

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.

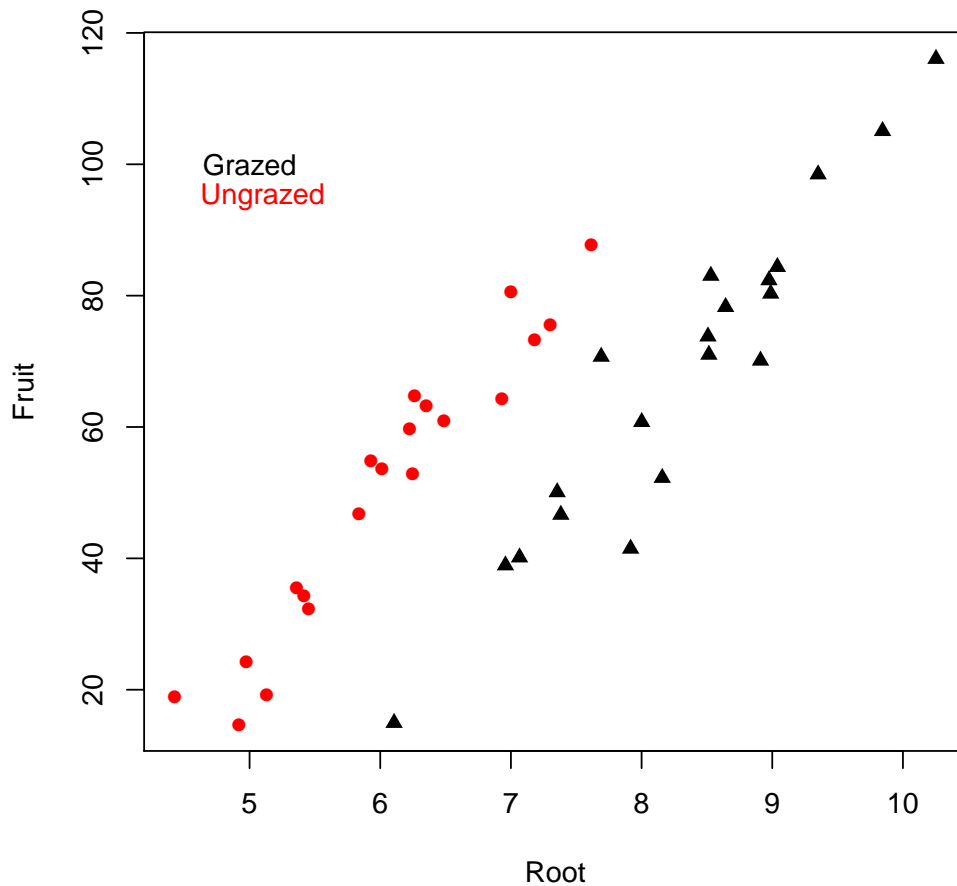
Day 6: Analysis of Covariance (ANCOVA) Part 2

ANCOVA with different values of the covariates

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

Plot making clever use of logic arithmetic

```
> plot(Fruit ~ Root, pch = 16 + (Grazing == "Grazed"), col = as.numeric(Grazing))
> text(5, 100, label = "Grazed")
> text(5.1, 95, label = "Ungrazed", col = 2)
```



Surprisingly the Grazed Plants seem to produce more fruits, but let's take a closer look.

To calculate the sum of squares manually we will use versions of our functions from before (Ancova-Anova writeup). A little general function for correction factors one that can be used to supply factor levels, and the function for the sum of products.

```

> 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)))
+ }
> SSXY <- function(x, y) sum(x * y) - (sum(x) * sum(y))/length(c(y))

```

Based on this we do the calculations for a model with interaction effects (differences in slope) and one without.

Note that we can **not** calculate a slope from the the simple regression sum of squares and the simple square of products, as the cavariates (Roots) have different values in the treatments (unlike photoperiode in the example in anova-ancova).

```

> sst <- sum(Fruit^2) - CF(Fruit)
> ssx <- sum(Root^2) - CF(Root)
> ssa <- Q.aov(Fruit, Grazing) - CF(Fruit)
> ssxy <- SSXY(Fruit, Root)
> b.wrong <- ssxy/ssx
> ssr <- b.wrong * ssxy
> sse <- sst - ssr
> multi.sst <- tapply(Fruit, Grazing, function(x) sum(x^2) - CF(x))
> multi.ssxy <- c(by(ipo, Grazing, function(d) SSXY(d$Fruit, d$Root)))
> multi.ssx <- tapply(Root, Grazing, function(x) sum(x^2) - CF(x))
> multi.b <- multi.ssxy/multi.ssx
> multi.ssr <- multi.b * multi.ssxy
> multi.sse <- multi.sst - multi.ssr
> combi.sst <- sum(multi.sst)
> combi.ssxy <- sum(multi.ssxy)
> combi.ssx <- sum(multi.ssx)
> combi.b <- sum(multi.b)
> combi.ssr <- sum(multi.ssr)
> combi.sse <- sum(multi.sse)

