Keeping Multiple Views Consistent: Constraints, Validations, and Exceptions in Visualization Authoring

Zening Qu, *Student Member, IEEE*, and Jessica Hullman, *Member, IEEE*

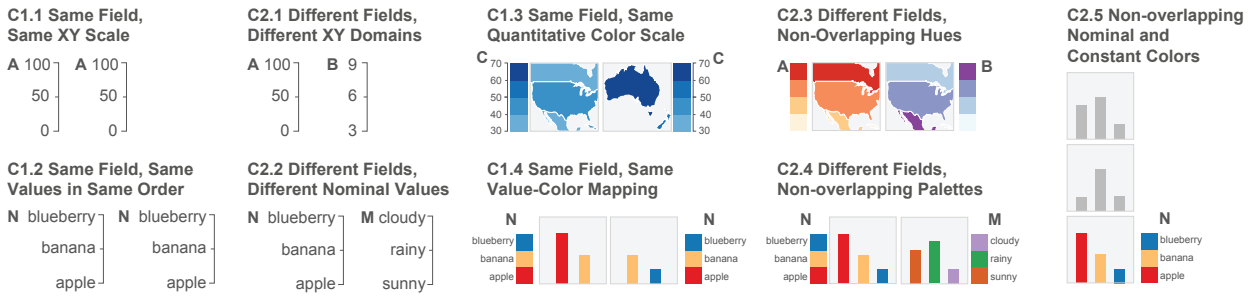

Fig. 1: Visualizations that contain multiple views have multiple scales (e.g., *xy* scales, *color* scales). We develop and study authors' reactions to a model that formalizes visual encoding consistency between scale pairs as the above constraints. C1.1, C1.2, C2.1, and C2.2 require a pair of quantitative or nominal fields to be encoded using the same or different *xy* scales across two views depending on whether the fields are the same or different. Applying a similar logic to color, C1.3 and C1.4 require the same quantitative and nominal fields to have the same color scales. C2.3 requires different quantitative fields to have different hues, while C2.4 requires palettes of different nominal fields to not overlap. C2.5 requires nominal palettes to not include constant colors of other views.

Abstract— Visualizations often appear in multiples, either in a single display (e.g., small multiples, dashboard) or across time or space (e.g., slideshow, set of dashboards). However, existing visualization design guidelines typically focus on single rather than multiple views. Solely following these guidelines can lead to effective yet inconsistent views (e.g., the same field has different axes domains across charts), making interpretation slow and error-prone. Moreover, little is known how consistency balances with other design considerations, making it difficult to incorporate consistency mechanisms in visualization authoring software. We present a wizard-of-oz study in which we observed how Tableau users achieve and sacrifice consistency in an exploration-to-presentation visualization design scenario. We extend (from our prior work) a set of encoding-specific constraints defining consistency across multiple views. Using the constraints as a checklist in our study, we observed cases where participants spontaneously maintained consistent encodings and warned cases where consistency was overlooked. In response to the warnings, participants either revised views for consistency or stated why they thought consistency should be overwritten. We categorize participants' actions and responses as constraint validations and exceptions, depicting the relative importance of consistency and other design considerations under various circumstances (e.g., data cardinality, available encoding resources, chart layout). We discuss automatic consistency checking as a constraint-satisfaction problem and provide design implications for communicating inconsistencies to users.

Index Terms—Visualization Design, Qualitative Study, Evaluation.

1 Introduction

Visualizations are often created and presented in multiples. In analysis, users experiment with different combinations of variables and encodings as they seek interesting patterns. The same quantitative field that is encoded by an *x* scale in one view may be encoded by a *color* scale in another view. Or, a field may be filtered differently across two views, resulting in different *x*-axis domains ($[min, max]$). When individual visualizations are combined for presentation in dashboards or slideshow presentations, authors may overlook inconsistencies between how the same variable is encoded differently across views, or cases where different variables look exactly the same across views, leading to confusion or misinterpretation among viewers.

Visualization systems typically leave it to the user to manually choose the encoding specifics to present data consistently. However, most existing visualization design criteria that users could rely on are not informative for multi-view designs. Instead, criteria focus on single visualizations, such as maximizing the ratio of data to ink in a single chart [28] or using the most perceptually effective visual encodings given a nominal or quantitative data field [16]. As a result, users often have to devise their own manual strategies to achieve encoding consistency. For example, the author of two charts showing *horsepower* and *mean horsepower* may manually align the scale domains to make the *mean* value more comparable (Fig. 2). Here, deciding whether and how to achieve encoding consistency requires careful consideration, because

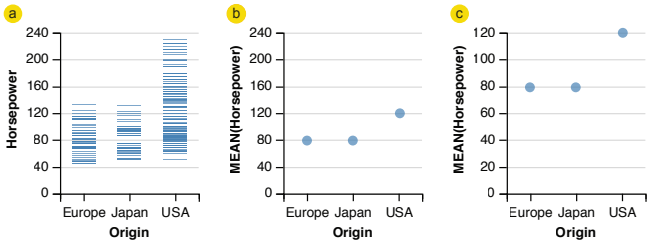

Fig. 2: Consistent *y*-scales (a, b) make it easier to relate *mean horsepower* to the raw values. If the scales were inconsistent (a, c), assessing the relation requires considerably more effort. Constraint C1.1 requires quantitative *xy* scales encoding the same field (*horsepower*) to have the same domain ($[0, 240]$).

- *Zening Qu is with University of Washington. E-mail: zqu@uw.edu.*
- *Jessica Hullman is with University of Washington. E-mail: jhullman@uw.edu.*

making the scales comparable means not fully exploiting the available space for depicting data. Ideally, visualization tools could help users identify and decide when and how to maintain consistency and balance with other design considerations. However, little is currently known about when and what types of encoding consistencies authors want to achieve as they create multiple views.

In this work, we take the first close look at how visualization authors perceive and achieve consistency between views when designing multiple views. We present the results of a wizard-of-oz inspired study, in which a "wizard" mimics an automated design assistant that evaluates the consistency of an author's work at multiple stages in the design process. To enable this study design, we contribute to prior work on consistency guidelines for visualization [9, 19, 30] by **extending a set of consistency constraints** for different encoding channels in 2D multivariate visualizations [19]. We provide an **initial validation of consistency constraints** by observing how authors perceive the importance of various forms of consistency between visualizations. Our work contributes the first detailed **characterization of authors' rationales** for tolerating inconsistencies under some conditions. Finally, we describe **design implications** for integrating consistency support in visualization authoring tools that emerged as a product of our study.

2 Related Work

Small multiples or trellis plots, a form of view faceting [18], are perhaps the most common form of visualization where identical encodings must be maintained across views to allow comparisons [4, 27]. Encoding consistency is also relevant in coordinated and multiple view (CMV) visualization systems, which assume an analyst viewing a set of visualizations simultaneously (e.g., [20]). Techniques like brushing and linking, in which a mark that is highlighted or colored in one frame is simultaneously highlighted or colored in other frames [3], help the viewer identify data more easily across views. Wang Baldonado et al. describe how consistency facilitates learning, makes comparisons easier, and helps prevent false inferences in CMV systems [30]. Their "Rule of Consistency" advocates consistency in system state through mechanisms like brushing and linking to ensure that view interactions like highlighting or zoom are applied across views of the same data. They also advocate consistency in interface affordances across views that a designer deems functionally equivalent, such as ensuring that two calendar views afford the same interactions. Other proposed view coordination operations from early CMV systems include allowing users to copy visual elements across frames to ensure consistent representation, and supporting aligning of multiple charts along common axes [7, 22]. More recently, six HCI and visualization experts identified consistency as one of the top ten factors that can explain the widest range of usability problems in visualizations [9]. However, a discussion of specific consistency rules is still missing from the literature.

More recently, design defaults in visualization recommender systems aim to ease cognitive processing of multiple charts, such as Voyager's consistent axes ranges [33, 34]. However, consistency guidelines have not been well integrated in visualization design guidance nor authoring tools. Common authoring systems, like Tableau [26], leave consistency concerns to the user with the exception of a few basic defaults, like Tableau's support for maintaining nominal color encodings despite different filters on a field. Instead, authoring suggestions that are incorporated to guide users tend to be based on principles that assume a single visualization target, such as *expressiveness* and *effectiveness* criteria for generating the single best chart given a dataset [16, 17].

