A simple analysis for the 'temperature' dataset

Yen-Chi Chen

12/5/2016

This dataset is from *All of Statistics* by Larry Wasserman. To start, we first read the data and use `head()` function to show the first few observations.

```
D <- read.table("temperature.txt", header=T, sep="\t")
head(D)
```

```
##                City JanTemp  Lat  Long
## 1        Mobile, AL      44 31.2  88.5
## 2    Montgomery, AL      38 32.9  86.8
## 3       Phoenix, AZ      35 33.6 112.5
## 4   Little Rock, AR      31 35.4  92.8
## 5   Los Angeles, CA      47 34.3 118.7
## 6 San Francisco, CA      42 38.4 123.0
```

To find out how many observations and variables in this dataset, we use `nrow()` and `ncol`:

```
nrow(D)
```

```
## [1] 56
```

```
ncol(D)
```

```
## [1] 4
```

Thus, there are 56 observations and 4 variables.

Generally, the next analysis we will do is to use the function `summary()` to give a brief summary of this dataset.

```
summary(D)
```

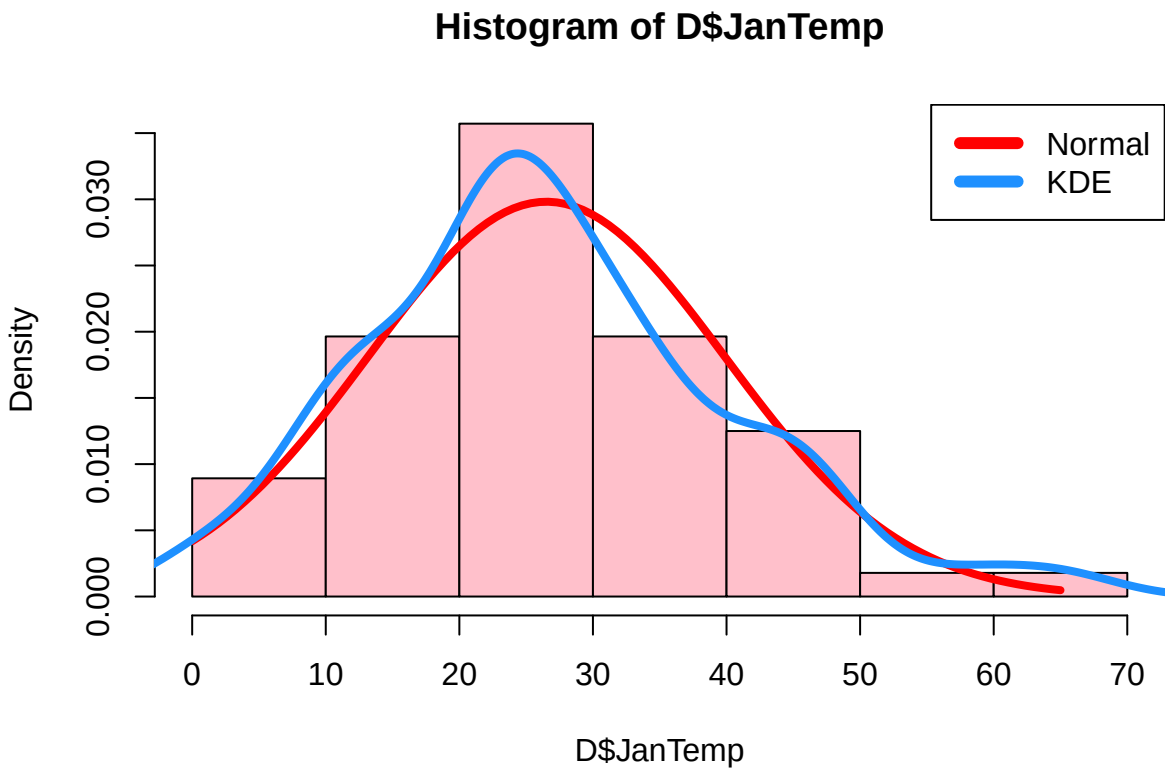
```
##                 City       JanTemp           Lat             Long       
##  Albany, NY       : 1   Min.   : 0.00   Min.   :25.00   Min.   : 70.50  
##  Albuquerque, NM  : 1   1st Qu.:18.75   1st Qu.:35.55   1st Qu.: 78.62  
##  Amarillo, TX     : 1   Median :24.50   Median :39.80   Median : 87.80  
##  Atlanta, GA      : 1   Mean   :26.52   Mean   :38.97   Mean   : 90.96  
##  Atlantic City, NJ: 1   3rd Qu.:33.25   3rd Qu.:42.62   3rd Qu.: 98.45  
##  Baltimore, MD    : 1   Max.   :65.00   Max.   :48.10   Max.   :123.20  
##  (Other)          :50
```

