Jade Radoni
STAT 641
HW #6
11/8/2021

Casus H.O.11, Devore Chp 7.9, 15.3:

1.) (1) Seeded: $\mathcal{U}^{*}_{0.95,0.95} = 2284.993$

Unseeded: $\mathcal{U}^{*}_{.95,.95} = 458.5596$

(2) Seeded 95% CI: (284.8813, 782.4843)

Unseeded 95% CI: (273.6416, 326.8917)

(3) Seeded 95% CI: (181.9196, 367.3327)

Unseeded 95% CI: (275.375, 328.625)

(4) The seeding seems to increase the variance of the amount of rainfall. In particular the seeding seems to generate more extreme rainfall events.

2) (1) (i) Asymptotic 95% CI: (2.7720, 2.8950)

(ii) Studentized bootstrap 95% CI: (2.828936, 2.833595)

(2) Parametric Bootstrap 95% CI: (2.775503, 2.890096)

3.) (1). No, I do not agree w/ the statisticians results. One of the conditions for the chi-squared GOF test to be valid is that All $E_i > 1$ (14.0.9, pg 3). Looking at the table given, we see that $E_1 = 0.12 < 1$. Thus the chi-squared GOF test isn't valid and we need to combine the first two groups.

	0-15	16-20	21-24	25-27	28-30	31+	Total
O_i	3	36	52	50	39	20	200
E_i	2.55	34.62	59.51	45.36	32.62	27.34	200
$\frac{(O_i-E_i)^2}{E_i}$	0.0794	0.06	0.53	0.48	1.25	1.97	4.3694

$P[\chi^2_4 \geq 4.3694] = 1 - \text{pchisq}(4.3494, 4) \doteq \boxed{0.3583153}$

$\Rightarrow$ Our data can be reasonably modeled by a Poisson distribution.

[continued]

(2) $P\left(-z_{\alpha/2} \le \frac{\sqrt{n}\,(\bar{y}-\lambda)}{\sqrt{\lambda}} \le z_{\alpha/2}\right)$

• $z_{\alpha/2} = \frac{\sqrt{n}\,(\bar{y}-\lambda)}{\sqrt{\lambda}} \iff z\sqrt{\lambda} = \sqrt{n}\,(\bar{y}-\lambda) \iff z\sqrt{\lambda} = \sqrt{n}\,\bar{y} - \sqrt{n}\,\lambda$

$\iff z^2\lambda = (\sqrt{n}\,\bar{y} - \sqrt{n}\,\lambda)^2$

$\iff z^2\lambda = n\bar{y}^2 - 2n\bar{y}\lambda + n\lambda^2$

$\iff 0 = n\bar{y}^2 - 2n\bar{y}\lambda + n\lambda^2 - z^2\lambda$

$\iff 0 = \underbrace{n}_{a}\lambda^2 - \underbrace{(2n\bar{y}+z^2)}_{b}\lambda + \underbrace{n\bar{y}^2}_{c}$

$$\lambda = \frac{-(-2n\bar{y}-z^2) \pm \sqrt{(-2n\bar{y}-z^2)^2 - 4n^2\bar{y}^2}}{2n}$$

$$= \frac{2n\bar{y}+z^2 \pm \sqrt{(-2n\bar{y}-z^2)^2 - (2n\bar{y})^2}}{2n}$$

* note: $x^2 - y^2 = (x+y)(x-y)$

$$= \frac{2n\bar{y}+z^2 \pm \sqrt{(-2n\bar{y}-z^2+2n\bar{y})(-2n\bar{y}-z^2-2n\bar{y})}}{2n}$$

$$= \frac{(2n\bar{y}+z^2) \pm \sqrt{(-z^2)(-4n\bar{y}-z^2)}}{2n}$$

$$= \frac{2n\bar{y}+z^2 \pm z\sqrt{4n\bar{y}-z^2}}{2n}$$

$$= \frac{2(200)(27.7) + \text{qnorm}(0.975)^2 \pm \text{qnorm}(0.975)\sqrt{4(200)(27.7) - \text{qnorm}(0.975)^2}}{2(200)}$$

$$\boxed{95\%\ \text{CI for } \lambda:\ (28.41975,\ 28.43895)}$$

4.) (1) Agresti-Coull: 95% CI: (0.1282745, 0.3390152)