Designing multiple views for presentation is a focus in narrative visualization, where models of the cognitive cost between pairs of views in a set have been proposed to help authors [11, 13]. However, the purpose of measuring the cost between views is for ordering views or designing animated transitions, rather than for ensuring consistency.

Munzner proposed that a general way of understanding the relation between multiple views is by examining whether two views have the same or different encodings, and whether they share all, some, or none of their data [18]. Our consistency constraints describe multiple view relations at the level of scales (instead of views), and describe detailed requirements for scale properties.

We take inspiration from Kosslyn [15], who proposed evaluating the semantic clarity of a graph by assessing, e.g., *representativeness*, whether elements represent the meaning of the viewer's preferred representation, *congruence*, whether the appearance of the marks is compatible with their meanings, and *between-level mapping*, whether every mark has only one meaning. Kosslyn advised applying the framework to multi-chart composite visualizations first by analyzing each chart separately, then by analyzing the composite view. However, examples are largely absent from the visualization literature. Similarly, the recently proposed principles of *invariance* and *unambiguity* state that the impression of a visualization should be determined by data, and that changes in the data should be clearly visible in the representation [14]. However, the algebraic process model comprised by these principles provides a theoretical contribution to demonstrate how a human can evaluate a single visualization. Our goal is to work towards automated support for encoding consistency in visualization authoring systems.

In prior work, we formulated an initial consistency model describing encoding-specific rules for consistency [19]. The model is organized around two high level constraints that operationalize Kosslyn's principles: **C1**: Encode the same data in the same way, and **C2**: Encode different data in different ways. The original model specifies constraints for *xy*, *color*, *size* and *shape* mappings, and was developed through analysis of media examples and visualization specification prototyping using declarative languages [25] to test specific cases. We continue to focus on *xy* and *color* mappings in the current work based on their prevalence. However, we also extend this model in several ways for our wizard-of-oz study. First, we formally define a scale and its constituent parts, and distinguish a scale from an axis or legend. We describe consistency using this new scale definition and add heuristics for identifying the same/different fields in Tableau (see Sec. 3). We add new constraints for nominal *xy* scales and constant color, add special cases for scales encoding *measure names*, and make several existing constraints stricter to solicit more reflections from participants. Finally, by studying participants' reactions to the resulting constraints, we provide an initial validation of the constraint model (see Sec. 5).

3 Consistency Constraint Model

In order to systematically observe how participants spontaneously maintain consistency and catch inconsistencies in our wizard-of-oz study, we adapt a consistency model from our prior work [19] that defines what counts as consistent and inconsistent views. The model starts from two high level constraints that can be specialized to specific *scales* (e.g., x or y scale, color scale). A scale is *a function and its inverse* [31], between a *data domain D* and *retinal range R*:

$$\textbf{Scale: } \{\textit{data values in } D\} \underset{f^{-1}}{\overset{f}{\rightleftharpoons}} \{\textit{retinal values in } R\}$$

There are two mappings in a scale. The mapping f encodes abstract data as retinal values (e.g., position values, color values) that can be perceived after rendering. The inverse mapping f^{-1} allows a viewer to decode data from the graph. We borrow the scale definition from ggplot2 [31] and the *domain* and *range* terminologies from D3 [5, 6]. We henceforth use "domain" to refer to data domain and use "range" to refer to the domain of retinal values.

We distinguish scale from *axis* and *legend*. The scale itself is a relation and not readily perceivable by the eyes [32]. However, a scale becomes perceivable after being visualized as an *x* or *y* axis, or a color, size or shape legend. Both the scale's *domain* and *range* should be visualized. For an axis, the domain is visualized by the tick labels and the range is visualized by the axis line. For a color legend, the domain is visualized by the data labels and the range is visualized by the colors in the legend. Therefore we sometimes use "domain" to talk about the data labels and use "range" to talk about the position or color ranges.

A scale is typically associated with one *field* and one *encoding resource*. A *field* is slice of the dataset that forms a meaningful semantic concept (e.g., *horsepower*, *origin* are distinct concepts). An *encoding resource* is a visual attribute for encoding data (e.g., *xy*, *color*, *size*, *shape*). The field and encoding resource define a scale's *type* (e.g., quantitative/nominal, *xy*/*color*). Multiple views may contain scales that encode the same field, or use the same encoding resource, or both [18],

making room for potential inconsistencies. We propose *two high level constraints* to check consistency between any two scales that have the same encoding resources (e.g., two position scales, two color scales):

C1 Same Field ⇒ Same Scales: The same data field is encoded the same way.

C2 Different Fields ⇒ Different Scales: Different data fields are encoded differently.

At a high level, C1 implies that the viewer's attention is not drawn to visual changes given a constant data field across views. When C1 is satisfied, C2 aims to ensure that any field that does change between visualizations is likely to draw the viewer's attention.

When two scales encode the same field, the axis or legend titles will naturally contain the same field name. In addition, the "same scales" requirement in C1 demands the scales' domain, range and mapping functions all to be the same. This may not be necessary for all scale and data types, but we intentionally keep C1 and its *xy* and *color* derivations strict to better elicit participants' actions and reactions to consistency concerns. On the other hand, when two scales encode different fields, the axis or legend titles will naturally differ, and the "different scales" requirement in C2 demands at least the domains (and sometimes the ranges and the mapping functions) to be different between the two scales. We present the specific constraints for *xy*, *color* channels together with their validations and exceptions in Sec. 5.

How to classify fields and their transformations as the same or different so as to match people's intuitions and expectations for consistency remains an unsolved problem. A field has a title (e.g., "horsepower"), a type (e.g., quantitative, nominal), a domain, a set of data values in the domain, and sometimes a unit. A field, or multiple fields, can be transformed to create a new field. For example, Tableau supports aggregation, count, binning and grouping [26]; SYSTATS supports mathematical, statistical, and multivariate transforms [32]. The field transformations can affect the new field's title (e.g., "average horsepower"), data type (e.g., count of nominal values), domain, data values and unit. One can build a model to classify fields as same or different using one or all of the above mentioned field properties and also infer from field transformations, but building and validating such a model with viewers is beyond the scope of this paper. Instead, for our wizard-of-oz study, we classified "same" and "different" fields from a Tableau user's perspective, using several heuristics. (1) Treat distinct dimensions and measures in the UI as different fields. (2) If the participant explicitly created a new dimension or measure and gave it a name, treat as a different field from the original field(s). (3) *Log*, *average*, *mean*, *median*, *min*, *max*, *bin* and *partition* transformations do not create new measures in the UI by default, therefore the new field is the same as the original field. (4) *Variance*, *standard deviation*, *quantile*, *rank*, *sum*, *count* and *group* transformations create a new field that is different from the original field. (5) A filtered field is generally considered the same field as the original field (e.g., after filtering out null values, we still consider a field same as original). (6) If the participant applied different filters on a field to create multiple subfields, and the subfield domains do not overlap, we consider the subfields different fields. For example, Fig. 3 compares the most expensive camera *brands* and the least expensive *brands*. We consider them different fields because the brand names do not overlap.

4 Study Design and Objectives

We present a qualitative study in which participants with prior visualization design experience authored multiple views for presentation. We use a wizard-of-oz style design, as used in prior visualization research to understand the implications of novel tools like whiteboard visualizations [29] and brain visualizations [2]. The wizard-of-oz approach enabled us to detect inconsistencies at various different decision points in view creation and evaluate their impact in a realistic context. Our specific research goals in using this study design were twofold. First, we wished to observe participants' spontaneous and warning-prompted strategies for maintaining encoding consistency across views. These behaviors indicate agreement with particular constraints, providing input for validating specific consistency constraints. Second, we wished to learn what types of rationales participants use to defend inconsistencies, and override particular constraints. Together these observations provide insight into how consistency concerns manifest in their work. In the interest of understanding how interactions with consistency unfold in an authentic authoring scenario, we noted where the participants' intentions were and were not well supported by the visualization authoring system. As a result, we later describe opportunities for enhancing authoring systems with novel forms of consistency support that were suggested by our observations.

4.1 Visualization Software, Participants and Datasets

We chose Tableau [26] for our study because (1) its dashboard and storyboard features allow easy transition from data exploration to presentation, and (2) it offers rich organization of multiple views: small-multiples in the same worksheet, worksheets in the same workbook, and worksheets organized as dashboards or storyboards.

