Working with Factors in R

Model parameterizations, contrasts, inferences on means, etc.

Dan Hall, Director of the SCC

Department of Statistics
Franklin College of Arts and Sciences
Statistical Consulting Center
UNIVERSITY OF GEORGIA

Table of Contents

Related Resources

- Companion videos for this talk can be found here `https://kaltura.uga.edu/media/t/1_cka61gmn` and here `https://kaltura.uga.edu/media/t/1_nkujgc63`.
- A companion script, factors.R, can be found here: `https://tinyurl.com/2m65myr5`.

Introduction

- Categorical variables are often called classification variables or factors, especially when used as explanatory variables in statistical models.
- Examples abound: sex (male, female), treatment (drug A, drug B, placebo), operating system (Windows, Linux, MacOS), etc.
- Factors are ubiquitous in data science and understanding how to use them is fundamental to the practice of statistics.
- So a discussion of factors would seem only suitable for complete novices.
- However, there are some tricky issues, especially in R where the `factor` object class can be quite confusing.
- Moreover, students usually learn to work with discrete explanatory variables (i.e., factors) after and less thoroughly than covariates.

Introduction

- Categorical variables are often called classification variables or factors, especially when used as explanatory variables in statistical models.
- Examples abound: sex (male, female), treatment (drug A, drug B, placebo), operating system (Windows, Linux, MacOS), etc.
- Factors are ubiquitous in data science and understanding how to use them is fundamental to the practice of statistics.
- So a discussion of factors would seem only suitable for complete novices.
- However, there are some tricky issues, especially in R where the `factor` object class can be quite confusing.
- Moreover, students usually learn to work with discrete explanatory variables (i.e., factors) after and less thoroughly than covariates.

Introduction

- Categorical variables are often called classification variables or factors, especially when used as explanatory variables in statistical models.
- Examples abound: sex (male, female), treatment (drug A, drug B, placebo), operating system (Windows, Linux, MacOS), etc.
- Factors are ubiquitous in data science and understanding how to use them is fundamental to the practice of statistics.
- So a discussion of factors would seem only suitable for complete novices.
- However, there are some tricky issues, especially in R where the `factor` object class can be quite confusing.
- Moreover, students usually learn to work with discrete explanatory variables (i.e., factors) after and less thoroughly than covariates.

Introduction

- Categorical variables are often called classification variables or factors, especially when used as explanatory variables in statistical models.
- Examples abound: sex (male, female), treatment (drug A, drug B, placebo), operating system (Windows, Linux, MacOS), etc.
- Factors are ubiquitous in data science and understanding how to use them is fundamental to the practice of statistics.
- So a discussion of factors would seem only suitable for complete novices.
- However, there are some tricky issues, especially in R where the `factor` object class can be quite confusing.
- Moreover, students usually learn to work with discrete explanatory variables (i.e., factors) after and less thoroughly than covariates.

Introduction

- Categorical variables are often called classification variables or factors, especially when used as explanatory variables in statistical models.
- Examples abound: sex (male, female), treatment (drug A, drug B, placebo), operating system (Windows, Linux, MacOS), etc.
- Factors are ubiquitous in data science and understanding how to use them is fundamental to the practice of statistics.
- So a discussion of factors would seem only suitable for complete novices.
- However, there are some tricky issues, especially in R where the `factor` object class can be quite confusing.
- Moreover, students usually learn to work with discrete explanatory variables (i.e., factors) after and less thoroughly than covariates.

Introduction

- Categorical variables are often called classification variables or factors, especially when used as explanatory variables in statistical models.
- Examples abound: sex (male, female), treatment (drug A, drug B, placebo), operating system (Windows, Linux, MacOS), etc.
- Factors are ubiquitous in data science and understanding how to use them is fundamental to the practice of statistics.
- So a discussion of factors would seem only suitable for complete novices.
- However, there are some tricky issues, especially in R where the `factor` object class can be quite confusing.
- Moreover, students usually learn to work with discrete explanatory variables (i.e., factors) after and less thoroughly than covariates.

Do you need to listen to a talk about factors?

- In R, do you understand the `factor` class? What is the difference between the `levels` and the `labels` of a factor? What is the mode of a factor? How do you convert an object to or from the factor class? How do you recode a factor (e.g., change, combine, or split levels; change reference category; reorder)?
- Do you know the difference between ordered and unordered factors in R?
- Are you comfortable with alternative parameterizations of a model involving factors? Do you know how to induce different parameterizations via *contrasts* in R?
- Do you know how to get simultaneous confidence intervals for a set (or family) of means, or to test a family of contrasts and adjust those inferences for multiple comparisons?
- Do you know how to test a custom contrast among a set of means (e.g., corresponding to the levels of a factor)?

Do you need to listen to a talk about factors?

- In R, do you understand the `factor` class? What is the difference between the `levels` and the `labels` of a factor? What is the mode of a factor? How do you convert an object to or from the factor class? How do you recode a factor (e.g., change, combine, or split levels; change reference category; reorder)?
- Do you know the difference between ordered and unordered factors in R?
- Are you comfortable with alternative parameterizations of a model involving factors? Do you know how to induce different parameterizations via *contrasts* in R?
- Do you know how to get simultaneous confidence intervals for a set (or family) of means, or to test a family of contrasts and adjust those inferences for multiple comparisons?
- Do you know how to test a custom contrast among a set of means (e.g., corresponding to the levels of a factor)?

Do you need to listen to a talk about factors?

- In R, do you understand the `factor` class? What is the difference between the `levels` and the `labels` of a factor? What is the mode of a factor? How do you convert an object to or from the factor class? How do you recode a factor (e.g., change, combine, or split levels; change reference category; reorder)?
- Do you know the difference between ordered and unordered factors in R?
- Are you comfortable with alternative parameterizations of a model involving factors? Do you know how to induce different parameterizations via *contrasts* in R?
- Do you know how to get simultaneous confidence intervals for a set (or family) of means, or to test a family of contrasts and adjust those inferences for multiple comparisons?
- Do you know how to test a custom contrast among a set of means (e.g., corresponding to the levels of a factor)?

Do you need to listen to a talk about factors?

