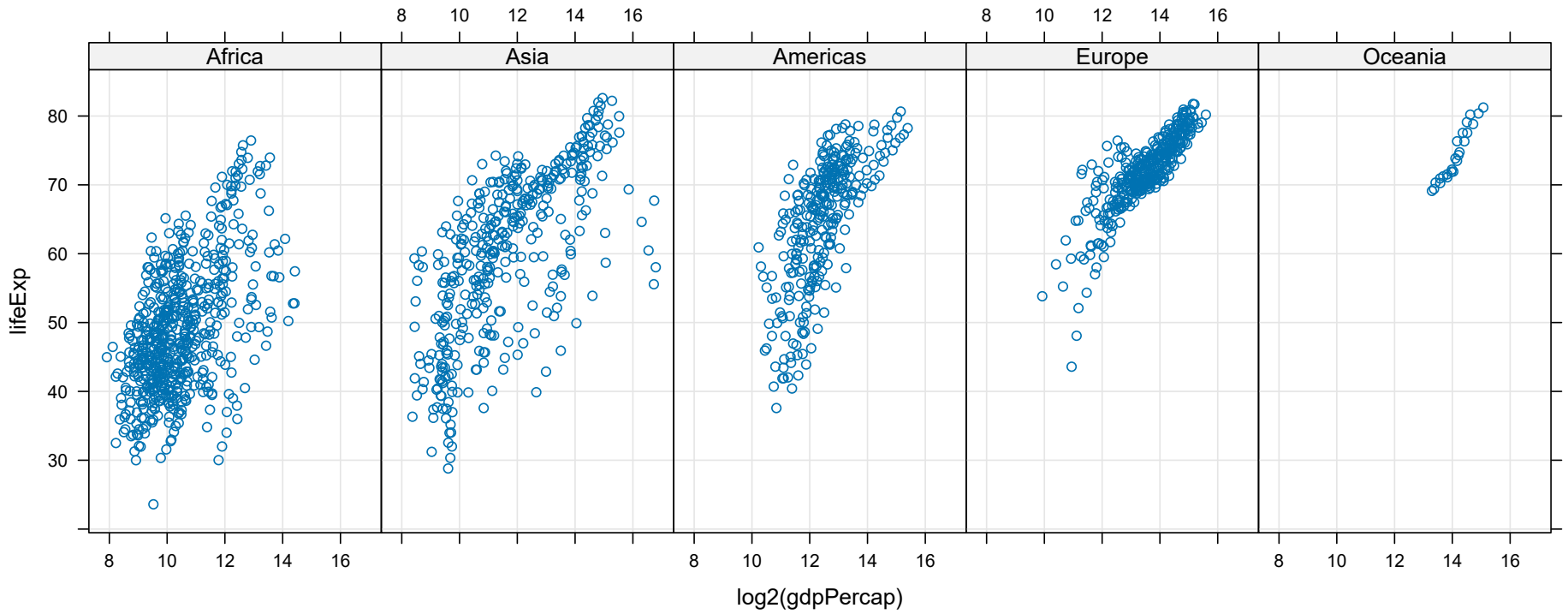
```
cloud(Ozone ~ Solar.R + Temp, data = airquality,
      scales = list(arrows = FALSE))
```



Conditioning / faceting

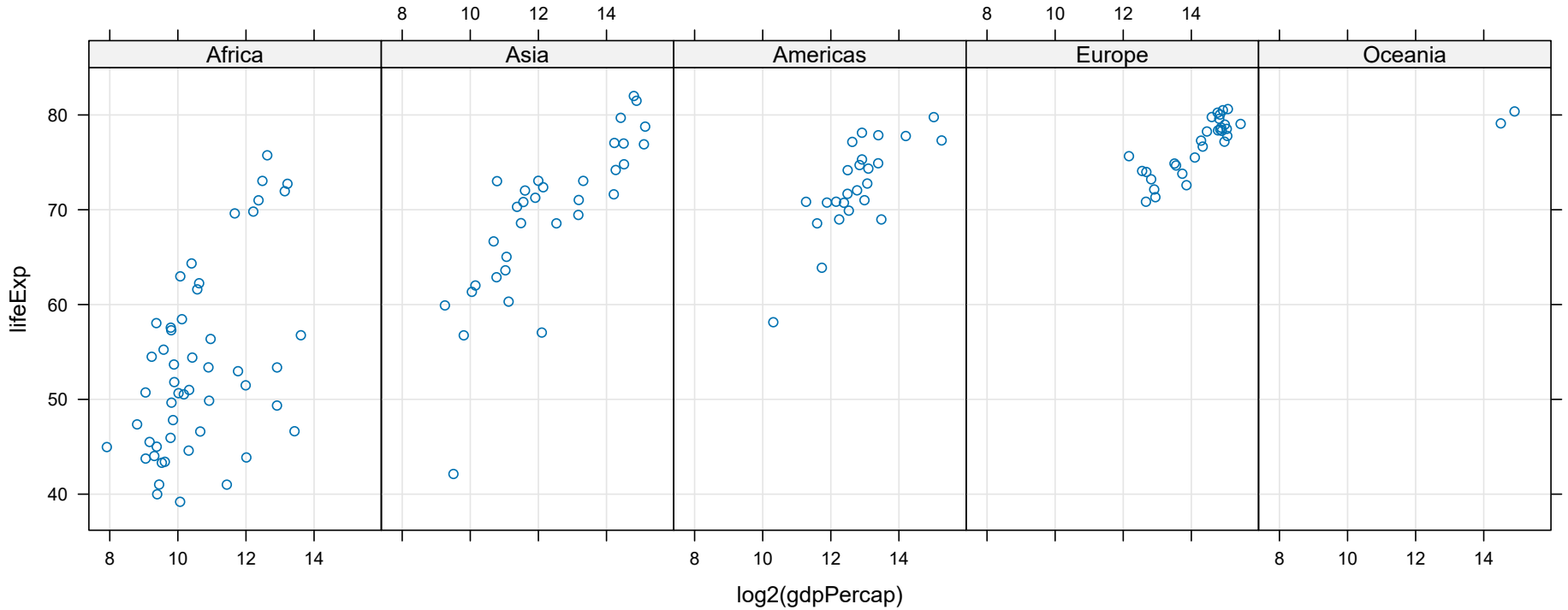
- Categorical variables can be compared using superposition

```
xyplot(lifeExp ~ log2(gdpPercap) | reorder(continent, lifeExp),
       data = gapminder, grid = TRUE)
```



