Erschienen in: SCHMIDT, Albrecht, ed., Kaisa VÄÄNÄNEN, ed., Tesh GOYAL, ed.. Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems. New York, NY: ACM, 2023, 823. ISBN 978-1-4503-9421-5
https://doi.org/10.1145/3544548.3580734

Interactive Context-Preserving Color Highlighting for Multiclass Scatterplots

Kecheng Lu
lukecheng0407@gmail.com
Shandong University
China

Khairi Reda
redak@iu.edu
Indiana University-Purdue University Indianapolis
United States

Oliver Deussen
oliver.deussen@uni-konstanz.de
University of Konstanz
Germany

Yunhai Wang*
cloudseawang@gmail.com
Shandong University
China

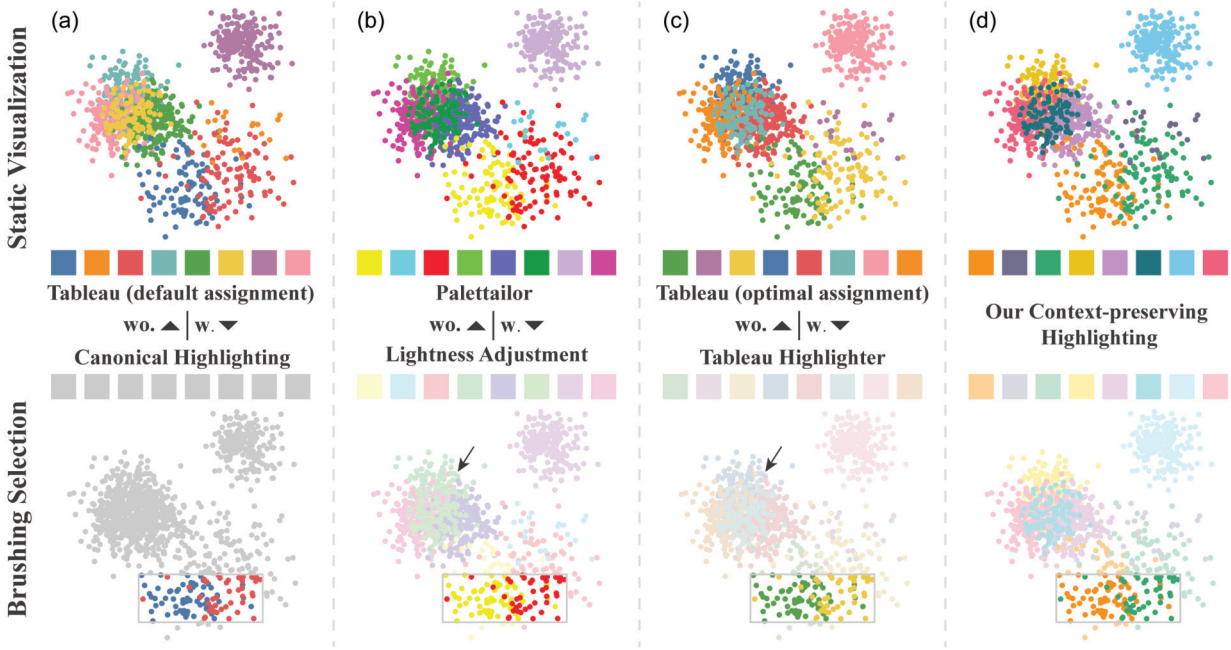

Figure 1: Results for applying different color-based highlighting methods for highlighting points in a multi-class scatterplot. (a) (top) colorization with Tableau palette and default assignment; (bottom) highlighting effect achieved by assigning a grey color to all non-selected data points; (b) (top) result for a Palettailor-generated palette [21]; (bottom) highlighting achieved by increasing lightness of non-selected data points; (c) (top) colorization by Tableau palette and optimal assignment; (bottom) highlighting by applying Tableau Highlighter function; (d) (top) colorization by our method using a salient color palette; (bottom) highlighting result by combining salient and faint color mapping schemes. Problematic areas are indicated by arrows. Our method allows highlighting a subset of data points while maintaining the discriminability of all non-selected points as well as the color consistency of all color pairs.

*corresponding author

ABSTRACT

Color is one of the main visual channels used for highlighting elements of interest in visualization. However, in multi-class scatterplots, color highlighting often comes at the expense of degraded color discriminability. In this paper, we argue for *context-preserving highlighting* during the interactive exploration of multi-class scatterplots to achieve desired pop-out effects, while maintaining good perceptual separability among all classes and consistent color mapping schemes under varying points of interest. We do this by first generating two contrastive color mapping schemes with large and

small contrasts to the background. Both schemes maintain good perceptual separability among all classes and ensure that when colors from the two palettes are assigned to the same class, they have a high color consistency in color names. We then interactively combine these two schemes to create a dynamic color mapping for highlighting different points of interest. We demonstrate the effectiveness through crowd-sourced experiments and case studies.

KEYWORDS

Color Palettes, Highlighting, Multi-Class Scatterplots, Discriminability

1 INTRODUCTION

Multi-class scatterplots are among the most commonly used representations for visualizing labeled quantitative data. They represent each data item with a color-coded point (or some other marks) positioned within two orthogonal dimensions, with color encoding class label (i.e., category). If used with judiciously designed color palettes, multi-class scatterplots can effectively display the distribution of classes and the relationships among them. A few automated colorization methods [21, 42] have been proposed for maximizing class discriminability in scatterplots while maintaining their aesthetic appeal. Typically, all classes in a visualization are given equal emphasis (see the first row in Fig. 1). This type of color assignment is sufficient for a static visualization.

For interactive visualizations, however, it is desirable to emphasize certain subsets of the data on-demand, for example, to allow the user to select or brush points of interest. The canonical strategy to support this interaction is to dim colors for the non-selected data by assigning a neutral grey. Although effective at inducing a highlight, this approach results in a loss of context (e.g. see Fig. 1 (a)-bottom). While modulating lightness or opacity for the non-selected data items can alleviate this issue [32], it can lead to poor class separability (e.g. see the small difference between the green and light-green classes in the bottom of Fig. 1 (b)). In short, existing color-based highlighting strategies cause a temporary loss of color coding [24], which disrupts exploration or, at minimum, the user's mental map. Even state-of-the-art commercial solutions, such as Tableau Highlighter [36], struggle at producing good color highlighting while preserving context (see Fig. 1 (c)-bottom). Designing color palettes that support both, *focus and context*, is therefore crucial for interactive visualization. Yet, this important design goal is unsupported by existing colorization approaches (e.g., Palettailor [21] and Colorgorical [9]), as those approaches do not allow for dynamically varying the visual emphasis of selected data points.

We present an interactive colorization technique for the *context-preserving* highlighting of scatterplots (and for other multi-class labeled visualizations). The core of this technique is the ability to interactively emphasize data points of interest while also maintaining a good perceptual separability for all classes, and furthermore ensuring color consistency under varying degrees of emphasis. Our technique allows the user to highlight elements in a visualization while still being able to see the local neighborhood and global context of those elements. This allows for creating compelling visualizations, such as multi-view interfaces that support effective brushing-and-linking without loss of context (see supplementary video).

The proposed technique works by generating two contrastive color mapping schemes that maximize and minimize the contrast over the background, while both optimize the discriminability for a given scatterplot configuration. We create the needed emphasis by interactively combining both palettes. To achieve effective highlighting, we model the contrast to the background as well as the class-neighborhood contrast. Additionally, our technique maintains color consistency for hue, saturation, and color names between highlighted and de-emphasized colors for each class, thus ensuring minimal disruption to the user's mental model. Fig. 1 (d) shows two pre-generated palettes of our method. The bottom of (d) shows the results of interactive highlighting. For the purpose of letting the selected data points of interest stand out, our results have a similar effect to Figs. 1 (a, b, c). However, in contrast to other methods, our method ensures good class separability and color names among selected and non-selected data points.

We evaluated our approach in two crowd-sourced studies[1]. First, we compared our colorization results with state-of-the-art palettes (e.g., Tableau [37] and Palettailor [21]). The results indicate that our method achieves a comparable highlighting effect while better maintaining class separability than the benchmark methods. We also created a web-based implementation of our technique as a color-design tool[2] and demonstrate its effectiveness in a case study. To summarize, the main contributions of this paper are as follows:

- We propose an interactive context-preserving approach for generating stable color palettes for multi-class visualizations. Our approach supports an interactive emphasis on data parts while maintaining overall class discriminability and relative color consistency.
- We present a simulated annealing-based optimization for generating highlightable palettes, while ensuring sufficient contrast with the background and neighboring classes, among other perceptual constraints.
- We empirically validate our techniques in two crowd-sourced experiments and present extensions of our method to a few interactions and other multi-class visualizations such as line and bar charts, within an open-source implementation[3].

2 RELATED WORK

We divide previous works into methods related to visual highlighting and to color design for visualization.

[1] Experimental data and analysis code are included with the submission as supplemental materials and are available at https://osf.io/679pb/.
[2] https://palettailor.github.io/highlighting/
[3] https://palettailor.github.io/highlighting/demo/

2.1 Highlighting in Interactive Visualization

In interactive visualization applications, it is a common task to highlight a subset of data points for directing the user attention or apply subsequent operations to [18, 32, 38]. Emphasis effects are created by manipulating visual variables (e.g. position, size, transparency and color lightness). Naidu [25] presents a crowd-sourced study to measure the highlighting effect of color-coded scatterplots and provides recommendations for effective color highlighting.

Previous studies have systematically evaluated emphasis effects in a wider range of scenarios [10, 23, 41]. Hall et al. [11] provide a systematic review of such effects and divide them into two classes: intrinsic and extrinsic effects. The former is created by the initial visual mapping, while the latter is the result of manipulating the visual variables of an existing visualization. Although the extrinsic emphasis is effective in many cases, it may conflict with the visual encoding of the given visualization. For example, changing transparency or color lightness in a multi-class scatterplot may result in new colors (see Fig. 1 (b)) that might lead to misunderstanding color-associated semantics or to similar colors (see Fig. 1 (c)) that do not allow visual discrimination anymore. To address this issue, our approach generates stable color mapping schemes for interactively highlighting multi-class scatterplots that attempt to balance between two goals: emphasizing points of interest *and* maintaining class discrimination and color consistency.

2.2 Color Design in Visualization

For a complete review of color design techniques for visualization, we refer readers to surveys such as [40, 46]. We limit our discussion to techniques related to color design for categorical data visualization and specifically to the optimization of color mappings, color palette generation, color palettes for highlighting as well as color consistency.

