(General) Linear Statistical Models (`lm`)

Regression

ANOVA (Analysis of Variance)

ANCOVA (Analysis of Covariance)

Note that much of what I plan to discuss will also extend to nonlinear models, such as Generalized Linear Models (`glm`), Nonlinear Least Squares (`nls`), Generalized Additive Models (`gam`), Regression Trees (`rpart`). Though of course, extensions will be needed for some of these.

Introduction to General Linear Model

References:

Montgomery DC and Peck EA. Introduction to Linear Regression Analysis, 2nd edition.

Draper NR and Smith H. Applied Regression Analysis, 3rd edition.

Neter J, Kutner MH, Nachtsheim CJ, and Wasserman W. Applied Linear Statistical Models, 4th edition.

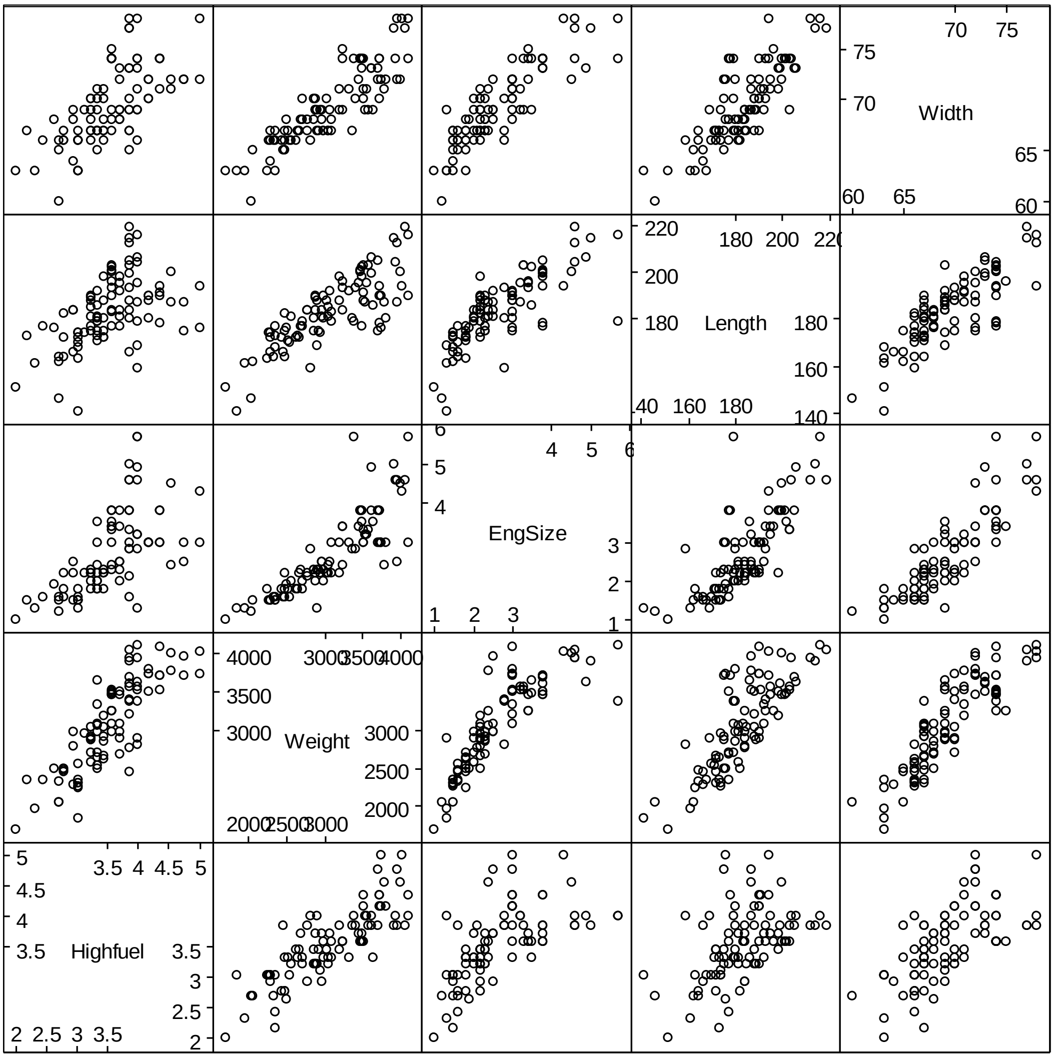
Ramsey FL and Schafer DW. The Statistical Sleuth, 2nd edition

etc

Regression (Quantitative Predictors)

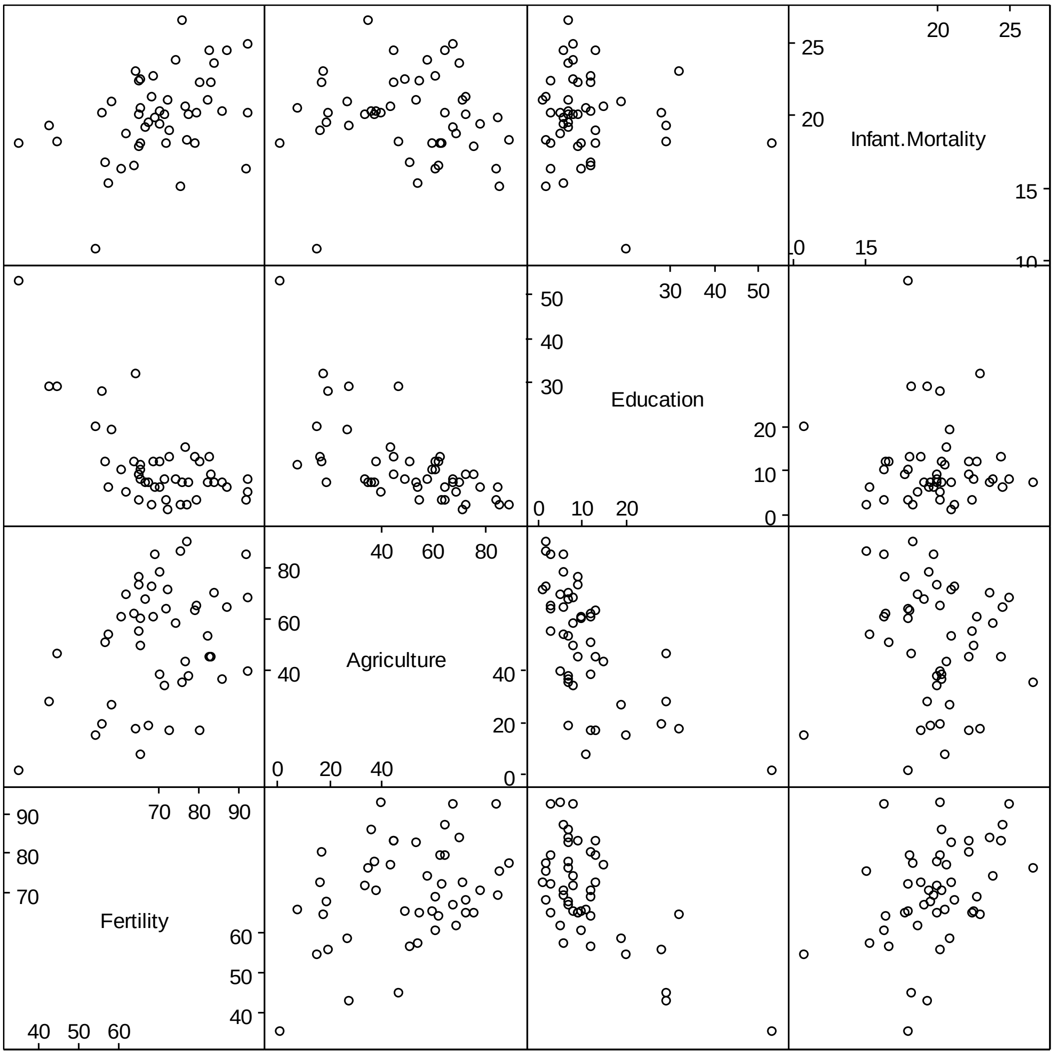
Examples:

Model EPA highway fuel (`Highfuel`) use in terms of car weight (`Weight`), engine size (`EngSize`), length (`Length`), and width (`Width`)



Scatter Plot Matrix

Model infant mortality (`Infant.Mortality`) in Switzerland in terms of education (Education), agriculture (`Agriculture`), and fertility (`Fertility`) for the dataset `swiss`.



Scatter Plot Matrix

Fit models of the form

$$Y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \ldots + \beta_k x_{ki} + \varepsilon_i ; \quad \varepsilon_i \sim \mathrm{N}(0, \sigma^2)$$

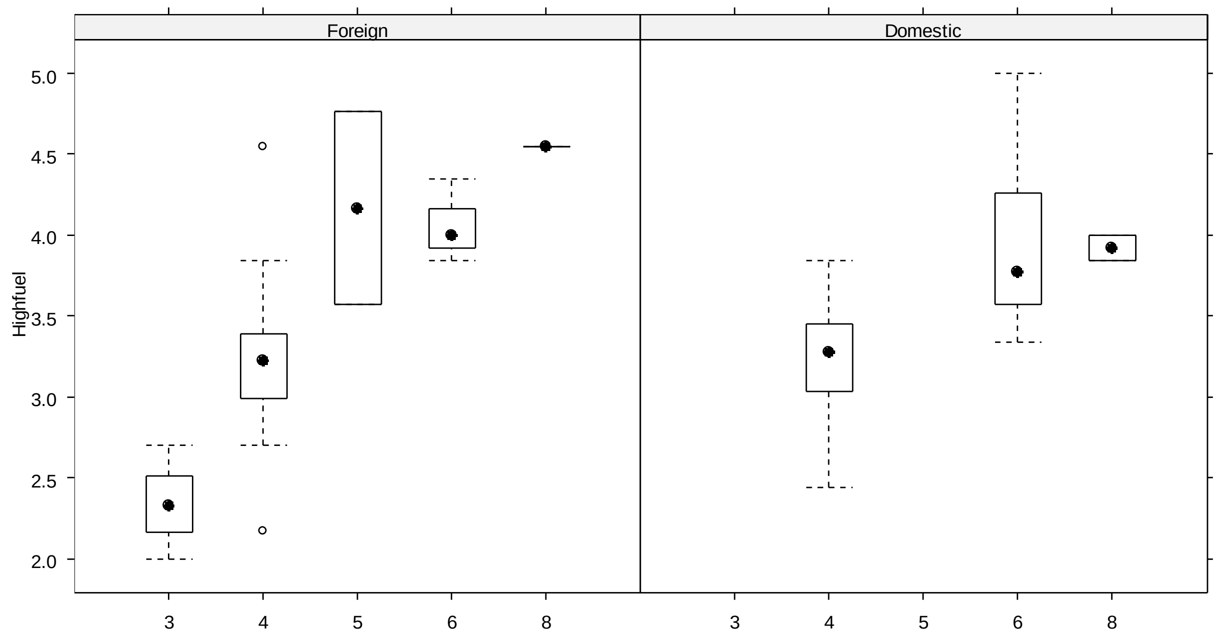
This also include polynomial regression as, for example could have $x_{ki} = x_{ji}^2$.

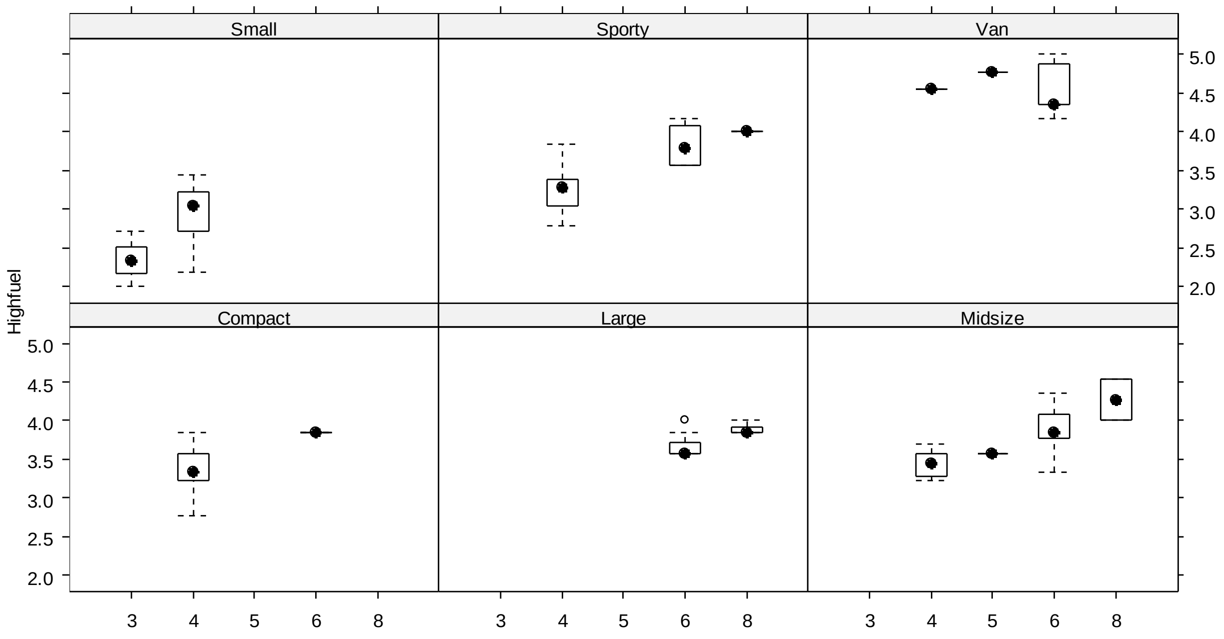
Linear regression refers to linear in the parameters, not the predictors. For example, polynomial or log transformations of the predictors is fine.

