Categorical Colormap Optimization with Visualization Case Studies

H. Fang, S. Walton, E. Delahaye, J. Harris, D. A. Storchak, and M. Chen, *Member, IEEE*

Abstract—Mapping a set of categorical values to different colors is an elementary technique in data visualization. Users of visualization software routinely rely on the default colormaps provided by a system, or colormaps suggested by software such as *ColorBrewer*. In practice, users often have to select a set of colors in a semantically meaningful way (e.g., based on conventions, color metaphors, and logological associations), and consequently would like to ensure their perceptual differentiation is optimized. In this paper, we present an algorithmic approach for maximizing the perceptual distances among a set of given colors. We address two technical problems in optimization, i.e., (i) the phenomena of local maxima that halt the optimization too soon, and (ii) the arbitrary reassignment of colors that leads to the loss of the original semantic association. We paid particular attention to different types of constraints that users may wish to impose during the optimization process. To demonstrate the effectiveness of this work, we tested this technique in two case studies. To reach out to a wider range of users, we also developed a web application called Colourmap Hospital.

Index Terms—Color, categorical colormap, optimization, seismological data visualization, London tube map.

1 Introduction

Color is one of the most prevalent visual channels used in visualization because of its merits in visual search and easy implementation [49]. *Categorical colormaps* (or qualitative colormaps) are commonly featured in visualization for mapping data values to distinct colors, allowing users to identify categories of values quickly through association [4]. The choice of colors can affect the users' ability to learn and remember the mapping [3, 35], and more importantly, their ability to differentiate the colors perceptually when such a mapping is used in a visualization [2, 22, 9].

Colormap design has been a subject of much study. Many schemes for choosing colors exist, including historical convention [6], system recommendations [1, 47, 20], and metaphoric colors [31]. There have been a number of design guidelines, e.g., [21, 7, 43, 39]. This paper is concerned with a classic question in specifying a categorical colormap, that is, when a visualization designer has selected a set of colors based on one of the above schemes or by following certain guidelines, how can this colormap be further optimized?

Lee *et al.* [29] provided an optimization method based on the visual saliency of each point against its surroundings in a visualization. While this is a useful solution for optimizing a specific visualization, it is not suitable for a system to generate numerous visualizations for different datasets using a consistent colormap. We thus devised a new algorithmic approach that enables users to define several types of constraints, for instance, (i) to fix a certain number of colors (e.g., for background colors), and (ii) to limit the variation of certain properties of the original colors (e.g., to maintain the intended color semantics).

In terms of measuring color differences, the merits of perceptually-uniform color spaces (e.g., CIE L*u*v* and L*a*b*) over commonly-used color spaces (e.g., RGB, HSB) is well documented in the color literature. There are several proposed metrics for measuring color distance in a perceptually-uniform space, including CIE76, CIE94, CIEDE2000 and CMC84 [22]. There is a general consensus that CIEDE2000 outperforms other metrics (e.g., [51]). We thus decided to use CIEDE2000 (cf. CIE76 in [29]), and focus our effort on the algorithmic issues in optimization.

In this work, we examine three algorithms in terms of convergence, speed and deployment. In particular, we provide a technical solution to address the phenomena of local maxima in optimization because the distance space based on CIEDE2000 is not totally convex. For example, as shown in Fig. 7(a), the lines of the London underground were introduced gradually into a colormap. Because of the need for maintaining the historic color mappings for passengers, this has created challenges for making new lines differentiable. While the Nelder-Mead simplex method has a low computational cost, optimization is often impeded by local maxima. Meanwhile the technique of genetic algorithm can address this issue, but it requires user-defined constraints to prevent the loss of the original semantic association.

We also report a case study where seismologists needed to re-specify the categories of depths of seismic events, and would like to improve an existing colormap for visualizing the depths of seismic events in two visual representations. As both visual representations are typically used by the data analysts hundreds of times a day, it is important for the colors in the colormap to be easily memorable and distinguishable. This work has enabled a scientific approach to the design of colormaps in this application.

Our contributions include (i) a novel approach for optimizing a categorical colormap algorithmically while enabling users to define initial metaphoric colors and set constraints for the optimization (Sections 3 and 5); (ii) a study of three optimization algorithms in this context and compared their relative merits in terms of convergence, speed, and deployment (Sections 4); (iii) two case studies that demonstrate the practical usability of our approach (Section 7); and (iv) a web-based application to enable wider access to some of the presented techniques.

2 Related Work

In this section, we first give a brief survey of color designs in visualization. We then give an overview of perceptually-uniform color spaces, and optimization in such spaces.

Color Designs in Visualization. Color is one of the most powerful visual channels in visualization, as it facilitates pre-attentive visual search and pop-out effect. In this respect, there is a common consensus that among the four commonly-used visual channels, *color* precedes *size*, *shape* and *orientation*. This has been well-supported in the psychology literature (e.g., [50, 38, 44, 37]). Bertin [4] considered that color is *associative*, *selective* and *ordered*. Ware [48] conducted empirical studies on colormaps in the context of visualization. Borgo *et al.* [5] and Demiralp *et al.* [8] studied the distance between visual objects featuring integrated visual channels (including colors) using crowdsourced experiments.

- *Hui Fang was with University of Oxford and International Seismological Centre, and is now with Edge Hill University, UK. Email: hui.fang@edgehill.ac.uk.*
- *Simon Walton and Min Chen are with University of Oxford, UK. Email: {simon.walton, min.chen}@oerc.ox.ac.uk.*
- *Emily Delahaye, James Harris and Dmitry A. Storchak are with International Seismological Centre, UK. Email: {emily, james, dmitry}@isc.ac.uk.*

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/

Visualization textbooks (e.g., by Stone [43], Ware [49], Munzner [33], and Tufte [45]) offer guidance about colormaps. Other sources of guidance include [12, 21, 7, 6, 39]. Several tools have been developed for helping users design colormaps. Bergman *et al.* [1] developed a rule-based approach taking spatial frequency into account. Meier *et al.* [32] developed an interactive tool for users to "mix and organize colors; explore color combinations; and solicit historical, theoretical, or expert sources." Harrower and Brewer [20] developed the popular web-based tool, *ColorBrewer*, providing pre-defined colormaps for sequential, diverging and qualitative forms of mapping.

Colors have a profound role in history, culture and everyday life [13, 19, 15]. Lin *et al.* [31] used automated web searches to determine the association between a given a set of terms and the colors in representative images. Kuhn *et al.* [27] proposed an automated method for recoloring images for dichromats.

Perceptually-uniform Color Spaces. For many decades, much effort has been made to develop *perceptually uniform color spaces* where every pair of equal distances measured corresponds with a pair of equally-perceived color differences. The development of *Lab* color spaces (lightness and two color dimensions) saw the evolution from Hunter's Lab space in 1948, to CIE 1976 L*a*b* color space, to the CIE 1994 and CIEDE2000 distance functions [23, 26, 22]. Today the difference measure ΔE in CIEDE2000 remains as the industrial standard. Although CIEDE2000 in general offers a better distance measure than other formulae, it is important to note that it does not guarantee perceptual uniformity in all conditions (e.g., varying light sources or materials). There are several parameters that should be tuned for each individual condition. The CIEDE2000 standard provides parameters for only two conditions, none of which represent computer displays. We will address this issue in Section 4.

Several works in the field of visualization considered the need for optimizing a pre-defined colormap. Kindlmann *et al.* [25] proposed a method for colormap optimization by using faces as measuring templates. Wang *et al.* [47] describe a rule-based system for optimizing the colors chosen for 2D and 3D visualizations, accommodating factors such as color discrimination, translucency and labeling. Chuang *et al.* [11] proposed to optimize colormaps for lowering energy consumption. Other optimization works, which do not involve a perceptually uniform space, include [40, 30].

Lee *et al.* [29] presented a color optimization algorithm that maximizes the perceptual distances among a set of categorical colors. Their work is based on CIE L*a*b* space, for which the authors pointed out its limitation, suggested a remedy of using CIE1994 and CIEDE2000, and reasoned about the algorithmic difficulty in using CIEDE2000. Our work differs from [29] in several ways: (i) We focused on the optimization of a color palette rather an individual visualization; (ii) We enabled users to define constraints in a more meaningful way and demonstrated the use of such constraints through a real-world application; (iii) We made use of CIEDE2000, studied a number of optimization algorithms and identified their relative merits in addressing the algorithmic difficulty; and (iv) We provided a web application to make this technique available to a wider range of users.

