hw3

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

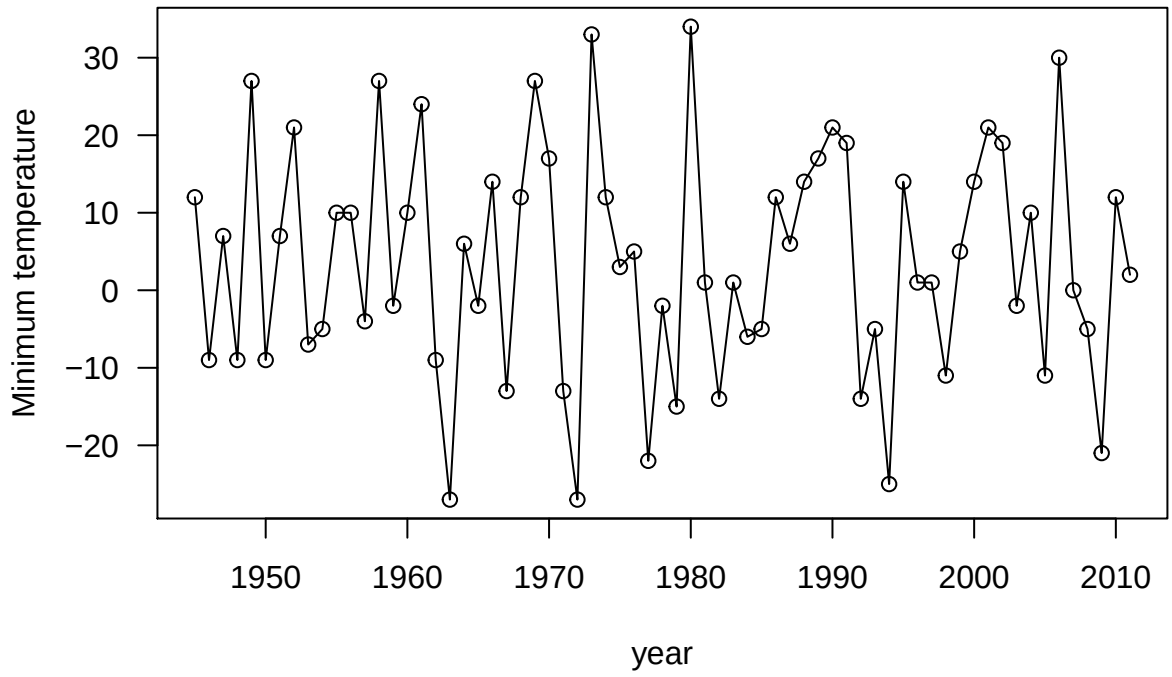
```
set.seed(3301)
dat = read.table("mspwinter.txt")
#1a) we must assume that the 67 minimum temperature measurements are realizations
#of random variables X1,...,X67 that are iid sequence of of random variables with some distribution.

#1a)We can also assume that the 65th minimuum temperature measurement of -21 degrees is assumed
#to be a realization of the unknown distribution which is
#the same for all the measurements with unknown mean mu and unknown variance

#1b) there is no visual evidence that the
#distribution of the response is changing over time.
dat$temp
```

```
##  [1]  12  -9   7  -9  27  -9   7  21  -7  -5  10  10  -4  27  -2  10  24  -9 -27
## [20]   6  -2  14 -13  12  27  17 -13 -27  33  12   3   5 -22  -2 -15  34   1 -14
## [39]   1  -6  -5  12   6  14  17  21  19 -14  -5 -25  14   1   1 -11   5  14  21
## [58]  19  -2  10 -11  30   0  -5 -21  12   2
```

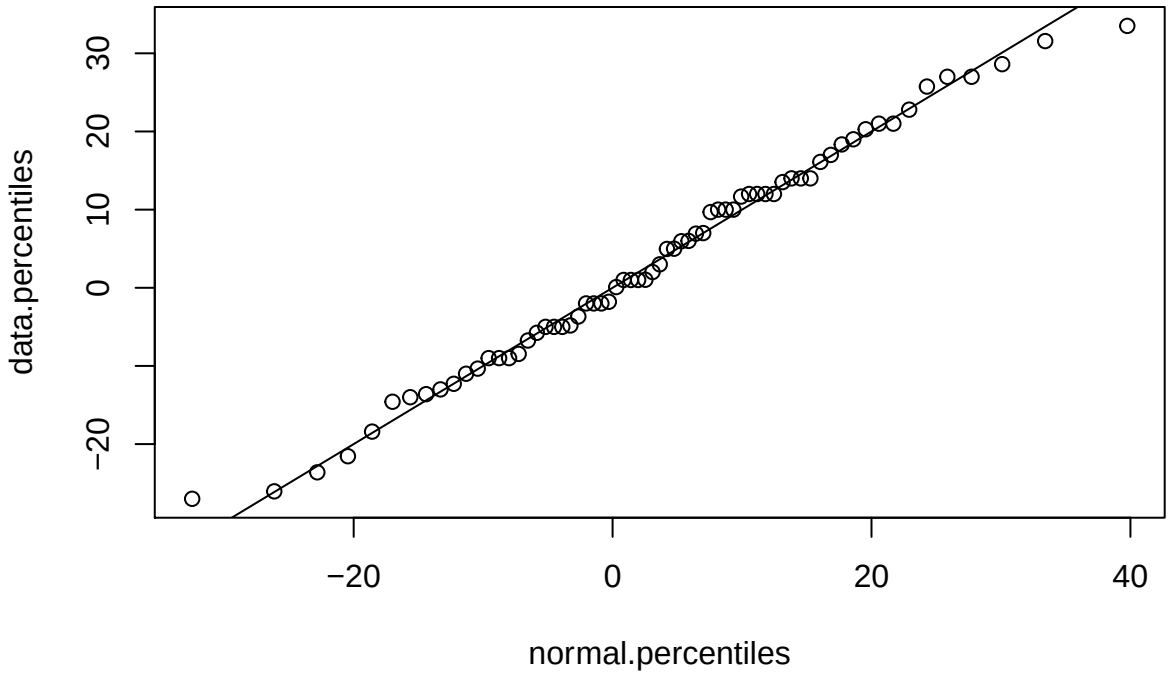
```
x.list = dat$temp
names(x.list) = dat$year
x.list
```

```
