Optimizing Color Assignment for Perception of Class Separability in Multiclass Scatterplots

Yunhai Wang, Xin Chen, Tong Ge, Chen Bao,
Michael Sedlmair, Chi-Wing Fu, Oliver Deussen and Baoquan Chen

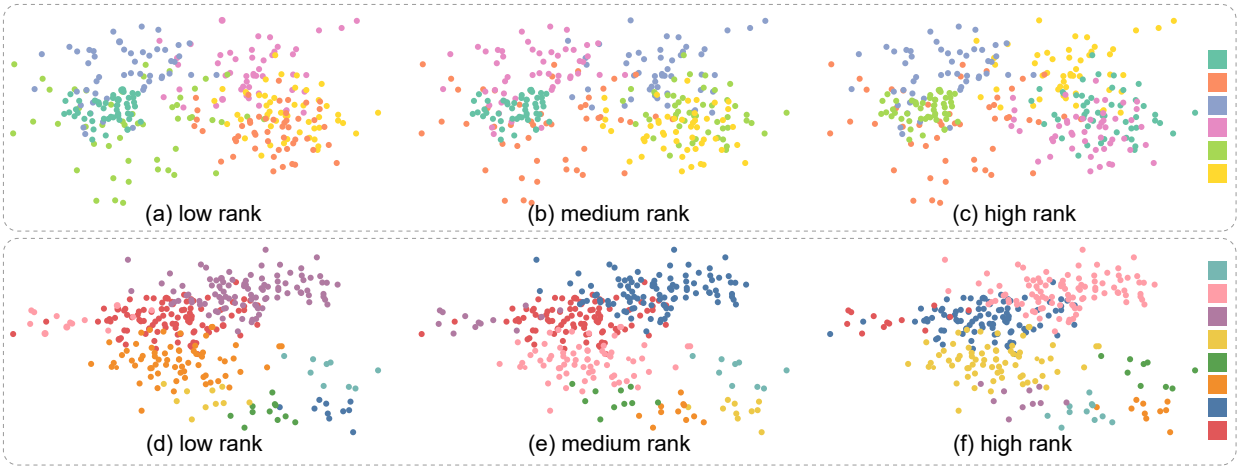

Fig. 1. The color assignment has a strong influence on the visual separability of class structures shown in multiclass scatterplots. Here we show synthetically labeled six-class (top row) and eight-class (bottom row) datasets, and color palettes from ColorBrewer [16] (a,b,c) and Tableau [42] (d,e,f). Scatterplots are displayed from left to right using color assignments produced by our method with the lowest (i.e., poor) scores ranked by our separation measure (a,d) towards the one with the highest (i.e., best) scores (c,f).

Abstract—Appropriate choice of colors significantly aids viewers in understanding the structures in multiclass scatterplots and becomes more important with a growing number of data points and groups. An appropriate color mapping is also an important parameter for the creation of an aesthetically pleasing scatterplot. Currently, users of visualization software routinely rely on color mappings that have been pre-defined by the software. A default color mapping, however, cannot ensure an optimal perceptual separability between groups, and sometimes may even lead to a misinterpretation of the data. In this paper, we present an effective approach for color assignment based on a set of given colors that is designed to optimize the perception of scatterplots. Our approach takes into account the spatial relationships, density, degree of overlap between point clusters, and also the background color. For this purpose, we use a genetic algorithm that is able to efficiently find good color assignments. We implemented an interactive color assignment system with three extensions of the basic method that incorporates top K suggestions, user-defined color subsets, and classes of interest for the optimization. To demonstrate the effectiveness of our assignment technique, we conducted a numerical study and a controlled user study to compare our approach with default color assignments; our findings were verified by two expert studies. The results show that our approach is able to support users in distinguishing cluster numbers faster and more precisely than default assignment methods.

Index Terms—Color perception, visual design, scatterplots.

1 Introduction

Scatterplots are one of the most commonly used visualization techniques for displaying two quantitative variables [8]. Encoding data points by the position of a visual mark (e.g. a dot) allows users to reveal a number of different relationships in the data, such as correlation, clusters shape and separation and other properties [29]. Multiclass scatterplots have been shown to be effective for exploring quantitative data accompanied with class information [35], referred to as *labeled data*. In such plots, the color of the points is typically used for encoding their class label. If the number of dimensions is larger than two, dimensionality reduction techniques [22] have to be used to make data points visible on the plane. For analyzing the data, inspecting the class separation is then a major task for most users [5]. During this task, the user's perception of the point class is heavily affected by the adopted color mapping scheme [14,23]. A good mapping makes class structures clearly visible, while a bad one results in different classes being barely separable, which even might lead to a misinterpretation of the data.

In practice, an appropriate color mapping scheme is often obtained by a two-step procedure: (i) selecting a good palette using a categorical color palette tool (e.g., ColorBrewer [16] or Colorgorical [15]) and (ii) assigning the selected colors to the classes through a trial-and-error process. Our goal is to automate and optimize the second step. Figure 1

- *Y Wang, X. Chen, T. Ge and C. Bao are with Shandong University. Email: {cloudseawang, chenxin199634, getong95, baochen95}@gmail.com.*
- *B. Chen is with Peking University. E-mail: baoquan.chen@gmail.com.*
- *C.-W. Fu is with the Chinese University of Hong Kong. E-mail: cwfu@cse.cuhk.edu.hk.*
- *M. Sedlmair is with VISUS, University of Stuttgart, Germany. E-mail: michael.sedlmair@visus.uni-stuttgart.de.*
- *O. Deussen is with Konstanz University, Germany and Shenzhen VisuCA Key Lab, SIAT, China. E-mail: oliver.deussen@uni-konstanz.de.*
- *Y. Wang and X. Chen are joint first authors.*

Manuscript received xx xxx. 201x; accepted xx xxx. 201x. Date of Publication xx xxx. 201x; date of current version xx xxx. 201x. For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org. Digital Object Identifier: xx.xxxx/TVCG.201x.xxxxxxx

shows some good and bad assignment examples generated by using the default Tableau color palette and the ColorBrewer template palette, which are both designed to be perceptually discriminable and legible. An un-optimized color assignment still results in barely discriminable class structures (see Figures 1(a,b,d,e)), while an optimized color assignment makes almost all the classes clearly visible (see Figures 1(c,f)). The few color assignment strategies that exist for scatterplots [24, 37] assume all classes to be associated with semantics and use these semantics to generate meaningful colors. In most cases, however, data sets do not come with such class semantics; for instance, when they stem from clustering algorithms [5]. In other cases, semantics cannot be easily associated with colors. To the best of our knowledge, there are no methods at present that allows a proper assignment of colors for general multiclass scatterplots.

To fill this gap, we formulate an optimization approach to *automatically generate a color assignment that maximizes the perceptual separability between classes*. To do so, we extend state-of-the-art visual class separation measures [1, 40] to incorporate color factors to model human perception of multiclass scatterplots, and use these measures to guide the automatic search of proper color assignments. A variety of visual separation measures exist, which imitate human perception in multiclass scatterplots. Almost all existing measures, however, do not consider the color assignment factors. Thus, their measured separation values might not align with human perception, if the scatterplot is visualized using an improper color mapping scheme. Following existing color design guidelines [48], we extend these measures by incorporating color factors for visual class separation by quantifying (i) the color difference between neighboring classes and (ii) the color contrast between each class and the background. By integrating these two factors into state-of-the-art class separation measures [1], our measure resembles human class separation judgments for color-coded multiclass scatterplots quite well, as will be shown below.

One straightforward method for finding an optimal color assignment is to evaluate all possible solutions and then rank the assignments by the score of our perceptual separation measure. When n is small, this method is feasible, but there is an exponential increase of required computational costs with an increase of n. To address this issue, our approach uses a customized genetic algorithm [27] to efficiently search for a color assignment scheme by maximizing the proposed class separation measure. Using this approach, we are able to deal with scatterplots having 15 classes in less than 2.5 seconds.

We evaluated our approach using 27 multiclass scatterplots [4] and quantitatively measured the quality of our results using Lee et al.'s class visibility measure [23]. For the tested datasets, our method is capable of producing results with high class visibility. We also ran two user studies: the first study investigated how our selected color assignment helps users estimate class separation. The second one aimed at learning if users subjectively prefer our results, and if yes, why they prefer them. Both studies confirmed the capability of our method to produce results that align well with human perceptual judgments.

We furthermore present an interactive color assignment system with three extensions of our methods for the interactive exploration of multiclass scatterplots. First, our system is able to suggest a set of good and diverse color assignments for the user to select the preferred one. Second, in some cases, the user might prefer specific colors for some classes, and accordingly, we extend our genetic algorithm to satisfy such user-provided constraints. Lastly, when the user is interested in some specific classes, our approach allows the generation of a color assignment that maximizes the class separation for such classes.

In summary, the main contributions of this paper are:

- We formulate an optimization approach to automatically generate a color assignment based on (i) incorporating color factors into state-of-the-art visual separation measures, and (ii) devising a customized genetic algorithm to rapidly generate proper color assignments (Section 4).
- We quantitatively evaluate the resulting color assignments using a class visibility measure [23], and conduct two user studies to show the usefulness of our approach (Section 5).
- We present three extensions that show how our method can be used to help the exploration of multiclass scatterplots (Section 6).

