Minitab and SAS Commands for – Analysis of Variance, Design, and Regression: Linear Modeling of Unbalanced Data

---

Ronald Christensen
Department of Mathematics and Statistics
University of New Mexico

This is a work in progress!

But it should be useful as is.

Contents

Preface

This is not a general introduction to either Minitab or SAS! It is merely an introduction to generating the results in the book. As much as practicable, the chapters and sections of this guide give commands to generate results in the corresponding chapters and sections of the book. You should be able to copy the SAS code given here into a .sas file and run it. (At the moment, that is quite questionable for anything from Chapter 19 on. Not guaranteed for anything.) An exception is that you will need to modify the locations associated with data files.

As dismaying as I find the fact, it seems that relatively few students of statistics read books from beginning to end. Even I do not expect students to read a computing manual from beginning to end. As a result, I have made a positive effort to be repetitive between chapters about ideas that I think are particular important. *My ideal is that people would read the first three chapters and then skip around as needed. Chapter 3 contains the core ideas.*

Ronald Christensen
Albuquerque, New Mexico
July, 2015

Chapter 1

Introduction

There is not a lot of computing associated with Chapter 1 of the book. This chapter introduces some elementary tools related to probability and graphing and some other features that are useful.

1.1 Getting started

There was a lot of Minitab and some SAS code in the previous version of the book, which is also available on the website where this manual is located (https://www.stat.unm.edu/~fletcher/anreg.pdf). I haven't had time to work on the Minitab and SAS code as much as I have on the R code. The plan was to construct this material by combining the material from the earlier version and adapting the material from the R manual in order to illustrate Minitab and SAS. It was my intention to convert the R code that was actually used into SAS code. That has been done, but *the results have not been tested*. Minitab requires far less explication.

1.1.1 Minitab

The first order of business is to obtain access to the program. For academic users, relatively inexpensive copies of Minitab can be rented for six months or a year. Go to estore.onthehub.com or just search for the Minitab website.

A key virtue of Minitab is that it is extremely easy to use. It is menu driven and very intuitive, but, of course, the menus construct the commands needed to run the program. The command language is itself quite simple to use. As of 2021 the newest version of Minitab runs off the net and is being continually improved, hence no version numbers. The current version displays five windows. On the left is a Navigator window for selecting the output to look at in the the Session window at the center top. The center bottom displays the Worksheet containing the data. The worksheet operates rather like a spreadsheet. On the top right is a Command Line window for entering and running commands. The bottom right is a History window of the commands run. If you use the menu structure, the History window shows you the commands that the menus generated.

In the Worksheet, variables (columns of numbers) are labeled C1, C2, etc. They can also be given alpha-numeric names like y or x1. Variable names can be typed into the worksheet or read in as part of data files. The name for, say, column C10 could also be specified with a command as `name c10 'var-name'`. Commands can reference either the column number or the variable name. I often prefer to use column numbers.

For Minitab we present two different techniques for obtaining results. Sometimes we describe menu choices and sometimes we give actual Minitab commands. (Everything carried over from the previous 1996 version of the book consists of commands.) Note that when using subcommands, individual commands are separated with a semicolon and the string of commands must end with a period. One command per line. A period is not needed if subcommands are not specified. Older versions of Minitab prompted one for a command with `MTB >` and if a semicolon indicated that a subcommand was coming, Minitab provided a `SUBC>` prompt until a period was entered.

Prior to its current online existence, Minitab had a checkered history relative to directly entering

commands. The oldest versions of Minitab used to provide prompts to enter commands into the Session window. As Minitab became more menu oriented, Minitab's Session window generally hid the commands that the menus were generating but you could get Minitab to display the commands.

To enter commands in Minitab 18, with the cursor in the `Session` window, select the `Editor` menu and click on `Show Command Line`. This opens a third window (there were no Navigator or History windows) that allows you to type in commands and that shows the commands associated with menu selections.

In Minitab 16, prior to making menu choices, with the cursor in the `Session` window, select the `Editor` menu and check `Enable Commands`. The commands generated by Minitab menus will then appear in the session window prior to the display of output. The Minitab code (commands) are the lines that start with `MTB >` or `  `. This option also allows one to type commands directly into Minitab. Sometimes, especially when performing repetitive operations, it is easier to type commands than go through a series of menus.

This work was originally done on Minitab 16 for Windows, a program I really like! **I cannot recommend Minitab 18 for unbalanced ANOVA.** *It often refuses to fit the models that you ask it to fit.* Minitab 18 also made it harder read in my data files and harder to specify models. More on these issues in the appropriate places. I have no experience with Minitab 17 and 19. Since Minitab 16 is not readily available anymore, **I have little intention of ever updating or improving the Minitab commands given here.** However, I have made a few modifications based on getting access to the web based version in 2021.

1.1.2 SAS

The first order of business is to obtain access to the program. For academic users, most universities provide access to SAS either through mainframe batch computing or through rental of PC versions of the program.

I only have access to SAS in batch mode, so the commands will all take the form of `filename.sas` files. In batch mode, I type in `sas filename` (**NOT** `sas filename.sas`) and SAS produces two new files, `filename.log` and `filename.lst`. The `.log` file contains information about how SAS worked — including error messages. The `.lst` file contains the SAS output.

The SAS code usually starts with

```
options ps=60 ls=80 nodate;
```

followed by a `data` statement and always involves lines starting with `proc`.

The work was done on SAS 9.2 for linux.

1.2 Plots and probabilities

1.2.1 Minitab

Minitab plots are very easy, but somewhat restrictive. The plots in the book were all constructed in R.

Menu choices to generate something like Figure 1.2

```
graph
probability distribution plot
vary parameters
select t distribution from list
list degrees of freedom as 3 8 3000
multiple graphs
overlaid on same graph
```

3000 is used as an approximation to ∞

Menu choices to generate something like Figure 1.3

```
graph
probability distribution plot
two distributions
distribution 1: select Chi-Square from list
degrees of freedom:  enter 8 in box
distribution 2: select F from list
numerator df:  enter 3
denominator df:  enter 18
multiple graphs
in separate panels
same scales for graphs
deselect both Same Y and Same X
```

1.2.2 SAS

SAS graphics are very powerful but I do not have access to them. The only plots I have produced in SAS are crude things that could have been produced by a dot matrix printer – if you remember that ancient equipment.

1.3 Reading data

This section is pretty much restricted to reading the data files for the book. The data files are available from my website (www.stat.unm.edu/~fletcher).

1.3.1 Minitab

Minitab is not good at reading my data files. The problem is that their standard method no longer allows "free format." To read the data files for the book, open them in some editor, count the number of columns of data, and remove any variable names. The data for Table 12.3 is in `tab12-3.dat`. The file has 6 columns, so use the command

```
read c1-c6;
file "c:\path\tab12-3.dat".
```

My "path" is

```
e-drive\books\anreg2\newdata
```

Yours will be different. If you are copying commands from a .pdf file, some characters do not always copy appropriately, like ~ and -. You may have to delete and retype them.

Small data files can also be copied and pasted directly into the worksheet.

Generally one would go to the `File` menu on the top left and choose `Open`. In the new window find the appropriate folder and file and `open` it. Check the preview and, for my data files, change the `Field delimiter` to `Space`. Typically, uncheck the `Data has column names` box but you can see whether to do that from the preview. If the preview looks ok, hit `OK`. *On my data files this rarely works* because I included extra spaces to make them look good in an editor.

One advantage of Minitab is that deleting outliers is easy. Just go into the worksheet and change the value to an asterisk. You might want to copy the data vector to a new vector before deleting outliers, in case you forget the original values.

The remainder of this subsection can be skipped if you are working with the current version of Minitab.

Minitab 18 and 19 work pretty much like the current online version except you have to open the Command Line window to enter the commands.

Minitab 16 and earlier versions did read my data files because they allowed "free format." On the top row, click `File` and choose `Open Worksheet`. This opens a new menu page. Near the

bottom of this page, locate `Files of Type`, hit the down arrow and choose `Data`. This activates the `Options` button; click it. Within this page of options make two changes. For `Variable Names`, check `None`. For `Field Definition`, check `Free Format`. Click the `OK` button. Now either write in the complete file name or browse to find the .dat file that you want and hit the `Open` button.

Check the worksheet to see that the data are correct! In the Worksheet, variables (columns of numbers) are labeled C1, C2, etc. Variable names can be added near the top of the Worksheet. The labels Ci continue to work, even if variable names have been defined.

To summarize the menu choices for Minitab 16:

```
File
Open Worksheet
Files of Type:  Data
Options
Variable Names: None
Field Definition:  Free Format
```

Enter the file name in the dialog box and hit `Open`. Check the worksheet to see that the data are correct. (I cannot imagine typing in commands to read the data.)

1.3.2 SAS

Read and print the data from Example 2.1.1 of the book.

```
options ps=60 ls=80 nodate;
data Koop;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\ex2-1-1.dat';
   input y ;
proc print data=Koop;
   var y;
run;
```

The `proc print` command is there so you can check that the data were read properly!

SAS is much more touchy about reading files than R and Minitab. To get `tab14-1.dat` to read correctly I had to add spaces to several of the last rows. Incidentally, reading `tab14-1.dat` (in Chapter 14 of course) provides an example of reading alpha-numeric data.

1.4 Elementary transformations

1.4.1 Minitab

Use the `Calc` menu or

```
name c1 'y'
let c2 = loge(c1)
let c3 = sqrt(c1)
let c4 = asin(sqrt(c1))
let c5 = c1**(1/3)
```

The cubed root is just to illustrate a power transformation.

In older versions of Minitab that prompted one for commands, this would have looked like

```
MTB > name c1 'y'
MTB > let c2 = loge(c1)
MTB > let c3 = sqrt(c1)
MTB > let c4 = asin(sqrt(c1))
MTB > let c5 = c1**(1/3)
```

1.4.2 SAS

Transformations need to be specified in the `data` statement, after reading the data, but before any `proc` statements. The following program illustrates syntax.

```
options ps=60 ls=80 nodate;
data Koop;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\ex2-1-1.dat';
   input y ;
x = (4*y + y - 3*y)/2;
y1=log(x);
y2=exp(x);
y22=sqrt(x);
y3=sin(x);
y4=cos(x);
y5=x**(1/3);
y6=arsin(x);
proc print ;
var y x y1 y2 y5;
run;
```

For more help google `sas data functions and call routines`

1.5 Housekeeping

1.5.1 Minitab

Minitab is easy to use but it is often not very clear about exactly what it is doing. I frequently use the `Help` menu an select `Methods and Formula` to learn about the exact procedures.

Some commands are easier to type than have menu generated. Get rid of column 1: `erase c1`. Copy column 1 into column 2: `copy c1 c2`. Copy columns 1 and 2 into matrix `m1`: `copy c1 c2 m1`.

One advantage of Minitab is that deleting outliers is easy. Just go into the Worksheet and change the value to an asterisk. You might want to copy the data vector to a new vector before deleting outliers, in case you forget the original values.

A particularly nice feature of Minitab is that when you close a session it will prompt you to save it as a Minitab project. I usually use the File menu to be sure of where I am saving it and what I am calling it. By double clicking a project file, you go back to the exact state in which you left your work.

1.5.2 SAS

Chapter 2

One-Sample

2.1 Read book data files

See Subsection 1.3.1 for reading data files into Minitab. See Subsection 1.3.2 for reading data files into SAS.

2.2 Parametric Inference

2.2.1 Minitab

Choose `Stat` from the top line and within the menu options choose `Basic Statistics`. This provides options for one sample t (`1-Sample t`) and z inferences. (`1-Sample z`). z inferences are based on the normal distribution, i.e., $df = \infty$.

You can also trick `greg` into doing this,

```
let c11 =  Y + 1 - Y
name c11 'J'
GReg 'Y' = J;
  NoConstant;
  Confidence 95.0;
  PContinuous 1;
  TPrediction;
  TCoef;
  TANOVA.
```

2.2.1.1 P Values

To find a P value using Minitab when the reference distribution is a t, start with the number $-|t_{obs}|$, where t_{obs} is the observed value of the test statistic. In other words, find the observed test statistic and make it a negative number. Then simply use this number with the 'cdf' command, specifying the t distribution and the degrees of freedom in the subcommand. The procedure for $t_{obs} = 1.51$ is illustrated below. The probability given by the cdf command *must be doubled* to get the appropriate P value.

```
cdf -1.51;
t 35.
```

As simple as this is, the menus are even easier – because you don't have to remember anything.

```
Calc
Probability Distributions
t
```

enter the `Degrees of freedom`, check `Input constant`, and enter -1.51 in the box.

2.2.2 SAS

There is probably specialized software in SAS for this. Below is a general linear model approach.

```
data Koop;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\ex2-1-1.dat';
   input y ;
   J = y + 1 - y;
proc glm data=Koop;
   model y = J/ solution noint;
run;
```

2.3 Prediction intervals

2.3.1 Minitab

```
let c11 =  Y + 1 - Y
name c11 'J'
GReg 'Y' = J;
  NoConstant;
  Confidence 95.0;
  PContinuous 1;
  TPrediction;
  TCoef;
  TANOVA.
```

2.3.2 SAS

There is probably one line to add to `proc glm` to get a prediction interval. Alas, I don't know it.

```
data Koop;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\ex2-1-1.dat';
   input y ;
   J = y + 1 - y;
proc glm data=Koop;
   model y = J/ solution noint;
      output out=new LCL = plow UCL= phigh LCLM= clow UCLM=chigh alpha=.01;
proc print data=new;
run;
```

2.4 Model testing

2.5 Normal plots

2.5.1 Minitab

Use the menus

```
graph
probability plot
single
```

Specify the variable for the plot in the appropriate place. This defaults to a normal plot, other options are available. These menu selections generate the following code

```
PPlot 'y';
  Normal;
  Symbol;
```

```
  FitD;
  Grid 2;
  Grid 1;
  MGrid 1.
```

As mentioned earlier, Minitab is easy to use but it is often not very clear about exactly what it is doing. (Not that any program really is.) For example, the normal plot produced by these commands includes a P value. For what? By going to the `Help` menu, selecting `StatGuide`, `Graphs`, and `Probability Plot` we find that the plot is using the Anderson-Darling statistic and giving the associated P value.

A computer program is necessary for finding the normal scores and convenient for plotting the data and computing W'. The following Minitab commands provide a normal plot and the W' statistic for a variable in c1.

```
name c1 'y'
nscores c1 c2
plot c1*c2
corr c1 c2
note    The correlation is printed out, e.g., 0.987.
note    This correlation is used in the next command.
let k1=.987**2
note    k1 is W'
print k1
```

2.5.2 SAS

A crude normal plot can be obtained as follows.

```
data Koop;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\ex2-1-1.dat';
   input y ;
proc rank data=new normal=blom;
   var y;
   ranks nscores;
proc plot;
   plot y*nscores/vpos=16 hpos=32;
run;
```

To get higher quality graphics you might try.

```
data Koop;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\ex2-1-1.dat';
   input y ;
ods graphics on;
proc rank data=new normal=blom;
   var y;
   ranks nscores;
proc plot;
   plot y*nscores/vpos=16 hpos=32;
run;
ods graphics off;
```

2.6 Transformations

See Section 1.4.

2.7 Inference about σ^2

2.7.1 Minitab

Choose `Stat` from the top line and within the menu options choose `Basic Statistics`. This provides an options for testing a variance: `1 Variance`.

2.7.2 SAS

Chapter 3

Defining Linear Models in Minitab

This chapter examines the syntax of Minitab models from the most elementary models to the quite sophisticated. We begin with an example, to remind those users who are already familiar with the statistical concepts, of the syntaxes used to specify models in Minitab, R, and SAS. On a first reading of the manual, you can skip this first example.

EXAMPLE 3.0.1. *Modeling Cheat Sheet.* We provide model syntax for models defined in Section 16.1 of the book. All three programs can fit the first form of the model. Minitab ONLY fits the first form. The R and SAS commands given below are for fitting the second form of the model.

$$\begin{aligned}[ABC] &\cong y_{ijkm} = G + A_i + B_j + C_k + [AB]_{ij} + [AC]_{ik} + [BC]_{jk} + [ABC]_{ijk} + e_{ijkm} \\ &\cong y_{ijkm} = [ABC]_{ijk} + e_{ijkm}.\end{aligned}$$

$$\begin{aligned}[AB][BC] &\cong y_{ijkm} = G + A_i + B_j + C_k + [AB]_{ij} + [BC]_{jk} + e_{ijkm} \\ &\cong y_{ijkm} = [AB]_{ij} + [BC]_{jk} + e_{ijkm}.\end{aligned}$$

$$\begin{aligned}[AB][C] &\cong y_{ijkm} = G + A_i + B_j + C_k + [AB]_{ij} + e_{ijkm} \\ &\cong y_{ijkm} = [AB]_{ij} + C_k + e_{ijkm}.\end{aligned}$$

$$\begin{aligned}[A_0][A_1][A_2][C] &\cong y_{ijkm} = G + A_{i0} + \gamma_1 x_j + \gamma_2 x_j^2 + A_{i1} x_j + A_{i2} x_j^2 + C_k + e_{ijkm}. \\ &\cong y_{ijkm} = A_{i0} + A_{i1} x_j + A_{i2} x_j^2 + C_k + e_{ijkm}.\end{aligned}$$

Model	Minitab	R	SAS
$[ABC]$	$A\|B\|C$	A:B:C-1	A*B*C / noint
$[AB][BC]$	$A\|B \quad B\|C$	A:B+B:C-1	A*B B*C / noint
$[AB][C]$	$A\|B \quad C$	A:B+C-1	A*B C / noint
$[A_0][A_1][A_2][C]$	$A\|X \quad A\|X2 \quad C$	A+A:X+A:X2+C-1	A A*X A*X2 C / noint

To fit different models, one needs to modify the part of the code that specifies the model. In Minitab's `glm`, models are usually specified in the `model` dialog box (or on the command line) and X and X2 have to be specified as covariates. In R, specifying models involves changes to, say, `lm(y ~ A:B+C-1)` where A, B, and C all have to be prespecified as `factor` variables. In SAS's `proc glm`, modeling involves changes to `model y = A*B C/noint;` where A, B, and C all have to be prespecified as `class` variables.

I think the following statements are true. In R the model `A*B*C` is equivalent to `A+B+C+A:B+A:C+B:C+A:B:C`. In Minitab and SAS the model `A|B|C` is equivalent to `A B A*B C A*C B*C A*B*C`. □

This chapter describes general approaches to specifying fixed effect linear models in Minitab. Chapter 3 in the book describes general approaches to statistical inference with Section 3.9 introducing various linear models that are particularly useful. Most of this chapter is devoted to a discussion of how to specify those linear models in Minitab. The chapter goes beyond those models because I think it is useful to consolidate in one place the fundamental ideas of specifying Minitab models. It does not, however, discuss the random effects models that appear in Chapter 19.

We assume that y is a measurement random variable and that x is some predictor variable or that $x \equiv (x_1, \ldots, x_p)'$ is a vector of predictor variables. *In a computer file* all of the observations on y consist of a column of numbers and the x observations are either a single column of numbers or p different columns of numbers, one column for each component of the vector x. The components of the vector x can either be measurement (continuous) variables, classification (categorical, factor, discrete) variables, or some combination of the two. We assumed in Section 3.9 of the book that

$$E(y) = m(x)$$

for some function m and described a number of different, commonly used, examples. When x contains only measurement variables, we construct regression models, when x contains only classification variables, we construct ANOVA models, when x contains a combination of the two, we construct ACOVA models.