(2) $L^*_{.95,.99} = 14.71807$

(3) 95% PI: (5.296746, 786.063146)

5.) (1) 95% CI: (0.8360288, 0.9853998)

(2) 95% CI: (32.6, 41.8)

(3) (19.43877, 71.07666)

6.) (1) D

(2) D

(3) C $\left[n = \frac{2.57829^2 (30)^2}{10^2} = 59.71407 \right]$

(4) B

(5) B

(6) C

(7) D

(8) D if were using parametric bootstrap (i.e. we know F) o.w. E.

```
R version 4.1.1 (2021-08-10) -- "Kick Things"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)

R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.

  Natural language support but running in an English locale

R is a collaborative project with many contributors.
Type 'contributors()' for more information and
'citation()' on how to cite R or R packages in publications.

Type 'demo()' for some demos, 'help()' for on-line help, or
'help.start()' for an HTML browser interface to help.
Type 'q()' to quit R.

> # rainfall data
> x_seeded <- c(151, 450, 124, 235, 357, 110, 302, 671, 118, 115, 275, 275, 2550, 243, 201, 199,
+           130, 119, 92, 91, 92, 98, 1650, 1200, 1180, 900, 700, 460, 340, 330)
> x_unseeded <- c(246, 268, 275, 348, 305, 311, 206, 279, 426, 269, 257, 299, 337, 329, 319, 312,

+           327, 342, 351, 205, 151, 426, 154, 353, 396, 441, 254, 263, 278, 281)
>
> # carbon fiber data
> x <- c(2.526, 2.546, 2.628, 2.669, 2.869, 2.710, 2.731, 2.751, 2.771, 2.772, 2.782,
+         2.789, 2.793, 2.834, 2.844, 2.854, 2.875, 2.876, 2.895, 2.916, 2.919, 2.957, 2.977,
+         2.988, 3, 3, 3, 3)
>
> # space shuttle data
> x <- c(.18, 3.1, 4.2, 6.0, 7.5, 8.2, 8.5, 10.3, 10.6, 24.2, 29.6, 31.7, 41.9, 44.1, 49.5,
+         50.1, 59.7, 61.7, 64.4, 69.7, 70.0, 77.8, 80.5, 82.3, 83.5, 84.2, 87.1, 87.3, 93.2,
+         103.4, 104.6, 105.5, 108.8, 112.6, 116.8, 118.0, 122.3, 123.5, 124.4, 125.4, 129.5,
+         130.4, 131.6, 132.8, 133.8, 137.0, 140.2, 140.9, 148.5, 149.2, 152.2, 152.9, 157.7,
+         160.0, 163.6, 166.9, 170.5, 174.9, 177.7, 179.2, 183.6, 183.8, 194.3, 195.1, 195.3,
+         202.6, 220.0, 221.3, 227.2, 251.0, 266.5, 267.9, 269.2, 270.4, 272.5, 285.9, 292.6,
+         295.1, 301.1, 304.3, 316.8, 329.8, 334.1, 346.2, 351.2, 353.3, 369.3, 372.3, 381.3,
+         393.5, 451.3, 461.5, 574.2, 656.3, 663.0, 669.8, 739.7, 759.6, 894.7, 974.9)
>
> # braided cord data
>
> x <- c(19.7, 21.6, 21.9, 23.5, 24.2, 24.4, 24.9, 25.1, 26.4, 26.9, 27.6, 27.7, 27.9,
+         28.4, 29.8, 30.7, 31.1, 31.1, 31.7, 31.8, 32.6, 34.0, 34.8, 34.9, 35.1, 36.6, 37.0,
+         37.7, 38.7, 38.7, 39.0, 39.6, 40.0, 41.4, 41.4, 41.8, 42.2, 43.5, 44.5, 45.0, 45.5,
+         45.9, 46.3, 46.7, 46.7, 47.0, 47.0, 47.4, 47.6, 48.6, 48.8, 57.9, 58.3, 67.9, 84.2,
+         97.3)
>
>
> # 1.)  An experiment was designed to evaluate whether or not rainfall can be increased by treat
ing
> #      clouds with silver iodide. Rainfall was measured from 60 clouds, of which 30 were chosen
 randomly
> #      to be seeded with silver iodide. The objective is to describe the effect that seeding ha
s on
> #      rainfall. The measurements are the amounts of rainfall in acre-feet from the 60 clouds.
>
>
> # NOTE: In what follows, you should first check whether the data are Normally distributed. If n
ot, apply
