

Stat 579 Statistical Computing II Homework 4

Erik Barry Erhardt

May 5, 2006

All code can be found in the Appendix.

Exercise 1 *Hardy-Weinberg Equilibrium with Newton-Raphson and Fisher Scoring evaluations of the MLE for θ .*

	Blood Type			
	$M(x_1)$	$MN(x_2)$	$N(x_3)$	Total (n)
Frequency	342	500	187	1029

Multinomial Distribution

$$\begin{aligned}L(\theta) &= \Pr(X_1 = x_1, X_2 = x_2, X_3 = x_3) \\&= \frac{n!}{\prod_{i=1}^3 x_i!} p_1(\theta)^{x_1} p_2(\theta)^{x_2} p_3(\theta)^{x_3} \\&= \frac{n!}{\prod_{i=1}^3 x_i!} (1 - \theta)^{2x_1} (2\theta(1 - \theta))^{x_2} \theta^{2x_3} \\&= \frac{n!}{\prod_{i=1}^3 x_i!} (1 - \theta)^{2x_1 + x_2} \theta^{x_2 + 2x_3} 2^{x_2}\end{aligned}$$

(a) *Derive the Multinomial log-likelihood, $\ell(\theta)$.*

$$\begin{aligned}\ell(\theta) &= \log(n!) - \sum_{i=1}^3 \log(x_i!) \\&\quad + (2x_1 + x_2) \log(1 - \theta) + (x_2 + 2x_3) \log(\theta) + (x_2) \log(2)\end{aligned}$$

(b) *Derive the derivatives of the log-likelihood and the expected information, $\dot{\ell}(\theta)$, $\ddot{\ell}(\theta)$, $I(\theta) = E(-\ddot{\ell}(\theta))$.*

$$\begin{aligned}\dot{\ell}(\theta) &= -\frac{2x_1 + x_2}{1 - \theta} + \frac{x_2 + 2x_3}{\theta} \\ \ddot{\ell}(\theta) &= -\frac{2x_1 + x_2}{(1 - \theta)^2} - \frac{x_2 + 2x_3}{\theta^2}\end{aligned}$$

$X_i \sim \text{Bin}(n, p_i(\theta))$ so $E(X_i) = np_i(\theta)$, giving

$$\begin{aligned}
 I(\theta) &= E(-\ddot{\ell}(\theta)) \\
 &= E\left(\frac{2x_1 + x_2}{(1-\theta)^2} + \frac{x_2 + 2x_3}{\theta^2}\right) \\
 &= n\left(\frac{2p_1(\theta) + p_2(\theta)}{(1-\theta)^2} + \frac{p_2(\theta) + 2p_3(\theta)}{\theta^2}\right) \\
 &= n\left(\frac{2(1-\theta)^2 + 2\theta(1-\theta)}{(1-\theta)^2} + \frac{2\theta(1-\theta) + 2\theta^2}{\theta^2}\right) \\
 &= 2n\left(\frac{(1-\theta) + \theta}{(1-\theta)} + \frac{(1-\theta) + \theta}{\theta}\right) \\
 &= 2n\left(\frac{1}{(1-\theta)} + \frac{1}{\theta}\right) \\
 &= 2n\left(\frac{\theta + (1-\theta)}{(1-\theta)\theta}\right) \\
 &= 2n\left(\frac{1}{(1-\theta)\theta}\right)
 \end{aligned}$$

(c) Likelihood equations, $L(\theta|\underline{d})$.

$$\begin{aligned}
 \dot{\ell}(\theta) = 0 &= -\frac{2x_1 + x_2}{1-\theta} + \frac{x_2 + 2x_3}{\theta} \\
 &= -\frac{-2x_1\theta - x_2\theta + x_2(1-\theta) + 2x_3(1-\theta)}{(1-\theta)\theta} \\
 0 &= (-2x_1 - x_2 - x_2 - 2x_3)\theta + x_2 + 2x_3 \\
 \hat{\theta} &= \frac{x_2 + 2x_3}{2x_1 + 2x_2 + 2x_3} \\
 &= \frac{500 + 2(187)}{2(342) + 2(500) + 2(187)} \\
 &= 0.424684159
 \end{aligned}$$

(d) Matlab functions to evaluate functions in parts (a) and (b).

Four Matlab functions `*multinomial_hardyweinberg.m` in the appendix.

(e) *Plot $\ell(\theta)$, guess at MLE.*

The plot in Figure 1 on page 4 suggests that the MLE is close to $\theta = 0.42$.

(f) *Newton-Raphson for MLE.*

Matlab function `newton_raphson_hardyweinberg.m` in the appendix.

Regardless of the starting value of $\theta \in (0, 1)$, after a few iterations, the MLE is decided at $\hat{\theta} = 0.424684159$.

(g) *Fisher scoring for MLE.*

Matlab function `fisher_scoring_hardyweinberg.m` in the appendix.

Regardless of the starting value of $\theta \in (0, 1)$, after a single iteration, the MLE is decided at $\hat{\theta} = 0.424684159$.

(h) *Compare NR and Fisher scoring methods.*

Both methods converge quickly to the MLE, though Fisher's scoring method unconditionally converges in a single step.

(i) *Sensitivity analysis.*

The plot in Figure 2 on page 4 indicates that both Newton-Raphson and Fisher's scoring methods will converge to the correct solution for all starting values. It indicates that Fisher's scoring does converge in a single step regardless of the initial value of θ .

