Generalized Linear Models in R

Charles J. Geyer

December 8, 2003

This used to be a section of my master's level theory notes. It is a bit overly theoretical for this R course. Just think of it as an example of literate programming in R using the `Sweave` function. You don't have to absorb all the theory, although it is there for your perusal if you are interested.

1 Bernoulli Regression

We start with a slogan

- Categorical *predictors* are no problem for linear regression. Just use "dummy variables" and proceed normally.

but

- Categorical *responses* do present a problem. Linear regression assumes normally distributed responses. Categorical variables can't be normally distributed.

So now we learn how to deal with at least one kind of categorical response, the simplest, which is Bernoulli.

Suppose the responses are

$$Y_i \sim \text{Bernoulli}(p_i) \tag{1}$$

contrast this with the assumptions for linear regression

$$Y_i \sim \text{Normal}(\mu_i, \sigma^2) \tag{2}$$

and

$$\boldsymbol{\mu} = \mathbf{X}\boldsymbol{\beta} \tag{3}$$

The analogy between (1) and (2) should be clear. Both assume the data are independent, but not identically distributed. The responses Y_i have distributions in the same family, but not the same parameter values. So all we need to finish the specification of a regression-like model for Bernoulli is an equation that takes the place of (3).

1.1 A Dumb Idea (Identity Link)

We could use (3) with the Bernoulli model, although we have to change the symbol for the parameter from $\boldsymbol{\mu}$ to $\mathbf{p}$

$$\mathbf{p} = \mathbf{X}\boldsymbol{\beta}.$$

This means, for example, in the "simple" linear regression model (with one constant and one non-constant predictor x_i)

$$p_i = \alpha + \beta x_i. \tag{4}$$

Before we further explain this, we caution that this is universally recognized to be a dumb idea, so don't get too excited about it.

Now nothing is normal, so least squares, t and F tests, and so forth make no sense. But maximum likelihood, the asymptotics of maximum likelihood estimates, and likelihood ratio tests do make sense.

Hence we write down the log likelihood

$$l(\alpha, \beta) = \sum_{i=1}^{n} \bigl[y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \bigr]$$

and its derivatives

$$\begin{aligned} \frac{\partial l(\alpha, \beta)}{\partial \alpha} &= \sum_{i=1}^{n} \left[\frac{y_i}{p_i} - \frac{1 - y_i}{1 - p_i} \right] \\ \frac{\partial l(\alpha, \beta)}{\partial \beta} &= \sum_{i=1}^{n} \left[\frac{y_i}{p_i} - \frac{1 - y_i}{1 - p_i} \right] x_i \end{aligned}$$

and set equal to zero to solve for the MLE's. Fortunately, even for this dumb idea, R knows how to do the problem.

Example 1.1 (Bernoulli Regression, Identity Link).
We use the data in the file `ex12.8.1.dat` in this directory, which is read by the following

```
> X <- read.table("ex12.8.1.dat", header = TRUE)
> names(X)

[1] "y" "x" "z"

> attach(X)
```

and has three variables `x`, `y`, and `z`. For now we will just use the first two.

The response `y` is Bernoulli. We will do a Bernoulli regression using the model assumptions described above. The following code does the regression and prints out a summary.

```
> out.quasi <- glm(y ~ x, family = quasi(variance = "mu(1-mu)"),
+     start = c(0.5, 0))
> summary(out.quasi, dispersion = 1)

Call:
glm(formula = y ~ x, family = quasi(variance = "mu(1-mu)"), start = c(0.5,
    0))

Deviance Residuals:
   Min      1Q  Median      3Q     Max
-1.552  -1.038  -0.678   1.119   1.827

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept) -0.354863   0.187324  -1.894   0.0582 .
x            0.016016   0.003589   4.462  8.1e-06 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for quasi family taken to be 1)

    Null deviance: 137.19  on 99  degrees of freedom
Residual deviance: 126.96  on 98  degrees of freedom
AIC: NA

Number of Fisher Scoring iterations: 9
```

We have to apologize for the rather esoteric syntax, which results from our choice of introducing Bernoulli regression via this rather dumb example.

As usual, our main interest is in the table labeled `Coefficients:`, which says the estimated regression coefficients (the MLE's) are $\hat{\alpha} = -0.34750$ and $\hat{\beta} = 0.01585$. This table also gives standard errors, test statistics ("z values") and P-values for the two-tailed test of whether the true value of the coefficient is zero.

The scatter plot with regression line for this regression is somewhat unusual looking. It is produced by the code

```
> plot(x, y)
> curve(predict(out.quasi, data.frame(x = x)), add = TRUE)
```

and is shown in Figure 1. The response values are, of course, being Bernoulli, either zero or one, which makes the scatter plot almost impossible to interpret (it is clear that there are more ones for high x values than for low, but it's impossible to see much else, much less to visualize the correct regression line).