We recruited 10 participants (4 males, 6 females, age 18 - 56) through an HCI email list at a large university. Their professions included graduate student, data librarian, market research consultant, and IT service representative. All participants were familiar with Tableau and had at least moderate experience with data visualization (i.e., had taken a class before or worked in visualization design, see supplementary material for details). All participants received two $25.00 Amazon gift certificates, one for each session.

Participants chose among possible datasets of cameras, cereals, country development, films, seat belts, and speed dating [1, 8]. We wanted participants to have a choice to increase the chances they would find a dataset genuinely interesting, making their work more authentic.

4.2 Detailed Procedure

Session 1 - Exploratory Analysis: Participants had about 50 minutes to get familiar with the dataset of their choice by making charts, and to export five or more charts from their exploration. We asked them to pretend that they were to publish their results online at the end of session 2, but we assured them that in session 1 they could focus on exploration and leave presentation to the next session. All participants were given a quick demo of Tableau's dashboard, storyboard, and annotation features, commonly used for organizing multiple views and highlighting data. To ensure that consistency violations were not due to a participant's lack of knowledge about the tool, we told participants that they could ask us for help if they encountered technical difficulties by sketching what they have in mind and we would make the changes between the sessions.

Before starting their exploration, participants completed a brief warm-up exercise on how to use a think-aloud protocol. In both sessions, we prompted the participants to talk about their goals and narrate their actions, particularly when they appeared to be making choices related to consistency across views. At the end of session 1, we exported all worksheets, dashboards, and storyboards created by the participant, and any sketches the participant created to specify intended revisions.
Between sessions: The experimenter made any requested revisions to an exported copy of participants' work. The consistency constraints (Sec. 5) were then applied to the generated views by comparing pairs of encodings of data fields across views. Each constraint violation was noted on a paper handout as a "system warning" that included screenshots of conflicting views and a warning message. For example, Bob (pseudonym) who created Fig. 5 received the following warning:

The data field "Price" has different scale domains in worksheets "Bang For the Buck Over Time" ([0, 600]), "Price Per Brand" ([0, 900]) and "Released in 2007" ([0, 6000]).

Session 2 - Presentation: Participants had about 50 minutes to refine the views for online presentation and reconcile system feedback where they felt it necessary. We first confirmed that participants were satisfied with any view modifications based on their sketches, then presented the paper handout and told participants that they had received warnings on their work from an automatic tool. We did not inform them in advance that the warnings would be related to consistency. Participants were encouraged to reflect on the warnings, including whether they would

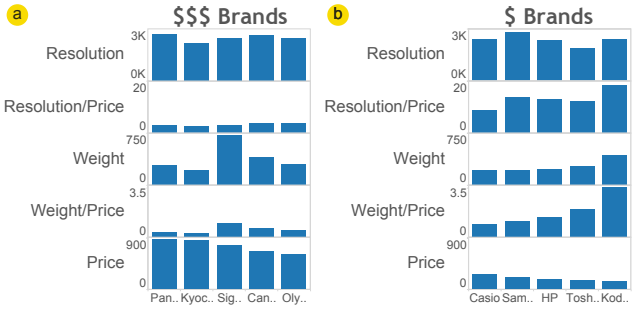

Fig. 3: Bob manually made the scale domains the same for five pairs of *y*-axes in a dashboard, to support the comparison of camera properties between the most and least expensive brands.

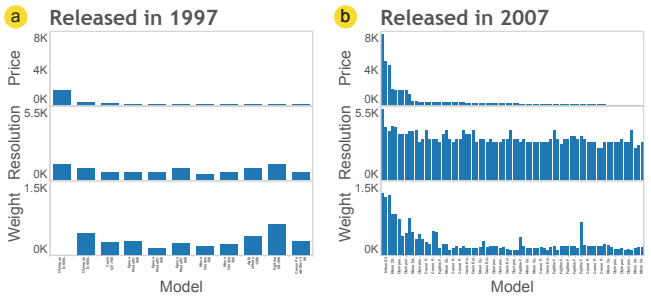

Fig. 4: Bob manually set the *price*, *resolution* and *weight* axes pairs to have the same domain in this dashboard in order to compare camera models released in 1997 and 2007.

revise their designs to try to resolve the inconsistencies or ignore them, and to talk aloud about their thoughts. After a short time, participants were asked whether they would ignore or revise for each warning.

For the remainder of session 2, as the participant continued to refine their visualizations for presentation, we surfaced additional warnings when the constraints were violated in new ways. The warnings were written down and placed near the participant's laptop, where she could read them when ready. After the participant had read the warnings, she was asked for her decision and rationale.

We observed and warned inconsistencies between views despite of their organization. That is, we applied constraints to all basic views in the same Tableau workbook, ignoring whether the views were on the same worksheet/dashboard/storyboard or not. Our "bag of views" approach elicited participants' reactions that started to form boundaries for various constraints (see Sec. 6.2).

Interview: Immediately after session 2, we conducted a four-question interview (see supplementary material) to gauge participants' opinions about the warnings and discuss where an automatic tool could help in the exploration-presentation process.

4.3 Counting Validations and Exceptions

We collected approximately 20 hours of audio and screen recordings in August 2016. In total, the 10 participants made 15 presentations (4 storyboards, 11 dashboards) containing 88 basic views (including trellis plot cells). The actual number of views created during exploration was much higher, as not all worksheets were selected for presentation.

We systematically applied the specific constraints in Sec. 5 to the data and summarized evidence that *validates* each constraint (participants achieving consistency on their own, or after warning) as well as *exceptions* to many constraints (based on the explicit reasons participants gave for not achieving consistency after being reminded).

We count instances of constraint validations by the number of scale revisions or verifications participants performed to make them consistent. If Bob aligned five pairs of *y* axes to achieve C1.1 consistency, we count these as ten instances of validations for C1.1. When participants ignored warnings generated by constraints, we identify the distinct reasons they gave as exceptions. For each exception, we count their instances by the number of scale revisions that *would* have to be made, if the constraint *were* to be satisfied. The count measures the "effect" of an exception by the number of scale revisions waived by it.

5 Findings

We present specific derivations of C1 and C2 (see Sec. 3) for *x*, *y* and *color* scales that encode quantitative or nominal fields. For each specific constraint C1.* or C2.*, we summarize their validations and exceptions using counting methods detailed in Sec. 4.3.

5.1 Quantitative XY Scales

C1.1 Same Field ⇒ Same XY Scale: If two *x* or *y* scales encode the same quantitative field, the scales should have the same data domain, retinal range, and mapping functions.

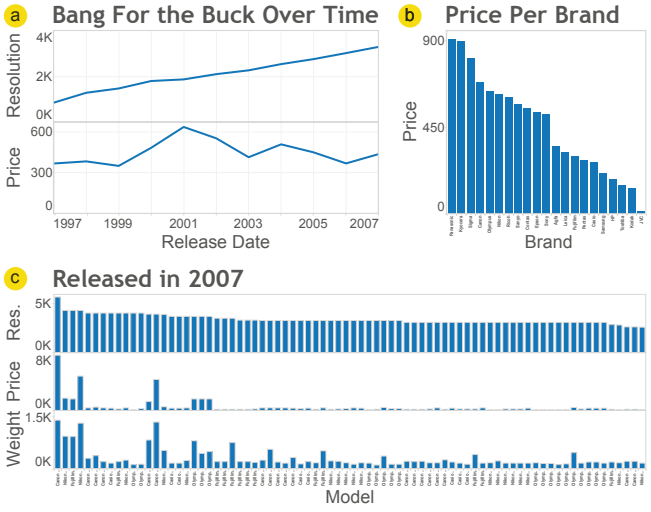

Fig. 5: Bob's dashboard of camera development and price changes over time. He overrode C1.1 which required *price* to have consistent scales in a, b and c. His rationale was that the *price* axes weren't juxtaposed and their *x*-axes encoded different fields. On the other hand, he manually set the *measure names* order in a and c to satisfy C1.2.

Recall that *xy* scales are visualized as axes. "Same domain" requires the two axes to have identical start and end tick labels. "Same retinal range" requires the two axes to have the same rendered length. "Same mapping functions" requires the function types to be the same. For example, a linear scale and a log scale encoding the same field would trigger a warning. Using these requirements to examine Fig. 2, which compares *horsepower* to *mean horsepower* (we treat as the same field using heuristics in Sec. 3), we can see that view a and b satisfies this constraint while view a and c fail the "same domain" requirement.

C1.1 Validation (28): Axis domains were manually made the same.