- In R, do you understand the `factor` class? What is the difference between the `levels` and the `labels` of a factor? What is the mode of a factor? How do you convert an object to or from the factor class? How do you recode a factor (e.g., change, combine, or split levels; change reference category; reorder)?
- Do you know the difference between ordered and unordered factors in R?
- Are you comfortable with alternative parameterizations of a model involving factors? Do you know how to induce different parameterizations via *contrasts* in R?
- Do you know how to get simultaneous confidence intervals for a set (or family) of means, or to test a family of contrasts and adjust those inferences for multiple comparisons?
- Do you know how to test a custom contrast among a set of means (e.g., corresponding to the levels of a factor)?

Do you need to listen to a talk about factors?

- In R, do you understand the `factor` class? What is the difference between the `levels` and the `labels` of a factor? What is the mode of a factor? How do you convert an object to or from the factor class? How do you recode a factor (e.g., change, combine, or split levels; change reference category; reorder)?

- Do you know the difference between ordered and unordered factors in R?

- Are you comfortable with alternative parameterizations of a model involving factors? Do you know how to induce different parameterizations via *contrasts* in R?

- Do you know how to get simultaneous confidence intervals for a set (or family) of means, or to test a family of contrasts and adjust those inferences for multiple comparisons?

- Do you know how to test a custom contrast among a set of means (e.g., corresponding to the levels of a factor)?

Do you need to listen to a talk about factors?

- Do you know how to use orthogonal polynomial contrasts for ordered factors? How about when the levels of the factor are not equally spaced?
- Do you known the difference between Type I, II, III tests, their proper usage and implementation in R?
- Do you know the difference between *main effects* and *simple effects*?
- Do you know the difference between marginal means, joint means, and raw means?
- Do you know how to estimate joint and/or marginal means from a fitted model and do inferences on them properly?
- Omitting the intercept in an ANOVA model doesn't alter the model (only its parameterization), but how does it change ANOVA table F tests?

Do you need to listen to a talk about factors?

- Do you know how to use orthogonal polynomial contrasts for ordered factors? How about when the levels of the factor are not equally spaced?
- Do you known the difference between Type I, II, III tests, their proper usage and implementation in R?
- Do you know the difference between *main effects* and *simple effects*?
- Do you know the difference between marginal means, joint means, and raw means?
- Do you know how to estimate joint and/or marginal means from a fitted model and do inferences on them properly?
- Omitting the intercept in an ANOVA model doesn't alter the model (only its parameterization), but how does it change ANOVA table F tests?

Do you need to listen to a talk about factors?

- Do you know how to use orthogonal polynomial contrasts for ordered factors? How about when the levels of the factor are not equally spaced?
- Do you known the difference between Type I, II, III tests, their proper usage and implementation in R?
- Do you know the difference between *main effects* and *simple effects*?
- Do you know the difference between marginal means, joint means, and raw means?
- Do you know how to estimate joint and/or marginal means from a fitted model and do inferences on them properly?
- Omitting the intercept in an ANOVA model doesn't alter the model (only its parameterization), but how does it change ANOVA table F tests?

Do you need to listen to a talk about factors?

- Do you know how to use orthogonal polynomial contrasts for ordered factors? How about when the levels of the factor are not equally spaced?
- Do you known the difference between Type I, II, III tests, their proper usage and implementation in R?
- Do you know the difference between *main effects* and *simple effects*?
- Do you know the difference between marginal means, joint means, and raw means?
- Do you know how to estimate joint and/or marginal means from a fitted model and do inferences on them properly?
- Omitting the intercept in an ANOVA model doesn't alter the model (only its parameterization), but how does it change ANOVA table F tests?

Do you need to listen to a talk about factors?

- Do you know how to use orthogonal polynomial contrasts for ordered factors? How about when the levels of the factor are not equally spaced?
- Do you known the difference between Type I, II, III tests, their proper usage and implementation in R?
- Do you know the difference between *main effects* and *simple effects*?
- Do you know the difference between marginal means, joint means, and raw means?
- Do you know how to estimate joint and/or marginal means from a fitted model and do inferences on them properly?
- Omitting the intercept in an ANOVA model doesn't alter the model (only its parameterization), but how does it change ANOVA table F tests?

Do you need to listen to a talk about factors?

- Do you know how to use orthogonal polynomial contrasts for ordered factors? How about when the levels of the factor are not equally spaced?
- Do you known the difference between Type I, II, III tests, their proper usage and implementation in R?
- Do you know the difference between *main effects* and *simple effects*?
- Do you know the difference between marginal means, joint means, and raw means?
- Do you know how to estimate joint and/or marginal means from a fitted model and do inferences on them properly?
- Omitting the intercept in an ANOVA model doesn't alter the model (only its parameterization), but how does it change ANOVA table F tests?

The `factor` Class in R—Basics

- Factors in R created with `factor()` function.

```
(sex <- rep(c("M","F"), each=5))  # a character vector, 5 Male, 5 Female
```

```
 [1] "M" "M" "M" "M" "M" "F" "F" "F" "F" "F"
```

```
(sexFac <- factor(sex))           # a factor
```

```
 [1] M M M M M F F F F F
Levels: F M
```

- Their mode is `numeric`, but `is.numeric()` returns `FALSE`.

```
mode(sexFac) # numeric? Really?
```

```
[1] "numeric"
```

```
is.numeric(sexFac)
```

```
[1] FALSE
```

- They are numeric vectors with a `levels` attribute. The elements of a factor are the indices of their levels.

```
attributes(sexFac)
```

```
$levels
[1] "F" "M"

$class
[1] "factor"
```

```
as.numeric(sexFac)
```

```
 [1] 2 2 2 2 2 1 1 1 1 1
```

The `factor` Class in R—Basics

- Can specify the levels and, optionally, labels to print in place of levels. Specifying levels can be useful to put them in a desired order and for other reasons.

```
sexFac # levels alphabetically ordered by default
```

```
 [1] M M M M M F F F F F
Levels: F M
```

```
(sexFac.MF <- factor(sex,levels=c("M","F"), labels=c("Male","Female"))) #set diff't order
```

```
 [1] Male   Male   Male   Male   Male   Female Female Female Female Female
Levels: Male Female
```

```
table(sexFac)    # a freq distribution for sexFac
```

```
sexFac
F M
5 5
```

```
table(sexFac.MF) # note the difference in the order of levels
```

```
sexFac.MF
  Male Female
     5      5
```