That finishes our discussion of the example. So why is it "dumb"? One reason is that nothing keeps the parameters in the required range. The p_i,

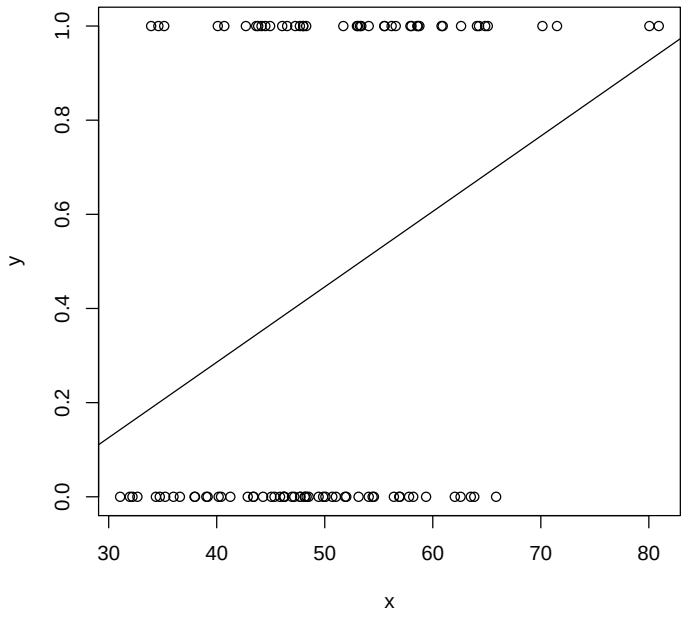

Figure 1: Scatter plot and regression line for Example 1.1 (Bernoulli regression with an identity link function).

being probabilities must be between zero and one. The right hand side of (4), being a linear function may take any values between $-\infty$ and $+\infty$. For the data set used in the example, it just happened that the MLE's wound up in $(0, 1)$ without constraining them to do so. In general that won't happen. What then? R being semi-sensible will just crash (produce error messages rather than estimates).

There are various ad-hoc ways one could think to patch up this problem. One could, for example, truncate the linear function at zero and one. But that makes a nondifferentiable log likelihood and ruins the asymptotic theory. The only simple solution is to realize that linearity is no longer simple and give up linearity.

1.2 Logistic Regression (Logit Link)

What we need is an assumption about the p_i that will always keep them between zero and one. A great deal of thought by many smart people came up with the following general solution to the problem. Replace the assumption (3) for linear regression with the following two assumptions

$$\boldsymbol{\eta} = \mathbf{X}\boldsymbol{\beta} \tag{5}$$

and

$$p_i = h(\eta_i) \tag{6}$$

where h is a smooth invertible function that maps $\mathbb{R}$ into $(0, 1)$ so the p_i are always in the required range. We now stop for some important terminology.

- The vector $\boldsymbol{\eta}$ in (5) is called the *linear predictor*.
- The function h is called the *inverse link function* and its inverse $g = h^{-1}$ is called the *link function*.

The most widely used (though not the only) link function for Bernoulli regression is the *logit* link defined by

$$g(p) = \operatorname{logit}(p) = \log\left(\frac{p}{1-p}\right) \tag{7a}$$

$$h(\eta) = g^{-1}(\eta) = \frac{e^\eta}{e^\eta + 1} = \frac{1}{1 + e^{-\eta}} \tag{7b}$$

Equation (7a) defines the so-called *logit* function, and, of course, equation (7b) defines the inverse logit function.

For generality, we will not at first use the explicit form of the link function writing the log likelihood

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n} \bigl[y_i \log(p_i) + (1 - y_i) \log(1 - p_i) \bigr]$$

where we are implicitly using (5) and (6) as part of the definition. Then

$$\frac{\partial l(\boldsymbol{\beta})}{\partial \beta_j} = \sum_{i=1}^{n} \left[\frac{y_i}{p_i} - \frac{1-y_i}{1-p_i} \right] \frac{\partial p_i}{\partial \eta_i} \frac{\partial \eta_i}{\partial \beta_j}$$

where the two partial derivatives on the right arise from the chain rule and are explicitly

$$\frac{\partial p_i}{\partial \eta_i} = h'(\eta_i)$$
$$\frac{\partial \eta_i}{\partial \beta_j} = x_{ij}$$

where x_{ij} denotes the i, j element of the design matrix $\mathbf{X}$ (the value of the j-th predictor for the i-th individual). Putting everything together

$$\frac{\partial l(\boldsymbol{\beta})}{\partial \beta_j} = \sum_{i=1}^{n} \left[\frac{y_i}{p_i} - \frac{1-y_i}{1-p_i} \right] h'(\eta_i) x_{ij}$$

These equations also do not have a closed form solution, but are easily solved numerically by R

Example 1.2 (Bernoulli Regression, Logit Link).
We use the same data in Example 1.1. The R commands for logistic regression are

```
> out.logit <- glm(y ~ x, family = binomial)
> summary(out.logit)

Call:
glm(formula = y ~ x, family = binomial)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5237  -1.0192  -0.7082   1.1341   1.7665

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -3.56633    1.15954  -3.076  0.00210 **
x            0.06607    0.02259   2.925  0.00345 **
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 137.19  on 99  degrees of freedom
Residual deviance: 127.33  on 98  degrees of freedom
```

```
AIC: 131.33

Number of Fisher Scoring iterations: 4
```

Note that the syntax is a lot cleaner for this (logit link) than for the "dumb" way (identity link). The regression function for this "logistic regression" is shown in Figure 2, which appears later, after we have done another example.

1.3 Probit Regression (Probit Link)

Another widely used link function for Bernoulli regression is the *probit* function, which is just another name for the standard normal inverse c. d. f. That is, the link function is $g(p) = \Phi^{-1}(p)$ and the inverse link function is $g^{-1}(\eta) = \Phi(\eta)$. The fact that we do not have closed-form expressions for these functions and must use table look-up or computer programs to evaluate them is no problem. We need computers to solve the likelihood equations anyway.

