Poisson IRWLS

Lieven Clement

1. Poisson model family

1.1 Structure of glm

$$\left\{\begin{array}{lcr} y_i & \sim & Poisson(\mu_i) \\ \log(\mu_i) & = & \eta_i \\ \eta_i & = & \mathbf{x}_i\beta \end{array}\right.$$

1.2 Poisson distribution

$$y_i \sim \frac{\mu_i^{y_i} e^{-\mu_i}}{y_i!}$$

1.2.1. In the form of the exponential family

$$y_i \sim \exp\left\{\frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + c(y_i,\phi)\right\}$$

$$y_i \sim \exp\left\{y_i \log(\mu_i) - \mu_i - \log(y_i!)\right\}$$

- Canonical model parameter $\theta_i = \log \mu_i$.
- $b(\theta_i) = \exp(\theta_i)$
- $c(y_i, \phi) = -\log(y_i!)$
- $\phi = 1$
- $a(\phi) = 1$
- $\mu_i = \frac{\partial b(\theta_i)}{\partial \theta_i} = \frac{\partial \exp(\theta_i)}{\partial \theta_i} = \exp(\theta_i)$
- $\text{Var}[y_i] = a(\phi)\frac{\partial^2 b(\theta_i)}{(\partial\theta_i)^2} = \frac{\partial^2 \exp(\theta_i)}{\partial \theta_i^2} = \exp(\theta_i)$.
- Mean is equal to variance for Poisson!

1.3 Poisson Log likelihood

For one observation:

$$l(\mu_i|y_i) = y_i \log \mu_i - \mu_i - \log y_i!$$

$$l(\mu_i|y_i) = y_i\theta_i - e^{\theta_i} - \log y_i!$$

- Note that $\theta_i = \eta_i$. The canonical parameter for the poisson equals the linear predictor!

$$\theta_i = \eta_i = \mathbf{x}_i^t\beta$$

Log-likelihood for all observations, given that they are independent:

$$l(\mu|\mathbf{y}) = \sum_{i=1}^{n}\left\{y_i\theta_i - e^{\theta_i} - \log y_i!\right\}$$

1.4 Poisson parameter estimation

Maximum likelihood: choose the parameters β so that the likelihood to observe the sample under the statistical model becomes maximum.

$$\text{argmax}_\beta l(\mu|\mathbf{y})$$

Maximization $\rightarrow$ set first derivative of likelihood to betas equal to zero. First derivative of likelihood is also called the Score function ($S(\theta)$):

$$S(\theta) = \frac{\partial l(\mu|\mathbf{y})}{\partial \beta} = 0$$

$$\begin{aligned}
S(\beta) &= \frac{\partial \sum_{i=1}^{n} \left\{ y_i\theta_i - e^{\theta_i} - \log y_i! \right\}}{\partial \beta} \\
&= \sum_{i=1}^{n} \frac{\partial \left\{ y_i\theta_i - e^{\theta_i} - \log y_i! \right\}}{\partial \beta} \\
&= \sum_{i=1}^{n} \frac{\partial \left\{ y_i\theta_i - e^{\theta_i} - \log y_i! \right\}}{\partial \theta_i} \frac{\partial \theta_i}{\partial \beta} \\
&= \sum_{i=1}^{n} \left\{ y_i - e^{\theta_i} \right\} \mathbf{x}_i^t \\
&= \sum_{i=1}^{n} \mathbf{x}_i \left\{ y_i - e^{\theta_i} \right\} \\
&= \mathbf{X}^T \left\{ \mathbf{Y} - \mu \right\}
\end{aligned}$$

Parameter estimator $\hat{\beta}$: Find $\hat{\beta}$ so that

$$\mathbf{X}^T \left\{ \mathbf{Y} - \mu \right\} = \mathbf{0}$$

Problem $\mu = \exp(\theta) = \exp(\eta) = \exp(\mathbf{X}\beta)$! Score equation is nonlinear in the model parameters! $\rightarrow$ Find roots of score equation by using Newton-Raphson method.