Conditioning / faceting

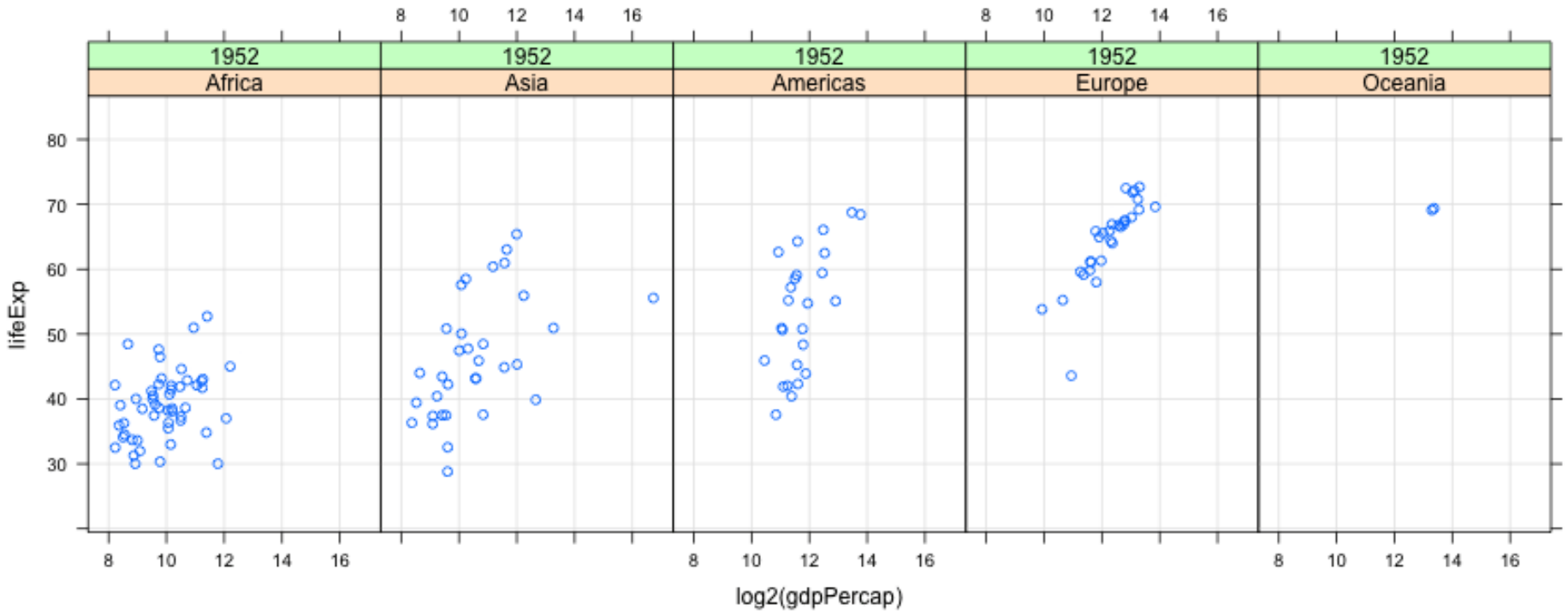
```
xyplot(lifeExp ~ log2(gdpPercap) | reorder(continent, lifeExp),
       data = gapminder, grid = TRUE, subset = (year == 2002))
```