3 Problem Statement

Let $\mathbf{C} = \{c_1, c_2, \ldots, c_n\}$ be a set of colors. Let $D(c_i, c_j)$ be a distance function for measuring the difference between two colors. We assume that D satisfies all essential conditions for a metric, i.e., non-negativity, identity of indiscernibles, symmetry, and triangle inequality. We denote the minimal distance among all pairs of colors as $D_{\min}(\mathbf{C})$, i.e.,:

$$D_{\min}(\mathbf{C}) \leq D(c_i, c_j), \quad \forall i, j\,(i \neq j) \tag{1}$$

The general optimization problem is to find a set of color transformations, $\Phi_i, i = 1, 2, \ldots, n$, such that the set of transformed colors

$$\mathbf{C}' = \{\Phi_1(c_1), \Phi_2(c_2), \ldots, \Phi_n(c_n)\}$$

has reached maximal value of $D_{\min}(\mathbf{C}')$.

The problem, however, is not as simple as the above general formulation. There are following additional considerations:

- *Color Space* — There are several color spaces, such as RGB, HSL, HSV, and CMYK, which are widely understood by users and supported by software for drawing, presentation, and image manipulation. Other color spaces, such as CIE XYZ, CIE L*u*v*, CIE UVW, and CIE L*a*b* were developed according to measurements of human color perception. There are also color spaces designed for television broadcasts and other industrial applications. The distance calculation can take place in any of such spaces, but will yield different results.
- *Distance Metrics* — Given colors are defined using 3- or 4-tuples, many distance functions can be candidates for such a function. These include the commonly-used Euclidean distance, the Manhattan distance, and their generalization, the Minkowski distance. There are also many other candidates, such as Chebyshev distance and normed vector space. Some color spaces, such as CIE L*a*b*, have predefined distance functions, while others allow for more flexible interpretations.
- *Constraints* — In practice, not all colors need to be or should be transformed. For example, the background color of a display canvas, and/or the foreground color for labeling may be fixed. In many cases, the users may wish to restrict the amount of changes allowed during color transformation. Such constraints normally relate to the semantics of a colormap, and may be defined for one or a few specific colors, or for all colors.

In both our case studies, to be detailed in Section 7, there is a combination of requirements for the above considerations. On one hand, the distance calculations must take place in a perceptually-uniform color space, while on the other hand, the constraints must be defined in a color space familiar to users. On one hand, optimization algorithms prefer to work on a convex distance space, while on the other hand the property of perceptually-uniformity may require a distance function that is not convex. On one hand, a color scheme may be used for displaying foreground information, and on the other hand, a metaphorically similar scheme is required for displaying background information. All these make the color optimization a non-trivial undertaking.

4 General Optimization

Global optimization techniques have been used previously in the field of visualization (e.g., [36, 29]). Fig. 1 shows four iteration steps in optimizing the distances among four colors. The four colors are shown in a special palette where each color (circle) is overlaid on all other colors (strips). The pairwise fitness diagram in (e) shows the CIEDE2000 distance of each pair of colors over time. The multi-band colormap is defined by minimal, average, and maximal distances ($D_{min}, D_{avg}, D_{max}$) at t_0, together with absolute 0-distance and the black-white distance (D_{BW}). The four ranges are:

- $[0, D_{min}]$ in Wine – At t_0, it shows the minimum. After t_0, any such color is highly undesirable and is expected to disappear.
- $(D_{min}, D_{avg}]$ in Yellow – At t_0, it shows any distance equal to or below the average. Colors in this range indicate undesirable distances, and should ideally be reduced over time.
- $(D_{avg}, D_{max}]$ in Green – At t_0, it shows any distance above the average at t_0. Colors in this range indicate desirable distances, and more of such distances are expected to appear after t_0.
- $(D_{max}, D_{BW}]$ in Grey – As colors in this range are above the maximum at t_0, they appear only after t_0. The more, the better.

In our work, we focus on optimizing a pre-defined initial categorical colormap such that the input colormap is improved to a level that cannot be further improved algorithmically. We consider the following three goals that potentially can be in conflict: (i) the perceptual distance between the colors should be maximized according to a distance metric; (ii) the user should be able to define *constraints* to limit undesired changes; and (iii) the fixed background color and foreground colors (if any) should be factored into the optimization process.

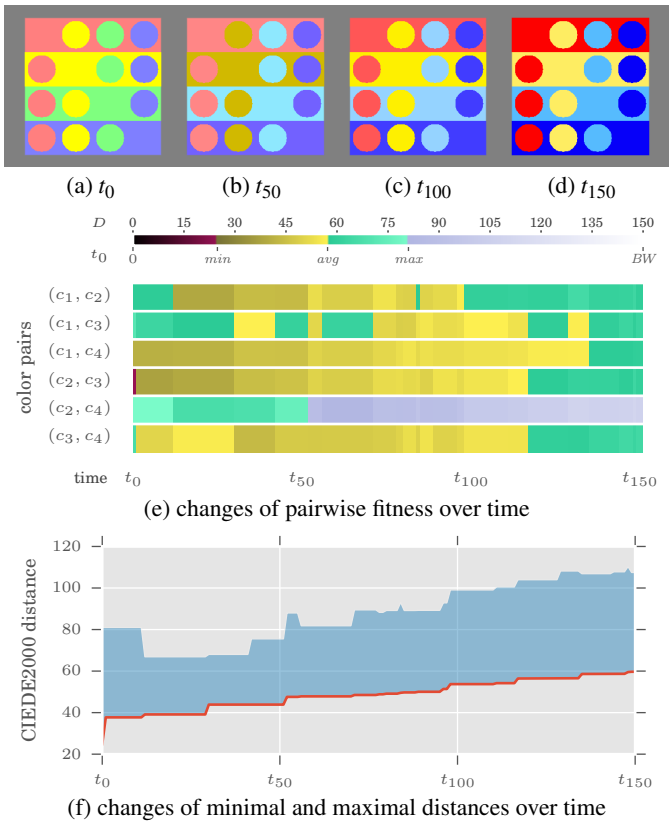


Fig. 1. Basic optimization. (a-d) Four colors at four iteration steps are shown in a special palette where each color (circle) is overlaid on all other colors (strips). (e) The pairwise fitness diagram shows the CIEDE2000 distance of each pair of colors over time. The four color bands in the colorbar are defined based on the distances at t_0. They are absolute 0-distance, minimal, average, and maximal distances, and the black-white distance. (f) The minimum and the maximum increase over time, though the priority is placed on the increase of the minimum.

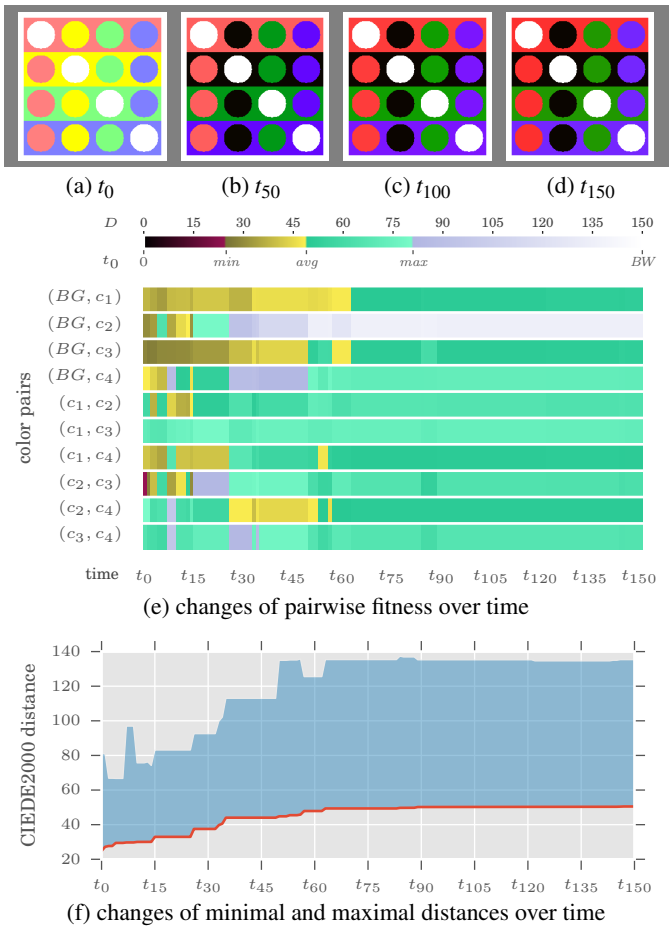


Fig. 2. Optimization with fixed white background (white circles in the palette). The resultant colormap differs noticeably from that in Fig. 1, and the optimization converges quicker.

4.1 Perceptual Color Metrics

There are many metrics for measuring color difference. Perception-based metrics attempt to simulate the color differences perceived by the human visual system. Most of these metrics perform in the CIE L*a*b* color space. CIE1976 is the simplest metric measuring Euclidean distances in L*a*b*. More advanced metrics, CIE1994 and CIEDE2000, were introduced to address perceptual non-uniformity in L*a*b*. Among CIE standards, CIEDE2000 [10, 42] is currently considered to be the closest to human perception. Its formula is widely available in the literature.