ANOVA (Qualitative predictors)

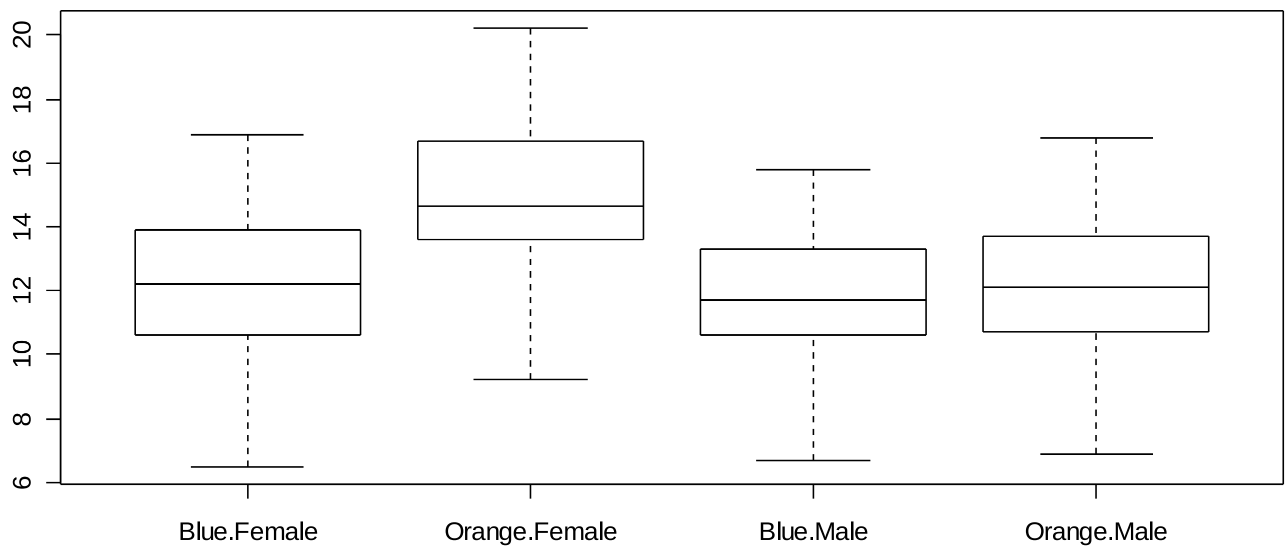
Example:

> Model EPA highway fuel (`Highfuel`) use in terms of car type (`Type`), number of cylinders (`Cylinder`), and where made (`Domestic`)





Model rear width (`RW`) of *Leptograpsus variegates* with sex (`sex`) and species (`sp`) in crabs dataset




```
boxplot(RW ~ sp * sex)
```

Fit models of the form

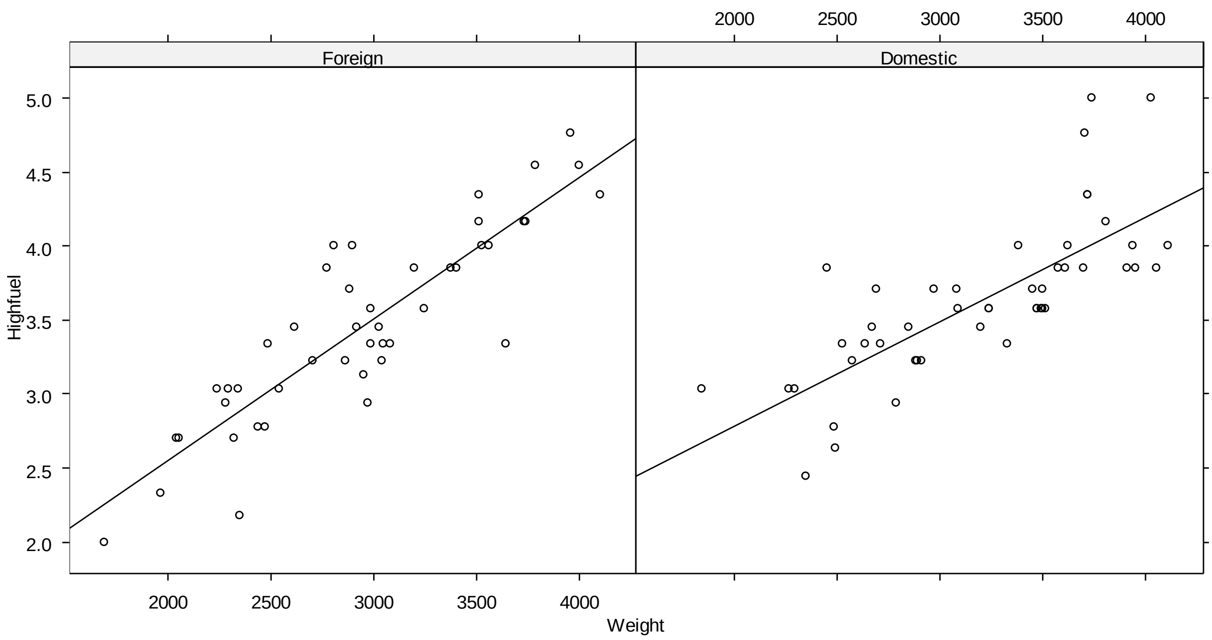
$$Y_{ijkl} = \mu + (\alpha\beta\gamma)_{jkl} + \varepsilon_{ijkl}; \quad \varepsilon_{ijkl} \sim \mathrm{N}(0, \sigma^2)$$

Have a different mean (potentially) for each combination of the factor levels.