Conditioning / faceting

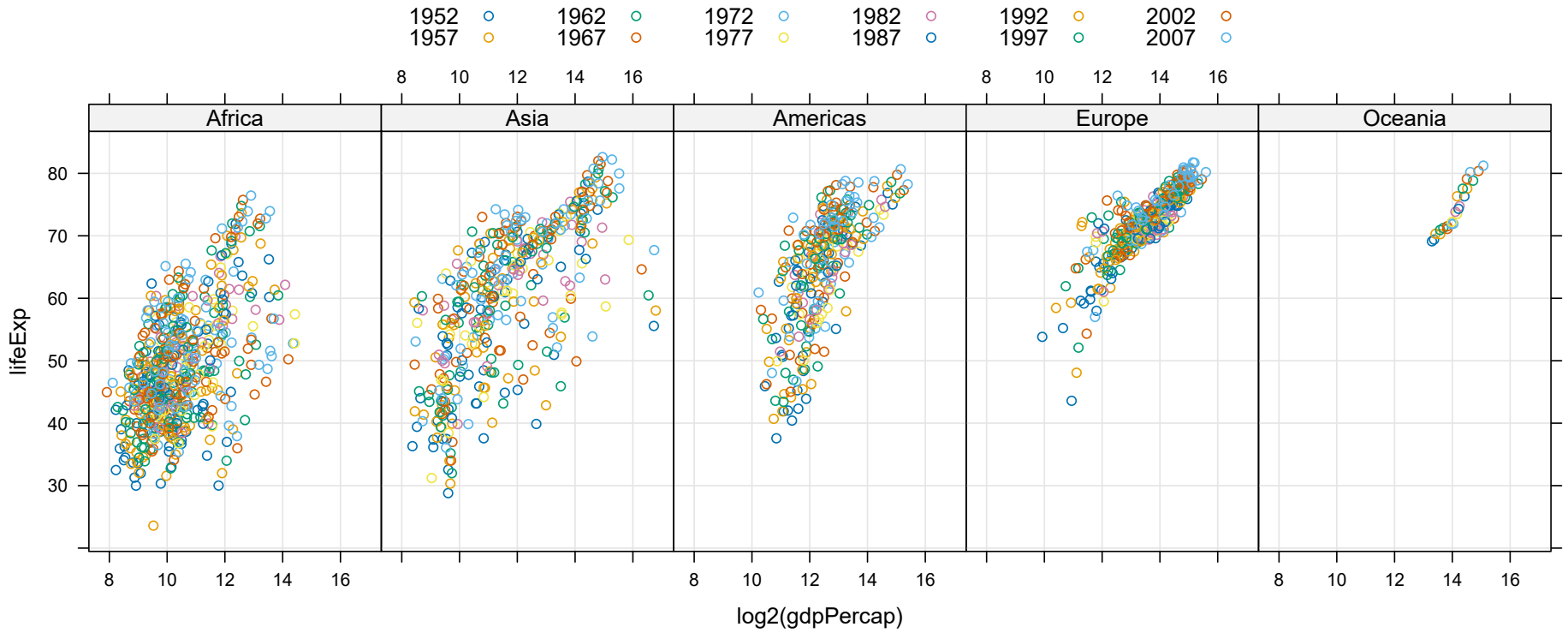
- For too many comparisons, single display page may not be enough

```
xyplot(lifeExp ~ log2(gdpPercap) | reorder(continent, lifeExp) + factor(year),
       data = gapminder, grid = TRUE, layout = c(5, 1))
```



Conditioning / faceting

```
xyplot(lifeExp ~ log2(gdpPercap) | reorder(continent, lifeExp),
       data = gapminder, grid = TRUE, group = year,
       auto.key = list(columns = 6))
```



Tables: Summary measures on categorical attributes

```
str(gapminder)
```

```
'data.frame':    1698 obs. of  6 variables:
 $ country  : chr  "Afghanistan" "Afghanistan" "Afghanistan" "Afghanistan" ...
 $ continent: chr  "Asia" "Asia" "Asia" "Asia" ...
 $ year     : int  1952 1957 1962 1967 1972 1977 1982 1987 1992 1997 ...
 $ lifeExp  : num  28.8 30.3 32 34 36.1 ...
 $ pop      : int  8425333 9240934 10267083 11537966 13079460 14880372 12881816 13867957 16317921 22227415 ...
 $ gdpPercap: num  779 821 853 836 740 ...
```

