# Qualpal: Qualitative Color Palettes for Everyone

**Johan Larsson** ◍ [1]

**1** Department of Mathematical Sciences, University of Copenhagen ᴿᴼᴿ

## Summary

qualpal is a C++ library, command-line tool, R package, and web app for creating qualitative color palettes with maximally distinct colors. It helps scientists and anyone working with data visualization choose colors that remain clear and accessible. qualpal features flexible palette generation from multiple input formats, such as the HSL and $LCH_{ab}$ color spaces or fixed sets of RGB colors, and can adapt palettes to color vision deficiencies (CVD) across the full dichromacy spectrum at any severity. At its core, qualpal is a lightweight C++ library with no external dependencies, making it easy to integrate with other software and programming languages.

## Statement of need

Effective visualization of categorical data requires color palettes with easily distinguishable colors—for both people with normal color vision and those with CVD. Designing a palette is therefore an optimization problem, where the goal is to maximize the minimum difference among the colors in the palette to make the palette as distinct as possible. This is a non-trivial problem, as the number of possible palettes grows exponentially with the number of colors in the palette. This means that any given palette can at best be optimal only for a specific number of categories (colors). Since users may also have specific requirements in terms of, for instance, hue, lightness, saturation, adaptation to CVD, or background color, it is impossible to provide a set of fixed palettes to cover these needs. Therefore, there is a need for flexible palette generation tools that can accommodate a wide range of user requirements and preferences.

To tackle this problem, Glasbey et al. (2007) developed an algorithm based on simulated annealing, which is currently available in the Python package glasbey (McInnes, 2025). Other tools include iWantHue (Jacomy, 2013/2025), Colorgorical (Gramazio et al., 2016), distinctipy (Roberts et al., 2024), and Palettailor (Lu et al., 2021). All of these packages rely on some metric to measure the distance between colors and use some form of optimization algorithm, such as simulated annealing, to find a set of colors that maximizes the minimum distance between them in the palette. We summarize these existing packages and their features in Table 1 and Table 2, respectively.

**Table 1:** Summary of related work and packages, in terms of their algorithms, metrics, input types, and implementation languages.

| Package | Algorithm | Metrics | Input | Language |
|---|---|---|---|---|
| Glasbey | Simulated annealing | CIE76 | $LCH_{ab}$, Fixed set | Python |
| iWantHue | $k$-means/force vectors | CIE76 | $LCH_{ab}$ | JavaScript |
| Colorgorical | Random sampling | CIEDE2000 | $LCH_{ab}$ | Python |
| distinctipy | Random sampling | $L_{uv}$ approx | Pastel filter | Python |

| Package | Algorithm | Metrics | Input | Language |
|---|---|---|---|---|
| Palettailor | Simulated annealing | CIEDE2000 | Hue, lightness | JavaScript |
| **qualpal** | Farthest points | CIEDE2000, DIN99d, CIE76 | HSL, $LCH_{ab}$, Fixed sets | C++ |

32 Existing packages have different strengths and weaknesses. `qualpal` is, however, the first C++
33 library, CLI tool, and R package for generating qualitative color palettes. It is also the first
34 package to implement a farthest point sampling algorithm for generating qualitative color
35 palettes, and the only one to support multiple types of CVD. In addition, it is the only package
36 to support input from the HSL color space, which represents an intuitive way to specify colors
37 in terms of hue, saturation, and lightness. It also supports multiple metrics for measuring color
38 distance, including CIEDE2000 (Sharma et al., 2005), DIN99d (Cui et al., 2002), and CIE76.

