Introduction to Robust Statistics

Klaus Nordhausen

Department of Mathematics and Statistics
University of Turku

Autumn 2015

Linear Regression in R

There are a lot of functions for regression specifications in R. The general idea for all of them is, that we fit a regression for a model formula specified in the function. The usual way is to assign the regression result to an object. All necessary information can then be extracted from the regression object.

The main regression functions are:

- `lm` main regression function, for the standard regression model
- `aov` special version of `lm`, calls the function `lm` for each stratum of a factor (Output looks like classical ANOVA, is for balanced designs)
- `gls` fits a general linear regression (belongs to the library nlme, fits models with unequal error variances or correlated errors)
- `glm` fits a generalized linear model (observations can have other distributions than normal)

Functions for Regression in R

There are a lot of functions in R for regression objects, they extract, sometimes depending on the regression type (generic functions), information from the regression object. The most important ones are:

- `summary` basic information of the model
- `coef` parameter estimates
- `fitted` fitted values
- `predict` makes predictions
- `residuals` all types of residuals
- `anova` ANOVA table
- ...

The following slides will explain more detailed how they work for `lm` type objects.

The lm function

The function `lm` is the function for the basic linear model. Its usage is

```
lm(formula, data, subset, weights, na.action,
method = "qr", model = TRUE, x = FALSE, y = FALSE, qr = TRUE
singular.ok = TRUE, contrasts = NULL, offset, ...)
```

Have we assigned a `lm` function to an object we can straight extract from there certain results using indexing. E.g. coefficients, residuals, fitted.values, rank, weights, df.residual, call, terms, contrasts, xlevels, y. But often the same with more options can be obtained using generic functions.

Statistical models in R

Summary statistics give only a glimpse at the data and often of an analysis inference and or modeling is the actual goal. R provides a lot of statistical tests as well as a lot of modeling functions. Before we can however use them we have to learn something about R's formulae definitions to be able to define models in R.

A basic formula in R has the form

```
y ~ x1 + x2 + x3
```

Where the part left of ~ is the dependent variable and the right part defines the independent variables.

Formulae and intercept

The intercept in a model formula is represented by a `1`. By default R assumes that an intercept is present, therefore mentioning the intercept or not makes no difference. If however the intercept should be removed a `-1` is needed in the formula.

These two models are equivalent, both have an intercept:

```
y ~ x1 + x2    and    y ~ x1 + x2 + 1
```

The same model without intercept must be defined as:

```
y ~ x1 + x2 - 1
```

Interactions and nested designs

Often in statistical models interactions between variables are suspected or variables are nested. This can be formulated also using R formulae. Several special operators are available for this. To name a few:

- `:`
  Used for interactions like `x1 : x2`
- `*`
  Main effects plus interactions, like `x1 * x2 = x1 + x2 + x1 : x2`.
- `^`
  Factor crossing up to a certain degree, like
  `(x1+x2+x3)^2 = x1+x2+x3 + x1:x2 + x1:x3 + x2:x3`.
- `-`
  Removing terms, like
  `(x1+x2+x3)^2 - x2:x3 = x1+x2+x3 + x1:x2 + x1:x3`.

Variable transformations in formulae

Common practice is use transformations of variables in statistical models. This can be done in R directly in the model formula. For example:

`log(y) ~ x1 + x2 + sin(x3)` is a correct formula.

However, due to the definition of interactions and so on, the special function `I` is of interest here. This function interprets the operators used inside it as expressions in their original meaning. For example:

- `y~I(x1-1)` extracts from x1 one unit before it enters the model and not the intercept. This is therefore different from `y~x1-1`.
- `y~I(x1^2)` squares variable x1 and has nothing to do with factor crossing.

lm Regression Objects

Assume we fitted with an appropriate model formula a regression model using the function `lm` and assigned that to the object `lm.out`. Then a lot of functions have a generic output when applied to this object. What exactly these functions are doing can be explored using the help. If we are for example interested to know what `summary` does to an `lm` object, we can ask the help for this by using `?summary.lm`. In general, for any generic function the specific help can be obtained this way.

If we just ask for the `lm.out` object we get only minimal output. That is the model formula and the estimated parameters.

update

After creating an regression object one often wants to do only a small change, like changing the contrast or removing or adding a variable. One could of course then just call the regression function again and make the changes there, but one could also use the function `update`. This function applies then on the old object the change which we defined in the `update` function.

Using for example +/- we could add or take independent variables to/ out from the model.

Assume `lm.out` contains the independent variables `x1` and `x2`. then we could add `x3` using:

```
lm.out.add <- update(lm.out, . ~ . + x3)
```

or we take variable x2 out

```
lm.out.minus <- update(lm.out, . ~ . - x2)
```

summary

The `summary` of a `lm` object is normally the first you look at.

It provides you with:

- the model formula
- a 5-point statistic for the residuals
- the parameter estimates including there standard errors, t-test statistics and their p-values
- the residual standard error with its df (the residual standard error is the estimate for s, which is the error variance)
- R^2 and RA^2
- the F-test for all versus only the intercept

anova for One Object

Do we have only one `lm` object, the function `anova` returns an ANOVA table.

This is however then a sequential analysis of variance table for that fit. This means, that the function returns a table where we can see in each row how much the residual sum of squares would be changed by taking that variable out of the model. The significance of this change is evaluated with an F-test. We start reading this table at the bottom. This means, that tables says nothing about if a variable belongs to the model, it makes only a statement if the variable improved the fit when you added it to the model. Therefore the order how you specified the model matters here. For instance if you would have the model formula `y ~ x + z + w` your ANOVA table would look different than when you would have used `y ~ w + z +x`. For the first model, the last row of the ANOVA table would evaluate if a model with x, z and w is equal to a model with only x and z. The next row compares then the models x and z against only z.

anova for Several Objects

We call models nested when there is a "largest" model and all other models could be seen as subsets of this "largest" model.

Do we submit now several `lm.objects` which are nested to the `anova` function the ANOVA table then compares the different models.

R however cannot make sure, that the models are nested. Therefore it just makes the assumption. It is a kind of convention to start the list with the largest model and arrange them then in descending order.
Then again we can start our comparison in the last row and compare the results sequentially.

na.action

Model comparisons based on likelihood tests make however the assumption that the design matrix is always the "same"'. This must be taken into account when the data has missing values. Normally, when there are missing values, we delete observations which have missing values in the independent variables which are used in the current model. Therefore often smaller models have more observations than larger models.

In R we can choose in `lm` therefore for at least two different `na.actions`:

- `na.omit` uses all observation that are possible (no missing values for residuals and fitted values and so on)
- `na.exclude` also makes residuals and fitted values comparable when missing values are at hand

plot

As mentioned earlier, most of the model assumptions of regressions can be evaluated using plots.

R provides by default 4 plots for diagnostics when an `lm.object` is submitted to the `plot` function. Those plots are:

- residuals vs. fitted
- Q-Qplot for the standardized residuals
- root of standardized residuals vs. fitted values
- a plot of residuals against leverages

It is often easier to evaluate the fit when plotting all four plots into one window using the `par()` function.

Other plots can be obtained using the `which` argument. For details see `?plot.lm`.

model.matrix

If one is interested how the design matrix looks one can use the function `model.matrix`.

This function returns for an `lm` object the design matrix where one for example can see which contrast was used for a factor and so on. Especially when there are factors in your model it might be a good idea to check this matrix so that you know how to interpret the result.

Contrasts in R I

As mentioned earlier, factors need dummy variables when they enter a regression model. Depending on that coding, the interpretation of the parameter estimates changes. Which parameters R uses by default can be found out using the command:

```
> getOption("contrasts")
       unordered           ordered
"contr.treatment"      "contr.poly"
```

There one can see what R uses as default contrasts for unordered factors and ordered factors.

Contrasts in R II

The contrasts discussed earlier have in R the following names:

- treatment contrast: `contr.treatment`
- helmert contrast: `contr.helmert`
- sum contrast: `contr.sum`
- polynomial contrast: `contr.poly`

To specify the characteristics of each contrast like which is the default comparison level in the treatment contrast see the help for contrast of interest. And also the function `relevel`.

Contrasts in R III

If one wants different contrasts than the default ones, there are two ways to change it. First we can change it globally, so that it effects all applications where we need contrasts. Then we use the option command and specify there the default contrast for unordered and ordered contrasts.

```
options(contrasts=c('default contrast for unordered factors',
 'default contrast for ordered factors'))
```

Or we change it only in our regression function call. Here can even use several different contrasts. If we call for example the regression function `lm` and we have two factors, named factor1 and factor2. For one we would like to have the treatment contrast and for the other the helmert contrast then we could use:

```
lm.out <- lm(..., contrasts=c(factor1=contr.treatment,
 factor2=contr.helmert))
```

Fitted Values

There is a generic function to extract fitted values from a regression object. That function is called `fitted`. However especially for `lm` objects there are also to other ways to extract fitted values.

