Gousto

Ahmed Ahmed

31/05/2022

Prepare for analyses

```
library(ggplot2)
library(dplyr)
library(broom)
library(ggpubr)
library(ggfortify)
library(formattable)
```

Data

Upload population data

```
population <-  read.csv("population_data.csv")
head(population)
```

	Time	Year	Population	Deaths	Births	Net.migration
	<int>	<int>	<int>	<int>	<int>	<int>
1	0	1971	55928000	645078	901648	-87870
2	1	1972	56096700	673938	833984	-33846
3	2	1973	56222900	669692	779545	-97153
4	3	1974	56235600	667359	737138	-79679
5	4	1975	56225700	662477	697518	-44641
6	5	1976	56216100	680799	675526	-20927

6 rows

Building linear regression model

$Population = \beta_0 + \beta_1 * time + e$

Applying the model using R, we have

```
population.lm <- lm(Population ~ Time, data = population)
summary(population.lm)
```

```
##
## Call:
## lm(formula = Population ~ Time, data = population)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1515782 -1085951  -328840  1034163  2220043
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 53936223     347089  155.40   <2e-16 ***
## Time          222954      12207   18.27   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1246000 on 48 degrees of freedom
## Multiple R-squared:  0.8742, Adjusted R-squared:  0.8716
## F-statistic: 333.6 on 1 and 48 DF,  p-value: < 2.2e-16
```

So our parameters are $\hat{\beta}_0$ = 5.393622310^{7} and $\hat{\beta}_1$ = 2.229537410^{5}

