EPISODE 1250

[INTRODUCTION]

[00:00:00] JM: Spatial technology impacts every person who uses a smartphone, drives a car, or flies in an airplane. It refers to all the technology used to acquire and interpret geographic information. In more advanced settings, geospatial technology is used for constructing dynamic maps, 3D visualizations, and scientific and governmental simulations. The company Makepath specializes in geospatial technology, and full stack application development. Makepath helps companies to create beautiful and simple visualizations from mountains of complex data. Using open source Python libraries and real world validation, they create analytics, web applications and other automation processes. They're passionate about the open source ecosystem and contribute to many ongoing projects. In this episode, we talked with Brendan Collins from Makepath. He's a founder and principal at Makepath and he previously worked as a principal of parietal and was a software developer at Anaconda. We talk about geospatial data science and Makepath.

[INTERVIEW]

[00:00:58] JM: Brendan, welcome to the show.

[00:01:00] BC: Hey, Jeffrey, thank you.

[00:01:02] JM: There is a growing ecosystem of tools around data visualization of geospatial data, and processing of geospatial data. Give me the state of the art of geospatial analytics and geospatial data platforms. Why is this important today? And why is it growing in importance?

[00:01:22] BC: Yeah, that's a great question. I think just to start decomposing what is geospatial data and geospatial technology into some of its component parts can be helpful. So on the spatial data science front and geospatial in general, we think a lot about the capture of data and the storage of that data, but then also the analysis and visualization of the data. And each one of these component parts advances at a different rate. We get new sensors that come online that get attached to different remote sensing vehicles. That might be a plane or a satellite. And we go through processes of understanding how to best store that data and get that to the people that can analyze it and derive information and insight from it. So as we move, say, from a satellite, to a blob store somewhere on the cloud, pre-processing satellite images, extracting relevant information and trying to figure out the best action based on that information. That entire tool chain has been evolving over the past, really, since the late 60s with people like Roger Tomlinson, defining what geographic information systems were and defining the fundamental data structures for storing things like arcs and vectors and rasters.

So the last 50 years, we've had an evolution of that technology. And the area that I'm most interested in is on the open source side. So as we look at taking geospatial technology, and allowing more people to access it more easily, and being able to have transparent tools that we can collaborate on, that building communities around open tools is one area where I see this technology really taking off right now with libraries like GeoPandas on the Python side, GDAL as one of the fundamental geospatial dependencies that's all open source. So I see in breaking down geospatial into its different components, the area that I know most about is on the analysis side.

So I look at what are the tools out there to analyze increasingly large datasets are increasingly growing datasets. So those technologies are not specific to geospatial in many instances. Those involve, “What are our tools for scaling and analysis horizontally across many machines and clusters of machines? So we can handle planetary size questions.” And then also, how do we make our algorithms faster and more accessible and more extendable? I would call that and kind of more on the vertical scaling side. So just to start the conversation there, certainly a large question, but I think on the on the analytics, it's about making the tools easier to access and being able to scale the tools to larger problems.

[00:04:13] JM: And when I think about the applications of geospatial analytics, I'm mostly thinking about like problems where there is a human in the loop, like where you have a human analyst looking at a visualization of geospatial data, for example, to decide where to open a new restaurant, for example, looking at foot traffic to decide where to open a new restaurant. Is that where most of the problem domain is? Or is there also a problem domain around just do calculations and kind of doing like machine learning around just calculating problem sets where you have something like essentially like a location graph?

[00:05:01] BC: So site selection for new locations, for a restaurant or a chain of businesses is certainly a geospatial problem because it involves the question of where? So anytime we're asking the question where, there's a good chance that geospatial may come in there. On that question involves is so fundamental, that we have to figure out different ways to represent the fundamental phenomenon of the world. And that's where we go in deeper into the geospatial world, is about how we represent the phenomena of the world in a computer. But specifically on machine learning, the question of where is often a decent place to start to find new features to build models on. So say, you're looking at a topic and you want to do feature engineering, and add additional dimension, say, to your data frame before you start training, then looking at geospatial technology to understand, say, the closest police station to the location. That would be a question you can answer using GeoPandas or using Xarray-Spatial, one of the open source libraries that we work on it Makepath, and then add that as a new dimension to your model for learning.