Color Map Optimization. Mapping each class to a proper color selected from a given palette is particularly helpful for categorical data visualization since no given order can be used here. A few factors have been identified for guiding searches within such mappings. For example, Lin et al. [19] propose to optimize the compatibility between class semantics and assigned colors. Setlur and Stone [33] produce better results by using co-occurrence measures of color name frequencies. Reda et al. argue generally for increasing the nameability of colors in colormaps [30, 31]. Kim et al. [15] incorporate color aesthetics and contrast into the optimization of color assignment for image segments. Szafir [39] find that mark size heavily affects color discriminability. Wang et al. [42] propose to maximize class discriminability based on color-based class separability, which takes into account spatial relationships between classes and in addition the contrast with the background color. Once the assignment is done, the color of each class can be further optimized for better serving additional purposes, such as reducing the power consumption of displays [4], improving the accessibility of visualizations for visually impaired users [22], or better class discrimination [17]. Almost all these methods aim to generate effective static visualizations, whereas our goal is to generate interactive visualizations with varying subsets of interest and maximizing class discriminability as well as the similarity between the perceived colors of each class.

Color Palette Generation. To create an appropriate categorical color palette, the commonly used approach is to select one from a library of carefully designed palettes provided by online tools such as ColorBrewer [12]. Fang et al. [6] suggest maximizing the perceptual distances among a set of colors while meeting various user-defined constraints. Likewise, Nardini et al. [26] provide an automatic optimization algorithm for improving continuous colormaps in Euclidean color space and integrate them into a test suite [27]. Colorgorical [9] further allows users to customize color palettes by generating them based on user-specified discriminability levels and preferences. Recently, Palettailor [21] takes a further step by automatically generating categorical palettes for different types of charts, such as scatterplots or line and bar charts. Rather than generating one palette at a time, our work produces a pair of contrastive palettes with different contrast over the background while maintaining color consistency between corresponding color pairs. This drives the dynamic generation of palettes to interactively highlight points of interest in multi-class scatterplots.

Color Design for Highlighting. To let important classes stand out, the commonly used practice is to assign them bright colors while using subdued colors for less-important classes. This can be achieved by using accentuated color palettes, which consist of a set of subdued colors and a set of bright (stronger, darker, or more saturated) colors. However, only few such palettes are available. For example, ColorBrewer [12] provides only a very small set of such palettes. Therefore, Wilke [45] suggests creating palettes by lightening some colors of an existing palette while darkening others. This method might be able to create desired palettes but it is often hard to maintain the discriminability between all classes in the given data. In contrast, our method can automatically create such palettes and assigns them to the input data, while maintaining the stability of palettes for varying the data points of interest.

Color Consistency. Multi-view visualizations are commonly used for multivariate analysis. Although a few design guidelines [43] have been proposed for constructing multi-view visualizations, few of them are related to color design. Qu et al. [29] recommend a set of color consistency constraints across views. Among them is a high-level constraint that the same data field should always be encoded in the same way. In our work, however, the highlighted data subset varies during the exploration. To ensure a relative consistency of the perceived color, we require the highlighted and de-highlighted colors of the same class to have the same hue and similar color names.

3 BACKGROUND

Given a multi-class scatterplot with m classes and n data items $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$, each $\mathbf{x}_t$ has a label $l(\mathbf{x}_t)$ and the i-th class has n_i data points. The goal of Palettailor [21] is to find a color mapping $\tau : L \mapsto c$ that maximizes the discriminability of the given multi-class scatterplot while ensuring that all colors can be referenced by names. Since each class is assigned a unique color, a palette P with m colors is formed. Palettailor finds τ by maximizing the following objective:

$$\arg\max_{\tau} E(\tau) = \omega_0 E_{PD} + \omega_1 E_{ND} + \omega_2 E_{CD}, \tag{1}$$

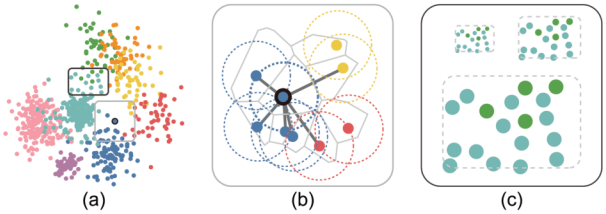

Figure 2: Illustrating the nearest neighbors of each data point and the influence of mark size in the perceived color difference for an input scatterplot (a) with two regions selected to zoom-in in (b,c). (b) The nearest neighbors for the selected point are defined by the α-shape graph; (c) a subset of data points selected from (a) shown in three different sizes.

consisting of a point distinctness term E_{PD}, a name difference term E_{ND}, and a color discrimination term E_{CD}. Each weight ω_i is a value range from 0 to 1 and each class C_i is assigned a unique color c_i. Besides these terms, a hard constraint is imposed to require the color difference between any two colors to be larger than a just noticeable difference threshold. E_{CD} is defined as the minimal CIELAB color distance [34] among every color pairs in τ; we describe the other terms in the following.

Point Distinctness. Given $\mathbf{X}$, an α-shape graph [21] is first constructed by connecting each point to its neighbors in the Delaunay graph and intersected within so-called α-balls (see an example in Fig. 2(b)). Then for each data point $\mathbf{x}_t$, its point distinctness is defined as:

$$\gamma(\mathbf{x}_t) = \frac{1}{|\Omega_t|} \sum_{\mathbf{x}_p \in \Omega_t} \frac{\Delta\epsilon(\tau(l(\mathbf{x}_t)), \tau(l(\mathbf{x}_p)))}{d(\mathbf{x}_t, \mathbf{x}_p)},$$

where Ω_t is set of nearest neighbors of $\mathbf{x}_t$, $\tau(l(\mathbf{x}_p))$ is the mapped color of $\mathbf{x}_p$, d is the Euclidean distance and $\Delta\epsilon$ is the CIELAB color distance [34]. By summing up the point distinctness of all data points, E_{PD} is defined as:

$$E_{PD} = \sum_{i}^{m} \frac{1}{n_i} \sum_{t=1}^{n} \gamma(\mathbf{x}_t)\delta(l(\mathbf{x}_t), i), \tag{2}$$

where $\delta(l(\mathbf{x}_t), i)$ is one, if the class label $l(\mathbf{x}_t)$ has a value of i, otherwise zero. If a class overlaps with different classes, the point distinctness value will be high, but small for a well-separated class.

Name Difference. Since color names are frequently used for communicating colors in visualization, a good palette should consist of colors associated with largely different names. Given a normalized color-term count matrix T, E_{ND} is defined as:

$$E_{ND} = \frac{2}{m(m-1)} \sum_{i \neq j \in m} D(T_{c_i}, T_{c_j}). \tag{3}$$

where T_{c_i} is the probability distribution of color names for a given color c_i and $D(T_{c_i}, T_{c_j})$ can be any distance measure for probability distributions; here we use the cosine distance [13].

To find the optimal τ in Eq. 1, a customized simulated annealing [1] algorithm is used, which starts with a random initial solution and a high temperature and progressively updates the solution and decreases the temperature to zero until reaching the convergence. This algorithm yields reasonable results in less than 10s for 20 classes, facilitating an interactive generation of palettes. However, a key limitation of Palettailor (and other automated colorization techniques) is that they do not support interactive highlighting, limiting their use in dynamic visualizations. We address this limitation by providing explicit support for the interactive emphasis on demand. Additionally, we also extend earlier colorization methods by modeling background contrast. This allows the highlighted data to pop out relative to the background, but also to other marks of the visualization. We also developed new constraints for the optimization to ensure discriminability and color consistency for emphasized and non-highlighted classes.

4 CONTEXT-PRESERVING HIGHLIGHTING

For a given multi-class scatterplot with m classes and n data items $\mathbf{X}$ and a background color c_b, our goal is to find a set of colors that creates the desired interactive emphasis effect for multi-class scatterplots. In line with the design requirements for pop-out effects and categorical data visualization [8, 14, 21], our problem can be formulated based on the following three design requirements:

(i) **DR1:** highlighting the selected data points as much as possible to deliberately attract user attention;
(ii) **DR2:** maximizing the visual discrimination between classes for efficiently exploring the data, for the selected and non-selected classes; and
(iii) **DR3:** maintaining color consistency for data points when they are dynamically highlighted or de-emphasized.

The resulting color mapping schemes satisfy DR1 by letting selections of interest pop out from the context while yielding better visual discrimination of classes for meeting DR2. Because the color for a class can vary depending on whether it is highlighted or not, we satisfy DR3 by requiring the two states (highlighted vs. de-emphasized) to have the same hue and saturation values while also ensuring similar color names. In doing so, we ensure a consistent perception of color appearance as data points of interest are interactively highlighted or de-emphasized.

4.1 Combination-based Highlighting

Most existing colorization techniques [9, 21] attempt to meet DR2. A key challenge for our technique, however, is to ensure that colors for classes of interest are sufficiently distinct to create the wanted pre-attentive 'pop out' effect (DR1). To meet this constraint, a widely used manual approach [24] is to modulate the color opacity or luminance contrast with the background for the non-selected data points. However, doing so will likely violate DR2 and DR3, because changing one or multiple colors might not preserve the ability of the viewer to visually discriminate all classes (see the bottom row of Figs. 1 (b,c)). On the other hand, simply extending existing colorization methods to enforce larger color differences for all classes might help meet DR1 and DR2, but the generated color mappings might differ noticeably when data points are dynamically emphasized (e.g., in response to user selection), causing user confusion (violating DR3).

To meet the three design requirements, we propose a combination-based highlighting method consisting of two steps. We first pre-generate the two color mapping schemes $\tau^s : L \mapsto c$ and $\tau^f : L \mapsto \tilde{c}$ for a given scatterplot, consisting of one palette $P^s = \{c_1, \cdots, c_m\}$ with salient colors over the background and a corresponding palette $P^f = \{\tilde{c}_1, \cdots, \tilde{c}_m\}$ with faint colors. For each data point $\mathbf{x}_i$, the color will be $\tau^s(l(\mathbf{x}_t))$ if selected for highlighting or $\tau^f(l(\mathbf{x}_t))$ otherwise. Since τ^s and τ^f both are required to meet DR2 and DR3, the overall color mapping will emphasize the data of interest (given the high saliency of τ^s) while also preserving the visual discriminability of all classes and ensuring good color consistency.

