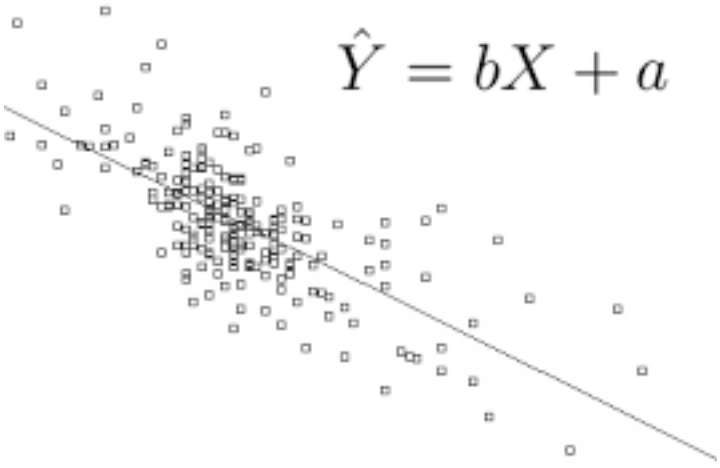
CALCULATING A TREND WITH T-SQL USING LINEAR REGRESSION

Practical guide



Srdjan Stojadinovic

TABLE OF CONTENT

THANKS TO

Julija Cincik reviewing and commenting on this paper.

WHAT YOU GOING TO LEARN?

Trend lines allow us to see the difference in various points over a period of time. This helps to understand the possible path the values might take in the future.

Finding regression line are very popular tool to illustrate possible outcome and they are used for different business purposes as:

- *Demand planning*
- *Pricing*
- *Performance*
- *Risk etc.*

You are going to learn about method, which will help you to calculate "best fit" line through scatterplots presuming that only tool you are using is restricted somehow to T-SQL.

To see this trending or for making this "predictions" we will use Microsoft and Google stock prices and hopefully this tutorial would give you enough knowledge to start experimenting with your own business data.

To understand basic principles this short tutorial will walk you through following:

- *Basic elements in math theory about Simple Linear regression.*
- *Math formula for calculating Linear Regression Coefficients*.
- *How to write SQL to implement calculation.*
- *How to test and find right result.*

WHAT IS SIMPLE LINEAR REGRESSION AND ITS BASIC ELEMENTS?

To get straight to the point - Linear Regression is used to study relation between two Quantitative variables.

We are predicting value of variable **Y** named **predictor** with variable **X** named **response**.

If you look through different literature, you will find **Predictor** and **Response** can be called differently e.g.

Y – Independent or explanatory variable.

X – Dependent or outcome variable.

RELATION BETWEEN VARIABLES

These two variables are related to each other with *statistical relationship*. This means that variables X and Y are not perfectly correlated to each other.

Just for your information, opposite of *statistical relationship* is *deterministic relationships.*

DETERMINISTIC RELATIONSHIPS

It is exact relation between two variables.

For example, relation between Celsius and Fahrenheit. When temperature is raising for each Celsius will temperature in Fahrenheit raise for 9/5*Cel+32.

STATISTICAL RELATIONSHIP

It is opposite from Deterministic relationship which means that relationship between the variables is not perfect and change in one variable will result in increase of another variable, but only approximately.

MATH END EQUATION

LINEAR REGRESSION EQUATION

Formula for find best-fitting line is **y=βx+α** where **β** variable is slope and **α** is intercept.

Now breaking down formula **y=βx+α**

$$\bar{Y} = m\bar{X} + b, where$$

$$\bar{X} = \frac{1}{n}\sum_{i=1}^{n} x_i \ (the\ average\ of\ x)$$

$$\bar{Y} = \frac{1}{n}\sum_{i=1}^{n} y_i \ (the\ average\ of\ y)$$

$$m = \frac{\sum_{i=1}^{n}(x_i - \bar{X})(y_i - \bar{Y})}{\sum_{i=1}^{n}(x_i - \bar{X})^2}$$