So there certainly is just the where question. So anytime that we have where, we potentially have a geospatial problem to solve, and we see that reflected in many of the technologies out there. So if you look at a database like PostGIS, which is a spatial extension for Postgres, that will allow you to ask the question, “Is this row in the database within 10 miles of this other row?” That's a where question that we can apply to relational databases using geospatial extensions like PostGIS.

[00:06:51] JM: Tell me more about the application domains. Let's talk a little bit more about the high-level application domains that are in need of better geospatial tooling right now. And then maybe we can dive a little bit deeper into the engineering.

[00:07:06] BC: Yeah. So one of the original kind of geospatial analyses that people point to is the Jon Snow cholera outbreak map in London in the mid-19th century, looking at the relationship between wells, and drinking water, and cholera outbreaks. So this is in epidemiology. We just went obviously through and are currently in the COVID pandemic. And the question of where in the COVID pandemic is really important, including things like contact tracing. Looking at the location that someone went as they traveled potentially with a COVID diagnosis. So the high-level applications apply to many different domains. So if we think of data science in general as being mathematics, plus computer science, plus a domain, then geospatial falls into that math and computer science area, and then as applied to a domain. So at Makepath, we’re a spatial data science firm in Austin, Texas, and we find ourselves working in finance, in oil and gas, in natural resource management, environmental conservation, retail, because all of these domains questions that involve where. And so at the high-level application, it can be things like quantifying the amount of carbon in a stand of forest so that we can try to attribute $1 value to the wood that's existing and capture an externality that is the storage of carbon. That's an application in the natural resource management area that we work with groups to try to quantify carbon and look at hitting climate change goals for companies.

Then on the finance side, there's many times geographic questions that can affect a certain sector where financial groups are looking at what's the effect exactly of a vessel getting stuck in the Suez Canal, for instance? And what are some of the downstream effects geographically? And how do you quantify that? A lot of that may be quantified off of things like ship trajectories, where we try to understand like how far were these ships going? And what's the implied cost to having them be delayed? So the question of where it comes in almost across all domains because where is the context in which we're doing things, right? So if we're trying to understand the world better so that we can make better decisions. And that's where at a high-level where the geospatial technologies come in?

[00:09:39] JM: What is the stack of technologies that you need to build geospatial analytics? So for example, I'm just taking a guess. But at the visualization layer, you've got a stack of technologies that may be render in a browser or render in – Or like you're using Python-based tools to just render on your machine. But there is some data pipeline that leads to an output that gets consumed by this rendering system. So I'd like to sort of get the architecture for the data munging and data production side of things, versus the data representation side of things. Like do I need to use like Spark or Presto and then like get an output that's like a CSV file? Or is it like a streaming data system? Or is there's some geospatial representation? And then what's the pipeline to convert that into a visualization?

[00:10:44] BC: Yeah, great question. So I think it's important to start with how at a fundamental level, how we represent geographic phenomena in a computer. So there's two predominant ways that we do that, that map to the types of phenomena that we see. So first is the vector data format. And while there's collision in terms between disciplines, vector is certainly one of them. You go between disciplines, and the word vector means something different or slightly different. In the geospatial world, when we say vector data, we're referring to points, lines and polygons, or discrete phenomena that we can represent as sequences of coordinates. Those are things like highway would be a line feature. While maybe a gas station is a point feature. So we represent – That's the first type. The second type would be how do we represent continuous phenomena? So there're things like rainfall and elevation, distance, and even soil types that can be better represented as a regular grid, then a sequence of coordinates. And this is the fundamental nature of things outside of geospatial, right. We think of Adobe Illustrator as our vector editor. We think of Adobe Photoshop as our raster editor. And those same two types of data make it into the geospatial world. And there're different formats to hold each of them.

When scaling, people are generally more comfortable in the geospatial world with vector data with these discrete features that you can have mapped attributes, and then you can do things like easily partition a file of points across a Spark cluster and do an analysis, a point analysis. The raster side is more the domain of image processing. So we think about our image formats, the classic ones, we're talking TIFs, JPEGs, PNGs in the sequence of getting, say, a visualization to the browser. At some point, we're going to want to be in a browser native raster format. And we know the PNG and the JPEG as raster formats that the browser understands.

