PharmaSUG 2026 - Paper DV-232

# AI-Recommended Color Palettes with QC for R Shiny Figures

Yi Guo, Pfizer Inc.

## ABSTRACT

In the pharmaceutical industry, the safe and controlled use of artificial intelligence (AI) remains an active topic of discussion. If AI is viewed as a programmer, concerns often focus on unforeseen bugs and outputs. When viewed as an artist, AI can offer unexpected inspiration and expand creative possibilities.

In this paper, we use AI as an artist-inspired tool to extend the dynamic color selection feature of an existing R Shiny figures application. Using the {openai} package, the app calls a large language model (LLM) via an API to generate candidate color palettes based on user-defined themes. Within the app, each subgroup is assigned a unique color picker with a default color and an option for manual adjustment, while natural language prompts are used to guide the LLM in generating candidate color palettes for subgroup-level visualization.

However, this alone is not sufficient, as an artist's aesthetic choices may not always meet regulatory submission requirements. To address this, we introduce a QC-oriented evaluation approach based on established color metrics, with primary emphasis on minimum thresholds for luminance difference ($\Delta\mathrm{L}$) and color difference ($\Delta\mathrm{E}$). Contrast ratio is used as a supporting check, with luminance as a reference. Color difference for colorblindness ($\Delta\mathrm{E}_{(cb)}$) is optionally considered.

Rather than treating AI outputs as final results, these metrics provide quantitative QC references to support interactive, human-in-the-loop color selection in regulated visualization workflows. By complementing visual judgment with reference values, this approach improves efficiency, reduces reviewer-to-reviewer variability, and lowers the chance of missing good color options.

## INTRODUCTION

Bad color can destroy good data. Figures, unlike tables and listings, provide a visual summary that helps reviewers quickly understand and interpret study results. By using the appropriate colors, trends or patterns become more visible and comparable. Poor color choices, however, can compromise clarity and weaken interpretability.

Currently, there is no direct guidance on color selection for figures in clinical trial reporting or regulatory submissions. Even when this area is mentioned, it is only briefly addressed, typically limited to very basic technical and readability considerations. For example, the FDA's Portable Document Format (PDF) Specifications document only requires that files be readable in grayscale and meet minimum image resolution requirements. In addition, Duke et al. (2015) provided general recommendations for using color to differentiate key data elements (e.g., treatment effects across study groups), emphasizing the importance of visual clarity in medical graphics. However, these recommendations focus on design principles rather than quantitative evaluation criteria within regulated workflows.

As a result, color selection often relies on manual trial-and-error, requiring iterative adjustments to ensure both visual clarity and grayscale compatibility. In many cases, choices are influenced by experience, existing templates, or therapeutic area conventions. For example, red is commonly used in figures, but it may be avoided in some contexts. One reason is that red is commonly associated with abnormal findings or safety signals. In addition, red appears similar to dark gray when printed in grayscale, and it can be also difficult to tell apart from green for people with color vision deficiency.

Although many predefined palettes are widely used in data visualization, including ColorBrewer, the viridis family, Tableau categorical palettes, and colorblind-friendly schemes such as Okabe-Ito, they are intended for general visualization purposes rather than regulated clinical reporting. While some support print-friendliness, perceptual uniformity, or accessibility, these features do not necessarily guarantee clear separation across multiple subgroups or distinguishability in grayscale output. As a result, palette selection for multi-subgroup figures may still be time-consuming and subjective in practice.

These considerations highlight the need for more structured and objective approaches to color evaluation in regulated figures. Therefore, we propose an AI-assisted framework for color palette generation within an existing R Shiny-based figure application. In this context, AI is applied to a presentation-level task, with outputs remaining fully reviewable and adjustable by the user. The palettes are evaluated using color science metrics informed by relevant regulatory and industry guidance. These metrics provide structured QC guidance that complements visual judgment and supports human-in-the-loop color selection in regulated workflows. The implementation is model-agnostic and can align with organization-approved LLM services or controlled environments.

## PURPOSE AND INTENDED USE OF AI IN THIS PAPER

AI adoption in clinical decision-making workflows is still evolving in the pharmaceutical industry. Although more governance frameworks are being proposed, expectations regarding validation, oversight, and predefined performance characteristics are still developing across organizations.

Clinical decision making, particularly statistical inference, requires strict control of Type I error ($\alpha$) while maintaining adequate power (1-$\beta$). As these decisions directly affect the results and their interpretations, well-established validation frameworks are essential. Although these concepts are not directly applicable to visualization or AI-assisted design works, they reflect a broader principle of risk control and reliability. Similarly, current regulatory visualization tools, such as eDish plots, adopt a conservative design philosophy intended to highlight potential safety signals rather than filter them.

In line with this principle, the “artist” AI is used to generate candidate palettes, which are then evaluated using predefined QC metrics. It improves visualization while having no impact on inferential operating characteristics, because it does not participate in data preparation or statistical computation. Therefore, this novel, structured framework provides a practical, low-risk entry point for AI adoption.

## HUMAN-IN-THE-LOOP WORKFLOW

The human-in-the-loop workflow integrates AI-assisted palette recommendation, manual color selection, and quantitative QC evaluation (Figure 1). Candidate colors are assessed against predefined criteria, and users can iteratively adjust colors until requirements are met. This ensures that final color selection remains under user control, with AI serving only as a recommendation tool and decisions guided by objective metrics.

The remainder of the paper is organized into two main parts: the generation layer (AI) and the evaluation layer (QC metrics), with practical visualization examples demonstrating their application.