We can now plot the regression line in ggplot2

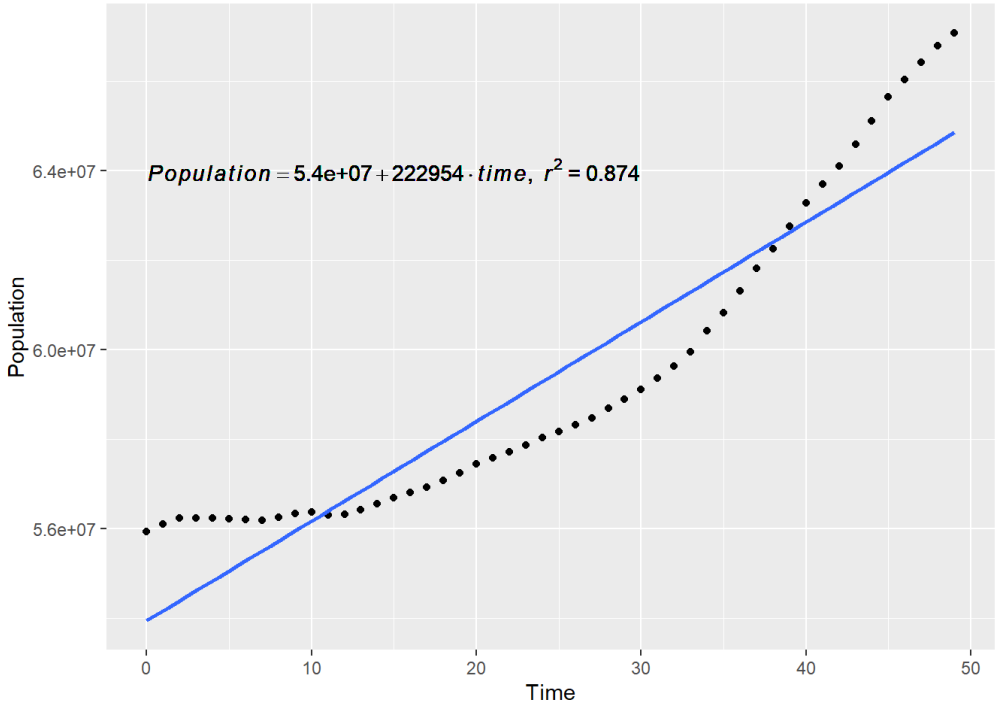
```
p <- ggplot(population,aes(Time,Population)) +
  geom_point() +
  geom_smooth(method='lm', se=FALSE)

lm_eqn <- function(population){
  m <- lm(Population ~ Time, population);
  eq <- substitute(italic(Population) == a + b %.% italic(time)*","~~italic(r)^2~"="~r2,
                   list(a = format(unname(coef(m)[1]), digits = 2),
                        b = format(unname(coef(m)[2]), digits = 2),
                        r2 = format(summary(m)$r.squared, digits = 3)))
  as.character(as.expression(eq));
}

p1 <- p + geom_text(x = 15, y = 6.4e7, label = lm_eqn(population), parse = TRUE)

p1
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Fitted values and residuals

We can add fitted values and residuals. Using the boom package I will produce several metrics useful for regression diagnostics.

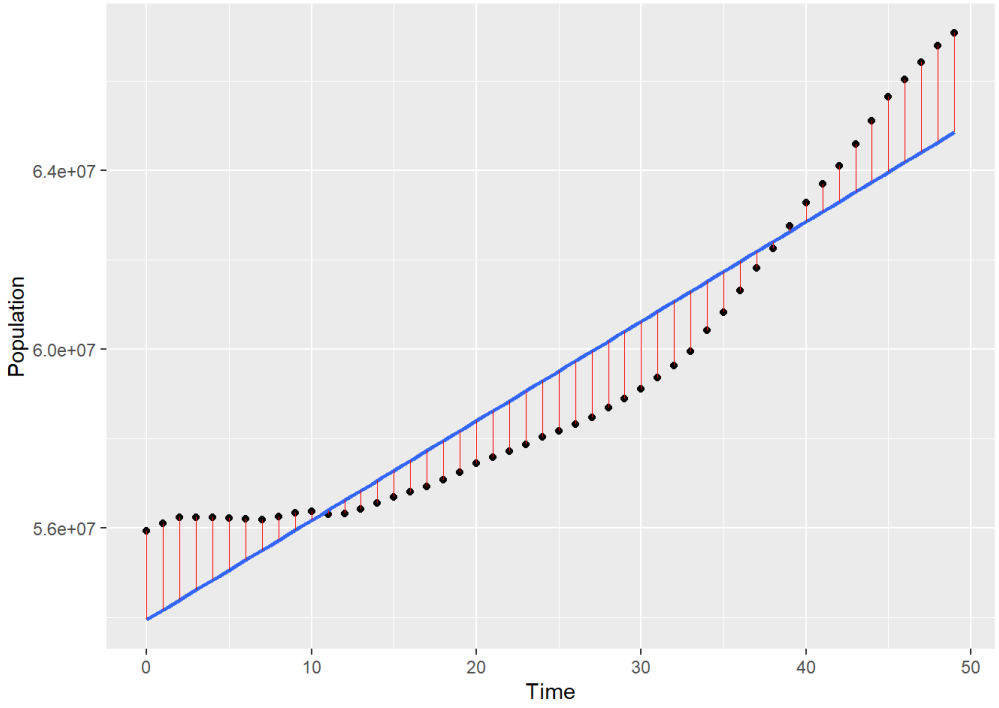
```
model.diag.metrics <- augment(population.lm)
head(model.diag.metrics)
```

Population <int>	Time <int>	.fitted <dbl>	.resid <dbl>	.hat <dbl>	.sigma <dbl>	.cooksd <dbl>	.std.resid <dbl>
55928000	0	53936223	1991777	0.07764706	1221890	0.11668771	1.6649976
56096700	1	54159177	1937523	0.07303721	1224077	0.10283155	1.6156126
56222900	2	54382131	1840769	0.06861945	1227650	0.08637830	1.5312895
56235600	3	54605085	1630515	0.06439376	1234533	0.06302611	1.3533181
56225700	4	54828038	1397662	0.06036014	1241087	0.04303726	1.1575584
56216100	5	55050992	1165108	0.05651861	1246562	0.02777603	0.9629884

6 rows

```
ggplot(model.diag.metrics, aes(Time, Population)) +
  geom_point() +
  stat_smooth(method = lm, se = FALSE) +
  geom_segment(aes(xend = Time, yend = .fitted), color = "red", size = 0.3)
```