Now how we go and move from, say, a planetary size, petabyte partitioned set of TIFs in a blob store to a PNG that is nicely color mapped for a user to see to gain some insight from the data? That is a lot of what we do on the open source side in a project called Datashader. So Datashader generated some maps – For anyone, I'm not sure if they can see behind me, to allow us to take large data sets and find structure in those data sets without using brittle heuristics, like sub-sampling or transparency and having an intelligent way to go from vectors, discrete sequences of coordinates to rasters, which are regularly gritted “images”.

So one of the areas when we're doing munging and processing is understanding, “Is this phenomenon better represented as a vector? Or is it better represented as a raster?” It's a fundamental question. And playing between the two is where you can do some interesting analysis. As we may imagine, doing something like a suitability analysis. If we're looking to decide on a new location for a farm, for instance, there're going to be many different factors that come into that, including discrete data, like distance to a river or a distance to a highway. We're going to have to deal with those river and highway features. And then also continuous data, like elevation and rainfall that can help us build a model to site select our new farm. And a common way to do that is to rasterize everything into a common resolution, and then do a thresholding to see if you're within the thresholds of each one of those dimensions for finding your new farm and being able to generate a single image that shows the suitability of a given pixel in the image at a given resolution for farming.

So the conversion between vector and raster is important. The data formats on how you store data are important. In general, when we're thinking about how we store data, I think about a couple of things in terms of data formats. And these are not specific to geospatial This is just data science in general, right? So I'm looking for binary formats that I can partition into chunks that I can add compression to, and that in the case of a vector set, I'm looking to store by column. So if it's a columnar store that includes good compression support that I can partitions so that I can run a cluster against it, and it's binary, I'm pretty happy.

Now there're other formats that I tend to also use things like GeoJSON, which is a very helpful format, text-based JSON to represent vector data or sequences have coordinates, right? And that's really nice because it's human readable. So as I'm developing an application or a dashboard, I can include a snippet of GeoJSON that represents a discrete area. And that's easy to update. And it's easy for folks to look at. And if it's small, than having it be binary on disk isn't that big of a worry to me.

So a lot of the interplay and building the applications has to do with making sure that we get the geospatial data into a format native to the browser, if that's what we're targeting, is like a browser visualization. We have to think a little bit about where in the browser we're rendering it. Are we rendering it on SVGs? Are we rendering it on a canvas or within WebGL? Those are all questions that we would want to have the data transformed into the appropriate format for targeting that area of the browser. So not just the browser, but what in the browser are you doing?

[00:16:56] JM: Okay, you've given a decent overview of the modern state of data visualization and data visualization, application architecture. And I'd like to ease into a discussion of what you are working on. So I guess, let's go by way of example. So to understand what Makepath is, could you give a few examples of how customers use your software?

[00:17:26] BC: Yeah, so Makepath is a services firm. We have a heavy focus on spatial data science and open source. And we take open source projects that we create and also are created by others and leverage them to solve our clients' spatial data problems. That's fundamentally what we do. We don't sell a software product, but we do create software communities. Right now, Xarray-Spatial, which is a spatial extension for the Xarray library is our main focus. And our focus there is to make the tool accessible to geo spatial analysts.

So what we're fundamentally doing there is taking tools from general data science, so things like that we know and love in the Python community like NumPy, and Pandas, and Dask, and Numba. We're taking these libraries and implementing the core spatial algorithms that we need for analysis using language that geospatial analysts understand. So there's a cross-domain area here where for instance, within NumPy, we would call slope, or the change in elevation, we would call that – We'd name it something like gradient or NP.gradient, or in geospatial land we call curvature, which is the second derivative of elevation. On the other side of the – That way we may call that acceleration within physics. So there's a lot of tools that already exists that can be leveraged for geospatial, but they're not named in ways that geospatial analysts understand necessarily. So we're going and implementing geospatial algorithms for raster data specifically with Xarray-Spatial, and then "spelling" the functions in ways that geospatial analysts will already understand and be familiar with.

[00:19:26] JM: Is there a relationship between what customers come to you looking for help to build and the open source projects that you build? Like, is there a flywheel where they come to you with problems and you're like, "Oh, this is actually a gap in the tooling ecosystem. We can build an open source project to alleviate this."