In particular, for the variables `JanTemp`, `Lat`, and `Long`, `summary()` gives an elegant summary of their distributions.

The variable `JanTemp`, temperature in January, is the most interesting one. So now we examine its histogram and attach density curves to it:

```
hist(D$JanTemp, probability=T,col="pink")
# add normal fit
x_seq <- seq(from=min(D$JanTemp), to=max(D$JanTemp),
             length.out=101)
y_seq <- dnorm(x_seq, mean=mean(D$JanTemp),
               sd=sd(D$JanTemp))
lines(x_seq,y_seq,lwd=4, col="red")
```

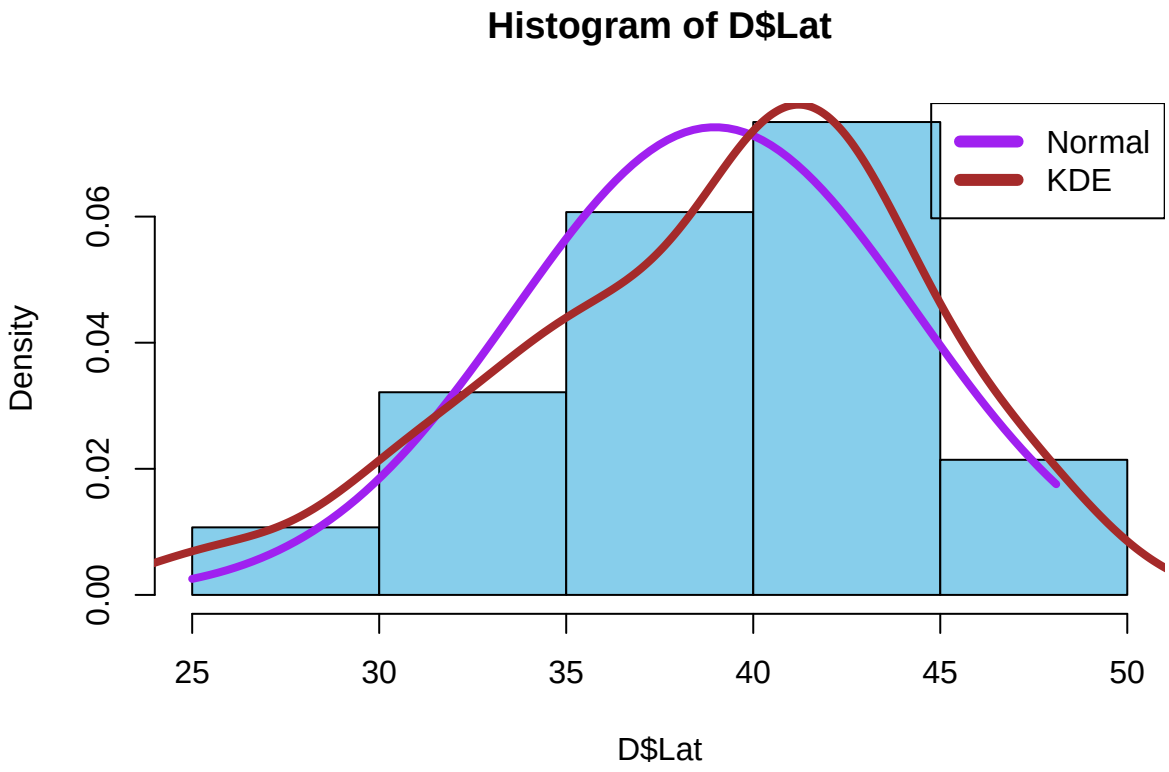
```r
# add KDE fit
lines(density(D$JanTemp), lwd=4, col="dodgerblue")
legend("topright",legend=c("Normal","KDE"),
       col=c("red","dodgerblue"), lwd=6)
```



