# Introduction to S

Statistics 135

Autumn 2005

# Introduction to S

When discussing the **S** environment, I will, or at least try to, make the following distinctions.

- **S** will be used when what is being discussed will work with **R** or **S-Plus**.
- **R** will be used when discussing **R** specific features
- **S-Plus** will be used when discussing **S-Plus** specific features

# Sample S Analysis

## 93 Cars Data

Data on 93 different 1993 model year cars

Available on Datasets page of the course web site.

Want to examine which variables are related with EPA City MPG ratings.

To read the dataset into **S**, use a command similar to

```
cars93.df <- read.table("93cars.txt", header=T, row.names=NULL)
```

For this study, a reduced version of the dataset (in terms of variables that will be examined), named `cars.df`.

```
> dim(cars.df)
[1] 93 10

> class(cars.df)
[1] "data.frame"
```

```
> cars.df[1:5,]
   Manu   Model Cylinder    Type EngSize Weight CityMPG HighMPG CityFuel
1 Acura Integra        4   Small     1.8   2705      25      31 4.000000
2 Acura  Legend        6 Midsize     3.2   3560      18      25 5.555556
3  Audi      90        6 Compact     2.8   3375      20      26 5.000000
4  Audi     100        6 Midsize     2.8   3405      19      26 5.263158
5   BMW    535i        4 Midsize     3.5   3640      22      30 4.545455
  HighFuel
1 3.225806
2 4.000000
3 3.846154
4 3.846154
5 3.333333
```

Lets get a simple summary of the data frame from **S**.

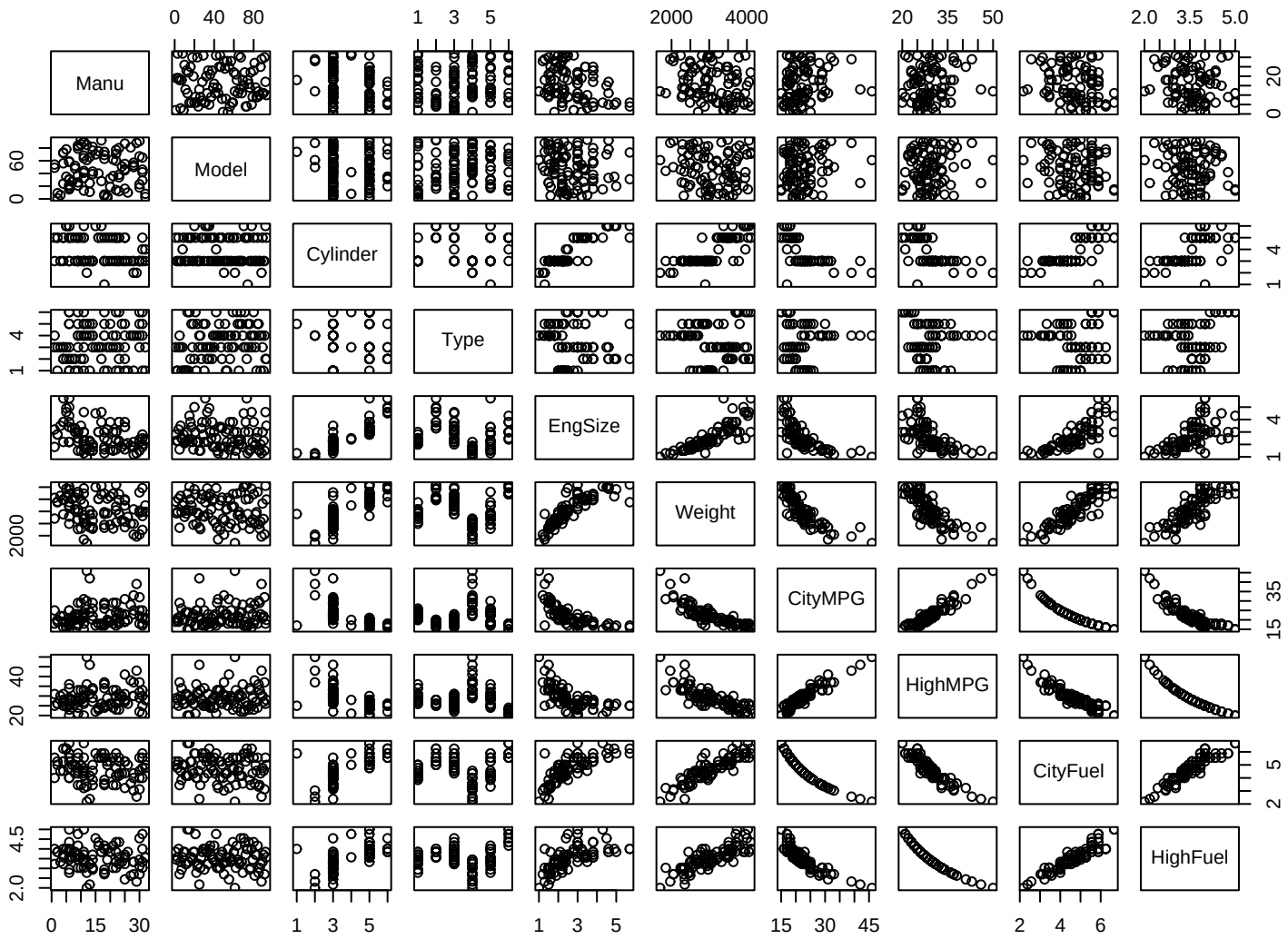
```
> summary(cars.df)
        Manu         Model     Cylinder      Type        EngSize
 Chevrolet: 8   100    : 1   *: 1     Compact:16   Min.   :1.000
 Ford     : 8   190E   : 1   3: 3     Large  :11   1st Qu.:1.800
 Dodge    : 6   240    : 1   4:49     Midsize:22   Median :2.400
 Mazda    : 5   300E   : 1   5: 2     Small  :21   Mean   :2.668
 Pontiac  : 5   323    : 1   6:31     Sporty :14   3rd Qu.:3.300
 Buick    : 4   535i   : 1   8: 7     Van    : 9   Max.   :5.700
 (Other)  :57   (Other):87
     Weight        CityMPG         HighMPG         CityFuel        HighFuel
 Min.   :1695   Min.   :15.00   Min.   :20.00   Min.   :2.174   Min.   :2.000
 1st Qu.:2620   1st Qu.:18.00   1st Qu.:26.00   1st Qu.:4.000   1st Qu.:3.226
 Median :3040   Median :21.00   Median :28.00   Median :4.762   Median :3.571
 Mean   :3073   Mean   :22.37   Mean   :29.09   Mean   :4.699   Mean   :3.541
 3rd Qu.:3525   3rd Qu.:25.00   3rd Qu.:31.00   3rd Qu.:5.556   3rd Qu.:3.846
 Max.   :4105   Max.   :46.00   Max.   :50.00   Max.   :6.667   Max.   :5.000
```

