Chapter 1. Getting to an Effective Visualization

Choosing or designing a good visualization is rarely a straightforward process. It is tempting to believe that there is one beautiful visualization that will show all the critical aspects of a dataset. That the *right* visual representation will reveal hidden insights. That a perfect, simple, and elegant visualization—perhaps just a line chart or a well-chosen scatterplot—will show precisely what the important variable was and how it varied in precisely the way to illustrate a critical lesson.

This is often the impression that we, at least, are left with after reading data science case studies. But in our experience, this does not match the reality of visual data analysis. It takes hard work, and trial and error, to get to an insightful visualization. We start by thinking about what we want to know, and we refine fuzzy questions into actionable, concrete tasks. We clean, reshape, and restructure the data into forms that we can put into a visualization. We work around limitations in the data, and we try to understand what the user wants to learn. We have to consider which visual representations to use and what interaction mechanisms to support. Along the way, we find other variables that tell us more about the dataset and that help clarify our thinking. And no single visualization is ever quite able to show all of the important aspects of our data at once—there just are not enough visual encoding channels.

Designing effective visualizations presents a paradox. On the one hand, visualizations are intended to help users learn about parts of their data that they don't know about. On the other hand, the more we know about the users' needs and the context of their data, the better we can design a visualization to serve them. The process described in this book embraces this paradox: it leverages the knowledge users have of their datasets, the context the data lives in, and the ways it was collected—including its likely flaws, challenges, and errors—in order to figure out the aspects of it that matter.

Put another way, this book is about the path from "I have some data…" to "We know this because of these clear, concise, and insightful visualizations." We believe that creating effective visualizations is itself a process of exploration and discovery. A good visualization design requires a deep understanding of the problem, data, and users.

Getting to Insight

We most often work with other people that have a dataset they are trying to make sense of. The process of designing a visualization usually starts when people walk into our office.

> *CLIENT: I have some data that I'd like to visualize. How should I draw it?*

The client seems to expect us to pull a visualization off the shelf, to sculpt that perfect visualization. We almost always frustrate them by asking what they hope to see.

> *Q: What is it about the data that you would like to visualize?*
>
> *CLIENT: I want to see how profitable our stores are.*
>
> *Q: What in your data indicates a store being profitable?*
>
> *CLIENT: It means that the store has lots of sales of high-profit items.*

Q: How does profit vary by store?

And so on.

By the end of this process, we often find that the clients do not have a visualization problem, but an operationalization one. Their struggles to choose a visualization stem from a lack of clarity about which attributes of the data are most important and how those attributes relate to one another. Once they can describe how the data attributes relate to the question they are trying to answer, finding an appropriate visualization becomes much easier.

We have learned over the years that designing effective visualizations to make sense of data is not an art—it is a systematic and repeatable process. We have systematized this process into what we believe are reproducible and clear steps.

This process tracks our understanding of four components:

Data

> What data is available, and what does it mean? What does the data look like, and what are its important aspects? Where did it come from, and why was it originally collected?

Tasks

> What needs to happen with the data? What are the low-level questions and tasks that will support high-level goals?

Stakeholders

> Who is involved with the data, the problem, and the goals? What can they say about the problem to help design an effective visualization? Who will view the final visualization, and what sorts of things do we expect them to learn from it? What domain knowledge do they bring to the table? What answers would they find satisfying?

Visualization

> How does the understanding of data, tasks, and stakeholders come together? What representations of this data will fulfill the tasks for the users?

Regardless of the visualization outcome, this process will almost certainly lead to new discoveries and insights. These discoveries help to inform the operationalization, but they will also likely steer the process down new and unexpected paths. The guidance and framework in this book are meant to help identify opportunities for discovering new knowledge and to make an otherwise messy process a bit more structured.

Hotmap: Making Decisions with Data

As an example of how visualizations can help you to better understand a problem, and help an organization make decisions, we can look back to 2006. Microsoft was rolling out its new mapping tool, Virtual Earth, a zoomable world map. The team behind Virtual Earth had lots of questions about how their users were using this new tool, so they collected usage data.

The usage data was based on traditional telemetry: it had great information on what cities were most viewed, how many viewers were in "street" mode versus "photograph" mode, and even information about viewers' displays. They instrumented search and navigation, and they collected counts for the number of times that users looked at certain sentinel regions. And because Virtual Earth was built on top of a set of progressively higher-resolution image tiles, the team was also archiving server logs that tracked how often individual tiles were downloaded.

Interviews with team members suggested that they did not have an intuitive notion of how their tool was being used. In conversation, one team member argued that people were likely to look at their own homes; another thought that the overhead photography would mostly be used over mountains. The goals were varied: they included seeing whether the user experience was well balanced across user needs and deciding how and where to invest in future rounds of photography.

We addressed these questions with a visualization tool called Hotmap. Figure 1-1 shows a screen capture from the visualization tool, focusing on the central United States. Hotmap uses a *heatmap* encoding of the tile access values. This is a visualization technique that uses a colormap to encode the access values at the geospatial locations of the tiles. Colored spots on the map are places where more users have accessed image tiles. The colormap is a logarithmic color scale, so bright spots have many more accesses than dim ones.

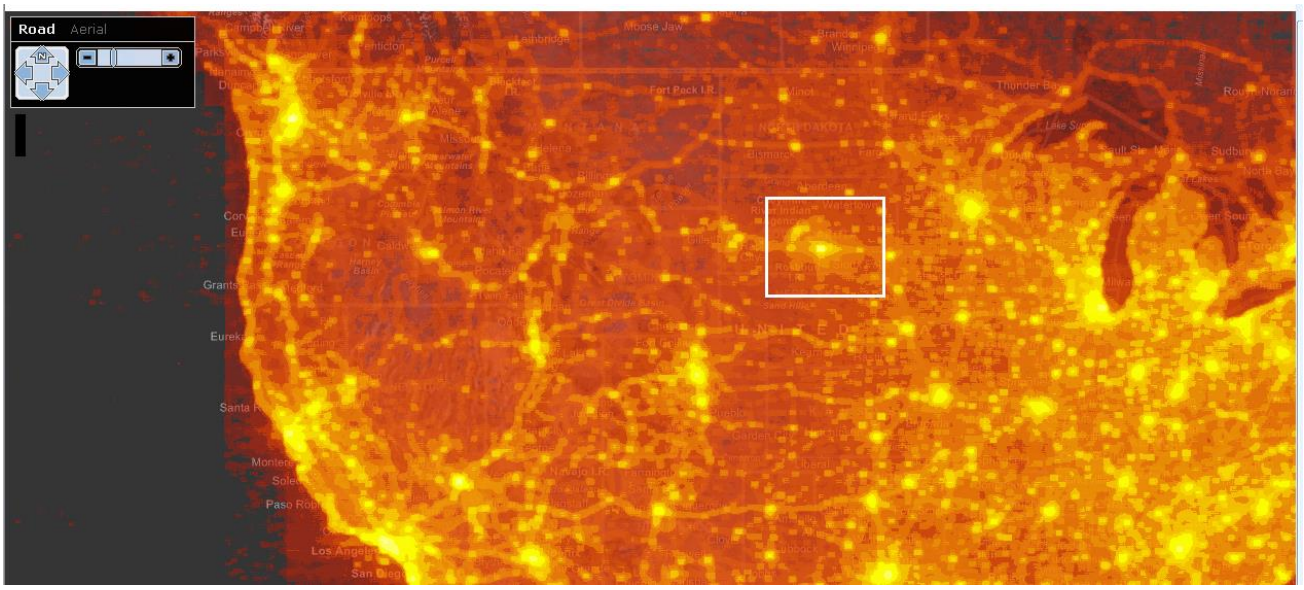

Figure 1-1. Hotmap, looking at the central United States. The white box surrounds an anomaly in South Dakota.

Some of the brightest areas correspond to major population centers—Chicago and Minneapolis on the right, Denver and Salt Lake City in the middle, and West Coast cities on the left. Near the center, though, is an anomalous shape: a bright spot where no big city exists. There is a star shape around the bright spot, and an arc of bright colors nearby. The spot is in a sparsely populated bit of South Dakota—there was no obvious reason to the team why users might zoom in there.

That point is, however, very close to the center of a map of the continental US. In fact, the team learned that the center of the star corresponds to the center of the default placement of the map in many browsers. The bright spot with the star most likely corresponds to users sliding around after inadvertently zooming in, trying to figure out where they've landed; the arc seems to correspond to variations in monitor proportions.

As a result of this usability challenge, many mapping tools—including Bing Maps (the successor product to Virtual Earth)—no longer offer a zoom slider, which keeps users from accidentally zooming all the way in on a single click.

A second screen capture, shown in Figure 1-2, reveals a bright spot off the coast of Ghana. This spot exhibits the same star pattern created by users scrolling around to try to figure out what part of the map they are viewing. This spot is likely only bright because it is at 0 degrees latitude, 0 degrees longitude, a point that GIS tools run into often. While computers might find (0,0) appealing, it is unlikely that there is much there for the typical Virtual Earth user to find interesting.[1]

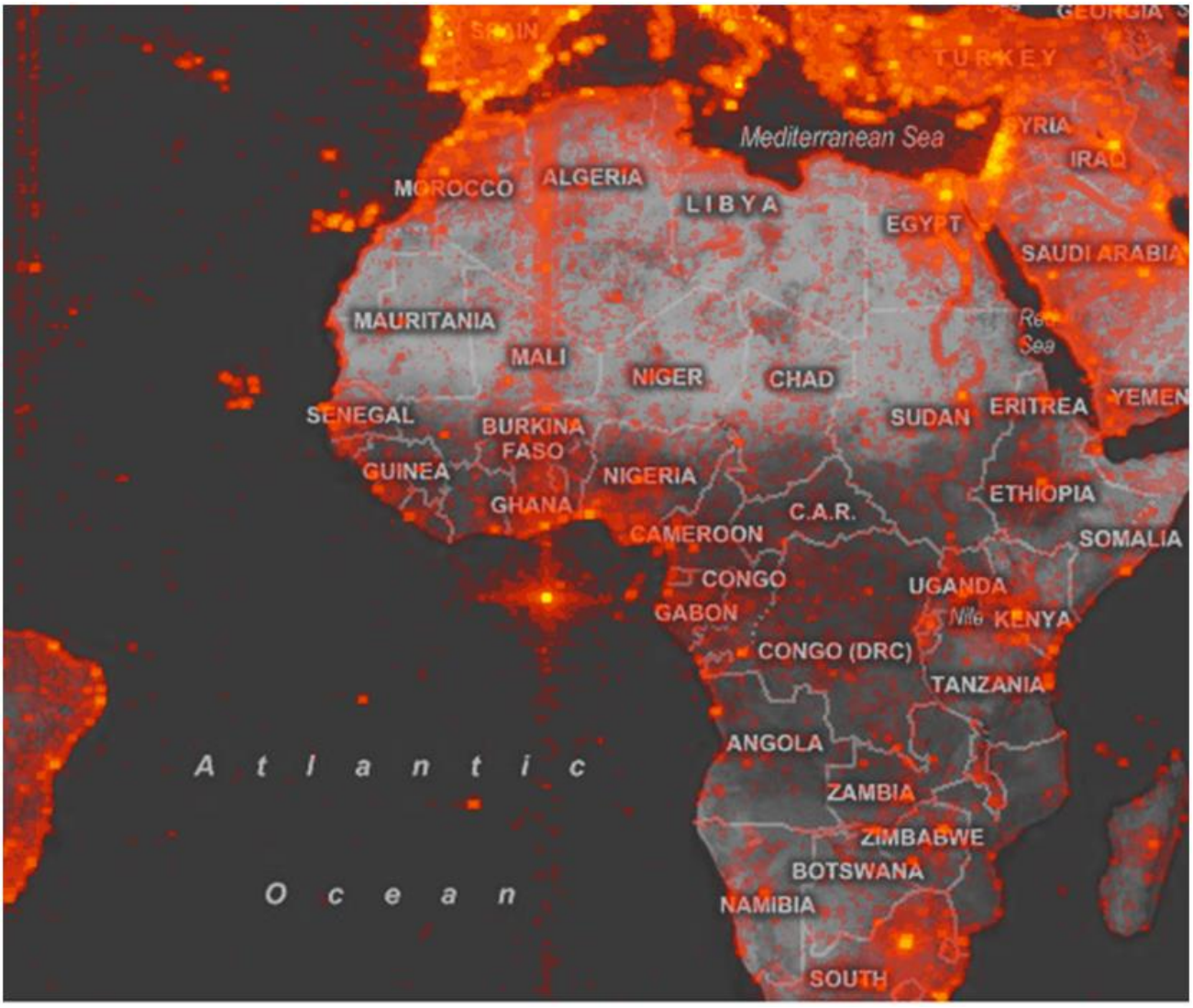

Figure 1-2. Hotmap, looking at the map origin (0,0).