Let us call our `lm` object again `lm.out`. Then we can get the fitted values using:

- `fitted(lm.out)`
- `fitted.values(lm.out)`
- `lm.out$fitted`

Residuals in R

As mentioned above, residuals are important features for model diagnostics.

R offers a lot of residual types which can be extracted from an regression object. The basic function for this purpose is `residuals`. Which types of residuals can be obtained from `lm` objects can be found out using the help for `residuals.lm`.

For the studentized residuals there is also a special function, `rstundent`, available as well a function `rstandard` for standardized residuals.

Furthermore can the basic residuals also just be extracted using:

```
lm.out$res
```

Predictions in R

The motivation to fit a regression model can have several reasons. One reason is to predict the dependent variable given new subjects or to predict the development in the future. It is quite easy to get predictions in R. One needs mainly two steps to get them.
First one has to create a dataframe (`data.new`) that contains the settings of the independent variables for which a prediction is wanted. Then one uses the function `predict` to obtain the predictions.
Assume one wants to predict for the `lm.out` object and one has a dataframe `data.new` for which one wants to predict. Then use:

```
predict(lm.out, data.new)
```

When we are also interested in confidence intervals we can add the interval argument and decide whether we want the real prediction interval or the main response interval.

Infuence Diagnostics in R

For a regression object, the function `influence.measures` will, applied on such an object return a dataframe containing all important influence measures as all DFBETAS, DFFITS, Cook's distance, covariance ratios as well as the leverage values for each observation (they are called here in the output "hat").

Observations assumed to be influential concerning any of the diagnostics are marked with an asterisk.

Model Selection in R

Automatic model selection is also possible in R. However not based on p-values but on AIC or BIC. The function for this is the function `step`.

It can perform all three different types of selections. Backward, forward and stepwise.

One can even specify minimal and maximal model between which we want to choose. In general one can punish here the number of parameters with any weight k. But only the settings k=2 (AIC) or k=log(n) (BIC) have then a theoretical criterion on which the selection would be based upon.

Cherry Tree Example I

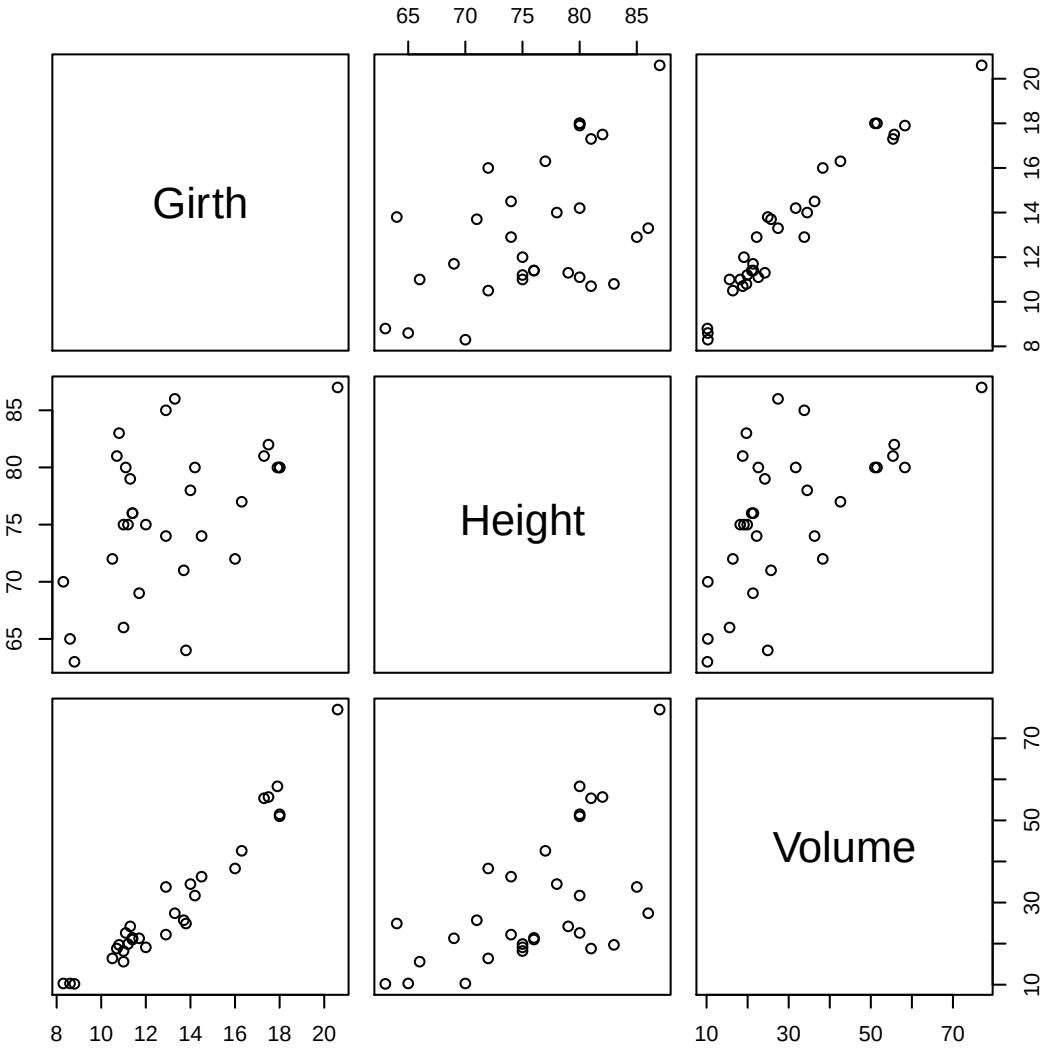
As a first example consider the `trees` data set from the `MASS` package. The data set contains the girth, height and volume of 31 felled black cherry trees. The aim is to obtain a model which can be used to predict the volume of a tree based on its height and girth.

```
> library(MASS)
> data(trees)
> head(trees)
  Girth Height Volume
1   8.3     70   10.3
2   8.6     65   10.3
3   8.8     63   10.2
4  10.5     72   16.4
5  10.7     81   18.8
6  10.8     83   19.7
```

Cherry Tree Example II

```
> str(trees)
'data.frame':   31 obs. of  3 variables:
 $ Girth : num  8.3 8.6 8.8 10.5 10.7 10.8 11 11 11.1 11.2 ...
 $ Height: num  70 65 63 72 81 83 66 75 80 75 ...
 $ Volume: num  10.3 10.3 10.2 16.4 18.8 19.7 15.6 18.2 22.6 1
> summary(trees)
     Girth           Height       Volume
 Min.   : 8.30   Min.   :63   Min.   :10.20
 1st Qu.:11.05   1st Qu.:72   1st Qu.:19.40
 Median :12.90   Median :76   Median :24.20
 Mean   :13.25   Mean   :76   Mean   :30.17
 3rd Qu.:15.25   3rd Qu.:80   3rd Qu.:37.30
 Max.   :20.60   Max.   :87   Max.   :77.00
> plot(trees)
```

Cherry Tree Example III



Cherry Tree Example IV

Let us first fit a marginal model for the two explaining variables.

```
> options(show.signif.stars=FALSE)
> fit.girth <- lm(Volume ~ Girth, data = trees)
> summary(fit.girth)

Call:
lm(formula = Volume ~ Girth, data = trees)

Residuals:
   Min     1Q Median     3Q    Max
-8.065 -3.107  0.152  3.495  9.587

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -36.9435     3.3651  -10.98 7.62e-12
Girth         5.0659     0.2474   20.48  < 2e-16

Residual standard error: 4.252 on 29 degrees of freedom
Multiple R-squared: 0.9353,     Adjusted R-squared: 0.9331
F-statistic: 419.4 on 1 and 29 DF,  p-value: < 2.2e-16
```

Cherry Tree Example V

```
> fit.height <- lm(Volume ~ Height, data = trees)
> summary(fit.height)

Call:
lm(formula = Volume ~ Height, data = trees)

Residuals:
    Min      1Q  Median      3Q     Max
-21.274  -9.894  -2.894  12.068  29.852

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -87.1236    29.2731  -2.976 0.005835
Height        1.5433     0.3839   4.021 0.000378

Residual standard error: 13.4 on 29 degrees of freedom
Multiple R-squared: 0.3579,     Adjusted R-squared: 0.3358
F-statistic: 16.16 on 1 and 29 DF,  p-value: 0.0003784
```

Cherry Tree Example VI

Model containing both explaining variables.

```
> fit.both <- lm(Volume ~ Girth + Height, data = trees)
> summary(fit.both)

Call:
lm(formula = Volume ~ Girth + Height, data = trees)

Residuals:
    Min      1Q  Median      3Q     Max
-6.4065 -2.6493 -0.2876  2.2003  8.4847

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -57.9877     8.6382  -6.713 2.75e-07
Girth         4.7082     0.2643  17.816  < 2e-16
Height        0.3393     0.1302   2.607   0.0145

Residual standard error: 3.882 on 28 degrees of freedom
Multiple R-squared: 0.948,      Adjusted R-squared: 0.9442
F-statistic:   255 on 2 and 28 DF,  p-value: < 2.2e-16
```

