

Consulting Handbook

Jarad Niemi, Rebekah Scott

2025-09-17

Table of contents

Preface	4
1 Introduction	5
I Client Interactions	6
2 Active Listening	8
II Statistical Methods	9
3 ANOVA Nuances	11
3.1 Sums of Squares Refresher	11
3.2 Changing SS Types	13
3.2.1 In SAS	13
3.2.2 In R	13
3.3 Example: Alpine Data	14
3.3.1 Type I SS	14
3.4 R	14
3.5 SAS	15
3.5.1 Type II SS	16
3.6 R	16
3.7 SAS	17
3.7.1 Type III SS	18
3.8 R	18
3.9 SAS	19
3.10 Bonus Code	20
3.10.1 After ANOVA in R: emmeans with Alpine Data	20
3.10.2 Broad vs Narrow Inference: Alpine Data Mixed Model	22
4 Experimental Design	24

III Case Studies	25
5 Example Study	27
References	28

Preface

In today's data-driven world, statistical consulting plays a vital role in supporting research across all domains. This handbook is designed to serve as a practical guide for statistical consultants in a university environment, and is divided into three sections: Client Interactions, Statistical Methodology, and Case Studies.

This handbook has been shaped by years of experience in [statistical consulting at Iowa State University](#) and inspired by [Statistical Consulting by Cabrera and McDougall](#). Our goal is to empower consultants to approach each project with confidence, curiosity, and a commitment to excellence.

We hope this handbook becomes a trusted companion in your consulting journey and enhance impact of your work across the university community.

1 Introduction

This is a book created from markdown and executable code.

See Knuth (1984) for additional discussion of literate programming.

```
1 + 1
```

```
[1] 2
```

Part I

Client Interactions

In summary, this book has no content whatsoever.

1 + 1

[1] 2

2 Active Listening

In summary, this book has no content whatsoever.

1 + 1

[1] 2

Part II

Statistical Methods

In summary, this book has no content whatsoever.

1 + 1

[1] 2

3 ANOVA Nuances

3.1 Sums of Squares Refresher

The three common types of Sums of Squares (SS) are equivalent for balanced data: Type I, II, and III (I believe SAS is to blame for the unhelpful naming). These SS only differ when data is unbalanced.

Sums of Squares can be thought of in many different ways, from calculating squared distances from the regression lines and group means to subtracting projection matrices with nested column spaces. We can consider the SS as **sequential F-tests for a null or alternative model** (from The Statistical Sleuth), where the scaled difference in SS (alternative - null) form the numerator in the F-test. This can also be thought of in a regression context with setting some coefficients to zero. Each of the three SS types have a particular style of F-tests, which are explained below.

Notice that **the F-test for the highest order interaction is the same for each SS type**.

Type I SS are sequential—the order of main effects matters! Notice that for effect A, it's just the SS of A given the intercept, but for effect B it's the SS of B given the intercept and A. Type I SS are the only SS that add to the total SS with unbalanced data; Type II and III SS do not add up to anything special with unbalanced data.

Variable	Null Model	Alternative Model
A	1 (Intercept-only)	A
B	A	A + B
AB	A + B	A + B + AB

If we have the variables in a meaningful order, we may be interested in the whole sequence of tests formed by Type I SS.

Type II SS are hierarchical. This SS ignores interactions when testing for main effects (i.e. the interaction is excluded from both the null and alternative model). Type II SS are rarely used.

Variable	Null Model	Alternative Model
A	B	A + B
B	A	A + B
AB	A + B	A + B + AB

Type III SS consider dropping one term at a time. Only Type III SS account for all other terms in the model. The corresponding ANOVA F-tests can be used to test for main effects and interactions.

Variable	Null Model	Alternative Model
A	B + AB	A + B + AB
B	A + AB	A + B + AB
AB	A + B	A + B + AB

Type IV SS are only relevant for factorial designs with missing cells (i.e. fractional factorials) and are equivalent to Type III when there are no missing cells.

Putting it all together, the below table is an alternative view of three SS types, taken from slide 60 of Dr. Dan Nettleton's STAT 0510 lecture [8: Analysis of Variance for Unbalanced Two-Factor Experiments](#). The model term(s) on the right being conditioned on form the null model, whereas the term on the left is being added to the null model to form the alternative model.