We observed 28 instances of validations (31% of total validations) from three participants spending conscious effort to achieve C1.1 consistency. In Fig. 3 and Fig. 4, Bob spontaneously enforced same domains for eight pairs of *y* scales to support comparison of the same field across two small multiples. For each scale pair, Bob manually picked the larger domain and applied it to both scales. The "same range" requirement was naturally fulfilled by the small multiples design. "Same mapping functions" was automatically satisfied by the default *linear* mapping option for all fields. In addition, Ada and Joy received six total C1.1 warnings prompting revisions. Ada manually set a pair of *x* scales to the same domain, even though the domain difference was very small: "The scale inconsistencies are very subtle (for human eyes). I wouldn't have caught on my own." Joy stated that she would have fixed five pairs of *x* axes if they were in her final presentation.

C1.1 Exception (15): Axes not juxtaposed, so scales can differ.

In Fig. 5-a, b and c, *price* had different domains and ranges. Upon

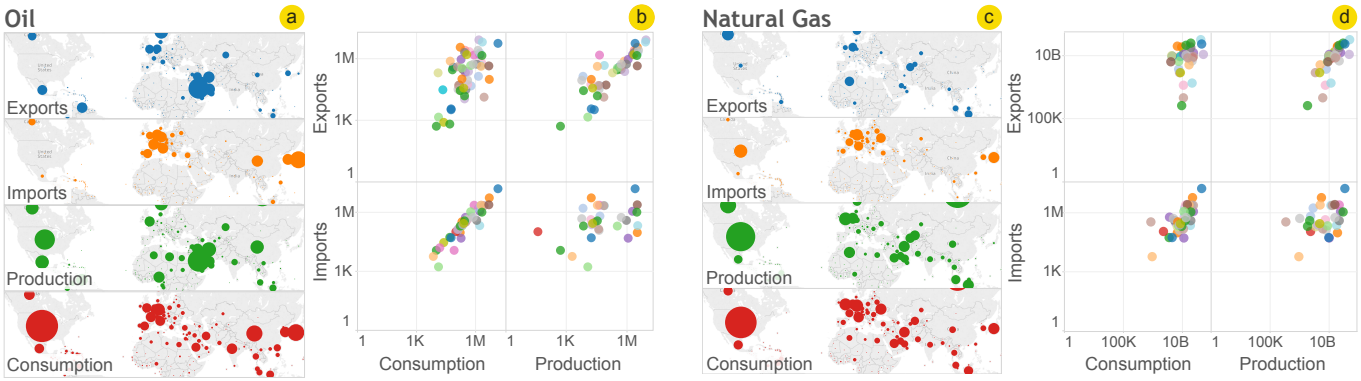

Fig. 6: Tim's oil & natural gas storyboard. He reasoned the overlapping nominal colors between the maps and the scatter plots was benign because in the scatter plots color encoded a high-cardinality field.

receiving an warning about this inconsistency, Bob reasoned he didn't want to revise because the *y* axes were not juxtaposed for comparison, and the perpendicular *x* axes encoded different fields (*release date*, *brand*, and *model*). Bob commented: "If these charts (containing *price*) were to go side-by-side, I would consider making the scales the same." For the same reason, he also allowed *resolution* to be inconsistent.

Bob's rationale was consistent with his design decisions for other sets of views. For example, in Fig. 3 and Fig. 4 where pairs of *y* axes *were* juxtaposed for comparison and the *x* axes *did* encode the same field (with different filters applied), he aligned the domains spontaneously (C1.1 validations). However, when Bob first made the two small multiples Fig. 4-a and b, they were separated as two single worksheets. At that point he chose not to make the *price*, *resolution* and *weight* scale pairs consistent in domain.

Zoe and Tim also had non-juxtaposed axes with inconsistent domains and used the same reason as Bob for not revising. Zoe had two time-series where the *y*-axes encoded *car accident occurrences* and the *x*-axes encoded *time* (by *month* and by *year*). She expected the *y* axis domains to be inconsistent because they were not juxtaposed and the *x* time units were different.

The juxtaposition exception suggest that if a comparison should be made, it may be more important for the scales encoding the same fields to be consistent, so that the "real" changes in data can show.

C1.1 Exception (18): That would be too much whitespace.

In addition to view juxtaposition, Bob was also concerned that making the scale domains the same would create too much whitespace when contemplating revisions for Fig. 3-5. He tried making scale domains the same and different and commented that making scale domains the same often means adding whitespace to one of the views, leaving details and trends harder to see in that view (e.g., Fig. 3-b *price*).

Although whitespace was a valid concern affecting many *x y* scales, it was not the deciding factor for Bob. After careful thinking, Bob made his decisions to revise versus ignore using the "juxtaposition for comparison" criteria. He left the inconsistent scales in Fig. 5 (counted as exceptions) and revised scales in Fig. 3 and Fig. 4 (counted as validations). Bob said if two views (Fig. 3-a and b) should be compared, revealing the quantitative differences across views on consistent scales becomes more important than preserving local details: "the *y* (axes) should have the same scales to support comparison, even if re-scaling will sacrifice a lot of screen space."

C2.1 Different Fields ⇒ Different XY Domains: If two *x* or *y* scales encode different quantitative fields, the two scale domains should differ, unless the field domains happen to be identical.

If two fields are different, the axis titles should naturally be different. Different fields also imply that the scale domains would automatically differ in most cases (e.g., it would be unlikely for *horsepower* and *miles per gallon* to have identical domains). Only when the different fields happen to have identical domains (e.g., probabilities, all within $[0,1]$) will the scales have the same data domain. Note that C2.1 does not enforce same or different retinal ranges. Two axes can have same or different lengths depending on view organization (e.g., layout). Nor does C2.1 impose same or different mapping functions. Linear, log, and power functions are commonly used for quantitative fields.

C2.1 Validation (10): Axis domains automatically differed.

As expected, numerous scale combinations satisfied C2.1 (i.e., scales encoding different fields typically have different domains). In most of these cases, the participant did not spend any conscious effort to achieve consistency. We do not count these "constraint naturally satisfied" cases because we want the validation count to quantify where authors' conscious, manual effort was spent (see Sec. 5.5 for analysis of effort).

We observed ten instances where Tim spent conscious effort related to C2.1. First, Tim consciously picked different mapping functions (*log* and *linear*) for two different fields (*military expenditures*, *military expenditures/GDP%*) in the same dashboard: "For percentages, don't make it log scale." Next, Tim manually set up the *log* scale for eight different fields (*exports*, *imports*, *production* and *consumption* for *oil* and *natural gas*, in Fig. 6-b and d). Tim's different choices in the two scenarios verified that the constraint should not prescribe same or different mapping functions for *x y* scales encoding different fields. The choice of mapping functions may be more dependent on the encoded data field than the consistency between scales.

5.2 Nominal XY Scales

C1.2 Same Field ⇒ Same Values in Same Order: If two x or y scales encode the same nominal (or ordinal) field, then the two scales should have the same data domain, retinal range, and mapping functions.

Following C1.2, two scales encoding the same field would have identical axes, which should facilitate comparison. "Same data domain" requires the two axes to contain the same nominal labels. "Same retinal range" requires the two axes to have the same lengths. "Same mapping functions" requires the nominal values to be sorted in the same order. The "same domain" requirement is intentionally strict and may not be satisfied in all cases. Even when two scales do not have exactly the same nominal values, we still require the overlapping nominal values to be sorted in the same order. For example, in an overview + detail design, the scale in the detail view will only encode a subset of values from the overview. It is helpful for the subset of values to be in consistent order with the overview.

We treat nominal *x* or *y* scale that encode *measure names* as a special instance of C1.2. *Measure names* is commonly used in Tableau to construct small multiples that compare different fields. For example, Fig. 5-c compares *resolution*, *price*, and *weight*. These fields are "nominal values" of *measure names*. The values of *measure names* are typically manually specified by the user in creating the particular trellis plot. Therefore, when two *x* or *y* scales encode *measure names*, we do not enforce same domain (or range), but only require the overlapping measures to be sorted in the same order.

