Evaluating Visualization Sets: Trade-offs Between Local Effectiveness and Global Consistency

Zening Qu
University of Washington HCDE
zqu@uw.edu

Jessica Hullman
University of Washington iSchool
jhullman@uw.edu

ABSTRACT

Evaluation criteria like expressiveness and effectiveness favor optimal use of space and visual encoding channels in a single visualization. However, individually optimized views may be inconsistent with one another when presented as a set in recommender systems and narrative visualizations. For example, two visualizations might use very similar color palettes for different data fields, or they might render the same field but in different scales. These inconsistencies in visualization sets can cause interpretation errors and increase the cognitive load on viewers trying to analyze a set of visualizations. We propose two high-level principles for evaluating visualization set consistency: (1) the same fields should be presented in the same way, (2) different fields should be presented differently. These two principles are operationalized as a set of constraints for common visual encoding channels (`x`, `y`, `color`, `size`, and `shape`) to enable automated visualization set evaluation. To balance global (visualization set) consistency and local (single visualization) effectiveness, trade-offs in space and visual encodings have to be made. We devise an *effectiveness preservation score* to guide the selection of which conflicts to surface and potentially revise for sets of quantitative and ordinal encodings and a *palette resource allocation* mechanism for nominal encodings.

Keywords

Visualization Set; Consistency Effectiveness Trade-offs

1. INTRODUCTION

Traditional visualization evaluation metrics, including design guidelines and heuristics such as Mackinlay's effectiveness and expressiveness criteria [16] and Tufte's data density theory [29] are designed to be applied to a single visualization independently. However, today we often see visualizations as sets rather than standalones. In applications such as recommender systems [34, 30, 18] and narrative visualizations [23, 13], the viewer needs to interpret many different visualizations in parallel or in quick succession. This process is often cognitively difficult due to inconsistencies in encodings across the views, in spite of authors' efforts to group similar visualizations together [34, 13] and use animation and annotations to guide viewers [12, 23].

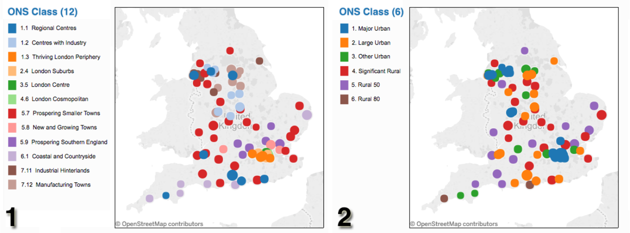
ONS Class (12)
1.1 Regional Centres
1.2 Centres with Industry
1.3 Thriving London Periphery
2.4 London Suburbs
3.5 London Centre
4.6 London Cosmopolitan
5.7 Prospering Smaller Towns
5.8 New and Growing Towns
5.9 Prospering Southern England
6.1 Coastal and Countryside
7.11 Industrial Hinterlands
7.12 Manufacturing Towns
1
© OpenStreetMap contributors
ONS Class (6)
1. Major Urban
2. Large Urban
3. Other Urban
4. Significant Rural
5. Rural 50
6. Rural 80
2
© OpenStreetMap contributors

Figure 1: Two maps show different UK healthcare group locations. The same colors represent different data in the two views, requiring viewers to maintain several meanings for each color value in memory as they analyze the set.

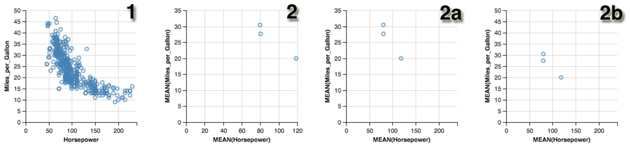

Figure 2: Two views depicting of `Horsepower` and `Miles_per_Gallon` (views 1 and 2) are inconsistent in `x` and `y` scales but show the same underlying fields. View 2 shows mean values for both variables, grouping models by country of origin. View 2a revises view 2 so that the `x` scale is consistent with view 1; View 2b makes both `x` and `y` scales consistent with view 1. Data source: [1].

For example, the two maps in Fig. 1 look similar. As a result of Gestalt principles like similarity [32], a viewer of these maps might think that they represent two overlapping fields because the same color values appear in each . Even when the viewer notes the non-overlapping meanings between the fields by reading the legend, she may have trouble moving between the plots. To do so requires her to intentionally suppress the meanings she just decoded in order to learn the new encoding. Had the map author realized the potentially confusing encoding reuse, he or she might have chosen a more consistent design given the semantics of the data fields, such as selecting a different nominal palette to apply to each map.

BELIV '16 Baltimore, Maryland USA
© 2016 ACM. ISBN 978-1-4503-2138-9. . . $15.00
DOI: 10.1145/1235

In other cases, sets of visualizations depict different filters and transformations of the same underlying data fields, but with different visual representations. Both visualizations in Fig. 2.1 and 2.2 represent the same fields Horsepower and Miles_per_Gallon, but Fig. 2.1 depicts the raw data for all observed cars, while Fig. 2.2 depicts MEAN(Horsepower) and MEAN(Miles_per_Gallon) for US, European, and Japanese cars. However, the two visualizations encode the data using different x and y scales, which hinders comparing the views without effort. Had 2.2 been generated with consistent scales such as in Fig. 2.2b, the viewer would more quickly recognize that the fields are the same as Fig. 2.1. She could also easily compare data across the views, for instance noting that cars from each of the countries fall near the densest center portion of the full distribution.

We propose that the traditional single-visualization evaluation metrics may not be sufficient in motivating cognitively efficient visualization sets: sets of views that align with viewers' natural impressions of encoding semantics and reduce the effort required to move between views in analysis. We outline our vision of an approach for evaluating sets of visualizations that can be automatically instantiated in design tools to support these goals.