The `factor` Class in R—Basics

- Use meaningful levels or, if not, use labels!
- Specifying levels induces an ordering, but that doesn't mean the factor is an *ordered* factor.
- An ordered factor is a special type of factor.
  - Order of levels stored with object.
  - `min()`, `max()` `<`, `>` can be used to compare ordered factors.
  - More importantly, we may wish to parameterize ordered factors differently than unordered factors in models.

```
opin <- c(-1,0,-1,1,-1)
(opinFac <- factor(opin,levels=-1:1,labels=c("disagree","neutral","agree")))
```

```
[1] disagree neutral  disagree agree    disagree
Levels: disagree neutral agree
```

```
(opinOrd <- factor(opin,levels=-1:1,labels=c("disagree","neutral","agree"),ordered=TRUE))
```

```
[1] disagree neutral  disagree agree    disagree
Levels: disagree < neutral < agree
```

```
max(opinOrd)  # max(opinFac) gives error
```

```
[1] agree
Levels: disagree < neutral < agree
```

The `factor` Class in R—Basics

- Until recently, `read.table()`, `data.frame()`, etc. automatically coerced character vectors into factors, unless option `stringsAsFactors` set to `FALSE`.
  - Thankfully, as of R 4.0.0, this behavor has been changed.
- A categorical variable is sometimes more conveniently manipulated as a factor, other times as a non-factor (e.g., numeric or character).
  - Keep two versions (e.g., keep both `opin` and `opinFac`).
- Be careful turning a factor into a "non-factor".

```
doseFac <- factor(c(0,500,500,0,1000))
as.character(doseFac)             # character version
```

```
[1] "0"    "500"  "500"  "0"    "1000"
```

```
as.numeric(as.character(doseFac)) # Right
```

```
[1]    0  500  500    0 1000
```

```
as.numeric(doseFac)               # Wrong!
```

```
[1] 1 2 2 1 3
```

Recoding factors

Many good tools in `forcats` package (part of tidyverse).

- Collapsing levels:
  - `fct_collapse()`.
- Expanding (adding) levels:
  - `fct_expand()`.
- Dropping levels:
  - `fct_drop()`.
- Re-ordering levels:
  - `fct_inorder()`, `fct_infreq()`, `fct_inseq()`, `fct_relevel()`, `fct_reorder()`, `fct_rev()`, `fct_shift()`.
- Combine levels based on frequency of occurrence:
  - `fct_lump_min()`, `fct_lump_prop()`, `fct_lump_n()`, `fct_lump_lowfreq()`.
- Recode levels:
  - `fct_recode()` (can be used to collapse levels).
- Create a factor from combination of levels of two factors:
  - `fct_cross()`.
- Examples in script, factors.R.

Models with Factors—Parameterizations and Contrasts

- Presentation focuses on linear models, but extends to GLMs, others.
- Linear models with factors are known as ANOVA and ANCOVA models.
- Such models can be equivalently formulated in multiple ways depending on how the factor(s) are handled.
  - These alternative approaches are different parameterizations of a model, not different models.

Example—One-way ANOVA Model:

- Two common parameterizations:

$$\text{Cell means version:} \quad y_{ij} = \mu_i + e_{ij}$$

$$\text{Effects version:} \quad y_{ij} = \mu + \alpha_i + e_{ij}$$

- μ_is are treatment means.
- α_is are treatment effects up/down from a constant μ.

Models with Factors—Parameterizations and Contrasts

$$\text{Cell means version:} \quad y_{ij} = \mu_i + e_{ij}$$

$$\text{Effects version:} \quad y_{ij} = \mu + \alpha_i + e_{ij},$$

$i = 1, ..., g$ (treatments), $j = 1, ..., n_i$ (replicates). Suppose $g = 4$.

- Models are equivalent. Four means for 4 treatments.
- Effects model is *overparameterized*. It captures 4 means with 5 parameters: $\mu, \alpha_1, \alpha_2, \alpha_3, \alpha_4$.
  - One parameter is redundant.
  - Redundancy does not have to be removed, but R always does.
  - Redundancy can be removed in several different ways. We could:
    - set $\mu = 0$ (becomes the cell means model)
    - set $\alpha_1 = 0$ (μ becomes mean in trt 1)
    - set $\alpha_i = 0$ for any i (μ becomes mean in trt i)
    - constrain $\alpha_1 + \alpha_2 + \alpha_3 + \alpha_4 = 0$ (μ becomes mean of the trt means, or grand mean).
  - Choices yield different parameterizations of the same model.
  - In R, choice is controlled via *contrasts*.

Models with Factors—Parameterizations and Contrasts

- R uses contrasts (contrast matrices) applied to each factor to avoid overparameterization.
  - *contrast matrices* is a misnomer. Should be called *coding matrices* (as in Venables' `codingMatrices` package).
- Contrast matrices remove parameter redundancy by coding the columns of the *model matrix* corresponding to a factor so they are not linearly dependent with the column for the constant term μ.
- R "contrast" functions implement these recodings:
  - `contr.treatment(g,i)`: equivalent to setting $\alpha_i = 0$.
    - Primary argument g is number of levels of the factor.
    - Default value of `i` is 1.
    - `contr.SAS(g)` is wrapper for `contr.treatment(g,g)`
  - `contr.sum(g)`: equivalent to "sum-to-zero" constraint, $\sum_i \alpha_i = 0$.
  - `contr.helmert(g)`: parameters become (orthogonal) contrasts b/w 2nd and 1st level, b/w 3rd level and avg of levels 1 & 2, etc.
  - `contr.poly(g)`: parameterizes ordered factor effects in terms of orthogonal polynomial contrasts (linear effect, quadratic effect,...,$g-1$st order effect).

Models with Factors—Parameterizations and Contrasts

How do I set the contrasts for the factors in my model?

- `contrasts()` function can query or change contrasts for a factor.
- There's a system option that sets contrasts to be used for unordered and ordered factors unless otherwise specifed.

```
options("contrasts")
```

```
$contrasts
        unordered           ordered
"contr.treatment"      "contr.poly"
```

```
sexFac <- factor(rep(c("M","F"), each=5)) # 5 Male, 5 Female
contrasts(sexFac) # returns the contrasts used by default
```

```
  M
F 0
M 1
```

```
# Change to sum-to-zero contrasts:
contrasts(sexFac) <- contr.sum(2) # 2 because sexFac has 2 levels
op <- options(contrasts=c("contr.SAS","contr.poly")) # change contrasts option & store previous value in op
ageFac <- factor(rep(c("Adult","Child"), times=5)) # A,C,A,C,...
contrasts(ageFac) # returns the contrasts used by default
```

```
      Adult
Adult     1
Child     0
```

```
options(op) # restore defaults
```

Models with Factors—Parameterizations and Contrasts

How do I set the contrasts for the factors in my model?

