0.1 Residual Plots

In the graph above, you can predict non-zero values for the residuals based on the fitted value. For example, a fitted value of 8 has an expected residual that is negative. Conversely, a fitted value of 5 or 11 has an expected residual that is positive.

The non-random pattern in the residuals indicates that the deterministic portion (predictor variables) of the model is not capturing some explanatory information that is leaking into the residuals. The graph could represent several ways in which the model is not explaining all that is possible.

Possibilities include:

- A missing variable
- A missing higher-order term of a variable in the model to explain the curvature
- A missing interction between terms already in the model

Identifying and fixing the problem so that the predictors now explain the information that they missed before should produce a good-looking set of residuals.

In addition to the above, here are two more specific ways that predictive information can sneak into the residuals:

The residuals should not be correlated with another variable. If you can predict the residuals with another variable, that variable should be included in the model. In Minitabs regression, you can plot the residuals by other variables to look for this problem.

Autocorrelation

Adjacent residuals should not be correlated with each other (**autocorrelation**). If you can use one residual to predict the next residual, there is some predictive information

present that is not captured by the predictors. Typically, this situation involves time-ordered observations. For example, if a residual is more likely to be followed by another residual that has the same sign, adjacent residuals are positively correlated. You can include a variable that captures the relevant time-related information, or use a time series analysis.

In Minitabs regression, you can perform the ***Durbin-Watson*** test to test for autocorrelation.

0.1.1 Residual Plots

A residual plot is a graph that shows the residuals on the vertical axis and the independent variable on the horizontal axis. If the points in a residual plot are randomly dispersed around the horizontal axis, a linear regression model is appropriate for the data; otherwise, a non-linear model is more appropriate.

Below the table on the left shows inputs and outputs from a simple linear regression analysis, and the chart on the right displays the residual (e) and independent variable (X) as a residual plot.

The residual plot shows a fairly random pattern - the first residual is positive, the next two are negative, the fourth is positive, and the last residual is negative. This random pattern indicates that a linear model provides a decent fit to the data.

Below, the residual plots show three typical patterns. The first plot shows a random pattern, indicating a good fit for a linear model. The other plot patterns are non-random (U-shaped and inverted U), suggesting a better fit for a non-linear model.

In the next lesson, we will work on a problem, where the residual plot shows a non-random pattern. And we will show how to "transform" the data to use a linear model with nonlinear data.

0.2 Diagnostic Plots for Linear Models with R

Plot Diagnostics for an `lm` Object

Six plots (selectable by `which`) are currently available:

1. a plot of residuals against fitted values,
2. a Scale-Location plot of *sqrt(— residuals —)* against fitted values,
3. a Normal Q-Q plot,
4. a plot of Cook's distances versus row labels,
5. a plot of residuals against leverages,
6. a plot of Cook's distances against leverage/(1-leverage).

By default, the first three and 5 are provided.

Residuals plots

lme allows to plot the residuals in the following ways:

```
res_lme=residuals(model_lme)
plot(res_lme)
qqnorm(res_lme)
qqline(res_lme)
plot(model_lme)
```

When the `plot` function calls the model object, the residual plot is produced.

```
plot(JS.roy1, which=c(1) )
```

LME models assume that the residuals of the model are normally distributed. A Normal probability plot can be constructed to check this assumption. Commonly used R commands can be used to construct the plot.

```
qqnorm(resid(JS.roy1),pch="*",col="red")
qqline(resid(JS.roy1),col="blue")
```

```
table(dat$method[1:255])
##
##   J   S
## 255   0
table(dat$method[256:510])
##
##   J   S
##   0 255
```

```
library(predictMeans)
CookD(model, group=method, plot=TRUE, idn=5, newwd=FALSE)
```

```
> shapiro.test(resid(JS.roy1)[256:510])


Shapiro-Wilk normality test


data:  resid(JS.roy1)[256:510]

W = 0.9395, p-value = 9.503e-09
```

```
plot(roy.NLME, resid(., type = "p") ~ fitted(.) | method,

abline = 0, id=.05)
```

```
library(predictMeans)

CookD(model, group=method, plot=TRUE, idn=5, newwd=FALSE)
```

```
blood.red <- blood[!(blood$subject %in% c(68,78,80)),]

dim(blood.red)

# 27 observations should be removed.


blood.NLME.red <-lme(BP ~ method-1 , random=~1|subject,data = blood.red)

plot(blood.NLME.red, resid(., type = "p") ~ fitted(.) | method, abline = 0, id=.05
```

```
> shapiro.test(resid(JS.roy1)[1:255])


Shapiro-Wilk normality test


data:  resid(JS.roy1)[1:255]
W = 0.9931, p-value = 0.2852
```

```
> shapiro.test(resid(JS.roy1)[256:510])


Shapiro-Wilk normality test


data:  resid(JS.roy1)[256:510]
W = 0.9395, p-value = 9.503e-09
```

```
data.frame( response = resid(JS.ARoy20091, type = "response"),
pearson  = resid(JS.ARoy20091, type = "pearson"),
normalized = resid(JS.ARoy20091, type = "normalized") )
```

```
response      pearson    normalized
1    -4.65805902 -0.761587227 -0.7615872269
2    -0.88701342 -0.145025661  0.0776238081
3    -5.16580898 -0.844603753 -0.8446037530
```

```
4     2.29041830  0.374480726  0.6450898404
5     7.87508366  1.287567009  1.2875670086
6    -6.57048659 -1.074266908 -1.5090772378
.....................................
```

