Transcript of Mick Crawley's R course 2010 Imperial College London, Silwood Park

Emanuel G Heitlinger

Disclaimer: The following document is a private transcript of Mick Crawley's R-course. I am a participant in this course and my writeup has in no way been approved by Mick Crawley (from whom the ideas behind the code and teaching concepts are) or any of his staff.

Day 7: Nested analysis and variance components

Nested Analysis

We take a look at a classic example for pseudoreplication.

```
> rat <- read.table("rats.txt", header = T)
> attach(rat)
> Treatment <- factor(Treatment)
> Rat <- factor(Rat)
> Liver <- factor(Liver)
```

In this experiments 3 treatments have been administated to 2 rats each. From these 6 rats three pieces of liver were taken and Glycogen content was measured for each of the 18 pieces twice, leading to a total of 36 datapoints.

Clearly it would be wrong to analyse the data as if there were 12 replicates for each treatment: The replicates for Treatment are only 2! 2 Rats: The spatial scale immediately below the interesting factor. These form of data provides a big risk of conducting the wrong analysis.

One very easy approach of conducting the right analysis is taking means of the pseudoreplicates.

```
> trm.mean <- as.vector(tapply(Glycogen, list(Treatment, Rat),
+     mean))
```

```
> trm <- factor(rep(c(1, 2, 3), 2))
> summary(aov(trm.mean ~ trm))

            Df Sum Sq Mean Sq F value Pr(>F)
trm          2 259.59 129.796   2.929 0.1971
Residuals    3 132.94  44.315
```

Not a very impressive experiment! But is there not more information in the data? On the level of comparing treatments, **no**! But we can learn something about the variance in different pieces of rat-liver and about the variance in the measurements (the measurement error). The means for theses factors are uninteresting as the factor-levels are uninformative.

First the calculations by hand. In nested designs we have to use a special correction factor, the uncorrected sum of squares from the spacial scale **above**.

So for the highest level everything stays the same:

$$SS_{treat} = \frac{\sum T^2}{n} - \frac{(\sum y)^2}{nk}$$

Where n is the number of observations in a treatment and k the number of treatments. But

$$SS_{Rats} = \frac{\sum R^2}{n_{rats}} - \frac{\sum T^2}{n_{treat}}$$
$$SS_{liver.bits} = \frac{\sum L^2}{n_{bits}} - \frac{\sum R^2}{n_{rat}}$$
$$SS_{measure} = \frac{\sum y^2}{1} - \frac{\sum L^2}{n_{bits}}$$

We can make use of our half way generalized functions for anova to calculate these sums of squares (note the use of combined factor levels to get the right n:

```
> CF <- function(x) sum(x)^2/length(x)
> Q.aov <- function(x, fac) {
+     sum(as.vector((tapply(x, fac, sum)^2)/tapply(x, fac, length)))
+ }
> ss.treat <- Q.aov(Glycogen, Treatment) - CF(Glycogen)
> ss.rats <- Q.aov(Glycogen, Treatment:Rat) - Q.aov(Glycogen, Treatment)
> ss.liver <- Q.aov(Glycogen, Treatment:Rat:Liver) - Q.aov(Glycogen,
+     Treatment:Rat)
```

```
> ss.measure <- sum(Glycogen^2) - Q.aov(Glycogen, Treatment:Rat:Liver)
> ss.total <- sum(Glycogen^2) - CF(Glycogen)
> ss.all <- c(SS.treat = ss.treat, SS.rats = ss.rats, SS.liver = ss.liver,
+     SS.measure = ss.measure, SS.total = ss.total)
> ss.all

  SS.treat    SS.rats   SS.liver SS.measure   SS.total
 1557.5556   797.6667   594.0000   381.0000  3330.2222
```

Next key thing to understand are the degrees of freedom. Take some time to see what the code does: It takes the number of observations on the level below minus one (to get the degrees of freedom) times the number of observations on that level.

```
> df.treat <- nlevels(Treatment) - 1
> df.rats <- (nlevels(Rat) - 1) * nlevels(Treatment)
> df.bits <- (nlevels(Liver) - 1) * nlevels(Rat:Treatment)
> df.measure <- (2 - 1) * nlevels(Rat:Treatment:Liver)
> df.total <- length(Glycogen) - 1
> df.all <- c(DF.treat = df.treat, DF.rats = df.rats, DF.bits = df.bits,
+     DF.measure = df.measure, DF.total = df.total)
```

Based on this it is easy to get the variances (mean squares):

```
> ms.all <- ss.all/df.all
> names(ms.all) <- sapply(names(ms.all), function(x) gsub("SS",
+     "MS", x))
> ms.all

  MS.treat    MS.rats   MS.liver MS.measure   MS.total
 778.77778  265.88889   49.50000   21.16667   95.14921
```

Next important thing to understand is that to calculate the F-statistics not the error-variance is used but the **variance of the spatial scale below**. The reason for this is that this time the spacial error-varaince is not a quantity excluding other error-variances. Or to say it another way, every variance still includes additional error for the spacial scale above.

```
> Fstat <- sapply(1:3, function(i) ms.all[i]/ms.all[i + 1])
> names(Fstat) <- sapply(names(Fstat), function(x) gsub("MS", "Fstat",
+     x))
> Fstat
```

```
Fstat.treat  Fstat.rats Fstat.liver
   2.928959    5.371493    2.338583
```