For a data frame, `summary` returns the 5 figure summary for numeric variables and counts for categorical variable (or numeric variables that look like they might be categorical).

Now, lets get a graphical summary of the data. In **R** we can give the commands

```
postscript("../city-plotR.eps", horiz=F, width=9, height=6)
                          # put figure into file
par(mar=c(5,4,1,1)+0.1)   # set plot margin sizes
plot(cars.df)             # generate summary plot
dev.off()                 # close graphics device and save figure
```

These commands will generate the plot

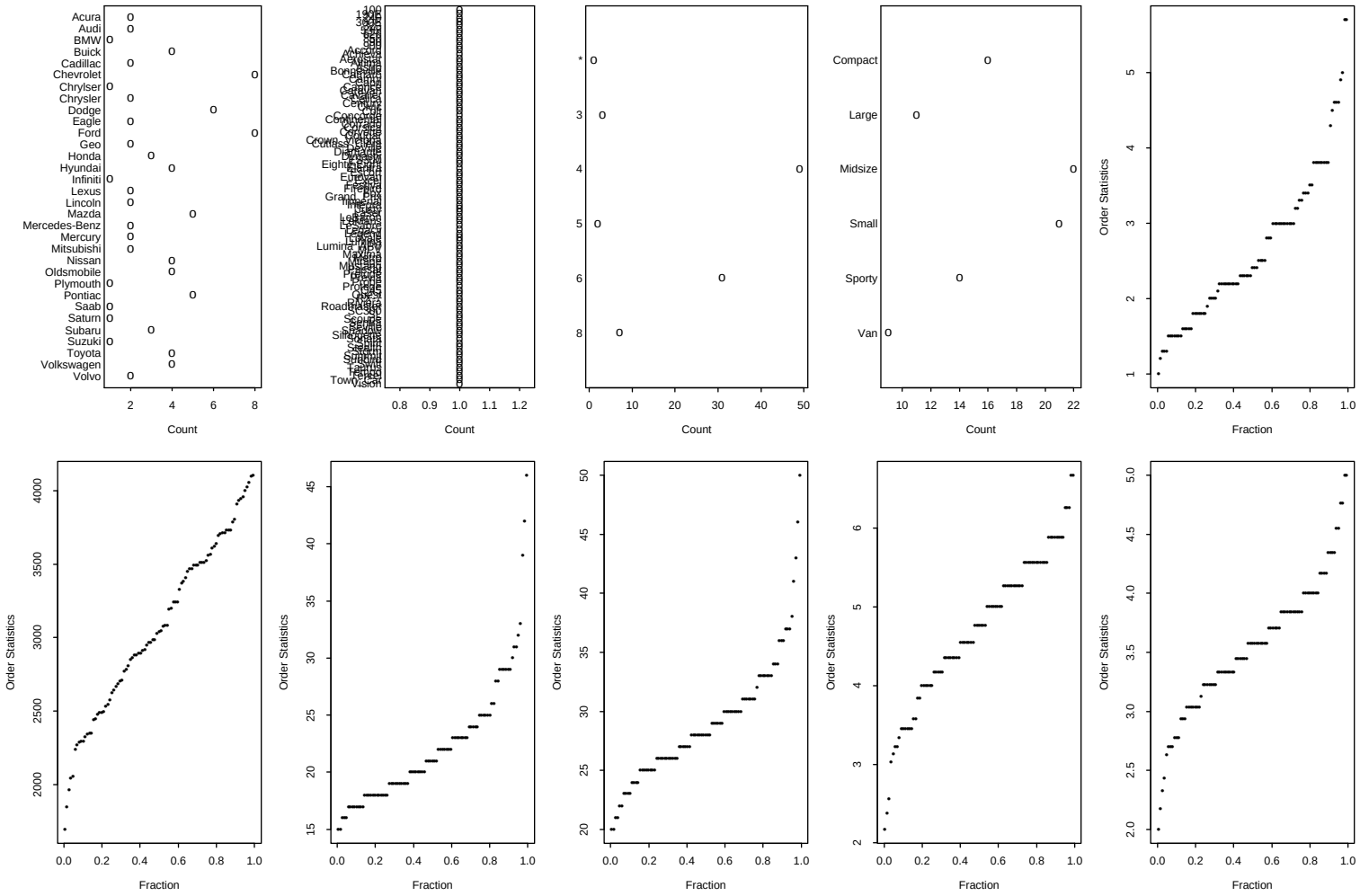

Manu
Model
Cylinder
Type
EngSize
Weight
CityMPG
HighMPG
CityFuel
HighFuel

**R** version

If we try the following similar commands in **S-Plus**

```
postscript("../city-plotS.eps", horiz=F, width=9, height=6.5)
                          # put figure into file
par(mar=c(5,4,5,1)+0.1) # set plot margin sizes
par(mfrow=c(2,5))         # set to 2 rows, 5 cols
plot(cars.df, ask=F)      # generate summary plot for each variable
dev.off()                 # close graphics device and save figure
```