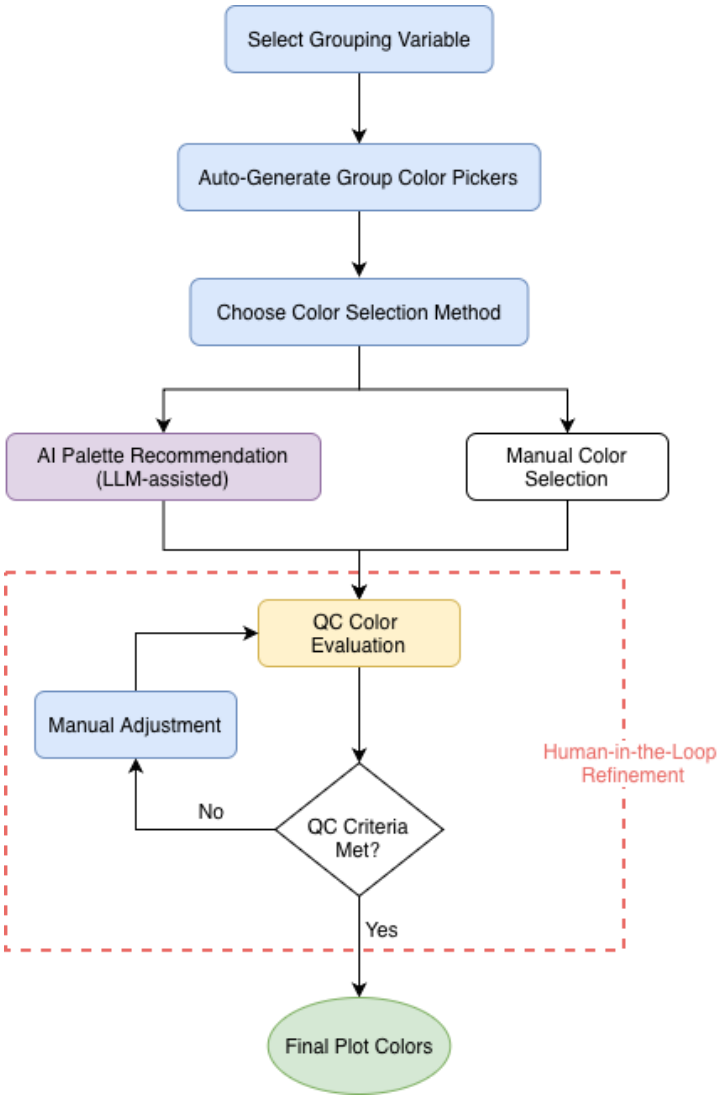



**Figure 1. Human-in-the-loop workflow for AI-recommended and manually selected color palettes with QC-guided iterative refinement in an R Shiny figures application.**

## AI INTEGRATION ARCHITECTURE

This section presents the generation layer of the proposed framework, including AI package selection, system architecture within the Shiny environment, reactive prompt design, API communication, and dynamic color injection for AI-assisted color palette generation.

### AI PACKAGE SELECTION

Because downstream QC requires strictly formatted HEX vectors, structured JSON output was treated as an essential design requirement to ensure consistent parsing and evaluation of generated color palettes. The {openai} package was selected in this work because it provides a stable R interface for API interaction and integrates easily with the Shiny framework.

## SYSTEM ARCHITECTURE IN THE SHINY ENVIRONMENT

The AI module operates within the visualization layer of the Shiny application and remains separate from statistical computation and dataset derivation processes. Structured API calls are triggered based on user-defined inputs to generate candidate color palettes. Model responses are parsed into machine-readable HEX values using {jsonlite} and passed to the visualization layer for rendering. This design enables adaptive visual customization while preserving the integrity of the underlying analytical workflow.

## API INTEGRATION AND SECURITY

API credentials are stored securely in a project-level environment file rather than embedded in source scripts, ensuring protection of sensitive information during development and version control.

### (1) API Key Setup

Before integration, an API key must be generated from the OpenAI developer platform and kept confidential. The key is stored in the project-level `.Renviron` file rather than written directly in R scripts, preventing exposure of sensitive credentials.

```
install.packages("usethis")
usethis::edit_r_environ()
```

After running the two lines of code above, run `Sys.getenv("OPENAI_API_KEY")` to verify that the API key is accessible within the R environment.

```
Sys.getenv("OPENAI_API_KEY")
```

### (2) Package Initialization

After setting up the API key, the {openai} package is installed and loaded.

```
install.packages("openai")
library(openai)
library(jsonlite)
```

## REACTIVE PROMPT DESIGN

The prompt is implemented using separate system and user messages. The system message defines the model role and output constraints, while the user message provides task-specific inputs.

The user prompt is dynamically constructed based on user-defined inputs, including the number of subgroup levels (automatically determined from the selected grouping variable, `n_groups`) and a user-provided palette description (`input$palette_prompt`). The palette description may specify the desired theme or stylistic preference, as well as practical constraints such as visual distinguishability or colorblind-friendly considerations.

### (1) Prompt Design Guidance

To ensure reliable structured responses, the prompt follows several design principles commonly used in LLM prompting.

a. **Role instruction**: The model is assigned a role to guide the task (e.g., a color palette assistant for statistical graphics in clinical trials). This is defined in the system message.
b. **Color count specification**: The required number of colors is specified explicitly based on the detected number of subgroup levels (e.g., `n_groups`).
c. **Structured JSON output**: The output is restricted to HEX color codes returned as a JSON array so that it can be parsed by the downstream QC workflow.
d. **Example output**: An example output is included to reinforce the expected structure and prevent the model from returning explanatory text (e.g., `[\"#112233\", \"#445566\"]`).

```
observeEvent(input$ask_ai, {
  req(input$palette_prompt)
  lv <- group_levels()
  req(lv)
  n_groups <- length(lv)

  system_prompt <- paste0(
    "You are a color palette assistant for statistical graphics in clinical trials.",
    "You respond ONLY with valid JSON."
  )

  user_prompt <- paste0(
    "There are ", n_groups, " groups in the plot.",
    "Based on this description: '", input$palette_prompt, "', ",
    "return EXACTLY ", n_groups, " contrasting hex color codes as a JSON array of
     strings.",
    "Ensure sufficient luminance difference, contrast and color difference for
     readability in both grayscale and color printing.",
    "No explanation, no extra text. Example output: [\"#112233\", \"#445566\"]."
  )

  ...
})
```