We envision several applications of an automated approach to evaluating visualization sets:

Scenario 1: An analyst is exploring the cars data set [1] using a visualization recommendation browser like Voyager [34]. The system generates multiple distinct views of the data, each optimized by single visualization evaluation criteria (e.g., Fig. 2.1, 2.2). Before presenting the views to the analyst, an automated "set evaluator assistant" checks for inconsistencies in the set using the constraints we propose in Sec. 6. In the case of Fig. 2 the evaluator detects that the x scales are inconsistent. For each constraint violation, the evaluator decides the appropriate next step using mechanisms we propose in Sec. 7 to balance loss of comparability of data in a single view with the added ease of mental integration of information across views from consistency. The evaluator concludes that revising Fig. 2.2's x and y axes to be consistent with those of Fig. 2.1 does not significantly impact perception of the data in Fig. 2.2, and therefore makes the revision.

Scenario 2: A designer is using an authoring tool such as Tableau Public [2] or Lyra [21] to create multiple visualizations about UK National Health Services (like Fig. 1) for a data story. After the designer assigns a categorical color palette to a nominal field (Fig. 1.1), a built in "set design assistant" records which colors are consumed. When the designer later applies the same or an overlapping scale to depict a different nominal field (Fig. 1.2), the design assistant detects the potential conflict and surfaces it for the designer to address.

The approach we propose begins with two high level design constraints: **C1**: The same data fields should be encoded the same way. **C2**: Different data fields should be encoded differently. We contribute an operationalization of these constraints for common quantitative, ordinal, and nominal encodings, which we developed through case studies using real-world visualization set examples. We further propose two mechanisms for negotiating trade-offs between local (single visualization) optimization of encodings (where maximizing comparisons in the visualization is often a goal), and global consistency constraints. Our *effectiveness preservation score* analyzes the loss in data comparisons of the rendered view if an encoding is applied consistently between two views. A *color and shape palette allocation* strategy helps negotiate trade-offs in nominal encoding resources (i.e., color hues and shapes). We conclude with a discussion of areas where future research is needed to determine how inconsistent encodings affect interpretation and how to quantify the loss in perceptual discriminability in a single view.

2. RELATED WORK

Most existing work on principles for evaluation, whether proposing value-driven approaches [25] or visualization-specific heuristics [27, 35] assume a single visualization and do not explicitly consider visualization sets.

Automatic visualization design has used evaluative metrics such as Mackinlay's *expressiveness* and *effectiveness* criteria in an automated generation system [16]. Expressiveness violations resemble hard constraints. Showing nominal data using a length encoding, for example, is not clearly more or less confusing than showing nominal data with an area encoding; both are undesirable. Effectiveness is better represented as a ranking function for encodings based on the accuracy with which viewers can decode the information. More recently, work in visualization recommender systems has approached automated recommendation of sets of views of relational data with the aim of including views that vary both the depicted data and the encodings used [34, 33]. Some design choices made by these systems aim to ease cognitive processing of multiple charts, such as VoyagerâĂŹs use of consistent colors for variables (which satisfies our constraints **C1.5** and **C2.3**). However, our work provides explicit mechanisms and more detailed requirements for ensuring encoding consistency across the set, which has not been a focus in the prior work.

Work on coordinated and multiple view (CMV) systems for exploratory analysis assumes an analyst viewing a set of visualizations at once (e.g,. [20]). In a classic CMV paradigm, the analyst manually generates the views. Instead of consistency between views, the focus is usually on enabling interactive coordination between views, such as brushing and linking where highlighting marks across two views is the common strategy for drawing attention to their shared identity. Work in narrative visualization sequence has proposed a model of the cognitive cost between pairs of views in a set, but for the purpose of ordering views rather than ensuring consistency [13].

Small multiples are the clearest example where maintaining identical encodings across views is used, in this case to allow the viewer to compare the views [28]. We believe that encoding consistency can be helpful both for supporting comparison and reducing semantic confusion beyond small multiples settings as well.

We take inspiration from Kosslyn [15], who proposes that evaluating the semantic clarity of a graph includes assessing: 1) whether elements represent the meaning of the viewer's preferred representation (representativeness), 2) whether the appearance of the marks is compatible with their meanings (congruence), 3) whether the design aligns with graphical conventions (schema availability), 4) whether every meaningful difference in the value of a variable is mapped to a detectable difference in marks (perceptual discriminability), and 5) whether every mark has only one meaning (between-level mapping principle). Kosslyn advises applying the framework to multi-chart composite visualizations first by analyzing each chart separately, then by analyzing the composite view. However, few examples of such analysis appear in his or others' work. More recently, Kindlmann proposed the principles of Invariance and Unambiguity [14]. These two principles align with our two high level principles that the same data should be presented in the same way and that different data should be presented differently. The difference is that Kindlmann's algebraic process model was a more theoretical contribution demonstrated as a way for a human designer to critique and improve a single visualization in her mind, whereas our high-level principles and specific consistency constraints are stated from a domain-specific-language (e.g., Vega) perspective for sets of visualizations and may be automated by a design or analysis tool.

3. METHODS

To develop our approach, we first examined real-world visualization sets from various sources including The Guardian, New York Times, Financial Times, and Tableau Public. We applied a variety of classic design guidelines (e.g., [16, 29, 31]), observing where threats to semantic interpretation of the data that we noticed were not characterized by these guidelines.

To further understand specific forms of conflicts, we prototyped a testbed for generating configurations of views with varying properties to test our approach. We used declarative visual specification grammars Vega [22, 4] and Vega-Lite [34, 3] because these languages make it easy to enumerate the visualization design space for a given data set.

4. COMPARISON SET

Our approach assumes a comparison set in applying our evaluative constraints: a set of visualizations in which all visualizations are supposed to be compared with each other. For instance, Figs. 1, 5, 6 and 7 are comparison sets.