```

Now we can calculate the value of a single common slope and SSR_{diff} a measure for the difference of the two different slopes. Finaly we can calculate the error sum of squares for the modell without differernt slopes.

```

> ssr <- combi.ssxy^2/combi.ssx
> single.b <- combi.ssxy/combi.ssx
> ssr.diff <- combi.ssr - ssr
> sse.inter <- sst - ssa - ssr - ssr.diff
> sse.common <- sst - ssa - ssr

```

We can look at our models, first the complicated one with different slopes, then the less complicated without the non-significant interaction term:

Source	SS	df	MS (var)	F	p
Grazing	2910.436	1	2910.436	62.38	0.08
Root	19148.939	1	19148.939	410.42	0.031
Different slopes	4.812	1	4.812	0.103	0.75
Error	1679.649	36	46.657		
Total	23743.837	39			

Source	SS	df	MS (var)	F	p
Grazing	2910.436	1	2910.436	63.929	0.079
Root	19148.939	1	19148.939	420.616	0.031
Error	1684.461	36	45.526		
Total	23743.837	39			

Compare this with R's ancova. The step function finds the right minimal adequate model.

```
> model <- lm(Fruit ~ Grazing * Root)
> summary.aov(model)
```

```
          Df Sum Sq Mean Sq  F value    Pr(>F)
Grazing    1  2910.4   2910.4   62.3795 2.262e-09 ***
Root        1 19148.9  19148.9  410.4201 < 2.2e-16 ***
Grazing:Root 1     4.8     4.8    0.1031    0.75
Residuals  36  1679.6    46.7
```

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

```
> model2 <- step(model)
```

```
Start:  AIC=157.5
```

```
Fruit ~ Grazing * Root
```

```
          Df Sum of Sq  RSS  AIC
- Grazing:Root  1    4.8122 1684.5 155.61
<none>                1679.7 157.50
```

```
Step:  AIC=155.61
```

```
Fruit ~ Grazing + Root
```

```
          Df Sum of Sq    RSS    AIC
<none>          1684.5 155.61
- Grazing    1     5264.4 6948.8 210.30
- Root       1    19148.9 20833.4 254.22
```

```
> summary.aov(model2)
```

```
          Df Sum Sq Mean Sq F value    Pr(>F)
Grazing    1  2910.4   2910.4   63.929 1.397e-09 ***
Root       1 19148.9 19148.9  420.616 < 2.2e-16 ***
Residuals 37  1684.5    45.5
```

```
---
```

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

Surprisingly the **probabilities** seem to be wrong. But this is because R calculates not the simple p-values but **exclusion p-values**. To explain this a little excursion:

From what we got so far we can calculate the two different intercepts:

$$a_1 = \bar{Y}_1 - b\bar{X}_1$$

$$a_2 = \bar{Y}_2 - b\bar{X}_2$$

```
> multi.a <- by(ipo, Grazing, function(d) mean(d$Fruit) - single.b *
+      mean(d$Root))
> c(multi.a[1], multi.a[2])
```

```
      Grazed  Ungrazed
-127.82936  -91.72611
```

```
> a.diff <- multi.a[[2]] - multi.a[[1]]
> a.diff
```

```
[1] 36.10325
```

This demonstrates, that the ungrazed plants produce more fruits than the grazed plants, which is the opposite to what is obtained by looking at the means.

```
> tapply(Fruit, Grazing, mean)
```

```
Grazed Ungrazed
67.9405 50.8805
```

The **standard error of the intercept**. Standard errors for one common intercepts can be obtained as follows:

$$SE_a = \sqrt{s^2 \left[\frac{1}{n} + \frac{(0 - \bar{x})^2}{SSX} \right]}$$

```
> SE.a <- sqrt(sse.common/37 * (1/length(Root[Grazing == "Grazed"]) +
+ (0 - mean(Root[Grazing == "Grazed"]))^2/combi.ssx))
```

And the **standard error of the difference of the two intercepts** (which is equivalent to the inclusion of grazing).

$$SE_{\hat{y}_1 - \hat{y}_2} = \sqrt{s^2 \left[\frac{2}{n} + \frac{(\bar{x}_1 - \bar{x}_2)^2}{SSX} \right]}$$

```
> SE.yy <- sqrt(sse.common/37 * (2/length(Root[Grazing == "Grazed"]) +
+ (mean(Root[Grazing == "Grazed"]) - mean(Root[Grazing == "Ungrazed"]))^2/combi.ssx))
```

