Handout R15
Psychology 501 Analysis of Covariance
Prof Colleen Moore and Regression
Montana State University
Prof Emerita, University of Wisconsin

This handout describes the basics of model comparisons for Type III sums of squares using R. The principles apply to all software. The principle apply to any problem where you want to find out what the contribution of a variable is after other variables are entered in a model. Section III below shows graphs for regression.

Overview:
Step 1. Fit a "full model" with all variables of interest.
Step 2. Fit other models, each omitting one variable.
Step 3. Compare the full model to the other models that each omit one variable.

There are at least 3 ways to do this in R. Use 'drop1' function, use 'anova' to compare models, and calculate models by hand. Get Type III SS's from 'Anova' in 'car' package is another option, but I don't show the 'car' package method here.

First, I illuX1 ate with one continuous variable and one dichotomous variable. Then I illuX1 ate for an orthogonally coded but unbalanced 2x2 factorial design.

Quick look at code:

```
> drop1=(modelname, test=c("F")) # gives F tests for all possible models when each variable is separately dropped from the model
# or
> anova(model1, model2)  # gives comparison of two models
##  Also, when using 'lm' if you plan to plot residuals put an 'na.action' option in the lm statement, like this:
> mod1=lm(dv~X1 + X2, na.action = na.exclude)
```

I.A. Bring in data and do some set up

```
> datafilename <-file.choose()  # navigate to your data
> R15 =read.table(datafilename,header=TRUE)  # bring it in
> R15   # check to see what R has
> R15
> R15=read.table(datafilename,header=TRUE)
> R15
     id cond  A  B C G1      DV1   DV2   covar1   covar2       X1       X2       X3
1  Pp54    1  1 -1 2  1 1.333333  1.00 5209.675 2.457333 7.911469 8.021652 8.756911
2  Pp56    1  1 -1 2  2 1.166667 10.50 5093.639 0.252308 7.143996 7.171335 7.727887
3  Pp58    1  1 -1 1  1 2.000000 -5.50 4959.157 0.651017 8.386417 8.499202 9.685800
4  Pp61    2 -1 -1 1  1 1.791667 14.25 5281.021 0.497929 6.986083 7.065525 7.712287
5  Pp66    3 -1  1 2  1 2.958333 -0.25 4321.810 1.273387 7.959967 8.070048 9.368265
. . .
       X4       X5       X6       X7       X8       X9       X10
1 7.208725 6.469940 8.142128 3.632653 4.912007 1.515945 -0.182611
2 6.562198 5.576745 7.487420 3.163733 6.400055 1.689282 -0.522616
```

```
3  7.177512 4.952458  8.346119 3.543761 5.238397 1.916049  0.083951
4  6.373180 5.152551  7.364401 2.965024 4.805220 1.602963  0.188704
5  6.617887 3.599694  7.937875 2.802151 5.232481 1.512760  1.445573
```