A comparison can be generated in various ways depending upon the application. A comparison set could be manually constructed by a designer for a narrative visualization (e.g., [13]). Or, a comparison set could be automatically generated by a visualization recommender like Voyager [34]. A set of visualizations may contain multiple comparison sets. For example, a designer of a multi-tab interactive visualization may define views under each tab as a comparison set, within which consistency should be evaluated.

5. TWO HIGH-LEVEL CONSTRAINTS

We propose two high level constraints for evaluating visualization set consistency:

C1: The same data field is encoded the same way.

C2: Different data fields are encoded differently.

C2 implies that the field differences between individual visualizations should be visually perceivable. **C1** implies that the viewer's attention is not drawn to visual changes given a constant data field across views – in other words, the configuration of views should not create unnecessary "noise" for the perception system. Assuming **C1**, **C2** aims to ensure that any field that does change between visualizations is likely to draw the viewer's attention. In devising these principles, we informally tested their usefulness in detecting confusing encodings in case studies involving real-world visualization set examples. With these examples we derived more detailed constraints (**C1.*** and **C2.***) in Sec. 6.

5.1 Defining "the Same Field"

Visualization sets can include different transformations of the same underlying data variables across views. For example Fig. 2.1 plots tuples representing car models by the data variables (`Horsepower`, `Miles_per_Gallon`). Fig. 2.2 shows the same variables with aggregation applied to the tuples (`MEAN(Horsepower)`, `MEAN(Miles_per_Gallon)`). We use the term "field" to refer to the subset of data presented in a view, which can be a function on data variables representing measures (outcome variables) and dimensions (independent variables). Our high level constraints require a way of inferring whether the same underlying data variable with different transformations should still be interpreted as the "same field" in multiple visualizations.

Ideally, a system determines whether two fields are "the same" or "different" by assessing the underlying *semantics* of each. For example, it is relatively easy to automatically identify transformations that derive from the same (raw) data variable and classify these as the same as the non-transformed variable. For example, `BIN(Horsepower)` and `MEAN(Horsepower)` both derive from `Horsepower` and thus should be classified as "same field" for our high-level constraints **C1** and **C2**. **C1** will then surface a potential conflict when `BIN(Horsepower)` and `MEAN(Horsepower)` are encoded using different scales.

For transformations that involve combinations of multiple different variables (including dimensions or measures and/or constants), it becomes more challenging to infer when two fields are semantically similar enough to count as the same. For example, if two fields show the same measure (e.g., total sales) but with transformations applied (e.g., total sales versus total sales for coffee only, adjusted for inflation), whether they should be considered the same fields depends on the greater context of the data and presentation scenario. Data "semantics", including determining the relationships between different variables and inferring when it is desirable to compare them across plots, remains a challenging and underexplored area in information visualization. However, we believe that it is possible to develop reasonable defaults that could be used to surface conflicts which a human user could then assess using their understanding of the context.

6. VISUAL ENCODING CONSTRAINTS

We propose a set of specific constraints that operationalize the two high level constraints for common encodings. **Field-channel mappings constraints** ensure that the same fields are mapped to the same channels across the views. These constraints detect conflicts in channel pairs by evaluating *what* fields are mapped to *what* channels. **Quantitative and nominal encoding constraints** are channel-specific constraints that detect conflicts between *how* a field is mapped to a channel across a set of views. These constraints can be hard, such that a conflict is believed to be always likely to cause confusion, and is therefore important to surface to a designer. Or a constraint can be soft, such that consistency is desirable for better cognitive efficiency but not achieving consistency may not cause error is lower priority for surfacing. Conflicts between constraints are identified by these two levels of processing, and the constraints processed in order from hard to soft. **Trade-off negotiation mechanisms** are used to determine the severity of a conflict for

Table 1: Data field types and their visual encoding channels. The channels are ranked (from top to bottom) by their effectiveness for encoding the specific type of data, based on [16, 8].

Quantitative	Ordinal	Nominal
x, y	x, y	x, y
size	color.quantitative	color.nominal
color.quantitative	size	shape

either form of constraint (Sec. 7). We describe terminology, then outline field-channel mappings and encoding specific constraints.

6.1 Terminology

To facilitate discussion we use Vega-Lite's definition of *channel* as a visual encoding resource that can be allocated to data fields [3]. Given a channel, a *scale* is a specific mapping function from a data *domain* (e.g., [0, 25], ['USA', 'Europe', 'Japan']) to a visual *range* (e.g., 0-100 pixels, [red, green, blue]). Hence all properties of how a scale is presented (e.g., number of axis ticks, axis label, etc) are considered part of a scale. We consider how quantitative, ordinal and nominal fields are encoded through five commonly used visual encoding channels [3]: `x`, `y`, `color`, `size`, and `shape` (see table 6.1). Note that `x` and `y` channels can encode all types of fields and are also the most effective [16, 8]. Symbol `size` typically only encodes quantitative data. Symbol `shape` most commonly encodes nominal data. Interestingly, `color` can vary in hue, lightness and saturation to encode all three data types. Therefore we split `color` into two channels: `color.nominal` and `color.quantitative`. `color.nominal` is perceived as unordered and can encode nominal data. `color.quantitative` is perceived as ordered and can encode ordinal and quantitative data.

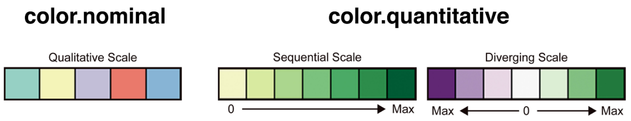

Figure 3: Two color channels: `color.nominal` and `color.quantitative`. `color.nominal` is perceived as unordered and `color.quantitative` is perceived as ordered. Diverging schemes are reserved for data with a meaningful midpoint. Source: [5].

6.2 Field-Channel Mappings Constraints

Each individual visualization contains at least one field-channel mapping. For example, the same field may appear as `size` in one view and `color.quantitative` in another. In comparing visualizations, higher level inconsistencies arise from differences in field-channel mappings. Prior to comparing pairs of encodings to channel-specific constraints, our approach is designed to detect field-channel mapping differences.