(j) *Summary.*

It is remarkably easy to maximize a univariate likelihood function, and to know when particular maximization methods will converge. This was a great example.

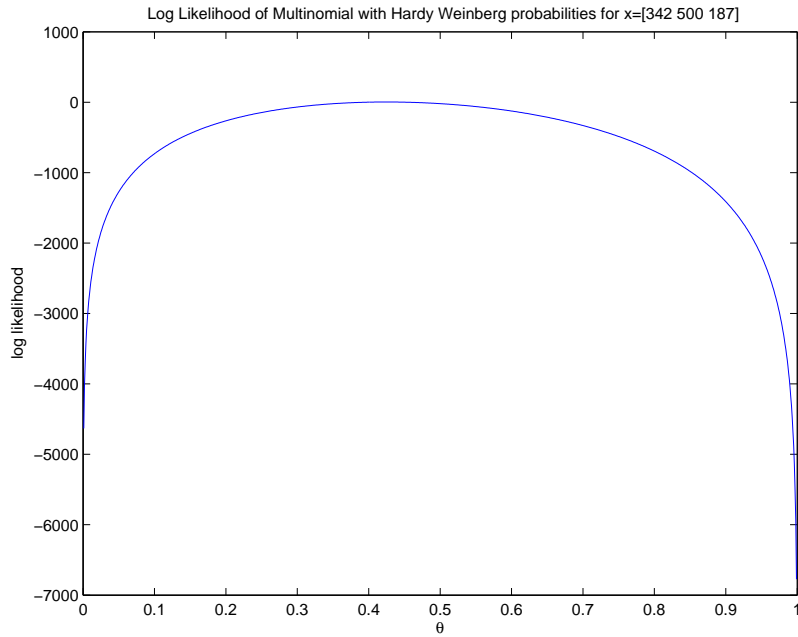


Figure 1: 4.1e log-likelihood $\ell(\theta)$ nearly maximized at $\theta = 0.42$.

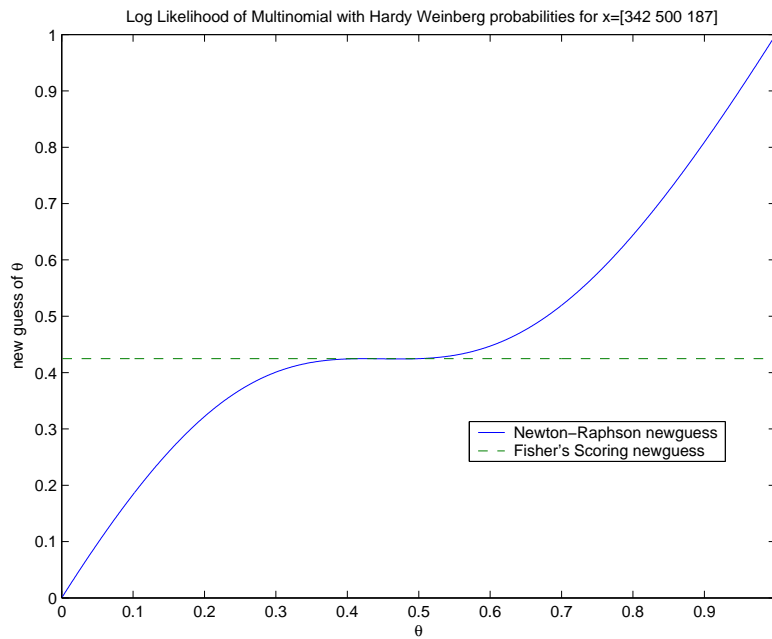


Figure 2: 4.1i Next value of θ in NR and FS iterative schemes.

Exercise 2 *Exponential Regression.* $Y_i \stackrel{ind}{\sim} \text{Exp}(\mu_i)$

$$\begin{aligned} y_i &= \exp\{\beta_0 + \beta_1(x_i - \bar{x})\}\varepsilon_i, \\ E(y_i) = \mu_i &= \exp\{\beta_0 + \beta_1(x_i - \bar{x})\}, \\ f(y_i|\mu_i) &= \frac{1}{\mu_i} \exp\left\{-\frac{y_i}{\mu_i}\right\} \end{aligned}$$

(a) Derive the log-likelihood, $\ell(\underline{\mu})$.

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

(b) Derive the derivatives of the mean vector and log-likelihood and derive the expected information.

Generalize to arbitrary number of predictors.

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

(b) 1. first partial of mean vector

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

(b) 2. first partial of log-likelihood

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

(b) 3. *second partial of log-likelihood*

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

(b) 4. *Expected information*

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

(c) Derive the matrix representations of the derivatives of the mean vector and log-likelihood and the expected information, $\dot{\ell}(\underline{\beta})$, $\ddot{\ell}(\underline{\beta})$, $I(\underline{\beta}) = E(-\ddot{\ell}(\underline{\beta}))$.

