Statistical Computing 1, Stat 590 Fall 2015 | Homework 11 | Name: Answer Key Prof. Erik B. Erhardt

Part I. (120 points) Do all calculations in LaTeX + R + knitr. Insert computer text output and graphics to support what you are saying. For this assignment, all R code should well commented and be visible (`echo=TRUE`) in the document where you have written it.

(120[pts]) **1. Exponential regression**

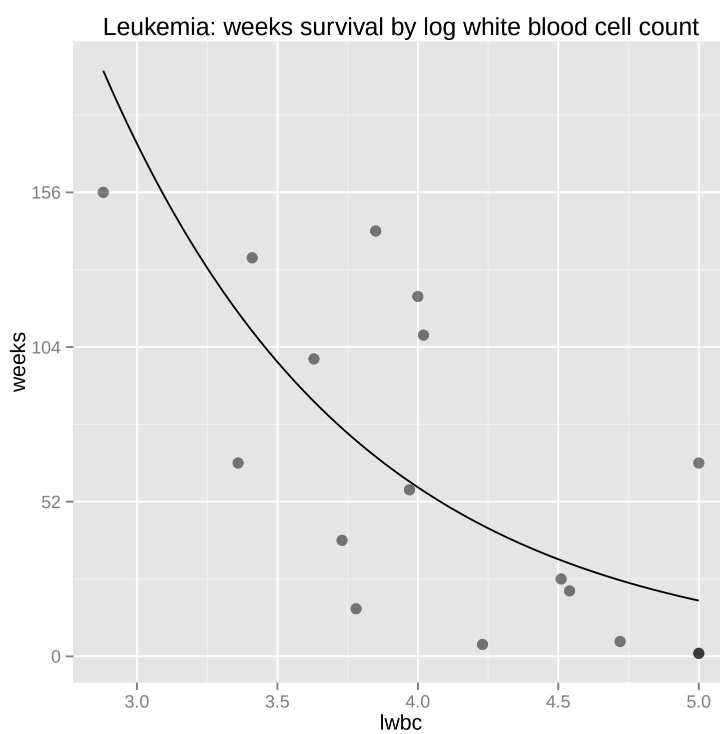
The data below from Feigl and Zelen (1965) are time to death, Y (weeks), in weeks from diagnois and $\log_{10}$(wbc) (the initial white blood cell count), x (lwbc), for 17 patients suffering from leukemia. The relation between Y and x is the main aspect of interest.

```
eleuk <- read.table("http://statacumen.com/teach/SC1/SC1_HW11_Exp-leuk.dat", header = TRUE)
```

	lwbc	weeks
1	3.36	65
2	2.88	156
3	3.63	100
4	3.41	134
5	3.78	16
6	4.02	108
7	4.00	121
8	4.23	4
9	3.73	39
10	3.85	143
11	3.97	56
12	4.51	26
13	4.54	22
14	5.00	1
15	5.00	1
16	4.72	5
17	5.00	65

Plot: Horizontal lines at 52-week (year) intervals.

```
library(ggplot2)
p <- ggplot(eleuk, aes(x = lwbc, y = weeks))
p <- p + scale_y_continuous(breaks = 52*c(0,1,2,3))
p <- p + stat_function(fun = f.exp.pred, args = list(out.FS$beta.MLE) )
p <- p + geom_point(alpha = 0.5, size = 3)
p <- p + labs(title = "Leukemia: weeks survival by log white blood cell count")
print(p)
```



A plot shows substantial random variation together with a tendency for Y to decrease with increasing x. Elaborate model fitting would be unnecessary for the analysis of these data in isolation. In particular, many different parametric representations of the systematic variations are consistent with the data.

We use the data to illustrate two main points. One is the use of general considerations to choose between alternative parametric regression relations. The other is the examination of the form of the random variation about a relation.

Three main aspects are involved in setting up a parametric description:

1. a specification of the form of systematic variation (mean), for example by giving the relation between the expected value $\mathrm{E}[Y_i]$ and x_i for the ith individual,
2. a specification of the general form of the random variation (residuals), and
3. a specification of the way in which systematic and random variation "combine", that is, by multiplication or by addition.