- Many model-fitting functions have a `contrasts=` option.

```
lm(rnorm(10)~sexFac+ageFac,contrasts=list(sexFac="contr.treatment",
                                          ageFac="contr.SAS"))
```

```
Call:
lm(formula = rnorm(10) ~ sexFac + ageFac, contrasts = list(sexFac = "contr.treatment",
    ageFac = "contr.SAS"))

Coefficients:
(Intercept)      sexFacM  ageFacAdult
    -0.2043       0.3771      -0.1152
```

Models with Factors—Parameterizations and Contrasts

In linear models, any of the standard inferences (tests, confidence intervals) we would wish to perform can be done in any parameterization of the model.

- Some parameterizations more convenient for some purposes than others, but the parameterization does not limit what we can do.
- Parameterization is essentially arbitrary, but that does not mean it doesn't matter.
- We do need to understand the parameterization we are working in.
- For Type III tests on main effects to be constructed correctly (e.g., by `Anova()` in `car` package) in models with interactions for unbalanced data, must use `contr.sum()`.

Example–Walking Age

- Infants randomized to 4 treatments to stimulate a walking response in newborns:
  - control, no exercise, passive exercise, active exercise.
- Response is the age (in mos.) when child first began to walk.
- See factors.R and video linked here.

Highlights:

- Note the default choices made when reading data, setting up factors.
- One-way ANOVA model fitted with `aov()`, a wrapper for `lm()`.
  - Fitted model object has two classes `aov` and `lm`.
  - Default summary for `aov` object is an ANOVA table. Parameter estimates less important. Parameterization is arbitrary.
- Treatment means and comparisons among them of main interest.
- Lots of parameterizations considered for `group`, the treatment factor.
  - A parameterization can be chosen so that (regression) coefficients of the model are quantities of interest.
- For an ordered factor, parameterization in terms of polynomial contrasts or consecutive differences may be appealing.
  - Use `scores=` option in `contr.poly()` for unequally spaces factor levels.

Example–Walking Age

Highlights (continued):

- Models with different parameterizations all give same ANOVA table, F test for group.
  - Exception is when model lacks an intercept. Then
    - the choice of contrast matrix (factor coding) does not matter,
    - regression parameters are treatment means,
    - and ANOVA table F test differs. Now it tests that all means are equal to 0, an uninteresting hypothesis!
    - Default calculation of R^2 is incorrect when model lacks an intercept.
- Regardless of parameterization, `emmeans` package can be used to estimate means and do inferences on them.
  - `emmeans()` function can give means and confidence intervals for them.
  - `contrast()` function (not `contrasts()`!) can estimate and test contrasts among means.

Inferences on Means—The `emmeans` Package

- An ANOVA model is a framework for inference on the effects of factors. Usually, want to do inference on the treatment means.
  - We do model-based inference.
- In one-way model, basic question is, *Are all treatment means the same?*
- Just a starting point. If not all equal,
  - which ones differ and by how much?
  - point estimates and intervals for each treatment mean;
  - Usually want inference on several quantities. How do we control type I error for all these inferences?
- Extremely useful tools for these tasks in the `emmeans` package.

Inferences on Means—The `emmeans` Package

The `emmeans` Package:

- Gives estimates of treatment means, associated SEs, and CIs based on a fitted model.
  - In a multifactor model, can get
    - joint means (at combinations of levels of the factors),
    - or marginal means (at each level of one factor, averaging over others).
- Can estimate, give CIs for, and test hypotheses on contrasts and other linear combinations of means.
- Implements multiplicity adjustments to control type I error rate/simultaneous coverage probability when doing multiple inferences.
- Can plot means and associated CIs, including interaction plots in multi-factor models.
- Works with a variety of different model classes (`lm`s and `aov`s, `mlm`s, `glm`s, `lmerMod`s, `glmerMod`s, `gam`'s, many others).
- Based on LSMEANS statement in SAS (package originally called `lsmeans`).

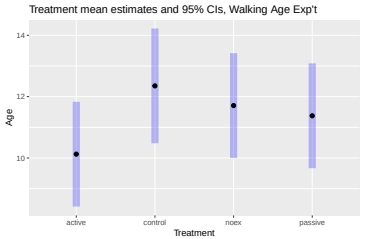
Inferences on Means—The emmeans Package

Example—Walking Age

```
walk.m1 <- aov(age~group,data=walkdata)
(walk.m1.emm <- emmeans(walk.m1,specs= ~ group,adjust="bonferroni"))
```

```
 group    emmean    SE df lower.CL upper.CL
 active     10.1 0.619 19     8.42     11.8
 control    12.3 0.678 19    10.48     14.2
 noex       11.7 0.619 19    10.00     13.4
 passive    11.4 0.619 19     9.67     13.1

Confidence level used: 0.95
Conf-level adjustment: bonferroni method for 4 estimates
```

```
plot(walk.m1.emm,horizontal=F,xlab="Age",ylab="Treatment")+ggtitle("Treatment mean estimates and 95% CIs, Walking Age Exp't")
```



Inferences on Means—The emmeans Package

Example—Walking Age (Continued)

```
# Dunnett intervals and tests for each pairwise diff with control
# (for 1-tailed tests, 1-sided intervals add argument: side="<"):
(diffsVsControl <- contrast(walk.m1.emm,method="trt.vs.ctrl",ref="control",infer=c(TRUE,TRUE),adjust="dunnettx"))
```

```
 contrast          estimate    SE df lower.CL upper.CL t.ratio p.value
 active - control    -2.225 0.918 19    -4.58    0.133 -2.423  0.0670
 noex - control      -0.642 0.918 19    -3.00    1.716 -0.699  0.8013
 passive - control   -0.975 0.918 19    -3.33    1.383 -1.062  0.5854

Confidence level used: 0.95
Conf-level adjustment: dunnettx method for 3 estimates
P value adjustment: dunnettx method for 3 tests
```

```
plot(diffsVsControl,xlab="Months")+ggtitle("Pairwise differences w/ control, Walking Age Exp't")
```

