Statistical Modeling in S-PLUS

Aidan Palmer

May 26, 1997

1 Introduction

This handout explains how to perform statistical modeling, such as regression, in S-PLUS. The reader is assumed to have a working knowledge of S-PLUS and an understanding of statistical modeling concepts.

To follow along with this handout, go to the directory `/afs/andrew/stat/data/S-Tutorials` and copy the files `geyser.dat`, `education.dat`, `copenhagenhousing.dat`, `neurological.dat` and `anthro.dat` to your home directory.

2 Model Formulae

Every modeling function in S-PLUS needs a model formula, that is, a predictor or predictors and a response. The format for such a formula is the response, a `~` (tilde) meaning "modeled by", and the predictors. The predictors should be separated by `+`, or `*` if an interaction between two predictors is being modeled. An intercept term is included with the predictors by default, to remove it use `-1` as a predictor.

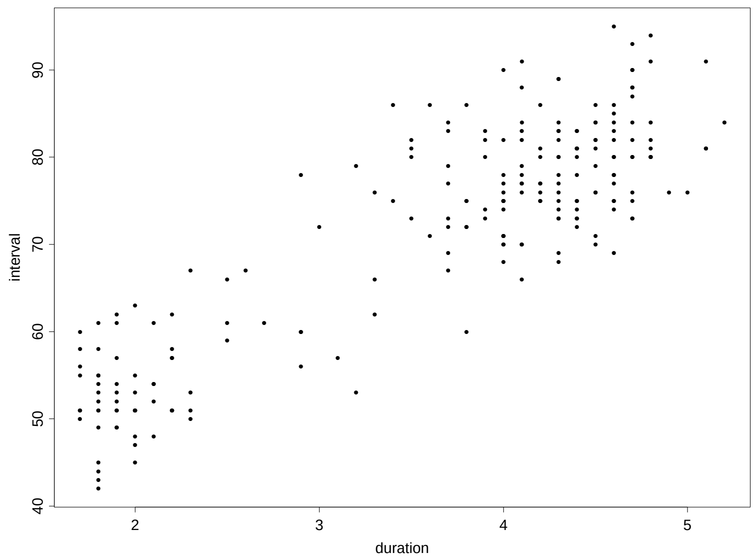
3 Regression

The file `geyser.dat` contains data about Old Faithful Geyser, a popular tourist attraction. A geologist was interested in predicting the length of time between eruptions based on the duration of eruptions. He observed the geyser several times in August 1978, and several more times in August 1979.

```
> geyser <- read.table("geyser.dat",col.names=c("date","interval","duration","is.aug.78","is.aug.79"
> attach(geyser)
```

The data file consists of what day the observation took place, the interval between eruptions (in minutes), the duration of the preceding eruption (in minutes), a 0/1 indicator variable for August 1978, and another 0/1 indicator variable for August 1979.

```
> motif()
> plot(duration, interval)
```



This picture shows an upward sloping relationship between `duration` and `interval`. The longer the eruption, the more time until the next eruption.

Let's start with only one predictor, `duration`. The formula would be "interval modeled by duration", which is `interval ~ duration`. The function to create a regression model is `lm`, for Linear Model.

```
> lm(interval ~ duration)
Call:
lm(formula = interval ~ duration)

Coefficients:
 (Intercept) duration
    33.90084 10.37363

Degrees of freedom: 221 total; 219 residual
Residual standard error: 6.170498
```

The `lm` function creates an object of class "linear model", and returns its value. The value includes the coefficients and other information. It is a good idea to store this object someplace so that you can update it and refer to it as you proceed with the analysis.

```
> geyser.lm <- lm(interval ~ duration)
```

The `summary` function, which you have probably used to generate summary statistics about a vector, comes in handy with model objects.

