Summary Applied Analysis of Variance and Experimental Design

Learning from Data

We are in the (abstract) situation where we have a "system" or a "process" with many input variables (predictors) and an output (response). The predictor could be a fertilizer and the output the biomass of a plant. We would like to find a cause-effect relationships. With observational data, we can typically just make a statement about an association between two variables. A potential danger is the existence of **confounding variables / confounders**. A confounder is a common cause for two variables (e.g. turbulent weather cause the seatbelt sign to switch on and the plane to shake, but the seatbelt sign is not a cause of the shaking plane).
In an observational study, we only observe subjects / objects in an existing (uncontrolled) situation. There are cross-sectional studies (e.g. a "snapshot" of the population at a given time-point), cohort studies which are prospective (what will happen if...?, e.g. determining the risk of exposed vs. non-exposed smokers) and case-control studies which are retrospective (why did it develop this way?, e.g. comparison of habits).
In an experimental study, we can observe the subjects / objects in a controlled setting.
Before designing an experimental study, we must have a focused and precise research question that we want to answer with experimental data. The study then consists of:

- The different interventions which we perform on the system (the different treatments)
- Experimental units: The "things" to which we apply the treatments
- A method that assigns treatments to experimental units (typically randomization)
- Response(s): The output that we measure

We distinguish between the following types of predictors:

- Predictors that are of primary interest and that can (ideally) be varied according to our "wishes"
- Predictors that are systematically recorded such that potential effects can later be eliminated in our calculations ("controlling for...")
- Predictors that can be kept constant and whose effects can therefore be eliminated (e.g. using always the same measurement device)
- Predictors that we can neither record nor keep constant (e.g. some special soil properties)

Randomization (the random allocation of experimental units to the different treatments) ensures that the only systematic difference between the different treatment "groups" is the treatment and protects from confounders. Typically, the order (time), locations and instruments should also be randomized. When we already know that some experimental units are more alike than others before doing the experiment, we do a randomization "within" homogeneous blocks which is called **blocking**. Blocking (typically) increases precision of an experiment.
An **experimental unit** is defined as the "thing" to which we apply the treatments by randomization and "should be able to receive any treatment independently of the other units". A measurement unit is the unit on which the response is being measured. They don't need to be the same. E.g., if we randomize different food supplies to cages of animals, the experimental unit is the cage. However, the measurement unit will be given by an individual animal of the cage. Typically, values of measurement units are aggregated such that we get one value per experimental unit. The experimental units should ideally be a random sample from the population of interest.
The response should be chosen such that it reflects useful information about the process under study. If not directly measurable, a **surrogate response** is used (e.g. a cell count for disease progression).
Different experimental units will always give different responses to the same treatment. We should design our experiment such that we get an idea of this so-called **experimental error**, e.g. with multiple experimental units receiving the same treatment. When we do multiple measurements on the same experimental unit, we call them **pseudoreplicates** (which causes statistical tests on the data to be invalid).
Blinding means that the evaluators don't know which treatment is given to which experimental unit. With **double-blinding**, neither the evaluators nor the patients know the assignment (this protects us from bias due to expectations).
A **control treatment** is a standard treatment used as a baseline. A **placebo** is a "null treatment" for situations where the act of applying a treatment potentially has an effect.
Categorical predictors are also called **factors**. We distinguish between unordered (or nominal; e.g. the eye color) and ordered (or ordinal; e.g. an income class) factors.

Completely Randomized Designs

One-Way Analysis of Variance

The goal is to compare $g \geq 2$ treatments and we have N experimental units that are assigned randomly to the g different treatment groups having n_i observations each. Let Y_{ij} be the jth observation in treatment group i (where $i = 1, \ldots, g$ and $j = 1, \ldots, n_i$). In the cell means model, each treatment is allowed to have its own expected value and we assume:

$$Y_{ij} \sim N(\mu_i, \sigma^2), \text{ independent}$$

(where μ_i is the expected value of treatment group i and σ^2 the variance which is equal for all groups). This can be rewritten as:

$$Y_{ij} = \mu_i + \epsilon_{ij} \ , \ \mu_i = \mu + \alpha_i \ , \ Y_{ij} = \mu + \alpha_i + \epsilon_{ij}$$

Where ϵ_{ij} i.i.d. $\sim N(0, \sigma^2)$. α_i is also called the ith treatment effect. This model isn't identifiable anymore because there are $g + 1$ parameters to model g mean values μ_i. Only $g - 1$ elements of the treatment effects are allowed to vary freely, it has $g - 1$ degrees of freedom (df). Therefore, side constraints are introduced, which are set in R with:

```
options(contrasts = c("contr.sum", "contr.poly"))
```

- **Name:** weighted sum-to-zero, **Side-Constraint:** $\sum_{i=1}^{g} n_i \alpha_i = 0$, **Interpretation of μ:** $\mu = \frac{1}{N} \sum_{i=1}^{g} n_i \mu_i$, **R:** -
- **Name:** sum-to-zero, **Side-Constraint:** $\sum_{i=1}^{g} \alpha_i = 0$, **Interpretation of μ:** $\mu = \frac{1}{g} \sum_{i=1}^{g} \mu_i$, **R:** *contr.sum*
- **Name:** reference group, **Side-Constraint:** $\alpha_1 = 0$, **Interpretation of μ:** $\mu = \mu_1$, **R:** *contr.treatment* where the first level in the output of *levels* is the reference group, can be changed with *relevel*, e.g.:

  ```
  d$col <- relevel(d$col, ref = "M")
  ```

The model is fitted using the least squares criterion, i.e.:

$$\widehat{\mu}, \widehat{\alpha}_i = \text{argmin}_{\mu,\, \alpha_i} \sum_{i=1}^{g} \sum_{j=1}^{n_i} (y_{ij} - \mu - \alpha_i)^2$$

$$\widehat{\mu}_i = \text{argmin}_{\mu_i} \sum_{i=1}^{g} \sum_{j=1}^{n_i} (y_{ij} - \mu_i)^2$$

Which gives us $\widehat{\mu}_i = \widehat{\mu} + \widehat{\alpha}_i = \overline{y}_{i\cdot}$. The error variance is estimated by the mean squared error:

$$\widehat{\sigma}^2 = MS_E = \frac{1}{N - g} SS_E \quad SS_E = \sum_{i=1}^{g} \sum_{j=1}^{n_i} (y_{ij} - \widehat{\mu}_i)^2$$

Or:

$$MS_E = \frac{1}{N - g} \sum_{i=1}^{g} (n_i - 1) s_i^2 \quad s_i^2 = \frac{1}{n_i - 1} \sum_{j=1}^{n_i} (y_{ij} - \widehat{\mu}_i)^2$$

Where s_i^2 is the empirical variance in treatment group i. The demoninator $N - g$ ensures that $\widehat{\sigma}^2$ is an unbiased estimator (the error estimate has $N - g$ degrees of freedom).
In R,

```
fit <- aov(response ~ predictor, data = data)
```

is used for fitting the model. The coefficients can be viewed with *coef(fit)*, e.g.:

```
## (Intercept)   grouptrt1   grouptrt2
##       5.032      -0.371       0.494
```

(Intercept) contains $\widehat{\mu} = 5.032$ (interpretation dependent on side constraints, without constraints expected value of control group). *grouptrt1* is $\widehat{\alpha}_2 = -0.371$ and *grouptrt2* $\widehat{\alpha}_3 = 0.494$. The function *dummy.coef(fit)* shows all coefficients, e.g.:

```
## Full coefficients are
##
## (Intercept):     5.032
## group:            ctrl   trt1   trt2
##                  0.000 -0.371  0.494
```

We can get the estimated cell means $\widehat{\mu}_i$ with *predict(fit, newdata = data.frame(group = c("ctrl", "trt1", "trt2")))*:

```
##     1     2     3
## 5.032 4.661 5.526
```

The output with *contr.sum* looks as follows:

```
options(contrasts = c("contr.sum", "contr.poly"))
fit2 <- aov(weight ~ group, data = PlantGrowth)
coef(fit2)
```

```
## (Intercept)      group1      group2
##       5.073      -0.041      -0.412
```

Now, *(Intercept)* is the global mean, *group1* the difference of the first (control) group and *group2* the difference of the second group. With *dummy.coef*, the full picture can be retrieved again:

```
## Full coefficients are
##
## (Intercept):     5.073
## group:            ctrl   trt1   trt2
##                 -0.041 -0.412  0.453
```

Tests

Our null hypothesis is that all groups share the same mean, i.e.:

$$Y_{ij} = \mu + \epsilon_{ij},\ \epsilon_{ij} \text{ i.i.d. } \sim N(0, \sigma^2).$$
$$H_0 : \mu_1 = \ldots = \mu_g$$