For any two visual encoding channels in two visualizations, there are three types of differences between field-channel mappings: *swap*, *shift* and *update*. These differences are possible between any two channels that can encode the same type of fields, namely: `x` and `y`, `color.quantitative` and `size`, `color.nominal` and `shape`, `x` and `color`, `x` and `size`, `x` and `shape`, `y` and `color`, `y` and `size`, and `y` and `shape`. Figure 4 illustrates swap, shift and update for the first three channel pairs. A *swap* means two channel pairs encode the same two fields but have swapped the field-channel mappings. In other words, in the first pair `channel1` encodes field `A` and `channel2` encodes field `B`; in the second pair `channel1` encodes field `B` and `channel2` encodes field `A`. A *shift* means two channel pairs have only one field in common, but that common field is not encoded by the same channel. For example, the first channel pair encodes field `A` and `B`, the second channel pair encodes field `A` and `C`; in the first pair, `A` is encoded by `channel1`; in the second pair, `A` is encoded by `channel2`. If two channel pairs only have one field in common and that field is encoded by the same channel, we call it an *update* between two channel pairs. If two channel pairs have no common fields, we also call it an *update*. From our experience, a *swap* or a *shift* between two channel pairs are problematic for global consistency because they encode the same data fields in different ways (in this case, different channels). An *update* situation encodes the same fields with same channels and different fields with different channels and thus does not hurt consistency at the field-channel mapping level. However, an *update* situation can still have lower level inconsistencies when we consider specifically *how* a field is encoded in a channel (see Sec. 6.3 and Sec. 6.4 for channel-specific constraints).

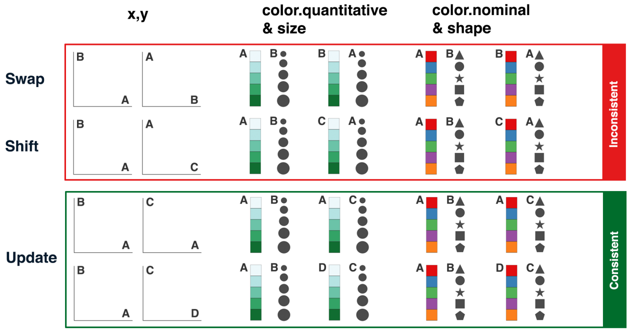

Figure 4: Field-channel mappings swap, shift and update for `x` and `y`, `color.quantitative` and `size`, and `color.nominal` and `shape` channel pairs.

To summarize, inconsistencies raising from field-channel mappings can be prohibited by the following two constraints:

C1.1 There should be no swap between two field-channel mappings.

C1.2 There should be no shift between two field-channel mappings.

Inconsistencies between field-channel mappings can be resolved by having the viewer reading the axes titles and legend, but maintaining awareness of how a field is mapped to different channels across views increases cognitive load. Additionally, like other constraint violations, field-channel mapping violations work against view comparability.

6.3 Quantitative Encoding Constraints

A quantitative field can be encoded by `x`, `y`, `size`, or `color.quantitative` channel. We describe encoding-specific constraints for quantitative fields.

6.3.1 *x / y*

In light of **C1** and **C2**, we derive:

C1.3 If two x (or y) channels encode the same field, the scales should be the same.

In this case, all scale properties should be the same: scale `domain` and `range`, as well as aspects of the scale depiction, such as axis ticks and labels and so on. Detecting conflicts with this constraint can help a human designer better achieve cognitive efficiency and avoid tasking the viewer with unnecessary effort to determine the differences between views, as in Fig. 2.1 and Fig. 2.2, where the `x` channels have inconsistent scales.

C2.1 If two x (or y) channels encode different fields, the scales should convey the difference.

We do not constrain visualizations sets against re-using `x` and `y` for encoding different fields because this quickly eliminates the two most effective channels for all data types. Additionally, we believe that reuse of `x` and `y` in sets of visualizations is a type of inconsistency between views that most viewers are accustomed to by convention. When two `x` channels encode different fields, viewers scan the axis to learn that they are different fields. Constraint **C2.1** should be naturally satisfied in most cases because it's unlikely for two different fields (e.g., `Horsepower` and `Miles_per_Gallon`) to have identical axes.

6.3.2 *size*

Size is typically used to encode quantitative fields and is often broken down to discrete steps due to Weber's law and the phenomena of just noticeable difference [10, 26, 31]. We propose that:

C1.4 The same field should have a consistent size scale in a comparison set

For example, the Human Development Trends slide show [11] keeps a consistent size scale for population throughout the presentation. Between Fig. 5.4 and 5.5, continent population and country population use the same scale. This helps viewers make more accurate judgements of the populations and also allows viewers to quickly find a specific country (e.g., China) through size given prior knowledge about the relative size of the country's population.

Where possible, we also propose:

C2.2 (Soft Constraint) A visualization set should not map more than one field to the size channel

Like `x`, `y` channels, we believe `size` can be reused for encoding different fields in some visualization sets without causing too much confusion for views based on convention. However, viewers have to consult legends to learn that `size` has different meanings in different views. Therefore we propose a soft constraint **C2.2** to discourage `size` reuse.

In cases where channel reusing becomes unavoidable, the field differences should be made readily apparent. The most obvious way to explicitly state field differences is through size legends. Additionally, if redundant coding [31] is possible (when a visualization set has some unassigned nominal encoding resources like color hues and symbol shapes, described further in Sec. 7.2), we could assign different color hues to the different fields, so that the field differences become more perceptually apparent.

6.3.3 *color.quantitative*

To encode quantitative data, the `color.quantitative` channel can vary hue, lightness and saturation to form either sequential or diverging color scales [6] (see Fig. 3. Diverging scales are used to encode data with a meaningful midpoint. In ordered color scales, it is common to see people associating darker colors with larger data values and lighter colors with smaller data values.