we get



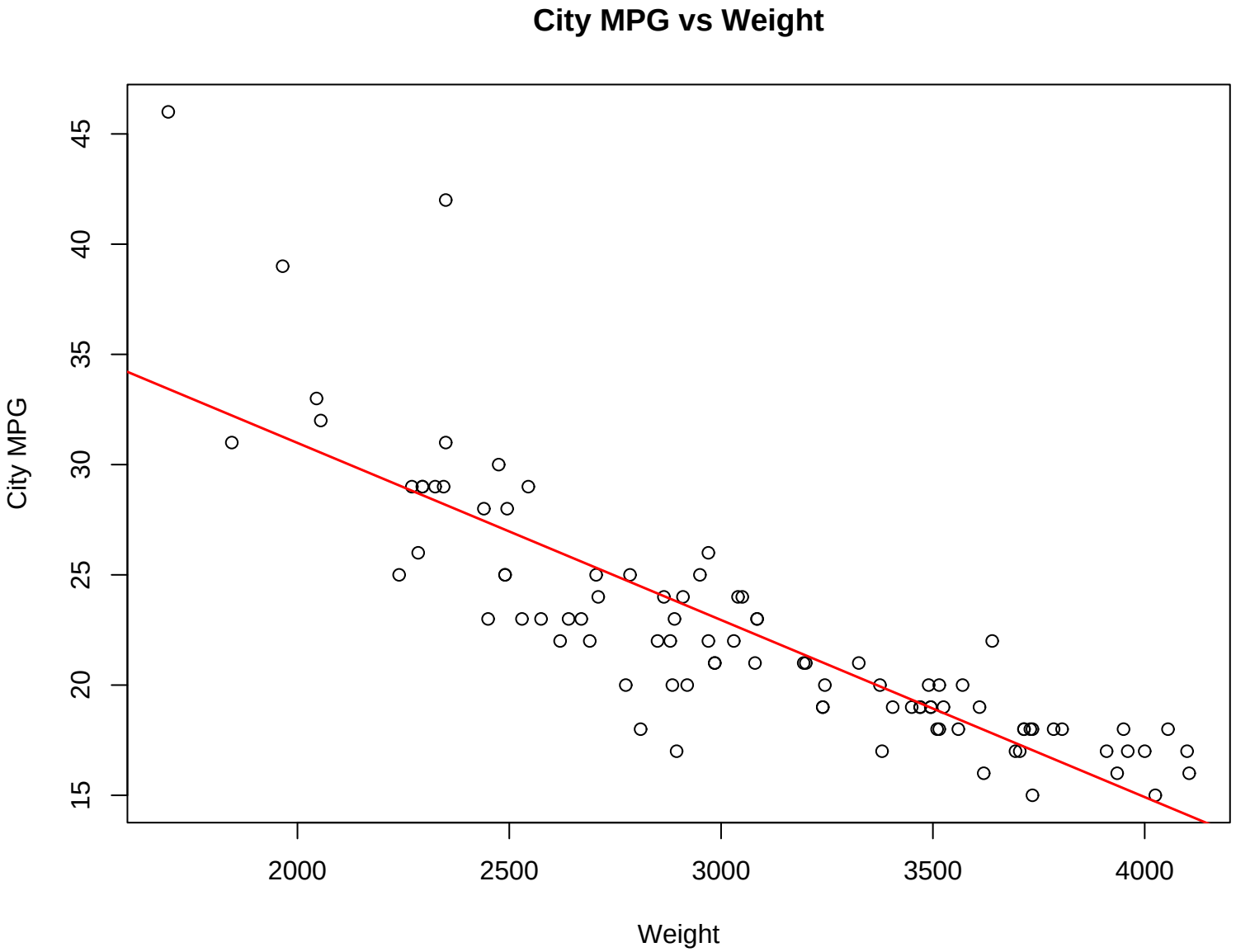

**S-Plus** version

This is an example where **S-Plus** and **R** act differently.

This example also exhibits the power of the object oriented nature of **S**. In this case, the programs recognized that a data frame was the argument of the `plot` function and used the method designed for data frames.