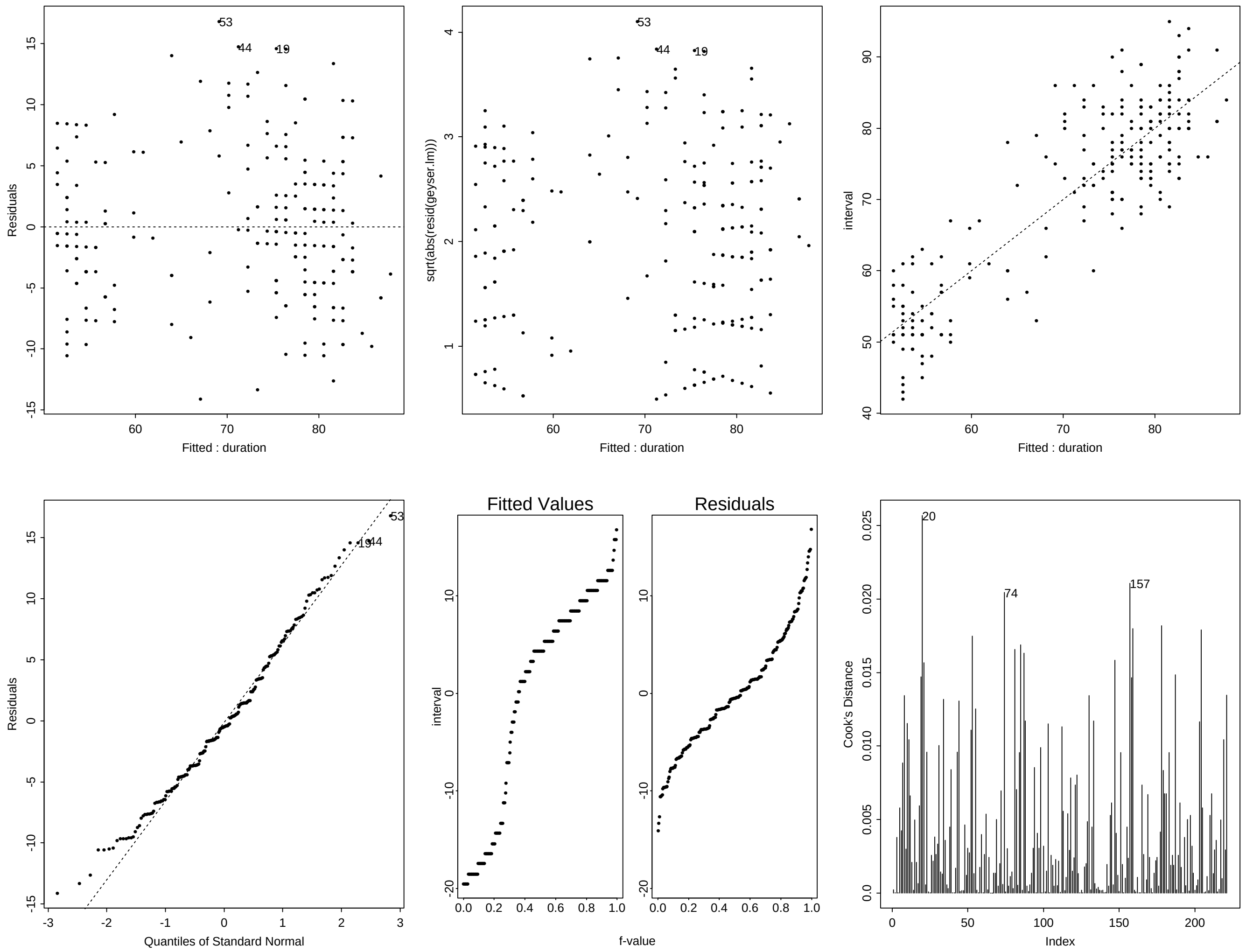
```
> summary(geyser.lm)

Call: lm(formula = interval ~ duration)
Residuals:
   Min     1Q  Median    3Q   Max
 -14.1 -4.507 -0.4701 4.194 16.83

Coefficients:
              Value Std. Error t value Pr(>|t|)
```

```
(Intercept) 33.9008  1.4409    23.5277  0.0000
   duration 10.3736  0.3850    26.9428  0.0000

Residual standard error: 6.17 on 219 degrees of freedom
Multiple R-Squared: 0.7682
F-statistic: 725.9 on 1 and 219 degrees of freedom, the p-value is 0

Correlation of Coefficients:
         (Intercept)
duration -0.9576
```

The `summary` command provides a bevy of useful information, including p-values, multiple R-Squared, and correlation. Based on this information, it looks like `duration` is a very good predictor of `interval`.