CIEDE2000 has three pre-determined parameters, K_L, K_C and K_H, to balance the weights of the color components for different visual media, with the default value 1 for each of them. One common adjustment is to set $K_L = 2$ for textiles, and $K_L = 1$ for graphic arts [26]. However, computer displays are neither textiles nor graphic arts. Different sets of K_L, K_C and K_H for computer displays were proposed in the literature, e.g., [52, 51, 53]. After comparing their experimental conditions and results, we set $K_L = 0.725$, $K_C = 1$ and $K_H = 1$ in our calculations.

In general, the larger the distance between a pair of colors, the more perceptually distinguishable they are. This notion is however probabilistic. This is partly because the perception of color differences varies from person to person and from chart to chart [12]. It is also because the CIEDE2000 distance space is highly nonlinear from the perspective of the RGB space. The transformation from RGB to CIE L*a*b* introduces the first distortion [18], and the distance calculation introduces the second distortion since CIEDE2000 is by definition not a Euclidean distance metric [10].

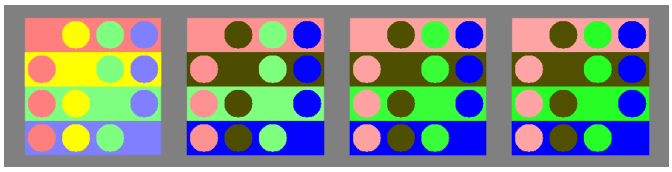
The visible color space is bounded. The more colors a colormap tries to accommodate, the smaller D_{min} will be. Green-Armytage proposed 26 possibly distinguishable colors after a series of empirical studies. The minimal pairwise distance is 6.86 (CIEDE2000 is unitless). Among 325 pairs of distances, 3.69% below 20, and 6.77% below 25. Since several colors among the 26 colors are very similar, it suggests that [20, 25] might be the borderline zone.

4.2 Fitness Function

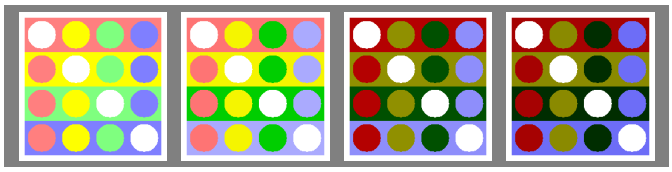
Our optimization strategy is to increase the minimal perceptual distance among different pairs of colors in the colormap. Although some other works, e.g., [29], used a strategy to maximize the average distance, we believe that improving the color pairs with the shortest distances yields greatest benefit to visualization. Thus a solution with a higher minimum and a lower average is more preferable than that with a lower minimum and a higher average. Based on Eq. 1, given m intermediate states $\mathbf{C}_1, \mathbf{C}_2, \ldots, \mathbf{C}_m$, the fitness function selects $k < m$ states such that $D_{\min}(\mathbf{C}_{selected}) > D_{\min}(\mathbf{C}_{unselected})$. At the final iteration, the state with the maximal $D_{\min}$ is selected as the solution.

5 Constrained Optimization

When optimizing a colormap algorithmically, users may potentially find themselves with a colormap that is perceptually optimized, but suboptimal for its application. For example, many categorical colormaps are designed purposely to associate colors with symbolic or metaphoric meanings. When new colors are introduced to an existing colormap, it is usually desirable to maintain the consistency and memorability of the old colors. The two case studies in Section 7 exemplify these two scenarios.



(a) Without background: t_0, t_{50}, t_{100}, t_{150}



(b) With a white background: t_0, t_{50}, t_{100}, t_{150}

Fig. 3. Optimization constrained by a fixed range. In both examples, hue is constrained by a 0-size range, which is typically used for maintaining color symbolism and metaphors.

In this work, we improved the usability of the general color optimization process in Section 4 with facilities for users to define various constraints. We enabled users to specify them in CIE L*a*b*, RGB, and HSL, while incorporating other color spaces will be a trivial undertaking. In practice, we have found that it is easy to control the optimization using constraints defined in HSL, though the optimization is always performed in CIE L*a*b* in conjunction with CIEDE2000.

It is also necessary or highly desirable to ensure that the introduced constraints will not undermine the standard formula of CIEDE2000. We thus avoid the use of a weighting function to combine multiple optimization objectives. We group constraint specifications into three categories: *Fixed Colors*, *Fixed Range*, and *Adaptive Range*.

Fixed Colors. The most common use of this type of constraint is for background and foreground colors. In our user interface, we purposely referred to this type of constraint as "Background/Foreground Colors". In some applications, it is necessary to fix more than two colors. For instance, when a map is used as a background, one may wish to choose three fixed colors for ocean, land and nation borders. The separation between background and foreground does not affect optimization, but is useful for the users to be aware of the difference in impact (e.g., their sizes and locations in a visualization).

Fig. 2 illustrates the optimization of four colors to be displayed on a fixed white background. Comparing to the color palettes in Fig. 1 (optimized without background), we can observe that the intensity values of the four colors at t_{100} in Fig. 2 are relatively darker, and the blue in the 4th column has been shifted towards purple to widen the CIEDE2000 distance from the dark grey and the green in the second and third columns respectively.

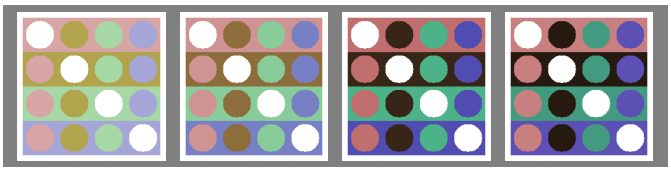
Fixed Range. With this facility, users can set a constraint on each component of a color to be optimized. In this work, the available components are H, S, V; R, G, B; and L*, a*, b* in the three color spaces supported. Given the initial value x of an arbitrary color component, X, and a fixed range defined as $[x_{min}, x_{max}]$, the evolution of x is always constrained within this fixed range. When the initial color x is not in the fixed range, we first translate it to a color $x' \in [x_{min}, x_{max}]$ as:

$$x' = \frac{(x - X_{min})(x_{max} - x_{min})}{(X_{max} - X_{min})} + x_{min}, \quad X_{min} \neq X_{max} \qquad (2)$$

where X_{min} and X_{max} are the overall range of component X.

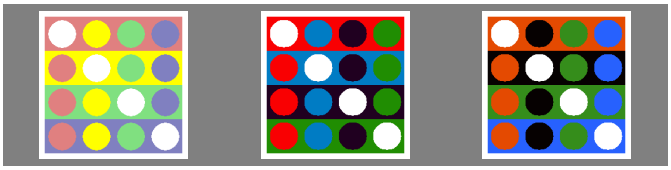
The user can set a fixed range as tight as $x_{min} = x_{max}$. For example, one may wish to fix the hue of the colors to be optimized in order to maintain their historic symbolism. Fig. 3 shows such an optimization with a 0-size range on hue. The transformations of colors thus take place only in terms of saturation and luminance in the HSL space. (Note that the actual optimizations are still performed in CIE L*a*b*.) Comparing with Fig. 2, we can see that the shifting towards purple was prevented in Fig. 3(b).

In addition to semantic requirements (e.g., historical convention, color symbolism and metaphors, and logological association), this type of constraint can also enable users to have more control over the solution to be generated to suit a particular, artistic style, display medium, or display environment. For example, overly saturated colors are often avoided by visualization designers typically for aesthetic reasons. In order to avoid saturated colors at t_{100} in Figs. 1, 2 and 3, one may set a desired saturation range [0.1, 0.4]. Under this fixed range constraint, the optimization process first transforms the initial value of the saturation component into the range using Eq. 2, and then starts the iteration as the normal constrained optimization. The results are shown in Fig. 4.



(a) t'_0 (b) t_{50} (c) t_{100} (d) t_{150}

Fig. 4. Optimization with a range constraint on saturation, which is set between [0.1,0.4]. The initial palette in (a) shows the results after the normalization of the initial colors using Eq. 2. The optimization is also constrained by a white background.



(a) t_0 (b) no range control (c) adaptive range

Fig. 5. This example shows the comparison of optimization with and without adopting the adaptive range. Without using the adaptive range control when running the GA algorithm, the result in (b) has a color swap effect while the adaptive range control makes smooth evolution.