Inferences on Means—The emmeans Package

Example—Walking Age. Custom Contrasts:

```
levels(walkdata$group)
```

```
[1] "active"  "control" "noex"    "passive"
```

```
# contrast to compare exercise absent groups to exercise present groups:
exerciseContrast <- c(1,-1,-1,1)/2 #avg of control & noex v. avg of others
contrast(walk.m1.emm,method=list(exerciseContrast),infer=c(T,T)) #infer asks for interval and test
```

```
 contrast                estimate    SE df lower.CL upper.CL t.ratio p.value
 c(0.5, -0.5, -0.5, 0.5)    -1.28 0.634 19    -2.61   0.0485 -2.016  0.0581

Confidence level used: 0.95
```

```
test(contrast(walk.m1.emm,method=list(exerciseContrast,c(1,0,0,-1),c(0,1,-1,0))),joint=TRUE) # joint F test (same as main effect test)
```

```
 df1 df2 F.ratio p.value
   3  19   2.142 0.1285
```

```
anova(walk.m1)
```

```
Analysis of Variance Table

Response: age
          Df Sum Sq Mean Sq F value Pr(>F)
group      3 14.778  4.9259  2.1422 0.1285
Residuals 19 43.690  2.2995
```

Multi-factor Models

Example—Soybean Weeds

An experiment in a randomized complete block design (RCBD) was conducted to study effects of soybean variety and herbicide use on weed biomass.

- Trt factors: Variety (16 levs); Herbicide (never, @2 wks, @4 wks).
- Blocking factor: Location (Rosemount, St.Paul)

All 16*3=48 treatments randomized to plots in each location.

- Do herbicide effects differ across varieties (interaction).
- If not, what are herbicide effects (*main effects*, averaged over variety).
- If so, what are herbicide effects for each variety (*simple effects*).

Potentially interested in two types of means:

- Joint means for each of the 48 treatments.
- Marginal means for each level of herbicide (avg'd over variety).
- Marginal means for each variety (avg'd over herbicide).

Multi-factor Models

Example—Soybean Weeds (continued)

Model:

$$y_{ijk} = \underbrace{\mu + \alpha_i + \beta_j + \gamma_{ij} + \tau_k}_{\equiv \mu_{ijk}} + e_{ijk}, \quad i = 1, ..., 16; j = 1, 2, 3; k = 1, 2.$$

- Joint means: $\bar{\mu}_{ij\cdot}$.
- Marginal means:
  - $\bar{\mu}_{i\cdot\cdot}$ mean for ith variety,
  - $\bar{\mu}_{\cdot j\cdot}$ mean for jth level of herbicide.
- In fact, all these means are averaged over block (marginal, in a sense).
- Estimates of above quantities obtained from the fitted model.

Multi-factor Models

Model-based Means vs. Raw Means:

- Raw means (Soybean Weeds example):
  - Joint: $\bar{y}_{ij\cdot}$ (simple avg of data in i, jth treatment).
  - Marginal: $\bar{y}_{i\cdot\cdot}$ and $\bar{y}_{\cdot j\cdot}$ (simple avg of data at each level of a factor).
- These ignore the model (a bad idea).
  - The model "adjusts for" nuisance variables (e.g., the blocking factor). We want our estimates to reflect such adjustments!
- Sometimes model-based estimates of means and raw means agree, but not in general (e.g., unbalanced designs).
  - Even when they agree, inferences typically don't agree.
- Validate the model and then use it as a framework for inference!

Multi-factor Models—Soybean Weeds (continued)

Examine the data and fit the model:

```
str(weedDat) # returns structure of the data frame containing the data.
```

```
'data.frame':   96 obs. of  7 variables:
 $ variety: chr  "Parker" "Lambert" "M89-792" "Sturdy" ...
 $ weeds  : num  750 870 1090 1110 1150 1210 1330 1630 1660 2210 ...
 $ herb   : num  2 2 2 2 2 2 2 2 2 2 ...
 $ loc    : chr  "R" "R" "R" "R" ...
 $ locFac : Factor w/ 2 levels "Rosemount","St.Paul": 1 1 1 1 1 1 1 1 1 1 ...
 $ herbFac: Factor w/ 3 levels "none","2 weeks",..: 2 2 2 2 2 2 2 2 2 2 ...
 $ varFac : Factor w/ 16 levels "Archer","Lambert",..: 15 2 9 16 14 5 10 11 7 1 ...
```

```
weeds.m1 <- aov(weeds~locFac+varFac+herbFac+varFac:herbFac,data=weedDat)
summary(weeds.m1)
```

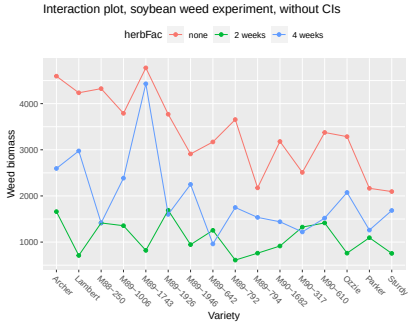
```
               Df   Sum Sq  Mean Sq F value   Pr(>F)
locFac          1 50634150 50634150  42.453 4.45e-08 ***
varFac         15 25587029  1705802   1.430    0.173
herbFac         2 85183540 42591770  35.710 3.71e-10 ***
varFac:herbFac 30 22426627   747554   0.627    0.912
Residuals      47 56057750  1192718
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

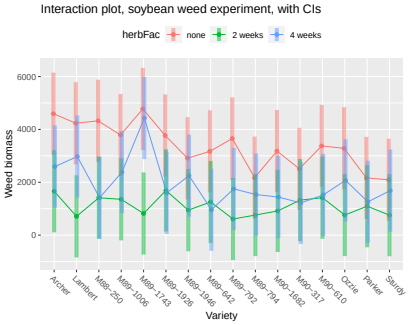
- Very little evidence of interaction.

Multi-factor Models—Soybean Weeds (continued)

Get estimated means from the model and plot the non-significant interaction:

```
weeds.m1.emm <- emmeans(weeds.m1,specs= list(jointMeans=       ~ varFac:herbFac,
                                             varietyMargMeans= ~ varFac,
                                             herbMargMeans=    ~ herbFac))
emmip(weeds.m1,herbFac~varFac) + theme(legend.position="top",axis.text.x= element_text(hjust=0,angle= -45)) +labs(title="Interaction plot, s
emmip(weeds.m1,herbFac~varFac,CIs=T) + theme(legend.position="top",axis.text.x= element_text(hjust=0,angle= -45)) +labs(title="Interaction p
```

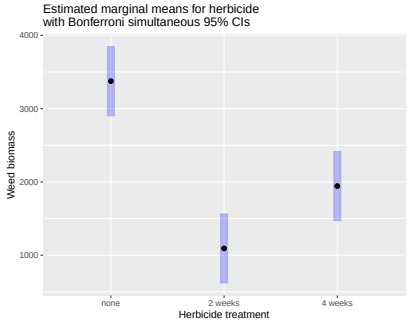


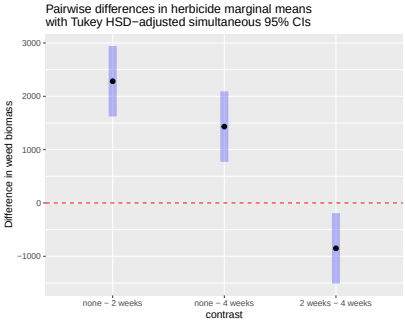


Multi-factor Models—Soybean Weeds (continued)

Comparisons among marginal means are reasonable with no interaction.