$$b = \bar{Y} - m\bar{X}$$

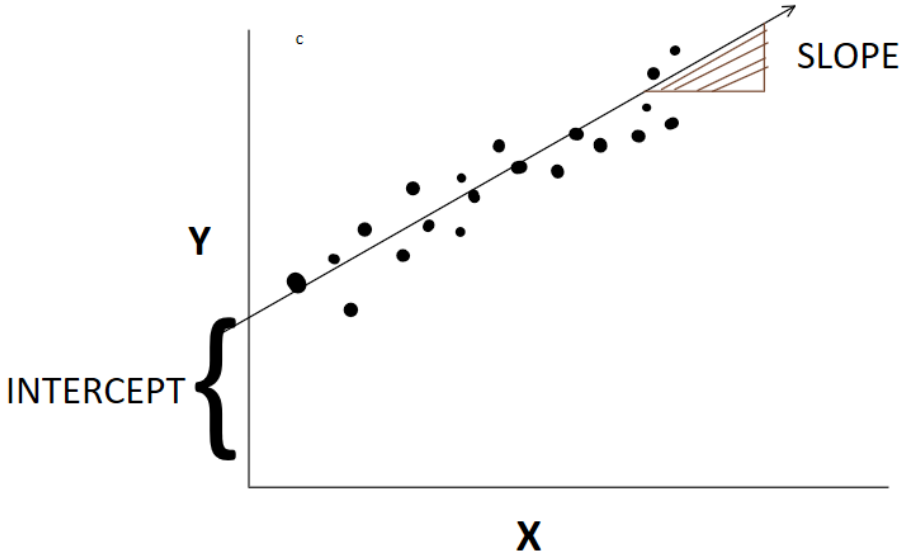
WHAT IS SLOPE?

The slope of a line is number that describes both the direction and steepness of the line. For example, a slope of 5/2 means that increases on **x**-axis by 2 increases **y**-value by 5 on average.

WHAT IS INTERCEPT?

The **α** - intercept of this line is the value of **y** at the point where the line crosses the **y** axis.

This drawing is showing example of Scatter Plots, Intercept and Slope.



WHAT IS R²

R^2 is a number between 0 and 1, where 0 indicates that the regression does not represent the data, and 1 is a perfect fit.

If R^2 is 0,866896156879491, then 86% of the variation can be explained by the regression, and the other 14% of the variation is unexplained. Basically, this says that the model is 86% more accurate than using a random guess (minimum error by using the mean).

We are going to calculate result for regression line in simple steps:

- ***Calculate slope****. (Always calculate* ***Slope*** *first because* ***Intercept*** *result is depending on* ***Slope****.)*
- ***Calculate intercept.***
- ***Calculate R².***

Theoretical stuff stops here and if you want to dive deeper into details, I would recommend reading tons of material on internet.

I was very glad to read following:

https://onlinecourses.science.psu.edu/stat501/node/251/

http://onlinestatbook.com/2/regression/intro.html

http://users.stat.ufl.edu/~winner/qmb3250/notespart2.pdf

SOLUTION

PREREQUISITES

We will create table **Historical_Stock_Prices** containing for now only Microsoft (MSFT) stock prices.

SQL Code	Comments

`CREATE TABLE [dbo].[Historical_Stock_Prices](`
`[Dato] [datetime] NULL,`
`[Price] [float] NULL`
`) ON [PRIMARY]`
	Code creates empty table.
	Price is "Closing Price".

Get data from ex. NASDAQ. I have chosen last 5 years stock price data.
https://www.nasdaq.com/symbol/msft/historical
Remember to format/change file so it can be imported into Excel and Copy/Pasted into SQL table afterwards.

Second possibility is that to make SSIS package to import file into table you just created.

SQL CODE

Her is SQL code which calculate Slope, Intercept and R^2:

```
WITH CTE as (
SELECT
	ROW_NUMBER() over(order by [Dato] asc) as
x,
	[Dato],
	Price Amount
FROM [dbo].Historical_Stock_Prices

),

CTE1 AS (
SELECT
	(COUNT(x) * SUM(X*Amount) - SUM(X) *
	SUM(Amount)) / (count(x) * SUM(X*X) -
	SUM(X) * SUM(X)) AS SLOPE,
	AVG(Amount) AVG_Amount,
	AVG(x) AVG_X,
	AVG(CAST(Amount as float)) - ((COUNT(x) *
	SUM(X*Amount) - SUM(X) * SUM(Amount)) /
	(count(x) * SUM(X*X) - SUM(X) * SUM(X))*
	AVG(x)) AS INTERCEPT
FROM CTE
)
```

CODE EXPLANATION

- First CTE is just getting data from table. I am using ROW_NUMBER function to generate numbers which help me calculate **X** (*dependent variable*) without converting dates into numbers.
- Second CTE(CTE1) is used to intermediate calculations and to get **Intercept** and **Slope** which makes easier to calculate R^2 in final calculation.