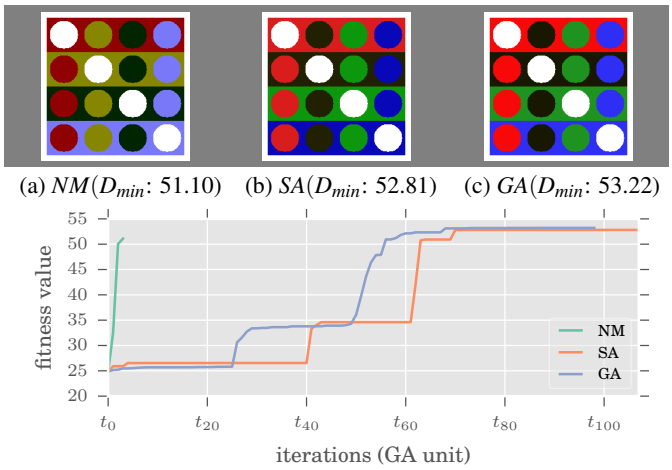
Adaptive Range. The adaptive range is defined by Δ, the percentage of changes allowed at each iteration. Let x_i be the value of an arbitrary color component X at the i^{th} iteration. The transformed color x'_i is constrained by the range $[x_i(1-\Delta), x_i(1+\Delta)]$, together with $[X_{min}, X_{max}]$, the overall range for X. After each iteration, the algorithm takes control of the search range by recalculating the actual range based on the current state of the solution. As a result, the adaptive range provides a mechanism to transform colors during optimization at a gentle pace.

An example of using adaptive range is shown in Fig. 5. When using the technique of genetic algorithm (GA) without the adaptive range control, the green in the initial palette in Fig. 5 (a) becomes a blue color in the resultant palette in (b), while the blue becomes green. With an adaptive range constraint, the problem of swapping in GA can be alleviated. Fig. 5(c) shows the result by setting Δ to 10% for each of the H, S, and V color components. The undesired swapping of green and blue is eradicated.

6 Optimization Algorithms

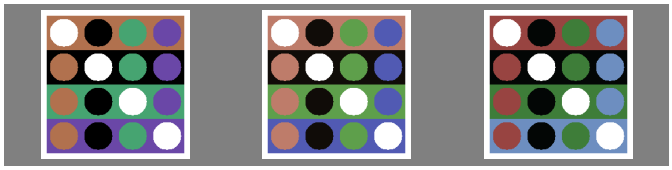
Given the problem statement in Section 3, a full search for an optimal solution is of a complexity $O((k_1 k_2 k_3)^n)$, where n is the number of colors, and k_1, k_2 and k_3 are the numbers of valid values of the three color components. For example, for a typical display in the RGB space, we have $k_1 = k_2 = k_3 = 256$, and 256^{3n} combinations in total.

The basic hill climbing algorithm avoids a full search by following the most promising path from an initial color. There are $m = 3n$ variables for all the components of the n colors. For each iteration, we may consider three possible options for transforming a color component: no change ($\Phi(x) = x$), increment ($\Phi(x) = x + \delta$), or decrement ($\Phi(x) = x - \delta$), where $\delta > 0$. Hence, each iteration would need to evaluate $3^m = 3^{3n}$ optional transformations for n colors. Therefore the full search and hill climbing are both intractable. To reduce the

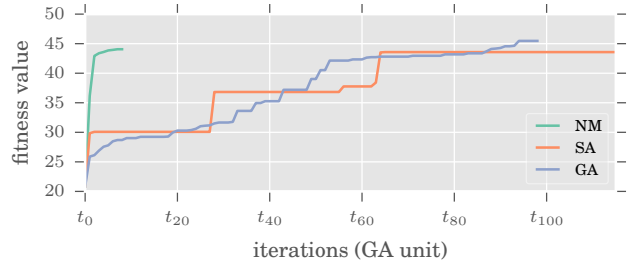


(a) $NM(D_{min}$: 51.10) (b) $SA(D_{min}$: 52.81) (c) $GA(D_{min}$: 53.22)

(d) Fitness (D_{min}) vs. Cost (number of iterations)



(e) $NM(D_{min}$: 44.06) (f) $SA(D_{min}$: 43.57) (g) $GA(D_{min}$: 45.47)



(h) Fitness (D_{min}) vs. Cost (number of iterations)

Fig. 6. Comparing the three algorithms in terms of the convergence of D_{min} and their computational cost. In (a-d), the setting of Fig. 3 is used, and in (e-h), the setting of Fig. 4 is used. D_{min} is the minimal distance among all color pairs. A larger value indicates a better solution.

computational cost, we considered the following three optimization algorithms, and compared their performance in MATLAB.

Nelder-Mead Simplex (NM) is a popular technique due to its lower computational cost [28], and it has been used extensively in visual computing (e.g., [24, 14]). Instead of evaluating 3^m optional transformations, it starts each iteration with $m+1$ options, whereby the term *simplex* implies $m+1$ vertices of a simplex in $\mathbb{R}^m$ space. The options are ordered according to the fitness function D_{min} given in Section 3. Four arithmetic operations, *reflection*, *expansion*, *contraction*, and *shrinkage* are then used to improve the simplex by moving away from the worst vertex. The deformed simplex becomes the $m+1$ options for the next iteration. We choose the first $m+1$ options by considering $2m$ transformations, each features either increment ($\Phi(x) = x+\delta$), or decrement ($\Phi(x) = x-\delta$) of just one single color component. We then use the fitness function to identify the m best options. Together with the initial color, we have the $m+1$ options for the first iteration. In our implementation, we adapted MATLAB's `fminsearch` to perform optimization in a constrained color space.

Simulating Annealing (SA) is a technique inspired by the physical process of cooling [41]. It alleviates the problem of local maxima (or minima) in hill climbing by exploring worse options probabilistically, while maintaining a record of the best solution throughout the process. The cooler the temperature, the less probable for a worse option to be explored further. In our implementation, the optimization commences with the input color as the initial best solution and the first option to explore. The initial temperature T_0 is set to 100 (i.e., 100% probability). At the i^{th} iteration, $T_i = T_0 * 0.95^i$. We made use of MATLAB's `simulannealbnd`, which also enables the control of the exploration boundary for constrained optimization.

Genetic Algorithm (GA) has been widely used in many applications [17]. The optimization process is defined as an evolution of a collection of ω chromosomes, each containing an optional transformation. In each generation (i.e., iteration), the ω chromosomes are evaluated using the fitness function D_{min}. $\upsilon \ll \omega$ best chromosomes are selected to progress to the next generation. Meanwhile, $\omega - \upsilon$ pairs of distinct parents are probabilistically chosen from the ω chromosomes. The chromosomes with better fitness values have more chances to be chosen. These parents result in $\omega - \upsilon$ children to progress to the next generation. The transformation contained in each child chromosome is a linear combination of the transformations in its two parent chromosomes. We used the `ga` function in MATLAB, which also includes a mutation operation for introducing the additional diversity in evolution. After some experimentation, we set $\omega = 80$, $\upsilon = 2$, and *mutation* $= 0.03$. The initial 80 chromosomes contain the set of n input colors, and 79 sets obtained by uniformly sampling the HSL space, and within the fixed or adaptive range if set by the user.

Comparison of the Algorithms. We compare the three algorithms in terms of the minimal distance D_{min} reached, and the speed for achieving convergence. Using a range of tests with different settings of colors and constraints, the comparison was fairly conclusive. Fig. 6 shows the testing results of the two examples in Figs. 3 and 4.

Fig. 6 shows a phenomenon of GA $>$ ((NM $\geq$ SA) $\vee$ (NM $\leq$ SA)) in terms of D_{min}. This was quite typical in our tests. In general, GA tended to achieve the largest D_{min}, while the ordering between NM and SA was less consistent.

In order to compare the computational costs in a consistent manner, we test the three algorithms by making sure that they will evaluate the same number of options in each iteration. In NM and SA, we grouped every 80 consecutive iterations into a macro iteration in order to be consistent with GA ($\omega = 80$). As exemplified by Figs. 6(d, h), NM converged much more quickly than GA and SA, while GA was slightly faster than SA in most cases.

Furthermore, NM and SA were more likely to be trapped in a local maximum. GA alleviated this problem, but would often deviate from the input colors, such as changing the order of color symbolism in Fig. 5(b). The adaptive range constraint introduced in Section 5 can prevent GA from such deviations.

7 Application Case Studies

This work was initially motivated by the need to design a number of categorical colormaps in a visualization system for visualizing seismological data. For this scientific visualization application, it was difficult to resolve conflicts of color symbolism by simply selecting each colormap using software such as *ColorBrewer*. To test the utility of the technique developed in a more general scenario, we applied it to the classic problem of visualizing the London Tube Map.

7.1 Case Study #1: The London Tube Map

The London tube map is iconic in the history of visualization. In 1863, the world saw its first underground railway when the Metropolitan Railway opened in London. Since then it has been developed into an integrated underground and overground network with 14 lines. When these lines were gradually introduced, as shown in Fig. 7(a), the colormap of the London Tube Map had to assign distinct colors to new lines while maintaining the old color symbolism. There is no doubt that the designers of every evolution of the tube map made good efforts to optimize the colormap, perhaps based mainly on their artistic intuition. Nevertheless, some colors in the current standard colormap for computer screens [16] have rather short distances in terms of CIEDE2000. For example, as shown in Fig. 7(a), the shortest CIEDE2000 distance is 15.6 between DLR and Waterloo & City.

