# Chapter 3: Linear Models

*Nathanael Aff*

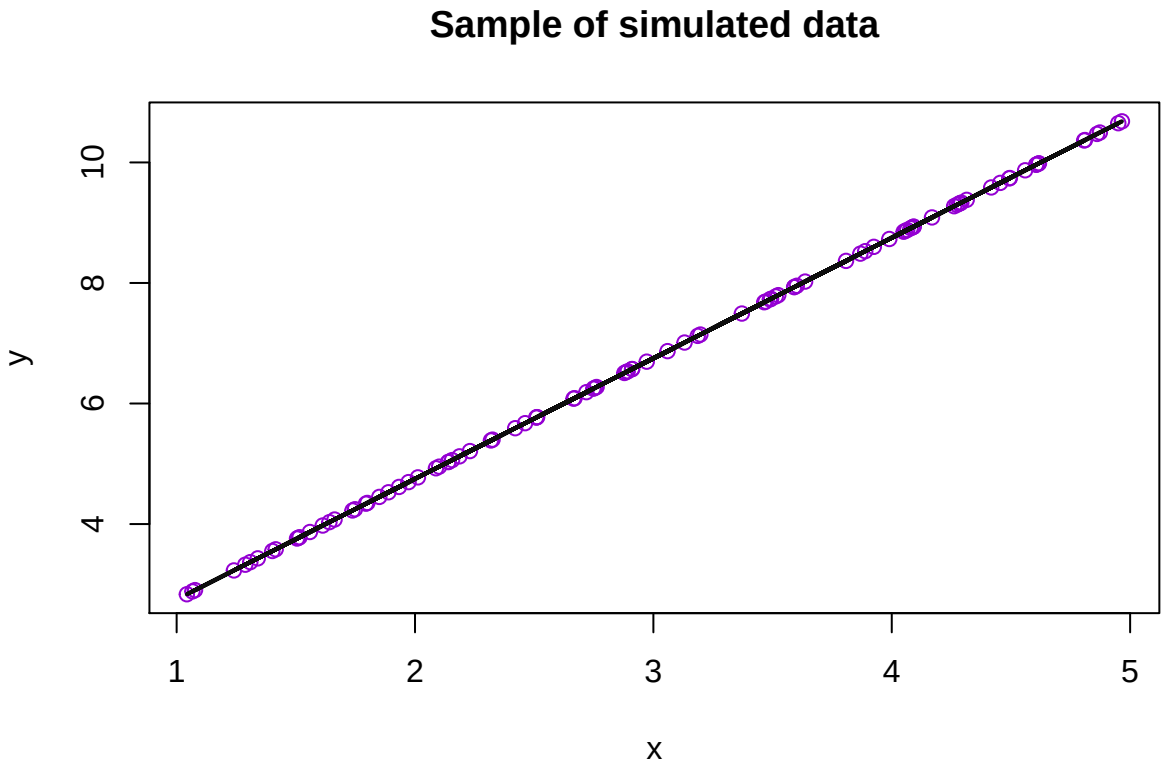
*June 11, 2018*

## Single Variable Linear Regression

Single variable regression, somtetimes called simple linear regression, models the relationship between two variables as a linear relationship.

The response variable $y$ must be a numeric variable and the predictor $x$ can be either numeric or a categorical (factor) variable.

```
# Our simulated function
fx <- function(x){
  y <- 2*x + 3/4
  return(y)
}

# Sample 100 points in the interval (1,5)
x <- runif(100, min=1, max = 5)

# Create the response variable y
y <- fx(x)

#Plot points
plot(x=x, y=y, col = "darkviolet", main="Sample of simulated data")

# Add a line to the plot
lines(x,y, col = "gray5", lwd = 2)
```




This plot shows the line of the true function $y = 2x + \frac{3}{4}$ and points sampled from this function.