The procedure for the probabilities is very much like that. It has to come from F-tables for the degrees of freedom from the actual spacial scale and the one below.

```
> p.val <- sapply(1:3, function(i) 1 - pf(Fstat[i], df.all[i],
+     df.all[i + 1]))
> names(p.val) <- sapply(names(p.val), function(x) gsub("Fstat",
+     "p.val", x))
> p.val

p.val.treat  p.val.rats p.val.liver
 0.19709896  0.01410908  0.05029073
```

Now first note, that the exactly same p-value for differences in treatments is archieved with this complicated analysis as with simply taking means for pseudoreplicates.

Then we can compare our archievements with R's anova for these kind of analyses:

```
> summary(aov(Glycogen ~ Treatment + Error(Treatment/Rat/Liver)))

Error: Treatment
          Df Sum Sq Mean Sq
Treatment  2 1557.6  778.78

Error: Treatment:Rat
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  3 797.67  265.89

Error: Treatment:Rat:Liver
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals 12    594    49.5

Error: Within
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals 18    381  21.167
```

Note in this output there are no F-values and probabilities, but trust me tay are right, as the Sum sq and MS are.

The difference in means between the so called **random effects** (rats, liverpieces and measurements) is not interesting, as their facto levels are uninformative. But we can work out how much these effects contribute to the overall variance, this might help to plann a better experimental design. We do this simply by substracting the variance for a certain level from the variance immediately below to get the additional variance. We express this variance as a percentage of the overall variance. This simple technique is called **variance component analysis**.

```
> var.diff <- (c(sapply(1:3, function(i) ms.all[[i]] - ms.all[[i +
+     1]]), ms.all[[4]]))
> ms.all[[1]] == sum(var.diff)
```

```
[1] TRUE
```

```
> var.comp <- (var.diff/sum(var.diff)) * 100
> names(var.comp) <- c("Treatment %", "Rats %", "Liver %", "Measurement %")
> var.comp
```

```
  Treatment %        Rats %       Liver % Measurement %
    65.858182     27.785704      3.638179      2.717934
```

This tells us that if we do a proper experiment we could hope to get significant results for different treatments. The variances for different Liver pieces and the measurement error are low compared to variance for individual rats. The experiment should concentrate on a bigger sample size for rats e.g. several genotypes of inbread lines would be clever to compare (this would allow for informative factor levels instead of rat 1 to many).

More complex examples and variance component analysis in R

A fly experiment

Read in some data: Three male flies were mated to four females (in total 12 females; note that female is not nested in male despite the repetetive factor levels of female). These mating was repeated once for each couple of flies.

```
> fly <- read.table("flies.txt", header = TRUE)
> attach(fly)
```

Beside aov there is lme (from the nmle-package) in R to analyse nested data.

```
> library(nlme)
> fm <- lme(eye ~ 1, random = ~1 | male/female)
> summary(fm)

Linear mixed-effects model fit by REML
 Data: NULL
       AIC      BIC    logLik
  138.5075 143.0495 -65.25376

Random effects:
 Formula: ~1 | male
        (Intercept)
StdDev:    4.207861

 Formula: ~1 | female %in% male
        (Intercept) Residual
StdDev:    9.743854 1.140905

Fixed effects: eye ~ 1
               Value Std.Error DF  t-value p-value
(Intercept) 66.63333  3.723998 12 17.89296       0

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.29844030 -0.58190558  0.03583860  0.55001891  1.41870043

Number of Observations: 24
Number of Groups:
            male female %in% male
               3               12
```

Note that it used restricted maximum likelyhood (REML) to estimate the model parameters, if we wanted to do model simplification we would have to specify method=ML.

We extract the variances from our lme model object using VarCorr and express them as percentage of the total variance.

```
> temp <- as.numeric(VarCorr(fm)[, "Variance"])
> vars <- temp[!is.na(temp)]
> comp <- vars/sum(vars) * 100
> names(comp) <- c("male %", "female %", "same m and f %")
> comp
```

```
      male %       female % same m and f %
   15.538414      83.319278       1.142308
```

Most variation is between females covered by the same male. Different males seem to mave not much of an influence and breeding attempts between the same individuals vary hardly.

Now we can ask wether the 15% variation attributable to males is significant. We make a model with a new factor, as female was nested in males, leading to no pseudoreplication in females.

```
> nf <- gl(12, 2)
> fm2 <- lme(eye ~ 1, random = ~1 | nf)
> summary(fm2)

Linear mixed-effects model fit by REML
 Data: NULL
       AIC      BIC    logLik
  136.7889 140.1953 -65.39443

Random effects:
 Formula: ~1 | nf
        (Intercept) Residual
StdDev:    10.38362 1.140907

Fixed effects: eye ~ 1
               Value Std.Error DF  t-value p-value
(Intercept) 66.63333  3.006527 12 22.16289       0

Standardized Within-Group Residuals:
        Min          Q1         Med          Q3         Max
-1.29169080 -0.58861037  0.01665371  0.54393003  1.42544637

Number of Observations: 24
Number of Groups: 12

> anova(fm, fm2)

    Model df      AIC      BIC    logLik   Test   L.Ratio p-value
fm      1  4 138.5075 143.0495 -65.25376
fm2     2  3 136.7889 140.1953 -65.39443 1 vs 2 0.2813399  0.5958
```

The new model ignoring males is not significantly poorer than the old one including the males. But note that the sample size was extremely small (3 for males). In a bigger experiment including more males this could become significant.