[00:19:47] BC: Yes, definitely. So there're two specific cases right now where we're working with companies to try to fill gaps in the open source ecosystem. The first with Xarray-Spatial is Microsoft's Planetary Computer. So Microsoft is launching that. And there're kind of announcements of that yesterday for some soft launches. But we're working alongside Microsoft to fill in a gap in raster processing. So right now we look at providing huge data sets to Microsoft users in a hosted environment with open source tools to do analysis. There's a powerful combination there. And in looking at distributing that analysis across a cluster of machines, Xarray-Spatial is one of the libraries that's actually trying to address that. So we look at our algorithms, and we consider them all for distributed and GPU situations where we want to leverage task schedulers and leverage CUDA to do better analysis at larger scale. So Microsoft right now is the main sponsor for Xarray-Spatial and we continue to work with them to fill in tool gaps, whether that be physical tools in passing kernels across raster data, zonal tools, things like distributed proximity tools and pathfinding, viewshed, which is like one step array tracing. Just like line of sight across a terrain. So those are some of the use cases that were gaps within distributed computing that we were interested in for the planetary computer, which is part of AI for Earth.

On another front, there're disciplines where there's existing tools, and they work well, but they just need to be a little push forward. So 2021 is the year of Bokeh for Mackepath. Bokeh is a visualization library written in Python that is based on the grammar of graphics book, understanding how to compose beautiful graphics. There are many geospatial features within Bokeh. But this year we're really focused on publication quality outputs. So do these tools give you something that you can submit to a journal and feel confident about? And that's what we really want to focus on this year on Bokeh is allowing for people to add LaTex to their Bokeh plots, and also export SVG and be able to support those on specifically bio science use cases. And that's through a sponsorship from the Chan Zuckerberg initiative in pushing forward bio science with open source.

[00:22:29] JM: Can you give me a little bit more on the background of Bokeh? I know this project has been around for a while. What's been evolution?

[00:22:36] BC: Yeah, that's great. And so going way back, so Peter Wang, who’s the CEO at Anaconda, he created a library that's called Chaco, which was a visualization library that later became Bokeh not in a direct line, but certainly through spirit. And Bokeh has been – Well, started by Peter Wang, has really been built out with Bryan Van de Ven. So Bryan has been leading that project, at least probably since 2014, is my guess, but I might be wrong on that. And he is the main force in organizing that and pushing it forward. So Bokeh, as it t has evolved, has gained new components, including Bokeh server, which is a WebSocket based server so that you can write things like interactive callbacks within Python and have them execute from JavaScript. And also adding in support for different visualization types and plot types and higher level plotting abstractions to allow people to express commonly made plots easily. So Bokeh provides a lot of the fundamental structures to create plots, but for a little bit was a little hard to use. So there are some higher-level abstractions on top of Bokeh, things like Hollow Views and Panel are two libraries that heavily use Bokeh and provide higher-level abstractions for layout and for plot types. But Bokeh has an amazing community of contributors and also does a lot of evangelism, sponsoring different groups like PyLadies and SciPy and hold sprints in places. We have a weekly Bokeh dev meeting, which anyone can join, and that helps push it forward. But yeah, that's a kind of a high-level description of Bokeh.

[00:24:33] JM: We've done a few shows on some of the modern Python tooling like Dask, and I guess there's also Ray, which is maybe not as closely related. It's different than Dask. It’s been a while since I did their shows. I don't remember the comparison exactly. But can you give me some detail on the modern state of Python parallelism and how that relates to geospatial?

[00:25:08] BC: Yeah. So one thing just to start as a public service announcement, if you find yourself in Python importing multiprocessing and writing your own multiprocessing tool, you should definitely be looking at Dask and Dask Bag, because it's the simplest way that I've seen to do multiprocessing in terms of multi core with in Python. And one of the great things about Dask is that it wasn't specifically developed for clusters. While there is a distributed, which is a distributed task scheduler that's packaged up with Dask, Dask works just fine on a single machine. So there are still issues where we think about parallelism and multi-threading versus multiprocessing within Python, and the global interpreter lock and different issues that come across when using threads and having problems with compute bound operations using threads. And in Python, that remains. There're certainly interesting ways to get around that in releasing the GIL using projects like Numba. So we use Numba to be able to write Python code, really, a subset of Python, to get around the global interpreter lock and be able to do real multi-threading in LLVM code, and that's the magic that Numba does for us.

So when we think about parallelism and scaling, there's one thing which is just making your tight loops fast when you're trying to get through tons of pixels. And then there's also being able to do that in a multi-threaded environment, or in like a CUDA environment. And those are things that we're addressing in Xarray-Spatial specifically for geospatial.