This second bright spot inspired a hunt for bugs. The team rapidly learned that Virtual Earth's search facility would sometimes fail, and instead of returning an error message, typos and erroneous searches would sometimes redirect the user to (0,0). Interestingly, the bug had been on the backlog for some time because the team had decided that it was not likely to surface often. Seeing this image made it clear that some users really *were* being confused by the error, so the team prioritized the bug.

Although the Virtual Earth team started out using the Hotmap visualization expecting to find out about how users interacted with maps, they gleaned much more than just a characterization of usage patterns. Like many—dare we say most?—new visualizations, the most interesting insights were those that the viewers were not anticipating to find.[2]

Where Visualization Is Useful

Is visualization the silver bullet to help us make sense of data? Not always. There are two questions to consider to help you decide if your data analysis problem is a good candidate for a visualization solution.

First, *could the analysis tasks be supported with an algorithm*? A crisp task such as "I want to know the total number of users who looked at Seattle" suggests that an algorithm, statistical test, or even a table of numbers might be the best way to answer the question. On the other hand, "How do users explore the map?" is much fuzzier. Fuzzy tasks are great candidates for a visualization solution because they require you to look at the data from different angles and perspectives, and to be able to make decisions and inferences based on your own knowledge and understanding.

The second question to consider is "Is all the necessary information contained in the dataset?" If there is information about the problem that is not in the dataset which requires an expert to interpret the data that is there, then visualization is a great solution. Going back to our fuzzy question about exploring a map, we can imagine that it is unlikely that there will be an explicit attribute in the data that classifies a user's exploration style. Instead, answering this question requires someone to interpret other aspects of the data to bring knowledge to bear about what aspects of the data imply an exploration style. Again, visualization enables this sort of flexible and user-centric analysis.

For all but the crispest questions about explicitly measured phenomena, visualization is probably a good tool to throw at a problem. In our experience, we have almost never come up against a problem that cannot benefit from some amount of visualization.

Further Reading

The Hotmap project is discussed in:

- Fisher, Danyel. "Hotmap: Looking at Geographic Attention." *IEEE Transactions on Visualization and Computer Graphics* 13 (2007): 1184–1191.
- Fisher, Danyel. "The Impact of Hotmap." *The Infovis 2009 Discovery Exhibition.* Redmond, WA: Microsoft, 2009.

[1] So many datasets have references to (0,0) that GIS practitioners refer to that location as "null island."
[2] See "Further Reading" for other stories of how Hotmap has been used.

Chapter 3. Data Counseling, Exploration, and Prototyping

The previous chapter outlined a way to analyze a real-world question and transform it into an actionable, operationalized task. This analysis involves many steps that require decisions along the way: identifying specific tasks that address the broad question; decomposing each task into specific objects, measures, and groupings; and finally building visualizations that validate and support these tasks. Carrying out this process effectively requires sophisticated domain expertise, knowledge of the data and the problem space, and a sense of what would be a good answer to the question. This chapter discusses a variety of techniques that support gaining this understanding through working with stakeholders and iterating on visualization prototypes.

We call this collaborative process *data counseling*. We chose this name because working with stakeholders is a back-and-forth process of conducting interviews; of diving deeply into a user's intents around data; and of understanding the stories of where the data comes from, what problems are associated with it, and what it can mean.[1] Data counseling is interwoven with exploring data, developing visualization prototypes, and collecting feedback on these preliminary results. This chapter describes techniques for these steps as well.

A major visualization project can require multiple interviews and rounds of prototypes in an intensively collaborative process. Recognizing the ecosystem of stakeholders involved in a project can help uncover needs and increase the impact of the data analysis results. A smaller project might entail just one or two informal interviews and putting the data into a graphing program like Tableau.

Sometimes an analyst needs to make sense of data without a team around them. The strategies in this chapter still apply to prototyping and refining visualization solutions in such conditions. These techniques are applicable at all scales.

Technique 1: Data Counseling

Data counseling is a technique that brings domain expertise into the operationalization process to help inform decisions about good proxies as well as to uncover insights using the resulting visualizations. This expertise is uncovered through interviews with a variety of stakeholders in a project. The goal of these interviews is to gain an understanding of the questions and data, as well as to get feedback on proxies, explorations, and visualization designs.

Arguably, the hardest part of data counseling is figuring out *who* the stakeholders are and *what* questions to ask them. The rest of this section describes some of the types of stakeholders that can be encountered during this process and provides guidance for conducting interviews.

Identifying Stakeholders

When it comes to tackling a problem, who is invested in the results? Who will use the results, and who will they present those results to? If the visualization produces valuable insights who will act on those insights, and what will they do with them? There is likely a whole ecosystem of people that have been, are, or would like to be involved with the data and the problem—the people who produce and store the data, the people who want to consume it, and those who will make decisions based on it.

All the people who are invested in the problem in some way, shape, or form are the *stakeholders*. Identifying these stakeholders is crucial for data counseling. Different stakeholders can give different perspectives on the data and the problem, and potentially provide unanticipated paths to insight. The process of interviewing and examining the data itself may uncover new stakeholders who can provide fresh perspectives.

There are a number of recurring types of stakeholders. This list is by no means complete, but it can work as guidance for identifying some of the important people in the ecosystem of a problem. A single person could embody one, some, or all of these roles:

Analyst

> A person who works directly with the data, searching and exploring to make discoveries. These stakeholders are the people most likely to use visualizations designed for the problem.

Data producer

> A person who collects, creates, and curates the data. Data producers can often shed light on the nuances and quirks of how the data was attained and can be invaluable during the data cleaning process.

Gatekeeper

> A person with the power to approve or block the project, including authorizing people to spend time talking about the data and problem. The gatekeeper's perspective can be useful for understanding the high-level goals and potential impact of the project. In some settings, a gatekeeper may require a proposal to carry out an analysis.

Decision maker

> A person who wants to use the insights gleaned from the data to execute on a decision. Decision makers are often one step removed from analysts, and act as the analysts' customers. They often have a different interpretation of goals and questions than those who are closer to the data.

Connector

> A person who may not be directly involved with the data or the question but can identify other people to talk with. Connectors can help fit together diverse perspectives on a problem and figure out what analysis needs to happen. In our experience, good connectors are worth their weight in gold.

Conducting Interviews

The operationalization process proceeds through information gleaned from interviews; later rounds of interviews provide feedback on intermediate results and final designs. The role of the interviewer is to ask questions that will guide the stakeholders toward elucidating the information necessary for working through an operationalization and designing visualizations.

Interviewing can be very challenging, but it can also one of the best parts of the work—how many jobs allow you, even *require* you, to talk to experts about the deepest, most interesting parts of their problems? Interviewing is not easy, though, and requires practice and experience. The necessary skills include how to keep a conversation moving along and on track, how to elicit meaningful responses, how to revise questions

based on responses, and how to interpret both subtle cues and detailed responses. While this section provides several strategies to help with these tasks, gaining competency in these skills is a matter of practice.

In conducting an interview, there is a sweet spot with regard to the amount of structure in the conversation. Unstructured interviews, on the one hand, resemble casual conversations—the interviewer goes in with little expectation of where the conversation will go and does little to guide it in any one direction. This style of interview can uncover unknown needs and goals, but it can take a significant amount of time to get to anything useful. On the other hand, formally structured interviews are like giving a stakeholder a verbal survey, where the interview is completely scripted and strictly guided. While efficient, this type of interview leaves little room for discovering new insights.

The most effective data counseling sessions aim for a spot in the middle: *semistructured interviews*. The interviewer does some preparation in developing initial questions. The rest of the interview is then open to exploring ideas that come up during the conversation. Be prepared, but also be open.

The initial set of questions for an interview should be open-ended and address the problem, data, and context in order to help understand where the interviewee sits: their perspective on the problem, how they see the scope of the problem, and how they expect to interact with it.

Some useful interview questions might include:

- What are the goals of the project? How do those goals fit with the organizational needs?
- Who would act on the insights and results of this analysis? What decisions are they looking to make?
- What questions can be answered with this data?
- What do you already know from the data, and what else do you expect to find?
- What do you want to do with the data that is not currently possible? If you could do that, what else would you want to do next?

These general questions are meant to get a conversation going and to help in establishing the start of the analysis process. They lead to more specific questions that help clarify understanding of the problem and the data, confront assumptions on the part of the stakeholder and the analyst, and shape the description of the problem into something that a visualization can solve.

Interviews often start in the wrong place

It is easy to begin this analysis in the wrong place. When people come to us with visualization questions, they often start with very specific questions: “How do I tie together two scatterplots with a gradient color pattern?” These types of questions tend to be the result of people struggling to force their data into the visualizations that they know best and finding that those either don’t fully support the extent of the data they have or don’t really support an insightful analysis.

The conversation searches out the more abstract question and often finds that the question the person really wants to solve has a very different visualization solution.

Librarians know this challenge well. When someone asks for, say, an issue of a news magazine, librarians are trained to gently probe for the underlying information need. What does the reader really want to know?

Sometimes the question might be better solved with an entirely different source: if the reader who wants the latest *Time* magazine is hoping it will contain a map of Somalia, an atlas would fulfill their need better.

What would it look like in the data?

The process of data counseling often entails chasing down particular meanings of unclear words and identifying good proxies. The question "What would this look like in the data?" can lead to illuminating results. For example, if a journalist were trying to find "good movies" in the database, in the interview we might ask questions like "What would the data show for *good* movies?" and "What would *bad* movies look like?" This can help interviewees nail down specifics.

Fairly often, the interviewee will not be sure what a "good movie" would look like. The process of articulating a list of possibilities, as outlined in Chapter 2, can be highly informative in itself.

Making questions specific

Low-level questions arise when trying to make general tasks more concrete and actionable, such as defining what a specific dimension means or how the objects that appear in a task actually look in the data. Finding these poorly defined terms in the interview is a cue to ask more questions to clarify those concepts more concretely. It can be useful to ask stakeholders what these terms mean within their workflow or to show an aspect of the data by pulling it up in a spreadsheet.

Certain action terms are also useful cues during these interviews: the verbs a stakeholder uses when discussing data can help inform the visualization. For example, interviewees might talk about *comparing* data items in describing a task. This invites a follow-up question: "Would you like to compare one item to another, or group many items together?" Similarly, words such as *select*, *identify*, or *group* can translate directly into tasks that can be supported with a visualization.

Other words, like *shape*, *structure*, and *size*, can help in deciding what kinds of visual encodings to use or what characteristics of the data the stakeholder is most interested in seeing. The visualization types described in Chapters 5 and 6 will help you recognize and know what to do with visualization keywords like these during interviews.

