Quiz 3

GLM Theory

```r
library(tidyverse)
```

```
-- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
v dplyr     1.1.4     v readr     2.1.5
v forcats   1.0.0     v stringr   1.5.1
v ggplot2   3.5.1     v tibble    3.2.1
v lubridate 1.9.3     v tidyr     1.3.1
v purrr     1.0.2
-- Conflicts ------------------------------------------ tidyverse_conflicts() --
x dplyr::filter() masks stats::filter()
x dplyr::lag()    masks stats::lag()
i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to becom
```

Question 1

Is it acceptable to use the quasi-poisson model when the data actually follow the Poisson distribution? Briefly explain why or why not.

It is not acceptable to use a quasi-poisson model when the data actually follows a Poisson distribution. In fact, the quasi-poisson exists for the case when the data doesn't satisfy assumptions of Poisson data, specifically the assumption that the mean and variance are equal and quasi-poisson will inflate the standard errors to handle variance greater than we would expect in Poisson data. If our data follows a Poisson distribution, then the mean=variance assumption should be satisfied, and quasi-poisson just takes away from the theoretical strength of our analysis.

Question 2

The support of the distribution is the integers where $y > 0$ or equivalently $[1, 2, ..., \infty)$.

Question 3

We start off with the PMF of the truncated Poisson, which is

$$P(Y=y)=\frac{e^{-\lambda}\lambda^{y}}{(1-e^{-\lambda})y!}$$

Note that our support is not dependent on an unknown parameter, meaning we can attempt to write this in exponential family form to see if the truncated Poisson is a member of the exponential family.

We will exponentiate the logarithm of the PMF.

$$exp(log\frac{e^{-\lambda}\lambda^{y}}{(1-e^{-\lambda})y!})$$

$$exp(log(e^{-\lambda})+log(\lambda^{y})-log(1-e^{-\lambda})-log(y!))$$

$$\frac{1}{y!}exp(log(\frac{e^{-\lambda}}{1-e^{-\lambda}})+ylog(\lambda))$$

We notice that this belongs to the exponential family.

The exponential family in canonical form can be written as:

$$f(y\mid\lambda)=h(y)exp(\eta(\lambda)T(y)-\psi(\lambda))$$

where

$$h(y)=\frac{1}{y!}$$

$$\psi(\lambda)=-log(\frac{e^{-\lambda}}{1-e^{-\lambda}})$$

$$\eta=log(\lambda)$$
$$T(y)=y$$

Question 4

Recall that the log partition function can be derived from the η and ψ components of our exponential family structure, specifically reparameterizing ψ in terms of η.

We found that $\eta = log(\lambda)$ or equivalently that $e^{\eta} = \lambda$.

From $\psi = -log(\frac{e^{-\lambda}}{1-e^{-\lambda}})$, we write the log partition function:

$$A = -log(\frac{e^{-e^{\eta}}}{1-e^{-e^{\eta}}})$$

By logarithm rules, we can say

$$A = e^{\eta} + log(1 - e^{-e^{\eta}})$$

From the log partition function, we can weaponize the following fact to find $E(Y)$

$$E(Y) = \frac{\partial A}{\partial \eta}$$

$$E(Y) = e^{\eta} + \frac{e^{-e^{\eta}} e^{\eta}}{1 - e^{-e^{\eta}}}$$

This is equivalent to

$$E(Y) = \frac{e^{\eta}}{1 - e^{-e^{\eta}}}$$

Since $e^{\eta} = \lambda$,

we can say

$$E(Y) = \frac{\lambda}{1 - e^{-\lambda}} = \frac{\lambda e^{\lambda}}{e^{\lambda} - 1}$$

Question 5

The canonical link can be viewed by η which connects our response to the predictors. Since $\eta = log(\lambda)$, $X\beta = log(\lambda)$ is the link function.

Question 6

To find the log likelihood of the truncated Poisson, I will first write out the likelihood by taking the product of PMF from n=1, ..., n.