It is preferable for (1) to choose where possible a relation that gives positive values for $\mathrm{E}[Y_i]$ for all possible parameter values and values of x. From this point of view, the relation

$$\mathrm{E}[Y_i] = \beta_0 \exp\{\beta_1(x_i - \bar{x})\},$$

where $\bar{x} = \sum x_i/n$, is preferable to, for example, a linear relation

$$\mathrm{E}[Y_i] = \gamma_0 + \gamma_1(x_i - \bar{x}).$$

If the random contribution of the ith observation is ε_i, simplicity of interpretation and fitting, and inspection of the data, suggesting multiplicative combination; that is, we consider an interpretation in which the proportional variation around the mean has the same form for all x. It is then of interest to compare that distribution with the exponential distribution, partly because that is about the simplest very dispersed distribution for a positive quantity, partly because of the interpretation of the exponential distribution in terms of the properties of the completely random process, the Poisson process, partly because use of the exponential distribution much simplifies more detailed analysis, and consistency with exponential form is of great interest.
These considerations lead to the model

$$\mathrm{E}[Y_i] = \beta_0 \exp\{\beta_1(x_i - \bar{x})\}\varepsilon_i, \tag{1}$$

where ε_i is a random term of unit mean and, conceivably, exponentially distributed.
Suppose Y is a positive random variable with density

$$f(y|\mu) = \frac{1}{\mu}\exp\{-y/\mu\}, \quad y \geq 0,$$

then Y is an Exponential random variable with $\mathrm{E}[Y] = \mu$ and $\mathrm{Var}[Y_i] = \mu^2$. We write $Y \sim \mathrm{Exponential}(\mu)$.
Note that Equation (1) is equivalent to $Y_i \sim \mathrm{Exponential}(\mu_i)$, where

$$\begin{aligned} \mu_i &= \beta_0 \exp\{\beta_1(x_i - \bar{x})\}, \quad \text{or} \\ \log(\mu_i) &= \log(\beta_0) + \beta_1(x_i - \bar{x}). \end{aligned}$$

We will fit this model, but reparameterize the intercept so that

$$\log(\mu_i) = \beta_0 + \beta_1(x_i - \bar{x}), \tag{2}$$

that is, our intercept is the log of the original intercept.
Assume the data are a random sample $Y_1, Y_2, \ldots, Y_n$ independent with $Y_i \sim \mathrm{Exponential}(\mu_i)$.

(a) (10 pts) Show that the log-likelihood function is

$$\ell(\underset{\sim}{\mu}|\underset{\sim}{y}) \;=\; -\sum_{i=1}^{n}\left(\frac{y_i}{\mu_i} + \log(\mu_i)\right).$$

For later reference, note that

$$\frac{\partial \ell}{\partial \beta_j} \;=\; \frac{y_i}{\mu_i^2} - \frac{1}{\mu_i}.$$

Solution:

$$\begin{aligned}
L(\underset{\sim}{\mu}|\underset{\sim}{y}) &= \prod_{i=1}^{n}\frac{1}{\mu_i}\exp\left\{-\frac{y_i}{\mu_i}\right\} \\
&= \frac{1}{\prod_{i=1}^{n}\mu_i}\exp\left\{-\sum_{i=1}^{n}\frac{y_i}{\mu_i}\right\} \\
\ell(\underset{\sim}{\mu}|\underset{\sim}{y}) &= -\sum_{i=1}^{n}\log(\mu_i) - \sum_{i=1}^{n}\frac{y_i}{\mu_i} \\
&= -\sum_{i=1}^{n}\left(\log(\mu_i) + \frac{y_i}{\mu_i}\right) \\
&= -\sum_{i=1}^{n}\left(\log(\exp\{\beta_0+\beta_1(x_i-\bar{x})\}) + \frac{y_i}{\exp\{\beta_0+\beta_1(x_i-\bar{x})\}}\right) \\
&= -\sum_{i=1}^{n}\left(\beta_0+\beta_1(x_i-\bar{x}) + \frac{y_i}{\exp\{\beta_0+\beta_1(x_i-\bar{x})\}}\right)
\end{aligned}$$

(b) (20 pts) In a standard exponential regression problem we assume that $\log(\mu_i)$ is linearly related to a vector $\underset{\sim}{x}_i^\top = [x_{i1}, x_{i2}, \ldots, x_{ip}]$ of known covariates and an unknown vector $\underset{\sim}{\beta} = [\beta_1, \beta_2, \ldots, \beta_p]^\top$ of regression parameters via

$$\underset{\sim}{\mu}_i \;\equiv\; \underset{\sim}{\mu}_i(\underset{\sim}{\beta}) \;=\; \exp\{\underset{\sim}{x}_i^\top\underset{\sim}{\beta}\}$$

or

$$\log(\underset{\sim}{\mu}_i) \;\equiv\; \log(\underset{\sim}{\mu}_i(\underset{\sim}{\beta})) \;=\; \underset{\sim}{x}_i^\top\underset{\sim}{\beta} \;=\; \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip}.$$