1.4.1. Newton-Raphson

1. Choose initial parameter estimate $\beta^k = \beta^1$
2. Calculate score $S(\beta)|_{\beta=\beta^k}$
3. Calculate derivative of the function for which you want to calculate the roots
4. Walk along first derivative until line (plane) of the derivative crosses zero
5. Update the betas β^{k+1}
6. Iterate from step 2 - 5 until convergence.

1.4.1.3. Derivative of score equation

$$
\begin{aligned}
\frac{\partial S(\beta)}{\partial \beta} &= \frac{\mathbf{X}^T \left\{\mathbf{Y} - \exp\left(\theta\right)\right\}}{\partial \beta} \\
&= -\mathbf{X}^T \begin{bmatrix} \frac{\partial \exp(\theta_1)}{\partial \theta_1} & 0 & \dots & 0 \\ 0 & \frac{\partial \exp(\theta_2)}{\partial \theta_2} & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \frac{\partial \exp(\theta_n)}{\partial \theta_n} \end{bmatrix} \frac{\partial \theta}{\partial \beta} \\
&= -\mathbf{X}^T \begin{bmatrix} \exp(\theta_1) & 0 & \dots & 0 \\ 0 & \exp(\theta_2) & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots \\ 0 & 0 & \dots & \exp(\theta_n) \end{bmatrix} \mathbf{X} \\
&= -\mathbf{X}^T \mathbf{W} \mathbf{X}
\end{aligned}
$$

1.4.1.3. Define line (plane) of derivative

- We know two points of the plane $(\beta^k, S(\beta^k))$ and $(\beta^{k+1}, 0)$
- We know the direction of the plane $S'(\beta) = \frac{\partial S(\beta)}{\partial \beta}$
- Equation of plane:

$$S(\beta) = \alpha_0 + S'|_{\beta^k} \beta$$

- Get β_{k+1}

$$
\begin{aligned}
\mathbf{0} &= \alpha_0 + S'|_{\beta^k} \beta^{k+1} \\
\beta^{k+1} &= -\left(S'|_{\beta^k}\right)^{-1} \alpha_0
\end{aligned}
$$

- Get α_0

$$
\begin{aligned}
S(\beta^k) &= \alpha_0 + S'|_{\beta^k} \beta^k \\
\alpha_0 &= -S'|_{\beta^k} \beta^k + S(\beta^k)
\end{aligned}
$$

- Get β_{k+1}

$$
\begin{aligned}
\beta^{k+1} &= \beta^k - \left(S'|_{\beta^k}\right)^{-1} S(\beta^k) \\
\beta^{k+1} &= \beta^k + \left(\mathbf{X}^T \mathbf{W} \mathbf{X}\right)^{-1} S(\beta^k)
\end{aligned}
$$

With $J(\beta) = I(\beta) = \mathbf{X}^T \mathbf{W} \mathbf{X}$ the Fisher information matrix. Because we use the canonical model parameters the observed Fisher information matrix equals the expected Fisher information matrix $J(\beta) = I(\beta)$. Hence, Newton-Raphson is equivalent to Fisher scoring

1.4.1.4 Iteratively Reweighted Least Squares for Poisson.

We can rewrite Fisher scoring in IRLS.

$$\begin{aligned}
\beta^{k+1} &= \beta^k + \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1} S(\beta^k) \\
\beta^{k+1} &= \beta^k + \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1} \mathbf{X}^T (\mathbf{Y} - \mu) \\
\beta^{k+1} &= \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1} \mathbf{X}^T\mathbf{W}\mathbf{X}\beta^k + \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1} \mathbf{X}^T\mathbf{W}\mathbf{W}^{-1} (\mathbf{Y} - \mu) \\
\beta^{k+1} &= \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1} \mathbf{X}^T\mathbf{W} \left[\mathbf{X}\beta^k + \mathbf{W}^{-1} (\mathbf{Y} - \mu)\right] \\
\beta^{k+1} &= \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1} \mathbf{X}^T\mathbf{W}\mathbf{z}
\end{aligned}$$

with $\mathbf{z} = \left[\mathbf{X}\beta^k + \mathbf{W}^{-1} (\mathbf{Y} - \mu)\right]$

So we can fit the model by performing iterative regressions of the pseudo data $\mathbf{z}$ on $\mathbf{X}$. In each iteration we will update $\mathbf{z}$, the weights $\mathbf{W}$ and the model parameters.