```
## `geom_smooth()` using formula 'y ~ x'
```



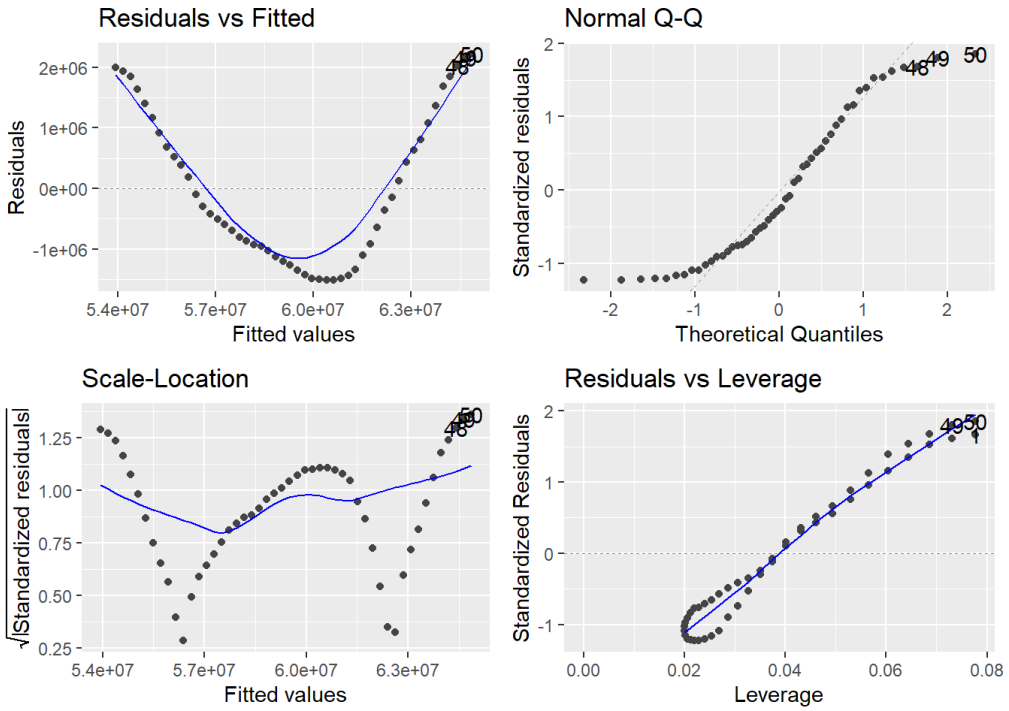
Regression assumptions

For linear regression we check that our data meet four main assumptions:-

- Independence of observation (no correlation). Only one independent variable so no need to worry about correlation between independent variables.
- Normality of residuals.Residuals are assumed to be normally distributed.
- Linearity of the data. Relationship between the independent(Time) and dependent(Population) is assumed to be linear.
- Independence of residuals error terms.

We can run regression diagnostics

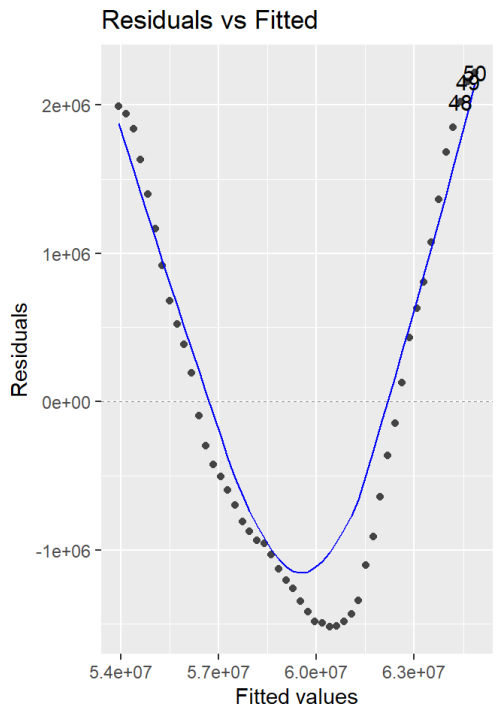
```
autoplot(population.lm)
```



Linearity of the data

This assumption can be checked by inspecting Residuals vs Fitted plot.

```
autoplot(population.lm,1)
```

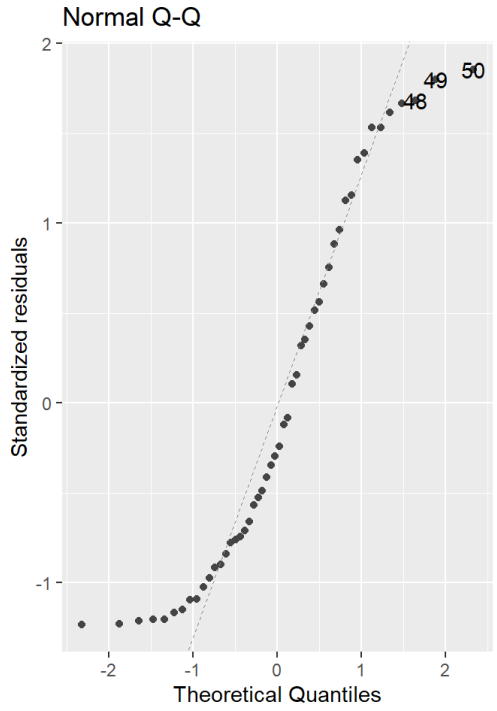


This plot shows if residuals have non-linear patterns.For a linear relationship we expect see evenly spread residuals around a horizontal line without distinct patterns.

Here, we see a a parabola. The non-linear relationship was not explained by the model and was left out in the residuals.

Normality of the residuals