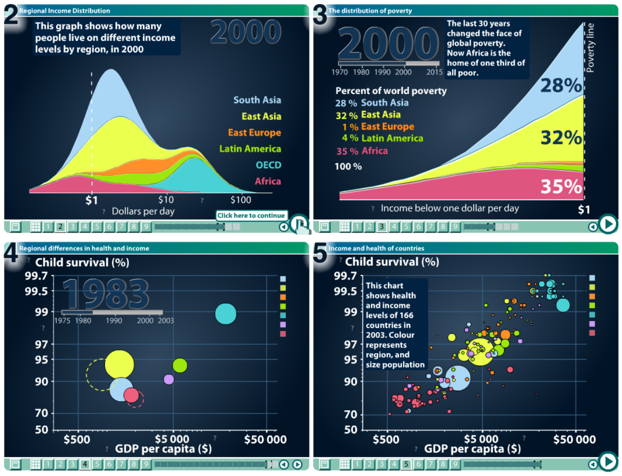
2
Regional Income Distribution
This graph shows how many people live on different income levels by region, in 2000
2000
South Asia
East Asia
East Europe
Latin America
OECD
Africa
$1
$10
$100
Dollars per day
Click here to continue
3
The distribution of poverty
2000
1970 1980 1990 2000 2015
The last 30 years changed the face of global poverty. Now Africa is the home of one third of all poor.
Poverty line
Percent of world poverty
28 % South Asia
32 % East Asia
1 % East Europe
4 % Latin America
35 % Africa
100 %
28%
32%
35%
Income below one dollar per day
$1
4
Regional differences in health and income
Child survival (%)
99.7
99.5
99
97
95
90
70
50
1983
1975 1980 1990 2000 2003
$500
$5 000
$50 000
GDP per capita ($)
5
Income and health of countries
Child survival (%)
This chart shows health and income levels of 166 countries in 2003. Colour represents region, and size population
99.7
99.5
99
97
95
90
70
50
$500
$5 000
$50 000
GDP per capita ($)

Figure 5: Screenshots of Human Development Trends, 2005 [11]. This classic interactive slide show presents global economic and health growth from 1970 to 2015. It contains field transformations such as calculate percentages (3), data drill down (5), and filter by time (3 and 4). Gapminder keeps a consistent size scale to show continent population as well as country population. It also keeps consistent mappings between continents and nominal colors.

Assuming each single visualization follows single visualization color design guidelines, we propose two constraints for consistent use of ordered color scales across a comparison set:

C1.5 If two visualizations use color to encode the same quantitative field, they should use the same color scale.

Specifically, the two ordered color scales should have the same scale `domain`, mapped to the same color scale `range`, and have the same color breaks. For example, in The Guardian "Wealth and Poverty in Africa" comparison set (Fig. 6), views 1-4 represent `population` percentage through a yellow to purple color scale. The only difference between the views is that each represents a different subset of income levels. However, in view 1, the color scale maps from $[0, 82.1\%]$ to the $[yellow, purple]$, whereas in views 2-4 the scales map from $[0, 25\%]$ to the same color range. This violation of **C1.5** makes it virtually impossible to compare the first view with the other views in the set. The visual similarity across all views may lead viewers to assume that views 2-4 apply the same scale as view 1, leading to interpretation errors.

C2.3 If two quantitative color channels encode different quantitative / ordinal fields, the two channels should have different color scales.

Continuing with the same example, Fig. 6 views 5 and 6 encode different fields (`Gini index` and `GDP per capita`) on the same color ranges as views 1-4. In addition to fixing the color scale applied to all `population%` views, we propose that the comparison set would be easier to process if views

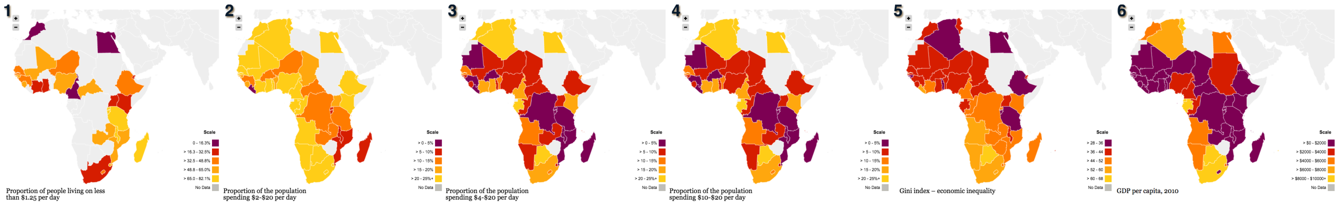

Figure 6: Wealth and poverty in Africa, The Guardian [19]. Note that views 1-4 use inconsistent color scales to encode the same field `population%`. Views 5-6 encode different fields using the same color scale.

5 and 6 mapped their distinct fields to distinct color scales (e.g., $[lightblue, darkblue]$, $[pink, red]$) that do not overlap with the scale in views 1-4.

6.4 Nominal Encoding Constraints

6.4.1 *color.nominal*

Color can vary in hue, lightness and saturation to create a qualitative scale that is suitable for encoding nominal data.

In light of **C1**, we derive:

C1.6 If two nominal color channels encode the same nominal field, the mappings from the nominal data values to the nominal colors should be identical

For example, the Gapminder comparison set (Fig. 5) consistently uses red for Africa, cyan for OECD, orange for East Europe and so on.

Conversely, in light of **C2**, we derive:

C2.4 If two nominal color channels encode different nominal fields, the same color should not be reused for different nominal values

When this constraint is violated, such as in Fig. 1 where color scales with many overlapping colors encode distinct fields, viewers will experience a higher cognitive load due to the need to "unlearn" a set of mappings every time they transition between the views.