Cherry Tree Example VII

```
> coef(fit.both)
(Intercept)       Girth      Height
-57.9876589   4.7081605   0.3392512
> confint(fit.both)
                   2.5 %      97.5 %
(Intercept) -75.68226247 -40.2930554
Girth         4.16683899   5.2494820
Height        0.07264863   0.6058538
> anova(fit.both)
Analysis of Variance Table

Response: Volume
          Df Sum Sq Mean Sq  F value  Pr(>F)
Girth      1 7581.8  7581.8 503.1503 < 2e-16
Height     1  102.4   102.4   6.7943 0.01449
Residuals 28  421.9    15.1
```

Cherry Tree Example VIII

A full model might here rather a model with a second degree polynomial for both variables.

```
> fit.full <- lm(Volume ~ Girth + I(Girth^2) + Height + I(Height^2), data = trees)
> summary(fit.full)

Call:
lm(formula = Volume ~ Girth + I(Girth^2) + Height + I(Height^2),
    data = trees)

Residuals:
   Min     1Q Median     3Q    Max
-4.368 -1.670 -0.158  1.792  4.358

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.955101  63.013630  -0.015    0.988
Girth       -2.796569   1.468677  -1.904    0.068
I(Girth^2)   0.265446   0.051689   5.135 2.35e-05
Height       0.119372   1.784588   0.067    0.947
I(Height^2)  0.001717   0.011905   0.144    0.886

Residual standard error: 2.674 on 26 degrees of freedom
Multiple R-squared: 0.9771,     Adjusted R-squared: 0.9735
F-statistic:   277 on 4 and 26 DF,  p-value: < 2.2e-16
```

Cherry Tree Example IX

A full model might here rather a model with a second degree polynomial for both variables.

```
> coef(fit.full)
 (Intercept)        Girth   I(Girth^2)       Height  I(Height^2)
-0.955101497 -2.796568781  0.265446012  0.119371566  0.001716694
> confint(fit.full)
                    2.5 %       97.5 %
(Intercept) -130.48147401 128.57127102
Girth         -5.81547684   0.22233928
I(Girth^2)     0.15919771   0.37169431
Height        -3.54890263   3.78764576
I(Height^2)   -0.02275384   0.02618722
> anova(fit.full)
Analysis of Variance Table

Response: Volume
            Df Sum Sq Mean Sq   F value    Pr(>F)
Girth        1 7581.8  7581.8 1060.5991 < 2.2e-16
I(Girth^2)   1  212.9   212.9   29.7850 1.009e-05
Height       1  125.4   125.4   17.5378  0.000286
I(Height^2)  1    0.1     0.1    0.0208  0.886452
Residuals   26  185.9     7.1
```

Cherry Tree Example X

The polynomials make the parameters difficult to interpret.

```
> with(trees, cor(Girth, Girth^2))
[1] 0.9930404
> with(trees, cor(Height, Height^2))
[1] 0.99887
> m.Girth <- with(trees, mean(Girth))
> m.Height <- with(trees, mean(Height))
> with(trees, cor(Girth-m.Girth, (Girth-m.Girth)^2))
[1] 0.4379578
> with(trees, cor(Height-m.Height, (Height-m.Height)^2
[1] -0.3133785
```

Cherry Tree Example XI

So lets us the centered variables.

```
> fit.full.c <- lm(Volume ~ I(Girth-m.Girth)+ I((Girth-m.Girth)^2)+ I(Height-m.Height) + I((Height-m.Height)^2)
> summary(fit.full.c)

Call:
lm(formula = Volume ~ I(Girth - m.Girth) + I((Girth - m.Girth)^2) +
    I(Height - m.Height) + I((Height - m.Height)^2), data = trees)

Residuals:
   Min     1Q Median     3Q    Max
-4.368 -1.670 -0.158  1.792  4.358

Coefficients:
                          Estimate Std. Error t value Pr(>|t|)
(Intercept)              27.573754   0.704033  39.165  < 2e-16
I(Girth - m.Girth)        4.236894   0.202220  20.952  < 2e-16
I((Girth - m.Girth)^2)    0.265446   0.051689   5.135 2.35e-05
I(Height - m.Height)      0.380309   0.093901   4.050  0.00041
I((Height - m.Height)^2)  0.001717   0.011905   0.144  0.88645

Residual standard error: 2.674 on 26 degrees of freedom
Multiple R-squared: 0.9771,     Adjusted R-squared: 0.9735
F-statistic:   277 on 4 and 26 DF,  p-value: < 2.2e-16
```

Cherry Tree Example XII

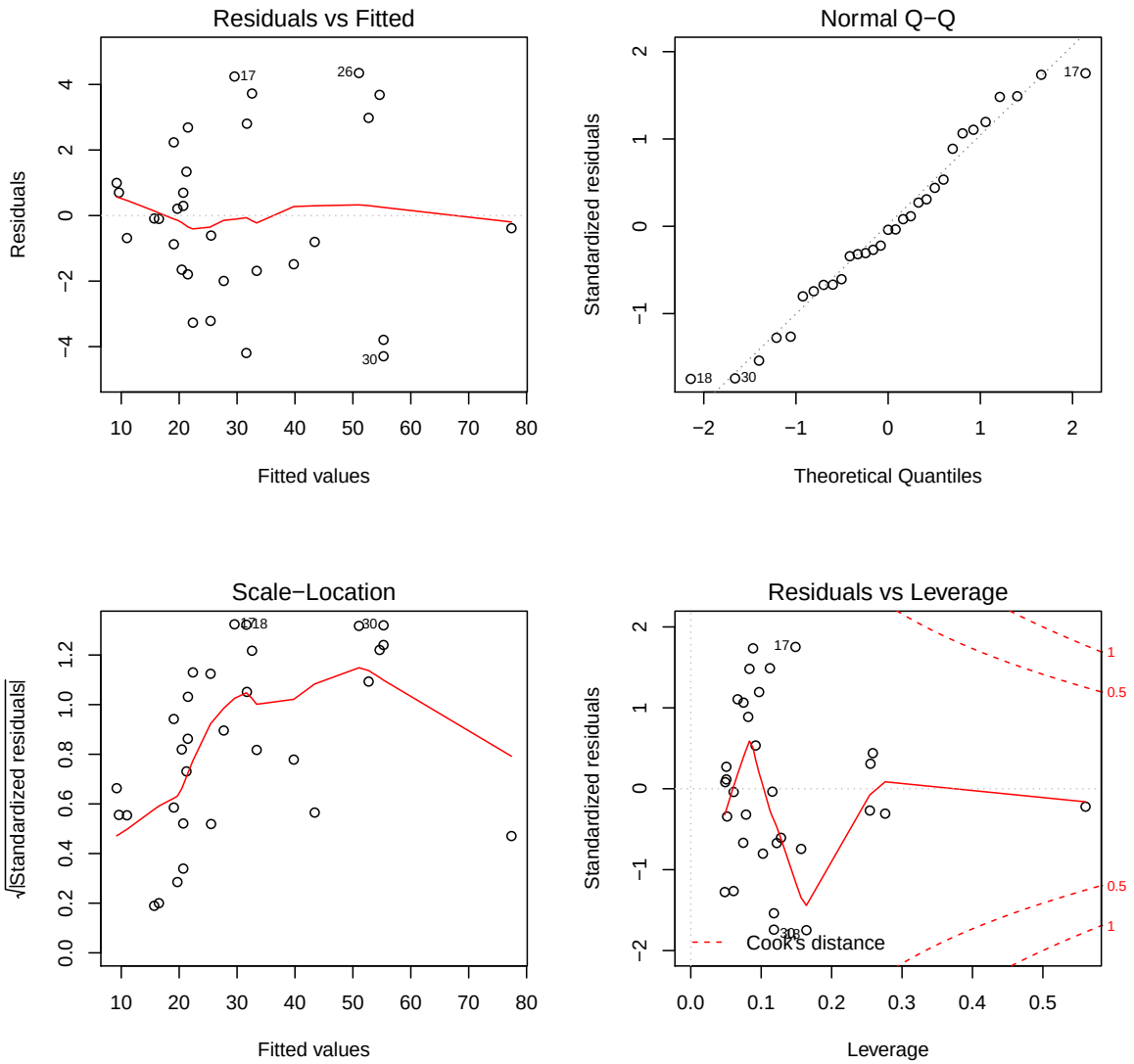
```
> fit.2 <- lm(Volume ~ I(Girth-m.Girth)+ I((Girth-m.Girth)^2) + I(Height-m.Height), data = trees)
> summary(fit.2)

Call:
lm(formula = Volume ~ I(Girth - m.Girth) + I((Girth - m.Girth)^2) +
    I(Height - m.Height), data = trees)

Residuals:
    Min      1Q  Median      3Q     Max
-4.2928 -1.6693 -0.1018  1.7851  4.3489

Coefficients:
                       Estimate Std. Error t value Pr(>|t|)
(Intercept)            27.61093    0.64314  42.931  < 2e-16
I(Girth - m.Girth)      4.23255    0.19630  21.561  < 2e-16
I((Girth - m.Girth)^2)  0.26862    0.04590   5.852 3.13e-06
I(Height - m.Height)    0.37639    0.08823   4.266 0.000218

Residual standard error: 2.625 on 27 degrees of freedom
Multiple R-squared: 0.9771,     Adjusted R-squared: 0.9745
F-statistic: 383.2 on 3 and 27 DF,  p-value: < 2.2e-16
```