Source	Type I	Type II	Type III
A	SS (A 1)	SS (A 1, B)	SS (A 1, B, AB)
B	SS (B 1, A)	SS (B 1, A)	SS (B 1, A, AB)
AB	SS (AB 1, A, B)	SS (AB 1, A, B)	SS (AB 1, A, B)

The following slide in Dr. Nettleton's lecture has this table for 3 factors (see slide 61).

Rather than using these default ANOVA tables, you can construct specific tests to address specific scientific questions. In R, this would look something like

```
anova(m1, m2)
```

where m1 is the null model and m2 is the alternative model.

When you test a null vs alternative model this way in R, the F statistic and p-value will only match the ANOVA table if the alternative model is the full model containing every term. The SS and Means Squares (MS) will all be the same. Only Type III SS always specify the alternative model as the full model.

3.2 Changing SS Types

3.2.1 In SAS

Specifying the SS type in SAS is very simple - you only need to specify `/ ss3` in the model statement for `proc glm` or `method = type3` in `proc mixed`. For example, for Type III

```
/* proc glm */
proc glm data=dat;
  class x1 x2;
  model y = x1 x2 x1*x2 / ss3;
run;

/* proc mixed */
proc mixed data=dat method = type3;
  class x1 x2;
  model y = x1 x2 x1*x2;
run;
```

3.2.2 In R

Specifying the SS type in R is tricky.

`anova` in base R calculates Type I SS. For example,

```
dat_lm <- lm(y ~ x1 * x2, data=dat)
anova(dat_lm)

# make sure treatment variables are factors!
```

The `car::Anova` function allows for Type II and III SS, however the so-called Type III SS does not match `emmeans::joint_tests` or SAS unless you switch to an orthogonal basis.

For Type II,

```
car::Anova(dat_lm, type = 2)
```

For Type III, use `emmeans::joint_tests`

```
emmeans::joint_tests(lm(y ~ x1 * x2, data=dat))
```

Or switch to an orthogonal basis using `contr.sum`, `contr.helmert` or `contr.poly` to use `car::Anova`. Note: `contr.SAS` is not orthogonal.

```
options(contrasts = c('contr.sum', 'contr.poly'))
dat_lm2 <- lm(y ~ x1 * x2, data=dat)
car::Anova(dat_lm2, type=3)
```

3.3 Example: Alpine Data

The alpine data have three factors: place (Cumb, Penn), species (grac, nive), site (Dry, Mid, Wet) with unequal unequal sample sizes, but no missing cells. We will analyze this as a three-way factorial. Credit to Dr. Philip Dixon for this data and example.

3.3.1 Type I SS

3.4 R

We calculate Type I using `anova` in base R,

```
# create the linear model
alpine.lm <- lm(lnwt ~ place*site*spp, data=alpine)

# type I (sequential) SS
anova(alpine.lm)
```

Analysis of Variance Table

Response: lnwt

	Df	Sum Sq	Mean Sq	F value	Pr(>F)	
place	1	5.1939	5.1939	19.4377	2.688e-05	***
site	2	0.5404	0.2702	1.0111	0.3676	
spp	1	15.9614	15.9614	59.7337	1.007e-11	***
place:site	2	0.4640	0.2320	0.8683	0.4229	
place:spp	1	0.5362	0.5362	2.0066	0.1598	
site:spp	2	15.8529	7.9264	29.6639	8.865e-11	***
place:site:spp	2	0.1563	0.0781	0.2925	0.7471	

Residuals 97 25.9192 0.2672

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

3.5 SAS

Which matches SAS.

```
proc glm;
  class place spp site;
  model lnwt = place site spp place*site place*spp site*spp place*site*spp / ss1;
run;
```

The GLM Procedure

Dependent Variable: lnwt

Source	DF	Type I SS	Mean Square	F Value	Pr > F
place	1	5.19393348	5.19393348	19.44	<.0001
site	2	0.54036876	0.27018438	1.01	0.3676
spp	1	15.96135063	15.96135063	59.73	<.0001
place*site	2	0.46403991	0.23201996	0.87	0.4229
place*spp	1	0.53616754	0.53616754	2.01	0.1598
spp*site	2	15.85289774	7.92644887	29.66	<.0001
place*spp*site	2	0.15629797	0.07814899	0.29	0.7471