Now lets look at the relationship between `Weight` and `CityMPG` with the following commands (which use the `plot` command differently)

```
postscript("../citympg.eps", horiz=F, width=8, height=6.5)
par(mar=c(5,4,4,1)+0.1)
par(mfrow=c(1,1))
# generate scatter plot
plot(cars.df$Weight, cars.df$CityMPG, xlab="Weight",
  ylab="City MPG", main="City MPG vs Weight")
# add least squares line
abline(lsfit(cars.df$Weight,cars.df$CityMPG), col=2, lwd=1.5)
dev.off()
```


City MPG vs Weight
City MPG
15
20
25
30
35
40
45
2000
2500
3000
3500
4000
Weight

A linear relationship doesn't look very good. But lets run the linear regression to see what we get with the following commands

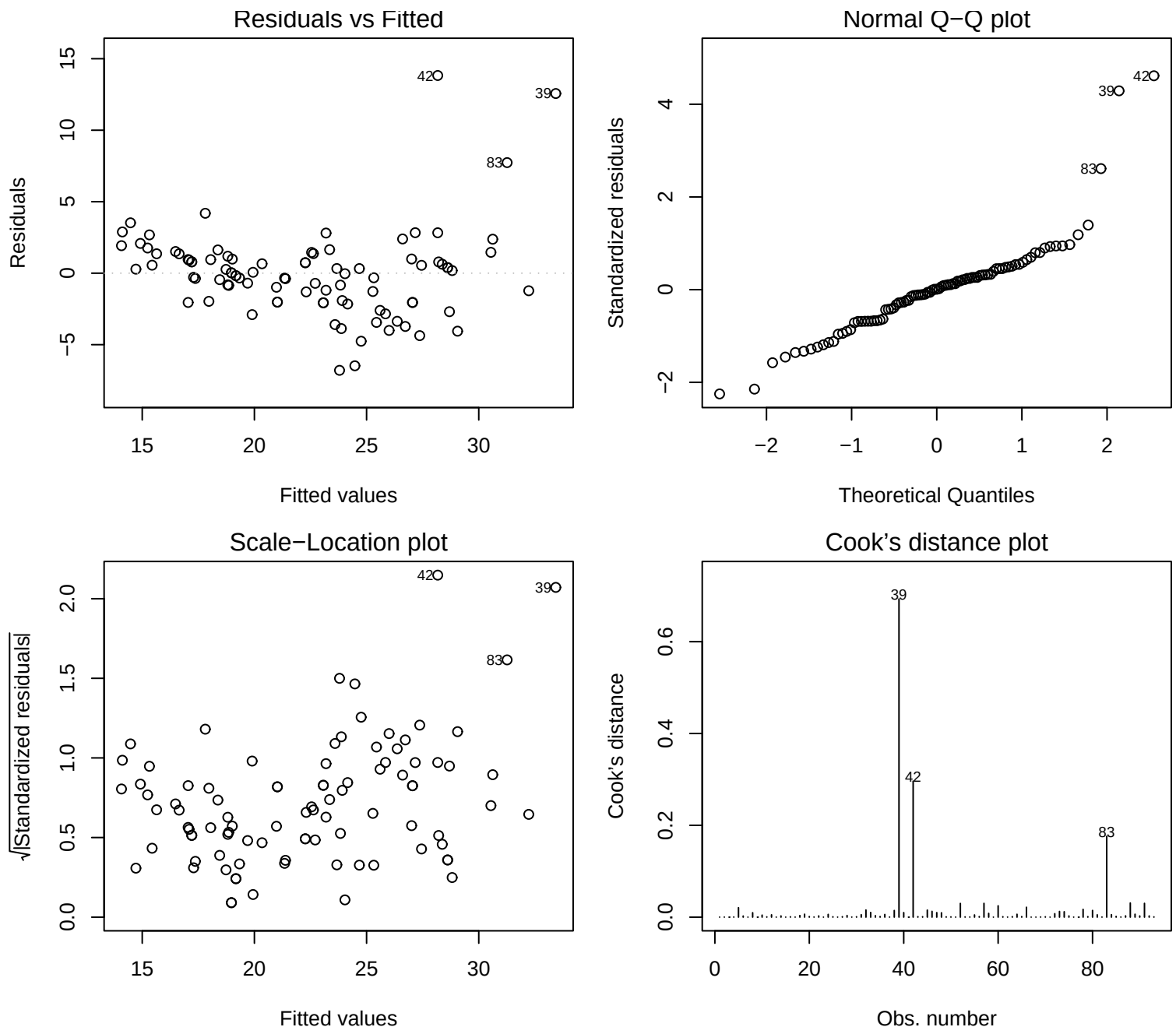
```
# Describe model
> citympg.lm <- lm(CityMPG ~ Weight, data=cars.df)

# View linear model object
> citympg.lm

Call:
lm(formula = CityMPG ~ Weight, data = cars.df)

Coefficients:
(Intercept)       Weight
  47.048353    -0.008032
```

```
# View summary of linear model object

Call:
lm(formula = CityMPG ~ Weight, data = cars.df)

Residuals:
     Min       1Q   Median       3Q      Max
-6.79458 -1.97110  0.02486  1.18550 13.82777

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 47.048353   1.679912   28.01   <2e-16 ***
Weight      -0.008032   0.000537  -14.96   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 3.038 on 91 degrees of freedom
Multiple R-Squared: 0.7109,     Adjusted R-squared: 0.7077
F-statistic: 223.8 on 1 and 91 DF,  p-value: < 2.2e-16
```

```
# Get ANOVA table
> anova(citympg.lm)
Analysis of Variance Table

Response: CityMPG
          Df  Sum Sq Mean Sq F value    Pr(>F)
Weight     1 2065.52 2065.52  223.75 < 2.2e-16 ***
Residuals 91  840.05    9.23
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Now lets see what **R** gives as default diagnotic plots for linear models with the commands