And finally the **standard error of the slope** (the common slope)

$$SE_b = \sqrt{\frac{s^2}{SSX}}$$

```
> SE.b <- sqrt((sse.common/37)/combi.ssx)
> SE.b
```

```
[1] 1.148771
```

These Standard errors can be found in the table generated by `summary.lm` and we can also generate the t-values.

```
> summary.lm(model2)
```

Call:

```
lm(formula = Fruit ~ Grazing + Root)
```

Residuals:

Min	1Q	Median	3Q	Max
-17.1920	-2.8224	0.3223	3.9144	17.3290

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-127.829	9.664	-13.23	1.35e-15 ***
GrazingUngrazed	36.103	3.357	10.75	6.11e-13 ***
Root	23.560	1.149	20.51	< 2e-16 ***

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

Residual standard error: 6.747 on 37 degrees of freedom

Multiple R-squared: 0.9291, Adjusted R-squared: 0.9252

F-statistic: 242.3 on 2 and 37 DF, p-value: < 2.2e-16

```
> c(SE.a = SE.a, SE.yy = SE.yy, SE.b = SE.b)
```

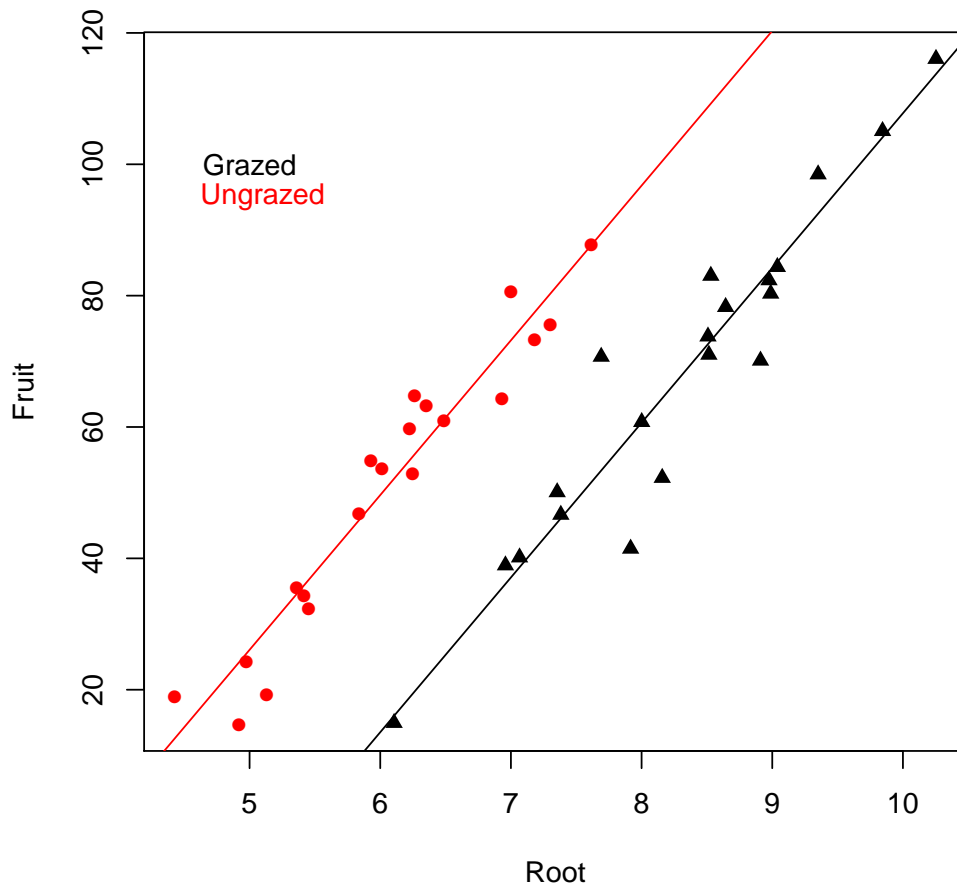
```
      SE.a      SE.yy      SE.b
9.664095 3.357396 1.148771
```

```
> tvals <- c(tval.Grazed = multi.a[[1]]/SE.a, tval.Ungrazed = a.diff/SE.yy,
+           tval.root = single.b/SE.b)
> tvals
```

```
      tval.Grazed tval.Ungrazed      tval.root
      -13.22725      10.75335      20.50892
```

The beauty of the summary output of these models is, that these p-values are presented, which are more important in modeling. So finally after we understood all the output, we can draw two lines in our plot.

```
> plot(Fruit ~ Root, pch = 16 + (Grazing == "Grazed"), col = as.numeric(Grazing))
> abline(multi.a[[1]], single.b)
> abline(multi.a[[2]], single.b, col = 2)
> text(5, 100, label = "Grazed")
> text(5.1, 95, label = "Ungrazed", col = 2)
```