## API COMMUNICATION

In this implementation, a lightweight chat model (e.g., gpt-4o-mini) was used for palette generation to support efficient responses in the Shiny application. Other OpenAI models can be substituted with minimal modification.

```
observeEvent(input$ask_ai, {
  ...
  user_prompt <- paste0(...)
  system_prompt <- paste0(...)

  ai_resp <- create_chat_completion(
  model = "gpt-4o-mini",
  temperature = 0,
  messages = list(
    list(role = "system", content = system_prompt),
    list(role = "user", content = user_prompt)
```

```
    )
  )
  ...
})
```

### DYNAMIC COLOR INJECTION

After the API call returns the generated palette, the HEX color codes are parsed and mapped to the detected subgroup structure. For each subgroup, a `colourInput()` control is created in the user interface with an initial default color. When the AI-generated palette is returned, the controls are programmatically updated using `updateColourInput()`. Users can still manually adjust the colors if needed. The finalized color vector is then passed to the plotting function.

```
observeEvent(input$ask_ai, {
  ...
  user_prompt <- paste0(...)
  system_prompt <- paste0(...)

  ai_resp <- create_chat_completion(...)

  #Extract response from AI
  txt <- ai_resp$choices[[1]]$message$content

  #Parse JSON into HEX palette
  cols <- jsonlite::fromJSON(txt)

  #Inject colors into UI
  for (i in seq_len(n_groups)) {
  updateColourInput(
    session,
    inputId = paste0("col_", i),
    value   = cols[i]
   )
  }

  ...
})
```

## COLOR SCIENCE METRICS FOR QUALITY CONTROL

This section presents the evaluation layer of the proposed framework. While the generation layer introduces candidate color palettes through AI-assisted prompting, the evaluation layer applies quantitative color science principles to assess their suitability for regulated visualization workflows.

### SELECTION AND RATIONALE FOR COLOR EVALUATION METRICS

Regulatory guidance for electronic submissions to the U.S. Food and Drug Administration emphasizes that graphical elements must remain distinguishable in both color and grayscale formats, as reviewers may examine documents in black and white.

Therefore, color difference ($\Delta E2000$) and luminance difference ($\Delta L$) were used as the primary metrics to ensure clear visual separation in both color and grayscale conditions. Contrast ratio relative to a white background, derived from relative luminance ($Y$), was included as a supporting check to ensure sufficient visibility. Relative luminance ($Y$) and color difference under color vision deficiency ($\Delta E_{2000(cb)}$) were included as reference and optional metrics, respectively.

The selected metrics are described below, and the corresponding R implementations are provided in Appendix 1.

#### (1) Relative Luminance ($Y$)

Hue and chroma define the perceptual differences between colors in a color display, while relative luminance ($Y$) determines how those colors are perceived in grayscale. Relative luminance is not a simple RGB average (e.g., $(R + G + B)/3$). It is computed as a weighted combination of red, green, and blue channels, reflecting human sensitivity to brightness differences. $Y$ is computed as follows:

$$Y = 0.2126R + 0.7152G + 0.0722B$$

where R, G, and B are normalized RGB values in the range [0,1].

For example, pure red (1, 0, 0) has a luminance value of 0.2126, whereas pure green (0, 1, 0) has a luminance value of 0.7152. Although both colors are equally intense in RGB magnitude, green appears substantially brighter in grayscale due to its higher perceptual weighting (Figure 2).

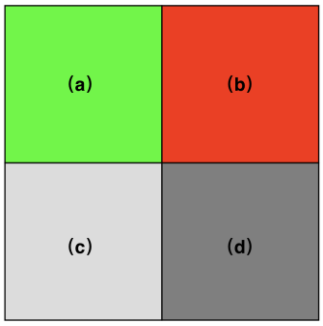



**Figure 2. Comparison of color palettes and their grayscale representations. (a) Green palette and (b) red palette, with corresponding grayscale conversions shown in (c) and (d).**

### (2) Luminance difference ($\Delta L$)

While luminance describes the brightness of an individual color, grayscale distinguishability depends on differences in luminance between colors. Therefore, pairwise luminance difference ($\Delta L$) was calculated as:

$$\Delta L = \left|Y_i - Y_j\right|$$

where $Y_i$ and $Y_j$ represent the luminance values of the two colors.

A larger $\Delta L$ indicates greater separation in grayscale. Because grayscale rendering preserves luminance but removes hue and chroma, $\Delta L$ directly reflects robustness in black-and-white reproduction.

### (3) Contrast Ratio

While relative luminance ($Y$) measures the brightness of an individual color, contrast ratio evaluates the separation between an element color and its background and is calculated as:

$$Contrast\ Ratio = \frac{L_1 + 0.05}{L_2 + 0.05}$$

where $L_1$ and $L_2$ denote the relative luminance ($Y$) of the lighter and darker colors, respectively.

In clinical trial reporting and publications, figures are typically presented on a white background. In this setting, the figure element color is $L_2$, since no color has higher luminance than white. For white (RGB = 255, 255, 255), after normalization (R = G = B = 1), the relative luminance is $L_1 = 1$.

For example, pure yellow (#FFFF00) has very high luminance ($Y \approx 0.93$), which is close to white ($Y = 1.0$). As a result, the contrast ratio between yellow and a white background is low (approximately 1.07:1). When converted to grayscale, yellow appears as a very light gray, reducing visual separation. For reference, Table 1 summarizes the relative luminance and contrast ratios against a white background for four representative colors.