Typically, $x_{i1} = 1$ and if so β_1 is an "intercept effect".
A goal here is to derive an NR/Fisher Scoring iteration procedure to compute the MLE of $\underset{\sim}{\beta}$. To this end, derive the derivatives of the mean vector and log-likelihood and derive the expected information:

1. first partial of mean vector

$$\frac{\partial \mu_i(\underset{\sim}{\beta})}{\partial \beta_j} \;=\; x_{ij}\mu_i(\underset{\sim}{\beta})$$

2. first partial of log-likelihood

$$\frac{\partial \ell(\mu(\utilde{\beta})|\utilde{y})}{\partial \beta_j} = \sum_{i=1}^{n} x_{ij} \left(\frac{y_i}{\mu_i(\utilde{\beta})} - 1 \right)$$

3. second partial of log-likelihood

$$\frac{\partial^2 \ell(\mu(\utilde{\beta})|\utilde{y})}{\partial \beta_j \partial \beta_k} = -\sum_{i=1}^{n} x_{ij} x_{ik} \frac{y_i}{\mu_i(\utilde{\beta})}$$

4. Expected information

$$\mathbf{I} = \mathrm{E}\left(-\frac{\partial^2 \ell(\mu(\utilde{\beta})|\utilde{y})}{\partial \beta_j \partial \beta_k} \right) = \utilde{x}_j^\top \utilde{x}_k$$

Solution: Gereralize to arbitrary number of predictors.

$$\begin{aligned} \mu_i \equiv \mu_i(\utilde{\beta}) &= \exp\{\utilde{x}_i^\top \utilde{\beta}\} \\ &= e^{x_{i1}\beta_1} e^{x_{i2}\beta_2} \cdots e^{x_{ip}\beta_p} \end{aligned}$$

1. first partial of mean vector

$$\begin{aligned} \frac{\partial \mu_i(\utilde{\beta})}{\partial \beta_j} &= x_{ij} e^{x_{i1}\beta_1} e^{x_{i2}\beta_2} \cdots e^{x_{ip}\beta_p} \\ &= x_{ij} \exp\{\utilde{x}_i^\top \utilde{\beta}\} \\ &= x_{ij} \mu_i(\utilde{\beta}) \end{aligned}$$

2. first partial of log-likelihood

$$\begin{aligned} \ell(\mu(\utilde{\beta})|\utilde{y}) &= -\sum_{i=1}^{n} \left(\log(\mu_i(\utilde{\beta})) + \frac{y_i}{\mu_i(\utilde{\beta})} \right) \\ &= -\sum_{i=1}^{n} \left(\log(\exp\{\utilde{x}_i^\top \utilde{\beta}\}) + \frac{y_i}{\exp\{\utilde{x}_i^\top \utilde{\beta}\}} \right) \\ &= -\sum_{i=1}^{n} \left(\utilde{x}_i^\top \utilde{\beta} + y_i \exp\{-\utilde{x}_i^\top \utilde{\beta}\} \right) \\ \frac{\partial \ell(\mu(\utilde{\beta})|\utilde{y})}{\partial \beta_j} &= -\sum_{i=1}^{n} \left(x_{ij} + x_{ij} y_i \exp\{-\utilde{x}_i^\top \utilde{\beta}\} \right) \\ &= \sum_{i=1}^{n} x_{ij} \left(y_i \exp\{-\utilde{x}_i^\top \utilde{\beta}\} - 1 \right) \\ &= \sum_{i=1}^{n} x_{ij} \left(\frac{y_i}{\mu_i(\utilde{\beta})} - 1 \right) \end{aligned}$$

3. second partial of log-likelihood

$$\begin{aligned}\frac{\partial^2 \ell(\mu(\underset{\sim}{\beta})|\underset{\sim}{y})}{\partial\beta_j\partial\beta_k} &= \frac{\partial}{\partial\beta_k}\left\{\sum_{i=1}^n x_{ij}\left(\frac{y_i}{\mu_i(\underset{\sim}{\beta})} - 1\right)\right\} \\ &= \frac{\partial}{\partial\beta_k}\left\{\sum_{i=1}^n x_{ij}\left(y_i \exp\{-\underset{\sim}{x}_i^\top \underset{\sim}{\beta}\} - 1\right)\right\} \\ &= \sum_{i=1}^n x_{ij}(-x_{ik}) y_i \exp\{-\underset{\sim}{x}_i^\top \underset{\sim}{\beta}\} \\ &= -\sum_{i=1}^n x_{ij}x_{ik}\frac{y_i}{\mu_i(\underset{\sim}{\beta})}\end{aligned}$$