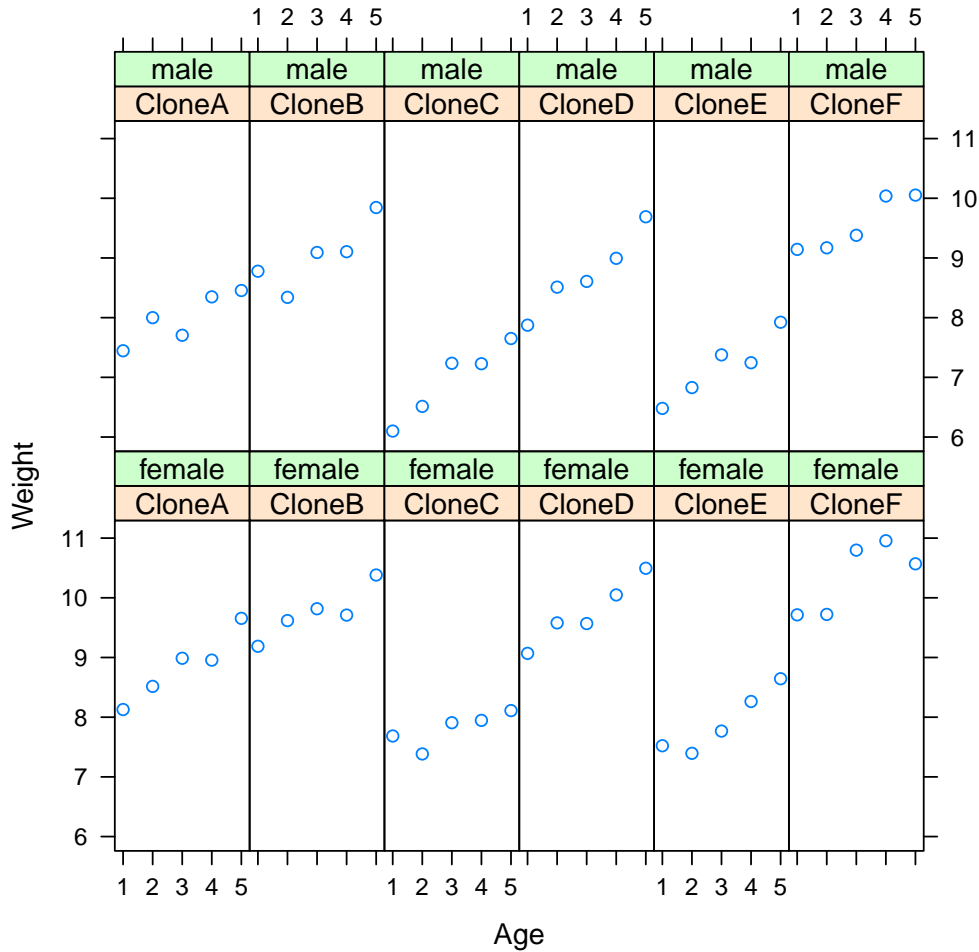


A more complex example

We read in a more complex dataset and finally we use lattice for what it is good instead of learning how to build graphics from first principles in base.

```
> gain <- read.table("Gain.txt", header = TRUE)
> attach(gain)
> library(lattice)
> pl <- xyplot(Weight ~ Age | Genotype + Sex)

> print(pl)
```



After a first already allows some conclusions about differences in intercepts and slopes and we start modeling with this in mind.

```
> m1 <- lm(Weight ~ Sex * Age * Genotype)
> summary(m1)
```