(c) 1. first partial of log-likelihood

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

(c) 2. second partial of log-likelihood

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

(c) 3. Expected information

$$\begin{aligned}
 I(\underline{\beta}) = E(-\ddot{\ell}(\underline{\beta})) &= E\left(\mathbf{X}' \mathbf{D}(\underline{y}) \mathbf{D}^{-1}(\underline{\mu}_i(\underline{\beta})) \mathbf{X}\right) \\
 &= \mathbf{X}' E\left(\mathbf{D}(\underline{y})\right) \mathbf{D}^{-1}(\underline{\mu}_i(\underline{\beta})) \mathbf{X} \\
 &= \mathbf{X}' \mathbf{D}(\underline{\mu}_i(\underline{\beta})) \mathbf{D}^{-1}(\underline{\mu}_i(\underline{\beta})) \mathbf{X} \\
 &= \mathbf{X}' \mathbf{I} \mathbf{X} \\
 &= \mathbf{X}' \mathbf{X}
 \end{aligned}$$

(d) *Iterative scheme via Newton-Raphson and Fisher Scoring.*

(d) **Newton-Raphson** *Iterative scheme.*

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

(d) **Fisher's Scoring** *Iterative scheme.*

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

(e) *Matlab function to compute the log-likelihood.*

Matlab function `log_likelihood_exponential_regression.m` in the appendix.

(f) & (g) *Matlab function to compute MLE of β using Newton-Raphson and Fisher Scoring.*

Matlab function `mle_exponential_regression.m` in the appendix.

(h) *Matlab script to read leukemia Survival data and fit the model using both NR and FS.*

Matlab script in the appendix. The MLE procedure for this particular log-likelihood is sensitive to initial conditions since the solution converges to a ridge, as discussed in (i). Using starting values for $(\beta_0, \beta_1) = (0, 0)$ the two methods both converge to approximately $(8.4742, -1.1092)$ with a log-likelihood value $\ell_{\hat{\beta}} = -14770$.

The plot in Figure 3 on page 10 shows the fitted exponential regression overlaid on the original data. The curve fits very well.

Table 1 on page 10 gives a few examples of convergence. Starts around $(0, 0)$ converge to the same location giving $\hat{\beta}_1 = -1.1092$, but can be different for $\hat{\beta}_0$, which is understandable given the discussion in part (i). For starting values $(10, 10)$, Fisher's method diverged immediately, while Newton-Raphson converged quickly in β_1 but was converging extremely slowly in β_0 , and stopped after 100 iterations.

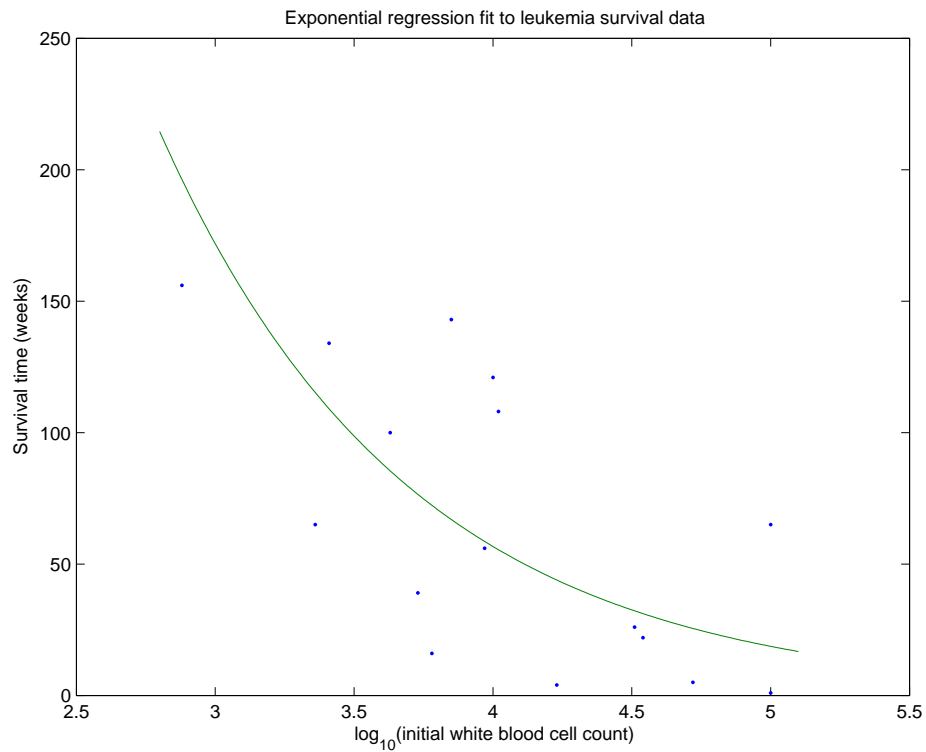


Figure 3: 2h. Plot of data points and fitted exponential regression.