Another way to examine regression output is graphically, using the `plot` command. When given regression output as its argument, `plot` will produce six diagnostic plots. To see them all, you must split the plotting window into six or more parts. For instance, make it into a 2 by 3 grid of windows.

```
> par(mfrow=c(2,3))
> plot(geyser.lm)
```



The `par(mfrow=c(2,3))` command divided the graphics window into six separate windows. To change it back to normal, do `par(mfrow=c(1,1))` (that is, one row by one column). This command will take effect when the next plot is executed.

The six plots show useful diagnostic information, such as plots of residuals vs. fitted values, a quantile-normal plot, and a Cook's distance plot (the influence of each data point on the final model, large values indicate high influence).

Other functions which can be used on model objects include `residuals` (returns a vector of the residuals) and `fitted` (returns a vector of the fitted values).

4 Multiple Regression

The file `education.dat` consists of school expenditure in 1970 (SE70) using PI68 (average personal income, 1968), Y69 (school-aged population per capita, 1969), and Urban70 (urban population per capita, 1970) for the 50 states and Washington, DC. The data also have variables for region (general) and locale (specific). The question of interest is, "which variables have an effect on school expenditures?"

```
> education <- read.table("education.dat",header=T)
> is.factor(education$Region)
[1] T
> is.factor(education$Locale)
[1] T
> attach(education)
```

Let's try a model with all of the numeric predictors.

```
> summary(education.lm <- lm(SE70 ~ PI68 + Y69 + Urban70))

Call: lm(formula = SE70 ~ PI68 + Y69 + Urban70)
Residuals:
    Min     1Q Median    3Q   Max
 -60.24 -15.74 -1.156 15.88 51.38

Coefficients:
                Value Std. Error   t value  Pr(>|t|)
(Intercept) -286.8388   64.9199    -4.4183    0.0001
       PI68    0.0807    0.0093     8.6738    0.0000
        Y69    0.8173    0.1598     5.1151    0.0000
    Urban70   -0.1058    0.0343    -3.0863    0.0034

Residual standard error: 26.69 on 47 degrees of freedom
Multiple R-Squared: 0.6896
F-statistic: 34.81 on 3 and 47 degrees of freedom, the p-value is 5.337e-12

Correlation of Coefficients:
        (Intercept)    PI68     Y69
   PI68 -0.3064
    Y69 -0.9398      0.0934
Urban70 -0.0711     -0.6784  0.0381
```

This model appears to fit well (note the high multiple R-squared and small p-values). However, `Urban70` and `PI68` are highly collinear, so `Urban70` should be removed from the model.

5 Updating Models

The simplest way to remove `urban70` would be to create a new model which does not include it. However, retyping the formulae can get tiresome, especially when a lot of terms are involved. A fancier way to change the model is with the `update` function.

```
> education.lm <- update(education.lm, . ~ . - Urban70)
```

This assigns a new value to `education.lm`. The new value is `education.lm`, updated to have the same response (the `.` before the `~` means the same thing is being predicted) and using the same predictors, except without `Urban70`. The new model looks like this:

```
> summary(education.lm)

Call: lm(formula = SE70 ~ PI68 + Y69)
Residuals:
    Min     1Q Median    3Q   Max
 -51.42 -18.17 -1.768 15.32 53.16

Coefficients:
                Value Std. Error   t value  Pr(>|t|)
(Intercept) -301.0892   70.2713    -4.2847    0.0001
       PI68    0.0612    0.0074     8.2532    0.0000
        Y69    0.8361    0.1733     4.8253    0.0000

Residual standard error: 28.97 on 48 degrees of freedom
Multiple R-Squared: 0.6267
F-statistic: 40.3 on 2 and 48 degrees of freedom, the p-value is 5.354e-11

Correlation of Coefficients:
     (Intercept)    PI68
PI68 -0.4839
 Y69 -0.9402      0.1624
```