For the J observations, the variance is 6.116252 whereas for the S observations, the denominator is 9.118144. (with the expected ratio of 1.490806)

```
> pearson %>%
+   as.numeric %>%
+   matrix(nrow=85) %>%
+   round(4)
[,1]    [,2]    [,3]    [,4]    [,5]    [,6]
[1,] -0.7616  0.2194  0.3829 -0.2983  0.3597 -0.0790
[2,] -0.1450  0.1820 -0.1450 -0.5014  0.1567  0.2663
[3,] -0.8446  0.4634  0.1364 -0.1630 -0.2727  0.1660
[4,]  0.3745 -0.2795 -0.2795 -0.2658 -0.2658  0.6115
[5,]  1.2876 -0.6744 -0.6744  0.8935 -0.0935 -0.8612
[6,] -1.0743  1.8687 -0.7473 -0.0383  0.2908 -0.3673
.....................................
```

We can plot the residuals against the fitted values, to assess the assumption of constant variance.

```
# standardized residuals versus fitted values
plot(JS.ARoy20091, resid(., type = "pearson") ~ fitted(.) ,
abline = 0, id = 0.05)
```

```
par(mfrow=c(1,2))
qqnorm((resid(JS.ARoy20091)[1:255]),
pch="*",col="red",
ylim=c(-40,40),
main="Method J")
qqline(resid(JS.ARoy20091)[1:255],col="blue")
qqnorm((resid(JS.ARoy20091)[256:510]),
pch="*",col="red",
ylim=c(-40,40),
main="Method S")
qqline(resid(JS.ARoy20091)[256:510],col="blue")
par(mfrow=c(1,1))
```

Residuals plots

When the `plot` function calls the model object, the residual plot is produced.

```
plot(JS.roy1, which=c(1) )
```

LME models assume that the residuals of the model are normally distributed. A Normal probability plot can be constructed to check this assumption. Commonly used R commands can be used to construct the plot.

```
qqnorm(resid(JS.roy1),pch="*",col="red")
qqline(resid(JS.roy1),col="blue")
```

```
table(dat$method[1:255])
##
##   J   S
## 255   0
table(dat$method[256:510])
##
##   J   S
##   0 255
```

```
plot(roy.NLME, resid(., type = "p") ~ fitted(.) | method,
abline = 0, id=.05)
```

```
data.frame( response = resid(JS.ARoy20091, type = "response"),
pearson  = resid(JS.ARoy20091, type = "pearson"),
normalized = resid(JS.ARoy20091, type = "normalized") )
```

```
response      pearson    normalized
1    -4.65805902 -0.761587227 -0.7615872269
2    -0.88701342 -0.145025661  0.0776238081
3    -5.16580898 -0.844603753 -0.8446037530
4     2.29041830  0.374480726  0.6450898404
5     7.87508366  1.287567009  1.2875670086
6    -6.57048659 -1.074266908 -1.5090772378
.........................................
```

For the J observations, the variance is 6.116252 whereas for the S observations, the denominator is 9.118144. (with the expected ratio of 1.490806)

```
> pearson %>%
+   as.numeric %>%
+   matrix(nrow=85) %>%
+   round(4)
[,1]    [,2]    [,3]    [,4]    [,5]    [,6]
[1,] -0.7616  0.2194  0.3829 -0.2983  0.3597 -0.0790
[2,] -0.1450  0.1820 -0.1450 -0.5014  0.1567  0.2663
[3,] -0.8446  0.4634  0.1364 -0.1630 -0.2727  0.1660
[4,]  0.3745 -0.2795 -0.2795 -0.2658 -0.2658  0.6115
[5,]  1.2876 -0.6744 -0.6744  0.8935 -0.0935 -0.8612
[6,] -1.0743  1.8687 -0.7473 -0.0383  0.2908 -0.3673
.........................................
```