Of course we have to tell the computer program whether any component of the vector x is a measurement or classification variable. Most computer programs have a default setting that, unless a variable is specified to be one thing, it is assumed to be the other. In SAS and R, the default is that any numeric variable is a measurement variable. In Minitab, the default changes with the specific program being used. Any variable that takes nonnumeric values is automatically taken as a classifier.

The modeling capabilities of Minitab are not as flexible as those in R and SAS. We can fit any model we need in Minitab but our choices for parameterizations of models are more limited. (Minitab requires *hierarchical models*, something we will discuss later.) Our modeling in Minitab will be focused on the `glm` (general linear model) command which is an option under the `Stat` menu and its `ANOVA` submenu. The `General Regression (GReg or greg)` command, found under the `Stat` menu's `Regression` submenu, is quite similar to the `glm` command and will also be a primary focus. One difference between these programs is that `glm`, by default, assumes that variables are categorical so that covariates must be specifically identified, whereas `GReg`, by default, assumes that numeric variables are measurements (continuous, covariates) so that categorical variables must be specifically identified. Both `glm` and `GReg` default to include an intercept term (grand mean) in every model, only GReg allows the intercept to be removed.

The fundamental form of these programs follow. For `glm` the core commands are

```
glm  y = model;
   covariates list;
   brief 3.
```

`list` is really a list of variable names that are measurement variables in the model. If there are no covariates, the subcommand can be dropped. If *all* of the predictor variables are covariates, the `brief 3` subcommand can be dropped. It is only needed to obtain estimates associated with categorical variables. *When using the `glm` menu, `y` goes in the "Responses" dialog box and `model` goes in the "Model" dialog box. The list of covariates is specified on the `Covariates` submenu. To specify `brief 3`, go to the `Results` submenu and check the last circle under `Display of Results`.*

For `greg` the core commands are

```
greg y = model;
  categorical list;
  tcoef;
  tanova.
```

`list` is really a list of variable names that are categorical variables in the model. If there are no categorical variables, the subcommand can be dropped. The subcommands `tcoef` and `tanova` are needed to get the table of coefficients and the ANOVA table printed. *When using the `GReg` menu, `y` goes in the "Response" dialog box, `model` goes in the "Model" dialog box, and the list goes in the "Categorical predictors" dialog box. The two other subcommands are produced by default.*

In Minitab, commands and subcommands are separated by semicolons and the entire string of commands *must* terminate with a period. The standard diagnostic quantities that were discussed in Chapter 7 and Section 10.1 of the book are obtained using the following subcommands.

```
  Resid 'ehat';
  SResid 'r';
  TResid 't';
  Hi 'lev';
  CookD 'C';
```

These subcommands apply to both `glm` and `greg` and other Minitab programs as well. *When using the menus, these are all choices available to check on the `Storage` submenu.*

We now return to the real subject of this chapter, how to define "`model`" appropriately in Minitab.

3.1 One sample

Most statistics programs have specialized software for data in which all the observations have the same mean value, like Minitab's `1 Sample t`.

Linear models programs often do not allow fitting *only* a common mean to all observations but you can generally trick them into fitting such a model by defining a predictor variable that is always 1 and not fitting a constant to the model.

```
let c20 = y+1-y
GReg y = c20;
  NoConstant;
  TCoef;
  TANOVA.
```

With these commands sitting in front of you, it is probably easiest just to type them, but you can also use the menus. You could use the `Calc` menu to create c20. In the `Reg` menu, select `GReg`. After specifying the response and the model, in the `Options` submenu deselect `Fit intercept`. *I do not think you can get this analysis out of `glm`.*

3.2 Two samples

For two samples, x should be a categorical variable with only two categories. Use

```
glm y = x;
  Brief 3.
```

The subcommand `Brief 3` is necessary to get a table of coefficients. To specify `brief 3` in the `glm` menu, go to the `Results` submenu and check the last circle under `Display of Results`. Alternatively, use

```
GReg y = x;
  Covariate x;
  TCoef;
  TANOVA.
```

In either case, the first group mean is the sum of the two coefficients and second group mean is the difference in the two coefficients. The t test for the second coefficient is the test for equality of means.

Chapter 4 of the book also considers some extensions, i.e., paired comparisons and unequal variances. The `glm` and `greg` commands assume equal variances, so they do not apply to the latter. As discussed in Chapter 4, paired comparisons can be treated as either one-sample that consists of the differences between the pairs or as a two-way ANOVA without interaction.

3.3 Regression

In regression we assume that the vector x contains only measurement variables.

3.3.1 Simple linear regression

With a single predictor x, simple linear regression is

$$m(x) = \beta_0 + \beta_1 x$$

or

$$y_h = \beta_0 + \beta_1 x_h + \varepsilon_h, \qquad h = 1, \ldots, n.$$

It is fitted in `glm` as

```
glm y = x;
  Covar x.
```

or in `greg` as

```
GReg y = x;
  TCoef;
  TANOVA.
```

Notice that the intercept term is not specified and that this has the same model structure as used for two samples. The key point is that x is a measurement variable not a categorical variable. We will see later that this same Minitab structure is used for one-way ANOVA when x is a classification variable with more than two categories.

If we wanted to force the regression through the origin the model becomes

$$y_h = \beta_1 x_h + \varepsilon_h, \qquad h = 1, \ldots, n$$

and is modeled in `greg` as

```
GReg y = x;
  NoCon;
  TCoef;
  TANOVA.
```

The model in question does not contain an intercept and the `NoConstant` subcommand is used to stop an intercept being fitted. I do not know how to keep `glm` from from fitting an intercept.

3.3.2 Polynomial regression

A cubic polynomial model

$$m(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$

or

$$y_h = \beta_0 + \beta_1 x_h + \beta_2 x_h^2 + \beta_3 x_h^3 + \varepsilon_h, \qquad h = 1, \ldots, n,$$

is modeled in `glm` as

```
glm y = x x*x x*x*x;
covar x.
```

or in greg as

```
greg y = x x*x x*x*x;
  tcoef;
  tanova.
```

3.3.3 Multiple regression

Now suppose x is a vector of measurement variables with $p = 3$, i.e., $x = (x_1, x_2, x_3)'$. The multiple regression model is

$$m(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

or

$$y_h = \beta_0 + \beta_1 x_{h1} + \beta_2 x_{h2} + \beta_3 x_{h3} + \varepsilon_h, \qquad h = 1, \ldots, n.$$

Typically, when programming it is safer not to define variables using subscripts, so we write x_j as `xj` and the model is written in `glm` as

```
glm y = x1 x2 x3;
covar x1 x2 x3.
```

or in greg as

```
greg y = x1 x2 x3;
  tcoef;
  tanova.
```

Notice that the intercept term is never specified.

3.3.4 Offsets

Suppose in the multiple regression model we knew that $\beta_2 = 5$ so that the model becomes

$$y_h = \beta_0 + \beta_1 x_{h1} + 5x_{h2} + \beta_3 x_{h3} + \varepsilon_h, \qquad h = 1, \ldots, n.$$

As far as I know, there is no direct way to model this in Minitab. However, for a linear model like this, we can rewrite the linear model as

$$y_h - 5x_{h2} = \beta_0 + \beta_1 x_{h1} + \beta_3 x_{h3} + \varepsilon_h, \qquad h = 1, \ldots, n$$

which can be handled in Minitab as

```
let yoff = y - 5*x2
glm yoff = x1 x3;
covar x1 x2 x3.
```

GReg works similarly.

It is not much of a problem that Minitab does not make provision for offset terms in linear models. It is a much larger problem that Minitab has no "offset" capability in its programs to treat the "generalized linear models" examined near the end of the book.

3.4 ANOVA

We begin by assuming that x is a single classification variable and then move to vectors of classification variables. `Glm` assumes that all predictor variables are categorical unless otherwise specified. GReg assumes that numeric predictor variables are covariates unless otherwise specified.

3.4.1 One-way ANOVA

A typical one-way ANOVA model is written

$$y_{ij} = \mu_i + \varepsilon_{ij}, \qquad i = 1,2,\ldots,a,\ j = 1,\ldots,N_i.$$

In this context, the generic predictor variable x should really be thought of as i, the subscript in the model that identifies the groups. *It is more trouble than it is worth to get Minitab to fit this model.*

The alternative form of the one-way model that contains a grand mean (intercept) is what most programs are designed to fit,

$$y_{ij} = \mu + \alpha_i + \varepsilon_{ij}, \qquad i = 1,2,\ldots,a,\ j = 1,\ldots,N_i.$$

The Minitab analysis is obtained from

```
glm y = x;
brief 3.
```

or

```
greg y = ii;
  Categorical x;
  TCoef;
  TANOVA.
```

For estimation Minitab uses the side condition $\sum_{i=1}^{a} \alpha_i = 0$ and reports estimates $\hat{\alpha}_1,\ldots,\hat{\alpha}_{a-1}$ with the understanding that $-\hat{\alpha}_a = \sum_{i=1}^{a-1} \hat{\alpha}_i$.

Alternatively, the code

```
greg y = x;
  Categorical x;
  Coding 1;
  TCoef;
  TANOVA.
```

gets Minitab to use the side condition $\alpha_1 = 0$ (like R) which forces the estimate of μ to be the sample mean of group 1 and the estimate of α_r to be the difference between the sample mean of group r and the sample mean of group 1, $r = 1,\ldots,a$. From the `GReg` menu, this option involves opening the `Options` submenu and checking the (1,0) choice for "Type of coding for categorical predictors".

Regardless of side conditions, $\hat{\mu} + \hat{\alpha}_r = \bar{y}_{r\cdot}$ for any r.

The one-way ANOVA with $a = 2$ model is identical to the model for two independent samples with equal variances and the model specification in Minitab is the same.

3.4.2 Two-way ANOVA

Now suppose x is a vector of classification variables with $p = 2$, i.e., $x = (x_1, x_2)'$. As illustrated earlier, rewrite x_j as `xj`. Because `x1` and `x2` are classification variables (but more than anything, to be consistent with the R and SAS presentations), *throughout this entire subsection* we relabel `x1` and `x2` as `ii` and `jj`, respectively, via

```
let c11 = x1
copy x2 c12
names c11 'ii' c12 'jj'.
```

(The columns `c11` and `c12` were selected haphazardly.)

Models similar to those discussed below also hold for $p > 2$ but get exponentially more complicated. The book examines in detail some cases with $p = 3$ and $p = 4$. Section 8 below also contains some discussion of higher-order models.

3.4.2.1 Interaction

The model

$$y_{ijk} = \mu_{ij} + \varepsilon_{ijk}, \qquad i = 1,\ldots,a,\ j = 1,\ldots,b,\ k = 1,\ldots,N_{ij},$$

is not readily fit by Minitab. Some of the output can be obtained by treating the model as a one-way ANOVA and using specialized software, see Chapter 12.

The equivalent interaction model can be written

$$y_{ijk} = \mu + \alpha_i + \eta_j + (\alpha\eta)_{ij} + \varepsilon_{ijk}, \qquad i = 1,\ldots,a,\ j = 1,\ldots,b,\ k = 1,\ldots,N_{ij}.$$

and the `glm` or `greg` analysis is obtained using

```
y = ii jj ii*jj
```

or, incorporating Minitab's shorthand for hierarchical models,

```
y = ii|jj
```

Henceforth, we will use this shorthand as much as possible. For estimation Minitab uses the default side conditions

$$\sum_{i=1}^{a}\alpha_i = 0; \qquad \sum_{j=1}^{b}\eta_j = 0; \qquad \sum_{j=1}^{b}(\alpha\eta)_{ij} = 0, \quad i = 1,\ldots,a; \qquad \sum_{i=1}^{a}(\alpha\eta)_{ij} = 0, \quad j = 1,\ldots,b.$$

For example, if $j \neq b$ Minitab reports estimates $\widehat{(\alpha\eta)}_{1j},\ldots,\widehat{(\alpha\eta)}_{a-1,j}$ with the understanding that $-\widehat{(\alpha\eta)}_{a,j} = \sum_{i=1}^{a-1}\widehat{(\alpha\eta)}_{ij}$. Similar computations must be made with the other side conditions to find the complete set of estimates.

Alternatively, we could use the script

```
greg y = ii|jj;
  Categorical ii jj;
  Coding 1;
  TCoef;
  TANOVA.
```

With the subcommand `Coding 1`, the estimated parameters for this overspecified model incorporate (like R) the side conditions,

$$\alpha_1 = 0; \qquad \eta_1 = 0; \qquad (\alpha\eta)_{1j} = 0, \quad j = 1,\ldots,b; \qquad (\alpha\eta)_{i1} = 0 \quad i = 1,\ldots,a.$$

From the `GReg` menu, this option involves opening the `Options` submenu and checking the (1,0) choice for "Type of coding for categorical predictors".

3.4.2.2 Additive effects

The additive effects model is

$$y_{ijk} = \mu + \alpha_i + \eta_j + \varepsilon_{ijk}, \qquad i = 1,\ldots,a,\ j = 1,\ldots,b,\ k = 1,\ldots,N_{ij}.$$

and is written in `glm` or `greg` as

```
y = ii jj
```

For estimation Minitab uses the default side conditions

$$\sum_{i=1}^{a}\alpha_i = 0; \qquad \sum_{j=1}^{b}\eta_j = 0.$$

Alternatively, we could use the script

```
greg y = ii jj;
  Categorical ii jj ;
  Coding 1;
  TCoef;
  TANOVA.
```

With the subcommand `Coding 1`, the estimated parameters for this overspecified model use (like R) the side conditions,

$$\alpha_1 = 0; \qquad \eta_1 = 0.$$

From the `GReg` menu, this option involves opening the `Options` submenu and checking the (1,0) choice for "Type of coding for categorical predictors".

3.4.2.3 Sequential fitting

As discussed in the book, although the models themselves are equivalent, some computer output changes depending on whether one specifies the two-way interaction model as `y = ii jj ii*jj` (`y = ii|jj`) or as `y = jj ii jj*ii` (`y = jj|ii`). Although the end models are the same, the process of getting to the end model is different and the sequential sums of squares (SeqSS) output contains information from the process in addition to information from the end result. For unbalanced data, these models typically give different numbers for the SeqSS, cf. Section 9.4 and Subsection 14.1.2 of the book. For balanced data, only the order of presentation changes. Parameter estimation depends only on the end model and does not change materially with different orderings of terms.

In particular, when you ask `glm` or `greg` to fit `y = ii jj ii*jj`, or equivalently `ii|jj`, they sequentially fit the models

```
y = ii
y = ii jj
y = ii jj ii*jj
```

In general, they fit a sequence of models determined by the order in which you added terms of the larger model! This is reflected in the sequential sums of squares. Incidentally, as long as the main effect `ii` comes before `jj`, it is irrelevant in these models whether you specify `ii jj ii*jj` or specify `ii jj jj*ii`.

Although this discussion has focused on the two-way with interaction model, issues of sequential fitting are pervasive with unbalanced data (including regression). This is not so much a problem as an opportunity. The extra computer output can save you from the chore of fitting some models. It never does any harm unless you misinterpret the results.

As discussed in Section 14.1 of the book, there is also a concept of *adjusted sums of squares*. In general, adjusted sums of squares are meant to be sums of squares for fitting terms last. In the additive effects model, the adjusted sums of squares for each main effect is the sequential sum of squares for that term when the term is being fitted to the model after the other main effect. However, *in the interaction model, most of the adjusted sums of squares are worthless*. The interaction adjusted sum of squares is fine, it is just the sum of squares for adding the interaction after both of the main effects. However, the sum of squares for a main effect fitted after an interaction should be zero, not the arbitrary numbers reported for main effects as adjusted sums of squares. The reported numbers are arbitrary because they depend on the choice of side conditions being used by the program to determine estimates. The side conditions are arbitrary choices, so these adjusted sums of squares are also arbitrary. An illustration of this phenomenon appears in Subsubsection 14.1.1.1 of this guide.

3.5 ACOVA and interaction

The following models were not discussed in Section 3.9 of the book but rather are discussed in Chapter 15. They involve a classification variable x_1 written as `x1` that denotes a group i and a

measurement variable x_2 written as `x2` *or equivalently as* `z` (to make the notation more similar to the book). Throughout the section we

```
let  c11 = x2
let  c12 = x1
name c11 'z'  c12 'ii'.
```

3.5.1 ACOVA: parallel lines

Minitab will not readily fit the nicest ACOVA model,

$$y_{ij} = \mu_i + \gamma z_{ij} + \varepsilon_{ij}, \qquad i = 1,2,\ldots,a,\ j = 1,\ldots,N_i.$$

but it will happily fit the alternative model

$$y_{ij} = \mu + \alpha_i + \gamma z_{ij} + \varepsilon_{ij}$$

as

```
y = ii z;
```

In `glm`, `z` has to be specified as a covariate and in `greg`, `ii` has to be specified as a "Categorical predictor". For estimation Minitab uses the default side condition

$$\sum_{i=1}^{a} \alpha_i = 0.$$

Alternatively, we could use the script

```
greg y = ii z;
  Categorical ii;
  Coding 1;
  TCoef;
  TANOVA.
```

With the subcommand `Coding 1`, the estimated parameters for this overspecified model use (like R) the side condition $\alpha_1 = 0$. From the `GReg` menu, this option involves opening the `Options` submenu and checking the (1,0) choice for "Type of coding for categorical predictors".

Note that this Minitab model is essentially the same form as that used for the two-way additive effects model in Subsubsection 3.4.2.2 except now one of the predictor variables is a measurement variable and the other is a classification variable.

3.5.2 Interaction: skew lines

We now consider ways of fitting lines to each group that can have different slopes as well as different intercepts. The simplest model to interpret is

$$y_{ij} = \mu_i + \gamma_i z_{ij} + \varepsilon_{ij}.$$

Minitab doesn't like this. The version most similar to generalizing the traditional ACOVA model is

$$y_{ij} = \mu + \alpha_i + \gamma_i z_{ij} + \varepsilon_{ij}.$$

Again, Minitab doesn't like this.

Minitab likes to specify a "hierarchical" version of the model (which I generally think is pretty silly but admittedly gives some useful computer output),

$$y_{ij} = \mu + \alpha_i + \beta z_{ij} + \gamma_i z_{ij} + \varepsilon_{ij}$$

written in Minitab as

```
y = ii z ii*z
```

or, even more simply, as

```
y = ii|z
```

Again, in `glm`, `z` has to be specified as a covariate and in `greg`, `ii` has to be specified as a categorical predictor. Also, `glm` will not print out coefficient estimates for the `ii` effects without being asked (the `brief 3` subcommand). But then `greg` will not print out anything without being asked.

For estimation Minitab uses the side conditions

$$\sum_{i=1}^{a} \alpha_i = 0 = \sum_{i=1}^{a} \gamma_i.$$

It reports estimates $\hat{\alpha}_1, \ldots, \hat{\alpha}_{a-1}$ with the understanding that $-\hat{\alpha}_a = \sum_{i=1}^{a-1} \hat{\alpha}_i$ and estimates $\hat{\gamma}_1, \ldots, \hat{\gamma}_{a-1}$ with the understanding that $-\hat{\gamma}_a = \sum_{i=1}^{a-1} \hat{\gamma}_i$.

Alternatively, we could use the script

```
greg y = ii|z;
  Categorical ii;
  Coding 1;
  TCoef;
  TANOVA.
```

With the subcommand `Coding 1`, the estimated parameters for this overspecified model incorporate (like R) the side conditions,

$$\alpha_1 = 0; \qquad \gamma_1 = 0.$$

From the `GReg` menu, this option involves opening the `Options` submenu and checking the (1,0) choice for "Type of coding for categorical predictors".