I. B. Find the effect of each variable by subtracting from the R-squared for the full model, using either a) the 'drop1' funcion, or b) R's 'anova' function to compare models.

```
> attach(R15)
>  mod1=lm(DV1 ~covar2 +X1 + C)
## mod1 has two possible confounding factors in it, plus the
variable we want to test 'X1 '

> summary(mod1)

Call:
lm(formula = DV1 ~ covar2 + X1 + C)

Residuals:
    Min      1Q  Median      3Q     Max
-1.0140 -0.3931 -0.0056  0.3846  1.3845

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  0.73616    0.68771   1.070   0.2924
covar2      -0.01988    0.15693  -0.127   0.9000
X1           0.20010    0.09030   2.216   0.0339 *
C           -0.36493    0.22770  -1.603   0.1188
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.6057 on 32 degrees of freedom
  (2 observations deleted due to missingness)
Multiple R-squared: 0.1845,Adjusted R-squared: 0.1081
F-statistic: 2.414 on 3 and 32 DF,  p-value: 0.08478

## Also get the anova table for the regression, remember these
are Type I SS, order matters!

> anova(mod1)
Analysis of Variance Table

Response: DV1
          Df  Sum Sq Mean Sq F value Pr(>F)
covar2     1  0.6631 0.66312  1.8076 0.1883
X1         1  1.0513 1.05126  2.8656 0.1002
C          1  0.9423 0.94227  2.5685 0.1188
Residuals 32 11.7393 0.36685


## use 'drop1' to find Type III SS.
> drop1(mod1, test=c("F"))
```

```
Single term deletions

Model:
DV1 ~ covar2 + X1 + C
       Df Sum of Sq    RSS     AIC F value  Pr(>F)
<none>              11.739 -32.341
covar2  1   0.00589 11.745 -34.323  0.0160 0.89998
X1      1   1.80165 13.541 -29.201  4.9111 0.03392 *
C       1   0.94227 12.681 -31.561  2.5685 0.11884
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1


## Compare models in ‘anova’. We should get identical results.
## First we have to calculate the models with one var deleted
## first calculate a regression with everything but X1

> mod2=lm(DV1 ~covar2 +C )
> summary(mod2)

Call:
lm(formula = DV1 ~ covar2 + C)

Residuals:
     Min       1Q   Median       3Q      Max
-1.30678 -0.25598  0.02744  0.40589  1.39889

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   2.0828     0.3406   6.115 6.87e-07 ***
covar2       -0.1772     0.1480  -1.198    0.240
C            -0.1488     0.2176  -0.684    0.499
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.6406 on 33 degrees of freedom
  (2 observations deleted due to missingness)
Multiple R-squared: 0.05939,     Adjusted R-squared: 0.002386
F-statistic: 1.042 on 2 and 33 DF,  p-value: 0.3641

## Now compare models by ‘anova’ function.
> anova(mod2,mod1)
Analysis of Variance Table

Model 1: DV1 ~ covar2 + C
Model 2: DV1 ~ covar2 + X1 + C
  Res.Df    RSS Df Sum of Sq      F  Pr(>F)
1     33 13.541
2     32 11.739  1    1.8016 4.9111 0.03392 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```

```
## interpretation is that 'X1 ' adds significantly to predicting
  DV1  over and above the effects of covar2  and C.

## We can calculate the other partial models and compare to the
'full model' to see the effects of covar2 and C 'over and above'
the effects of each other pair of variables.

> mod3=lm(DV1~X1 +C)  ## omit covar2
> mod4=lm(DV1~X1 + covar2)  ## omit C
> anova(mod3,mod1)
Analysis of Variance Table

Model 1: DV1 ~ X1 + C
Model 2: DV1 ~ covar2 + X1 + C
  Res.Df    RSS Df Sum of Sq     F Pr(>F)
1     33 11.745
2     32 11.739  1 0.0058875 0.016    0.9  # effect of covar2
> anova(mod4,mod1)
Analysis of Variance Table

Model 1: DV1 ~ X1 + covar2
Model 2: DV1 ~ covar2 + X1 + C
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     33 12.681
2     32 11.739  1   0.94227 2.5685 0.1188  # effect of C

## results match those from the 'drop1' function.
## the 'drop1' function gives us the model comparisons in one
step
```

II. A. Comparing models to create Type III SS for a factorial design. This design is unbalanced, and I've coded the factors 'A' and 'B ' orthogonally. Let's calculate 3 partial models, each omitting one variable from the factorial design (A, B, AxB).
-- 'drop1' doesn't exclude main effects that also have an interaction term in the model, so we'll use 'anova'.

```
# first the 'full model'. Could also fit these with 'aov'.
> mod11=lm(DV1 ~A *B )  # * means to include interaction
> mod12=lm(DV1 ~A +B ) # omits interaction
> mod13=lm(DV1 ~B +A:B ) # omits A  main effect
> mod14=lm(DV1 ~A +A:B )  # omits B  main effect
```

Look at the 'anova' of the full model. Remember, these are Type I SS that are order dependent !!

