Logistic Regression Example: Cow Shock data

by Michael Escobar

February 11, 2013

1 Introduction

This dataset looks at the reaction to cows to mild shocks. It looks to see if cows reacted when they were subject to mild electrical currents. The cows are recorded as either reacting or not, so the outcome is binary.

2 Description of Data

This data was obtained from Weisberg, S (pg 267 & 275, 1985, *Applied Linear Regression*). There it is mentioned that the data was collected by R. Norell. The purpose was to determine the effects of high-voltage power lines on livestock. To look at some similar papers (and perhaps which discuss this data), see Appleman and Gustafson (1985, "Source of stray voltage and effects on cow health and performance", *J Diary Sci, 68*, 1554-1567) or Norell, Appleman, Gustafson (1983, "Behavioral studies of dairy cattle sensitivity to electrical currents," *Trans. Am. Soc. Agric. Eng. 26*, 1506).

There were 7 cows which were each subjected to 30 shocks which were at 6 different intensities in random order. Then, the experiment was repeated (for a second block). The intensities were of 0, 1, 2, 3, 4, and 5 milliamps. (Note: Weisberg reports that a shock of 15 milliamps are considered painful for many humans.)

The purpose of this experiment was to look at the probability of response for the different current values. There are also a number of other variables which might effect the probability. The main underlying goal is to test the sensitivity of cows to electrical current due to stray votages on farms. Stray votages can effect cows in different ways so it is important to understand what levels do effect cows.

The data and further description are contained in an appendix to this handout.

3 Statistical methods

The basic approach is to look at this in the general linear model frame work. Therefore, here we need to discuss the random component, the link function, and the linear/systematic component. Also, we need to discuss the priors that are used.

3.1 Random component

The response that is recorded is the number of responses. For a given cow, block, and current intensity there are 5 different trials. There are 84 different unique combination

of cow, block and current intensity levels. Here we consider each unique combination as the "observation". Therefore, one might consider modelling the response as a binomial distribution if we condition on the expected response probability for 5 trials on each of the 84 combinations. et `i` indicate one of the 84 different combinations, `Y[i]` be the number of responses for the `i`-th combination, and let `p[i]` be the probability of a response under the `i`-th condition. Then, this can be expressed in a WinBug model as:

```
Y[i] ~ dbin(p[i],N[i])
```

That is, we specify that the observations `Y[i]` hav a binomial distribution with the probability of a response is `p[i]$ for observation \verbi+`.

3.2 Link function

The link function relates the random component to a linear component of the model. In frequentist theory, one does this by modelling the some function of the mean of the observations modelled in the random component as a linear function of parameters and covariates. This function of the mean is called the link function. So, in this case, one wants a function of `p[i]` to be a linear function of parameters and covariates. For models such as this one with a binomial random component, a common function to use a link function is the logistic function. If $\text{logit}(x)$ is the logistic function for a variable x, then it can be defined as:

$$\text{logit}(x) = \log\left(\frac{x}{1-x}\right)$$

There are other choices such as the probit link function which are sometimes used for binomial data, but for this example, only the logit function is used.

In Winbugs, one uses the notation `logit(p[i])` for the logit of `p[i]`.

3.3 Linear component

Now, to create this general linear model, we assume that the `logit(p[i])` can be expressed as a linear function of the covariates. To do this, we need to use a linear model to model `logit(p[i])` as a linear model of the different effects. That is, this is done by specifying parameters as linear function of X's which are used to represent the covariates for each of the `i` observations. Here we consider some possible issues.

- The variable "cow" is a categorical variable. There are 7 different cows so this is a 7 level variable. For this model, the redundant parameters technique is used. This technique is discussed in last weeks handout.

- The variable "block" is binary. So, this could be considered as a 2 level categorical variable. Alternatively, we could model this with one unknow parameter, say, `b.block` so that `-b.block` is the value for block 1 and `+b.block` is the value for block 2.