3.6 Interaction in multiple regression

The multiple regression model with $p = 2$ and continuous predictors $x = (x_1, x_2)$ has

$$m(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

or

$$y_h = \beta_0 + \beta_1 x_{h1} + \beta_2 x_{h2} + \varepsilon_h, \qquad h = 1, \ldots, n.$$

This is an additive model in x_1 and x_2, cf. Section 9.9 of the book. As we have already seen, the `glm` and `greg` model for this is `y = x1 x2`. It is essentially the same as that used for the two-way ANOVA additive effects model and the parallel lines ACOVA model except that now both of the predictor variables are measurement variables. Moreover, the effects of the two predictors simply add together.

A useful way of incorporating some interaction into the model is to fit

$$m(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2,$$

that is,

$$y_h = \beta_0 + \beta_1 x_{h1} + \beta_2 x_{h2} + \beta_3 x_{h1} x_{h2} + \varepsilon_h, \qquad h = 1, \ldots, n.$$

This is no longer an additive model in the original variables x_1 and x_2 (although it *is* an additive model in the variables x_1, x_2, $x_1 x_2$). The main reason for introducing this model here is to illustrate that the Minitab regression model has essentially the same structure as the Minitab models for two-way ANOVA with interaction and the ACOVA interaction model with skew lines. The Minitab model is

```
y = x1 x2 x1*x2
```

or

```
y = x1|x2
```

With both variables being measurement variables there is yet another option for fitting the model. `Let c13 = x1*x2`, `name c13 'x1x2'`, then use the model `y = x1 x2 x1x2`.

3.7 Hierarchical and nested models

The model

$$y_{ijk} = \mu + \alpha_i + \eta_j + (\alpha\eta)_{ij} + \varepsilon_{ijk}, \qquad i = 1,\ldots,a,\ j = 1,\ldots,b,\ k = 1,\ldots,N_{ij}.$$

is said to be *hierarchical* because the model includes not only the interaction terms $(\alpha\eta)_{ij}$ but also the lower order "main effect" terms α_i and η_j as well as the grand mean μ. A model is hierarchical whenever a model that includes an mth order interaction between some variables also includes all of the lower order interactions among those variables as well as all the main effects and the grand mean. Thus, if a model contains $(\alpha\eta\gamma)$ terms, it must also include $(\alpha\eta)$ terms, $(\alpha\gamma)$ terms, $(\eta\gamma)$ terms, and the main effect terms, the αs, ηs, and γs, as well as the grand mean μ. I have been at some pains in the book to point out that for ANOVA models all of these lower order terms are meaningless, so I have little regard for this concept of a hierarchical model. (The term "hierarchical" takes on alternate meanings in alternate contexts, e.g., Bayesian statistics.)

As mentioned at the beginning of the chapter, Minitab assumes that all models are hierarchical. It even uses a similar concept of hierarchy for measurement variables. (I should mention that when using the rather artificial, but computationally very convenient, concept of interaction between continuous (measurement) variables employed by all of Minitab, R and SAS, the lower order terms are **NOT** meaningless.) `Glm` refuses to fit models that are not hierarchical. `GReg` gives incorrect answers for models that are not hierarchical. (`GReg` gives the correct answer for *some* model, it just is not the model you wish it was and is not any model that is easy to determine.)

Minitab (like R) incorporates a special operator to simplify defining hierarchical models. With `x1` and `x2` as classification variables that define i and j, respectively, in Minitab the model displayed earlier can be written either as as

```
y = x1 x2 x1*x2
```

or

```
y = x1|x2
```

If you care about hierarchical models, you might care about nested models which are models that are not quite hierarchical. For example,

$$y_{ijk} = \mu + \alpha_i + (\alpha\eta)_{ij} + \varepsilon_{ijk}, \qquad i = 1,\ldots,a,\ j = 1,\ldots,b,\ k = 1,\ldots,N_{ij}$$

is said to have the $(\alpha\eta)_{ij}$ effects nested within the α_i effects. Again, fixed α_i effects are irrelevant in this model, so I find it hard to care about this concept of nesting. Nonetheless, Minitab has specific code for defining nested models. For factor variables `x1 x2`, the model above could be written

```
y = x1 x2(x1)
```

Ironically, even though the original model is already nonhierarchical, if you leave out the main effect `x1`, `glm` complains that the model is not hierarchical.

3.8 Higher-order models

When discussing Minitab, we will consider "higher order" models to be those that involve $p > 2$, although the distinction is probably more common only when the number of classification variables is greater than 2. The reason we do this is because the methods in Minitab for defining higher order models are pretty much interchangeable regardless of whether the predictors are measurements or

classifications. When used on measurement variables, the Minitab commands introduce interaction similar to Section 3.6.

For example, the code

```
y = x1|x2|x3
```

can determine a variety of models. If all the predictors are measurements, the model is

$$y_h = \beta_{000} + \beta_{100}x_{h1} + \beta_{010}x_{h2} + \beta_{001}x_{h3} + \beta_{110}x_{h1}x_{h2} + \beta_{101}x_{h1}x_{h3} + \beta_{011}x_{h2}x_{h3} + \beta_{111}x_{h1}x_{h2}x_{h3} + \varepsilon_h, \qquad h = 1,\ldots,n.$$

When all the predictors are classifications with $x_1 \equiv i$, $x_2 \equiv j$, and $x_3 \equiv k$ the code means

$$y_{ijks} = \mu + \alpha_i + \eta_j + \gamma_k + (\alpha\eta)_{ij} + (\alpha\gamma)_{ik} + (\eta\gamma)_{jk} + (\alpha\eta\gamma)_{ijk} + \varepsilon_{ijks}, \qquad s = 1,\ldots,N_{hij}.$$

If only `x1` is a measurement variable, it means

$$y_{jks} = \mu + \beta_1 x_{jks1} + \eta_j + \gamma_k + \beta_{1j}x_{jks1} + \beta_{1k}x_{jks1} + (\eta\gamma)_{jk} + \beta_{1jk}x_{jks1} + \varepsilon_{jks}, \qquad s = 1,\ldots,N_{jk}.$$

If both `x1` and `x2` are measurement variables, it means

$$y_{ks} = \beta_{00} + \beta_{10}x_{ks1} + \beta_{01}x_{ks2} + \gamma_k + \beta_{11}x_{ks1}x_{ks2} + \beta_{10k}x_{ks1} + \beta_{01k}x_{ks2} + \beta_{11k}x_{ks1}x_{ks2} + \varepsilon_{jks}, \qquad s = 1,\ldots,N_k,$$

Minitab will also let us subtract out the three factor term

```
y = x1|x2|x3 - x1*x2*x3
```

to give us a second order interaction model. If all the predictors are measurements, the model is

$$y_h = \beta_{000} + \beta_{100}x_{h1} + \beta_{010}x_{h2} + \beta_{001}x_{h3} + \beta_{110}x_{h1}x_{h2} + \beta_{101}x_{h1}x_{h3} + \beta_{011}x_{h2}x_{h3} + \varepsilon_h, \qquad h = 1,\ldots,n.$$

When all the predictors are classifications with $x_1 \equiv i$, $x_2 \equiv j$, and $x_3 \equiv k$ the code means

$$y_{ijks} = \mu + \alpha_i + \eta_j + \gamma_k + (\alpha\eta)_{ij} + (\alpha\gamma)_{ik} + (\eta\gamma)_{jk} + \varepsilon_{ijks}, \qquad s = 1,\ldots,N_{hij}.$$

If only `x1` is a measurement variable, it means

$$y_{jks} = \mu + \beta_1 x_{jks1} + \eta_j + \gamma_k + \beta_{1j}x_{jks1} + \beta_{1k}x_{jks1} + (\eta\gamma)_{jk} + \varepsilon_{jks}, \qquad s = 1,\ldots,N_{jk}.$$

If both `x1` and `x2` are measurement variables, it means

$$y_{ks} = \beta_{00} + \beta_{10}x_{ks1} + \beta_{01}x_{ks2} + \gamma_k + \beta_{11}x_{ks1}x_{ks2} + \beta_{10k}x_{ks1} + \beta_{01k}x_{ks2} + \varepsilon_{jks}, \qquad s = 1,\ldots,N_k.$$

In fact, you can even write `y = x1|x2|x3 - x1*x2*x3 - x2*x3` in place of `y = x1|x2 x1|x3`, which is a somewhat redundant version (it includes the main effect `x1` twice) of `y = x1 x2 x3 x1*x2 x1*x3`.

Similarly, we can fit third order interaction models for $p = 4$,

```
y = x1|x2|x3|x4 - x1*x2*x3*x4
```

The regression model with all measurement variables is

$$\begin{aligned} y_h = {} & \beta_{000} + \beta_{1000}x_{h1} + \beta_{0100}x_{h2} + \beta_{0010}x_{h3} + \beta_{0001}x_{h4} \\ & + \beta_{1100}x_{h1}x_{h2} + \beta_{1010}x_{h1}x_{h3} + \beta_{1001}x_{h1}x_{h4} + \beta_{0110}x_{h2}x_{h3} + \beta_{0101}x_{h2}x_{h4} + \beta_{0011}x_{h3}x_{h4} \\ & + \beta_{1110}x_{h1}x_{h2}x_{h3} + \beta_{1101}x_{h1}x_{h2}x_{h4} + \beta_{1011}x_{h1}x_{h3}x_{h4} + \beta_{0111}x_{h2}x_{h3}x_{h4} + \varepsilon_h, \qquad h = 1,\ldots,n. \end{aligned}$$

The ANOVA model with all classification variables is

$$\begin{aligned} y_{hijks} = {} & \mu + \alpha_h + \eta_i + \gamma_j + \delta_k \\ & + (\alpha\eta)_{hi} + (\alpha\gamma)_{hj} + (\alpha\delta)_{hk} + (\eta\gamma\delta)_{ij} + (\eta\delta)_{ik} + (\gamma\delta)_{jk} \\ & + (\alpha\eta\gamma)_{hij} + (\alpha\eta\delta)_{hik} + (\alpha\gamma\delta)_{hjk} + (\eta\gamma\delta)_{ijk} + \varepsilon_{hijks}, \qquad s = 1,\ldots,N_{hijk}. \end{aligned}$$

I will leave it to you to worry about the various kinds of ACOVA models that mix measurements with classifications.

Defining Linear Models in SAS

This chapter examines the syntax of SAS models from the most elementary models to the quite sophisticated. We begin with an example, to remind those users who are already familiar with the statistical concepts, of the syntaxes used to specify models in Minitab, R, and SAS. On a first reading of the manual, you can skip this first example.

EXAMPLE 3.8.1. *Modeling cheat sheet*. We provide model syntax for models defined in Section 16.1 of the book. All three programs can fit the first form of the model. Minitab ONLY fits the first form. The R and SAS commands given below are for fitting the second form of the model.

$$\begin{aligned}[ABC] &\cong y_{ijkm} = G + A_i + B_j + C_k + [AB]_{ij} + [AC]_{ik} + [BC]_{jk} + [ABC]_{ijk} + e_{ijkm} \\ &\cong y_{ijkm} = [ABC]_{ijk} + e_{ijkm}.\end{aligned}$$

$$\begin{aligned}[AB][BC] &\cong y_{ijkm} = G + A_i + B_j + C_k + [AB]_{ij} + [BC]_{jk} + e_{ijkm} \\ &\cong y_{ijkm} = [AB]_{ij} + [BC]_{jk} + e_{ijkm}.\end{aligned}$$

$$\begin{aligned}[AB][C] &\cong y_{ijkm} = G + A_i + B_j + C_k + [AB]_{ij} + e_{ijkm} \\ &\cong y_{ijkm} = [AB]_{ij} + C_k + e_{ijkm}.\end{aligned}$$

$$\begin{aligned}[A_0][A_1][A_2][C] &\cong y_{ijkm} = G + A_{i0} + \gamma_1 x_j + \gamma_2 x_j^2 + A_{i1} x_j + A_{i2} x_j^2 + C_k + e_{ijkm}. \\ &\cong y_{ijkm} = A_{i0} + A_{i1} x_j + A_{i2} x_j^2 + C_k + e_{ijkm}.\end{aligned}$$

Model	Minitab	R	SAS
$[ABC]$	$A\|B\|C$	A:B:C-1	A*B*C / noint
$[AB][BC]$	$A\|B$ $B\|C$	A:B+B:C-1	A*B B*C / noint
$[AB][C]$	$A\|B$ C	A:B+C-1	A*B C / noint
$[A_0][A_1][A_2][C]$	$A\|X$ $A\|X2$ C	A+A:X+A:X2+C-1	A A*X A*X2 C / noint

To fit different models, one needs to modify the part of the code that specifies the model. In Minitab's `glm`, models are usually specified in the `model` dialog box (or on the command line) and X and X2 have to be specified as covariates. In R, specifying models involves changes to, say, `lm(y ~ A:B+C-1)` where A, B, and C all have to be prespecified as `factor` variables. In SAS's `proc glm` and `proc genmod`, modeling involves changes to `model y = A*B C/noint;` where A, B, and C all have to be prespecified as `class` variables.

I think the following statements are true. In R the model `A*B*C` is equivalent to `A+B+C+A:B+A:C+B:C+A:B:C`. In Minitab and SAS the model `A|B|C` is equivalent to `A B A*B C A*C B*C A*B*C`. □

This chapter describes general approaches to specifying fixed effect linear models in SAS. Chapter 3 in the book describes general approaches to statistical inference with Section 3.9 introducing

various linear models that are particularly useful. Most of this chapter is devoted to a discussion of how to specify those linear models in SAS. The chapter goes beyond those models because I think it is useful to consolidate in one place the fundamental ideas of specifying SAS models. It does not, however, discuss the random effects models that appear in Chapter 19.

Modeling in SAS is pretty much the same for all of the procedures we will focus on:

```
proc glm                                    /* Extensive ANOVA table output */
proc genmod              /* Coefficients, SSE, dfE, MSE (Error=Deviance) */
proc logistic
```

and we will also examine `proc reg`.

We assume that y is a measurement random variable and that x is some predictor variable or that $x \equiv (x_1, \ldots, x_p)'$ is a vector of predictor variables. *In a computer file* all of the observations on y consist of a column of numbers and the x observations are either a single column of numbers or p different columns of numbers, one column for each component of the vector x. The components of the vector x can either be measurement (continuous) variables, classification (categorical, factor, discrete) variables, or some combination of the two. We assumed in Section 3.9 of the book that

$$E(y) = m(x)$$

for some function m and described a number of different, commonly used, examples. When x contains only measurement variables, we construct regression models, when x contains only classification variables, we construct ANOVA models, when x contains a combination of the two, we construct ACOVA models.

Of course we have to tell the computer program whether any component of the vector x is a measurement or classification variable. Most computer programs have a default setting that, unless a variable is specified to be one thing, it is assumed to be the other. SAS assumes that all numeric variables are measurement variables, so classification variables taking on numeric values have to be specified as such. Variables that take nonnumeric values are **NOT** automatically taken as classifiers.

Also, most computer programs for linear models default to include an intercept term (grand mean) in every model.

A fairly generic `glm` program that illustrates most of the key features for three predictor variables, the first of which is a classifier, follows:

```
options ps=60 ls=80;
data Name;
   infile 'filename';
   input y x1 x2 x3;
proc print;
run;
proc glm data=Name;
   class x1
   model y = x1 x2 x3/ solution noint;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc print data=new;
proc plot;
   plot sr*yhat/  vpos=16 hpos=32;
run;
proc rank data=new normal=blom;
   var sr;
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
```

This includes printing of the standard diagnostic quantities that were discussed in Chapter 7 and Section 10.1 of the book, a crude plot of the standardized residuals versus the fitted values, and a crude normal plot. With minor modifications, this program also works for `prog genmod`, `proc logistic`, and `proc reg`. Most modifications will occur on the `model` line after the / with *solution* being unnecessary for other procedures and `genmod` requiring specifications of a `link` and a `distribution`.

If you have a lot of observations you might want to skip the first `proc print;` However, *one should always check that the data have been read in correctly* and SAS can be a little touchy about reading things.

The primary subject of this chapter is what to put in the `model` statement before the /.

3.9 One sample

If all the observations have the same mean value, you often have to use specialized software for this data structure, like SAS's `proc ??`.

Often linear models programs do not allow fitting *only* a common mean to all observations but you can generally trick them into fitting such a model by defining a predictor variable that is always 1 and not fitting a constant to the model. In SAS define the model via

```
J = y + 1 - y;
model y = J/noconst;
```

3.10 Two samples

For two samples, x should be a categorical variable with only two categories, specified by

```
class x;
```

To follow the discussion in Chapter 4 of the book, I recommend fitting the model

```
model y = x /noint;
```

The `/noint` in the model tells it not to fit an intercept, so that the parameter estimates become the group sample means. You should compare this output to that obtained by fitting

```
model y = x;
```

in which the parameter estimates are typically one group sample mean and the difference between the sample means of the two groups.

It turns out that if all you want is a test of whether the group means are equal and if the two categories are coded as numbers, you do not need to specify that x defines categories. The usual summary output for fitting

```
model y = x;
```

still gives the test of whether the groups differ. This trick does not work if there are more than two categories!

Chapter 4 of the book also considers some extensions, i.e., paired comparisons and unequal variances. The `glm` command assumes equal variances, so it does not apply to the latter. As discussed in Chapter 4, paired comparisons can be treated as either one-sample that consists of the differences between the pairs or as a two-way ANOVA without interaction.

3.11 Regression

In regression we assume that the vector x contains only measurement variables. In SAS, this is the default.

3.11.1 Simple linear regression

With a single predictor x, simple linear regression is

$$m(x) = \beta_0 + \beta_1 x$$

or

$$y_h = \beta_0 + \beta_1 x_h + \varepsilon_h, \qquad h = 1, \ldots, n.$$

It is modeled in SAS as

```
model y = x;
```

Notice that the intercept term is not specified and that this has the same structure as models used for two samples. The key point is that x is a measurement variable not a categorical variable. We will see later that this same SAS structure is used for one-way ANOVA when x is a classification variable with more than two categories.

If we wanted to force the regression through the origin the model becomes

$$y_h = \beta_1 x_h + \varepsilon_h, \qquad h = 1, \ldots, n.$$

and is coded in SAS as

```
model y = x / noint;
```

The model in question does not contain an intercept and the `/ noint` is used to stop an intercept being fitted.

When x is a measurement variable, the two models

```
model model y = x / noint;
model model y = x;
```

are different models. It turns out that if x is a classification variable,

```
model y = x / noint;
model y = x;
```

are equivalent models.

3.11.2 Polynomial regression

A cubic polynomial model

$$m(x) = \beta_0 + \beta_1 x + \beta_2 x^2 + \beta_3 x^3$$

or

$$y_h = \beta_0 + \beta_1 x_h + \beta_2 x_h^2 + \beta_3 x_h^3 + \varepsilon_h, \qquad h = 1, \ldots, n,$$

is modeled in SAS as

```
model y = x x*x x*x*x;
```

or as

```
x2 = x*x;
x3 = x2*x;
model y = x x2 x3;
```

where *the arithmetic needs to be done in the* `data` *step*.

Does SAS also have an option for fitting orthogonal polynomials?

3.11.3 Multiple regression

Now suppose x is a vector of measurement variables with $p = 3$, i.e., $x = (x_1, x_2, x_3)'$. The multiple regression model is

$$m(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3$$

or

$$y_h = \beta_0 + \beta_1 x_{h1} + \beta_2 x_{h2} + \beta_3 x_{h3} + \varepsilon_h, \qquad h = 1, \ldots, n.$$

Typically, when programming it is safer not to define variables using subscripts, so we write x_j as `xj` and the model is written in SAS as

```
model y = x1 x2 x3;
```

Notice that the intercept term is not specified.