2 Related Work

Existing related work can be divided into two categories: visual class separation measures and color design in visualization.

2.1 Visual Class Separation Measures

Scatterplots support a number of different analysis tasks such as correlation estimation and object clustering [33]. As mentioned above, for multiclass scatterplots, the main task is to investigate the visual separability of classes in labeled data [5]. Sedlmair et al. [36] developed a taxonomy of factors that influence the human perception of visual class separation, where most factors are derived from the positions of the data points. They suggested that the design of reliable separability measures should be guided by this taxonomy.

By combining different factors, some visual separation measures have been proposed in the past. Distance Consistency [40] defines the class separability as the proportion of data points whose closest *class center* has the same class label. The Class Density and Histogram Density measures [43] are based on *class density*, which is described by a density image and a histogram. Distribution Consistency [40] is also based on class density but it computes the proportion of nearest data points with the same class label directly in data space. Aupetit and Sedlmair [1] generalized such neighborhood approaches by factorizing them into two aspects: neighborhood graphs and class purity functions. By combining 143 neighborhood graphs and 14 class purity functions, they proposed a set of 2002 new visual separation measures.

In the machine learning community, there are some separation measures developed for the qualitative evaluation of clustering and classification algorithms, such as the Silhouette Index [32], Fisher's discriminant ratio [18], and Dunn's index [11]. All these measures are also based on the factors summarized in the taxonomy of Sedlmair et al. [36], although they are not intended to work in visual space. Therefore, using them for measuring visual class separation might not well align with human judgment.

To learn how well the measures predict human judgments, Sedlmair and Aupetit [34] proposed a machine learning framework. They found that Distance Consistency (DSC) is better than others but its accuracy is still not perfect. Using their proposed 2002 new measures [1], a large scale evaluation was conducted using the same framework. The results showed that their proposed "0.35-Observable Neighbors of each point of the target class" (GONG) performs the best, much better than DSC. Meanwhile, they found that the "average Class-Proportion of the 2-Nearest-Neighbors of each point in the target class" (KNNG) performs slightly worse than GONG, but has a much lower computational cost.

Recently, Wang et al. [47] extended KNNG by incorporating the density information and used it to achieve a perception-driven dimensionality reduction (DR) technique, which performs better than the state-of-the-art DR methods. Likewise, our work also extends KNNG but with an additional factor, *color*, which is an essential element in visualization but overlooked by the current taxonomy [36].

2.2 Color Design in Visualization

Color is one of the most commonly used visual channels. Creating an appropriate color map for visualization has attracted much attention. A complete review of color map design is beyond the scope of this paper; we refer the reader to Silva et al. [39] and Zhou and Hansen [50]. Therefore, we restrict our discussion to techniques for finding categorical color maps for visualizing labeled data.

Color palette creation. Creating a categorical color palette for maximizing visual discrimination between classes is a demanding task for the visualization of labeled data [45, 48]. A few guidelines for the manual design of color palettes have been provided in the past, such as "color should be well separated" [45] or "colors should cooperate with each other" [49]. However, most visualization creators would like to avoid creating palettes from scratch.

Also a few automatic or semi-automatic tools have been provided. Bergman et al. [3] developed a rule-based approach that uses the varying sensitivity of the human visual system for spatial frequencies as a basic rule for creating color palettes. Healey [17] proposed to create palettes by using colors named with the ten Munsell hues that maximize the perceptual distance between colors. Maxwell [26] further considered the spatial characteristics of classes to create color palettes for multivariate data. Harrower and Brewer [16] developed ColorBrewer, a widely used tool, which provides a large number of pre-defined well-discriminable color palettes. Recently, Gramazio et al. [15] proposed Colorgorical, a tool that not only incorporates aesthetics for color palette creation but also allows users to customize palettes. Our work assumes that a high-quality palette was already selected (or designed) for a labeled data visualization, such as the ones provided by the Tableau palette library [42] and by ColorBrewer [16]. Our work then focuses on optimally assigning these colors to a multiclass scatterplot.

Color palette optimization. A selected color palette might further be optimized for color harmony [46], energy consumption [7], class visibility [23], and perceptual distance [12]. The last two tasks are the closest to our work, which attempt to optimize the class discrimination.

Class visibility [23] is defined by the perceptual intensity of a class, and the perceptual intensity is based on the saliency of each point against its neighborhood. Based on the class visibility, Lee et al. [23] presented a method that perceptually optimizes a given color palette to better reveal all the class structures. Similar to this class visibility method, our proposed color-based visual separation method also considers the spatial distribution of each class, but it additionally incorporates the color contrast against the background, which could heavily influence the perception of class structures [48]. Second, the class visibility does not consider the class density, whereas our measured color contrast with the background is weighted by the class density. Last, the class visibility is defined on the pixel, thus it might not accurately characterize the data patterns. In contrast, our method works with the data points in multiclass scatterplots of many intertwined points instead of just maps and focuses on the search of good color assignments for improving the visual separation between classes.

Rather than optimizing colors for a specific visualization, Fang et al. [12] proposed to maximize the perceptual distance among a set of given colors while incorporating a set of user-defined constraints. They further compared three optimization algorithms in solving this problem and found that a Genetic Algorithm (GA) can alleviate the issue of sticking to the local maxima. Similar to this work, our color optimization framework also enables users to define constraints and uses GA to solve the optimization. Moreover, we show that our color assignment optimization can improve the visualization produced by this method in terms of class discrimination (see Section 5).

Color assignment. Given a target color palette and a set of class labels, color assignment aims to find a unique optimal color for each class, which has not been extensively studied so far. Most methods focus on associating colors with semantics, which are modeled by collecting representative images from Google Image Search. Lin et al. [24] proposed a method to select such semantically resonant colors and experimentally demonstrated the benefits of this method. Setlur et al. [37] improved this method by using a co-occurrence measure of color name frequencies from Google n-grams and showed better results than Lin et al. [24]. Most classes shown in scatterplots, however, might not have clear semantics, especially the ones generated by clustering algorithms or customizable tools, thus such method cannot be directly applied to general multiclass scatterplots.

It should be noted that we are not the first to work on color assignment without semantics in visualization and computer graphics. Hurter et al. [19] proposed an optimization method for assigning colors to lines of a metro map that assigns close routes with the most distinguishable colors. Kim et al. [21] proposed a perception-driven color assignment method for assigning colors to unordered image segments, where color aesthetics as well as contrast are incorporated. Our method can be regarded as a task-driven color coding [44], where our task is to maximize the perceived class separation in the scatterplots.

3 Preliminaries: Formal Definitions

In this section, we provide formal definitions of some components of our approach. We start by describing a state-of-the-art separation measure and then introduce the color factors that influence the separability of color-coded classes. In general we suppose to have a multiclass (m classes) scatterplot with 2D data points $\mathbf{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_n\}$, where each $\mathbf{x}_i$ is associated with a class label $l(\mathbf{x}_i)$ and the j-th class (with n_j data points) consists of $\{\mathbf{x}_1^j, \cdots, \mathbf{x}_{n_j}^j\}$, $j \in \{1, \cdots, m\}$.

3.1 Class Separation Measure: KNNG

Aupetit and Sedlmair [1] proposed the above-mentioned KNNG measure and showed that it performs slightly worse than the best state-of-the-art measure but has lower computation complexity. Because of this characteristic, Wang et al. [47] extended the measure to guide the process of dimensionality reduction. Our method can be regarded as a further extension of KNNG.

KNNG is built upon a k-nearest neighbor graph with $k = 2$, where each data point $\mathbf{x}_i$ is connected to its two nearest neighbors, denoted as $\Omega(\mathbf{x}_i)$. For each $\mathbf{x}_i$, we compute its separation degree as

$$\mathbf{s}(\mathbf{x}_i) = \frac{1}{|\Omega(\mathbf{x}_i)|} \sum_{\mathbf{x}_j \in \Omega(\mathbf{x}_i)} \delta(l(\mathbf{x}_i), l(\mathbf{x}_j)) \,, \tag{1}$$

where $\delta(l(\mathbf{x}_i), l(\mathbf{x}_j))$ returns one if $\mathbf{x}_i$ and $\mathbf{x}_j$ have the same class label, else zero. The final KNNG value is then the average separation degree over all data points of the same class in relation to the entire dataset. Since finding $\Omega(\mathbf{x}_i)$ for a data point requires us to test its nearby data points in $\mathbf{X} \backslash \{\mathbf{x}_i\}$, the overall time complexity is $O(n \log n)$, which is feasible for most applications.

3.2 Color-based Separation Factors

Sedlmair et al. [36] created a taxonomy of visual class separation factors in scatterplots, which guided the recent development of class separation measures [1, 47]. Almost all existing measures that include KNNG [1, 40, 43] are purely based on the position of the data points, whereas the color associated with the data points is completely overlooked. However, human judgments for color-coded labeled data are influenced by a number of color factors, as summarized by Ware [48]. Here, we only briefly review two factors: *distinctness* and *contrast with the background*, which are most related to the design of color palettes for multiclass scatterplots.

