Sigmoid Curves

Before we get into it, I have posted a link to a second source as I know seeing it from multiple views helps me a lot when learning a subject. If you couldn’t care less on how the math actually works or how this is built, you can jump to the ‘building a sigmoid graph’ section and just copy step for step there.

[Cleartelligence Sigmoid Curves](https://www.cleartelligence.com/post/fun-with-curves-in-tableau-part-3-sigmoid-curves)

## Understanding the foundation

We will be using the superstore data as a reference, but understand this can be applied to many different metrics once we understand the main goal here. The sigmoid graph is not much different at all from a normal line graph. From each point going across the x-axis, we have a y-value where we started and a y-value that will end at the next x-axis value. For example, in February we have $59,751 in sales and in March we have $210,672 in sales. This leads to a line climbing the area between February and March starting and ending between these two points. See the figure below for a visual aid.

Chart, line chart

Description automatically generatedThe main thing to focus on here, is the line passing from each point to point. In this representation, it of course is a straight line but it is this gap we will be working with on setting up our sigmoid functions. Each month has a starting point, and ending point, and for our purposes down the road we can make use of the delta between the two. You could either say ‘we need to travel from $59, 751 to $210,672’ OR you could say ‘we need to travel $150, 921 UP from $59, 751’. Both these stories give the same result and we will need to use both ways to guide our sigmoid functions. In a linear function like the one to the left, we only pay attention to the two reference points and draw a straight line between the two. In reality, there is really a formula going on for the points between the two main points (the famous Y=MX + B formula we all remember from algebra in school). Another way to think about this is that instead of just one point for each month, there are actually a whole bunch of points on the x-axis between each month (lets say each hour of every day for example) that each have a corresponding y value using the linear equation formula. When you plot all of these on the graph, there are so many that they all bleed together and seem like one line. While we don’t usually care about these specifics on linear graphs, we need to care for sigmoid functions as each of these tiny x-values will be used to determine each point on the curve. As a visual example, when working in tableau you can drag your mouse across the top of a graph and you will see the corresponding circle below follow the path of your graph even though you wont see a ‘value’ until you hit the actual data point (in our case I am using the months of order dates). That is as in depth as we need to go for now, just understand that between each main data point, there are a lot of smaller points that give a function the ability to connect the dots. To review, in order to build our function we will need the starting point, ending point, and the delta between the two.

## What is a sigmoid function?

Chart, line chart

Description automatically generatedWithout getting into the specific mathematics just yet, you can think of a sigmoid function as a simple ‘S’ curve. It is a curve that starts out with a very low slope that increases as you reach the MIDDLE of the travel distance and then slows down again the closer you get to the end of the travel distance. You can already tell visually that this is the curve between each point in our sigmoid graph! The curves might of course be longer or wider depending on the starting point and ending point but the general curve remains the same. As eluded in the linkedin post that probably brought you here, this is where it can possibly be misleading when used on a graph if you are not careful. The curve between two points is going to happen regardless of the delta between the two points. So if for example the sales for February to March were $100,000 to 102,000, we would still see a small increase and decreasing curve between the two points and when tight enough, can mislead someone into thinking there was growth/loss where there really wasn’t. This can be fixed by adjusting the axis limits/tick marks or by doing away with this option all together if it does not aid to the story you are tying to tell. Without going into a deeper mathematical discussion on the sigmoid curve and its uses/limitations, just know that it is a “S” curve going from one point to another and for our purposes, the x-axis distance between each point will be 12 (going from -6 to 6). This range is what we will be using in reference to the previous section talking about the smaller points between each main point. So in this case, between each month we have 12 points ranging from -6 to 6. To learn more about why the -6 to 6 range, look up sigmoid functions or DM me!