Cherry Tree Example XIII

Some model comparison if we would need the dropped term

```
> anova(fit.full.c, fit.2)
Analysis of Variance Table

Model 1: Volume ~ I(Girth - m.Girth) + I((Girth - m.Girth)^2) + I(Height -
    m.Height) + I((Height - m.Height)^2)
Model 2: Volume ~ I(Girth - m.Girth) + I((Girth - m.Girth)^2) + I(Height -
    m.Height)
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     26 185.86
2     27 186.01 -1  -0.14865 0.0208 0.8865
> AIC(fit.full.c)
[1] 155.4959
> AIC(fit.2)
[1] 153.5207

> par(mfrow=c(2,2))
> plot(fit.2)
```

Cherry Tree Example XIV



Cherry Tree Example XV

```
> influence.measures(fit.2)
Influence measures of
         lm(formula = Volume ~ I(Girth - m.Girth) + I((Girth - m.Girth)^2) +
         I(Height - m.Height), data = trees) :

     dfb.1_ dfb.I.G.m dfb.I...m dfb.I.H.m   dffit cov.r   cook.d    hat inf
1   0.04991   0.13389  -0.14205  -0.00093 -0.1865 1.584 8.99e-03 0.2761   *
2  -0.03231  -0.08786   0.11558  -0.06343  0.1777 1.539 8.17e-03 0.2552   *
3  -0.03156  -0.09458   0.14379  -0.12951  0.2559 1.524 1.69e-02 0.2585   *
4  -0.00421   0.00502  -0.00166   0.00137 -0.0100 1.238 2.59e-05 0.0609
5  -0.07507   0.17931  -0.02688  -0.18423 -0.2478 1.238 1.57e-02 0.1221
6  -0.08885   0.22124  -0.02482  -0.25914 -0.3180 1.269 2.57e-02 0.1567
7  -0.00577  -0.00123   0.00123   0.00965 -0.0128 1.315 4.22e-05 0.1159
8  -0.04817   0.04115   0.00383  -0.01559 -0.0786 1.205 1.60e-03 0.0516
9   0.07653  -0.10653  -0.00540   0.11557  0.1680 1.228 7.25e-03 0.0921
10  0.01258  -0.00800  -0.00264   0.00299  0.0181 1.222 8.53e-05 0.0490
11  0.17072  -0.17166  -0.03627   0.18490  0.3036 1.059 2.29e-02 0.0749
12  0.01920  -0.01091  -0.00577   0.00715  0.0261 1.222 1.77e-04 0.0506
13  0.04532  -0.02574  -0.01361   0.01687  0.0616 1.212 9.82e-04 0.0506
14  0.18030   0.04963  -0.08648  -0.17184  0.2635 1.125 1.75e-02 0.0815
15 -0.26742   0.02490   0.13671  -0.00418 -0.2912 0.953 2.07e-02 0.0483
16 -0.31618  -0.10002   0.20935   0.08713 -0.3255 0.970 2.59e-02 0.0608
17  0.43839  -0.21691  -0.26133   0.60135  0.7635 0.843 1.34e-01 0.1487
```

Cherry Tree Example XVI

18	-0.45747	0.19053	0.28697	-0.64193	-0.8073	0.861	1.50e-01	0.1642	
19	-0.21900	-0.15609	0.15914	0.16550	-0.2692	1.176	1.84e-02	0.1024	
20	-0.08363	-0.09513	0.06361	0.13358	-0.1546	1.543	6.19e-03	0.2543	*
21	0.28824	0.10047	-0.20135	0.03192	0.2963	1.035	2.18e-02	0.0665	
22	-0.17029	-0.04552	0.11730	-0.06045	-0.1877	1.176	8.99e-03	0.0745	
23	0.40571	0.29310	-0.28977	-0.19675	0.4576	0.904	4.99e-02	0.0834	
24	-0.14078	-0.18431	0.08282	0.15398	-0.2296	1.263	1.35e-02	0.1281	
25	-0.06442	-0.06968	0.03137	0.02652	-0.0917	1.243	2.17e-03	0.0785	
26	0.24006	0.31964	0.01334	0.04704	0.5629	0.795	7.31e-02	0.0885	
27	0.13904	0.19651	0.04097	0.06727	0.3950	1.036	3.84e-02	0.0971	
28	0.12795	0.31223	0.12530	-0.03997	0.5429	0.930	7.02e-02	0.1124	
29	-0.11772	-0.32571	-0.15272	0.04587	-0.5784	0.913	7.92e-02	0.1181	
30	-0.13508	-0.37373	-0.17524	0.05263	-0.6637	0.819	1.01e-01	0.1181	
31	0.08569	-0.01382	-0.18947	-0.04927	-0.2456	2.627	1.56e-02	0.5606	*

Cherry Tree Example XVII

The model in this form might not be fully satisfactory.

However to make a reasonability check, assume the tree would be a perfect cylinder with height h and radius r, then

- the girth of the circle is $g = 2\pi r$
- the volume of the cylinder is $v = \pi r^2 h$

Anorexia Example I

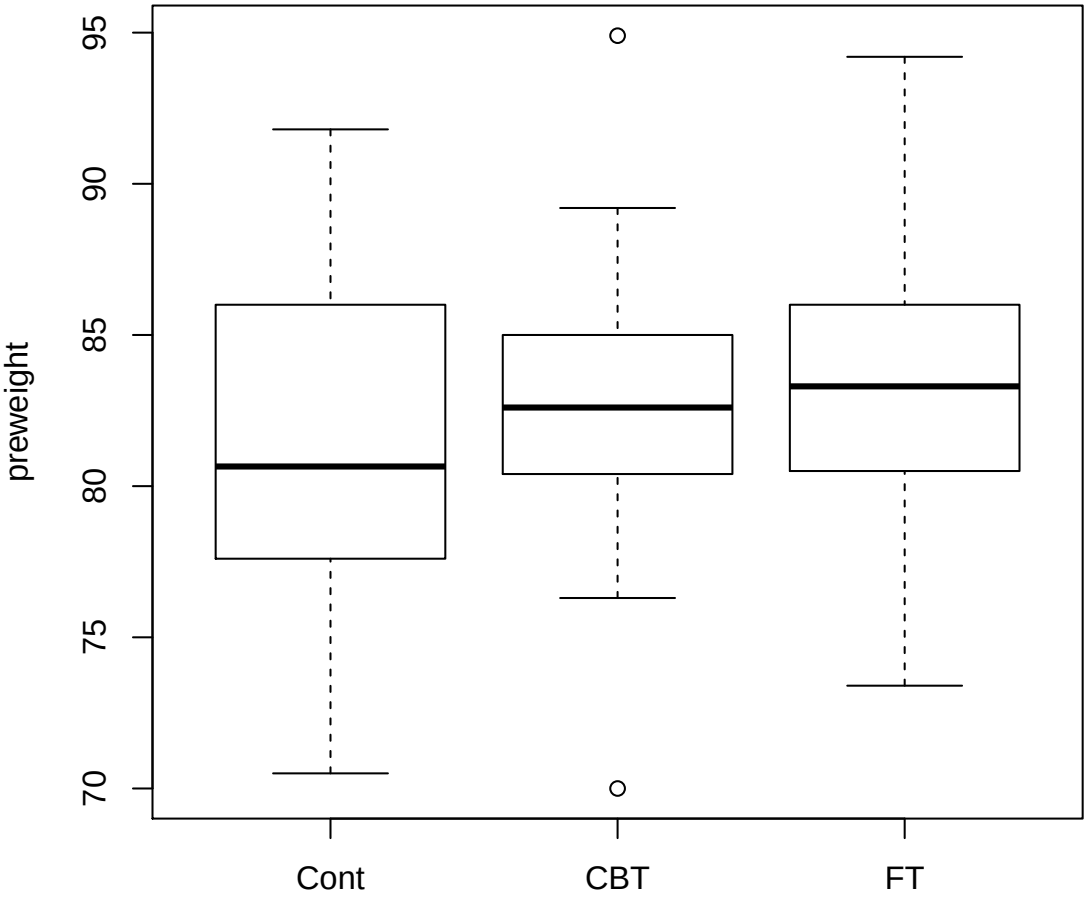
This data set contains the effect of different forms of therapy on the body weight of girls suffering from anorexia.

```
> library(MASS)
> options(show.signif.stars=FALSE)
> data(anorexia)
> str(anorexia)
'data.frame':   72 obs. of  3 variables:
 $ Treat : Factor w/ 3 levels "CBT","Cont","FT": 2 2 2 2 2 2 2 2 2 2 ...
 $ Prewt : num  80.7 89.4 91.8 74 78.1 88.3 87.3 75.1 80.6 78.4 ...
 $ Postwt: num  80.2 80.1 86.4 86.3 76.1 78.1 75.1 86.7 73.5 84.6 ...
> sixth <- seq(6,nrow(anorexia),by=6)
> anorexia[sixth,]
   Treat Prewt Postwt
6   Cont  88.3   78.1
12  Cont  88.7   79.5
18  Cont  86.0   75.4
24  Cont  77.5   81.2
30   CBT  82.6   81.9
36   CBT  80.5   82.1
42   CBT  80.4   71.3
48   CBT  76.5   75.7
54   CBT  80.8   96.2
60    FT  86.7  100.3
66    FT  81.6   77.8
72    FT  87.3   98.0
```