Distinctness is one of the key factors in the human saliency detection process [13], referring to how good an object can be discriminated from others. To achieve distinctness, Ware [48] suggested that choosing a color set should consider the separation not only between the colors themselves but also between the colors and the background, the marker size as well as the distribution of the data points [33]. Margolin et al. [25] defined the color distinctness of a data point as its color difference with the neighboring points, where the color difference can be measured by many metrics, such as the Euclidean RGB distance or by the CIE76, CIE94 and CIEDE2000 distance measures [6, 38].

Contrast with background is an essential perceptual factor that dramatically influences the readability of color-coded objects. For example, the yellow class in Figure 1(d) is hard to be recognized but is clearly shown in Figure 1(f). By measuring the color difference with the background in terms of luminance [20], Kim et al. [21] integrated this factor to optimize color assignments for showing image segments. Likewise, we also include this factor in our optimization for color assignment.

Although other factors such as unique hues, color blindness and cultural conventions might also influence the perception of multiclass scatterplots, they either should be considered by the design of the color palettes or are not of general interest. Thus, we base our color assignment optimization on the above two factors.

4 Class Separability Driven Color Assignment

To visualize a set of labeled 2D data points $\mathbf{X}$ of m classes $\mathbf{M} = \{1, \cdots, m\}$ in a scatterplot, with a given color palette $\mathbf{C} =$

$\{C_1,\ldots,C_p\}$ $(p \geq m)$ and a background color C_b, we need a mapping $\tau : \mathbf{M} \mapsto \mathbf{C}$ that assigns the colors to classes. Most existing visualization tools like Tableau [42] assume that classes as well as colors are ordered and assign colors simply by following the order. However, most multiclass datasets shown in scatterplots do not inherently contain such ordering information, so the assignment is just a random order based on the point ordering in the data file. This might not produce effective visualizations. Figures 1(b,d) are generated in this way. It is obvious that some classes have a poor separation from the rest.

To address this issue, we first introduce some color-based class separation measures that seek to imitate human perception of class separability in color-coded multiclass scatterplots. Based on such measures, we formulate color assignment as an optimization problem, by which we seek a color mapping that makes all the classes in the scatterplot easily recognizable to humans.

4.1 Class Separation guided Color Assignment

As reviewed in Section 3.2, we assume that distinctness and contrast with the background are the two main factors in the design of a proper color mapping. In the case of a multiclass scatterplot, each class will have its own spatial distribution, which should also be considered. To achieve this goal, we first construct a k-nearest neighbor graph and then compute these two factors for each data point based on its local neighborhood.

Point distinctness. Suppose the set of k-nearest neighbors of data point $\mathbf{x}_i$ is Ω_i, the color distinctness of $\mathbf{x}_i$ under the color mapping τ is:

$$\alpha(\mathbf{x}_i) \;=\; \frac{1}{|\Omega_i|}\sum_{\mathbf{x}_j\in\Omega_i} \Delta\varepsilon(C_r,C_s)\; g(d(\mathbf{x}_i,\mathbf{x}_j))\;, \tag{2}$$

where $C_r = \tau(l(\mathbf{x}_i))$, $C_s = \tau(l(\mathbf{x}_j))$, $\Delta\varepsilon$ is the CIEDE2000 distance metric [38], and $g(d(\mathbf{x}_i,\mathbf{x}_j))$ is a distance-based function to assign large weights to nearby points and small weights to far-away points. Thus, it can be regarded as the degree of influence of point $\mathbf{x}_j$ to point $\mathbf{x}_i$, an appropriate function is $g(d) = 1/d$. A good color assignment aims to assign colors, so that nearby points have larger color differences than points that are far away. Note that if we set $g(d) = 1$, this measure is equivalent to the point saliency, which is the base of the class visibility [23]. The larger the $\alpha(\mathbf{x}_i)$, the larger the point distinctness is.

Point contrast with background. Other than examining the color difference with the background, the point's contrast also depends on the spatial distribution of the neighboring points [30]. If most points in Ω_i are close to $\mathbf{x}_i$ and have the same label, $\mathbf{x}_i$ should have a large separation degree (see Eq. (1)); see the points in Figure 2(a). However, if most points in Ω have different labels, $l(\mathbf{x}_i)$ would become hard to be identified; see the examples in Figure 2(b). To maximize the visibility of all classes, points with lower class separability degrees should be assigned with colors that have larger contrast to the background. Hence, we define the point contrast as

$$\beta(\mathbf{x}_i) \;=\; \frac{1}{|\Omega_i|}\sum_{\mathbf{x}_j\in\Omega_i} \Delta L(C_r,C_b)\; ns(\mathbf{x}_i)\;, \tag{3}$$

where $\Delta L(C_r,C_b)$ is the luminance difference between the assigned point color $C_r = \tau(l(\mathbf{x}_i))$ and the background color C_b, and $ns(\mathbf{x}_i)$ is the position-based *non-separability* degree. A class's non-separability degree can be regarded as the reverse of the separability degree. Classes with larger separability degrees should have smaller non-separability degrees, and vice versa. Below, we explore how to define non-separability, which is an essential part in $\beta(\mathbf{x}_i)$ above.

Non-separability. Although existing class separation measures such as KNNG can help compute such non-separability degrees, most of them are based on a class purity function (see Eq. (1)) without considering any density information. Figure 2(b) shows an example that illustrates this drawback of KNNG. To address this issue, we incorporate the distance between $\mathbf{x}_i$ and its neighboring points into Eq. (1) and

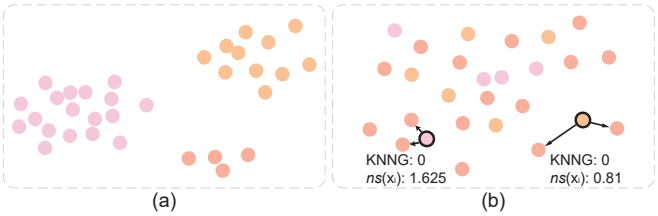

Fig. 2. Illustration of the separation between color-coded classes, where the same color mapping is assigned to classes with different class separability degrees. (a) Classes with large separation degrees can easily be separated although their assigned colors are similar; (b) Classes with poor separation degrees can hardly be separated although their assigned colors are the same as in (a). The points with black circles illustrate the drawback of KNNG in characterizing the separation degree.

compute the within-class separation degree by measuring the compactness of $\mathbf{x}_i$ and other points belonging to the same class as

$$a(\mathbf{x}_i) = \frac{1}{|\Omega_i|}\sum_{\mathbf{x}_j\in\Omega_i} \delta(l(\mathbf{x}_i),l(\mathbf{x}_j))\; g(d(\mathbf{x}_i,\mathbf{x}_j))\;, \tag{4}$$

and based on that calculate the non-separability degree between $\mathbf{x}_i$ and the data points of other classes by

$$b(\mathbf{x}_i) = \frac{1}{|\Omega_i|}\sum_{\mathbf{x}_j\in\Omega_i} \big(1-\delta\left(l(\mathbf{x}_i),l(\mathbf{x}_j)\right)\big)\; g(d(\mathbf{x}_i,\mathbf{x}_j))\;. \tag{5}$$

Then, the non-separability of the point $\mathbf{x}_i$ is defined as

$$ns(\mathbf{x}_i) = b(\mathbf{x}_i) - a(\mathbf{x}_i)\;. \tag{6}$$

Hence, a negative $ns(\mathbf{x}_i)$ indicates that most neighbor points have the same label as $\mathbf{x}_i$, and vice versa. In other words, $ns(\mathbf{x}_i)$ performs similarly to density-aware KNNG [47] in characterizing class distribution, but it reflects the non-separability with the un-normalized values, so that the density between points is more accurately described.

Objective function. Based on point distinctness and contrast with the background, we can now define our objective function as maximizing the sum of the separation degree of all the points in each class:

$$\arg\max_{\tau} E(\tau) \;=\; \sum_{j=1}^{m}\sum_{i=1..n_j} \{\; \lambda\,\alpha(\mathbf{x}_i^j) \;+\; (1-\lambda)\beta(\mathbf{x}_i^j)\;\}\;, \tag{7}$$

where λ is a weight parameter to balance the two factors. For a proper choice of λ, we refer readers to Section 4.3.

4.2 Optimization by using a Genetic Algorithm

For a palette of p colors and a scatterplot of m $(m \leq p)$ classes, there are $p!/(p-m)!$ color assignment choices. Just for $m = p = 10$, we already have more than three million possible assignment choices. To find the optimal assignment with the maximal energy with respect to Eq. (7) in such huge search space, we use a genetic algorithm [27] which is especially suitable for finding a near optimal solution for combinatorial optimization problems with a large parameter space. Compared with another optimization method [31], GA can produce more diverse solutions, so it is more likely to find the global optimum.