```
# Bonferroni-adjusted CIs for each herbicide marginal mean:
plot(summary(weeds.m1.emm$herbMargMeans,adjust="bonferroni"),horizontal=FALSE) +
  labs(x="Weed biomass",y="Herbicide treatment",title="Estimated marginal means for herbicide\nwith Bonferroni simultaneous 95% CIs")
# Tukey HSD adjusted confidence intervals for each pairwise difference in marginal herbicide means:
plot(contrast(weeds.m1.emm$herbMargMean,method="pairwise"),
     horizontal=FALSE) + geom_vline(xintercept=0,color="red",linetype=2) +
  labs(x="Difference in weed biomass",title="Pairwise differences in herbicide marginal means\nwith Tukey HSD-adjusted simultaneous 95% CIs"
```



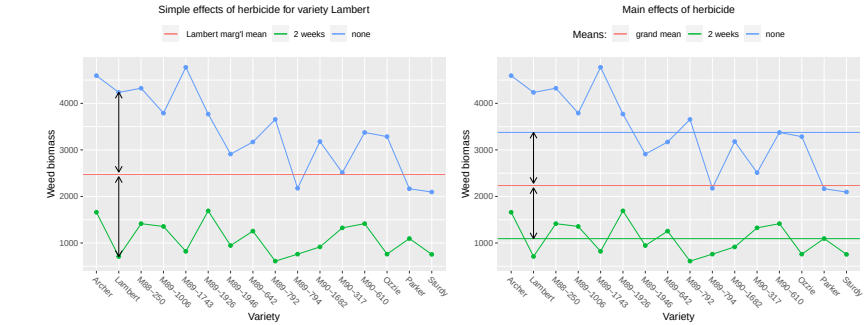


Multi-factor Models—Simple vs. Main Effects.

An "effect" in this context: for each level of a factor, the deviation up or down from the mean across all levels of the factor.

- Simple effects are in terms of joint means. E.g., differences among levels of herbicide within each variety.
- Main effects are in terms of marginal means. E.g., differences among marginal means of herbicide averaged over variety.

To simplify, suppose only 2 herbicide levels: *none*, *2 weeks*.



Multi-factor Models—Simple Effects and Effect Slices

In presence of significant interaction, main effects (usually) not appropriate.

- Interaction implies simple effects differ, so main effect tests and comparisons of marginal means are too simplistic. They simplify or even distort the true story.
- Instead, test hypothesis of no simple effects of one factor at each level of the other (i.e., tests of **effect slices**),
- and/or use joint means to make specific comparisons across levels of one factor within each level of the other.

Soybean Weeds. Herbicide effects sliced by variety:

```
ts <- test(contrast(weeds.m1.emm$jointMeans,simple="herbFac"),joint=TRUE)
ts[1:3,] # first 3 of 16 tests
```

```
   varFac df1 df2 F.ratio     p.value note
1  Archer   2  47   3.770 0.030312283    d
4 Lambert   2  47   5.350 0.008065586    d
7 M88-250   2  47   4.733 0.013403684    d
```

Multi-factor Models—Simple Effects and Effect Slices

In the presence of interaction, testing effect slices is an appropriate alternative to testing main effects.

- In the Soybean Weed example, that replaces a single 2-df test by 16 tests. Doesn't that inflate Type I error rate?
- Yes! So apply some multiplicity adjustment.

Here we use `p.adjust()` function to add p-values adjusted by

- Holm's method (controls familywise error rate), and
- Benjamini-Hochberg method (controls false discovery rate).

```
ts$fdr_adjpval <- format.pval(p.adjust(ts$p.value,method="fdr"),digits=3,eps=.0001)
ts$holm_adjpval <- format.pval(p.adjust(ts$p.value,method="holm"),digits=3,eps=.0001)
ts[1:6,] # first 6 of 16 tests
```

```
     varFac df1 df2 F.ratio      p.value note fdr_adjpval holm_adjpval
1    Archer   2  47   3.770 0.0303122833    d      0.0970       0.3637
4   Lambert   2  47   5.350 0.0080655864    d      0.0645       0.1210
7    M88-250  2  47   4.733 0.0134036844    d      0.0715       0.1877
10 M89-1006   2  47   2.505 0.0925044831    d      0.1693       0.9067
13 M89-1743   2  47   8.047 0.0009876509    d      0.0158       0.0158
16 M89-1926   2  47   2.527 0.0906722193    d      0.1693       0.9067
```

Multi-factor Models—Type I, II and III Tests

For data from experiments with crossed treatment factors, we typically include main effects and all interactions among the factors.

- E.g., the Soybean Weed example, herbicide is crossed with variety.
  - We include `herbFac`, `varFac` and `herbFac:varFac` to assess whether factors interact and, if not, how each factor alone affects the mean response.

We **do not** test interaction and, if not significant, drop it and re-fit.

- The model reflects the design, allowing means for every treatment. It's a framework for inference on all questions about how treatment means differ.

But in a model that allows for interaction, how do we test main effects?
When data are unbalanced, there is more than one answer to that question!

- Type I (sequential) tests
- Type II tests
- Type III tests

Multi-factor Models—Type I, II and III Tests

- Type I: model is built up one term at a time and significance of each effect is relative to the previous model that lacked that term.
  - This approach rarely tests interesting hypotheses on main effects.
- Type II: the significance of an effect is based on how much it improves a *hierarchical model*[1] that lacks the effect.
  - Each effect is tested based on adjusting for all other effects of the same order and higher order effects not containing the effect.
  - For main effects in an unbalanced two-way model, this tests a null hypothesis defined in terms of a weighted average of treatment means, weighted by relative sample sizes.
  - Unless sensible to average over factor levels in proportion to their sample sizes[2], this is unappealing.

[1] Hierarchy principle: interactions aren't allowed unless subordinate effects also included.

[2] Might be sensible if replication is proportional to population prevalence so that factor levels aren't equally relevant to the population.

Multi-factor Models—Type I, II and III Tests

- Type III: significance of an effect is based on its effect when dropped from the model.
  - For main effects in a two-way model, equivalent to testing if the marginal means for the factor are all equal.
  - **But the previous interpretation only applies when sum-to-zero constraints are used to parameterize the model!**
    - Otherwise, the type 3 tests will be nonsensical.

In R,

- `anova(fittedModel)` and `summary(fittedModel)` give Type I tests.
  - Often inappropriate and can be very different from Types II and III.
- For Types II and III use `Anova(fittedModel,type=)` from `car` package.
- Type III is usually the right choice for experimental data, **but must use sum-to-zero constraints** when fitting the model.

Multi-factor Models—Type I, II and III Tests

Soybean Weed Example:

- Suppose varieties 3-16 in Rosemount and 1-14 in St.Paul.
  - This gives an unbalanced design with different treatments in each block.