$\beta_0^{(0)}$	$\beta_1^{(0)}$	Method	its	ℓ	step size	$\mathcal{Z}'\mathcal{Z}$	$\hat{\beta}_0$	$\hat{\beta}_1$
0	2	F	5	-14770	1.0669	0.0040794	8.4778	-1.1093
0	2	N	85	-14770	1.9999	9.8951e-005	8.4767	-1.1091
0	0	F	3	-14770	0.18295	0.88936	7.5985	-1.1093
0	0	N	80	-14770	1.9999	0.00043815	8.4742	-1.1092
10	10	F	1	-Inf	-1.0001	4.2464e+027	4.0983e+027	-1.1117e+027
10	10	N	99	-14770	1.9999	0.11745	-34.767	-1.1097

Table 1: 2h

(i) *Sensitivity analysis.*

The first three plots in Figure 4 on page 12 show the log-likelihood we'd like to maximize as a function of (β_0, β_1) . The range in these plots is $\beta_0 \in [-200 : 4 : 200], \beta_1 \in [-5 : .1 : 5]$. 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 fourth plot shows the length of the vector that is the difference between initial and new estimates of $\underline{\beta}$, that is $\|\underline{\beta}^{(1)} - \underline{\beta}^{(0)}\|$, in a single iteration of the NR MLE algorithm. The range in this plot is $\beta_0 \in [-15 : .1 : 15], \beta_1 \in [-15 : .1 : 15]$, with the vertical axis going to 14×10^{22} . 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.

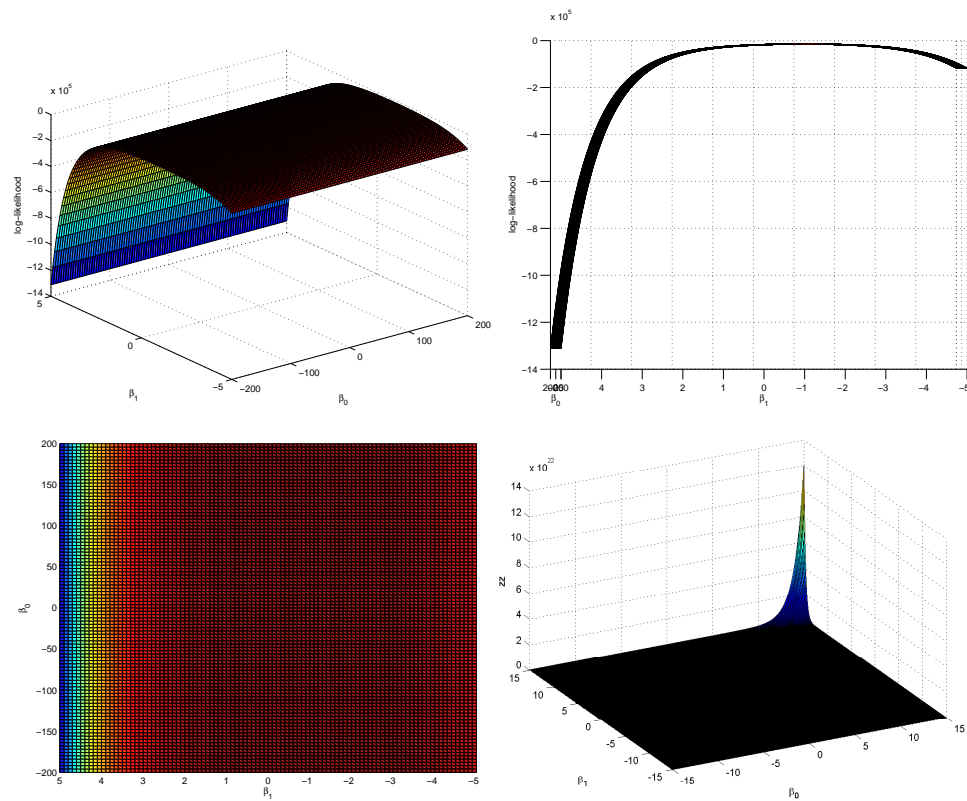


Figure 4: 2i. Three plots of the log-likelihood, and one of the norm of differences in beta vectors in one iteration of NR.

Exercise 3¹ *Nonparametric and parametric bootstrap distributions of the MLE for θ from Exercise 1.*

(a) *Nonparametric Bootstrap.*

The nonparametric bootstrap starts with the observed counts of M, MN and N of $x_1 = 342, x_2 = 500$, and $x_3 = 187$. Therefore, our sample has 342 Ms, 500 MNs, and 187 Ns, and the bootstrap procedure resamples with replacement these Ms, MNs, and Ns. From each bootstrap resample of size $n = 1029$, the MLE will be calculated as it was in exercise 1.

(b) *Parametric Bootstrap.*

The parametric bootstrap uses the observed counts of M, MN and N of $x_1 = 342, x_2 = 500$, and $x_3 = 187$, along with the Multinomial model. The estimated proportions in the multinomial model are estimated from the observed data $p_1 = 342/1029 = 0.332, p_2 = 500/1029 = 0.486$, and $p_3 = 187/1029 = 0.182$. Counts of M, MN and N are simulated from a multinomial distribution with these parameters, using the program written in homework 1. From each parametric sample of size $n = 1029$, the MLE will be calculated as it was in exercise 1.

(c) *Comparison.*

The plot in Figure 5 on page 14 shows both the nonparametric and parametric bootstrap with $B = 10000$ resamples. It suggests that either bootstrap method produces the same information about the sampling distribution of MLE $\hat{\theta}$. That means that the multinomial is an appropriate model for these data (which we knew).

¹I'd do much more, but I'm very ready for the semester to end.