Now Dask, Dask really helps us, because Dask provides the Dask array, which is a distributed ND array. And we need that for our raster processing, specifically to do things in a distributed context, like dealing with edge effects. So as we're, say, applying a focal kernel across a distributed image, we need to be able to handle those edge effects in a graceful way that's easy to express to the user. So doing things like having overlapping partitions and figuring out which partitions we need to load in to do an analysis in blocks, in chunks, is one of the items that we're able to express through Dask.

So when I'm thinking in concurrency and parallelism, that obviously applies to one machine, but then also the cluster situation. And also, obviously, compression is a big component there, where there're some large big data problems that seem like you may need data a Dask cluster to solve, when really, you just need better compression. And so we look at projects like [inaudible 00:27:53] who's based in Spain, that is one of the compression libraries that we look at and using data formats that help us to reduce the size of problems to not need to involve the extra complexity of a Dask cluster, even though that's becoming easier and easier all the time with the hosted Dask clusters from coiled computing, if you're familiar with coil computing. They're a great way to hook up your cloud provider, whether it be like AWS, or Azure, to a Dask cluster without having to go into all the workers and install your dependencies and administer the Dask cluster.

So just to start, the fact that Dask clusters are easier to get going now with coiled, plus libraries, domain-specific libraries that use the Dask data structures, things like the Dask data frame, the Dask array and the Dask Bag. Those are the path forward, at least for geospatial that we're focused on, is how to bring these tools that are perfectly – That work really, really well, but weren't designed for a geospatial context, and wrap them with algorithms and names that the geospatial folks know.

But it continues. I mean, the issues that come up in distributed computing are certainly a level more complex than on a local machine. So we always do our analysis first on a subset of data on a local machine. And then we ask the question, "If I rent a huge cloud server, can I just run this there? And would that be enough?” And if I can't rent a machine with 128 cores to solve my problem, then I will then start thinking about HPC and clusters, and that's where I definitely go to coil to set that up, because I'm going to be running on a cloud somewhere and I don't want to do the administration part of setting up a cluster.

[00:29:52] JM: And can you explain, like if I am trying to run some kind of data visualization algorithm, like a clustering kind of algorithm, like I want – Like let's say I am trying to find some sort of measure – Let's say I want a visualization that is a measurement of the most likely places to have a very good new Starbucks. Like if I'm Starbucks, I've got a ton of Starbucks has already, I know a lot about the United States, I know where coffee shops are. This is like a NP complete problem. It's like super hard to solve. But I want a visualization that gives near good representation of what might be a good visualization of where I should place my newest Starbucks. I want to parallelize that. I want to rent a bunch of servers in the cloud. How are those servers collaborating on creating that data visualization? Is it just – Like can you just sort of delegate all of this to Dask and have Dask like figure out how to parallelize this thing? Or do you have to do some hard programming work?

[00:31:06] BC: You shouldn't have to do too much hard programming work. All the tools that you'd need to do that sort of analysis are available within the Py Data ecosystem. So in thinking about that site selection for a Starbucks, we would make sure that we had all the spatial data for the components that we were interested in. We'd make sure that we have all the highways and roads, demographic information based on census, information about past sales and any other dimensions that we know from existing Starbucks locations, and be able to do that cluster analysis and rate – Maybe we rank the clusters by not so good to outstanding location in some maybe pay means or something like that to find those bins. Then we have a little bit of a question of presentation to the user and at what scale. So when we look at that at the global scale, we want to be able to intelligently resample, whatever we're doing, so that the users can make out structure at a global scale, and be able to zoom into an area and then see local scale structure. And that might mean that we're rerunning a suitability analysis using a different study area dynamically as we're as we're going. So being able to have a cluster –

Maybe I should just back up here and say, fundamentally, what Dask does. So dask executes tasks, graphs, and there're different data structures to help you form those graphs. But at just a fundamental level, that's all that Dask is doing, is it's coordinating work between workers and a task scheduler to execute a graph and retrieve outputs from that graph. So Dask provides that piece of infrastructure. Now, Datashader is a library that would allow you to actually build that visualization and do things like resampling, and color mapping in a distributed context, because Datashader takes the Dask data frame as an input to its operations for its graphics pipeline.