Breaking out of dead ends

Interviews can reach a point that feels like a dead end: the stakeholder has answered the planned question but the problem still seems inscrutable. There are a couple of strategies that can be used here:

- Try to rephrase the stakeholder's response back to them. This strategy allows the stakeholder to correct any misinterpretations and it also can prompt the stakeholder to explain their ideas in a more familiar terms.
- Try to ask the same or similar questions in different ways. Often a specific phrase or choice of words will click with the stakeholder and cause them to respond in a way that makes more sense.
- Try exploring a different conversational topic. It's not unusual that another topic will illuminate this one.

One of the most important things you can do in interviews is to keep the stakeholders talking. The more they talk, the more likely it is that they will share a response that clarifies a topic or opens up a new avenue of inquiry.

The interviewer's toolbox

There are several tools that are a part of the interview toolkit. Commonly used interviewing tools include pen and paper, voice recorder, camera, and video recorder. We advocate for voice recording of interviews, in large part because it is difficult to take detailed notes while also trying to think of follow-up questions based on what is being said. We try to transcribe an interview shortly after it is conducted to ensure the context is fresh in our minds. We rarely transcribe an interview word for word, but instead transcribe the most important or complex details. Transcribing is useful for analyzing the interview results, as well as for making it easy to refer back to the conversation later in the design process.

Conducting Contextual Interviews

In general, a first interview does not get into detailed analysis; it can be useful to get a general overview of the problem, identify stakeholders, and establish the stakeholders' expertise. It is in follow-up interviews where details begin to emerge.

There is often a big difference between talking to someone in a conference room as opposed to sitting at their desk. In a conference room, people will often tell a very general story; at their desks, they will more likely show their processes in very specific ways. For follow-up interviews, the *contextual interview* is a particularly useful tool. Contextual interviews take place in the stakeholder's work environment and consist of demonstrations of the tools and data inspection methods that the stakeholder currently uses. These types of interviews can bring to light aspects of a problem that might not have come up in a strictly verbal interview. They help show how the data works in practice: what happens with current capabilities and how users handle and understand the data they see.

A contextual interview often starts by asking the stakeholder to either walk through a specific analysis task they have already performed or conduct some of their work for that day with the interviewer present. The stakeholder talks through each step they are taking; the interviewer can interrupt with clarifying questions or use these as launching points for further explorations. The following starting questions can help the interviewer understand what works, what does not work, and what does not yet exist:

- What is the work process you currently use? What tools are involved?
- What challenges do you have in analyzing the data?
- Are there limitations within the system? If so, how do you work around them?
- Do you understand what the system is doing to the data and any algorithms that are being applied, or is this a black box?

Technique 2: Exploring the Data

While talking with stakeholders can be very informative, there is no substitute for reaching deep into the data. We like to start exploring the data as early as possible, for a few reasons. First, it is useful to understand how it is structured and what data is available. Second, each operationalization needs to be checked against the data, and third, it helps to start addressing fine-grained tasks.

In Chapter 2, for example, exploring the data helped us determine which fields were available in trying to evaluate what would make for a good director. It also helped us choose appropriate cutoffs in trying to define terms like *highly rated movie*.

Working through the analysis, then, brings forward a variety of visualization tools to explore the data. Tools like Excel and Tableau, or R and Pandas, make it easy to rapidly generate visualizations that can highlight distributions of data, its major dimensions, and the values within. These tools also make it easy to check whether a dataset makes sense—for example, to confirm that a hierarchy really is layered appropriately or to ensure that there are only a small number of categories for a specific dimension of the data.

Sometimes it becomes clear that the problem is so specialized that it needs a bespoke visualization tool created from scratch. This happens when the data to be explored is not amenable to an off-the-shelf tool. Both Hotmap, described in the Preface, and the case study in Chapter 8 are examples of situations in which it was necessary to explore complex data with novel visuals. These bespoke tools can be created using visualization-specific languages like Vega, D3, or Processing.

These can be fast-and-loose data prototypes: the goal is to get ideas up and going as quickly as possible, as opposed to carefully considering software architecture for long-term use and reusability. Rapidly discarding bad prototypes is as much a critical part of visualization as it is of other design areas.

Making the decision to create a custom data exploration tool requires weighing the development time against the significance of the analysis—if it is possible to get 80% of the way to a good decision using Excel, then it may not be worth spending three months to develop a custom solution. Bespoke exploration visualization tools instead come into play later, when an operationalization is well established and verified, and the focus is on going back and answering the high-level question.

If the high-level goals of a project can often be met with off-the-shelf tools then it is great to be considered a hero for quickly resolving the problem! Custom tools, in contrast, can be great ways to better understand what is lacking in the current ecosystem of visualizations. These insights are invaluable when designing a custom visualization tool.

One challenge is knowing when, and how, to start digging into the data. Oftentimes the stakeholders already have some way of analyzing or visualizing the data that they find to be insufficient for their question. This is usually good place to start. For example, are they looking at many static visualizations? Add interactivity to support exploration. Are they using only one kind of visualization? Take a different perspective on the data and visualize it in a different way using another type of visualization. Use these early data explorations for a deeper conversation about what works and what doesn't, and why. This process also provides a chance to better understand the stakeholders' perspectives on the data. Chapter 8 discusses a case study where scientists had a pre-existing set of technologies; adding interactivity and new representations helped reveal that there were entirely new questions to ask, too.

Technique 3: Rapid Prototyping for Design

Even from fairly early stages in the process, prototypes of the final tool can be a helpful model. The intention behind prototypes is to explore the visualization *design space*, as opposed to the *data space*. A typical project usually entails a series of prototypes; each is a tool to gather feedback from stakeholders and help explore different ways to most effectively support the higher-level questions that they have. The repeated feedback also helps validate the operationalization along the way.

Rapid prototyping is a process of trying out many visualization ideas as quickly as possible and getting feedback from stakeholders on their efficacy. Throughout the data counseling process, multiple rounds of

rapid prototyping can help in understanding how the problem is formulated. In early phases, sketches on a whiteboard can help to better understand what types of visualizations to use and how stakeholders might interact with them. Later on, higher-fidelity techniques can explore the space of possible visualization designs. The design concept of "failing fast" informs this: by exploring many different possible visual representations, it quickly becomes clear which tasks are supported by which techniques.

The Range of Prototypes

The term *prototypes* refers to a broad range of techniques and tools, from paper to programming. The fidelity of these prototypes, as well as the time and energy required to create them, lives on a spectrum (Figure 3-1).

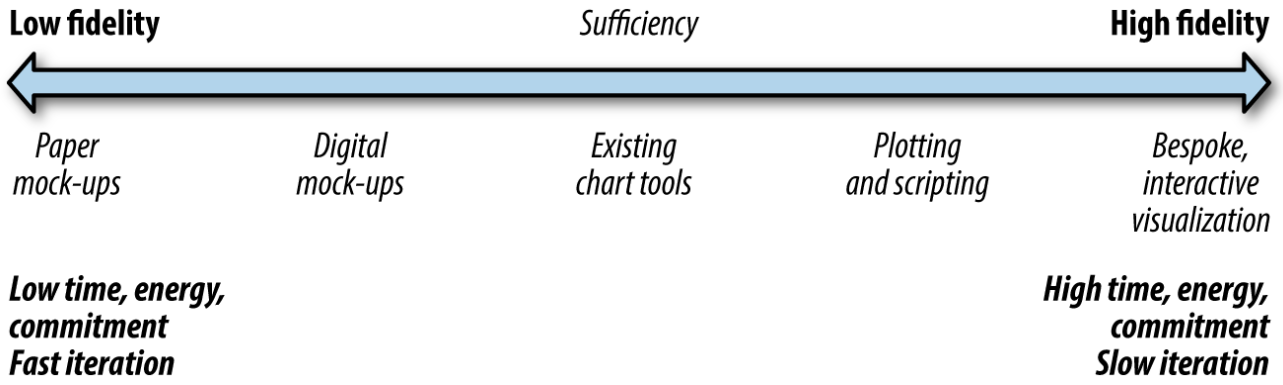

Figure 3-1. Prototypes range from low-fidelity sketches to high-fidelity working models.

One end of the spectrum is characterized by *low-fidelity* (or *lo-fi*) prototypes. These include mock-ups quickly sketched on paper or a whiteboard with impressions of what the data might look like, and fast, digital mock-ups that may include some controls for explaining interaction ideas, such as slide jumps in PowerPoint or Keynote. Figure 3-2 is an example of a quick interface mock-up made during the design process. Lo-fi digital mock-ups can also incorporate charts generated in a tool like Excel or Tableau with fake or sampled data to explore possible visualization representation ideas. These lo-fi prototypes are great for communicating the gist of an idea in an interview, or for recording high-level ideas when planning out how to explore the data. Lo-fi prototypes are, by nature, fast and easy to produce.

Lo-fi sketches play a critical role in interviews. Creating these prototypes can help us to understand what the implications of the data might be, and clarify the usefulness of different proxies. If a diagram is confusing to explain and design on a whiteboard, it may require too much detail to fit on a screen.

Communicating ideas with lo-fi prototypes can rapidly help establish whether the visualization designer is on the same page as the stakeholders. Drawing pictures of possible interfaces can start new conversations about the problem and its constraints. Figure 3-2 shows one instance: a stakeholder was discussing relational data, and drawing this data on the whiteboard allowed the stakeholder to see what it might feel like to visualize the data as a network. The multiple colored lines allowed the stakeholder to start thinking about how to view multiple modalities of the data; the directed edges were actually built from a sample of the data. Drawing this prototype helped the client realize that there was more structure to the data then they had been communicating: every node in the graph represented by a box actually occurred at a specific time, and it was important in the analysis to expose the temporal dimension of the data.

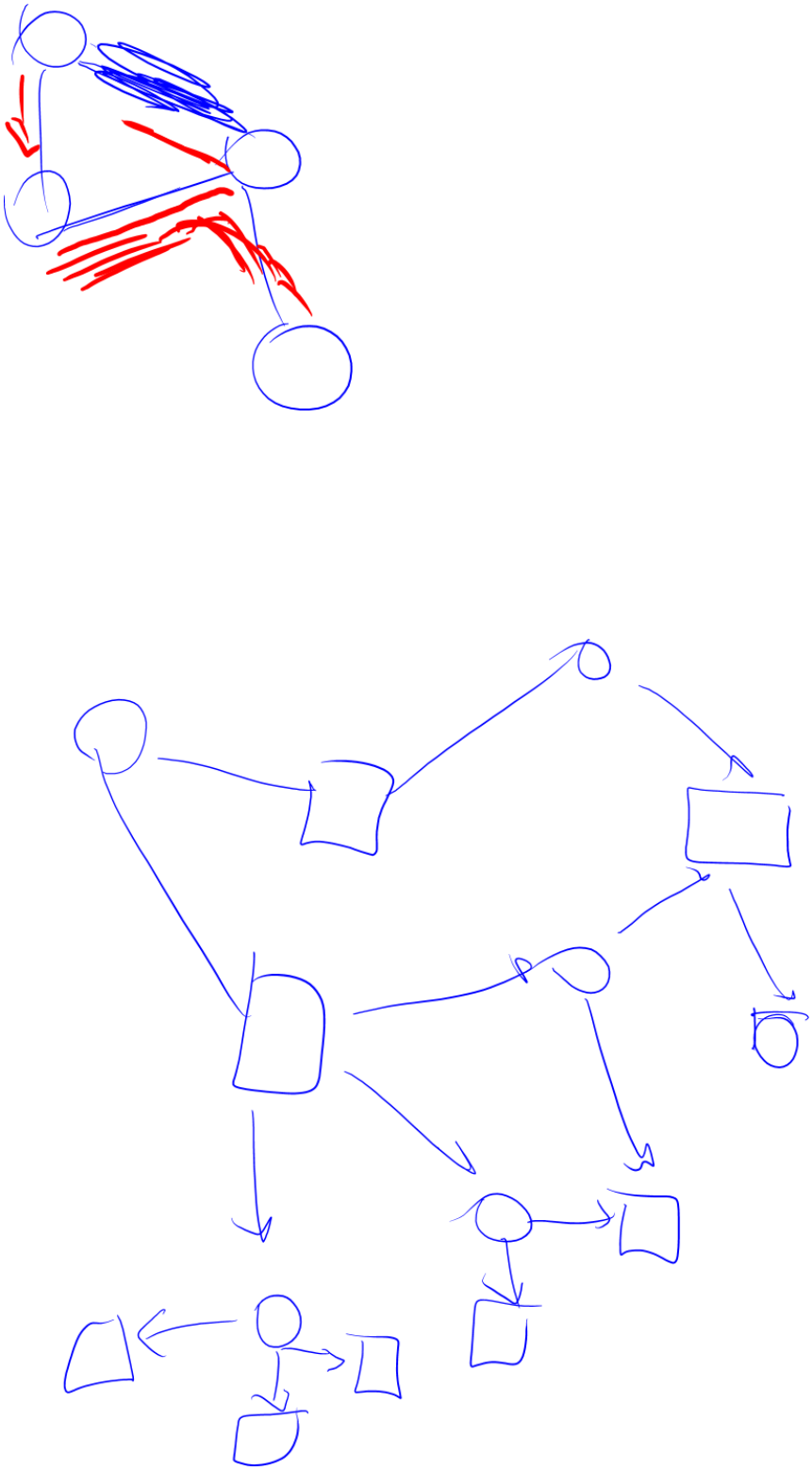

Figure 3-2. A lo-fi prototype exploring the idea of a weighted, directed graph layout. This sketch was hand-drawn on a whiteboard during an interview session, based on sample data, by manually looking at the spreadsheet and drawing out the relationships.