ANCOVA (Combination of qualitative and quantitative predictors)

Example:

Model EPA highway fuel (`Highfuel`) use in terms of car weight (`Type`) and where made (`Domestic`)



Fit models of the form

$$Y_{ji} = \beta_{0j} + \beta_{1j}x_{1ji} + \beta_{2j}x_{2ji} + \ldots + \beta_{kj}x_{kji} + \varepsilon_{ji}; \quad \varepsilon_{ji} \sim \mathrm{N}(0, \sigma^2)$$

Have a different regression surface for each combination of the factor levels.

In fact these three situations are all special cases of a common model. They can all be written in the form

$$Y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \ldots + \beta_k x_{ki} + \varepsilon_i; \quad \varepsilon_i \sim \mathrm{N}(0,\ \sigma^2)$$

where the x_j are functions of the quantitative variables and levels of the qualitative variables.

There is a short hand notation for this model, which was briefly discussed in the first assignment. It can be written in matrix notation as

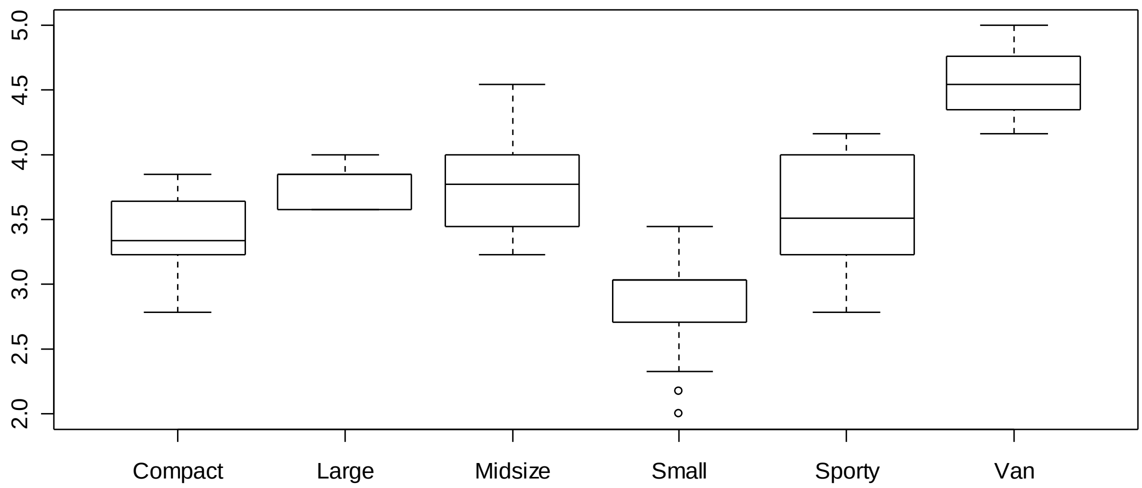
$$Y = X\beta + \varepsilon$$

where Y, β, and ε are column vectors (of length n, k + 1, and n) and X is a matrix with (n rows and k + 1 columns). The least squares estimates of β is given by

$$\hat{\beta} = \left(X^T X\right)^{-1} X^T Y$$

For example consider 1-way ANOVA, where there is one qualitative variable as a predictor.

An example of this model is the situation where fuel use is modeled by car type



This data could be described with the model

$$Y_{ji} = \mu + \alpha_j + \varepsilon_{ji}$$

It can be converted to the other form by setting

$$x_{1i} = \begin{cases} 1 & \text{car } i \text{ is Compact} \\ 0 & \text{otherwise} \end{cases}$$

$$x_{2i} = \begin{cases} 1 & \text{car } i \text{ is Large} \\ 0 & \text{otherwise} \end{cases}$$

…

$$x_{5i} = \begin{cases} 1 & \text{car } i \text{ is Sporty} \\ 0 & \text{otherwise} \end{cases}$$

Need 1 less x variable than the number of levels of categorical factor.

Note that there are other, equally valid ways, of defining x variables for this problem.

The model objects in S-Plus/R make it easy to deal with defining these other variables for fitting the model.

The basic way of defining a model is of the form

y ~ x1 + x2 + … + xk

where xj could be a qualitative variable, a quantitative variable, or a combination of variables

For example, for the Infant Mortality example

```
Infant.Mortality ~ Education + Agriculture +
Fertility
```

describes the model

$$y_i = \beta_0 + \beta_1 x_{1i} + \beta_2 x_{2i} + \beta_3 x_{3i} + \varepsilon_i$$

To fit this model we can use the `lm()` command

```
> swiss.lm <- lm(Infant.Mortality ~ Education + Agriculture +
      Fertility, data=swiss)
```

```
> summary(swiss.lm)

Call:
lm(formula = Infant.Mortality ~ Education + Agriculture +
    Fertility, data = swiss)

Residuals:
    Min      1Q  Median      3Q     Max
-8.1086 -1.3820  0.1706  1.7167  5.8039

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 10.14163    3.85882   2.628  0.01185 *
Education    0.06593    0.06602   0.999  0.32351
Agriculture -0.01755    0.02234  -0.785  0.43662
Fertility    0.14208    0.04176   3.403  0.00145 **

Residual standard error: 2.625 on 43 degrees of freedom
Multiple R-Squared: 0.2405,     Adjusted R-squared: 0.1875
F-statistic:  4.54 on 3 and 43 DF,  p-value: 0.007508

> anova(swiss.lm)

Analysis of Variance Table
Response: Infant.Mortality
            Df  Sum Sq Mean Sq F value   Pr(>F)
Education    1   3.850   3.850  0.5585 0.458920
Agriculture  1  10.215  10.215  1.4820 0.230103
Fertility    1  79.804  79.804 11.5780 0.001454 **
Residuals   43 296.386   6.893

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

For the `Highfuel` vs `Type` example

```
> type.lm <- lm(Highfuel ~ Type, data=cars93)

> summary(type.lm)

Call:
lm(formula = Highfuel ~ Type, data = cars93)