Let's check for an interaction between the two significant terms. Note that when you enter an interaction, the main effects are included as well.

```
> education.lm <- update(education.lm, . ~ PI68*Y69)
> summary(education.lm)

Call: lm(formula = SE70 ~ PI68 + Y69 + PI68:Y69)
Residuals:
    Min     1Q Median    3Q   Max
 -44.55 -17.02 -1.733 13.56 52.48

Coefficients:
                Value Std. Error   t value  Pr(>|t|)
(Intercept)  245.8215  328.7258     0.7478    0.4583
       PI68   -0.1017    0.0960    -1.0592    0.2949
        Y69   -0.6669    0.8995    -0.7414    0.4621
   PI68:Y69    0.0004    0.0003     1.7016    0.0954

Residual standard error: 28.41 on 47 degrees of freedom
Multiple R-Squared: 0.6484
F-statistic: 28.89 on 3 and 47 degrees of freedom, the p-value is 9.728e-11
```

```
Correlation of Coefficients:
         (Intercept)    PI68     Y69
    PI68 -0.9826
     Y69 -0.9974      0.9815
PI68:Y69  0.9778     -0.9971 -0.9820
```

The interaction term really messed up the model, so we should go back to the previous model by removing it.

```
> education.lm <- update(education.lm, . ~ . - PI68:Y69)
```

The `PI68:Y69` term means "the `PI68` and `Y69` interaction", so if we subtract `PI68:Y69` we are left with the main effects only.

6 Factors in Models

When a factor is used as a predictor in a model, S-PLUS calculates parameter estimates for each level of the factor by using Helmert contrasts. **Helmert contrasts are a real nuisance when it comes to interpreting the coefficients, so use treatment contrasts instead.** Type:

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

This means use treatment contrasts for regular factors, and the default polynomial contrasts for ordered factors. Since you probably don't want to type this every time you run S-PLUS, put this line in your `.First` function. The `.First` function is executed every time S-PLUS begins.

```
> .First <- function()
+ {
+ options(contrasts=c("contr.treatment","contr.poly"))
+ }
```

When treatment contrasts of a factor are used in a model, the first level of that factor (alphabetically) is taken as the baseline. The coefficients of the other levels represent the difference between their effect and the baseline effect.

Suppose we had a vector "Animal" in three levels: Birds, Cats, and Dogs, and we were using this vector in a model. We are interested in the difference between the three types of animals. S-PLUS would create two dummy vectors, "AnimalCats" and "AnimalDogs". For birds, the entry in both dummy vectors would be 0. For cats, AnimalCats would be 1 and AnimalDogs would be 0. Dogs would be coded by AnimalCats = 0 and AnimalDogs = 1. The two dummy vectors AnimalCats and AnimalDogs would then be used in the model. The resulting coefficient of AnimalCats would represent the difference between birds and cats, after accounting for other variables, and the coefficient of AnimalDogs would represent the difference between birds and dogs, after accounting for other variables.

Let's try using treatment contrasts in fitting terms for location to the education model. There are two factors we could use: `Region` and `Locale`. We can't use both, because each unit of `Region` consists of several units of `Locale`. If we include `Locale` in the model, `Region` would provide no useful information (S-PLUS would give a "computed fit is singular" warning about it). Perhaps we should use `Locale`, since it is more specific.

```
> summary(lm(SE70 ~ PI68 + Y69 + Locale))

Call: lm(formula = SE70 ~ PI68 + Y69 + Locale)
Residuals:
    Min     1Q Median    3Q   Max
```

```
 -50.97 -14.97 -3.194 15.71 54.13

Coefficients:
                    Value Std. Error   t value  Pr(>|t|)
   (Intercept) -234.4908   79.5045    -2.9494    0.0053
          PI68    0.0449    0.0092     4.8930    0.0000
           Y69    0.7608    0.1944     3.9142    0.0003
  LocaleESTCNT  -17.8049   20.7650    -0.8575    0.3963
  LocaleMIDATL   37.2958   20.2641     1.8405    0.0731
LocaleMOUNTAIN   17.1916   16.4521     1.0450    0.3023
  LocaleNEWENG    7.2652   16.4033     0.4429    0.6602
 LocalePACIFIC   47.4117   16.9029     2.8049    0.0077
  LocaleSTHATL   11.9626   15.2510     0.7844    0.4374
  LocaleWNRCNT   21.7977   16.0552     1.3577    0.1822
  LocaleWSTCNT  -15.0130   19.4544    -0.7717    0.4448

Residual standard error: 26.44 on 40 degrees of freedom
Multiple R-Squared: 0.7408
F-statistic: 11.43 on 10 and 40 degrees of freedom, the p-value is 6.413e-09
```