<pre>SELECT
SLOPE,
AVG_Amount-(SLOPE*AVG_x) as INTERCEPT,
(INTERCEPT * SUM(Amount) +SLOPE *
SUM(x*Amount)-
SUM(Amount)*SUM(Amount)/COUNT(x)) /
(SUM(Amount*Amount) - SUM(Amount)*
SUM(Amount) / COUNT(x)) AS R2
FROM CTE1, CTE
GROUP BY INTERCEPT,SLOPE,AVG_Amount,AVG_x</pre>
	- Last calculation (SELECT) is putting everything together. Using **Slope** and **Intercept** from CTE1 to calculate R^2.
	As you can see, I am calculating again **Intercept** just to show that it can be calculated using intermediate variables AVG_Amount and AVG_X from CTE1.

TEST OF RESULT IN EXCEL

Calculating stock price last 5 years through shown SQL Code are following numbers.

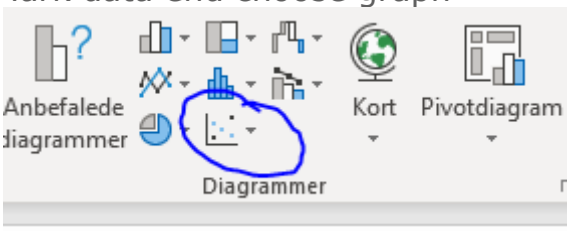
SLOPE	INTERCEPT	R2
0,054647	27,45644236	0,866896

We are going to test this result in Excel:

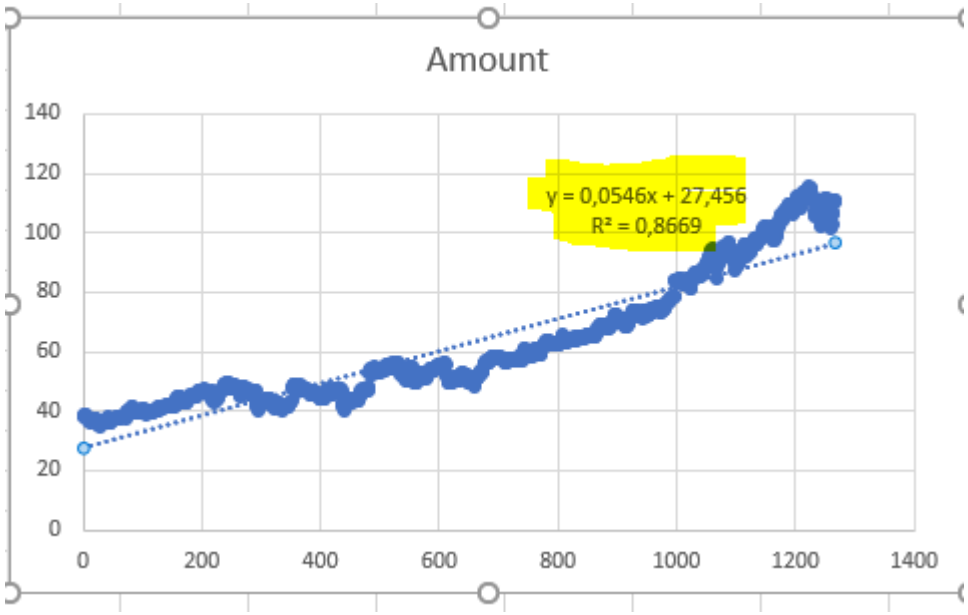
- Run this query in SQL Management Studio. It will return all rows from Historical_Stock_Prices table.

```
SELECT
    ROW_NUMBER() over(order by [Dato] asc) as x,
    [Dato],
    Price Amount
FROM [dbo].Historical_Stock_Prices
```

- Copy data with headers (Right click on data and then chose "Copy")
- Paste data in Excel
- Mark data end choose graph



- Adjust Trendline and set check marks to "Show equation" and "R^2" value.
- You should now see following diagram.