This is the single mean model and is a special case of the cell means model with $\alpha_1 = \ldots = \alpha_g = 0$. The alternative is therefore:

$$H_A : \mu_k \neq \mu_l \text{ for at least one pair } k \neq l.$$

The total variation of the response around the overall mean can be decomposed into variation "between groups" and variation "within groups":

$$\underbrace{\sum_{i=1}^{g}\sum_{j=1}^{n_i}(y_{ij} - \overline{y}_{..})^2}_{SS_T} = \underbrace{\sum_{i=1}^{g}\sum_{j=1}^{n_i}(\overline{y}_{i.} - \overline{y}_{..})^2}_{SS_{\text{Trt}}} + \underbrace{\sum_{i=1}^{g}\sum_{j=1}^{n_i}(y_{ij} - \overline{y}_{i.})^2}_{SS_E},$$

Where SS_T is the total sum of squares, SS_{Trt} the treatment sum of squares (between groups) and SS_E the error sum of squares (within groups). SS_{Trt} can also be interpreted as the reduction in residual sum of squares when comparing the cell means with the single mean model. This information can be summarized in the ANOVA table (see Appendix). If all the groups share the same (theoretical) mean, we expect the treatment sum of squares to be small. The idea is now to compare the variation between groups with the variation within groups. Under H_0, it holds that:

$$F = \frac{MS_{\text{Trt}}}{MS_E} \sim F_{g-1,\, N-g}$$

Where:

$$MS_{\text{Trt}} = \frac{SS_{Trt}}{g-1}$$

Under H_0, MS_{Trt} is also an estimator for σ^2 and therefore $F = \frac{MS_{Trt}}{MS_E} \approx 1$.
We reject the null hypothesis if the observed F value lies in a "extreme" region of the corresponding distribution, more precise we reject H_0 in favor of H_A if F is larger than the 95% quantile. The F-test is a omnibus test because it compares all group means simultaneously. Increasing the denominator degrees of freedom will decrease the corresponding quantile (which gives more power). In R, *summary(fit)* gives the ANOVA table and p-value. As the global test can also be interpreted as a test for comparing two different models, namely the cell means and the single means model, there's also another approach. *anova* can be used to compare the two models:

```
## Fit single mean model (1 means global mean):
fit.single <- aov(weight ~ 1, data = PlantGrowth)
## Compare with cell means model:
anova(fit.single, fit)
```

To perform statistical inference for the individual α_i's, *summary.lm(fit)* (for tests; retrieves all the parameters including standard errors) and *confint(fit)* (for confidence intervals) can be used. Interpretation depends on the side-constraint, an example output of *summary.lm(fit)* looks like this:

```
Coefficients:
                    Estimate Std. Error t value Pr(>|t|)
(Intercept)           3.3600     0.1965   17.10 1.39e-07
treatmentCommercial   4.1200     0.2779   14.82 4.22e-07
```

Checking Model Assumptions

Statistical inference (p-values, confidence intervals, ...) is only valid if the model assumptions are fulfilled. So far, this means

- are the errors independent?
- are the errors normally distributed?
- is the error variance constant?
- do the errors have mean zero?

The errors ϵ_{ij} can't be observed, but the residuals r_{ij} can be used as estimates:

$$r_{ij} = y_{ij} - \widehat{\mu}_i.$$

In a QQ-plot, the empirical quantiles are plotted against the theoretical quantiles (of a standard normal distribution). We should more or less see a straight line if the distribution assumption is correct. This is done in R with *plot(fit, which = 2)*. If the QQ-plot suggests non-normality, we can try to use a transformation of the response to accommodate this problem.

The Tukey-Anscombe plot plots the residuals r_{ij} vs. the fitted values $\widehat{\mu}_i$. It allows us to check whether the residuals have constant variance and whether the residuals have mean zero. For the one-way ANOVA situation we could also read off the same information from the plot of the data itself and the residuals always have mean zero (per group). In R, the plot is generated by *plot(fit, which = 1)*.

Transformations Affect Interpretation

Whenever we transform the response we implicitly also change the interpretation of the model parameters. Therefore, while it is conceptually attractive to model the problem on an appropriate scale of the response, this typically has the side effect of making interpretation (much) more difficult. For example, if we use the logarithm,

$$\log(Y_{ij}) = \mu + \alpha_i + \epsilon_{ij}$$

all the α_i's (and their estimates) have to be interpreted on the log-scale. For example, if we us *contr.treatment* and we have $\widehat{\alpha}_2 = 1.5$. This means: on the log-scale we estimate that the average value of group 2 is 1.5 larger than the average value of group 1 (additive shift). What about the original scale? We know that $\mathbb{E}[\log(Y_{ij})] = \mu + \alpha_i$, but the expected value on the original scale does (in general) not directly follow the transformation, i.e. $\mathbb{E}[Y_{ij}] \neq e^{\mu+\alpha_i}$. However, we can make a statement about the median. On the log-scale the median is equal to the mean (because we have a symmetric distribution around $\mu + \alpha_i$) Hence,

$$\text{median}(\log(Y_{ij})) = \mu + \alpha_i$$

In contrast to the mean, any quantile directly transforms with a strictly monotone increasing function. As the median is nothing else than the 50%-quantile, we have

$$\text{median}(Y_{ij}) = e^{\mu+\alpha_i}$$

Similarly, for the ratio

$$\frac{\text{median}(Y_{2j})}{\text{median}(Y_{1j})} = \frac{e^{\mu+\alpha_2}}{e^{\mu}} = e^{\alpha_2}$$

Hence, we can make a statement that on the original scale the median of group 2 is $e^{\widehat{\alpha}_2} = e^{1.5} = 4.48$ as large as the median of group 1. This means that additive effects on the log-scale become multiplicative effects on the original scale. Unfortunately, the statement is only about the median and not the mean on the original scale.

If we also consider a confidence intervals for α_2, e.g. $[1.2, 1.8]$, the transformed version $[e^{1.2}, e^{1.8}]$ is a confidence interval for e^{α_2} which is the ratio of medians on the original scale.

Contrasts and Multiple Testing

Contrasts

The F-test is rather unspecific. It basically gives us a "Yes/No" answer for the question "is there any treatment effect at all?". It doesn't tell us what specific treatment (or treatment combination) is significant. Such kinds of questions can typically be formulated as a so-called contrast. For instance, we could have the null hypothesis / alternative:

$$H_0 : \mu_1 - \mu_2 = 0$$
$$H_A : \mu_1 - \mu_2 \neq 0.$$

This can be encoded with a vector $c \in \mathbb{R}^g$:

$$H_0 : \sum_{i=1}^{g} c_i \mu_i = 0.$$

Typically, the side constraint $\sum_{i=1}^{g} c_i = 0$ is enforced which ensures that the contrast is about differences between treatments and not about the overall level of our response. A contrasts true (but unknown) value $\sum_{i=1}^{g} c_i \mu_i$ is estimated with:

$$\sum_{i=1}^{g} c_i \widehat{\mu}_i$$

In R, *glht* of the package *multcomp* is used (on a normal ANOVA fit):

```
library(multcomp)
fit.gh <- glht(fit, linfct =
      mcp(group = c(1, -1/2, -1/2)))
summary(fit.gh)
```

This gives us a p-value if the contrast(s) is (are) zero or not. *confint* gives a confidence interval for the contrast(s). To test many contrasts at the same time:

```
M <- rbind(new.vs.old   = c(1/2, -1/2, 1/2, -1/2),
           co2.vs.mixed = c(1, 0, -1, 0))
fit.mc <- glht(fit, linfct = mcp(treatment = M))
summary(fit.mc, test = adjusted("none"))
```

Every contrast has an associated sum of squares:

$$SS_c = \frac{\left(\sum_{i=1}^{g} c_i \overline{y}_{i\cdot}\right)^2}{\sum_{i=1}^{g} \frac{c_i^2}{n_i}}$$

With one degree of freedom (therefore $MS_c = SS_c$). This is the square of the t-statistic of the corresponding null hypothesis for the model parameter $\sum_{i=1}^{g} c_i \mu_i$ (without the MS_E). We have:

$$\frac{MS_c}{MS_E} \sim F_{1,\, N-g}$$

Two contrasts c and c^* are orthogonal if

$$\sum_{i=1}^{g} \frac{c_i c_i^*}{n_i} = 0.$$

In this case, the corresponding estimates are (statistically) independent. This means that if we know something about one of the contrasts, this does not help us in making a statement about the other one. If we have g treatments, we can find $g-1$ different orthogonal contrasts (one dimension is already used by the global mean). A set of orthogonal contrasts partitions the treatment sum of squares meaning that if $c^{(1)}, \ldots, c^{(g-1)}$ are orthogonal contrasts it holds that:

$$SS_{c^{(1)}} + \cdots + SS_{c^{(g-1)}} = SS_{\text{Trt}}$$

Multiple contrasts are all orthogonal if and only if for the matrix C that represents them, $C^T C$ is diagonal.

