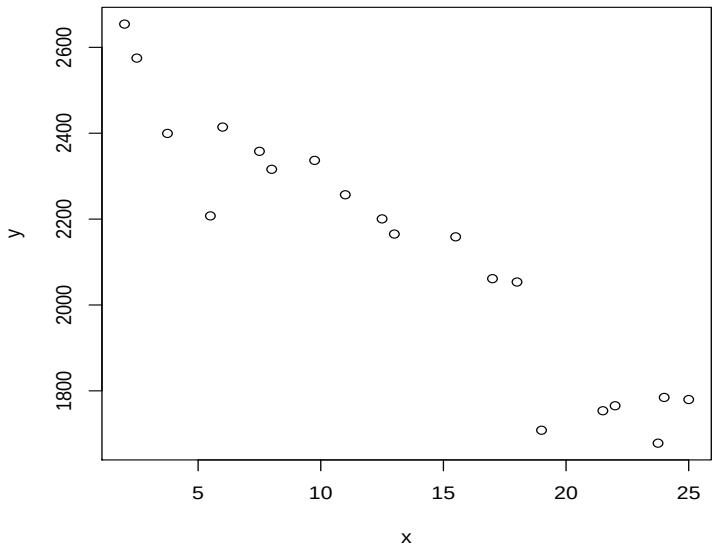
```
> #read in the data from a csv file
> dat1=read.table("Data Sets/Chapter 2/Examples/ex-2-1.csv", header=T, sep=",")
> #if the data has no header and is separated by space in a txt file then use
> #dat1=read.table("Data Sets/Chapter 2/Examples/ex-2-1.txt", sep=" ")
> str(dat1) #check the data

'data.frame':   20 obs. of  3 variables:
 $ Observation..i               : int  1 2 3 4 5 6 7 8 9 10 ...
 $ Shear.Strength..yi..psi.     : num  2159 1678 2316 2061 2208 ...
 $ Age.of.Propellant..xi..weeks.: num  15.5 23.8 8 17 5.5 ...

> dat1 #print out the dat

   Observation..i Shear.Strength..yi..psi. Age.of.Propellant..xi..weeks.
1               1                  2158.70                         15.50
2               2                  1678.15                         23.75
3               3                  2316.00                          8.00
4               4                  2061.30                         17.00
5               5                  2207.50                          5.50
6               6                  1708.30                         19.00
7               7                  1784.70                         24.00
8               8                  2575.00                          2.50
9               9                  2357.90                          7.50
10             10                  2256.70                         11.00
11             11                  2165.20                         13.00
12             12                  2399.55                          3.75
13             13                  1779.80                         25.00
14             14                  2336.75                          9.75
15             15                  1765.30                         22.00
16             16                  2053.50                         18.00
17             17                  2414.40                          6.00
18             18                  2200.50                         12.50
19             19                  2654.20                          2.00
20             20                  1753.70                         21.50

> y=dat1[,2] #assign the second column in dat1 to be variable y
> x=dat1[,3] #assign the third column in dat1 to be variable x
> plot(y~x) #plot the data
```




```
> fm1=lm(y~x)  #fit a simple linear model with E(y)=beta0+beta1x1
> summary(fm1) # the properties of the coefficients for the fitting line

Call:
lm(formula = y ~ x)

Residuals:
    Min      1Q  Median      3Q     Max
-215.98  -50.68   28.74   66.61  106.76

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 2627.822     44.184   59.48  < 2e-16
x            -37.154      2.889  -12.86 1.64e-10

Residual standard error: 96.11 on 18 degrees of freedom
Multiple R-squared: 0.9018, Adjusted R-squared: 0.8964
F-statistic: 165.4 on 1 and 18 DF,  p-value: 1.643e-10

> anova(fm1) #the analysis of variance table

Analysis of Variance Table

Response: y
          Df  Sum Sq Mean Sq F value    Pr(>F)
x          1 1527483 1527483  165.38 1.643e-10
Residuals 18  166255    9236
```

```
> resid(fm1) # residuals, vector for e

         1          2          3          4          5          6
106.758301 -67.274574 -14.593631  65.088687 -215.977609 -213.604131
         7          8          9         10         11         12
 48.563824  40.061618   8.729573  37.567141  20.374323 -88.946393
        13         14         15         16         17         18
 80.817415  71.175153 -45.143358  94.442278   9.499187  37.097528
        19         20
100.684823 -75.320154

> df.residual(fm1) # degree of freedom for the residual sum of squares

[1] 18

> coef(fm1)  # estimates for beta0 and beta1

(Intercept)           x
 2627.82236   -37.15359

> #assign some notations
> ybar=mean(y);ybar

[1] 2131.358

> xbar=mean(x);xbar

[1] 13.3625

> ei=resid(fm1)
> yhat=fitted(fm1)    #y.hat
> df=df.residual(fm1)
> print(cbind(x,y,yhat, ei))  #cbind: combine the vectors by column

       x       y     yhat          ei
1  15.50 2158.70 2051.942  106.758301
2  23.75 1678.15 1745.425  -67.274574
3   8.00 2316.00 2330.594  -14.593631
4  17.00 2061.30 1996.211   65.088687
5   5.50 2207.50 2423.478 -215.977609
6  19.00 1708.30 1921.904 -213.604131
7  24.00 1784.70 1736.136   48.563824
8   2.50 2575.00 2534.938   40.061618
9   7.50 2357.90 2349.170    8.729573
10 11.00 2256.70 2219.133   37.567141
11 13.00 2165.20 2144.826   20.374323
12  3.75 2399.55 2488.496  -88.946393
```

```
13 25.00 1779.80 1698.983   80.817415
14  9.75 2336.75 2265.575   71.175153
15 22.00 1765.30 1810.443  -45.143358
16 18.00 2053.50 1959.058   94.442278
17  6.00 2414.40 2404.901    9.499187
18 12.50 2200.50 2163.402   37.097528
19  2.00 2654.20 2553.515  100.684823
20 21.50 1753.70 1829.020  -75.320154
```

```
> sum(ei^2)       # Residual sum of squares
```

```
[1] 166254.9
```

```
> sum(ei^2)/df    # Residual Mean Squares
```

```
[1] 9236.381
```

