1 Fisher Scoring

The Fisher Scoring algorithm can be implemented using weighted least squares regression routines. Given a starting value for $\beta$ (or $\pi$), we construct the vector of weights $W$ and the working response $Z$, and then find $\hat{\beta}$ by regressing $Z$ on $X$ using weights $W$. We construct new weights and working responses $Z$, and then iterate until we obtain convergence.

For the logit link in binary regression

- $Z = \log(\pi/(1-\pi)) + (Y - \pi)/(\pi \times (1-\pi))$
- $W = \pi \times (1-\pi)$

Here is a no-frills function to perform $k$ iterations of Fisher Scoring under the logit link (equivalent to Newton Raphson because observed and expected information are the same). The input $X$ is assumed to contain a column of ones. No error checking or comments! (bad prof)

```r
Fisher.it <- function(Y, X, pi0, niter=1, print=F) {
  pi <- pi0
  for (i in 1:niter) {
    W <- pi *[1-pi]
    Z <- log(pi/(1-pi)) + (Y - pi)/(pi *[1-pi])
    lmobj <- lm(Z ~ X - 1, weights=W)
    beta <- lmobj$coef
    eta <- X %*% beta
    pi <- exp(eta)/(1 + exp(eta))
    if (print) {
      print(paste("Iteration ", as.character(i), ": Betahat"))
      print(beta)
    }
  }
  XWX <- t(lmobj$R) %*% lmobj$R
  return(beta, XWX, pi, W)
}
```

Start with initial probabilities $pi0 = 0.75$ if $Y=1$, $0.25$ if $Y=0$.

After one iteration:

```r
> out <- Fisher.it(oris$failure, X, pi0, 1)

> out$beta
  X1       X2
9.422777 -0.1492647
```
Run for 4 iterations:

```r
> out = Fisher.it(orings$failure, X, pi0, 4, print=T)
```

```
[1] "Iteration  1 : Betahat"
  X1   X2
9.422777 -0.1492647

[1] "Iteration  2 : Betahat"
  X1   X2
10.76226 -0.1696113

[1] "Iteration  3 : Betahat"
  X1   X2
10.87462 -0.1713095

[1] "Iteration  4 : Betahat"
  X1   X2
10.87535 -0.1713205

> # Asymptotic Covariance Matrix
> solve(out$XWX)
```

```
X1   X2
X1 32.5232111 -0.473880521
X2 -0.4738805  0.006962009
```

```
# Standard errors
> sqrt(diag(solve(out$XWX)))
[1] 5.70291251 0.08343865
```

```
# correlation between betahat0 and betahat1
> -.473880521/( 5.70291251*0.08343865)
[1] -0.9958751
```

#Splus:

```
> summary(oring.logit)
```

```
Coefficients:
    Value Std. Error t value
(Intercept) 10.8753321  5.69793801  1.908643
    temp  -0.1713202  0.08336339 -2.055102

Number of Fisher Scoring Iterations: 4

Correlation of Coefficients:
    (Intercept)
    temp -0.9958713
```

Pretty close! difference in SE(beta0)?
2 Residuals and Diagnostics

- ordinary residuals
  \[(y_i - \mu_i)\]

- Person residuals
  \[((y_i - \mu_i)/\sqrt{\text{Var}(y_i)})\]

- Deviance residuals (sum squared deviance residuals = deviance)
  \[\text{sign}(\hat{\eta}_i - \tilde{\eta}_i)\sqrt{2I(\hat{\eta}_i, y_i) - 2I(\tilde{\eta}_i, y_i)}\]

- Cook’s Distance
  \[(\hat{\beta}(i) - \tilde{\beta})'X'WX(\hat{\beta}(i) - \tilde{\beta})/(p) \approx (1/p)(e^F)^2 h_{ii}/(1 - h_{ii})\]

Studentized forms of the Pearson and Deviance residuals are defined as \(e^P\) and \(e^D\) by taking the above and dividing by the leverage

\[h_{ii} = \text{diag}(H) \quad H = W^{1/2}X(X'WX)^{-1}X'W^{1/2}\]

In Splus, the `residuals` function will return each of the un-standardized residuals. Cook’s Distance can be approximated using the linear model results applied at the last step of the Fisher Scoring algorithm. The function below uses the `lm.influence` function, which provides estimates of \(\beta\) with single case deletions, and the leverage values.

```r
> CooksD _ function(fit)
{
  fit.s <- summary.glm(fit)
  fit.infl <- lm.influence(fit)
  R <- fit$R
  XWX <- t(R) %*% R
  D <- rep(0, length(fit$residuals))
  Dcoef <- scale(fit.infl$coefficients, center = fit$coefficients, scale = F)
  disper <- fit.s$dispersion  #estimate of stdev^2 for Gaussian
  p <- sum(fit.infl$hat)
  for(i in 1:length(D)) {
    D[i] <- (t(Dcoef[i, ]) %*% XWX %*% Dcoef[i, ])/(p * disper)
  }
  D
}

Construct diagnostics and plots:

```r
> h _ lm.influence(oring.logit)$hat
> pearson.residual _ residuals(oring.logit, type="pearson")/sqrt(1-h)
> dev.residual _ residuals(oring.logit, type="deviance")/sqrt(1-h)
> coosd _ CooksD(oring.logit)
```

```r
> par(mfrow=c(2,2))
> index _ 1:24
> plot(index, h)
> qqnorm(pearson.residual)
> plot(index, coosd)
> qqnorm(dev.residual)
```
Diagnostics for the oring data using the logit link.
Residual plots for the oring data using the logit link.