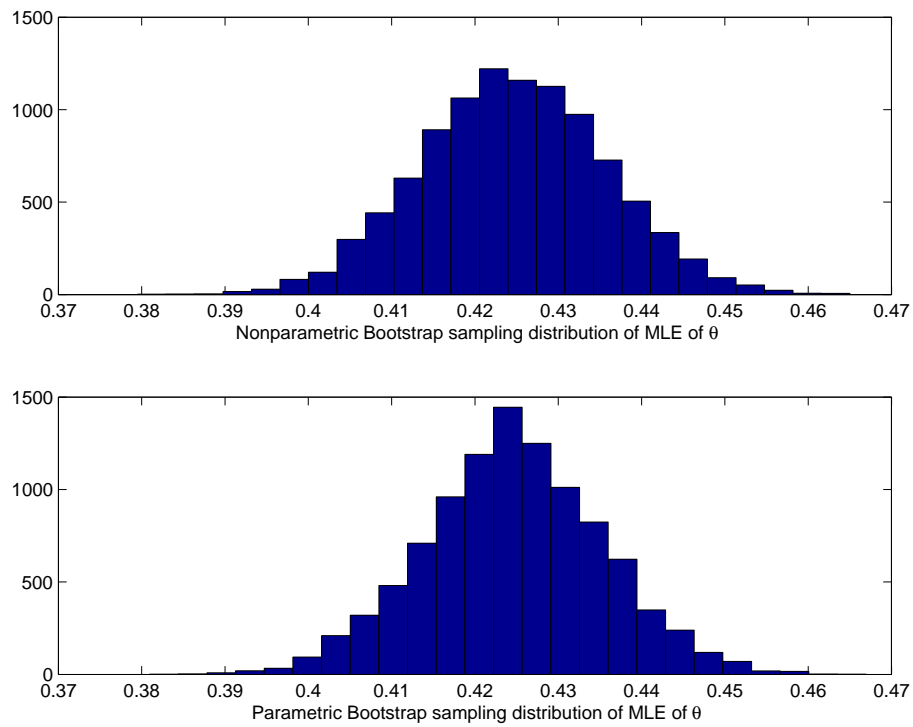


Figure 5: 4.3 Nonparametric and Parametric Bootstraps of MLE $\hat{\theta}$ from exercise 1.

Appendix

Matlab code used for the analysis above

```

%%% Exercise 1 =====
% 1c =====
x=[342 500 187];
thetahat = (x(2)+2*x(3))/(2*(sum(x)))
% 1d =====
function [ll] = log_likelihood_multinomial_hardyweinberg(x,theta)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% log_likelihood_multinomial_hardyweinberg.m
% Stat 579 Statistical Computing II Homework 2
% Create a M-file to calculate the log-likelihood.
% input (x,theta) where
%   x is a vector (3x1) of counts, and
%   theta is a scalar probability.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author:
% Erik Barry Erhardt, M.S., Ph.D. student Statistics
% Dept of Math & Stats, Univ. of New Mexico, Albuquerque, NM 87131, U.S.A.
% Office: Humanities 328, MSC03 2150
% erike AT stat.unm.edu   http://www.stat.unm.edu/~erike/
% (505)277-0757         Fax: (505)277-5505
% American Statistical Association Albuquerque Chapter Representative
%
% Date: 5/2/2006
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if nargin~=2;error('log_likelihood_multinomial_hardyweinberg requires 2 arguments');return;end;
if length(find(x<0)) > 0;error('Counts in x must all be nonnegative');return;end;
if length(find(theta<=0)) > 0 | length(find(theta>1)) > 0;error('Proportion in theta must between 0 and 1');return;end;
ll = gammaln(sum(x)) - sum(gammaln(x)) + (2*x(1)+x(2))*log(1-theta) + (x(2)+2*x(3))*log(theta) + x(2)*log(2); % log-lik
% EOF

function [dll] = dlog_likelihood_multinomial_hardyweinberg(x,theta)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% dlog_likelihood_multinomial_hardyweinberg.m
% Stat 579 Statistical Computing II Homework 2
% Create a M-file to calculate the derivative of log-likelihood.
% input (x,theta) where
%   x is a vector (3x1) of counts, and
%   theta is a scalar probability.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author:
% Erik Barry Erhardt, M.S., Ph.D. student Statistics
% Dept of Math & Stats, Univ. of New Mexico, Albuquerque, NM 87131, U.S.A.
% Office: Humanities 328, MSC03 2150
% erike AT stat.unm.edu   http://www.stat.unm.edu/~erike/
% (505)277-0757         Fax: (505)277-5505
% American Statistical Association Albuquerque Chapter Representative
%
% Date: 5/2/2006
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if nargin~=2;error('dlog_likelihood_multinomial_hardyweinberg requires 2 arguments');return;end;
if length(find(x<0)) > 0;error('Counts in x must all be nonnegative');return;end;
if length(find(theta<=0)) > 0 | length(find(theta>1)) > 0;error('Proportion in theta must between 0 and 1');return;end;
dll = -(2*x(1)+x(2))/(1-theta) + (x(2)+2*x(3))/theta; % derivative log-likelihood
% EOF

function [ddll] = ddlog_likelihood_multinomial_hardyweinberg(x,theta)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% ddlog_likelihood_multinomial_hardyweinberg.m
% Stat 579 Statistical Computing II Homework 2
% Create a M-file to calculate the 2nd derivative of log-likelihood.
% input (x,theta) where
%   x is a vector (3x1) of counts, and
%   theta is a scalar probability.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author:
% Erik Barry Erhardt, M.S., Ph.D. student Statistics
% Dept of Math & Stats, Univ. of New Mexico, Albuquerque, NM 87131, U.S.A.
% Office: Humanities 328, MSC03 2150
% erike AT stat.unm.edu   http://www.stat.unm.edu/~erike/