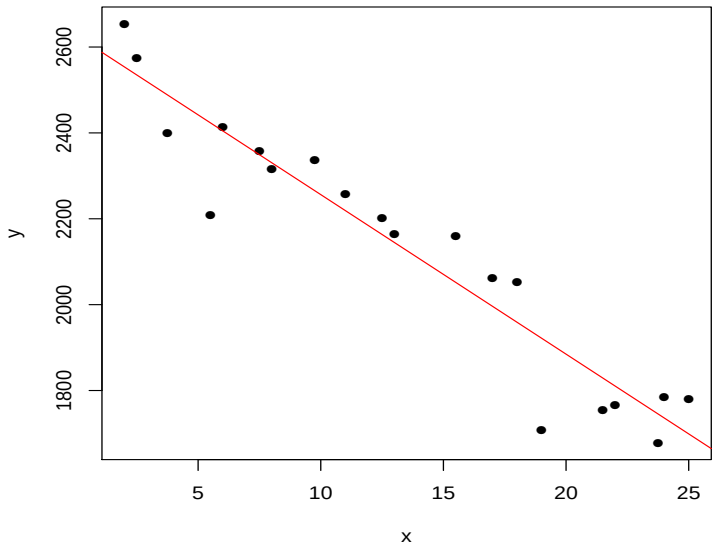
```
> s2.hat=sum(ei^2)/df
> sum((y-ybar)^2)
```

```
[1] 1693738
```

```
> sum((yhat-ybar)^2)
```

```
[1] 1527483
```

```
> #plot the data and the regression line
> fm1=lm(y~x)
> plot(y~x, pch=16)  #pch: change the symbol from o to solid dot
> abline(coef(fm1), col=2)
```



Confident interval for β_0 and β_1.

```
> coef(fm1)

(Intercept)           x
 2627.82236   -37.15359

> b0=coef(fm1)[1];b1=coef(fm1)[2]
> confint(fm1)

                2.5 %     97.5 %
(Intercept) 2534.99540 2720.64931
x            -43.22338  -31.08380
```