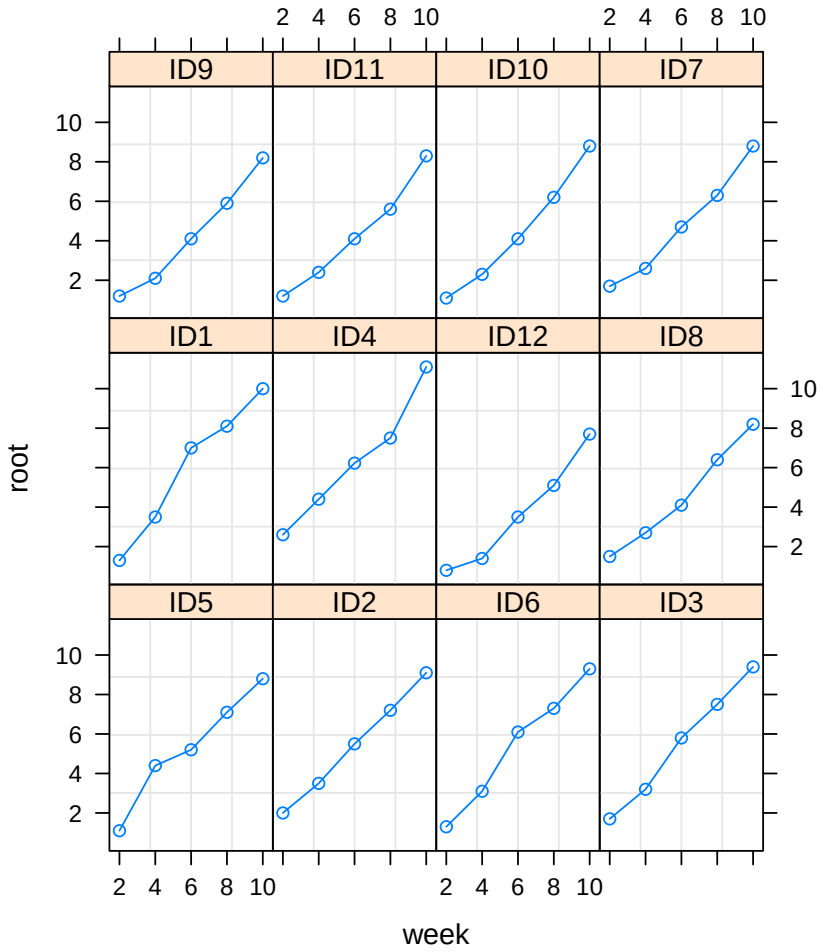
Mixed effect models for temporal pseudoreplication

We read in data from a classical field experiment. We have two levels for treatment with fertilizer, 6 replicate plants for each of this levels and 5 measurements for each plant (temporal pseudoreplicates). Root biomass is the response.

```
> grow <- read.table("fertilizer.txt", header = TRUE)
> attach(grow)
```

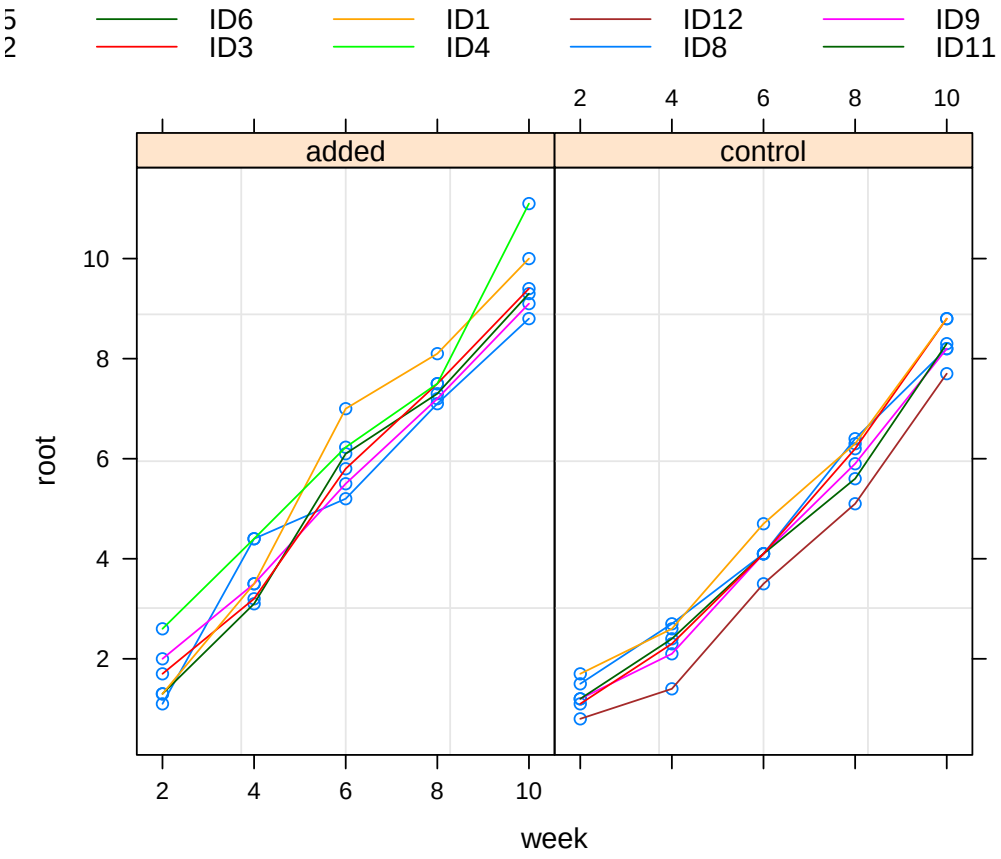
First we do a graphical exploration of the data: We use a grouped Data object.

```
> library(lattice)
> gr <- groupedData(root ~ week | plant, outer = ~fertilizer, grow)
> grpl <- plot(gr)

> print(grpl)
```