- The variable "current" could be either a continuous or a categorical variable. That is, we could model each intensity level with a separate parameter or we could model it as a linear (or higher order polynomial) function. Also, note that when the current is zero, there is never a response. Therefore, the probability of a response of zero when current is zero and the logit is negative infinite. So, we might consider removing these observations from the data. Therefore, here are some options for modelling the current effect:
  - If we model it as a linear function, then the current effect could be `b.curr*curr[i]`.
  - If we model it as a categorical effect, then the current effect could be `b.curr[curr[i]+1]`. The `+1` term is included since the variable `curr[i]` takes on the consecutive integer values from 0 to 5. So, by use `[curr[i]+1]`, the indices have consecutive integer values from 1 to 6. (Indices in WinBugs start from 1 and are consecutive integers after that.)
  - If we remove the observations when current is zero and we model the current effect as a categorical variable, then we could model the effect by `b.curr[curr[i]]`. Note, then in a frequentist analysis, the observations when current is zero causes problems because the logit(0) is negative infinity.
- Also, it is possible to include "extra-binomial" variation in this part of the model. A problem with the binomial model is that the conditional variance in the random component is completely determined by the mean. That is, the conditional variance of `Y[i]` given `p[i]` is `N*p[i]*(1-p[i])`. Now, it is possible that there are source of variation which are not accounted for in the model. For example, suppose there are covariates which are not included in the model or which are not even recorded. These extra source of variation can be included here by including a random effect in the systematic part of the model. This is sometimes referred to as accounting for "overdispersion".

So, if we have current as a categorical term and we include an adjustment for overdispersion as well as the other effects discussed above, one way to model the above in WinBugs is as follows:

```
logit(p[i]) <- b.cow[cow[i]] + b.block*(2*block[i]-3) +
                  b.curr[curr[i]+1] +b[i] +b.0
```

Also to account for the redundancy, the following lines are added to the model:

```
b.adj[i]<-b[i] -mean(b[])
b.cow.adj[jc]<-b.cow[jc] -mean(b.cow[])}
b.curr.adj[jcr]<-b.curr[jcr]-mean(b.curr[])}

b.0.adj<-b.0 +mean(b.cow[]) +mean(b.curr[])  + mean(b[])
```

Note: the above equations are inside different `for` loops. Below the entire model file is given and it is then obvious which `for` loop each of the equations are in.

3.4 Priors

So far, there are the following parameters in the model: `b.cow`, `b.block`, `b.curr`, `b`, and `b0`. From frequentist theory, it is known that the MLE's for these parameters is approximately normally distributed. If the posterior is dominated by the likelihood and then posterior will also be approximately normal. So, it might be reasonable to consider approximate normal priors for these parameters. When there are several parameters being sampled from the same distribution, then we might put a prior on the precision parameter. For the below, we first describe a prior which puts a gamma prior on the precision. Then, we describe a prior where the standard deviation has a flat, uniform prior.

The following priors are considered for these parameters:

- The parameter `b.cow` has 7 different levels. Since there are 7 levels, then if we assume they come from the same normal distribution, we could then use the 7 values to estimate the parameters of this distribution. So, let the mean be `b.cow.0` which has a fixed normal prior. The precision parameter, `tau.cow` is given a gamma distribution. So, the prior for `b.cow` could be:

```
for(jc in 1:7){
b.cow[jc]~dnorm(b.cow.0,tau.cow)
b.cow.adj[jc]<-b.cow[jc] -mean(b.cow[])}
b.cow.0~dnorm(0,.1)
tau.cow~dgamma(.1,.4)
```

  Note that the `b.cow.adj` is inside the `for` loop and is because of redundant parametrization. The parameters `b.cow`'s are not identifiable, but the parameters `b.cow.adj`'s are identifiable. The fixed priors for `b.cow.0` and `tau.cow` are set so that the log(odd ratio) are somewhere between -4 and 4. Note, if the log(odds ratio) is about 4, then the odd ratio is on the order 54. So, the log(odds ratio) of 4 or -4 is fairly large. (Well, actually, these priors provide a wider range than -4 and 4.)