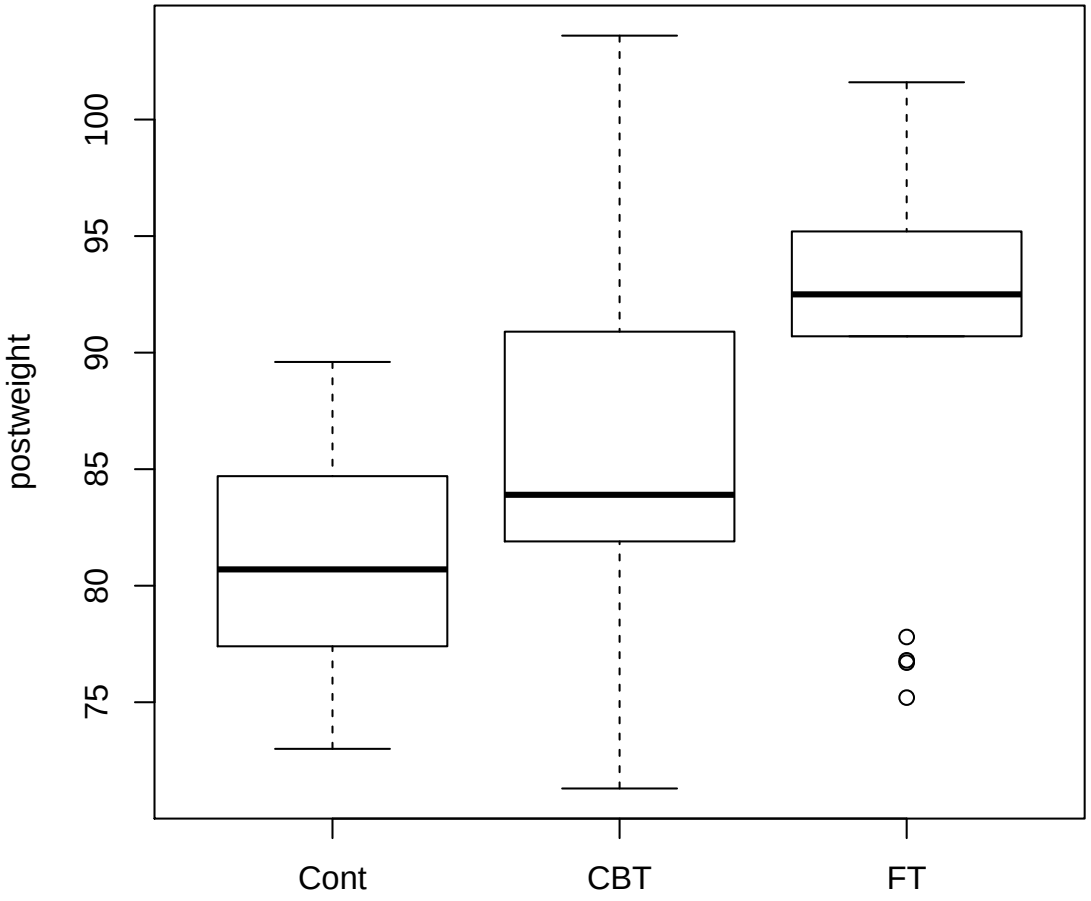
Anorexia Example II

```
> summary(anorexia)
  Treat        Prewt           Postwt
 CBT :29   Min.   :70.00   Min.   : 71.30
 Cont:26   1st Qu.:79.60   1st Qu.: 79.33
 FT  :17   Median :82.30   Median : 84.05
           Mean   :82.41   Mean   : 85.17
           3rd Qu.:86.00   3rd Qu.: 91.55
           Max.   :94.90   Max.   :103.60
> anorexia$TREAT <- relevel(anorexia$Treat, ref="Cont")
> boxplot(Prewt~TREAT, data=anorexia, ylab="preweight")
> boxplot(Postwt~TREAT, data=anorexia, ylab="postweight")
```

Anorexia Example III



Anorexia Example IV



Anorexia Example V

```
> Prewt.mean <- with(anorexia, tapply(Prewt, TREAT, mean))
> Prewt.sd <- with(anorexia, tapply(Prewt, TREAT, sd))
> Postwt.mean <- with(anorexia, tapply(Postwt, TREAT, mean))
> Postwt.sd <- with(anorexia, tapply(Postwt, TREAT, sd))
> n.group <- with(anorexia, tapply(Prewt, TREAT, length))
>
> ANOREX.SUMMARY <- cbind(n.group, Prewt.mean, Prewt.sd, Postwt.mean, Postwt.sd)
> ANOREX.SUMMARY
     n.group Prewt.mean Prewt.sd Postwt.mean Postwt.sd
Cont      26   81.55769 5.707060    81.10769  4.744253
CBT       29   82.68966 4.845495    85.69655  8.351924
FT        17   83.22941 5.016693    90.49412  8.475072
```

Anorexia Example VI

```
> anfit1 <- lm(Postwt~TREAT, data=anorexia)
> model.matrix(anfit1)[sixth,]
   (Intercept) TREATCBT TREATFT
6            1        0       0
12           1        0       0
18           1        0       0
24           1        0       0
30           1        1       0
36           1        1       0
42           1        1       0
48           1        1       0
54           1        1       0
60           1        0       1
66           1        0       1
72           1        0       1
> summary(anfit1)
Call:
lm(formula = Postwt ~ TREAT, data = anorexia)
Residuals:
     Min       1Q   Median       3Q      Max
-15.2941  -3.7299  -0.0021   4.7809  17.9034
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   81.108      1.429  56.746   <2e-16
TREATCBT       4.589      1.968   2.331   0.0227
TREATFT        9.386      2.273   4.129   0.0001

Residual standard error: 7.288 on 69 degrees of freedom
Multiple R-squared: 0.2005,     Adjusted R-squared: 0.1773
F-statistic: 8.651 on 2 and 69 DF,  p-value: 0.0004443
```

Anorexia Example VII

```
> anfit1b <- lm(Postwt~TREAT-1, data=anorexia)
> model.matrix(anfit1b)[sixth,]
   TREATCont TREATCBT TREATFT
6          1        0       0
12         1        0       0
18         1        0       0
24         1        0       0
30         0        1       0
36         0        1       0
42         0        1       0
48         0        1       0
54         0        1       0
60         0        0       1
66         0        0       1
72         0        0       1
> summary(anfit1b)
Call:
lm(formula = Postwt ~ TREAT - 1, data = anorexia)
Residuals:
     Min       1Q   Median       3Q      Max
-15.2941  -3.7299  -0.0021   4.7809  17.9034
Coefficients:
          Estimate Std. Error t value Pr(>|t|)
TREATCont   81.108      1.429   56.75   <2e-16
TREATCBT    85.697      1.353   63.32   <2e-16
TREATFT     90.494      1.768   51.20   <2e-16

Residual standard error: 7.288 on 69 degrees of freedom
Multiple R-squared: 0.993,      Adjusted R-squared: 0.9927
F-statistic:  3284 on 3 and 69 DF,  p-value: < 2.2e-16
```

Anorexia Example VIII

```
> anfit1c <- lm(Postwt~TREAT, data=anorexia, contrast=list(TREAT="contr.sum"))
> model.matrix(anfit1c)[sixth,]
   (Intercept) TREAT1 TREAT2
6            1      1      0
12           1      1      0
18           1      1      0
24           1      1      0
30           1      0      1
36           1      0      1
42           1      0      1
48           1      0      1
54           1      0      1
60           1     -1     -1
66           1     -1     -1
72           1     -1     -1
> summary(anfit1c)
Call:
lm(formula = Postwt ~ TREAT, data = anorexia, contrasts = list(TREAT = "contr.sum"))
Residuals:
     Min       1Q   Median       3Q      Max
-15.2941  -3.7299  -0.0021   4.7809  17.9034
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 85.76612    0.88186  97.256  < 2e-16
TREAT1      -4.65843    1.20775  -3.857 0.000255
TREAT2      -0.06957    1.17822  -0.059 0.953087

Residual standard error: 7.288 on 69 degrees of freedom
Multiple R-squared: 0.2005,     Adjusted R-squared: 0.1773
F-statistic: 8.651 on 2 and 69 DF,  p-value: 0.0004443
```