```

```

% (505)277-0757          Fax: (505)277-5505
% American Statistical Association Albuquerque Chapter Representative
%
% Date: 5/2/2006
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if nargin==2;error('ddlog_likelihood_multinomial_hardyweinberg requires 2 arguments');return;end;
if length(find(x<0)) > 0;error('Counts in x must all be nonnegative');return;end;
if length(find(theta<=0)) > 0 | length(find(theta>1)) > 0;error('Proportion in theta must between 0 and 1');return;end;
ddll = -(2*x(1)+x(2))/(1-theta)^2 - (x(2)+2*x(3))/theta^2; % 2nd derivative log-likelihood
% EOF

function [ei] = EInformation_multinomial_hardyweinberg(x,theta)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% EInformation_multinomial_hardyweinberg.m
% Stat 579 Statistical Computing II Homework 2
% Create a M-file to calculate the Expected information of log-likelihood.
% input (x,theta) where
%   x is a vector (3x1) of counts, and
%   theta is a scalar probability.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author:
% Erik Barry Erhardt, M.S., Ph.D. student Statistics
% Dept of Math & Stats, Univ. of New Mexico, Albuquerque, NM 87131, U.S.A.
% Office: Humanities 328, MSC03 2150
% erike AT stat.unm.edu   http://www.stat.unm.edu/~erike/
% (505)277-0757          Fax: (505)277-5505
% American Statistical Association Albuquerque Chapter Representative
%
% Date: 5/2/2006
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
if nargin==2;error('EInformation_multinomial_hardyweinberg requires 2 arguments');return;end;
if length(find(x<0)) > 0;error('Counts in x must all be nonnegative');return;end;
if length(find(theta<=0)) > 0 | length(find(theta>1)) > 0;error('Proportion in theta must between 0 and 1');return;end;

ei = 2*sum(x)/((1-theta)*theta); % Expected information
% EOF
% 1e  =====
x=[342 500 187];
theta=.001:.001:0.999;
ll=zeros(size(theta));
for i_theta=theta
    ii_theta = find(theta==i_theta);
    ll(ii_theta)=log_likelihood_multinomial_hardyweinberg(x,i_theta);
end;
plot(theta,ll);
xlabel('\theta');ylabel('log-likelihood');title('log-likelihood of Multinomial with Hardy Weinberg probabilities for x');
plot_name = strcat('sc2hw04-1e.eps'); % plot name
print(gcf, '-depsc2', plot_name); % print plot

% if  =====
function root = newton_raphson_hardyweinberg(x,theta)
% NR routine for finding root of log-likelihood
% Requires first and second derivatives of the log-likelihood
%
% The iteration is controlled by:
%
% eps = absolute convergence criterion
% maxit = maximum allowable number of iterations
%
% Input: theta = starting value
% Output: number of steps, root, and log-likelihood at root
maxit = 35;
eps = 1e-6;

thetaold = 0; % arbitrary: needed so argument in while loop defined
thetaneu = theta;

i = 0; % initial iteration index
while (i <= maxit) & (abs(thetaneu - thetaold) > eps);
    thetaold = thetaneu; % old guess is current guess
    thetaneu = thetaold - dlog_likelihood_multinomial_hardyweinberg(x,thetaold)./ddlog_likelihood_multinomial_hardyweinberg(x,thetaold);
    disp([i thetaold thetaneu dlog_likelihood_multinomial_hardyweinberg(x,thetaneu)]); % display history
    i = i+1; % increment iteration
end
root = thetaneu;