```
> anova(mod11)
Analysis of Variance Table
```

```
Response: DV1
          Df  Sum Sq Mean Sq F value  Pr(>F)
A          1  1.2736 1.27362  3.1495 0.08546 .
B          1  0.1808 0.18076  0.4470 0.50856
A:B        1  0.0013 0.00126  0.0031 0.95584
Residuals 32 12.9403 0.40438
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

## Next compares full model with the one omitting ‘A ’ main
effect, so this should give us the Type III SS for A . Notice
that the F and the p-level don’t match the
Type I SS result above

> anova(mod13,mod11)
Analysis of Variance Table

Model 1: DV1 ~ B + A:B
Model 2: DV1 ~ A * B
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     33 14.086
2     32 12.940  1    1.1459 2.8336  0.102

##  the p-level and F are close, but not exact


> anova(mod14,mod11)  # tests ‘B ’ main effect
Analysis of Variance Table

Model 1: DV1  ~ A  + A:B
Model 2: DV1  ~ A  * B
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     33 13.122
2     32 12.940  1   0.18132 0.4484 0.5079  #effect of B

## finally, test the interaction

> anova(mod12,mod11)  # mod 12 omits A  x B
Analysis of Variance Table

Model 1: DV1  ~ A  + B
Model 2: DV1  ~ A  * B
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     33 12.941
2     32 12.940  1 0.0012595 0.0031 0.9558


## this F and p match Type I but only because the interaction is
## entered last in R’s default calculation
## interpretation: the A  x B  interaction is non-sig.
```

Then do some typing to accumulate your results in a Summary Table:

	df	SS	MS	F	P
A	1	1.1459	1.1459	2.8336	0.102
B	1	0.18132	0.18132	0.4484	0.5079
A :B	1	0.0013	0.00126	0.0031	0.9558
Residuals	32	12.9403	0.40438		

II. B. Calculate by hand instead of using 'anova' to compare models. From this you learn how to compare two multiple regressions and calculate the F's from the R-squared differences.

Step 1: Use 'summary' for each model in II.A. to get the R-squared. List the R-squared values and their dfs. The dfs are for the model as a whole. (Summaries not shown here).

```
R-sq full model (mod11) = 0.1011, model df= 3, 32
R-sq omit 'A ' main effect (mod13) = 0.02152, df=2,33
R-sq omit 'B ' main effect (mod14) = 0.08852, df=2, 33
R-sq omit A x B  (mod12) = 0.1010, model df = 2, 32
```

Step 2: Find an error term. Calculate (1 - R-sq) for full model, divide by df-denominator. This is your error term for testing all effects because this is a between-groups design

```
> denom=(1-0.1011)/32
```

Step 3: Calculate the numerator of the F-test for each effect. Because this is a 2x2 design, each effect has 1 df.

```
> numA =(0.1011-0.02152)/1 #(R-sq full – R-sq omit A )/(df A )
```

Step 4: Calculate F value:

```
> FA =numA /denom; FA   # cA  F for A
[1] 2.832974   # matches results within rounding error
```

Step 5: find p-value for F