## Building a sigmoid graph

### Densify the table

Ok, so now we have a basic understanding of the two graphs and now can start putting them together. Unfortunately, there is a bit of trickery we are going to have to play with inside Tableau to make this work. The first thing we are going to have to do is densify the table of data we are working with. There are many different ways to do this, but I am going to refer to one way as I believe it gets the point across. Our end goal here is basically to have two rows of data for each record. So for the superstore data, there is one row per every order-id. In excel, you can create a simple table that has a column “JoinID” and another as “Path” that will have rows with value 1 and the other 100 (the actual difference in numbers do not mean much for our purposes but to keep it simple, just do 100). The “JoinID” will just be values of 1. In tableau, you can create a calculated field or a joinfield in the datasource tab and you need to do so with a value of 1. Simply forward here, we are doing a 1 to 1 join. This is going to duplicate every row of data in the table with one ‘Path’ value of 1 and another of 100. Once this is done, you will navigate to the newly created ‘path’ measure and create bins (right click and select create -> bin) and make the bin size to 1. You could make a parameter value that will adjust the bin size and I encourage you to do so if you wish to see how the number of points affects the graph, but this requires some manipulation of the sigmoid formula so if you wish to keep it simple just put 1 for the size.

### Index calculation

We are going to create an index calculation that will be used in tandem with the final calculation of the sigmoid curve. Remember, we do not want to just index across each month, but across all of the tiny space between each month which is what the bins we created in the previous step are for.

*SalesCurve – Index*

((INDEX() – 1) \* .12) – 6

Note here for the names I am starting each with ‘SalesCurve’. This keeps it more organized when needing to work on the function components but is not totally necessary. The ‘INDEX()’ here of course is going to be the bin Path we created, with the rest of the components being the math that moves from -6 to positive 6 using our bin path. I will go into a little depth here to explain why/how this works but if you do not need or want to know, you can skip to the next calculation.

If you plug our binpath into the index of the equation above, you will start at a value of -6 (since the first index is 1, the expression in parenthesis closes to 0 and we are left with negative 6) and make tiny even increments all the way to positive 6. If you wanted to change the number of data points from 100 to another number or even a variable, you would also have to change the ‘.12’ value in the index formula. We get .12 because with 100 x-values, there is .12 worth of space from -6 to positive 6. As an example shown below, I only use 10 data points and thus the formula will be

((INDEX() -1) \* 1.2) -6

Giving a result as follows…



The more data points you have, the smoother the curve will look with the idea mentioned earlier about a lot of points bleeding together to form one line, so that is why I use 100 points for this example but now that you understand how this works, you can change at your will.

### Sales calculation

In our example with the superstore data, we are going to track the total sales each month. Since we are indexing on the binPath, we need to get the sum of sales of each month separately than the main index. We can do this by using WINDOW\_SUM in tableau, with the window being the given order month. The purpose of this function is to get the *end point of the month* in sales. We talked about this earlier on how there were two ways to get this value and for using this calculation as a variable moving forward, we are just going to collect it in a sum.

*SalesCurve – Sales*

WINDOW\_SUM(SUM([Sales])/2)

Note that we divide by 2 here within the window because our table has twice the rows since we applied the densification, so to get the real amount we need to divide by two. You can use this method OR use a fixed level of detail calculation for sales and use that moving forward. Either method works though I try to use the least amount of LOD in a workbook as possible to help performance but depending on the size of your data sample it may not matter one way or the other.

### Previous Sales calculation

To find out what the starting point is for each months sales, we need to know what the previous months sales were.

*SalesCurve – Previous*

ZN(LOOKUP([SalesCurve – Sales], -1))

Here I am using the LOOKUP function which is a table calculation in tableau that will revert to the previous windows value (getting a little messy here with naming conventions but you get the general point). I am looking up the previous windows (Order month) [SalesCurve – Sales] value which is just the total sales for that month. The negative 1 refers to the last window and the ZN function is a logical function that returns 0 if the result is NULL. This is for the very first window as we do not have a previous window at the first window, so we need to start from 0. You can get creative here with a different starting point but can get messy if not uniform across all the calculations so for now leave at 0.

