# Checking Assumptions

September 29, 2017

## checking the assumption

Building a linear regression model is only half of the work. In order to actually be usable in practice, the model should conform to the assumptions of linear regression.

- **Assumption 1:** The regression model is linear in parameters
- **Assumption 2:** The mean of residuals is zero.
  Check the mean of the residuals. If it zero (or very close), then this assumption is held true for that model.

```
mean(linearMod$residuals)
[1] 1.23764e-17
```

- **Assumption 3:** Homoscedasticity of residuals or equal variance
  plot the model using *plot*(*lm.mod*). This produces four plots. The top-left and bottom-left plots shows how the residuals vary as the fitted values increase.

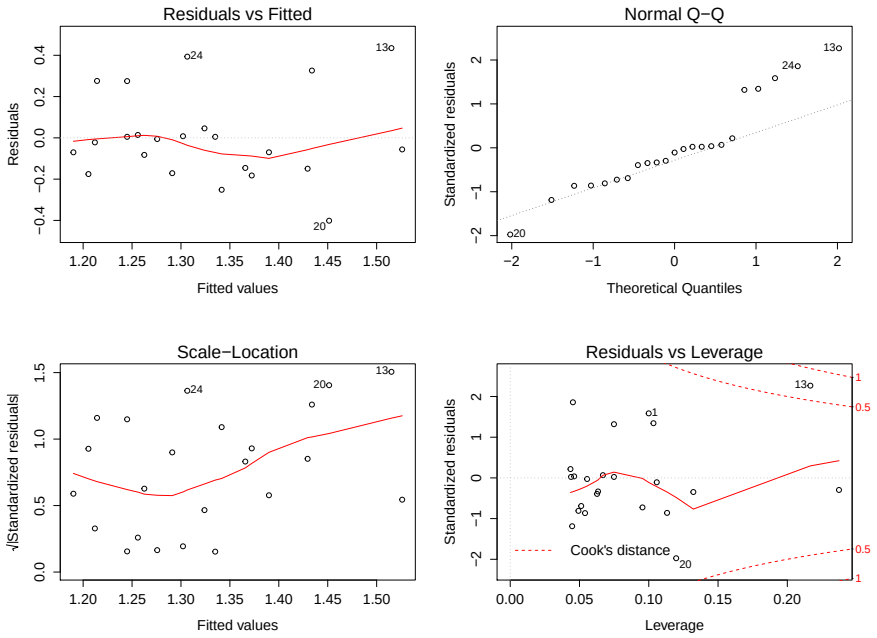


Figure 1: four plots

From the first plot (top-left), as the fitted values along x increase, the residuals decrease and then increase. This pattern is indicated by the red line, which should be approximately flat if the disturbances are homoscedastic. The plot on the bottom left also checks this, and is more convenient as the disturbance term in Y axis is standardized.

- **Assumption 4:** No autocorrelation of residuals

  When the residuals are autocorrelated, it means that the current value is dependent of the previous (historic) values and that there is a definite unexplained pattern in the Y variable that shows up in the disturbances.

  – Method 1: Visualise with acf plot

```
library(ggplot2)
acf(linearMod$residuals)  # highly autocorrelated from the picture.
```

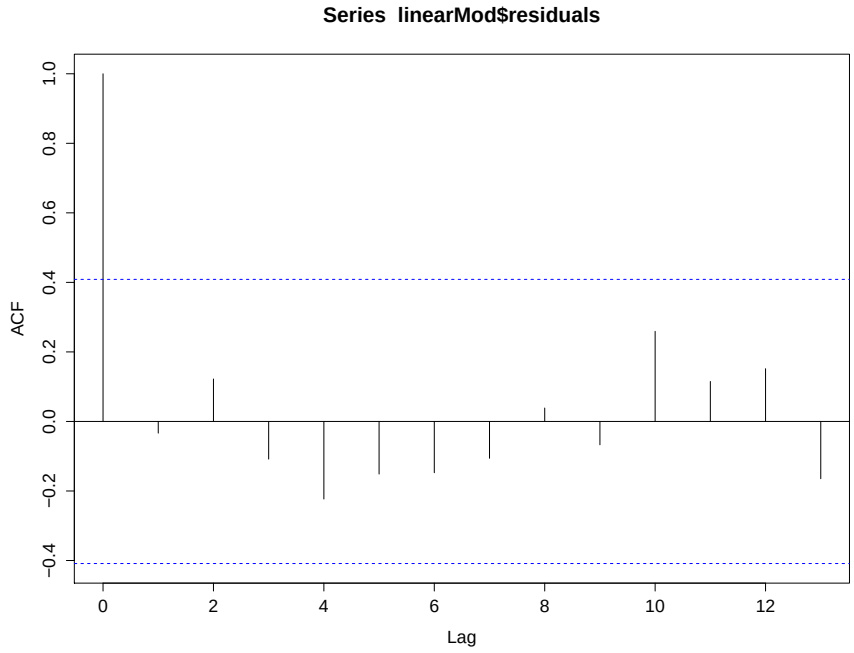


Figure 2: checking if residual uncorrelated

The X axis corresponds to the lags of the residual, increasing in steps of 1. The very first line (to the left) shows the correlation of residual with itself (Lag0), therefore, it will always be equal to 1. If the residuals were not autocorrelated, the correlation (Y-axis) from the immediate next line onwards will drop to a near zero value below the dashed blue line (significance level).