**Table 1. Example Luminance and Contrast Values for Representative Colors**

| Color | Relative Luminance ($Y$) | Contrast Ratio vs White |
|---|---|---|
| Yellow (#FFFF00) | ~0.93 | ~1.07:1 |
| Green (#00FF00) | ~0.72 | ~1.37:1 |
| Red (#FF0000) | ~0.21 | ~4.0:1 |
| Black (#000000) | 0.00 | 21:1 |

### (4) Color Difference ($\Delta E_{2000}$)

While luminance difference and contrast ratio address grayscale robustness and background visibility, full-color distinguishability requires evaluation of perceptual color difference. Color difference ($\Delta E$) is defined in the CIELAB color space by the International Commission on Illumination (CIE) and quantifies the perceptual distance between two colors.

In the CIELAB space, the original color difference formulation ($\Delta E_{76}$) is defined as:

$$\Delta E_{76} = \sqrt{(L_1^* - L_2^*)^2 + (a_1^* - a_2^*)^2 + (b_1^* - b_2^*)^2}$$

where $\mathrm{L}^*$, $\mathrm{a}^*$, and $\mathrm{b}^*$ represent lightness and chromatic components.

However, $\Delta\mathrm{E}_{76}$ does not fully account for perceptual non-uniformity. Among several formulations, $\Delta\mathrm{E}_{2000}$ (CIEDE2000) incorporates corrections for lightness, chroma, and hue interactions and is widely regarded as the most perceptually accurate standard. In this study, $\Delta\mathrm{E}_{2000}$ was used to assess pairwise color distinguishability under full-color display conditions. Larger $\Delta\mathrm{E}$ values indicate greater perceptual separation.

***Ordering of $\Delta L$ and $\Delta E$: Rationale***

Luminance difference ($\Delta\mathrm{L}$) is evaluated before $\Delta\mathrm{E}$, as grayscale separation serves as a foundational layer of visual robustness. In the CIELAB color space, lightness ($\mathrm{L}^*$) is derived from luminance ($\mathrm{Y}$) through a monotonic transformation; therefore, a large luminance difference corresponds to a large lightness difference. Since $\Delta\mathrm{E}$ incorporates lightness difference as one of its components, $\Delta\mathrm{L}$ is assessed first to ensure grayscale robustness prior to evaluating full-color perceptual separation.

### (5) Color Difference for Color Vision Deficiency ($\Delta\mathrm{E}_{2000(\mathrm{cb})}$)

To further assess perceptual robustness, color difference was optionally evaluated under simulated color vision deficiency (CVD) conditions, including protanopia, deuteranopia, and tritanopia. Pairwise $\Delta\mathrm{E}_{2000}$ values were calculated after applying each simulation to the original colors.

Unlike $\Delta\mathrm{E}_{2000}$ under normal vision, color differences under CVD simulations are more variable and model-dependent, with no universally accepted thresholds. These values are therefore used to support qualitative assessment rather than strict criteria.

In this work, all CVD-based $\Delta\mathrm{E}_{2000}$ metrics were treated as optional indicators. Among these, protanopia and deuteranopia represent the most common forms of color vision deficiency, while tritanopia is less prevalent. When trade-offs are required, users may focus on protanopia and deuteranopia results, particularly when it is not feasible to optimize color distinguishability across all simulated conditions.

## PRIORITY AND THRESHOLD RECOMMENDATIONS

Table 2 summarizes the evaluation metrics, their relative priority, and their recommended thresholds. For separation metrics ($\Delta\mathrm{L}$, $\Delta\mathrm{E}_{2000}$, and $\Delta\mathrm{E}_{2000(\mathrm{cb})}$), the minimum pairwise value across all color combinations is evaluated to assess worst-case distinguishability.

Relative luminance ($\mathrm{Y}$) is used to avoid proximity to background brightness, while contrast ratio provides a direct measure of visibility against the background. For both metrics, worst-case values are reported in the Shiny application.

**Table 2. Color Quality Control Metrics and Recommended Thresholds**

| Color Metrics | Purpose | Recommended Reference | Role | Priority |
|---|---|---|---|---|
| Relative Luminance ($\mathbf{Y}$) | Background proximity control; avoid blending with background | Avoid proximity to background luminance (e.g., $Y \leq 0.90$ on white background) | Reference | * |
| Luminance Difference ($\mathbf{\Delta L}$) | Grayscale separation | ≥ 0.05 (normalized to a 0-1 scale) | Primary check | *** |
| Contrast Ratio | Background visibility | Contrast vs white background ≥ 2:1 | Supporting check | ** |
| Color Difference ($\mathbf{\Delta E_{2000}}$) | Full-color perceptual separation | ≥ 10 preferred | Key metric | *** |
| Color Difference for CVD ($\mathbf{\Delta E_{2000(cb)}}$) | Colorblind robustness (CVD simulation) | ≥ 8 preferred | Optional check | ** (advisory) |

Note: $\mathbf{\Delta E_{2000(cb)}}$ values include protanopia, deuteranopia, and tritanopia.

The threshold values were selected based on a combination of established guidelines, published practices, and empirical observations, and are intended as practical reference points rather than strict universal criteria. Key considerations underlying these selections are discussed below.

**Color Difference ($\mathbf{\Delta E_{2000}}$)**: A threshold of $\Delta E_{2000} \geq 10$ was used as a practical criterion under normal vision, informed by commonly referenced perceptual ranges and corresponding to clearly distinguishable colors.

**Table 3. $\mathbf{\Delta E_{2000}}$ Ranges and Perceptual Interpretation**

| $\mathbf{\Delta E_{2000}}$ Value | Perception |
|---|---|
| 0 | No difference, colors are identical |
| < 3 | Limited perceptual difference |
| 3-10 | Noticeable difference |
| ≥ 10 | Clear visual separation (used in this study) |
| 100 | Colors are exact opposite |

**Luminance difference** ($\mathbf{\Delta L}$): Larger differences (e.g., ≥ 0.10) provide more robust separation; however, when multiple colors (typically more than three) are used, it may be difficult to satisfy stricter thresholds across all pairs. Therefore, a minimum threshold of 0.05 was adopted as a practical criterion to balance perceptual distinguishability and feasibility.