> #       a Box-Cox transformation
>
> x_seeded = sort(x_seeded)
>
> # carbon fiber data ()
> x_unseeded = sort(x_unseeded)
>
> ##### NOTE: x_seeded needs a boxcox transformation, x_unseeded is already normal
>
```

```
> y = x_seeded
> n = length(y)
> yt0 = log(y)
> s = sum(yt0)
> varyt0 = var(yt0)
> Lt0 = -1*s - .5*n*(log(2*pi*varyt0)+1)
> th = 0
> Lt = 0
> t = -3.01
> i = 0
> while(t < 3)
+ {t = t+.001
+ i = i+1
+ th[i] = t
+ yt = (y^t -1)/t
+ varyt = var(yt)
+ Lt[i] = (t-1)*s - .5*n*(log(2*pi*varyt)+1)
+ if(abs(th[i])<1.0e-10)Lt[i]<-Lt0
+ if(abs(th[i])<1.0e-10)th[i]<-0
+ }
>
> # The following outputs the values of the likelihood and theta and yields
> # the value of theta where likelihood is a maximum
> out = cbind(th,Lt)
> Ltmax= max(Lt)
> Ltmax
[1] -208.3935
> imax= which(Lt==max(Lt))
> thmax= th[imax]
> thmax
[1] -0.391
>
>
> iLtci = which(Ltmax - Lt <= 0.5 * qchisq(0.95, 1))
> iLtciL = min(iLtci)
> iLtciU = max(iLtci)
> thLci = th[iLtciL]
> thUci = th[iLtciU]
>
> # NOTE: b/c our 95% ci for theta max contains 0, I will use a log transformation instead.
>
> x_seeded_logtrans = log(x_seeded)
> shapiro.test(x_seeded_logtrans)

        Shapiro-Wilk normality test

data:  x_seeded_logtrans
W = 0.92486, p-value = 0.03592

>
>
> # (1) Place 95/95 lower tolerance intervals on the amount of rainfall amounts from both seeded
and unseeded
> #     clouds. If you had to transform one or both of the datasets, create a bound for the trans
formed data,
> #     then back-transform to get a bound on the original scale.
>
> # look at tables to get value of 2.22 (h.O 11 pg 45)
>
> upper_unseeded = mean(x_unseeded) + 2.22*sd(x_unseeded)
> upper_unseeded
[1] 458.5596
>
> upper_seeded = mean(x_seeded_logtrans) + 2.22*sd(x_seeded_logtrans)
> upper_seeded = exp(upper_seeded)
> upper_seeded
[1] 2284.993
>
>
> # (2) Place 95% confidence intervals on the average rainfall from both seeded and unseeded clou
```

```
ds. If you
> #     had to transform one or both of the datasets, use the studentized bootstrap, because our
confidence
> #     interval procedures for a mean are not appropriate for transformed data.
>
> tval_.975_df29 = 2.04523
> ub_unseeded = mean(x_unseeded) + tval_.975_df29*sd(x_unseeded)/sqrt(length(x_unseeded))
> lb_unseeded = mean(x_unseeded) - tval_.975_df29*sd(x_unseeded)/sqrt(length(x_unseeded))
> CI95_unseeded = c(lb_unseeded, ub_unseeded)
> CI95_unseeded
[1] 273.6416 326.8917
>
>
> n= length(x_seeded)
> thest = mean(x_seeded)
> V = var(x_seeded)/n
> B = 9999
> W = numeric(B)
> W = rep(0,times =B)
> for (i in 1:B)
+   W[i] = mean(sample(x_seeded,replace=T))
> Z = sqrt(n)*(W-thest)/W
> Z = sort(Z)
> LZ = Z[250]
> UZ = Z[9750]
> thL = thest-UZ*sqrt(V)
> thU = thest-LZ*sqrt(V)
> CI95_seeded = c(thL,thU)
> CI95_seeded
[1] 281.7764 788.9982
>
> # (3) Place 95% confidence intervals on the median rainfall from both seeded and unseeded cloud
s. Note
> #     that, since with the Normal distribution, the median equals the mean, you can just apply
confidence
> #     interval approaches for a mean. If you had to transform one or both of the datasets, go a
head and use
> #     the confidence interval approach for a mean and back-transform.
>