Lo-fi *slideware* can help ensure that designs will make sense to users, especially when incorporating interactive features into the design. The slideware in Figure 3-3 shows one step in the feedback cycle, illustrating the result of a specific interaction mechanism. This image was manually assembled in a variety of different tools. The prototype sketch is meant to help the user understand how the final interaction will work.

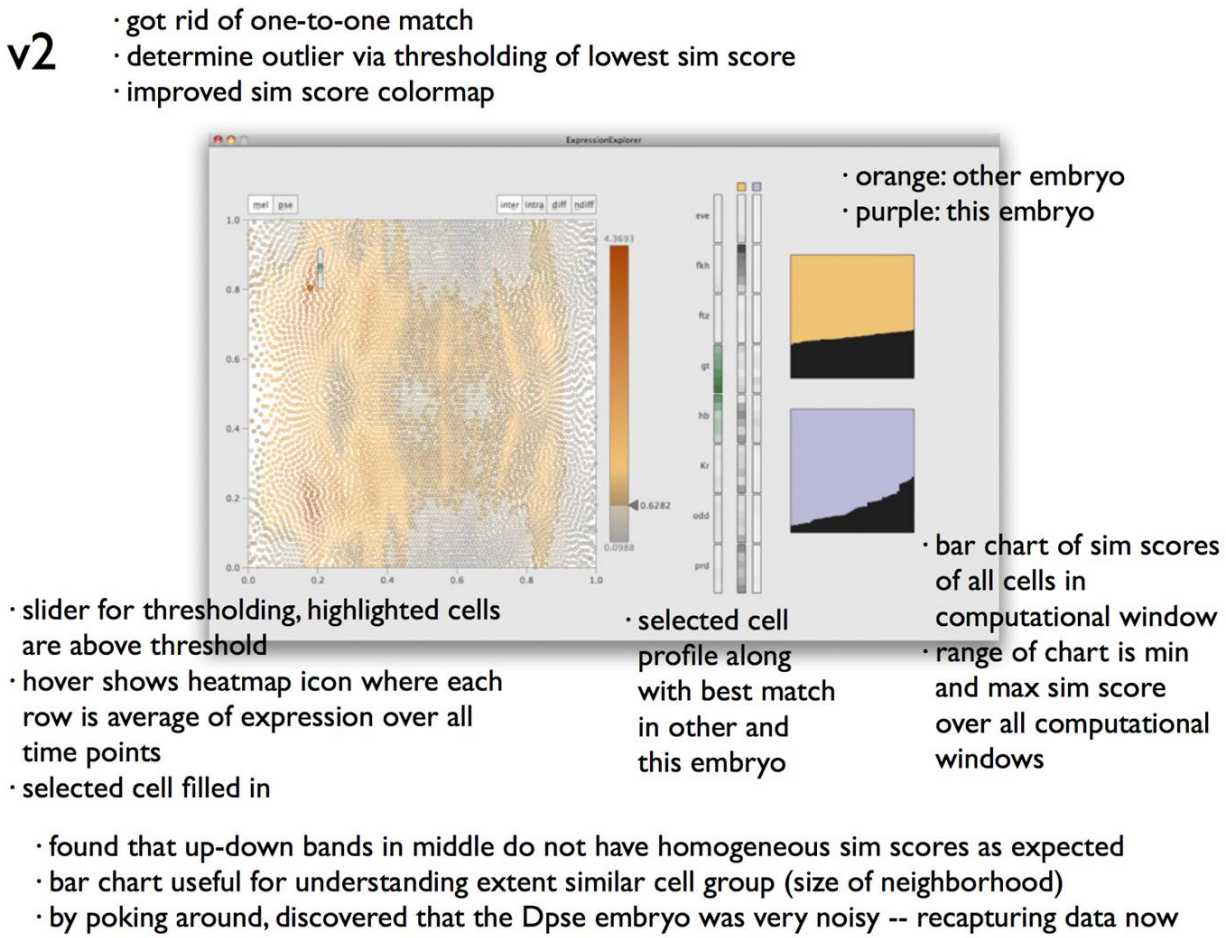

Figure 3-3. This slideware image of a design stage shows iteration from a previous version. The images were created in a variety of different tools.

On the other end of the prototype spectrum are *high-fidelity* (or *hi-fi*) custom visualizations which must be created from scratch. These hi-fi prototypes are meant to largely contain the core functionality of an envisioned visualization tool, including all necessary visualizations of the data and interaction mechanisms. They will often, however, gloss over many backend issues such as smooth integration with existing workflows or fully fleshed out features for saving and loading. Just as for bespoke visualizations created for our own data exploration, languages like D3 or Processing can help in creating hi-fi prototypes rapidly.

Hi-fi prototypes are meant to be thrown away. In our experience, however, hi-fi prototypes are often the tools that get deployed and adopted by some users, particularly to those frantic to get into their data. Regardless, the point is not to worry about the code other than to confirm that ideas can work.

Using a Data Management Pipeline

Using a data analytics engine to handle data management can allow an analyst to rapidly iterate through different ways of looking at the data. Many off-the-shelf data visualization systems provide both a data pipeline, which consists of database connectors and data cleaning and shaping facilities, and a visualization system. Until recently, however, doing this prototyping meant that analysts were constrained to the visual mappings available in the system.

A few tools have begun to allow developers to incorporate their own custom visuals: Google Sheets, Microsoft Excel, and Microsoft PowerBI all have custom add-in mechanisms. PowerBI also provides dashboard tools like cross-filtering between custom add-ins.

Eliciting Feedback

Identifying stakeholders during data counseling is useful not only to help with the operationalization, but also to garner feedback on prototypes. With rapid prototyping as a strategy, we go back to our stakeholders early and often with our visualizations to ensure that the operationalization has directed an effective visualization.

Eliciting useful feedback, however, goes beyond asking stakeholders if they like what they see—approval is necessary, but not sufficient. Part of the problem with seeking approval is that interviewees (sometimes unconsciously) wish to give positive feedback to an interviewer; this can be particularly problematic within a friendly team. A stakeholder might say they like a visualization that is not informative but looks nice.

We like to focus instead on what the visualization can and cannot do. A contextual interview where the stakeholder uses the visualization can be particularly insightful for uncovering weaknesses in the design or problems with the operationalization. Keeping questions focused on aspects of the data that are being shown forces the stakeholder to more directly confirm or refute the efficacy of what they are seeing.

And, Repeat

It is very difficult to get a good (or even adequate) operationalization of a problem the first time around. Getting this right often requires multiple interviews with stakeholders, interspersed with some data exploration and rapid prototyping.

For example, consider an operationalization that leads to a distribution of values in a histogram. That distribution helps show that there are outliers at one end of the range that had not been in the original problem description; stakeholders may then realize the outliers are actually quite interesting, which leads to a new task and a new representation.

The process is often a very iterative one. Talk with some stakeholders, try some ideas with the data, share those ideas with the stakeholders. And, repeat.

Conclusion

This chapter looked at several core techniques for supporting operationalization: data counseling, data exploration, and rapid prototyping. These techniques bring a variety of different perspectives on the problem and the data in order to build, refine, and support an operationalization of a problem. All of these techniques are useful on their own, but using them in combination provides a powerful suite of tools.

Chapter 4 looks at the nature of the data itself. Understanding the types of data, and the tasks that can be carried out with it, leads to Chapter 5 and a look at the core visualizations for basic data types.

[1] This process is closely related to *task analysis* in interface design. The distinction is that task analysis is typically oriented towards creating interfaces; this process, instead, works with data, which warrants a unique set of considerations on the part of the designer.

Chapter 6. Multiple and Coordinated Views

The previous chapter provided a gallery of single-chart visualizations. This chapter brings those views together into interactive, connected visuals called *multiple linked views* (MLVs). An MLV leverages multiple visualizations by linking the information shown in each view to the others through user interactions.

MLVs are vital to understanding large and complex data. They allow many different attributes to be viewed at once by splitting them up across a set of views and partitioning the data items to find interesting trends. They can be designed to help guide a user toward the most interesting data items, to show multiple perspectives on data, or to allow the user to dive more deeply into a dataset. In an MLV system, a dataset is shown in multiple simple visualizations, with the data items shown in the different charts corresponding to each other. The charts in each visualization can be used to highlight, control, or filter the data items shown in the others.

There are a number of well-defined MLV design patterns, each of which supports a different set of analysis tasks. This chapter covers five of the best-known patterns: small multiples, scatterplot matrices (SPLOMs), overview+detail, multiform views and dashboards, and overlays. Small multiples and SPLOMs are series of small visualizations that use the same view but show different parts of the data. Overview+detail pairs two views, one as an overview of the complete dataset and the other as a detailed view of a subset of the data. Multiform views and dashboards use different types of visualizations with each view optimized for a subset of attributes. Lastly, overlays are multiple visualizations drawn on a common, shared axis.

This chapter discusses appropriate tasks and provides examples for each of these. There are characteristic interactions that go with most of these design pattern. The chapter describes the ways that users might interact with these visualizations. Table 6-1 provides an overview.

MLV type	Supported task	Data	Interaction	What is shared
Small multiples	Understand and identify differences between subsets or measures of the data	Each view shows a partitioned subset, or a different measure, of the data	Usually static	Different data, same attributes, same view; or, same data, different attributes, same view
SPLOM	Understand relationships and correlation between the attributes	Each scatterplot shows all of the data items for every pair of attributes	Brushing and linking highlights the same data items in different views	Same data, different attributes, same view
Multiform views and dashboards	Understand relationships and correlation between the attributes	Each view shows all of the data items, but different attributes	Brushing and linking highlights the same data items in different views	Same data, different attributes, different views
Overview+detail	Find interesting data items and understand those in detail	Large datasets where all data items and attributes cannot easily be shown at once	Selection in overview; navigation in detail	Same data, different attributes, different views

MLV type	Supported task	Data	Interaction	What is shared
Overlay	Compare two datasets that share a common attribute	Different, but joinable, datasets	Usually static	Different data, shared attribute, shared axis

Table 6-1. The design space of multiple linked views

Small Multiples

The small multiples design pattern—sometimes also called a *trellis chart*—focuses on showing subsets of a dataset, meaningfully partitioned. The pattern allows an analyst to quickly look across these subsets and compare them in order to find trends, patterns, and outliers.