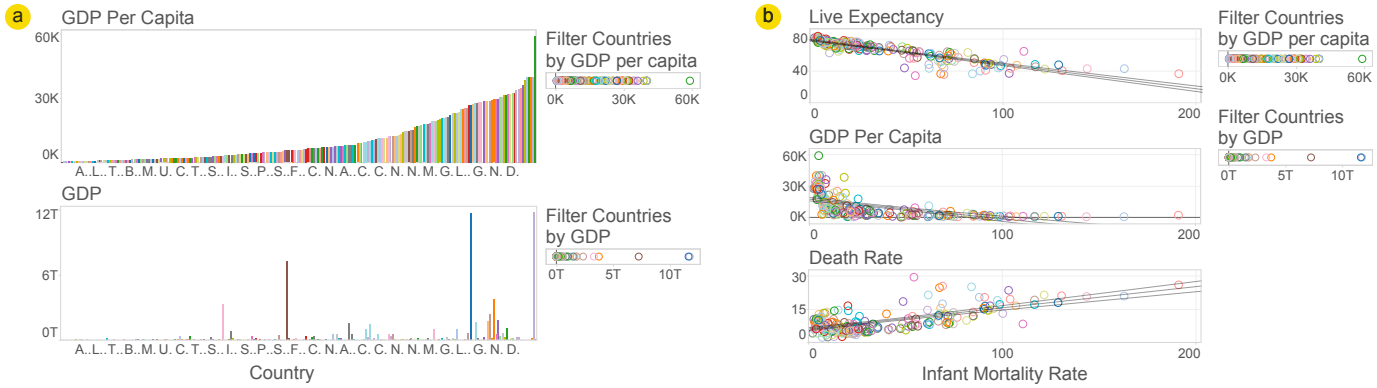

Fig. 7: Kai's dashboards on world economy and health. He made sure the color for each country stayed consistent in all views (including filters).

a

Go Out	Female	Male
Several times a week	21	21
Twice a week	20	21
Once a week	19	19
Twice a month	11	9
Once a month	3	5
Several times a year	1	4
Almost never	1	1

b

Go on Dates	Female	Male
Several times a week	2	4
Twice a week	6	8
Once a week	12	16
Twice a month	21	19
Once a month	16	20
Several times a year	20	19
Almost never	17	17

Fig. 8: Iva created two separate worksheets counting the frequency of people going out versus going on dates. She devoted considerable effort to make the color scale consistent across tables, including when interactive filters were applied.

C1.2 Validation (9): Apply same filter and sort to get identical axes.
Three participants spontaneously maintained C1.2 consistency. Kai had two bar charts with *country* names on *x* in a top-bottom layout (Fig. 7-a). He intentionally ensured each country's bar was vertically aligned. Whenever he revised filters or sorting criteria (which could affect the *x* values), Kai conscientiously verified that the two *country* axes still had the same nominal values, in the same order. Ann had two *country* heatmap tables in a left-and-right layout. Ann applied the same *null* filters and sorted *country* names alphabetically in both heatmaps to achieve C1.2 consistency. Similarly, Iva found when she added interactive filters to her heatmap table (Fig. 8-a), the table rows could change. This bothered Iva: "I don't like the graph to change (on filter)". She manually made the table rows show the raw data domain.

C1.2 Validation (10): Measure names were manually ordered.
When working on Fig. 3-ab and Fig. 4-ab, Bob duplicated worksheets to ensure measure names have consistent hand-picked order. When working on Fig. 5, Bob adjusted *measure names* order three times in response to three instances of C1.2 warnings.

Before the first adjustment, Bob received a warning saying that in his chart "Bang For the Buck Over Time," *measure names* was ordered [*resolution*, *price*] while in his chart "Release 2007," *measure names* was ordered [*price*, *resolution*, *weight*]. Bob remarked "that's a good warning" and changed "Release 2007" *measure names* order to [*resolution*, *price*, *weight*] to match "Bang For the Buck."

Later in the session, Bob felt the chart title "Bang For the Buck Over Time" emphasized *price*, so *price* should appear at the top of the chart. After revising, Bob received the same measure names order warning again, provoking him to think deeply about this design. He played with different orders, considering each chart's message and the consistency between them. He decided *resolution* should be the most important property for cameras and therefore should be on top for both the "Bang For the Buck" and "Release 2007" charts. Bob preferred to change the title to better fit the order [*resolution*, *price*], because having the same order in both charts seemed more important.

Bob received a third warning between Fig. 5 ([*resolution*, *price*, *weight*]) and Fig. 3 ([*resolution*, *weight*, *price*]). Bob said: "I'd definitely take a warning for that. And I will have to think about it pretty hard.". He again experimented with orders but didn't find a satisfying solution: "That's difficult, and I don't have a good answer for that."

C1.2 Exception (3): Nominal values sorted by different metrics.
In Joy's cereal nutrition dashboard (Fig. 9), cereal *name* appeared in three charts, sorted by *rating*, *serving size* and *calories*. When asked if she would adjust the *name* order, Joy said: "Each one (chart) individually is easiest to look at by its measure, instead of by *name*. I think it's more important for each graph to individually make sense than all of them being sorted in the same way."

We note that in Joy's dashboard, the nominal *xy* axes with different orders are not juxtaposed for comparison, which may have motivated the rationalization of the inconsistency (just like C1.1 exception). However, Joy did not bring up "juxtaposition" as a reason.

C2.2: Different Fields ⇒ Different Nominal Values: If two x or y scales encode different nominal (or ordinal) fields, the two scale domains should differ, unless the field domains coincide.
Like C2.1, we expected C2.2 to be automatically satisfied in many cases except when the nominal fields encoded happen to have exactly the same values (e.g., Likert scale answers).

C2.2 Validation (4): Nominal axis labels naturally differed.
Like C2.1 validations, participants created many examples where axes domains naturally differed, complying with C2.2. We only count the instances that involved conscious effort.

In Fig. 3, Bob compared two different fields *$$$ brands* and *$ brands* (by filtering the *brands* sorted by camera *price*). The two *x* scales differ in domain, satisfying C2.2. Similarly, in Fig. 4, Bob filtered and compared camera *models released in 1997* versus *models released in 2007*. Bob's confirmed: "I'm fully aware that the models are different (in different release years), and that's the way it should be".

We also observed a case where the two fields encoded happen to have identical domains, matching the "unless" clause in C2.2. In Fig. 8, Iva's two heatmap table rows contained survey responses: *go out frequency* and *go on dates frequency*. The two ordinal fields had identical *domains*. Iva manually sorted the values in the same order.

5.3 Quantitative Color Scales

C1.3 Same Field ⇒ Same Color Scale: If two color scales encode the same quantitative (or ordinal) field, they should have the same data domain, retinal range, and mapping functions.
Color scales are often illustrated by legends. "Same domain" means the start and end numbers should be identical between two color scales (or legends). The "same range" requirements means the start and end colors should be identical. "Same mapping functions" means the same numbers should be mapped to the same colors. For stepped color scales, this requires the break points to be the same.

C1.3 Validation (2): Color scale reused.
We did not observe many instances of validation related to C1.3 because using quantitative color scales to encode *the same field* across views was rare in our study. Iva spontaneously made colors consistent when applying different interactive filters to a heatmap to generate multiple views (Fig. 8-a). The color scale's range and mapping functions were

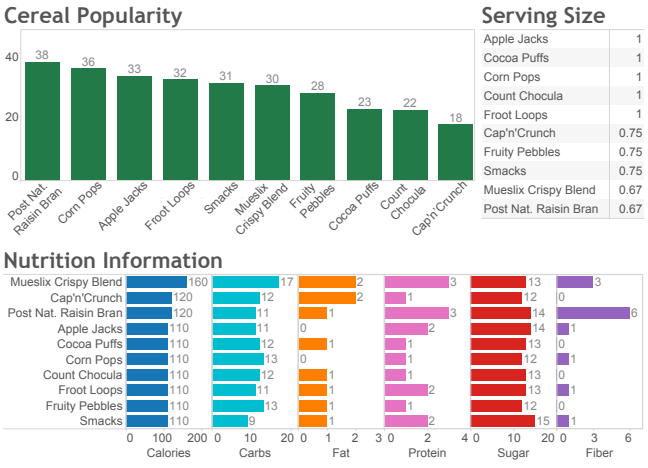

Fig. 9: Joy's dashboard of the ten most popular cereals. She used different constant colors to distinguish fields (e.g., *calories*, *carbs*) and sorted cereal *names* by different quantitative fields.

handcrafted by Iva to map the *high-middle-low* values of *participant count* to *green-yellow-white*. By default, the color scale *domain* in the heatmap would change when different filters were applied. Bothered by this inconsistency, Iva hard coded the color scale *domain* to [0, 21], so that "you get the context" when different filters are applied. After handcrafting the color scale's domain, range and mapping functions for heatmap a, Iva used it as a template to create heatmap b.