Example 1.3 (Bernoulli Regression, Probit Link).
We use the same data in Example 1.1. The R commands for probit regression are

```
> out.probit <- glm(y ~ x, family = binomial(link = "probit"))
> summary(out.probit)

Call:
glm(formula = y ~ x, family = binomial(link = "probit"))

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5263  -1.0223  -0.7032   1.1324   1.7760

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.20894    0.68649  -3.218  0.00129 **
x            0.04098    0.01340   3.057  0.00223 **
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 137.19  on 99  degrees of freedom
Residual deviance: 127.27  on 98  degrees of freedom
AIC: 131.27

Number of Fisher Scoring iterations: 4
```

Note that there is a huge difference in the regression coefficients for our three examples, but this should be no surprise because the coefficients for the three

regressions are not comparable. Because the regressions involve different link functions, the *meaning* of the regression coefficients are not the same. Comparing them is like comparing apples and oranges, as the saying goes. Thus Bernoulli regression in particular and generalized linear models in general give us yet another reason why *regression coefficients are meaningless*. Note that Figure 2 shows that the estimated regression functions $E(Y \mid X)$ are almost identical for the logit and probit regressions despite the regression coefficients being wildly different. Even the linear regression function used in our first example is not so different, at least in the middle of the range of the data, from the other two.

> *Regression functions (response predictions) have a direct probabilistic interpretation* $E(Y \mid X)$.
>
> *Regression coefficients don't.*

The regression function $E(Y \mid X)$ for all three of our Bernoulli regression examples, including this one, are shown in Figure 2, which was made by the following code

```
> plot(x, y)
> curve(predict(out.logit, data.frame(x = x), type = "response"),
+     add = TRUE, lty = 1)
> curve(predict(out.probit, data.frame(x = x), type = "response"),
+     add = TRUE, lty = 2)
> curve(predict(out.quasi, data.frame(x = x)), add = TRUE, lty = 3)
```

The `type="response"` argument says we want the predicted mean values $g(\boldsymbol{\eta})$, the default being the linear predictor values $\boldsymbol{\eta}$. The reason why this argument is not needed for the last case is because there is no difference with an identity link.

2 Generalized Linear Models

A *generalized linear model* (GLM) is a rather general (duh!) form of model that includes ordinary linear regression, logistic and probit regression, and lots more. We keep the regression-like association (5) between the regression coefficient vector $\boldsymbol{\beta}$ and the *linear predictor* vector $\boldsymbol{\eta}$ that we used in Bernoulli regression. But now we generalize the probability model greatly. We assume the responses Y_i are independent but not identically distributed with densities of the form

$$f(y \mid \theta, \phi, w) = \exp\left(\frac{y\theta - b(\theta)}{\phi/w} - c(y, \phi)\right) \tag{8}$$

We assume

$$Y_i \sim f(\cdot \mid \theta_i, \phi, w_i),$$

that is, the *canonical parameter* θ_i is different for each case and is determined (in a way yet to be specified) by the linear predictor η_i but the so-called *dispersion*

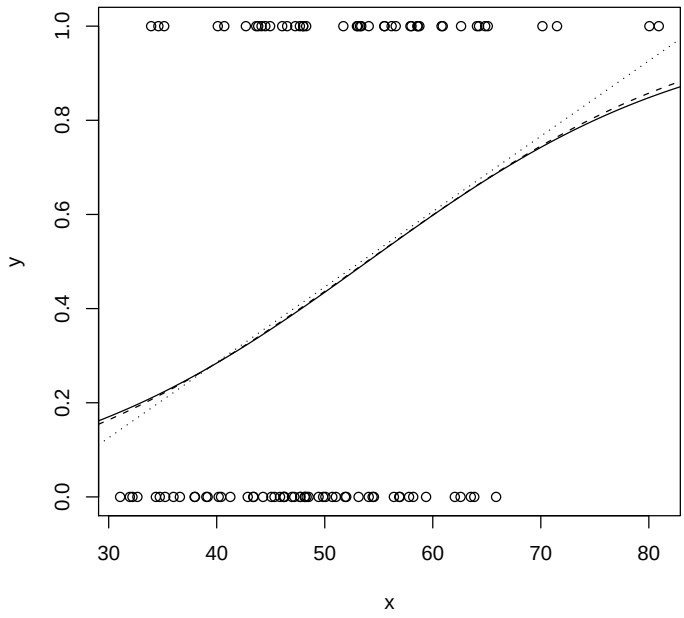

Figure 2: Scatter plot and regression functions for Examples 1.1, 1.2, and 1.3. Solid line: regression function for logistic regression (logit link). Dashed line: regression function for probit regression (probit link). Dotted line: regression function for no-name regression (identity link).

parameter ϕ is the same for all Y_i. The *weight* w_i is a known positive constant, not a parameter. Also $\phi > 0$ is assumed ($\phi < 0$ would just change the sign of some equations with only trivial effect). The function b is a smooth function but otherwise arbitrary. Given b the function c is determined by the requirement that f integrate to one (like any other probability density).

