```
/*  A SAS program to perform a regression
    analysis of the effects of the
    composition of Portland cement on the
    amount of heat given off as the cement
    hardens.  */

data set1;
  input run x1 x2 x3 x4 y;
/* label y = evolved heat (calories)
        x1 = tricalcium aluminate
        x2 = tricalcium silicate
        x3 = tetracalcium aluminate ferrate
        x4 = dicalcium silicate;  */
  cards;
  1  7 26  6 60  78.5
  2  1 29 15 52  74.3
  3 11 56  8 20 104.3
  4 11 31  8 47  87.6
  5  7 52  6 33  95.9
  6 11 55  9 22 109.2
  7  3 71 17  6 102.7
  8  1 31 22 44  72.5
  9  2 54 18 22  93.1
 10 21 47  4 26 115.9
 11  1 40 23 34  83.8
 12 11 66  9 12 113.2
 13 10 68  8 12 109.4
run;
```

```
proc print data=set1 uniform split='*';
  var y x1 x2 x3 x4;
  label y = 'Evolved*heat*(calories)'
        x1 = 'Percent*tricalcium*aluminate'
        x2 = 'Percent*tricalcium*silicate'
        x3 = 'Percent*tetracalcium*aluminate*ferrate'
        x4 = 'Percent*dicalcium*silicate';
  run;

/*  Regress y on all four explanatory
    variables and check residual plots
    and collinearity diagnostics */

proc reg data=set1 corr;
  model y = x1 x2 x3 x4 / p r ss1 ss2
                          covb  collin;
  output out=set2 residual=r
                      predicted=yhat;
  run;


/*  Examine smaller regression models
    corresponding to subsets of the
    explanatory variables  */


proc reg data=set1;
  model  y = x1 x2 x3 x4 /
            selection=rsquare cp aic
            sbc mse stop=4 best=6;
  run;
```

```
/*  Regress y on two of explanatory
    variables and check residual plots
    and collinearity diagnostics */

proc reg data=set1 corr;
  model y = x1 x2 / p r ss1 ss2
                      covb  collin;
  output out=set2 residual=r
                     predicted=yhat;
  run;


/*  Use the GLM procedure to identify
    all estimable functions  */

proc glm data=set1;
  model y = x1 x2 x3 x4 / ss1 ss2 e1 e2 e p;
  run;
```

OBS	Evolved heat (calories)	Percent tricalcium aluminate	Percent tricalcium silicate	Percent tetracalcium aluminate ferrate	Percent dicalcium silicate
1	78.5	7	26	6	60
2	74.3	1	29	15	52
3	104.3	11	56	8	20
4	87.6	11	31	8	47
5	95.9	7	52	6	33
6	109.2	11	55	9	22
7	102.7	3	71	17	6
8	72.5	1	31	22	44
9	93.1	2	54	18	22
10	115.9	21	47	4	26
11	83.8	1	40	23	34
12	113.2	11	66	9	12
13	109.4	10	68	8	12

Correlation

Variable	x1	x2	x3	x4	y
x1	1.0000	0.2286	-0.8241	-0.2454	0.7309
x2	0.2286	1.0000	-0.1392	-0.9730	0.8162
x3	-0.8241	-0.1392	1.0000	0.0295	-0.5348
x4	-0.2454	-0.9730	0.0295	1.0000	-0.8212
y	0.7309	0.8162	-0.5348	-0.8212	1.0000

The REG Procedure

Model: MODEL1

Dependent Variable: y

Number of Observations Read	13
Number of Observations Used	13

Analysis of Variance

Source	DF	Sum of Squares	Mean Square	F Value	Pr > F
Model	4	2664.52051	666.13013	111.78	<.0001
Error	8	47.67641	5.95955		
Corrected Total	12	2712.19692			

Root MSE	2.44122	R-Square	0.9824
Dependent Mean	95.41538	Adj R-Sq	0.9736
Coeff Var	2.55852		