C2.5 If a nominal color channel encodes a nominal field and a quantitative channel encodes a quantitative / ordinal field, the color semantics established by the two channels should not contradict

For example, Fig.7 1 establishes nominal color semantics for `race`: white people is green, black is blue, Hispanic is yellow and Asian is red. The ordered color channels in views 2-5 encode `population` filtered by `race`. These channels could all have used a gray scale to encode `population`, but the author decided to honor the nominal `race-color` mappings established in view 1 so he carefully picked the meaningful colors for each scale. By doing so, he not only made views 2-5 more distinct but also associated them better with view 1. Unfortunately, the color semantics are challenged by view 6, which inconsistently reuses the same blue for foreign-born population. A distinct hue would better represent the independence of `black` and `foreign-born` that is otherwise implied.

6.4.2 *shape*

Shape typically encodes nominal fields. The constraints we derive for shape are similar to that of `color.nominal`:

C1.7 If two shape channels encode the same nominal field, they should maintain the same mappings from nominal values to symbol shapes

C2.6 If two shape channels encode different nominal fields, they should use distinct symbol shapes.

Like color hue constraint violations, violating shape constraints is likely to increase cognitive processing and lead to confusion.

7. GLOBAL VS. LOCAL TRADE-OFFS

Our constraint-based approach is intended to allow automated detection of potential global inconsistencies. Surfacing potential conflicts can alert a designer to reconcile them if desired, or can be used as input to a revision stage in a fully automated scenario. A naive reconciliation approach is to detect and try to resolve *all* conflicts for optimal global consistency. For example, a "set evaluator assistant" built into a tool like Tableau or Lyra could point out every conflict in a finished set of visualizations to a designer, implying that he or she should fix them all. Or an automated visualization recommender could try to resolve all conflicts by directly revising the visualization set. However, while the naive approach may successfully achieve *global* consistency, is likely to suggest revisions that can significantly compromise *local* effectiveness. In the case of a design assistant in a visualization creation tool, surfacing all possible conflicts is also likely annoy or overwhelm the user. Ordering constraints a priori by their anticipated consequence on interpretation could allow ranking (and triaging) of suggested revisions. Our process proceeds by first considering hard constraints, then soft constraints. However, the severity of a detected conflict may not be predictable from the constraint alone (e.g., it may depend on the data and the space for presentation). We therefore propose mechanisms for negotiating global-local tradeoffs for quantitative and qualitative encodings.

7.1 Trade-Offs in Quantitative Comparisons: Effectiveness Preservation Score

Our negotiation mechanism for quantitative encoding violations is based on an assumption that an effective single visualization maximizes the accuracy with which a viewer can compare data values. A classic example is Edward Tufte's advocation for high data-ink ratio, high data density designs [29]. A common design practice for achieving maximal accuracy in decoding data is to set the *scale domain* of any rendered encoidng to be large enough to encode all values in the *data domain* but not much larger (e.g., adding only slight padding to ensure readability of extreme values in mapping a variable like horsepower to an x-axis). We operationalize this assumption and use it to quantify the impacts to each single visualization for which a consistency conflict is detected. This quantification is used to prioritize which detected conflicts to surface, either for resolving by a human designer or fully automated revisions.

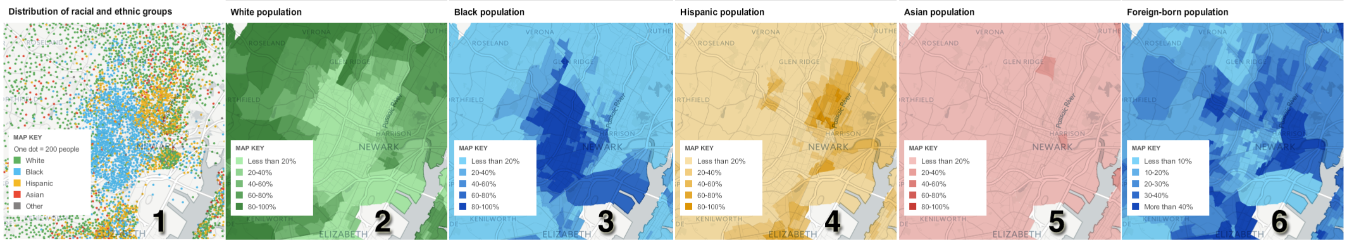

Figure 7: Cropped screenshots of New York Times "Mapping America: Every City, Every Block" [17]. All views show population percentage. For views 1-5, white people = green, black people = blue, Hispanic = yellow, Asian = red. For view 6, foreigners = blue. View 6 has a different `domain` (`[0, 40%+]`) than views 2-5.

Consider the conflicts in `x` and `y` scales between Fig. 2.1 and Fig. 2.2. To make the views consistent, one potential revision is to set the domains of the `x` and `y` position encodings in view 2 to the domains of `x` and `y` in view 1 (e.g., Fig. 2.2b). Or one could do the reverse: set the domains of the `x` and `y` encodings in view 1 to the domains of `x` and `y` in view 2. However, the latter revision results in view 1 no longer representing all values in its original domain. In applying our trade-off negotiation approach to quantify the impact of conflicts, for any detected encoding conflict we therefore only consider revisions to the scale with a smaller domain. However in practice, there may be cases where a designer prefers to lose data visibility in some views to achieve consistency.

Specifically, we devise a measure that we call the *effectiveness preservation score* to operationalize the intuition that a single visualization may not want to apply a consistent quantitative encoding if the loss in ability to make data comparisons in the revised view is large. For each local revision proposed by a constraint, we define its *effectiveness preservation (EP) score* as:

$$EP = \frac{\#\ Comparison_{revised}}{\#\ Comparison_{original}}$$

where
$\#\ Comparison_{revised}$ is the number of possible pairwise visual comparisons supported by the *revised* visualization, and $\#\ Comparison_{original}$ is the number of possible pairwise visual comparisons supported by the *original* visualization (without applying the proposed revision).

