Video Title: "Video 2: Understanding data sets as texts"
From "Rhetmap.org: Composing Data for Future Re-Use and Visualization"
by Chris Lindgren and Jim Ridolfo
(Kairos XX.X PraxisWiki)

Welcome back to The Missing Tutorial. This is video number two, understanding data sets as texts. The purpose of this video are the following two main objectives: First, we're going to learn how to read common data formats and structures that you will encounter out there in the wild, and the second main objective is understanding the links between these types of data with their textual modes and how we can maybe start to see some connections between potential visual modes. Of course, there are many, but I'm gonna show you some of the more obvious connections to get us thinking more clearly about their interconnections, okay?

Let's get going.

Okay, how to read CSV data, or comma-separated values. In this section, I'm going to review this particular example, CSV data format, and the data involved here in relationship to the chart that it creates that you see right here. I'll also talk about how this is actually one of the original charts that I was looking through in the D3 gallery as I was thinking about trying to create this, this temporal chart. And finally, I'm also going to compare this CSV data to Ridolfo's tabular data that you see right here. So we can start to see some cross connections between essentially the textual modes that we see in the CSV data and the Google Sheets data within the visual modes that we see represented in these charts.

So let's begin.

Here we have CSV data, and there's some ways we can describe CSV data. One description is called a flat file because it's in two dimensions. And by two dimensions, we can represent that visually as the following: Notice how there's line breaks between, let's say, these different lines that we see here. So those represent rows. And as we can see, every new line break, we can assume then, that's a new row of observations, of data. So I can draw a little line down here saying, okay, if we go down linearly, we can then see one dimension of the data, and then if we go horizontally this way, we can see another dimension at work.

Let's take a peek here. We have a couple things going on. We have this first line right here. Notice how it's different. This is called the header, and this includes the name of the columns. And then every line thereafter is assumed to be the beginning of the data. That is, for each column then, you can see now how the commas come into place. So if I highlight some of these commas here in this first line, oops, almost missed one. We have in the header, we have, let's see here, year. Let me switch colors for a sec. We have year and then we have the actual value of 1980. Then we have variableA, followed by underneath it, if we follow the scheme here, the second comma-separated value would be 70 and so on and so forth. And this repeats throughout the entire file. So that's how you read a CSV file. Let me erase some of that. If we compare that to Ridolfo's data in a table over here, we see similar things at work in a table.

So if I grab my highlighter, or actually I'm gonna grab my lines, we can see here we have the data moving vertically and horizontally in the same way. The only thing that might be different is you're wondering, like, okay, well, what about this little space right here? Ooh, that's ugly. This little space right here. Well, it's an assumption built into Ridolfo's table. We might label that in a CSV as weeks, and then we would assume then, instead of variable A, B, C, and so on, we'd have this particular year as labels, years as labels. We can modify them as needed, but then you can kind of see how, okay, the idea of two dimensions here is we have, we can compare week one to then the year and do that multiple times. And so the idea of two dimensions and the ability to cross compare something across a horizontal and vertical axis are embedded in both the CSV and table textual modes, if you will.

Now if we cross over now to the charts, recall how the CSV data, the textual modes, we have the, again, horizontal and vertical two dimensional modes. We have the header with the column information that then track the separation of values per column with the commas. So we have 1980, the year through 2015, and then we have all those variables A through D with different values that are then charted as follows:

So let's take a peek here. We have along the bottom, x-axis moving horizontally, the years. And then remember how every year had those different variables? So those four variables are then posted along the y-axis. So that's the second dimension. Same as the CSV. So it's just another way to represent, correct? So we can pull out the data, show it differently, so we can perhaps see it better over time. This was also influential in my own thinking, since I thought, oh, my variables are years. So each market year will have its own line, and that means if we go back to the tabular data, what becomes the x-axis? Well, it's the weeks. Each week is listed across the x-axis, if we go here. Here are our weeks along the x-axis, along of course, y then would be the number of postings and then our variables are the years. So we can compare for each year and each week of that year how many posts were there is the embedded question in that particular data set.