Variance-covariance matrix of the model parameters?

In the IRWLS algorithm, the data is weighted according to the variance of $\mathbf{Y}$. We correct for the fact that the data are heteroscedastic. Count data have a mean variance relation (e.g. in Poisson case $\mathrm{E}[Y] = \mathrm{var}[Y] = \mu$). The IRWLS also corrects for the scale parameter ϕ in $\mathbf{W}$. (Note that the scale parameter for Poisson is $\phi = 1$).

So IRWLS the variance-covariance matrix for the model parameter equals

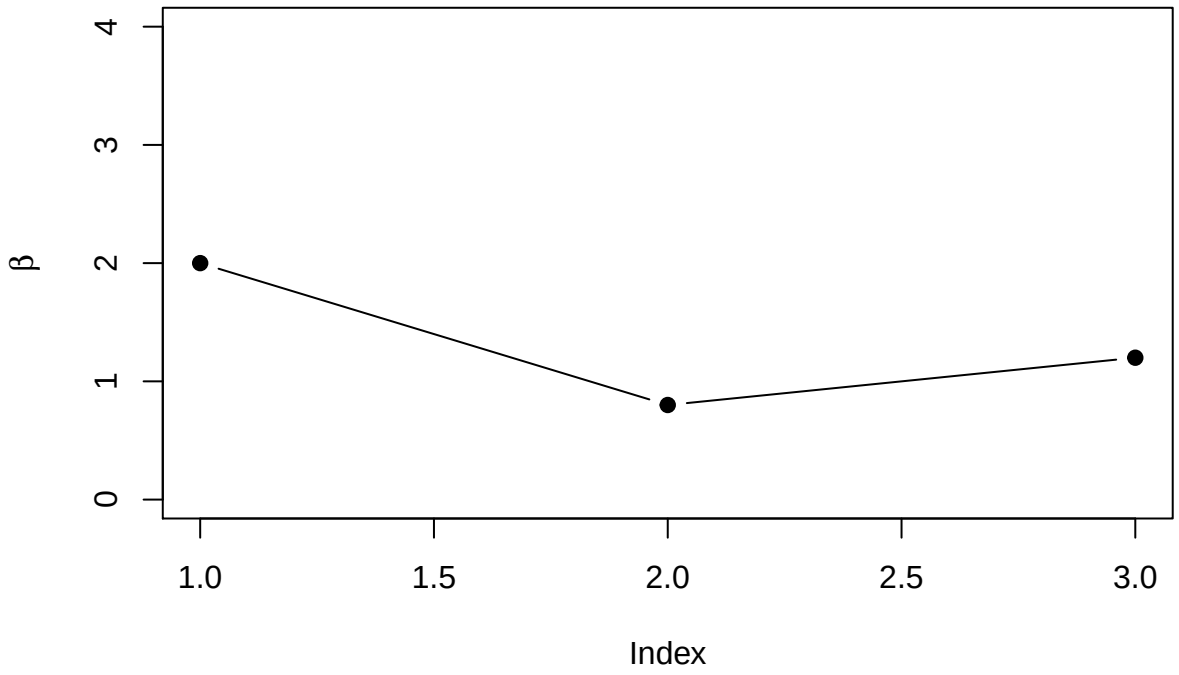
$$\mathbf{\Sigma}_{\hat{\beta}} = \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1}.$$

Note, that the Fisher Information Matrix (FIM) equals the inverse of the variance-covariance matrix of the experiment. The larger the FIM the more information we have on the experiment to estimate the model parameters. FIM ↑, precision ↑, SE ↓

2. Simulate poisson data

- We simulate data for 100 observations.
- Covariates x are simulated from normal distribution
- The β are chosen at $\beta_0 = 2$, $\beta_1 = 0.8$, $\beta_2 = 1.2$