4. Expected information

$$\begin{aligned}\mathbf{I} = \mathrm{E}\left(-\frac{\partial^2 \ell(\mu(\underset{\sim}{\beta})|\underset{\sim}{y})}{\partial\beta_j\partial\beta_k}\right) &= \mathrm{E}\left(\sum_{i=1}^n x_{ij}x_{ik}\frac{y_i}{\mu_i(\underset{\sim}{\beta})}\right) \\ &= \mathrm{E}\left(\sum_{i=1}^n x_{ij}x_{ik} y_i \exp\{-\underset{\sim}{x}_i^\top \underset{\sim}{\beta}\}\right) \\ &= \sum_{i=1}^n x_{ij}x_{ik}\mathrm{E}(y_i) \exp\{-\underset{\sim}{x}_i^\top \underset{\sim}{\beta}\} \\ &= \sum_{i=1}^n x_{ij}x_{ik} \exp\{\underset{\sim}{x}_i^\top \underset{\sim}{\beta}\}\exp\{-\underset{\sim}{x}_i^\top \underset{\sim}{\beta}\} \\ &= \sum_{i=1}^n x_{ij}x_{ik} \\ &= \underset{\sim}{x}_j^\top \underset{\sim}{x}_k\end{aligned}$$

(c) (15 pts) Using matrix notation, write the "structural" part of the model as

$$\log(\underset{\sim}{\mu}) = \mathbf{X}\underset{\sim}{\beta}$$

where $\mathbf{X}$ is an n-by-p design matrix with ith row $\underset{\sim}{x}_i^\top$ and $\log(\underset{\sim}{\mu})$ is an n-by-1 vector with ith element $\log(\mu_i)$.
Using obvious notation, derive the matrix representations of the derivatives of the mean vector and log-likelihood and the expected information, $\dot{\ell}(\underset{\sim}{\beta})$, $\ddot{\ell}(\underset{\sim}{\beta})$, $\mathbf{I}(\underset{\sim}{\beta}) = \mathrm{E}(-\ddot{\ell}(\underset{\sim}{\beta}))$.

1. first partial of log-likelihood

$$\dot{\ell}(\underset{\sim}{\beta}) = \left[\frac{\partial\ell}{\partial\beta_j}\right]_{p\times 1} = \mathbf{X}^\top\mathbf{D}^{-1}(\mu(\underset{\sim}{\beta}))(\underset{\sim}{y} - \mu(\underset{\sim}{\beta}))$$

where $\mathbf{D}^{-1}(\mu(\underset{\sim}{\beta}))$ is an n-by-n diagonal matrix with diagonal elements μ_i, and $\underset{\sim}{y} = [y_1, y_2, \ldots, y_n]^\top$.

2. second partial of log-likelihood

$$-\ddot{\ell}(\underset{\sim}{\beta}) = \left[-\frac{\partial^2 \ell}{\partial\beta_j \partial\beta_k}\right]_{p\times p} = \mathbf{X}^\top \mathbf{D}(\underset{\sim}{y}/\underset{\sim}{\mu})\mathbf{X}$$

where $\mathbf{D}(\underset{\sim}{y}/\underset{\sim}{\mu})$ is an n-by-n diagonal matrix with diagonal elements y_i/μ_i.

3. Expected information

$$\mathbf{I}(\underset{\sim}{\beta}) = \mathrm{E}(-\ddot{\ell}(\underset{\sim}{\beta})) = \mathbf{X}^\top\mathbf{X}.$$

Recall that the MLE of $\underset{\sim}{\beta}$ solves the likelihood equations $\dot{\ell}(\underset{\sim}{\beta}) = \underset{\sim}{0}_p$, a p-by-1 vector of zeros, and that for large n

$$\hat{\underset{\sim}{\beta}} \ \dot{\sim} \ \mathrm{Normal}_p(\underset{\sim}{\beta}, \mathbf{I}^{-1}(\hat{\underset{\sim}{\beta}})),$$

where the information matrix can be estimated by either $\mathbf{I}(\hat{\underset{\sim}{\beta}})$ or $-\ddot{\ell}(\underset{\sim}{\beta})$.