4.2 Modeling Contrastive Color Mappings

We formulate the search for a pair of color mapping τ^s and τ^f and their resulting palettes P^s and P^f as an optimization problem with the **objective function** $E(\tau^s, P^s, \tau^f, P^f)$:

$$\underset{\tau^s, P^s, \tau^f, P^f}{\arg\max} \; E(\tau^s, P^s, \tau^f, P^f) = \omega_0 \big(E_{PD}(\mathbf{X}, \tau^s) + E_{PD}(\mathbf{X}, \tau^f)\big) \quad (4)$$

$$+\omega_1 \big(E_{BC}(\mathbf{X}, \tau^s) - E_{BC}(\mathbf{X}, \tau^f)\big) \quad (5)$$

$$+\omega_2 \big(E_{ND}(P^s) + E_{ND}(P^f)\big) \quad (6)$$

$$+\omega_3 \, E_{CC}(P^s, P^f) \quad (7)$$

where the first two terms are based on the score of each color mapping scheme and the last two measure the score and compatibility of two resulting color palettes.

Each weight ω_i is a value in the range $[0, 1]$; we set all of them to 1 by default. The terms E_{PD} and E_{ND} are designed to meet DR2 while ensuring that colors are nameable. The second term E_{BC} satisfies DR1 by maximizing and minimizing the luminance contrast of the color mappings τ^s and τ^f over the background, respectively. The last term E_{CC} meets DR3 by requiring the corresponding colors in the two palettes to have similar perceived colors. To ensure all colors found by color mapping τ have enough discriminability, we apply a hard constraint in the form of the JND threshold. Since the perceived color difference varies across mark sizes (see Fig. 2(c)), we define the JND threshold based on the size-dependent model proposed by Szafir[39]. E_{PD} and E_{ND} are defined in Eq. 2 and Eq. 3. In the following, we will introduce the new terms for background contrast E_{BC} and color consistency E_{CC} and then describe how we solve the overall optimization problem.

Background Contrast. We define the contrast of each data point $\mathbf{x}_t$ to the background based on two factors: position-based class separability among its neighboring points and luminance difference to the background. The former measures by the difference between two separation degrees [2]:

$$\rho(\mathbf{x}_t) = b(\mathbf{x}_t) - a(\mathbf{x}_t)\,,$$

where $b(\mathbf{x}_t)$ is the between-class separation degree and $a(\mathbf{x}_t)$ is the within-class separation degree. The measures are defined as weighted sums of the non-separability of $\mathbf{x}_t$ from its neighborhood stemming from the same class and from other classes:

$$a(\mathbf{x}_t) = \frac{1}{|\Omega_t|} \sum_{\mathbf{x}_p \in \Omega_t} \frac{\delta(l(\mathbf{x}_t), l(\mathbf{x}_p))}{d(\mathbf{x}_t, \mathbf{x}_p)},$$

$$b(\mathbf{x}_t) = \frac{1}{|\Omega_t|} \sum_{\mathbf{x}_p \in \Omega_t} \frac{1 - \delta(l(\mathbf{x}_t), l(\mathbf{x}_p))}{d(\mathbf{x}_t, \mathbf{x}_p)}.$$

When most neighbor points of $\mathbf{x}_t$ have the same label as $\mathbf{x}_t$, $\rho(\mathbf{x}_t)$ is negative. However, a negative $\rho(\mathbf{x}_t)$ would reduce the contrast of the palette P^s over the background, which conflicts with the objective of Eq. 7. To address this issue, we use an exponential function to let $\rho(\mathbf{x}_t)$ always be positive and then normalize it to the range [0,1]. The contrast to the background of the ith class is:

$$\beta_i(\mathbf{X}, \tau) = \frac{1}{n_i} \sum_{t=1}^{n} \rho(\mathbf{x}_t) \Delta L(\tau(l(\mathbf{x}_t)), \mathbf{c}_b) \delta(l(\mathbf{x}_t), i). \quad (8)$$

where $\tau(l(\mathbf{x}_t)$ is the color of point $\mathbf{x}_t$, ΔL is the absolute luminance difference between point and background color in CIELAB space. The background contrast is defined as the sum of the background contrasts of each class:

$$E_{BC}(\mathbf{X}, \tau) = \sum_{i=1}^{m} \beta_i(\mathbf{X}, \tau). \quad (9)$$

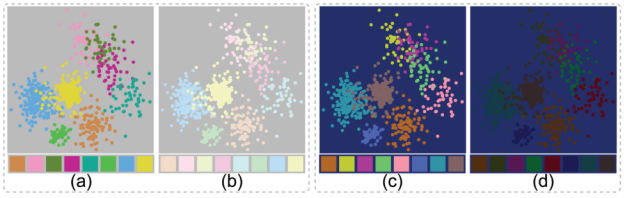

Figure 3: Results generated for two different backgrounds: grey (a,b) and blue (c,d). (a,c) the salient palettes and the colorized scatterplots; (b,d) the faint palettes and the colorized scatterplots.

This term depends on the non-separability and color luminance difference to the background, which means a class overlapping with other classes should have a larger β_i than a separated class. As shown in the top of Fig. 1(d) (green and dark blue classes), this yields better class separability. Since we take the contrast with the input background color into account, this model is able to adapt to different backgrounds, see Fig. 3 for illustration.

Color Consistency. To ensure colors are perceived similarly for highlighted and de-emphasized states, hue and saturation should be stable when data emphasis is dynamically changed. We also require colors to have similar color names, which helps the user in maintaining a mental map as data is selected and deselected. Hence, the following term measures name similarity between color pairs across the two contrastive palettes:

$$E_{CC}(P^s, P^f) = -\frac{1}{m} \sum_{i \in m} ND(P^s(i), P^f(i)), \quad (10)$$

$$\text{subject to} \quad H(P^s(i)) = H(P^f(i)) \text{ and } S(P^s(i)) = S(P^f(i)), \quad (11)$$
$$\text{for } i = 1, \ldots, m$$

We impose the hard constraint that $P^s(i)$ and $P^f(i)$ assigned to the ith class should have the same hue $H(P^s(i))$ and saturation $S(P^s(i))$ values. In other words, the difference between colors in

(a) Without Name Similarity (b) With Name Similarity

Figure 4: Comparing palettes generated by without (a) and with (b) name similarity terms. The results with solid borders are created by the salient palettes and the ones with the dashed border are created by the faint palettes.

$P^s(i)$ and $P^f(i)$ is only in the lightness channel and thus the objective in Eq. 7 can be simplified to a four-dimensional optimization problem. Fig. 4 compares the results generated by only imposing the hard constraint in Eq. 11 and the complete color consistency term. Incorporating name similarity not only yields contrastive palettes with highly similar color names but also enlarges name differences in each palette (see the four pink colors of the right palette in Fig. 4(a)).

Foreground Contrast Constraint. To ensure a visual pop-out effect for the highlighted data, we require the corresponding points to have large color contrast to the background, so that highlighted data is perceived as a clear 'foreground' layer. Since highlighted points are colorized by the P^s palette, we meet this requirement by imposing a hard constraint that each color in P^s should have a larger luminance contrast to the background than all colors in P^f:

$$\forall i, \forall j, \Delta L(P^s(i), \mathbf{c}_b) > \Delta L(P^f(i), \mathbf{c}_b). \tag{12}$$

For short, we refer to this constraint as the foreground contrast constraint.

Homogeneous Lightness Constraint. Previous studies [44] show that detecting targets over uniform backgrounds is more efficient than over complex ones. Thus, assigning uniform lightness to all colors in τ^f allows us to better meet DR1. However, this might reduce visual discrimination among non-highlighted classes, which would violate DR2. Fig. 6(b) shows an example with uniform lightness; note how the assigned pink and red colors in Fig. 6(a) become very similar in Fig. 6(b). To find a trade-off between these two requirements, we impose a constraint that all colors in P^f should have a small standard deviation in lightness so as to yield a relatively homogeneous background:

$$0 \leq \mathrm{SD}(\{L(P^f(1)), \ldots, L(P^f(m))\}) \leq \sigma \tag{13}$$

where $L(P^f(i))$ is the lightness of color $P^f(i)$ and σ is a small value specified by the user. To meet this constraint, we first find a uniform lightness level for all colors in each τ and subsequently perturb the lightness of each color $P^f(i)$ within a range of $[-\sigma, \sigma]$. Figs. 6(c, d) show how an appropriate σ can help to meet DR1 and DR2. Note how all colors in Fig. 1(d) are discriminable (see §4.4 for a formal analysis of this parameter). Also, observe how the non-highlighted points are assigned similarly faint colors though not identical lightness levels.

Algorithm 1 Simulated Annealing for Palette Generation

1: randomly initialize P^s and l
2: set an initial temperature T and $P^f = P^s$
3: set the lightness of all colors in P^f to l
4: $\Delta E = 0$
5: **while** $T > 0$ **do**
6: $\quad Q^s = P^s, Q^f = P^f$
7: $\quad$ **if** $random(0,1) < exp(\Delta E/T)$ **then**
8: $\quad\quad$ randomly choose a new $\hat{l}$ in the neighborhood of l
9: $\quad\quad$ set the lightness of all colors in Q^f to $\hat{l}$
10: $\quad$ **end if**
11: $\quad$ **if** $random(0,1) < 0.5$ **then**
12: $\quad\quad$ randomly exchange two colors from Q^s
13: $\quad\quad$ exchange the corresponding colors in Q^f
14: $\quad$ **else**
15: $\quad\quad$ randomly disturb one color c from Q^s
16: $\quad\quad$ update Q^f with Q^s via Eq. 11 and Eq. 13
17: $\quad\quad$ **while** Q^s, Q^f not satisfying *Eq.* 12 **do**
18: $\quad\quad\quad$ disturb color c from Q^s
19: $\quad\quad\quad$ update Q^f with Q^s via Eq. 11 and Eq. 13
20: $\quad\quad$ **end while**
21: $\quad$ **end if**
22: $\quad$ **while** $\min_{c_i, c_j \in Q^s} \Delta\varepsilon(c_i, c_j) < \eta$ or $\min_{c'_i, c'_j \in Q^f} \Delta\varepsilon(c'_i, c'_j) < \eta$ **do**
23: $\quad\quad$ randomly disturb c_i or c_j to get a new Q^s
24: $\quad\quad$ update Q^f with Q^s via Eq. 11 and Eq. 13
25: $\quad$ **end while**
26: $\quad \Delta E = E(Q^s, Q^f) - E(P^s, P^f)$
27: $\quad$ **if** $\Delta E > 0$ **then**
28: $\quad\quad P^s = Q^s, P^f = Q^f, l = \hat{l}$
29: $\quad$ **else**
30: $\quad\quad$ **if** $random(0,1) < \exp(\Delta E/T)$ **then**
31: $\quad\quad\quad P^s = Q^s, P^f = Q^f, l = \hat{l}$
32: $\quad\quad$ **end if**
33: $\quad$ **end if**
34: $\quad T = \alpha T$
35: **end while**

4.3 Optimization for Contrastive Palettes

We implement the above constraints in a simulated annealing algorithm to generate a pair of color mapping schemes τ^s and τ^f (see pseudocode in Algorithm 1). Before presenting our algorithm, we map each class label to an index range of $[1, m]$ and assume that the color c_i in the palette P is assigned to the ith class. After initializing a high "temperature" and a palette P^s with m random colors, this method iteratively updates the palettes with three major steps in each iteration: i) finding a neighboring solution of P^s, ii) using P^s to update P^f and finding a neighboring solution of P^f; and iii) refining the two temporary palettes Q^s and Q^f to meet the JND constraint. In the following, we describe the last two major steps.