Call:

```
lm(formula = Weight ~ Sex * Age * Genotype)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.40218	-0.12043	-0.01065	0.12592	0.44687

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	7.80053	0.24941	31.276	< 2e-16	***
Sexmale	-0.51966	0.35272	-1.473	0.14936	
Age	0.34950	0.07520	4.648	4.39e-05	***
GenotypeCloneB	1.19870	0.35272	3.398	0.00167	**
GenotypeCloneC	-0.41751	0.35272	-1.184	0.24429	
GenotypeCloneD	0.95600	0.35272	2.710	0.01023	*
GenotypeCloneE	-0.81604	0.35272	-2.314	0.02651	*
GenotypeCloneF	1.66851	0.35272	4.730	3.41e-05	***
Sexmale:Age	-0.11283	0.10635	-1.061	0.29579	
Sexmale:GenotypeCloneB	-0.31716	0.49882	-0.636	0.52891	
Sexmale:GenotypeCloneC	-1.06234	0.49882	-2.130	0.04010	*
Sexmale:GenotypeCloneD	-0.73547	0.49882	-1.474	0.14906	
Sexmale:GenotypeCloneE	-0.28533	0.49882	-0.572	0.57087	
Sexmale:GenotypeCloneF	-0.19839	0.49882	-0.398	0.69319	
Age:GenotypeCloneB	-0.10146	0.10635	-0.954	0.34643	
Age:GenotypeCloneC	-0.20825	0.10635	-1.958	0.05799	.
Age:GenotypeCloneD	-0.01757	0.10635	-0.165	0.86970	
Age:GenotypeCloneE	-0.03825	0.10635	-0.360	0.72123	
Age:GenotypeCloneF	-0.05512	0.10635	-0.518	0.60743	
Sexmale:Age:GenotypeCloneB	0.15469	0.15040	1.029	0.31055	
Sexmale:Age:GenotypeCloneC	0.35322	0.15040	2.349	0.02446	*
Sexmale:Age:GenotypeCloneD	0.19227	0.15040	1.278	0.20929	
Sexmale:Age:GenotypeCloneE	0.13203	0.15040	0.878	0.38585	
Sexmale:Age:GenotypeCloneF	0.08709	0.15040	0.579	0.56616	

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

Residual standard error: 0.2378 on 36 degrees of freedom

Multiple R-squared: 0.9742, Adjusted R-squared: 0.9577

F-statistic: 59.06 on 23 and 36 DF, p-value: < 2.2e-16

> summary.aov(m1)

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Sex	1	10.374	10.3735	183.4398	1.065e-15 ***
Age	1	10.770	10.7700	190.4503	6.028e-16 ***
Genotype	5	54.958	10.9917	194.3710	< 2.2e-16 ***
Sex:Age	1	0.049	0.0489	0.8654	0.3584
Sex:Genotype	5	0.147	0.0294	0.5195	0.7598
Age:Genotype	5	0.168	0.0336	0.5946	0.7041

```
Sex:Age:Genotype 5 0.349 0.0698 1.2347 0.3132
Residuals        36 2.036 0.0566
```

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

On the first few only the main effects seem significant all the interaction terms seem to cancel out. We start simplification and trust R's step function.

```
> m2 <- step(m1)
```

```
Start:  AIC=-155.01
```

```
Weight ~ Sex * Age * Genotype
```

	Df	Sum of Sq	RSS	AIC
- Sex:Age:Genotype	5	0.34912	2.3849	-155.51
<none>			2.0358	-155.01

```
Step:  AIC=-155.51
```

```
Weight ~ Sex + Age + Genotype + Sex:Age + Sex:Genotype + Age:Genotype
```

	Df	Sum of Sq	RSS	AIC
- Sex:Genotype	5	0.146901	2.5318	-161.92
- Age:Genotype	5	0.168136	2.5531	-161.42
- Sex:Age	1	0.048937	2.4339	-156.29
<none>			2.3849	-155.51

```
Step:  AIC=-161.92
```

```
Weight ~ Sex + Age + Genotype + Sex:Age + Age:Genotype
```

	Df	Sum of Sq	RSS	AIC
- Age:Genotype	5	0.168136	2.7000	-168.07
- Sex:Age	1	0.048937	2.5808	-162.78
<none>			2.5318	-161.92

```
Step:  AIC=-168.07
```

```
Weight ~ Sex + Age + Genotype + Sex:Age
```

	Df	Sum of Sq	RSS	AIC
- Sex:Age	1	0.049	2.749	-168.989
<none>			2.700	-168.066
- Genotype	5	54.958	57.658	5.612

Step: AIC=-168.99
 Weight ~ Sex + Age + Genotype

	Df	Sum of Sq	RSS	AIC
<none>			2.749	-168.989
- Sex	1	10.374	13.122	-77.201
- Age	1	10.770	13.519	-75.415
- Genotype	5	54.958	57.707	3.662

> summary(m2)

Call:
 lm(formula = Weight ~ Sex + Age + Genotype)

Residuals:

Min	1Q	Median	3Q	Max
-0.40005	-0.15120	-0.01668	0.16953	0.49227

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	7.93701	0.10066	78.851	< 2e-16 ***
Sexmale	-0.83161	0.05937	-14.008	< 2e-16 ***
Age	0.29958	0.02099	14.273	< 2e-16 ***
GenotypeCloneB	0.96778	0.10282	9.412	8.07e-13 ***
GenotypeCloneC	-1.04361	0.10282	-10.149	6.21e-14 ***
GenotypeCloneD	0.82396	0.10282	8.013	1.21e-10 ***
GenotypeCloneE	-0.87540	0.10282	-8.514	1.98e-11 ***
GenotypeCloneF	1.53460	0.10282	14.925	< 2e-16 ***

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

Residual standard error: 0.2299 on 52 degrees of freedom
 Multiple R-squared: 0.9651, Adjusted R-squared: 0.9604
 F-statistic: 205.7 on 7 and 52 DF, p-value: < 2.2e-16

> summary.aov(m2)

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Sex	1	10.374	10.3735	196.23	< 2.2e-16 ***
Age	1	10.770	10.7700	203.73	< 2.2e-16 ***
Genotype	5	54.958	10.9917	207.93	< 2.2e-16 ***