> ub_unseeded = median(x_unseeded) + tval_.975_df29*sd(x_unseeded)/sqrt(length(x_unseeded))
> lb_unseeded = median(x_unseeded) - tval_.975_df29*sd(x_unseeded)/sqrt(length(x_unseeded))
> CI95_unseeded = c(lb_unseeded, ub_unseeded)
> CI95_unseeded
[1] 275.375 328.625
>
>
> ub_seeded = median(log(x_seeded)) + tval_.975_df29*sd(log(x_seeded))/sqrt(length(log(x_seeded))
)
> ub_seeded = exp(ub_seeded)
> lb_seeded = median(log(x_seeded)) - tval_.975_df29*sd(log(x_seeded))/sqrt(length(log(x_seeded))
)
> lb_seeded = exp(lb_seeded)
>
> CI95_seeded = c(lb_seeded, ub_seeded)
> CI95_seeded
[1] 181.9196 367.3327
>
>
>
> # (4) What can you conclude about the effect of the seeding on the amount of rainfall?
>
> # It increases the variance of the rainfall.
>
>
>
> # 2.) Twenty-eight bundles of impregnated carbon fibers of length 20 mm are exposed to graduall
y
> #     increasing stress until they finally fail. The stress at failure are recorded as follows.
 The maximum stress
> #     that can be applied to the fibers is 3 and four of the fibers had not failed at that stre
```

```
ss so a value of 3 was
> #     assigned to the four fibers:
> library(survival)
> library(MASS)
>
> x <- c(2.526, 2.546, 2.628, 2.669, 2.869, 2.710, 2.731, 2.751, 2.771, 2.772, 2.782,
+        2.789, 2.793, 2.834, 2.844, 2.854, 2.875, 2.876, 2.895, 2.916, 2.919, 2.957, 2.977,
+        2.988, 3, 3, 3, 3)
> length(x)
[1] 28
>
> # 1.) Estimate with a 95% confidence interval the average stress to failure for the carbon fibe
rs without
> #     specifying the distribution of the stress to failure values. Do this two ways: (i) using
an asymptotic
> #     CI based on the results of the R function survfit, and (ii) using the studentized bootstr
ap, treating
> #     the censored observations as true stress values (i.e., ignoring the censoring).
>
> # (i)
> xcens = c(rep(1,times = length(x) - 4), rep(0, times = 4))
>
> Surv(x, xcens)
 [1] 2.526  2.546  2.628  2.669  2.869  2.710  2.731  2.751  2.771  2.772
[11] 2.782  2.789  2.793  2.834  2.844  2.854  2.875  2.876  2.895  2.916
[21] 2.919  2.957  2.977  2.988  3.000+ 3.000+ 3.000+ 3.000+
>
> cords.surv <- survfit(Surv(x, xcens) ~ 1,conf.type="log-log")
> summary(cords.surv)
Call: survfit(formula = Surv(x, xcens) ~ 1, conf.type = "log-log")

 time n.risk n.event survival std.err lower 95% CI upper 95% CI
 2.53     28       1    0.964  0.0351       0.7724        0.995
 2.55     27       1    0.929  0.0487       0.7435        0.982
 2.63     26       1    0.893  0.0585       0.7036        0.964
 2.67     25       1    0.857  0.0661       0.6629        0.944
 2.71     24       1    0.821  0.0724       0.6230        0.921
 2.73     23       1    0.786  0.0775       0.5840        0.898
 2.75     22       1    0.750  0.0818       0.5461        0.872
 2.77     21       1    0.714  0.0854       0.5091        0.846
 2.77     20       1    0.679  0.0883       0.4732        0.818
 2.78     19       1    0.643  0.0906       0.4381        0.789
 2.79     18       1    0.607  0.0923       0.4039        0.760
 2.79     17       1    0.571  0.0935       0.3706        0.729
 2.83     16       1    0.536  0.0942       0.3381        0.698