The log likelihood is thus

$$l(\boldsymbol{\beta}) = \sum_{i=1}^{n} \left(\frac{y_i \theta_i - b(\theta_i)}{\phi / w_i} - c(y_i, \phi) \right) \tag{9}$$

Before we proceed to the likelihood equations, let us first look at what the identities derived from differentiating under the integral sign

$$E_\theta\{l_n'(\theta)\} = 0 \tag{10}$$

and

$$E_\theta\{l_n''(\theta)\} = -\operatorname{var}_\theta\{l_n'(\theta)\} \tag{11}$$

and their multiparameter analogs

$$E_{\boldsymbol{\theta}}\{\nabla l_n(\boldsymbol{\theta})\} = 0 \tag{12}$$

and

$$E_{\boldsymbol{\theta}}\{\nabla^2 l_n(\boldsymbol{\theta})\} = -\operatorname{var}_{\boldsymbol{\theta}}\{\nabla l_n(\boldsymbol{\theta})\} \tag{13}$$

tell us about this model. Note that these identities are exact, not asymptotic, and so can be applied to sample size one and to any parameterization. So let us differentiate one term of (9) with respect to its θ parameter

$$\begin{aligned} l(\theta, \phi) &= \frac{y\theta - b(\theta)}{\phi / w} - c(y, \phi) \\ \frac{\partial l(\theta, \phi)}{\partial \theta} &= \frac{y - b'(\theta)}{\phi / w} \\ \frac{\partial^2 l(\theta, \phi)}{\partial \theta^2} &= -\frac{b''(\theta)}{\phi / w} \end{aligned}$$

Applied to this particular situation, the identities from differentiating under the integral sign are

$$\begin{aligned} E_{\theta,\phi}\left\{ \frac{\partial l(\theta, \phi)}{\partial \theta} \right\} &= 0 \\ \operatorname{var}_{\theta,\phi}\left\{ \frac{\partial l(\theta, \phi)}{\partial \theta} \right\} &= -E_{\theta,\phi}\left\{ \frac{\partial^2 l(\theta, \phi)}{\partial \theta^2} \right\} \end{aligned}$$

or

$$\begin{aligned} E_{\theta,\phi}\left\{ \frac{Y - b'(\theta)}{\phi / w} \right\} &= 0 \\ \operatorname{var}_{\theta,\phi}\left\{ \frac{Y - b'(\theta)}{\phi / w} \right\} &= \frac{b''(\theta)}{\phi / w} \end{aligned}$$

From which we obtain

$$E_{\theta,\phi}(Y) = b'(\theta) \tag{14a}$$

$$\operatorname{var}_{\theta,\phi}(Y) = b''(\theta)\frac{\phi}{w} \tag{14b}$$

From this we derive the following lemma.

Lemma 1. *The function b in* (8) *has the following properties*

(i) *b is strictly convex,*

(ii) *b' is strictly increasing,*

(iii) *b'' is strictly positive,*

unless $b''(\theta) = 0$ for all θ and the distribution of Y is concentrated at one point for all parameter values.

Proof. Just by ordinary calculus (iii) implies (ii) implies (i), so we need only prove (iii). Equation (14b) and the assumptions $\phi > 0$ and $w > 0$ imply $b''(\theta) \geq 0$. So the only thing left to prove is that if $b''(\theta^*) = 0$ for any one θ^*, then actually $b''(\theta) = 0$ for all θ. By (14b) $b''(\theta^*) = 0$ implies $\operatorname{var}_{\theta^*,\phi}(Y) = 0$, so the distribution of Y for the parameter values θ^* and ϕ is concentrated at one point. But now we apply a trick using the distribution at θ^* to calculate for other θ

$$\begin{aligned} f(y \mid \theta, \phi, w) &= \frac{f(y \mid \theta, \phi, w)}{f(y \mid \theta^*, \phi, w)} f(y \mid \theta^*, \phi, w) \\ &= \exp\left(\frac{y\theta - b(\theta)}{\phi/w_i} - \frac{y\theta^* - b(\theta^*)}{\phi/w_i}\right) f(y \mid \theta^*, \phi, w) \end{aligned}$$

The exponential term is strictly positive, so the only way the distribution of Y can be concentrated at one point and have variance zero for $\theta = \theta^*$ is if the distribution is concentrated at the same point and hence has variance zero for all other θ. And using (14b) again, this would imply $b''(\theta) = 0$ for all θ. □

The "unless" case in the lemma is uninteresting. We never use probability models for data having distributions concentrated at one point (that is, constant random variables). Thus (i), (ii), and (iii) of the lemma hold for any GLM we would actually want to use. The most important of these is (ii) for a reason that will be explained when we return to the general theory after the following example.

Example 2.1 (Binomial Regression).
We generalize Bernoulli regression just a bit by allowing more than one Bernoulli variable to go with each predictor value $\mathbf{x}_i$. Adding those Bernoullis gives a binomial response, that is, we assume

$$Y_i \sim \text{Binomial}(m_i, p_i)$$

where m_i is the number of Bernoulli variables with predictor vector $\mathbf{x}_i$. The density for Y_i is

$$f(y_i \mid p_i) = \binom{m_i}{y_i} p_i^{y_i} (1 - p_i)^{m_i - y_i}$$

we try to match this up with the GLM form. So first we write the density as an exponential

$$\begin{aligned} f(y_i \mid p_i) &= \exp\left[y_i \log(p_i) + (m_i - y_i) \log(1 - p_i) + \log \binom{m_i}{y_i} \right] \\ &= \exp\left[y_i \log\left(\frac{p_i}{1 - p_i} \right) + m_i \log(1 - p_i) + \log \binom{m_i}{y_i} \right] \\ &= \exp\left\{ m_i \bigl[\bar{y}_i \theta_i - b(\theta_i) \bigr] + \log \binom{m_i}{y_i} \right\} \end{aligned}$$

where we have defined

$$\begin{aligned} \bar{y}_i &= y_i / m_i \\ \theta_i &= \operatorname{logit}(p_i) \\ b(\theta_i) &= -\log(1 - p_i) \end{aligned}$$

So we see that

- The *canonical parameter* for the binomial model is $\theta = \operatorname{logit}(p)$. That explains why the logit link is popular.
- The *weight* w_i in the GLM density turns out to be the number of Bernoullis m_i associated with the i-th predictor value. So we see that the weight allows for grouped data like this.
- There is nothing like a dispersion parameter here. For the binomial family the dispersion is known; $\phi = 1$.