Multiple Testing

If we simply want to do all pairwise t-tests using pooled standard deviation (without adjustment for multiple testing), we can use:

```
with(d, pairwise.t.test(y, tr, p.adjust.method = "none"))
```

The (overall) error rate increases with increasing number of tests. If we perform m (independent) tests using an individual significance level α, the probability of making at least one false rejection (if all $H_{0,j}$ are true) is:

$$1 - (1-\alpha)^m.$$

If we perform m tests, whereof m_0 null hypotheses are true, we have the following potential outcomes:

	H_0 **true**	H_0 **false**	**Total**
Significant	V	S	R
Not significant	U	T	$m-R$
Total	m_0	$m-m_0$	m

The family-wise error rate is defined as the probability of rejecting at least one of the true H_0's:

$$\text{FWER} = P(V \geq 1)$$

We say that a procedure controls the family-wise error rate in the strong sense at level α if FWER $\leq \alpha$ for any configuration of true and non-true null hypotheses.
The false discovery rate (FDR) is the expected fraction of false discoveries:

$$\text{FDR} = E\left[\frac{V}{R}\right].$$

Controlling FDR at level 0.2 means that in our list of "significant findings" we expect only 20% that are not "true findings" (so called false positives). If a procedure controls FWER at level α, FDR is automatically controlled at level α too. On the other side, a procedure that controls FDR at level α might have a much larger error rate regarding FWER.
We call a set of confidence intervals simultaneous confidence intervals at level $(1-\alpha)$ if the probability that all intervals cover the corresponding true parameter value is $(1-\alpha)$.
Bonferroni: In this approach, we use a more restrictive (individual) significance level of $\alpha^* = \alpha/m$. It controls the family-wise error rate in the strong sense. Equivalently, we can also multiply the "original" p-values by m and keep using the original α. The confidence intervals based on the adjusted significance level are simultaneous (we use level $1-\alpha/m$ instead of $1-\alpha$). In R, we do the adjustment by

```
summary(fit.gh, test = adjusted("bonferroni"))
```

Bonferroni-Holm: Bonferroni-Holm is less conservative and uniformly more powerful than Bonferroni. It works as follows:

1. Sort p-values from small to large: $p_{(1)} \leq p_{(2)} \leq \cdots \leq p_{(m)}$
2. For $j = 1, 2, \ldots$: Reject null hypothesis if $p_{(j)} \leq \frac{\alpha}{m-j+1}$
3. Stop when reaching the first non-signifcant p-value.

In R, we use

```
summary(fit.gh, test = adjusted("holm"))
```

Scheffé: The Scheffé procedure controls for the search over any possible contrast. This means we can try out as many contrasts as we like and still get honest p-values. The price for this very nice property is low power. It works as follows: Calculate F-ratio as if ordinary contrast and use the distribution $(g-1) \cdot F_{g-1,\, N-g}$ instead of $F_{1,\, N-g}$ to calculate p-values / critical values. In R, this has to be done manually:

```
fit.sch <- glht(fit,
linfct = mcp(group = c(1/2, -1, 1/2)))

## p-value according to Scheffe (g = 3, N - g = 27)
pf((summary(fit.sch)$test$tstat)^2 / 2, 2, 27,
    lower.tail = FALSE)
```

Tukey Honest Significant Difference (HSD) A special case for a multiple testing problem is the comparison between all possible pairs of treatments. There are a total of $g*(g-1)/2$ pairs. We could perform all pairwise t-tests with the function *pairwise.t.test* (that uses a pooled standard deviation estimate from all groups). There is a better (more powerful) alternative which is called Tukey Honest Significant Difference. Think of a procedure that is custom tailored for this situation. It gives us both p-values and confidence intervals. In R, this is done using

```
TukeyHSD(fit)
```

Multiple Comparison with a Control (MCC) In the same spirit, if we want to compare all treatment groups with a control group, we have a so called multiple comparisons with a control problem. The corresponding procedure is called Dunnett procedure which is implemented in the add-on package multcomp.

```
fit.dunnett <- glht(fit, linfct = mcp(treatment = "Dunnett"))
```

Relationship to the F-Test Pairwise comparisons etc. can also be done if the omnibus F-test isn't significant, as they have built-in multiple-testing correction and conditioning on a significant F-test makes them over-conservative.

Factorial Treatment Structure

In practice, treatments are often combinations of the level of two or more factors which is called factorial treatment structure. If we see all possible combinations of the levels of two (or more) factors, we call them crossed. In R, we can count the number of observations for every combination of the levels with *xtabs(∼ factor1 + factor2, data = data)*. In a factorial treatment structure, there are typically questions about both factors and / or their possible interplay (we could also use the cell means / one-way ANOVA model for analysis, but then we would ignore the special structure and answering these questions using contrasts is complicated).
Such kind of data can be visualized with an interaction plot (R: *interaction.plot(x.factor = factor1, trace.factor = factor2, response = response)*). For every combination, the average response is calculated. The *x.factor* is plotted on the x-axis, and settings corresponding to the same level of *trace.factor* are connected with a line. Parallel lines indicate no interplay.

Two-Way ANOVA model

We assume a general setup with a factor A with a levels, a factor B with b levels and n replicates for every combination of A and B (a balanced design). Hence, we have a total of $N = a \cdot b \cdot n$ observations. The two-way ANOVA model with interaction is

$$Y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \epsilon_{ijk}$$

where α_i is the main effect of factor A at level i, β_j is the main effect of factor B at level j, $(\alpha\beta)_{ij}$ is the interaction effect between A and B for the level combination i, j (it is not the product $\alpha_i\beta_j$) and ϵ_{ijk} are i.i.d. $N(0, \sigma^2)$ errors.
Typically, sum-to-zero constraints are used, i.e. $\sum_{i=1}^{a} \alpha_i = 0$, $\sum_{j=1}^{b} \beta_j = 0$, $\sum_{i=1}^{a} (\alpha\beta)_{ij} = 0$ and $\sum_{j=1}^{b} (\alpha\beta)_{ij} = 0$. The main effects have $a - 1$ and $b - 1$ degrees of freedom, respectively and the degrees of freedom of the interaction term are $(a - 1)(b - 1)$. Parameters are estimated using least squares, which gives: $\widehat{\mu} = \overline{y}_{...}$, $\widehat{\alpha}_i = \overline{y}_{i..} - \overline{y}_{...}$, $\widehat{\beta}_j = \overline{y}_{.j.} - \overline{y}_{...}$ and $\widehat{(\alpha\beta)}_{ij} = \overline{y}_{ij.} - \widehat{\mu} - \widehat{\alpha}_i - \widehat{\beta}_j$. We estimate therefore the expected value of the response Y_{ijk} for A at level i and B at level j as

$$\widehat{\mu} + \widehat{\alpha}_i + \widehat{\beta}_j + \widehat{(\alpha\beta)}_{ij} = \overline{y}_{ij.}$$

which is simply the cell mean.
In R, we use

```
aov(response ~ factor1 * factor2, data = data)
```

which is equivalent to *response ∼ factor1 + factor2 + factor1:factor2*. *response ∼ factor1 + factor2* would fit a main effects model.

Tests

As for the one-way ANOVA case, the total sum of squares SS_T can be partitioned into different sources

$$SS_T = SS_A + SS_B + SS_{AB} + SS_E$$

where

- $SS_A = \sum_{i=1}^{a} bn(\widehat{\alpha}_i)^2$ ("between rows")
- $SS_B = \sum_{j=1}^{b} an(\widehat{\beta}_j)^2$ ("between columns")
- $SS_{AB} = \sum_{i=1}^{a} \sum_{j=1}^{b} n(\widehat{\alpha\beta})_{ij}^2$ ("correction")
- $SS_E = \sum_{i=1}^{a} \sum_{j=1}^{b} \sum_{k=1}^{n} (y_{ijk} - \overline{y}_{ij.})^2$ ("within cells")
- $SS_T = \sum_{i=1}^{a} \sum_{j=1}^{b} \sum_{k=1}^{n} (y_{ijk} - \overline{y}_{...})^2$ ("total")

The two-way ANOVA table is in the appendix. Tests can be constructed based on the corresponding F-distribution: For the interaction $H_0 : (\alpha\beta)_{ij} = 0$ for all i, j; H_A: at least one $(\alpha\beta)_{ij} \neq 0$; Under H_0:

$$\frac{MS_{AB}}{MS_E} \sim F_{(a-1)(b-1),\, ab(n-1)}$$

For the main effect A: $H_0 : \alpha_i = 0$ for all i; H_A: at least one $\alpha_i \neq 0$; Under H_0:

$$\frac{MS_A}{MS_E} \sim F_{a-1,\, ab(n-1)}$$

For the main effect B: $H_0 : \beta_j = 0$ for all j; H_A: at least one $\beta_j \neq 0$; Under H_0:

$$\frac{MS_B}{MS_E} \sim F_{b-1,\, ab(n-1)}$$

The ANOVA table is read bottom to top. First, it is checked if the interaction term is needed or not. If there is no evidence of interaction, we continue with the inspection of the main effects. If there is interaction, we would (typically) stop inspecting whether the main effects are significant or not, i.e. drop no terms from the model. With appropriate contrasts, some specific questions could be inspected. In R, this is often done with the *interaction* function which generates a hyper-factor that has as levels all possible combinations of the provided factors, e.g.:

```
d$exp.loc <- interaction(d$location, data$exposure)
```

Then, a normal ANOVA analysis with a contrast can be used where the hyper-factor is the factor.
If we continue with individual analyses after the two-way model, a significant interaction term means we should do the individual models (of one factor) per level (of the other factor). We can improve the tests by "re-using" the MS_E with the corresponding degrees of freedom of the full model.