We may wish to optimize the colormap with a need for preserving the historical color symbolism. We can, for instance, fix the colors of

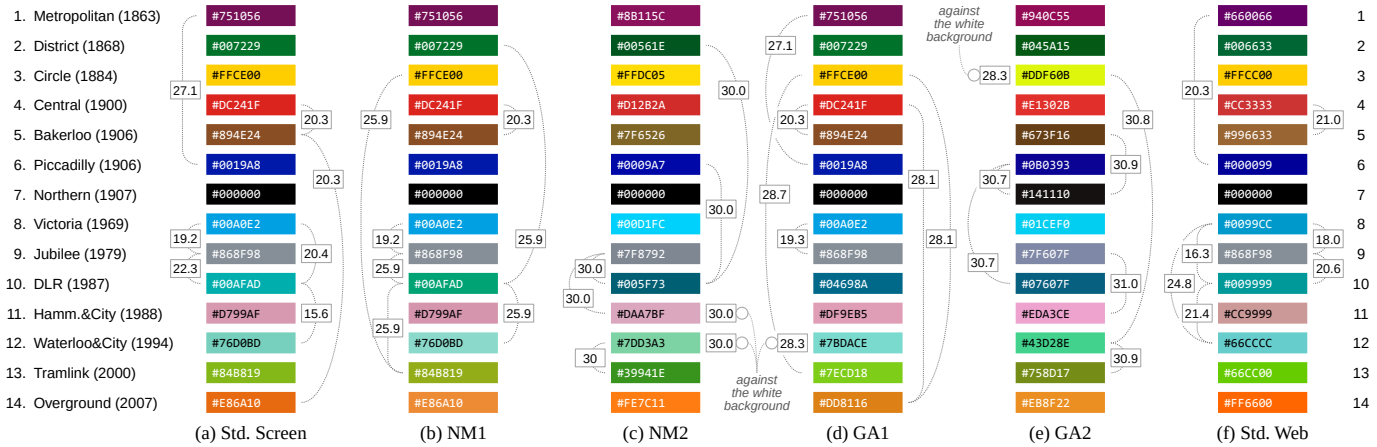


Fig. 7. The London underground has massively expanded over one and a half centuries. (a) Many pairwise CIEDE2000 distances in the TfL (Transport for London) standard screen colormap [16] are rather close. (b) To maintain color symbolism for all lines, we fix the colors of pre-1980 lines, and optimize those of post-1980 with some constraints using the Nelder-Mead simplex method. (c) We can further increase the distances by relaxing the constraints for all lines. (d) Using the genetic algorithm method, the colors of post-1980 lines can break out of local maximum regions in comparison with (b). (e) Slightly relaxing the colors of all lines allows us to address the short distances among pre-1980 colors. (f) Meanwhile, the TfL standard web colormap also features some short CIEDE2000 distances. Seven shortest distances in each map are labelled. Colors being displayed may be affected by device-dependent settings.

the nine pre-1980 lines, and focus on the five post-1980 lines. Specified in the HSL space, the hues of the five colors are constrained each by $\pm 5\%$, and saturation and luminance by $\pm 10\%$. Using the Nelder-Mead simplex method (NM), which is computationally most efficient among the three algorithms tested, we have improved the distance between DLR and Waterloo & City from 15.6 to 25.9 as shown in Fig. 7(b). Consequently, the distance of 19.2 between Victoria and Jubilee becomes the shortest. We may then decide to tackle this problem by removing the constraints for the post-1980 lines while relaxing the constraints for the pre-1980 lines from fixed HSL to H: $\pm 5\%$, and S, L: $\pm 10\%$. This yields a colormap in Fig. 7(c). We can notice that all short distances have been significantly improved.

Switching to Genetic Algorithm (GA), we may repeat the optimization processes using the same constraints for NM. Comparing (b) and (d) in Fig. 7, we can see that GA obtained a noticeably better set of colors for the five post-1980 lines. This suggests that the application of NM in (b) may have suffered from the presence of local maxima. Meanwhile, using the same relaxed constrains as in Fig. 7(c), GA delivered a comparable colormap as shown in (e). In the fact, GA achieved a slightly better map in terms of CIEDE2000 distances, as there are 15 pairwise distances in the range [30, 31] in (c) and 9 in (e). However, a few colors in (e) became quite different from the corresponding ones in (a). The changes with the Circle and Tramlink lines are particularly noticeable. Nevertheless, together NM and GA can provide designers with different options to balance between preserving historical symbolism and maximizing perceptual distances.

7.2 Case study #2: Visualizing Seismological Data

The International Seismological Centre (ISC) has been collecting and verifying earthquake data for more than a century. As part of a visualization system for supporting the workflow of a team of data analysts, there are some seven views, each requiring a different colormap. As the data analysts use the system throughout each working day, they need to be able to remember color mappings in each view with little need for checking the legends. Color symbolism thus plays a critical role in supporting familiarization and memorization. Here we describe how color optimization helps the design of two related views, where depths are categorized into different bands.

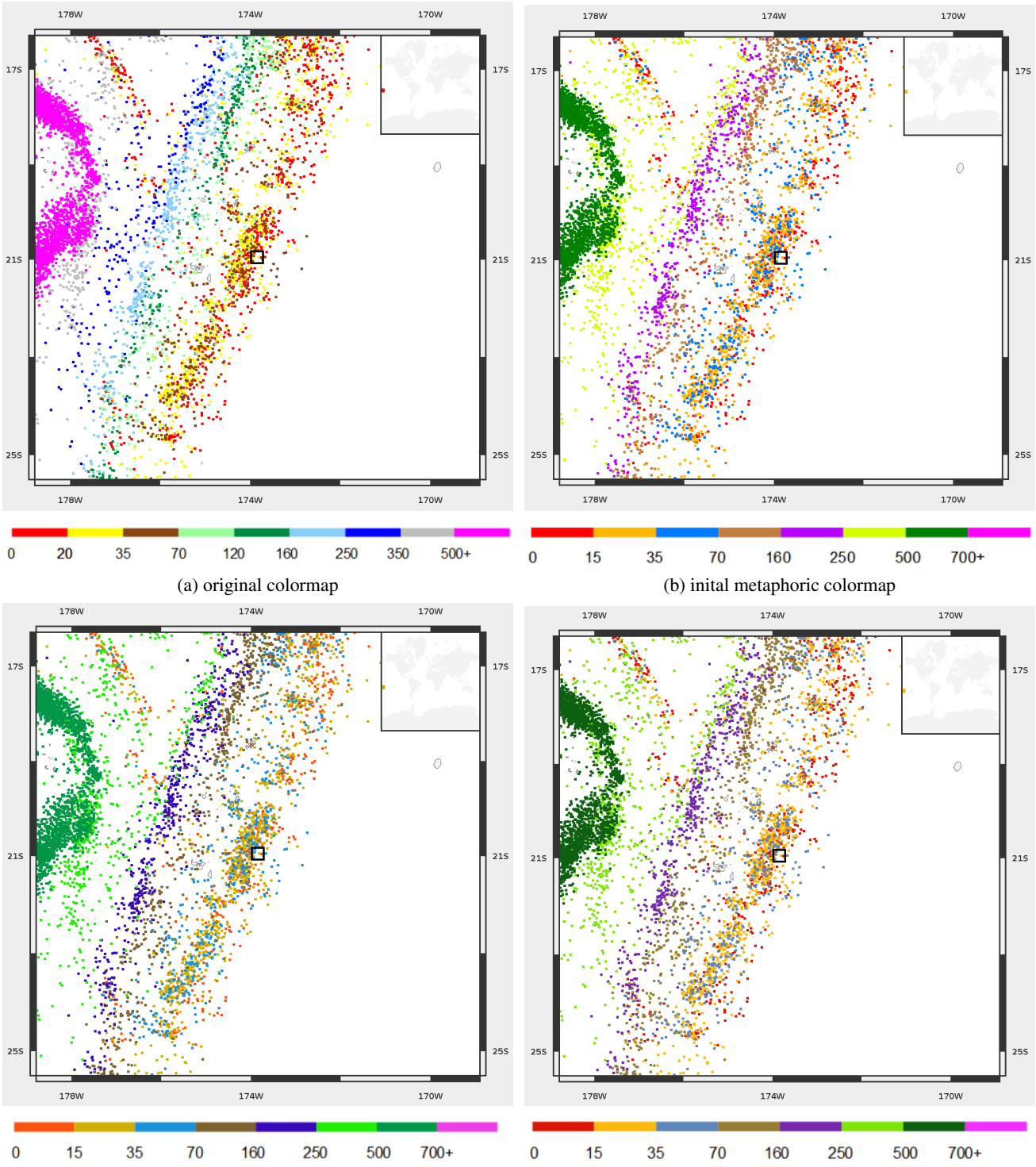
Hypocentres Overview. This view, which is also referred to as the *Historical Seismicity Map*, enables data analysts to examine whether the depth estimations reported by different agencies for a seismic event are consistent with the historical events in the region. Fig. 8(a) shows a view of Hypocentres Overview, with the old colormap that was manually designed by a former staff member many years ago. While the use of a categorical colormap is essential for rapid comparison, there is no record about its design rationale. For scientific and operational reasons, domain experts decided to modify the categorization scheme (from 9 to 8 bands), and took the opportunity to define a new colormap featuring more meaningful color symbolism. Note that the partitioning of the numerical range was purposely designed to be non-uniform, reflecting both geological and precisional factors.