## 1945 1946 1947 1948 1949 1950 1951 1952 1953 1954 1955 1956 1957 1958 1959 1960
##   12   -9    7   -9   27   -9    7   21   -7   -5   10   10   -4   27   -2   10
## 1961 1962 1963 1964 1965 1966 1967 1968 1969 1970 1971 1972 1973 1974 1975 1976
##   24   -9  -27    6   -2   14  -13   12   27   17  -13  -27   33   12    3    5
## 1977 1978 1979 1980 1981 1982 1983 1984 1985 1986 1987 1988 1989 1990 1991 1992
##  -22   -2  -15   34    1  -14    1   -6   -5   12    6   14   17   21   19  -14
## 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003 2004 2005 2006 2007 2008
##   -5  -25   14    1    1  -11    5   14   21   19   -2   10  -11   30    0   -5
## 2009 2010 2011
##  -21   12    2
```

```
n=length(x.list)
xbar= mean(x.list)
plot(dat$year, dat$temp, type="o", las=1, xlab="year", ylab="Minimum temperature")
```




```
#1c)
probs=ppoints(length(x.list))
data.percentiles=quantile(x.list, probs)
normal.percentiles=qnorm(probs, mean=xbar, sd=sd(x.list))
plot(normal.percentiles, data.percentiles)
abline(0,1)
```




```
moe=qt(1-.01/2, n-1)*sd(x.list)/sqrt(n)
LB = xbar-moe #lower and upper bound for the mean of the temperatures
```

```
UB = xbar+moe
c(LB, UB)
```

```
## [1] -1.166076  8.449658
```

```
#check if the minimum temperature measurements have a normal distribution
```

```
#1d

#Yes, the interpretation is incorrect, because it is approximately 99% confident that the unknown mean
#distribution used to model the minimum temperature in the