Single Replicates

If we only have a single observation in each "cell" ($n = 1$) we cannot do statistical inference anymore with a model including the interaction as we have no idea of the experimental error (for every treatment combination we only have one observation). However, we can still fit a main effects only model. If the data generating mechanism actually contains an interaction, we are fitting a wrong model. The consequences are that the corresponding tests will be too conservative, meaning p-values will be too large. This is not a problem as the type 1 error rate is still controlled. We "just" lose power.

Quite often, we can get rid of interactions if we look at the problem on a different scale, i.e., if we transform the response appropriately. A famous example is the logarithm. Effects that are multiplicative on the original scale become additive on the log-scale, i.e., no interaction is needed on the log-scale.

Tukey One-Degree of Freedom Interaction A very special form of interaction where only one parameter λ is used (here, $\alpha_i\beta_j$ is the product of the main effects):

$$Y_{ij} = \mu + \alpha_i + \beta_j + \lambda\alpha_i\beta_j + \epsilon_{ij}$$

Unbalanced Data

With unbalanced data, the sum of squares cannot be uniquely partitioned into different sources anymore (for some part of the variation, it is not clear to what source we should attribute it). The problem is solved by using a model comparison approach. The sum of squares of a factor can be interpreted as the reduction of residual sum of squares when adding the factor to the model. But now (in contrast to the balanced case), it matters if the other factors are in the model.

With $SS(B|1, A)$, the reduction in residual sum of squares when comparing the model $(1, A, B)$ $(= y \sim A + B)$ with $(1, A)$ $(= y \sim A)$. The 1 is the overall mean μ. Interpretation of the corresponding test is as follows: "Do we need factor B in the model if we already have factor A (or after having controlled for factor A)?" There are different ways / types of model comparison approaches:

1. Type I (sequential): Sequentially build up model (depends on the "ordering" of the model terms!): $SS(A|1)$, $SS(B|1, A)$, $SS(AB|1, A, B)$
2. Type II (hierarchical): Control for the influence of the largest hierarchical model not including the term of interest: $SS(A|1, B)$, $SS(B|1, A)$, $SS(AB|1, A, B)$
3. Type III (fully adjusted): Control for all other terms: $SS(A|1, B, AB)$, $SS(B|1, A, AB)$, $SS(AB|1, A, B)$

summary of an *aov* gives type 1 (hence the order matters). For type II, the function *Anova* of *car* can be used, for example:

```
library(car)
Anova(fit, type = "II")
```

For type III, the command *drop1* (which reports deletion of single terms) can be used (but contrast option has to be *contr.sum* in this case). The following command can be used to get the type III sum of squares for all variables in the model:

```
drop1(fit, scope = ~., test = "F")
```

It can be shown that for type I and II the null hypothesis is weighted by the sample sizes wheres for type III it is $H_0 : \alpha_1 = \ldots = \alpha_a$ which means that the unweighted means are equal (which we generally want to test).

Block Designs

Quite often we already know that experimental units are not homogeneous. Using a completely randomized design in such a situation would still be a valid procedure. However, making explicit use of the special "structure" of the experimental units typically helps reducing variance. For instance, in a paired t-test we use blocking (e.g. on persons and therefore eliminate the person-to-person variation). This is extended to $g > 2$.

Randomized Complete Block Designs

Assume that we can divide our experimental units into r groups, also known as blocks, containing g experimental units each. The randomized complete block design (RCBD) uses a restricted randomization scheme: Within every block (e.g. location), the treatments are randomized to the experimental units (e.g. plots of land). The design is called complete because we see the complete set of treatments within every block. In the most basic form, we assume that we don't have replicates within a block (i.e. we only see every treatment once in every block). We therefore fit a main effects model:

$$Y_{ij} = \mu + \alpha_i + \beta_j + \epsilon_{ij}$$

Where the α_i's are the treatment effects and β_j's the block effects. According to this model we implicitly assume that blocks only cause additive shifts. Typically, we are not inspecting the p-value of the block factor. However, we can do a quick check to verify whether blocking was efficient or not. We would like the block factor to explain a lot of variation, hence if the mean squares of the block factor are much larger than the error mean square MS_E we would conclude that blocking was efficient.

Instead of a single treatment factor we can also have a factorial treatment structure within every block, e.g. a two-factor factorial which we would model as $Y \sim Block + A * B$. Here, we could actually test the interaction between A and B even if every level combination of A and B appears only once in every block. As we have multiple blocks, we have multiple observations for every level combination of A and B! However, a randomized complete block design can only be used with one blocking factor.

Multiple Block Factors

We can also block on more than one factor. A special case is the so-called Latin Square Design where we have two block factors and one treatment factor having g levels each (e.g. block factors R_1 to R_4 and C_1 to C_4 and a treatment factor with levels A to D). In a Latin Square Design each treatment (the Latin letters) appears exactly once in each row and once in each column. We also say it is a so-called row-column designs. An example is shown in the appendix. A Latin Square blocks on both rows and columns simultaneously. We can use the model

$$Y_{ijk} = \mu + \alpha_i + \beta_j + \gamma_k + \epsilon_{ijk}$$

to analyze data from a Latin square design. Here, α_i's are the treatment effects and β_j and γ_k are the block effects with the usual side-constraints. The design is balanced having the effect that our usual estimators and sums of squares are "working". In R we would use the model formula $Y \sim$ *Block1 + Block2 + Treat*.

Graeco-Latin Squares: If we have another blocking criterion with g levels (denoted by Greek letters, e.g. with levels α, β, γ, δ), we can use a Graeco-Latin Squares design. The conditions are that the Latin letters (treatments) occur once in each row and column and the Greek letters (third block factor) occur once in each row and column, i.e. we have two superimposed Latin Squares. In addition, each Latin letter occurs exactly once with each Greek letter. We use the main effects model to analyze the data:

$$Y_{ijkl} = \mu + \alpha_i + \beta_j + \gamma_k + \delta_l + \epsilon_{ijkl}$$

Where $alpha_i$ is the treatment, β_j the block factor 1 (rows), γ_k the block factor 2 (columns), δ_l the block factor 3 (Greek letters). An example of the design is in the appendix.

Precision

In a RCB design, the squared standard errors are $\frac{\sigma^2_{RCB}}{r}$ (where r is the number of blocks) and in a completely randomized design $\frac{\sigma^2_{CRD}}{n}$. If we want to have the same precision, we have to ensure that:

$$\frac{\sigma^2_{RCB}}{r} = \frac{\sigma^2_{CRD}}{n}$$

Therefore, if we knew both squared standard errors, we would have to use a ratio of $\frac{n}{r} = \frac{\sigma^2_{CRD}}{\sigma^2_{RCB}}$. σ^2_{RCB} is estimated by the MS_E of our RCB and σ^2_{CRD} can be estimated using a weighted average of MS_E and MS_{Block}. The relative efficiency is then defined as:

$$RE = \frac{\hat{\sigma}^2_{CRD}}{\hat{\sigma}^2_{RCB}}$$

And gives us the ratio $\frac{n}{r}$ (which is interpreted as how many experimental units would be needed by a CRD to achieve the same efficiency / precision). Easier for a quick check is to look at the ratio $\frac{MS_{Block}}{MS_E}$, because:

$$\frac{MS_{Block}}{MS_E} > 1 \Leftrightarrow \text{Relative Efficiency} > 1$$

Random and Mixed Effects Models

Up to now, treatment effects (the α_i's) were fixed, unknown quantitites that we tried to estimate. This means we were making a statement about a specific, fixed set of treatments.