 2.84     15       1    0.500  0.0945       0.3064        0.666
 2.85     14       1    0.464  0.0942       0.2756        0.633
 2.87     13       1    0.429  0.0935       0.2457        0.600
 2.88     12       1    0.393  0.0923       0.2167        0.565
 2.88     11       1    0.357  0.0906       0.1886        0.530
 2.90     10       1    0.321  0.0883       0.1615        0.493
 2.92      9       1    0.286  0.0854       0.1354        0.456
 2.92      8       1    0.250  0.0818       0.1106        0.418
 2.96      7       1    0.214  0.0775       0.0871        0.378
 2.98      6       1    0.179  0.0724       0.0651        0.337
 2.99      5       1    0.143  0.0661       0.0450        0.295
> print(cords.surv,print.rmean=TRUE)
Call: survfit(formula = Surv(x, xcens) ~ 1, conf.type = "log-log")

         n     events     *rmean *se(rmean)     median    0.95LCL    0.95UCL
   28.0000    24.0000     2.8311     0.0249     2.8490     2.7720     2.8950
    * restricted mean with upper limit =  3
>
> # (ii)
> n= length(x)
> thest = mean(x)
> V = var(x)/n
> B = 9999
> W = numeric(B)
> W = rep(0,times =B)
```

```
> for (i in 1:B)
+   W[i] = mean(sample(x,replace=T))
> Z = sqrt(n)*(W-thest)/W
> Z = sort(Z)
> LZ = Z[250]
> UZ = Z[9750]
> thL = thest-UZ*sqrt(V)
> thU = thest-LZ*sqrt(V)
> CI95_x = c(thL,thU)
> CI95_x
[1] 2.828914 2.833541
>
> # (2) Estimate with a 95% confidence interval the average stress to failure for the carbon fibe
rs assuming
> #      the distribution of the stress to failure values has a Weibull distribution. Do this usi
ng the parametric
> #      bootstrap. To estimate the Weibull parameters, use the survreg function
>
> fit <- survreg(Surv(x, xcens) ~ 1, dist = "weibull")
> shape_est <- 1 / fit$scale
> scale_est <- exp(fit$coef)
>
> mu = scale_est*gamma(1+1/shape_est)
> mu
(Intercept)
   2.835246
> sigma = sqrt((scale_est^2)*(gamma(1+2/shape_est) - gamma(1+1/shape_est)^2))
> sigma
(Intercept)
  0.1549954
>
>
>
> aD = shape_est
> bD = scale_est
>
> n = length(x)
> B = 9999
> W = matrix(0,B,n)
> cv = numeric(B)
> cv = rep(0,B)
> a = numeric(B)
> a = rep(0,B)
> b = numeric(B)
> b = rep(0,B)
> mleest = matrix(0,B,2)
> {
+   for (i in 1:B)
+     W[i,] = rweibull(n,shape = aD, scale = bD)
+ }
> {
+   for (i in 1:B)
+     mleest[i,] = coef(fitdistr(W[i,],"weibull"))
+ }
There were 50 or more warnings (use warnings() to see the first 50)
> a = mleest[,1]
> b = mleest[,2]
> mu = (b*gamma(1+1/a))
> R = sort(mu)
> L = R[250]
> U = R[9750]
> ci = c(L, U)
> ci
[1] 2.776550 2.890274
>
> mean(x)
[1] 2.831143
>
>
>
```

```
>
> #3.) The National Institute for Standards and Technology conducted a study to develop standards
> #    for asbestos concentration. Asbestos dissolved in water was spread on a filter, and punche
s of 3-mm
> #    diameter were taken from the filter and mounted on a transmission electron microscope. An
operator
> #    counted the number of asbestos fibers on each of 200 grid squares yielding the following c
ounts: (the
> #    researcher no longer had the original counts just the following grouped data and the mean
of the 200 counts
>
> 1-pchisq(4.3694,4)
[1] 0.3583153
>