3.11.4 Offsets

Suppose in the multiple regression model we knew that $\beta_2 = 5$ so that the model becomes

$$y_h = \beta_0 + \beta_1 x_{h1} + 5x_{h2} + \beta_3 x_{h3} + \varepsilon_h, \qquad h = 1, \ldots, n.$$

This can be modeled in proc `genmod` as

```
xoff =  5*x2
model y = x1 x3/ offset=xoff;
```

where *the arithmetic needs to be done in the `data` step* rather than the `prog genmod` step. This will not work in `proc glm`. (It probably also works for `proc logistic` but probably not for `proc reg`.)

We can rewrite the linear model as

$$y_h - 5x_{h2} = \beta_0 + \beta_1 x_{h1} + \beta_3 x_{h3} + \varepsilon_h, \qquad h = 1, \ldots, n.$$

which can be modeled in `proc glm`, `prog reg`, or in `proc genmod` when using `link=identity` as

```
yy = y - 5*x2
model yy = x1 x3;
```

Again, *the arithmetic needs to be done in the `data` step*.

For nonlinear "generalized linear models," such as those treated near the end of the book, one must use the "offset" command.

3.12 ANOVA

We begin by assuming that x is a single classification variable and then move on to vectors of classification variables. By default, SAS assumes that x is a measurement variable, so we have to specify that x is a classifier (even if x is nonnumerical). When x is a vector, we have to specify that each variable in the vector is a classifier.

3.12.1 One-way ANOVA

A typical one-way ANOVA model is written

$$y_{ij} = \mu_i + \varepsilon_{ij}, \qquad i = 1, 2, \ldots, a, \ j = 1, \ldots, N_i.$$

In this context, the generic predictor variable x should really be thought of as i, the subscript in the model that identifies the groups. To specify that x is categorical, write

```
class x;
```

But just to avoid any possible confusion, I prefer to define a new version of x that is categorical,

```
ii = x; class ii;
```

(In some contexts we might want to go back and forth between thinking of x as measurement or classification in which case it is handy to have both versions.) Note that `ii=x` needs to go in the `data` step, while `class ii` is part of the `proc`.

The model is specified as

```
model y = ii / noint;
```

The model in question does not contain an intercept with the `/ noint` used to stop an intercept from being fitted.

An alternative form of the one-way model that contains a grand mean (intercept) is what most programs are designed to fit,

$$y_{ij} = \mu + \alpha_i + \varepsilon_{ij}, \qquad i = 1,2,\ldots,a,\ j = 1,\ldots,N_i.$$

The model is written as

```
model y = ii;
```

For estimation SAS uses the side condition $\alpha_a = 0$ which forces the estimate of μ to be the sample mean of group a and the estimate of α_r to be the difference between the sample mean of group r and the sample mean of group a, $r = 1,\ldots,a$. As discussed in the book, other programs use other side conditions and it is important to be able to interpret the different output.

The one-way ANOVA with $a = 2$ model is identical to the model for two independent samples with equal variances and the model specification in SAS is the same.

3.12.2 Two-way ANOVA

Now suppose x is a vector of classification variables with $p = 2$, i.e., $x = (x_1, x_2)'$. As illustrated earlier, rewrite x_j as `xj`. Because `x1` and `x2` are classification variables, we specify

```
ii= x1; class ii;
jj= x2; class jj;
```

throughout this entire subsection. Models similar to those discussed below hold for $p > 2$ but get exponentially more complicated. The book examines in detail some cases with $p = 3$ and $p = 4$. Section 8 below also contains some discussion of higher-order models.

3.12.2.1 Interaction

The model

$$y_{ijk} = \mu_{ij} + \varepsilon_{ijk}, \qquad i = 1,\ldots,a,\ j = 1,\ldots,b,\ k = 1,\ldots,N_{ij},$$

is written in SAS as

```
model y = ii * jj / noint;
```

Remember that this is essentially a one-way ANOVA model!

Alternatively, the equivalent interaction model can be written

$$y_{ijk} = \mu + \alpha_i + \eta_j + (\alpha\eta)_{ij} + \varepsilon_{ijk}, \qquad i = 1,\ldots,a,\ j = 1,\ldots,b,\ k = 1,\ldots,N_{ij}.$$

and written in SAS as

```
model y = ii jj ii*jj;
```

SAS, unlike many programs, does not seem to have a shorthand way to specify the model. When estimating parameters for this overspecified model, the following side conditions are used by SAS,

$$\alpha_a = 0; \qquad \eta_b = 0; \qquad (\alpha\eta)_{aj} = 0, \quad j = 1,\ldots,b; \qquad (\alpha\eta)_{ib} = 0 \quad i = 1,\ldots,a.$$

3.12.2.2 Additive effects

The additive effects model is

$$y_{ijk} = \mu + \alpha_i + \eta_j + \varepsilon_{ijk}, \qquad i = 1,\ldots,a,\ j = 1,\ldots,b,\ k = 1,\ldots,N_{ij}.$$

and is written in SAS as

```
model y = ii jj;
```

When estimating parameters, the following side conditions are used by SAS,

$$\alpha_a = 0; \qquad \eta_b = 0.$$

3.12.2.3 Sequential fitting: Type I sums of squares

SAS has inflicted upon the world the abomination of Types I, II, III, and IV sums of squares. Type I sums of squares are *sequential sums of squares*. Type III sums of squares are *adjusted sums of squares*. "Sequential" and "adjusted" are the (far more intuitive) terms used by Minitab.

As discussed in the book, although the models themselves are equivalent, some computer output changes depending on whether one specifies the two-way interaction model as

```
model y = ii jj ii*jj;
```

or as

```
model y = jj ii jj*ii;
```

Although the end models are the same, the process of getting to the end model is different and the `proc glm` Type I sums of squares output contains information from the process in addition to information from the end result. For unbalanced data, these models typically give different numbers for the `proc glm` Type I sums of squares. For balanced data, only the order of presentation changes. Parameter estimation depends only on the end model and does not change materially with different orderings of terms.

In particular, when you ask `proc glm` to fit

```
model y = ii jj ii*jj;
```

it sequentially fits the models

```
model y = ii;
model y = ii jj;
model y = ii jj ii*jj;
```

In general, `proc glm` fits a sequence of models determined by the order in which you added terms of the larger model! This is reflected in the Type I sums of squares. Incidentally, it is irrelevant in these models whether you specify `ii jj ii*jj` or specify `ii jj jj*ii`.

For Type I sums of squares, `proc glm` treats `ii*jj ii jj` as equivalent to `ii*jj` because fitting an `ii` term or a `jj` term after an `ii*jj` term actually contributes nothing.

Although this discussion has focused on the two-way with interaction model, issues of sequential fitting are pervasive with unbalanced data (including regression). This is not so much a problem as an opportunity. The extra computer output can save you from the chore of fitting some models. It never does any harm unless you misinterpret the results.

As discussed in Section 14.1 of the book, there is also a concept of *adjusted sums of squares*. In `proc glm` these are known as Type III sums of squares. In general, adjusted sums of squares are meant to be sums of squares for fitting terms last. In the additive effects model, the adjusted sums of squares for each main effect is the sequential sum of squares for that term when the term is being fitted to the model after the other main effect. However, *in the interaction model, most of the adjusted sums of squares are worthless*. The interaction adjusted sum of squares is fine, it is just the sum of squares for adding the interaction after both of the main effects. However, the sum of squares for a main effect fitted after an interaction should be zero, not the arbitrary numbers

reported for main effects as adjusted sums of squares. The reported numbers are arbitrary because they depend on the choice of side conditions being used by the program to determine estimates. The side conditions are arbitrary choices, so these adjusted sums of squares are also arbitrary. An illustration of this phenomenon appears in Subsubsection 14.1.1.1 of this guide.

3.13 ACOVA and interaction

The following models were not discussed in Section 3.9 of the book but rather are discussed in Chapter 15. They involve a classification variable x_1 written as `x1` and a measurement variable x_2 written as `x2` *or equivalently as* `z` (to make the notation more similar to the book). Throughout the section we assume

```
z=x2; ii = x1; class ii;
```

The first two of these need to go in the `data` step of your SAS program.

3.13.1 ACOVA: parallel lines

The ACOVA model is

$$y_{ij} = \mu_i + \gamma z_{ij} + \varepsilon_{ij}, \qquad i = 1,2,\ldots,a,\ j = 1,\ldots,N_i.$$

written in SAS as

```
model y = ii z / noint;
```

The a lines associated with the groups have intercepts μ_i and a common slope γ.

The alternative model

$$y_{ij} = \mu + \alpha_i + \gamma z_{ij} + \varepsilon_{ij}.$$

is written in SAS as

```
model y = ii z;
```

When estimating parameters, the side condition $\alpha_a = 0$ is used by SAS so that μ is the intercept of the line for the first group, α_r is the intercept of the rth group minus the intercept of the last, in other words, $\mu + \alpha_r$ is the intercept of the rth group, and again γ is the slope for all of the lines.

Note that this second SAS model is essentially the same form as that used for the two-way additive effects model in Subsubsection 3.4.2.2 except now one of the predictor variables is a measurement variable and the other is a classification variable.

3.13.2 Interaction: skew lines

We now consider ways of specifying lines for each group that can have different slopes as well as different intercepts. The simplest model to interpret is

$$y_{ij} = \mu_i + \gamma_i z_{ij} + \varepsilon_{ij}.$$

In this model μ_i is the intercept and γ_i is the slope for the line associated with the i group. The model in SAS is

```
model y = ii ii*z / noint;
```

The version most similar to generalizing the traditional ACOVA model is

$$y_{ij} = \mu + \alpha_i + \gamma_i z_{ij} + \varepsilon_{ij}$$

in which the line for the ith group has intercept $\mu + \alpha_i$ and slope γ_i. Written in SAS this model is

```
model y = ii z*ii;
```

and uses the side condition $\alpha_a = 0$ for estimation. It does not matter if we write `z*ii` or `ii*z`.

Some folks like to specify a "hierarchical" version of the model (which I generally think is pretty silly but admittedly gives some useful computer output),

$$y_{ij} = \mu + \alpha_i + \beta z_{ij} + \gamma_i z_{ij} + \varepsilon_{ij}$$

written in SAS as

```
model y = ii z ii*z;
```

For this model SAS uses $\alpha_a = 0 = \gamma_a$. More on hierarchical models in Section 7.

3.14 Interaction in multiple regression

The multiple regression model with $p = 2$ and continuous predictors $x = (x_1, x_2)$ has

$$m(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

or

$$y_h = \beta_0 + \beta_1 x_{h1} + \beta_2 x_{h2} + \varepsilon_h, \qquad h = 1, \ldots, n.$$

This is an additive model in x_1 and x_2, cf. Section 9.9 of the book. The SAS model for this

```
model y = x1 x2;
```

is essentially the same as that used for the two-way ANOVA additive effects model and the parallel lines ACOVA model except that now both of the predictor variables are measurement variables. Moreover, the effects of the two predictors simply add together.

A useful way of incorporating some interaction into the model is to fit

$$m(x) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_1 x_2,$$

that is,

$$y_h = \beta_0 + \beta_1 x_{h1} + \beta_2 x_{h2} + \beta_3 x_{h1} x_{h2} + \varepsilon_h, \qquad h = 1, \ldots, n.$$

This is no longer an additive model in the original variables x_1 and x_2 (although it *is* an additive model in the variables x_1, x_2, x_1x_2). The main reason for introducing this model here is to illustrate that the SAS model has essentially the same structure as the SAS models for two-way ANOVA with interaction and the ACOVA interaction model with skew lines. The SAS command is

```
model y = x1 x2 x1*x2;
```

(This may not work in `proc reg`.)

With both variables being measurement variables there is yet another option for fitting the model,

```
x1x2 = x1*x2;
model y = x1 x2 x1x2;
```

As usual, the arithmetic needs to be done in the `data` step.

3.15 Hierarchical and nested models

The model

$$y_{ijk} = \mu + \alpha_i + \eta_j + (\alpha\eta)_{ij} + \varepsilon_{ijk}, \qquad i = 1, \ldots, a,\ j = 1, \ldots, b,\ k = 1, \ldots, N_{ij}.$$

is said to be hierarchical because the model includes not only the interaction terms $(\alpha\eta)_{ij}$ but also the lower order "main effect" terms α_i and η_j as well as the grand mean μ. A model is hierarchical whenever a model that includes an mth order interaction between some variables also includes all of the lower order interactions among those variables as well as all the main effects. Thus, if a model contains $(\alpha\eta\gamma)$ terms, it must also include $(\alpha\eta)$ terms, $(\alpha\gamma)$ terms, $(\eta\gamma)$ terms, and the

main effect terms, the αs, ηs, and γs, as well as the grand mean μ. I have been at some pains in the book to point out that all of these lower order terms are meaningless, so I have little regard for this concept of a hierarchical model. (The term "hierarchical" takes on alternate meanings in alternate contexts, e.g., Bayesian statistics.) Some programs, like Minitab, insist that all models be hierarchical, using a similar concept of hierarchy for measurement variables. Minitab and R incorporate special operators to simplify defining hierarchical models. (As far as I know) SAS does not! In SAS the model displayed earlier must be written as

```
model y = x1 x2 x1*x2
```

where `x1` and `x2` are the classification variables that define i and j, respectively.

I should point out that when using the rather artificial (but computationally very convenient) concept of interaction between continuous (measurement) variables employed by all three of Minitab, R, and SAS, the lower order terms are **NOT** meaningless.

If you care about hierarchical models, you might care about nested models which are models that are not quite hierarchical. For example,

$$y_{ijk} = \mu + \alpha_i + (\alpha\eta)_{ij} + \varepsilon_{ijk}, \qquad i = 1,\ldots,a,\ j = 1,\ldots,b,\ k = 1,\ldots,N_{ij}$$

is said to have the $(\alpha\eta)_{ij}$ effects nested within the α_i effects. Again, fixed α_i effects are irrelevant in this model, so I find it hard to care about this concept of nesting. Nonetheless, SAS has specific code for defining nested models. With `class x1 x2;` the model above could be written

```
model y = x1(x2);    ????
```

which by definition is the same model as

```
model y = x1   x1*x2; ???
```

3.16 Higher-order models

This section has not been validated.

When discussing SAS, we will consider "higher order" models to be those that involve $p > 2$, although the distinction is probably more common only when the number of classification variables is greater than 2. The reason we do this is because the methods in SAS for defining higher order models are pretty much interchangeable regardless of whether the predictors are measurements or classifications. When used on measurement variables, the SAS commands introduce interaction similar to Section 3.6 but do not introduce polynomial (power) terms.

For example, the code

```
model y = (x1 x2 x3)@2;
```

should fit something!

From the first two ways of writing the code you might be tempted to think that if all the predictors are measurements the model is

$$y_h = \sum_{r=0}^{2}\sum_{s=0}^{2}\sum_{t=0}^{2} \beta_{rst} x_{h1}^r x_{h2}^s x_{h3}^t + \varepsilon_h, \qquad h = 1,\ldots,n. \tag{1}$$

It is not!? The model is actually

$$y_h = \beta_{000} + \beta_{100}x_{h1} + \beta_{010}x_{h2} + \beta_{001}x_{h3} + \beta_{110}x_{h1}x_{h2} + \beta_{101}x_{h1}x_{h3} + \beta_{011}x_{h2}x_{h3} + \varepsilon_h, \qquad h = 1,\ldots,n.$$

When all the predictors are classifications with $x_1 \equiv i$, $x_2 \equiv j$, and $x_3 \equiv k$ the code means

$$y_{ijks} = \mu + \alpha_i + \eta_j + \gamma_k + (\alpha\eta)_{ij} + (\alpha\gamma)_{ik} + (\eta\gamma)_{jk} + \varepsilon_{ijks}, \qquad s = 1,\ldots,N_{hij}.$$

If only `x1` is a measurement variable, it means

$$y_{jks} = \mu + \beta_1 x_{jks1} + \eta_j + \gamma_k + \beta_{1j}x_{jks1} + \beta_{1k}x_{jks1} + (\eta\gamma)_{jk} + \varepsilon_{jks}, \qquad s = 1,\ldots,N_{jk}$$

If both `x1` and `x2` are measurement variables, it means

$$y_{ks} = \beta_{00} + \beta_{10}x_{ks1} + \beta_{01}x_{ks2} + \gamma_k + \beta_{11}x_{ks1}x_{ks2} + \beta_{10k}x_{ks1} + \beta_{01k}x_{ks2} + \varepsilon_{jks}, \qquad s = 1,\ldots,N_k$$

The models

```
model y = (x1 x2 x3)@3
model y = x1*x2*x3
```

are also equivalent.

Similarly, we can fit mth order interaction models, where m is a positive integer no greater than p using, say, for $p = 4$,

```
model y = (x1 x2 x3 x4)@m
```

With $p = 4$, the all measurement variable model is **not**

$$y_h = \sum_{r=0}^{m}\sum_{s=0}^{m}\sum_{t=0}^{m}\sum_{u=0}^{m}\beta_{rstu}x_{h1}^r x_{h2}^s x_{h3}^t x_{h4}^u + \varepsilon_h, \qquad h = 1,\ldots,n.$$

The actual model for $m = 3$ and $p = 4$ has third order interactions and is

$$\begin{aligned} y_h \quad = \quad & y_h = \beta_{000} + \beta_{1000}x_{h1} + \beta_{0100}x_{h2} + \beta_{0010}x_{h3} + \beta_{0001}x_{h4} \\ & + \beta_{1100}x_{h1}x_{h2} + \beta_{1010}x_{h1}x_{h3} + \beta_{1001}x_{h1}x_{h4} + \beta_{0110}x_{h2}x_{h3} + \beta_{0101}x_{h2}x_{h4} + \beta_{0011}x_{h3}x_{h4} \\ & + \beta_{1110}x_{h1}x_{h2}x_{h3} + \beta_{1101}x_{h1}x_{h2}x_{h4} + \beta_{1011}x_{h1}x_{h3}x_{h4} + \beta_{0111}x_{h2}x_{h3}x_{h4} + \varepsilon_h, \qquad h = 1,\ldots,n. \end{aligned}$$

The ANOVA model that use all classifications is

$$\begin{aligned} y_{hijks} \quad = \quad & \mu + \alpha_h + \eta_i + \gamma_j + \delta_k \\ & + (\alpha\eta)_{hi} + (\alpha\gamma)_{hj} + (\alpha\delta)_{hk} + (\eta\gamma\delta)_{ij} + (\eta\delta)_{ik} + (\gamma\delta)_{jk} \\ & + (\alpha\eta\gamma)_{hij} + (\alpha\eta\delta)_{hik} + (\alpha\gamma\delta)_{hjk} + (\eta\gamma\delta)_{ijk} + \varepsilon_{hijks}, \qquad s = 1,\ldots,N_{hijk}. \end{aligned}$$

I will leave it to you to worry about the various kinds of ACOVA models that mix measurements with classifications.

Chapter 4

Two Samples

There are two ways to specify the two samples of data. You can either have one vector for each sample, say y_1 and y_2, or you can have one vector for all the data, say y, but also have an associated classification variable, say x, that identifies the sample for each observation. To use use a linear model program to analyze the data, the (x,y) data structure is needed (except for paired comparisons).

4.1 Two correlated samples: paired comparisons

You need to know not only what sample an observation is in but also what pair it is in, so the (x,y) data structure is awkward for these problems. Using the y_1 and y_2 data structure, the vectors will have to have the same length and it is just assumed that the ith entry of y_1 is paired with the ith entry of y_2. One way to test equality of means is to use

4.1.1 Minitab

Enter the data from `tab4-1.dat` with variables `Pair`, `y1`, and `y2`. Choose `Stat` from the top line and within the menu options choose `Basic Statistics`. This provides an option for `paired t` that allows data to be entered either as two columns of numbers or one column of differences.