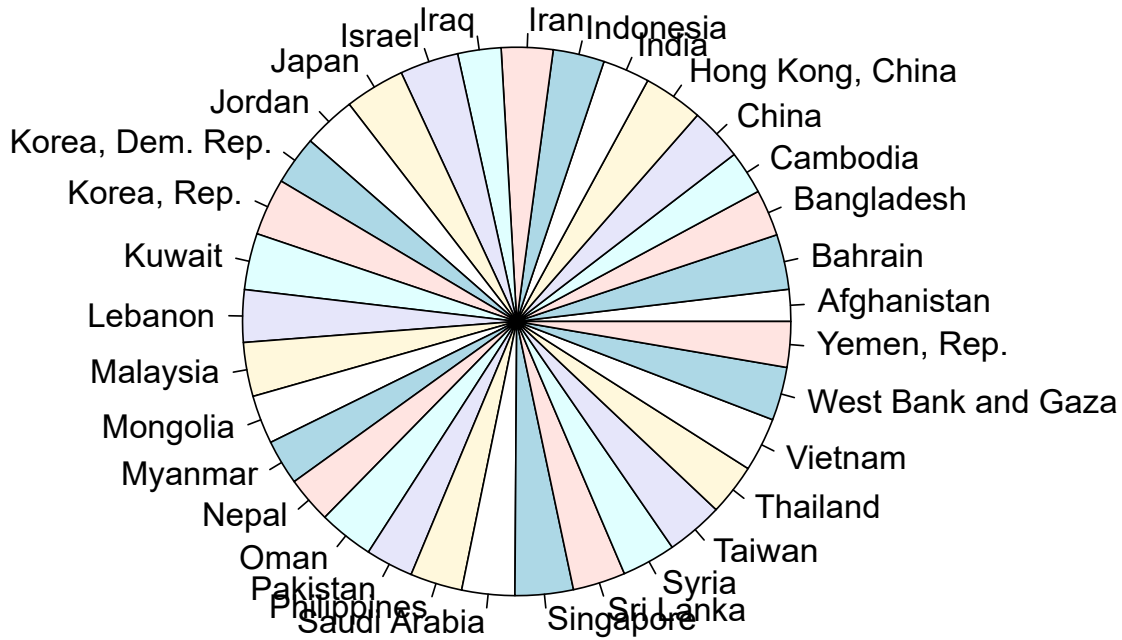
Tables: Summary measures on categorical attributes

```
gapminder_sub <-
    subset(gapminder,
           continent == "Asia" & year %in% c(1967, 1987, 2007))
xtabs(lifeExp ~ country + year, data = gapminder_sub)
```

```
                  year
country                1967     1987     2007
  Afghanistan       34.02000 40.82200 43.82800
  Bahrain           59.92300 70.75000 75.63500
  Bangladesh        43.45300 52.81900 64.06200
  Cambodia          45.41500 53.91400 59.72300
  China             58.38112 67.27400 72.96100
  Hong Kong, China  70.00000 76.20000 82.20800
  India             47.19300 58.55300 64.69800
  Indonesia         45.96400 60.13700 70.65000
  Iran              52.46900 63.04000 70.96400
  Iraq              54.45900 65.04400 59.54500
  Israel            70.75000 75.60000 80.74500
```

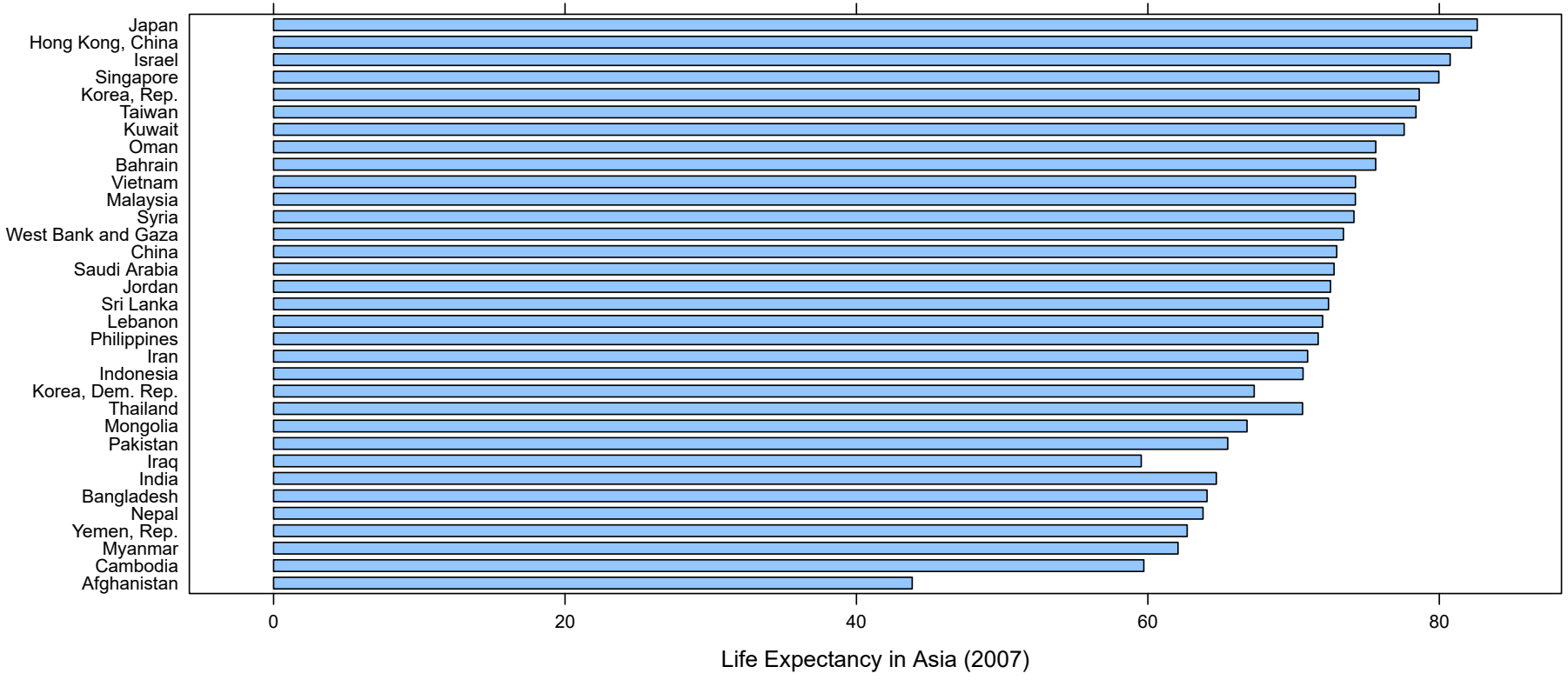
Pie charts

```
letab <- xtabs(lifeExp ~ country + year, data = gapminder_sub)
pie(letab[, "2007"])
```