- The priors for the parameter `b.curr` are modelled similar to the `b.cow` parameters. That is, it is modelled with the following WinBugs code:

```
for(jcr in 1:6){
b.curr[jcr]~dnorm(b.curr.0,tau.curr)
b.curr.adj[jcr]<-b.curr[jcr]-mean(b.curr[])}
b.curr.0~dnorm(0,.1)
tau.curr~dgamma(.1,.4)
```

- The parameter for the block effect, `b.block` is just a single parameter, here a simple normal prior is used with fixed parameters. The following is the WinBugs code:

```
b.block~dnorm(0,.01)
# log(odds) somewhere between -4 and 4...  so this is wide
```

Note that for a normal with precision .01, the standard deviation of this normal is 10. So, the 95% range for logit(p) is between -20 to 20. So, this is a wide range for the odds.

- The overdispersion parameter, `b[i]` has 84 different samples, so a normal prior is used with mean 0 and the precision is the parameter `tau`. The parameter `tau` has a gamma prior with fixed parameters. So, inside the `for` loop, there is the code:

  ```
  b[i]~dnorm(0, tau)
  b.adj[i]<-b[i] -mean(b[])
  ```

  and outside the `for` loop is the code:

  ```
  tau~dgamma(.1,.4)
  ```

- The parameter `b0` is the grand mean or y-intercept parameter. Here, a normal prior with fixed parameters is used. So, this is represented by the following code:

  ```
  b.0~dnorm(0,.001)
  b.0.adj<-b.0 +mean(b.cow[]) +mean(b.curr[])  + mean(b[])
  ```

As an alternative to modelling the precision parameters with a gamma, one could consider modelling the standard deviations with a flat prior. One possibility is to use a uniform prior with the left end point zero and letting the right end point be some large number. The reason that some people are advocating such a prior is that with the gamma prior, the shape parameter (the "α" parameter) for the posterior is something like $n/2$ plus the α from the prior. Therefore, the gamma distribution will have most of its mass away from zero and infinity. If believes that perhaps "all the means" are the same, then one would want to allow the posterior distribution of the precision to put possibly have mass near infinity. This is similar to having the posterior of the standard deviation possible condense on zero.

To do this, one would replace the statements like:

```
tau~dgamma(.1,.4)
tau.cow~dgamma(.1,.4)
tau.curr~dgamma(.1,.4)
```

with statements like:

```
sd~dunif(0,20)
sd.cow~dunif(0,20)
sd.curr~dunif(0,20)
tau<-1/sd/sd
tau.cow<-1/sd.cow/sd.cow
tau.curr<-1/sd.curr/sd.curr
```

Note that the above are examples of non-conjugate priors in a WinBugs model.

3.5 WinBugs Model file

Given the above, we can now specify the Winbug model file. This file contains the following model choices:

- The current intensity is modelled as a categorical variable.
- There is a term included in the model to account for binomial overdispersion.
- The variables are model using redundant parameters. That is, the primary categorical parameters are not identifiable. There is a set of adjusted parameters which are identifiable.

The file "CowShockSimpleRprog.txt" contains a series of R commands which run WinBugs. It contains the following elements:

- Sets the working directory.
- Reads the data from a file and writes the data to a data frame using the command `read.table`. Then, this data frame is attached.
- calls the R2WinBUGS package in R. This is contains programs which first writes the necessary information to a series of files. Then, these programs run WinBUGS in batch mode and then writes the information to a list variable.
- Uses the `cat` command to write the model file for WinBugs. This allows one to hold all the information about the WinBugs program in one large R file.
- Creates preliminary values needed to run Winbugs. This includes `data` which is list that contains the data variable names (as character values), a function `inits` which will be used to generate initial values of the parameters, and the character variable `parameters` which contains the parameters which are monitored.
- A call to the function `bugs` which will call WinBUGS and run WinBUGS in batch mode. The output to this command is a large list which contans information about the output of the MCMC runs.