You can also handle this as a one sample problem on the differences using a regression program.

```
let c11 =  y1 + 1 - y1
let c12 = y1 - y2
name c11 'J' c12 'd'
GReg 'd' = J;
  NoConstant;
  Confidence 95.0;
  PContinuous 1;
  TPrediction;
  TCoef;
  TANOVA.
```

`GReg` is easier to run from its menus.

4.1.2 SAS

There is probably specialized software available be we can use the following linear model approach.

```
data Weld;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\tab4-1.dat';
   input Pair y1 y2;
   d = y1-y2
   J = y1 + 1 - y1
```

```
proc glm data=Weld;
   model d = J/ solution noint;
run;
```

4.2 Two independent samples with equal variances

Both data structures work fine for this problem, but if you have unequal group sizes with the (y_1, y_2) structure you might need to fill out the group having a smaller sample size with missing data symbols to make the data vectors the same length.

4.2.1 Minitab

Enter the data from `tab4-2.dat` and name the first two columns `x` and `y` **or** enter the data from `tab4-2a.dat` with variables `Obs`, `y1`, and `y2`. Choose `Stat` from the top line and within the menu options choose `Basic Statistics`. This provides an option for `2 Sample t` tests. *To get an equal variances analysis you need to check the appropriate box!*

Alternatively, you could use the more general linear modeling methods to analyze the data. Read `tab4-2.dat` with variables `x` and `y`. Choose `Stat` from the top line and within the menu options choose `ANOVA`. This provides an option for fitting `glm` (general linear model). Specify `y` as the dependent variable and `x` as the model. In fact, if all you care about is the t test, you could use the regression menus rather than `ANOVA`'s `glm`.

For two samples, x should be a categorical variable with only two categories. Use

```
glm y = x;
  Brief 3.
```

The subcommand `Brief 3` is necessary to get a table of coefficients. To specify `brief 3` in the `glm` menu, go to the `Results` submenu and check the last circle under `Display of Results`. Alternatively, use

```
GReg y = x;
  Covariate x;
  TCoef;
  TANOVA.
```

In either case, the first group mean is the sum of the two coefficients and second group mean is the difference in the two coefficients. The t test for the second coefficient is the test for equality of means.

4.2.2 SAS

Before addressing specialized SAS software, we consider the linear models approach.

```
data pts;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\tab4-2.dat';
   input x y ;
proc glm;
   class x;
   model y = x / solution noint;
proc glm;
   class x;
   model y = x / solution;
run;
```

The following code does nothing more than read in the two data structures. I expect in the future to figure out how to use specialized software from SAS on these.

```
data pts;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\tab4-2.dat';
   input x y ;
proc print;
run;
```

```
data pts;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\tab4-2a.dat';
   input Obs y1 y2 ;
proc print;
run;
```

4.3 Two independent samples with unequal variances

Assuming unequal variances, the linear models approach does not apply (unless you know a constant of proportionality between the variances – are rare event). Specialized software is required. As discussed in the book, testing equality of means when the variances are unequal is a far less interesting thing to do than testing equality of means when the variances are equal.

4.3.1 Minitab

Enter the data from `tab4-3.dat` and name the first three columns `Index`, `y1`, and `y2`. Choose `Stat` from the top line and within the menu options choose `Basic Statistics`. This provides options for `2 Sample t` tests. The default is unequal variances.

4.3.2 SAS

I am not familiar with the specialized software needed. The following code just reads in the data.

```
data turt;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\tab4-3.dat';
   input Index y1 y2;
proc print;
run;
```

4.4 Testing equality of the variances

4.4.1 Minitab

Enter the data from `tab4-3.dat` with variables `Index`, `y1`, and `y2`. Choose `Stat` from the top line and within the menu options choose `Basic Statistics`. This provides an options for testing two variances: `2 Variances`.

The only thing a computer is needed for is to get the F percentiles reported in Example 4.4.1 of the book. The commands are brief

```
invcdf .995;
f 23 23.
invcdf .005;
f 23 23.
```

The menu approach is actually easier because you do not have to remember syntax, even though it takes much longer to describe.

```
Calc
Probability Distributions
F
```

Once you are in the `F` menu, check the circle for `Inverse cumulative probability`. Enter the numbers (23 and 23) into the dialog boxes for `Numerator degrees of freedom` `Denominator degrees of freedom`. Check the circle for `Input Constant` and enter .995 (or .005) into the dialog box.

4.4.2 SAS

I am not familiar with the specialized software required.

Chapter 5

Contingency Tables

5.1 One binomial sample

5.1.1 Minitab

Choose `Stat` from the top line and within the menu options choose `Basic Statistics`. This provides an option for `1 Proportion`.

5.1.2 SAS

5.2 Two independent binomial samples

5.2.1 Minitab

Choose `Stat` from the top line and within the menu options choose `Basic Statistics`. This provides an option for `2 Proportions`.

5.2.2 SAS

5.3 One multinomial sample

5.4 Two multinomial samples

5.4.1 Minitab

Check out

`Stat`
`Tables`

Otherwise, Minitab commands for generating the analysis of Swedish birth rates are given below. Column c1 contains the observations, the O_{ij}s. Column c2 contains indices from 1 to 12 indicating the month of each observation and c3 contains indices for the two sexes. The subcommand 'colpercents' provides the proportions discussed in the analysis. The subcommand 'chisquare 3' gives the observations, estimated expected values, and Pearson residuals along with the Pearson test statistic.

```
read 'tab5-3a.dat' c1 c2 c3
table c2 c3;
frequencies c1;
colpercents;
chisquare 3.
```

5.4.2 SAS

```
options ps=60 ls=80 nodate;
data tab8-3;
```

```
infile 'tab5-3a.dat';
input O M S;
proc genmod data=tab8-3;
   class M S;
model O = M S / link=log obstats dist=poisson;
run;
```

5.5 Several independent multinomial samples

5.5.1 Minitab

Check out

```
Stat
Tables
```

5.5.2 SAS

Chapter 6

Simple Linear Regression

6.1 An example

6.1.1 Minitab

We can do this in `glm` or `GReg` as illustrated in Chapter 3 but this code is for `Regress`.

Commands given below generate the table of coefficients and the analysis of variance table. Column c3 contains the test scores y and column c2 contains the composite socioeconomic statuses x. The primary command is to regress c3 on 1 predictor variable, c2. This same command allows for more predictor variables and we will use that capability in subsequent chapters on regression. In our example, the subcommand 'predict -16.04' was used; this subcommand gives the estimate of the line (prediction) when $x = -16.04$, the standard error for the estimate of the line, the 95% confidence interval for the value of the line at $x = -16.04$, and the 95% prediction interval when $x = -16.04$.

```
name c2 'test' c3 'socio'
regress c3 1 c2;
predict -16.04.
```

Below are the complete commands (generated by the menus) for reading in the data and getting the table of coefficients, ANOVA table, a prediction, and the primary diagnostic statistics defined in Chapters 7 and 9.

```
WOpen "C:\E-drive\Books\ANREG2\NewData\TAB6-1.DAT";
  FType;
    Text;
  VNames;
    None;
  DecSep;
    Period;
  TDelimiter;
    DoubleQuote.
Name c4 "RESI1" c5 "SRES1" c6 "TRES1" c7 "HI1" c8 "COOK1"
Regress 'y' 1 'x';
  Residuals 'RESI1';
  SResiduals 'SRES1';
  Tresiduals 'TRES1';
  Hi 'HI1';
  Cookd 'COOK1';
  GFourpack;
  RType 1;
  Constant;
  Predict -16.04;
  Brief 2.
```

Columns 4, 5, 6, 7 and 8 contain the residuals $\hat{\varepsilon}_i$, the standardized residuals r_i, the standardized deleted residuals t_i, the leverages h_i (or m_{ii})m and Cook's distances C_i.

6.1.1.1 Regression through the origin

The Minitab commands for fitting a regression through the origin are given below.

```
name c1 'test' c2 'socio'
regress c1 1 c2;
noconstant.
```

The subcommand `noconstant` can be replaced by `nocon`.

6.1.2 SAS

SAS has at least three different procedures that will do this. The following does NOT display `proc reg`.

```
options ps=60 ls=80 nodate;
data coleman;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\tab6-1.dat';
   input School x y ;
print coleman;
proc glm data=coleman;
   model y = x;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc plot;
   plot ehat*yhat/  vpos=16 hpos=32;
proc rank data=new normal=blom;
   var sr;
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
proc genmod data=coleman;
model y = x/link=identity dist=normal obstats;
run;
```

Regression through the origin

```
options ps=60 ls=80 nodate;
data coleman;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\tab6-1.dat';
   input School x y ;
proc glm data=coleman;
model y = x/ noint;
```

6.6 An alternative model

6.6.1 Minitab

6.6.2 SAS

6.7 Correlation

6.7.1 Minitab

```
corr c2 c3}
```

6.7.2 SAS

6.8 Two sample problems

6.8.1 Minitab

6.8.2 SAS

6.9 A multiple regression

6.9.1 Minitab

This is easy. Read in `tab6-4.dat` with variables `input School x1 x2 x3 x4 x5 y`. See Subsection 3.3.3 for a general description. If you want to use regress,you need to specify the number of predictor variables:

```
Regress 'y' 5 'x1' x2 x3 x4 x5
```

As far as I can tell the single quotes do not seem to matter (outside of the `name` command). In fact, I cannot even figure out when Minitab uses them and when it doesn't.

6.9.2 SAS

```
options ps=60 ls=80 nodate;
data coleman;
   infile 'tab6-4.dat';
   input School x1 x2 x3 x4 x5 y ;
proc print;
run;
proc glm data=coleman;
   model y = x1 x2 x3 x4 x5;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc print data=new;
proc plot;
   plot sr*yhat /  vpos=16 hpos=32;
proc rank data=new normal=blom;
   var sr;
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
proc genmod data=coleman;
model y = x1 x2 x3 x4 x5/link=identity dist=normal obstats;
proc print data=obstats;
run;
```

Chapter 7

Model Checking

7.1 Recognizing Randomness

7.1.1 Minitab

The plots and sample correlations similar to these can be obtained with the Minitab commands given below.

```
random 25 c10-c15;
normal 0 1.
plot c11*c15
plot c12*c15
plot c13*c15
plot c14*c15
plot c11*c12
plot c13*c12
plot c14*c12
plot c14*c13
note     OBTAIN SAMPLE CORRELATION MATRIX
corr c11-c15
```

7.1.2 SAS

7.2 Checking assumptions: residual analysis

7.2.1 Minitab

The beginning of Chapter 3 discusses general commands for `glm` and `greg` including storage of diagnostic statistics. It is very easy to plot the stored residuals etc. using the `Graph` menu but it is even easier to ask for residual plots on the various menus like `glm`, `greg`, and `regress`.

The following are older commands based on regress. We now illustrate the Minitab commands necessary for obtaining the leverages and standardized deleted residuals for the *Coleman Report* data. Both sets of values are obtained by using subcommands of the regress command. The 'hi' subcommand gives leverages, while the 'tresid' subcommand gives standardized deleted residuals. The last command gives the index plot for leverages.

```
names c1 'test' c2 'socio'
regress c1 1 c2;
hi c13;
tresid c14.
tsplot c13
```

We now illustrate the Minitab commands necessary for the analysis in Example 7.2.1.

```
names c1 'test' c2 'socio'
regress c1 1 c2;
```

```
   sresid c11;
   fits c12.
   names c11 'r' c12 'yhat'
   note     PLOT STD. RESIDS AGAINST PRED. VALUES
   plot c11*c12
   note     PLOT STD. RESIDS AGAINST x
   plot c11*c2
   note     COMPUTE NORMAL SCORES (RANKITS) FOR THE
   note     STANDARDIZED RESIDUALS
   nscores c11 c10
   note     MAKE NORMAL PLOT
   plot c11*c10
   note     COMPUTE W' STATISTIC
   corr c11 c10
   note     CORR PRINTS OUT A NUMBER LIKE 0.978
   let k1=.978**2
   print k1
```

7.2.2 SAS

This is just the standard program from Chapter 3 applied to the data of Section 6.1.

```
options ps=60 ls=80 nodate;
data coleman;
   infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\tab6-1.dat';
   input School x y ;
print coleman;
proc glm data=coleman;
   model y = x;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc plot;
   plot ehat*yhat sr*R/  vpos=16 hpos=32;
proc rank data=new normal=blom;
   var sr;
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
proc genmod data=coleman;
model y = x/link=identity dist=normal obstats;
run;
```

7.3 Transformations

We talked about basic transformations in Chapters 1 and 2. We focus on Box-Cox.

7.3.1 Minitab

Check out the Box-Cox menu in GReg. Glm doesn't really do it. But you can always brute force it as follows.

Below are given Minitab commands for performing the Box–Cox transformations and the constructed variable test. To perform multiple regression using the 'regress' command, you need to specify the number of predictor variables, in this case 2.

```
   name c1 'temp' c2 'press'
   note     CONSTRUCT THE GEOMETRIC MEAN
   let c9 = loge(c2)
   mean c9 k1
    MEAN    =      2.9804
   let k2 = expo(k1)
   note     PRINT THE GEOMETRIC MEAN
   print k2
 K2       19.6960
   note     CONSTRUCT THE z VARIABLES
   note     FOR DIFFERENT LAMBDAS
   let c20=(c2**.5-1)/(0.5*k2**(.5-1))
   let c21=(c2**.25-1)/(.25*k2**(.25-1))
   let c22=(c2**.333333-1)/(.333333*k2**(.333333-1))
   let c23=loge(c2)*k2
   let c24=(c2**(-.5) - 1)/(-.5*k2**(-1.5))
   let c25=(c2**(-.25) -1)/(-.25*k2**(-1.25))
   note     REGRESS z FOR LAMBDA = 0.5 ON c1
   regress c20 1 c1
   note     4 MORE REGRESSIONS ARE NECESSARY
   note
   note     CONSTRUCT THE VARIABLE w
   let c3=c2*(c9-k1-1)
   note     PERFORM THE MULTIPLE REGRESSION ON x AND THE
   note     CONSTRUCTED VARIABLE w
   regress c2 2 c1 c3
```

7.3.2 SAS

This program is NOT doing Box-Cox.

```
options ps=60 ls=80;
data hook;
   infile 'tab7-1.dat';
   input Case Temp Pres;
proc print;
run;
proc glm data=hook;
   model Pres = Temp/ solution;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc print data=new;
proc plot;
   plot sr*yhat/  vpos=16 hpos=32;
run;
proc rank data=new normal=blom;
   var sr;
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
```

Chapter 8

Lack of Fit and Nonparametric Regression

8.1 Polynomial regression

8.1.1 Minitab

The cubic model in `glm` is

```
glm y = x x*x x*x*x;
covar x3.
```

The quartic model in `greg` is

```
greg y = x x*x x*x*x x*x*x*x;
  tcoef;
  tanova.
```

Specialized program for fitting polynomials up to degree 3 is

```
Fitline y x;
  Poly 2.
```

In what follows we illustrate `regress` commands for fitting quadratic, cubic, and quartic models. These include the prediction subcommand used with the quadratic model for $x = 205$. Note that the prediction subcommand requires us to enter both the value of x and the value of x^2 when using the quadratic model.

```
names c1 'y' c2 'x'
note     FIT QUADRATIC MODEL
let c22=c2**2
regress c2 2 c2 c22;
pred 205 42025.
note     FIT CUBIC MODEL
let c23=c2**3
regress c1 3 c2 c22 c23
note     FIT QUARTIC MODEL
let c24=c2**4
regress c1 4 c2 c22-c24
```

8.1.2 SAS

```
options ps=60 ls=80;
data hook;
   infile 'tab7-1.dat';
   input Case Temp Pres;
proc print;
run;
proc glm data=hook;
   model Pres = Temp Temp*Temp/ solution;
```

```
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc print data=new;
proc plot;
   plot sr*yhat/  vpos=16 hpos=32;
run;
proc rank data=new normal=blom;
   var sr;
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
```

8.2 Polynomial regression and leverages

There are no new computing skills involved in this section.

8.3 Other basis functions

8.3.1 SAS

SINES AND COSINES

```
options ps=60 ls=80;
data hook;
   infile 'tab7-1.dat';
   input Case Temp Pres;
rg=30.5;
mn=180.5;
x=(Temp-mn)/rg;
c1=cos(pi*1*x);
c2=cos(pi*2*x);
s1=sin(pi*1*x);
s2=sin(pi*2*x);
run;
proc glm data=hook;
   model Pres =  Temp  c1  s1  c2  s2/ solution;
run;
```

COSINES

```
options ps=60 ls=80;
data hook;
   infile 'tab7-5.dat';
   input Temp Pres;
rg=30.5;
mn=180.5;
x=(Temp-mn)/rg;
c1=cos(pi*1*x);
c2=cos(pi*2*x);
c3=cos(pi*3*x);
c4=cos(pi*4*x);
run;
```

```
proc glm data=hook;
   model Pres =  Temp  c1  c2  c3 c4/solution;
run;
```

HAAR WAVELETS **Laura Kapitula from Grand Valley State University contributed the following (because I did not know how to do it).**

Laura writes, " I used arrays below, but you do not have to use arrays, I also kept in the equivalent code commented out. Note that SGPLOT is an ODS graphics procedure and is available with BASE SAS and as part of SAS Univ. edition if anyone uses that."

```
/*Indicator Variables in SAS*/
data hook;
  set hook;
rg=30.5;
mn=180.5;
x=(Temp-mn)/rg;
array h(4);
do i=1 to 4;
   h(i)=((i-1)/4 <=x <i/4);
end;
/* above is equivalent to*/
/*
h1=(0<=x<.25);
h2=(.25<=x<.5);
h3=(.5<=x<.75);
h4=(x>=.75);
*/
run;

ods graphics;
proc glm plots=all;
model Pres = Temp h1 h2 h3 ;
output out=fits predicted=pred residual=resid;
run;

proc sgplot data=fits;
  scatter x=temp y=pres;
  series x=temp y=pred;
  label pres= "Pressure" pred="Fit from Haar Wavelet"
  x="Temperature";
run;

proc sgplot data=fits;
  scatter x=temp y=resid;
    label pres= "Pressure" pred="Fit from Haar Wavelet"
  x="Temperature";
run;
```

8.4 Partitioning methods

8.4.1 Minitab

We give three different sets of Minitab commands that provide the last of our analyses. In the text we discussed the output from fitting the regression command. Alternatively, one can fit this model using a general linear model procedure like Minitab's glm.

```
regress Pres 3 x h x1
glm c2 = h h*x;
covar x.
glm Pres = h x h*x;
covar x.
```

Note that in the glm commands, there does not exist a variable called 'h*x'. This is a term that we are telling glm to construct out of two variables that do exist. The only difference between the two "glm" commands, is that when the model does not specify an x term, glm fits x before fitting h. The table of coefficients provided by the two glm commands are the same but different from that provided by regress. The glm command by default only gives results for the intercept and terms that involve the covariate x.

8.4.2 SAS

Have an alternate data file with the partitioning variable.

```
options ps=60 ls=80 nodate;
data lsq;
   infile 'tab7-1a.dat';
   input Case h x y ;
proc print data=lsq;
var h x y;
proc glm data=lsq;
   class h;
   model y = x h h*x ;
proc genmod data=lsq;
   class h;
   model y = h x h*x/ link=id dist=normal;
run;
```

The following seems to be R code that needs to be straightened out for SAS. The first command is just an R logical command to create the variable h that was read in from tab7-5a.dat.

```
H1=1*as.logical(Temp>=191)
H2=1-H1
TH1=Temp*H1
TH2=Temp*H2
```

8.4.3 Utt's Method

Chapter 9

Multiple Regression and Diagnostics

9.1 Example

9.1.1 Minitab

Of course you can also do this in `glm` and `greg`.