Random Effects models

One-Way ANOVA

In this point of view, treatments are random samples from a large population of treatments. For example, a random sample of school classes that were drawn from all school classes in a country or machines that were randomly sampled from a large population of machines. Typically, we are interested in making a statement about some properties of the whole population. Such data can be modelled with

$$Y_{ij} = \mu + \alpha_i + \epsilon_{ij}$$

where α_i i.i.d. $\sim N(0, \sigma^2_\alpha)$, ϵ_{ij} i.i.d. $\sim N(0, \sigma^2)$. The α_i is also called a random effect. This introduces a new parameter σ^2_α which is the variance of the random effect. The expected value of Y_{ij} is $E[Y_{ij}] = \mu$ (but $E[Y_{ij} \,|\alpha_i] = \mu + \alpha_i$) . The variance is $Var(Y_{ij}) = \sigma^2_\alpha + \sigma^2$ and the correlation:

$$Cor(Y_{ij}, Y_{kl}) = \begin{cases} 0 & i \neq k \\ \sigma^2_\alpha/(\sigma^2_\alpha + \sigma^2) & i = k, j \neq l \\ 1 & i = k, j = l \end{cases}$$

Observations from a different "group" (e.g. from a different machine) are uncorrelated, while observations from the same "group" are correlated with an intraclass correlation (ICC) of $\sigma^2_\alpha/(\sigma^2_\alpha + \sigma^2)$. It is large if $\sigma^2_\alpha \gg \sigma^2$ which means that observations from the same "group" (e.g. machine) are very similar to each other. Therefore, values sharing the same α_i are correlated (while all are independent in the fixed effects model). The same holds for multiple random effects. For them, the correlation is the sum of shared variance components divided by the sum of all variance components. Parameter estimation for σ^2_α and σ^2 is typically done using restricted maximum likelihood estimation (REML). In R, the *lmer* function is used. A random effect is specified using *(1 | column)* which means that all variables sharing the same *column* (could also be an interaction) will get the same random effect α_i. An example call looks like this:

```
library(lmerTest)
fit.sire <- lmer(weight ~ (1 | sire), data = animals)
```

The output of *summary* contains:

```
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  sire     (Intercept) 116.7    10.81
##  Residual             463.8    21.54
```

Which means $\widehat{\sigma}^2_\alpha = 116.7$ and $\widehat{\sigma}^2 = 463.8$. With *confint(fit.sire, oldNames = FALSE)*, it's possible to get confidence intervals. The intervals are usually longer than if we would fit the model with the normal *aov* function because we are making inference about the whole population (whereof the fixed effects model makes inference about the specific ones we have seen / measured). Estimates (conditional means) for the random effects can be retrieved with *ranef*. These can then also be plotted in a QQ-plot to verify the normality assumption, e.g.:

```
## QQ-plots of random effects
qqnorm(ranef(fit.sire)$sire[,1], main = "sire")
## QQ-plots of residuals
qqnorm(resid(fit.sire), main = "residuals")
```

With *exactRLRT(fit.sire)* from the *RLRsim* library, we can do (simulation based) tests for variance components of random effects (i.e. if they are zero), which we are generally not that interested in (more in the confidence intervals).

Multiple factors

It's also possible to model multiple factors (with interaction) as random models, e.g.

$$Y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \epsilon_{ijk}$$

with

$$\alpha_i \text{ i.i.d.} \sim N(0, \sigma_\alpha^2),\ \beta_j \text{ i.i.d.} \sim N(0, \sigma_\beta^2),$$
$$(\alpha\beta)_{ij} \text{ i.i.d.} \sim N(0, \sigma_{\alpha\beta}^2)$$

In R, the model is fitted like this:

```
fit.trigly <- lmer(y ~ (1 | day) + (1 | machine)
  + (1 | machine:day), data = trigly)
```

Nesting

When we have nested factors, a factor can have a different meaning depending on another factor. Let's consider this example: The strength of a chemical paste product was measured for a total of 60 samples coming from 10 randomly selected delivery batches each containing 3 randomly selected casks ("Fässer"). Hence, two samples were taken from each cask. Cask 1 in batch 1 has nothing to do with cask 1 in batch 2 and so on. The "1" of cask has a different meaning for every batch. Hence, cask and batch are not crossed. We say cask is nested in batch. This can be modelled with:

$$Y_{ijk} = \mu + \alpha_i + \beta_{j(i)} + \epsilon_{k(ij)}$$

where α_i is the (random) effect of batch and $\beta_{j(i)}$ is the (random) effect of cask within batch with the usual assumptions α_i i.i.d. $\sim N(0, \sigma_\alpha^2)$, $\beta_{j(i)}$ i.i.d. $\sim N(0, \sigma_\beta^2)$, There are multiple ways to tell *lmer* about the nesting structure. We can use the notation *(1 | batch/cask)* which means that we want to have a random effect per batch and per cask within batch. We could also use *(1 | batch) + (1 | cask:batch)* which means that we want to have a random effect per batch and a random effect per combination of batch and cask (which is the same as a nested effect). If we already have a combination of batch and cask in the data (e.g. sample with values A:a, A:b, B:a, ...), *(1 | batch) + (1 | sample)* can also be used. But we can't use *(1 | batch) + (1 | cask)* because then all casks with the same number (across batches) would share the same effect. In a fully nested design, every factor is nested in its predecessor.

Mixed Effects Model

Mixed effects models are models which contain both random and fixed effects. For instance, if we have three brands of machines and six workers are chosen randomly among the employees of a factory to operate each machine three times, we can model this as:

$$Y_{ijk} = \mu + \alpha_i + \beta_j + (\alpha\beta)_{ij} + \epsilon_{ijk}$$

Where α_i is the fixed effect of machine i (with the usual side constraint), β_j is the random effect of worker j and $(\alpha\beta)_{ij}$ is the corresponding (random) interaction with β_j i.i.d. $\sim N(0, \sigma_\beta^2)$, $(\alpha\beta)_{ij}$ i.i.d. $\sim N(0, \sigma_{\alpha\beta}^2)$. The model is fitted as:

```
fit <- lmer(score ~ Machine + (1 | Worker) +
            (1 | Worker:Machine), data = Machines)
```

The *lmerTest* allows to get p-values for the fixed effect (Machine) by calling *anova(fit)* with the following sample output:

```
##      Sum Sq Mean Sq NumDF DenDF F.value    Pr(>F)
## Mach 38.051  19.025     2    10  20.576 0.0002855 ***
```

The degrees of freedom come from the fact that we are comparing the variation between different machines (with 2 df) to the variation due to the interaction between machine and workers (having $2 * 5 = 10$ df).
With *rand(fit)*, we can do conservative tests for the variance components of the random effects.
The interpretation of the intercept (in the *coef* output) depends on the chosen model. For a model with only fixed effects, it corresponds to the reference (e.g. reference treatment, reference subject) whereas for a mixed model, it corresponds to the reference of the fixed effect (e.g. treatment) and the expected value over all elements of the random effect (e.g. subjects).
If we use a purely fixed effects model and a mixed effects model, the p-values of the purely fixed effects model will be much more significant. This is because in the mixed effects model, we're doing inference about the population average (but see only some part of the population) whereas in the fixed effects model, we're making a statement about the observed data.

Split-Plot Designs

Split-Plot designs are experimental designs that contain experimental units of different "size".

Example

Imagine a farmer that randomizes and applies two fertilization "schemes" ("control" and "new") to eight plots of land. In addition, each plot is divided into four subplots. In each plot, four different strawberry varieties are randomized to the subplots. He is interested in the effect of fertilization scheme and strawberry variety on fruit mass. In total, there are $8 * 4 = 32$ observations.
To set up a correct model we have to follow the randomization procedure that was applied. There were two randomizations involved here:

- fertilization schemes were randomized and applied to *plots* of land
- strawberry varieties were randomized and applied to *subplots*.