#Twin Cities on January 15th is between -1.166076 and 8.44965 farhenheit
```

```
#1e
set.seed(3301); mu=4; sigma=15;alpha=.01
x.list1 = round(mu+sigma*rnorm(67))
```

```
#1e part1
xbar1= mean(x.list1)
moe1 = qt(1-alpha/2, 67-1)*sd(x.list1)/sqrt(67)
conf.int = xbar1+c(-1,1)*moe1
conf.int
```

```
## [1] 0.2461529 9.8732501
```

```
#1e part2
set.seed(3303)
mu =4; sigma=15; alpha=.01
n=67; reps = 5000
c.list = numeric(reps)
for(r in 1:reps){ #for more precision I simulted multiple reps
  captured.list = numeric(21)
  for(i in 1:21){
    r.list = round(mu+sigma*rnorm(67))
    rbar = mean(r.list)
    s = sd(r.list)
    conf.interval=rbar+c(-1,1)*qt(1-alpha/2,n-1)*s/sqrt(n)
    captured.list[i] = 1*(conf.interval[1]<mu)*(mu<conf.interval[2])
  }
  c.list[r] = mean(captured.list)
}
mean(c.list)
```

```
## [1] 0.9898762
```

```
#2a
set.seed(3301)
mu = 68; sigma= 3; theta=2; reps=5000; alpha=.01; n1=1000
captured.list3 = numeric(reps)
captured.list4 = numeric(reps)
x.list3 = numeric(n1)
```

```r
varx= sigma^2 + (theta^2)/3

w.list = mu + sigma*rnorm(n1)
q.list = runif(n=n1, min=-theta, max=theta)
x.list3= q.list+ w.list
xbar3 = mean(x.list3) #Xbar
s3= sd(x.list3)#sd(X)
mean.conf = xbar3 +c(-1,1) *qt(1-alpha/2, n-1)*s3/sqrt(n1)

exsq  = (x.list3 - 68)^2 #(X-68)^2