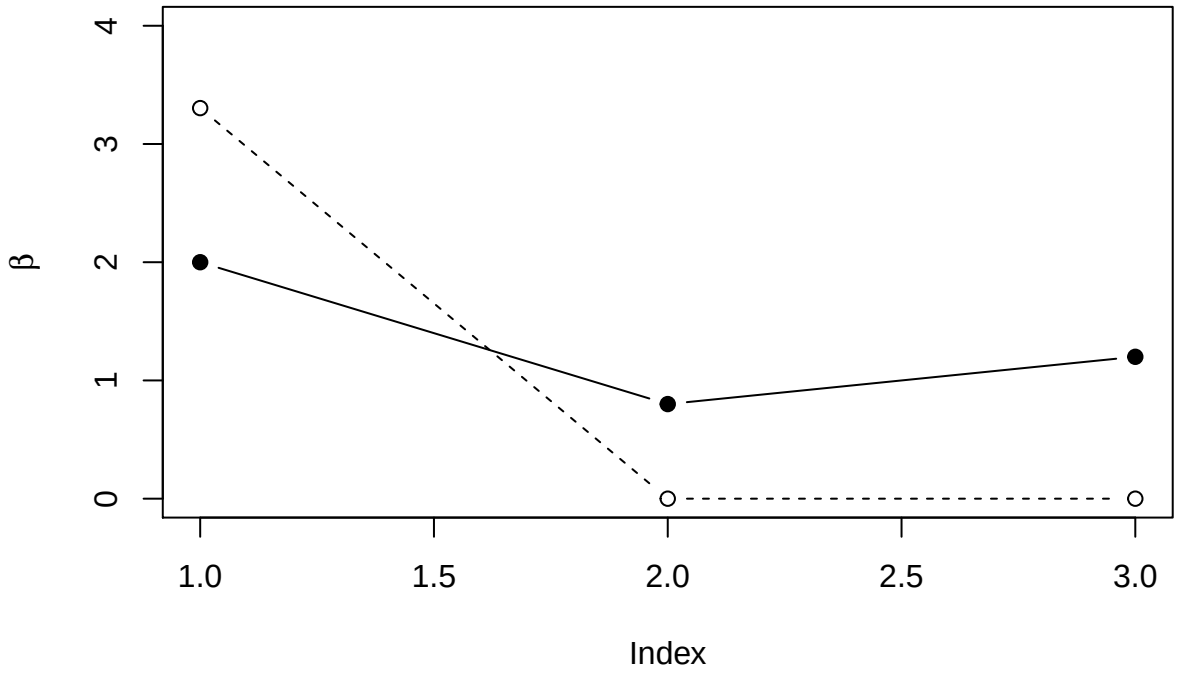
```
set.seed(300)
xhlp<-cbind(1,rnorm(100),rnorm(100))
betasTrue<-c(2,0.8,1.2)
etaTrue<-xhlp%*%betasTrue
y<-rpois(100,exp(etaTrue))
plot(betasTrue,ylab=expression(beta),ylim=c(0,4),pch=19,type="b")
```



3. Initial estimate

This is a very poor initial estimate used to illustrate the algorithm. Otherwise convergence for this simple example is way to quick

```
iteration=0
betas<-c(log(mean(y)),0,0)
plot(betasTrue,ylab=expression(beta),ylim=c(0,4),pch=19,type="b")
lines(betas,type="b",lty=2)
```