Parameter Estimates

Variable	DF	Parameter Estimate	Standard Error	t Value	Pr > |t|	Type I SS	Type II SS
Intercept	1	63.16602	69.93378	0.90	0.3928	118353	4.86191
x1	1	1.54305	0.74331	2.08	0.0716	1448.75413	25.68225
x2	1	0.50200	0.72237	0.69	0.5068	1205.70283	2.87801
x3	1	0.09419	0.75323	0.13	0.9036	9.79033	0.09319
x4	1	-0.15152	0.70766	-0.21	0.8358	0.27323	0.27323

Collinearity Diagnostics

Number	Eigenvalue	Condition Index
1	4.11970	1.00000
2	0.55389	2.72721
3	0.28870	3.77753
4	0.03764	10.46207
5	0.00006614	249.57825

Collinearity Diagnostics

	Proportion of Variation				
Number	Intercept	x1	x2	x3	x4
1	0.00000551	0.00036889	0.00001833	0.00021022	0.00003641
2	8.812348E-8	0.01004	0.00001265	0.00266	0.00010070
3	3.060952E-7	0.00057551	0.00031981	0.00159	0.00168
4	0.00012679	0.05745	0.00278	0.04569	0.00088373
5	0.99987	0.93157	0.99687	0.94985	0.99730

Output Statistics

Obs	Dependent Variable	Predicted Value	Std Error Mean Predict	Residual	Std Error Residual	Student Residual	-2-1 0 1 2	Cook's D
1	78.5000	78.4929	1.8109	0.007071	1.637	0.00432	| | |	0.000
2	74.3000	72.8005	1.4092	1.4995	1.993	0.752	| |* |	0.057
3	104.3000	105.9744	1.8543	-1.6744	1.588	-1.054	| **| |	0.303
4	87.6000	89.3333	1.3265	-1.7333	2.049	-0.846	| *| |	0.060
5	95.9000	95.6360	1.4598	0.2640	1.957	0.135	| | |	0.002
6	109.2000	105.2635	0.8602	3.9365	2.285	1.723	| |*** |	0.084
7	102.7000	104.1289	1.4791	-1.4289	1.942	-0.736	| *| |	0.063
8	72.5000	75.6760	1.5604	-3.1760	1.877	-1.692	| ***| |	0.395
9	93.1000	91.7218	1.3244	1.3782	2.051	0.672	| |* |	0.038
10	115.9000	115.6010	2.0431	0.2990	1.336	0.224	| | |	0.023
11	83.8000	81.8034	1.5924	1.9966	1.850	1.079	| |** |	0.172
12	113.2000	112.3007	1.2519	0.8993	2.096	0.429	| | |	0.013
13	109.4000	111.6675	1.3454	-2.2675	2.037	-1.113	| **| |	0.108

The REG Procedure

Model: MODEL1

Dependent Variable: y

R-Square Selection Method

Number of Observations Read	13
Number of Observations Used	13

Number in Model	R-Square	C(p)	AIC	MSE	SBC	Variables in Model
1	0.6744	139.1583	58.8383	80.26883	59.96815	x4
1	0.6661	142.9553	59.1672	82.32597	60.29712	x2
1	0.5342	203.0030	63.4964	114.85844	64.62630	x1
1	0.2860	315.9455	69.0481	176.04815	70.17804	x3
2	0.9787	2.6886	25.3830	5.77400	27.07785	x1 x2
2	0.9726	5.4882	28.6828	7.44242	30.37766	x1 x4
2	0.9353	22.4391	39.8308	17.54438	41.52565	x3 x4
2	0.8470	62.6435	51.0247	41.50442	52.71951	x2 x3
2	0.6799	138.6599	60.6172	86.80675	62.31201	x2 x4
2	0.5484	198.5303	65.0933	122.48683	66.78816	x1 x3
3	0.9824	3.0156	24.9187	5.30773	27.17852	x1 x2 x4
3	0.9823	3.0458	24.9676	5.32774	27.22742	x1 x2 x3
3	0.9814	3.4829	25.6553	5.61716	27.91511	x1 x3 x4
3	0.9730	7.3094	30.4953	8.15096	32.75514	x2 x3 x4
4	0.9824	5.0000	26.8933	5.95955	29.71808	x1 x2 x3 x4

This output was produced by the e option in the model statement of the GLM procedure. It indicates that all five regression parameters are estimable.