varbar = mean(exsq)#Exp{(X-68)^2}=variance(X)
varconf = varbar+c(-1,1) *qt(1-alpha/2,n1-1)*sd(exsq)/sqrt(n1)
print("99% approx confidence interval of E(X)")
```

```
## [1] "99% approx confidence interval of E(X)"
```

```r
mean.conf
```

```
## [1] 67.88471 68.42214
```

```r
print("99% approx confidence interval of Var(X)")
```

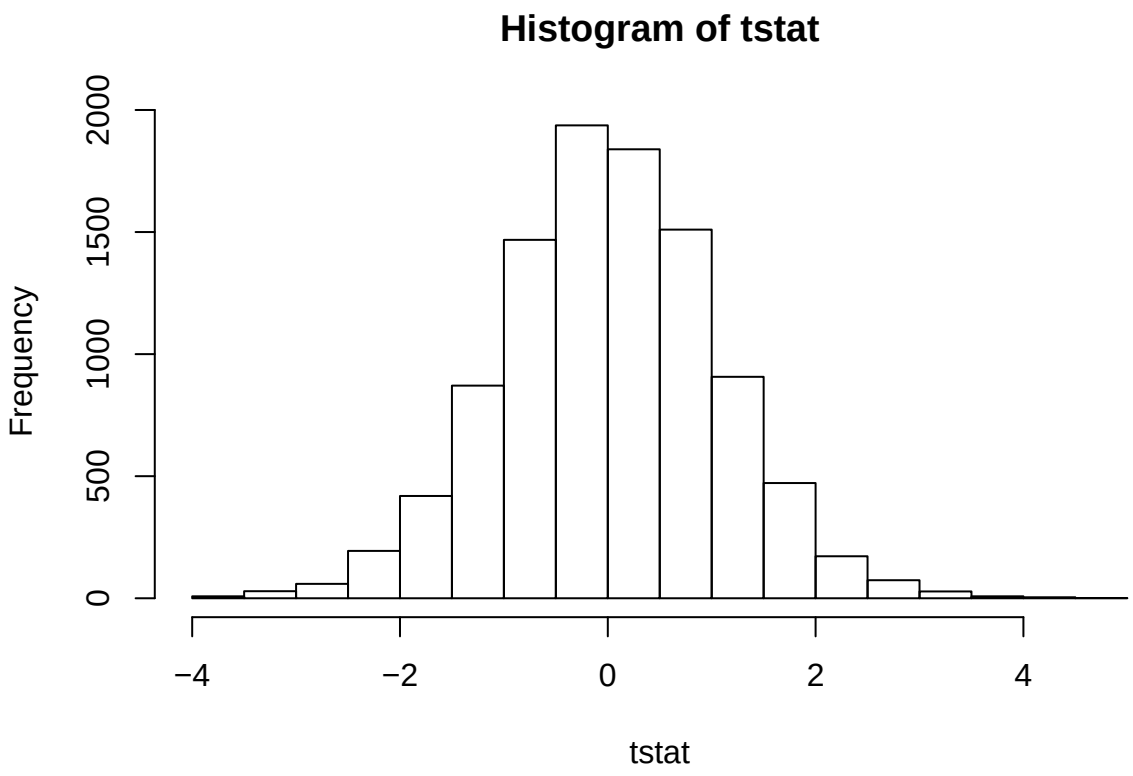
```
## [1] "99% approx confidence interval of Var(X)"
```

```r
varconf
```

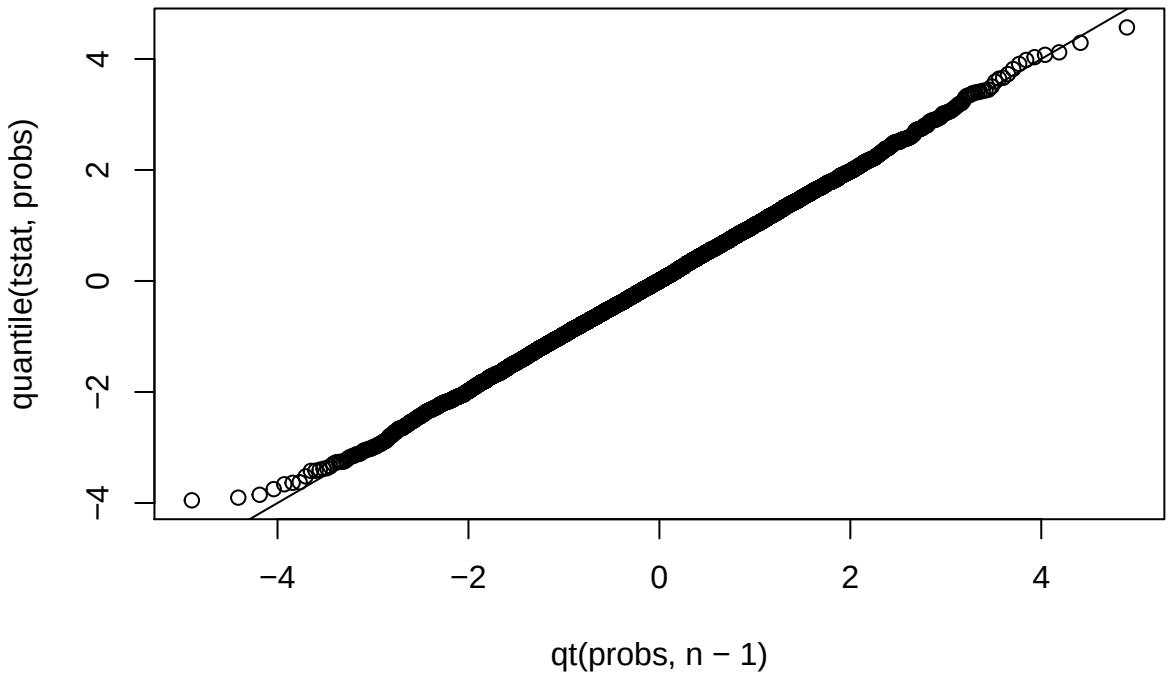
```
## [1]  9.073905 11.479823
```

```r
# the actual values of the mean and the variance is captured
```

```r
#2b
set.seed(3301)
mu = 65; sigma= 3; theta=1.5; reps=10000; n=20; alpha=.01
x.list4= numeric(n)
tstat = numeric(reps)
for(i in 1:reps){
  w1.list = mu + sigma*rnorm(n)
  q1.list = runif(n=20, min=-theta, max=theta)
  x.list4 = w1.list + q1.list
  spart=0
  xbar4 = sum(x.list4, na.rm= TRUE)/n
  s = sqrt(sum((x.list4 -xbar4)^2, na.rm=TRUE)/(n-1))
  tstat[i]= sqrt(n)*(xbar4-mu)/s
}

hist(tstat)
```




```
probs = ppoints(reps)
plot( qt(probs, n-1), quantile(tstat, probs))
abline(0,1)
```




```
#I plotted a histogram of the distribution of T to check if it resembles a normal distribution
#the distribution of T is well approximated by the t-distribution.