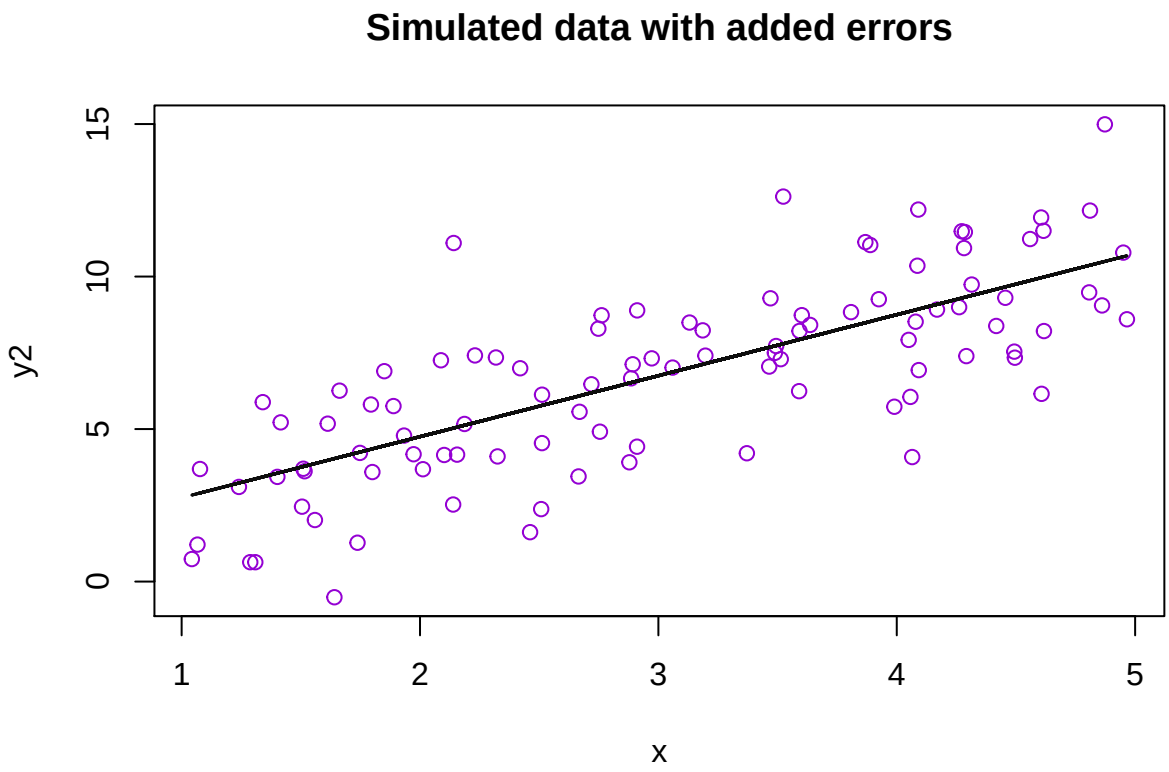
Of course, almost no data will have a perfectly linear relationship between $x$ and the response $y$.

The difference between the "best fit" line and the linear model is the random error denoted $\varepsilon$. The points we see when we collect real data, are modeled as having a linear relationship plus this random error.

$$y = f(x) + \varepsilon$$

We can simulate this by adding a random error to the data generated by our linear function. We use the `rnorm()` function to simulate errors from a random *Normal* distribution. The first parameter is the number of points to simulate and the next two *parameters* are the *mean* and *standard deviation* of the random normal we want to simulate.

```
errors <- rnorm(100, 0, 2)

# Create the response variable y and add errors
y2 <- fx(x) + errors

# Plot error points
plot(x, y2, col = "darkviolet", main= "Simulated data with added errors")
lines(x, y, col = "gray5", lwd = 1.5)
```




## Residuals

The dark grey line above is the line that 'generated' our data.

We can now see how well fitting a linear model in R will reproduce the equation of the "true" model.

First, we will create a data frame with $x$ and $y$ as columns, since we will usually be dealing with data in data frames or matrices.