Residuals:
     Min        1Q   Median        3Q       Max
-0.87891 -0.19098  0.04712  0.22671  0.77217

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.37677    0.08886  38.002  < 2e-16 ***
TypeLarge    0.37248    0.13921   2.676  0.00891 **
TypeMidsize  0.39651    0.11678   3.395  0.00103 **
TypeSmall   -0.49786    0.11795  -4.221 5.95e-05 ***
TypeSporty   0.14754    0.13007   1.134  0.25980
TypeVan      1.20983    0.14809   8.169 2.24e-12 ***

Residual standard error: 0.3554 on 87 degrees of freedom
Multiple R-Squared: 0.658,      Adjusted R-squared: 0.6383
F-statistic: 33.48 on 5 and 87 DF,  p-value: < 2.2e-16
```

```
> anova(type.lm)

Analysis of Variance Table
Response: Highfuel
          Df  Sum Sq Mean Sq F value    Pr(>F)
Type       5 21.1446  4.2289  33.476 < 2.2e-16 ***
Residuals 87 10.9906  0.1263
```

Contrasts for factors

As mentioned earlier, there are different ways of assigning the predictor variables. S-Plus and R have 4 built in ways of handling that. The previous example was run with

```
options(contrasts=c("contr.treatment", "contr.poly"))
```

which used a parametrization similar to what I described before. The other options are

```
options(contrasts=c("contr.sum", "contr.poly"))

options(contrasts=c("contr.helmert", "contr.poly"))
```

Note that the different options give different parameter estimates, but the same fitted values, residuals, etc.

The default choice in S-Plus is

```
options(contrasts=c("contr.helmert", "contr.poly"))
```

The default in R is

```
options(contrasts=c("contr.treatment", "contr.poly"))
```

```
> options(contrasts=c("contr.sum", "contr.poly"))
> type.sum.lm <- lm(Highfuel ~ Type, data=cars93)
> summary(type.sum.lm)

Call:
lm(formula = Highfuel ~ Type, data = cars93)

Residuals:
     Min       1Q   Median       3Q      Max
-0.87891 -0.19098  0.04712  0.22671  0.77217

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.64819    0.03880  94.024  < 2e-16 ***
Type1       -0.27141    0.08228  -3.299  0.00141 **
Type2        0.10106    0.09572   1.056  0.29397
Type3        0.12510    0.07303   1.713  0.09029 .
Type4       -0.76928    0.07427 -10.358  < 2e-16 ***
Type5       -0.12388    0.08672  -1.428  0.15676

Residual standard error: 0.3554 on 87 degrees of freedom
Multiple R-Squared: 0.658,      Adjusted R-squared: 0.6383
F-statistic: 33.48 on 5 and 87 DF,  p-value: < 2.2e-16
> anova(type.sum.lm)
Analysis of Variance Table
Response: Highfuel
          Df  Sum Sq Mean Sq F value    Pr(>F)
Type       5 21.1446  4.2289  33.476 < 2.2e-16 ***
Residuals 87 10.9906  0.1263
```

```
> options(contrasts=c("contr.sum", "contr.poly"))

> contrasts(cars93$Type)

        [,1] [,2] [,3] [,4] [,5]
Compact    1    0    0    0    0
Large      0    1    0    0    0
Midsize    0    0    1    0    0
Small      0    0    0    1    0
Sporty     0    0    0    0    1
Van       -1   -1   -1   -1   -1

> options(contrasts=c("contr.treatment", "contr.poly"))

> contrasts(cars93$Type)

        Large Midsize Small Sporty Van

Compact     0       0     0      0   0
Large       1       0     0      0   0
Midsize     0       1     0      0   0
Small       0       0     1      0   0
Sporty      0       0     0      1   0
Van         0       0     0      0   1

> options(contrasts=c("contr.helmert", "contr.poly"))

> contrasts(cars93$Type)