Table 1. Metaphoric colors defined before color optimization.

Depth(km)	Metaphor (Alliterative Association)
Shallow Ranges	
< 15	California (Cherry, Crimson); Shallow (Scarlet)
$[15, 35)$	Mexico (Mango)
$[35, 70)$	Bismarck Sea (Blue)
Intermediate Ranges	
$[70, 160)$	Hindu (Honey)
$[160, 250)$	Andean (Amethyst)
Deep Ranges	
$[250, 500)$	Japan (Jasmine); Deep (Daisy, Daffodil)
$[500, 700)$	Fiji (Forest Green)
≥ 700	Possibly Problematic (Pink)

Table 1 shows a set of modified depth bands, and initial metaphoric colors chosen by the domain experts and visualization designers in a brainstorming meeting. Statistics about different regions at each depth band were made available during the meeting. Each color metaphor was based on the name of a geographical region, which is well-known for having seismic events with depth in a specific band. The name of a region is then mapped to a color using alliterative associations (a form of logology). In some cases, additional semantic meanings, such as "shallow" and "deep", provide additional alliterative associations. For example, historically California had many earthquakes at a shallow depth. Hence the shallowest band (< 15km) is exemplified by this region, and is mapped to red through alliterative associations cherry/crimson/california and scarlet/shallow. Because earthquakes in a shallow range usually cause more damages, the red color also draws attention to such events. Fig. 8(b) shows the same view with the metaphoric colormap resulting from the brainstorming meeting.

In addition, there are three background colors, RGB(255,255,255),



(a) original colormap

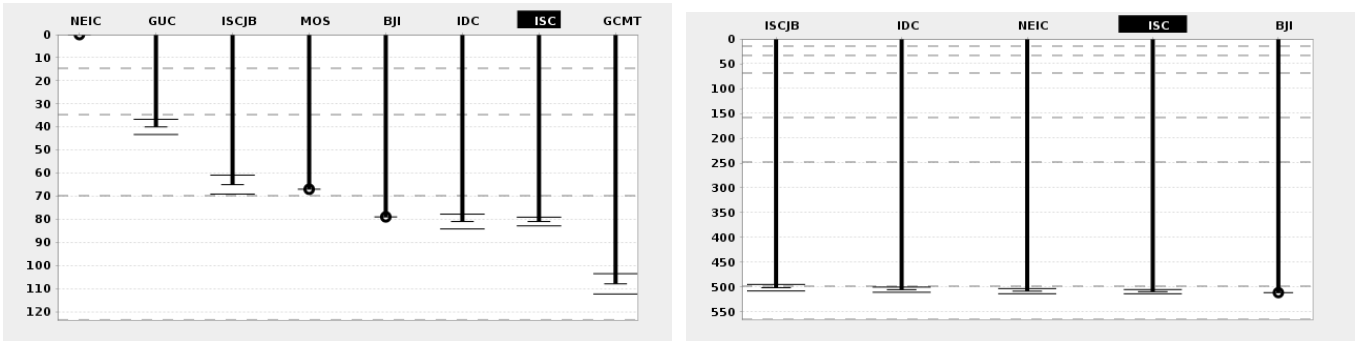
(b) inital metaphoric colormap

(c) optimized colormap with fixed background colors

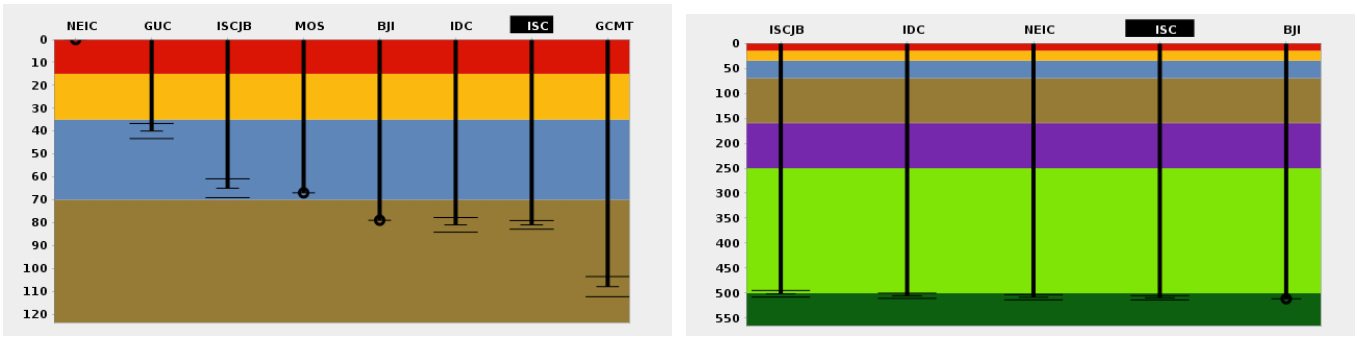
(d) optimized with hue constraint and fixed background colors

Fig. 8. Four versions of a historical seismicity map (of a region near Tonga) exemplify the evolution from the original 9-band map in (a), to an 8-band map with metaphoric colors in (b), to a map optimized without hue constraint in (c), and finally to a map in (d) optimized with hue constraint to preserve color metaphors. By comparing (b) and (d), we can observe that the band [250, 500) becomes more distinguishable from the white background and the band [15, 35) becomes more separable from [0, 15) and [70, 160). Meanwhile, the loss of some color symbolism in (c), e.g., for bands [0, 15) and [15, 35), is avoided in (d).

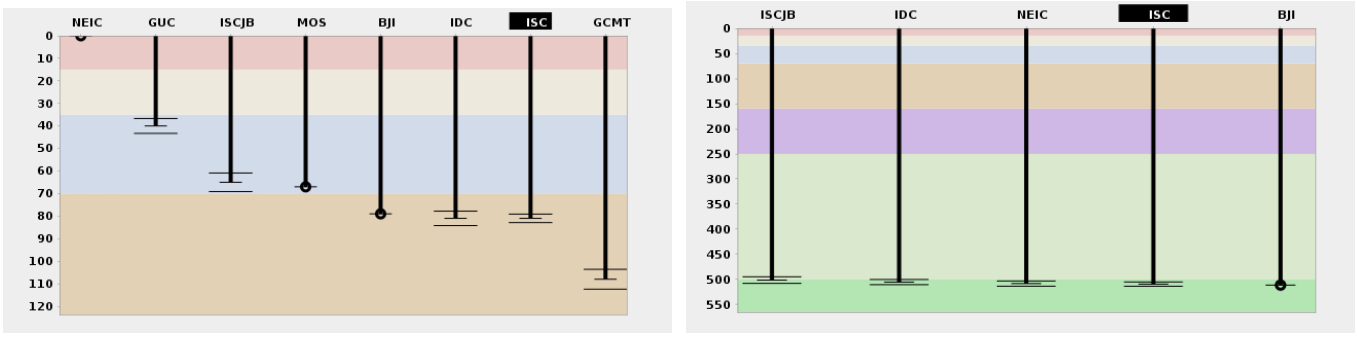
RGB(240,240,240), and RGB(0,0,0), depicting ocean, land, and nation borders respectively. As the colormap is expected to be used for any arbitrary area on the earth, it was difficult to optimize the 11 colors intuitively by inspecting numerous views of different areas. Figs. 8(c, d) show the same view with two colormaps resulting from GA optimization. In the case of (c), we did not set any constraints except fixing the three background colors. This allows colors to be transformed more freely. We can see that the hues of some colors may have shifted too much to maintain the original alliterative associations. For example, the first band becomes more of an orange, connecting to neither california or shallow. We therefore added a $\pm 5\%$ constraint on each hue component of the eight metaphoric colors. As shown in Fig. 8(d),



(a) a traditional design with white background and dashed lines for separating depth bands



(b) a color-coded background using the same metaphoric colormap as the historical seismicity view



(c) a color-coded background using a transformed colormap optimized with user-defined constraints

Fig. 9. The depthmap view of two seismic events: an event at an intermediate depth (left column), and an event at a large depth (right column). (a) The traditional design could not facilitate a rapid perception of different y-scales between the two visualizations. (b) The background color-coded using the colormap in Fig. 8(d) solved the problem in (a), but the colors were not optimal. (c) A visually more pleasing colormap was obtained using the optimization technique with user-defined constraints.

the new colormap maximized the color separation while maintaining the original metaphors and alliterative associations.