```
postscript("../citympgdiag.eps", horiz=F, width=9, height=7)
par(mar=c(5,4,1,1)+0.1, mfrow=c(2,2))
plot(citympg.lm)
dev.off()
```


Residuals vs Fitted
Residuals
Fitted values
42
39
83
Normal Q−Q plot
Standardized residuals
Theoretical Quantiles
Scale−Location plot
√|Standardized residuals|
Fitted values
Cook's distance plot
Cook's distance
Obs. number

The plots suggest that the linear assumption isn't reasonable. In addition there is a problem with the constant variance assumption and there are some points that are influential and outliers.

Lets consider a different model by transforming `CityMPG`. One possibility is

$$\text{CityFuel} = \frac{100}{\text{CityMPG}}$$

This corresponds to the number of gallons needed to go 100 miles.

This transformation is based on a couple of ideas.

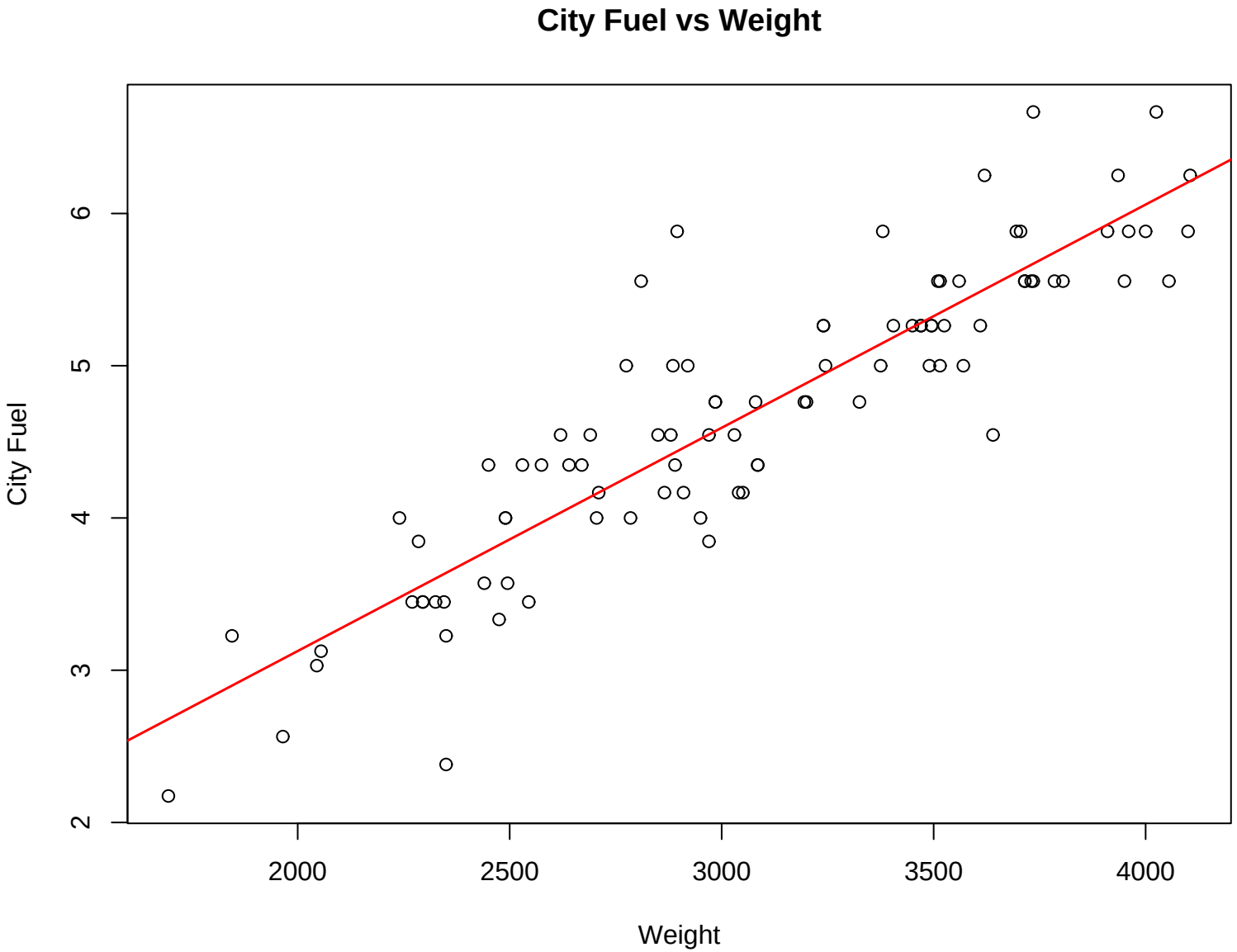
1. Most of the world reports fuel use with something like this. For example, Canada reports the number of litres to go 100 km. Europe I believe uses something similar.

2. The physics. The energy required to move an object is proportional to its weight. The energy used to move the car should be proportional to the gasoline consumption (at least as a first approximation).

Lets see if the relation between `Weight` and `CityFuel` works "better".

Again lets start with a scatterplot of the relationship between `Weight` and `CityFuel` with the commands

```
postscript("../cityfuel.eps", horiz=F, width=8, height=6.5)
par(mar=c(5,4,4,1)+0.1)
par(mfrow=c(1,1))
plot(CityFuel ~ Weight, data=cars.df, xlab="Weight",
  ylab="City Fuel", main="City Fuel vs Weight")
abline(lsfit(cars.df$Weight,cars.df$CityFuel), col=2, lwd=1.5)
dev.off()
```


City Fuel vs Weight
City Fuel
2
3
4
5
6
2000
2500
3000
3500
4000
Weight

Well this looks better. Now lets run the regression

```
> cityfuel.lm <- lm(CityFuel ~ Weight, data=cars.df)

> cityfuel.lm

Call:
lm(formula = CityFuel ~ Weight, data = cars.df)