C2.3 Different Fields ⇒ Non-overlapping Hues: If two color scales encode different quantitative (or ordinal) fields, the color scales should differ in data domain and retinal range.

Similar to C2.1, "different domains" require the two color scales to encode different numerical domains. This should be naturally satisfied in most cases, unless the color scales coincidentally encode different fields that have identical domains. Because color hues tend to be interpreted as nominal, we expected different hues might be associated with different fields. "Different retinal ranges" forbids two color scales from containing the same hues. When examining the ranges, we check the start, end, and (for diverging scales) middle hues of two quantitative color scales. This constraint does not require the scales' mapping functions to be different (e.g., two scales can both use Jenks' natural breaks [12] to map data to color).

C2.3 Validation (2): Hues can denote different quantitative fields.

As expected, we observed hues associated with different quantitative color scales with Ada, who used a *green* scale for *HIV infection rate* and a *red* scale for *infant mortality rate* in two choropleth maps. Both scales had five color steps. Ada confirmed with us in the study that the same number of color steps (i.e., same mapping functions) for different fields was fine for her design as it is a common practice.

C2.3 Exception (7): Hues can also denote high & low, good & bad.

To our surprise, not all participants used distinct hues for different fields. We also observed hues used to denote categorical semantics (e.g., *high-low*, *good-bad*) derived from the original quantitative domain.

For example, Tom associated {*red, yellow, green*} to high, middle, and low values of three different quantitative fields. He said he didn't want viewers to relearn the color scheme: "I think that having the color scheme (hues) the same would be important, because if you are going through a whole bunch of these (charts), you get trained to think that red is the top and green is the bottom."

Later, Tom and Ann both associated *green* with "good" (e.g., *positive GDP growth rate*, *low external debt*) and *red* with "bad" (e.g., *negative GDP growth rate*, *high external debt*). Participants noted that their color choices for "good" and "bad" were not arbitrary: "It would be a major problem if you switch (red and green mappings)." (Tom) Participants also spontaneously avoided using *red* and *green* for other semantics when they were using them for "good" and "bad".

We conjecture that using hues to denote different fields, high-low semantics, or good-bad semantics might be three parallel strategies that cannot be used at the same time in the same set of views. For example, when Tom used hues for *good-bad*, he was no longer using hues to denote *high-low* or different fields. He expected that "text can help" to convey the concept of different fields: "if the variable that's been shown changes, that's been changed by the title of the map/graph, in my opinion." Tom also brainstormed when he *might* use hues for different fields: "If I have four maps next to each other, and they all show different variables, and I want to show the differences between the variables, *maybe* you do color differently but I think you can still have all colors as the same because you want to compare the four maps."

5.4 Nominal Color Scales

C1.4 Same Field ⇒ Same Value-Color Mapping: If two color scales encode the same nominal field, they should have the same data domain, retinal range and mapping functions.

Using C1.4, the same nominal values in a field would be presented by the same colors across charts. For example, *temperature*={*hot, cold*} will always be associated with *color*={*red, blue*}.

Just as nominal *xy* scales can encode *measure names* (see C1.2), nominal colors can encode *measure names* to help viewers distinguish different fields. As a special instance of C1.4, if two visualizations use color to encode *measure names*, the measure-color mappings should be identical. We do not require the scale domain or range to be identical as *measure names* are hand-picked in Tableau.

C1.4 Validation (6): Color meanings assigned and reused.

Three participants demonstrated conscious effort in ensuring that the same field was mapped to consistent colors across charts. Iva and Mia assigned semantically meaningful colors to nominal fields *gender* and *hot/cold* and reused the customized encodings throughout their presentation. Kai also checked to make sure that the same countries had the same colors in two charts (Fig. 7-a and b).

C1.4 Validation (12): Allocated color palette for measure names.

Many participants spontaneously hand-picked unique constant colors to distinguish different fields. For example, Joy commented on Fig. 9: "I like the different colors. It makes them (*calories*, *carbs*, *fat* etc.) distinct from each other." Similarly, Tim allocated distinct colors for *imports*, *exports*, *production* and *consumption* in Fig. 6-a and c.

In total, four participants (Zoe, Tim, Ada, Joy) allocated five nominal color palettes to five sets of *measure names*, but only one participant spontaneously kept the *measure name* colors consistent across views that appeared in a dashboard, presentation, or worksheet. The other three participants received six instances of C1.4 warnings saying either the same field had different colors or different fields had the same color. Participants considered these warnings to be important. They either reselected colors (five instances) or stated they would have reselected colors if the conflicting charts were presented together (one instance).

We also notice that participants were willing to revise colors even when the conflicting charts occurred on separate worksheets or different pages in a storyboard. One possible explanation is that changing the constant color for a chart is relatively easy in Tableau compared to other consistency revisions. Another possibility is that participants thought it would be beneficial for viewers' recognition of data if they could re-apply what they learned for field-color associations across pages.

C2.4 Different Fields ⇒ Non-overlapping Palettes: If two color scales encode different nominal fields, the two scales should differ in data domain, retinal range, and mapping functions.

C2.4 requires the same color should not be reused for different nominal values. If different fields are encoded, the "different domains" requirement should be naturally satisfied, unless by coincidence. "Different retinal range" requires two nominal palettes should *contain no overlapping colors*. For nominal scales, the mapping functions are defined by the data values and the retinal values. Hence the mapping functions should also differ. C2.4 intends to reduce the chance of misinterpreting the meanings of colors and lower the cognitive load of learning and re-learning the mappings when the viewer transitions between views.

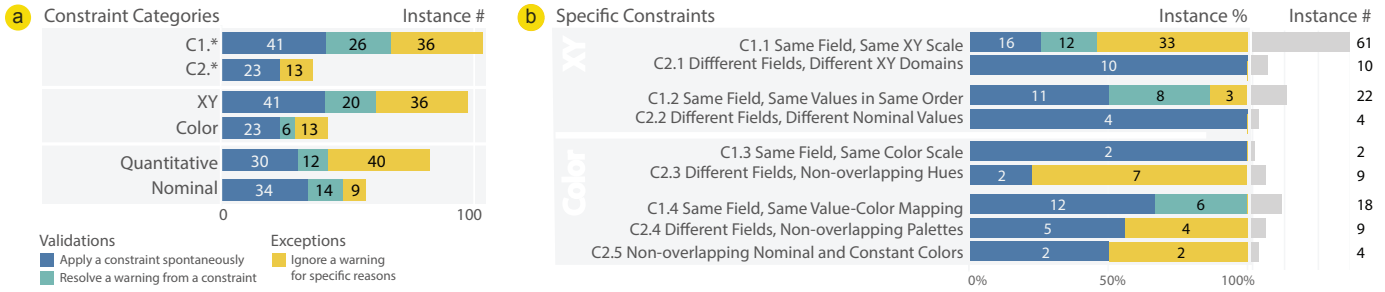

Fig. 10: Validation and exception counts by constraint origin, encoding resource, and data type (a) and their ratios per constraint (b).

C2.4 Validation (5): Avoided multiple meanings for the same color.

Multiple participants devoted conscious effort to ensuring singular color meanings across views. Iva spontaneously chose non-overlapping color palettes for two nominal fields: *gender*={*female, male*} $\rightleftharpoons$ {*pink, blue*}, *looking for serious relationship*={*true, false*} $\rightleftharpoons$ {*red, gray*}. Iva reasoned: "I wouldn't use this (pink) color (for *serious relationship*) because that's the girl color that I used." On the other hand, Mia received a warning between *cereal type*={*hot, cold*} $\rightleftharpoons$ {*red, blue*} and *cereal names* encoded by many different colors including *red* and *blue*, which meant different things across charts. Mia devoted considerable effort to resolving this issue. She reasoned that encoding a high-cardinality field like *cereal names* with nominal colors was not an effective design anyways, so she calculated a new ordinal field with low cardinality, and hand-picked colors other than *red* and *blue*.

C2.4 Exception (4): High-cardinality overlaps are more forgivable.