```
> pf(FA , 1, 32, lower.tail = F, log.p = FALSE)
[1] 0.1020804
##  'pf' is the function to find the prob of a given F-value
```

-- Finish up the rest of the effects on your own

III. Plot some interesting things

III. A.

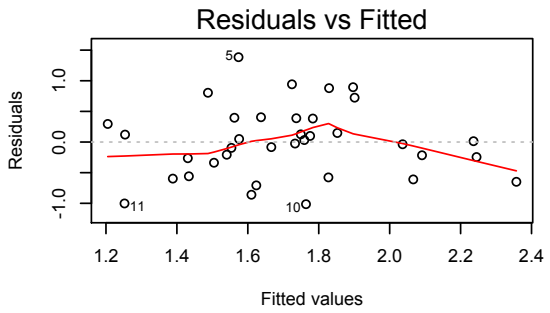
Plot the model fit and regression diagnostics

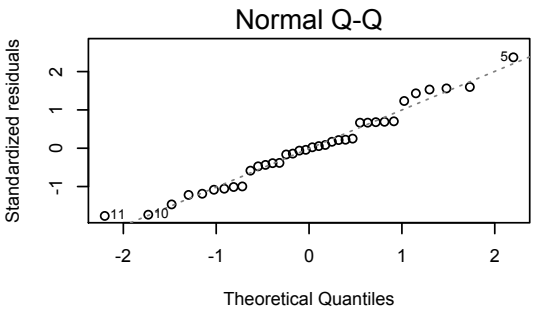
```
> plot.lm(mod1, 1:6)
## this command gives 6 graphs in sequence:
     1. residuals vs fitted values
     2. normal Q-Q of the residuals for the model
```

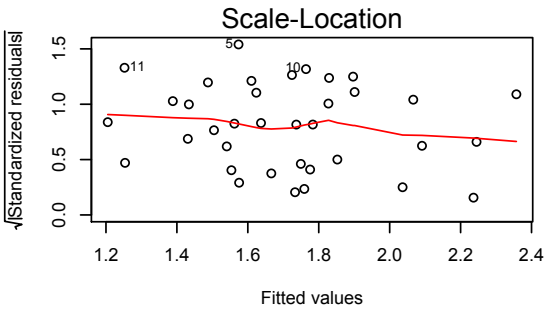
3. sq rt of standardized resid vs fitted values
4. Cook's distance arranged by observation number
5. Standardized residuals vs 'leverage'
6. Cook's distance vs 'leverage'

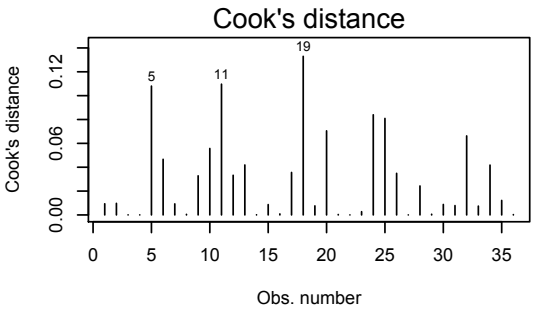
1. Residuals vs predicted values should show no pattern, or be spread horizontally along the zero point of the residuals. If ther is a pattern it implies a poor fit. Curvilinear? Heteroskedacity (a cone shaped pattern)?
2. QQ plot should cluster along the line
3. The standardized residuals give a better look at outliers that may be influential
4. Cook's distance is a measure of the 'influence' a data point has on the regression coefficients. Influence is discrepancy times 'leverage'.
5. Leverage is measured by the 'hat' values, the degree to which an observation influences the predicted (fitted) values. The hat value (h) is the degree to which the observation deviates from the mean of the X predictor variable (or centroid of the several X predictors).
6. See 4 and 5.

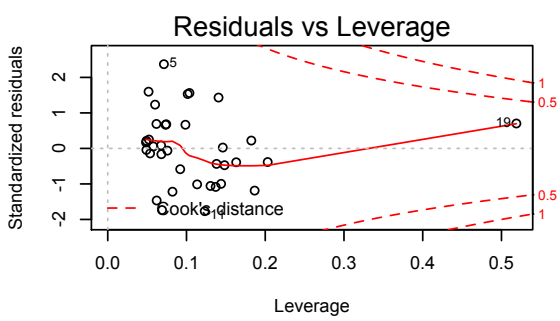
```
## set up to put the 6 graphs on one page
> par(mfrow = c(3,2)) # number of rows and columns to make
> plot.lm(mod1,1:6)
```

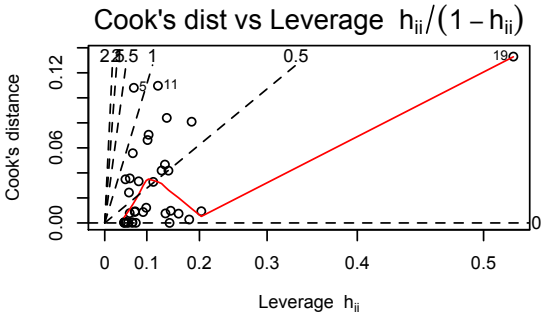












III.B. Fundamentals and Handy Functions

1. Make a correlation matrix