So we look at these other libraries that build on top of Dask for domain specific problems. So if it's converting all of your points into an image, and you have a trillion points that are divided up into maybe one billion chunk partitions, then you'd be able to spell that or write that out in Python in a sequential way without doing things like addressing individual partitions or addressing individual workers. That is the job of what Dask does.

The logic on what to do on that data may fall on the Dask side if it's a kind of data frame type problem, or like a NumPy, universal function type operation, like doing a standard deviation, or doing a rolling window. Those are the sorts of operations that you can do using the Dask data frame and the NumPy universal functions. But then for other things like, "Hey, I want to color map using the mins and maxes across all partitions so I can stretch, say, my purple value to green value. I want that stretched using the min and global min-max of my data set. But I don't want to address individual chunks in half and basically write that blocked algorithm myself." That's what Dask can do for you with Datashader and other domain-specific libraries on top of it that integrate with Dask. So Xarray-Spatial that we're doing integrates with Dask for doing classic 1D classification algorithms that geospatial folks know. These are things in the geospatial side like calculating bin quantiles and natural breaks, which is like 1DK means. Those are some of the classification algorithms that we do. But we also do zonal algorithms, where we're trying to do summary stats for specific zones within an image. And those are also things that geospatial folks want to be able to do and they want to be able to do at scale. And they're not going to necessarily come in with the distributed computing and HPC background to understand how to use MPI, or the other distributed tools that computer scientists are really great at, but geographers need a little help in expressing that sort of analysis.

[00:35:46] JM: Okay, very interesting. Are there some specific bottlenecks in application development given the current set of tools? Like is there a need for better tools? Is there some specific improvement or set of improvements that needs to be made?

[00:36:05] BC: Packaging is always a battle. It’s not particularly pupil dilating stuff to build better package managers and be able to install and manage dependencies, it is super important. Conda is a package manager that continues to try to solve that problem. But I would say packaging, and removing what I call like creativity killing problems. So I have an idea, I have a great idea for a new analysis that's going to knock someone's socks off. But then in the process of going through it, I have a problem with a dependency or some other dependency, and then I'm troubleshooting an environment problem, and I'm not focusing on my core question or the core work that I want to do. So moving in marketing speak, time to value for analysts is super important. And packaging is one of those areas that continues to be a pain sometimes. It's gotten better. But that's one area where we're focused on Xarray-Spatial is not having C extensions. So we need the speed of C as we're implementing, say, like an image processing algorithm.

But when we include C extensions, we need to do builds for different platforms. We need to make sure that people have certain build dependencies installed locally. And we get around that by using a just-in-time compiler called Numba. And so if there's one library that goes with Dask, in my mind, it would be Numba, because Dask covers our horizontal scaling and Numba covers our vertical scaling of making our algorithms faster. So what we're able to do is stay in one language, avoid the C extensions, which makes packaging easier, and then still gain the speed that we want and allow people to extend the library in Python without having to contact switch to another language, which I guess maybe a second area is just extensibility and transparency. So allowing the folks that are using the tools to see the code and extend the code for their own use cases in an easy way. Like I use GDoll, but I don't regularly extend the GDAL. And the process of extending GDAL is intimidating to me, and that I don't have a huge C++ background. But it still is an intimidating thing for me to go in and think about extending GDAL, extending something like Xarray-Spatial, which has a much more limited breath than GDAL.

I mean, GDAL is kind of the ultimate raster library that we have in the open source. And we're basing a lot of what we do on them. And if it wasn't for them, we wouldn't be able to do what we're doing. But it is much easier to extend Xarray-Spatial and to read the code and understand the code than the GDAL case of really using GDAL as a dependency that's wrapped up with

Python bindings. I guess just to summarize, it’s packaging, extensibility and transparency into code, which obviously ties into extensibility. But those are the major things I see.

[00:39:13] JM: As we begin to wind down, I'd love to get your perspective on the industry and how data visualization fits into the modern software industry, geo geospatial data visualization specifically, fits into the modern industry and how you expect it to change in the next 5 to 10 years. Are there certain extreme changes coming soon that you see? For example, one of the companies I see you've partnered with is SafeGraph, and we've done a bunch of shows with SafeGraph. I really think it's an interesting company, because they're basically like a self-serve data platform. And I think democratization of data is a big deal. If you have democratization of data, you have all kinds of new applications that can be built. So maybe that's one trend you could explore. But I'm just curious to get your thoughts on where things are going.