EP is a ratio between 0 and 1. If a EP is close to 1, it means the proposed revision preserves local effectiveness (operationalized as number of pairwise comparisons) relatively well and should be accepted. If a EP is close to 0, it means the proposed revision will severely impact local effectiveness and should be rejected. Users can define a threshold for EP to specify how much compromise in local effectiveness is acceptable for a detected conflict to be surfaced. For Fig. 2, the EP score would identify that no pairwise comparisons in view 2 are lost by revising view 2 for consistency with view 1's scales (Fig. 2.2b). Hence, this conflict would be surfaced for a human designer in a design assistant scenario, optionally with the suggestion for revision, or automatically done in a fully automated visualization recommender system.

To calculate the number of possible visual comparisons in a visualization requires a model that can predict the discriminability of different configurations of marks and attributes. For example, to determine the number of comparisons supported by a particular rendered y position encoding of a quantitative variable in a scatterplot requires predicting when the occlusion of rendered points will precludes comparing their precise values. An important avenue for future work toward negotiating between evaluative criteria for single and sets of visualizations is to build better *perceptual kernels* [9], distance matrices representing perceptual differences between and within visual variables.

In cases where the best solution is to break consistency, various strategies can be used to eliminate potential confusion on the part of the viewer, such as presenting the conflicting view using a consistent encoding initially, then animating a transition to the conflicting encoding to bring the user's attention to the difference. Or, textually annotating the conflicts across multiple views can warn the viewer to carefully read axes guides and legends. In some situations, log scales are an alternative way to avoid a conflict while still allowing comparability of values in both views.

7.2 Trade-Offs in Nominal Comparisons: Palette Allocation

Conflicts between nominal encodings can be thought of as "harder" constraints than those between quantitative encodings, due to the preattentive nature of assumptions that two identical hues (e.g., red points across two scatterplots) should depict the same underlying data. We propose several additional mechanisms to help prioritize nominal conflicts for surfacing, and suggests revisions to negotiate conflicts.

In general, when different nominal data fields in a comparison set, unused hues and/or symbol shapes (geometric shapes, ISOTypes, texts), can be put to use to maximize symbol distinctiveness. When overlapping values are detected between multiple rendered nominal palettes (i.e., hues in a categorical color scale, or shapes being used to encode different data), a tool could allocate preferred nominal encodings (e.g., color hue over shape, and the most discriminable categorical hue palettes) for the nominal fields that appear most frequently across the set of views, applying less optimal nominal encodings for fields that appear less. The development of larger palette sets (e.g., the addition of more shapes, such as icons chosen for the particular dataset [24]) can support allocation while still avoiding conflicts.

Another strategy treats the hues across a set of sequential or diverging color palettes used to encode fields in differing views as a color palette, detecting repeated hues used to show different fields. In such cases, an automated revision assistant could surface the conflict to a designer to fix, or even suggest different hues to encode different quantitative fields across a set of visualizations (e.g., Fig. 7).

7.3 Task Dependency

We note that the optimal use of the EP score and palette allocation techniques, as well as other aspects of our approach, can be task dependent. For example, consider two simple visualizations depicting population data in the US for different demographic groups for two different years. The first visualization is a one dimensional scatterplot where each mark represents different education level groupings in 1980. Population is encoded redundantly using both `size` and `x` position. The domain of the `x` position encoding is [0, 72 million], and the range is [0, 200px]. The second visualization is identical to the first, except that marks represent the same education levels but in the year 2016. In visualization 2, the domain of the `x` position encoding is therefore different ([0, 140 million]), but the range is the same: [0, 200px].

The question of whether x is encoded consistently depends on the viewer's goal. If the goal is to get a better sense of the ratios of the population with different education levels at the two time points, then the lack of consistency in `x` position is not necessarily problematic. However, if the goal is to get a better sense of the increase in absolute population across the two years, the inconsistency bars easy mental integration of information across the two views.

8. IMPLEMENTATION

Our approach to visualization set evaluation and revision can be thought of as a pipeline with two primary steps: 1) detecting conflicts, and 2) deciding whether a conflict is worth surfacing for a human author or automated revision by assessing the loss of effectiveness for each possible revision. These steps can be implemented in various ways.

One approach is to encode definitions of field-channel mapping and encoding-specific constraints in a constraint solver, which searches a pre-determined design space of channels and encodings for states that reduce conflicts. Another approach is to model the set as a graph with two types of edges, representing 1) when visualizations have the same field and 1) when they use an identical encoding. Patterns of edges representing conflicts can be identified. Future work should determine how feasible it is for either approach to find a satisfactory sets of designs given different configurations of views.

9. DISCUSSION

Our work represents a first step toward devising an evaluative approach that integrates considerations similar to APT's expressiveness and effectiveness but for sets of visualizations. Our approach is designed to detect conflicts that can make a set of visualizations likely to produce interpretation errors. We believe that this goal is especially important for visualizations intended for casual users. Some conflicts may allow for clear-cut suggestions from an automated design assistant on how to potentially revise the encodings (e.g., use consistent hues across nominal and filtered quantitative views). However, we believe that whereever possible, human input is the best way to identify the most appropriate design revisions to address a conflict in a given context. Understanding when conflicts result in erroneous spontaneous interpretations, how to determine discriminability for different encodings, and how to account for contextual effects are important topics for future work.

9.1 Determining spontaneous interpretations

Visualization interpretation is driven by both top-down (e.g., a priori interests) and bottom-up (perceptual, Gestalt) factors [7]. We believe a constraint-based framework for evaluating visualization sets, such as we have presented, can help designers and visualization recommenders avoid unnecessarily increasing the cognitive load of the viewer. However, to apply our framework in visualization construction or recommender systems may require interpretation experiments to develop specific enough knowledge on how encodings and conjunctions of encodings are likely to be interpreted. Consider two scatterplots depicting two distinct sets of `x` and `y` variables but using the same marks (e.g., circles), the same default mark hue (e.g., blue) and the same palette to encode a distinct third quantitative variable as size in each view. Will the combined encoding similarities produce confusion despite the common reuse of `x`, `y`, and `size` to depict data? The perceptual literature on the separability of encoding conjunctions can help, but further work may be required to build more specific predictive models.