- Numbers marked with yellow (Intercept and Slope) are equal with result from our SQL code.

EXTENDING OF EXAMPLE

Let's extend our example with following:

- TREND calculation - can be used in different Reporting tools (Maybe *Reporting Services)* without statistical features to draw regression line.

- Additional data - import Google Stock prices into our table together with Microsoft to compare performance of those two stocks.

For this purpose, we will need to add to our table one extra attribute (Ticker) to distinguish between those two stocks.

Run code below in SQL Management Studio to create missing attribute:

```
ALTER TABLE dbo.Historical_Stock_Prices ADD Ticker varchar(50) NULL
```

Next step is to add/update existing and new records with Ticker values.
Run UPDATE and SET "MSFT" value on new Ticker attribute.

When importing GOOGLE data remember to set "Ticker" with "GOOGL" value.

EXTENDED SQL CODE WITH MULTIPLY COMPANY STOCK PRICES AND TREND MEASUREMENTS

```
--drop table #temp
WITH CTE as (
SELECT
	Ticker,
	ROW_NUMBER() over(partition by Ticker
order by [Dato] asc) as x,
	[Dato],
	SUM(Price) Amount
FROM [PLAYGROUND].[dbo].Historical_Stock_Prices
GROUP BY
	Ticker,[Dato]

),

CTE1 AS (
SELECT
	CTE.Ticker,
	(COUNT(x) * SUM(X*Amount) - SUM(X) *
SUM(Amount)) / (count(x) * SUM(X*X) - SUM(X) *
SUM(X)) AS SLOPE,
	AVG(Amount) AVG_Amount,
	AVG(x) AVG_X,
	AVG(CAST(Amount as float)) - ((COUNT(x) *
SUM(X*Amount) - SUM(X) * SUM(Amount)) /
(count(x) * SUM(X*X) - SUM(X) * SUM(X))* AVG(x))
AS INTERCEPT
FROM CTE
GROUP BY
	CTE.Ticker
)

SELECT
	CTE.Ticker,
	SLOPE,
	AVG_Amount-(SLOPE*AVG_x) as INTERCEPT,
	(AVG(INTERCEPT) * SUM(Amount) +AVG(SLOPE)
* SUM(x*Amount)-
SUM(Amount)*SUM(Amount)/COUNT(x)) /
(SUM(Amount*Amount) - SUM(Amount)* SUM(Amount) /
COUNT(x)) AS R2
INTO #temp
FROM CTE1 inner join CTE on
cte.Ticker=CTE1.Ticker
GROUP BY
	INTERCEPT, SLOPE, AVG_Amount, AVG_x,
CTE.Ticker

SELECT * FROM #temp

SELECT
	s.Ticker,
	Cast(Dato as Date) Dato,
	s.Price as Amount,
	(SUM(SLOPE)*ROW_NUMBER() OVER(PARTITION
BY s.Ticker ORDER BY [Dato] ASC))+SUM(INTERCEPT)
as TREND

FROM [dbo].Historical_Stock_Prices s  INNER JOIN
#temp ON #temp.Ticker=s.Ticker
GROUP by s.Ticker,[Dato], s.Price
```

Code explanation

- Same explanation as first one just I am grouping on **Ticker** as well.
  - First CTE is just used to get data from table.
    I am using ROW_NUMBER function to generate numbers which help me calculate **X** (*dependent variable*) without converting dates into numbers.

- Adding **Ticker** grouping.
  - Second CTE(CTE1) is used to intermediate calculations and to get **Intercept** and **Slope** which makes easier to calculate R^2 in final calculation.