```

```

[(i-1) root dlog_likelihood_multinomial_hardyweinberg(x,root)]

% 1g
=====
function root = fisher_scoring_hardyweinberg(x,theta)
% Fisher's Scoring routine for finding root of log-likelihood
% Requires first and second derivatives of the log-likelihood
%
% The iteration is controlled by:
%
% eps = absolute convergence criterion
% maxit = maximum allowable number of iterations
%
% Input: theta = starting value
% Output: number of steps, root, and log-likelihood at root
maxit = 35;
eps = 1e-6;

thetaold = 0; % arbitrary: needed so argument in while loop defined
thetaneu = theta;

i = 0; % initial iteration index
while (i <= maxit) & ( abs(thetaneu - thetaold) > eps );
thetaold = thetaneu; % old guess is current guess
thetaneu = thetaold + dlog_likelihood_multinomial_hardyweinberg(x,thetaold)/EInformation_multinomial_hardyweinberg(x,thetaold);
disp([i thetaold thetaneu dlog_likelihood_multinomial_hardyweinberg(x,thetaneu)]); % display history
i = i+1; % increment iteration
end

root = thetaneu;
[(i-1) root dlog_likelihood_multinomial_hardyweinberg(x,root)]

% 1i
=====

x=[342 500 187];
theta=.001:.001:0.999;
nr=zeros(size(theta)); % Newton-Raphson
fs=zeros(size(theta)); % Fisher's scoring
for i_theta=theta
ii_theta = find(theta==i_theta);
nr(ii_theta) = i_theta - dlog_likelihood_multinomial_hardyweinberg(x,i_theta)/ddlog_likelihood_multinomial_hardyweinberg(x,i_theta);
fs(ii_theta) = i_theta + dlog_likelihood_multinomial_hardyweinberg(x,i_theta)/EInformation_multinomial_hardyweinberg(x,i_theta);
end;
plot(theta,nr,'-', theta,fs,'--');
xlabel('\theta');ylabel('new guess of \theta');title('log-likelihood of Multinomial with Hardy Weinberg probabilities');
legend('Newton-Raphson newguess', 'Fisher's Scoring newguess',0)
plot_name = strcat('sc2hw04-1i.eps'); % plot name
print(gcf, '-depsc2', plot_name); % print plot

%% Exercise 2
% 2e
=====

function [ll] = log_likelihood_exponential_regression(y,X,beta)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% log_likelihood_exponential_regression.m
% Stat 579 Statistical Computing II Homework 2
% Create a M-file to calculate the log-likelihood.
% input (x,theta) where
% x is a vector (3x1) of counts, and
% theta is a scalar probability.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author:
% Erik Barry Erhardt, M.S., Ph.D. student Statistics
% Dept of Math & Stats, Univ. of New Mexico, Albuquerque, NM 87131, U.S.A.
% Office: Humanities 328, MSC03 2150
% erike AT stat.unm.edu http://www.stat.unm.edu/~erike/
% (505)277-0757 Fax: (505)277-5505
% American Statistical Association Albuquerque Chapter Representative
%
% Date: 5/2/2006
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%if nargin<2,error('log_likelihood_exponential_regression requires 2 arguments');return;end;
X = X-(diag(mean(X))*ones(size(X)))'; % center X's
ll = -X*beta-y*exp(-X*beta); % log likelihood
ll = sum(ll);
% EOF

% 2fg
=====

```

```

function [beta,il,zz,likn,covm,ithist] = mle_exponential_regression(y,X,beta,method)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% mle_exponential_regression.m
% Stat 579 Statistical Computing II Homework 2
% Create a M-file to calculate the log-likelihood.
% input (x,theta) where
%   x is a vector (3x1) of counts, and
%   theta is a scalar probability.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Author:
% Erik Barry Erhardt, M.S., Ph.D. student Statistics
% Dept of Math & Stats, Univ. of New Mexico, Albuquerque, NM 87131, U.S.A.
% Office: Humanities 328, MSC03 2150
% erike AT stat.unm.edu   http://www.stat.unm.edu/~erike/
% (505)277-0757           Fax: (505)277-5505
% American Statistical Association Albuquerque Chapter Representative
%
% Date: 5/2/2006
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%
% lrmle1: Fisher scoring estimation of LR model (with line search)
%
% input:   X = n-by-a design matrix
%          y = n-by-1 vector of responses
%          beta= a-by-1 vector of starting values for regression est
%          method = 'N' Newton-Raphson or 'F' Fisher's Scoring
%
% output:  bn   = a-by-1 vector of MLES for regression estimates
%          il   = number of iterations
%          zz   = SS of abs diff in estimates (no convergence if large)
%          likn = log-like at MLE
%          covm = covariance matrix of MLE
%          ithist = iteration history (il,zz,likn,bn)
if (method=='N' & method ~= 'F');error('method must be either N or F');return;end;

eps1 = 1e-4;           % absolute convergence criterion for beta
eps2 = 1e-5;           % absolute convergence criterion for log-like
itmax = 50;            % max number of iterations
alp   = -1.00005:1e-3:2.00005; % line search values
%beta = beta;          % 1st guess
likn = log_likelihood_exponential_regression(y,X,beta); % log-like at first guess
liko = likn + 100;     % dummy value
zz   = 5;              % set dummy for norm of diff between est
il = 1;                % start loop
while ( il < itmax ) & ( zz > eps1 ) & ( abs(likn - liko) > eps2 );
betao = beta;          % current beta, log-like, means
liko = likn;           % and diag matrix of variances
if method=='N';
del = (-X'*diag(y'*exp(-X*beta))*X) \ X'*diag(exp(-X*beta))*(y-exp(X*beta));
end;
if method=='F';
del = (X'*X) \ X'*diag(exp(-X*beta))*(y-exp(X*beta));
end;

is = 1;                % line search for best multiplier
for is=1:length(alp);
if method=='N';
btemp = betao - alp(is).*del;
end;
if method=='F';
btemp = betao + alp(is).*del;
end;
ltem(is) = log_likelihood_exponential_regression(y,X,btemp);
end;

[likn,iwhere] = max(ltem); % locate max of log-like and location
if method=='N';
beta = betao - alp(iwhere).*del; % update beta, zz for check at next step
end;
if method=='F';
beta = betao + alp(iwhere).*del; % update beta, zz for check at next step
end;
% likn = loglike(y,m,p(x,bn));
zz = sqrt( (beta-betao)'*(beta-betao) );
ithist(il,:) = [il likn alp(iwhere) zz beta' ]; % save iteration summaries