Returning to the general GLM model (a doubly redundant redundancy), we first define yet another parameter, the *mean value parameter*

$$\mu_i = E_{\theta_i, \phi}(Y_i) = b'(\theta_i).$$

By (ii) of Lemma 1 b' is a strictly increasing function, hence an invertible function. Thus the mapping between the canonical parameter θ and the mean value parameter μ is an invertible change of parameter. Then by definition of "link function" the relation between the mean value parameter μ_i and the linear predictor η_i is given by the link function

$$\eta_i = g(\mu_i).$$

The link function g is required to be a strictly increasing function, hence an invertible change of parameter.

If, as in logistic regression we take the linear predictor to be the canonical parameter, that determines the link function, because $\eta_i = \theta_i$ implies $g^{-1}(\theta) = b'(\theta)$. In general, as is the case in probit regression, the link function g and the function b' that connects the canonical and mean value parameters are unrelated.

It is traditional in GLM theory to make primary use of the mean value parameter and not use the canonical parameter (unless it happens to be the same as the linear predictor). For that reason we want to write the variance as a function of μ rather than θ

$$\operatorname{var}_{\theta_i,\phi}(Y_i) = \frac{\phi}{w} V(\mu_i) \tag{15}$$

where

$$V(\mu) = b''(\theta) \qquad \text{when} \qquad \mu = b'(\theta)$$

This definition of the function V makes sense because the function b' is an invertible mapping between mean value and canonical parameters. The function V is called the *variance function* even though it is only proportional to the variance, the complete variance being $\phi V(\mu)/w$.

2.1 Parameter Estimation

Now we can write out the log likelihood derivatives

$$\begin{aligned} \frac{\partial l(\boldsymbol{\beta})}{\partial \beta_j} &= \sum_{i=1}^{n} \left(\frac{y_i - b'(\theta_i)}{\phi / w_i} \right) \frac{\partial \theta_i}{\partial \beta_j} \\ &= \sum_{i=1}^{n} \left(\frac{y_i - \mu_i}{\phi / w_i} \right) \frac{\partial \theta_i}{\partial \beta_j} \end{aligned}$$

In order to completely eliminate θ_i we need to calculate the partial derivative. First note that

$$\frac{\partial \mu_i}{\partial \theta_i} = b''(\theta_i)$$

so by the inverse function theorem

$$\frac{\partial \theta_i}{\partial \mu_i} = \frac{1}{b''(\theta_i)} = \frac{1}{V(\mu_i)}$$

Now we can write

$$\frac{\partial \theta_i}{\partial \beta_j} = \frac{\partial \theta_i}{\partial \mu_i} \frac{\partial \mu_i}{\partial \eta_i} \frac{\partial \eta_i}{\partial \beta_j} = \frac{1}{V(\mu_i)} h'(\eta_i) x_{ij} \tag{16}$$

where $h = g^{-1}$ is the inverse link function. And we finally arrive at the likelihood equations expressed in terms of the mean value parameter and the linear predictor

$$\frac{\partial l(\boldsymbol{\beta})}{\partial \beta_j} = \frac{1}{\phi} \sum_{i=1}^{n} \left(\frac{y_i - \mu_i}{V(\mu_i)} \right) w_i h'(\eta_i) x_{ij}$$

These are the equations the computer sets equal to zero and solves to find the regression coefficients. Note that the dispersion parameter ϕ appears only multiplicatively. So it cancels when the partial derivatives are set equal to zero. Thus the regression coefficients can be estimated without estimating the dispersion (just as in linear regression).

Also as in linear regression, the dispersion parameter is not estimated by maximum likelihood but by the method of moments. By (15)

$$E\left\{\frac{w_i(Y_i - \mu_i)^2}{V(\mu_i)}\right\} = \frac{w_i}{V(\mu_i)}\operatorname{var}(Y_i) = \phi$$

Thus

$$\frac{1}{n}\sum_{i=1}^{n}\frac{w_i(y_i - \hat{\mu}_i)^2}{V(\hat{\mu}_i)}$$

would seem to be an approximately unbiased estimate of ϕ. Actually it is not because $\hat{\boldsymbol{\mu}}$ is not $\boldsymbol{\mu}$, and

$$\hat{\phi} = \frac{1}{n-p}\sum_{i=1}^{n}\frac{w_i(y_i - \hat{\mu}_i)^2}{V(\hat{\mu}_i)}$$

is closer to unbiased where p is the rank of the design matrix $\mathbf{X}$. We won't bother to prove this. The argument is analogous to the reason for $n-p$ in linear regression.