When a high-cardinality nominal field (e.g., *cereal names*, *country*) is encoded by color, it is less likely that the average viewer will remember the meaning of all colors well enough to recall it when viewing subsequent visualizations. Perhaps for this reason, C2.4 conflicts seemed less problematic to some participants when working with high-cardinality fields. For example, Tim used overlapping color palettes to encode *measure names* (Fig. 6-a,c) and *country names* (Fig. 6-b,d). He argued for his use of color: "Color may not be the best for countries, but it helps distinguish different countries." However, not all participants reacted the same when handling color conflicts involving high-cardinality fields. As we have seen in C2.4 validations, Mia painstakingly resolved the same type of inconsistency instead of using Tim's rationale.

C2.5 Non-overlapping Nominal and Constant Colors: If a color scale encodes a nominal field and another color scale does not encode any field (and just has a constant color), the two scales' retinal ranges should not overlap.

C2.5 addresses conflicts that can be created despite the previous two color constraints: even when the color scale is unmapped in a chart, a default constant mark color (e.g., all blue bars) could contradict nominal color encodings in another view.

C2.5 Validation (2): Changed constant color to avoid ambiguity.

Iva spontaneously changed the constant color of a bar chart from *blue* to *gray* because she had already used same *blue* to encode *male* in the same dashboard.

C2.5 Exception (2): High-cardinality overlaps are more forgivable.

Participants treated high-cardinality color conflicts as more forgivable, perhaps because such colors are less likely to be memorable. Tim and Kai gave this reason to overwrite C2.4 and C2.5, respectively. Before Kai finalized Fig. 7-b, he experimented with removing the nominal colors, making all points *blue*. He was pretty satisfied with this design and kept it for a while, even though the same *blue* color was also used in Fig. 7-a. In his view, the nominal colors in Fig. 7-a were for distinguishing different countries, not for establishing memorable mappings: "you might have multiple countries shown as peach color". In contrast, in Iva's dashboard the nominal color scale encoded a low-cardinality *gender* field, and the participant spontaneously avoided the constant color conflict.

5.5 Reflections on the Consistency Constraint Model

Though based on a small sample, the detailed validation and exception instances we observed enable the initial evaluation of the consistency constraint model (Fig. 10). Overall, we observed more validations than exceptions: 90 instances of validations and 49 instances of exceptions associated with five distinct constraints. For validations, we distinguish participants' conscious, spontaneous consistency maintenance strategies (blue) from those revisions triggered by our consistency warnings (teal). The spontaneous strategies depict where participants spend their mental and manual effort, achieving consistency on their own. The revisions triggered by warnings represent inconsistencies that are more likely to be overlooked. They point to where an automatic tool can be particularly helpful. The exceptions shed light on conditions when consistency should give room to other design considerations.

The lengths of the stacked bars in Fig. 10-a represent how often a type of constraint became relevant during exploration and presentation. C1.* constraints are more frequently a concern than C2.* constraints, as many C2.* requirements can be naturally satisfied. Constraints related to *xy* are considered more often than constraints related to *color*, probably because position encoding is used more often. We also notice that all revisions were made to comply with C1.* constraints. Out of the 26 revision instances, about half (14) were related to nominal scales encoding *measure names* (C1.2 and C1.4 special cases), which means participants tended to agree that *measure names* consistency was important but couldn't always achieve it on their own. Participants did not raise any explicit exceptions associated with *measure names*.

Fig. 10-b compares validation and exception instances of constraints. C1.4, C2.1, C2.2, and C1.3 were not challenged by exceptions in our study. On the other extreme, C1.1 and C2.3 saw high portions of exceptions, signaling "juxtaposition for comparison, whitespace" and "hue semantics" should be considered in the consistency trade-off.

Some of our observations suggest a more nuanced notion of the same or different fields may be necessary. For example, Tim dealt with many "similar" fields (e.g., *oil imports*, *gas imports*). Tim clearly separated the oil views and the gas views to convey the field difference, but he also intentionally matched the *measure names* order for *oil* and *natural gas* fields in Fig. 6-a&c, b&d to convey the field similarity.

Finally, we observed participants using several other strategies to achieve consistency that were not covered by our constraints. For example, Joy worked to assign consistent value labels across charts. Ada and Kai worked to apply consistent trendlines across charts. As an extreme example, although Kai knew that compared to logarithmic or exponential trend lines, linear trend line was a worse fit for *GDP per capita* in Fig. 7-b, he still chose the linear trend line because the other two fields used linear fits. Future work might incorporate consistency constraints for text elements and statistics, as well as considering consistent interactions, which we did not study here.

6 Design Implications for Authoring Tools

Our work provides an initial foundation for consistency support mechanisms in visualization authoring tools. We discuss two implications of our results: 1) the potential for *automated detection of inconsistencies* during view creation, and 2) the design of *mixed-initiative interface mechanisms* for suggesting detected inconsistencies to a user.

6.1 Automatic Inconsistency Detection & Resolution

We envision an automatic consistency design assistant embedded in visualization authoring tools that can detect inconsistencies and propose revisions using a constraint-satisfaction approach [21, 23]. In Sec. 3 we formalized scales as $\langle datadomain, retinalrange, mappingfunctions \rangle$ tuples. The scale components can be declaratively represented using grammars such as [24, 25]. By applying the constraints on scale representations, the system could output a high-level conclusion about whether two views are consistent as well as details about any inconsistent scale components, solving the detection problem. Revisions for resolving inconsistencies could be proposed given a built-in model of visualization effectiveness (e.g., codifying expressiveness and effectiveness [16] as criteria for guiding the search through alternative possible scales). However, searching for scale representations that satisfy all constraints in a large design space may be time and resource consuming. More than one solution may exist. Feedback from the user may be necessary to direct and accelerate the search results.

6.2 Designing A Mixed-Initiative Interface for Consistency

In the interview, all participants stated that having an automatic tool to monitor consistency would be helpful. However, participants also expressed concerns about how such a tool would surface warnings. Several of their responses echo known design principles of mixed-initiative systems [10]: warning at the right time (i.e., during presentation rather than exploration), minimizing intrusiveness, and minimizing warning repetition. Additionally, our study observations and participants' explicit comments about tool possibilities point to several more specific suggestions for designing automated support for encoding consistency in visualization software.

Acknowledge Different Boundaries for Different Constraints Participants' reactions to some constraints suggest that different circumstances, such as distinct view configurations, impact when a constraint is perceived as worth surfacing. For example, C1.1 was perceived by participants as applicable primarily when the axes of interest are juxtaposed. On the other hand, participants did not appear to perceive C1.2 for *measure names* to require that charts to be juxtaposed (on the same page). We also observed participants' reactions validating that the quantitative and nominal color constraints (C2.3, C1.4) apply to views across pages. A constraint model implemented in a consistency tool may set up different applicability boundaries for different constraints.

Enable Personalization of Constraint Models Participants varied at times during their sessions in how important they considered consistency constraints. For example, we observed various different reactions to C2.3, using non-overlapping hues in color scales for different quantitative variables. A tool could allow for customization of a constraint model to fit users' preferences in a single session (e.g., *equate field 1 and field 2 as same data*) and across sessions (e.g., *suppress warnings about nominal color inconsistency when colors represent measure names*). Such rules could be learned implicitly from the users' patterns of warning dismissal over time.

Estimate the Cost for Revision Though we did not attempt to systematically measure revision difficulty, some participants seemed more likely to revise their designs based on warnings when the revisions were relatively easy to achieve (e.g., changing constant color of a chart). Modeling how difficult a violated constraint is likely to be to fix would allow a tool to account for revision difficulty in selecting and prioritizing which warnings to surface.

Give Revision Previews & Accept/Reject Options Modeling potential revisions that satisfy constraints for different chart configurations could ease the users' perception of the burden of manual revision, and alleviate concerns voiced by our participants that the suggestion may change something more important than consistency. As Bob described, "When a tool is giving me warnings, it would be nice to see a little window that has a preview of the adjusted view. Then I can have the opportunity to see what my data *could* look like, and accept / reject." Such a model could incorporate design principles for single visualizations (such as expressiveness and effectiveness [16]) to support revision suggestions that minimize negative impacts to individual charts.

Clearly Convey Inconsistency Some of our participants had developed sophisticated notions of some consistency concerns but had not considered others. Visual or textual warnings should be carefully worded and closely tied to the specific chart elements to ensure that users understand the potential impacts of inconsistency before making revision decisions. Quantifying or otherwise representing the impact of inconsistency on individual plots (e.g., what percentage of pairwise point comparisons in a scatterplot are lost if the axis scale is changed, assuming a perceptual model) could help in communicating the potential impact of inconsistencies.