```

```

il = il+1; % update counter
end
% bn and likn are final est of beta and log-like. get inv info at MLE
covm = inv( X'*X );
end

% EOF
% EOF

% 2h
=====
clear
format compact
format short g
X=[1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00 1.00;
   3.36 2.88 3.63 3.41 3.78 4.02 4.00 4.23 3.73 3.85 3.97 4.51 4.54 5.00 5.00 4.72 5.00]';
y=[ 65 156 100 134 16 108 121 4 39 143 56 26 22 1 1 5 65]';
beta=[0 0]';
method='F';
[betahat,il,zz,likn,covm,ithist] = mle_exponential_regression(y,X,beta,method); ithist
method='N';
[betahat,il,zz,likn,covm,ithist] = mle_exponential_regression(y,X,beta,method); ithist

0 2 F 5 -14770 1.0669 0.0040794 8.4778 -1.1093
0 2 N 85 -14770 1.9999 9.8951e-005 8.4767 -1.1091
0 0 F 3 -14770 0.18295 0.88936 7.5985 -1.1093
0 0 N 80 -14770 1.9999 0.00043815 8.4742 -1.1092
10 10 F 1 -Inf -1.0001 4.2464e+027 4.0983e+027 -1.1117e+027
10 10 N 99 -14770 1.9999 0.11745 -34.767 (slowly converging to 8.5) -1.1097

beta=[8.4742 ; -1.1092]; % MLE of beta parameters
x=2.8:.01:5.1; % range of x to plot
y_hat= exp((beta(1)+(x).*beta(2))); % fitted values of y
plot(X(:,2),y,'.', x,y_hat,'-'); % plot it
xlabel('log_{10}(initial white blood cell count)');ylabel('Survival time (weeks)');
title('Exponential regression fit to leukemia survival data')
plot_name = strcat('sc2hw04-2h.eps'); % plot name
print(gcf, '-depsc2', plot_name); % print plot

% 2i
=====
%NR
%beta0=-15:.1:15;
%beta1=-15:.1:15;
beta0=-200:4:200;
beta1=-5:.1:5;
betamap = zeros(length(beta0),length(beta1));
zz = zeros(length(beta0),length(beta1));
for i_beta0 = beta0;
    for i_beta1 = beta1;
        ii_beta0=find(beta0==i_beta0);
        ii_beta1=find(beta1==i_beta1);
        beta=[i_beta0;i_beta1];
        % NR
        del = (-X'*diag(y'*exp(-X*beta))*X) \ X'*diag(exp(-X*beta))*(y-exp(X*beta));
        betan = beta - del;
        zz(ii_beta0,ii_beta1) = sqrt( (beta-betan)'*(beta-betan) );
        betamap(ii_beta0,ii_beta1) = log_likelihood_exponential_regression(y,X,beta);
    end;
end;
figure(1);clf;
surf(beta0,beta1,betamap');
xlabel('\beta_0');
ylabel('\beta_1');
zlabel('log-likelihood');
figure(2);clf;
surf(beta0,beta1,zz');
xlabel('\beta_0');
ylabel('\beta_1');
zlabel('zz');

% surface plot of the log-likelihood
plot_name = strcat('sc2hw04-2i1.eps'); % plot name
print(gcf, '-depsc2', plot_name); % print plot
% rotated surface plot of the log-likelihood
plot_name = strcat('sc2hw04-2i2.eps'); % plot name
print(gcf, '-depsc2', plot_name); % print plot
% top-down surface plot of the log-likelihood

```

```

    plot_name = strcat('sc2hw04-2i3.eps'); % plot name
    print(gcf, '-depsc2', plot_name); % print plot
% surface plot of the zz
    plot_name = strcat('sc2hw04-2i4.eps'); % plot name
    print(gcf, '-depsc2', plot_name); % print plot

%% Exercise 3 =====
x=[342 500 187];
thetahat = (x(2)+2*x(3))/(2*(sum(x)))
% 3a =====
% nonparametric bootstrap
rand('state',2653);
MN = [1.*ones(342,1); 2.*ones(500,1); 3.*ones(187,1)]; % create a vector of M=1, MN=2, N=3
n = length(MN); % total number n
B = 10000; % number of bootstrap samples
ind = unidrnd(n,n,B); % sample indicies from discrete uniform
bs = MN(ind); % bootstrap sample
bs_x = [sum((bs==1)); sum((bs==2)); sum((bs==3))]; % determine counts
bs_thetahat = (bs_x(2,:)+2.*bs_x(3,:))/(2.*(sum(bs_x))); % calculate MLE
subplot(2,1,1)
hist(bs_thetahat,25) % plot histogram
axis([.37 .47 0 1500])
xlabel('Nonparametric Bootstrap sampling distribution of MLE of \theta')
%plot_name = strcat('sc2hw04-3a.eps'); % plot name
%print(gcf, '-depsc2', plot_name); % print plot

% 3b =====
% parametric bootstrap
x=[342 500 187];
B = 10000; % number of bootstrap samples
bs_x=multigen(sum(x),x/sum(x),B)'; % draw from multinomial distribution
bs_thetahat = (bs_x(2,:)+2.*bs_x(3,:))/(2.*(sum(bs_x))); % calculate MLE
subplot(2,1,2)
hist(bs_thetahat,25) % plot histogram
axis([.37 .47 0 1500])
xlabel('Parametric Bootstrap sampling distribution of MLE of \theta')
%plot_name = strcat('sc2hw04-3b.eps'); % plot name
%print(gcf, '-depsc2', plot_name); % print plot
plot_name = strcat('sc2hw04-3.eps'); % plot name
print(gcf, '-depsc2', plot_name); % print plot

```