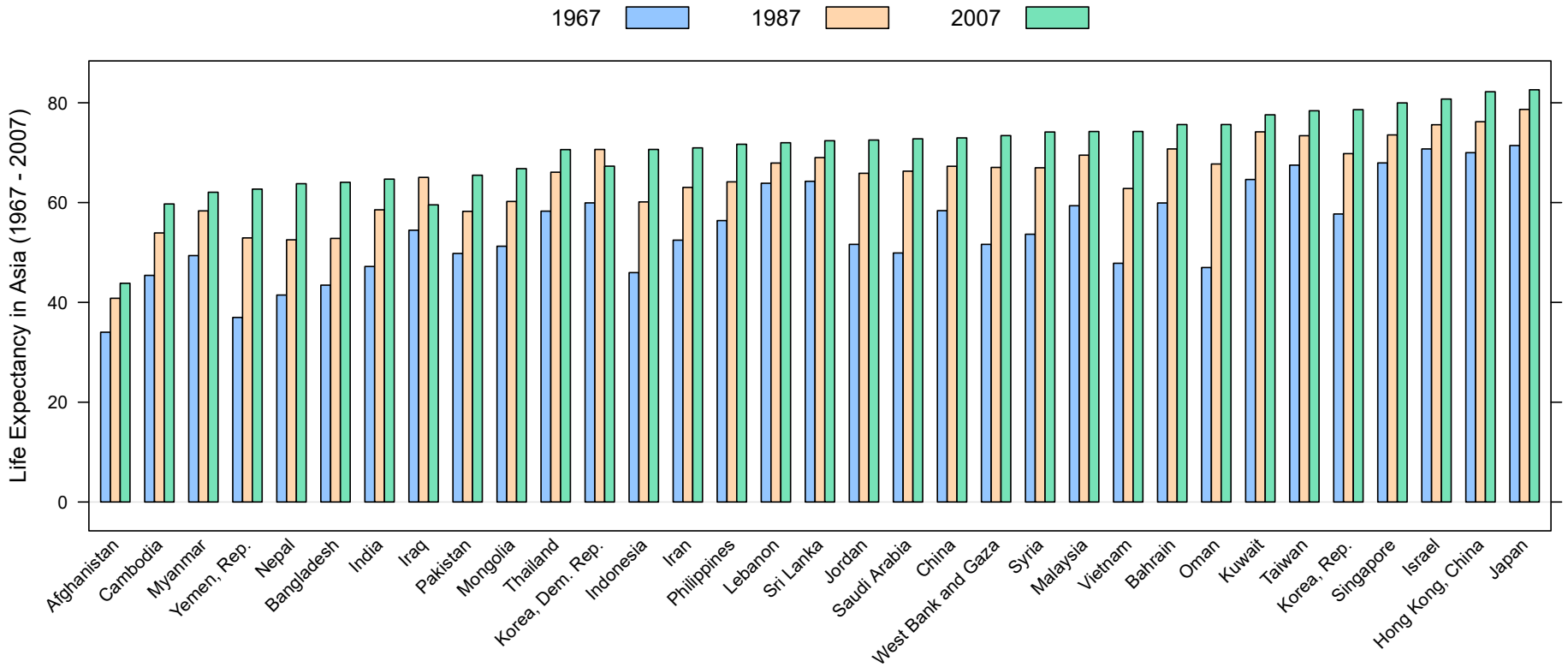
Bar charts

```
letab <- xtabs(lifeExp ~ reorder(country, lifeExp, max) + year,
               data = gapminder_sub)
barchart(letab[, "2007"], origin = 0,
         xlab = "Life Expectancy in Asia (2007)")
```