```
# Fit the model to the unblanaced data. All 3 modelsbelow are equivalent.
# Important to use 3rd one for Type III tests
weeds.m1.un  <- aov(weeds~locFac+varFac*herbFac,data=weedDat.un)
weeds.m1a.un <- aov(weeds~varFac*herbFac+locFac,data=weedDat.un)
weeds.m1b.un <- aov(weeds~locFac+varFac*herbFac,data=weedDat.un,contrasts=list(locFac=contr.sum,varFac=contr.sum,herbFac=contr.sum))
# Type I tests depend on the order of the terms in the formula within the aov() function (not good!).
anova(weeds.m1.un);anova(weeds.m1a.un)
```

```
Analysis of Variance Table

Response: weeds
              Df   Sum Sq  Mean Sq F value    Pr(>F)
locFac         1 43545600 43545600 48.5258 4.226e-08 ***
varFac        15 24332306  1622154  1.8077    0.07408 .
herbFac        2 70519779 35259889 39.2925 1.130e-09 ***
varFac:herbFac 30 32559288  1085310  1.2094    0.29245
Residuals     35 31407928   897369
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Analysis of Variance Table

Response: weeds
              Df   Sum Sq  Mean Sq F value    Pr(>F)
varFac        15 37275633  2485042  2.7693  0.006504 **
herbFac        2 70519779 35259889 39.2925 1.130e-09 ***
locFac         1 30602272 30602272 34.1022 1.256e-06 ***
varFac:herbFac 30 32559288  1085310  1.2094  0.292454
Residuals     35 31407928   897369
```

Multi-factor Models—Type I, II and III Tests

Soybean Weed Example (continued):

```
# Type III tests:
Anova(weeds.m1b.un,type=3)
```

```
Anova Table (Type III tests)

Response: weeds
                 Sum Sq Df  F value    Pr(>F)
(Intercept)    364844813  1 406.5715 < 2.2e-16 ***
locFac          30602272  1  34.1022 1.256e-06 ***
varFac          24332306 15   1.8077   0.07408 .
herbFac         61967112  2  34.5271 5.238e-09 ***
varFac:herbFac  32559288 30   1.2094   0.29245
Residuals       31407928 35
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

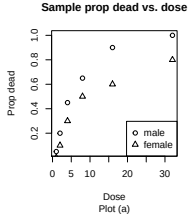
```
# Type III tests with wrong type of contrasts give garbage results, but no warnings, so must know what you are doing!:
Anova(weeds.m1.un,type=3) # WRONG!
```

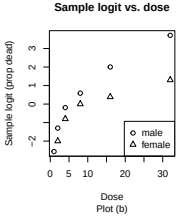
```
Anova Table (Type III tests)

Response: weeds
                 Sum Sq Df F value    Pr(>F)
(Intercept)    53572721  1 59.6997 4.523e-09 ***
locFac         30602272  1 34.1022 1.256e-06 ***
varFac         29292533 15  2.1762   0.02904 *
herbFac         8992300  2  5.0104   0.01220 *
varFac:herbFac 32559288 30  1.2094   0.29245
Residuals      31407928 35
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

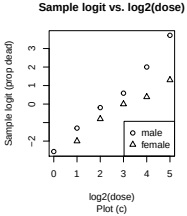
Factors and Marginal Means in Logistic Regression

A logistic regression example—Tobacco Budworms

		doseFac	1	2	4	8	16	32
sexFac	deadFac							
Male	No		19	16	11	7	2	0
	Yes		1	4	9	13	18	20
Female	No		20	18	14	10	8	4
	Yes		0	2	6	10	12	16







Factors in Logistic Regression—Tobacco Budworms

A main effects model: y_{ij}'s indep, $y_{ij} \sim \text{Bin}(20, \pi_{ij})$

$$\text{logit}(\pi_{ij}) = \underbrace{\mu + \alpha_i + \beta_j}_{\equiv \eta_{ij}}, \quad i = 1, 2; j = 1, ..., 6$$

```
bud.mainmod <- glm(cbind(dead,alive)~sexFac+doseFac,data=budworm,family=binomial(link="logit"),contrasts=list(doseFac=contr.poly))
gof(bud.mainmod) # adequate fit
```

```
D  = 5.0128, df = 5, P(>D) = 0.4143176
X2 = 3.7014, df = 5, P(>X2) = 0.5931545
```

```
# Test main effects with LR tests:
Anova(bud.mainmod,type=2,test.statistic="LR") # don't use anova()!
```

```
Analysis of Deviance Table (Type II tests)

Response: cbind(dead, alive)
         LR Chisq Df Pr(>Chisq)
sexFac     10.14  1   0.001451 **
doseFac   113.79  5  < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Factors in Logistic Regression—Tobacco Budworms

Marginal means of the linear predictor:

- Avg log odds of of death for sex i, averaged over dose: $\bar{\eta}_{i\cdot} = \frac{1}{6}(\eta_{i1} + \cdots + \eta_{i6})$.
- Avg log odds of death at dose j, averaged over sex: $\bar{\eta}_{\cdot j} = \frac{1}{2}(\eta_{1j} + \eta_{2j})$.