Solution:

1. first partial of log-likelihood

$$\begin{aligned}
\dot{\ell}(\underset{\sim}{\beta}) &= \left[\frac{\partial \ell}{\partial \beta_j}\right]_{p\times 1} \\
&= \left[\sum_{i=1}^n x_{ij}\left(\frac{y_i}{\mu_i(\underset{\sim}{\beta})} - 1\right)\right]_{p\times 1} \\
&= \left[\sum_{i=1}^n x_{ij}\left(\frac{y_i - \mu_i(\underset{\sim}{\beta})}{\mu_i(\underset{\sim}{\beta})}\right)\right]_{p\times 1} \\
&= \left[\underset{\sim}{x}_j^\top \mathbf{D}^{-1}(\mu(\underset{\sim}{\beta}))(\underset{\sim}{y} - \mu(\underset{\sim}{\beta}))\right]_{p\times 1} \\
&= \mathbf{X}^\top \mathbf{D}^{-1}(\mu(\underset{\sim}{\beta}))(\underset{\sim}{y} - \mu(\underset{\sim}{\beta}))
\end{aligned}$$

2. second partial of log-likelihood

$$\begin{aligned}
-\ddot{\ell}(\underset{\sim}{\beta}) &= \left[-\frac{\partial^2 \ell}{\partial\beta_j \partial\beta_k}\right]_{p\times p} \\
&= \left[\sum_{i=1}^n x_{ij}\frac{y_i}{\mu_i(\underset{\sim}{\beta})}x_{ik}\right]_{p\times p} \\
&= \left[\underset{\sim}{x}_j^\top \mathbf{D}(\underset{\sim}{y})\mathbf{D}^{-1}(\mu_i(\underset{\sim}{\beta}))\underset{\sim}{x}_k\right]_{p\times p} \\
&= \mathbf{X}^\top \mathbf{D}(\underset{\sim}{y})\mathbf{D}^{-1}(\mu_i(\underset{\sim}{\beta}))\mathbf{X}
\end{aligned}$$

3. Expected information

$$
\begin{aligned}
\mathbf{I}(\underset{\sim}{\beta}) = \mathrm{E}(-\ddot{\ell}(\underset{\sim}{\beta})) &= \mathrm{E}\left(\mathbf{X}^\top \mathbf{D}(\underset{\sim}{y})\mathbf{D}^{-1}(\mu_i(\underset{\sim}{\beta}))\mathbf{X}\right) \\
&= \mathbf{X}^\top \mathrm{E}\left(\mathbf{D}(\underset{\sim}{y})\right)\mathbf{D}^{-1}(\mu_i(\underset{\sim}{\beta}))\mathbf{X} \\
&= \mathbf{X}^\top \mathbf{D}(\mu_i(\underset{\sim}{\beta}))\mathbf{D}^{-1}(\mu_i(\underset{\sim}{\beta}))\mathbf{X} \\
&= \mathbf{X}^\top \mathbf{X}
\end{aligned}
$$

(d) (10 pts) Write down an iterative scheme to compute successive estimates of $\underset{\sim}{\beta}$ via Newton Rhapson and Fisher Scoring.

Solution: Newton-Raphson iterative scheme.

$$
\begin{aligned}
\hat{\beta}^{(i+1)} &= \hat{\beta}^{(i)} - [\ddot{\ell}(\hat{\beta}^{(i)})]^{-1}\dot{\ell}(\hat{\beta}^{(i)}) \qquad \text{in general} \\
\hat{\underset{\sim}{\beta}}^{(i+1)} &= \hat{\underset{\sim}{\beta}}^{(i)} - [\mathbf{X}^\top \mathbf{D}(\underset{\sim}{y})\mathbf{D}^{-1}(\mu_i(\underset{\sim}{\beta}^{(i)}))\mathbf{X}]^{-1}\mathbf{X}^\top \mathbf{D}^{-1}(\mu(\underset{\sim}{\beta}^{(i)}))(\underset{\sim}{y} - \mu(\underset{\sim}{\beta}^{(i)}))
\end{aligned}
$$

Fisher's Scoring iterative scheme.

$$
\begin{aligned}
\hat{\beta}^{(i+1)} &= \hat{\beta}^{(i)} + [\mathbf{I}(\beta^{(i)})]^{-1}\dot{\ell}(\hat{\beta}^{(i)}) \qquad \text{in general} \\
\hat{\underset{\sim}{\beta}}^{(i+1)} &= \hat{\underset{\sim}{\beta}}^{(i)} + [\mathbf{X}^\top \mathbf{X}]^{-1}\mathbf{X}^\top \mathbf{D}^{-1}(\mu(\underset{\sim}{\beta}^{(i)}))(\underset{\sim}{y} - \mu(\underset{\sim}{\beta}^{(i)}))
\end{aligned}
$$

(e) (10 pts) Write a function to compute the log-likelihood function in Part (a) allowing for two vectors of input $\underset{\sim}{y}$ and $\underset{\sim}{\mu}$ and matrix $\underset{\sim}{X}$.