Hence, an experimental unit for fertilizer is given by a *plot* of land, while for strawberry variety, the experimental unit is given by a *subplot*. Fertilizer is the so-called whole-plot factor and strawberry variety the split-plot factor. A whole-plot is given by a plot of land and a split-plot by a subplot of land. As we have two different sizes of experimental units, we also need two error terms to model the corresponding experimental errors. We need one error term "acting" on the plot level and another one on the subplot level. Let Y_{ijk} be the mass of the kth replicate of a plot with fertilization scheme i and strawberry variety j. We use the model

$$Y_{ijk} = \mu + \alpha_i + \eta_{k(i)} + \beta_j + (\alpha\beta)_{ij} + \epsilon_{ijk},$$

where α_i is the fixed effect of fertilization scheme, β_j the fixed effect of strawberry variety and $(\alpha\beta)_{ij}$ is the corresponding interaction term. $\eta_{k(i)}$ is the whole-plot error where the nesting structure ensures that we get a whole-plot error per plot of land. ϵ_{ijk} is the split-plot error (the "usual" error). We assume $\eta_{k(i)}$ i.i.d. $\sim N(0, \sigma_\eta^2)$, ϵ_{ijk} i.i.d. $\sim N(0, \sigma^2)$.
$\alpha_i + \eta_{k(i)}$ can be thought of as the "reaction" of an individual plot on the ith fertilization scheme. All plot specific properties are included in the whole-plot error $\eta_{k(i)}$. The fact that all subplots on the same plot share the same whole-plot error has the side-effect that observations from the same plot are modeled as correlated data. The part $\beta_j + (\alpha\beta)_{ij} + \epsilon_{ijk}$ is the "reaction" of the subplot on the jth variety (including a potential interaction with the ith fertilization scheme). All subplot specific properties can now be found in the split-plot error ϵ_{ijk}. If we only consider fertilization scheme, we do a completely randomized design here (with plots as experimental units). The first part of the model formula is actually the corresponding model equation. On the other side, if we only consider variety, we could treat the plots as blocks and would have a randomized complete block design on this "level" including an interaction term (this is what we see in the second part of the model formula).

In R, we use a mixed model. The whole-plot error (acting on plots) can easily be incorporated with *(1 | plot)*. The split-plot error (acting on the subplot level) is automatically included as it is on the level of individual observations. We therefore have:

```
fit <- lmer(mass ~ fertil * variety + (1 | plot),
data = d)
```

With *anova(fit)*, we can look at the F-tests:

```
##      Sum Sq Mean Sq NumDF DenDF F.value    Pr(>F)
## f    137.413 137.413     1     6  68.240 0.0001702 ***
## v    96.431  32.144     3    18  15.963 2.594e-05 ***
## f:v 4.173   1.391     3    18   0.691 0.5695061
```

The interaction is not significant while the two main effects are. We have 6 denominator degrees of freedom for fertilizer because we basically performed a completely randomized design with eight experimental units and a treatment factor having two levels, i.e. $df = 8 - 1 - 1$.

Properties

Typically, split-plot designs are suitable for situations where one of the factors can only be varied on a "large" scale. E.g., fertilizer or irrigation on (large) plots of land. While "large" was literally large in the previous example, this is not always the case. Let us consider an example with a machine running under different settings using different source material. While it is easy to change the source material it is much more tedious to change the machine settings. Hence, we don't want to change the machine settings too often. We could think of an experimental design where we change the machine setting and keep using the same setting for different source materials. This means we are not completely randomizing machine setting and source material. This would be another example of a split-plot design where machine settings is the whole-plot factor and source material is the split-plot factor.
The price on the whole-plot level is less precision (or less power) because we have much less observations on this level.

Split-Split Plot Design

If we have more than two factors, a split-split plot design can be performed, where we have an additional "layer" and therefore three "sizes" of experimental units: Whole plots, split plots and split-split plots.
For example, consider the following experiment design: The yield of oats from a split-plot field trial using 3 varieties and 4 levels of manurial treatment. The experiment was laid out in 6 blocks of 3 main plots, each split into 4 sub-plots. The varieties were applied to the main plots and the nitrogen treatments to the sub-plots.
A whole plot is given by a plot of land in a block (B), the whole-plot factor is variety (V). A block design (RCB) was used at the whole-plot level. A split plot is given by a subplot of land, the split-plot factor is given by nitrogen treatment (N). The mathematical model is:

$$Y_{ijk} = \mu + \alpha_i + \gamma_k + \eta_{ik} + \beta_j + (\alpha\beta)_{ij} + \epsilon_{ijk}$$

Where α_i is the fixed effect of variety, γ_k the fixed effect of block, η_{ik} the whole-plot error ($\mathcal{N}(0, \sigma_\eta^2)$), β_j the fixed effect of nitrogen treatment, $(\alpha\beta)_{ij}$ the interaction between variety and nitrogen treatment and ϵ_{ijk} the split-plot error. In R, we would model this as:

```
fit <- lmer(Y ~ B + V * N + (1 | B:V), data=oats)
```

Incomplete Block Designs

The block designs we have seen so far were complete, meaning that every block contained all treatments. In practice, this is not always possible. For example, the physical size of a block might be too small. There are also situations where it is not advisable to have too many treatments in each block. If we do a food tasting experiment, we typically want to restrict the number of different "recipes" (treatments) we want to show to an individual rater. In an incomplete block design (IBD), we have to decide what subset of treatments we use on an individual block.

We call a design disconnected if we can build two groups of treatments such that it never happens that we see members of both groups together in the same block. In such designs, certain quantities can't be estimated (e.g. comparing treatment in different groups). If we fit a standard main effect model, two of the treatment coefficients will be set to zero (not only one as usual when using *contr.treatment* and looking at the coefficients with *dummy.coef*), e.g.:

```
## (Intercept):      0.307025
## block:                 1          2 ...
##                0.0000000 -0.2788532 ...
## treat:                 A          B ...
##                0.0000000  0.9989451 ...
```

R automatically drops one of the treatment levels out of the model (which might be dangerous if you are not carefully inspecting the output) in these cases.

Balanced Incomplete Block Designs

A possible optimality criterion is that we can estimate all treatment differences with the same precision, i.e. all confidence intervals for $\alpha_i - \alpha_j$ have the same width (for any pair i, j). This is fulfilled by a balanced incomplete block design (BIBD). A BIBD is an incomplete block design where all pairs of treatment occur together in the same block equally often ($= \lambda$). The following notation is used:

- g: Number of treatments ("number of different chocolate chip cookie brands")
- b: Number of blocks ("number of raters")
- k: Number of units per block ($k < g$) ("number of cookie brands each raters gets to see")
- r: Number of replicates per treatment ("how often do we see a certain cookie brand across all raters?")
- N: Total number of units

We have $N = b * k = g * r$. For every setting $k < g$, it's possible to find a BIBD by taking all possible $\binom{g}{k}$ (*choose* in R) subsets. The corresponding design is called an unreduced balanced incomplete block design. In R, the combinations are retrieved with *combn*, e.g. *combn(x = 4, m = 3)* which would generate 4 blocks because $\binom{4}{3} = 4$ which would give the following outputs (where the columns are blocks):

```
##      [,1] [,2] [,3] [,4]
## [1,]    1    1    1    2
## [2,]    2    2    3    3
## [3,]    3    4    4    4
```

Typically, we would randomize the order or the placement of the treatments within a block.
In practice we cannot always afford to do an unreduced BIBD as the required number of blocks might be too large. Whether a BIBD exists for a certain desired setting of number of blocks b, block size k and number of treatments g, is a combinatorial problem. A necessary (but not sufficient, i.e. even if the condition is fulfilled, it might be the case that you can't find a BIBD) condition for a BIBD to exist is

$$\frac{r \cdot (k-1)}{g-1} = \lambda.$$

The intuition behind the formula is that a treatment appears in r different blocks. In each block, there are $k-1$ available other units, leading to a total of $r * (k-1)$ available units. There are $g-1$ available treatments that must be divided among them in order to have a balanced design. Hence, $r \cdot (k-1) = \lambda \cdot (g-1)$ must hold. In R, the package *ibd* provides some functionality to find (B)IBDs. E.g. the function *bibd* can be used to find a BIBD:

```
des.bibd <- bibd(v = 6, b = 10, r = 5, k = 3, lambda = 2)
```

where v is the number of treatments and the rest as defined above. In *des.bibd$design*, each row corresponds to a block:

```
##        [,1] [,2] [,3]
##  [1,]    4    5    6
##  [2,]    1    4    5
##  [3,]    1    3    4
##  [4,]    1    3    6
##  [5,]    3    5    6
##  [6,]    1    2    6
##  [7,]    2    4    6
##  [8,]    2    3    5
##  [9,]    1    2    5
## [10,]    2    3    4
```

With *des.bibd$NNP*, we can view the concurrency matrix. When no BIBD is possible, *ibd* can be used to find a nearly balanced design: *des.ibd <- ibd(v = 6, b = 9, k = 3)* In the concurrency matrix (*des.ibd$conc.mat*), we see how often any pair of treatments appear together in the same block, e.g.:

```
##        [,1] [,2] [,3] [,4]
## [1,]    1    0    1    0
## [2,]    0    1    1    0
## [3,]    1    1    3    1
## [4,]    0    0    1    1
```

isGYD(d) can be used to check if an incomplete design is balanced, where d is a matrix with blocks as rows (i.e. the same output format as *des.bibd*)
In a partially balanced incomplete block design, some treatment pairs occur together more often than other pairs.