(S-PLUS also produces a huge matrix of correlations, which has been omitted to save space).

Locale has nine levels, the first of which, ENRCNT (East North-Central), is taken as the baseline. The coefficients of the others represent their difference from the East North-Central. From the p-values, only the Pacific locale has significantly different spending than expected using income and school-aged population per capita. Specifically, the mean Pacific spending on education is 47.4117 higher than in the East North-Central.

7 Removing Outliers

If you plot this model, you will see that there is one state with relatively large Cook's distance, Alaska. To exclude outliers from the analysis, tell S-PLUS to perform the model on only a subset of the data.

```
> summary(lm(SE70 ~ PI68 + Y69 + Locale, subset=-50))
```

The above will produce a summary of a regression excluding Alaska. Another way to do the exact same thing is:

```
> summary(lm(SE70 ~ PI68 + Y69 + Locale, data=education[-50,]))
```

The above means use `education`, without row 50 but with all columns. I think the first way is the "proper" thing to do. My understanding is that the `data` field exists to allow modeling without attaching to a data frame.

8 General Linear Models

Generalized Linear Models are fit in much the same way as regression models. Use the `glm` command and specify the family of model to fit (for instance poisson or binomial). Arguments like `subset=` and `data=` demonstrated above work in exactly the same way for generalized linear models. It is possible to perform linear regression in this format, by using `glm` with `family=gaussian`.

8.1 A Poisson Model

In 1960, researchers studied of tenants' satisfaction with their housing conditions in Copenhagen. The data (from the file `copenhagenhousing.dat`) are broken down into a four-dimensional table, with the dimensions being: type of housing (tower blocks, apartments, atrium houses, terraced houses), tenant's influence on management (low, medium, high), contact with other residents (low, high), and level of satisfaction (low, medium, high). The number of respondents in each category can be studied with a poisson model.

```
> housing <- read.table("copenhagenhousing.dat",col.names=c("HousingType",
+ "MgmtInfluence","Contact","Satisfaction","NumberRespondents"))
> attach(housing)
```

Now that the data have been read in, fit a generalized linear model to the data. If you use `summary` to examine the results, you will get a long list of coefficients and correlations, which you will have to sort through to find what you are looking for. An easier way to parse the results is through an ANOVA (analysis of variance) table. We don't want to use up all of the degrees of freedom, so we should omit the four-way interaction, which is unlikely to be significant anyway.

```
> copenhagen.glm <- glm(NumberRespondents ~ HousingType*MgmtInfluence*Contact*Satisfaction
+ - HousingType:MgmtInfluence:Contact:Satisfaction, family=poisson)
> anova(copenhagen.glm)
Analysis of Deviance Table

Poisson model

Response: NumberRespondents

Terms added sequentially (first to last)
                                    Df Deviance Resid. Df Resid. Dev
                               NULL                    71   833.6570
                        HousingType  3 376.3000        68   457.3570
                      MgmtInfluence  2  78.5163        66   378.8407
                            Contact  1  38.8321        65   340.0086
                       Satisfaction  2  44.6569        63   295.3518
          HousingType:MgmtInfluence  6  16.8914        57   278.4604
                HousingType:Contact  3  39.0578        54   239.4026
              MgmtInfluence:Contact  2  16.6992        52   222.7033
           HousingType:Satisfaction  6  60.6687        46   162.0346
         MgmtInfluence:Satisfaction  4 102.0653        42    59.9693
               Contact:Satisfaction  2  16.0175        40    43.9518
  HousingType:MgmtInfluence:Contact  6   5.2896        34    38.6622
HousingType:MgmtInfluence:Satisfaction 12  22.5550        22    16.1072
     HousingType:Contact:Satisfaction  6   9.2788        16     6.8284
   MgmtInfluence:Contact:Satisfaction  4   0.8841        12     5.9443
```