Finding the neighboring Q^f near P^s (line 6-21). Based on the current solution P^s and P^f, we first update Q^f by finding a new

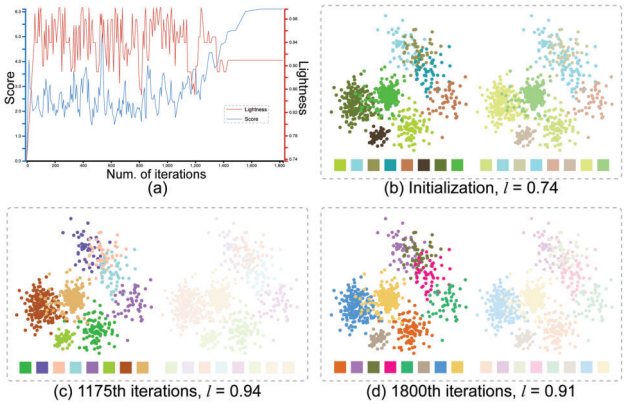

Figure 5: Convergence of our method. (a) curve on E versus the number of iterations(blue) and curve of l versus the number of iterations (red); (b) colorized results and palettes with random initialization; (c,d) results after 1175 and 1800 iterations.

lightness value for all colors (line 6-10). To rapidly produce a homogeneous background, we set a large probability for accepting a uniform lightness for all colors initially and decrease it with increasing number of iterations (see the probability distribution in the supplemental material). Then, we try to find a neighbor solution Q^s of P^s by randomly exchanging two colors (line 12) or choosing a new color around the neighborhood of one selected color (line 15). After generating a new solution for Q^s, we update Q^f in the terms of Eq. 11 and Eq. 13 (line 16). Namely, we maintain the hue and saturation of each color chosen from Q^s and perturb the latest uniform lightness value (see lines 28 and 31) by a random value in $[l-\sigma, l+\sigma]$ to increase the discriminability. Finally, we disturb the solution until satisfying Eq. 12 (see line 17-20). Fig. 5 (a) shows the evolution curve of l, which has large variations at the beginning and then gradually converges to a stable value. Figs. 5(b, c, d) show scatterplots visualized with palettes yielded at the initialization, the 1175th, and the 1800th iteration. l quickly reaches a reasonable value but the colors keep changing to further improve class discrimination (see Figs. 5(c, d)).

Refine Palettes via JND constraints (line 22-25). To ensure that all colors in Q^s and Q^f can be discriminated, we compute the minimal distance between each pair of colors in both palettes and see if it satisfies the hard JND constraint. If their difference is smaller than a mark size-dependent JND threshold η [39], we randomly perturb the corresponding color pair in Q^s and use Q^s to update Q^f until all color pairs meet the constraint. Since η is a JND threshold modulated by the given mark size, we recalculated η when the size is changed. JND modulation ensures that smaller marks are allocated larger color difference relative to other classes for discriminability.

Due to the stochastic nature of this algorithm, random initialization of palettes and lightness does not influence the final solution in our experiments. The time complexity of each iteration is $O(m^2)$ and the time complexity for the whole algorithm is $O(tm^2)$ for a total number of iterations t. Our method allows us to yield reasonable palettes for scatterplots with 20 classes in less than 10s.

4.4 Parameter Analysis for Background Complexity

A key parameter that affects the likelihood that the emphasized data will pop out is the complexity of the background. Namely σ, which controls the standard deviation for the lightness of the non-highlighted data (i.e., for the colors of the faint palette P^f). A small σ will reduce the discriminability of all background classes (see the two red colors in Fig. 6(b)), while a large value will degrade user performance in identifying the highlighted data (see the yellow and pink classes in Fig. 6(d)). Experimenting with different levels, we found a default value of $\sigma = .05$ to be a good trade-off between emphasizing classes of interest while preserving background discriminability (see Fig. 6(c)).

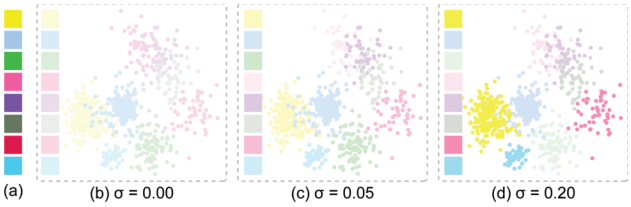

Figure 6: Effect of the lightness standard deviation σ on the faint palette: (a) salient palette P^s; (b,c,d) faint palettes P^f generated by different σ (left) and resulting scatterplots (right).

5 EVALUATION

Considering that oftentimes there are multiple tasks involved in the analysis of a scatterplot, we evaluated the effectiveness of our method from two different perspectives: *static visualization* and *interactive exploration*. For static visualizations, all classes have equal importance and the main task is to discriminate different classes (see Table 1). Here, we use a *counting task*, prompting participants to count the number of unique classes in a visualization, thus measuring how discriminable the classes are. For interactive exploration, classes of interest should be highlighted and other classes need to be de-emphasized. In this case, the most important part is to find the classes of interest. However, during interactive exploration, different classes would be highlighted. When a class gets out of focus and its color faints, the viewer still needs to recognize which class it is. Furthermore, people often need to distinguish the context around a highlighted class to examine the data distribution. To achieve these evaluation goals, we created three tasks to examine the efficiency of our method: (1) a *highlighting task*, (2) a *matching task* and (3) a *selecting task*. The last two are designed to evaluate the effectiveness of our technique in preserving color consistency and class discrimination of non-selected data points, referred as context-preserving tasks. We conducted two controlled experiments across the four tasks by crowdsourcing 150 participants through Amazon Mechanical Turk (AMT).

Benchmark Methods. We compared our method with two existing colorization methods as benchmarks: 1) Palettailor [21], a state-of-the-art automated colorization tool designed for generating discriminable and optimized categorical palettes; and 2) Tableau [37],

Table 1: Layout of the two crowdsourced experiments: *Tableau (D)* indicates *Tableau with default assignment, Tableau (O)* indicates *Tableau with optimal assignment, Palettailor (L)* indicates *Palettailor with lightness adjustment, Palettailor (A)* indicates *Palettailor with alpha blending, Tableau (D+H)* indicates *Tableau Highlighter with default assignment, Tableau (O+H)* indicates *Tableau Highlighter with optimal assignment, Our Method (S)* indicates *Our Method (static), Our Method (I)* indicates *Our Method (interactive).*

Experiment 1: Static Visualization		*Experiment 2: Interactive Exploration*			
Methods	*Tasks*	*Methods*	*Tasks*		
Palettailor	Counting Task	Palettailor (L)	Highlighting Task	Matching Task	Selecting Task
		Palettailor (A)			
Tableau (D)		Tableau (D+H)			
Tableau (O)		Tableau (O+H)			
Our Method (S)		Our Method (I)			

an interactive data visualization software with designer-crafted palettes. Tableau also has a purposefully designed tool called Highlighter for emphasizing a specific class while maintaining the context of the other classes. We therefore include Tableau Highlighter among our benchmark methods. However, Palettailor only supports static data visualization. Hence, we compared our method with two commonly used extrinsic emphasis techniques: adjusting the lightness contrast or the opacity value of a given palette to emphasize desired classes [18].

Figure 7: Illustration of tasks: (a) *counting task*; (b) *selecting task*; (c) *highlighting task* and (d) *matching task*. The scatterplot is from one of the datasets used in the experiment. Here, we apply a Tableau-provided color palette with randomized rotation to avoid learning effects.

Tasks & Measures.

- *Counting task (global discrimination).* Following previous methodologies [21, 42], we asked participants to identify how many classes (i.e., different colors) they can see in a given scatterplot. As shown in Fig.7(a), they entered their answer by selecting from multiple options that were displayed below the scatterplot. We recorded the answer and *response time* for each trial, and computed the relative *error* as the proportion of the total number of classes. For example, a participant answering with 8 classes when there were actually 10 would be reported as an error of 0.2.
- *Selecting task (local discrimination).* As shown in Fig.7(b), we put a circle around a randomly selected point of the highlighted class in a scatterplot and then asked participants to select all colors from the palette that appear within the circle. The radius of the circle was 10× the radius of each point. Participants entered their answers by selecting colors from a palette displayed below the scatterplot. We recorded user selection and *response time* for each trial, and computed the relative *error* as the proportion of the actual number of classes.
- *Highlighting task.* Following the methodology of Mairena et al. [23], we asked participants to examine the scatterplot first and then point out which class they believe is being emphasized, as shown in Fig. 7(c). We allow participants to click any representative point in the scatterplot as a way of selecting the class. For each trial, we measured the *binary error* (i.e., whether the class chosen by a participant is the intentionally highlighted one). We also tracked the *response time*.
- *Matching task (color consistency).* As shown in Fig.7(d), we asked participants to select the cluster from a scatterplot whose color most closely matches the indicated color. Participants could click any representative point in the scatterplot as a way of selecting the class. For each trial, we measured the *error (0/1)* (i.e., whether the class chosen by a participant was the correct one) and the *response time*.

Hypotheses.

We expect our methods to outperform the benchmarks in preserving context and color consistency. That is, we expect to attain

the benefits of visual focus without sacrificing performance on tasks requiring context. Specifically, we pose the following hypotheses:

H1. Our palette generation method for static visualization is comparable to the benchmark conditions in the *counting task*.

H2. Our palette generation method for interactive exploration is comparable to the best benchmark conditions in the *highlighting task*.

H3. Our palette generation method for interactive exploration outperforms the benchmark conditions in the two context-preserving tasks (*selecting task* and *matching task*).