```
Residuals    52  2.749  0.0529
```

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

We arrived at a model only containing main effects. It means that the slopes do not vary, but the intercepts do. This is reasonable if you remember our plot.

But what about factor levels? Don't some of the levels have the same intercept? We will come back to this when we talk about contrasts. For the moment we just look at the output of `summary.lm`:

Genotype B and D seem to have a comparable intercept, so do C and E. We create a factor with 3 new levels accordingly.

I prefer nested ifelse statement over logic arithmetic.

```
> ngen <- factor(ifelse(Genotype == "CloneB" | Genotype == "CloneD",
+   "BD", ifelse(Genotype == "CloneC" | Genotype == "CloneE",
+   "CE", ifelse(Genotype == "CloneA", "A", "F"))))
> ngen
```

```
[1] A  A  A  A  A  BD BD BD BD BD CE CE CE CE CE BD BD BD BD BD CE CE CE CE CE
[26] F  F  F  F  F  A  A  A  A  A  BD BD BD BD BD CE CE CE CE CE BD BD BD BD BD
[51] CE CE CE CE CE F  F  F  F  F
Levels: A BD CE F
```

We use this in a third model.

```
> m3 <- lm(Weight ~ Sex + Age + ngen)
> summary(m3)
```

Call:

```
lm(formula = Weight ~ Sex + Age + ngen)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-0.42651	-0.16687	0.01211	0.18776	0.47736

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	7.93701	0.10308	76.996	< 2e-16 ***
Sexmale	-0.83161	0.06080	-13.679	< 2e-16 ***
Age	0.29958	0.02149	13.938	< 2e-16 ***
ngenBD	0.89587	0.09119	9.824	1.28e-13 ***

```
ngenCE      -0.95950    0.09119 -10.522 1.10e-14 ***
ngenF       1.53460    0.10530  14.574 < 2e-16 ***
```

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

Residual standard error: 0.2355 on 54 degrees of freedom
Multiple R-squared: 0.962, Adjusted R-squared: 0.9585
F-statistic: 273.7 on 5 and 54 DF, p-value: < 2.2e-16