By representing each candidate τ as a *genome*, our genetic algorithm performs the search in a heuristic way. Since the color assignment for multiclass scatterplots requires each class to have a unique color, the genome we work with in essence is a table that assigns such a unique number of a color to each bin (class). Each color assignment is a permutation of these numbers. After generating a number of random permutations as the initial *population*, our algorithm evaluates each solution by using Eq. (7) and then iteratively improves the solution through performing steps of *selection*, *crossover*, and *mutation* until it converges, as outlined in Algorithm 1.

Algorithm 1 Genetic Algorithm for optimizing color assignment

Input: An initial population $P = \{\tau_1, ..., \tau_s\}$, each τ_i being a color assignment solution
Output: The fittest individual τ'_s

1: **repeat**
2: Perform selection on P
3: Perform crossover on P
4: Perform mutation on P
5: **until** the fitness of the fittest individual cannot be improved or reaching the maximum iteration
6: **return** τ'_s

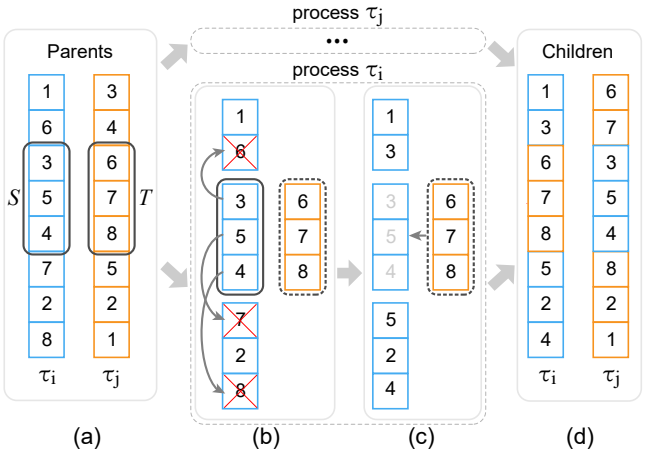

Fig. 3. Illustration of the crossover process: (a) Two genomes τ_i and τ_j with segments (boxed in black) are to be exchanged. (b) To replace segment $\{3,5,4\}$ in genome i by $\{6,7,8\}$ from genome j, we first remove any 6,7,8 color number from i, and then (c) assign 3,5,4 randomly to the locations where we have removed 6,7,8. (d) After that, we put $\{6,7,8\}$ into the genome i in the child population.

Selection. The idea here is to select individuals with high fitness scores from the existing population and use them to breed a new generation. To balance between "exploitation" and "exploration," many different selection methods have been developed [2]. Here, we use the method of a *roulette wheel* selection, which each time randomly selects an individual with a sufficiently high fitness score.

Crossover. This is a significant step in our genetic algorithm. With a certain probability, we combine two individuals to produce new offsprings. We perform the crossover by using a *two-point crossover* method, which selects a point on the genome and then exchanges b consecutive genes between the two genomes. However, when doing so, we have to ensure that the color permutation in each genome is only shuffled and any color number still appears only once.

Suppose we have genomes τ_i and τ_j, each with a segment of b bins, say $S = \{s_1, \ldots, s_b\}$ in τ_i and $T = \{t_1, \ldots, t_b\}$ in τ_j, to be exchanged in the crossover. We take the following steps to perform the operation: without loss of generality, we describe the steps to process τ_i, since τ_j can be processed in the same way (see the illustration in Figure 3): (i) find the bins with colors (short for color numbers) in τ_i that contain a color in T and randomly replace them with colors in S, and (ii) then replace the colors of S with the colors of T.

Mutation. To increase the diversity within a population and to avoid being trapped in local optima, GA performs mutations on some randomly-selected genomes from time to time. When an individual is selected for a mutation, the genes at two randomly selected positions are simply swapped.

GA parameters. For a quick convergence, it is important to appropriately set the algorithm's parameters: population size, crossover and mutation rate. There are, however, no general guidelines for setting up these parameters for different situations. Following the empirical suggestions of Jong et al. [10], we set the population size to 50, crossover rate to 0.6 and mutation rate to 0.01. These values are assumed to be the best for most GA applications.

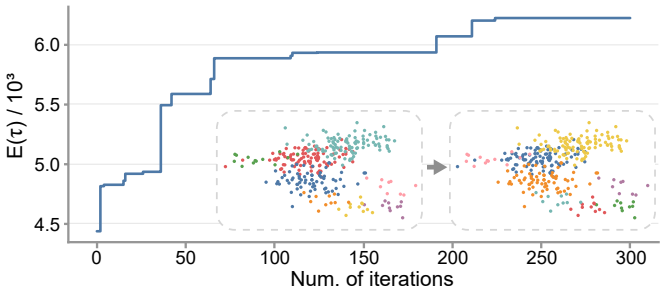

Fig. 4. Value of the objective function E(τ) (Eq. (7)) versus the number of iterations during the genetic optimization. The process converges after 300 iterations.

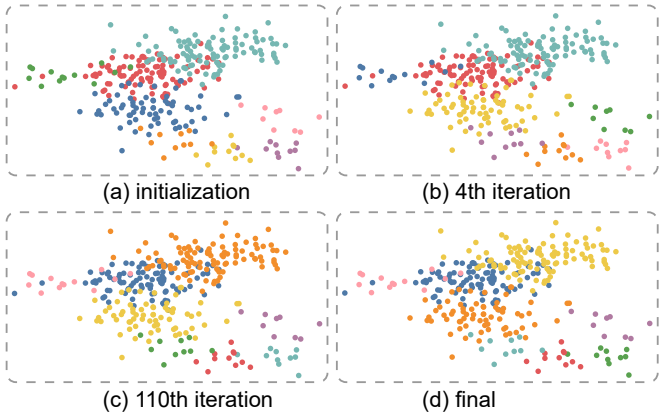

Fig. 5. Exploring the convergence of our genetic algorithm: (a) result after the random initialization; (b) result after 4 iterations; (c) result after 110 iterations; and (d) final result after 280 iterations.

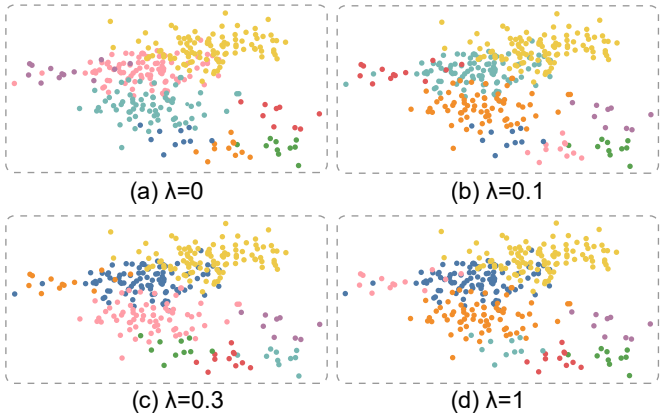

Fig. 6. Exploring the influence of λ on the selected color assignment: (a) result generated by only considering the color contrast with background; (b) result generated with λ set to 0.1; (c) result generated with λ set to 0.3; and (d) result generated by considering only the point distinctness.

Figure 4 shows the convergence curve. Our method converges to a reasonable solution, but then jumps through a number of smaller and bigger steps towards the final value. This is a reasonable behavior, since mutations help us not to stuck in a local optimum. Figure 5 confirms this by showing intermediate results of the optimization for a multiclass scatterplot, the result of the 110th iteration already shows a clearly visible class separation.

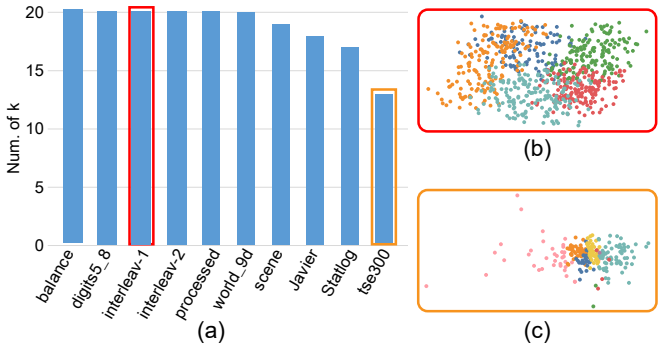

Fig. 7. Exploring the sensitivity of our objective to parameter k. (a) The bar chart shows the consistency of the selected color assignment with different k, where the selected color assignment has high consistency for most data; (b,c) two example scatterplots corresponding to the datasets highlighted in red and yellow boxes (a,b).

4.3 Parameter Study

Our algorithm comes with two free parameters that need to be set: weight (λ) and the number of nearest neighbors (k).

Weight λ. Figure 6 illustrates the influence of λ on the selected color assignment. Considering only the color contrast with the background might lead to some neighboring classes with small color differences, such as the blue and cyan classes in Figure 6(a). Similarly, considering only the distinctness might lead to some non-separable classes that are assigned with low contrast colors to the background, such as the pink class in Figure 6(d). Thus, finding a good λ to balance these two terms is very important. We found that $\lambda = 0.3$ works well for most data in our experiment; see an example in Figure 6(c).