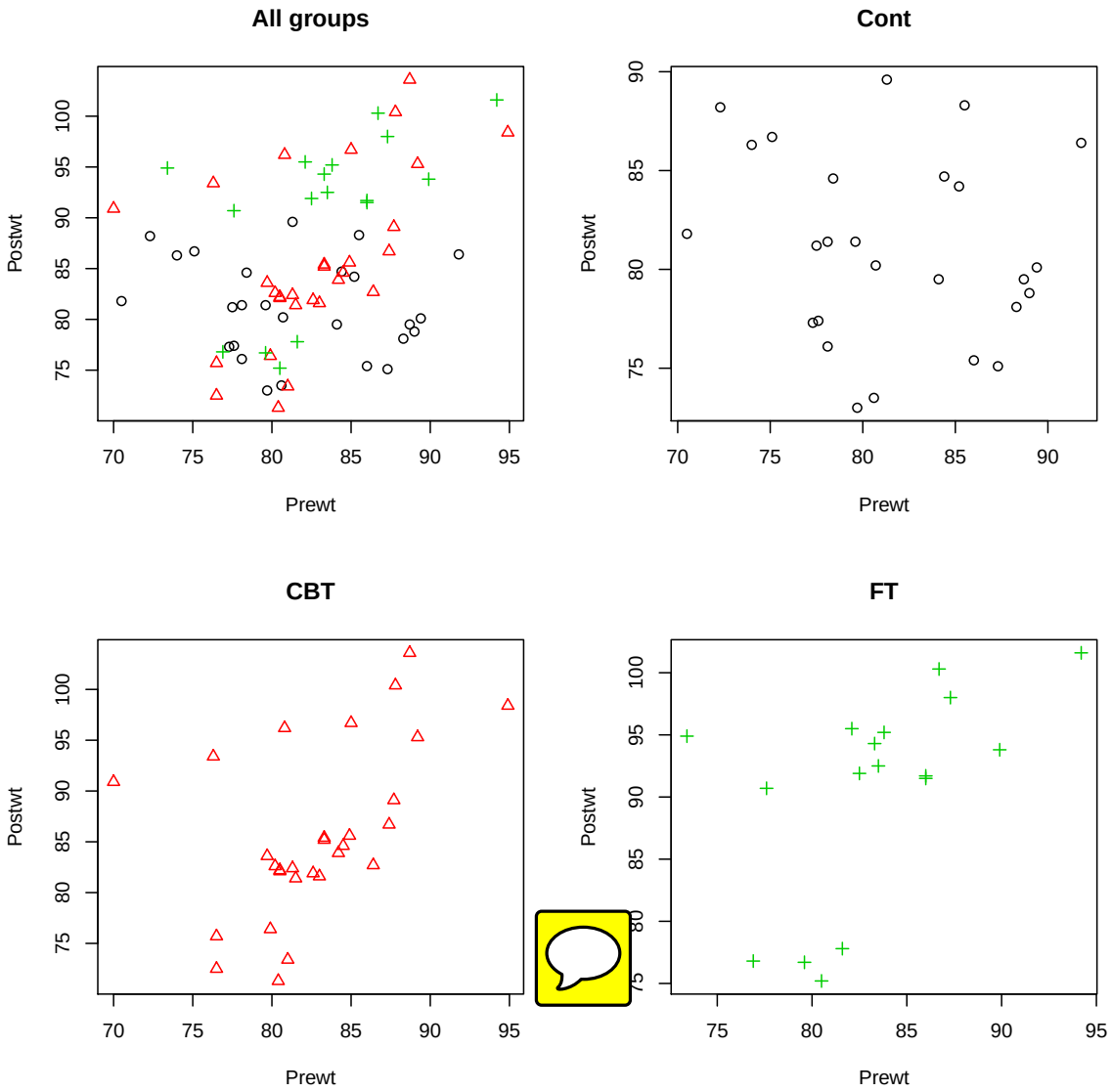
Anorexia Example IX

```
> anfit1d <- lm(Postwt~TREAT, data=anorexia, contrast=list(TREAT="contr.helmert"))
> model.matrix(anfit1d)[sixth,]
   (Intercept) TREAT1 TREAT2
6            1     -1     -1
12           1     -1     -1
18           1     -1     -1
24           1     -1     -1
30           1      1     -1
36           1      1     -1
42           1      1     -1
48           1      1     -1
54           1      1     -1
60           1      0      2
66           1      0      2
72           1      0      2
> summary(anfit1d)
Call:
lm(formula = Postwt ~ TREAT, data = anorexia, contrasts = list(TREAT = "contr.helmert"))
Residuals:
     Min       1Q   Median       3Q      Max
-15.2941  -3.7299  -0.0021   4.7809  17.9034
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  85.7661     0.8819  97.256  < 2e-16
TREAT1        2.2944     0.9842   2.331 0.022667
TREAT2        2.3640     0.6744   3.505 0.000806

Residual standard error: 7.288 on 69 degrees of freedom
Multiple R-squared: 0.2005,     Adjusted R-squared: 0.1773
F-statistic: 8.651 on 2 and 69 DF,  p-value: 0.0004443
```

Anorexia Example X

```
> par(mfrow=c(2,2))
> plot(Postwt~Prewt, data=anorexia, col=as.numeric(TREAT), pch=as.numeric(TREAT),
+ main="All groups")
> plot(Postwt~Prewt, data=anorexia, col=as.numeric(TREAT), pch=as.numeric(TREAT),
+ main="Cont", subset=TREAT=="Cont")
> plot(Postwt~Prewt, data=anorexia, col=as.numeric(TREAT), pch=as.numeric(TREAT),
+ main="CBT", subset=TREAT=="CBT")
> plot(Postwt~Prewt, data=anorexia, col=as.numeric(TREAT), pch=as.numeric(TREAT),
+ main="FT", subset=TREAT=="FT")
```