**Table 2:** Summary of features of existing packages, in terms of color vision adaptation (CVD), availability of a web app (Web), command-line interface (CLI), ability to extend existing palettes, option to adapt to a background color, and possibility to create palettes with related blocks (such as pairs).

| Package | CVD | Web | CLI | Extend | Back-ground | Blocks |
|---|---|---|---|---|---|---|
| Glasbey | ✓ | | | ✓ | | ✓ |
| iWantHue | ✓ | ✓[1] | | | | ✓ |
| Colorgorical | | ✓[2] | | | | |
| distinctipy | ✓ | | | | | |
| Palettailor | | ✓[3] | | | ✓ | |
| **qualpal** | ✓ | ✓[4] | ✓ | ✓ | ✓ | |

# Summary of the algorithm

40 Qualpal begins with a set of input colors. These can either be a fixed set of colors provided by
41 the user, one of the built-in palettes, or a subspace in the $LCH_{ab}$ or HSL color spaces. In the
42 latter case, we use a quasi-random Halton sequence (Halton, 1964) to spread colors throughout
43 this subspace. The input colors are then (optionally) projected into a color space corresponding
44 to one or several CVD types, such as protanopia or deuteranopia, using simulation methods
45 described by Machado et al. (2009).

46 After this, we compute a full color distance matrix for the colors in the input set, using the
47 CIEDE2000 (Sharma et al., 2005) color difference metric by default. Finally, we run a farthest
48 point sampling algorithm loosely based on the work by Schlömer et al. (2011), which iteratively
49 swaps colors between a candidate palette and its complement set of colors until no swap
50 can improve the minimum distance between colors in the candidate palette. Optionally, a
51 background color can be included in this step, in which case the palette is optimized to be
52 distinct from it. The algorithm is deterministic (unlike the other algorithms from **??**) and takes
53 roughly 0.1 seconds to generate a 10-color palette from a set of 1000 input colors.

---

[1] https://medialab.github.io/iwanthue/
[2] http://vrl.cs.brown.edu/color (but down at the time of writing)
[3] https://iamkecheng.github.io/palettailor/
[4] https://qualpal.cc

## Acknowledgements

## References

Cui, G., Luo, M. R., Rigg, B., Roesler, G., & Witt, K. (2002). Uniform colour spaces based on the DIN99 colour-difference formula. *Color Research & Application*, *27*(4), 282–290. https://doi.org/10.1002/col.10066

Glasbey, C., van der Heijden, G., Toh, V. F. K., & Gray, A. (2007). Colour displays for categorical images. *Color Research & Application*, *32*(4), 304–309. https://doi.org/10.1002/col.20327

Gramazio, C. C., Laidlaw, D. H., & Schloss, K. B. (2016). Colorgorical: Creating discriminable and preferable color palettes for information visualization. *IEEE Transactions on Visualization and Computer Graphics*, *23*(1), 521–530. https://doi.org/10.1109/TVCG.2016.2598918

Halton, J. H. (1964). Algorithm 247: Radical-inverse quasi-random point sequence. *Commun. ACM*, *7*(12), 701–702. https://doi.org/10.1145/355588.365104

Jacomy, M. (2025). *iWantHue*. Médialab Sciences Po. https://github.com/medialab/iwanthue (Original work published 2013)

Lu, K., Feng, M., Chen, X., Sedlmair, M., Deussen, O., Lischinski, D., Cheng, Z., & Wang, Y. (2021). Palettailor: Discriminable colorization for categorical data. *IEEE Transactions on Visualization and Computer Graphics*, *27*(02), 475–484. https://doi.org/10.1109/TVCG.2020.3030406

Machado, Gustavo. M., Oliveira, Manuel. M., & Fernandes, Leandro. A. (2009). A physiologically-based model for simulation of color vision deficiency. *IEEE Transactions on Visualization and Computer Graphics*, *15*(6), 1291–1298. https://doi.org/10.1109/tvcg.2009.113

McInnes, L. (2025). *Glasbey* (Version 0.3.0). https://github.com/lmcinnes/glasbey

Roberts, J., Crall, J., Ang, K.-M., & Brandt, Y. (2024). *Distinctipy* (Version v1.3.4). Zenodo. https://doi.org/10.5281/zenodo.10480933

Schlömer, T., Heck, D., & Deussen, O. (2011). Farthest-point optimized point sets with maximized minimum distance. *Proceedings of the ACM SIGGRAPH Symposium on High Performance Graphics*, 135–142. https://doi.org/10.1145/2018323.2018345

Sharma, G., Wu, W., & Dalal, E. N. (2005). The CIEDE2000 color-difference formula: Implementation notes, supplementary test data, and mathematical observations. *Color Research & Application*, *30*(1), 21–30. https://doi.org/10.1002/col.20070