To understand the fised effect it might be better to group the 6 replicate in each treatment.

```
> grpl2 <- plot(gr, outer = T)

> print(grpl2)
```



Now we start modelling.

Our fixed effects or root fertilizer the random effects are week|plant. We can write it like this as we have a continuous random effect instead of 1| for categorical random effects.

```
> model <- lme(root ~ fertilizer, random = ~week | plant)
> summary(model)

Linear mixed-effects model fit by REML
 Data: NULL
       AIC      BIC    logLik
  171.0236 183.3863 -79.51181

Random effects:
```

```
 Formula: ~week | plant
 Structure: General positive-definite, Log-Cholesky parametrization
            StdDev    Corr
(Intercept) 2.8639832 (Intr)
week        0.9369412 -0.999
Residual    0.4966308

Fixed effects: root ~ fertilizer
                     Value Std.Error DF   t-value p-value
(Intercept)        2.799710 0.1438367 48 19.464500   0e+00
fertilizercontrol -1.039383 0.2034158 10 -5.109644   5e-04
 Correlation:
                  (Intr)
fertilizercontrol -0.707

Standardized Within-Group Residuals:
       Min         Q1        Med         Q3        Max
-1.9928118 -0.6586834 -0.1004301  0.6949714  2.0225381

Number of Observations: 60
Number of Groups: 12
```

From all this complex output we only need to learn, that the unfertilized controls have a about -1.03 reductio in root mass. With a standard error of about 0.203 on 10 redidual degrees of freedom and this is highly significant.

We can compare this to an anova restricted to week 10:

```
> model2 <- aov(root ~ fertilizer, subset = (week == 10))
> summary(model2)

            Df Sum Sq Mean Sq F value   Pr(>F)
fertilizer   1 4.9408  4.9408  11.486 0.006897 **
Residuals   10 4.3017  0.4302
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ

> summary.lm(model2)

Call:
aov(formula = root ~ fertilizer, subset = (week == 10))

Residuals:
```

```
    Min      1Q  Median      3Q     Max
-0.8167 -0.3667 -0.1333  0.4042  1.4833

Coefficients:
                  Estimate Std. Error t value Pr(>|t|)
(Intercept)         9.6167     0.2678  35.915 6.65e-12 ***
fertilizercontrol  -1.2833     0.3787  -3.389   0.0069 **
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ

Residual standard error: 0.6559 on 10 degrees of freedom
Multiple R-squared: 0.5346,        Adjusted R-squared: 0.488
F-statistic: 11.49 on 1 and 10 DF,  p-value: 0.006897
```

```
> detach(grow)
```

The model is in about the same range as the complex model. lme gave us a little bit lower estimate of the difference but a higher significance. Most importantly a correctly specified complex model allows to avoid arbitrary decissions of which data subset to use in a time-series. The reason for the differences are that lme uses Best Linerar Unbiased Predictors (BLUPs), read about them elsewhere!

Analysis of split plot experiments

Split plot experiments can be destinguished from other spatial pseudoreplicated designs by their informative factor levels. Here the means of the factors are interesting and should be compared. Clever designs have the most interesting factor on the deepest spacial scale, hence the most real replicates for this.

```
> spyield <- read.table("splityield.txt", header = TRUE)
> attach(spyield)
```

In this data a field was split in four blocks. In each of these one half was irrigated the other half not. The irrigation sub-blocks were seeded in three densities each, and finally three fertilizer treatments were applied to each of the irrigation-density sub-sub-blocks.