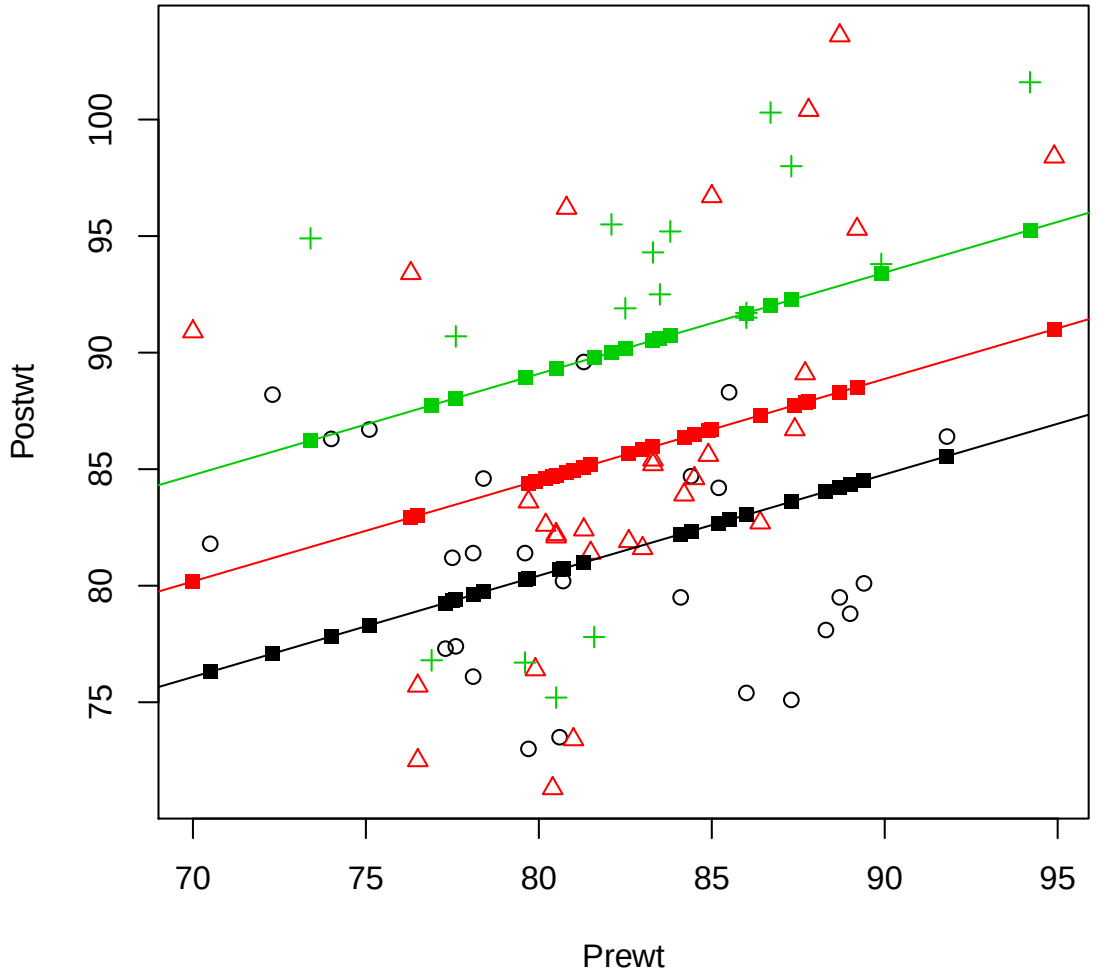
Anorexia Example XI



Anorexia Example XII

```
> coef(anfit2)
(Intercept)    TREATCBT     TREATFT       Prewt
 45.6740435   4.0970655   8.6601282   0.4344612
>
> plot(Postwt~Prewt, data=anorexia, col=as.numeric(TREAT), pch=as.numeric(TREAT))
> abline(coef(anfit2)[1],coef(anfit2)[4], col=1)
> abline(coef(anfit2)[1]+coef(anfit2)[2],coef(anfit2)[4], col=2)
> abline(coef(anfit2)[1]+coef(anfit2)[3],coef(anfit2)[4], col=3)
> with(anorexia,points(Prewt,fitted(anfit2),pch=15, col=as.numeric(TREAT)))
```

Anorexia Example XIII



Anorexia Example XIV

```
> anfit3 <- lm(Postwt~TREAT*Prewt, data=anorexia)
> summary(anfit3)

Call:
lm(formula = Postwt ~ TREAT * Prewt, data = anorexia)

Residuals:
     Min       1Q   Median       3Q      Max
-12.8125  -3.8501  -0.9153   4.0010  15.9640

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     92.0515    18.8085   4.894 6.67e-06
TREATCBT       -76.4742    28.3470  -2.698  0.00885
TREATFT        -77.2317    33.1328  -2.331  0.02282
Prewt           -0.1342     0.2301  -0.583  0.56173
TREATCBT:Prewt   0.9822     0.3442   2.853  0.00578
TREATFT:Prewt    1.0434     0.4000   2.609  0.01123

Residual standard error: 6.565 on 66 degrees of freedom
Multiple R-squared: 0.3794,     Adjusted R-squared: 0.3324
F-statistic:  8.07 on 5 and 66 DF,  p-value: 5.5e-06
```

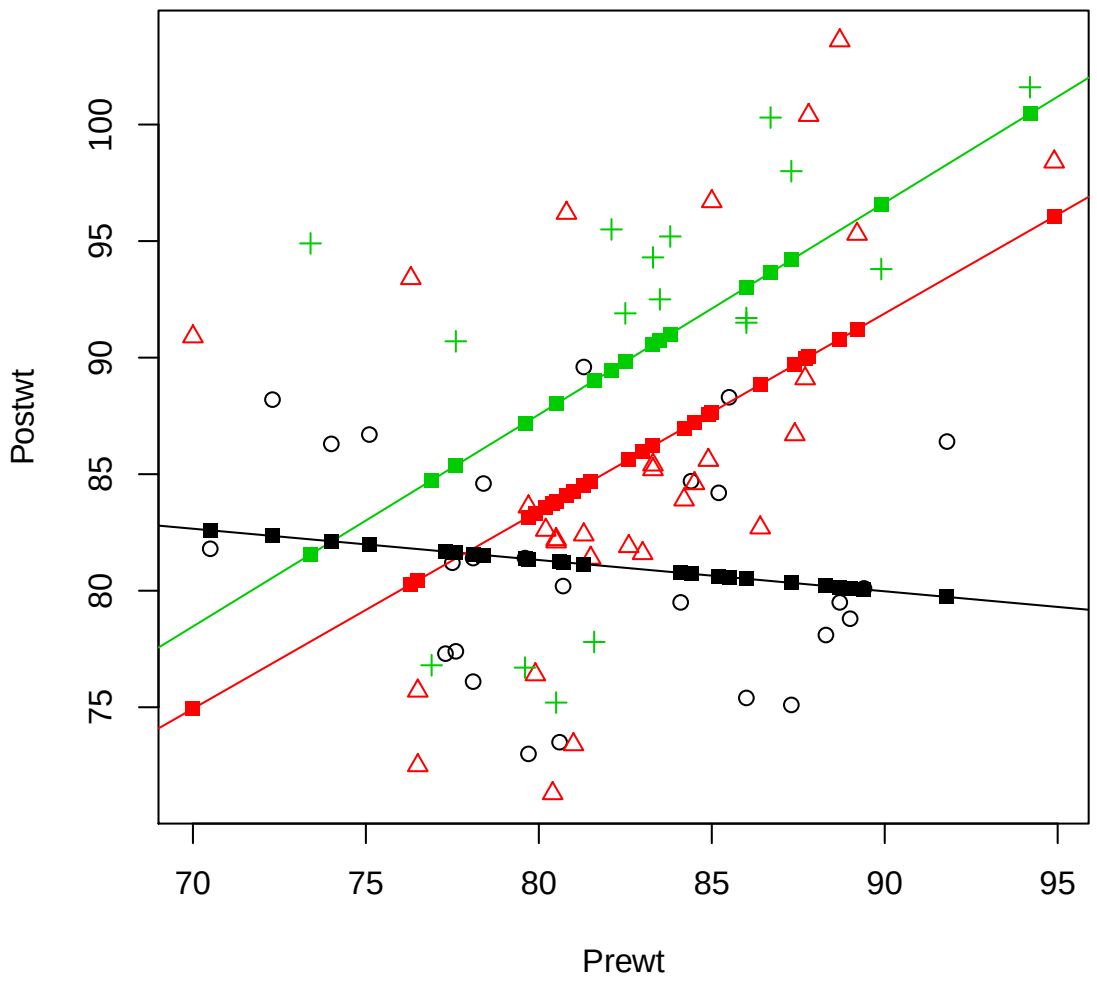
Anorexia Example XV

```
> model.matrix(anfit3)[sixth,]
   (Intercept) TREATCBT TREATFT Prewt TREATCBT:Prewt TREATFT:Prewt
6            1        0       0  88.3            0.0           0.0
12           1        0       0  88.7            0.0           0.0
18           1        0       0  86.0            0.0           0.0
24           1        0       0  77.5            0.0           0.0
30           1        1       0  82.6           82.6           0.0
36           1        1       0  80.5           80.5           0.0
42           1        1       0  80.4           80.4           0.0
48           1        1       0  76.5           76.5           0.0
54           1        1       0  80.8           80.8           0.0
60           1        0       1  86.7            0.0          86.7
66           1        0       1  81.6            0.0          81.6
72           1        0       1  87.3            0.0          87.3
```

Anorexia Example XVI

```
> coef(anfit3)
   (Intercept)       TREATCBT        TREATFT          Prewt TREATCBT:Prewt
    92.0514708    -76.4742268    -77.2317177     -0.1341845      0.9821661
 TREATFT:Prewt
     1.0434107
>
> plot(Postwt~Prewt, data=anorexia, col=as.numeric(TREAT), pch=as.numeric(TREAT))
> abline(coef(anfit3)[1],coef(anfit3)[4], col=1)
> abline(coef(anfit3)[1]+coef(anfit3)[2],coef(anfit3)[4]+coef(anfit3)[5], col=2)
> abline(coef(anfit3)[1]+coef(anfit3)[3],coef(anfit3)[4]+coef(anfit3)[6], col=3)
> with(anorexia,points(Prewt,fitted(anfit3),pch=15, col=as.numeric(TREAT)))
```