Number of neighbors k. The original KNNG is defined on the k nearest neighborhood graph with $k = 2$, but our objective function can be defined for any k nearest neighborhood graph. To understand the sensitivity of our objective function to k, we randomly selected ten datasets, constructed the graphs with a k ranging from one to twenty for each dataset, and computed the best color assignment using each of the graphs. Based on the selected color assignments, we computed the number of same color assignments selected by different k ranging from one to twenty, referred to as consistency. Figure 7 (a) shows results using a bar chart, for six datasets our method selects the same color assignment no matter what k was chosen; the consistency value is larger than 75% for the other cases. To show why some data has high or low consistency, two selected data sets are shown in Figures 7 (b) and (c). We can see that the one with lower consistency corresponds to data with a large variation of class densities. Since a larger k demands more computation in the construction of the nearest neighborhood graph, we set k to two in our experiments as in the original KNNG.

4.4 Implementation & Performance

We implemented our method using JavaScript (see code in supplemental material) and tested it on a computer with an Intel Core i5-7400 processor with 8GB memory. The k nearest neighborhood graph is constructed by using the FLANN library [28]. To support the interactive search of the optimal color assignment, we decompose the computation of color distances and class non-separability degrees and the GA algorithm. The performance of GA algorithm heavily depends on the number of classes, namely, the length of each genome, rather than the number of data points in each scatterplot, as to be shown later in Figure 8(c).

5 Evaluation

To confirm that our method resembles the human perception of class separation, we evaluated the quality of its selected color assignments by: (i) judging their quality with an existing numeric measure [23]; (ii) conducting a lab study to verify that they can improve human class separation judgments and measure how close they match the user preferences; and (iii) comparing with expert-chosen color assignments.

5.1 Evaluation with Numerical Measures

To perform a quantitative evaluation, we took 27 multiclass scatterplots of *real* datasets gathered from the UCI repository [4]. For visual encodings, we took the Tableau 20 default palette [42] with white background as the input. We run our method with default parameters $k = 2$ and $\lambda = 0.3$.

Measure. We computed the quality of our results by using Kim et al.'s class visibility measure [23]. For this measure, the quality is defined on the whole scatterplot and thus we take the sum of the visibility of all classes. Although directly comparing values of our measure and class visibility [23] does not say much, the relative difference of the quality measures generated for the best, medium, and worst color assignments are comparable. Hence, we score all possible color assignments with our measure and pick the best, medium, and worst ones.

Once we have these three color assignments, we can compute the relative differences

$$\mathrm{dE}_y = \frac{E_y - E_{\mathrm{worst}}}{E_{\mathrm{worst}}} \quad \text{and} \quad \mathrm{dV}_y = \frac{V_y - V_{\mathrm{worst}}}{V_{\mathrm{worst}}}, \tag{8}$$

where E_y and V_y refer to our objective values $E(\tau)$ (see Eq. (7)) and class visibility [23], respectively, computed with the best or medium color assignment. Hence, the values of dE_y and dV_y represent the relative distance between the best or medium color assignment and the worst one, according to E_y and V_y, respectively. Although there are several major differences between these two measures as discussed in Section 2.2, we expect that both measures will give consistent quality orderings for the selected color assignments.

Results. Figure 8(a) shows the four variables $\mathrm{dE}_{\mathrm{best}}$, $\mathrm{dE}_{\mathrm{medium}}$, $\mathrm{dV}_{\mathrm{best}}$, and $\mathrm{dV}_{\mathrm{medium}}$ for each dataset, with $\mathrm{dE}_{\mathrm{worst}}$ and $\mathrm{dV}_{\mathrm{worst}}$ being the base. If a value is out of the plot range, we treat it as an outlier and draw a dark halo in the plot to indicate them. The scatterplot shown in Figure 7(b) is an example of such an outlier.

The ranges of $\mathrm{dE}_{\mathrm{best}}$ and $\mathrm{dE}_{\mathrm{medium}}$ are much larger than the ones of $\mathrm{dV}_{\mathrm{best}}$, and $\mathrm{dV}_{\mathrm{medium}}$. For most dataset, the ranking order of $\mathrm{dE}_{\mathrm{best}}$ and $\mathrm{dE}_{\mathrm{medium}}$ and the ones of $\mathrm{dV}_{\mathrm{best}}$ and $\mathrm{dV}_{\mathrm{medium}}$ are consistent, only the two rankings of the *digits5_8* dataset are inconsistent. After carefully investigating this dataset, we found that this dataset has very strong overlap between classes, so that pixel-based class visibility cannot accurately characterize the separation between the classes.

To facilitate the comparison between our measure and class visibility, we summarize the resulting relative difference associated with each kind of color assignments in the boxplot shown in Figure 8(b). It shows that our measure covers a larger range than class visibility, facilitating it to search for the best color assignment.

5.2 Lab Study

As our goal is to optimize color assignments with respect to human perception, it is necessary to test our results with human subjects. We thus run two human-subject studies, including a controlled lab study (this section) and an expert study (next section).

Goals and tasks. The main goal of the lab study is to test the effects of our method on task efficiency and subjective preferences of users. Specifically, we hypothesize that the colorings suggested by our approach will lead to reduced time and errors *(H1)*, as well as fitting to the subjective preferences of the users *(H2)*. According to these goals, the study consists of two parts.

Part 1—Efficiency: we sought to measure the effectiveness of color assignments by asking users to count the number of classes in a scatterplot and to choose one of the several given numerical options. The actual number of classes in our scatterplots ranges from six to eight, and the number of given options ranges from one to twelve. We recorded the *time taken* by each user for each trial, counted their *errors*, and computed the number of mismatch between the actual number of classes

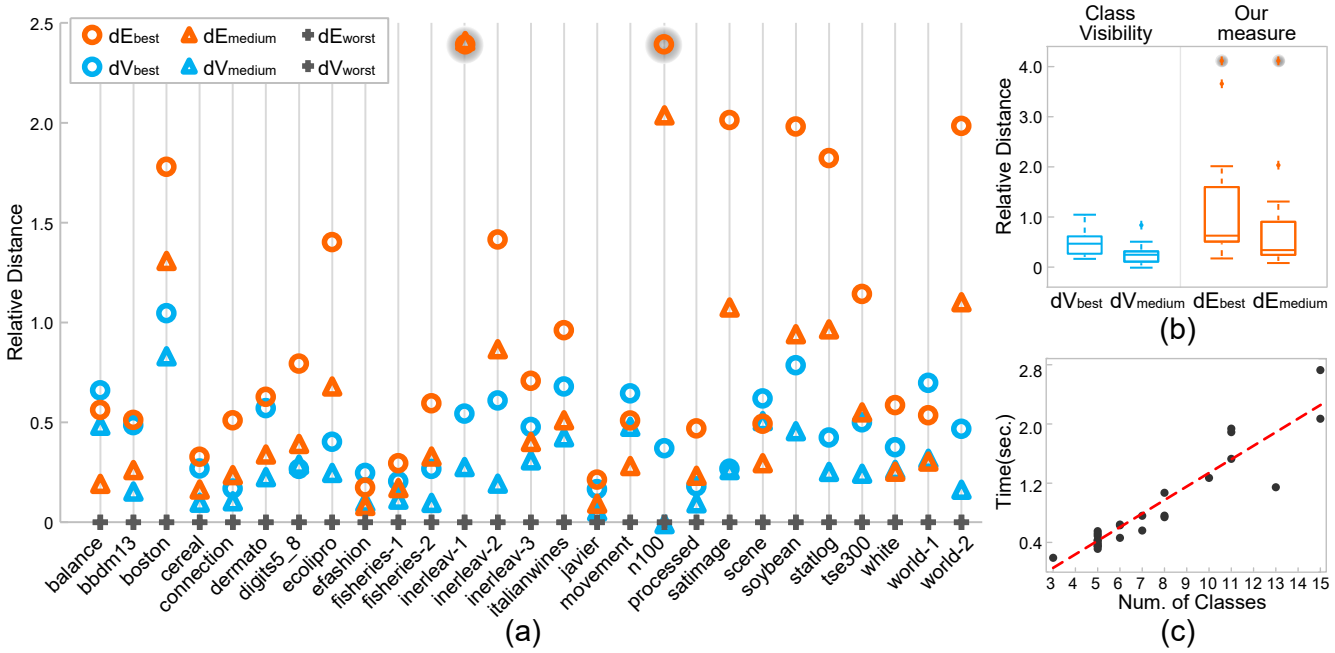

Fig. 8. (a) Comparison of our measure and class visibility using the values of dE_{best}, dE_{medium}, dV_{best}, and dV_{medium} for each dataset; (b) boxplots summarizing the values of the four variables in (a); and (c) scatterplot with a red trend line, showing the relationship between computation time and the number of classes.