To obtain the statistics in this section, use the Minitab command 'regress.' The form of the command has regress, the *y* variable, the number of predictors (excluding the intercept), and a list of the predictor variables.

```
names c1 'x1' c2 'x2' c3 'x3' c4 'x4' c5 'x5' c6 'y'
regress c6 on 5 c1-c5
```

Below are given the Minitab commands for obtaining the diagnostics. On the 'regress' line, 5 predictors are specified, so the next 5 columns are taken to contain the predictor variables. The standardized residuals are placed in the next column listed and the predicted values are placed in the column listed after that. Thus the standardized residuals are in c21 and the predicted values are in c22. The subcommands 'tresid', 'hi', and 'cookd' indicate the standardized deleted residuals, leverages, and Cook distances, respectively. The *t* statistics are in c23. The leverages are in c24. The Cook distances are in c25.

```
regress c8 on 5 c2-c6 c21 c22;
tresid c23;
hi c24;
cookd c25.
```

To delete a case in Minitab, just replace the dependent variable value in the worksheet with an asterisk.

9.1.2 SAS

```
options ps=60 ls=80 nodate;
data coleman;
   infile 'tab6-4.dat';
   input School x1 x2 x3 x4 x5 y ;
proc print;
run;
proc glm data=coleman;
   model y = x1 x2 x3 x4 x5;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc print data=new;
proc plot;
   plot sr*yhat /  vpos=16 hpos=32;
proc rank data=new normal=blom;
   var sr;
```

```
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
```

or

```
proc genmod data=coleman;
model y = x1 x2 x3 x4 x5/link=identity dist=normal obstats;
proc print data=obstats;
run;
```

9.2 Predictions

9.2.1 Minitab

To obtain the predictions in this section, use the Minitab subcommand 'predict.'

```
   names c1 'x1' c2 'x2' c3 'x3' c4 'x4' c5 'x5' c6 'y'
   regress c6 on 5 c1-c5;
   predict 2.07 9.99 -16.04 21.6 5.17 .
```

9.2.2 SAS

This is still R

```
#Predictions
new = data.frame(x1=2.07, x2=9.99,x3=-16.04,x4= 21.6, x5=5.17)
predict(lm(y~x1+x2+x3+x4+x5),new,se.fit=T,interval="confidence")
predict(lm(y~x1+x2+x3+x4+x5),new,interval="prediction")
```

If response is a matrix a linear model is fitted separately by least-squares to each column of the matrix.

Chapter 10

Diagnostics and Variable Selection

Probably do these things in `proc reg`

10.1 Diagnostics

We did this in the last chapter.

10.2 Best subset model selection

10.2.1 Minitab

You can do this through the `Regression` menu.

Below are given Minitab commands for obtaining Table 10.8.

```
breg c8 on c2-c6;
best 2.
```

10.2.2 SAS

```
options ps=60 ls=80 nodate;
data coleman;
   infile 'tab6-4.dat';
   input School x1 x2 x3 x4 x5 y ;
proc print;
run;
proc reg data=coleman;
   model y = x1 x2 x3 x4 x5/selection=rsquare best=2;
proc reg data=coleman;
   model y = x1 x2 x3 x4 x5/selection=Cp best=2;
proc reg data=coleman;
   model y = x1 x2 x3 x4 x5/selection=adjrsq;
proc reg data=coleman;
   model y = x1 x2 x3 x4 x5/selection= backward;
proc reg data=coleman;
   model y = x1 x2 x3 x4 x5/selection=forward;
proc reg data=coleman;
   model y = x1 x2 x3 x4 x5/selection=stepwise;
run;
```

10.3 Stepwise model selection

10.3.1 Minitab

You can do this through the `Regression` menu.

Minitab's 'stepwise' command provides stepwise variable selection with a default in which variables are only added if the greatest F statistic is greater than 4 and only removed if the smallest F statistic is less than 4. There are options for forcing variables out of the model, forcing variables into the model, specifying an initial model, and setting the comparison values for the F statistics. Backwards elimination is obtained by entering all the variables into the initial model and resetting the F value for entering a variable to a very large number, say, 100000. Forward selection is obtained by setting the F value for removing a variable to 0. The commands are given below.

```
names c2 'x1' c3 'x2' c4 'x3' c5 'x4' c6 'x5' c8 'y'
note       STEPWISE:  starting with full model
stepwise c8 on c2-c6;
enter c2-c6.
note       STEPWISE:  starting with intercept model
stepwise c8 on c2-c6
note       BACKWARDS ELIMINATION
stepwise c8 on c2-c6;
enter c2-c6;
fenter = 100000.
note       FORWARD SELECTION
stepwise c8 on c2-c6;
fremove = 0.
```

10.3.2 SAS

```
options ps=60 ls=80 nodate;
data coleman;
   infile 'tab6-4.dat';
   input School x1 x2 x3 x4 x5 y ;
proc print;
run;
proc reg data=coleman;
   model y = x1 x2 x3 x4 x5/selection= backward sls=.05;
proc reg data=coleman;
   model y = x1 x2 x3 x4 x5/selection=forward sle=.05;
proc reg data=coleman;
   model y = x1 x2 x3 x4 x5/selection=stepwise  sle=.05 sls=.05;
run;
```

`sls` is the *significance level* needed to *stay* in the model `sle` is the *significance level* needed to *enter* the model.

10.4 Model Selection and Case Deletion

Nothing new here.

10.5 LASSO

10.5.1 Minitab

Minitab has a "predictive modeler" add-on that can be purchased to do LASSO as well as some other penalized (regularized) regressions, splines, CART, random forests,

10.5.2 SAS

Use `Proc GLMSELECT`. "GLM" is a reference to "general linear model" not to "generalized linear model."

```
options ps=60 ls=80 nodate;
data coleman;
   infile 'tab6-4.dat';
   input School x1 x2 x3 x4 x5 y ;
proc print;
run;
proc glmselect data=coleman;
   model y = x1 x2 x3 x4 x5/selection=lasso;

run;
```

This procedure allows other selection methods (but not best subset selection) and some stopping rules, not all of which we discussed.

Chapter 11

Multiple Regression: Matrix Formulation

Introduce reading matrices and simple computations.

11.3 Least squares estimation of regression parameters

11.3.1 Minitab

```
read m1
read m11
let m2=tran(m1)
let m3= prod m2 m1
let m4 = inverse(m3)
let m5= prod m4 m2
let m6=prod m5 m11
print m6
```

11.3.2 SAS

Figuring this out is one of my lowest priorities.

11.5 Residuals, standardized residuals, and leverage

11.6 Principal Component Regression

11.6.1 Minitab

You can do this through the `Regression` menu. This is an up to date version generated by the menus.

```
PCA 'x1' 'x2' 'x3' 'x4' 'x5';
  NComponents 5;
  Coefficients c17;
  Scores c11-c15.
Regress 'y' 5 C11 C12 C13 C14 C15 ;
  Constant;
  Brief 2.
```

This is an older version.

Minitab commands for the principal components regression analysis are given below. The basic command is 'pca.' The 'scores' subcommand places the principal component variables into columns c12 through c16. The 'coef' subcommand places the eigenvectors into columns c22 through c26. If one wishes to define principal components using the covariances rather than the correlations, simply include a pca subcommand with the word 'covariance.'

```
pca c2-c6;
scores c12-c16;
coef c22-c26.
regress c8 on 5 c12-c16 c17 c18
plot c17 c18
```

Chapter 12

One-Way ANOVA

12.1 Example

12.1.1 Minitab

Besides `glm` and `greg`, Minitab has specialized programs for fitting one-way ANOVA:

```
Oneway y ii
```

and another program `AOVOneway` for when the data consist of *a* columns of numbers, one for each group.

12.1.2 SAS

SAS code:

```
options ps=60 ls=80 nodate;
data anova;
   infile 'tab12-1.dat';
   input A R;
   LA = log(A);
proc glm data=anova;
   class R ;
   model LA = R / solution noint;
   means R / lsd alpha=.01 ;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc plot;
   plot ehat*yhat sr*R/  vpos=16 hpos=32;
proc rank data=new normal=blom;
   var sr;
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
proc genmod data=anova;
   class R ;
   model LA = R / noint;
run;
```

12.2 Theory

12.3 Regression analysis of ANOVA data

12.4 Modeling contrasts

12.5 Polynomial regression and one-way ANOVA

12.5.1 Minitab

12.5.2 SAS

```
options ps=60 ls=80;
data asi;
   infile 'tab12-9.dat';
   input x y;
proc print;
run;
proc glm data=Name;
   class x
   model y = x / solution noint;
run;
proc glm data=Name;
   model y = x / solution noint;
run;
proc glm data=Name;
   model y = x x*x x*x*x x*x*x*x x*x*x*x*x / solution noint;
run;
```

12.6 Weighted Regression

12.6.1 Minitab

`Options` menu in `glm` or `greg`.

To get the weighted regression from Minitab, suppose that c1 contains the plate lengths, c2 contains the sample sizes, and c3 contains the means. The commands are as follows:

```
   regress c3 on 1 c1;
   weights c2.
```

A complete set of commands for generating an analysis such as this are given for the next example.

The Minitab commands for this analysis are given below. To obtain pure error and lack of fit from the full data one fits both the simple linear regression and the one-way ANOVA. The ages from Table 7.14 are in c1 and the costs are in c2.

```
  regress c2 on 1 c1
  note     THE AGES IN c1 ARE NOT INTEGERS, SO WE
  note     MULTIPLY THEM BY TWO TO MAKE INTEGER
  note     GROUP LABELS FOR THE ONE-WAY
  let c3=2*c1
  oneway c2 c3
  note     PUT THE MEANS FROM THE ONE-WAY INTO c6, THE
  note     AGES INTO c7, AND THE SAMPLE SIZES INTO c8.
  set c6
172.5 664.333333 633 900.3333333 1202 987 1068.5
end
set c7
```

```
.5 1 4 4.5 5 5.5 6
end
  set c8
2 3 3 3 3 1 2
end
  note     DO THE WEIGHTED REGRESSION
  regress c6 on 1 c7;
  weight c8.
```

12.6.2 SAS

Not done

```
options ps=60 ls=80;
data Name;
   infile 'filename';
   input y x1 x2 x3;
proc print;
run;
proc glm data=Name;
   class x1
   model y = x1 x2 x3/ solution noint;
run;
```

Chapter 13

Multiple Comparisons

13.0.1 Minitab

Check out `Comparisons` menu in `glm`. `GReg` doesn't really do it.

Minitab will do Tukey on unbalanced anova - no idea why the method would be reasonable.

Minitab can be used to obtain the F and t percentage points needed for Bonferroni's method. In this section we have used $t(0.99881, 39)$, $t(0.9997, 39)$, and $F(0.991\bar{6}, 1, 39)$. To obtain these, use Minitab's inverse cumulative distribution function command.

```
invcd \! f .99881;
t 39.
invcd \! f .9997;
t 39.
invcd \! f .9916666;
f 1 39.
```

13.0.2 SAS

Need to examine the `means` line within `proc glm`. I plan to google it when I have time. This is clearly set up to do LSD and other options should be available.

```
options ps=60 ls=80 nodate;
data anova;
   infile 'tab6-1.dat';
   input S LAB ;
   LS = log(S);
proc glm data=anova;
   class LAB;
   model LS = LAB;
   means LAB / lsd alpha=.01 lines;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=r;
proc plot;
   plot ehat*yhat  r*LAB  /  vpos=16 hpos=32;
proc rank data=new normal=blom;
   var  r;
   ranks nscores;
proc plot;
   plot  r*nscores/vpos=16 hpos=32;
run;
```

Chapter 14

Two-Way ANOVA

14.1 Unbalanced two-way ANOVA

Minitab commands

The rat data are contained in Table 14.1. An appropriate data file might look like the first four columns of Table 14.4 and we use C1, C2, C3, and C4 to indicate those columns. Below are Minitab commands for the initial analysis of variance and the ANOVA tables in Table 14.1.

```
names c1 'Index" c2 'Litters' c3 'Mothers' c4 'Weight'
glm c4 = c2|c3;
fits c5;
sresid c6;
cookd c7;
hi c8;
tresid c9.
glm c4 = c3|c2;
```

In the 'glm' command, $c2|c3$ could be replaced by $c2\ c3\ c2*c3$. The $c2$ and $c3$ terms indicate main effects for Litters and Mothers, respectively. Litter by Mother interaction is indicated by $c2*c3$.

Minitab's glm command reports both sequential and adjusted sums of squares for each effect and by default performs tests with the adjusted sums of squares. This should be changed in the "Options" menu, especially when fitting interaction. The adjusted tests for main effects in models that contain interaction are just silly. Adjusted tests are for fitting a term last, and the main effects add nothing to a model that already contains the "interaction" parameters. The reported tests depend on the peccadillos of the programming. We have reported and analyzed the sequential sums of squares.

To delete the outlier in Minitab, just replace the 68.0 for case 12 with an asterisk ($*$) and repeat the commands given above.

14.1.0.1 Adjusted sums of squares

I have often stated that in an interaction model, adjusted sums of squares for main effects are worthless. We have earlier discussed that GReg allows two different sets of side conditions when fitting ANOVA models. Below I have fitted the rat data using both sets of side conditions. Note that sequential sums of squares all agree and that all four of the interaction sum of squares are the same (sequential vs adjusted and two different sets of side conditions).

This is `GReg`'s default set of side condtions (and `glm`'s only side conditions).

```
GReg 'Weight' = Mother| Litter;
  Categorical 'Mother' 'Litter';
  Coding -1;
  TCoef;
  TANOVA.
```

These side conditions give this ANOVA table.

```
Analysis of Variance
Source           DF   Seq SS   Adj SS   Adj MS        F         P
Regression       15  1659.31  1659.31  110.621  2.03945  0.033327
  Mother          3   771.61   671.74  223.913  4.12815  0.011416
  Litter          3    63.63    27.66    9.219  0.16996  0.916118
  Mother*Litter   9   824.07   824.07   91.564  1.68811  0.120053
Error            45  2440.82  2440.82   54.240
Total            60  4100.13
```

The other side conditions are used below

```
GReg 'Weight' = Mother| Litter;
  Categorical 'Mother' 'Litter';
  Coding 1;
  TCoef;
  TANOVA.
```

They give this ANOVA table

```
Analysis of Variance
Source           DF   Seq SS   Adj SS   Adj MS        F         P
Regression       15  1659.31  1659.31  110.621  2.03945  0.033327
  Mother          3   771.61   582.25  194.083  3.57821  0.020993
  Litter          3    63.63   591.69  197.232  3.63625  0.019675
  Mother*Litter   9   824.07   824.07   91.564  1.68811  0.120053
Error            45  2440.82  2440.82   54.240
Total            60  4100.13
```

The adjusted sums of squares for Mother and Litter are completely different! In truth, all four of those numbers should be 0 because main effects never add any explanatory power when fitted after an interaction term that involves them.

14.1.1 SAS

```
options ps=60 ls=80 nodate;
data anova;
   infile 'tab14-1.dat';
   length Litter $1 Mother $1 ;
   input Litter Mother Weight;
proc print;
run;
proc glm data=anova;
   class Litter Mother ;
   model Weight = Litter Mother Litter*Mother / solution;
   means Litter / lsd alpha=.01 ;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc print data=new;
proc plot;
   plot ehat*yhat sr*Litter/  vpos=16 hpos=32;
run;
proc rank data=new normal=blom;
   var sr;
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
```

```
proc glm data=anova;
   class Litter Mother ;
   model Weight = Litter Mother Mother*Litter / solution;
proc glm data=anova;
   class Litter Mother ;
   model Weight = Mother Litter Litter*Mother / solution;
proc glm data=anova;
   class Litter Mother ;
   model Weight =  Litter*Mother Litter Mother / solution;
run;
```

had to modify the data file with blank spaces to get it to read correctly.

14.2 Modeling contrasts

14.2.1 Minitab

14.2.2 SAS

14.3 Regression modeling

14.3.1 Minitab

14.3.2 SAS

14.4 Homologous factors

14.4.1 Minitab

14.4.2 SAS

Not done!!! Need to specify correct models.

```
options ps=60 ls=80 nodate;
data anova;
   infile 'tab14-1.dat';
   length Litter $1 Mother $1 ;
   input Litter Mother Weight;
proc print;
run;
proc glm data=anova;
   class Litter Mother ;
   model Weight = Litter Mother Litter*Mother / solution;
run;
```

Chapter 15

ACOVA and Interactions

15.1 One covariate example

Minitab commands

The following code looks out of date. I do not see `ancova` in the Minitab 16 menus.

The following Minitab commands were used to generate the analysis of these data. The means given by the 'ancova' subcommand 'means' are the adjusted treatment means.

```
names c1 'body' c2 'heart' c3 'sex'
note   Fit model (\thesection.1).
oneway c2 c3
note   Fit model (\thesection.2).
ancova c2 = c3;
covar c1;
resid c10;
fits c11;
means c3.
plot c10 c11
plot c10 c3
note   Split the data into females and males and
note   perform two regressions to fit model (\thesection.5).
copy c1 c2 to c11 c12;
use c3=1.
regress c12 on 1 c11
copy c1 c2 to c21 c22;
use c3=2.
regress c22 on 1 c21
```

15.1.1 SAS

```
options ps=60 ls=80 nodate;
data acova;
   infile 'tab15-1.dat';
   input B H S;
proc glm data=acova;
   class S ;
   model H = B S;
   lsmeans S;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=r;
proc plot;
   plot ehat*yhat  r*S r*B /  vpos=16 hpos=32;
proc rank data=new normal=blom;
```

```
   var  r;
   ranks nscores;
proc plot;
   plot  r*nscores/vpos=16 hpos=32;
proc genmod data=acova;
   class S;
   model H = S  B/ link=id dist=normal ;
proc glm data=acova;
   class S ;
   model H = S S*B;
   lsmeans S ;
proc genmod data=acova;
   class S;
   model H = S  S*B/ link=id dist=normal noint;
run;
```

15.2 Regression modeling

15.2.1 Minitab

15.2.2 SAS

15.3 ACOVA and two-way ANOVA

15.3.1 Minitab

15.3.2 SAS

15.4 Near replicate lack-of-fit tests

15.4.1 Minitab

15.4.2 SAS

15.5

15.5.1 Minitab

15.5.2 SAS

Chapter 16

Multifactor Structures

All of the computational issues can be illustrated from Section 1 alone.

16.1 Unbalanced three-factor analysis of variance

16.1.1 Minitab

Unlike R and SAS, Minitab requires that models be hierarchical. `A B A*B` is equivalent to `A|B` but the model `A*B` is not allowed unless preceded by its individual terms.