So that's how we can start to see some connections between the textual and visual modes. It may seem obvious, but hopefully we can, it'll open your eyes to, perhaps, some other kinds of charts that you see out there that might use something like a CSV data, and how to structure that information. But the main goal here, remember, how do we read something like this, this CSV data. It is a highly common, prevalent format that you might encounter, you will encounter. So it's good to know how they work and the best way to structure them and the assumptions that are usually built into them and how they compare something across two dimensions.

Okay, so let's move on to the JavaScript object notation data format.

Welcome back. This is how to read JSON, or JavaScript Object Notation. And I'm going to show you, again, the relationships between the textual modes of JSON objects and how they could be potentially rendered visually as a certain kind of diagram. So here on the left side, we see the visualization. And on the right side, we actually, this is the data for this visualization. So my plan is to walk you through the connections between the data and the visualization, how to read this JSON object, because it looks, if you've never seen one before, it might look kind of intimidating. How do you read this? There's visual cues that you might grab onto right away, with the hierarchies, but let's walk through it together. So I'll supply, again, these URLs and the links to these particular items, but if they're not around, you can definitely just follow along right here.

Okay, so first, how did I even know where to look for this data? If you're curious about learning about data and data visualizations out there, with some examples you find in the wild, one thing I'd like to show you is, well, you have browsers and you can always view the source of a page. It's a very powerful thing, right? So if I right click, click on View Source, I get this. So if you have an underlying knowledge about how to read HTML, webpages, you can navigate your way a little bit. And once you get stronger with your programming with JavaScript, you should be able to navigate these as well. But once you start also reading data, you should be able to assume what kinds of... At one point, this project obviously has to use data. So I can sometimes narrow down where it is and then look at it. That helps me understand what's going on in the visualization and how they're connected, the textual data values with the actual visualization somebody created. So it's a significant part of the process for me when I learn something as well.

So if I click Command + F4, Control + F, which is finding something on a page textually, I'll type in dot, in this case, I know it's .json, and I could also assume so. Also then see how there are some hits. We're not learning how to read JavaScript yet in the tutorial, but we can see at this moment right here that this is where they actually pull in the data to then visualize it later in this code. D3.json is, D3 represents the idea of the D3 code library that's being used here that you can go back up here where they pull it in. So the browser knows to use it 'cause it has a method called JSON, which will retrieve some kind of file, in this case, a JSON file. Sure enough, in the folder that houses this same HTML document, the path here to the data file is data.json.

So what I did is, I copied this, and if you look again on the right side here, you'll see the URL is exactly the same except it has /data.json and sure enough within the same folder as the HTML file, there's the data. Cool, so if you wanna peruse things in the wild, View Page Source, again, is always a beautiful thing. All right. What I wanna do now is compare this data over here with how they visualized it on the left side. Well, it's not too... Well, it's cumbersome to be able to walk you through the data, the JSON object here without being able to collapse different things and highlight certain parts of it, rather than this one big blob of it. (laughs)

So here's what I did. I copied, or I selected everything, copied it, and then I went to a prettifier app on the web. And you can just type in JSON pretty or prettify or format, whatever. It should give you something where I copy it in and then I can click on Make Pretty. It was actually pretty pretty already (laughs) but here's what this does for me.

Check this out.

Note here that there's a curly brace, and a curly brace in JavaScript and some other programming languages denotes this idea of what follows is an object. And in this case, a JSON object is what's called an object array.

So here's what I'm gonna do. I'm gonna close this completely. Notice how there has to be a closing curly brace. That's what's called scope. Scope is, you start something denoted by some, usually some kind of syntax. So in this case, the notation is curly braces, object begin, and close curly braces, object close. So the complete object is encapsulated by a curly brace, curly braces as you see here.