Coefficients:
(Intercept)       Weight
   0.193667     0.001466
```

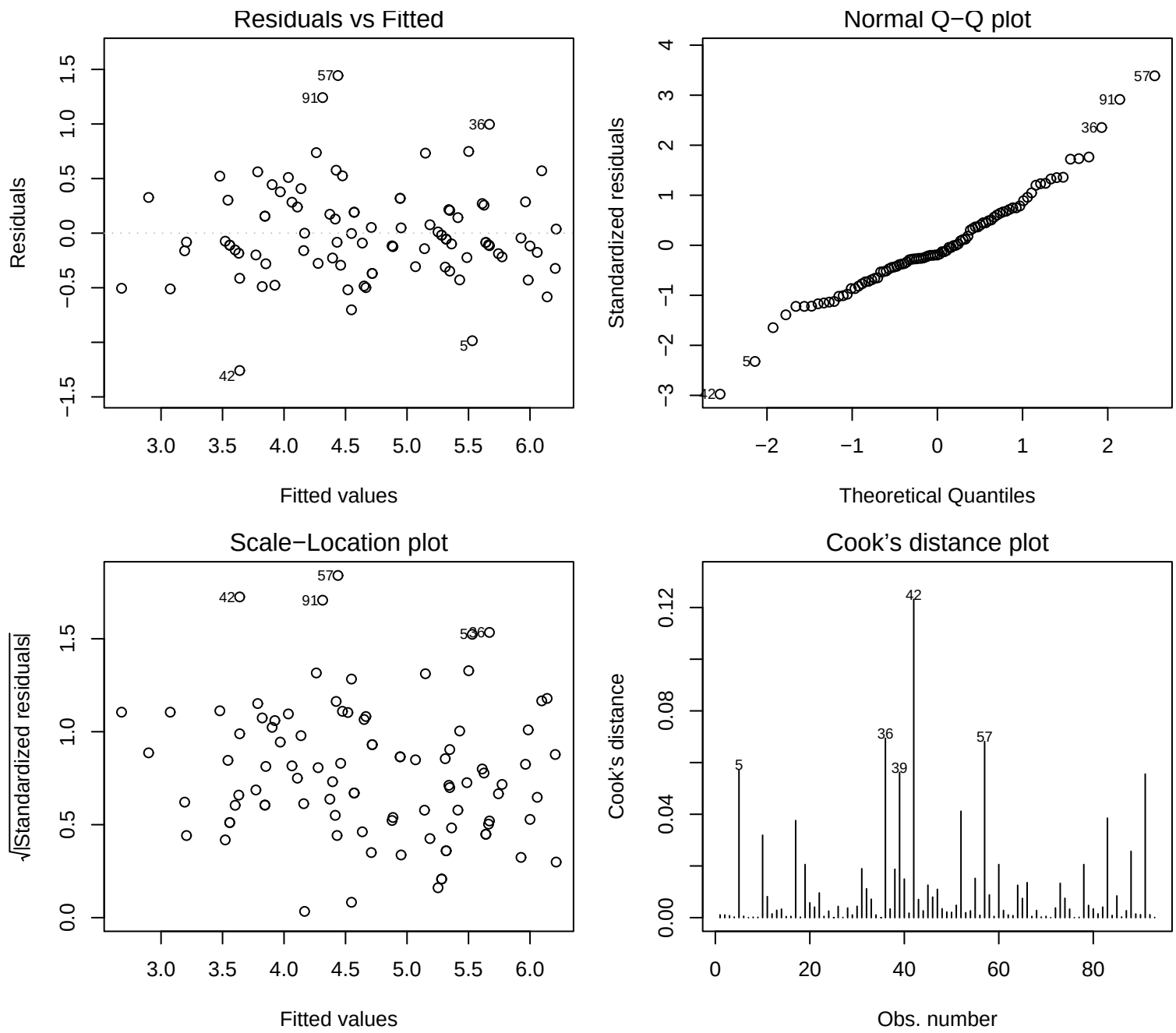
```
> summary(cityfuel.lm)

Call:
lm(formula = CityFuel ~ Weight, data = cars.df)

Residuals:
     Min       1Q   Median       3Q      Max
-1.25835 -0.22775 -0.08177  0.23933  1.44395

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 1.937e-01  2.371e-01   0.817    0.416
Weight      1.466e-03  7.579e-05  19.347   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4288 on 91 degrees of freedom
Multiple R-Squared: 0.8044,     Adjusted R-squared: 0.8023
F-statistic: 374.3 on 1 and 91 DF,  p-value: < 2.2e-16
```

```
> anova(cityfuel.lm)
Analysis of Variance Table

Response: CityFuel
          Df Sum Sq Mean Sq F value    Pr(>F)
Weight     1 68.825  68.825  374.31 < 2.2e-16 ***
Residuals 91 16.732   0.184
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