Done by hands

```
> Sxy=sum( (y-mean(y)) * (x-mean(x)) ); Sxy

[1] -41112.65

> Sxx=sum( (x-mean(x))^2 ); Sxx

[1] 1106.559

> se.b0=sqrt(s2.hat * (1/20+mean(x)^2/Sxx));se.b0

[1] 44.18391

> se.b1=sqrt(s2.hat/Sxx) ;se.b1

[1] 2.889107

> t.lim=qt(0.025,18)  # .025 quantile for t(18)
> c(b1 - t.lim * se.b1, b1 + t.lim * se.b1)

        x         x
-31.08380 -43.22338
```

Confident interval and prediction interval. use the command **predict()**

```
> fm1=lm(y~x)
> plot(y~x, xlab="age of Propellant, x", ylab="Shear Strength, y")
> abline(coef(fm1), col=2)  #col=2: change the colour of the line to be red
```

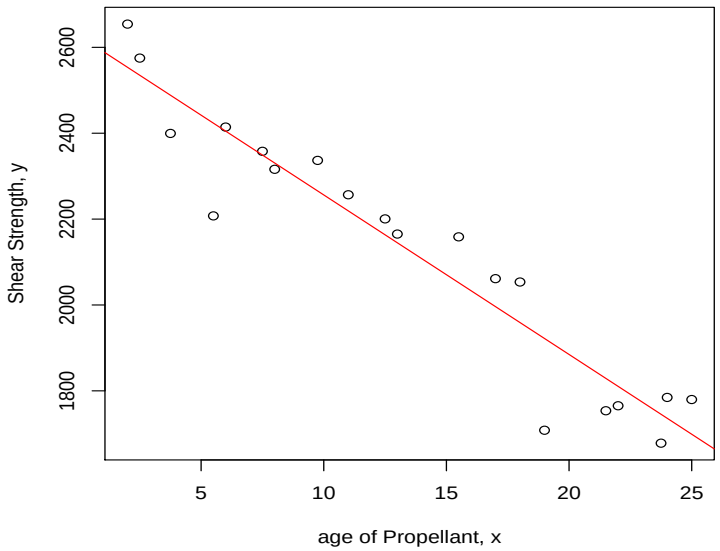


figure 2.4

predict(fm1, new, interval="confidence"): find the confidence interval for mean response for a set of given explanatory variables (say new).

Y.CI: matrix with 3 columns, one mean, lower limits and upper limits.

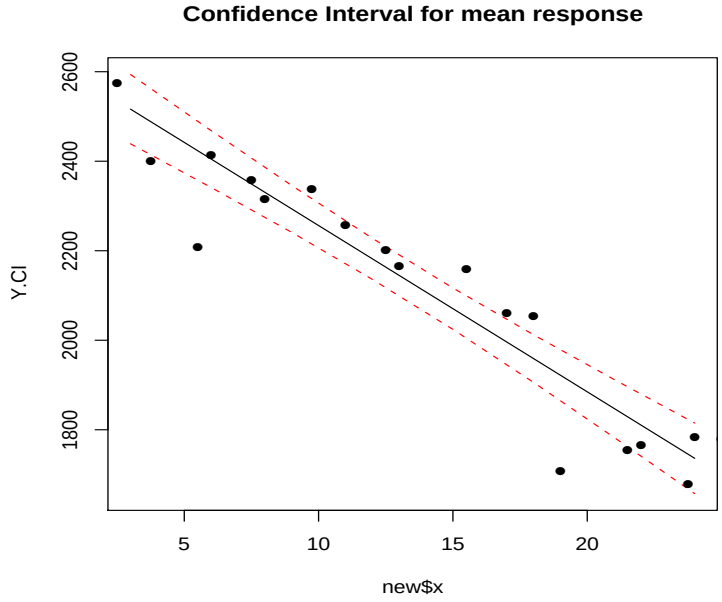
matplot: Plot the columns of one matrix against the columns of another. x vs Y.CI[,1], x vs Y.CI[,2], x vs Y.CI[,3]