It seems that the normal fit does a good job.

For the next one, we focus on variable `Lat` and try to show its histogram:

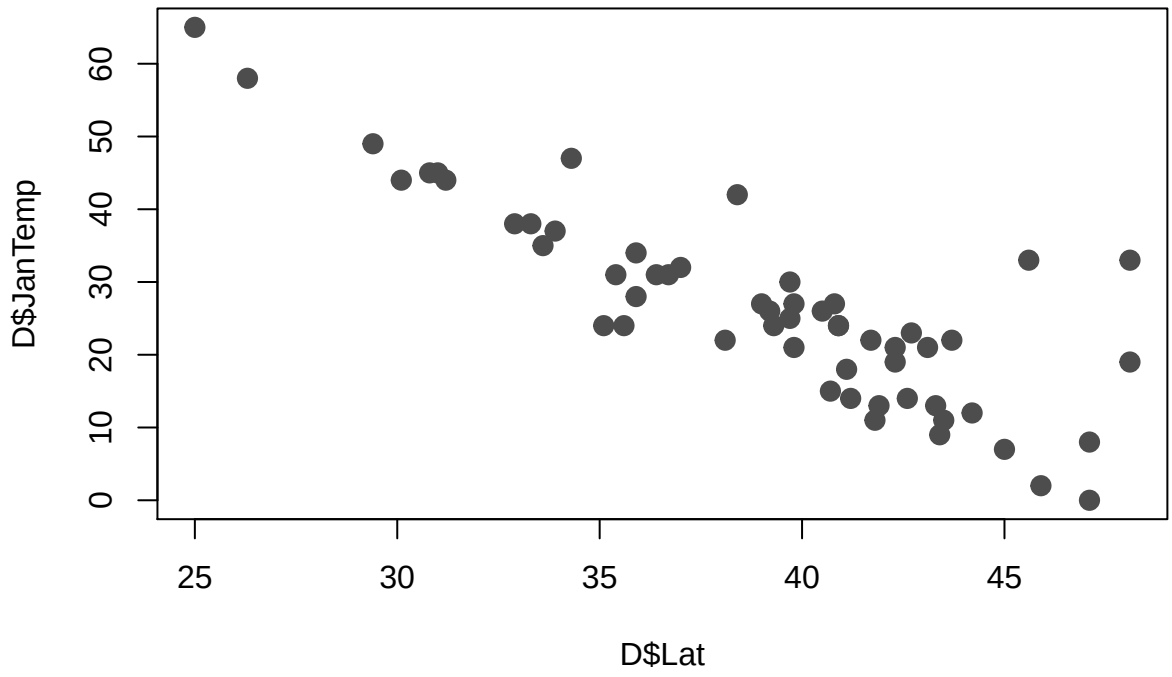
```r
hist(D$Lat, probability=T,col="skyblue")
x_seq <- seq(from=min(D$Lat), to=max(D$Lat),
             length.out=101)
y_seq <- dnorm(x_seq, mean=mean(D$Lat),
               sd=sd(D$Lat))
lines(x_seq,y_seq,lwd=4, col="purple")
lines(density(D$Lat), lwd=4, col="brown")
legend("topright",legend=c("Normal","KDE"),
       col=c("purple","brown"), lwd=6)
```



Again, it seems that the normal fit also works well.