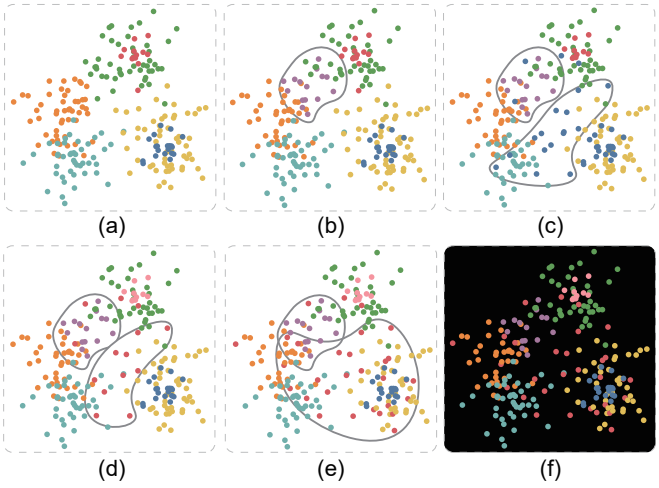

Fig. 9. Illustrating the generation of our test datasets used in part 1 of the user study. (a) The scatterplot with six classes is taken as the base for the creation of the other datasets; (b)-(e) two seven-class and two eight-class scatterplots created by a few randomly-selected points highlighted by lassos from the base (a) as the new classes; and (f) the background color of the scatterplot shown in (e) is changed to black.

and their choice (dependent variables). Figure 9 shows several examples of the scatterplots we used in this task. Each scatterplot stimulus has a different $E(\tau)$ score from our method (independent variable). We expect the participants to spend less time and produce fewer errors on "good" scatterplots (with large $E(\tau)$).

Part 2—Subjective preference: in the second part, we offered users two scatterplots per trial, where one score is larger than the other. Participants were then asked to choose the plot they "perceptually" preferred. We expect that most people would choose the picture with the higher $E(\tau)$ (independent variable), and that there would be only a few neutral selections; see Figure 10 for an example pair.

Pilot studies. We conducted a pilot study involving five students from our university to quickly iterate on our study design. In this study, we

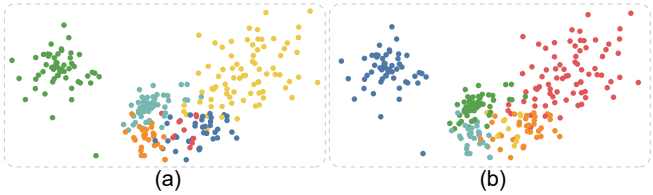

Fig. 10. An example picture for part 2 of the lab study. Users were asked to choose the scatterplot they would "perceptually" preferred from (a) and (b), whose $E(\tau)$ are 2873 and 1021, respectively.

randomly selected five scatterplots used in Section 5.1, each scatterplot was colorized with the best, medium, and worst color assignments. Since there are three kinds of combinations for each scatterplot, i.e., "best" vs. "medium", "medium" vs. "worst", and "best" vs. "worst", we thus have $5(scatterplots) \times 3(combinations) = 15$ pairs for the study in part 2. We showed the 15 scatterplots in random order to each participant to perform part 1 of the pilot studies and then showed the 15 scatterplot pairs in random order to perform part 2. In part 1, we found the errors to be close to zero for almost all scatterplots, while the result in part 2 showed some randomness. We performed a follow-up interview with each participant and asked them why they were able to quickly and accurately count the number of classes in part 1 and why they made a random choice for some scatterplots in part 2.

The answers hinted at two factors that influence the results in part 1: strong learning effect and large point sizes. The learning effect was caused by the five selected scatterplots with different distributions, so that participants could easily remember the number of classes. The large point size, on the other hand, reduced the task difficulty, even for the scatterplots with the worst color assignments. Such findings suggested us to synthesize scatterplots with similar distributions and to assign a proper point size. Regarding part 2, we found that the participants randomly chose one plot, if they found the two plots to be similar. Accordingly, we added the third option "No preference" in our study interface.

Datasets. To reduce the learning effect as found in the pilot studies, we used different datasets in the two parts of our lab study. Hence, we first create a synthetic scatterplot with six classes. Each class followed a

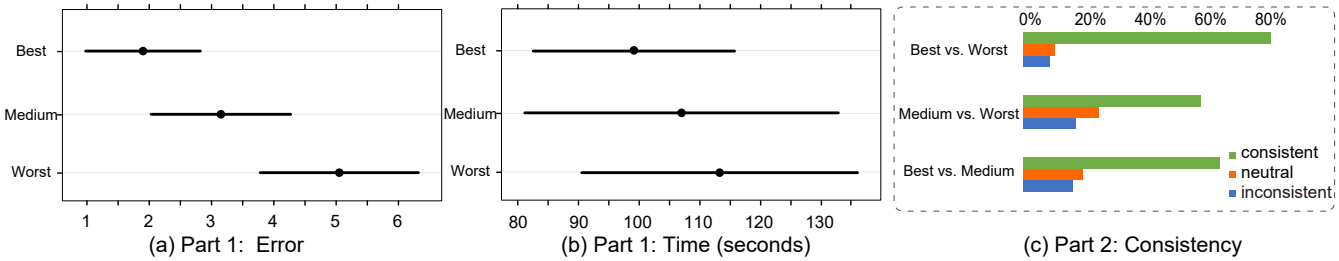

Fig. 11. Results of the lab studies. For part 1, we show mean values and deviation as 95% CIs of (a) user error, (b) time (lower values are better). For part 2, we display the consistency between people's choice and the rating of our method (c). If the user chose the better scatterplot of a pair, the choice was marked with "consistent," otherwise "inconsistent." If they choose "No preference," the result was "neutral."

Gaussian distribution with strong overlaps between some of the classes; see Figure 9(a). Taking this dataset as the base, we created the other four datasets (2x7 classes and 2x8 classes) by randomly selecting some points to form new classes. For example, the dataset with 7 classes shown in Figure 9(b) was created by randomly selecting some points from the orange and green classes and regarding them as a new class. Such points selected from the base were highlighted by a lasso in each set, as shown in Figures 9 (b) to (e). Doing so, we generated five synthetic scatterplots (1x6 classes, 2x7 classes, and 2x8 classes), where all the datasets consistently have 230 data points.

For our color palette, we chose the "Tableau 10" palette from the Tableau software system [42]. Each sample was displayed twice over a white and a black background; see Figure 9(f). Each picture was colored with three types of color assignments: large $E(\tau)$, median $E(\tau)$, and small $E(\tau)$; see Figures 9 (a) to (e). The total number of trials was $5(scatterplots) \times 2(backgrounds) \times 3(cases) = 30$. All the pictures were scaled to 1048×1048px, with a point size of 2px. We intentionally zoomed the scatterplots to half of their original size as our task was to count the number of classes, where a point size of 2px seemed to be a good tradeoff between readability and complexity. For each sample, six trials were performed, we alleviated the learning effect by rotating each scatterplot by an angle of $t * 30^\circ$ with randomized t.

In part 2 of the lab study, we randomly selected five different scatterplot samples from the ones used in the numerical evaluation; see Section 5.1. In addition, we used three different color palettes, "Tableau 20" and two from ColorBrewer. Again, each dataset was displayed over a white and a black background. In part 2, the scatterplots were shown in pairs. Hence, the total number of trials was $5(scatterplots) \times 3(color\ palettes) \times 2(backgrounds) \times 3(combinations) = 90$. All the images were shown in their original resolution of 1235×666.6px, with a point size of 4px, since preference tasks (part 2) should be insensitive to the point size.

Participants. We recruited 20 participants (13 males and 7 females). All were from the local university with a major in Computer Science. Their ages range from 21 to 31 (median 23). All participants passed our color deficiency test, had a normal or corrected-to-normal vision, and were used to using computing devices.

Device. The study was run on a quad-core PC with a 27" LCD widescreen with a mouse and a keyboard as the input and a 3840×2160 pixel display as the output. The monitor was calibrated using test images for a faithful color reproduction. All participants were seated at around 60cm from the display in a constantly illuminated room.

Procedure. We applied the following procedure in the lab study: (i) explaining the tasks by the researcher, followed by training; (ii) performing part 1 of the study; (iii) a ten-minute break; (iv) performing part 2 of the study; and (v) a short interview about part 2. In the interview, we were particularly interested in inconsistent choices made by the participants, so we asked them why they did not choose the good ones assumed by our method and what are the factors that influence their choices. Overall, the participants need five minutes on average to finish part 1 (min: 3 minutes and max: 12 minutes), and 13 minutes on average to finish part 2 (min: 8 minutes and max: 20 minutes).

Results. We used an estimation-based approach with effect sizes and confidence intervals [9]. Part 1 results are summarized in Figures 11 (a)-(b). The results were consistent with our hypotheses. The *error* was taken as the total number of errors made by a participant in this task. The result shows that the participants tend to make fewer mistakes in counting classes when $E(\tau)$ is high, in other words, when we optimized the color assignment. In terms of *time*, the results are less clear but still show a tendency that our color assignment method makes it more efficient for people to distinguish between classes. We assume the reason for the less apparent result is that the used color palettes are already quite good, so that even the worst color assignment helps the participant quickly finish the task.

Part 2 results are summarized in Figure 11(c). We measured the consistency of the participant's choices to our ratings in percentage. When a participant chose the better color assignment from the two pictures (e.g., if the participant chose the "good" picture from a "best vs. medium" pair), this trial was marked as "consistent;" otherwise as "inconsistent." A value of "neutral" represented the percentage of options with "no preference." Figure 11 (c) shows that most participants' choices are consistent with our method's ratings. This indicates that our method aligns well with human perception.