**Contrast Ratio**: A minimum contrast ratio of approximately 1.6:1 against a white background was used as a practical baseline, informed by empirical observation of the default base R palette (`palette()`) (Figure 3). Higher thresholds (e.g., 3:1 or 4.5:1), typically recommended for text readability, may be overly restrictive for multi-group graphical elements. Therefore, a relaxed threshold was adopted, with

distinguishability supported by perceptual color difference ($\Delta E$) and luminance-based metrics.

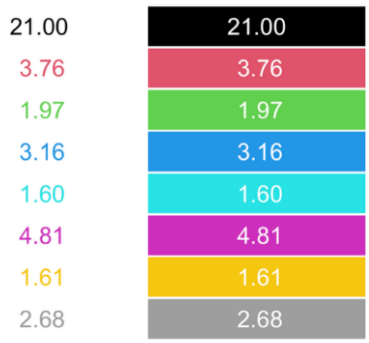



**Figure 3. Contrast ratios of colors in the default base R palette (`palette()`) against a white background. All colors exceed a contrast ratio of 1.6:1 and are visually distinguishable, supporting its use as a relaxed empirical baseline.**

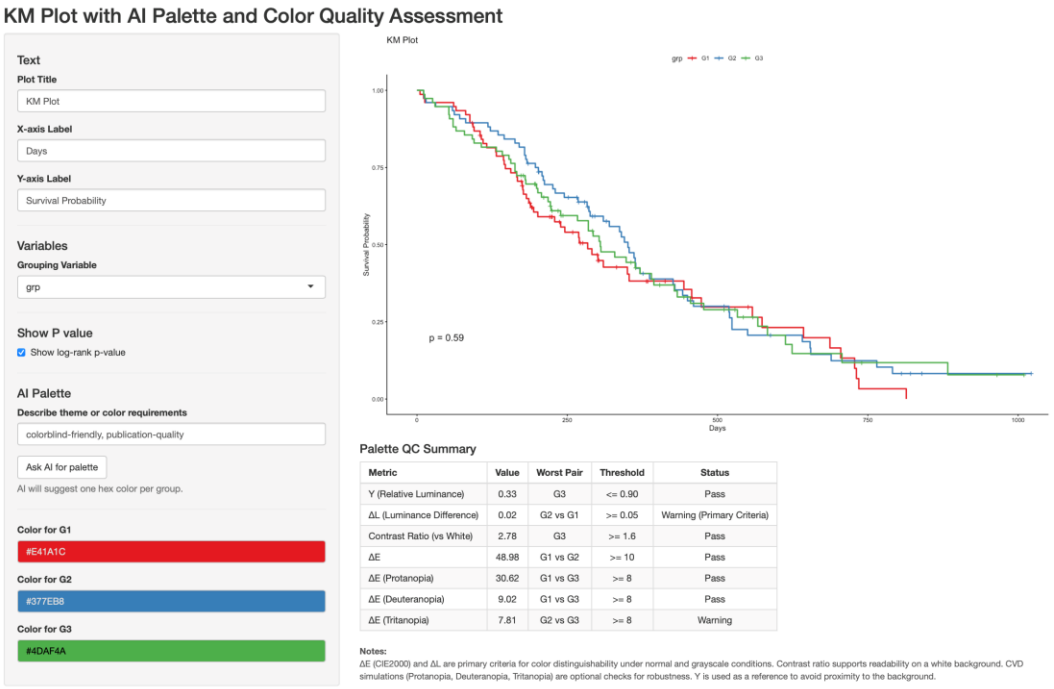
**Color Difference for CVD ($\Delta E_{2000(cb)}$)**: Under color vision deficiency (CVD) simulations, perceptual color relationships are altered due to distortion of the color space, and distances between colors may either decrease or increase depending on the color pair. Therefore, a slightly relaxed threshold (≥ 8) was used as a practical screening criterion, compared to the ≥ 10 threshold under normal vision. This choice balances perceptual distinguishability with the practical constraints of multi-group visualizations.

**Relative luminance ($Y$)**: A value of Y ≤ 0.90 (on a white background) was adopted as a practical guideline. It is used as a reference check to avoid colors being overly close to the white background.

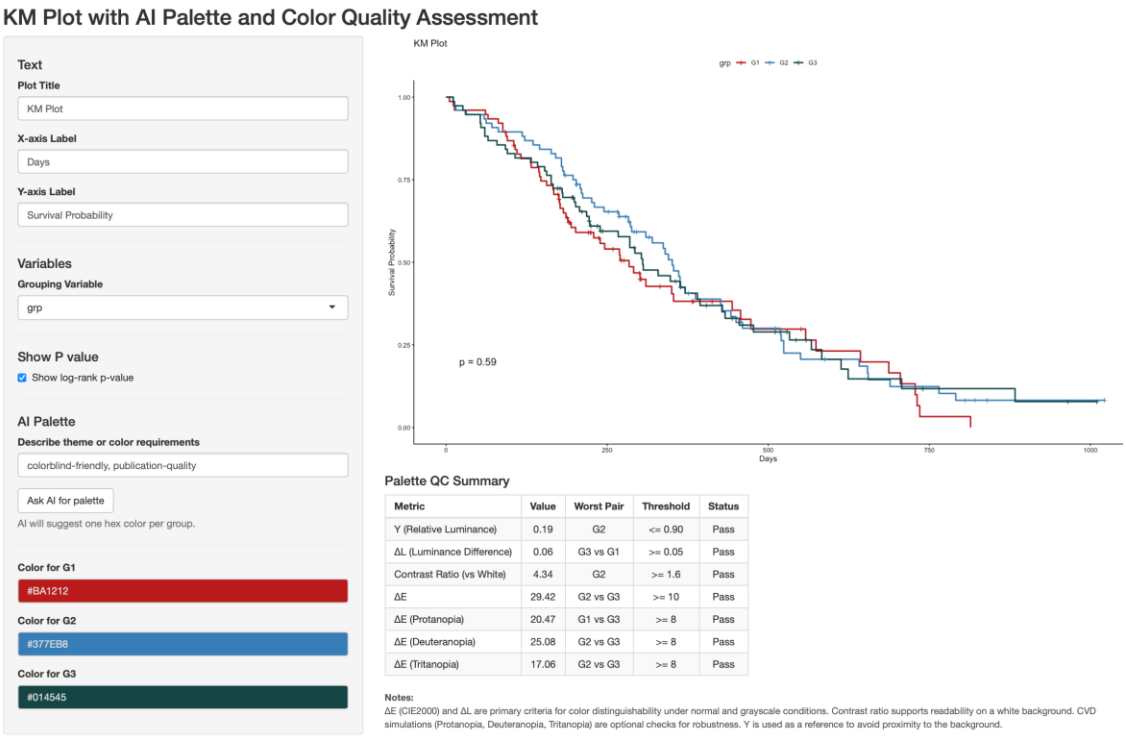
## MULTI-SUBGROUP VISUALIZATION EXAMPLES