4. Iteratively reweighted least squares

4.1. Pseudo data

$$z_i = \eta_i + \frac{\partial \eta_i}{\partial \mu_i}(y_i - \mu_i)$$

$$z_i = \eta_i + e^{-\eta_i} y_i - 1$$

4.2 Weight matrix?

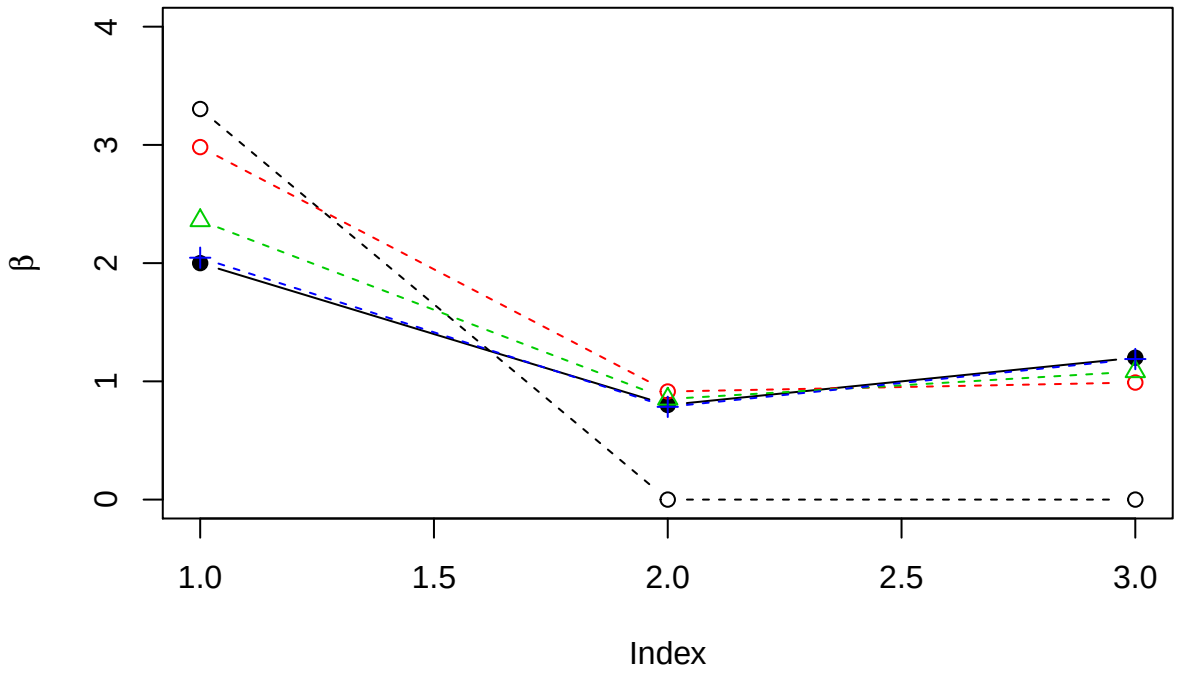
$$[w_{ii}] = var_{y_i}^{-1} \left(\frac{\partial \mu}{\partial \eta}\right)^2$$

$$[w_{ii}] = e^{\eta_i}$$

4.3 Run this update step multiple times

First 3 times (colors are black 0, red iteration 1, green iteration 2, blue iteration 3)