Dataset Generation. The scatterplot datasets used in our studies were generated as follows. First, to avoid learning effects, we chose three different class numbers: 6, 8 and 10 classes. Each class was generated using Gaussian random sampling and random placement in a 600×600 area. Following the procedure described in Lu et al. [21], all scatterplots belonged to one of four possible configurations of class size and density: small & dense ($n = 50, \sigma = 20$), small & sparse ($n = 20, \sigma = 50$), large & dense ($n = 100, \sigma = 50$), and large & sparse ($n = 50, \sigma = 100$). In total, we generated 3 (class number) $\times$ 4 = 12 scatterplots.

Engagement Checks. In addition to the analyzed trials, we also generated multiple engagement checks to verify that participants were paying attention to the task. Engagement checks comprised a scatterplot with only 4 fully-separated classes, each with a very distinctive color. For the *highlighting task*, we randomly chose one class to be emphasized, and assigned the other classes a lightness value of 0.9 to let them fade into the white background. For the other tasks, we assigned highly distinctive colors to allow for an easy class discrimination. We excluded participants from the analysis who failed more than one engagement check.

Procedure. Each participant went through the following steps: (i) viewing an instruction for the task and completing three training trials; (ii) completing each analyzed trial as accurately as possible; (iii) providing demographic information. The three training trials were identical to the subsequent real test. We implemented different response mechanisms for the four tasks. For the highlighting and matching tasks, participants clicked a data point belonging to the class they thought was the correct one. For the counting and selecting tasks, participants entered their class count or selected the corresponding colors by choosing from multiple options displayed below the visualization.

Analysis. Following previous research [21], we analyzed the results using 95% confidence intervals, and conducted Mann-Whitney tests to compare the differences between the conditions. In addition, we computed the effect size using *Cohen's d* (i.e., the difference in means of the conditions divided by the pooled standard deviation). We calculated an ANOVA-type statistics (ATS) without normality assumption (using the R-package GFD [7]) to examine the interaction effect between variables.

5.1 Experiment 1: Static Visualization

We conducted this experiment to examine how well our method supports people to visually distinguish different classes in a static visualization through a *counting task*.

Conditions. In this experiment, we included four conditions:

(1) *Palettailor:* This method represents the state-of-the-art automated colorization algorithm for multi-class scatterplots with the best class discriminability, corresponding to Fig. 1(b)-top.
(2) *Tableau with default assignment:* This method represents the default visualization effect for designer-crafted categorical palettes. We assigned each color to each class in turn, to mimic how Tableau performs the color assignment, as shown in Fig. 1(a)-top.
(3) *Tableau with optimal assignment:* This method represents the state-of-the-art for designer-crafted categorical palettes. We applied the optimal discrimination assignment approach [42] to the Tableau-10 palette, to mimic the best discriminable result from a manual selection of the user, as shown in Fig. 1(c)-top.
(4) *Our method (static):* Assigning colors using a salient palette generated by our automated colorization method described in Sec. 4.3 with default settings. This condition reflects the fully-automated colorization option of our method for static visualization, as shown in Fig. 1(d)-top.

Experimental Design. We used a *within-subject* design: each participant completed all four conditions. To avoid ordering effects, we randomly shuffled the display order of the given 48 stimuli (4 conditions $\times$ 12 scatterplots). For each stimulus, we also randomly rotated the scatterplot. Furthermore, we added three engagement checks to ensure participants were paying attention to the experiment.

5.1.1 *Counting task.*

We asked participants to identify how many classes (i.e., distinct colors) they find in a given scatterplot, as shown in Fig.7(a). Participants choose an answer from multiple options given below the scatterplot. We expected to see that our method will be comparable to other state-of-the-art conditions w.r.t. error and response time. We conducted this task through AMT with 30 participants. According to the completion time in the study (the details can be found in the supplementary materials), we paid each participant $1.75 for the task based on the US minimum hourly wage. No participant claimed color vision deficiency on their informed consent.

Results. Fig.8 shows the results of the visual discriminability experiment. While *Palettailor* achieves the best performance, our method performs better than the two *Tableau* conditions. In particular, *Tableau with default assignment* exhibited the worst performance. That said, there is no significant difference between these four conditions, implying a statistically similar performance. In terms of response time, we found that *Tableau with optimal assignment* and *Tableau with default assignment* take less time than our method and *Palettailor*. However, again these differences were not statistically significant. The results overall indicate that our palette generation method is comparable to the benchmarks for the *counting task* (**H1** confirmed).

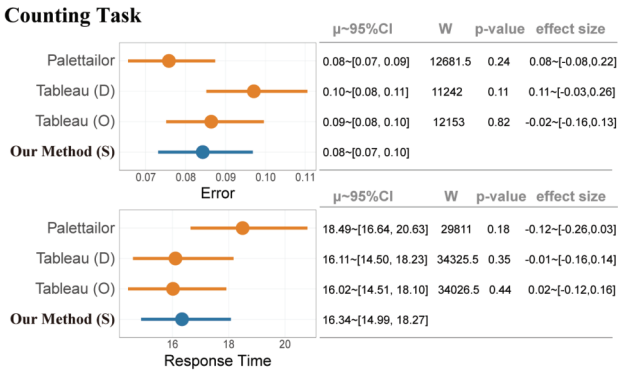

Figure 8: Confidence interval plots and statistical tables for the counting task. Error bars represent 95% confidence intervals. Each table shows the statistical test results of our experimental condition (*Our Method (S)* with the three benchmark conditions (*Palettailor*, *Tableau (D)* and *Tableau (O)*), showing the mean with 95% confidence interval (μ ~ 95%CI), W-value and p-value from the Mann-Whitney test, as well as effect size (d ~ 95%CI).

We did not find a significant interaction between *colorization methods* and *cluster number* ($F(3, 1432) = 0.1342; p > 0.1$). The effectiveness of the different methods on visual discriminability seems insensitive to the number of clusters.

5.2 Experiment 2: Interactive Exploration

We designed three tasks to examine the efficiency of our method for interactive exploration: a *highlighting task* for measuring the emphasis effectiveness, and two tasks for measuring the context-preserving performance: a *matching task* and a *selecting task*.

Conditions. We included five conditions, the illustrations for different conditions can be found in the supplementary materials:

(1) *Palettailor with lightness adjustment:* This condition represents a common highlighting strategy: applying lightness adjustments to a given colorized scatterplot that has good class discriminability to begin with. We maintain the original lightness level of the emphasized class while adjusting the lightness of all other classes. The adjusted lightness value depends on the background color. For example, if the background is white, the lightness should be high.

(2) *Palettailor with alpha blending:* This condition represents another highlighting strategy: applying alpha blending to a given colorized scatterplot that has a good class discriminability to begin with. We set the opacity of the class to be emphasized to 1.0 while adjusting the opacity of all other classes to 0.2, which is recommended by Bartram et al. [3].

(3) *Tableau Highlighter with default assignment:* For each color in the Tableau-10 palette, we obtain its corresponding faint color for the non-highlighted classes from the Tableau Highlighter. We applied this strategy to the default assignment of the Tableau palette.

(4) *Tableau Highlighter with optimal assignment:* Similar to the above, but with an optimal assignment of the Tableau palette.

(5) *Our Method (interactive):* Combining colors from the two contrastive palettes (salient and faint colors).

Experimental Design. Similar to the first experiment, we used a *within-subject* design: each participant completed all 5 conditions across 12 scatterplots *with a randomly chosen class to be highlighted* (60 stimuli in total). To avoid ordering effects, we randomly shuffled the display order of stimuli. For each stimulus, we additionally randomly rotated the scatterplot. We also included four engagement checks to ensure participants were paying attention.

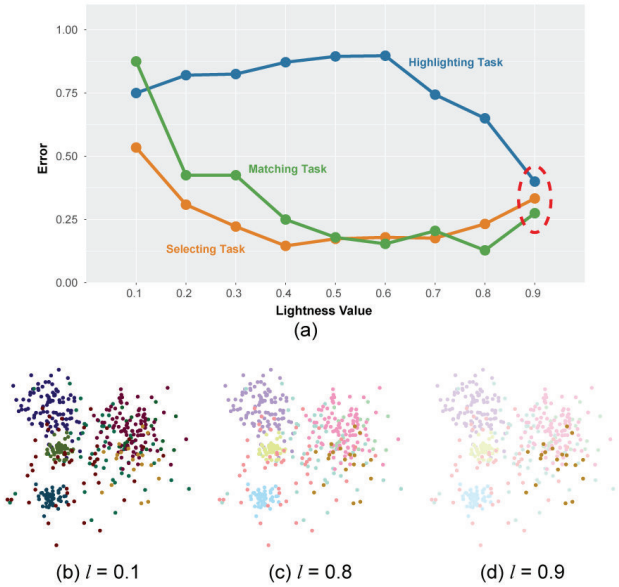

Figure 9: (a) Results of the pilot study for selecting a proper lightness value: larger error value implies lower performance for each task. (b, c, d) Example trials used in the study with different lightness values. A lightness value of 0.9 was selected as a sweet spot for the experiments.

5.2.1 *Pilot for Selecting Lightness Value.*

One potential issue for using lightness to emphasize the desired class is that we cannot choose a value arbitrarily. We therefore conducted a pilot study across all three tasks, to determine an appropriate lightness level to assign to the non-highlighted classes. We used four 8-class scatterplots in our pilot, and utilized *Palettailor with lightness adjustment*. The lightness value varied incrementally within a range of [0.1, 0.9] and a step of 0.1. In total, we included 4 (scatterplots) × 9 (lightness levels) = 36 trials, plus 3 engagement checks. The trials were presented in random order. We recruited 10 participants for each task (30 participants in total) through AMT for a pilot. Participants who failed more than one engagement check were excluded, with new recruits taking their place, until we reached 10 participants. Each participant went through all 36 stimuli. All participants were US residents with a task-approval rate larger than 97% and indicated normal color vision on their informed consent.

In Fig. 9(a), we plot the average error rate for each lightness value of the three tasks: *highlighting task*, *selecting task* and *matching task*.

On a white background, error decreased for the highlighting task as the lightness value increased. This is due to the non-emphasized classes fading into the background, enabling the highlighted class to stand out, as shown in Figs.9(b, c, d). Conversely, the errors for the selecting and matching tasks improved with an increased lightness value. To reach the best performance among these tasks, we chose 0.9 as a reasonable lightness value for our experiment.