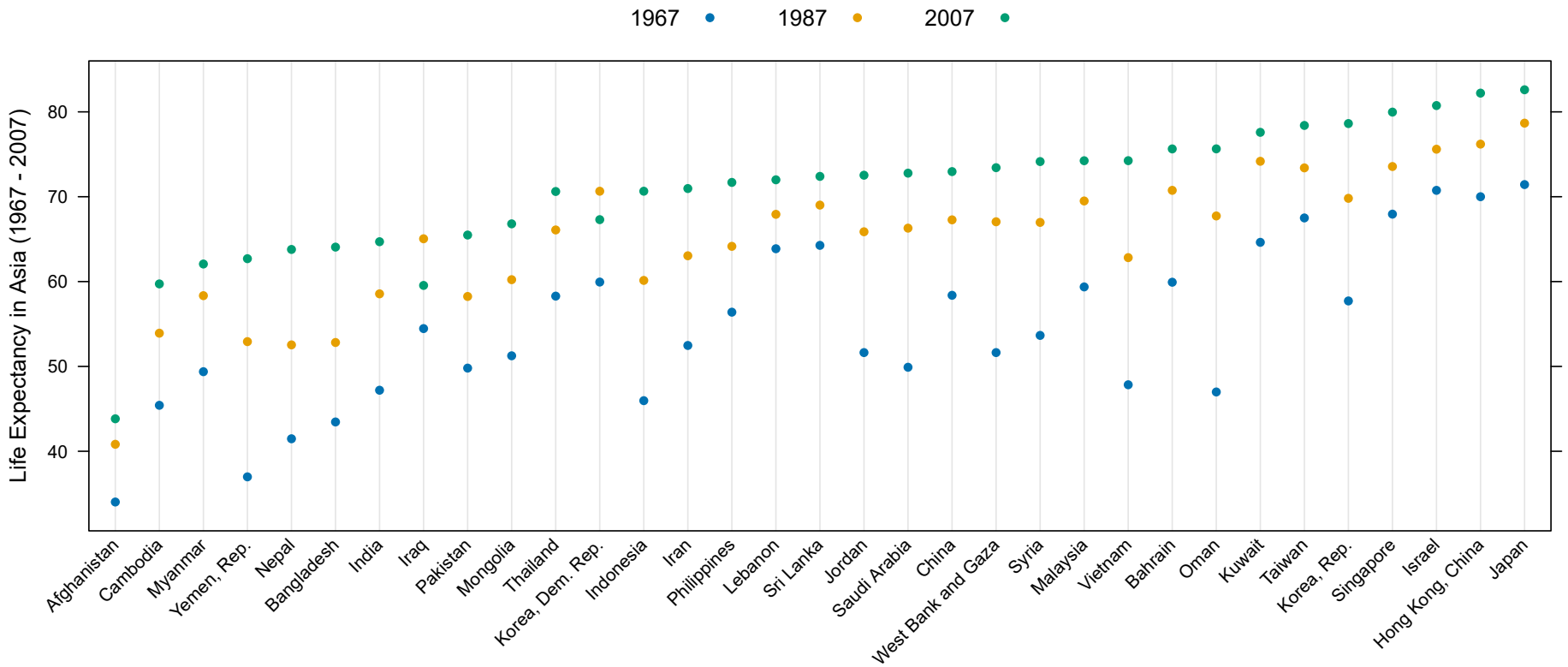
Bar charts

```
barchart(letab, ylab = "Life Expectancy in Asia (1967 - 2007)",
         horizontal = FALSE, stack = FALSE,
         auto.key = list(columns = 3),
         scales = list(x = list(rot = 45)))
```



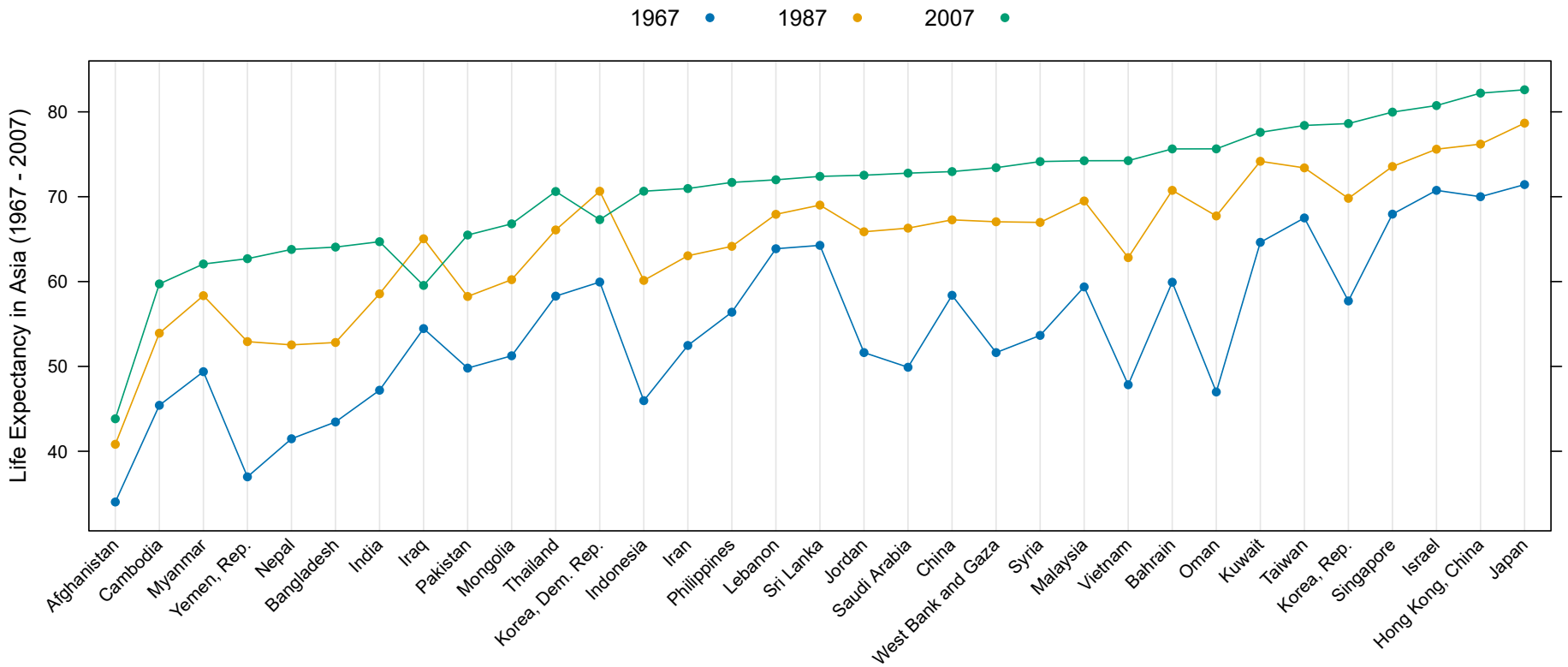
Dot plots

```
dotplot(letab, ylab = "Life Expectancy in Asia (1967 - 2007)",
        horizontal = FALSE, par.settings = simpleTheme(pch = 16),
        auto.key = list(columns = 3),
        scales = list(x = list(rot = 45)))
```