The file is below:

```
#
#  CowShock  -- simple program file
#  Michael Escobar
#  Feb 8, 2011
#
 WorkDir<- "c:/mike workstation/Bayescourse9/Examples/CowShock"
# Note: change this to the file where everything is
 setwd(WorkDir)
#  specify where the WinBugs files are:
#  WinDir="c:/Program Files/WinBUGS14/"
WinDir="c:/MyProg/WinBugs14/"

CowShock=read.table("CowShock.dat",header=TRUE,sep = "")
attach(CowShock)

library(R2WinBUGS)
Nobs=84
data<-list("Nobs", "cow", "block","Y","curr","N")

# Making the WinBUGS model file
cat("
#
#######################
#
#  CowShockModRedOvrGam.txt
#  --model has: redundant parameters,overdisper, gamma precisions
#
model
{
for (i in 1:Nobs)
{
Y[i] ~ dbin(p[i],N[i])
logit(p[i]) <- b.cow[cow[i]] + b.block*(2*block[i]-3) + b.curr[curr[i]+1] +b[i] +b.0
b[i]~dnorm(0, tau)    # overdispersion parameter
b.adj[i]<-b[i] -mean(b[])
}
b.0~dnorm(0,.001)
b.0.adj<-b.0 +mean(b.cow[]) +mean(b.curr[])  + mean(b[]) # redundant parameter design
#
for(jc in 1:7){
b.cow[jc]~dnorm(b.cow.0,tau.cow)
b.cow.adj[jc]<-b.cow[jc] -mean(b.cow[])}
```

```
b.cow.0~dnorm(0,.1)
#
b.block~dnorm(0,.01)
# log(odds) somewhere between -4 and 4...  so this is wide
for(jcr in 1:6){
b.curr[jcr]~dnorm(b.curr.0,tau.curr)
b.curr.adj[jcr]<-b.curr[jcr]-mean(b.curr[])}
b.curr.0~dnorm(0,.1)
#
tau~dgamma(.1,.4)
tau.cow~dgamma(.1,.4)
tau.curr~dgamma(.1,.4)
}", file="AROGmod.txt")

inits<-function(){list(b.cow=rnorm(7,0,1),b.block=rnorm(1,0,1),
                       b.curr=rnorm(6,0,1),tau=runif(1,.5,2),
                       tau.cow=runif(1,.5,2),tau.curr=runif(1,.5,2),
                       b.cow.0=rnorm(1,0,1), b.curr.0=rnorm(1,0,1),
                       b.0=rnorm(1,0,1),b=rnorm(Nobs,0,1))
                       }
parameters=c("b.cow","b.cow.adj","b.curr","b.curr.adj","b.0","b.0.adj",
      "b.block","b","b.adj","tau","tau.cow","tau.curr")
CowShockROG<-bugs(data,inits,parameters, model.file="AROGmod.txt",
        n.chains=3,n.iter=5100,n.burnin=100,
        n.thin=1, bugs.directory=WinDir#,debug=TRUE
                  )
print(CowShockROG)
```

3.6 Different Models Considered

For this data several different models are fit. The following is a code for the models on the whole data sets (including when current has the value of 0):

ROG Redundent modelling of categorical variables (cow and current), overdispersing modelled, and gamma prior for precisions.

ROdd Same as ROG, but program initialized with more disperse starting points.

ROU Redundent modelling of categorical variables (cow and current), overdispersing modelled, and uniform priors for standard deviations.

R Redundent modelling of categorical variables (cow and current), no overdispersing modelled, and gamma prior for precisions.

all 6 sets of models.

RU Redundent modelling of categorical variables (cow and current), no overdispersing modelled, and uniform priors for standard deviations.

LO Current is modelled as a linear effect, overdispersing modelled, and gamma prior for precisions.

LOU Current is modelled as a linear effect, overdispersing modelled, and uniform prior for standard deviations.

L Current is modelled as a linear effect, no overdispersing modelled, and gamma prior for precisions.

LU Current is modelled as a linear effect, no overdispersing modelled, and uniform prior for standard deviations.

When the data is subsetted to remove the observations when the value of current is zero, then the same models are run. The codes then have an "NZ" added to the end of the code.