        [,1] [,2] [,3] [,4] [,5]
Compact   -1   -1   -1   -1   -1
Large      1   -1   -1   -1   -1
Midsize    0    2   -1   -1   -1
Small      0    0    3   -1   -1
Sporty     0    0    0    4   -1
Van        0    0    0    0    5
```

Unordered vs Ordered factors

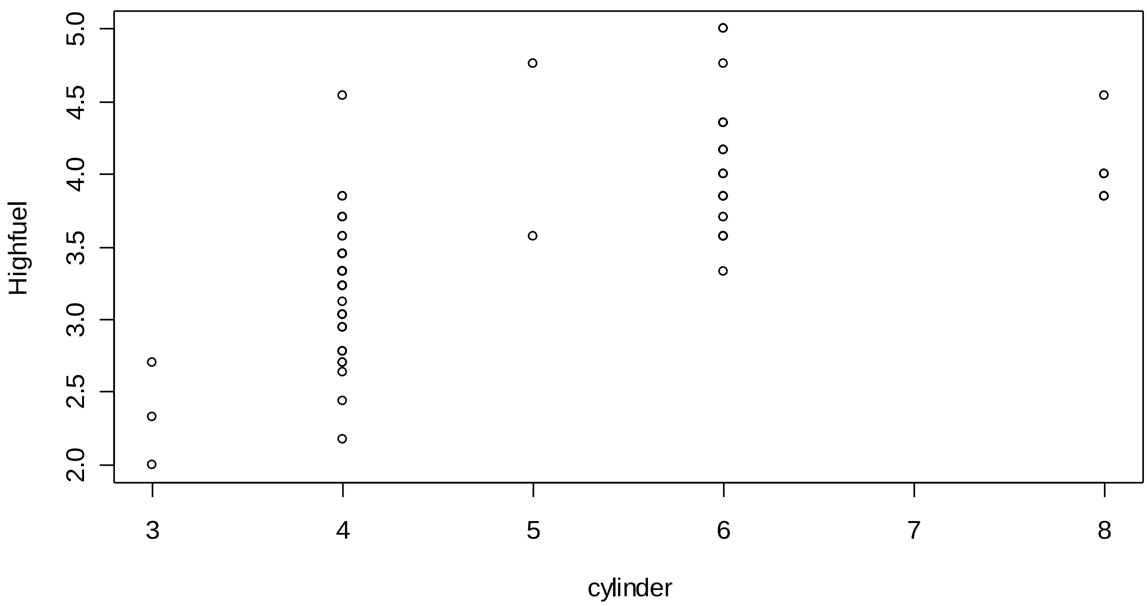
Some categorical variables have a natural ordering to them, such as the number of cylinders in an engine. Most categorical variables don’t, for example regligon. You might have fun arguing with people on how to order Christianity, Islam, Judism, Shinto, etc.

In the case where order makes sense, S-Plus/R has a set of contrast which allow for looking trends. They are based on orthogonal polynomials, assuming the levels are equally spaced.

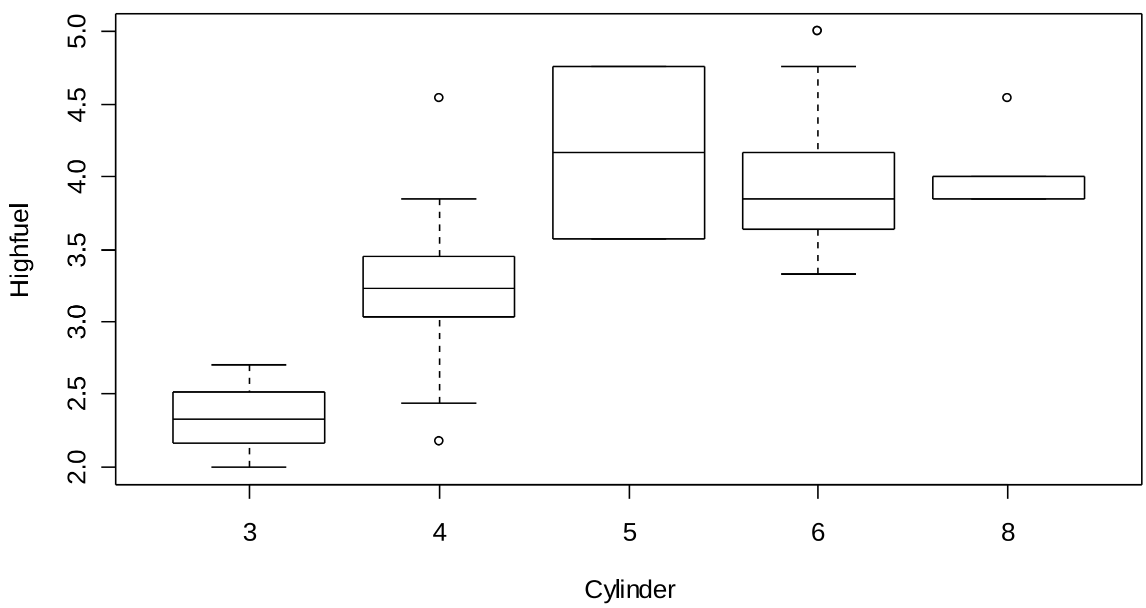
```
> contrasts(cars93$Cylinder)

  4 5 6 8
3 0 0 0 0
4 1 0 0 0
5 0 1 0 0
6 0 0 1 0
8 0 0 0 1

> cars93$CylinderO <- as.ordered(cars93$Cylinder)

> contrasts(cars93$CylinderO)

             .L         .Q            .C         ^4
3 -6.324555e-01  0.5345225 -3.162278e-01  0.1195229
4 -3.162278e-01 -0.2672612  6.324555e-01 -0.4780914
5 -3.287978e-17 -0.5345225  1.595204e-16  0.7171372
6  3.162278e-01 -0.2672612 -6.324555e-01 -0.4780914
8  6.324555e-01  0.5345225  3.162278e-01  0.1195229

> summary(Cylinder.ord.lm)

Call:
lm(formula = Highfuel ~ CylinderO, data = cars93)

Residuals:
      Min         1Q    Median        3Q       Max
-1.056216 -0.221020 -0.004322  0.218147  1.315326

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  3.54797    0.08237  43.073  < 2e-16 ***
CylinderO.L  1.29547    0.17964   7.211 1.92e-10 ***
CylinderO.Q -0.75963    0.21588  -3.519 0.000692 ***
CylinderO.C  0.04834    0.10641   0.454 0.650763
CylinderO^4  0.29654    0.21331   1.390 0.168023
```

Numeric vs factors

```
plot(Highfuel ~ cylinder,data=cars93)
```




```
plot(Highfuel ~ Cylinder,data=cars93)
```



```
> is.factor(cars93$cylinder)

[1] FALSE

> is.factor(cars93$Cylinder)

[1] TRUE

> is.numeric(cars93$cylinder)

[1] TRUE

> is.numeric(cars93$Cylinder)

[1] FALSE
```

The way that numeric variables and factors are treated in model definitions is different. S-Plus/R recognizes how each of the variables is defined and does the appropriate thing. Note you do need to be careful, as when data is read in, for example with `read.table`, assumptions are made about how each variable is defined. For example, any variable that appears to be numeric, is classified as numeric. If its really a factor, it will then need to be reassigned as a factor with the `as.factor` command. Labels can be giving to the various factor labels with the `levels` command.

```
> summary(cylinder.lm)

Call:

lm(formula = Highfuel ~ cylinder, data = cars93)

Residuals:

      Min         1Q    Median         3Q        Max
-1.074287 -0.272838  0.001887  0.200075  1.297254

Coefficients:

            Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.0561     0.1853  11.095  < 2e-16 ***
cylinder      0.2980     0.0361   8.257 1.20e-12 ***

Residual standard error: 0.4492 on 90 degrees of freedom

Multiple R-Squared: 0.431,      Adjusted R-squared: 0.4247

F-statistic: 68.17 on 1 and 90 DF,  p-value: 1.202e-12

> summary(Cylinder.lm)

Call:

lm(formula = Highfuel ~ Cylinder, data = cars93)

Residuals:

      Min         1Q    Median         3Q        Max
-1.056216 -0.221020 -0.004322  0.218147  1.315326

Coefficients:

            Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.3428     0.2344   9.994 4.17e-16 ***
Cylinder4     0.8874     0.2415   3.674 0.000411 ***
Cylinder5     1.8239     0.3707   4.921 4.05e-06 ***
Cylinder6     1.6455     0.2455   6.703 1.95e-09 ***
Cylinder8     1.6692     0.2802   5.957 5.31e-08 ***

Residual standard error: 0.406 on 87 degrees of freedom

Multiple R-Squared: 0.5507,     Adjusted R-squared:  0.53

F-statistic: 26.66 on 4 and 87 DF,  p-value: 1.92e-14
```

Interactions

Look at two models fit with `Weight` and `Domestic`