– Method 2: Runs test to test for randomness

```
> library(randtests)
> ?runs.test
> runs.test(linearMod$residuals)
```

```
Runs Test

data:  linearMod$residuals
statistic = -0.43693, runs = 11, n1 = 11, n2 = 11, n = 22, p-value = 0.6622
alternative hypothesis: nonrandomness
```

With a p-value 0.662, we fail to reject the null hypothesis that it is random. This means there is not a definite pattern in the residuals.

- Method 3: Durbin-Watson test

```
> lmtest::dwtest(linearMod)

Durbin-Watson test

data:  linearMod
DW = 1.8024, p-value = 0.3153
alternative hypothesis: true autocorrelation is greater than 0
```

### How to Rectify?

## Check Assumptions Automatically

The gvlma() function from gvlma offers a way to check the important assumptions on a given linear model.

```
> par(mfrow=c(2,2))  # draw 4 plots in same window
> gvlma::gvlma(linearMod)
>plot(linearMod)

Call:
lm(formula = short.velocity ~ blood.glucose)

Coefficients:
(Intercept)  blood.glucose
1.09781        0.02196


ASSESSMENT OF THE LINEAR MODEL ASSUMPTIONS
USING THE GLOBAL TEST ON 4 DEGREES-OF-FREEDOM:
Level of Significance =  0.05

Call:
gvlma::gvlma(x = linearMod)

Value p-value                Decision
Global Stat        3.47573  0.4816 Assumptions acceptable.
Skewness           1.21255  0.2708 Assumptions acceptable.
Kurtosis           0.05701  0.8113 Assumptions acceptable.
Link Function      0.95131  0.3294 Assumptions acceptable.
```

```
Heteroscedasticity 1.25486  0.2626 Assumptions acceptable.
```

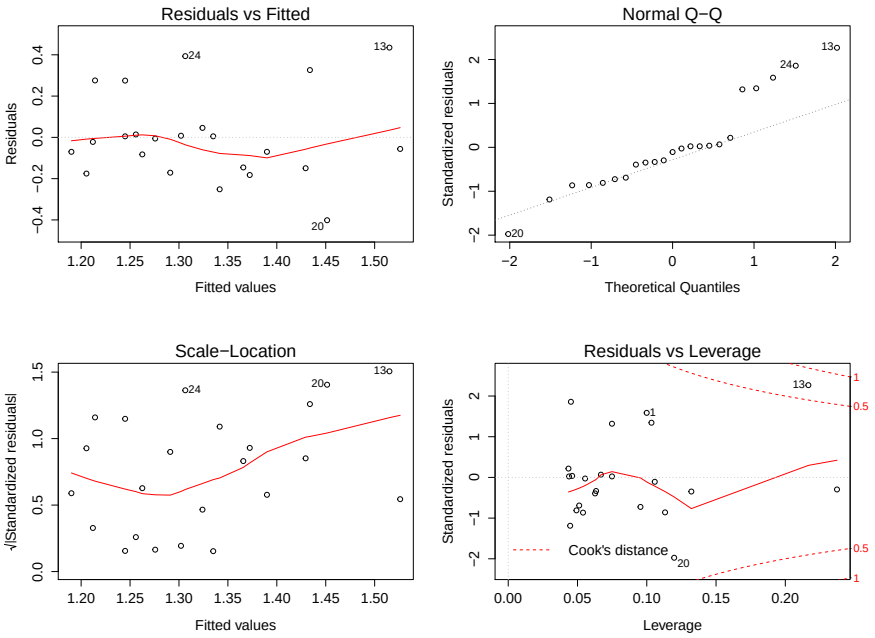


Figure 3: gvlma plot

in the data and having even 2 or 3 outliers can impact the quality of the model. So the immediate approach to address this is to remove those outliers and re-build the model.
From the above plot the data points: 24, 20 and 13 are marked as outliers. Lets remove them from the data and re-build the model.
mod ¡- lm(dist speed, data=cars[-c(23, 35, 49), ]) gvlma::gvlma(mod)

