Inverse Design of Reflectance Palettes That Survive a Change of Light

Satyajeeth Suresh Kannan
Innovation Academy · Mary's Lab
github.com/JeetuSK0808/maryslab

Preprint · August 2026 · Mary's Lab technical report TR-10

ABSTRACT

Ordinary colorimetry runs forward: measure a surface, predict its appearance. We run it backward. Given a requirement — N categories, mutually separated, under every illuminant in a declared set — we compute reflectance spectra that satisfy it, searching the 38-dimensional space of physical reflectances rather than the 3-dimensional space of colors. The extra dimensions are the point: they are the metameric freedom that allows a palette to hold separation under one light while also holding it under another. We report achieved separation as a function of class count for one and two viewing conditions. Adding a second condition costs separation at four of the five class counts — at 5 classes, 0.402 falls to 0.3867 — which quantifies the price of illuminant robustness that a forward workflow never sees. At 8 classes the ordering reverses; we attribute that to search noise rather than to a real advantage, and analyse it rather than omit it.

Keywords: inverse design; evolution strategies; palette design; metamerism; spectral optimization

1. Introduction

A designer choosing a categorical palette works in a color space: pick swatches, check contrast, ship. The artifact that results, if it is printed or manufactured, has reflectance spectra — determined not by the designer but by whatever pigments the process happens to use to hit those colors.

This is a lost opportunity as well as a risk. The choice of spectrum behind a color is exactly the freedom that determines how the palette behaves when the light changes. Working in color space discards it before the question is asked.

Inverse design puts the requirement first. We state the constraint — mutual separation, under multiple conditions — and solve for spectra that satisfy it.

2. Method

Reflectances are parameterised by a low-order polynomial basis, which serves two purposes: it reduces the search dimension, and it constrains solutions to the smooth curves real materials produce rather than the high-frequency noise that would otherwise satisfy the objective mathematically.

The objective is the minimum pairwise separation across all declared viewing conditions — a minimum over both pairs and conditions. As in TR-09 this is non-smooth, and the optimiser is a (1+1) evolution strategy for the same reason. A mild smoothness reward is added to the objective to keep designs manufacturable.

Every returned spectrum is verified after the search: bounded in [0,1] at every wavelength, and pushed through the forward path to confirm the appearance the optimiser claimed. Optimisers will happily return infeasible points if the objective rewards them, and the check is cheap.

3. Evaluation

Inputs are synthetic and the outputs are targets, not formulations. A designed reflectance is a curve a material would need to have; whether a pigment system can achieve it, at cost and with stability, is a chemistry question outside this instrument.

Table 1. Achieved minimum pairwise separation as a function of class count, for one viewing condition (D65) and two (D65 and A). Identical seeds and budgets; the only difference is the number of conditions the palette must satisfy simultaneously.

classes	one condition	two conditions	cost of robustness
3	0.6998	0.6016	—
4	0.5128	0.4621	—
5	0.402	0.3867	—
6	0.3441	0.2743	—
8	0.2321	0.2443	—

Separation falls with class count under both regimes, as the palette-ceiling analysis of TR-03 predicts it must. The informative comparison is between columns: requiring the palette to hold under a second illuminant reduces achievable separation at 3, 4, 5 and 6 classes. That difference is the price of illuminant robustness, and it is invisible to a workflow that designs under one light and hopes.

At 8 classes the columns invert — the two-condition run reports 0.2443 against 0.2321 for one condition. Taken at face value this would say that adding a constraint improves the optimum, which is impossible: the two-condition feasible set is a subset of the one-condition set, so its optimum cannot exceed it. The inversion is therefore evidence about the search rather than about the problem. Both runs are single seeds at a fixed budget, and at 8 classes the objective is a minimum over 28 pairs, which makes the landscape harder and the achieved value noisier; the one-condition run simply landed worse. We report it because a table that showed a clean monotone gap at every class count would be reporting the seed, not the instrument. The correct reading of the last row is that the two runs are within search noise of each other, and that the gap at 8 classes is not resolved by this experiment.