We can plot the residuals against the fitted values, to assess the assumption of constant variance.

```
# standardized residuals versus fitted values
plot(JS.ARoy20091, resid(., type = "pearson") ~ fitted(.) ,
abline = 0, id = 0.05)
```

```
par(mfrow=c(1,2))
qqnorm((resid(JS.ARoy20091)[1:255]),
pch="*",col="red",
ylim=c(-40,40),
main="Method J")
qqline(resid(JS.ARoy20091)[1:255],col="blue")
qqnorm((resid(JS.ARoy20091)[256:510]),
pch="*",col="red",
ylim=c(-40,40),
main="Method S")
qqline(resid(JS.ARoy20091)[256:510],col="blue")
par(mfrow=c(1,1))
```

This code will allow you to make QQ plots for each level of the random effects. LME models assume that not only the within-cluster residuals are normally distributed, but that each level of the random effects are as well. Depending on the model, you can vary the level from 0, 1, 2 and so on

```
qqnorm(JS.ARoy20091, ~ranef(.))


#  qqnorm(JS.ARoy20091, ~ranef(.,levels=1)
```

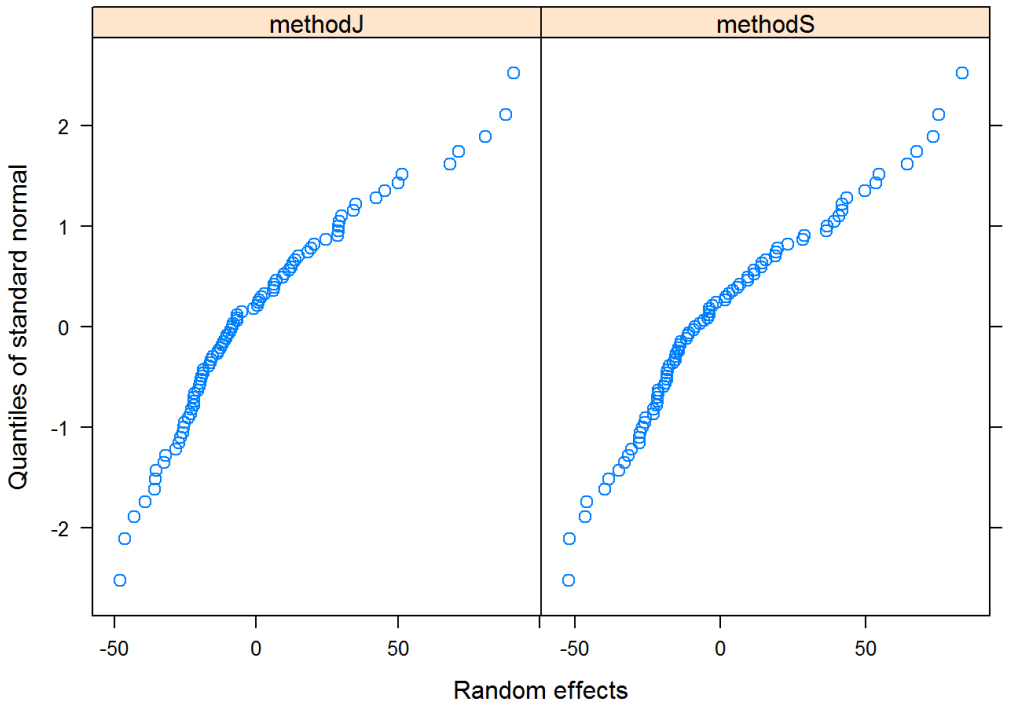

Figure 1:

```
data.frame( response = resid(JS.roy1, type = "response"),

pearson  = resid(JS.roy1, type = "pearson"),

normalized = resid(JS.roy1, type = "normalized") )
```

```
response      pearson    normalized

1    -4.65805902 -0.761587227 -0.7615872269
```

```
2    -0.88701342 -0.145025661  0.0776238081
3    -5.16580898 -0.844603753 -0.8446037530
4     2.29041830  0.374480726  0.6450898404
5     7.87508366  1.287567009  1.2875670086
6    -6.57048659 -1.074266908 -1.5090772378
.....................................
```