```
> cor(R15 [,3:10], use="pairwise.complete.obs")
                A           B           C          G1         DV1          DV2
A       1.000000000  0.04494666 -0.005578849  0.04494666  0.2974409 -0.16315074
B       0.044946657  1.00000000  0.411857914  0.13636364  0.1422254 -0.54227087
C      -0.005578849  0.41185791  1.000000000 -0.01692567 -0.1360627 -0.03742446
G1      0.044946657  0.13636364 -0.016925668  1.00000000 -0.2236895 -0.03562463
DV1     0.297440906  0.14222540 -0.136062717 -0.22368945  1.0000000 -0.31379543
DV2    -0.163150742 -0.54227087 -0.037424455 -0.03562463 -0.3137954  1.00000000
covar1  0.048613367 -0.06836784 -0.003671114 -0.11465838  0.2310290  0.01439627
covar2 -0.188510603 -0.28218889  0.090522730 -0.27030882 -0.2146228  0.14564538
             covar1      covar2
A       0.048613367 -0.18851060
B      -0.068367836 -0.28218889
C      -0.003671114  0.09052273
G1     -0.114658384 -0.27030882
```

```
DV1     0.231029013 -0.21462282
DV2     0.014396271  0.14564538
covar1  1.000000000 -0.03026831
covar2 -0.030268315  1.00000000
```

2. Test sig of a single correlation coefficient

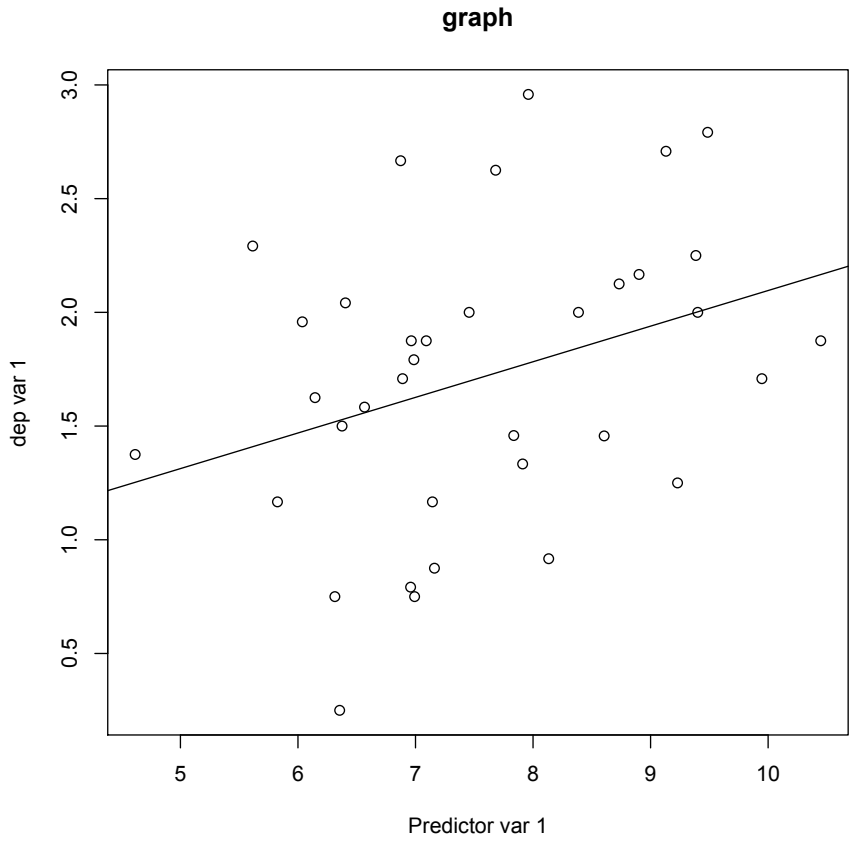
```
> cor.test(DV1 ,DV2, method="pearson")

      Pearson's product-moment correlation

data:  DV1 and DV2
t = -1.9271, df = 34, p-value = 0.06236
alternative hypothesis: true correlation is not equal to 0
95 percent confidence interval:
 -0.5823001  0.0164346
sample estimates:
       cor
-0.3137954
```

3. Make a single scatter plot