This pattern is common in everyday interfaces. In online shopping sites, a search query is presented as a grid. In this grid, the results are partitioned based on specific products, and each individual view in the grid gives quick information about what the product looks like, its price, and its rating. Similarly, weather forecasts typically show forecasts for several days with small multiples. Here, the forecast data is partitioned by day; each individual view shows information about the day's temperatures, cloud cover, and precipitation. The layout supports quick skims down the views to get a sense of how the weather will change over the course of the days.

The views in a small multiple must maintain *consistency*, so it is easy to read down or across each individual view to directly compare the data subsets across each attribute. Maintaining the same view of the data while varying the data items is a hallmark of small multiples.

A small-multiples display shows the same visualization repeated across a row, column, or grid of views. Small multiples come in two variants: they can be split by dimension, or they can show differing measures. When a visualization has been split by dimension, each individual view is a visualization of the same attributes, but the views show different subsets of the data, split along a partitioning attribute. The partitioning attribute is typically an ordinal or categorical value—as in Figure 6-1—a binned continuous value.

When a visualization shows multiple measures, each individual view shows most of the same attributes for all the data. Each individual view varies one dimension or measure from its neighbors. In Figure 6-9, the three views all show the same dimensions, but vary on the measure: population, engineers, and hurricanes per state.

Small multiples are particularly good for supporting *comparison* of subsets of the data *across* several attributes of interest—words such as these in an operationalized task point to a small multiples design pattern. For example, the small multiples of choropleth maps in Figure 6-1 support comparison of the percentage of the population in states in the US across different salary ranges.

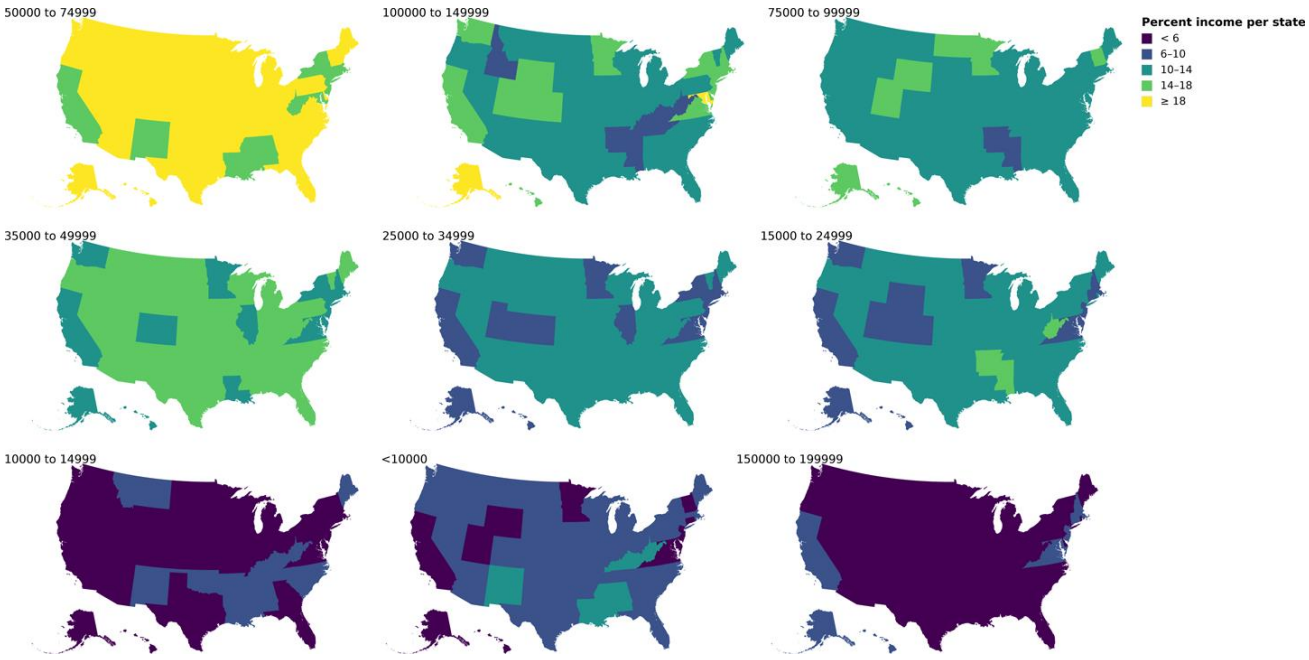

Figure 6-1. Small multiples of choropleths. In each choropleth, the percentages of the states' populations for a specific salary range are shown—the small multiple views are partitioned over the set of salary ranges.

In Figure 6-1, percentage and geographic location are shown in each view, and the data is partitioned by salary range. Each individual view in the small multiples display supports statements like "The state of Alaska has a comparatively lower percentage of residents in the lowest salary range." The full display supports statements like "California and Virginia stand out for their unusual distribution of salary at both the lowest and highest ranges."

One virtue of a small multiples view that is split by dimension is that all the views share the same spatial placement and the same color scale. The shared color scale is useful here because the multiple visualizations have the same meaning. The reader learns to interpret that *yellow* means a high percentage while *purple* is low, and can then effectively look across the charts to compare a specific color, pattern, or spatial location.

Often it is not obvious from a task which attributes should partition versus define the views, and trying different combinations can be useful. It is often fruitful to explore different small multiples during the early EDA stages to not only help in understanding the data, but also enable further refinement of the operationalization.

CONDITIONING AND GENERATIVE GRAMMARS

The concept of *conditioning* on a variable cuts across many visualization types. This is the statistical term for partitioning one attribute by another. Any single-view visualization can be changed into a small multiples view by conditioning on one or two dimensions; many visualizations can be overlaid by choosing another color to represent conditioning on a second dimension.

A bar chart is a small multiple of single bars, partitioned on an attribute. A clustered bar chart is a hierarchy of small multiples within a small multiples display. This sort of logic drives *The Grammar of Graphics*, which reduces every point on a visualization to a mark drawn in a particular way as described by the data; tools like *ggplot* and Vega explore this philosophy further. See "Further Reading".

Scatterplot Matrices

Scatterplot matrices (SPLOMs) are related to small multiples in that they use the same visualization—a scatterplot—across a matrix layout. Instead of showing subsets of data items across a few choice attributes, as in a small multiples display, they instead show the *complete* dataset in a matrix of scatterplots. More specifically, a SPLOM pairs each (usually) continuous numeric attribute against every other attribute in a (diagonally symmetric) matrix layout of scatterplots (Figure 6-2). As with other scatterplots, as discussed in Chapter 5, items in the scatterplot may also be colored and sized by additional attributes.

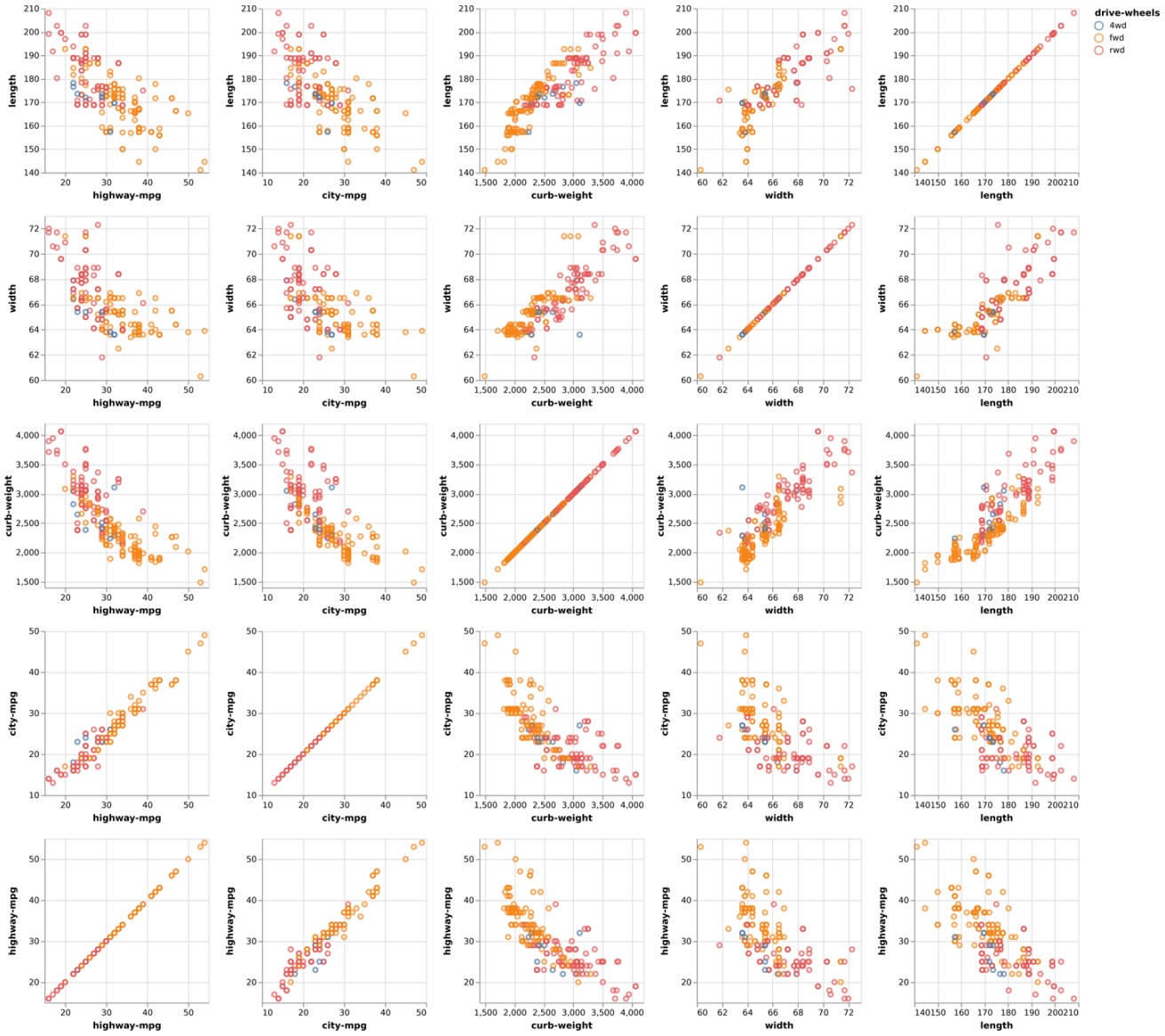

Figure 6-2. A SPLOM comparing attributes of cars in the scatterplots, with a color encoding indicating whether the cars are all-, rear-, or front-wheel drive. This chart helps show, for example, that rear- and front-wheel-drive cars can be separated by their curb weight in conjunction with city-mpg or highway-mpg more than by their width and length.

A SPLOM is primarily useful for characterizing relations between attributes in the earliest stages of EDA. Finding these relationships can help in narrowing down which attributes may be of most interest for further study during later EDA stages.

Overview+Detail

The overview+detail design pattern is essential for navigating and exploring large datasets in order to find interesting data items. It applies to any dataset that is too big—in terms of the number of data items, the number of attributes, or both—to show all at once. This design pattern includes an *overview* visualization that helps in finding interesting subsets of the data, and a linked *detail view* that shows the low-level attribute values associated with the selected subsets. Words in a task such as *locate* and *find* may indicate an overview+detail design pattern.

This design pattern is common in email clients, which provide an overview of the inbox showing the sender, subject, and date for all emails. Selection of an email in this overview then triggers a detailed email view to show the complete contents of the message. This is a flexible, user-driven process that supports a number of tasks: reading new emails, finding emails sent last night, or going back to a specific topic from yesterday, to name a few. The overview pane gives just enough information to make the selection, which is then shown in its entirety in the detail view. This detail view updates with each new selection.