2.2 Fisher Information, Tests and Confidence Intervals

The log likelihood second derivatives are

$$\frac{\partial^2 l(\boldsymbol{\beta})}{\partial\beta_j\partial\beta_k} = \sum_{i=1}^{n}\left(\frac{y_i - b'(\theta_i)}{\phi/w_i}\right)\frac{\partial^2\theta_i}{\partial\beta_j\partial\beta_k} - \sum_{i=1}^{n}\left(\frac{b''(\theta_i)}{\phi/w_i}\right)\frac{\partial\theta_i}{\partial\beta_j}\frac{\partial\theta_i}{\partial\beta_k}$$

This is rather a mess, but because of (14a) the expectation of the first sum is zero. Thus the j, k term of the expected Fisher information is, using (16) and $b'' = V$,

$$\begin{aligned}
-E\left\{\frac{\partial^2 l(\boldsymbol{\beta})}{\partial\beta_j\partial\beta_k}\right\} &= \sum_{i=1}^{n}\left(\frac{b''(\theta_i)}{\phi/w_i}\right)\frac{\partial\theta_i}{\partial\beta_j}\frac{\partial\theta_i}{\partial\beta_k} \\
&= \sum_{i=1}^{n}\left(\frac{V(\mu_i)}{\phi/w_i}\right)\frac{1}{V(\mu_i)}h'(\eta_i)x_{ij}\frac{1}{V(\mu_i)}h'(\eta_i)x_{ik} \\
&= \frac{1}{\phi}\sum_{i=1}^{n}\left(\frac{w_i h'(\eta_i)^2}{V(\mu_i)}\right)x_{ij}x_{ik}
\end{aligned}$$

We can write this as a matrix equation if we define $\mathbf{D}$ to be the diagonal matrix with i, i element

$$d_{ii} = \frac{1}{\phi}\frac{w_i h'(\eta_i)^2}{V(\mu_i)}$$

Then

$$\mathbf{I}(\boldsymbol{\beta}) = \mathbf{X}'\mathbf{D}\mathbf{X}$$

is the expected Fisher information matrix. From this standard errors for the parameter estimates, confidence intervals, test statistics, and so forth can be derived using the usual likelihood theory. Fortunately, we do not have to do all of this by hand. R knows all the formulas and computes them for us.

3 Poisson Regression

The Poisson model is also a GLM. We assume responses

$$Y_i \sim \text{Poisson}(\mu_i)$$

and connection between the linear predictor and regression coefficients, as always, of the form (5). We only need to identify the link and variance functions to get going. It turns out that the canonical link function is the log function (left as an exercise for the reader). The Poisson distribution distribution has the relation

$$\text{var}(Y) = E(Y) = \mu$$

connecting the mean, variance, and mean value parameter. Thus the variance function is $V(\mu) = \mu$, the dispersion parameter is known ($\phi = 1$), and the weight is also unity ($w = 1$).

Example 3.1 (Poisson Regression).
The data set in the file `ex12.10.1.dat` is read by

```
> X <- read.table("ex12.10.1.dat", header = TRUE)
> names(X)

[1] "hour"  "count"

> attach(X)
```

simulates the hourly counts from a not necessarily homogeneous Poisson process. The variables are `hour` and `count`, the first counting hours sequentially throughout a 14-day period (running from 1 to $14 \times 24 = 336$) and the second giving the count for that hour.

The idea of the regression is to get a handle on the mean as a function of time if it is not constant. Many time series have a daily cycle. If we pool the counts for the same hour of the day over the 14 days of the series, we see a clear pattern in the histogram. The R `hist` function won't do this, but we can construct the histogram ourselves (Figure 3) using `barplot` with the commands

```
> hourofday <- (hour - 1)%%24 + 1
> foo <- split(count, hourofday)
> foo <- sapply(foo, sum)
> barplot(foo, space = 0, xlab = "hour of the day", ylab = "total count",
+     names = as.character(1:24), col = 0)
```

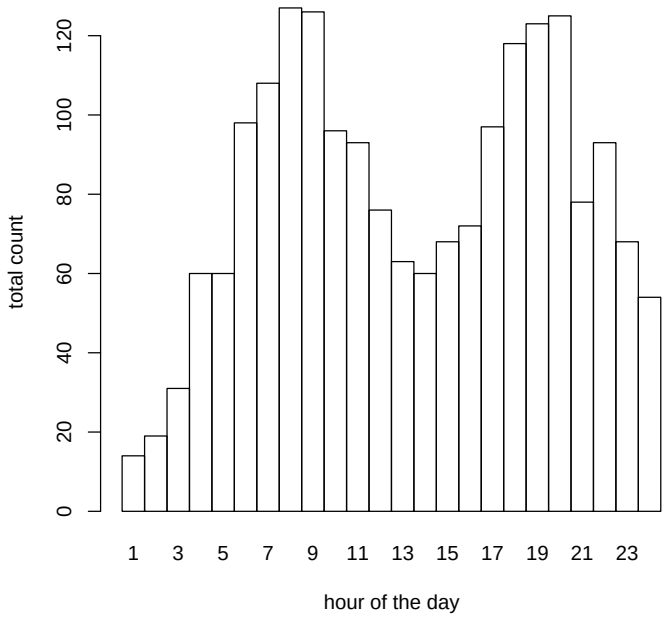

Figure 3: Histogram of the total count in each hour of the day for the data for Example 3.1.