type="l", plot type =line, lty: line type, col: line color.

```
> new = data.frame(x = seq(3, 24, 3)) #table 2.6
> predict(fm1, new, interval="confidence")

       fit      lwr      upr
1 2516.362 2438.937 2593.786
2 2404.901 2341.375 2468.426
3 2293.440 2241.099 2345.781
4 2181.979 2136.079 2227.879
5 2070.518 2024.289 2116.748
6 1959.058 1905.853 2012.263
7 1847.597 1782.886 1912.308
8 1736.136 1657.349 1814.923

> Y.CI=predict(fm1, new, interval="confidence");Y.CI  #assign the CIs to Y.CI

       fit      lwr      upr
1 2516.362 2438.937 2593.786
2 2404.901 2341.375 2468.426
3 2293.440 2241.099 2345.781
4 2181.979 2136.079 2227.879
```

```
5 2070.518 2024.289 2116.748
6 1959.058 1905.853 2012.263
7 1847.597 1782.886 1912.308
8 1736.136 1657.349 1814.923
```

```
> matplot(new$x, Y.CI, col=c(1,2,2), lty=c(1,2,2), type="l",
+     main="Confidence Interval for mean response")
> points(x,y,pch=16) #add data to the plot
```



predict(fm1, new, interval="prediction"): find the prediction interval for a set of new given explanatory variables (say new).

Y.PI: matrix with 3 columns, one mean, lower limits and upper limits.

```
> Y.PI=predict(fm1, new, interval="prediction");Y.PI
```

```
       fit      lwr      upr
1 2516.362 2300.114 2732.609
2 2404.901 2193.232 2616.570
3 2293.440 2084.855 2502.025
4 2181.979 1974.916 2389.042
5 2070.518 1863.382 2277.655
6 1959.058 1750.254 2167.861
7 1847.597 1635.569 2059.625
8 1736.136 1519.398 1952.875
```

```
> matplot(new$x, Y.PI, col=c(1,4,4), lty=c(1,4,4), type="l",
+     main="Prediction Interval for new obs")
> points(x,y,pch=16) #add data to the plot
```

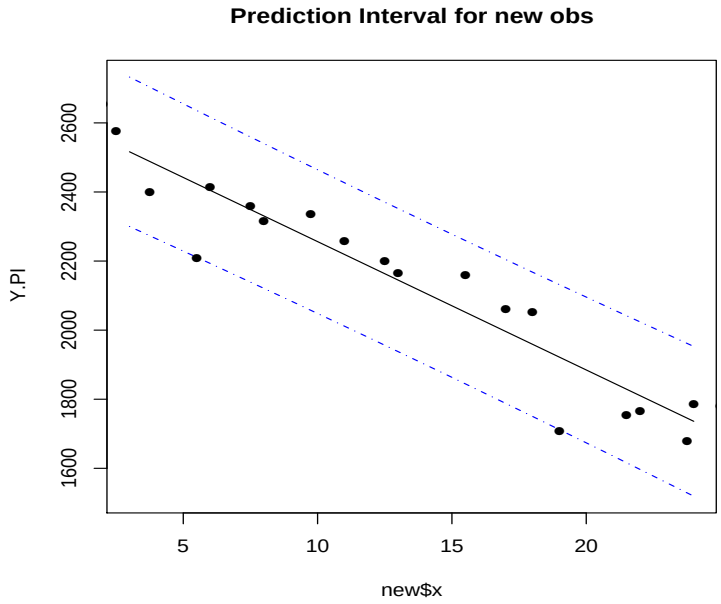


figure 2.5
add legend to the plot on the top-right corner.

```
> Y.mat=cbind(Y.CI, Y.PI[,-1])  #Y.mat, mean, lower.CI, upper.CI, lower.PI, upper.PI
> matplot(new$x, Y.mat, col=c(1,2,2,4,4), lty=c(1,2,2,4,4),
    type="l", xlab="x", ylab="response")
> points(x,y, pch=16) #add data to the plot
> legend("topright", legend=c("confint","pred.int"), col=c(2,4), lty=c(2,4))
```