Overview+detail is also frequently used as a navigation aid to provide context for movement around virtual spaces, such as maps, video games, or images. For example, Figure 6-3 is a screenshot from a photo viewing application. The overview is a small, contextual view that supports panning and zooming; the detail view shows a zoomed, detailed portion of the photo.

Figure 6-3. Some photo-viewing apps support overview+detail for panning and zooming an image. Here, the overview in the bottom-right corner indicates which part of the image is being shown in the larger detail view.

The overview+detail design pattern supports guided navigation to help find interesting subsets of the data. This pattern looks across many data items, using either one or several attributes as a measure of *interestingness*. The overview may use an attribute that is shown in other views, or a new, summarizing metric created from the underlying data. A selection in the overview triggers an update in the detail view to show the underlying details about the selected subset. This is typically a one-way interaction where selection in the overview drives what is shown in the detail view, but not vice versa.

What makes for *interestingness*? It is any attribute that helps figure out what subsets are worth looking at in the dataset. If the analyst is looking for places where data has extreme values, for example, it might be the count of items in that area, the maximum value of an attribute, or the average. It might be a synthetic value, such as an anomaly score. The case study in Chapter 8 has an overview that uses a distance function to help guide users to the most interesting bits of detail.

Figure 6-4 shows a dataset of genes located within a chromosome.[1] There are thousands of genes, far too many to reasonably show in a single view. Instead, the overview on the left shows regions of interest, each of which contains a set of related genes, shown as colored bars along the chromosome. These bars can be selected, triggering the detail view on the right. This detail view shows the individual genes within the selected region.

Overviews and details can nest for larger and more complex datasets. Figure 6-4 is actually part of a larger system that has two levels of overview to support looking across complete genomes—this system is shown in Figure 6-5. This visualization tool, called MizBee, supports selection of chromosomes of interest in the left view, regions of interest in the middle view, and detailed analysis of genes in the right view. A video of this system in action can be found on the book's website.

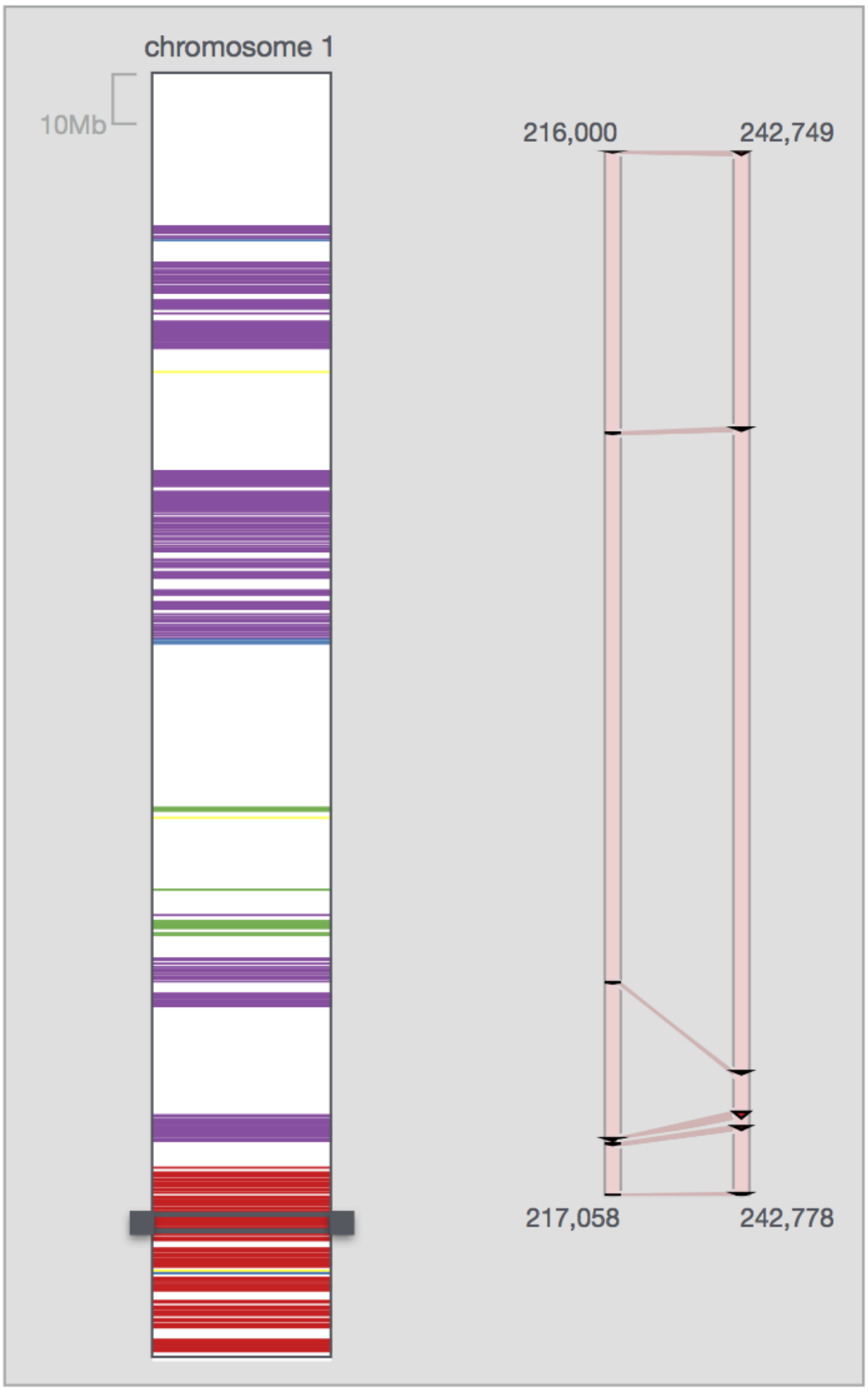

Figure 6-4. Overview+detail views in a tool for exploring genetic data. The overview on the left shows regions of interest, color-coded based on a similarity function. Selecting a region of interest triggers the detail view on the right to show the location of individual genes within the selected region.

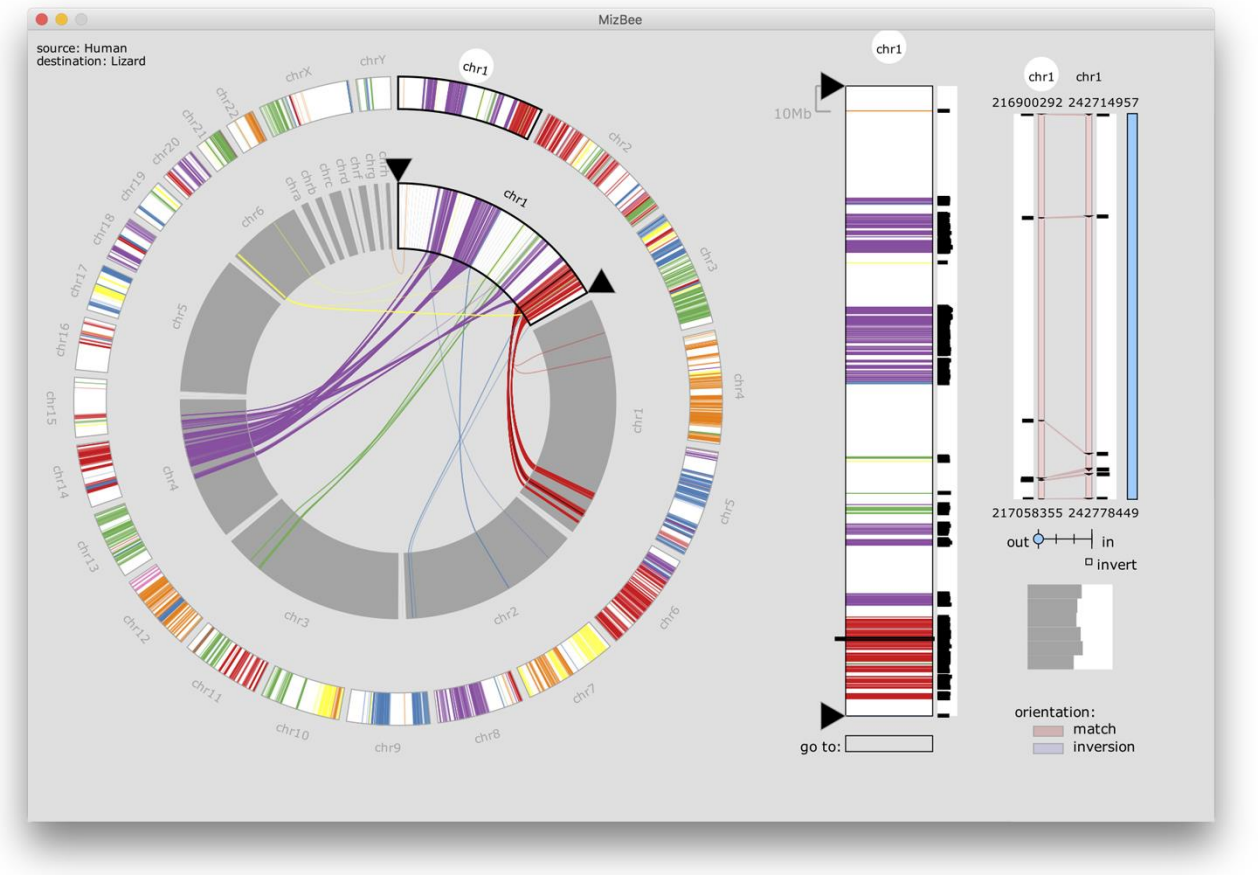

Figure 6-5. This tool for visualizing comparative genomics data has two levels of overview. Selecting a chromosome in the overview on the left triggers a more detailed overview of the selected chromosome in the middle view. In this middle view, regions of interest are selected to then trigger low-level details about individual genes in the detail view on the right.

Overview+detail supports open-ended exploration across the complete dataset and is useful for helping a user build a mental model of what is interesting within a dataset and what more detailed relationships should be explored. Choosing a good proxy for interestingness is essential to an effective overview+detail—oftentimes, the overview is designed to support a range of metrics that the user can switch between dynamically. It is often later in the process that a good understanding of what is interesting emerges. Furthermore, developing overview+detail tools typically requires some significant programming, making them somewhat heavyweight for early EDA. These tools are often developed later in the process, and often serve as the final visualization design.

Multiform Views and Dashboards

A single view can usually only effectively show three or four attributes. When trying to determine how multiple attributes are related to each other, then, a multiform linked visualization can show the connections between multiple attributes. A multiform visualization shows attributes across multiple visualizations, each tailored to most effectively show a small subset of the attributes. In this design pattern, no one view is best or primary and any one view by itself is insufficient.

Each view shows all of the data items, but just a portion of the data attributes. The views themselves are each designed to be most effective for showing one or a few of the attributes, and many views can be shown at once to support finding patterns, trends, and correlations across many attributes. Instead of primarily supporting

characterization of patterns in the *data items*, like a small multiples visualization, a multiform visualization supports characterization of patterns in the *attributes*.

This design pattern uses an interaction technique called *brushing and linking*, where selecting data items in one view triggers highlighting of those selected items in the other views, supporting fine-scale analysis of relationships across the attributes.

For example, in Figure 6-6 the view on the left is showing 2D spatial locations for each data item, with a metric encoded at each point, using color. We know from Chapter 5 that color is relatively ineffective for precisely comparing quantitative values; thus, the linked bar chart view on the right shows the metric for each data point using spatial encoding (height). The views are linked such that when the mouse hovers over a data item in the left view, the corresponding bar is highlighted in the right view—the interaction can be viewed in the video on the book's website.