The GLM Procedure

General Form of Estimable Functions

Effect	Coefficients
Intercept	L1
x1	L2
x2	L3
x3	L4
x4	L5

This output was produced by the e1 option in the model statement of the GLM procedure. It describes the null hypotheses that are tested with the equential Type I sums of squares.

Type I Estimable Functions

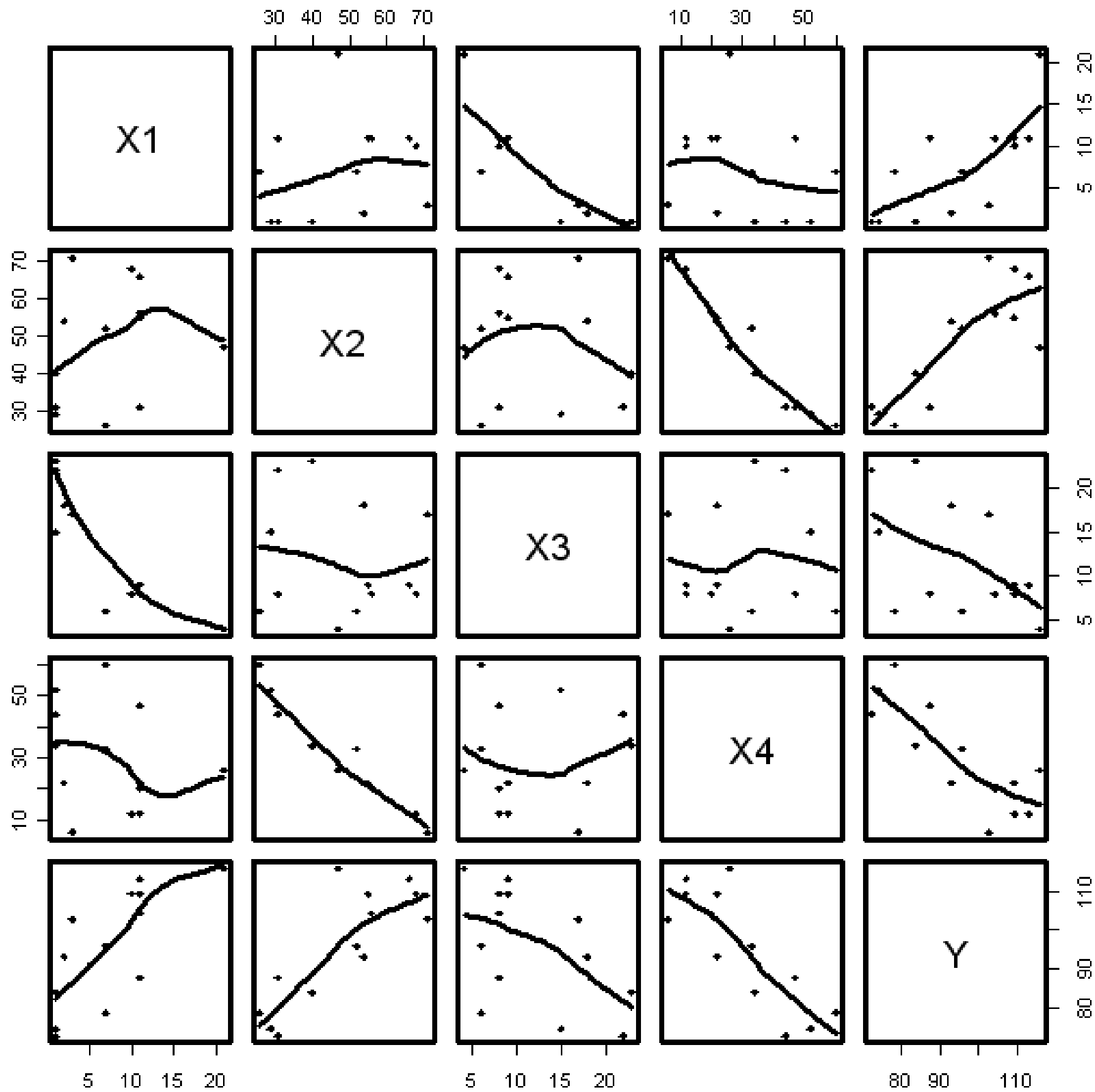
	---------------Coefficients---------------			
Effect	x1	x2	x3	x4
Intercept	0	0	0	0
x1	L2	0	0	0
x2	0.6047*L2	L3	0	0
x3	-0.8974*L2	0.0213*L3	L4	0
x4	-0.6984*L2	-1.0406*L3	-1.0281*L4	L5