Anorexia Example XVII



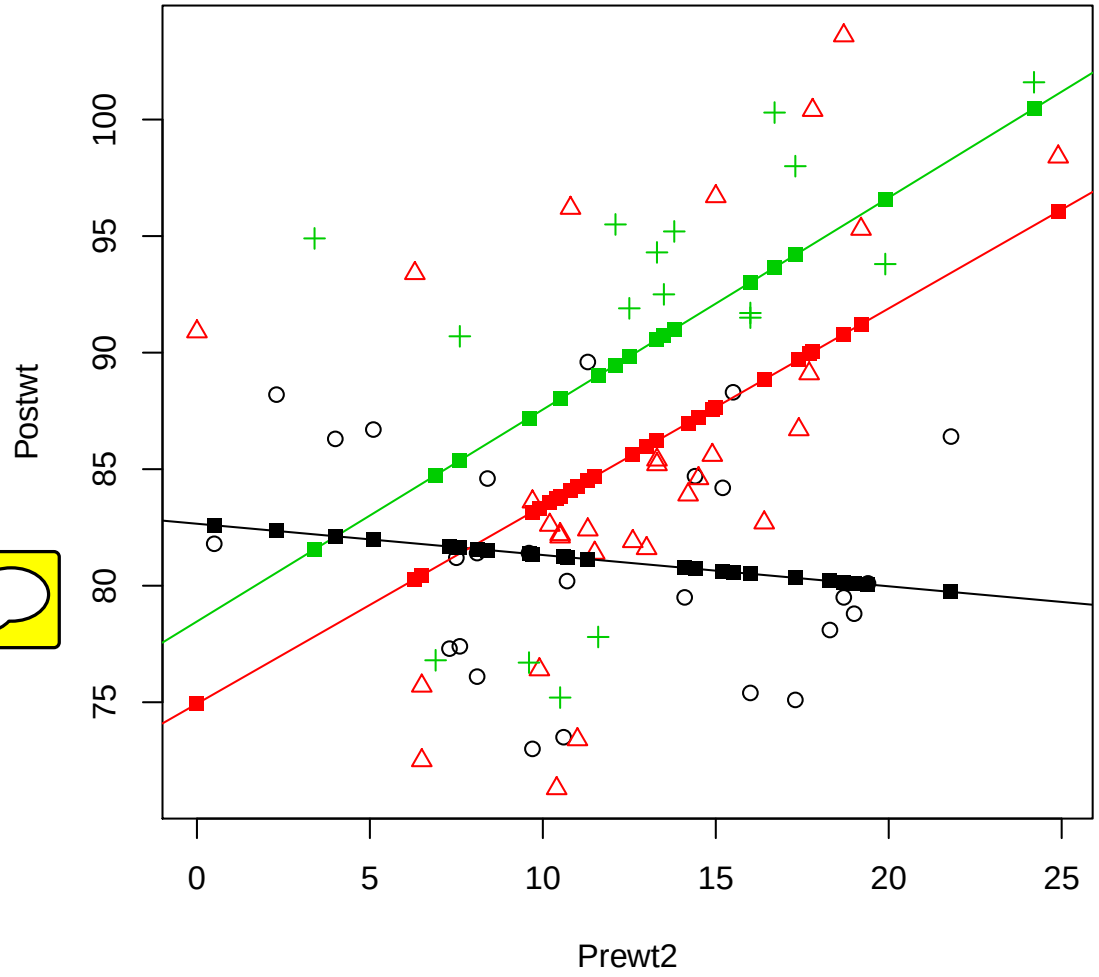
Anorexia Example XVIII

```
> min(anorexia$Prewt)
[1] 70
> anorexia$Prewt2 <- anorexia$Prewt - min(anorexia$Prewt)
> anfit4 <- lm(Postwt~TREAT*Prewt2, data=anorexia)
> summary(anfit4)

Call:
lm(formula = Postwt ~ TREAT * Prewt2, data = anorexia)

Residuals:
     Min       1Q   Median       3Q      Max
-12.8125  -3.8501  -0.9153   4.0010  15.9640

Coefficients:
                Estimate Std. Error t value Pr(>|t|)
(Intercept)      82.6586     2.9545  27.978  < 2e-16
TREATCBT         -7.7226     4.5577  -1.694  0.09490
TREATFT          -4.1930     5.4771  -0.766  0.44667
Prewt2           -0.1342     0.2301  -0.583  0.56173
TREATCBT:Prewt2   0.9822     0.3442   2.853  0.00578
TREATFT:Prewt2    1.0434     0.4000   2.609  0.01123

Residual standard error: 6.565 on 66 degrees of freedom
Multiple R-squared: 0.3794,     Adjusted R-squared: 0.3324
F-statistic:  8.07 on 5 and 66 DF,  p-value: 5.5e-06
```

Anorexia Example XIX

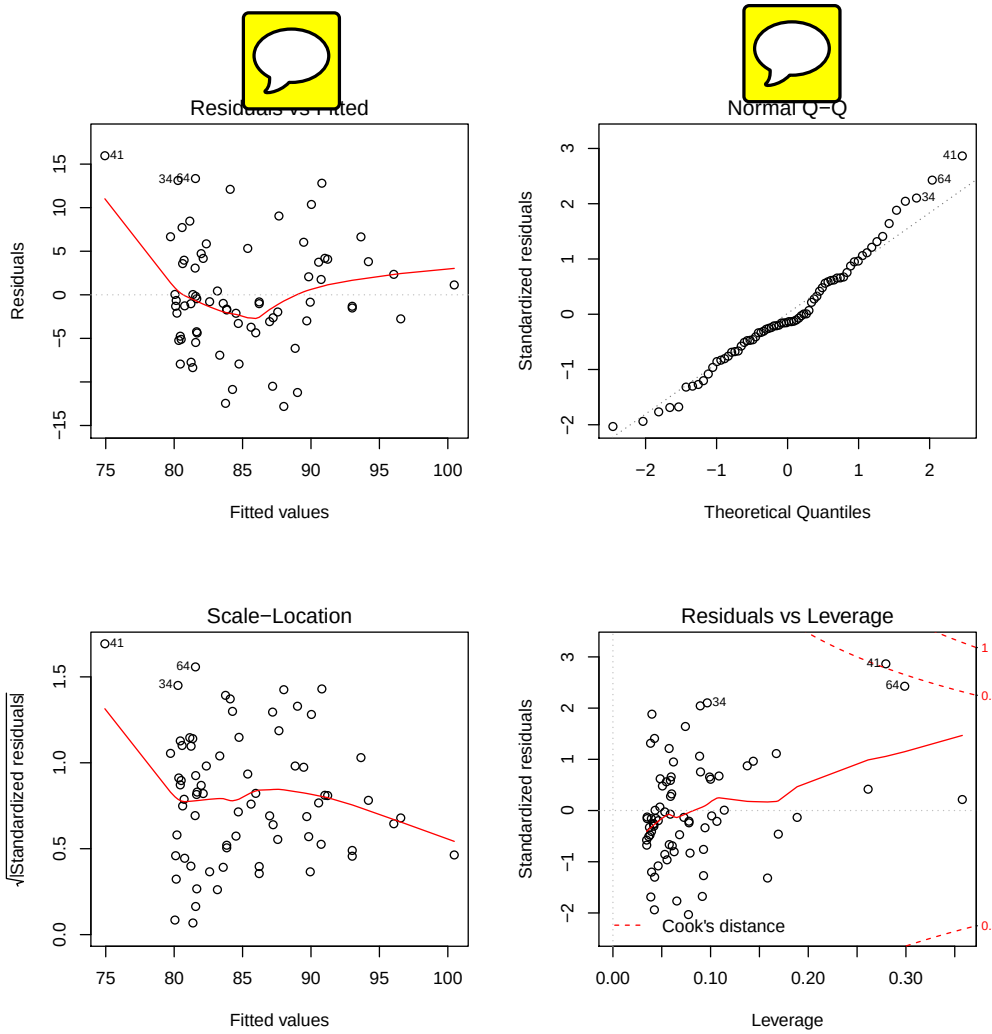
```
> coef(anfit4)
    (Intercept)        TREATCBT         TREATFT          Prewt2 TREATCBT:Prewt2
     82.6585555      -7.7225981      -4.1929661      -0.1341845       0.9821661
 TREATFT:Prewt2
      1.0434107
> plot(Postwt~Prewt2, data=anorexia, col=as.numeric(TREAT), pch=as.numeric(TREAT))
> abline(coef(anfit4)[1],coef(anfit4)[4], col=1)
> abline(coef(anfit4)[1]+coef(anfit4)[2],coef(anfit4)[4]+coef(anfit4)[5], col=2)
> abline(coef(anfit4)[1]+coef(anfit4)[3],coef(anfit4)[4]+coef(anfit4)[6], col=3)
> with(anorexia,points(Prewt2,fitted(anfit4),pch=15, col=as.numeric(TREAT)))
```

Anorexia Example XX



Anorexia Example XXI

```
> par(mfrow=c(2,2))
> plot(anfit4)
```



Anorexia Example XXII

```
> IM.fit4 <- influence.measures(anfit4)
> INFobs <- apply(IM.fit4$is.inf, 1, any)
> round(IM.fit4$infmat[INFobs,],3)
   dfb.1_ dfb.TREATCBT dfb.TREATFT dfb.Prw2 dfb.TREATCBT: dfb.TREATFT:  dffit cov.r cook.d   hat
15 -0.064        0.042       0.035    0.057        -0.038       -0.033 -0.064 1.349  0.001 0.189
33  0.000       -0.140       0.000    0.000         0.171        0.000  0.246 1.460  0.010 0.261
41  0.000        1.440       0.000    0.000        -1.317        0.000  1.892 0.686  0.530 0.279
63  0.000        0.000      -0.096    0.000         0.000        0.119  0.159 1.699  0.004 0.358
64  0.000        0.000       1.379    0.000         0.000       -1.208  1.647 0.892  0.418 0.299
70  0.000        0.000       0.097    0.000         0.000       -0.137 -0.207 1.294  0.007 0.169
```