Hypocentre Depth View. This view displays a specially designed bar chart for showing the estimated depths of an event reported by different agencies. It is designed for observing potential inconsistencies among different reports. As shown in Fig. 9, the reported depths were listed in an ascending order from left to right. There are two types of depth values, *fixed* or *free*. The former, denoting a heuristic solution, is depicted by a black circle at the bottom of a depth bar. The latter, denoting a computational solution, is depicted by error bars showing the standard deviation.

Domain experts had a critical requirement that the y-axis should be scaled according to the maximum values reported, and horizontal grid lines should correspond to the eight depth bands as discussed above. While the scaling might appear to be counterintuitive to visualization designers (cf. [46, 34]), there were strong scientific reasons for this requirement in this application. For example, estimations for an event at a shallow depth are normally more accurate than those for a deep event. It is necessary to see the smaller depth values in detail, while a coarse visual approximation of larger depth values is acceptable. The same numerical difference between two depth values may suggest high inconsistency in shallow ranges, but fairly acceptable in deep ranges. The same standard deviation would also be judged differently in different ranges. In addition, it was desirable to help data analysts relate a depth value to its categorization (i.e., one of the eight bands).

The initial sketch of the depth view, as illustrated by two examples in Fig. 9(a), used dashed lines as the separator of depth bands. We were unconvinced about its effectiveness as it was difficult to relate a depth value to a depth category. We then color-coded each band of the background canvas using the colormap in Fig. 8(d). The results are shown in Fig. 9(b). Domain experts found this approach highly effective for viewing a depth value with its spatial mapping of length as well as its categorization. However, the dark background colors were thought to be "very hard" on eyes, especially considering the very frequent use of this view in daily tasks. Using GA, we transformed the colormap in Fig. 8(d) to a pastel version by constraining hues with $\pm 5\%$ variation, saturations in the range of [0.3, 0.5], and luminance in the range of [0.8, 0.9]. The results, as shown in Fig. 9(c), achieved better contrast between the background colors and the foreground black for depth bars. In addition, the color-coded background offers another important visual cue for data analysts to judge the level of scaling of the y-axis. Comparing Fig. 9(c), a user can unconsciously become

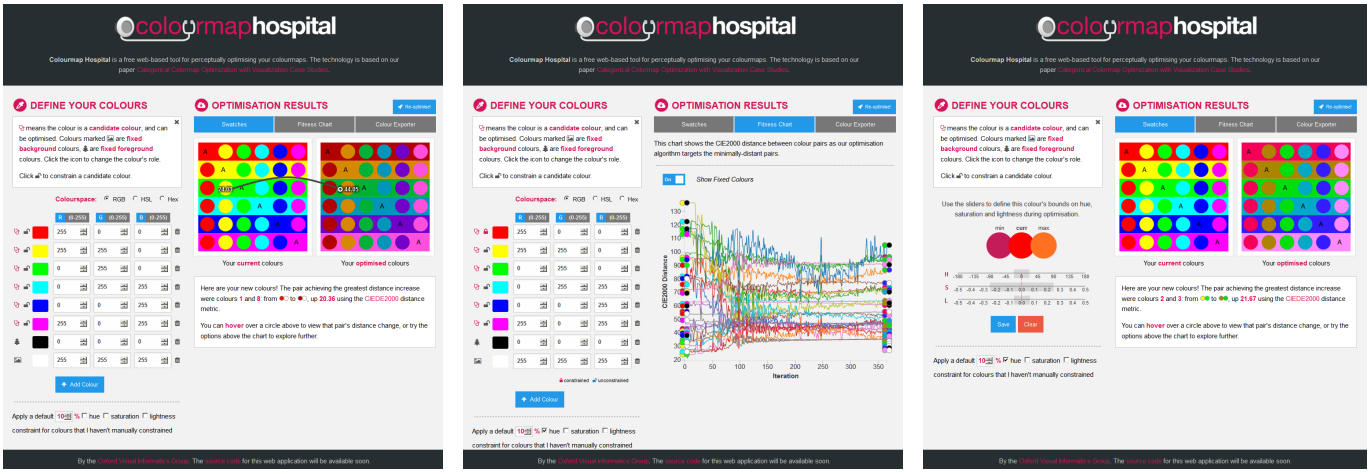


Fig. 10. Three screenshots of *Colourmap Hospital* (http://bit.ly/cmhosp). Left: optimizing six colors with fixed background and foreground. Middle: setting global constraints and viewing the convergence plot. Right: setting individual constraints on the red, and re-optimizing with all constraints.

accustomed to the association between color patterns and levels of scaling. To avoid over-complication in such association, the system always displays a whole band. Hence, there are only 8 scaling factors, and thereby 8 background patterns.

User Feedbacks. The merits of CIE L*a*b* and CIEDE200 have been extensively studied and widely accepted in color science. Hence our evaluation focused on specific sets of colors generated using the aforementioned optimization techniques. As this work was carried out at the ISC where one visualization researcher was embedded in a team of data analysts, we adopted user-centered design and agile development for all views in the visualization system. There were weekly meetings involving the lead analyst, the database manager, and visualization researchers. Major landmarks in design and development were examined in detail through brainstorm meetings or software testing involving other data analysts.

For the colormaps used in the aforementioned two views, we collect feedback from the data analysts through questionnaires and face-to-face meetings. The data analysts confirmed the following merits of color optimization:

- Using metaphors provided significant benefits, including being easy to remember, and easy to articulate when training new analysts and communicating with the external stakeholders.
- When the team assigned the metaphoric meaning to the categorical colormap, there were doubts among the users about the quality of the manually-defined colormap (Fig. 8(b)). After applying the color optimization, the users felt more confident about the quality of the new colormap (Fig. 8(d)).
- Among the three design options shown in Fig. 9), the users overwhelmingly preferred the version in the third row. They welcomed the colormaps featuring the same metaphor but different saturation and luminance, enabling them to relate the Hypocentres Overview and Hypocentre Depth View with ease.
- The optimization tool was also used in designing the colormaps for other views in the system.

8 Colourmap Hospital - A Web-based Tool

In order to make the developed technique available to a wider community, we have produced a web-based tool called *Colourmap Hospital*. The application enables a user to specify a set of colors with constraints, optimize these colors, and view the optimization results. Fig. 10 shows three screenshots of this tool, which is available at `http://bit.ly/cmhosp`.

Users specify the colors to be optimised in either RGB (numeric or hexadecimal) or HSL, and can switch between spaces at any time. A live swatch view showing pairwise comparisons of the colors is updated on the right-hand side during specification. Constraints can be specified with two mechanisms. (i) An individual constraint can be specified by clicking the lock icon to the left of a color. Through three sliders, a user can set $+/-$ ranges around the current hue, saturation and luminance values. (ii) The globally-applicable constraint, once enabled, applies to all colors that do not have an individual constraint. A user can set this constraint by entering a percentage value, and assigning it to hue, saturation or/and luminance. For example, entering 10% and checking the "hue" checkbox places a 36-degree hue boundary around all colors without any individual constraint.

The usability of the tool was evaluated through a consultancy meeting with 10 participants, all of whom had experience of using colormaps in visualization. Following a brief introduction, participants performed five optimization tasks using the tool. A short survey and discussions confirmed the usability strongly in terms of distinguishable colors, and moderately in terms of pleasing or satisfactory results and easy to use. Participants also suggested design aspects to be improved. Further details are given in the supplementary material.

The optimization itself is computed in a remote Python server application. Because of the need to provide color optimization at an interactive rate through a web interface, the tool currently supports only the Nelder-Mead simplex method, since its computational cost is significantly lower than the other two algorithms tested. The optimisation is computed in CIE L*a*b* with CIEDE2000. Testing results have confirmed that the Python implementation is consistent with the MATLAB implementation described in Section 6.

9 Conclusions

Designing a categorical colormap is an elementary yet important task in data visualization. There is no doubt that colormaps designed manually by experienced visualization researchers and visual artists or based the recommendation of software such as *ColorBrewer* are adequate for many applications. Nevertheless, designers can benefit from an optimization tool in a number of scenarios. For example, the tool can be used to refine a design sketch computationally, enabling the designers to focus more effort on other requirements for a colormap (e.g., preserving historical or creating new color symbolisms). It can also be used to check and update the existing colormaps used in various systems. Our two application case studies demonstrate the utility of such a tool.