All right, well let's open it up a little bit. And now I'm gonna do something else. I'm gonna do this. I'm gonna close out this one. Inside this JSON object, there's actually only two values, two objects that are, as you can see here, look, there's a comma that separates them. One's called name. This is called, on one side, a key, colon, and then on the other side of the colon, there's always gonna be some kind of data type. In this case, we see two quotation marks that are empty, actually. Quotation marks like this mean a string. That is, usually find some kind of text in there.

So right now we see the value of name, or sorry, key, name, value, well, nothing. If we look at the visualization, check this out, we have the first instance of this chart that is hierarchal with nothing here.

Hmm. Let's go back to the data. Notice how there's a second value, a second object, with a key name children. And then, now let's take a look. Instead of it being a string data type, we have something else, colon, value of what? What is that? We have an open square bracket and a closed square bracket. That denotes what's called an array, a list of something. Let's open up that list of children. There's quite a bit. Let's collapse it, though. This is the practice you need to do. Oh, there's only two objects that are in this array of children.

Okay, well let's take a look at the visualization again. I'm gonna open up that first instance that's unnamed and now we have one, two children. It's pretty on the nose with regards to that key name, by the way, right? Children. So I thought that might help us here.

All right, well let's look at the line numbers here that are represented. We have four and it just goes right away to 57. That's gonna tell you there's a lot of content in between. So let's open up the first child. We have the name, key, value, a string of text-interactive tools. Some particular patterns you're starting to see already. I'm gonna notice here how there's another one, key free true, I don't even know what that means because I haven't looked at the code yet. There's a description with another text value. Again, I'm not sure how that's used. I might, maybe a tool tip, if I hover over. I don't know, let me check. Yeah, there we go. We see a little tool tip, interactive authoring tools. So they coded it so that when you hover, you get that message.

Cool.

Okay, well now we see another instance of a key name children with another array of objects. Because, again, new object is always, when you see that curly brace, that means it has to also be closed somewhere. We have two children under Interactive Tools, apparently. Let's check that idea. Sure enough, we have browser-based and desktop. Is that it? Browser-based, same thing, description. And so it's following a pattern. You can start to see the patterns. It also has children, which we saw. I'll close that out. And then the second child should be, what, desktop? Sure enough, it is. And it also, let's see, it has two children. Let's see here, maybe I missed something here. Browser-based, oh, it has many children. I just clicked the wrong one, so there we go. So it has an array of objects and it has four. So we should have four and two, four in the browser-based, one, two, three, four. Two in the desktop, boom.

So just through this quick glancing of, okay, here's a JSON object. There's a hierarchy being constructed of objects that are in lists, separated by commas, but you also have to consider how there's a key value pair to create those objects. So we have an object, in this case, the first hierarchical level is no name, and then we have children and then a list of children. We know that there's two on that level. We have Interactive Tools and coding, so I'm gonna open up that one with, it has its children. And we could keep going down the line, right?

I won't belabor the point, but I hope you can start to see that, in this case, it's rather on the nose, again, that hierarchy is represented visually and textually within the JSON object. We have denotations through keys, listed as you see here, that are separated by commas, but you can always start new levels of hierarchy by creating a new object, or a list of objects even separated by commas themselves. So we have Interactive Tools, open, close, coding, open, close. It can also have a list of children objects itself, right? Here's a list of children. Here's the first one. Browser-based, we close it. Comma, start up the next one, desktop.

One last thing I'll have you kind of understand and see and make sure you understand is that the commas separate every new object. But if you get to the last object, you'll see how there is actually no comma, correct? So the comma is to suggest that there's something after. It tells the computer, there's something new coming, so check for it. So if I were to put a comma here and try to run the code for this visualization, it wouldn't fly. It would actually throw an error and tell me they're expecting something. So that's one little small detail to also note.

All right, you may be wondering, why am I having you learn how to read JSON? One, it's, again, a very common data format that you're going to encounter, especially if you code with JavaScript in web-based data visualizations, including, especially, if you're using the code library D3.js code library. Another reason specific to this tutorial is that I had to retrieve Ridolfo's data as a JSON object. So even though the market comparison data that Ridolfo created is in a Google Sheet in a tabular format, that originally I talked about it in relationship to CSV, the CSV data format. The way I had to retrieve this dynamically is through a JSON object.