```
autoplot(population.lm,2)
```

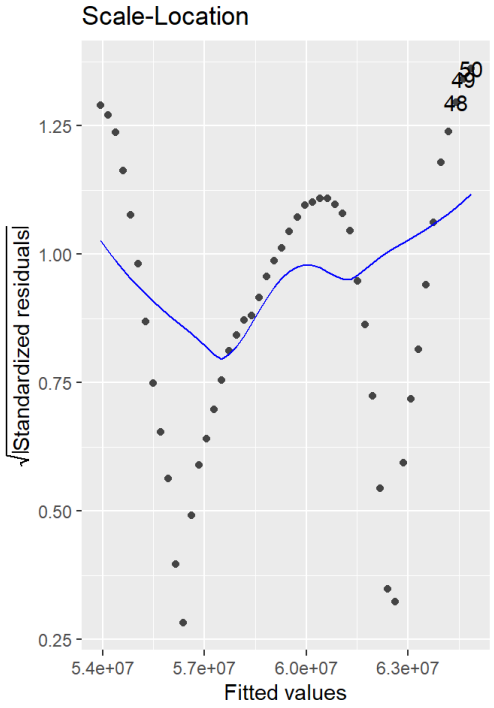


This plot shows if residuals are normally distributed. Ideally, residuals are lined up on the straight dashed line.

This assumptions seems to hold well, apart from the tails.

Homogeneity of variance

```
autoplot(population.lm,3)
```



This plot shows if residuals are constant along the ranges of independent variable.

It can be seen that the variability (variances) of the residuals increases at the beginning and end, then decrease at the middle. Suggesting a non-constant variances in the residuals errors (or heteroscedasticity).

Prediction

For prediction we have two distinctive measures:-

- Confidence Interval: prediction of the mean response.
- Prediction Interval: prediction of a future value.

For our model we have;

$$Population = \beta_0 + \beta_1 * time + e$$

Since $E(\epsilon) = 0$.Let $time^*$ denote the value of independent variable time. We can predict our point estimate, as follows

$$\hat{popul}ation = \hat{\beta}_0 + \hat{\beta}_1 * time^*$$

For the CI, our estimate only accounts for the variance in $\tilde{\beta}_0$ and $\tilde{\beta}_1$ i.e.

$$V(\tilde{\mu_{pop}}) = \sigma^2[\frac{1}{n} + \frac{(time^* - \overline{time}^2)}{\sum time_i - \frac{(\sum time_i)^2}{n}}]$$

For the PI,we need to account for the variance in the parameters and the error term. So we have,

$$V(Population) = \sigma^2[1 + \frac{1}{n} + \frac{(time^* - \overline{time}^2)}{\sum time_i - \frac{(\sum time_i)^2}{n}}]$$

Replacing σ by its estimate s, we have a confidence interval(CI)

$$\hat{\beta}_0 + \hat{\beta}_1 * time^* \pm t_{\frac{\alpha}{2}, n-2} * s\sqrt{[\frac{1}{n} + \frac{(time^* - \overline{time}^2)}{\sum time_i - \frac{(\sum time_i)^2}{n}}]}$$

and Prediction interval (PI)

$$\hat{\beta}_0 + \hat{\beta}_1 * time^* \pm t_{\frac{\alpha}{2}, n-2} * s\sqrt{[1 + \frac{1}{n} + \frac{(time^* - \overline{time}^2)}{\sum time_i - \frac{(\sum time_i)^2}{n}}]}$$

Plot

Prepare data frame for predictions entries.