```
> summary(weight.domestic.lm)

Call:
lm(formula = Highfuel ~ Weight + Domestic, data = cars93)

Residuals:
      Min         1Q    Median         3Q        Max
-0.781506 -0.244967  0.002068  0.180682  0.922104

Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)       9.923e-01  1.853e-01   5.355  6.5e-07 ***
Weight            8.354e-04  6.065e-05  13.774  < 2e-16 ***
DomesticDomestic -3.449e-02  7.120e-02  -0.484    0.629

> summary(weight.domestic.int.lm)

Call:
lm(formula = Highfuel ~ Weight * Domestic, data = cars93)

Residuals:

     Min       1Q   Median       3Q      Max
-0.78647 -0.21346 -0.03952  0.17163  0.99145

Coefficients:

                       Estimate Std. Error t value Pr(>|t|)
(Intercept)           6.264e-01  2.504e-01   2.501   0.0142 *
Weight                9.597e-04  8.347e-05  11.498   <2e-16***
DomesticDomestic      7.421e-01  3.721e-01   1.994   0.0492 *
Weight:DomesticDomes -2.529e-04  1.190e-04  -2.125   0.0364 *
```

Both models fit give regression lines for `Highfuel` vs `Weight`. The first one fits a model of the form

$$y_i = \beta_0 + \beta_1 w_i + \beta_2 d_i + \varepsilon_i$$

The second model is of the form

$$y_i = \beta_0 + \beta_1 w_i + \beta_2 d_i + \beta_3 w_i d_i + \varepsilon_i$$

where d_i is 1 for domestic cars and 0 for foreign cars

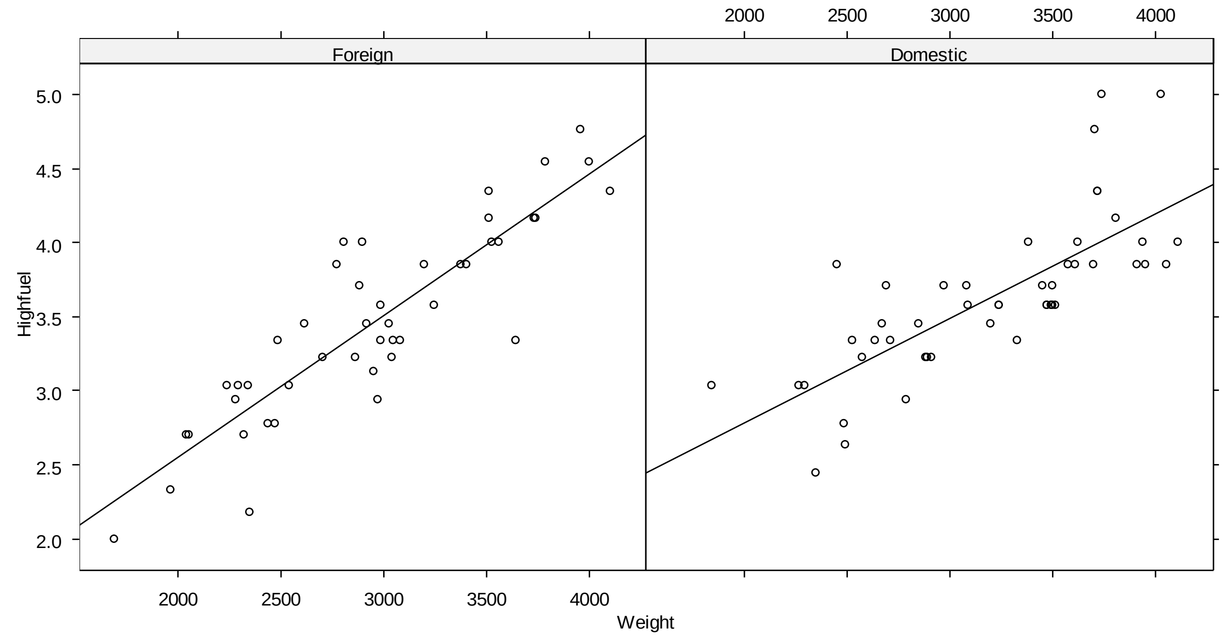
These can be rewritten as

$$y_i = \begin{cases} \beta_0 + \beta_1 w_i + \varepsilon_i & \text{Foreign car} \\ \underbrace{(\beta_0 + \beta_2)}_{\beta_0^*} + \beta_1 w_i + \varepsilon_i & \text{Domestic car} \end{cases}$$

and

$$y_i = \begin{cases} \beta_0 + \beta_1 w_i + \varepsilon_i & \text{Foreign car} \\ \underbrace{(\beta_0 + \beta_2)}_{\beta_0^*} + \underbrace{(\beta_1 + \beta_3)}_{\beta_1^*} w_i + \varepsilon_i & \text{Domestic car} \end{cases}$$

This second model has an example of an interaction. In this case the effect of weight differs depending on whether the car is domestically made or not. It fitting the equivalent on what was displayed in the figure



Interactions in models can be indicated with : and *.

A : means just the interaction of interest. A * means that interaction plus all lower level interactions and main effects, eg.

```
y ~ A*B
```

is the same as

```
y ~ A + B + A:B
```

Note you can fit the model

```
y ~ A:B
```

but you usually don't want to. It corresponds to the regression equation

$$y_i = \beta_0 + \beta_1 a_i b_i + \varepsilon_i$$

Note that A and B in the above can be any combinations of numerical variables and factors

Suppose you wanted to fit the model

```
y ~ A + B + C + A:B + A:C + B:C
```

A short hand for this model is

```
y ~ (A + B + C)^2
```

This will not pick up and A:A type term as it is regarded the same as A. For factors, this is the correct thing to do. However it may not be what you want with numeric predictors. For example you might want to fit the model

$$y_i = \beta_0 + \beta_1 w_i + \beta_2 w_i^2 + \varepsilon_i$$

To do this you would need a model statement like

```
y ~ w + I(w^2)
```

The function `I` inhibits the interpretation or conversion of objects.

Note that is S-Plus the `I` isn't needed but in R it is

S-Plus:

```
> lm(y~x+x^2)
Call:
lm(formula = y ~ x + x^2)

Coefficients:
 (Intercept)           x      I(x^2)
 -0.03241224 -0.09234361 0.008718185
```

R:

```
> lm(y~x+x^2)