```
> plot(X1,DV1) # the variable names
> plot(X1,DV1,xlab="Predictor var 1", ylab="dep var
1",main="graph")# with labels
> > abline(lm(DV1~X1 )) # adds linear regression to graph
```



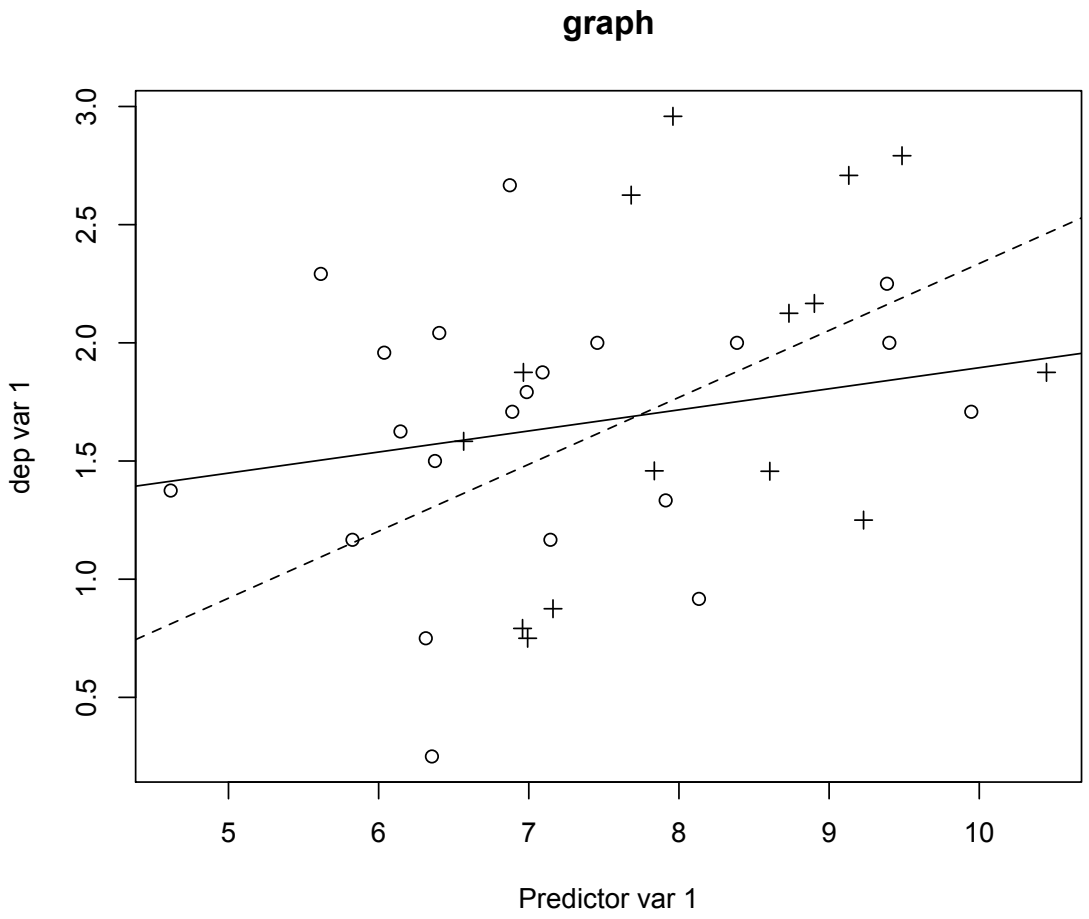
```
## Next,scatter plot with different symbols for different groups
## The option 'pch' selects the plot character. Because I coded B
as 1 and -1 I added 2 to it to select characters 1 and 3.

> plot(X1,DV1,xlab="Predictor var 1", ylab="dep var
1",main="graph", pch=(B+2))

## fit linear regressions to subsets, get the summaries so that
you have the coefficients, plug the coefficients into the
'abline' statement, then plot the two regression lines

> mod3=lm(X1 ~DV1 ,subset(R15 ,B ==1))
> mod4=lm(X1 ~DV1 ,subset(R15 ,B ==-1))