It looks like three of the three-way interactions are not significant (they only reduce the residual deviance by a small amount for their degrees of freedom). Use `update`, just like for regression, to remove them from the model.

```
> anova(copenhagen.glm <- update(copenhagen.glm, . ~ . - HousingType:MgmtInfluence:Contact
+ - HousingType:Contact:Satisfaction - MgmtInfluence:Contact:Satisfaction))
Analysis of Deviance Table

Poisson model
```

```
Response: NumberRespondents

Terms added sequentially (first to last)
                                      Df Deviance Resid. Df Resid. Dev
                                  NULL                   71   833.6570
                           HousingType  3 376.3000       68   457.3570
                         MgmtInfluence  2  78.5163       66   378.8407
                               Contact  1  38.8321       65   340.0086
                          Satisfaction  2  44.6569       63   295.3518
             HousingType:MgmtInfluence  6  16.8914       57   278.4604
                   HousingType:Contact  3  39.0578       54   239.4026
                 MgmtInfluence:Contact  2  16.6992       52   222.7033
              HousingType:Satisfaction  6  60.6687       46   162.0346
            MgmtInfluence:Satisfaction  4 102.0653       42    59.9693
                  Contact:Satisfaction  2  16.0175       40    43.9518
HousingType:MgmtInfluence:Satisfaction 12  21.8200       28    22.1318
```

Suppose the question of interest is, "which variables have an effect on resident satisfaction?" To answer that question, look at the terms interacting with `Satsifaction`. It appears that housing type, influence on management, and contact with other residents all have an effect on tenant satisfaction. Also, it appears that housing type and influence on management interact to have an effect on satisfaction.

Now, to check the direction of the effects:

```
> copenhagen.glm$coefficients
 (Intercept)
 4.135457
...
```

There is a lot of output, and it is not in a particularly legible format. Here is a subset of the output, after some editing:

```
 HousingTypeAtriumhouses:   -1.239491
 HousingTypeTerracedhouses: -1.444709
 HousingTypeTowerblocks:    -1.007473

 HousingTypeAtriumhousesSatisfactionLow:     0.2352837
 HousingTypeAtriumhousesSatisfactionMed:     0.432526
 HousingTypeTerracedhousesSatisfactionMed:   0.2217337
 HousingTypeTerracedhousesSatisfactionLow:   0.3838231
 HousingTypeTowerblocksSatisfactionLow:     -0.5432325
 HousingTypeTowerblocksSatisfactionMed:     -0.3314662
```

These coefficients say that fewer people live in atrium houses than apartments, fewer people live in terraced houses than apartments and fewer people live in tower blocks than apartments.

The next batch of coefficients reveals that tower blocks are the most preferred type of housing (fewer people have low or medium satisfaction), followed by apartments, and then by terraced houses and atrium houses.

You can examine the rest of the coefficients on your own.

8.2 A Binomial Model (logistic regression)

The file `neurological.dat` concerns patients suffering from a mild neurological disorder. Patients were treated with one of two drugs or a placebo, and the number recovering was recorded.