- Adding **Ticker** and **TREND** calculation. TREND calculation is very useful when showing Trend line in different Reporting tool and which have no ability to draw regression line as Excel can.
  - Last calculation (SELECT) is putting everything together. Using **Slope** and **Intercept** from CTE1 to calculate R^2.
    *As you can see, I am calculating again **Intercept** just to show that it can be calculated using intermediate variables AVG_Amount and AVG_X from CTE1.*

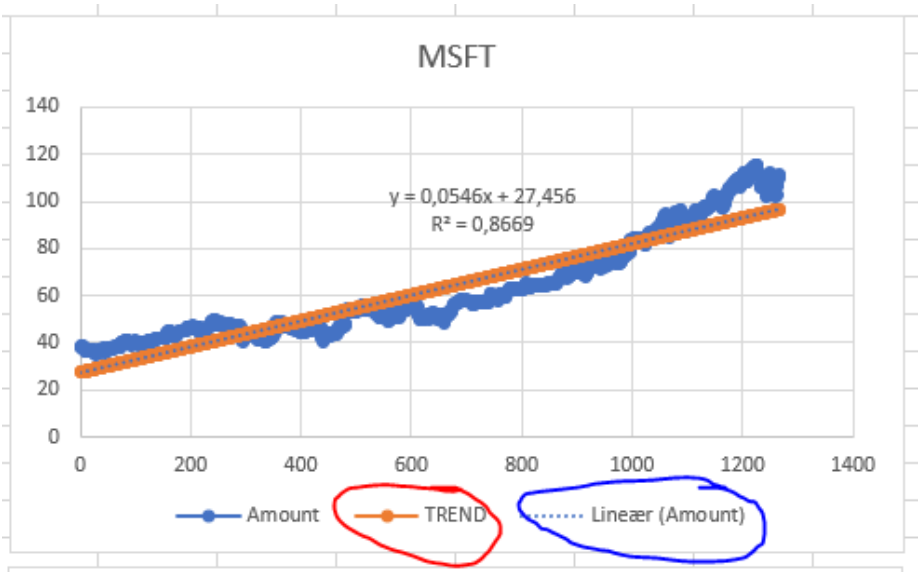
TEST OF RESULT IN EXCEL WITH MSFT AND GOOGL

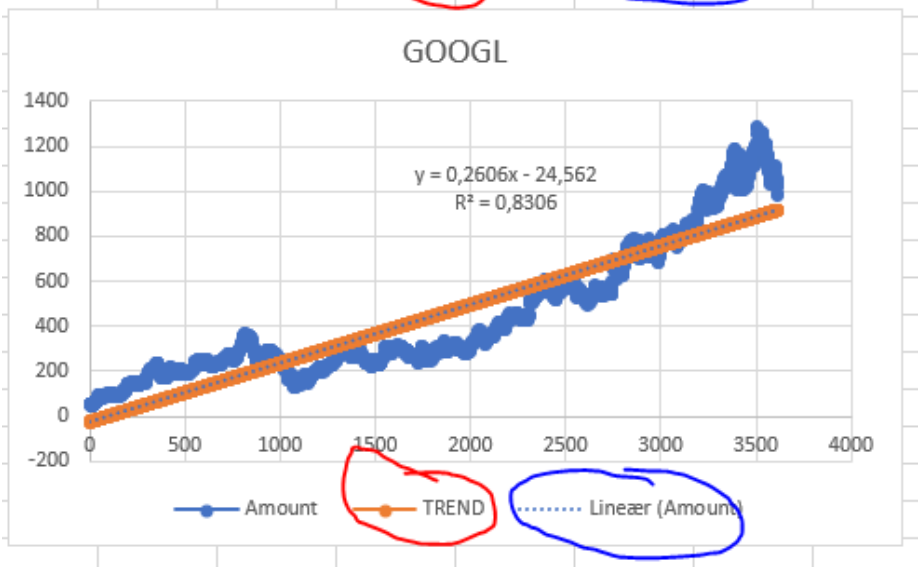
Running this SELECT statement (`SELECT * FROM #temp`) would return following:

Ticker	SLOPE	INTERCEPT	R2
GOOGL	0,260568	-24,56151319	0,830628
MSFT	0,054647	27,45644236	0,866896

#**Temp** table is generated with SQL code which can be viewed on previous page. This result is used to compare result from Excel.

Next picture represents SQL calculated values perfectly matching TREND line (Red) and regression line from Excel (Stipple).





CONCLUSION

In this article, we:

- walked through basic linear regression elements
- explored SQL solution regarding Linear Regression
- verified SQL results - explained how to test and compare SQL numbers in Excel.

Result is small framework for calculating simple linear regression. Just changing data input in SQL code allow user to calculate a regression for any data

I hope this article would encourage you in right direction to start experiment with your own data.