Solution:

```
f.exp.l <- function(y, X, beta) {
  # Exponential regression, log-likelihood
  # standard linear model objects: y, X, beta

  # center the Xs
  #X[,2] <- X[,2] - mean(X[,2])
  #X <- X - matrix(rep(1, nrow(X)), ncol = 1) %*% colMeans(X)

  # exponential log-likelihood
  l <- sum(-(X %*% beta + y * exp(-X %*% beta)))
  return(l)
}
```

(f) (15 pts) Write a function to compute the MLE of $\underset{\sim}{\beta}$ for the exponential regression model $\log(\underset{\sim}{\mu}) = \mathbf{X}\underset{\sim}{\beta}$ using NR. Include a "line-search" or "step-size" adjustment as part of the routine. The "input" and "output" structure of this function should mimic my logistic regression functions in the notes.

Solution:

```
f.exp.NR <- function(X, y, beta.1, method = "F"
                    , eps1 = 1e-6, eps2 = 1e-7, maxit = 100) {
  # NR/Fisher's Scoring routine for estimation of Exponential model (with line search)
  # Input:
  #   X         = n-by-(r+1) design matrix
  #   y         = n-by-1 vector of success counts
  #   beta.1    = (r+1)-by-1 vector of starting values for regression est
  #   method    = "N" Newton-Raphson or "F" Fisher's Scoring
  # Iteration controlled by:
  #   eps1  = absolute convergence criterion for beta
  #   eps2  = absolute convergence criterion for log-likelihood
  #   maxit = maximum allowable number of iterations
  # Output:
  #   out   = list containing:
  #     beta.MLE  = beta MLE
  #     NR.hist   = iteration history of convergence differences
  #     beta.hist = iteration history of beta
  #     beta.cov  = beta covariance matrix (inverse Fisher's information matrix at MLE)
  #     note      = convergence note

  beta.2 <- rep(-Inf, length(beta.1)) # init beta.2
  diff.beta <- sqrt(sum((beta.1 - beta.2)^2)) # Euclidean distance

  llike.1 <- f.exp.l(y, X, beta.1) # update loglikelihood
  llike.2 <- f.exp.l(y, X, beta.2) # update loglikelihood
  diff.like <- abs(llike.1 - llike.2) # diff
  if (is.nan(diff.like)) { diff.like <- 1e9 }

  i <- 1                              # initial iteration index

  alpha.step <- seq(-1, 2, by = 0.1)[-11]  # line search step sizes, excluding 0

  NR.hist <- data.frame(i, diff.beta, diff.like, llike.1, step.size = 1) # iteration history
  beta.hist <- matrix(beta.1, nrow = 1)
  while ((i <= maxit) & (diff.beta > eps1) & (diff.like > eps2)) {
    i <- i + 1                        # increment iteration

    # update beta
    beta.2 <- beta.1                  # old guess is current guess

    mu.2 <- exp(X %*% beta.2)        # mean
    score.2 <- t(X) %*% diag(as.vector(1/mu.2)) %*% (y - mu.2) # score function, first derivati
    ldd <- -t(X) %*% diag(as.vector(y / mu.2)) %*% X  # second derivative
    EI <- t(X) %*% X  # expected information

    # this increment version solves for (beta.2-beta.1) without inverting Information
    if (method == "N") { # NR
      increm <- - solve(ldd, score.2) # solve for increment
    }
    if (method == "F") { # Fisher's Scoring
      increm <- solve( EI, score.2) # solve for increment
    }
```

```
    # line search for improved step size
    llike.alpha.step <- rep(NA, length(alpha.step)) # init llike for line search
    for (i.alpha.step in 1:length(alpha.step)) {
      llike.alpha.step[i.alpha.step] <- f.exp.l(y, X
                    , beta.2 + alpha.step[i.alpha.step] * increm)
    }
    # step size index for max increase in log-likelihood (if tie, [1] takes first)
    ind.max.alpha.step <- which(llike.alpha.step == max(llike.alpha.step))[1]

    beta.1 <- beta.2 + alpha.step[ind.max.alpha.step] * increm   # update beta

    diff.beta <- sqrt(sum((beta.1 - beta.2)^2)) # Euclidean distance

    llike.2 <- llike.1               # age likelihood value
    llike.1 <- f.exp.l(y, X, beta.1) # update loglikelihood
    diff.like <- abs(llike.1 - llike.2) # diff

    # iteration history
    NR.hist   <- rbind(NR.hist, c(i, diff.beta, diff.like, llike.1, alpha.step[ind.max.alpha.ste
    beta.hist <- rbind(beta.hist, matrix(beta.1, nrow = 1))
  }

  # prepare output
  out <- list()
  out$beta.MLE  <- beta.1
  out$iter      <- i - 1
  out$NR.hist   <- NR.hist
  out$beta.hist <- beta.hist
  out$beta.cov  <- solve(EI) # variance matrix for betas

  if (!(diff.beta > eps1) & !(diff.like > eps2)) {
    out$note <- paste("Absolute convergence of", eps1, "for betas and"
                    , eps2, "for log-likelihood satisfied")
  }
  if (i > maxit) {
    out$note <- paste("Exceeded max iterations of ", maxit)
  }
  return(out)
}
```

(g) (10 pts) Incorporate a `method = "F"` option in your function in Part (f) to use either NR (`"N"`) or Fisher Scoring (`"F"`) and add the Fisher Scoring increment.