We do this a last time by hand using the semi-automated functions:

```
> sst <- sum(yield^2) - CF(yield)
> ssb <- Q.aov(yield, block) - CF(yield)
> ssi <- Q.aov(yield, irrigation) - CF(yield)
```

Here things get different. The error term is the interaction between blocks and all factors at that plot size or larger. We calculate the interaction term using combined factor lavels again and substract the block and irrigation sum of squares. The large plot error sum of squares is therefore:

```
> ssbi <- Q.aov(yield, block:irrigation) - ssb - ssi - CF(yield)
> SS.big <- c(ssb, ssi, ssbi)
```

At this point we can do the first anova table. We just need to get the degrees of freedom right to to variances an F-statistics. For the error it is here 8 large blocks in the experiment minus 1 minus the irrigation and block df.

```
> df.b <- nlevels(block) - 1
> df.i <- nlevels(irrigation) - 1
> df.bi <- nlevels(block:irrigation) - 1 - df.i - df.b
> DF.big <- c(df.b, df.i, df.bi)
> ms.i <- ssi
> ms.bi <- ssbi/df.bi
> fstat.i <- ms.i/ms.bi
> tval.i <- 1 - pf(fstat.i, df.i, df.bi)
```

Now we can move on a spatial scale below to the density. We calculate density; density and irrigation interaction; and density, irrigation and block interaction.

```
> ssd <- Q.aov(yield, density) - CF(yield)
> ssid <- Q.aov(yield, density:irrigation) - ssi - ssd - CF(yield)
> ssbid <- Q.aov(yield, density:irrigation:block) - ssid - ssd -
+     sum(SS.big) - CF(yield)
> SS.med <- c(ssd, ssid, ssbid)
```

We get the degrees of freedom to calculate variances and Fstats.

```
> df.d <- nlevels(density) - 1
> df.id <- df.d * df.i
> df.bid <- nlevels(irrigation:density:block) - 1 - df.d - df.id -
+     sum(DF.big)
> DF.med <- c(df.d, df.id, df.bid)
> MS.med <- SS.med/DF.med
> fstat.d <- MS.med[1]/MS.med[3]
> fstat.id <- MS.med[2]/MS.med[3]
> tval.d <- 1 - pf(fstat.d, df.d, df.bid)
> tval.id <- 1 - pf(fstat.id, df.id, df.bid)
```

Finally we can move on to the innermost block.

```
> ssf <- Q.aov(yield, fertilizer) - CF(yield)
> ssif <- Q.aov(yield, fertilizer:irrigation) - ssi - ssf - CF(yield)
> ssdf <- Q.aov(yield, fertilizer:density) - ssd - ssf - CF(yield)
> ssidf <- Q.aov(yield, fertilizer:density:irrigation) - ssid -
+     ssif - ssdf - ssi - ssd - ssf - CF(yield)
> SS.small <- c(ssf, ssif, ssdf, ssidf)
> sse <- sst - sum(c(SS.big, SS.med, SS.small))
```

And from this using degrees of freedom via variance and F statistics to the probabilities.

```
> df.f <- nlevels(fertilizer) - 1
> DF.small <- c(df.f = nlevels(fertilizer) - 1, df.if = df.i *
+     df.f, df.df = df.f * df.d, df.idf = df.i * df.d * df.f)
> df.err <- length(yield) - 1 - sum(DF.big) - sum(DF.med) - sum(DF.small)
> MS.small <- SS.small/DF.small
> MS.small

     df.f     df.if     df.df    df.idf
988.72222 476.72222  76.22222  58.68056

> ms.err <- sse/df.err
> Fstat.small <- MS.small/ms.err
> Fstat.small

      df.f      df.if      df.df     df.idf
11.4493111  5.5203989  0.8826462  0.6795154

> tval.small <- sapply(1:4, function(i) 1 - pf(Fstat.small[[i]],
+     DF.small[[i]], df.err))
> tval.small

[1] 0.0001417596 0.0081077586 0.4840525910 0.6106671944
```

So this gives three tables this time:

Finally we can model in R and will understand the output just calculated by hand.

```
> mod <- aov(yield ~ irrigation * density * fertilizer + Error(block/irrigation
> summary(mod)
```

Source	SS	df	MS (var)	F	p
Block	194.444	3			
Irrigation	8277.556	1	8277.556	17.59	0.025
Error	1411.778	3	470.593		

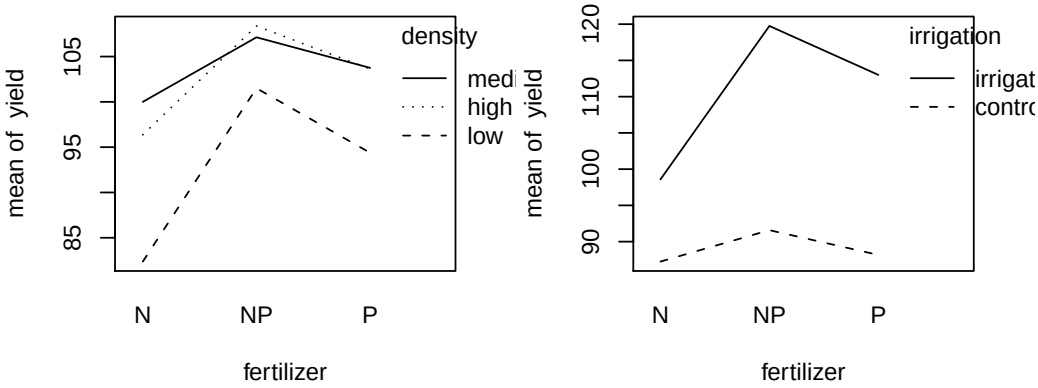
Source	SS	df	MS (var)	F	p
Density	1758.361	2	879.181	3.784	0.053
Irrigation:Density	2747.028	2	1373.514	5.912	0.016
Error	2787.944	12	232.329		