```
population[nrow(population)+30,] <- NA
population
```

	Time	Year	Population	Deaths	Births	Net.migration
	<int>	<int>	<int>	<int>	<int>	<int>
1	0	1971	55928000	645078	901648	-87870

	Time	Year	Population	Deaths	Births	Net.migration
	<int>	<int>	<int>	<int>	<int>	<int>
2	1	1972	56096700	673938	833984	-33846
3	2	1973	56222900	669692	779545	-97153
4	3	1974	56235600	667359	737138	-79679
5	4	1975	56225700	662477	697518	-44641
6	5	1976	56216100	680799	675526	-20927
7	6	1977	56189900	655143	657038	-13795
8	7	1978	56178000	667177	686952	42325
9	8	1979	56240100	675576	734572	30604
10	9	1980	56329700	661519	753708	-64389

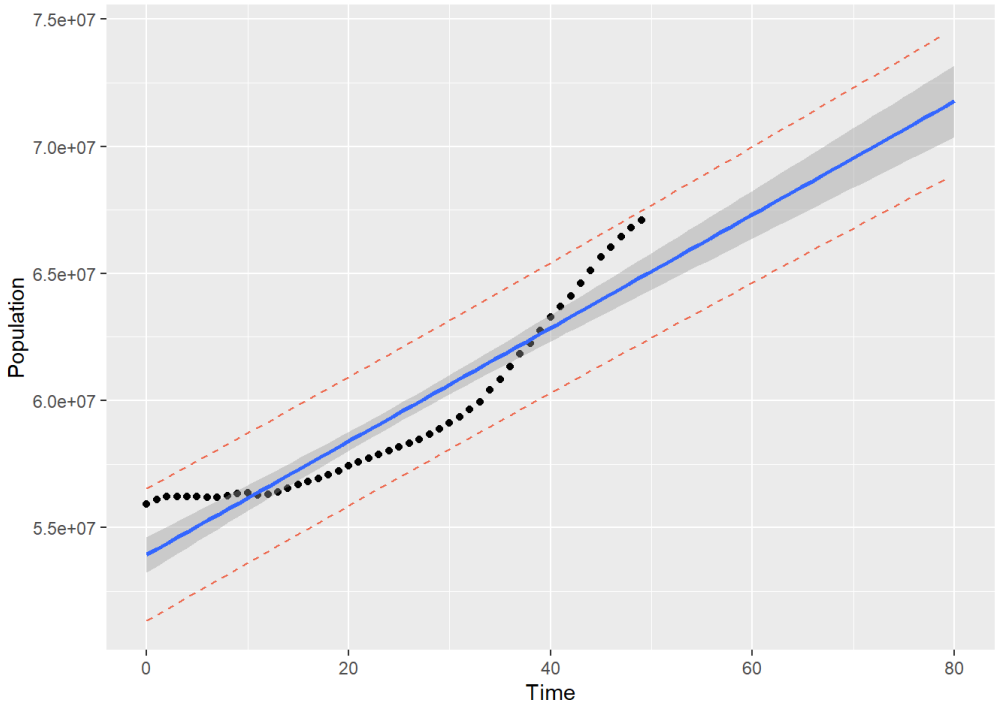
1-10 of 80 rows Previous **1** 2 3 4 5 6 … 8 Next

```
population[51:80,1] <- 50:79
```

```
predictions <- predict(population.lm, data.frame(Time=c(0:79)), interval="predict")
all_data <- cbind(population, predictions)
```

```
ggplot(all_data, aes(x = Time, y = Population)) + #define x and y axis variables
  geom_point()+xlim(0,80) + #add scatterplot points
  stat_smooth(method = lm,fullrange = TRUE) + #confidence bands
  geom_line(aes(y = lwr), col = "coral2", linetype = "dashed") + #lwr pred interval
  geom_line(aes(y = upr), col = "coral2", linetype = "dashed") #upr pred interval
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Our point estimate is then

```
(point_est <- comma(all_data[nrow(all_data),"fit"]))
```

```
## [1] 71,549,569.00
```

Similar to our google sheets workings, but now we can obtain our 95% prediction interval

```
comma(c(all_data[nrow(all_data),"upr"],all_data[nrow(all_data),"lwr"]))
```

```
## [1] 74,410,845.65 68,688,292.35
```

```
(error_range <- comma(all_data[nrow(all_data),"upr"] - all_data[nrow(all_data),"lwr"]))
```

```
## [1] 5,722,553.31
```

```
error_range/point_est
```

```
## [1] 0.08
```