3.7 Deviance Information Criteria

The Deviance information criteria (DIC) is going to be discussed later in this course. Basically it is one way to calculate how good the models fit the data. Some of the pro's and con's are going to be discussed at a later time. However, one thing to note is that it is fairly easy to compute. Basically, one calculates the deviance statistics from the data and the parameters at each iteration of the MCMC chain. Then one can calculate the posterior mean of the deviance. Also, to control for the number of parameters, one can compute a measure of the "effective number of parameters" which is called "pD". The sum of the mean of the deviance and pD the to get the statistics DIC. Basically, this is similar to other model fit statistics such as AIC and BIC. It is desirable to have a low DIC for a model.

4 Results

4.1 Values of the DIC

The following is the DIC values for the different models when the whole data set is used:

Model	pD	DIC	Deviance
ROG	34.03	167.97	133.94
ROdd	33.89	167.96	134.08
ROU	35.15	167.79	132.64
R	12.01	182.03	170.02
RU	12.06	181.88	169.82
LO	33.63	167.61	133.98
LOU	34.32	168.11	133.79
L	8.22	183.91	175.70
LU	8.39	184.33	175.95

The following is the DIC values for the different models when only the observations when the current is not zero:

Model	pD	DIC	Deviance
ROGNZ	33.45	166.88	133.42
ROddNZ	33.12	166.76	133.64
ROUNZ	34.94	166.57	131.62
RNZ	11.28	180.26	168.98
RUNZ	11.41	180.56	169.15
LONZ	31.95	164.42	132.47
LOUNZ	33.71	164.82	131.10
LNZ	8.22	178.61	170.39
LUNZ	8.48	179.22	170.74

From the above, we see that the lowest models are the models when there is an overdispersion parameter. Also, there is little different in this dataset between the models where the precision is modelled with a gamma distribution or where the standard deviation is model with a uniform distribution. Also, there is little difference between the models where the current effect is modelled with a categorical variable or as a linear term. Also, note that it is inappropriate to use these two table to compare whether one should have dropped the observations where the currrent was zero. This is because one should only compare the DIC when the same data is used.

Therefore, the DIC statistics would seem to point to the models where the overdispersion is modelled. There is no preference between the gamma or uniform priors for the precision/standard deviation. Since there is no preference between modelling current as a categorical or as linear effect, then by the usual preference for parsimony, this would mean that one would prefer to model current as a linear term.

To look at the difference between modelling the zero valued current, let us look at the fitted parameter values between two of these models. The first output is from the model with all the observations, precision modelled with a gamma, and with an overdispersion parameter. Here is some truncated output:

```
> print(CowShockROG)
Inference for Bugs model at "AROGmod.txt", fit using WinBUGS,
 3 chains, each with 5100 iterations (first 100 discarded)
 n.sims = 15000 iterations saved
               mean   sd  2.5%   25%   50%   75% 97.5% Rhat n.eff
b.cow.adj[1]    0.9  0.5   0.0   0.6   0.9   1.3   2.1  1.0  6300
b.cow.adj[2]    0.3  0.5  -0.7   0.0   0.3   0.6   1.3  1.0  5300
b.cow.adj[3]   -0.2  0.5  -1.2  -0.5  -0.2   0.1   0.7  1.0  3700
b.cow.adj[4]    0.2  0.5  -0.7  -0.1   0.2   0.5   1.1  1.0 15000
b.cow.adj[5]   -0.7  0.5  -1.6  -1.0  -0.6  -0.3   0.2  1.0  4400
b.cow.adj[6]   -1.0  0.5  -2.0  -1.3  -1.0  -0.7   0.0  1.0 15000
b.cow.adj[7]    0.5  0.5  -0.4   0.2   0.5   0.8   1.4  1.0 15000
b.curr.adj[1]  -7.2  3.4 -15.9  -8.1  -6.4  -5.2  -3.8  1.0   250
b.curr.adj[2]  -1.6  0.8  -3.0  -2.1  -1.7  -1.3   0.2  1.0   450
b.curr.adj[3]  -0.4  0.8  -1.5  -0.8  -0.5   0.0   1.5  1.0   410
b.curr.adj[4]   1.9  0.8   0.7   1.4   1.7   2.2   3.7  1.0   320
b.curr.adj[5]   3.4  0.9   2.1   2.8   3.2   3.8   5.4  1.0   300
b.curr.adj[6]   4.0  1.0   2.6   3.4   3.8   4.4   6.2  1.0   520
b.0.adj        -0.8  0.7  -2.5  -1.0  -0.7  -0.5  -0.1  1.0   280
b.block        -0.6  0.2  -1.1  -0.8  -0.6  -0.5  -0.2  1.0  2700
tau             1.0  0.7   0.3   0.5   0.8   1.2   3.0  1.0  1200
tau.cow         1.6  1.3   0.2   0.7   1.2   2.0   5.1  1.0  6900
tau.curr        0.1  0.1   0.0   0.0   0.1   0.1   0.2  1.0   290
deviance      133.9 12.2 111.7 125.2 133.5 142.2 159.1  1.0  2700

For each parameter, n.eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = Dbar-Dhat)
pD = 34.0 and DIC = 168.0
DIC is an estimate of expected predictive error (lower deviance is better).
>
```