So how does that look? Well, let's take a look. If I get to the code here, and I'm not assuming you're learning how to read code yet, but what I can do is show you something real quick, what it looks like as a JSON object. So I can how you here in my documented code, I have a link to

Ridolfo's JSON feed version of his data. So I'm gonna go back. So here it is again in a Google Sheet, and Google enables me to then render it as the following.

Wow, right? (laughs) Can you imagine opening this up for the first time and trying to make sense of it? It's giving me a lot more than I ever would need in the data. But how would we even approach this?

Let's use that Prettify app again. And let me just show you what's going on here. It'll be good practice.

So I'm gonna open this up in a new window since we don't need to see everything, and I'm going to replace what we had before. I just copied and pasted Ridolfo's JSON feed into here. I'm gonna make it pretty over here. What do we notice right away? Open, close curly brace, suggesting one JSON object. But inside of it, let's take a look. We have, at the first level, three key value pairs, version in coding and feed. You can take a guess where I had to kind of dig into, what level. It's probably feed, right? So these are just headers, if you will, of version numbers and then coding information. Coding is just a standard unit code, international standard for all character types. That's about as much as we need to know here.

But inside feed, we have a lot going on as well. Let's take a look. I have ID, updated. I'm just gonna keep going through the line. Category, title, link, author, some other things.

Oh, there we go.

Inside feed, I have one, two, three, four, five, six, seven, eight, nine, 10, 11, and 12 items at the next level. So within the data object, I have feed. Inside feed, I have a lot of things. So this is how I went about it. This is how I read JSON objects. I try to simplify them. For the sake of time, I want us to look at the last one, under entry. So Google has defined a standard for their Sheets JSON object and entries are where we have our data. Inside there we can see, look, entry is actually gonna be what? What is this called again? It's a square bracket, so it's opening it up and closed. That means it's a list of something. Most likely it's gonna be what? A list of objects. So we could just keep going through here and...

(humming)

Keep going, how many are there? If you knew the data and how weeks Ridolfo looks at, it's 17. And sure enough, if you counted them all, we got 'em right there, weeks one through 17. Inside each entry, we then have multiple things we could look at, ID, updated, category, title, content, link, and then all of a sudden, we have something that should look familiar all of a sudden. We have this gobbly-gook (laughs) stuff right here that means something with regards to how, one, Google defines certain cell values. But also it may be perhaps how Ridolfo might have defined something. I'm not sure. I never looked into that. But what I knew is that there's a key of this value that then has a list of objects inside of it, or multiple objects inside of it. It's actually not a list, it's just multiple objects.

So the first object, we have even, inside each one, another nested... Sorry, that was what I was alluding to. It has another nested key value pair. So we might take a guess that T means the cell value and how it's been typed in the Sheets. When I say typed, I mean the data type. So T might represent what? Text of some kind or maybe some kind of open text value. So here we have a string, week one, and then all the week one values per, what, year. You can see how the year data is not here, right? That year label, I should say. So that's something I noted right away. I'm thinking to myself, hmm. I have week... I have each week as an object inside of entry, and I get to these particular per instances. Let me close out that one. And I have week two here, and rinse and repeat. But I don't have the year information.

Hmm.

How am I going to restructure this 'cause I'm gonna need that, correct? I can't just not have that label. So how do I put them together? It's not anywhere else either. So that's the one thing that I was thinking, like, okay there's gonna be some data processing there. And I'm gonna have to think about how to actually reorganize the information so that I can also make sure I bundle the year data in relationship to this week, per week postings. So that's, if you will, how I came to the realization, oh, okay, I got some work to do before I create that temporal chart that's multi-lined.

This has hopefully helped you provide some tools to take something like a JSON blob object and turn it into something you can really simplify, start to see the hierarchy and the different types of data that can be included inside of a JSON object. There are other things that can be included, and of course this will look differently based on things you find in the wild. But again, this is a baseline video to help you get there and feel more comfortable about at least where to begin.