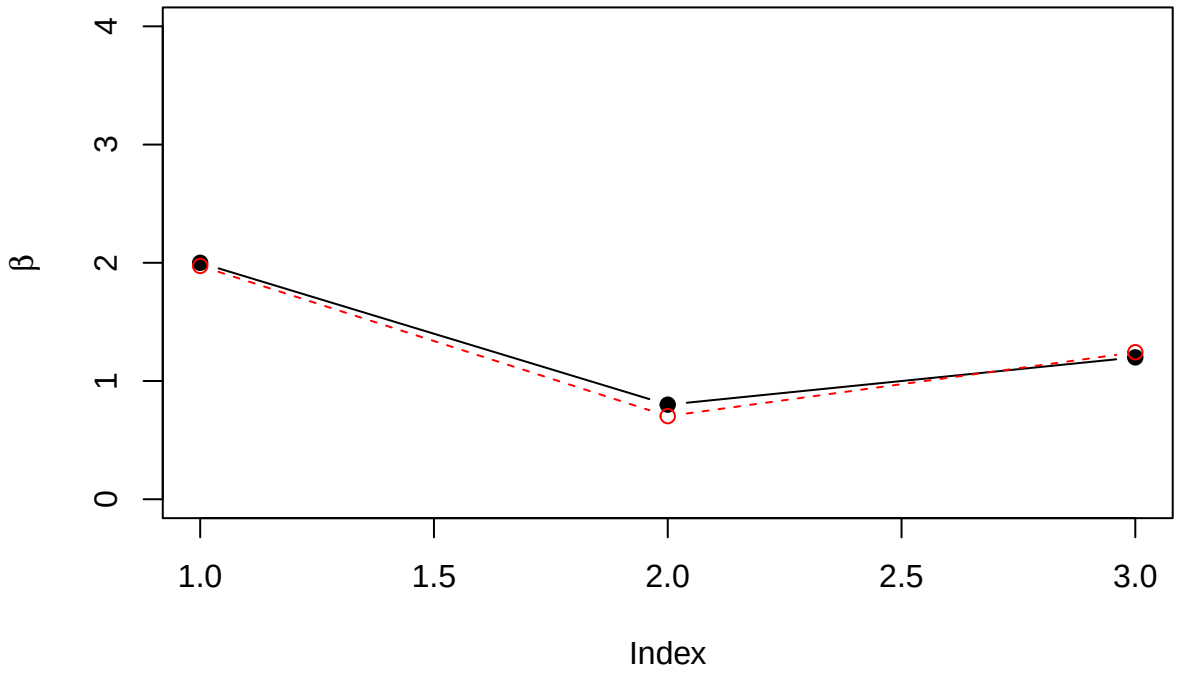
```
plot(betasTrue,ylab=expression(beta),ylim=c(0,4),pch=19,type="b")
lines(betas,type="b",lty=2)

#Calculate current eta
eta<-xhlp%*%betas

iteration=0
for (i in 1:3)
{
#start IRLS UPDATE STEP
iteration=iteration+1
#calculate pseudo data based on current betas
z=eta+exp(-eta)*(y-exp(eta))
#calculate new weights: diagonal elements
w<-c(exp(eta))

#update betas
lmUpdate<-lm(z~-1+xhlp,weight=w)
#eta<-xhlp%*%betas
eta<-lmUpdate$fitted
betas<-lmUpdate$coef
lines(betas,type="b",col=iteration+1,pch=iteration,lty=2)
}
```



5. Comparison with glm function

5.1 Smarter initialisation

```
z<-log(y+.5)
betas<-lm(z~-1+xhlp)$coef
plot(betasTrue,ylab=expression(beta),ylim=c(0,4),pch=19,type="b")
lines(betas,col=2,type="b",lty=2)
```



```
#calculate current eta
eta<-xhlp%*%betas
```

5.2. Evaluation Stopping Criterion

- Residual deviance: Is 2 log of LR between best possible fit and current fit

$$LR = \frac{L_{\text{best}}}{L_{\text{current}}}$$

$$D = 2(\log L_{\text{best}} - \log L_{\text{current}})$$

$$D = 2(l_{\text{best}} - l_{\text{current}})$$

- Best fit: $\mu = y$
- Optimal poisson:

$$l_{\text{best}} = \sum [y_i \log(y_i) - y_i - \log(y_i!)]$$

- Current fit

$$l_{\text{current}} = \sum [y_i \eta_i - e^{\eta_i} - log(y_i!)]$$

- Deviance D:

$$D = 2 \sum [y_i log(y_i) - y_i \eta_i - (y_i - e^{\eta_i})]$$

- Problem to calculate it if y=0 but by apply l'Hopital's rule we know

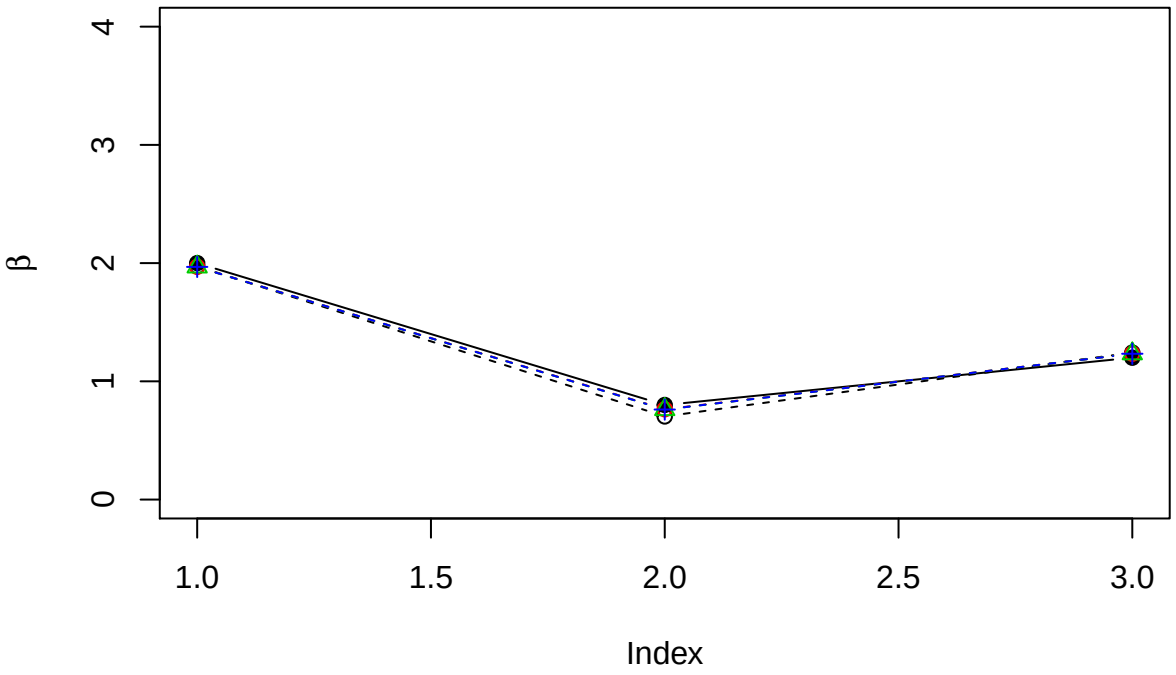
$$\lim_{y_i \to 0} y_i \log(y_i) = 0$$

```
ylogy<-function(y)
{
return(ifelse(y==0,rep(0,length(y)),y*log(y)))
}

deviance<-2*sum(ylogy(y)-y*eta-(y-exp(eta)))

devianceOld<-1e30
```

5.3 Run this update step multiple times until convergence

```
plot(betasTrue,ylab=expression(beta),ylim=c(0,4),pch=19,type="b")
lines(betas,type="b",lty=2)

tol<-1e-6
iteration=0
while(((devianceOld-deviance)/devianceOld)>tol)
{
#start IRLS UPDATE STEP
iteration=iteration+1
#calculate pseudo data based on current betas
z=eta+exp(-eta)*(y-exp(eta))
#calculate new weights: diagonal elements
w<-c(exp(eta))
```

```
#update betas
lmUpdate<-lm(z~-1+xhlp,weight=w)
#eta<-xhlp%*%betas
eta<-lmUpdate$fitted
betas<-lmUpdate$coef
lines(betas,type="b",col=iteration+1,pch=iteration,lty=2)

#criterion for convergence
devianceOld<-deviance
deviance<-2*sum(ylogy(y)-y*eta-(y-exp(eta)))
cat("iteration",iteration,"Deviance Old",devianceOld,"Deviance", deviance,"\n")
}
```




```
## iteration 1 Deviance Old 129.1127 Deviance 114.3748
## iteration 2 Deviance Old 114.3748 Deviance 114.3374
## iteration 3 Deviance Old 114.3374 Deviance 114.3374
```

5.4 Comparison with glm function in R

5.4.1. Variance β?

$$\Sigma_\beta = \left(\mathbf{X}^T\mathbf{W}\mathbf{X}\right)^{-1}$$

```
varBeta=solve(t(xhlp)%*%diag(w)%*%xhlp)
```

5.4.2. Fit GLM

Use -1 because intercept is already in xhlp

```
glmfit=glm(y~-1+xhlp,family=poisson)
comp=data.frame(glmfit=c(glmfit$deviance,glmfit$coef,summary(glmfit)$coef[,2]),ourFit=c(deviance,betas,
```

```r
row.names(comp)=c("deviance",paste("beta",1:3,sep=""),paste("se",1:3,sep=""))
comp
```

```
##                glmfit       ourFit
## deviance 114.33739500 114.33739500
## beta1      1.96805691   1.96805691
## beta2      0.76136641   0.76136641
## beta3      1.23330031   1.23330031
## se1        0.03814382   0.03814378
## se2        0.01891208   0.01891203
## se3        0.02556665   0.02556659
```