We define the PMF of the truncated Poisson as

$$f_Y = \frac{e^{-\lambda}\lambda^y}{(1-e^{-\lambda})y!}$$

The likelihood is then

$$L(\lambda) = \prod_{i=1}^{n} \frac{e^{-\lambda}\lambda^{y_i}}{(1-e^{-\lambda})y_i!}$$

$$L(\lambda) = \frac{e^{-n\lambda}\lambda^{\sum_{i=1}^{n} y_i}}{(1-e^{-\lambda})^n \prod_{i=1}^{n} y_i!}$$

In attempt to find the log likelihood, we will take the logarithm.

$$log\ L(\lambda) = log(\frac{e^{-n\lambda}\lambda^{\sum_{i=1}^{n} y_i}}{(1-e^{-\lambda})^n \prod_{i=1}^{n} y_i!})$$

By logarithm properties

$$log\ L(\lambda) = -n\lambda + \sum_{i=1}^{n} y_i log\lambda - nlog(1-e^{-\lambda}) - \sum_{i=1}^{n} log(y_i!)$$

Using our link $log(\lambda) = X\beta$, we know $\lambda = e^{X\beta}$, so we can equivalently write our log likelihood as

$$logL(\beta) = -ne^{X\beta} + \sum_{i=1}^{n} y_i X\beta - nlog(1-e^{-e^{X\beta}}) - \sum_{i=1}^{n} log(y_i!)$$

This is equivalently

$$= \sum_{i=1}^{n}[-e^{X\beta} + y_i X\beta - log(1-e^{-e^{X\beta}}) - log(y_i!)]$$

The vectorized form can be written as

$$logL(\beta) = \sum_{i=1}^{n}[-e^{X_i^T\beta} + y_i X i^T\beta - log(1-e^{-e^{X_i^T\beta}}) - log(y_i!)]$$

This matches the stated log likelihood.

Question 7

To find the score function, I will take the partial derivative of the log likelihood with respect to β.

$$S(\beta) = \frac{\partial\ logL(\beta)}{\partial\beta} = \sum_{i=1}^{n}[-e^{X_i^T\beta}X_i + y_iX_i - \frac{-e^{-e^{X_i^T\beta}} \times -e^{X_i^T\beta}X_i}{1-e^{-e^{X_i^T\beta}}}]$$

$$S(\beta) = \frac{\partial\ logL(\beta)}{\partial\beta} = \sum_{i=1}^{n}[-e^{X_i^T\beta}X_i + y_iX_i - \frac{e^{-e^{X_i^T\beta}} \times e^{X_i^T\beta}X_i}{1-e^{-e^{X_i^T\beta}}}]$$

$$S(\beta) = \frac{\partial\ logL(\beta)}{\partial\beta} = \sum_{i=1}^{n}[y_iX_i - \frac{e^{-e^{X_i^T\beta}} \times e^{X_i^T\beta}X_i + e^{X_i^T}X_i(1-e^{-e^{X_i^T\beta}})}{1-e^{-e^{X_i^T\beta}}}]$$

This allows us to simplify our score function to

$$S(\beta) = \sum_{i=1}^{n}[y_iX_i - \frac{e^{X_i^T\beta}}{1-e^{-e^{X_i^T\beta}}}X_i]$$

The score function in matrix form is:

$$S(\beta) = X^T(y - \frac{e^{X\beta}}{1-e^{-e^{X\beta}}})$$

The hessian function is found by taking the second partial of the log likelihood with respect to β. This is also found by taking the partial derivative of the score function with respect to β

$$\begin{aligned} H(\beta) = \frac{\partial}{\partial\beta}S(\beta) &= \frac{\partial}{\partial\beta}[\sum_{i=1}^{n}[y_iX_i - \frac{e^{X_i^T\beta}}{1-e^{-e^{X_i^T\beta}}}X_i]] \\ &= -\sum_{i=1}^{n}\frac{\partial}{\partial\beta}(\frac{e^{X_i^T\beta}}{1-e^{-e^{X_i^T\beta}}}X_i) \end{aligned}$$