And so let's move on to the final video that has to do with a particular data structure called a matrix. In this part, we're gonna learn how to read what's called a matrix, which is just a list of lists that help you develop some cross comparisons to see what kinds of relationships are between those different variables.

So one common visualization that you might be aware of is a relational chart called a chord diagram. And in this example here on Nadieh Bremer's page Visual Cinnamon is a very particular kind of chord diagram. But it's a great example because she breaks down the relationship between the matrix that she developed in relationship to her chord diagram idea. And in this case, we have two main categories of variables. We have variable A, B, and C on one side. And on the other, we have X, Y, and Z. And in this example case, these are just abstracted general names for, as we'll see below, values having to do with education versus occupation.

So we have six variables that have been broken up into two main categories visually here on this chart, and it then shows you the persistent links, the amount by the bandwidth, right, and connecting to certain other variables over in the other set. How did that become a thing and how is that actually connected to the data set itself?

If we look down here, this is what we call a matrix. So a matrix, as Bremer notes here, are meant to help you cross compare things. So how do you read this? We actually have, again, kind of this idea of a two dimensional thing. But it's ordered in such a way that there's actually more going on than just, let's say, a simple CSV. We have repeating variables. So we have, again, X, Y, Z, C, B, A connect to the main variables we just saw before. And in this particular case, we have lines that Bremer provides for us that are implicit in the structure of the data.

So what we do is we do it like this. We compare the idea of how many times this X connect to X, X to Y, X to Z. Obviously not any because in this case, they are part of the same category, but now we have X to C. Oh, they can be compared or there's a link there 10 times, X to B five, X to A 15 times. So if we go back up to the chart real quick and compare X to those X occupations to these values of C, B, A in education, we have X to A, X to C here in yellow, and then X to B here, and B was five, I believe. So that's why it's the thinnest band between the two.

So that's how you read a matrix, and she breaks it down easily that I won't belabor here. But the main goal to see is how you organize your information in the matrix to facilitate cross comparison and really correlation between existing variables in a data set. How many times do they overlap? Another name for it is co-occurrence. And so how many times do these co-occur?

And that's how you read a matrix.

And how does this look in a code and why did I say it was a list within a list? If you look down at Bremer's note, is this what it would look like in JavaScript, you can define a matrix. And then you can see here. Remember how square brackets mean that's a list? And it ends here. But within this list we have one, two, three, four, five, six lists, each representing the different variables. Now we can see how this idea, we have, again, where we actually get those values so cross comparisons connect back up to here and that facilitation of it, right?

So that's how matrices work. There's actually a lot more statistical knowledge bound up matrices that I won't belabor here, but the whole idea is, how can we test inter-relationships between variables in a data set? That's what a matrix is designed to do. It's a more complex form of comparison, based on what kinds of questions you have of the data. And then obviously, too, that's connected to the way you'd wanna visualize it. One example that I think is widely used is the chord diagram.

And again, Nadieh Bremer's work and her blog posts are really good at breaking down, how did she get from her data to the actual visualization. So here's a great example. If you care to read more, this is why I highlighted this blog post here.

Okay, those are the three main types of data formats that I really wanted to highlight in this tutorial series. So I hope you have a better sense about how to read CSV, read a JSON, Object Notation JSON file, as well as understand that curly braces and square brackets mean different things. Commas help delimit the amount of things within, let's say, these lists and objects and objects that have lists inside of them, right? So we can see already that these forms of data sets and different formats provide a really interesting, complex capacity to define relationships and then ask questions and facilitate an analysis. That's what we're trying to do with visualization is test ideas we have about something in the world. And then visualizations are meant to then highlight, illuminate some aspect of that.

So I hope we can take this new knowledge about how to read data sets in different formats and structures into the third video of this series where we actually look at the code of the Rhetmap case. And that's the next part. How do we take a look at how I took Ridolfo's data and processed it into the visualization that we see on the web right now?