Type II Estimable Functions

	----Coefficients----			
Effect	x1	x2	x3	x4
Intercept	0	0	0	0
x1	L2	0	0	0
x2	0	L3	0	0
x3	0	0	L4	0
x4	0	0	0	L5

```
> # This file is stored as   cement.r
>
> # The data file is stored under the name
> # cement.dat.  It has variable names on the
> # first line.  We will enter the data into
> # a data frame.
>
> cement <- read.table("c:/stAT504/cement.txt",header=T)
> cement
   run X1 X2 X3 X4     Y
1    1  7 26  6 60  78.5
2    2  1 29 15 52  74.3
3    3 11 56  8 20 104.3
4    4 11 31  8 47  87.6
5    5  7 52  6 33  95.9
6    6 11 55  9 22 109.2
7    7  3 71 17  6 102.7
8    8  1 31 22 44  72.5
9    9  2 54 18 22  93.1
10  10 21 47  4 26 115.9
11  11  1 40 23 34  83.8
12  12 11 66  9 12 113.2
13  13 10 68  8 12 109.4
>
```

```
> #  Compute correlations and round the results
> #  to four significant digits
>
> round(cor(cement[-1]),4)
        X1      X2      X3      X4       Y
X1  1.0000  0.2286 -0.8241 -0.2454  0.7309
X2  0.2286  1.0000 -0.1392 -0.9730  0.8162
X3 -0.8241 -0.1392  1.0000  0.0295 -0.5348
X4 -0.2454 -0.9730  0.0295  1.0000 -0.8212
Y   0.7309  0.8162 -0.5348 -0.8212  1.0000
>
> #  Create a scatterplot matrix with smooth
> #  curves.  Unix users  should first use motif( )
> #  to open a graphics wundow
>
> points.lines <- function(x, y)
+   {
+  points(x, y)
+  lines(loess.smooth(x, y, 0.90))
+   }
>
>  par(pch=18, mkh=.15, cex=1.2, lwd=3)
>  pairs(cement[ ,-1], panel=points.lines)
>
>
```


X1
X2
X3
X4
Y
30 40 50 60 70
10 30 50
5 10 15 20
30 40 50 60 70
5 10 15 20
10 30 50
80 90 110

```
> #  Fit a  linear regression
>
> cement.out <- lm(Y~X1+X2+X3+X4, cement)
>
> summary(cement.out)

Call:
lm(formula = Y ~ X1 + X2 + X3 + X4, data = cement)

Residuals:
   Min     1Q Median     3Q    Max
-3.176 -1.674  0.264  1.378  3.936

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 63.16602   69.93378   0.903   0.3928
X1           1.54305    0.74331   2.076   0.0716 .
X2           0.50200    0.72237   0.695   0.5068
X3           0.09419    0.75323   0.125   0.9036
X4          -0.15152    0.70766  -0.214   0.8358
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.441 on 8 degrees of freedom
Multiple R-squared: 0.9824,     Adjusted R-squared: 0.9736
F-statistic: 111.8 on 4 and 8 DF,  p-value: 4.707e-07
```

```
> anova(cement.out)
Analysis of Variance Table

Response: Y
          Df  Sum Sq Mean Sq  F value    Pr(>F)
X1         1 1448.75 1448.75 243.0978 2.854e-07 ***
X2         1 1205.70 1205.70 202.3144 5.814e-07 ***
X3         1    9.79    9.79   1.6428    0.2358
X4         1    0.27    0.27   0.0458    0.8358
Residuals  8   47.68    5.96
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
>
>


> #  Create a function to evaluate an orthogonal
> #  projection matrix.  Then create a function
> #  to compute type II sums of squares.
> #  This uses the ginv( )  function in the MASS
> #  library, so you must attach the MASS library
>
>
> library(MASS)
>
>
> #=======================================
> # project( )
> #--------------
> # calculate orthogonal projection matrix
> #=======================================
```

```
> project <- function(X)
+     { X%*%ginv(crossprod(X))%*%t(X)  }
> #======================================
>
>
>
> #=======================================
> # typeII.SS( )
> #-----------------
> # calculate Type II sum of squares
> #
> # input  lmout = object made by the lm( ) function
> #                y = dependent variable
> #
> #=======================================
>
> typeII.SS <- function(lmout,y)
+ {
+                       #  generate the model matrix
+    model <- model.matrix(lmout)
+
+                             #  create list of parameter names
+    par.name <- dimnames(model)[[2]]
+
+                             # compute number of parameters
+    n.par <- dim(model)[2]
+
+                             # Compute residual mean square
+   SS.res <- deviance(lmout)
+ df2    <- lmout$df.resid
+ MS.res <- SS.res/df2
```