To demonstrate the idea of comparing null and alternative models instead of using the default ANOVA table, let's test if we need to include site in our model given that we are already including place. This hypothesis test corresponds site in the Type I SS ANOVA table above.

```
null.lm <- lm(lnwt ~ place, data=alpine)
alt.lm <- lm(lnwt ~ place + site, data=alpine)

anova(alt.lm)
```

Analysis of Variance Table

Response: lnwt

Df	Sum Sq	Mean Sq	F value	Pr(>F)
----	--------	---------	---------	--------

```
place      1  5.194  5.1939  9.2607 0.002958 **
site       2  0.540  0.2702  0.4817 0.619069
Residuals 105 58.890  0.5609
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
anova(null.lm, alt.lm)
```

Analysis of Variance Table

```
Model 1: lnwt ~ place
Model 2: lnwt ~ place + site
  Res.Df  RSS Df Sum of Sq    F Pr(>F)
1     107 59.43
2     105 58.89  2   0.54037 0.4817 0.6191
```

Again, the SS and MS for site match the ANOVA output, however the F statistic and p-value do not match because they are calculated from the MSE of the alternative model only containing place and site.

```
# anova
# mean sq / mse (of full model)
0.2702 / 0.2672 # 1.01 F-stat
```

```
[1] 1.011228
```

```
# mean sq / mse (of alt model)
0.2702 / 0.56093 # 0.4817 F-stat
```

```
[1] 0.4817
```

Be careful!

3.5.1 Type II SS

3.6 R

Next, we will do Type II SS in R using `car::Anova` with `type = 2`


```
# type II (hierarchical) SS
car::Anova(alpine.lm, type=2)
```

Anova Table (Type II tests)

Response: lnwt

	Sum Sq	Df	F value	Pr(>F)
place	3.0097	1	11.2635	0.00113 **
site	0.9711	2	1.8172	0.16797
spp	16.2202	1	60.7025	7.435e-12 ***
place:site	0.2484	2	0.4647	0.62970
place:spp	0.1497	1	0.5603	0.45594
site:spp	15.8529	2	29.6639	8.865e-11 ***
place:site:spp	0.1563	2	0.2925	0.74708
Residuals	25.9192	97		

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

3.7 SAS

Which matches SAS

```
proc glm;
  class place spp site;
  model lnwt = place site spp place*site place*spp site*spp place*site*spp / ss2;
run;
```

The GLM Procedure

Dependent Variable: lnwt

Source	DF	Type II SS	Mean Square	F Value	Pr > F
place	1	3.00971222	3.00971222	11.26	0.0011
site	2	0.97112780	0.48556390	1.82	0.1680
spp	1	16.22022467	16.22022467	60.70	<.0001
place*site	2	0.24835684	0.12417842	0.46	0.6297
place*spp	1	0.14972544	0.14972544	0.56	0.4559
spp*site	2	15.85289774	7.92644887	29.66	<.0001
place*spp*site	2	0.15629797	0.07814899	0.29	0.7471

3.7.1 Type III SS

3.8 R

Now, the more complex Type III SS in R. We will show the results from `emmeans::joint_tests`, which match SAS. We will also show `car::Anova` with and without an orthogonal basis.

```
# use emmeans::joint_tests
emmeans::joint_tests(alpine.lm)
```

model term	df1	df2	F.ratio	p.value
place	1	97	10.959	0.0013
site	2	97	0.740	0.4796
spp	1	97	58.084	<.0001
place:site	2	97	0.393	0.6763
place:spp	1	97	0.561	0.4559
site:spp	2	97	29.741	<.0001
place:site:spp	2	97	0.292	0.7471

```
# or switch to an orthogonal basis for car::Anova
options(contrasts = c('contr.sum', 'contr.poly'))
alpine2.lm <- lm(lnwt ~ place*site*spp, data=alpine)
car::Anova(alpine2.lm, type=3)
```