The data consist of five columns: sex, cured or not cured, number on placebo, number on drug A, number on drug B.

```
> neuro <- read.table("neurological.dat")
> neuro
  V1 V2 V3 V4 V5
1  F  1 40  5 26
2  F  0 43  7 32
3  M  1 11 48 52
4  M  0  6 20 20
```

This is not a good format in S-PLUS. It would be much better to have "number cured" and "number not cured" in the same row, with each of the treatments in a different row. Manipulating the data into such a form would be complicated, and since the data set is so small we may as well just do it by hand (if you have read the handout on objects, you know how to do this).

```
> neuro
  Sex Treatment Cured NotCured
1   F   Placebo    40       43
2   F   Drug A     5        7
3   F   Drug B     26       32
4   M   Placebo    11       6
5   M   Drug A     48       20
6   M   Drug B     52       20
```

Also, we should make sure all of these variables are in the right format:

```
> neuro$Sex <- as.factor(neuro$Sex)
> neuro$Treatment <- as.factor(neuro$Treatment)
> neuro$Cured <- as.numeric(neuro$Cured)
> neuro$NotCured <- as.numeric(neuro$NotCured)
```

For logistic regression, there are a pair of response variables: the number of successes and the number of failures. Bind the two responses together with `cbind` and put them in the model.

```
> attach(neuro)
> anova(glm(cbind(Cured,NotCured) ~ Sex + Treatment, family=binomial))
Analysis of Deviance Table

Binomial model

Response: cbind(Cured, NotCured)

Terms added sequentially (first to last)
          Df Deviance Resid. Df Resid. Dev
     NULL                     5   19.71457
      Sex  1 19.07467         4    0.63990
Treatment  2  0.04528         2    0.59462
```

It appears that the two drugs were really no different from the placebo. The big effect was from gender, and looking at the data above, men were much more likely to recover (70%) than women (40%).

9 Cox Proportional Hazards

The Cox Proportional Hazards model is used in survival analysis, which generally involves studying the effects of various variables on survival time. For instance, a study of the Herero people investigated the effect of gender on mortality. These data are in the file `anthro.dat`. The file contains a header as well as row names (codes for each subject).

The study ended in 1989. After that, we have no information on how much longer the surviving subjects lived. Record the last year that each subject was seen, and a 0/1 variable for whether they died before 1989. Those who did not die before 1989 are considered "censored".

I generated a vector (`year.last.seen`) which is the year of death (for people who died before 1989) or 1989 (for people who did not die before 1989). I then subtracted year of birth from this to get the age at which a person was last seen (`age.last.seen`). I created a vector (`is.dead`) with deaths represented as 1's and people still alive at the end of the study represented as 0's.

Question: How do you do this?

I used these two vectors to create a survival object:

```
> anthro.surv <- Surv(age.last.seen, is.dead)
> anthro.surv[1:30]
 [1] 25+ 33+ 28+  5+ 35+ 33+ 31+ 29+ 22  22+ 20+  0  21+ 15+ 13+  8+  7+ 15+ 32+
[20] 21  23+ 19+ 15+  3  38+ 36+ 34+ 32+ 28+ 25+
```

A survival object consists of survival times, followed by a `+` if that observation was censored (in this case meaning that they were still alive at the end of the study). This means that the first person was 25 and still alive at the end of the study, the second person was 33 and still alive, etc. The ninth person died at age 22, the 12th person died at age 0, etc.

Now that we have a vector in the survival analysis format, we can use that as the response variable in a Cox proportional hazards model. This sort of model uses the same format as before. This time there is a new argument included, `na.action=na.omit`, to skip over the several people whose gender was not recorded. The relevant variables appear to be year of birth, number of living siblings, and gender:

```
> coxph(anthro.surv ~ birth.year + sibs + sex, na.action=na.omit)
Call:
coxph(formula = anthro.surv ~ birth.year + sibs + sex, na.action = na.omit)


              coef exp(coef) se(coef)     z       p
birth.year -0.0323     0.968  0.00754 -4.29 1.8e-05
      sibs  0.0623     1.064  0.02405  2.59 9.6e-03
       sex  0.7681     2.156  0.15709  4.89 1.0e-06

Likelihood ratio test=52.2  on 3 df, p=2.73e-11  n=1262 (6 observations deleted
due to missing)
```

The above shows that `sex` does have a significant effect on mortality. Sex is listed alphabetically (female, male) so men have a higher hazard rate of dying than women.