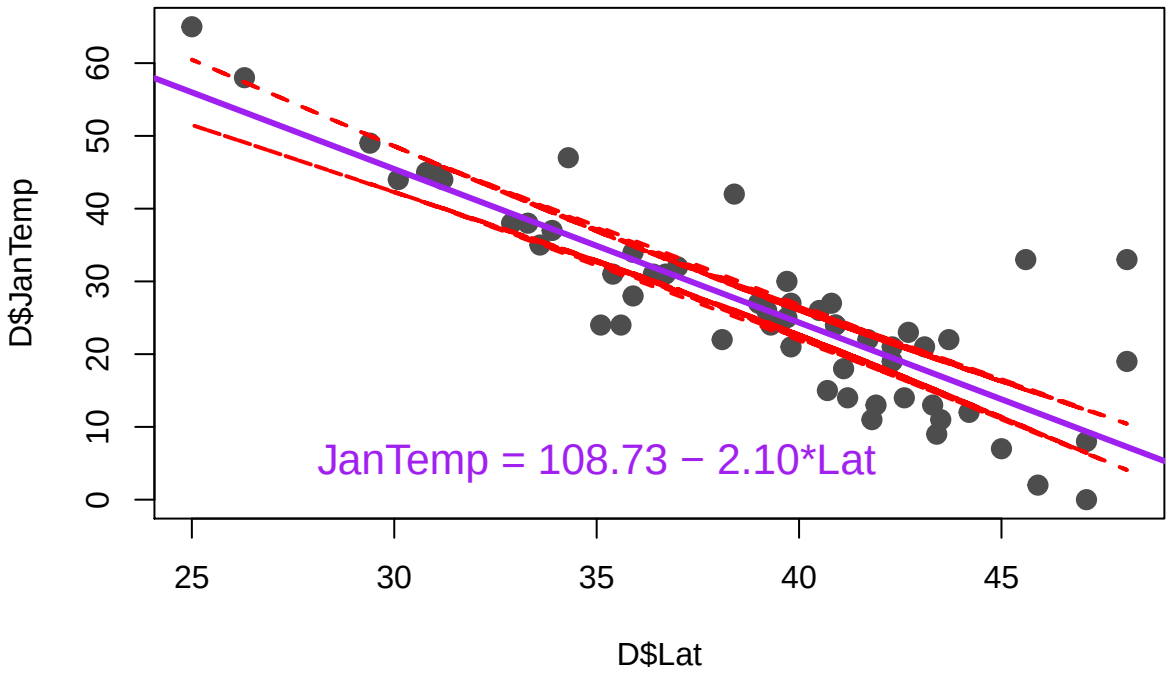
Now we examine the scatter plot of the two variables `Lat` and `JanTemp`:

```
plot(x= D$Lat,y=D$JanTemp, pch=20, cex=2,
     col="gray30")
```



We observe a clear decreasing pattern so now we try to fit a linear regression to it.

```
plot(x= D$Lat,y=D$JanTemp, pch=20, cex=2,
     col="gray30")
fit <- lm(JanTemp~Lat, data=D)
abline(fit, lwd=3, col="purple")
# adding a formula
text(x=35, y=5,labels="JanTemp = 108.73 - 2.10*Lat",
     cex=1.3, col="purple")
# adding a confidence band
prd<-predict(fit,newdata=D,interval =
               c("confidence"),
             level = 0.90,type="response")
lines(D$Lat,prd[,2],col="red",lty=2, lwd=2)
lines(D$Lat,prd[,3],col="red",lty=2, lwd=2)
```



We see a strange pattern in the confidence band. This is because the points are now ordered so when we use the `lines()` to connect points, it went back and forth on the x-axis. To handle this problem, we use the following procedure to order data points:

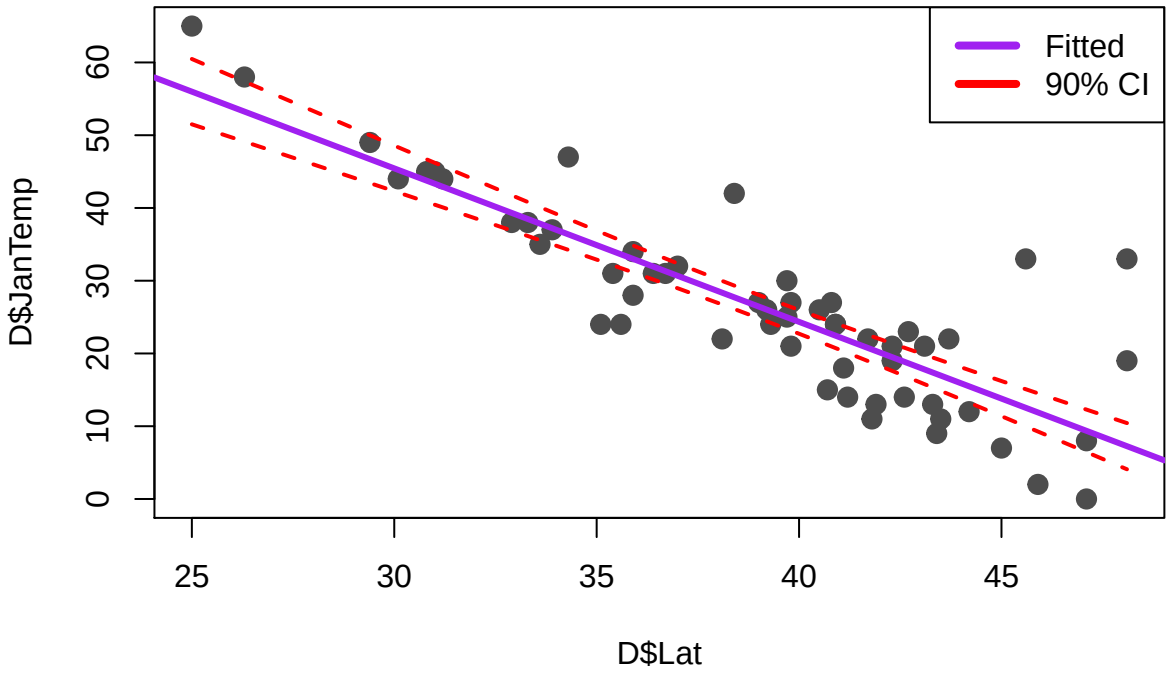
```
w = order(D$Lat)
D$Lat[w]
```

```
##  [1] 25.0 26.3 29.4 30.1 30.8 31.0 31.2 32.9 33.3 33.6 33.9 34.3 35.1 35.4
## [15] 35.6 35.9 35.9 36.4 36.7 37.0 38.1 38.4 39.0 39.2 39.3 39.7 39.7 39.8
## [29] 39.8 40.5 40.7 40.8 40.9 40.9 41.1 41.2 41.7 41.8 41.9 42.3 42.3 42.6
## [43] 42.7 43.1 43.3 43.4 43.5 43.7 44.2 45.0 45.6 45.9 47.1 47.1 48.1 48.1
```

Thus, we use `w` to order the points representing the confidence band:

```
# get the ordering
plot(x= D$Lat,y=D$JanTemp, pch=20, cex=2,
     col="gray30")
abline(fit, lwd=3, col="purple")
lines(D$Lat[w],prd[w,2],col="red",lty=2, lwd=2)
lines(D$Lat[w],prd[w,3],col="red",lty=2, lwd=2)
```