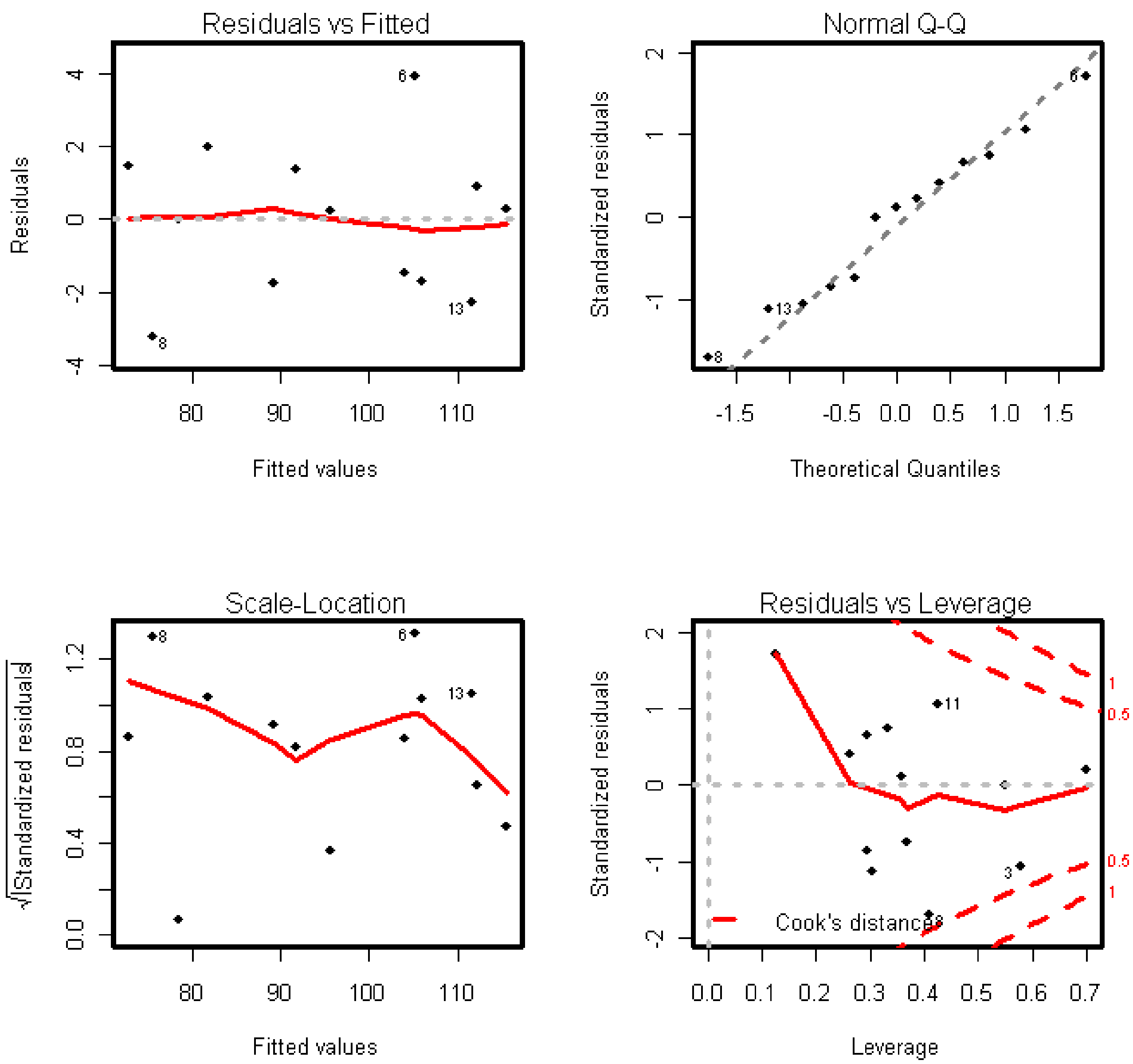
```
+
+ result <- NULL            # store results
+
+                                   # Compute Type II SS
+ for (i in 1:n.par) {
+     A <- project(model)-project(model[,-i])
+     SS.II  <- t(y) %*% A %*% y
+     df1    <- qr(project(model))$rank -
+                          qr(project(model[ ,-i]))$rank
+     MS.II  <- SS.II/df1
+     F.stat <- MS.II/MS.res
+     p.val  <- 1-pf(F.stat,df1,df2)
+     temp   <- cbind(df1,SS.II,MS.II,F.stat,p.val)
+     result <- rbind(result,temp)
+ }
+
+  result <- rbind(result,c(df2,SS.res,MS.res,NA,NA))
+  dimnames(result) <- list(c(par.name,"Residual"),
+   c("Df","Sum of Sq","Mean Sq","F Value","Pr(F)"))
+  cat("Analysis of Variance (TypeII Sum of Squares)\n")
+  round(result,6)
+ }
>
> #===========================================
>
>
>
```

```
> typeII.SS(cement.out, cement$Y)
Analysis of Variance (TypeII Sum of Squares)
            Df Sum of Sq   Mean Sq  F Value    Pr(F)
(Intercept)  1  4.861907  4.861907 0.815818 0.392790
X1           1 25.682254 25.682254 4.309427 0.071568
X2           1  2.878010  2.878010 0.482924 0.506779
X3           1  0.093191  0.093191 0.015637 0.903570
X4           1  0.273229  0.273229 0.045847 0.835811
Residual     8 47.676412  5.959551       NA       NA
```

```
>
> # Venables and Ripley have supplied functions
> # studres( ) and stdres( ) to compute studentized
> # and standardized residuals, respectively.
> # You must attach the MASS library before