Row-Column Incomplete Block Designs

In these designs, we have two block factors (rows and columns) and one or both of them are incomplete blocks.

Youden Squares

A Youden Square is rectangular such that the columns (or rows; can be flipped) form a BIBD and for the rows (or columns), each treatment appears equally often in each row (column). The columns therefore form a BIBD, the rows an RCB.

Analysis

The analysis of an incomplete block design is "as usual". We use a block factor and a treatment factor leading to

$$Y_{ij} = \mu + \alpha_i + \beta_j + \epsilon_{ij}.$$

As we are faced with an unbalanced design we typically use sum of squares for treatment effects that are adjusted for block effects (i.e. *drop1*), e.g.:

```
fit <- aov(dishes ~ session + detergent, data = dish)
drop1(fit, test = "F")
```

Where we would get the p-values for session and detergent (but would only be interested in detergent).
In this approach, we analyze the treatment effects while controlling for the block effects, which is a so called intrablock analysis of the (B)IBD. It is also possible to recover some information by comparing different blocks, which would be called an interblock analysis.
If we don't use adjusted sum of squares (*drop1*), it's important to first list the block factor in the model and then the treatment, as we're interested in seeing if the treatment has any influence on the response, after controlling for the variation between blocks.

Power

A statistical test controls by construction the type I error rate with the significance level α. This means the probability that we falsely reject the null hypothesis H_0 is less than α. The type II error occurs if we fail to reject the null hypothesis even though the alternative hypothesis H_A holds. The probability of a type II error is denoted by β (and we aren't controlling it). There is no universal β, it depends on the specific alternative H_A that is assumed. The power of a statistical test is:

$$P(\text{reject } H_0 \mid \text{a certain setting in } H_A \text{ holds}) = 1 - \beta.$$

For calculating power, no data is needed but a precise specification of the parameter setting under the alternative that we believe in ("what would happen if...").
If we plan an experiment with low power, it means that we waste time and money because with high probability we are not getting a significant result. A "rule of thumb" is that power should be larger than 80%. Power depends on:

- Design of the experiment (balanced, unbalanced, without blocking, with blocking, ...)
- Significance level α
- Parameter setting under the alternative (incl. error variance σ^2
- Sample size n

We can mainly maximize power using the first (design) and last item (sample size) from above.
For some designs (like a completely randomized design), there are closed-form formulas to calculate power. When the design is getting more complex, simulations are usually used.

For example, in a simple one-way ANOVA model with five groups and the null hypothesis:

$$H_0 : \mu_1 = \ldots = \mu_5$$

And the alternative hypothesis:

$$H_A : \mu_k \neq \mu_l \text{ for at least one pair } k \neq l.$$

One possible assumption for the alternative would be:

$$\mu_1 = 57,\ \mu_2 = 63,\ \mu_3 = 60,\ \mu_4 = 60,\ \mu_5 = 60.$$

In addition, the error variance has to be specified, which is assumed to be $\sigma^2 = 7.5$. For this design, the *power.anova.test* function can be used. It needs the number of groups, the variance between the group means μ_i, the error variance σ^2 and the sample size within each group (n). It is then called like this:

```
mu     <- c(57, 63, rep(60, 3))
sigma2 <- 7.5
power.anova.test(groups = length(mu), n = 4,
 between.var = var(mu), within.var = sigma2)
```

And a possible output would be:

```
##            power = 0.5452079
```

Which would mean we have a 54% chance to get a significant result under the above setting. We can also leave away n and use the argument *power* to get the required sample size (per group) for a certain power.
We can also simply simulate the data many times, do the corresponding test and measure the proportion of simulation runs that rejected H_0 (which is then the power). Because this is simply a binomial distribution (sum of independent Bernoulli trials), the function *binom.test(x, n)* can be used to get a confidence interval on the estimated power, where x is the number of times H_0 was rejected (the number of "successes") and n the number of overall simulations. If this confidence interval is too wide, n should be increased, i.e. one should do more simulation runs. For simulation, the function

```
rnorm(n * g, mean = rep(means, each = n), sd = sigma)
```

can be helpful, which (in the above case) generates a vector of normally distributed variables with mean *mean* and standard deviation *sigma*.

Notation

$y_{i\cdot} = \sum_{j=1}^{n_i} y_{ij}$	sum of group i
$y_{\cdot\cdot} = \sum_{i=1}^{g}\sum_{j=1}^{n_i} y_{ij}$	sum of all observations
$\overline{y}_{i\cdot} = \frac{1}{n_i}\sum_{j=1}^{n_i} y_{ij}$	mean of group i
$\overline{y}_{\cdot\cdot} = \frac{1}{N}\sum_{i=1}^{g}\sum_{j=1}^{n_i} y_{ij}$	overall (or total) mean

Design Examples

Latin Square Design Example

	C_1	C_2	C_3	C_4
R_1	A	B	C	D
R_2	B	C	D	A
R_3	C	D	A	B
R_4	D	A	B	C

Graeco-Latin Square Design Example

	C_1	C_2	C_3	C_4
R_1	$A\alpha$	$B\gamma$	$C\delta$	$D\beta$
R_2	$B\beta$	$A\delta$	$D\gamma$	$C\alpha$
R_3	$C\gamma$	$D\alpha$	$A\beta$	$B\delta$
R_4	$D\delta$	$C\beta$	$B\alpha$	$A\gamma$

Row-Column Incomplete Block Design Example

	C_1	C_2	C_3	C_4	C_5	C_6	C_7
R_1	3	4	5	6	7	1	2
R_2	5	6	7	1	2	3	4
R_3	6	7	1	2	3	4	5
R_4	7	1	2	3	4	5	6

The rows are complete blocks (there are 7 treatments), the columns form a BIBD, which is a so called row-orthogonal design.

R Details

Factors

Categorical variables in R are also called factors and the different values levels. With *str*, we can check the structure of data (if it's encoded as a factor). If it's not, we can use *factor*, e.g.:

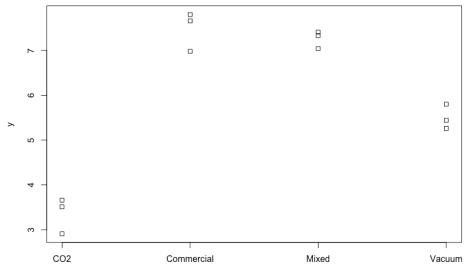
```
data$column = factor(data$column)
```

With *levels*, we can check the levels of a factor.

Visualization

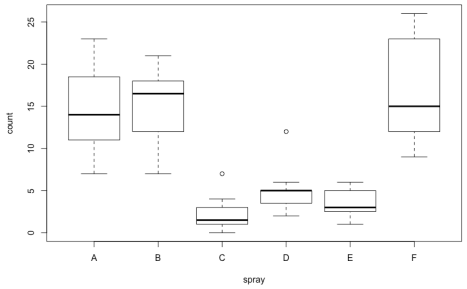
The *stripchart* function can be used to generate a stripchart:

```
stripchart(y ~ treatment, data = meat, vertical = TRUE)
```



And *boxplot* for boxplots:

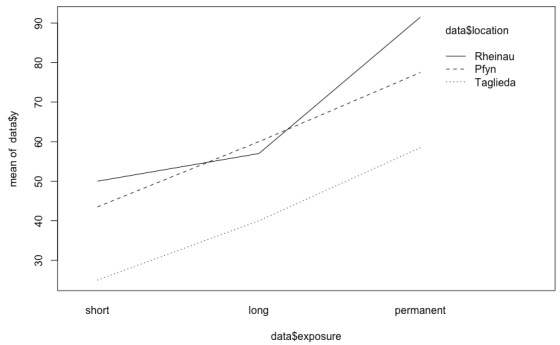
```
boxplot(count ~ spray, data = InsectSprays)
```



The thick line in the middle is the median value, the rectangle the interquartile range (IQR), i.e. the lower value is the 25th percentile and the higher value the 75th percentile. The whiskers (top / bottom lines) are usually bounded by 1.5 * IQR, values outside are displayed as outliers.