Next we use the `lm()` function to fit a model. The argument `y~x` passed to `lm()` is a model **formula**. It means we are modeling the response `y` by (`~`) the predicttor `x`.

```
# Create a data frame with x and y as columns
df <- data.frame(x=x, y2=y2)

# Fit a linear model
fit.lm <- lm(y2~x, data = df)

# Look at the summary output for our linear model
summary(fit.lm)
```

```
##
## Call:
## lm(formula = y2 ~ x, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
```

```
## -4.7833 -1.4393  0.0974  1.5037  6.2265
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.4291     0.5818   0.737    0.463
## x             2.0756     0.1790  11.592   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.049 on 98 degrees of freedom
## Multiple R-squared:  0.5783, Adjusted R-squared:  0.574
## F-statistic: 134.4 on 1 and 98 DF,  p-value: < 2.2e-16
```

### Columns in the summary table

Note the table under the header **Coefficients**. The table has four columns:

- **Estimate** : The estimate of the coefficient of each predictor and the intercept.
- **Std. Error** : The standard error of the coefficient.
- **t-value** : The t-value testing the null hypothesis: Does $\beta = 0$ (is the coefficient equal to 0).
- **Pr(>|t|)** : The p-value for the previous null hypothesis. A low p-value indicates that the coefficient for that predictor *is not* 0 which, in turn, means that the predictor has some relationship to the response variable.
- **Significance indicator** : The final column has symbols indicating whether the p-value is significant.

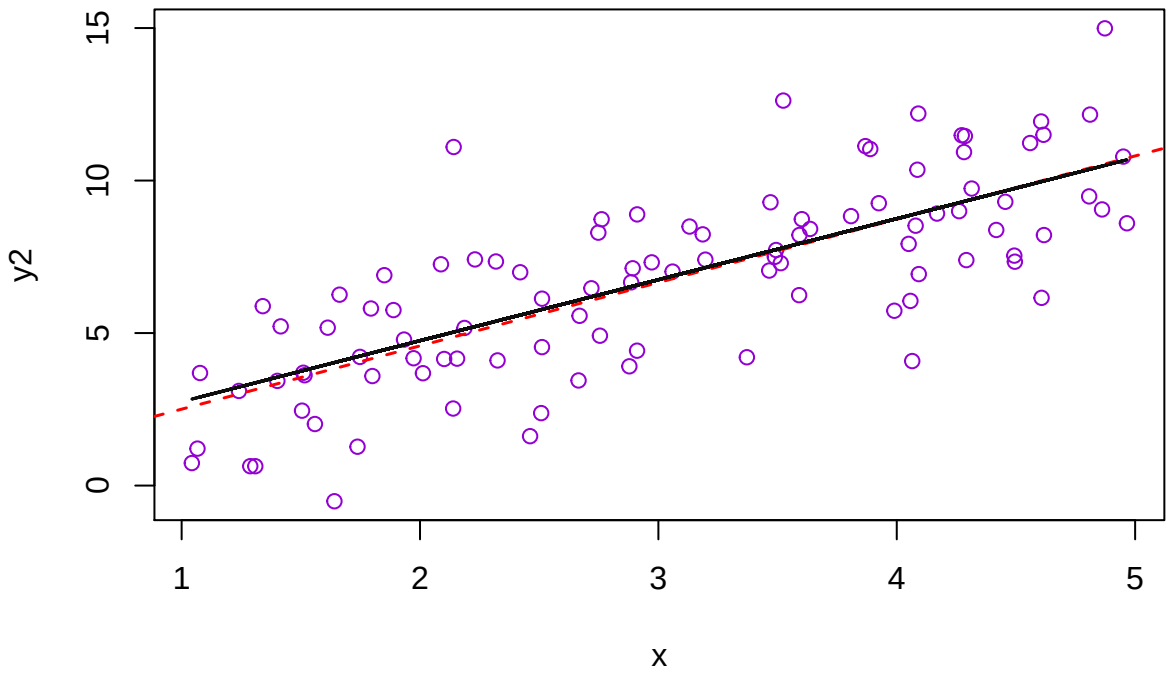
### Goodness of fit

Below the coefficient table there are several measures of the goodness of fit of the linear model. These include:

- R-squared
- Adjusted R-squared
- F-statistic
- P-value for F-statistic

## Comparing the

```r
# Plot the points
plot(x, y2, col="darkviolet")

# Plot a line for the model using the 'abline' function
abline(fit.lm, col="red", lwd=1.5, lty=2)
lines(x,y, col="gray5", lwd=1.5)
```




## Model diagnostics.

The single and multiple linear regression has several assumptions.

1. Residuals are normally distributed.
2. Variance of the residuals is constant across the range of the predictors.
3. The relationship between the predictors and the response is *linear*.

## Residuals

One of the assumptions of our model is that our residuals are normal. For this example, we generated the errors so we know they are normally generated.

We can also test this assumption using the fit from our linear model. The `lm` function returns the residuals of the linear fit. The `residuals` function can be used to select the model residuals.

### Are the residuals normally distributed?

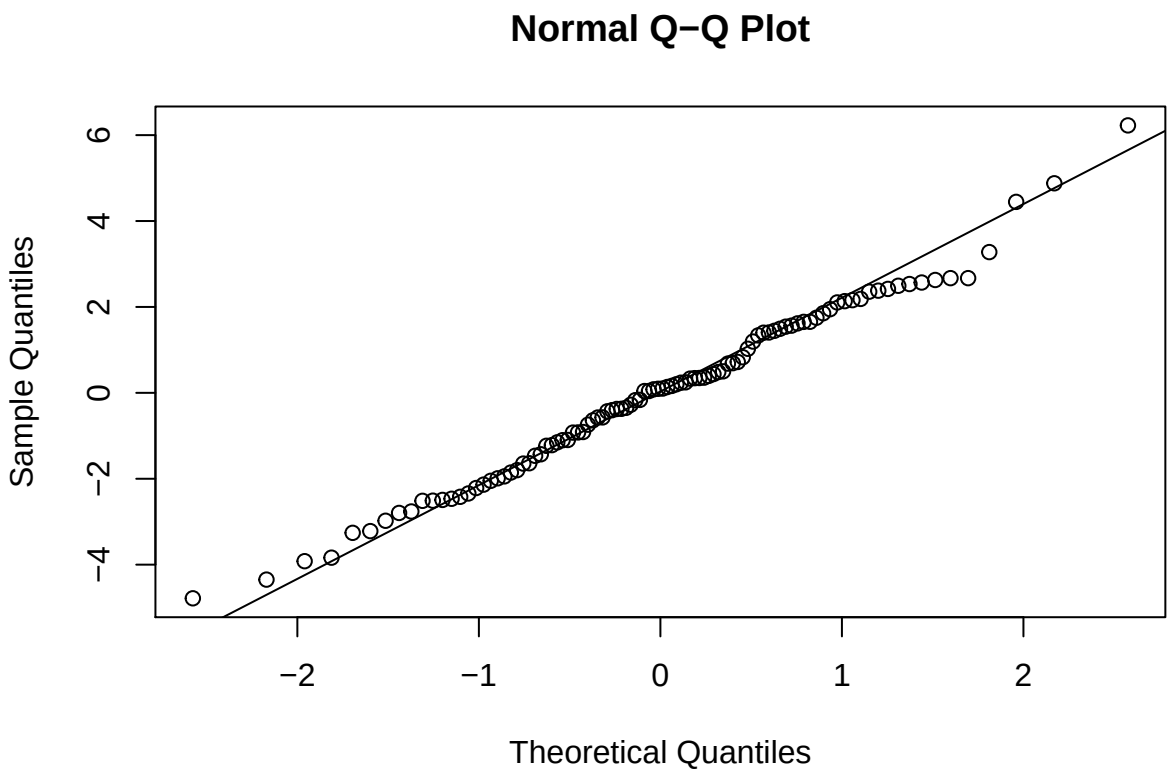
A quantile-quantile plot can be used to visualize the normality of the data.