> UB = (2*(200)*(27.7)+qnorm(.975)^2 + qnorm(.975)*sqrt(4*200*27.7-qnorm(.975)^2))/(400)
> LB = (2*(200)*(27.7)-qnorm(.975)^2 + qnorm(.975)*sqrt(4*200*27.7-qnorm(.975)^2))/(400)
>
> CI=c(LB,UB)
> CI
[1] 28.41975 28.43895
>
> # 4.) The space shuttle uses epoxy spherical vessels in an environment of sustained pressure. A
 study
> #     of the lifetimes of epoxy strands subjected to sustained stress was conducted. The data g
iving the lifetimes
> #     (in hours) of 100 strands tested at a prescribed level of stress is given in the followin
g table
>
>
> x <- c(.18, 3.1, 4.2, 6.0, 7.5, 8.2, 8.5, 10.3, 10.6, 24.2, 29.6, 31.7, 41.9, 44.1, 49.5,
+        50.1, 59.7, 61.7, 64.4, 69.7, 70.0, 77.8, 80.5, 82.3, 83.5, 84.2, 87.1, 87.3, 93.2,
+        103.4, 104.6, 105.5, 108.8, 112.6, 116.8, 118.0, 122.3, 123.5, 124.4, 125.4, 129.5,
+        130.4, 131.6, 132.8, 133.8, 137.0, 140.2, 140.9, 148.5, 149.2, 152.2, 152.9, 157.7,
+        160.0, 163.6, 166.9, 170.5, 174.9, 177.7, 179.2, 183.6, 183.8, 194.3, 195.1, 195.3,
+        202.6, 220.0, 221.3, 227.2, 251.0, 266.5, 267.9, 269.2, 270.4, 272.5, 285.9, 292.6,
+        295.1, 301.1, 304.3, 316.8, 329.8, 334.1, 346.2, 351.2, 353.3, 369.3, 372.3, 381.3,
+        393.5, 451.3, 461.5, 574.2, 656.3, 663.0, 669.8, 739.7, 759.6, 894.7, 974.9)
>
> # (1)  Estimate with a 99% confidence interval the probability that an epoxy strand subjected t
o the
> #      prescribed stress will survive for 300 hours. Use the Agresti-Coull approach.
>
> y_tild = length(which(x>=300)) + round((0.5*qnorm(.995)^2))
> n_tild = length(x)+round(qnorm(.995)^2)
> p_tild = y_tild/n_tild
> p_tild
[1] 0.2336449
>
> U_Agresti_Coull = p_tild + qnorm(.995)*sqrt(p_tild*(1-p_tild))/sqrt(n_tild)
> L_Agresti_Coull = p_tild - qnorm(.995)*sqrt(p_tild*(1-p_tild))/sqrt(n_tild)
>
> CI_Agresti_Coull = c(L_Agresti_Coull,U_Agresti_Coull)
> CI_Agresti_Coull
[1] 0.1282745 0.3390152
>
>
> # (2)  Estimate with 99% certainty the time, L.95, such that at least 95% of epoxy strands unde
r the prescribed
> #      stress would have lifetimes greater than L.95. You can assume that the lifetimes follow
an exponential
> #      distribution. (see pg 47 on H.O. 11)
>
>
>
> p = .95
> gamma = .99
> n = length(x)
> y_bar = mean(x)
>
> U_tol_bound = 1 - y_bar*(2*n/qchisq(1-gamma,2*n))*log(p)
```

```
> U_tol_bound
[1] 14.71807
>
>
> # (3)  Using the above data, predict with 95% certainty the lifetime of a strand subjected to t
he prescribed
> #      stress. Again, assume an exponential distribution.
>
> LB = y_bar*qf(1-.975,2,2*n)
> UB = y_bar*qf(.975,2,2*n)
> PI_95 = c(LB,UB)
> PI_95
[1]   5.296746 786.063146
>
>
> # 5.)  An experiment was conducted to determine the strength of a certain type of braided cord
after
> #      weathering. The strengths of 56 pieces of cord that had been weathered for 30 days were
investigated. The
> #      56 pieces of cord were placed simultaneously in a tensile strength device. The device ap
plies an increasing
> #      amount of force until the cord fails. The following strength readings (psi) are given be
low
>
>
> x <- c(19.7, 21.6, 21.9, 23.5, 24.2, 24.4, 24.9, 25.1, 26.4, 26.9, 27.6, 27.7, 27.9,
+        28.4, 29.8, 30.7, 31.1, 31.1, 31.7, 31.8, 32.6, 34.0, 34.8, 34.9, 35.1, 36.6, 37.0,
