

3.4: Residual Analysis

The `summary()` function provides a substantial amount of information to help us evaluate a regression model's fit to the data used to develop that model. To dig deeper into the model's quality, we can analyze some additional information about the observed values compared to the values that the model predicts. In particular, *residual analysis* examines these residual values to see what they can tell us about the model's quality.

Recall that the residual value is the difference between the actual measured value stored in the data frame and the value that the fitted regression line predicts for that corresponding data point. Residual values greater than zero mean that the regression model predicted a value that was too small compared to the actual measured value, and negative values indicate that the regression model predicted a value that was too large. A model that fits the data well would tend to over-predict as often as it under-predicts. Thus, if we plot the residual values, we would expect to see them distributed uniformly around zero for a well-fitted model.

The following function calls produce the residuals plot for our model, shown in Figure 3.3.

```
> plot(fitted(int00.lm), resid(int00.lm))
```

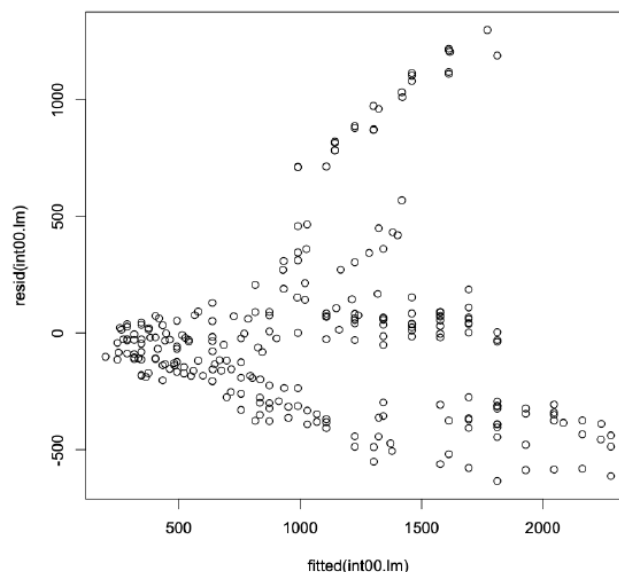


Figure 3.3: The residual values versus the input values for the one-factor model developed using the Int2000 data.

In this plot, we see that the residuals tend to increase as we move to the right. Additionally, the residuals are not uniformly scattered above and below zero. Overall, this plot tells us that using the clock as the sole predictor in the regression model does not sufficiently or fully explain the data. In general, if you observe any sort of clear trend or pattern in the residuals, you probably need to generate a better model. This does not mean that our simple one-factor model is useless, though. It only means that we may be able to construct a model that produces tighter residual values and better predictions.

Another test of the residuals uses the *quantile-versus-quantile*, or Q-Q, plot. Previously we said that, if the model fits the data well, we would expect the residuals to be normally (Gaussian) distributed around a mean of zero. The Q-Q plot provides a nice visual indication of whether the residuals from the model are normally distributed. The following function calls generate the Q-Q plot shown in Figure 3.4:

```
> qqnorm(resid(int00.lm))
> qqline(resid(int00.lm))
```

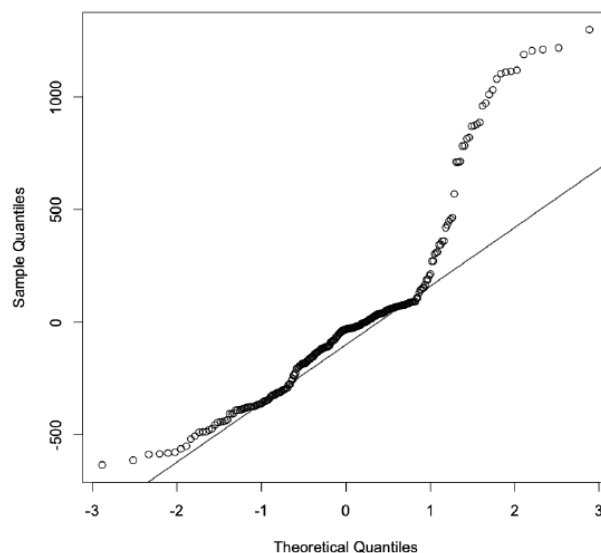


Figure 3.4: The Q-Q plot for the one-factor model developed using the Int2000 data.

If the residuals were normally distributed, we would expect the points plotted in this figure to follow a straight line. With our model, though, we see that the two ends diverge significantly from that line. This behavior indicates that the residuals are not normally distributed. In fact, this plot suggests that the distribution's tails are "heavier" than what we would expect from a normal distribution. This test further confirms that using only the clock as a predictor in the model is insufficient to explain the data.

Our next step is to learn to develop regression models with multiple input factors. Perhaps we will find a more complex model that is better able to explain the data.

This page titled [3.4: Residual Analysis](#) is shared under a [CC BY-NC 4.0](#) license and was authored, remixed, and/or curated by [David Lilja \(University of Minnesota Libraries Publishing\)](#) via [source content](#) that was edited to the style and standards of the LibreTexts platform.