The "q-q" plot plots the quantiles of a variable against the theoretical quantiles of a normal distribution. If the resulting points lie close to the line $y = x$, then the variable being tested is likely approximately normally distributed.

The plot below shows the residuals of our simulated data. Since we generated these residuals using a normal distribution using the `rnorm()` function, we know they are normally distributed.

You can see that even for normally distributed data the points shouldn't be expected to line up perfectly along the $x = y$ line.

```
# Generate a Quantile-Quantile plot of
qqnorm(residuals(fit.lm))
qqline(residuals(fit.lm))
```

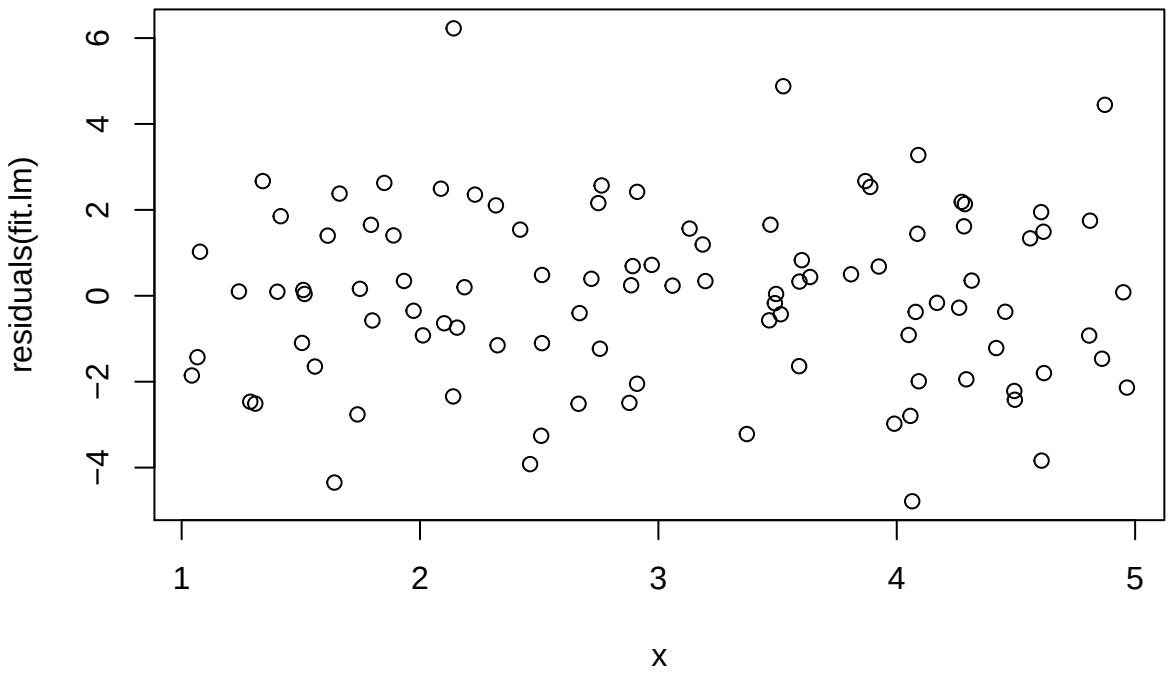



### Is there constant variance of the residuals (or error term)?

We can visualize whether there is constant variance of the error terms by plotting the residuals against our predictor $x$. If the amount of variance around 0 changes for different values of $x$, then our data may have non-constant variance.