### ValueDiff calculation

Now we are going to calculate the difference between the two values as we mentioned earlier.

*SalesCurve – ValueDiff*

[SalesCurve – Sales] – [SalesCurve – Previous]

### Startingpoint calculation

Lets get the starting value for each month that the sigmoid curve will start at. I know that it seems we already have this in the [SalesCurve – Previous] calculation, and we do, but we need another reference point to make our final calculations and avoid circular references.

SalesCurve – StartingPoint

RUNNING\_SUM([SalesCurve – ValueDiff]) – [SalesCurve – ValueDiff]

We use the running\_sum calculation here which is also a table calculation in tableau just like window\_sum. We are getting the current total of sales minus the current delta value, which would leave us the starting point of sales for the current month.

### Final path calculation

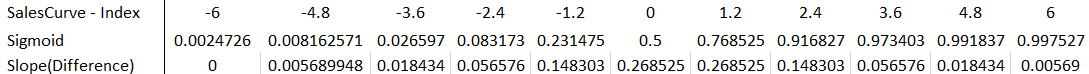
What a journey! We are almost there I promise. We are going to use a combination of the formulas we just created in conjunction with the sigmoid function. I include the sigmoid function itself within the last formula for simplicity, but to be more formal you can separate it out if you wish.

*SalesCurve – Curve*

[SalesCurve – StartingPoint] + (1 / (1 + EXP(-[SalesCurve – Index]))) \* [SalesCurve – ValueDiff]

I will go a little deeper into the sigmoid function itself here, but you can jump to the last section of formatting from here if you wish. The sigmoid function is the one in parentheses and defined below.

Here x is the index we move across. The purpose mathematically for the sigmoid function in our case is the *slope* of which our value increases. I wont go too far into the derivation and math theory behind it, but that is the basic understanding to its purpose in our [SalesCurve – Curve] formula. To show with values how this work, we can plot our index values as shown before starting with -6 to 6.

Notice the difference in the ‘Sigmoid’ values gets larger and larger as we approach 0 which is the middle point of our range. In the image of on “S”, this is where the slope is the largest and is reflected in the plot points above. The slope then starts to decrease in value once again as we approach the final value and move farther away from the center. The sigmoid function itself is getting closer and closer to the value of 1 as we go across the index values. You can see how now, generally speaking, that if we start at one point and multiply our ValueDiff (amount to get to the next value) by the sigmoid function, we will slowly get to our final value with the slope changing in steepness as we get closer and farther away from the middle. As a final summary point, you can see at the end of the sigmoid path, the value is almost exactly the value of 1 and in our [SalesCurve – Curve] formula, that would lead to

[SalesCurve – StartingPoint] + (1)\*[SalesCurve – ValueDiff] = FinalValue (ending point)

### Tableau Formatting/Setup

To set this up, we are going to drop the PathBin into the Columns section. Right click the PathBin pill and select ‘Show Missing Values’. From there we can move it to the marks card as a detail and select the type to be ‘Line’. Next, move the order date field to the Columns section and drill down to the months option and get rid of year, quarter sections.

Now drag the [SalesCurve – Index] to the columns and [SalesCurve – Curve] to the rows. On the [SalesCurve – Index] and [SalesCurve – Curve] pills, right click and select ‘Calculate using…’ and select the PathBin. Now on the [SalesCurve – Curve] pill, right click and select ‘Edit Calculation’. All of the drop down options in the nested calculations field should be using the PathBin but just in case move them all to that checkbox if not already. For the [SalesCurve-Previous] and [SalesCurve – StartingPoint] in the nested calculations drop down, select calculate using the time period (in our case the order month). Now, we have a curved line that should look something like this!

A picture containing calendar

Description automatically generatedTo finish clean up, right click the x axis to edit and make the starting and ending points fixed to be from -6 to 6. This will close that little gap between each month. From there, you can remove column lines, make it an area chart, or what other formatting preferences you wish! We finally made it!