The diagnostic plots with the commands

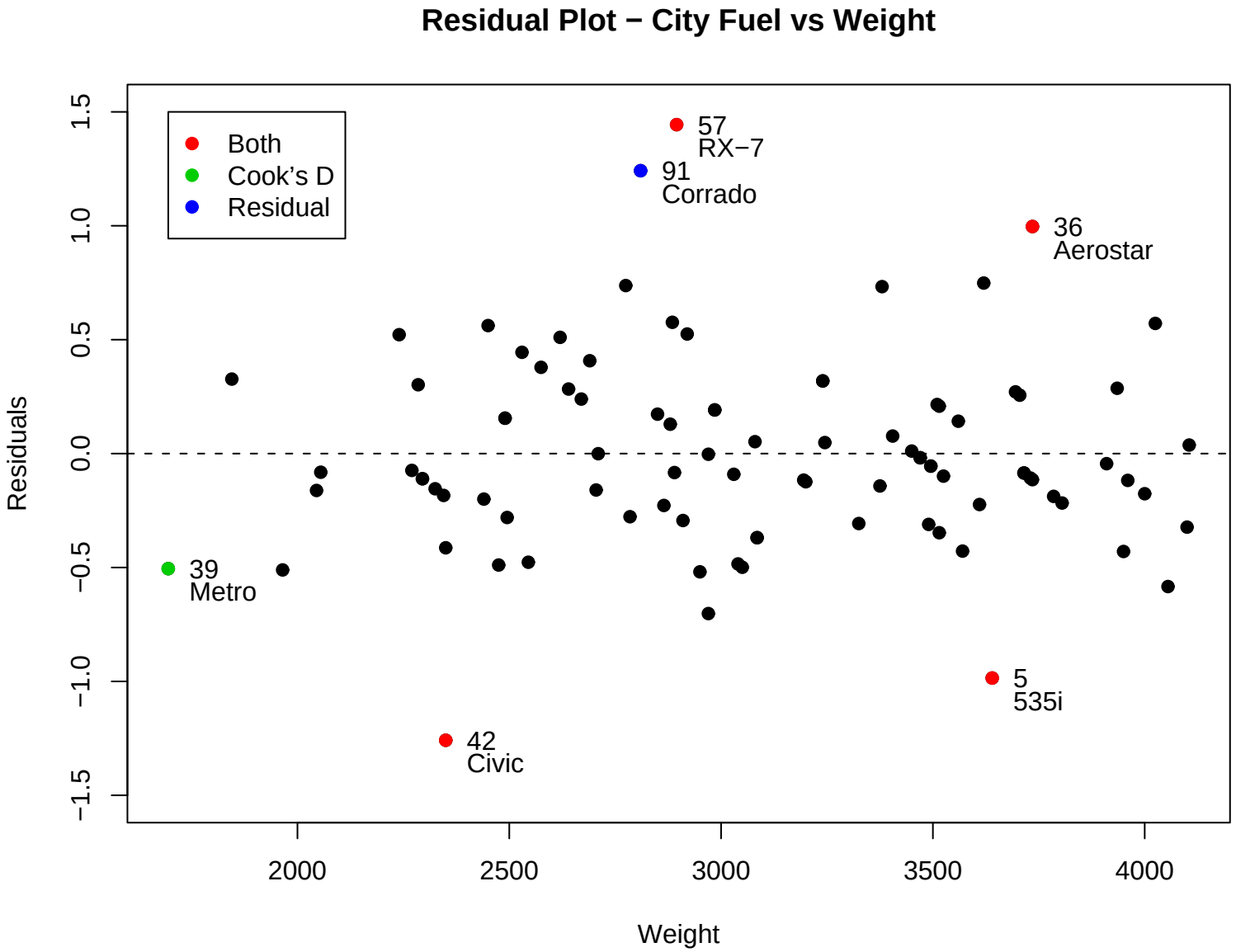
```
postscript("../cityfueldiag.eps", horiz=F, width=9, height=7)
par(mar=c(5,4,1,1)+0.1, mfrow=c(2,2))
plot(cityfuel.lm, id.n=5)   # change number of points flagged
dev.off()
```

give


Residuals vs Fitted
Residuals
Fitted values
57
91
36
5
42
Normal Q−Q plot
Standardized residuals
Theoretical Quantiles
Scale−Location plot
√|Standardized residuals|
Fitted values
Cook's distance plot
Cook's distance
Obs. number

This looks much better except for some possible outliers and influential points flagged by **R** (based on my choice of 5 points per plot). Lets see what they are

```
> cbind(cars.df,predict(cityfuel.lm),resid(cityfuel.lm))[c(5,36,39,42,57,91),]
         Manu    Model Cylinder    Type EngSize Weight CityMPG HighMPG
5         BMW     535i        4 Midsize     3.5   3640      22      30
36       Ford Aerostar        6     Van     3.0   3735      15      20
39        Geo    Metro        3   Small     1.0   1695      46      50
42      Honda    Civic        4   Small     1.5   2350      42      46
57      Mazda     RX-7        *  Sporty     1.3   2895      17      25
91 Volkswagen  Corrado        6  Sporty     2.8   2810      18      25

   CityFuel HighFuel predict(cityfuel.lm) resid(cityfuel.lm)
5  4.545455 3.333333             5.530741         -0.9852865
36 6.666667 5.000000             5.670033          0.9966338
39 2.173913 2.000000             2.678925         -0.5050122
42 2.380952 2.173913             3.639305         -1.2583530
57 5.882353 4.000000             4.438400          1.4439526
91 5.555556 4.000000             4.313771          1.2417847
```


Residual Plot − City Fuel vs Weight
Both
Cook's D
Residual
57
RX−7
91
Corrado
36
Aerostar
39
Metro
42
Civic
5
535i
Residuals
1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
2000
2500
3000
3500
4000
Weight