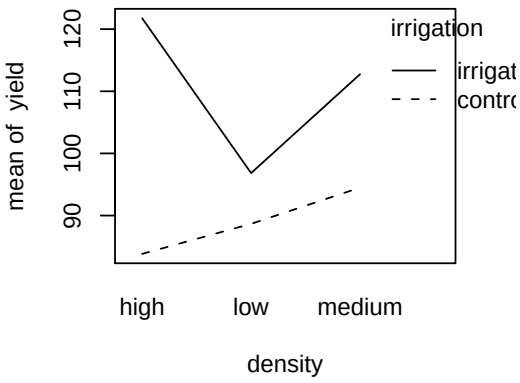
```
Error: block
          Df Sum Sq Mean Sq F value Pr(>F)
Residuals  3 194.44  64.815

Error: block:irrigation
           Df Sum Sq Mean Sq F value  Pr(>F)
irrigation  1 8277.6  8277.6  17.590 0.02473 *
Residuals   3 1411.8   470.6
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ

Error: block:irrigation:density
                   Df Sum Sq Mean Sq F value  Pr(>F)
density             2 1758.4  879.18  3.7842 0.05318 .
irrigation:density  2 2747.0 1373.51  5.9119 0.01633 *
Residuals          12 2787.9  232.33
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ

Error: block:irrigation:density:fertilizer
                              Df  Sum Sq Mean Sq F value    Pr(>F)
fertilizer                     2 1977.44  988.72 11.4493 0.0001418 ***
irrigation:fertilizer          2  953.44  476.72  5.5204 0.0081078 **
density:fertilizer             4  304.89   76.22  0.8826 0.4840526
irrigation:density:fertilizer  4  234.72   58.68  0.6795 0.6106672
Residuals                     36 3108.83   86.36
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ
```

And plot the model.

Source	SS	df	MS (var)	F	p
Fertilizer	1977.444	2	988.722	11.449	0
Irrigation:Fertilizer	953.444	2	476.722	5.52	0.008
Density:Fertilizer	304.889	4	76.222	0.883	0.484
Irrigation:Density:Fertilizer	234.722	4	58.681	0.68	0.611
Error	3108.833	36	86.356		

```
> par(mfrow = c(2, 2))
> interaction.plot(fertilizer, density, yield)
> interaction.plot(fertilizer, irrigation, yield)
> interaction.plot(density, irrigation, yield)
```





An even more complex split-split-split-split-plot experiment

```
> splot <- read.table("splitplot.txt", header = TRUE)
> attach(splot)
```

I will not do this by hand...

```
> model <- aov(Biomass ~ Insect * Mollusc * Rabbit * Lime * Competition *
+     Nutrient + Error(Block/Rabbit/Lime/Competition/Nutrient))
> summary(model)

Error: Block
               Df Sum Sq Mean Sq F value   Pr(>F)
Insect          1 414.61  414.61 34.2712 0.004248 **
Mollusc         1   8.75    8.75  0.7229 0.443088
Insect:Mollusc  1  11.06   11.06  0.9139 0.393209
Residuals       4  48.39   12.10
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ

Error: Block:Rabbit
                     Df Sum Sq Mean Sq   F value    Pr(>F)
Rabbit                1 388.79  388.79 4563.5924 2.877e-07 ***
Insect:Rabbit         1   0.40    0.40    4.6985   0.09607 .
Mollusc:Rabbit        1   0.01    0.01    0.1600   0.70963
Insect:Mollusc:Rabbit 1   0.25    0.25    2.9078   0.16335
Residuals             4   0.34    0.09
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ

Error: Block:Rabbit:Lime
                          Df Sum Sq Mean Sq   F value    Pr(>F)
Lime                       1 86.637  86.637 1918.2643 8.148e-11 ***
Insect:Lime                1  0.034   0.034    0.7556   0.41001
Mollusc:Lime               1  0.122   0.122    2.7005   0.13894
Rabbit:Lime                1  0.146   0.146    3.2284   0.11010
Insect:Mollusc:Lime        1  0.052   0.052    1.1425   0.31631
Insect:Rabbit:Lime         1  0.004   0.004    0.0794   0.78529
Mollusc:Rabbit:Lime        1  0.091   0.091    2.0043   0.19458
Insect:Mollusc:Rabbit:Lime 1  0.467   0.467   10.3353   0.01233 *
Residuals                  8  0.361   0.045
---
```