## One more graph to consider

There is one more thing left to be explained. That is, the plot in the bottom right. It is the plot of standardized residuals against the leverage. Leverage is a measure of how much each data point influences the regression. The plot also contours values of Cooks distance, which reflects how much the fitted values would change if a point was deleted.
A point far from the centroid with a large residual can severely distort the regression. For a good regression model, the red smoothed line should stay close to the mid-line and no point should have a large cooks distance (i.e. should not have too much influence on the model.)

```
> influence.measures(linearMod)
Influence measures of
lm(formula = short.velocity ~ blood.glucose, data = thuesen[-c(13,      20, 24), ]) :

dfb.1_ dfb.bld.   dffit cov.r   cook.d    hat inf
1  -0.68675  1.04129  1.2747 0.556 5.58e-01 0.1503   *
2   0.00535  0.01243  0.0464 1.180 1.14e-03 0.0539
3   0.00916 -0.00493  0.0131 1.190 9.14e-05 0.0582
```

```
4  -0.03648  0.04715  0.0508 1.743 1.37e-03 0.3572   *
5   0.03393 -0.02228  0.0417 1.202 9.19e-04 0.0700
6   0.65890 -0.52078  0.7002 0.836 2.11e-01 0.1119
7   0.01845 -0.00396  0.0394 1.177 8.19e-04 0.0505
8  -0.01689 -0.12065 -0.3615 0.932 6.13e-02 0.0563
9  -0.09219  0.05746 -0.1182 1.172 7.31e-03 0.0655
10  0.03475 -0.10560 -0.1976 1.135 2.01e-02 0.0700
11  0.02057 -0.01445  0.0239 1.216 3.03e-04 0.0788
12 -0.04541  0.03610 -0.0481 1.264 1.22e-03 0.1148
14  0.14750 -0.22705 -0.2814 1.239 4.08e-02 0.1433
15  0.49710 -0.34921  0.5777 0.805 1.44e-01 0.0788
17 -0.19459  0.16252 -0.2002 1.279 2.09e-02 0.1468
18  0.02375  0.01521  0.1014 1.155 5.38e-03 0.0512
19  0.06198 -0.15770 -0.2729 1.091 3.74e-02 0.0751
21  0.02252 -0.04404 -0.0654 1.228 2.26e-03 0.0915
22 -0.42267  0.34171 -0.4427 1.093 9.59e-02 0.1237
23 -0.13259  0.05131 -0.2312 1.060 2.68e-02 0.0526
```

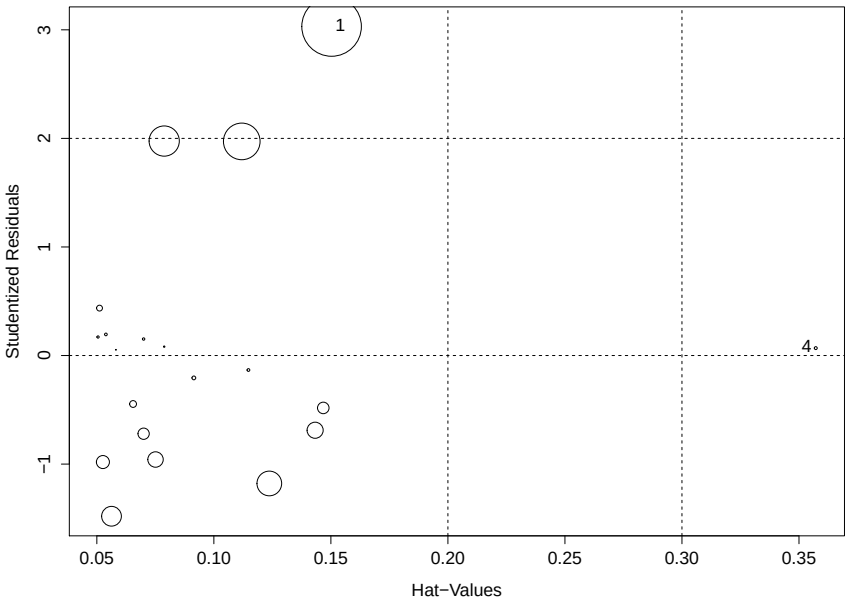


Figure 4: influence plot

# Prediction

# R Codes

```
########################################checking Assumption
mean(linearMod$residuals)
par(mfrow=c(2,2))  # set 2 rows and 2 column plot layout
```

```
plot(linearMod)
##################################uncorrelated residuals
par(mfrow=c(1,1))
# Method 1: Visualise with acf plot
library(ggplot2)
acf(linearMod$residuals)  # highly autocorrelated from the picture.
# Method 2: Runs test to test for randomness
library(randtests)
?runs.test
runs.test(linearMod$residuals)
#Method 3: Durbin-Watson test
library(lmtest)
lmtest::dwtest(linearMod)

###Check assumptions automatically
library(gvlma)
?gvlma
par(mfrow=c(2,2))  # draw 4 plots in same window
gvlma::gvlma(linearMod)
plot(linearMod)
linearMod <- lm(short.velocity~blood.glucose, data=thuesen[-c(13,20, 24), ])
gvlma::gvlma(linearMod)
summary(gvlma(linearMod))

plot(linearMod)
###################################Cook's distance
par(mfrow=c(1,1))
influence.measures(linearMod)
influencePlot(linearMod)
```