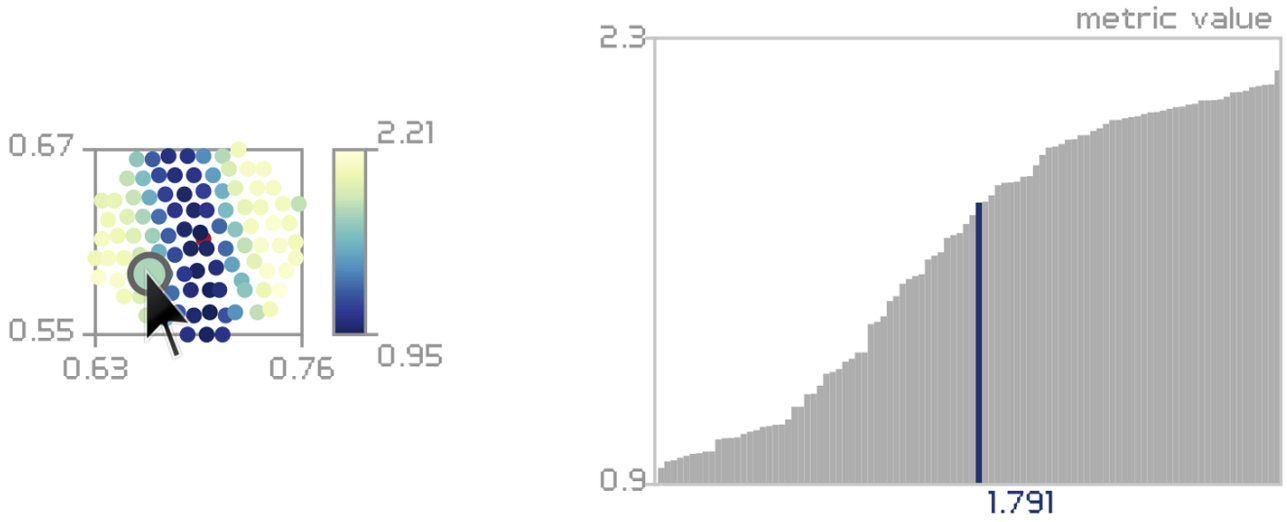

Figure 6-6. In this multiform visualization, both views are showing the same attribute across all the data items. On the left, that attribute is encoded with color, along with the spatial location of each data item. To overcome the perceptual challenges of making fine-scale comparisons of the attribute values using color, the linked view on the right encodes the attribute using a bar chart. The two views are linked together with brushing.

A multiform visualization such as this can give users access to a broad set of attributes across the complete dataset, making it particularly useful in the middle of the data counseling process.

Dashboards are a type of multiform visualization used to summarize and monitor data. These are most useful when proxies have been well validated and the task is well understood. This design pattern brings a number of carefully selected attributes together for fast, and often continuous, monitoring—dashboards are often linked to updating data streams. While many allow interactivity for further investigation, they typically do not depend on it. Dashboards are often used for presenting and monitoring data and are typically designed for at-a-glance analysis rather than deep exploration and analysis. An example of a business dashboard is shown in Figure 6-7.

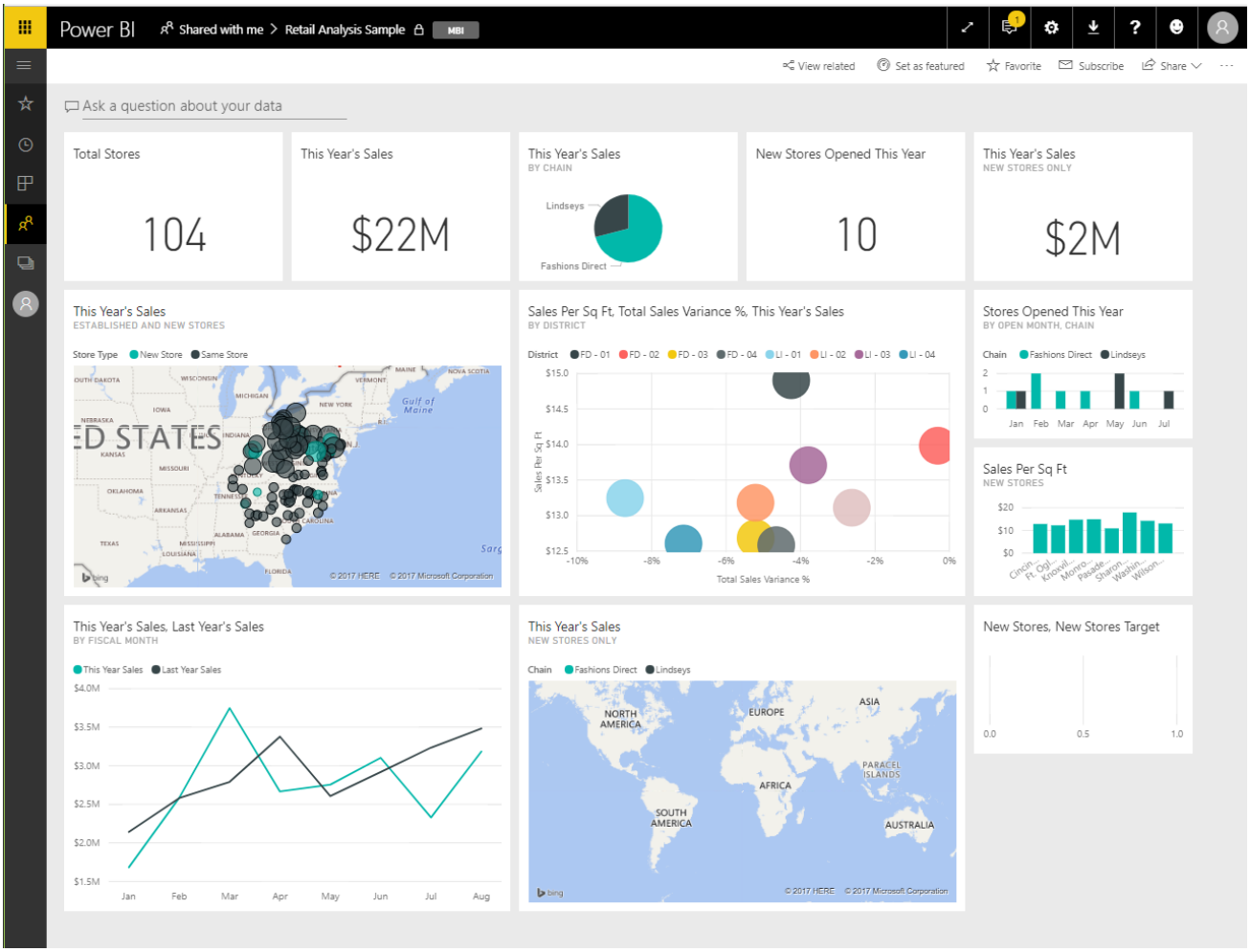

Figure 6-7. This sample business intelligence dashboard represents a number of different measures and dimensions of a dataset: single numbers summarize important features; scatterplots, bar charts, line charts, and maps address specific tasks. The cells are linked together: choosing a specific element in one panel acts as a filter or highlight against the others.

Overlays

A final MLV design pattern, overlays, uses a shared coordinate system to orient views together—these views are similar in that they share a common coordinate system, but could be different in the visualization type they use.This variation makes it easy to find patterns and trends among a small number of attributes along a common attribute, such as time or space. This form often occurs with geospatial and temporal data; the weather map in Figure 6-8 is an example. In this visualization, three different attributes are layered in the same view: temperature using color, pressure using isolines (contours), and wind speed and direction using wind-barb icons. All three attributes are using the same coordinate system, namely geospatial location over the continental United States. By visualizing these three attributes together, it is easier to make inferences about relationships between them.

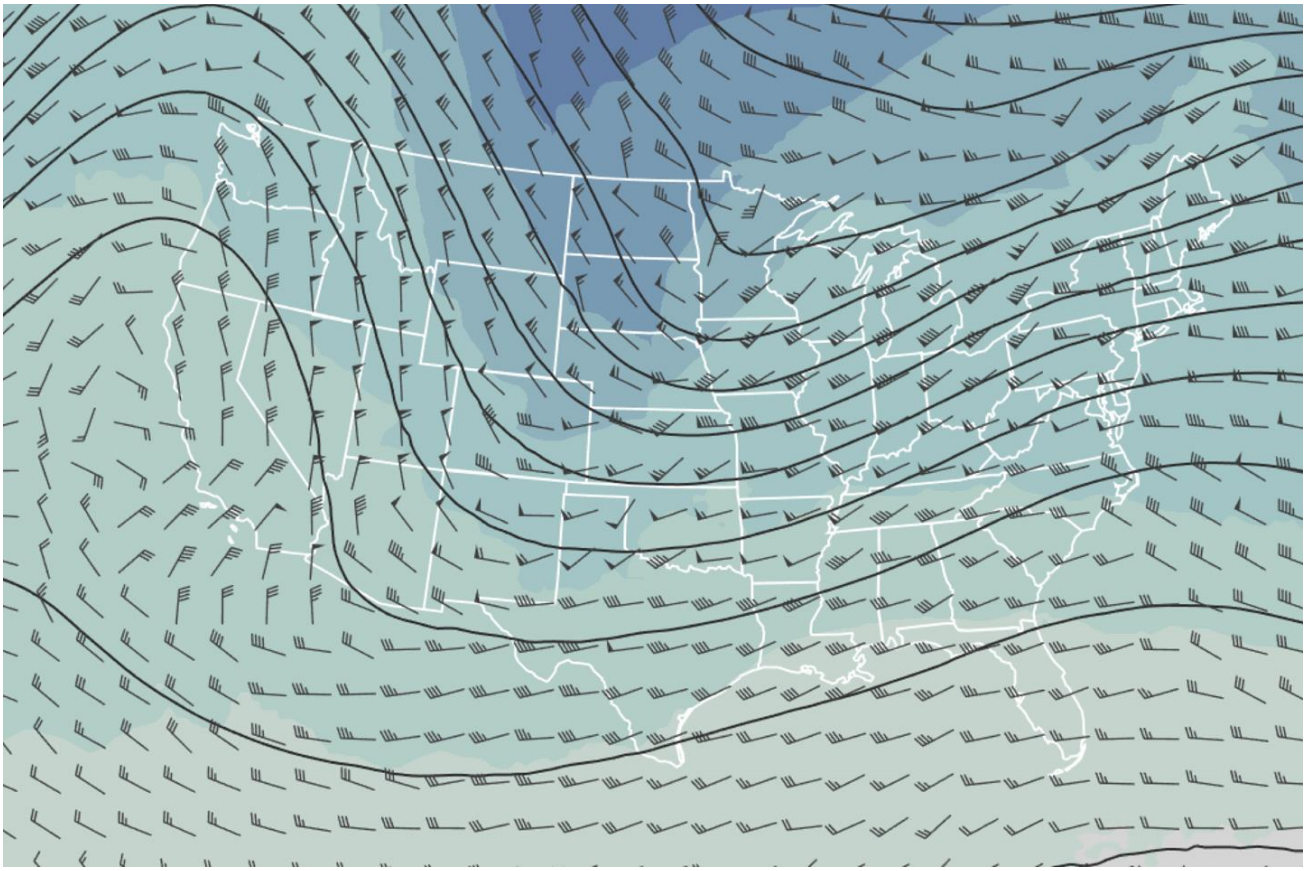

Figure 6-8. Overlays are often seen using a geospatial attribute as a common axis. In this weather map, temperature, pressure, and wind speed are overlaid on a map of the United States.

Besides comparing attributes, overlays are also good for saving pixels and presenting more information in a single display more compactly. On the other hand, they can add more visual complexity. Making detailed judgments about the weather attributes in Figure 6-8 requires a fair amount of attention—too much detail in an overlay can overwhelm a user quickly. Interaction can help with visual clutter, such as highlighting a specific layer of the overlay when a label in the legend is rolled over. In general, overlays are a great option when your analysis requires a small set of attributes to compare and the shared coordinate system is familiar.