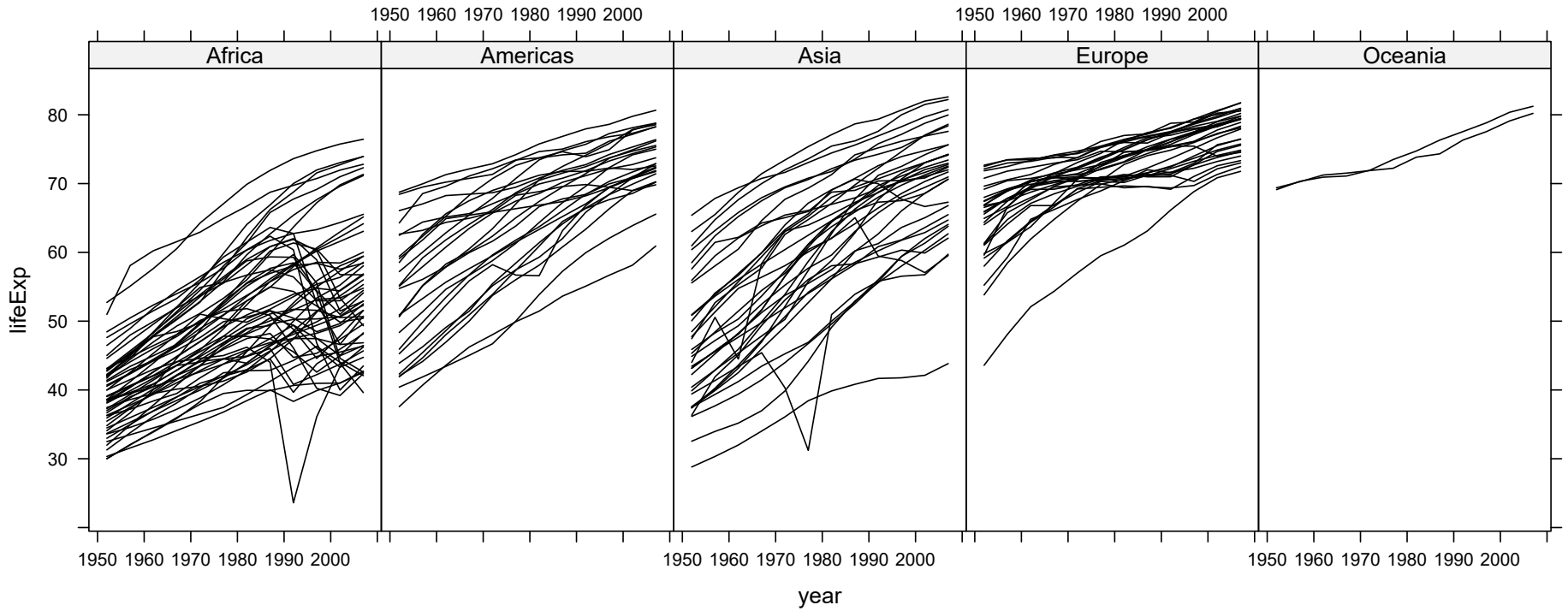
Dot plots

```
dotplot(letab, ylab = "Life Expectancy in Asia (1967 - 2007)",
        horizontal = FALSE, par.settings = simpleTheme(pch = 16),
        auto.key = list(columns = 3), type = "o",
        scales = list(x = list(rot = 45)))
```



Time-series plots

```
xyplot(lifeExp ~ year | continent, data = gapminder,
       groups = country, type = "l", col = "black")
```



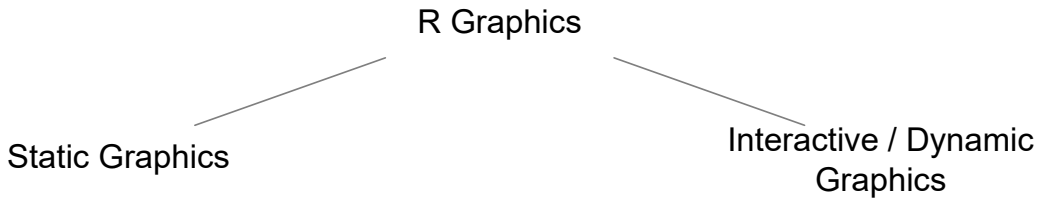
Using the R Graphics System

- We have seen examples of statistical graphics
- But how do we create such plots with our own data?