```

```r
#2c
set.seed(3301)
mu0=68;mu =67; sigma=3; theta1=6; alpha1 =.01; reps = 1e3
est.pval=function(n, mu0, mu, sigma, reps)
{
  pvalue.list=numeric(reps)
  x.list5= numeric(n)
  for(r in 1:reps)
  {
    w2.list = mu + sigma*rnorm(n)
    q2.list = runif(n=n, min=-theta1, max=theta1)
    for( k in 1:n) {
      x.list5[k] = w2.list[k] + q2.list[k]
    }
    spart1=0
    xbar5 = sum(x.list5, na.rm= TRUE)/n
    for(j in 1:n){
      spart1 = spart1 + (x.list5[j] - xbar5)^2
    }
    s = sqrt(spart1/(n-1))
    t.list= sqrt(n)*(xbar5-mu0)/s
    pvalue.list[r] = 2*pt(-abs(t.list), n-1)
  }
  return(pvalue.list)
}
estpval = est.pval(n=254, mu0= 68, mu= 67, sigma= sigma, reps=1e3 )
power1 = mean(estpval<.01) #power of this test is ~ 80% when n =254
print("The power is")
```

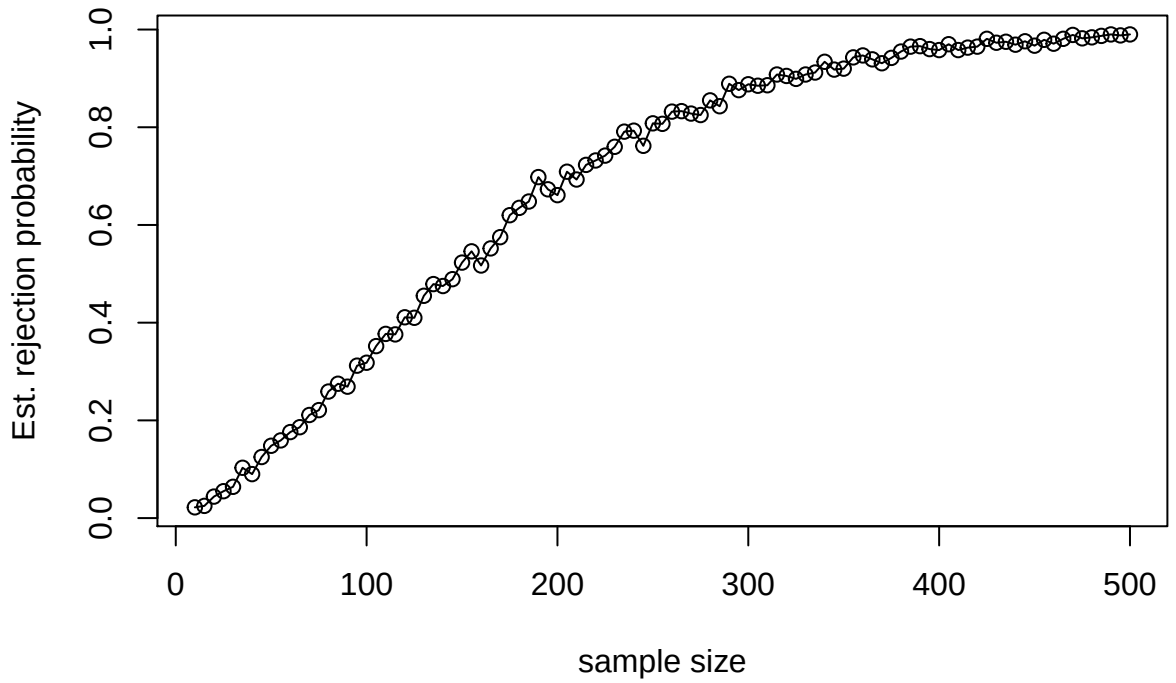
```
## [1] "The power is"
```

```r
power1
```

```
## [1] 0.803
```

```r
#estimate power vs n plotted to justify my solution
numpts =100
fast.est.power.ttest=function(n, mu0, alpha, mu, sigma, reps=1e3)
  {
  x.mat=matrix(mu+sigma*rnorm(reps*n) + runif(n= n*reps, min = -theta1, max=theta1), nrow=reps, ncol=n)
  xbar.list=apply(x.mat, 1, mean)
  s.list=apply(x.mat, 1, sd)
  t.list=(xbar.list - mu0)/(s.list/sqrt(n))
  prop.rejected = mean(abs(t.list) > qt(1-alpha/2, n-1))
return( prop.rejected )
}
set.seed(3301)
numpts = 100
n.seq = seq(from=10, to=500, by=5)
power.est = numeric(length(n.seq))
for(i in 1:length(n.seq))
{
```

```
  power.est[i] = fast.est.power.ttest(n=n.seq[i], mu0=68,alpha=.01,  mu=67, sigma=3,reps = 1e3)
}
plot(n.seq, power.est, type="o", xlab="sample size",
ylab="Est. rejection probability")
```




```
#the plot proves that around power of 80% the n size is 254
```

```
#2d
set.seed(3301)
n=180;mu0=68;mu=67;sigma=3;theta=6
alpha=0.01
score=0.05
reps = 30000
power.est = fast.est.power.ttest(n=180,mu0=68,alpha=0.01,mu=67,sigma=3,reps=30000)
s.ci = prop.test(x=reps*power.est,n=reps,correct=FALSE,conf.level=(1-score))$conf.int
s.ci
```

```
## [1] 0.6164978 0.6274710
## attr(,"conf.level")
## [1] 0.95
```