Axis Alignment and Scale Consistency

One important aspect of all of these forms of multiple visualizations is finding and aligning shared axes. In general, if two different parts of a visualization are meant to show the same scale, they should be aligned and sized the same way. In a small multiple view of histograms, for example, ensuring that the bins are consistent among the histograms makes it far easier for the user to compare bins to each other. Similarly, when overlaying several series with different ranges, it is worth considering whether the percentage change is most important, which would allow for a common y-axis.

The principles of alignment and consistency play into Figure 6-9, which illustrates the value of maintaining consistent axes while showing independent color scales. The three maps show very different data—the population, percentage of the population that are engineers, and number of hurricanes. The shared coordinate system and aligned axes help the reader compare the maps; the different color palettes emphasize that the attributes, scales, and meaning are very different between the three charts.

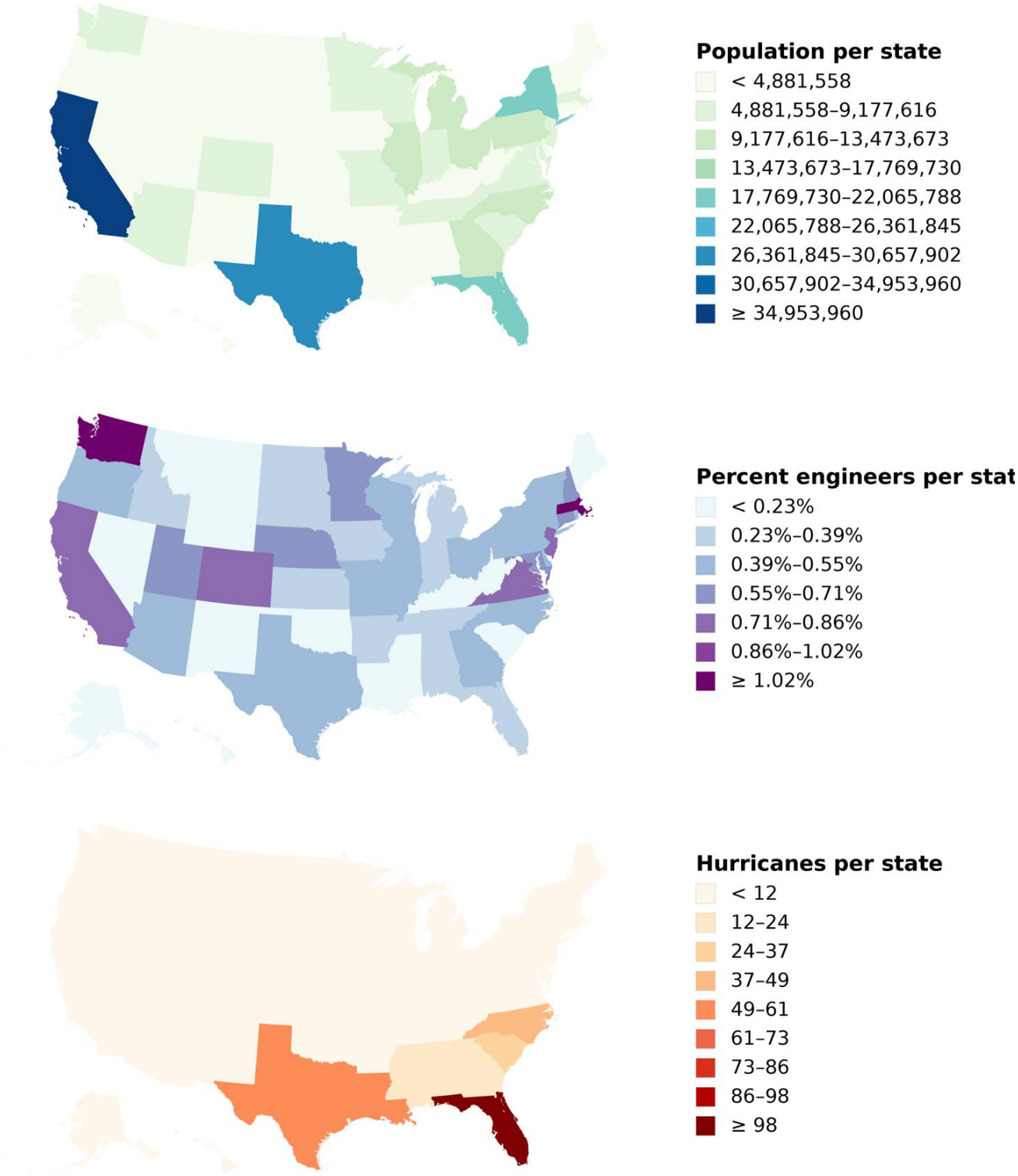

Figure 6-9. These three choropleths illustrate the value of aligning scales and maintaining coordinate systems. The maps show the population of each state, the percentage of the population that are engineers, and the number of hurricanes. (The very different map styles suggest that engineers do not cause hurricanes.)

Interacting with Multiple Linked Views

Many of these MLV design patterns support the concept of linking views by interacting with the data. The role of linked-view interactions is to select data in one view that is then reflected in another view, using the data as a selector. This is broadly referred to as *brushing and linking*.

The concept of brushing and linking brings together two subtly different types of interaction intentions: cross-highlighting and cross-filtering. For example, in a SPLOM it is common to select data items in one of the scatterplots to see how they are reflected in others; this is known as *cross-highlighting*. Cross-highlighting can be implemented when individual data points correspond in multiple charts.

Cross-filtering means that the selection on one chart removes data items from other charts. It might make sense, instead, to cross-filter on ranges or values of attributes—for example, by dragging along an attribute to mean "all data items with these values along this attribute."

Interestingly, there's little consensus on the exact specification of these two different intentions. Selecting a region can mean filtering to only a set of data items, or it can mean highlighting those points.

Figure 6-10 shows a cross-selection tool in action. The dataset (from World Bank Development Data) shows a series of countries, listed by the percentage of their population aged 15–64, and the percentage over 64. A scatterplot on the right side plots these two numbers against each other. The user has made a selection within the scatterplot; this highlights the corresponding data in both the lists.

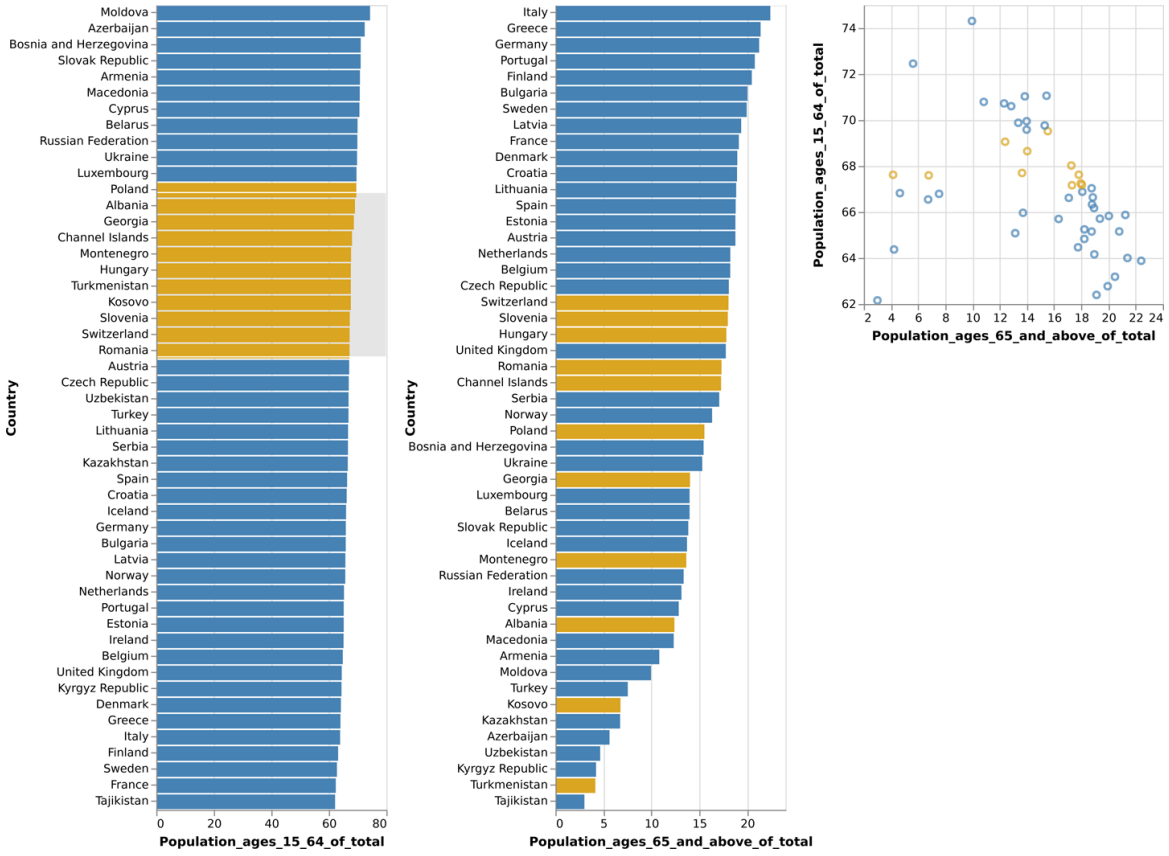

Figure 6-10. Cross-selection. The three views—two sorted bar charts and a scatterplot—are linked together. The user has selected a region of the scatterplot (grey box, orange dots), and the selected values correspondingly light up on the bar charts. This is based on the World Bank dataset.

MLVs and the Operationalization Process

The different types of MLVs tend to be useful at different stages in the operationalization process.

The SPLOM tends to be used early. It is designed to help an analyst dig around in an unfamiliar dataset, or to look at lots of different attributes and see whether there are any interesting correlations between them. SPLOMs are usually an exploratory tool, used when a user can't yet decide which dimensions will make good partitions or measures. It can help clarify the nature of the data and identify which dimensions will be interesting to visualize and explore.

Similarly, small multiples are a good way to rapidly scan how subsets of the data items compare to each other across several attributes. They are often useful at the early stages but continue to be useful later, when the final task is to compare aspects of the data by a partition.

Interactive multiform views often occur in the middle parts of the operationalization process. Brushing and linking between two views can help identify the parts of the data where interesting phenomena occur. The middle part of analysis is also where overview+detail visualizations are helpful—when you know enough about the data to be able to identify an *interestingness* measure and can use it to more richly explore the details of individual data items. Overlays are great here too, when it is known which few attributes are likely to be most important.

At the end of the operationalization, a dashboard is often the result of relentless pruning of ideas for proxies. Having examined the attributes and their interactions, the user now knows which proxies are useful for answering questions, and which attributes are most important. It begins to make sense to create multiple visualizations for different tasks. Each highly curated visualization helps to answer a definite and specific question.

Conclusion

Multiple linked views are design patterns that provide important support for making sense of complex and large datasets. Breaking up the data across multiple views avoids overwhelming a user with extremely dense visualizations, and also allows for optimization of each view based on the characteristics of the underlying data and task.

These design patterns are often used in conjunction with each other. Chapter 7 uses a variety of different visuals, with interactive linking between an overview-and-detail and a series of overlays.
Similarly, Chapter 8 illustrates an example of an MLV system that combines an overview+detail with small multiples, small multiples with overlays, and a two-level overview flow.

Further Reading

Meyer, Miriah, Tamara Munzner, and Pfister Hanspeter. "MizBee: A Multiscale Synteny Browser." *IEEE Transactions on Visualization and Computer Graphics* 15 (2009): 897–904. This paper describes the MizBee system, which combines the overview+detail pattern with small multiples, overlays, and multiform views in a single tool to address a series of tasks in biology.

Datasets

In addition to the datasets in Chapter 5, this chapter also uses:

World Bank development data

> Data from the World Bank about development indicators of countries, compiled from officially recognized international sources' World Statistics eXplorer.

[1] This visualization tool is discussed in "MizBee: A Multiscale Synteny Browser." See "Further Reading".