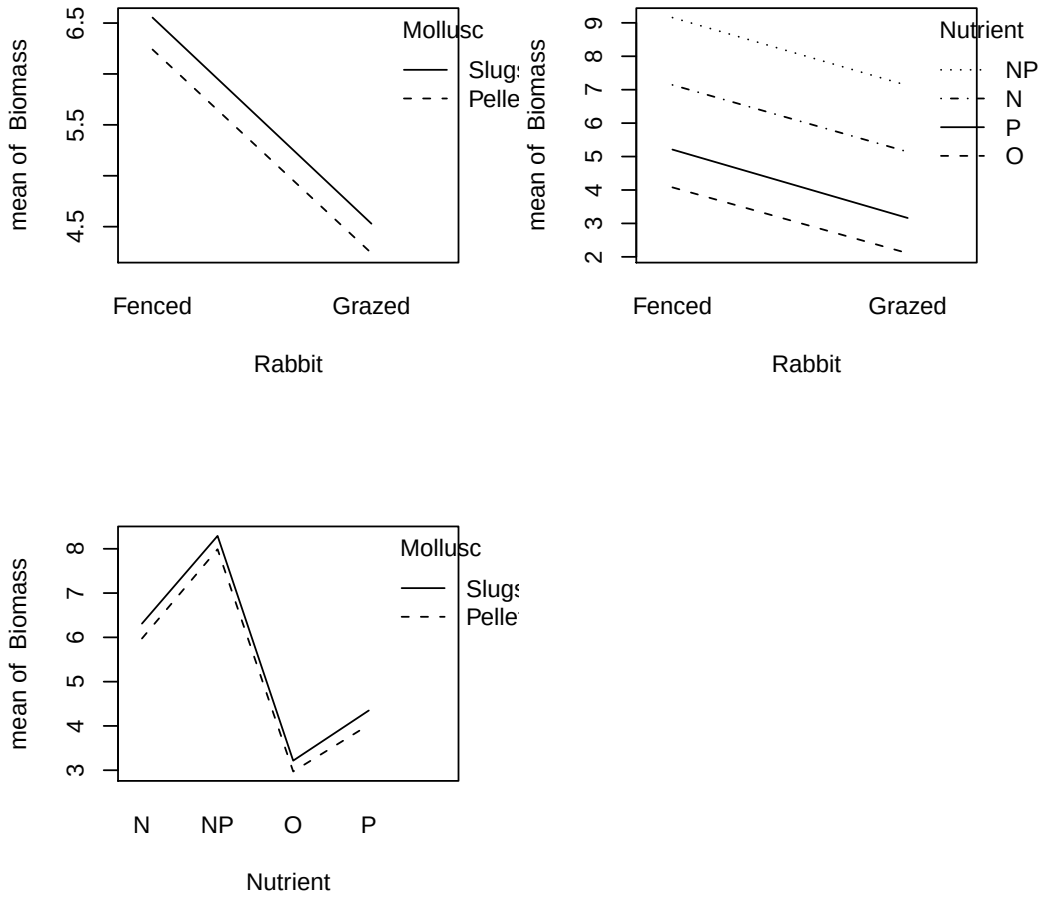
```
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ

Error: Block:Rabbit:Lime:Competition
                                    Df  Sum Sq Mean Sq   F value  Pr(>F)
Competition                          2 214.415 107.207 1188.3171 < 2e-16 ***
Insect:Competition                   2   0.150   0.075    0.8323 0.44425
Mollusc:Competition                  2   0.156   0.078    0.8664 0.43008
Rabbit:Competition                   2   0.198   0.099    1.0981 0.34576
Lime:Competition                     2   0.023   0.011    0.1255 0.88252
Insect:Mollusc:Competition           2   0.413   0.207    2.2899 0.11763
Insect:Rabbit:Competition            2   0.122   0.061    0.6768 0.51537
Mollusc:Rabbit:Competition           2   0.022   0.011    0.1225 0.88509
Insect:Lime:Competition              2   0.053   0.026    0.2919 0.74879
Mollusc:Lime:Competition             2   0.030   0.015    0.1641 0.84939
Rabbit:Lime:Competition              2   0.013   0.007    0.0742 0.92868
Insect:Mollusc:Rabbit:Competition    2   0.031   0.015    0.1702 0.84427
Insect:Mollusc:Lime:Competition      2   0.062   0.031    0.3444 0.71123
Insect:Rabbit:Lime:Competition       2   0.376   0.188    2.0812 0.14135
Mollusc:Rabbit:Lime:Competition      2   0.501   0.250    2.7751 0.07737 .
Insect:Mollusc:Rabbit:Lime:Competition 2 0.011   0.006    0.0635 0.93855
Residuals                           32   2.887   0.090
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ

Error: Block:Rabbit:Lime:Competition:Nutrient
                                         Df  Sum Sq Mean Sq   F value
Nutrient                                  3 1426.02  475.34 6992.5894
Insect:Nutrient                           3    0.21    0.07    1.0457
Mollusc:Nutrient                          3    0.12    0.04    0.5891
Rabbit:Nutrient                           3    0.09    0.03    0.4280
Lime:Nutrient                             3    0.03    0.01    0.1713
Competition:Nutrient                      6    0.72    0.12    1.7751
Insect:Mollusc:Nutrient                   3    0.11    0.04    0.5488
Insect:Rabbit:Nutrient                    3    0.78    0.26    3.8404
Mollusc:Rabbit:Nutrient                   3    0.93    0.31    4.5539
Insect:Lime:Nutrient                      3    0.06    0.02    0.2886
Mollusc:Lime:Nutrient                     3    0.48    0.16    2.3318
Rabbit:Lime:Nutrient                      3    0.38    0.13    1.8419
Insect:Competition:Nutrient               6    0.56    0.09    1.3634
Mollusc:Competition:Nutrient              6    0.35    0.06    0.8510
Rabbit:Competition:Nutrient               6    0.43    0.07    1.0567
```

```
                                                  6    0.45    0.08    1.1062
Insect:Mollusc:Rabbit:Nutrient                    3    0.41    0.14    2.0159
Insect:Mollusc:Lime:Nutrient                      3    0.70    0.23    3.4184
Insect:Rabbit:Lime:Nutrient                       3    0.43    0.14    2.1324
Mollusc:Rabbit:Lime:Nutrient                      3    0.06    0.02    0.2701
Insect:Mollusc:Competition:Nutrient               6    0.56    0.09    1.3764
Insect:Rabbit:Competition:Nutrient                6    0.73    0.12    1.7844