$A|XA|X2$ tells Minitab to fit A twice.

```
GLM 'Y' = A|X X2 A*X2;
  Covariates 'X' 'X2';
  Brief 2 .
```

16.1.2 SAS

```
options ps=60 ls=80 nodate;
data anova;
infile 'tab16-13.dat';
input y a b c;
z=b;
proc glm data=anova;
class a b c ;
model y = a*b*c /solution;
run;
proc glm data=anova;
class a b c ;
model y = a b c a*b a*c b*c a*b*c /solution;
run;
proc glm data=anova;
class a b c ;
model y = a*b + c/noint solution;
   means a*b  c/ bon;
run;
prog genmod data=anova;
class a b c ;
model y = c a a*z a*z*z / link=identity dist=normal obstats;
run;
prog  data=anova;
class a b c ;
model y = c a a*z / solution noint;
run;
```

Table 16.1: *Moisture data, indices, and predictors.*

	A	B	C	X	X2	A2		A	B	C	X	X2	A2
y	i	j	k	x	x^2		y	i	j	k	x	x^2	
8	1	1	1	1	1	1	11	1	2	2	2	4	1
17	1	2	1	2	4	1	16	1	3	2	3	9	1
22	1	3	1	3	9	1	3	2	1	2	1	1	2
7	2	1	1	1	1	2	17	2	2	2	2	4	2
26	2	2	1	2	4	2	32	2	3	2	3	9	2
34	2	3	1	3	9	2	5	3	1	2	1	1	2
10	3	1	1	1	1	2	16	3	2	2	2	4	2
24	3	2	1	2	4	2	33	3	3	2	3	9	2
39	3	3	1	3	9	2	4	1	1	2	1	1	1
13	1	2	1	2	4	1	10	1	2	2	2	4	1
20	1	3	1	3	9	1	15	1	3	2	3	9	1
10	2	1	1	1	1	2	5	2	1	2	1	1	2
24	2	2	1	2	4	2	19	2	2	2	2	4	2
9	3	1	1	1	1	2	29	2	3	2	3	9	2
36	3	3	1	3	9	2	4	3	1	2	1	1	2
5	1	1	2	1	1	1	34	3	3	2	3	9	2

16.2

Because it is the easiest program I know, most of the analyses in this book were done in Minitab. We now present and contrast R and SAS code for fitting $[AB][C]$ and discuss the fitting of other models from this section. Table 16.7 illustrates the variables needed for a full analysis. The online data file contains only the y values and indices for the three groups. Creating X and X2 is generally easy. Creating the variable A2 that does not distinguish between salts 2 and 3 can be trickier. If we had a huge number of observations, we would want to write a program to modify A into A2. With the data we have, in Minitab it is easy to make a copy of A and modify it appropriately in the spreadsheet. Similarly, it is easy to create A2 in R using `A2=A` followed by `A2[(A2 == 3)] = 2`. For SAS, I would probably modify the data file so that I could read A2 with the rest of the data.

SAS code for fitting $[AB][C]$ follows. The code assumes that the data file is the same directory (folder) as the SAS file.

```
options ps=60 ls=80 nodate;
data anova;
   infile 'tab16-1.dat';
   input y A B C;
   X = B;
   X2=X*X;
proc glm data=anova;
   class A B C ;
   model y = A*B C ;
   means C / lsd alpha=.01 ;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc plot;
   plot ehat*yhat sr*R/  vpos=16 hpos=32;
proc rank data=new normal=blom;
   var sr;
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
```

16.3 Comparison of model definitions

Consider the following models

$$\begin{aligned}[ABC] &\cong y_{ijkm} = G + A_i + B_j + C_k + [AB]_{ij} + [AC]_{ik} + [BC]_{jk} + [ABC]_{ijk} + e_{ijkm} \\ &\cong y_{ijkm} = [ABC]_{ijk} + e_{ijkm}.\end{aligned}$$

$$\begin{aligned}[AB][BC] &\cong y_{ijkm} = G + A_i + B_j + C_k + [AB]_{ij} + [BC]_{jk} + e_{ijkm} \\ &\cong y_{ijkm} = [AB]_{ij} + [BC]_{jk} + e_{ijkm}.\end{aligned}$$

$$\begin{aligned}[AB][C] &\cong y_{ijkm} = G + A_i + B_j + C_k + [AB]_{ij} + e_{ijkm} \\ &\cong y_{ijkm} = [AB]_{ij} + C_k + e_{ijkm}.\end{aligned}$$

$$\begin{aligned}[AB][C] &\cong y_{ijkm} = G + A_i + B_j + C_k + [AB]_{ij} + e_{ijkm} \\ &\cong y_{ijkm} = [AB]_{ij} + C_k + e_{ijkm}.\end{aligned}$$

$$\begin{aligned}[A_0][A_1][A_2][C] &\cong y_{ijkm} = G + A_{i0} + \gamma_1 x_j + \gamma_2 x_j^2 + A_{i1} x_j + A_{i2} x_j^2 + C_k + e_{ijkm}. \\ &\cong y_{ijkm} = A_{i0} + A_{i1} x_j + A_{i2} x_j^2 + C_k + e_{ijkm}.\end{aligned}$$

All three programs can fit the first form of the model. Minitab ONLY fits the first form. The R and SAS commands given below are for fitting the second form of the model.

Model	Minitab	R	SAS
$[ABC]$	$A\|B\|C$	A:B:C-1	A*B*C / noint
$[AB][BC]$	$A\|B \quad B\|C$	A:B+B:C-1	A*B B*C / noint
$[AB][C]$	$A\|B \quad C$	A:B+C-1	A*B C / noint
$[A_0][A_1][A_2][C]$	$A\|X \quad A\|X2 \quad C$	A+A:X+A:X2+C-1	A A*X A*X2 C / noint
$[A_0][A_1][C], A_{21} = A_{31}$	$A \quad A2\|X \quad C$	A+A2:X+C-1	A A2*X C / noint
$[A_0][A_1][C], A_{21} = A_{31}, A_{20} = A_{30}$	$A2 \quad A2\|X \quad C$	A2+A2:X+C-1	A2 A2*X C noint

To fit different models, one needs to modify the part of the code that specifies the model. In Minitab's `glm`, models are usually specified in the `model` dialog box (or on the command line) and X and X2 have to be specified as covariates. In R, specifying models involves changes to, say, `lm(y ~ A:B+C-1)` where A, B, and C all have to be prespecified as `factor` variables. In SAS's glm, modeling involves changes to `model y = A*B C/noint;` where A, B, and C all have to be prespecified as `class` variables. □

16.4 Balanced three factors

16.4.1 Minitab

16.4.2 SAS

```
options ps=60 ls=80 nodate
data abraid;
   infile 'tab16-8.dat';
   input y,S,F,P,rep;
```

```
   pp=P;
   p2=pp*pp;
proc print;
run;
proc glm data=abraid;
   class S F P;
   model y = S*F*P;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=sr;
proc glm data=abraid;
   class S F P;
   model y = S*F F*P   / noconstant;
proc glm data=abraid;
   class S F P;
   model y = S*F F*pp F*p2 / noconstant;
proc glm data=abraid;
   class S F P;
   model y = S*F F*pp  / noconstant;
proc print data=new;
proc plot;
   plot sr*yhat /  vpos=16 hpos=32;
proc rank data=new normal=blom;
   var sr;
   ranks nscores;
proc plot;
   plot sr*nscores/vpos=16 hpos=32;
run;
```

To finish the analysis we need to figure out how to define two additional variables in SAS,

$$p0 = (1,2,3,1,2,3,0,0,0,0,0,0,1,2,3,1,2,3,0,0,0,0,0,0)'$$

and

$$SF = c(11,11,11,11,11,11,12,12,12,12,12,12,21,21,21,21,21,21,21,21,21,21,21,21)'.$$

We can then add the following commands to the previous ones.

```
proc glm data=abraid;
   class S F P;
   model y = S*F  p0  / noconstant;
proc glm data=abraid;
   class S F P SF;
   model y = SF  p0   / noconstant;
run;
```

16.5 Higher order structures

Nothing to do.

Chapter 17

Basic Experimental Design

17.4 Randomized complete block designs

17.4.1 Minitab

The following Minitab commands generate the analysis of variance. Column c1 contains the spectrometer data, while column c2 contains integers 1 through 4 indicating the appropriate treatment, and c3 contains integers 1 through 3 that indicate the block. The predicted values are given by the 'fits' subcommand.

```
names c1 'y' c2 'Trts' c3 'Blks'
glm c1 = c3 c2;
  Pairwise Treatment;
    Bonferroni;
sresid c10;
fits  c11.
```

Get out all pairwise comparisons

17.4.2 SAS

```
options ps=60 ls=80 nodate;
data rcb;
   infile 'tab17-1.dat';
   input B T Y ;
proc glm data=rcb;
   class B T;
   model Y = B T;
   means B T / lsd alpha=.01 lines;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=r;
proc plot;
   plot ehat*yhat  r*B r*T /  vpos=16 hpos=32;
proc rank data=new normal=blom;
   var  r;
   ranks nscores;
proc plot;
   plot  r*nscores/vpos=16 hpos=32;
run;
```

17.5 Latin squares

17.5.1 Minitab

The following Minitab commands will give the sums of squares, means, and residuals necessary for the analysis. Here c1 is a column containing the mangold root yields, c2 has values from 1 to 5 indicating the row, c3 has values from 1 to 5 indicating the column, and c4 has values from 1 to 5 indicating the treatment.

```
   names c1 'y' c2 'Rows' c3 'Cols' c4 'Trts'
   glm c1 = c2 c3 c4;
```

17.5.2 SAS

```
options ps=60 ls=80 nodate;
data lsq;
   infile 'tab17-6.dat';
   input Y Row Col T ;
proc glm data=lsq;
   class Row Col T;
   model Y = Row Col T;
   means Row Col T / Tukey alpha=.05 lines;
   output out=new r=ehat p=yhat cookd=c h=hi rstudent=tresid student=r;
proc plot;
   plot ehat*yhat  r*Row r*Col r*T /  vpos=16 hpos=32;
proc rank data=new normal=blom;
   var  r;
   ranks nscores;
proc plot;
   plot  r*nscores/vpos=16 hpos=32;
run;
```

17.6 Balanced incomplete blocks

Detergents

17.6.1 SAS

```
options ps=60 ls=80 nodate;
data bib;
infile 'TAB17-7A.DAT';
input row blk trt y det amt ctl;
   a=amt;
   a2=a*a;
   a3=a2*a;
proc glm data=bib;
class det amt ctl blk trt;
model y = blk trt;
proc glm data=bib;
class det amt ctl blk trt;
model y = blk ctl det amt det*amt;
proc glm data=bib;
class det amt ctl blk trt;
```

```
model y = blk ctl det det*a det*a2 /noint solution;
proc genmod data=bib;
class det amt ctl blk trt;
model y = blk ctl det det*a det*a2 /noint link=identity dist=normal;
run;
```

17.7 Youden squares

17.7.1 Minitab

The Minitab commands for the mangold root analysis are given below.

```
names c1 'y' c2 'Rows' c3 'Cols' c4 'Trts'
glm c1 = c2 c3 c4;
means c4;
fits c11;
sresids c12;
tresids c13;
hi c14;
cookd c15.
```

17.7.2 SAS

Not done

```
options ps=60 ls=80;
data Name;
   infile 'filename';
   input y x1 x2 x3;
proc print;
run;
proc glm data=Name;
   class x1
   model y = x1 x2 x3/ solution noint;
run;
```

Chapter 18

Factorial Treatments

18.1 RCB Analysis

18.1.1 Minitab

The following Minitab commands generate this analysis of variance. Here c1 is a column containing the spectrometer data, c2 has values from 1 to 4 indicating the treatment, c3 has values from 1 to 3 indicating the block, c4 has values 1 and 2 indicating the disk type, and c5 has values 1 and 2 indicating the window type.

```
names c1 'y' c2 'Trts' c3 'Blks' c4 'Disk' c5 'Window'
glm c1 = c3 c4|c5;
```

In the 'glm' command, $c4|c5$ could be replaced by $c4\ c5\ c4*c5$. The $c4$ and $c5$ terms indicate main effects for disks and windows, respectively. Disk by window interaction is indicated by $c4*c5$.

18.1.2 SAS

Not done

```
options ps=60 ls=80;
data Name;
   infile 'filename';
   input y x1 x2 x3;
proc print;
run;
proc glm data=Name;
   class x1
   model y = x1 x2 x3/ solution noint;
run;
```

18.4 Interaction in a Latin square

18.4.1 Minitab

18.4.2 SAS

$$Trts = (4,1,0,5,3,2,1,2,4,3,0,5,5,3,2,4,1,0,0,4,1,2,5,3,2,5,3,0,4,1,3,0,5,1,2,4)'$$

```
options ps=60 ls=80;
data potato;
   infile 'tab18-2.dat';
   input R C N P y;
   Trts=10*N+P;
   pp=P;
   p2=pp*pp;
```

```
proc print;
run;
proc glm data=potato;
   class R C N P Trts
   model y = R C Trts/ solution noint;
run;
```

18.5 A balanced incomplete block design

18.5.1 SAS

```
options ps=60 ls=80 nodate;
data bib;
infile 'tab18-5A.DAT';
input row blk trt y det amt ctl;
   a=amt;
   a2=a*a;
   a3=a2*a;
proc glm data=bib;
class det amt ctl blk trt;
model y = blk trt;
proc glm data=bib;
class det amt ctl blk trt;
model y = blk ctl det amt det*amt;
proc glm data=bib;
class det amt ctl blk trt;
model y = blk ctl det det*a det*a2 /noint solution;
proc genmod data=bib;
class det amt ctl blk trt;
model y = blk ctl det det*a det*a2 /noint link=identity dist=normal;
run;
```

Chapter 19

Dependent Data

At the moment, I would **not completely trust** anything in this chapter except the Minitab projects. Most of it should work or be close to working. Certainly they are not complete analyses.

19.1 The analysis of split plot designs

In longitudinal data analysis these models are known as random intercept models because every individual has a separate random effect associated with it.

19.1.1 Minitab

Because of the missing whole plot in this version of Garner's data, Minitab 18 refuses to fit the appropriate model. Minitab 16 only gives the degrees of freedom and the sums of squares for the effects but that is the hard work. You can easily construct the ANOVA table from those. Error 1 is the Rep by Laundry interaction and it has 2 rather than the usual 3 degrees of freedom because of the missing whole plot.

```
names c1 'Rep' c2 'L' c3 'T' c4 'y';''
glm c4=c1|c2 c3 c2*c3
```

See also the Minitab project `SEC19-1UNBAL.MPJ` although since this was constructed in 16 I don't think 18 reads it.

Equivalently we can use the constructed interaction column "inter" to fit

```
glm c4=c1|c2 c5
```

The interaction column just treats each of the 16 combinations of a Laundry and a Test as a separate group. Fitting the model with C5 in Minitab gives reduced output but yields the appropriate *dfE* and *SSE*.

To get the *dfE* and *SSE* for the model that sets $A = D$ in Laundry 1 fit

```
glm c4=c1|c2 c6
```

To fit the other models (for Laundry 1 or the interaction) just replace C6 with a different column.

19.1.2 SAS

```
data garn;
  infile 'tab19-1.dat";
  var Rep L T y;
proc glm;
   class Rep L T;
   model y =  Rep  L Rep*L  T  T*L/solution;
```

No idea if the following model is going to work.

```
data garn;
  infile 'tab19-1.dat";
  var Rep L T y;
proc glm;
   class Rep L T;
   model y =  (Rep  L)@2  T  T*L;
```

This SAS program was in my files. **Not sure about it.**

```
proc mixed data=DUMMYALL;
  class Time A B WholePlot Subplot;
  model y = Time A B A*B Time*A Time*B Time*A*B / ddfm=kr
  random WholePlot / group=Time;
  repeated Time / subject=Subplot type=ARH(1);
  lsmeans Time*A / slice=Time;
  lsmeans Time*B / slice=Time;
  lsmeans Time*A*B / slice=Time;
run;
```

19.1.3 Whole Plot Analysis

19.1.3.1 SAS

```
data garnw;
   infile 'tab19-1w.dat";
  var Rep L y;
proc print;
run;
proc glm;
   class Rep L;
 model y =  Rep  L/solution;
run;
```

19.2 A four factor example

Use the data in file TAB19-6.DAT.

19.2.1 Minitab

Using the clunky Minitab 18 method for reading my data files,

```
  read c1-c6;
  file "c:\e-drive\books\anreg2\newdata\tab19-6.dat".
  names c1 'y' c2 'surf' c3 'fill' c4 'prop' c5 'rep' c6 'rota'
  GLM c1 = c2|c3|c4 c5(c2 c3 c4) c6 c6*c2 c6*c3 c6*c4 c6*c2*c3 c6*c2 &
CONT>     *c4 c6*c3*c4 c6*c2*c3*c4;
    Random c5.
```

The term `c5(c2 c3 c4)` tells Minitab to fit `rep` `nested within` `surf`, `fill`, and `prop`. The whole plots are a completely randomized design, so there is no blocking in the whole plots and the nested term is needed to get Error 1 computed (but not called Error 1). The subcommand to treat `reps` as random gets Minitab to construct the whole plot F tests correctly.

19.2.2 SAS

This should give the proper ANOVA table. Treat the S*F*P*RP as Error 1 and construct whole plot the F tests yourself

```
data abraid;
  infile 'tab19-5.dat';
  var y S F P RP  RoT ;
pp = P;
p2=pp*pp;
proc glm;
    class S F P RoT RP;
    model y =  S F P S*F S*P F*P S*F*P S*F*P*RP  RoT RoT*S RoT*F RoT*P RoT*S*F RoT*S*P RoT*F*P
```

This might also work.

```
data abraid;
  infile 'tab19-5.dat';
  var y S F P RP  RT ;
pp = P;
p2=pp*pp;
proc glm;
    class S F P RT RP;
    model y =  (S F P)@3  S*F*P*RP  RT*(S F P)@3;
```

19.2.3 Whole Plot Analysis with Error 1 residual plots

```
data abraid;
    infile 'C:\\E-drive\\Books\\ANREG2\\newdata\\tab19-5a.dat';
    var yy1 yy2 yy3 S F P rep;
    yw=(yy1+yy2+yy3)/3;
    pp = P;
    p2=pp*pp;
proc glm;
    class S F P RT RP;
model yw = S F P S*F S*P F*P S*F*P;
proc glm;
  class  S F P RT RP;
model yw = (S F P)@3;
```

19.2.4 Final Models

This has a lot of funky logical stuff (from R) that needs to be redone for SAS.

```
data  abraid   ;
  infile 'tab19-5.dat";
  var y s f p rep rot;

mtilde = 2*rot/rot
mtilde = mtilde - (rot<2)
```

```
mtildea=mtilde[f==1]
mtildeb=mtilde[f==2]

ya = y[f==1]
sa = s[f==1]
pa = p[f==1]
repa = rep[f==1]
rota = rot[f==1]

Mtildea=factor(mtildea);
Sa=factor(sa);
Pa=factor(pa);
RTa=factor(rota);
RPa=factor(repa);
p2a=pa*pa;
r2a=rota*rota;
proc glm;
   class  Sa F P RTa Mtildea RP;
   model ya = Sa  pa  RTa  Sa*Mtildea;



yb = y[f==2]
sb = s[f==2]
pb = p[f==2]
repb = rep[f==2]
rotb = rot[f==2]

Mtildeb=factor(mtildeb)
Sb=factor(sb)
Pb=factor(pb)
RTb=factor(rotb)
RPb=factor(repb)
p2b=pa*pb
r2b=rotb*rotb

    proc glm;  class  S F P RT RP;
bsfpb  model yb = Sb  RTb  Sb*Mtildeb)
anova(bsfpb)
coefb=summary(bsfpb)
coefb
bsfpb$fit
```

19.2.5 Unbalanced SubPlots

The key point in this subsection is that every model needs to have `S*F*P*RP` in it!

```
    proc glm;
      class  S F P RT RP;
model yw =  S*F*P*RP  RT  RT*(S F P)@3;
```

19.3 Multivariate analysis of variance

Use the data in file TAB19-6A.DAT.