The following is from the subsetted data set where the observations with a current of zero are removed. Again, this is from the the model with the precision modelled with a gamma and with an overdispersion parameter. Here is some truncated output:

```
> print(CowShockROGNZ)
Inference for Bugs model at "AROGNZmod.txt", fit using WinBUGS,
 3 chains, each with 5100 iterations (first 100 discarded)
 n.sims = 15000 iterations saved
               mean   sd  2.5%   25%   50%   75% 97.5% Rhat n.eff
```

```
b.cow.adj[1]    1.0  0.5   0.0   0.6   0.9   1.3   2.1  1.0  2300
b.cow.adj[2]    0.3  0.5  -0.7   0.0   0.3   0.6   1.3  1.0  3700
b.cow.adj[3]   -0.2  0.5  -1.2  -0.5  -0.2   0.1   0.7  1.0 15000
b.cow.adj[4]    0.2  0.5  -0.7  -0.1   0.2   0.5   1.1  1.0  4100
b.cow.adj[5]   -0.7  0.5  -1.6  -1.0  -0.6  -0.3   0.2  1.0  3800
b.cow.adj[6]   -1.0  0.5  -2.1  -1.4  -1.0  -0.7  -0.1  1.0   940
b.cow.adj[7]    0.5  0.5  -0.4   0.2   0.5   0.8   1.5  1.0  5700
b.curr.adj[1]  -3.0  0.5  -4.1  -3.3  -2.9  -2.6  -2.1  1.0  1100
b.curr.adj[2]  -1.8  0.4  -2.7  -2.0  -1.7  -1.5  -1.0  1.0   780
b.curr.adj[3]   0.4  0.4  -0.4   0.1   0.4   0.7   1.3  1.0  6300
b.curr.adj[4]   1.9  0.5   1.0   1.5   1.8   2.2   2.9  1.0   650
b.curr.adj[5]   2.5  0.6   1.4   2.1   2.4   2.8   3.7  1.0  1900
b.0             3.9  4.9  -5.0   0.7   3.4   6.8  13.3  1.2    14
b.0.adj         0.6  0.2   0.2   0.5   0.6   0.7   1.0  1.0  1400
b.block        -0.6  0.2  -1.1  -0.8  -0.6  -0.5  -0.2  1.0  2700
tau             1.0  0.9   0.3   0.5   0.8   1.2   3.5  1.0   260
tau.cow         1.5  1.3   0.2   0.7   1.1   1.9   4.9  1.0  1500
tau.curr        0.2  0.1   0.0   0.1   0.2   0.3   0.6  1.0  5100
deviance      133.4 12.3 111.0 124.7 132.7 141.5 159.0  1.0   500

For each parameter, n.eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor (at convergence, Rhat=1).

DIC info (using the rule, pD = Dbar-Dhat)
pD = 33.5 and DIC = 166.9
DIC is an estimate of expected predictive error (lower deviance is better).
>
```

Note, that when comparing the `b.curr.adj` between the two models, the parameter `b.curr.adj[1]` for the first model is when current is 0. So, in the first dataset, `b.curr.adj[2]` is for current of 1 milliamp while in the second model, `b.curr.adj[1]` is for the current level of 1 milliamp. Also, when looking at the two sets of output, the `b.curr.adj` for the first model has values consistently about 1.5 higher than the similar value for the second model. So, it appears that the two models are fitting roughly the same values.