+        37.7, 38.7, 38.7, 39.0, 39.6, 40.0, 41.4, 41.4, 41.8, 42.2, 43.5, 44.5, 45.0, 45.5,
+        45.9, 46.3, 46.7, 46.7, 47.0, 47.0, 47.4, 47.6, 48.6, 48.8, 57.9, 58.3, 67.9, 84.2,
+        97.3)
>
>
>
> # (1)  The manufacturer of the cords would like to estimate the proportion of weathered cords h
aving tensile
> #      strength less than 50 psi. Provide a 95% confidence interval based on the information fr
om the failure
> #      data from the 56 cords.
>
>
> # Using Wald CI
>
> p_hat = length(which(x<=50))/length(x)
> n = length(x)
>
> UB = p_hat + qnorm(.975)*sqrt(p_hat*(1-p_hat))/sqrt(n)
> LB = p_hat - qnorm(.975)*sqrt(p_hat*(1-p_hat))/sqrt(n)
> CI_Phat = c(LB,UB)
> CI_Phat
[1] 0.8360288 0.9853998
>
> # (2)  The manufacturer is planning a sales campaign to promote its cord and would like to stat
e a tensile
> #      strength value for its cord. Provide the manufacturer with a 95% confidence interval tha
t would provide
> #      an estimate of the median tensile strength value for the weathered braided cord.
>
> # Use distribution free extimate for Q(u) (H.O. 11 pg 31)
>
> n=length(x)
> L=.95
> P=.5
> s=ceiling(n*P)-1
> r=floor(n*P)+1
> cov=0
> while(s<n-1 && r>1 && cov<L)
+ {s=s+1
+ cov=pbinom(s-1,n,P)-pbinom(r-1,n,P)
+ if(cov>=L) break;
+ r=r-1
```

```
+ cov=pbinom(s-1,n,P)-pbinom(r-1,n,P)
+ }
> r
[1] 21
> s
[1] 36
> cov
[1] 0.9559535
>
> x = sort(x)
>
> CI = c(x[r], x[s])
> CI
[1] 32.6 41.8
>
>
> # (3) In order to determine if the braided cord has tensile strength that falls within federal
specifications,
> #     the manufacturer wants to determine an interval of strength values, (TL, TU ), such that
the manufacturer
> #     would be 95% confident that the interval would contain at least 90% of all strength value
s for its
> #     braided cords
>
> shapiro.test(x)

        Shapiro-Wilk normality test

data:  x
W = 0.85279, p-value = 6.915e-06

> hist(x)
>
> # do box-cox transformation to normality
>
> y = x
> n = length(y)
> yt0 = log(y)
> s = sum(yt0)
> varyt0 = var(yt0)
> Lt0 = -1*s - .5*n*(log(2*pi*varyt0)+1)
> th = 0
> Lt = 0
> t = -3.01
> i = 0
> while(t < 3)
+ {t = t+.001
+ i = i+1
+ th[i] = t
+ yt = (y^t -1)/t
+ varyt = var(yt)
+ Lt[i] = (t-1)*s - .5*n*(log(2*pi*varyt)+1)
+ if(abs(th[i])<1.0e-10)Lt[i]<-Lt0
+ if(abs(th[i])<1.0e-10)th[i]<-0
+ }
>
> # The following outputs the values of the likelihood and theta and yields
> # the value of theta where likelihood is a maximum
> out = cbind(th,Lt)
> Ltmax= max(Lt)
> Ltmax
[1] -217.9497
> imax= which(Lt==max(Lt))
> thmax= th[imax]
> thmax
[1] -0.452
>
>
> iLtci = which(Ltmax - Lt <= 0.5 * qchisq(0.95, 1))
> iLtciL = min(iLtci)
```

```
> iLtciU = max(iLtci)
> thLci = th[iLtciL]
> thUci = th[iLtciU]
> thLci
[1] -1.149
> thUci
[1] 0.195
>
> # note that our 95% CI for thmax contains 0 => use log trans of the data.
> length(x)
[1] 56
> x = log(x)
> shapiro.test(x)

        Shapiro-Wilk normality test

data:  x
W = 0.96876, p-value = 0.1543

>
> # look at tables to get value of 1.99 (h.O 11 pg 45)
> UB = mean(x) + 1.999*sd(x)
> UB = exp(UB)
>
> LB = mean(x) - 1.999*sd(x)
> LB = exp(LB)
>
> TI_95 = c(LB,UB)
> TI_95
[1] 19.43877 71.07666

> >
```