Furthermore, we looked at the results from the interviews to understand why the participants made inconsistent choices. Five of the participants mentioned that when they saw two similar pictures, they chose the one that is visually more pleasant. For example, one participant said that he did not like pictures that looked "half dark and half bright," which he thought was "unbalanced." The other four participants all mentioned that they did not like pictures in which "some neighboring classes have incompatible colors, for example, red and green." This is reasonable because our algorithm does not take color harmony into account, which will be part of our future work.

We summarize our lab study results as follows:

- our selected color assignments make the classes separation easy to be perceived;
- there are no significant benefits of our selected color assignments in terms of time; and
- our selected color assignments are typically preferred by users.

5.3 Expert Studies

To compare the performance of our method with expert-chosen color assignments, we invited two experts (1 male and 1 female) who have more than 20 years of experience in color design. In this study, we asked the experts to create their favorable color assignments for multiclass scatterplots, and then examined how close their results were to the results generated by our method.

Study design. We used the two multiclass scatterplots and the color palettes shown in Figure 1 in this study. To help the experts judge the class separabilities, we show the points of different classes in different gray levels. We also plot a convex hull of each class as a guidance for them to observe the class separability, once the expert

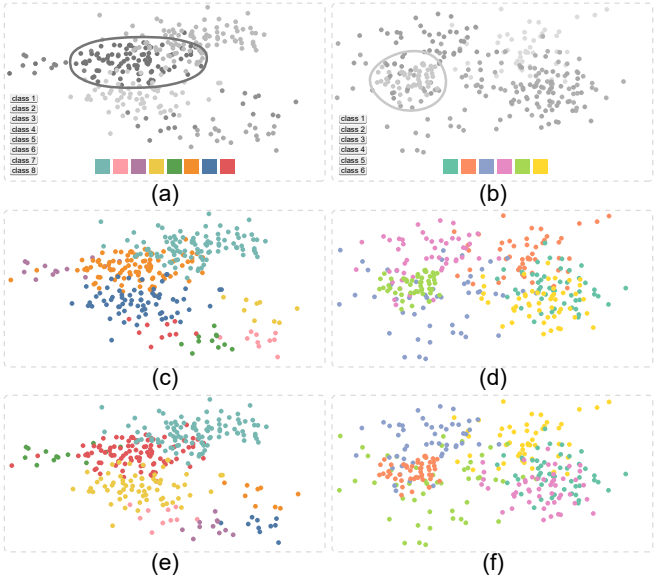

Fig. 12. The inputs and results of two expert studies. (a,b) The grey-scale scatterplots used as the input; (c,d) results generated by one expert; (e,f) results generated by the other expert.

clicks on the corresponding class name; see Figures 12(a) & (b). After selecting a color from the palette, the expert can assign the selected color to the class by brushing the corresponding points. We informed the experts that a good color assignment would foster the separation between classes and that they were allowed to iteratively improve the assignment until they were satisfied with the results.

Results. The two experts spent about five minutes on each scatterplot and generated the results shown in Figures 12 (c) to (f), where all the classes look well-separated like those generated by our method; see Figures 1 (c) and (f). To quantitatively compare the results, we computed the scores of the expert results according to Eq. (7) and determined how this would rank among all the possible 40320 and 720 color assignment permutations for the eight and six classes, respectively; note that 40320 = 8! and 720 = 6!. The results shown in Figures 12 (c) and (d) are ranked 756th (top 1.8%) among the 40320 assignments and 19th among the 720 assignments (top 2.6%), respectively, while the ones shown in Figures 12(e) and (f) are ranked 4053rd (top 10.1%) and 96th (top 13.3%), respectively. These scores show that both results made by the experts closely match with our measure. In particular, the results produced by the first expert are ranked within the top 3%. In summary, we believe that our method resembles and optimizes for human perception of class separation.

6 Interactive Color Assignment System

To further assist the users in selecting appropriate colors, we develop an interactive color assignment system that allows the users to interactively find desired color assignments for multiclass scatterplots. It runs in a web-based environment, which is available as an online tool[1]. After the user uploads a multiclass scatterplot, our system can automatically suggest color assignments by using the default color palette. The users may also select or design their desired color palettes. Besides these basic interactions, we provide three extensions to facilitate the user to intuitively find the desired color assignments.

Top K suggestions. Our basic GA optimization algorithm only reports the single optimal color assignment to the user. In many cases, however, users would like to have more diverse choices. To address this issue, we extend the GA algorithm by choosing the top K unique assignments at each iteration besides recording the best fit. In our system, the default K is six; see the project web page for results.

[1] http://www.color-assignment.net/

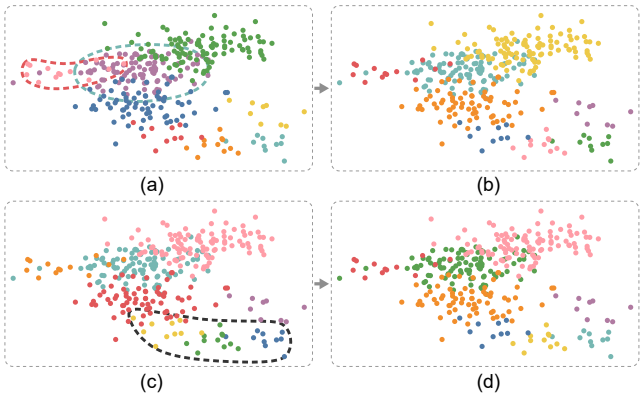

Fig. 13. Two extensions of our approach. (a) a scatterplot where two classes are assigned pink and green indicated by the color of lasso; (b) the result generated by our method; (c) scatterplot with three classes of interest selected; and (d) result generated by our method, where the separation between the two selected classes is maximized.

Pre-assignment of colors. In some cases, users want to assign specific colors to certain classes due to their domain knowledge. Our GA algorithm can easily support this extended function by fixing the pre-assigned colors on the corresponding genes. An example is shown in Figures 13 (a) to (b), where the user chooses the pink and purple classes, re-assigns them with red and green colors, respectively, and then lets the system re-generate the color assignment for the other classes.

Classes of interest. Lastly, users may want to make certain classes more distinguishable, especially for classes that are particularly interesting to the users. By applying the GA algorithm first to these classes, our method finds colors with maximal separation for these classes. After finding colors for these classes, our method searches for the colors of the other classes from the rest of the colors in the palette. In Figures 13 (c) and (d), the user would like to enhance the separability of three classes on the right bottom of the screen, so these three classes are first optimized with maximized color differences. Then the system assigns the remaining colors in the palette to the other classes.

7 Conclusion

We present a method for the color assignment of multiclass scatterplots that takes into account the spatial relationship, density, degree of overlap between point clusters, as well as the background color. These aspects are combined to a new perceptual metric, which is used by an optimization method based on a genetic algorithm to create good color assignments automatically. We evaluated the approach numerically, performed a controlled user study as well as two expert studies. The studies demonstrate that our approach creates good correspondences between optimization results and human preferences for color assignment.

There are various other analytical tasks in multiclass scatterplots [33], such as relative mean value judgments, correlation patterns, and outlier detection. While our current evaluation only investigates the perception of class separability, in the future, we would like to conduct a more thorough evaluation to assess the effectiveness of the color assignments also with respect to these other tasks. Furthermore, we plan to incorporate point sizes of scatterplot dots to point distinctness, whose influence on the color difference has been verified by Szafir et al. [41]. Finally, we want to study color assignments for people with color vision deficiency where the color palette and color assignment might both need to be adapted.

Acknowledgments

This work is supported by the grants of NSFC (61772315), Science Challenge Project (TZ2016002), Shandong Provincial Natural Science Foundation (ZR2016FM12), Leading Talents of Guangdong Program (00201509) and Fundamental Research Funds of Shandong University.

REFERENCES

[1] M. Aupetit and M. Sedlmair. SepMe: 2002 new visual separation measures. In *IEEE Pacific Visualization Symposium*, pp. 1–8, 2016. doi: 10.1109/pacificvis.2016.7465244

[2] J. E. Baker. Adaptive selection methods for genetic algorithms. In *Proc. of the 1st International Conference on Genetic Algorithms*, pp. 101–111. L. Erlbaum Associates Inc., 1985.

[3] L. D. Bergman, B. E. Rogowitz, and L. A. Treinish. A rule-based tool for assisting colormap selection. In *Proc. IEEE Conf. on Visualization*, pp. 118–125, 1995. doi: 10.1109/visual.1995.480803

[4] C. Blake and C. J. Merz. UCI repository of machine learning databases. `https://archive.ics.uci.edu/ml/datasets.html`, 1998.

[5] M. Brehmer, M. Sedlmair, S. Ingram, and T. Munzner. Visualizing dimensionally-reduced data: Interviews with analysts and a characterization of task sequences. In *Proc. of the Fifth Workshop on Beyond Time and Errors: Novel Evaluation Methods for Visualization*, pp. 1–8, 2014. doi: 10.1145/2669557.2669559

[6] Central Bureau of the CIE. Improvement to industrial colour-difference evaluation. Publications of the CIE, 2001.