Anova Table (Type III tests)

Response: lnwt

	Sum Sq	Df	F value	Pr(>F)
(Intercept)	1723.11	1	6448.5415	< 2.2e-16 ***
place	2.93	1	10.9592	0.001309 **
site	0.40	2	0.7403	0.479627
spp	15.52	1	58.0841	1.697e-11 ***
place:site	0.21	2	0.3927	0.676296
place:spp	0.15	1	0.5605	0.455864
site:spp	15.89	2	29.7405	8.453e-11 ***
place:site:spp	0.16	2	0.2925	0.747078
Residuals	25.92	97		

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

```
# otherwise, car::Anova does not match joint_tests or SAS
car::Anova(alpine.lm, type=3)
```

Anova Table (Type III tests)

Response: lnwt

	Sum Sq	Df	F value	Pr(>F)
(Intercept)	141.978	1	531.3386	< 2.2e-16 ***
place	0.645	1	2.4151	0.123428
site	3.808	2	7.1256	0.001296 **
spp	0.392	1	1.4670	0.228759
place:site	0.005	2	0.0098	0.990255
place:spp	0.001	1	0.0043	0.947784
site:spp	7.460	2	13.9598	4.697e-06 ***
place:site:spp	0.156	2	0.2925	0.747078
Residuals	25.919	97		

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

The `joint_tests` and orthogonal `car::Anova` output matches SAS, however `car::Anova` on the original `alpine.lm` object differs.

3.9 SAS

```
proc glm;
  class place spp site;
  model lnwt = place site spp place*site place*spp site*spp place*site*spp / ss3;
run;
```

The GLM Procedure

Dependent Variable: lnwt

Source	DF	Type III SS	Mean Square	F Value	Pr > F
place	1	2.92839766	2.92839766	10.96	0.0013
site	2	0.39565116	0.19782558	0.74	0.4796
spp	1	15.52056508	15.52056508	58.08	<.0001
place*site	2	0.20986856	0.10493428	0.39	0.6763

place*spp	1	0.14977498	0.14977498	0.56	0.4559
spp*site	2	15.89385223	7.94692612	29.74	<.0001
place*spp*site	2	0.15629797	0.07814899	0.29	0.7471

Let us again look at the idea of comparing null and alternative models by testing for the three-way interaction given that we are including all other model terms. This hypothesis test corresponds to the three-way interaction in the Type III SS ANOVA table above.

```

null.lm2 <- lm(lnwt ~ place + site + spp +
               place:site + place:spp + site:spp,
               , data=alpine)
alt.lm2 <- lm(lnwt ~ place*site*spp, data=alpine)

anova(null.lm2, alt.lm2)

```

Analysis of Variance Table

```

Model 1: lnwt ~ place + site + spp + place:site + place:spp + site:spp
Model 2: lnwt ~ place * site * spp
      Res.Df    RSS Df Sum of Sq    F Pr(>F)
1         99 26.076
2         97 25.919  2    0.1563 0.2925 0.7471

```

This time, the F statistic and p-value match because the alternative model is the full model, as with all hypothesis tests associated with Type III SS.

3.10 Bonus Code

3.10.1 After ANOVA in R: emmeans with Alpine Data

To quote Philip, “R doesn’t make ‘after the ANOVA’ especially easy. The `emmeans` (successor to `lsmeans`) package implements a powerful set of after the ANOVA tools. Basic approach is to make a `emmeans` object for the factor of interest, then look at properties of that object.

This example code includes calculating cell means, means within and averaging over place, contrast examples, and much more.

```

# analysis using emmeans (all three factors are of interest)
alpine.emm <- emmeans(alpine.lm, c('spp','site','place'))

# cell means with se, df and 95% CI
alpine.emm

# means within each place
emmip(alpine.lm, spp ~ site | place)

# cell means for site and spp averaging over place
emmeans(alpine.lm, c('site','spp'))

# F tests of site effects at each place and spp
joint_tests(alpine.lm, by=c('place','spp'))

# pairwise comparisons between all cells (not recommended!)
emmeans(alpine.lm, pairwise ~ place:site:spp)

# or between site and spp averaged over place
emmeans(alpine.lm, pairwise ~ site:spp)

# can specify same request as a set of factors and desired contrast(s)
emmeans(alpine.lm, c('site','spp'), contr = 'pairwise')

# sites for each spp averaging over place
emmeans(alpine.lm, pairwise ~ site | spp)