5.2.2 Highlighting task.
To evaluate whether our approach enables viewers to intuitively identify the emphasized class from a scatterplot, we conducted this task through AMT with 30 participants being accepted. The user interface is shown in Fig.7(c). According to the completion time in the study (the details of the pilot study can be found in the supplementary materials), we paid each participant $1.00 for the task based on the US minimum hourly wage. No participant claimed color vision deficiency on their informed consent.

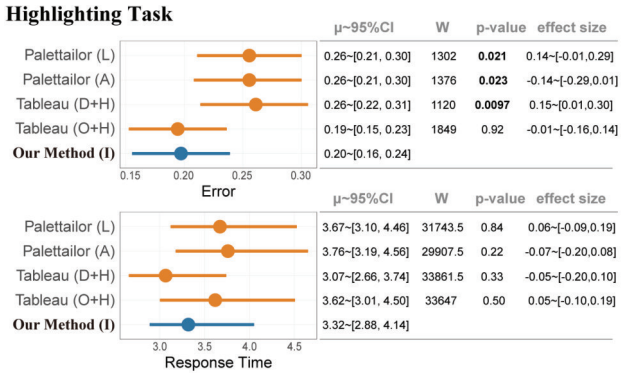

Figure 10: Confidence interval plots and statistical tables for the highlighting task. Error bars represent 95% confidence intervals. Each table shows the statistical test results of our experimental condition with the benchmark conditions (*Palettailor (L)* indicates *Palettailor with lightness adjustment, Palettailor (A)* indicates *Palettailor with alpha blending, Tableau (D+H)* indicates *Tableau Highlighter with default assignment, Tableau (O+H)* indicates *Tableau Highlighter with optimal assignment, Our Method (I)* indicates *Our Method (interactive)*).

Results. Fig. 10 shows the results of the experiment for the *highlighting task*. *Our Method* exhibited a significantly lower error rate than *Palettailor with lightness adjustment* ($p = 0.021$), *Palettailor with alpha blending* ($p = 0.023$), and *Tableau Highlighter with default assignment* ($p = 0.0097$). There was, however, no significant difference compared to *Tableau Highlighter with optimal assignment* ($p = 0.92$). There were no significant differences in response time between our method and the benchmarks as the P value is more than 0.05 ($p > 0.05$). We also did not find a significant interaction effect between the *colorization methods* and the *number of clusters* ($F(4, 1790) = 0.2685; p > 0.1$), meaning that visual emphasis is not affected by the number of clusters, which is consistent with the behavior of a popout effect.

The results indicate that our palette generation method outperforms commonly-used highlighting methods (e.g., lightness adjustment, alpha blending, and *Tableau Highlighter with default assignment*), while being comparable to the best-case scenario of a state-of-the-art commercial system such as *Tableau Highlighter with optimal assignment*. The results thus suggest an effective visual emphasis for our method, exceeding the performance of commonly applied manual highlighting techniques. We therefore consider **H2** to be confirmed.

5.2.3 Matching task.
As shown in Fig.7(d), we asked participants to select the cluster from the scatterplot whose color most closely matches the indicated color. The purpose of this color matching task was to examine whether our approach can maintain class recognition even when class color is changed in response to interactive highlighting. We conducted this task through AMT with 30 participants being accepted. According to the completion time in the study, we paid each participant $1.25. No participant claimed color vision deficiency on their informed consent.

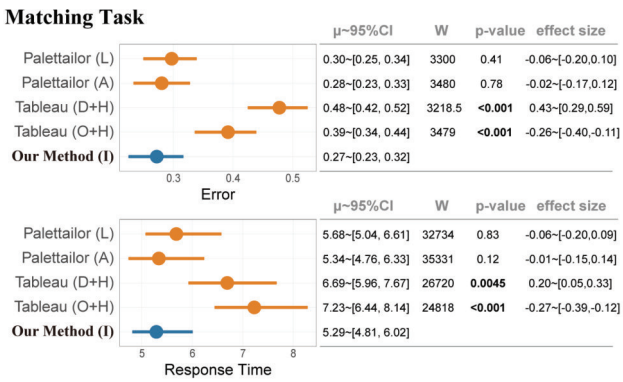

Figure 11: Confidence interval plots and statistical tables for the color-matching task.

Results. Fig. 11 shows the results of the *matching task*. Our method leads to a significantly lower error rate and response time compared to *Tableau Highlighter with default assignment* and *Tableau Highlighter with optimal assignment*, while it is slightly better than *Palettailor with lightness adjustment* and *Palettailor with alpha blending*. No significant interaction between *colorization methods* and *cluster number* was found ($F(4, 1790) = 2.163; p > 0.05$). The result indicates that our palette generation method has a better performance than the benchmark conditions for the *matching task* w.r.t. color consistency, which confirms **H3**.

5.2.4 Selecting task.
We asked participants to select all colors from the palette that appear within a given circle. The user interface is shown in Fig.7(b). For this task, participants need to discriminate different colors around a small area, to examine how well the different methods can preserve the context of emphasized data. We conducted this task through AMT with 30 participants being accepted. We paid each participant $2.00 for an hourly wage consistent with the US

minimum. No participant claimed color vision deficiency on their informed consent.

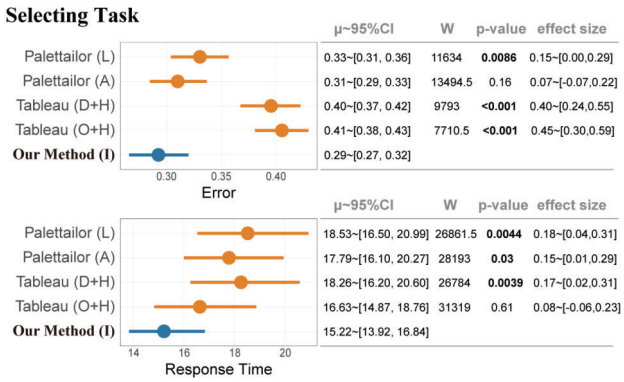

Figure 12: Confidence interval plots and statistical tables for the selecting task.

Results. Fig.12 shows the results of the visual separability experiment for local discrimination. Our method exhibits a significantly lower error rate relative to all other benchmark conditions, except *Palettailor with alpha blending*. Although non-significant, we still achieved a better error rate than *Palettailor with alpha blending* ($p = 0.16$). As for the completion time, our method achieves better performance than all other conditions, with a significantly shorter time than *Palettailor with alpha blending* ($p = 0.03$). These results support **H3**. No interaction was found between *colorization methods* and *cluster number* ($F(4, 1790) = 0.5798; p > 0.1$).

5.3 Discussion

We evaluated the effectiveness of our approach against the benchmark conditions through two crowdsourced experiments for two different scenarios (static visualization and interactive exploration). In the *counting task* for a static visualization (see Fig.8), we found that *Palettailor* outperformed the *Tableau* conditions and *Our Method (static)*. This is reasonable since the design goal of *Palettailor* is to maximize class discriminability. *Our Method (static)* seems to be slightly better than *Tableau with optimal assignment*. Notably, the latter achieves better performance than *Tableau with default assignment*, which indicates that an optimal assignment approach [42] does indeed improve discriminability for visualization. The results suggest that while *Palettailor* outperforms our method in the *counting task* for the global discriminability, the advantage is not substantial, thus representing a small overhead to pay for the ability to emphasize the desired classes.

For interactive exploration, our method shows a better performance. In the *highlighting task*, we found that participants intuitively select the emphasized class in our approach. There is a significant advantage for *Our Method* over some of the benchmark conditions (*Palettailor with lightness adjustment*, *Palettailor with alpha blending* and *Tableau Highlighter with default assignment*). This indicates that our method attains better visual emphasis than most benchmarks while being comparable to the best-case scenario *Tableau Highlighter with optimal assignment*). Interestingly, *Palettailor with alpha blending* did not yield good highlighting performance. One reason is that colors from Palettailor might have a similar lightness to the background, e.g., light yellow class in Fig.1(b)-top. Another reason is that blended colors could inadvertently attract attention away from the desired class, some examples can be found in the supplementary materials.

As for the two context-preserving tasks, first, we found that in the color *matching task*, *Our Method (interactive)* performed better than *Tableau Highlighter with default or with optimal assignment*, while achieving similar performance to *Palettailor*, both with lightness adjustment and alpha blending. This is likely because our method, like other lightness adjustment approaches, works by only perturbing lightness while maintaining the original hue and saturation. When the background color is achromatic (white), during alpha blending, the hue will not be changed, thus achieving good performance. However, for a chromatic background, alpha blending might result in poor class discriminability and color consistency (see Fig. 13). Since *Our Method* also preserves name similarities for de-emphasized colors, it slightly outperforms *Palettailor with lightness adjustment* and *Palettailor with alpha blending*. An example illustration of this phenomenon can be found in the supplementary materials. For the *selecting task*, we found that our method achieves the best performance among all benchmark conditions, even though there was no significant difference to *Palettailor with alpha blending*. However, *Our Method* leads to a significantly shorter response time than the alpha blending approaches, likely because the latter potentially introduces new blended colors that could distract the viewer.

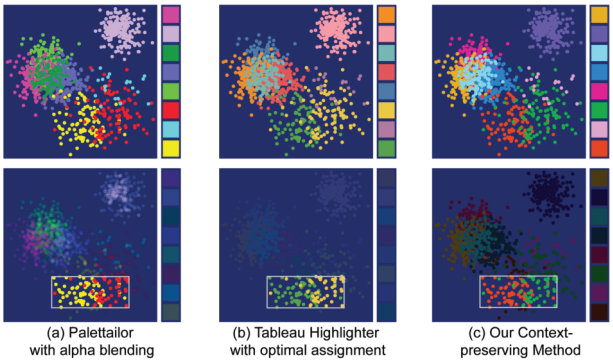

Figure 13: Results generated for different methods with a blue background: (a) (top) palette and colorized scatterplot from Palettailor; (bottom) a highlighting effect is achieved by reducing the opacity of non-selected data points; (b) (top) Tableau palette and colorized scatterplot; (bottom) achieving a highlighting effect by applying Tableau Highlighter function; (c) (top) salient palette and colorized scatterplot by our context-preserving highlighting method; (bottom) highlighting result by combining salient and faint color palette.