And compare it to model two.

```
> anova(m2, m3)
```

Analysis of Variance Table

Model 1: Weight ~ Sex + Age + Genotype

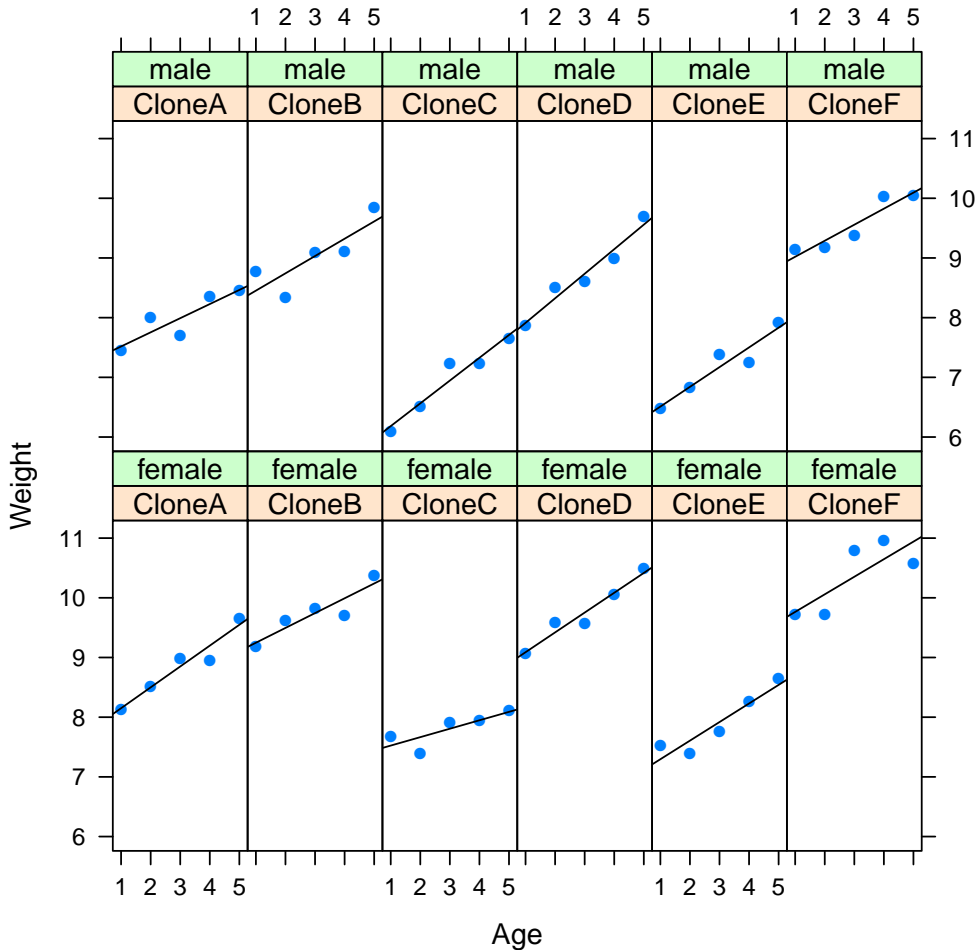
Model 2: Weight ~ Sex + Age + ngen

	Res.Df	RSS	Df	Sum of Sq	F	Pr(>F)
1	52	2.7489				
2	54	2.9938	-2	-0.24489	2.3163	0.1087

Model 2 is not significantly better than model 1 and we prefer parsimony.
Finally we can plot the model:

```
> pl1 <- update(pl, panel = function(x, y) {
+   panel.xyplot(x, y, pch = 16)
+   panel.abline(lm(y ~ x))
+ })

> print(pl1)
```



produces single linear models all with their own slope and intercept. **We want to plot our minimal adequate model!**

First we need to get the “raw value for the intercepts”. Remember, how R presents them as differences from the first level! Extract them in the order needed in lattice. So here the procedure for model 2.

```

> interc <- c(m2$coefficients[[1]], sapply(4:8, function(i) m2$coefficients[[1]]
+   m2$coefficients[[i]], m2$coefficients[[1]] + m2$coefficients[["Sexmale"]]
+   sapply(4:8, function(i) m2$coefficients[[1]] + m2$coefficients[[i]] +
+     m2$coefficients[["Sexmale"]]))
> pl3 <- update(pl, panel = function(x, y, ...) {
+   panel.xyplot(x, y, pch = 16)
+   for (i in 1:12) {
+     if (panel.number() == i) {

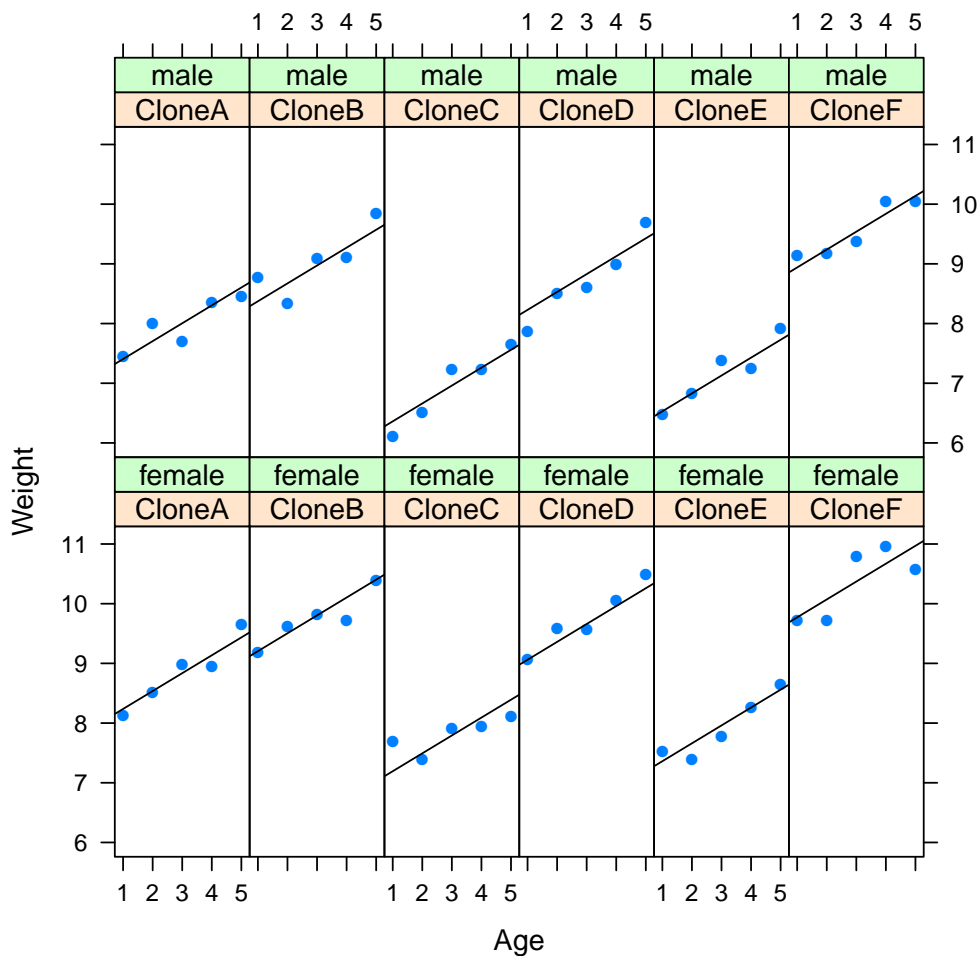
```

```

+           panel.abline(a = interc[i], b = m2$coefficients[["Age"]])
+       }
+   }
+ })

> print(pl3)