Non-constant variance is called *heteroskedasticity* and can affect the accuracy of the linear model.

```
# Plot of residuals against predictor 'x'
plot(x, residuals(fit.lm))
```




## Is there a linear relationship between our predictors and response?

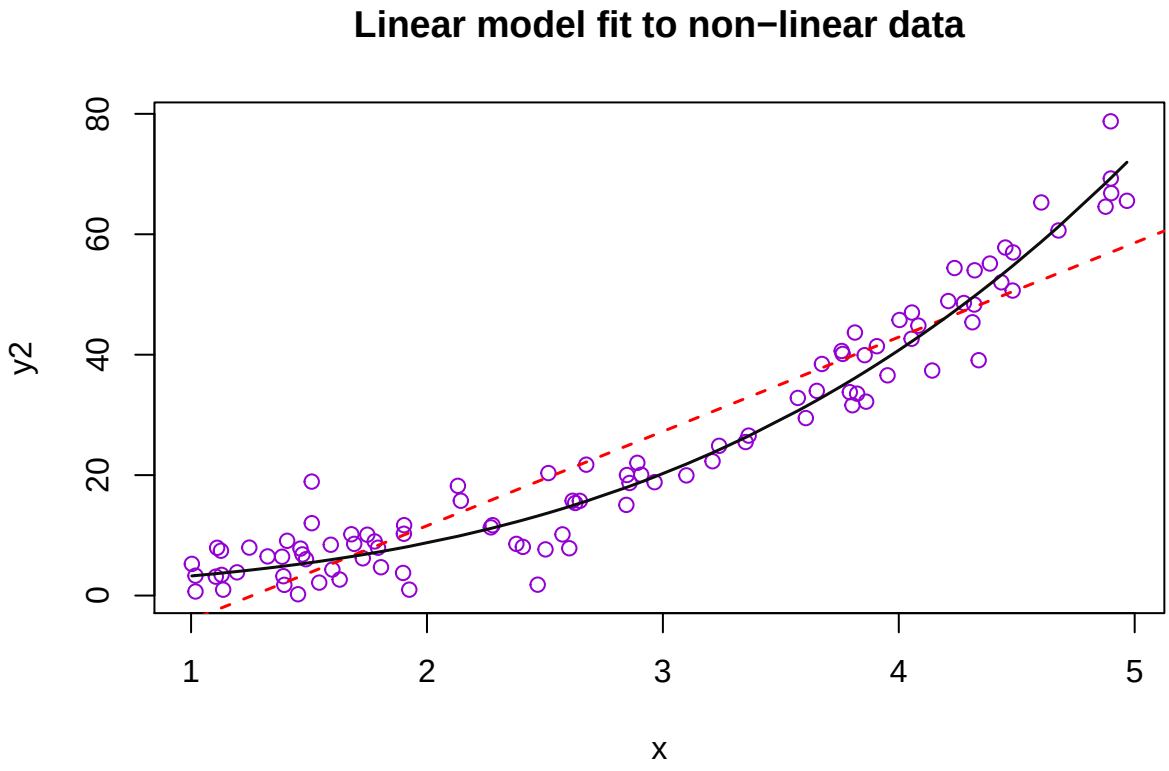
The plot shown above can also show whether there is a non-linear relationship between the predictor and the response.

Here, we'll generate some more data from an equation that includes a *non-linear* term. we do this by addint the term $(1/2)x^3$ to our original equation. The equation generating the data is now

$$y = (1/2)x^3 + 2x + 3/4.$$

```
# Our simulated function
fx2 <- function(x){
  y <- (1/2)*x^3 + 2*x + 3/4
  return(y)
}

# Sample 100 points in the interval (1,5)
x <- sort(runif(100, min=1, max = 5))

# Create the response variable y
y <- fx2(x)

# Add the random error term
error <- rnorm(100, 0, 4)
y2 <- y + error
```