For some plots of this, they are contained in a file.

To run all these models and plots, one can do this with the file: `CowShockAllModelData.txt`

5 Appendix: Description of the Data Set

5.1 Description of Data

First, here is a description of the model file with a description of the data:
++++++++++++++++++++++++++++++++++

```
# Description of the data:
#
#
# The data in shock.dat summarizes an experiment carried out by R.
# Norell.  Of interest is the effect of small electrical currents on
# farm animals, with the eventual goal of understanding the effects of
# high-voltage powerlines on livestock.  This experiment was carried out
# with 7 cows, and 6 shock intensities, 0, 1, 2, 3, 4, and 5 milliamps
# shocks on the order of 15 milliamps are painful for many humans;
# Dalziel el al., 1941).  Each cow was given 30 shocks, five at each
# intensity, in random order.  The entire experiment was then repeated,
# so each cow received a total of 60 shocks.  For each shock, the
# response, mouth movement, was either present or absent.  All the
# observations in the second block were taken after all of the
# observations in the first block; results may differ between blocks
# due to fatigue of the animals or due to learning.
#
#
# Column 1: Cow number
# Column 2: Block number
# Column 3: Response number.  (The number of responses out of five.)
# Column 4: The current level.  A value from 0 to 5 in milliamps. ## no zero
# Column 5: The total trials for each current level, Cow, Block
# combination.
#
# (data from Weisberg, 1985, page 275.)
# --------------------------------
#
```

5.2 Data file

The data file (CowShock.dat):

cow	block	y	curr	N	
1	1	0	0	5	
1	1	1	1	5	
1	1	1	2	5	
1	1	5	3	5	
1	1	5	4	5	
1	1	5	5	5	
1	2	0	0	5	
1	2	0	1	5	
1	2	4	2	5	
1	2	5	3	5	
1	2	5	4	5	
1	2	5	5	5	
2	1	0	0	5	
2	1	0	1	5	
2	1	2	2	5	
2	1	5	3	5	
2	1	5	4	5	
2	1	5	5	5	
2	2	0	0	5	
2	2	0	1	5	
2	2	0	2	5	
2	2	4	3	5	
2	2	5	4	5	
2	2	5	5	5	
3	1	0	0	5	
3	1	0	1	5	
3	1	1	2	5	
3	1	3	3	5	
3	1	5	4	5	
3	1	5	5	5	
3	2	0	0	5	
3	2	0	1	5	
3	2	0	2	5	
3	2	3	3	5	
3	2	5	4	5	
3	2	5	5	5	
4	1	0	0	5	
4	1	0	1	5	
4	1	3	2	5	
4	1	3	3	5	
4	1	4	4	5	

4	1	4	5	5
4	2	0	0	5
4	2	2	1	5
4	2	2	2	5
4	2	3	3	5
4	2	5	4	5
4	2	5	5	5
5	1	0	0	5
5	1	1	1	5
5	1	1	2	5
5	1	3	3	5
5	1	4	4	5
5	1	5	5	5
5	2	0	0	5
5	2	1	1	5
5	2	0	2	5
5	2	1	3	5
5	2	4	4	5
5	2	3	5	5
6	1	0	0	5
6	1	2	1	5
6	1	2	2	5
6	1	2	3	5
6	1	5	4	5
6	1	5	5	5
6	2	0	0	5
6	2	0	1	5
6	2	0	2	5
6	2	2	3	5
6	2	0	4	5
6	2	1	5	5
7	1	0	0	5
7	1	2	1	5
7	1	3	2	5
7	1	5	3	5
7	1	5	4	5
7	1	5	5	5
7	2	0	0	5
7	2	0	1	5
7	2	2	2	5
7	2	3	3	5
7	2	3	4	5

7 2 5 5 5