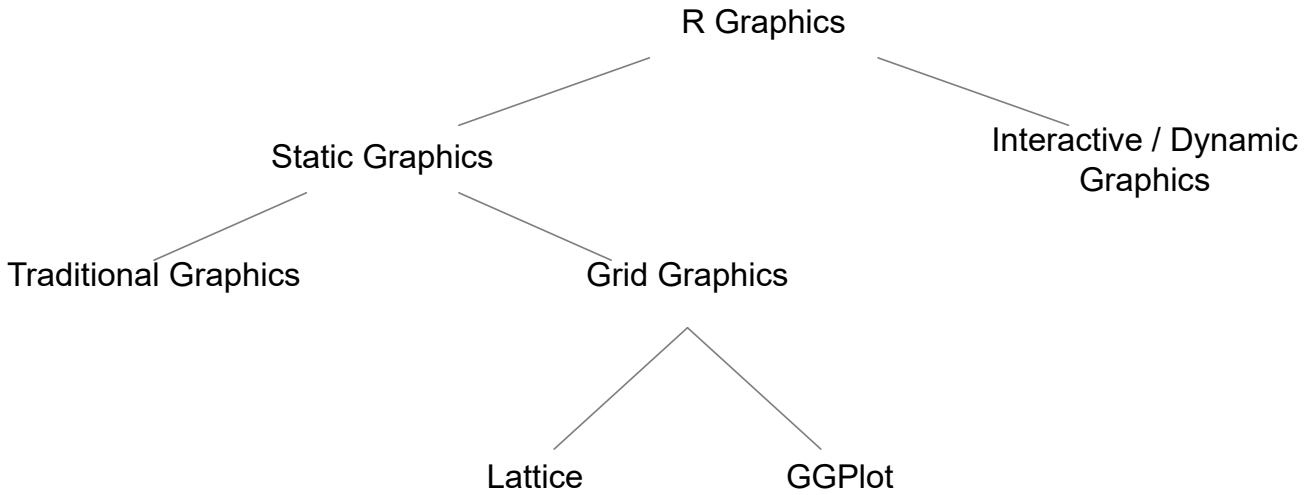
Using the R Graphics System

- We have seen examples of statistical graphics
- But how do we create such plots with our own data?
- We will now discuss the graphics tools in R in more detail

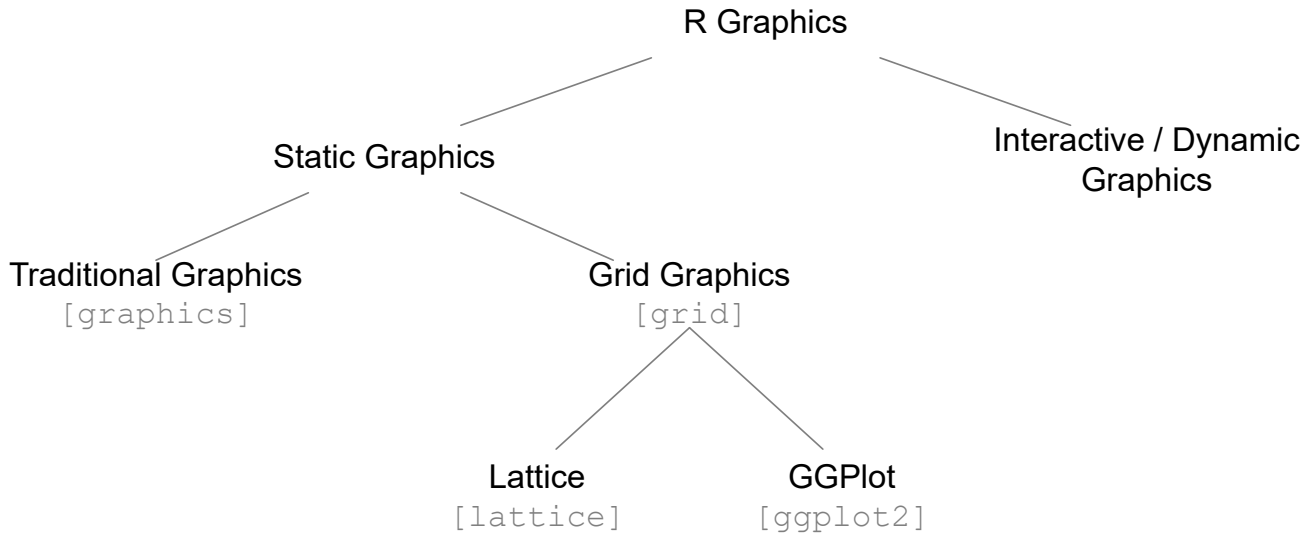
Using the R Graphics System



Using the R Graphics System



Using the R Graphics System



Using the R Graphics System

- A good way to start exploring (any package)

```
help(package = "graphics")
help(package = "ggplot2")
```

- Demo code available on website