### KAPLAN–MEIER (KM) PLOT

In this example, a KM plot based on simulated data demonstrates the color selection workflow for multi-subgroup visualization. Figures 4A and 4B represent two stages of this process. Figure 4A shows the initial color palette generated using “Ask AI for palette” feature with default settings; however, it does not fully meet QC standards. Figure 4B shows the refined version after manual adjustment guided by QC color metrics.




**(A) Initial color palette generated using the "Ask AI for palette" feature with the default prompt "colorblind-friendly, publication-quality".**

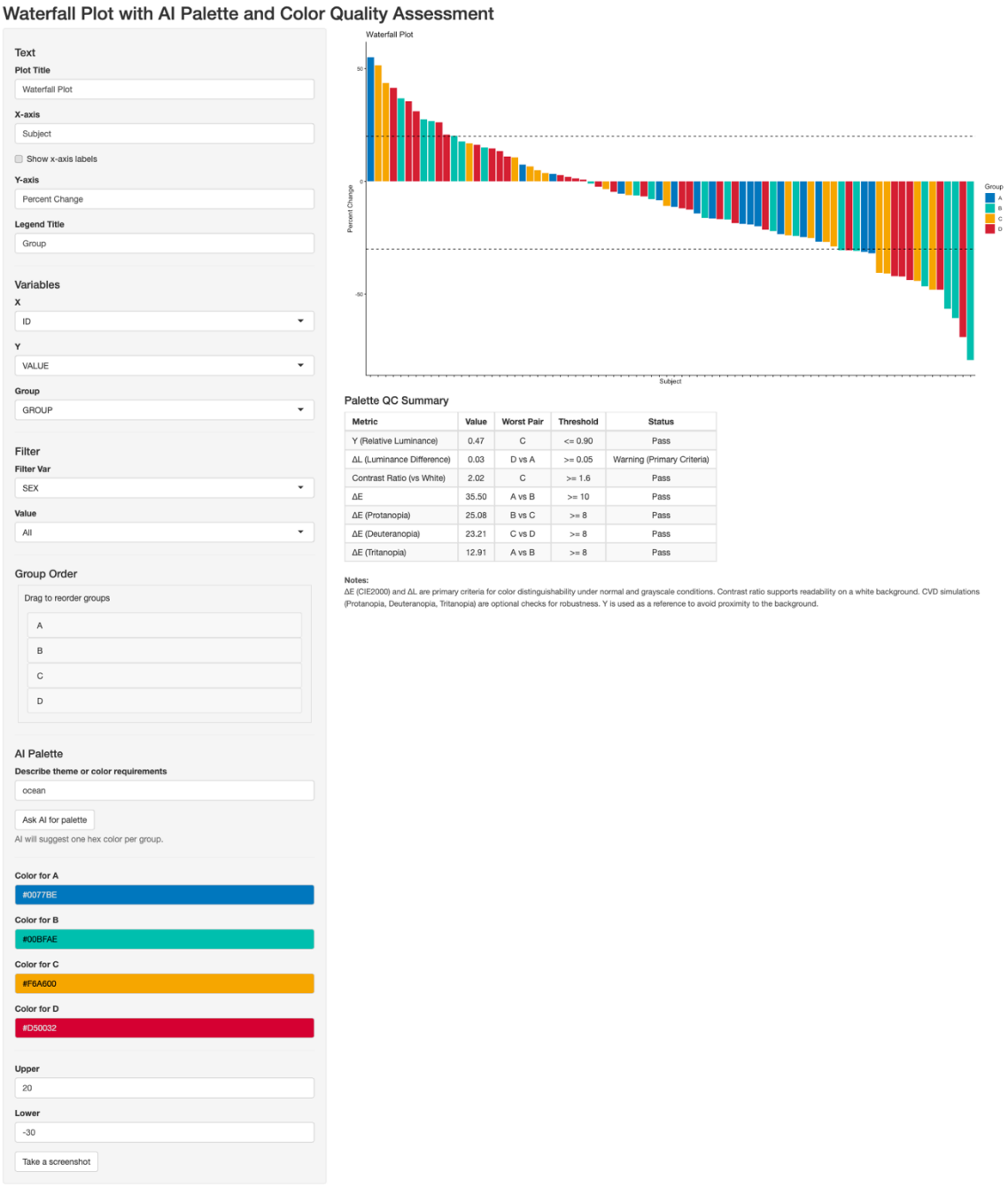



**(B) Final color palette after manual refinement guided by QC color metrics.**

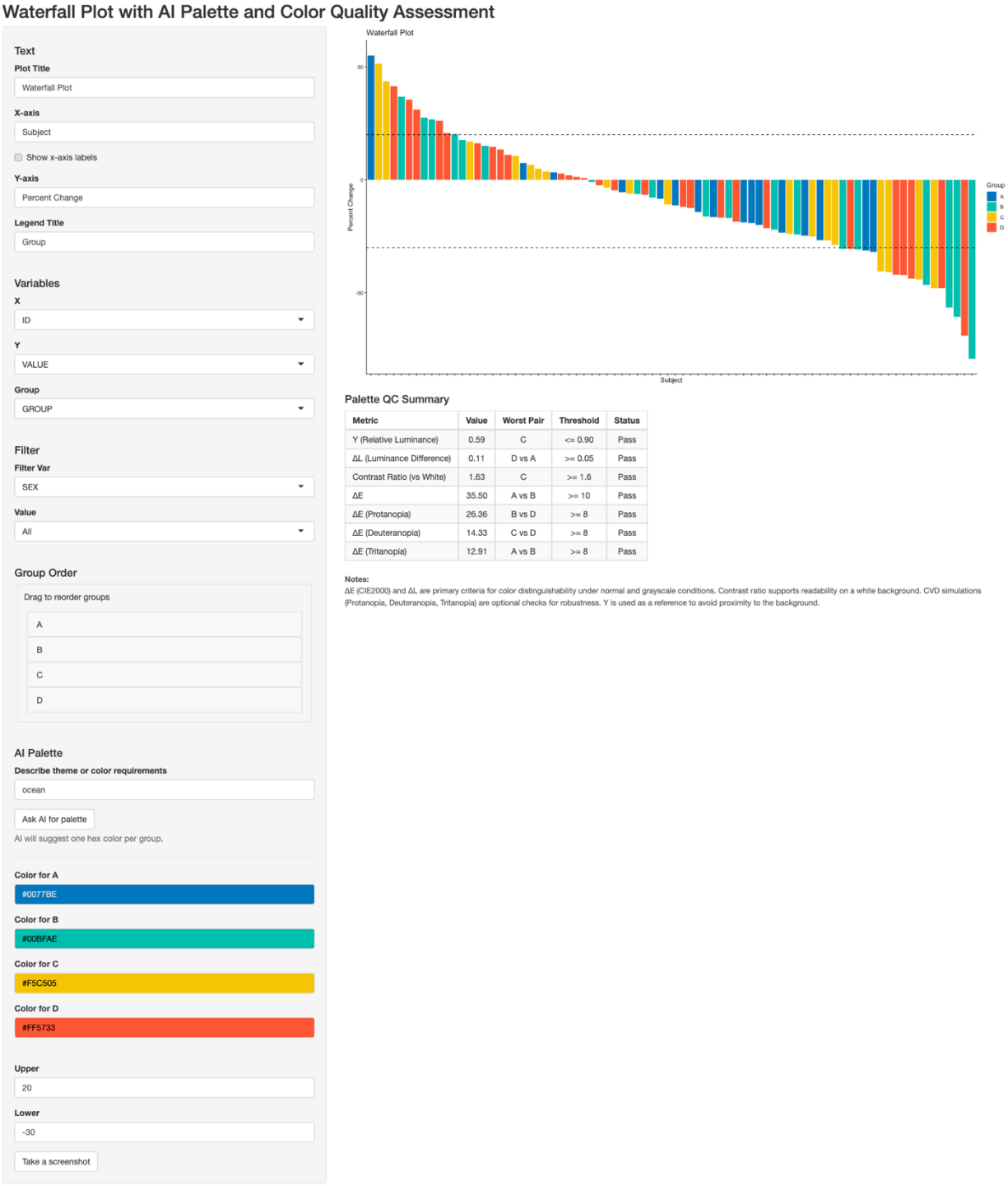
**Figure 4. KM plot color palettes before and after refinement.**

## WATERFALL PLOT

Another example uses a waterfall plot with four subgroups based on simulated data. In Figure 5A, the palette generated using the keyword “ocean” shows insufficient luminance separation between group A and D. This affects grayscale distinguishability. After a minor adjustment guided by QC metrics, the updated palette in Figure 5B achieves acceptable visual separation.




**(A) Initial color palette generated using the “Ask AI for palette” feature with a modified prompt (“ocean”); minimum pairwise luminance difference does not meet the threshold.**




**(B) Refined color palette after minor QC-guided adjustments; improved grayscale separation.**

**Figure 5. Waterfall plot color palettes before and after refinement.