We can use quotient rule to find this derivative

$$\frac{\partial}{\partial\beta}(\frac{e^{X_i^T\beta}}{1-e^{-e^{X_i^T\beta}}}X_i) = \frac{(1-e^{-e^{X_i^T\beta}})e^{X_i^T\beta}X_iX_i^T - e^{X_i^T}e^{-e^{X_i^T\beta}}e^{X_i^T\beta}X_i}{(1-e^{-e^{X_i^T\beta}})^2}$$

We use this to write the Hessian

$$H(\beta) = -\sum_{i=1}^{n} \frac{(e^{X_i^T\beta} - e^{-e^{X_i^T\beta}} e^{X_i^T\beta} - e^{2X_i^T\beta} e^{-e^{X_i^T\beta}})}{(1 - e^{-e^{X_i^T\beta}})^2} X_i X_i^T$$

In matrix form, we can write the Hessian as:

$$H(\beta) = -X^T W X$$

where $W = \frac{e^{X\beta} - e^{-e^{X\beta}} e^{X\beta} - e^{2X\beta} e^{-e^{X\beta}}}{(1 - e^{-e^{X\beta}})^2}$

Question 8

For simplicity, I use $m = exp(X\beta)$ and $g = \frac{m}{(1-exp(-m))}$

```
trunc_pois_Newton_Raphson <- function(x, y, b_init = rep(1, ncol(x)),
                                     tol = 1*10^(-8))
{

  change <- Inf
  b_old <- b_init

  while(change>tol)
  {
    eta <- x %*% b_old
    em <- exp(eta)
    w <- diag(as.vector((em-(em * exp(-em))-(em^2)*exp(-em))/(1-exp(-em)^2)),
              nrow = nrow(x),
              ncol = nrow(x))
    g <- em/(1-exp(-em))

    b_new <- b_old + solve(t(x)%*% w %*% x) %*% t(x) %*% (y-g)

    change <- sqrt(sum((b_new - b_old)^2))

    b_old <- b_new
  }

  b_new
}
```

I will test my algorithm using the college dataset we used on a previous assignment.

```
college <- read_csv("data/college-data.csv")
```

```
Rows: 593 Columns: 15
-- Column specification --------------------------------------------------------
Delimiter: ","
chr  (5): name, state, state_code, type, degree_length
dbl (10): room_and_board, in_state_tuition, in_state_total, out_of_state_tui...

i Use `spec()` to retrieve the full column specification for this data.
i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
x <- cbind(rep(1, nrow(college)),
           as.numeric(college$out_of_state_tuition)/2000,
           as.factor(college$type))

x <- as.matrix(x)


college <- college |>
  na.omit() |>
  mutate(total = round(out_of_state_total/1000))


y <- as.vector(college$total)
```

```r
trunc_pois_Newton_Raphson(x, y)
```

```
             [,1]
[1,]  3.749873567
[2,]  0.001541439
[3,] -0.015000046
```

Note that this result appears relatively similar to the result for the zero truncated glm function I found online. Citation (https://stats.oarc.ucla.edu/r/dae/zero-truncated-poisson/.)

Cross Check with Zero Truncated GLM

```r
library(VGAM)
```

```
Loading required package: stats4

Loading required package: splines
```

```r
college <- college |>
  mutate(
    out_tuition = out_of_state_tuition/2000
  )

model <- vglm(total ~ out_tuition + type, family = pospoisson(), data = college)

summary(model)
```

```
Call:
vglm(formula = total ~ out_tuition + type, family = pospoisson(),
    data = college)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) 2.851016   0.027530 103.560   <2e-16 ***
out_tuition 0.052979   0.001289  41.096   <2e-16 ***
typePublic  0.002543   0.018548   0.137    0.891
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Name of linear predictor: loglink(lambda)

Log-likelihood: -1580.193 on 545 degrees of freedom

Number of Fisher scoring iterations: 4

No Hauck-Donner effect found in any of the estimates
```

Question 9

By default, glm() calls iteratively re-weighted least squares.

Question 10

No, the pareto 1 distribution is not in the exponential family.