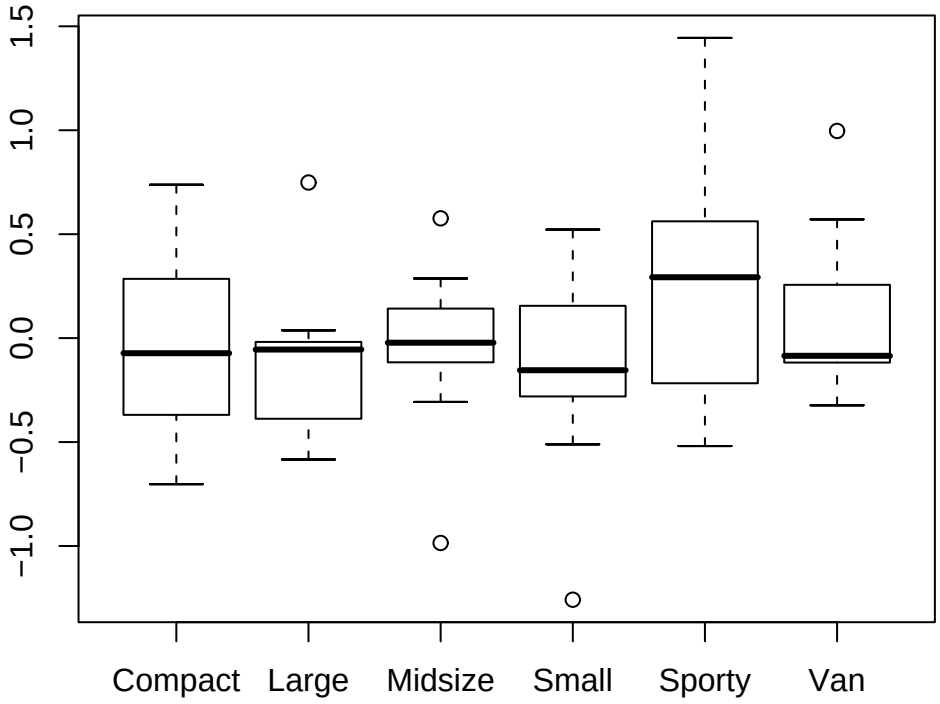
The previous plot was generated by the following commands

```
postscript("../cityfuelresid.eps", horiz=F, width=8, height=6.5)
par(mar=c(5,4,4,1)+0.1)
par(mfrow=c(1,1))
plot(cars.df$Weight, resid(cityfuel.lm), xlab="Weight", ylab="Residuals",
  main="Residual Plot - City Fuel vs Weight", ylim=c(-1.5, 1.5), pch=16)
points(cars.df$Weight[intpoints1], resid(cityfuel.lm)[intpoints1],
  pch=16, col=2)
points(cars.df$Weight[intpoints2], resid(cityfuel.lm)[intpoints2],
  pch=16, col=3)
points(cars.df$Weight[intpoints3], resid(cityfuel.lm)[intpoints3],
  pch=16, col=4)
text(cars.df$Weight[intpoints] + 50, resid(cityfuel.lm)[intpoints],
  intpoints, adj=c(0,0.5))
text(cars.df$Weight[intpoints] + 50, resid(cityfuel.lm)[intpoints] - 0.1,
  cars.df$Model[intpoints], adj=c(0,0.5))
abline(h=0, lty=2)
legend(min(cars.df$Weight), 1.5, c("Both", "Cook's D", "Residual"),
   pch=16, col=2:4)
dev.off()
```

By examining these 6 observations, I noticed a few things.

- The car with the most extreme residual (Mazda RX-7) doesn't have anything listed for Cylinder. The RX-7 has a rotary engine.

- The cars with the biggest positive residuals are a Van and 2 sports car, which shouldn't be too surprising. Maybe we should take an additional look at Type.




- The two cars with the best gas mileage, relative to their weight (Honda Civic and BMW 535i), are considered to be good cars, but nothing stands out.

The boxplot on the previous page was created with the command `boxplot(resid(cityfuel.lm) ~ cars.df$Type)`

Based on the boxplot, maybe Type could be a useful predictor. This can be examined by

```
> cityfuelt.lm <- lm(CityFuel ~ Weight + Type, data=cars.df)

> cityfuelt.lm

Call:
lm(formula = CityFuel ~ Weight + Type, data = cars.df)

Coefficients:
(Intercept)       Weight    TypeLarge  TypeMidsize    TypeSmall
   0.545480     0.001334     0.005667     0.082420    -0.163803
 TypeSporty      TypeVan
   0.312313     0.256599
```

```
> summary(cityfuelt.lm)

Call:
lm(formula = CityFuel ~ Weight + Type, data = cars.df)

Residuals:
     Min       1Q   Median       3Q      Max
-1.13523 -0.26021 -0.03129  0.22347  1.16311

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.5454797  0.4616236   1.182   0.2406
Weight       0.0013338  0.0001541   8.655 2.49e-13 ***
TypeLarge    0.0056671  0.2023308   0.028   0.9777
TypeMidsize  0.0824202  0.1556397   0.530   0.5978
TypeSmall   -0.1638025  0.1666887  -0.983   0.3285
TypeSporty   0.3123130  0.1523732   2.050   0.0434 *
TypeVan      0.2565994  0.2232962   1.149   0.2537
---
```

```
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1

Residual standard error: 0.4163 on 86 degrees of freedom
Multiple R-Squared: 0.8258,     Adjusted R-squared: 0.8137
F-statistic: 67.95 on 6 and 86 DF,  p-value: < 2.2e-16

> anova(cityfuelt.lm)
Analysis of Variance Table

Response: CityFuel
          Df Sum Sq Mean Sq  F value Pr(>F)
Weight     1 68.825  68.825 397.1471 <2e-16 ***
Type       5  1.829   0.366   2.1104 0.0718 .
Residuals 86 14.904   0.173
---
Signif. codes:  0 ’***’ 0.001 ’**’ 0.01 ’*’ 0.05 ’.’ 0.1 ’ ’ 1
```