## put values from the models into 'abline' statements
> abline(a=-.4964,b=0.2832,lty=2)
> abline(a=1.0028,b=.0892,lty=1)
```



4. Matrix of scatter plots.

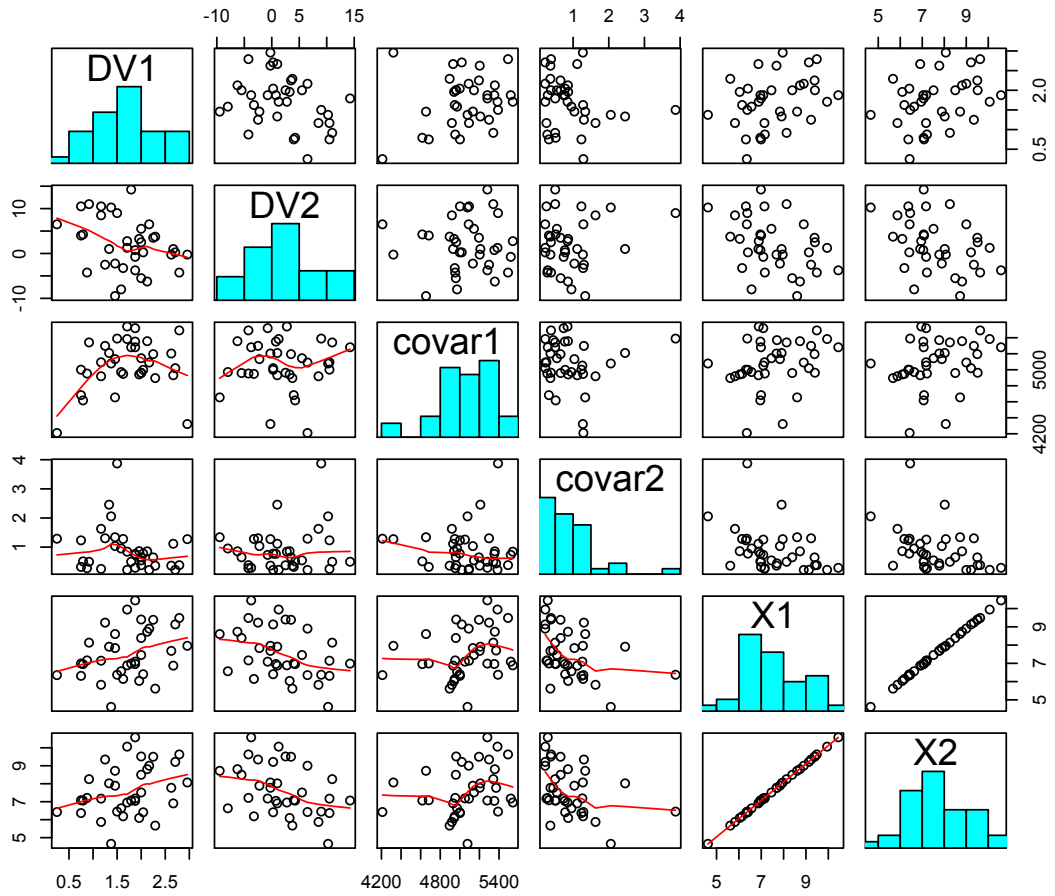
```
> pairs(R15 [,7:12], na.action=na.exclude)  # simple

## A function to use with 'pairs' to put histograms down the
diagonal of the matrix of scatter plots. Paste this into R, and
then hit return. Then put 'diag.panel=panel.hist' into the
'pairs' function. Enter this in R exactly as written here.

panel.hist <- function(x, ...)
 {
     usr <- par("usr"); on.exit(par(usr))
     par(usr = c(usr[1:2], 0, 1.5) )
     h <- hist(x, plot = FALSE)
    breaks <- h$breaks; nB <- length(breaks)
    y <- h$counts; y <- y/max(y)
     rect(breaks[-nB], 0, breaks[-1], y, col="cyan", ...)
 }


>
## now use it in the 'pairs' function, and can add the Lowess
smoothed curve in the lower plot to see how nonlinear they are

> pairs(R15 [,7:12], diag.panel=panel.hist,
lower.panel=panel.smooth, na.action=na.exclude)
```


DV1
DV2
covar1
covar2
X1
X2
-10 0 5 15
1 2 3 4
5 7 9
0.5 2.0
-10 0 10
4200 5000
1 2 3 4
5 7 9
5 7 9
0.5 1.5 2.5
4200 4800 5400
5 7 9