19.3.1 Minitab

Minitab commands for obtaining Tables 19.14 through 19.18 and the corresponding residuals and predicted values are given below. The three columns c1, c2, and c3 contain y_1, y_2, and y_3, respectively. Columns c4, c5, and c6 contain indices for S, F, and P. Some older versions of Minitab do not allow the last three subcommands that provide the multivariate analysis of variance. Incidentally, Minitab uses T^2/dfE as its definition for the Lawley–Hotelling trace.

```
glm c1-c3 = c4|c5|c6;
means c4 c5 c6 c4*c5 c4*c6 c5*c6 c4*c5*c6;
resid c11 c12 c13;
fits c21 c22 c23;
manova c4 c5 c6 c4*c5 c4*c6 c5*c6 c4*c5*c6;
sscp;
eigen.
```

When using the menus choose `Stat`, then `AVOVA`, then `MANOVA` (even though it is just using `glm` with a `manova` subcommand). The subcommands can be generated by various menu options.

19.3.2 SAS

Not really sure how to do MANOVA in SAS. Should be a minor generalization of `proc glm`.

```
abraid = data    ;  infile 'tab19-5a.dat",
  var yy1 yy2 yy3 S F P RP;
pp = P;
p2=pp*pp;
proc glm;
  class  S F P ;
  model yy1 yy2 yy3 = S F P S*F S*P F*P S*F*P/ noconstant ;
manova h= S F P S*F S*P F*P S*F*P/ printh printe summary;

proc glm;
      class  S F P RT RP;
model  yy1 yy2 yy3 =  (S F P)@3;
 manova;
proc glm;
      class  S F P RT RP;
 model  yy1 yy2 yy3 = S*F  F*P   / noconstant ;
 manova h= S*F  F*P ;
proc glm;
      class  S F P RT RP;
 model  yy1 yy2 yy3 = S*F  F*p  F*p2 / noconstant ;
  manova;
proc glm;
      class  S F P RT RP;
 model  yy1 yy2 yy3 = S*F + F*p  / noconstant ;
 manova;
```

We need to define

$$p0 = (1,2,3,1,2,3,0,0,0,0,0,0,1,2,3,1,2,3,0,0,0,0,0,0)$$

$$SF = (11,11,11,11,11,11,12,12,12,12,12,12,21,21,21,21,21,21,21,21,21,21,21,21)$$

Now also run

```
    proc glm;
      class  S F P RT RP;
model yy1  yy2 yy3 = S*F  p0  / noconstant ;
 manova;
    proc glm;
    class  S F P RT RP SF;
model yy1 yy2 yy3  = SF  p0   / noconstant ;
 manova;
```

19.4 Random effects models

19.4.1 Minitab

19.4.2 SAS

Chapter 20

Logistic Regression

20.1 Models for binomial data

No real computing.

20.2 Simple linear logistic regression

20.2.1 Minitab

```
names c1 'dose' c2 'died' c3 'trials' c4 'y'
let c4=c2/c3
let c11=loge(c1)
Name c12 "PRES2" c13 "SPRE2" c14 "HI2" c15 "EPRO2"
Blogistic 'died' 'trials' = C11;
  ST;
  Logit;
  Presiduals 'PRES2';
  Spresiduals 'SPRE2';
  Hi 'HI2';
  Eprobability 'EPRO2';
  Brief 2.
```

20.2.2 SAS

SAS contains a powerful program for logistic regression, PROC LOGISTIC, but it pools cases for goodness-of-fit tests and diagnostics. The first line below controls printing. The next four lines involve defining and reading the data. The remaining lines specify the model and that a logistic regression is to be performed.

```
options ps=60 ls=80 nodate;
data mice;
   infile 'tab21-1.dat';
   input x Ny N y;
proc logistic data=mice descending;
   model y=x ;
run;
```

Alternatively, to get usable deviance values from SAS, we can use the generalized linear model program PROC GENMOD. To perform logistic regression one must specify an appropriate link and distribution. One also specifies the number of deaths for each case (Ny) divided by the number of trials for each case (N).

```
options ps=60 ls=80 nodate;
data mice;
```

```
   infile 'tab21-1.dat';
   input x Ny N y;
proc genmod data=mice ;
   model Ny/N = x / link=logit dist=binomial;
run;
```

20.3 Model testing

20.3.1 Minitab

20.3.2 SAS

20.4 Fitting logistic models

20.4.1 Minitab

20.4.2 SAS

20.5 Binary data

20.5.1 Minitab

The data are in a file TAB21-3.DAT that contains six columns similar to Table 20.3. ID indicates the case, flt is the flight, f indicates failure of any O-rings, and temp is temperature. The fourth column s is one minus f. The last column in the file contains the actual number of O-rings that failed on a flight.

```
  names c1 'Case' c2 'Flt' c3 'F' c4 'S' c4 'Temp' c6 'no'
  Blog 'F' = Temp;
    Logit;
    Brief 2.
```

20.5.2 SAS

The data are in a file TAB21-3.DAT that contains six columns similar to Table 20.3. ID indicates the case, flt is the flight, f indicates failure of any O-rings, and temp is temperature. The fourth column s is one minus f. The last column in the file contains the actual number of O-rings that failed on a flight.

The following commands are for the generalized linear model procedure in SAS, GENMOD.

```
options ps=60 ls=80 nodate;
data oring;
   infile 'TAB21-3.dat';
   input ID flt f s temp no;
   n = 1;
proc genmod data = oring;
   model f/n = temp  / link=logit  dist=binomial;
run;
```

The first line controls printing of the output. The next four lines define the data. The variable "n" is used to specify that there is only one trial in each of the 23 binomials. PROC GENMOD needs the data specified: "data = oring". GENMOD also needs information on the model. "link = logit" and "dist = binomial" both are needed to specify that a logistic regression is being fitted. "model f/n = temp" indicates that we are modeling the number of failures in "f" out of "n" trials using the predictor "temp" (and implicitly an intercept).

As displayed in Subsection 20.2.4, the SAS program for logistic regression, PROC LOGISTIC, provides more specialized output.

```
options ps=60 ls=80 nodate;
data oring;
   infile 'tab21-3.dat';
   input ID flt f s temp no;
   n = 1;
proc genmod data = oring;
   model f/n = temp  / link=logit  dist=binomial;
run;
```

20.6 Multiple logistic regression

The data are in `chapman.dat`. They do not appear as a table in the book.

20.6.1 Minitab

Multiple logistic regression not very different from the simple. Best to do variable selection by appropriately modifying `breg` or `step`. Will show an example when I get around to it. Also see book.

The data are in a file 'chapman.dat' with eight columns: the case index, *Ag*, *S*, *D*, *Ch*, *H*, *W*, and *CN*. The file looks like this.

```
  1 44 124  80 254 70 190 0
  2 35 110  70 240 73 216 0
  3 41 114  80 279 68 178 0
  4 31 100  80 284 68 149 0
       data continue
199 50 128  92 264 70 176 0
200 31 105  68 193 67 141 0
```

20.6.2 SAS

Below are SAS commands for obtaining a multiple logistic regression. The data are in a file 'chapman.dat' with eight columns: the case index, *Ag*, *S*, *D*, *Ch*, *H*, *W*, and *CN*. The file looks like this.

```
  1 44 124  80 254 70 190 0
  2 35 110  70 240 73 216 0
  3 41 114  80 279 68 178 0
  4 31 100  80 284 68 149 0
       data continue
199 50 128  92 264 70 176 0
200 31 105  68 193 67 141 0
```

A simple way to fit the logistic regression model (20.6.4) in SAS is to use PROC GENMOD. The first line controls printing. The next four lines involve defining and reading the data and creating a variable "n" that gives the total number of possible successes for each case. The remaining lines specify the model and that a logistic regression is to be performed.

```
options ps=60 ls=80 nodate;
data chapman;
   infile 'chapman.dat';
   input ID Ag S D Ch H W CN;
   n = 1;
   y = CN;
proc genmod;
   model y/n = Ag S D Ch H W/ dist = binomial;
```

```
proc genmod data=chapman ;
   model CN/n = Ag Ch W / link=logit dist=binomial;
run;
```

A more powerful program for logistic regression (but one that pools cases for goodness-of-fit tests and diagnostics) is PROC LOGISTIC.

```
options ps=60 ls=80 nodate;
data chapman;
   infile 'chapman.dat';
   input ID Ag S D Ch H W CN;
proc logistic data=chapman descending;
   model CN=Ag Ch W ;
run;
```

On the line with "proc logistic", one specifies the data being used and the command "descending". The command "descending" is used so that the program models the probabilities of events coded as 1 rather than events coded as 0. In other words, it makes the program model the probability of a coronary incident rather than the probability of no coronary incident.

`proc logistic` can do backward elimination, forward selection, and an inferior form of best subset selection. It is better to do best subset selection using `prog reg` with input from a previous `proc logistic` run. Sometime I will give an example. Also see book.

20.7 ANOVA type logit models

The muscle tension data are listed in the file 'tab21-10.dat' with one column for the number of high tension scores, one column for the low tension scores, and three columns of indices that specify the level of weight (high is 1), muscle type, and drug, respectively.

```
 3  3 1 1 1
21 10 1 1 2
23 41 1 2 1
11 21 1 2 2
22 45 2 1 1
32 23 2 1 2
 4  6 2 2 1
12 22 2 2 2
```

20.7.1 Minitab

`Blog` accepts classifiers.

20.7.2 SAS

The following commands fit the model [WM][WD][MD] using SAS PROC GENMOD. Note that the variable "n" is the total number of individuals with each level of weight, muscle type, and drug. The "class" command is used to distinguish ANOVA type factors from regression predictors.

```
options ps=60 ls=80 nodate;
data tension;
   infile 'TAB21-9.DAT';
   input H L W M D;
   n = H+L;
proc genmod data=tension;
   class W M D;
   model H/n = W*M W*D M*D / link=logit dist=binomial;
```

```
run;
```

20.8 Ordered categories

20.8.1 Minitab

To fit the data in Table 20.15 they need to be manipulated. In Minitab, the data are read into columns c1 through c5. The following commands allow for fitting model (20.8.1). Here "EPRO2" gives the fitted probabilities from which the odds can be computed and placed into a separate column.

```
  names c1 "h" c2 "i" c3 "k" c4 "j" c5 "count"
sort the data so that all the yes counts are together
and all the no counts are together
  sort c4 c1 c2 c3 c5 c11 c12 c13 c14 c15
Now c11=j, c12=h, c13=i, c14=k, and c15=count
separate the yes data from the no data into a new worksheet
  Unstack (C12 C13 C14 C15);
    Subscripts C11;
    NewWS;
    VarNames.
Now c4 has the yes counts and c8 has the no counts with
c1=c5=h, c2=c6=i, c3=c7=k
  let c10=c4+c8
  Name c13 "EPRO2"
  Blogistic C4 C10 = c1|c2  c3;
    ST;
    Factors c1 c2 c3;
    Logit;
    Eprobability 'EPRO2';
    Brief 2.
  Name c14 "Odds"
  let c14 = c13/(1-c13)
```

20.8.2 SAS

```
data    abop;
   infile 'tab21-14a.dat";
   var Case R S A Yes No Total;
class R S A;
y=Yes/Total
proc genmod;
   model y = R*S + A/ dist = binomial,weights=Total;
```

Chapter 21

Log-Linear Models

Minitab has programs for fitting logistic regression but none for generalized linear models.

21.1 Models for two-factor tables

21.1.1 Minitab

21.1.2 SAS

The following SAS code fits models (21.1.2), (21.1.4), and (21.1.5) and gives fitted values and Pearson residuals. By comparing the fitted values and Pearson residuals for models (21.1.2) and (21.1.4) with each other and results from Section 5.5 you can see that both models fit the independence model. Similarly, the fitted values and Pearson residuals confirm that fitting model (21.1.5) fits the Roman Catholics and Protestants like Table 5.11 while leaving the Jewish entries untouched.

```
data occ;
   infile 'tab21-2.dat';
   var y R O RCP Q;
proc genmod;
    class R O RCP Q;
  model y = R O/dist = poisson;
proc genmod;
    class R O RCP Q;
   model y = O RCP*Q/dist = poisson;
proc genmod;
    class R O RCP Q;
   model y = O*RCP RCP*Q / dist = poisson;
```

21.2 Models for three-factor tables

21.2.1 Minitab

21.2.2 SAS

21.3 Estimation and odds ratios

21.3.1 Minitab

21.3.2 SAS

21.4 Higher dimensional tables

21.4.1 Minitab

21.4.2 SAS

The muscle tension data are listed in file `tab21-9a.dat` as counts for each cell with indices for tension, weight, muscle type, and drug, respectively. The file is as given below.

```
 3 1 1 1 1
21 1 1 1 2
23 1 1 2 1
11 1 1 2 2
22 1 2 1 1
32 1 2 1 2
 4 1 2 2 1
12 1 2 2 2
 3 2 1 1 1
10 2 1 1 2
41 2 1 2 1
21 2 1 2 2
45 2 2 1 1
23 2 2 1 2
 6 2 2 2 1
22 2 2 2 2
```

This file is different from that used in Chapter 21, Tab21-9.dat. We can fit the log-linear model [WMD][TWM][TWD][TMD] using SAS PROC GENMOD.

```
options ps=60 ls=80 nodate;
data tension;
   infile 'tab21-9a.dat';
   input y T W M D;
proc genmod data=tension;
   class T W M D;
   model y = W*M*T T*W*M T*W*D T*M*D / link=log
                                       dist=poisson;
run;
```

The "class" command specifies that a variable is not acting like a predictor variable in regression, but rather that it gives indices for specifying the levels of an analysis of variance type factor. To fit other specific models such as [TM][WM][MD] or [T][WM][D], the model statement uses T∗M W∗M M∗D or T W∗M D, respectively.

For the abortion data, the data file 'abort.dat' has five columns, the first four being indices for race, sex, age, and opinion, and the last being the cell counts. To fit [RSO][RSA][ROA][SOA] with GENMOD use

```
options ps=60 ls=80 nodate;
data abort;
   infile 'tab21-4.dat';
   input R S A O y;
proc genmod data=abort;
   class R S A O;
   model y = R*S*O R*S*A R*O*A S*O*A / link=log
                                       dist=poisson;
run;
```

The BMDP-4F commands for fitting the model [RSO][RSA][ROA][SOA] are

```
/ INPUT     FILE = 'C:\tab21-4.dat'.
         FORMAT = FREE.
         VARIABLES = 5.
/ VARIABLE  NAMES = R, S, A, O, N.
/ TABLE     INDICES = R, S, A, O.
            COUNT = N.
/ STAT      ALL.
/ FIT       MODEL = RSO, RSA, ROA, SOA.
/ PRINT     LINE = 79.
/ END
```

21.5 Ordered categories

21.5.1 Minitab

21.5.2 SAS

Models with quantitative factors can be fit easily using several computer packages, e.g., R and SAS PROC GENMOD. For example the model $\log(\mu_{hijk}) = u_{RSO(hij)} + u_{A(k)} + O_{1j}k + O_{2j}k^2$ can be fitted using SAS PROC GENMOD as given below.

```
options ps=60 ls=80 nodate;
data abort;
   infile 'tab21-4.dat';
   input R S A O y;
   A1 = A;
   A2 = A * A;
proc genmod;
   class R O S A;
   model y = R*S*O O*A / dist = poisson;
proc genmod data=abort;
   class R S O A;
   model y = R*S*O A O*A1 O*A2 / link=log dist=poisson;
run;
proc genmod;
   class R O S A;
   model y = R*S*O  A  O*A1 / dist = poisson;
run;
```

The terms $O_{1j}k$ and $O_{2j}k^2$ are really interactions between the factor O and the predictor variables age and age squared. In the model statement, they are simply specified as interactions.

21.6 Offsets

21.6.1 Minitab

Minitab doesn't seem to do offsets.

21.6.2 SAS

SAS GENMOD commands for fitting models (21.6.3) and (21.6.2) are, respectively,

```
options ps=60 ls=80 nodate;
data bis;
   infile 'tab22-5.dat';
   input index l y;
   ll = log(l);
    J=y+1-y;
proc genmod data=bis;
   model y = ll / link=log dist=poisson;
run;
proc genmod data=bis;
   model y = J/ link=log dist=poisson  noconstant offset=ll;
run;
```

21.7 Relation to logistic models

21.7.1 Minitab

21.7.2 SAS

21.8 Multinomial responses

21.8.1 Minitab

Ordinal logistic regression menu. `OLog`
Nominal logistic regression menu. `NLog`

21.8.2 SAS

21.9 Logistic discrimination and allocation

21.9.1 Minitab

21.9.2 SAS

Chapter 22

Exponential and Gamma Regression: Time to Event Data

Minitab doesn't really do these.

Might do exponential as regression with life data. Mostly does accelerated failure time and incorporates censoring.

22.1 Exponential regression

22.1.1 Minitab

22.1.2 SAS

The data file contains *h*, the case number; *y*, the survival time in weeks; *w*, the white blood cell count; *lw*, the log to base 10 of the white blood cell count; and *a*, the AG score. SAS GENMOD commands that deliver all the necessary deviances except the one for model (3) are

```
To fit the model involving the offset, use the GENMOD commands
\begin{verbatim}
options ps=60 ls=80 nodate;
data feigl;
   infile 'tab22-1.dat';
   input  y w  a;
   lw = log10(w)
   nlw = -lw;
run;
proc genmod data=feigl;
   class a;
   model y = a a*lw/ link=log
                     dist=gamma
                     noscale noint;
run;
proc genmod data=feigl;
   class a;
   model y = a lw / link=log
                    dist=gamma type1 noscale;
run;
proc genmod data=feigl;
   class a;
   model y = nag a*lw / link=log dist=gamma noscale noint;
run;
```

In the model statement, the link specifies that the log of the expected value is given a linear structure.

The pair "dist=gamma' and "noscale" together specify the exponential distribution for the data. The command "noint" indicates that the linear model is fitted without an intercept term. The command "type1" in the second call of proc genmod provides deviances for models (1) and (2) as well as model (4).

22.2 Gamma regression

22.2.1 Minitab

22.2.2 SAS

The SAS GENMOD commands are exactly as in Subsection 22.1.1 except that one no longer includes the "noscale" command.

To fit the model involving the offset, use the GENMOD commands

```
options ps=60 ls=80 nodate;
data feigl;
   infile 'tab22-1.dat';
   input  y w  a;
   lw = log10(w)
   nlw = -lw;
run;
proc genmod data=feigl;
   class a;
   model y = a / link=log dist=gamma
                 offset=nlw;
run;
```

Chapter 23

Nonlinear Regression

23.1 Minitab

See

```
NLinear;
  Response C1;
  Continuous C2;
  Parameter "Theta1" 1;
  Parameter "Theta2" 1;
  Expectation Theta1 * C2 / ( Theta2 + C2 );
  NoDefault;
  GFCurve;
  TMethod;
  TStarting;
  TConstraints;
  TEquation;
  TParameters;
  TSummary;
  TPredictions.
```

23.2 SAS

Chapter 24

More Stuff

```
data oakes;
   infile 'oakes.dat';
   var n I J;
proc genmod;
class I J;
model n = I + J + I*j/ distribution = poisson;
```

Chapter 25

Bsplines

12 is much better although the fit is too wiggly in the first section and misses the point of inflection. We now create the B-spline basis. You need to have three additional knots at the start and end to get the right basis. I have chosen to the knot locations to put more in regions of greater curvature. I have used 12 basis functions for comparability to the orthogonal polynomial fit.

```
> library(splines)
> knots = c(0,0,0,0,0.2,0.4,0.5,0.6,0.7,0.8,0.85,0.9,1,1,1,1)
> bx = splineDesign(knots,x)
> gs model y =  bx)
> matplot(x,bx,type="l",main="B-spline basis functions")
> matplot(x,cbind(y,gs$fit),type="pl",ylab="y",pch=18,lty=1,main="Spline fit")
The basis functions themselves are shown

> gs model y =  bx / noconstant ;
```

Index