```



For model 3 with the reduced factor levels a new plot is needed

```

> pl4 <- xyplot(Weight ~ Age | ngen + Sex)
> interc2 <- c(m3$coefficients[[1]], sapply(4:6, function(i) m3$coefficients[[1]]
+ m3$coefficients[[i]], m3$coefficients[[1]] + m3$coefficients[["Sexmale"]])
+ sapply(4:6, function(i) m3$coefficients[[1]] + m3$coefficients[[i]] +
+ m3$coefficients[["Sexmale"]]))

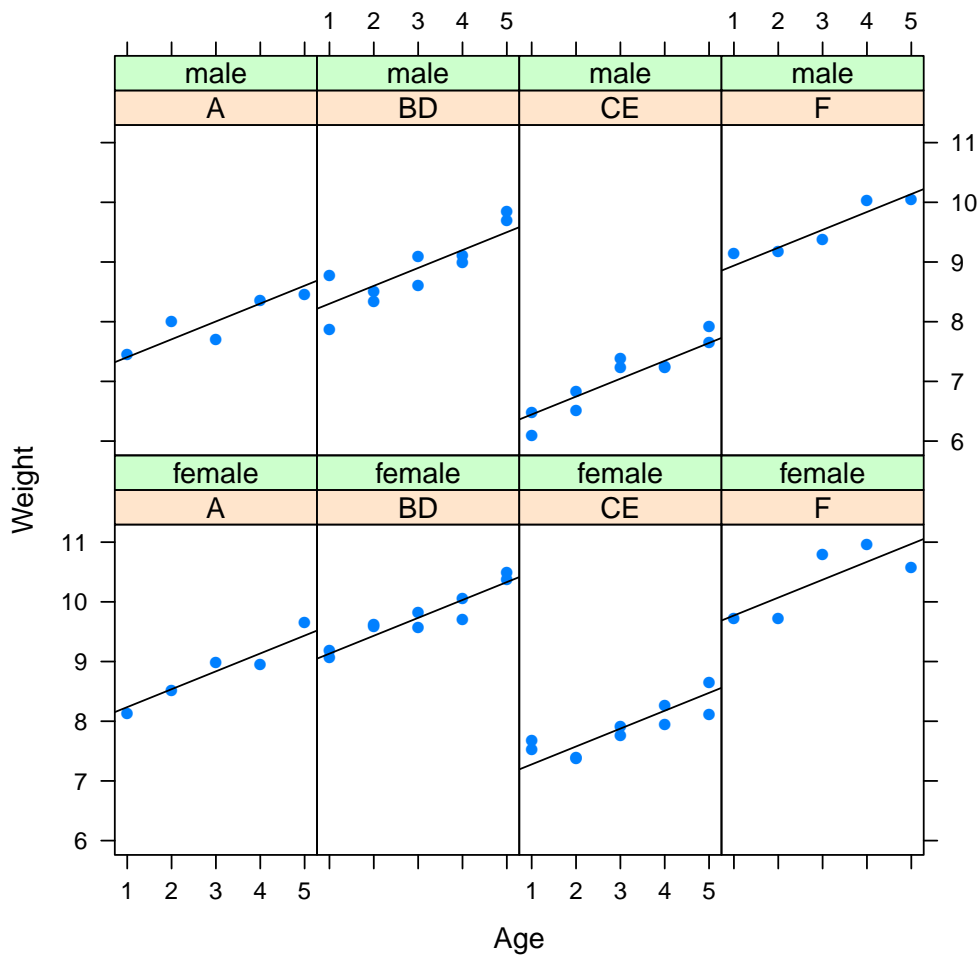
```

```

> pl4 <- update(pl4, panel = function(x, y, ...) {
+   panel.xyplot(x, y, pch = 16)
+   for (i in 1:8) {
+     if (panel.number() == i) {
+       panel.abline(a = interc2[i], b = m3$coefficients[["Age"]])
+     }
+   }
+ })

> print(pl4)

```



The result looks good. It was quite hard to achieve, even with lattice. Or have I just used a stupid approach?