In contrast, if we pool the counts for each day of the week, the histogram is fairly even (not shown). Thus it seems to make sense to model the mean function as being periodic with period 24 hours, and the obvious way to do that is to use trigonometric functions. Let us do a bunch of fits

```
> w <- hour/24 * 2 * pi
> out1 <- glm(count ~ I(sin(w)) + I(cos(w)), family = poisson)
> summary(out1)

Call:
glm(formula = count ~ I(sin(w)) + I(cos(w)), family = poisson)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-3.78327  -1.18758  -0.05076   0.86991   3.42492

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)  1.73272    0.02310   75.02  < 2e-16 ***
I(sin(w))   -0.10067    0.03237   -3.11  0.00187 **
I(cos(w))   -0.21360    0.03251   -6.57 5.04e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 704.27  on 335  degrees of freedom
Residual deviance: 651.10  on 333  degrees of freedom
AIC: 1783.2

Number of Fisher Scoring iterations: 5

> out2 <- update(out1, . ~ . + I(sin(2 * w)) + I(cos(2 * w)))
> summary(out2)

Call:
glm(formula = count ~ I(sin(w)) + I(cos(w)) + I(sin(2 * w)) +
    I(cos(2 * w)), family = poisson)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-3.20425  -0.74314  -0.09048   0.61291   3.26622

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    1.65917    0.02494  66.516  < 2e-16 ***
I(sin(w))     -0.13916    0.03128  -4.448 8.66e-06 ***
```

```
I(cos(w))     -0.28510    0.03661  -7.787 6.86e-15 ***
I(sin(2 * w)) -0.42974    0.03385 -12.696  < 2e-16 ***
I(cos(2 * w)) -0.30846    0.03346  -9.219  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 704.27  on 335  degrees of freedom
Residual deviance: 399.58  on 331  degrees of freedom
AIC: 1535.7

Number of Fisher Scoring iterations: 5
```

```
> out3 <- update(out2, . ~ . + I(sin(3 * w)) + I(cos(3 * w)))
> summary(out3)

Call:
glm(formula = count ~ I(sin(w)) + I(cos(w)) + I(sin(2 * w)) +
    I(cos(2 * w)) + I(sin(3 * w)) + I(cos(3 * w)), family = poisson)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-3.21035  -0.78122  -0.04986   0.48562   3.18471

Coefficients:
              Estimate Std. Error z value Pr(>|z|)
(Intercept)    1.655430   0.025152  65.818  < 2e-16 ***
I(sin(w))     -0.151196   0.032532  -4.648 3.36e-06 ***
I(cos(w))     -0.301336   0.038250  -7.878 3.32e-15 ***
I(sin(2 * w)) -0.439789   0.034464 -12.761  < 2e-16 ***
I(cos(2 * w)) -0.312843   0.033922  -9.222  < 2e-16 ***
I(sin(3 * w)) -0.063440   0.033805  -1.877   0.0606 .
I(cos(3 * w))  0.004311   0.033632   0.128   0.8980
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 704.27  on 335  degrees of freedom
Residual deviance: 396.03  on 329  degrees of freedom
AIC: 1536.1

Number of Fisher Scoring iterations: 5
```

It seems from the pattern of “stars” that maybe it is time to stop. A clearer indication is given by the so-called *analysis of deviance* table, “deviance” being

another name for the likelihood ratio test statistic (twice the log likelihood difference between big and small models), which has an asymptotic chi-square distribution by standard likelihood theory.

```
> anova(out1, out2, out3, test = "Chisq")

Analysis of Deviance Table

Model 1: count ~ I(sin(w)) + I(cos(w))
Model 2: count ~ I(sin(w)) + I(cos(w)) + I(sin(2 * w)) + I(cos(2 * w))
Model 3: count ~ I(sin(w)) + I(cos(w)) + I(sin(2 * w)) + I(cos(2 * w)) +
    I(sin(3 * w)) + I(cos(3 * w))
  Resid. Df Resid. Dev  Df Deviance P(>|Chi|)
1       333     651.10
2       331     399.58   2   251.52 2.412e-55
3       329     396.03   2     3.55      0.17
```

The approximate P-value for the likelihood ratio test comparing models 1 and 2 is $P \approx 0$, which clearly indicates that model 1 should be rejected. The approximate P-value for the likelihood ratio test comparing models 2 and 3 is $P = 0.17$, which fairly clearly indicates that model 2 should be accepted and that model 3 is unnecessary. $P = 0.17$ indicates exceedingly weak evidence favoring the larger model. Thus we choose model 2.

The following code

```
> plot(hourofday, count, xlab = "hour of the day")
> curve(predict(out2, data.frame(w = x/24 * 2 * pi), type = "response"),
+     add = TRUE)
```

draws the scatter plot and estimated regression function for model 2 (Figure 4).

I hope all readers are impressed by how magically statistics works in this example. A glance at Figure 4 shows

- Poisson regression is obviously doing more or less the right thing,
- there is no way one could put in a sensible regression function without using theoretical statistics. The situation is just too complicated.

4 Overdispersion

So far we have seen only models with unit dispersion parameter ($\phi = 1$). This section gives an example with $\phi \neq 1$ so we can see the point of the dispersion parameter.