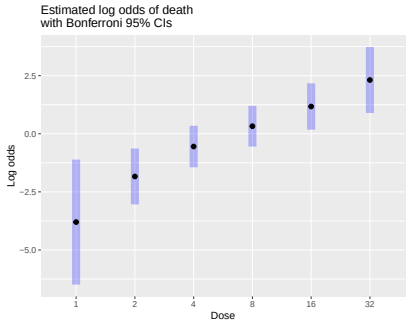
```
(bud.mainmod.emm <- emmeans(bud.mainmod,adjust="bonferroni",specs= list(doseMargMeans= ~ doseFac,sexMargMeans= ~ sexFac)))
```

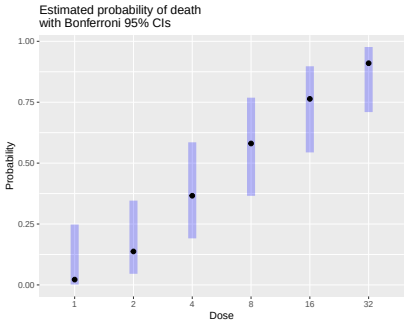
```
$doseMargMeans
 doseFac emmean    SE  df asymp.LCL asymp.UCL
 1       -3.799 1.019 Inf    -6.488    -1.110
 2       -1.837 0.455 Inf    -3.038    -0.636
 4       -0.549 0.339 Inf    -1.443     0.345
 8        0.325 0.332 Inf    -0.550     1.200
 16       1.173 0.378 Inf     0.176     2.170
 32       2.313 0.538 Inf     0.893     3.733

Results are averaged over the levels of: sexFac
Results are given on the logit (not the response) scale.
Confidence level used: 0.95
Conf-level adjustment: bonferroni method for 6 estimates

$sexMargMeans
 sexFac emmean    SE  df asymp.LCL asymp.UCL
 Male    0.149 0.285 Inf     -0.49     0.788
 Female -0.940 0.293 Inf     -1.60    -0.284

Results are averaged over the levels of: doseFac
Results are given on the logit (not the response) scale.
Confidence level used: 0.95
Conf-level adjustment: bonferroni method for 2 estimates
```

Factors in Logistic Regression—Tobacco Budworms

Plots of the marginal means for `doseFac` on log odds and probability scales (note the `type=` argument):

```
plot(bud.mainmod.emm$doseMargMeans,adjust="bonferroni",horizontal=FALSE,ylab="Dose",xlab="Log odds",type="link") +
  ggtitle("Estimated log odds of death\nwith Bonferroni 95% CIs")
plot(bud.mainmod.emm$doseMargMeans,adjust="bonferroni",horizontal=FALSE,ylab="Dose",xlab="Probability",type="response") +
  ggtitle("Estimated probability of death\nwith Bonferroni 95% CIs")
```





Factors in Logistic Regression—Tobacco Budworms

Contrasts in marginal means:

- Doses are evenly spaced on the $\log_2$ scale. Suppose we wish to test that dose effects on log odds of death are linear on that scale (recall plots).
  - Because we coded `doseFac` with `contr.poly()` contrasts, the coefficients for this factor can tell us whether linear, higher-order effects are significant:

```
arm::display(bud.mainmod,detail=TRUE) # briefer summary than given by summary() function
```

```
glm(formula = cbind(dead, alive) ~ sexFac + doseFac, family = binomial(link = "logit"),
    data = budworm, contrasts = list(doseFac = contr.poly))
            coef.est coef.se z value Pr(>|z|)
(Intercept)    0.15     0.29    0.52    0.60
sexFacFemale -1.09     0.35   -3.09    0.00
doseFac.L     4.84     0.73    6.60    0.00
doseFac.Q    -0.64     0.66   -0.97    0.33
doseFac.C     0.45     0.54    0.82    0.41
doseFac^4     0.01     0.44    0.03    0.98
doseFac^5    -0.01     0.36   -0.04    0.97
---
  n = 12, k = 7
  residual deviance = 5.0, null deviance = 124.9 (difference = 119.9)
```

Factors in Logistic Regression—Tobacco Budworms

- With this coding of `doseFac`, we can test nonlinear dose effects with joint test of the last 4 coefficients of the model. Or, under any parameterization, we can do the test on the marginal means.[3]

```
# Wald test that nonlinear effects of dose are null, computed two different ways:
coef.names <- names(coef(bud.mainmod)); lht(bud.mainmod,coef.names[-c(1:3)]) # lht stands for linear hypothesis test. From car package.
```

```
Linear hypothesis test

Hypothesis:
doseFac.Q = 0
doseFac.C = 0
doseFac^4 = 0
doseFac^5 = 0

Model 1: restricted model
Model 2: cbind(dead, alive) ~ sexFac + doseFac

  Res.Df Df  Chisq Pr(>Chisq)
1      9
2      5  4 1.4684     0.8322
```

```
test(contrast(bud.mainmod.emm$doseMargMeans,method=as.data.frame(contr.poly(6)[,-1])),joint=T) # F test w/ inf df equiv to chisq test above
```

```
 df1 df2 F.ratio p.value
   4 Inf   0.367  0.8322
```

[3] This is a Wald test, which is asymptotically equivalent to a likelihood ratio test. LRTs have somewhat better properties, so an LRT would be a better choice here and is easy to implement as `anova(update(bud.mainmod,.~sexFac+ldose),bud.mainmod,test="LRT")`.

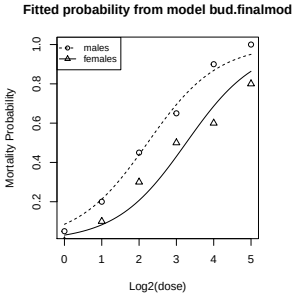
Factors in Logistic Regression—Tobacco Budworms

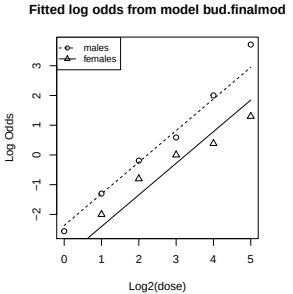
Conclusion:

- log odds of death are linear in $\log_2(\text{dose})$ with different intercepts for each sex:

$$\text{logit}(\pi_{ij}) = \alpha_i + \beta \log_2(\text{dose}).$$

```
bud.finalmod <- update(bud.mainmod,.~0+sexFac+ldose)
```





Questions?

Questions?

Thanks

- If you need assistance with R or with any statistical design or analysis task, please contact the SCC.
  - www.stat.uga/consulting
- We can help!

Thank you!