**

## PRATICAL CONSIDERATIONS FOR COLOR SELECTION

### AI-ASSISTED PALETTE GENERATION

When using AI to generate candidate color palettes, clear and specific prompts are essential. Vague or high-level descriptions may lead to inconsistent or less useful results. In practice, users are encouraged to provide explicit requirements, such as preferred color themes, contrast expectations, or constraints on specific groups (e.g., assigning a specific color to a group). More structured input increases the likelihood that the generated palettes align with intended visualization goals and reduces the need for extensive manual adjustment.

### APPLICATION OF QC METRICS

As the number of subgroups increases, it becomes more difficult to satisfy all QC criteria simultaneously. In such cases, the proposed metrics should be interpreted as practical reference points rather than strict pass/fail thresholds, with flexibility allowed based on visual judgment. When color alone is insufficient to achieve clear separation, alternative visual encodings, such as line types, increased line widths, marker shapes, or fill patterns, are recommended.

## CONCLUSION

This work demonstrates a practical approach to using AI as an assistant to select optimal color palettes for Shiny figures. User input guides the selection process, allowing flexible choices from a wide range of possible color combinations. It supports more dynamic Shiny visualizations with improved interactivity and user experience. Since color selection does not affect statistical results or clinical decisions, it becomes a suitable and low-risk entry point for AI adoption.

The broader implication of this work lies in its conceptual model, in which AI outputs are constrained and evaluated using domain-specific QC criteria. Within this human-in-the-loop framework, we aim to provide a useful reference for incorporating AI into other clinical workflows, while maintaining transparency, control, and compliance in automated processes.