For the J observations, the variance is 6.116252 whereas for the S observations, the denominator is 9.118144. (with the expected ratio of 1.490806)

```
> pearson %>%
+   as.numeric %>%
+   matrix(nrow=85) %>%
+   round(4)
[,1]    [,2]    [,3]    [,4]    [,5]    [,6]
[1,] -0.7616  0.2194  0.3829 -0.2983  0.3597 -0.0790
[2,] -0.1450  0.1820 -0.1450 -0.5014  0.1567  0.2663
[3,] -0.8446  0.4634  0.1364 -0.1630 -0.2727  0.1660
[4,]  0.3745 -0.2795 -0.2795 -0.2658 -0.2658  0.6115
[5,]  1.2876 -0.6744 -0.6744  0.8935 -0.0935 -0.8612
[6,] -1.0743  1.8687 -0.7473 -0.0383  0.2908 -0.3673
.....................................
```

We can plot the residuals against the fitted values, to assess the assumption of constant variance.

```
# standardized residuals versus fitted values
plot(JS.roy1, resid(., type = "pearson") ~ fitted(.) ,
abline = 0, id = 0.05)
```

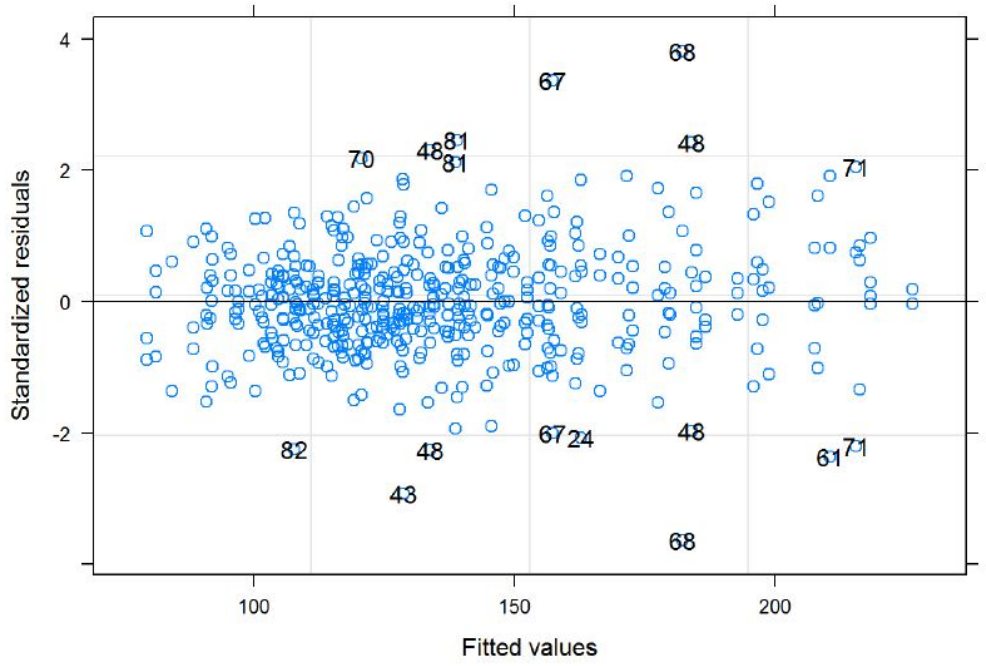

Figure 2:

- The **Scale-Location** plot, also called Spread-Location or S-L plot, takes the square root of the absolute residuals in order to diminish skewness (sqrt(—E—)) is much less skewed than — E — for Gaussian zero-mean E).

- The **Residual-Leverage** plot shows contours of equal Cook's distance, for values of cook.levels (by default 0.5 and 1) and omits cases with leverage one with a warning. If the leverages are constant (as is typically the case in a balanced aov situation) the plot uses factor level combinations instead of the leverages for the x-axis. (The factor levels are ordered by mean fitted value.)

```
par(mfrow=c(4,1))

plot(fittedmodel)

par(opar)
```

0.2.1 Residual Plots

A residual plot is a graph that shows the residuals on the vertical axis and the independent variable on the horizontal axis. If the points in a residual plot are randomly dispersed around the horizontal axis, a linear regression model is appropriate for the data; otherwise, a non-linear model is more appropriate.

Below the table on the left shows inputs and outputs from a simple linear regression analysis, and the chart on the right displays the residual (e) and independent variable (X) as a residual plot.

```
x & 60 & 70 & 80 & 85 & 95 \\ \hline

y & 70 & 65 & 70 & 95 & 85 \\ \hline

y.hat 65.411 71.849 78.288 81.507 87.945

e 4.589 -6.849 -8.288 13.493 -2.945
```

The residual plot shows a fairly random pattern - the first residual is positive, the next two are negative, the fourth is positive, and the last residual is negative. This random pattern indicates that a linear model provides a decent fit to the data.

Below, the residual plots show three typical patterns. The first plot shows a random pattern, indicating a good fit for a linear model. The other plot patterns are non-random (U-shaped and inverted U), suggesting a better fit for a non-linear model.

Bibliography