Solution: Included as a switch above.

(h) (20 pts) Put it all together! Write a script that will read the survival data given at the start of the program, fit the model in Equation (2) as defined in the data description, and print and plot relevant summary output using both NR and Fisher Scoring.

Put some thought into the choice of appropriate starting values.

Qualitatively discuss the fit of the model, including an equation for the estimated mean survival time as a function of $\log_{10}$(wbc).

Solution:

```
eleuk <- read.table("http://statacumen.com/teach/SC1/SC1_HW11_Exp-leuk.dat", header = TRUE)

# create data variables: y, X
n <- nrow(eleuk)
y <- matrix(eleuk$weeks, ncol = 1)
X.temp <- eleuk$lwbc

# design matrix
X <- matrix(c(rep(1,n), X.temp), nrow = n)
colnames(X) <- c("Int", "lwbc")
r <- ncol(X) - 1  # number of regression coefficients - 1

# initial beta vector
beta.init <- c(0, rep(0, r))

# fit betas using our exponential regression NR/FS function
out.NR <- f.exp.NR(X, y, beta.init, "N")
out.NR
## $beta.MLE
##           [,1]
## Int   8.477498
## lwbc -1.109298
##
## $iter
## [1] 6
##
## $NR.hist
##   i    diff.beta    diff.like     llike.1 step.size
## 1 1          Inf 1.000000e+09 -1062.00000       1.0
## 2 2 2.115517e+00 8.803439e+02  -181.65614       2.0
## 3 3 2.798853e+00 9.435738e+01   -87.29876       2.0
## 4 4 3.884532e+00 3.407787e+00   -83.89097       1.5
## 5 5 2.146064e-01 1.391297e-02   -83.87706       1.0
## 6 6 5.208143e-03 1.115922e-05   -83.87705       1.0
## 7 7 4.493651e-06 7.986500e-12   -83.87705       1.0
##
## $beta.hist
##          [,1]        [,2]
## [1,] 0.000000  0.00000000
## [2,] 2.115152 -0.03929616
## [3,] 4.900935 -0.30946358
## [4,] 8.689748 -1.16648186
## [5,] 8.482514 -1.11071509
## [6,] 8.477502 -1.10929913
## [7,] 8.477498 -1.10929792
##
## $beta.cov
##             Int       lwbc
## Int   2.7383886 -0.6542095
## lwbc -0.6542095  0.1597237
out.FS <- f.exp.NR(X, y, beta.init, "F")
```

```
out.FS
## $beta.MLE
##           [,1]
## Int   8.477496
## lwbc -1.109297
##
## $iter
## [1] 8
##
## $NR.hist
##   i    diff.beta    diff.like      llike.1 step.size
## 1 1          Inf 1.000000e+09 -1062.00000       1.0
## 2 2 3.109546e+01 9.241512e+02  -137.84884       0.1
## 3 3 9.268232e+00 2.200089e+01  -115.84795       0.5
## 4 4 4.624114e+00 2.067392e+01   -95.17403       2.0
## 5 5 8.618616e+00 1.075806e+01   -84.41597       2.0
## 6 6 1.173446e-01 5.380403e-01   -83.87793       1.1
## 7 7 2.752409e-02 8.843883e-04   -83.87705       1.0
## 8 8 2.916286e-03 1.398456e-06   -83.87705       1.1
## 9 9 7.509020e-05 1.231044e-09   -83.87705       1.1
##
## $beta.hist
##            [,1]      [,2]
##  [1,]  0.000000  0.000000
##  [2,] 30.520749 -5.950779
##  [3,] 21.544713 -3.641903
##  [4,] 16.976702 -2.923775
##  [5,]  8.560754 -1.065722
##  [6,]  8.448561 -1.100109
##  [7,]  8.474739 -1.108612
##  [8,]  8.477569 -1.109317
##  [9,]  8.477496 -1.109297
##
## $beta.cov
##             Int       lwbc
## Int   2.7383886 -0.6542095
## lwbc -0.6542095  0.1597237
```