## REFERENCES


U.S. Food and Drug Administration (2016). Portable Document Format Specifications. https://www.fda.gov/files/drugs/published/Portable-Document-Format-Specifications.pdf (accessed March 26, 2026)

Duke SP, Bancken F, Crowe B, Soukup M, Botsis T, Forshee R (2015). Seeing is believing: good graphic design principles for medical research. *Statistics in Medicine*, 34(22), 3040–3059. https://doi.org/10.1002/sim.6549

Guo Y, Salzano M, Sun N, Healey S (2025). Zero to Hero: the making of a comprehensive R Shiny figures app. PharmaSUG 2025 Proceedings, Paper DV-219.

Guo Y (2025). An integrated R Shiny solution for dynamic subgroup adjustments and customizations. PharmaSUG 2025 Proceedings, Paper DV-160.

OpenAI (2024). openai: R package for OpenAI API. https://cran.r-project.org/web/packages/openai/index.html (accessed March 26, 2026)

Bransen M (2024). Tutorial: using the OpenAI API in R. https://tilburg.ai/2024/03/tutorial-openai-api-in-r/ (accessed March 26, 2026)

OpenAI (n.d.). OpenAI API documentation. https://developers.openai.com/api/docs (accessed March 26, 2026)

Prompt Engineering Guide (2026). Prompting guide. https://www.promptingguide.ai (accessed March 26, 2026)

OpenAI (2024). Introducing structured outputs in the API. https://openai.com/index/introducing-structured-outputs-in-the-api/ (accessed March 26, 2026)

Dhaliwal P (2025). Structured prompting techniques (XML and JSON). https://codeconductor.ai/blog/structured-prompting-techniques-xml-json/ (accessed March 26, 2026)

Bray T (2014). The JavaScript Object Notation (JSON) data interchange format (RFC 7159). https://datatracker.ietf.org/doc/html/rfc7159 (accessed March 26, 2026)

Ihaka R et al. (2025). Compute contrast ratios. https://colorspace.r-forge.r-project.org/reference/contrast_ratio.html (accessed March 26, 2026)

World Wide Web Consortium (2025). Web Content Accessibility Guidelines (WCAG) 2.2. https://www.w3.org/WAI/WCAG22/quickref/ (accessed March 26, 2026)

Sharma G, Wu W, Dalal EN (2005). The CIEDE2000 color-difference formula: implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, 30(1), 21–30.

Schuessler Z (n.d.). Delta E explained. https://zschuessler.github.io/DeltaE/learn/ (accessed March 26, 2026)

Sapusek S (2025). Dissecting Delta E and the mathematical difference between colors. https://www.printing.org/content/2025/02/18/dissecting-delta-e-and-the-mathematical-difference-between-colors (accessed March 26, 2026)

Wild N (2025). Everything you need to know about color Delta E. https://blog.hybridhelix.com/everything-you-need-to-know-about-color-delta-e/ (accessed March 26, 2026)

## CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Yi Guo
Pfizer Inc.
yi.guo@pfizer.com

## APPENDIX 1. R FUNCTIONS FOR COLOR QUALITY ASSESSMENT

```r
#install.packages("farver")
#install.packages("colorspace")
#install.packages("dichromat")

library(farver)
library(colorspace)
library(dichromat)

#-------------------------------------------------
# Relative Luminance
#-------------------------------------------------
relative_luminance <- function(hex) {
  rgb <- hex2RGB(hex)@coords[1, ]
  f <- function(x) {
    #Convert sRGB to linear RGB (gamma correction)
    ifelse(x <= 0.03928, x / 12.92, ((x + 0.055) / 1.055)^2.4)
  }
  0.2126 * f(rgb[1]) + 0.7152 * f(rgb[2]) + 0.0722 * f(rgb[3])
}

#-------------------------------------------------
# Luminance Difference
#-------------------------------------------------
deltaL <- function(cols) {
  lum <- sapply(cols, relative_luminance)
  outer(lum, lum, function(a, b)
    abs(a - b))
}

#-------------------------------------------------
# Contrast Ratio (vs White Background)
#-------------------------------------------------
contrast_vs_white <- function(hex) {
  contrast_ratio(hex, "#FFFFFF")
}

#-------------------------------------------------
# Color Difference
#-------------------------------------------------
deltaE2000 <- function(cols) {
  rgb_mat <- decode_colour(cols)
  compare_colour(rgb_mat, from_space = "rgb", method = "cie2000")
}

#-------------------------------------------------
# Color Difference under CVD
#-------------------------------------------------

#Protanopia
deltaE2000_protan <- function(cols) {
```

```r
  cb_cols <- dichromat(cols, type = "protan")
  rgb_mat <- decode_colour(cb_cols)
  compare_colour(rgb_mat, from_space = "rgb", method = "cie2000")
}

#Deuteranopia
deltaE2000_deutan <- function(cols) {
  cb_cols <- dichromat(cols, type = "deutan")
  rgb_mat <- decode_colour(cb_cols)
  compare_colour(rgb_mat, from_space = "rgb", method = "cie2000")
}

#Tritanopia
deltaE2000_tritan <- function(cols) {
  cb_cols <- dichromat(cols, type = "tritan")
  rgb_mat <- decode_colour(cb_cols)
  compare_colour(rgb_mat, from_space = "rgb", method = "cie2000")
}

#-------------------------------------------------
# Test
#-------------------------------------------------

cols <- c("#b06389", "#a2c4c9", "#f1e9d4")

#Relative Luminance
sapply(cols, relative_luminance)

#Luminance Difference
deltaL(cols)

#Contrast Ratio
sapply(cols, contrast_vs_white)

#Color Difference
deltaE2000(cols)

#Color Difference under CVD
deltaE2000_protan(cols)
deltaE2000_deutan(cols)
deltaE2000_tritan(cols)
```