[7] J. Chuang, D. Weiskopf, and T. Möller. Energy aware color sets. *Computer Graphics Forum*, 28(2):203–211, 2009. doi: 10.1111/j.1467-8659.2009.01359.x

[8] W. S. Cleveland. *Visualizing Data*. Hobart Press, 1993.

[9] G. Cumming. *Understanding the new statistics: Effect sizes, confidence intervals, and meta-analysis*. Routledge, 2013. doi: 10.1037/e588422013-001

[10] K. A. De Jong. *Analysis of the behavior of a class of genetic adaptive systems*. PhD Thesis, University of Michigan Ann Arbor, 1975.

[11] J. C. Dunn. Well-separated clusters and optimal fuzzy partitions. *Journal of cybernetics*, 4(1):95–104, 1974. doi: 10.1080/01969727408546059

[12] H. Fang, S. Walton, E. Delahaye, J. Harris, D. Storchak, and M. Chen. Categorical colormap optimization with visualization case studies. *IEEE Trans. Vis. & Comp. Graphics*, 23(1):871–880, 2017. doi: 10.1109/tvcg.2016.2599214

[13] M. S. Gazzaniga. *The cognitive neurosciences*. MIT press, 2004.

[14] M. Gleicher, M. Correll, C. Nothelfer, and S. Franconeri. Perception of average value in multiclass scatterplots. *IEEE Trans. Vis. & Comp. Graphics*, 19(12):2316–2325, 2013. doi: 10.1109/tvcg.2013.183

[15] C. C. Gramazio, D. H. Laidlaw, and K. B. Schloss. Colorgorical: Creating discriminable and preferable color palettes for information visualization. *IEEE Trans. Vis. & Comp. Graphics*, 23(1):521–530, 2017. doi: 10.1109/tvcg.2016.2598918

[16] M. Harrower and C. A. Brewer. ColorBrewer.org: an online tool for selecting colour schemes for maps. *The Cartographic Journal*, 40(1):27–37, 2003. doi: 10.4324/9781351191234-18

[17] C. G. Healey. Choosing effective colours for data visualization. In *Proc. IEEE Conf. on Visualization*, pp. 263–270, 1996. doi: 10.1109/visual.1996.568118

[18] T. K. Ho and M. Basu. Complexity measures of supervised classification problems. *IEEE Trans. Pat. Ana. & Mach. Int.*, 24(3):289–300, 2002. doi: 10.1109/34.990132

[19] C. Hurter, M. Serrurier, R. Alonso, G. Tabart, and J.-L. Vinot. An automatic generation of schematic maps to display flight routes for air traffic controllers: structure and color optimization. In *Proc. Int. Conf. on advanced visual interfaces*, pp. 233–240, 2010. doi: 10.1145/1842993.1843034

[20] D. Jameson and L. M. Hurvich. Theory of brightness and color contrast in human vision. *Vision Research*, 4(1-2):135–154, 1964. doi: 10.1016/0042-6989(64)90037-9

[21] H.-R. Kim, M.-J. Yoo, H. Kang, and I.-K. Lee. Perceptually-based color assignment. *Computer Graphics Forum*, 33(7):309–318, 2014. doi: 10.1111/cgf.12499

[22] J. A. Lee and M. Verleysen. *Nonlinear dimensionality reduction*. Springer Science & Business Media, 2007. doi: 10.1007/978-0-387-39351-3

[23] S. Lee, M. Sips, and H.-P. Seidel. Perceptually driven visibility optimization for categorical data visualization. *IEEE Trans. Vis. & Comp. Graphics*, 19(10):1746–1757, 2013. doi: 10.1109/tvcg.2012.315

[24] S. Lin, J. Fortuna, C. Kulkarni, M. Stone, and J. Heer. Selecting semantically-resonant colors for data visualization. *Computer Graphics Forum*, 32(3pt4):401–410, 2013. doi: 10.1111/cgf.12127

[25] R. Margolin, A. Tal, and L. Zelnik-Manor. What makes a patch distinct? In *Proc. IEEE Conf. on Comp. Vis. and Pat. Rec.*, pp. 1139–1146, 2013. doi: 10.1109/cvpr.2013.151

[26] B. A. Maxwell. Visualizing geographic classifications using color. *The Cartographic Journal*, 37(2):93–99, 2000. doi: 10.1179/caj.2000.37.2.93

[27] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT press, 1998.

[28] M. Muja and D. G. Lowe. FLANN, fast library for approximate nearest neighbors. In *Proc. Int. Conf. on Computer Vision Theory and Applications*, 2009.

[29] T. Munzner. *Visualization analysis and design*. CRC press, 2014.

[30] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung. Saliency filters: Contrast based filtering for salient region detection. In *Proc. IEEE Conf. on Comp. Vis. and Pat. Rec.*, pp. 733–740, 2012. doi: 10.1109/cvpr.2012.6247743

[31] D. Pham and D. Karaboga. *Intelligent optimisation techniques: genetic algorithms, tabu search, simulated annealing and neural networks*. Springer Science & Business Media, 2012.

[32] P. J. Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *Journal of Computational and Applied Mathematics*, 20:53–65, 1987. doi: 10.1016/0377-0427(87)

[33] A. Sarikaya and M. Gleicher. Scatterplots: Tasks, data, and designs. *IEEE Trans. Vis. & Comp. Graphics*, 24(1):402–412, 2018. doi: 10.1109/tvcg.2017.2744184

[34] M. Sedlmair and M. Aupetit. Data-driven evaluation of visual quality measures. *Computer Graphics Forum*, 34(3):201–210, 2015. doi: 10.1111/cgf.12632

[35] M. Sedlmair, T. Munzner, and M. Tory. Empirical guidance on scatterplot and dimension reduction technique choices. *IEEE Trans. Vis. & Comp. Graphics*, 19(12):2634–2643, 2013. doi: 10.1109/tvcg.2013.153

[36] M. Sedlmair, A. Tatu, T. Munzner, and M. Tory. A taxonomy of visual cluster separation factors. *Computer Graphics Forum*, 31(3pt4):1335–1344, 2012. doi: 10.1111/j.1467-8659.2012.03125.x

[37] V. Setlur and M. C. Stone. A linguistic approach to categorical color assignment for data visualization. *IEEE Trans. Vis. & Comp. Graphics*, 22(1):698–707, 2016. doi: 10.1109/tvcg.2015.2467471

[38] G. Sharma, W. Wu, and E. N. Dalal. The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, 30(1):21–30, 2005. doi: 10.1002/col.20070

[39] S. Silva, B. S. Santos, and J. Madeira. Using color in visualization: A survey. *Computers & Graphics*, 35(2):320–333, 2011. doi: 10.1016/j.cag.2010.11.015

[40] M. Sips, B. Neubert, J. P. Lewis, and P. Hanrahan. Selecting good views of high-dimensional data using class consistency. *Computer Graphics Forum*, 28(3):831–838, 2009. doi: 10.1111/j.1467-8659.2009.01467.x

[41] D. A. Szafir. Modeling color difference for visualization design. *IEEE Trans. Vis. & Comp. Graphics*, 24(1):392–401, 2018. doi: 10.1109/tvcg.2017.2744359

[42] Tableau Software. The tableau visualization system. `http://www.tableausoftware.com/`.

[43] A. Tatu, G. Albuquerque, M. Eisemann, J. Schneidewind, H. Theisel, M. Magnork, and D. Keim. Combining automated analysis and visualization techniques for effective exploration of high-dimensional data. In *Proc. IEEE Symposium on Visual Analytics Science and Technology*, pp. 59–66, 2009. doi: 10.1109/vast.2009.5332628

[44] C. Tominski, G. Fuchs, and H. Schumann. Task-driven color coding. In *Proc. Int. Conf. on Information Visualisation*, pp. 373–380, 2008. doi: 10.1109/iv.2008.24

[45] B. E. Trumbo. A theory for coloring bivariate statistical maps. *The American Statistician*, 35(4):220–226, 1981. doi: 10.2307/2683294

[46] L. Wang, J. Giesen, K. T. McDonnell, P. Zolliker, and K. Mueller. Color design for illustrative visualization. *IEEE Trans. Vis. & Comp. Graphics*, 14(6):1739–1754, 2008. doi: 10.1109/tvcg.2008.118

[47] Y. Wang, K. Feng, X. Chu, J. Zhang, C.-W. Fu, M. Sedlmair, X. Yu, and B. Chen. A perception-driven approach to supervised dimensionality reduction for visualization. *IEEE Trans. Vis. & Comp. Graphics*, 24(5):1828–1840, 2018. doi: 10.1109/tvcg.2017.2701829

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

[49] A. Zeileis, K. Hornik, and P. Murrell. Escaping RGBland: selecting colors for statistical graphics. *Computational Statistics & Data Analysis*, 53(9):3259–3270, 2009. doi: 10.1016/j.csda.2008.11.033

[50] L. Zhou and C. D. Hansen. A survey of colormaps in visualization. *IEEE Trans. Vis. & Comp. Graphics*, 22(8):2051–2069, 2016. doi: 10.1109/tvcg.2015.2489649