[00:40:07] BC: If I can drop it just a terrible platitude, I would say that a picture's worth 1000 words to some degree. With all the data that's coming out, and companies like SafeGraph, that are doing amazing work on the data side, it can be really difficult to tell your story as just a pure data company. And that's a lot of the work that we were doing with SafeGraph is being able to show examples how to use data to move from information into insight. So what are we going to do with this data? How do we communicate it so that those insights pop? And those might not come from the analyst, exactly. Those might come from someone else in the organization, or someone outside the organization that looks at a picture and can see, “Hey, there's something weird here.”

So the ability to tell the story is fundamental. And visualization really speaks to that and speaks to one of our senses, which is our sight and our ability to interpret different degrees of color to some degree. So being able to tell your story, and being able to make that story very simple for folks is great. Within the area of exploratory data analysis, being able to see the relationships between the dimensions in your data is extremely helpful to find outliers and to understand the relationships between variables. Data visualization and scatter plots has been used obviously for a really long time. But how do you scale that to larger data problems? If you have a trillion points, and you want to create a scatterplot, to understand the relationship between variables, or multiple distributions, how do we do that in a way that doesn't use brittle heuristics? And that's a focus of Datashader. And I see that growing and continuing to be used as a way to intelligently aggregate your data to the pixels on your screen so that you cannot use things like subsampling, or transparency, or different stacking heuristics and having reproducible tools that allow you to visualize things without having to go in and tweak it specifically for that data set.

There're been great advancements just on the color front. Like if you look at something like Cubehelix, and all the amazing color ramps that have come out over the last really like decade, but then it becomes super popular, like Veritas or these different color ramps that have been tuned to be perceptually accurate for the human eye. And to take into account the fact that we have a nonlinear interpretation of color. And we skew our rods and cones skew green, like we were able to see many more shades of green than we are of other parts of the visible spectrum. And so knowing advances in neuroscience can help advances in visualization. That's one where they go together. And as these disciplines grow, and we see these interdisciplinary activities where computers are being used more and more in new scientific fields, those tools are going to trickle down.

And we're going to take examples from, say, physics, and we're going to apply them to geography and examples from geography, and we're going to apply them to oil and gas. That's one where the geo folks are great at tiling. If there's one thing we know in geospatial world, it's being able to tile a large data set. And we know that from Google Maps and MapQuest back in the day of chopping datasets into 256 by 256 tiles. There're many datasets that need that. I think if you look under the hood of non-geospatial applications, you'll find geospatial technology helping with things like tiling, and we take tiling, and we apply it to domains outside of geo, specifically looking at large acoustic plots, looking at large amounts of temperature data over time, applying some Fourier transforms and building some tiles for signal processing. That's an area where geo and other domains are going to be collaborating.

And what I hope to see is open source technology being used more within the public sector. So state and local governments that are finding that open source tools are reliable enough and easy enough to use, that they can be used by their analysts in context where there might be high-turnover, where you need continuity between different groups, and improving the open source tools so that local and state governments and national governments can use these tools to solve their problems, reduce redundancy between organizations, and also avoid expensive license fees. So that's one of the areas that I hope to see over the next decade increase is open source adoption within the public sector. We're not specifically looking at the public sector very much as we do this, but I do see the need for it as I interact with GIS meetup groups, and I talked to folks internationally, the role of open source and extending this technical abilities to places that aren't necessarily going to go out and get a very expensive proprietary package just for geospatial.

[00:45:35] JM: Awesome. Well, Brendan, is there anything you want to add before we close off?

[00:45:41] BC: Well, one, just thanks so much for having me here. It's really great to get the word out about Makepath and Xarray-Spatial and our open source tools. And just a big thanks to a lot of our partners that have helped us so far in getting where we are including Anaconda, and QuanSite, SafeGraph, and Microsoft, and CCI and all these guys. And they're helping support our open source tools. We also would love to invite any listeners to get involved. If there are folks that have experience in open source, that's great. If you don't have experience in open source, that's great too. You can come to the project and we can find some issues to help you get going. I encourage anyone who is interested in doing open source to simply start, and you can go to github.com/makepath/xarrayspatial to go to our GitHub page and check out some issues and take it from there.

[00:46:33] JM: Brendan, thanks for coming on the show.

[00:46:34] BC: Yeah, thank you, Jeffrey.

[END]