It is necessary to note that a numerically optimized colormap is not necessarily the most suitable colormap. As shown in Fig. 9, the colormaps in (b) represents an optimized colormap with better distance measure than those in (c). Hence, the humans' judgement about the quality in different contexts is still essential in practice. Nevertheless, with the tool, the colormaps in (b) can be transformed to those in (c) much more easily than with manual transformation.

We wish to further develop the techniques of color optimization. In particular, there are needs for optimizing continuous colormaps.

REFERENCES

[1] L. Bergman, B. Rogowitz, and L. Treinish. A rule-based tool for assisting colormap selection. In *Visualization, 1995. Visualization '95. Proceedings., IEEE Conference on*, pages 118–125, 444, Oct 1995.

[2] R. Berns. *Billmeyer and Saltzmans Principles of Color Technology*. Wiley-Interscience publication. Wiley, 2000.

[3] L. Berry. The interaction of color realism and pictorial recall memory. In *Proc. Association for Educational Communications and Technology*, 1991.

[4] J. Bertin. *Semiology of Graphics*. Esri Press, 1983.

[5] R. Borgo, K. Proctor, M. Chen, H. Jänicke, T. Murray, and I. Thornton. Evaluating the impact of task demands and block resolution on the effectiveness of pixel-based visualization. *IEEE Trans. Vis. Comp. Graphics*, 16(6):963–972, 2010.

[6] D. Borland and R. Taylor. Rainbow color map (still) considered harmful. *Computer Graphics and Applications, IEEE*, 27(2):14–17, March 2007.

[7] C. A. Brewer. Color use guidelines for data representation. *Proceedings of the Section on Statistical Graphics, American Statistical Association*, pages 55–60, 1999.

[8] J. H. C. Demiralp, M. S. Bernstein. Learning perceptual kernels for visualization design. *IEEE Trans. Vis. Comp. Graphics*, 20(12):1933–1942, 2014.

[9] Çağatay Demiralp, M. Bernstein, and J. Heer. Learning perceptual kernels for visualization design. *IEEE Trans. Visualization & Comp. Graphics (Proc. InfoVis)*, 2014.

[10] Central Bureau of the CIE. *Improvement to Industrial Colour Difference Evaluation*. Number 142-2001. 2001.

[11] J. Chuang, D. Weiskopf, and T. Mölla. Energy aware color sets. *Computer Graphics Forum*, 28:203–211, 2009.

[12] W. S. Cleveland and R. McGill. A color-caused optical illusion on a statistical graph. *The American Statistician*, 37(2):101–105, 1983.

[13] J. Cohen and M. Matthen. *Color Ontology and Color Science*. MIT Press, 2010.

[14] D. Cristinacce and T. Cootes. Automatic feature localisation with constrained local modes. *Pattern Recognition*, 41(10):3054–3067, 2008.

[15] J. Eckstut and A. Eckstut. *The Secret Language of Color*. Black Dog & Leventhal Publishers, 2013.

[16] T. for London. *Colour Standard*, volume Issue 3. 2009.

[17] D. Fouskakis and D. Draper. Stochastic optimization: a review. *International Statistical Review*, 70(3):315–349, 2002.

[18] T. Fujiwara. Color space conversion (3): Xyz-lab conversion (in japanese), accessed in June, 2016.

[19] J. Gage. *Colour and Meaning: Art, Science and Symbolism*. Thames and Hudson, 2000.

[20] M. Harrower and C. Brewer. Colorbrewer.org: An online tool for selecting colour schemes for maps. *The Cartographic Journal*, 40(1):27–37, 2003.

[21] C. G. Healey. Choosing effective colours for data visualization. In *Proceedings of the 7th Conference on Visualization '96*, VIS '96, pages 263–ff., Los Alamitos, CA, USA, 1996. IEEE Computer Society Press.

[22] R. W. G. Hunt and M. R. Pointer. *Measuring Colour, 4th Ed.* The Wiley-IS&T Series in Imaging Science and Technology. Wiley, 2011.

[23] R. S. Hunter. Photoelectric color-difference meter. *Journal of the Optical Society of America*, 48(12):985–993, 1958.

[24] A. Kammoun, F. Payan, and M. Antonini. Sparsity-based optimization of two lifting-based wavelet transforms for semi-regular mesh compression. *Computers and Graphics*, 36(4):272–282, 2012.

[25] G. Kindlmann, E. Reinhard, and S. Creem. Face-based luminance matching for perceptual colormap generation. In *Visualization, 2002. VIS 2002. IEEE*, pages 299–306, Nov 2002.

[26] G. A. Klein. *Industrial Color Physics*. Springer, 2010.

[27] G. R. Kuhn, M. M. Oliveira, and L. A. F. Fernandes. An efficient naturalness-preserving image-recoloring method for dichromats. *IEEE Transactions on Visualization and Computer Graphics*, 14(6):1747–1754, 2008.

[28] J. Lagarias, J. Reeds, M. Wright, and P. Wright. Convergence properties of the nelder-mead simplex method in low dimensions. *SIAM Journal of Optimization*, 9(1):112–147, 1998.

[29] S. Lee, M. Sips, and H.-P. Seidel. Perceptually driven visibility optimization for categorical data visualization. *IEEE Trans. Vis. Comp. Graphics*, 19(10):394–404, 2013.

[30] H. Levkowitz and G. Herman. Color scales for image data. *IEEE Computer Graphics and Applications*, 12(1):72–80, 1992.

[31] S. Lin, J. Fortuna, C. Kulkarni, M. Stone, and J. Heer. Selecting semantically-resonant colors for data visualization. *Computer Graphics Forum (Proc. EuroVis)*, 2013.

[32] B. J. Meier, A. M. Spalter, and D. B. Karelitz. Interactive color palette tools. *IEEE Computer Graphics and Applications*, 24(3):64–72, 2004.

[33] T. Munzner. *Visualization Analysis and Design*. A K Peters, 2014.

[34] A. V. Pandey, K. Rall, M. Sattarthwaite, O. Nov, and E. Bertini. How deceptive are deceptive visualizations?: An empirical analysis of common distortion techniques. In *Proc. ACM CHI Conference on Human Factors in Computing Systems (CHI)*, 2015.

[35] D. Pavey and R. Osborne. *Colour Symbolism: Outline History of Its Aesthetics and Psychology*. Lulu.com, 2009.

[36] D. Pineo and C. Ware. Data visualization optimization via computational modeling of perception. *IEEE Trans. Vis. Comp. Graphics*, 18(2):309–320, 2012.

[37] P. C. Q. Wang and M. Green. Familiarity and pop-out in visual search. *Perception & psychophysics*, 56(5):495–500, 1994.

[38] P. Quinlan and G. Humphreys. Visual search for targets defined by combinations of color, shape, and size: an examination of the task constraints on feature and conjunction searches. *Perception & psychophysics*, 41(5):455–527, 1987.

[39] T.-M. Rhyne. Color theory methods for visualization. IEEE VisWeek Tutorial.

[40] P. Robertson and J. OCallaghan. The generation of color sequences for univariate and bivariate mapping. *IEEE Computer Graphics and Applications*, 6(2):24–32, 1986.

[41] M. V. S. Kirkpartrick, C. Gelatt. Optimization by simulated annealing. *Science*, 220(4598):671–680, 1983.

[42] G. Sharma, W. Wu, and E. N. Dalal. The ciede2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research and Application*, 30(1):27–37, 2005.

[43] M. Stone. *A Field Guide to Digital Color*. Ak Peters Series. A.K. Peters, 2003.

[44] A. Treisman and S. Gormican. Feature analysis in early vision: evidence from search asymmetries. *Psychological Review*, 95(1):15–48, 1988.

[45] E. Tufte. *Envisioning information*. Graphics Press, 1990.

[46] E. R. Tufte. *The Visual Display of Quantitative Information*. Graphics Press USA, 2001.

[47] L. Wang, J. Giesen, K. McDonnell, P. Zolliker, and K. Mueller. Color design for illustrative visualization. *Visualization and Computer Graphics, IEEE Transactions on*, 14(6):1739–1754, Nov 2008.

[48] C. Ware. Color sequences for univariate maps: Theory, experiments and principles. *IEEE Computer Graphics and Applications*, 8:41–49, 1988.

[49] C. Ware. *Information Visualization: Perception for Design, 2nd edition*. Morgan Kaufmann, 2004.

[50] L. Williams. The effects of target specification on objects fixated during visual search. *Acta psychologica*, 27(1):355–415, 1967.

[51] H. Xu, H. Yaguchi, and S. Shioiri. Testing cielab-based color-difference formulae using large color differences. *Optical Review*, 8(6):487–494, 2001.

[52] Y. Yang, J. Ming, and N. Yu. Color image quality assessment based on ciede2000. *Adv. in MM*, 2012, 2012.

[53] S. Y. Zhu, M. R. Luo, and G. Cui. New experimental data for investigating uniform color spaces. volume 4421, pages 626–629, 2002.