> # using these functions.
>
> cement.res <- cbind(cement$Y,cement.out$fitted,
+                               cement.out$resid,
+                               studres(cement.out),
+                               stdres(cement.out))
> dimnames(cement.res) <- list(cement$run,
+        c("Response","Predicted","Residual",
+                     "Stud. Res.","Std. Res."))
```

```
> round(cement.res,4)
   Response Predicted Residual Stud. Res. Std. Res.
1      78.5   78.4929   0.0071     0.0040    0.0043
2      74.3   72.8005   1.4995     0.7299    0.7522
3     104.3  105.9744  -1.6744    -1.0630   -1.0545
4      87.6   89.3333  -1.7333    -0.8291   -0.8458
5      95.9   95.6360   0.2640     0.1264    0.1349
6     109.2  105.2635   3.9365     2.0324    1.7230
7     102.7  104.1289  -1.4289    -0.7128   -0.7358
8      72.5   75.6760  -3.1760    -1.9745   -1.6917
9      93.1   91.7218   1.3782     0.6472    0.6721
10    115.9  115.6010   0.2990     0.2100    0.2237
11     83.8   81.8034   1.9966     1.0919    1.0790
12    113.2  112.3007   0.8993     0.4061    0.4291
13    109.4  111.6675  -2.2675    -1.1326   -1.1131


> # Produce plots for model diagnostics
>
> par(mfrow=c(2,2))
> plot(cement.out)
```



Residuals vs Fitted
Residuals
Fitted values
Normal Q-Q
Standardized residuals
Theoretical Quantiles
Scale-Location
√|Standardized residuals|
Fitted values
Residuals vs Leverage
Standardized residuals
Cook's distance
Leverage

>