An interaction plot can be used to visualize interactions between two factors:

```
with(data, interaction.plot(x.factor = exposure,
           trace.factor = location, response = y))
```



Data Generation

Generate 10 "A"'s, followed by 10 "B"'s (two methods):

```
c(rep("A", times = 10), rep("B", times = 10))
rep(c("A", "B"), each = 10)
```

Alternate "A", "B" 10 times:

```
rep(c("A", "B"), times = 10)
```

Toss a coin 20 times (0.5 prob. for "A", "B"):

```
sample(c("A", "B"), 20, replace = TRUE)
```

Choose 10 "A" at random, the rest "B":

```
sample(rep(c("A", "B"), times = 10), 20, replace = FALSE)
```

Calculations

Overall mean of column "blood" (two methods):

```
mean(b.data$blood)
aggregate(blood ~ 1, data = b.data, mean)
```

Group means per treatment:

```
aggregate(blood ~ treat, data = b.data, mean)
```

Tests

If we want to check if we can simultaneously drop A and B from the model:

```
fit.null <- aov(Y ~ 1, data = dat)
fit.main <- aov(Y ~ A + B, data = dat)
anova(fit.null, fit.main)
```

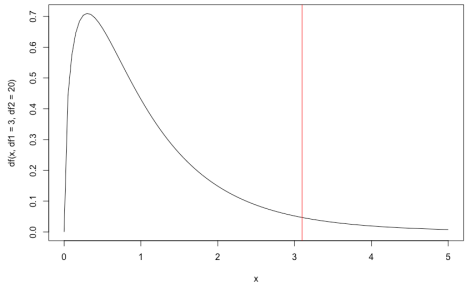
Mathematical Background

F Distribution

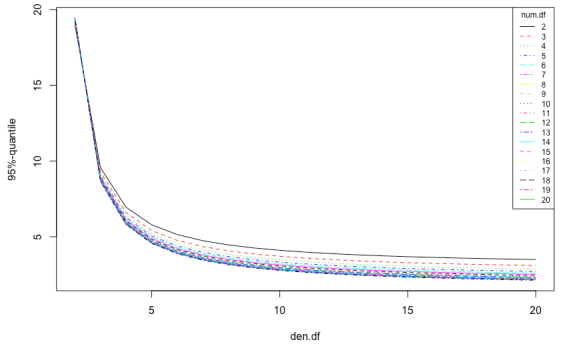
We have:

$$F_{n,m} = \frac{\frac{1}{n}(X_1^2 + \ldots + X_n^2)}{\frac{1}{m}(Y_1^2 + \ldots + Y_m^2)}$$

Where X_i, Y_j are i.i.d. $\mathcal{N}(0,1)$. The $F_{1,m}$-distribution is a special case, it's the square of a t_m-distribution. The F distribution with 3 denominator degrees of freedom and 20 numerator degrees of freedom looks like this (where the red line is the 95%-quantile):



The 95%-quantile behaves like this, depending on the denominator / numerator degrees of freedom:



Two Sample t-Test for Unpaired Data

We have X_i i.i.d. $\sim \mathcal{N}(\mu_X, \sigma^2)$ and Y_j i.i.d. $\sim \mathcal{N}(\mu_Y, \sigma^2)$ with X_i, Y_j independent. For the t-Test, $H_0 : \mu_X = \mu_Y$ and $H_A : \mu_X \neq \mu_Y$ (or one-sided). Then:

$$T = \frac{(\bar{X}_n - \bar{Y}_m)}{s_{pool}\sqrt{\frac{1}{n} + \frac{1}{m}}} \sim t_{n+m-2} \text{ under } H_0$$

Two Sample t-Test for Paired Data

We have independent $D_i = X_i - Y_i$ and:

$$\bar{D} = \frac{1}{n}\sum_{i=1}^{n} D_i \sim \mathcal{N}(\mu_D; \sigma_D/\sqrt{n})$$

H_0 / H_A as before and:

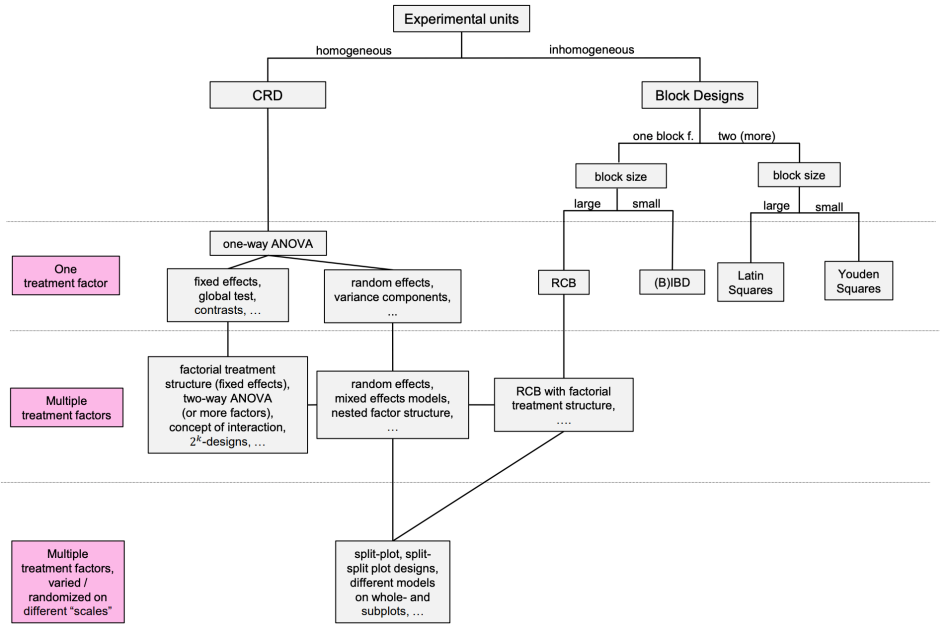
$$T = \sqrt{n}\frac{\bar{D}}{S_D} \sim t_{n-1} \text{ under } H_0$$

Binomial Coefficient

$$\binom{n}{k} = \frac{n!}{k! * (n-k)!}$$

Appendix

Overview



ANOVA table

Source	df	Sum of squares	Mean squares	F-ratio
Treatment	$g-1$	SS_{Trt}	$MS_{\text{Trt}} = \frac{SS_{\text{Trt}}}{g-1}$	$\frac{MS_{\text{Trt}}}{MS_E}$
Error	$N-g$	SS_E	$MS_E = \frac{SS_E}{N-g}$	

Where g = treatments / levels of factor, N = overall experimental units.

Two-Way ANOVA table

Source	df	Sum of squares	Mean squares	F-ratio
A	$a-1$	SS_A	$MS_A = \frac{SS_A}{a-1}$	$\frac{MS_A}{MS_E}$
B	$b-1$	SS_B	$MS_B = \frac{SS_B}{b-1}$	$\frac{MS_B}{MS_E}$
AB	$(a-1)(b-1)$	SS_{AB}	$MS_{AB} = \frac{SS_{AB}}{(a-1)(b-1)}$	$\frac{MS_{AB}}{MS_E}$
Error	$ab(n-1)$	SS_E	$MS_E = \frac{SS_E}{ab(n-1)}$	

Where a = levels of factor A, B = levels of factor B, n = replicates for every combination of A, B.

Examples

Split-Plot Design, ANOVA skeleton

Three new types of pizzas in six different packings are investigated by 90 consumers on a 0-10 scale. Each person rates the six packings of just one type of pizza, that is pizzas are randomized to persons and each person tastes the different packings in random order.
This is a split-plot design with persons as whole plots and rating orders (or time slots) as split plots. Pizza type is the whole-plot factor, packing the split-plot factor. We have:

$$Y_{ijk} = \mu + \beta_i + \eta_{k(i)} + \alpha_j + (\alpha\beta)_{ij} + \epsilon_{k(ij)}$$

Where $\eta_{k(i)}$ is the whole-plot error (per person). The ANOVA skeleton is given by:

Plot level	Source	df	MS	F
Whole plots	pizza	2	MS_B	$\frac{MS_B}{MS_\eta}$
	residual	87	MS_η	
Split plots	packing	5	MS_A	$\frac{MS_A}{MS_E}$
	packing:pizza	10	MS_{AB}	$\frac{MS_{AB}}{MS_E}$
	residual	435	MS_E	
	total	539		

Split-Plot Design with Blocking

A soil scientist wanted to investigate the effects of nitrogen supplied in four different forms and later evaluate those effects combined with those of thatch accumulation (two, five or eight years of accumulation) on the quality of an established turf. A golf green had been constructed and seeded with grass on the experimental plots. The nitrogen treatment plots were arranged on the golf green in a randomized complete block design with two block levels. Each of the eight experimental plots was split into three subplots to which the levels of the second treatment factor were randomly assigned.
This is a split-plot design with whole-plot factor nitrogen, split-plot factor thatch and a block factor block.

$$Y_{ijkl} = \mu + \gamma_i + \alpha_j + \beta_k + (\alpha\beta)_{jk} + \eta_{l(ij)} + \epsilon_{l(ijk)}$$

Where $l = 1$, γ_i fixed effect of block ($i = 1, 2$), α_j fixed main effect of nitrogen ($j = 1, 2, 3, 4$), β_k fixed main effect of thatch ($k = 1, 2, 3$), $(\alpha\beta)_{jk}$ interaction, $\eta_{l(ij)}$ the error on the whole-plot level and $\epsilon_{l(ijk)}$ the error on the split-plot level.