```

?'contrast-methods' lists all the options for built-in contrasts

Remember that you must know the order of the groups to properly specify the contrast. You can see the order of the group in many different ways including

- Look at the order of the groups in the emmeans() output.
- Print the emmeans object.
- Look at the sorted order of unique factor values: `sort(unique(factor))`

```

# let's focus on the sites
alpine.emm1 <- emmeans(alpine.lm, 'site')

# all pairwise comparisons is easy
site.pairs <- pairs(alpine.emm1) # Tukey adjustment by default

```

```

# specifying summary( ) with adjust= allows you to choose the adjustment
# from tukey, bonferroni, none, scheffe, sidak, dunnetttx
summary(site.pairs, adjust='none')

# and specifying summary( ) with infer=c(T, F) gives you CI's
summary(site.pairs, infer=c(T,F))
# first value for infer is confidence intervals, second is tests
# so infer=c(F,F) just gives estimates, infer=c(T,T) gives both CI's and tests

# or use confint() to which you can add adjust= to specify adjustments
confint(site.pairs)

# can specify desired contrasts either in a call to emmeans() or contrast()
# notice you can provide multiple contrasts in one call, but you must always
# give a list even if you are only doing one contrast
contrast(alpine.emm1, list('D-MW'=c(1, -0.5, -0.5), 'M-W'=c(0, 1, -1)))

alpine.emm2 <- emmeans(alpine.lm, c('site', 'spp'))
alpine.c1 <- contrast(alpine.emm2, by='spp',
                     list('D-MW'=c(1, -0.5, -0.5),
                          'M-W'=c(0, 1, -1)))

# can get CI's for the contrasts
summary(alpine.c1, infer=c(T,F), adjust='bonferroni')

```

3.10.2 Broad vs Narrow Inference: Alpine Data Mixed Model

Let's treat the alpine data as a repeated experiment with a single pooled place*anything interaction, random effect. The places are random (not ideal because only 1 df), so inference is broad-sense to population of places.

```

library(lme4)
alpine.lmm <- lmer(lnwt ~ site+spp+site:spp + (1|place) + (1|place:site:spp),
  data=alpine)
summary(alpine.lmm) # mostly to see estimated variance components

joint_tests(alpine.lmm)
# uses Kenward-Rogers df adjustments, different implementation than SAS
# (slightly different when all variance components > 0, and quite different
# when one or more = 0). R seems to account for estimating the 0, SAS ignores the 0.
# to see the difference, remove the interaction variance from the model

```

```

alpine2.lmm <- lmer(lnwt ~ site+spp+site:spp + (1|place),
  data=alpine)
summary(alpine2.lmm) # mostly to see estimated variance components
joint_tests(alpine2.lmm)

# matches SAS with or without interaction variance

```

```

proc glimmix;
  class place spp site;
  model lnwt = site spp site*spp / ddfm = kr;
  random place place*site*spp;
  lsmeans site*spp;
  title 'Repeated experiment analysis, Kenward-Roger df adjustment';
run;

```

4 Experimental Design

In summary, this book has no content whatsoever.

Frey et al. (2024) explains why you can't remove blocks in a designed experiment.

1 + 1

[1] 2

Part III

Case Studies

In summary, this book has no content whatsoever.

1 + 1

[1] 2

5 Example Study

In summary, this book has no content whatsoever.

1 + 1

[1] 2

References

- Frey, Janina, Jens Hartung, Joseph Ogutu, and Hans-Peter Piepho. 2024. “Analyze as Randomized—Why Dropping Block Effects in Designed Experiments Is a Bad Idea.” *Agronomy Journal* 116 (3): 1371–81. <https://doi.org/10.1002/agj2.21570>.
- Knuth, Donald E. 1984. “Literate Programming.” *Comput. J.* 27 (2): 97–111. <https://doi.org/10.1093/comjnl/27.2.97>.