Call:
lm(formula = y ~ x + x^2)

Coefficients:
(Intercept)            x
   -0.12723      0.02098
```

```
> summary(weight2.lm)

Call:
lm(formula = Highfuel ~ Weight + I(Weight^2), data = cars93)

Residuals:
     Min       1Q   Median       3Q      Max
-0.76605 -0.23896  0.01345  0.19332  0.91241

Coefficients:
              Estimate Std. Error t value Pr(>|t|)
(Intercept)  8.278e-01  8.696e-01   0.952    0.344
Weight       9.430e-04  5.855e-04   1.611    0.111
I(Weight^2) -1.879e-08  9.607e-08  -0.196    0.845

Residual standard error: 0.3355 on 90 degrees of freedom
Multiple R-Squared: 0.6848,     Adjusted R-squared: 0.6778
F-statistic: 97.78 on 2 and 90 DF,  p-value: < 2.2e-16
```

Removing terms from models

It is also possible to remove terms from models. For example

```
y ~ A + B + C + A:B + A:C + B:C
```

could also have been written as

```
y ~ A*B*C – A:B:C
```

so it can be used as a short hand to write more complicated models.

Another situation where it is more useful is to compare two models. Lets go back to the crab example and compare two models

```
RW ~ sex * sp
```

and

```
RW ~ sex + sp
```

One way of doing this is by

```
> crab.int.lm <- lm(RW ~ sex * sp)
> crab.add.lm <- update(crab.int.lm, . ~ . - sex:sp)
```

Note that is could also be done with

```
> crab.add2.lm <- lm(RW ~ sex + sp)
> crab.int2.lm <- update(crab.add2.lm, . ~ . + sex:sp)
```

To see whether the interaction model gives a better description we can look at the command

```
> anova(crab.add.lm,crab.int.lm)

Analysis of Variance Table


Model 1: RW ~ sex + sp

Model 2: RW ~ sex * sp

  Res.Df    RSS  Df Sum of Sq       F    Pr(>F)

1    197 1074.4

2    196 1016.4   1      58.0 11.184 0.0009884 ***
```

Another situation where removing a term can be useful is to get rid of the intercept. For example to fit a regression through the origin you can do

```
> weight.orig.lm <- lm(Highfuel ~ Weight - 1, cars93)

> summary(weight.orig.lm)

Call:
lm(formula = Highfuel ~ Weight - 1, data = cars93)

Residuals:
      Min        1Q    Median        3Q       Max
-0.820327 -0.227628 -0.009304  0.320788  1.050421

Coefficients:
        Estimate Std. Error t value Pr(>|t|)
Weight 1.141e-03  1.263e-05   90.33   <2e-16 ***
```

```
> anova(weight.orig.lm,weight.lm)

Analysis of Variance Table

Model 1: Highfuel ~ Weight - 1

Model 2: Highfuel ~ Weight

  Res.Df     RSS Df Sum of Sq      F    Pr(>F)
1     92 13.3643
2     91 10.1325  1    3.2318 29.025 5.564e-07 ***
```

Removing the intercept is also useful in some ANOVA models as it gives a different parametrization.

For example

```
> type.lm

Call:

lm(formula = Highfuel ~ Type, data = cars93)

Coefficients:

(Intercept)    TypeLarge  TypeMidsize    TypeSmall
     3.3768       0.3725       0.3965      -0.4979

TypeSporty      TypeVan
0.1475       1.2098

> type.noint.lm

Call:

lm(formula = Highfuel ~ Type - 1, data = cars93)

Coefficients:

TypeCompact    TypeLarge  TypeMidsize    TypeSmall
      3.377        3.749        3.773        2.879

TypeSporty      TypeVan
3.524        4.587
```

In the first the intercept is the mean for Compact cars and the others are the deviations for the other types. In the second, each is the mean for that type

Another example is

```
> weight.domestic.int.lm

Call:

lm(formula = Highfuel ~ Weight * Domestic, data = cars93)

Coefficients:

            (Intercept)            Weight     DomesticDomestic
              0.6263581         0.0009597            0.7420544

Weight:DomesticDomestic
             -0.0002529

> weight.domestic.int2.lm

Call:

lm(formula = Highfuel ~ Domestic/Weight - 1, data = cars93)

Coefficients:

        DomesticForeign         DomesticDomestic
              0.6263581                1.3684125

DomesticForeign:Weight           DomesticDomestic:Weight
             0.0009597                         0.0007069
```

The first gives the differences in the intercept and slope for domestic cars from foreign cars where the second gives the intercept and slope for domestic cars.

The `/` is another way of describing interactions. The form is `a / x`, where `a` is a factor and `x` could be numeric, a factor, or a combination of things. This model says fit the model described by `x` for each level of `a`. The specification `a/x - 1` is equivalent to `a + a:x - 1` in terms of parametrization.

Prediction

It is easy to make predictions for new or hypothesized observations with the `predict` command. The form of the function is `predict(fit, newdata)`, where `fit` is result of the `lm` command and `newdata` is a `dataframe` including all the variables used in fitting the model. For example

```
> newdata

  Weight Domestic

1   2000  Foreign

2   3000 Domestic

3   4000  Foreign

4   2000 Domestic

5   3000  Foreign

6   4000 Domestic

> predict(weight.domestic.int.lm,newdata)

       1        2        3        4        5        6

2.545835 3.489005 4.465312 2.782141 3.505573 4.195869
```

Also to exhibit that different parametrizations give the same fitted values

```
> predict(weight.domestic.int.lm,newdata)
       1        2        3        4        5        6
2.545835 3.489005 4.465312 2.782141 3.505573 4.195869
> predict(weight.domestic.int2.lm,newdata)
       1        2        3        4        5        6
2.545835 3.489005 4.465312 2.782141 3.505573 4.195869
> weight.domestic.int.lm
Call:
lm(formula = Highfuel ~ Weight * Domestic, data = cars93)
Coefficients:
            (Intercept)                  Weight        DomesticDomestic
              0.6263581               0.0009597               0.7420544
Weight:DomesticDomestic
             -0.0002529
> weight.domestic.int2.lm
Call:
lm(formula = Highfuel ~ Domestic/Weight - 1, data = cars93)
Coefficients:
   DomesticForeign    DomesticDomestic   DomesticForeign:Weight
         0.6263581           1.3684125                0.0009597
DomesticDomestic:Weight
              0.0007069
```