The results indicate that our method maintains class discriminability for all classes while still achieving an intuitive highlighting effect. An added benefit to our context-preserving approach is that it automatically adapts to different backgrounds, thus producing

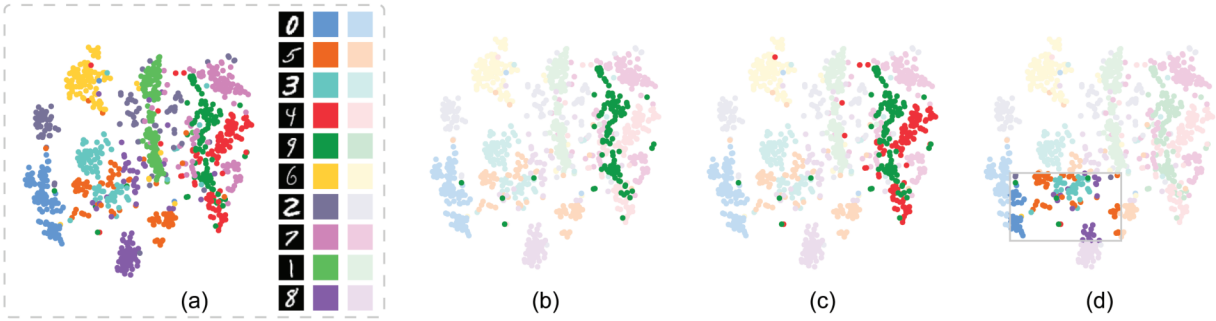

Figure 14: Exploring the MNIST dataset [16] with our context-preserving highlighting technique. Result of static visualization (a) and the corresponding highlighting results by different selection methods: (b,c) legend selection, (d) brushing selection.

more satisfactory results for chromatic backgrounds than Palettailor and Tableau Highlighter (see Fig.13). A detailed analysis of response time, including the influence of class number, along with an analysis of potential speed-accuracy tradeoffs, can be found in the supplementary materials.

Our evaluation has some **limitations**: First, we only tested two state-of-the-art colorization methods (*Palettailor* and *Tableau*, and their corresponding highlighting strategies). This choice was done to mitigate fatigue effects on participants. Whether other palettes (e.g., ColorBrewer's collection and Colorigorical) would lead to similar results remains to be seen. Second, our experiment only focused on color-based highlighting; however, many methods exist using other visual variables to emphasize classes such as shape and mark size. Third, the experimental setup is idealized: the scatterplots are relatively simple and the emphasis is applied to entire classes. The evaluation should therefore be extended with more complex datasets and tasks. We also did not measure participant preference (e.g. from an aesthetics standpoint, where designer-crafted palettes might perform better than auto-generated results), leaving this aspect as future work. Finally, although we made attempts to reduce learning effects (e.g., random display order, randomly rotating scatterplots), some residual learning could still have happened due to stimuli rotation.

6 SYSTEM AND CASE STUDIES

To aid designers in crafting categorical color palettes with contextual highlighting effects, we developed a web-based design tool that embodies our methodology[4]. Details of the system can be found in the supplementary materials. The interface allows users to select and highlight data via a variety of interactions, including clicking individual data points, clicking color legend to select an entire class, and brushing to select points that lie within a range. In the following, we present two extensions of our technique and conducted two case studies on real-world datasets.

Extensions for Bar and Line Charts. In addition to scatterplots, our color mapping method can be easily extended to other categorical visualization types such as bar or line charts. This is achieved by treating each bar or line segment as a mark and then using the same method to compute their class contrasts, where the detailed description can be found in the supplementary materials.

Extensions for Multi-view Visualizations. Our technique can be extended to generate consistent color mapping schemes for multi-view visualizations of the same multi-dimensional data. For example, the line chart in Fig. 15 displays trends of different classes, the bar chart shows the total number of each class. Following one of the multi-view consistency principles that the same nominal values in a field should be encoded by the same colors across different charts [29], we generate the color mapping scheme for the view with most overlap between classes and apply this scheme to the other views.

6.1 Handwritten Digits Dataset

Here, we analyzed the MNIST data of handwritten digits [16], which contains 784 data dimensions with ten classes. We project this dataset onto a 2D scatterplot using t-SNE with 1000 random distinct samples. As shown in Fig. 14(a), our technique first colorizes the scatterplot with an overall good class discriminability. The user can click on the legend color to select the corresponding class – in this case, the green class (see Fig. 14(b)), which represents the number 9. She finds that this class is heavily overlapping with red, so she also clicks to select the latter (see Fig. 14(c)). She speculates that this might be caused by the similar appearance of the two numbers. To further investigate similar overlaps, she brushes over the scatterplot to select the left bottom region: the orange, sky blue, and purple classes representing 5, 3, and 8, respectively. During this exploration, our technique produces consistently good pop-out effects, as the emphasized data is interactively selected and de-selected (see Figs. 14(b, c, d)). Additionally, class separability and color consistency are well maintained regardless of which data subset is highlighted.

6.2 Air Quality Dataset

We conducted a second case study with a real-world dataset, this time using line and bar charts. Here, we analyzed an air quality dataset provided by Vito et al. [5] containing hourly recordings of a multi-sensor gas device deployed in an Italian city for two months in 2004. The dataset contains five classes corresponding to different gases: *CO*, *NMHC* (non-metanic hydrocarbons), NO_x , NO_2 and O_3.

[4]https://palettailor.github.io/highlighting/

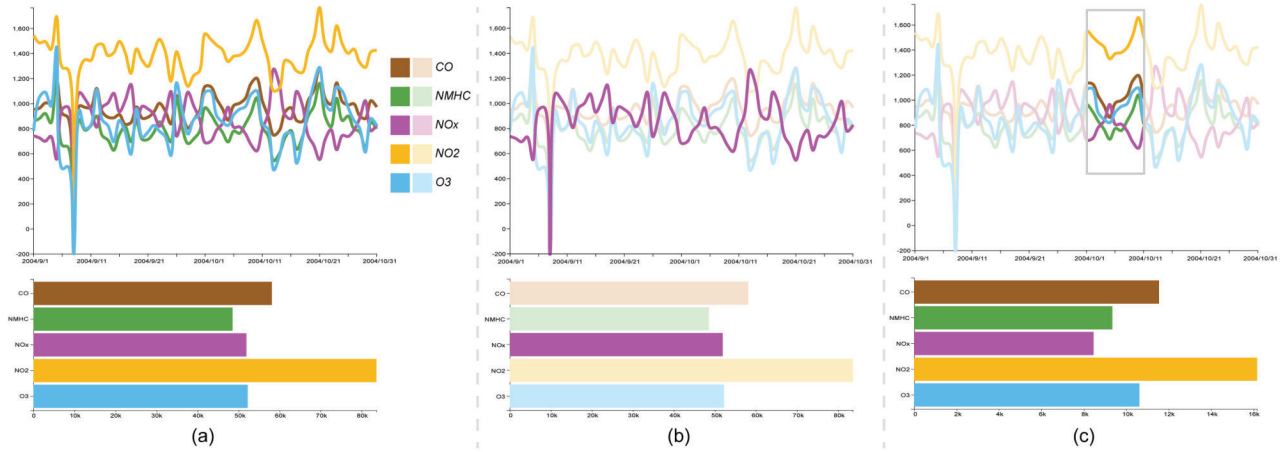

Figure 15: Visualizing an air quality dataset [5] using two linked views with our context-preserving highlighting technique. (a) (top) colorizing the line chart by using a salient color palette; (bottom) applying the salient palette to a bar chart; (b) (top) highlighting result by combining salient and faint color mapping schemes with legend selection; (bottom) corresponding bar chart; (c) (top) highlighting result for the brushing selection; (bottom) corresponding bar chart for the selected data. Our method produces a good highlighting effect while maintaining class discriminability during interactive exploration.

Fig. 15 shows line and bar charts colorized using our technique, where each gas type is represented by a unique color. The line charts represent the gas change over time and the bar charts represent the total amount of each gas type. We explore one class by interactively highlighting it through a legend selection. Fig. 15(b) emphasizes the pink class, which represents NO_x. Our method achieves good overall class discriminability while allowing the user to still investigate any of the de-emphasized classes. The brush selection results shown in Fig. 15(c), show that our technique maintains good separability between all trendlines, for both selected and non-selected classes. This ability to interactively vary the highlight while still maintaining context makes our method especially suitable for interactive visual exploration.

7 CONCLUSION AND FUTURE WORK

We presented an interactive context-preserving color highlighting approach for multi-class scatterplots. Our method allows viewers to intuitively identify points of interest, while ensuring visual discriminability of all classes in a visualization, and maintaining a stable color mapping scheme during interactive exploration. This goal is achieved by generating two contrastive palettes and then dynamically combining these two palettes, thus allowing for an interactively-variable focus effect. In addition to modeling intra-class discriminability, our method also ensures sufficient contrast with the background. We evaluated our approach through a crowdsourcing study, which empirically demonstrates reliable highlighting and good class discrimination for our generated palettes. To help users generate such designs, we extended this method to other categorical visualizations such as bar charts and lines. In addition, we propose a web-based tool that implements our approach, enabling a quick, data-driven generation of palettes for a context-preserving emphasis effect.

Our user study focuses on contextual highlighting to points of interest in single view visualizations. In the future, we will investigate its effectiveness on tasks spanning multiple views (e.g., comparison tasks [28]). In addition to color, other channels (e.g., shape [20] and mark size [35]) are known to have an effect on visual prominence, which could interact with our color-based highlighting approach. Future work could explore the possibility of modeling these factors to produce reliable intrinsic highlighting across multiple visual channels.

Second, our approach produces colors that might not be friendly to people with color vision deficiency. Future work could thus extend our palette generation techniques to incorporate physiologically based models of color-vision deficiency [22]. Such an extension could allow for color optimization with accessibility constraints. Aesthetic preferences should also be concerned in the automated colorization method to better serve users.

Lastly, we evaluated the effectiveness of our palettes against a limited number of highlighting techniques. However, since there are many different highlighting methods, such as shape, size, and animation, it would be interesting to fully investigate the strengths and limitations of these approaches for engendering a highlight effect.

ACKNOWLEDGMENTS

This work is supported by the grants of the NSFC (62132017, 62141217), and Shandong Provincial Natural Science Foundation (ZR2022JQ32). The authors would like to thank Mi Feng, Michael Sedlmair, and Qiong Zeng for their fruitful discussion and support.