9.2 Determining discriminability

Additional work is needed to inform the perceptual models on discriminability used to calculate the effectiveness preservation scores in our approach. Our work aligns with [14, 15] which stated that more significant changes in data should lead to more noticeable changes in the visual impression and vice versa. We believe that the approach of [9], which builds crowdsourced "perceptual kernels" that summarize perceptual differences between and within visual variables, is promising.

9.3 Contextual effects

Our approach is designed to detect conflicts between pairs of views and determine the severity of particular conflicts. It is possible that the degree to which conflicts cause confusion for viewers is a function of the number of views in which an encoding is consistent versus those in which it is not. For example, consider the interactive slideshow excerpted in Fig. 6, in which 4 views use the same yellow to purple color encoding to show the percent of population with a given income level, but 3 of these views using the encoding consistently and one using it inconsistently.

10. CONCLUSION

We presented a technique for evaluating the consistency of visualization sets. Our approach uses two high level constraints: Don't show the same data in different ways, and Don't show different data in the same way, to detect of constraints across specific encoding pairs in a set of views. We present an initial formulation of specific constraints plus mechanisms to prioritize conflicts for surfacing and/or automated revision while monitoring loss in local (single) visualization effectiveness. We describe areas for future work toward making automated visualization set evaluation a feasible approach for information visualization tools.

11. ACKNOWLEDGMENTS

We thank Kanit Wongsuphasawat for his comments in improving this paper.

12. REFERENCES

[1] Auto mpg data set. UCI Machine Learning Repository https://archive.ics.uci.edu/ml/datasets/Auto+MPG. Accessed: 2016-06-11.

[2] Tableau public. https://public.tableau.com/s/.

[3] Vega-lite visualization grammar. https://vega.github.io/vega-lite/.

[4] Vega visualization grammar. https://vega.github.io/vega/.

[5] C. Brewer. Colorbrewer: Color advice for cartography. http://colorbrewer2.org/#, 2009.

[6] C. A. Brewer. Color use guidelines for mapping and visualization. *Visualization in Modern Cartography*, 2:123–148, 1994.

[7] P. A. Carpenter and P. Shah. A model of the perceptual and conceptual processes in graph comprehension. *Journal of Experimental Psychology: Applied*, 4(2):75, 1998.

[8] W. S. Cleveland and R. McGill. Graphical perception: Theory, experimentation, and application to the development of graphical methods. *Journal of the American Statistical Association*, 79(387):531–554, 1984.

[9] C. D. Demiralp, M. S. Bernstein, and J. Heer. Learning perceptual kernels for visualization design. *IEEE TVCG*, 20(12):1933–1942, 2014.

[10] G. T. Fechner. Elements of psychophysics, 1860. 1948.

[11] Gapminder. Human development trends, 2005. https://www.gapminder.org/downloads/human-development-trends-2005/. Accessed: 2016-06-12.

[12] J. Heer and G. G. Robertson. Animated transitions in statistical data graphics. *IEEE TVCG*, 13(6):1240–1247, 2007.

[13] 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.

[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] S. C. Matthew Bloch and A. McLean. Mapping america: Every city, every block. New York Times http://www.nytimes.com/projects/census/2010/explorer.html. Accessed: 2016-06-11.

[18] D. B. Perry, B. Howe, A. M. Key, and C. Aragon. Vizdeck: Streamlining exploratory visual analytics of scientific data. 2013.

[19] C. Provost and the Guardian Interactive team. Wealth and poverty in africa. The Guardian http://www.theguardian.com/global-development/interactive/2011/dec/25/wealth-poverty-africa-interactive. Accessed: 2016-06-13.

[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*, pages 61–71. IEEE, 2007.

[21] A. Satyanarayan and J. Heer. Lyra: An interactive visualization design environment. In *Computer Graphics Forum*, volume 33, pages 351–360. Wiley Online Library, 2014.

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

[23] E. Segel and J. Heer. Narrative visualization: Telling stories with data. *IEEE TVCG*, 16(6):1139–1148, 2010.

[24] V. Setlur and J. D. Mackinlay. Automatic generation of semantic icon encodings for visualizations. In *SIGCHI*, pages 541–550. ACM, 2014.

[25] J. Stasko. Value-driven evaluation of visualizations. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, pages 46–53. ACM, 2014.

[26] S. S. Stevens. On the psychophysical law. *Psychological Review*, 64(3):153, 1957.

[27] A. Tarrell, A. Fruhling, R. Borgo, C. Forsell, G. Grinstein, and J. Scholtz. Toward visualization-specific heuristic evaluation. In *Proceedings of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, pages 110–117. ACM, 2014.

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

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

[30] S. van den Elzen and J. J. van Wijk. Small multiples, large singles: A new approach for visual data exploration. In *Computer Graphics Forum*, volume 32, pages 191–200. Wiley Online Library, 2013.

[31] C. Ware. *Information visualization: perception for design*. Elsevier, 2012.

[32] M. Wertheimer. Laws of organization in perceptual forms. 1938.

[33] K. Wongsuphasawat, D. Moritz, A. Anand, J. Mackinlay, B. Howe, and J. Heer. Towards a general-purpose query language for visualization recommendation. In *Proceedings of the Workshop on Human-In-the-Loop Data Analytics*, page 4. ACM, 2016.

[34] 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.

[35] T. Zuk, L. Schlesier, P. Neumann, M. S. Hancock, and S. Carpendale. Heuristics for information visualization evaluation. In *Proceedings of the 2006 AVI workshop on Beyond Time and Errors: Novel EvaLuation Methods for Information Visualization*, pages 1–6. ACM, 2006.