6.3 Limitations

We did not study how consistency impacts viewers' perception. In this way, our work begins from the same premise as Wang Baldonado et al. [30], who take it as a premise in proposing guidelines for CMV systems that consistency makes comparisons easier, reduces inaccurate inferences, and facilitates learning despite a potentially complex set of charts. Future work might use controlled studies to identify what types of inconsistencies are most likely to lead to misinterpretations among viewers with varying experience levels.

Our study results are based on in-depth, qualitative observation of a small sample of moderately experienced Tableau users from one university. While studying ordinary users allowed us to confirm that an automatic tool could help authors achieving consistency and we received a lot of valuable inputs from our participants, experts might be able to provide more insightful justifications for their consistency decisions. The similar institutional background of our participants might also have biased participants' decisions through visualization education. It is important to corroborate our findings with other author populations such as professional designers, users of other tools, and authors from more diverse organizations in future work. Because participants were aware that they were participating in a research study, they may have deduced that the experimenter was involved in developing the "tool" that provided the warnings. This may have lead to acquiescence bias, where participants express more positive opinions to please the experimenter.

For the purposes of our study, we focused on constraints that were limited to *xy* and *color* encodings of *quantitative* and *nominal* data. Our constraint model could be further expanded and tested for ordinal, temporal and geographical data encoded by other visual attributes. Additionally, we do not cover constraints to deal with interaction consistency, such as the benefits of making sure that functionally equivalent views afford the same interactions [30].

Although we systematically applied constraints to views created by participants between sessions and played back our screen recordings in our analysis, we might still have missed certain validations and exceptions during the sessions.

7 Conclusion

We presented a wizard-of-oz study in which participants were presented with warnings about encoding inconsistencies across views. Our work extends and validates a constraint model describing desirable consistencies in *xy* and *color* encodings across views based on the identity of data in those views. Our wizard-of-oz study provides the first close investigation of visualization authors' perceptions and practices around encoding consistency. Considering how sets of visualizations make data similarities and differences across views immediately recognizable to viewers is an important but often overlooked aspect of visualization design. Consistency warnings impacted this process by reminding authors of consistency among other design considerations.

We identified existing strategies used by authors to achieve particular forms of consistency, exceptional cases where authors wished to overwrite consistency concerns, and perceived tool needs, setting the foundation for future development of encoding consistency principles and tool support. For example, future work could pursue constraint satisfaction approaches to surface consistency warnings, as well as additional development and study of how the authoring process can be supported by human-in-the-loop tools.

REFERENCES

[1] R datasets. https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/00Index.html. Accessed: 2016-09-13.

[2] D. Akers. Wizard of oz for participatory design: inventing a gestural interface for 3d selection of neural pathway estimates. In *CHI'06 Extended Abstracts on Human Factors in Computing Systems*, pp. 454–459. ACM, 2006.

[3] R. A. Becker and W. S. Cleveland. Brushing scatterplots. *Technometrics*, 29(2):127–142, 1987.

[4] R. A. Becker, W. S. Cleveland, and M.-J. Shyu. The visual design and control of trellis display. *Journal of computational and Graphical Statistics*, 5(2):123–155, 1996.

[5] M. Bostock and J. Heer. Protovis: A graphical toolkit for visualization. *IEEE transactions on visualization and computer graphics*, 15(6), 2009.

[6] M. Bostock, V. Ogievetsky, and J. Heer. D^3 data-driven documents. *IEEE transactions on visualization and computer graphics*, 17(12):2301–2309, 2011.

[7] M. C. Chuah, S. F. Roth, J. Mattis, and J. Kolojejchick. Sdm: Selective dynamic manipulation of visualizations. In *Proceedings of the 8th annual ACM symposium on User interface and software technology*, pp. 61–70. ACM, 1995.

[8] J. R. Eagan. Igr 204 project datasets. IGR 204: Visualization class at Télécom ParisTech. Accessed: 2016-09-13.

[9] C. Forsell and J. Johansson. An heuristic set for evaluation in information visualization. In *Proceedings of the International Conference on Advanced Visual Interfaces*, pp. 199–206. ACM, 2010.

[10] E. Horvitz. Principles of mixed-initiative user interfaces. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, pp. 159–166. ACM, 1999.

[11] J. Hullman, S. Drucker, N. H. Riche, B. Lee, D. Fisher, and E. Adar. A deeper understanding of sequence in narrative visualization. *IEEE TVCG*, 19(12):2406–2415, 2013.

[12] G. F. Jenks and F. C. Caspall. Error on choroplethic maps: definition, measurement, reduction. *Annals of the Association of American Geographers*, 61(2):217–244, 1971.

[13] Y. Kim, K. Wongsuphasawat, J. Hullman, and J. Heer. Graphscape: A model for automated reasoning about visualization similarity and sequencing. In *Proceedings of ACM Computer-Human Interaction (CHI)*, 2017.

[14] G. Kindlmann and C. Scheidegger. An algebraic process for visualization design. *IEEE TVCG*, 20(12):2181–2190, 2014.

[15] S. M. Kosslyn. Understanding charts and graphs. *Applied Cognitive Psychology*, 3(3):185–225, 1989.

[16] J. Mackinlay. Automating the design of graphical presentations of relational information. *ACM Transactions On Graphics (TOG)*, 5(2):110–141, 1986.

[17] J. Mackinlay, P. Hanrahan, and C. Stolte. Show me: Automatic presentation for visual analysis. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1137–1144, 2007.

[18] T. Munzner. *Visualization analysis and design*. CRC Press, 2014.

[19] Z. Qu and J. Hullman. Evaluating visualization sets: Trade-offs between local effectiveness and global consistency. In *BELIV*, pp. 44–52, 2016.

[20] J. C. Roberts. State of the art: Coordinated & multiple views in exploratory visualization. In *Coordinated and Multiple Views in Exploratory Visualization, 2007. CMV'07. Fifth International Conference on*, pp. 61–71. IEEE, 2007.

[21] F. Rossi, P. Van Beek, and T. Walsh. *Handbook of constraint programming*. Elsevier, 2006.

[22] S. F. Roth, M. C. Chuah, S. Kerpedjiev, J. A. Kolojejchick, and P. Lucas. Toward an information visualization workspace: combining multiple means of expression. *Human-computer interaction*, 12(1):131–185, 1997.

[23] S. Russell, P. Norvig, and A. Intelligence. A modern approach. *Artificial Intelligence. Prentice-Hall, Egnlewood Cliffs*, 25:27, 1995.

[24] A. Satyanarayan, D. Moritz, K. Wongsuphasawat, and J. Heer. Vega-lite: A grammar of interactive graphics. *IEEE Transactions on Visualization and Computer Graphics*, pp. 1–1, 2017.

[25] A. Satyanarayan, K. Wongsuphasawat, and J. Heer. Declarative interaction design for data visualization. In *UIST*, pp. 669–678. ACM, 2014.

[26] C. Stolte, D. Tang, and P. Hanrahan. Polaris: A system for query, analysis, and visualization of multidimensional relational databases. *IEEE Transactions on Visualization and Computer Graphics*, 8(1):52–65, 2002.

[27] E. R. Tufte. Envisioning information. *Optometry & Vision Science*, 68(4):322–324, 1991.

[28] E. R. Tufte and P. Graves-Morris. *The visual display of quantitative information*, vol. 2. Graphics Press Cheshire, CT, 1983.

[29] J. Walny, B. Lee, P. Johns, N. H. Riche, and S. Carpendale. Understanding pen and touch interaction for data exploration on interactive whiteboards. *IEEE Transactions on Visualization and Computer Graphics*, 18(12):2779–2788, 2012.

[30] M. Q. Wang Baldonado, A. Woodruff, and A. Kuchinsky. Guidelines for using multiple views in information visualization. In *Proceedings of the working conference on Advanced visual interfaces*, pp. 110–119. ACM, 2000.

[31] H. Wickham. A layered grammar of graphics. *Journal of Computational and Graphical Statistics*, 19(1):3–28, 2010.

[32] L. Wilkinson. *The grammar of graphics*. Springer Science & Business Media, 2005.

[33] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager: Exploratory analysis via faceted browsing of visualization recommendations. *IEEE TVCG*, 22(1):649–658, 2016.

[34] K. Wongsuphasawat, Z. Qu, D. Moritz, R. Chang, F. Ouk, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Voyager 2: Augmenting visual analysis with partial view specifications. In *Proceedings of ACM Computer-Human Interaction (CHI)*, 2017.