The MLE procedure for this particular log-likelihood is sensitive to initial conditions since the solution converges to a ridge, as discussed in Part i. Using starting values for $(\beta_0, \beta_1) = (0, 0)$ the two methods both converge to approximately 8.477 for the constant term, -1.109 for the linear term, with a log-likelihood value $\ell(\underset{\sim}{\beta}) = -83.88$. The plot below indicates that the model follows the pattern of the data well.

```
f.exp.pred <- function(x, beta) {
  # Exponential regression, predicted values
  # univariate x, vector beta

  # exponential log-likelihood
  #y <- beta[1] * exp(beta[2] * (x - mean(x)))
  y <- exp(beta[1] + beta[2] * x)
  return(y)
}

# Plot at top of HW
#library(ggplot2)
#p <- ggplot(eleuk, aes(x = lwbc, y = weeks))
#p <- p + scale_y_continuous(breaks = 52*c(0,1,2,3))
```

```
#p <- p + stat_function(fun = f.exp.pred, args = list(out.FS$beta.MLE) )
#p <- p + geom_point(alpha = 0.5, size = 3)
#p <- p + labs(title = "Leukemia: weeks survival by log white blood cell count")
#print(p)
```

The examples below show that convergence depends on starting values, as well as the method used (NR or FS). More discussion in Part i.

```
out.NR.00.00 <- f.exp.NR(X, y, c( 0,  0), "N"); out.NR.00.00$iter
## [1] 6
out.FS.00.00 <- f.exp.NR(X, y, c( 0,  0), "F"); out.FS.00.00$iter
## [1] 8
out.NR.00.02 <- f.exp.NR(X, y, c( 0,  2), "N"); out.NR.00.02$iter
## Error in solve.default(ldd, score.2):  Lapack routine dgesv:  system is exactly
singular:  U[2,2] = 0
## Error in eval(expr, envir, enclos):  object 'out.NR.00.02' not found
out.FS.00.02 <- f.exp.NR(X, y, c( 0,  2), "F"); out.FS.00.02$iter
## [1] 7
out.NR.10.10 <- f.exp.NR(X, y, c(10, 10), "N"); out.NR.10.10$iter
## Error in while ((i <= maxit) & (diff.beta > eps1) & (diff.like > eps2)) {:  missing
value where TRUE/FALSE needed
## Error in eval(expr, envir, enclos):  object 'out.NR.10.10' not found
out.FS.10.10 <- f.exp.NR(X, y, c(10, 10), "F"); out.FS.10.10$iter
## [1] 30
```

(i) (10 pts) Explore convergence of NR and Fisher Scoring as a function of the starting values. That is, does the choice of starting values affect whether the routine converges, and if so, how rapidly? Discuss.

Solution: The left plot shows the log-likelihood we'd like to maximize as a function of (β_0, β_1). The domain in these plots is $\beta_0 \in [-15, 15], \beta_1 \in [-10, 10]$. As with many log-likelihoods, there is a very steep increase in one direction, then a slow decrease in the other direction. We observe that the pair (β_0, β_1) converge to the closest point along the ridge in the plots. Convergence of both the Newton-Raphson and Fisher's scoring methods for obtaining the MLE are sensitive to the initial conditions. We should consider our estimate of β_0 to not be reliable.

The right plot shows the length of the vector that is the difference between initial and MLE of $\underset{\sim}{\beta}$, that is $\|\underset{\sim}{\beta}^{(1)} - \underset{\sim}{\beta}^{(0)}\|$. The domain in this plot is the same. The largest differences occur for larger positive values of both β_0 and β_1.

From these plots, I feel comfortable choosing an initial (β_0, β_1) near the center of the range of these plots. Doing so gives the estimates I presented in part (h). Going outside this area gives different estimates for β_0 which appear unreliable based on the iterations, or failure to converge.

```
# ranges of betas to explore
beta0 <- seq(-15, 15, length = 50)
beta1 <- seq(-10, 10, length = 50)
df <- data.frame(expand.grid(beta0, beta1))
colnames(df) <- c("beta0", "beta1")
df$l <- NA # log-likelihood
df$d <- NA # distance between initial and beta MLE
```

```
for (ii in 1:nrow(df)) {
  beta.x <- c(df$beta0[ii], df$beta1[ii])
  df$l[ii] <- f.exp.l(y, X, beta.x)  #£
  df$d[ii] <- sqrt(sum((out.NR$beta.MLE - beta.x)^2))
}

library(lattice)
wireframe(l ~ beta0 * beta1
  , data = df
  , main = "log-likelihood"
  , shade = TRUE
  , screen = list(z = -50, x = -70)
)

wireframe(d ~ beta0 * beta1
  , data = df
  , main = "Euclidian distance from beta MLE"
  , shade = TRUE
  , screen = list(z = -50, x = -70)
)
```