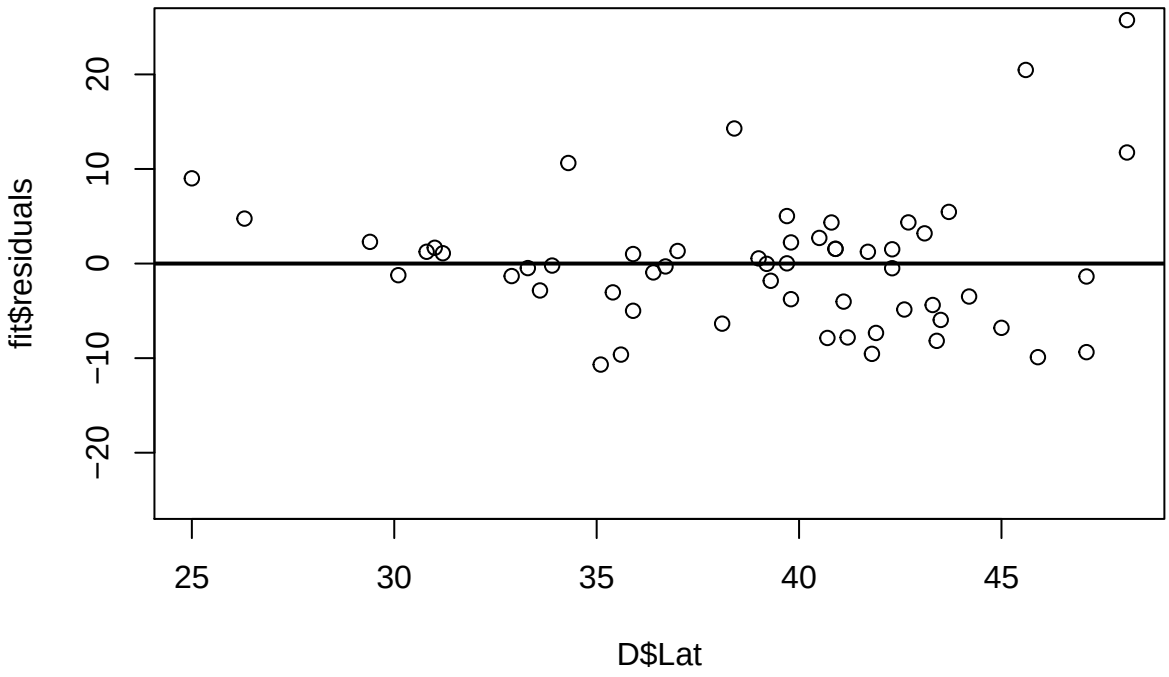
```
legend("topright",c("Fitted","90% CI"),
       col=c("purple","red"), lwd=4)
```



Now the confidence band looks very nice.

To make sure there is not hidden pattern the linear regression cannot capture, we examine the covariate versus residual plot:

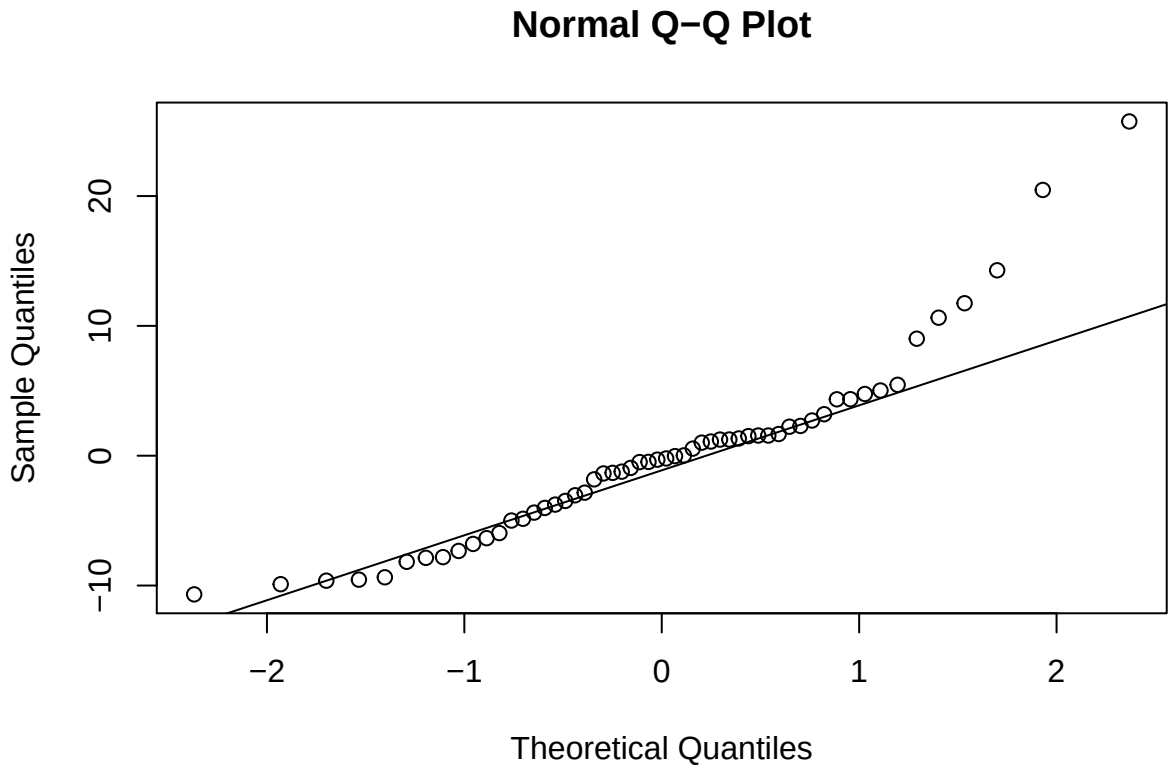
```
plot(x=D$Lat, y=fit$residuals, ylim=c(-25,25))
abline(h=0, lwd=2)
```



It does seem to be okay.

To test if the normality assumption on the errors are reasonable, we check the residual QQ plot:

```
qqnorm(fit$residuals)
qqline(fit$residuals)
```



Well, most points seem to follow the normal distribution but top right part is deviated from the normal line while the bottom left part is still on the QQline. This implies that the errors might be from a right-skew distribution.

Finally, we end this note on testing the null hypothesis

$$H_0 : \texttt{Slope} = 1.5.$$

Just recall that the function `summary()` applies to the regression fit object `fit` shows the details about the linear model:

```
summary(fit)
```

```
##
## Call:
## lm(formula = JanTemp ~ Lat, data = D)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -10.6812  -4.5018  -0.2593   2.2489  25.7434
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 108.7277     7.0561   15.41   <2e-16 ***
## Lat          -2.1096     0.1794  -11.76   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Residual standard error: 7.156 on 54 degrees of freedom
## Multiple R-squared:  0.7192, Adjusted R-squared:  0.714
## F-statistic: 138.3 on 1 and 54 DF,  p-value: < 2.2e-16
```

and the command

```
summary(fit)$coefficients
```

```
##               Estimate Std. Error   t value     Pr(>|t|)
## (Intercept) 108.727742  7.0561039  15.40903 2.034022e-21
## Lat          -2.109588  0.1793963 -11.75937 1.623964e-16
```

gives us a matrix about the fitted coefficients.

Using the Z-test, the test statistics T for the null hypothesis is

$$T = \frac{\hat{\beta} - 1.5}{\hat{\sigma}_\beta},$$

where $\hat{\beta}$ is the estimated slope and $\hat{\sigma}_\beta$ is the standard error of the estimated slope. By the central limit theory, T converges to a standard normal distribution. Thus, the p-value will be

$$2 \times P(Z \geq |T|),$$

where Z is a random variable follows the standard normal distribution. To compute the p-value of the Z-test, we use the following command

```
T_stats = (-1.5-summary(fit)$coefficient[2,1])/(summary(fit)$coefficient[2,2])
2*(1-pnorm(abs(T_stats)))
```

```
## [1] 0.0006788147
```

The first line computes the test statistics T and the second line computes the p-value. The p-value is pretty small, which means we do have a significant evidence to reject H_0 (under some common significance levels such as $\alpha = 0.05, 0.01$, or 0.001).