Mollusc:Rabbit:Competition:Nutrient               6    0.43    0.07    1.0472
Insect:Lime:Competition:Nutrient                  6    0.26    0.04    0.6321
Mollusc:Lime:Competition:Nutrient                 6    0.42    0.07    1.0229
Rabbit:Lime:Competition:Nutrient                  6    0.41    0.07    0.9943
Insect:Mollusc:Rabbit:Lime:Nutrient               3    0.35    0.12    1.7150
Insect:Mollusc:Rabbit:Competition:Nutrient        6    0.26    0.04    0.6346
Insect:Mollusc:Lime:Competition:Nutrient          6    0.40    0.07    0.9882
Insect:Rabbit:Lime:Competition:Nutrient           6    0.28    0.05    0.6918
Mollusc:Rabbit:Lime:Competition:Nutrient          6    0.35    0.06    0.8703
Insect:Mollusc:Rabbit:Lime:Competition:Nutrient   6    0.99    0.16    2.4239
Residuals                                       144    9.79    0.07
                                                   Pr(>F)
Nutrient                                       < 2.2e-16 ***
Insect:Nutrient                                 0.374357
Mollusc:Nutrient                                0.623110
Rabbit:Nutrient                                 0.733211
Lime:Nutrient                                   0.915667
Competition:Nutrient                            0.108184
Insect:Mollusc:Nutrient                         0.649746
Insect:Rabbit:Nutrient                          0.011086 *
Mollusc:Rabbit:Nutrient                         0.004432 **
Insect:Lime:Nutrient                            0.833596
Mollusc:Lime:Nutrient                           0.076689 .
Rabbit:Lime:Nutrient                            0.142217
Insect:Competition:Nutrient                     0.233305
Mollusc:Competition:Nutrient                    0.532754
Rabbit:Competition:Nutrient                     0.391469
Lime:Competition:Nutrient                       0.361602
Insect:Mollusc:Rabbit:Nutrient                  0.114331
Insect:Mollusc:Lime:Nutrient                    0.019078 *
Insect:Rabbit:Lime:Nutrient                     0.098722 .
Mollusc:Rabbit:Lime:Nutrient                    0.846884
Insect:Mollusc:Competition:Nutrient             0.227981
Insect:Rabbit:Competition:Nutrient              0.106248
```

```
Mollusc:Rabbit:Competition:Nutrient              0.397423
Insect:Lime:Competition:Nutrient                 0.704370
Mollusc:Lime:Competition:Nutrient                0.412849
Rabbit:Lime:Competition:Nutrient                 0.431503
Insect:Mollusc:Rabbit:Lime:Nutrient              0.166552
Insect:Mollusc:Rabbit:Competition:Nutrient       0.702370
Insect:Mollusc:Lime:Competition:Nutrient         0.435588
Insect:Rabbit:Lime:Competition:Nutrient          0.656564
Mollusc:Rabbit:Lime:Competition:Nutrient         0.518394
Insect:Mollusc:Rabbit:Lime:Competition:Nutrient  0.029138 *
Residuals
---
Signif. codes:  0 âĂŸ***âĂŹ 0.001 âĂŸ**âĂŹ 0.01 âĂŸ*âĂŹ 0.05 âĂŸ.âĂŹ 0.1 âĂŸ âĂŹ
```

Dauntingly complex! We can use interaction.plot to display the iteraction terms two at a time.

```
> par(mfrow = c(2, 2))
> interaction.plot(Rabbit, Mollusc, Biomass)
> interaction.plot(Rabbit, Nutrient, Biomass)
> interaction.plot(Nutrient, Mollusc, Biomass)
```



It demonstrates that the strong effect of fertilizer (bottom left) is hardly altered by mollusc or rabbit. Clearly this model needs simplification.