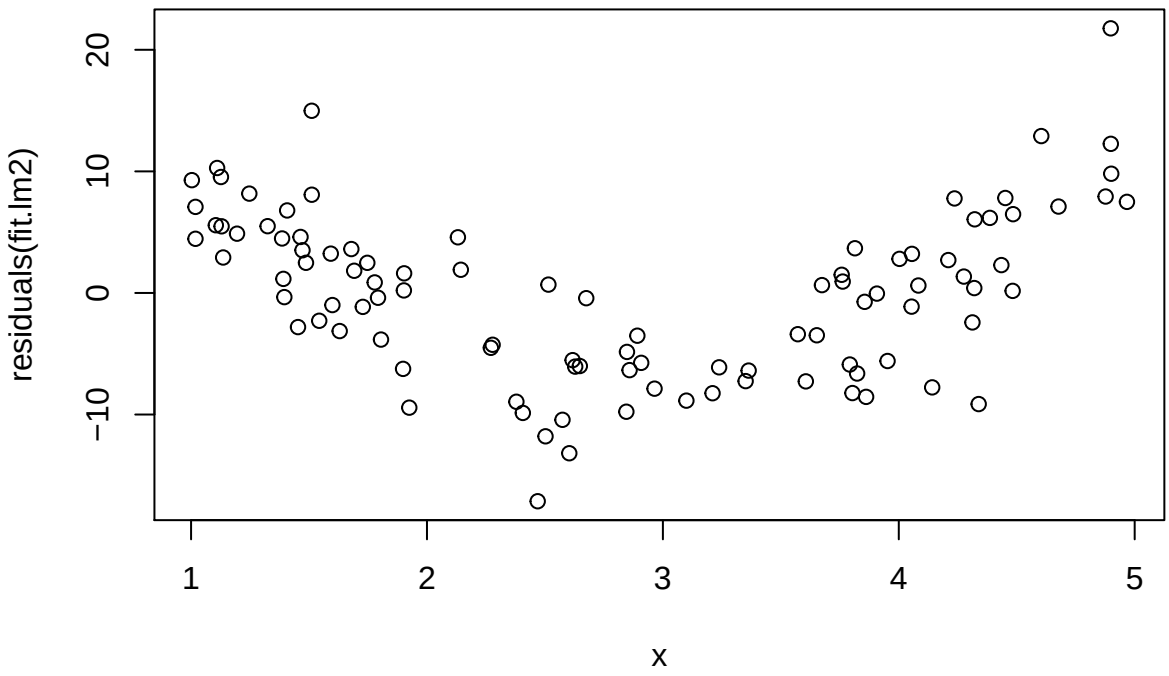
```r
# Fit a linear model to the data
fit.lm2 <- lm(y2 ~ x)

# Plot the points
plot(x, y2, col="darkviolet", main = "Linear model fit to non-linear data")

# Plot a line for the model using the 'abline' function
abline(fit.lm2, col="red", lwd=1.5, lty=2)
lines(x,y, col="gray5", lwd=1.5)
```




**Residual plot for non-linear data**

Now if we plot the residuals against our $x$ variable, we see a distinct pattern. The pattern indicates that our predictor has a non-linear relationship with the response variable.

```r
# Plot of residuals against predictor 'x'
plot(x, residuals(fit.lm2))
```




Just because the relationship between the predictor and the response is non-linear, does not mean that the linear model will not find a strong *linear* relationship between the variables.

Remember that the model that generated our data had both a linear and non-linear term, and a line can be a good approximation of a curve if the curvature is not too strong.

```
# Look at the summary output for our linear model
summary(fit.lm)
```

```
##
## Call:
## lm(formula = y2 ~ x, data = df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.7833 -1.4393  0.0974  1.5037  6.2265
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   0.4291     0.5818   0.737    0.463
## x             2.0756     0.1790  11.592   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.049 on 98 degrees of freedom
## Multiple R-squared:  0.5783, Adjusted R-squared:  0.574
## F-statistic: 134.4 on 1 and 98 DF,  p-value: < 2.2e-16
```