The reason $\phi = 1$ for binomial regression is that the mean value parameter $p = \mu$ determines the variance $mp(1 - p) = m\mu(1 - \mu)$. Thus the variance function is

$$V(\mu) = \mu(1 - \mu) \tag{17}$$

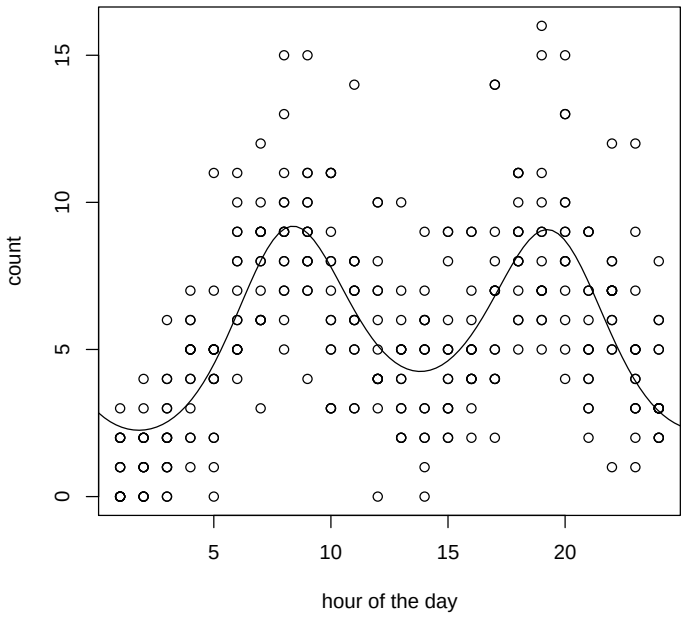

Figure 4: Scatter plot and regression curve for Example 3.1 (Poisson regression with log link function). The regression function is trigonometric on the scale of the linear predictor with terms up to the frequency 2 per day.

and the weights are $w_i = m_i$, the sample size for each binomial variable (this was worked out in detail in Example 2.1).

But what if the model is wrong? Here is another model. Suppose

$$Y_i \mid W_i \sim \text{Binomial}(m_i, W_i)$$

where the W_i are IID random variables with mean μ and variance τ^2. Then by the usual rules for conditional probability

$$E(Y_i) = E\{E(Y_i \mid W_i)\} = E(m_i W_i) = m_i \mu$$

and

$$\begin{aligned} \text{var}(Y_i) &= E\{\text{var}(Y_i \mid W_i)\} + \text{var}\{E(Y_i \mid W_i)\} \\ &= E\{m_i W_i (1 - W_i)\} + \text{var}\{m_i W_i\} \\ &= m_i \mu - m_i E(W_i^2) + m_i^2 \tau^2 \\ &= m_i \mu - m_i (\tau^2 + \mu^2) + m_i^2 \tau^2 \\ &= m_i \mu (1 - \mu) + m_i (m_i - 1) \tau^2 \end{aligned}$$

This is clearly larger than the formula $m_i \mu (1 - \mu)$ one would have for the binomial model. Since the variance is always larger than one would have under the binomial model.

So we know that if our response variables Y_i are the sum of a random mixture of Bernoullis rather than IID Bernoullis, we will have overdispersion. But how to model the overdispersion? The GLM model offers a simple solution. Allow for general ϕ so we have, defining $\overline{Y}_i = Y_i / m_i$

$$\begin{aligned} E(\overline{Y}_i) &= \mu_i \\ \text{var}(\overline{Y}_i) &= \frac{\phi}{m_i} \mu_i (1 - \mu_i) \\ &= \frac{\phi}{m_i} V(\mu_i) \end{aligned}$$

where V is the usual binomial variance function (17).

Example 4.1 (Overdispersed Binomial Regression).
The data set in the file `ex12.11.1.dat` is read by

```
> X <- read.table("ex12.11.1.dat", header = TRUE)
> names(X)

[1] "succ" "fail" "x"

> attach(X)
```

contains some data for an overdispersed binomial model. The commands

```
> y <- cbind(succ, fail)
> out.binom <- glm(y ~ x, family = binomial)
> summary(out.binom)

Call:
glm(formula = y ~ x, family = binomial)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.73992  -1.10103   0.02212   1.06517   2.24202

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -1.92156    0.35321   -5.44 5.32e-08 ***
x            0.07436    0.01229    6.05 1.45e-09 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 118.317  on 49  degrees of freedom
Residual deviance:  72.476  on 48  degrees of freedom
AIC: 135.79

Number of Fisher Scoring iterations: 4

> out.quasi <- glm(y ~ x, family = quasibinomial)
> summary(out.quasi)

Call:
glm(formula = y ~ x, family = quasibinomial)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.73992  -1.10103   0.02212   1.06517   2.24202

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) -1.92156    0.41713  -4.607 3.03e-05 ***
x            0.07436    0.01451   5.123 5.30e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for quasibinomial family taken to be 1.394695)

    Null deviance: 118.317  on 49  degrees of freedom
Residual deviance:  72.476  on 48  degrees of freedom
```

```
AIC: NA

Number of Fisher Scoring iterations: 4
```

fit both the binomial model (logit link and $\phi = 1$) and the "quasi-binomial" (logit link again but ϕ is estimated with the method of moments estimator as explained in the text). Both models have exactly the same maximum likelihood regression coefficients, but because the dispersions differ, the standard errors, z-values, and P-values differ.

Your humble author finds this a bit unsatisfactory. If the data are really overdispersed, then the standard errors and so forth from the latter output are the right ones to use. But since the dispersion was not estimated by maximum likelihood, there is no likelihood ratio test for comparing the two models. Nor could your author find any other test in a brief examination of the literature. Apparently, if one is worried about overdispersion, one should use the model that allows for it. And if not, not. But that's not the way we operate in the rest of statistics. I suppose I need to find out more about overdispersion (this was first written three years ago and I still haven't investigated this further).