REFERENCES

[1] EHL Aarts. 1989. A stochastic approach to combinatorial optimization and neural computing. *Simulated Annealing and Boltzmann Machines* (1989).
[2] M. Aupetit and M. Sedlmair. 2016. SepMe: 2002 New visual separation measures. In *IEEE Pacific Visualization Symposium*. 1–8. https://doi.org/10.1109/PACIFICVIS.2016.7465244
[3] Lyn Bartram and Maureen C Stone. 2010. Whisper, don't scream: Grids and transparency. *IEEE Transactions on Visualization and Computer Graphics* 17, 10 (2010), 1444–1458. https://doi.org/10.1109/TVCG.2010.237

[4] J. Chuang, D. Weiskopf, and Torsten Möller. 2009. Energy Aware Color Sets. *Computer Graphics Forum* 28 (2009). https://doi.org/10.1111/j.1467-8659.2009.01359.x

[5] S. De Vito, E. Massera, M. Piga, L. Martinotto, and G. Di Francia. 2008. On field calibration of an electronic nose for benzene estimation in an urban pollution monitoring scenario. *Sensors and Actuators B: Chemical* 129, 2 (2008), 750–757. https://doi.org/10.1016/j.snb.2007.09.060

[6] Hui Fang, Simon Walton, Emily Delahaye, James Harris, DA Storchak, and Min Chen. 2016. Categorical colormap optimization with visualization case studies. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2016), 871–880. https://doi.org/10.1109/TVCG.2016.2599214

[7] Sarah Friedrich, Frank Konietschke, and Markus Pauly. 2017. GFD: An R Package for the Analysis of General Factorial Designs. *Journal of Statistical Software, Code Snippets* 79, 1 (2017), 1–18. https://doi.org/10.18637/jss.v079.c01

[8] M. Gleicher. 2018. Considerations for Visualizing Comparison. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 413–423. https://doi.org/10.1109/TVCG.2017.2744199

[9] C. C. Gramazio, D. H. Laidlaw, and K. B. Schloss. 2017. Colorgorical: creating discriminable and preferable color palettes for information visualization. *IEEE Transactions on Visualization and Computer Graphics* 23, 1 (2017), 521–530. https://doi.org/10.1109/TVCG.2016.2598918

[10] Amy L Griffin and Anthony C Robinson. 2014. Comparing color and leader line highlighting strategies in coordinated view geovisualizations. *IEEE Transactions on Visualization and Computer Graphics* 21, 3 (2014), 339–349. https://doi.org/10.1109/TVCG.2014.2371858

[11] K Wm Hall, Charles Perin, Peter G Kusalik, Carl Gutwin, and Sheelagh Carpendale. 2016. Formalizing emphasis in information visualization. In *Computer Graphics Forum*, Vol. 35. 717–737. https://doi.org/10.1111/cgf.12936

[12] Mark Harrower and Cynthia A. Brewer. 2003. ColorBrewer.org: an online tool for selecting colour schemes for maps. *The Cartographic Journal* 40, 1 (2003), 27–37. https://doi.org/10.1179/000870403235002042

[13] Jeffrey Heer and Maureen Stone. 2012. Color naming models for color selection, image editing and palette design. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1007–1016. https://doi.org/10.1145/2207676.2208547

[14] L. Itti, C. Koch, and E. Niebur. 1998. A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 20, 11 (1998), 1254–1259. https://doi.org/10.1109/34.730558

[15] Hye-Rin Kim, Min-Joon Yoo, Henry Kang, and In-Kwon Lee. 2014. Perceptually-Based Color Assignment. *Computer Graphics Forum* 33, 7 (2014), 309–318. https://doi.org/10.1111/cgf.12499

[16] Yann LeCun, Corinna Cortes, and CJ Burges. 2010. MNIST handwritten digit database. *ATT Labs [Online]. Available: http://yann.lecun.com/exdb/mnist* 2 (2010).

[17] S. Lee, M. Sips, and H. Seidel. 2013. Perceptually Driven Visibility Optimization for Categorical Data Visualization. *IEEE Transactions on Visualization and Computer Graphics* 19, 10 (2013), 1746–1757. https://doi.org/10.1109/TVCG.2012.315

[18] Jie Liang and Mao Lin Huang. 2010. Highlighting in information visualization: A survey. In *Proceedings of the International Conference Information Visualisation*. IEEE, 79–85. https://doi.org/10.1109/IV.2010.21

[19] Sharon Lin, Julie Fortuna, Chinmay Kulkarni, Maureen Stone, and Jeffrey Heer. 2013. Selecting Semantically-Resonant Colors for Data Visualization. *Computer Graphics Forum* 32, 3 (2013), 401–410. https://doi.org/10.1111/cgf.12127

[20] Tingting Liu, Xiaotong Li, Chen Bao, Michael Correll, Changehe Tu, Oliver Deussen, and Yunhai Wang. 2021. Data-Driven Mark Orientation for Trend Estimation in Scatterplots. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–16. https://doi.org/10.1145/3411764.3445751

[21] K. Lu, M. Feng, X. Chen, M. Sedlmair, O. Deussen, D. Lischinski, Z. Cheng, and Y. Wang. 2021. Palettailor: discriminable colorization for categorical data. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2021), 475–484. https://doi.org/10.1109/TVCG.2020.3030406

[22] G. M. Machado, M. M. Oliveira, and L. A. F. Fernandes. 2009. A Physiologically-based Model for Simulation of Color Vision Deficiency. *IEEE Transactions on Visualization and Computer Graphics* 15, 6 (2009), 1291–1298. https://doi.org/10.1109/TVCG.2009.113

[23] Aristides Mairena, Carl Gutwin, and Andy Cockburn. 2021. Which emphasis technique to use? Perception of emphasis techniques with varying distractors, backgrounds, and visualization types. *Information Visualization* (2021), 14738716211045354. https://doi.org/10.1177/14738716211045354

[24] Tamara Munzner. 2014. *Visualization analysis and design*. CRC press.

[25] Supriya Manikonda Keshavaiah Naidu. 2019. *Measuring Effective Highlight Colors in Color-coded Scatterplots*. Ph.D. Dissertation. University of Colorado at Boulder.

[26] Pascal Nardini, Min Chen, Michael Böttinger, Gerik Scheuermann, and Roxana Bujack. 2021. Automatic improvement of continuous colormaps in Euclidean colorspaces. *Computer Graphics Forum* 40, 3 (2021), 361–373. https://doi.org/10.1111/cgf.14313

[27] Pascal Nardini, Min Chen, Roxana Bujack, M Bottinger, and Gerik Scheuermann. 2020. A testing environment for continuous colormaps. *IEEE Transactions on Visualization and Computer Graphics* 27, 2 (2020), 1043–1053. https://doi.org/10.1109/tvcg.2020.3028955

[28] B. Ondov, N. Jardine, N. Elmqvist, and S. Franconeri. 2019. Face to face: evaluating visual comparison. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 861–871. https://doi.org/10.1109/TVCG.2018.2864884

[29] Zening Qu and Jessica Hullman. 2017. Keeping multiple views consistent: constraints, validations, and exceptions in visualization authoring. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2017), 468–477. https://doi.org/10.1109/TVCG.2017.2744198

[30] Khairi Reda, Amey A Salvi, Jack Gray, and Michael E Papka. 2021. Color nameability predicts inference accuracy in spatial visualizations. In *Computer Graphics Forum*, Vol. 40. Wiley Online Library, 49–60.

[31] Khairi Reda and Danielle Albers Szafir. 2020. Rainbows revisited: Modeling effective colormap design for graphical inference. *IEEE transactions on visualization and computer graphics* 27, 2 (2020), 1032–1042.

[32] Anthony C Robinson. 2011. Highlighting in geovisualization. *Cartography and Geographic Information Science* 38, 4 (2011), 373–383. https://doi.org/10.1559/15230406384373

[33] V. Setlur and M. C. Stone. 2016. A Linguistic Approach to Categorical Color Assignment for Data Visualization. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (2016), 698–707. https://doi.org/10.1109/TVCG.2015.2467471

[34] Gaurav Sharma, Wencheng Wu, and Edul N Dalal. 2005. The CIEDE2000 color-difference formula: implementation notes, supplementary test data, and mathematical observations. *Color Research & Application* 30, 1 (2005), 21–30. https://doi.org/10.1002/col.20070

[35] Stephen Smart and Danielle Albers Szafir. 2019. Measuring the separability of shape, size, and color in scatterplots. In *Proceedings of the CHI Conference on Human Factors in Computing Systems*. 1–14. https://doi.org/10.1145/3290605.3300899

[36] Tableau Software. [n.d.]. Highlight Data Points in Context. https://help.tableau.com/current/pro/desktop/en-us/actions_highlight_highlighter.htm/.

[37] Tableau Software. [n.d.]. The tableau visualization system. http://www.tableausoftware.com/.

[38] Hendrik Strobelt, Daniela Oelke, Bum Chul Kwon, Tobias Schreck, and Hanspeter Pfister. 2015. Guidelines for effective usage of text highlighting techniques. *IEEE Transactions on Visualization and Computer Graphics* 22, 1 (2015), 489–498. https://doi.org/10.1109/TVCG.2015.2467759

[39] Danielle Albers Szafir. 2018. Modeling Color Difference for Visualization Design. *IEEE Transactions on Visualization and Computer Graphics* 24, 1 (2018), 392–401. https://doi.org/10.1109/TVCG.2017.2744359

[40] C. Tominski, G. Fuchs, and H. Schumann. 2008. Task-driven color coding. In *Proceedings of 12th International Conference Information Visualisation*. 373–380. https://doi.org/10.1109/IV.2008.24

[41] Manuela Waldner, Alexey Karimov, and Eduard Gröller. 2017. Exploring visual prominence of multi-channel highlighting in visualizations. In *Proceedings of the 33rd Spring Conference on Computer Graphics*. 1–10. https://doi.org/10.1145/3154353.3154369

[42] Yunhai Wang, Xin Chen, Tong Ge, Chen Bao, Michael Sedlmair, Chi-Wing Fu, Oliver Deussen, and Baoquan Chen. 2019. Optimizing color assignment for perception of class separability in multiclass scatterplots. *IEEE Transactions on Visualization and Computer Graphics* 25, 1 (2019), 820–829. https://doi.org/10.1109/TVCG.2018.2864912

[43] Michelle Q. Wang Baldonado, Allison Woodruff, and Allan Kuchinsky. 2000. Guidelines for Using Multiple Views in Information Visualization. In *Proceedings of the Working Conference on Advanced Visual Interfaces*. 110–119. https://doi.org/10.1145/345513.345271

[44] Colin Ware. 2019. *Information visualization: perception for design*. Morgan Kaufmann.

[45] Claus O Wilke. 2019. *Fundamentals of data visualization: a primer on making informative and compelling figures*. O'Reilly Media.

[46] L. Zhou and C. D. Hansen. 2016. A Survey of Colormaps in Visualization. *IEEE Transactions on Visualization and Computer Graphics* 22, 8 (2016), 2051–2069. https://doi.org/10.1109/TVCG.2015.2489649