A designer can now make the trade explicitly: accept lower separation for robustness, reduce the class count, or restrict the illuminant requirement. All three are legitimate; choosing among them requires the number.

Reproducing these numbers

Clone `github.com/JeetuSK0808/maryslab` and run `node papers/build/experiments.mjs`. Every figure in this report is written by that script into `papers/build/results.json`; the instrument itself is exercised by `design_spectral_palette`. The engines carry a 401-vector conformance suite shared by a TypeScript reference implementation, a C++20 core, and a WebAssembly build, so a reported number is a property of the specification rather than of one implementation.

4. Real-data evaluation: designing versus choosing

Inverse design is only worth the machinery if it beats the obvious alternative, which is not "design badly" — it is *don't design at all, and pick from what exists*. A practitioner with a material library selects the most separated N materials in it and ships. That baseline has never been measured here because there was no library to measure it against.

`usgs-splib07a-1` is one: 1752 measured reflectances. We selected from it greedily by farthest-point in Oklab — take the most distant material, then repeatedly the one furthest from everything already chosen — and compared the resulting minimum pairwise separation against what the optimiser designs from scratch.

Table 2. Designed palettes against the best selection from 1,752 measured materials, D65, minimum pairwise ΔE_OK. Both columns are lower bounds: one seed for the optimiser, greedy selection for the corpus.

classes	designed	best from corpus	design advantage	corpus vs theoretical ceiling
3	0.6998	0.4341	+61%	0.359
4	0.5128	0.354	+45%	0.385
5	0.402	0.2791	+44%	0.317
6	0.3441	0.2756	+25%	0.409
8	0.2321	0.2256	+3%	0.374

Design wins, and the margin collapses. At three classes the optimiser reaches 0.6998 against 0.4341 from the library — about three-fifths more separation. By eight classes the two are within a few percent of each other.

The shape of that curve is the useful finding, and it has a straightforward reading. A small palette has room to be clever: the optimiser places three reflectances wherever it likes in a 38-dimensional space, while the library offers only the materials somebody happened to measure. As the class count grows, both are forced to spread across the whole available gamut, the freedom stops mattering, and selection catches up. The practical advice inverts depending on where you sit — for a small categorical palette, designing is worth real effort; for eight or more classes, a good library gets you essentially the same separation for none.

The last column is a separate and more sobering number. The best selection from 1,752 real materials reaches only about a third of the theoretical ceiling TR-03 certifies. Measured materials are nowhere near optimal colours, which are step functions no pigment produces. The gap between what is physically conceivable and what has actually been measured is much larger than the gap between designing and choosing.

Reproducing these numbers

Clone `github.com/JeetuSK0808/maryslab` and run `node papers/build/experiments.mjs`. Every figure in this report is written by that script into `papers/build/results.json`; the instrument itself is exercised by `design_spectral_palette`. The engines carry a 401-vector conformance suite shared by a TypeScript reference implementation, a C++20 core, and a WebAssembly build, so a reported number is a property of the specification rather than of one implementation.

5. Limitations

This section states what the result does not establish. It is written to be read before the abstract is quoted.

- **The design-versus-selection comparison is between two lower bounds.** The designed column is one seed at a fixed budget; the corpus column is greedy farthest-point selection, not the optimal subset, which is combinatorial. A better optimiser or an exhaustive search would move either column up, and the percentages should be read as indicative rather than exact.
    - **Designed is not manufacturable.** A smooth, non-negative reflectance in [0,1] is necessary for a real material, not sufficient. The output is a target for formulation.
    - **A seeded result is not an optimum.** It is the best found within a budget; the tables report what one seed achieved. No variance across seeds is reported, so a difference between two cells smaller than the search's run-to-run spread — as at 8 classes — should not be read as a difference between the conditions.
    - **The optimiser satisfies the stated objective, including the parts not meant.** A palette maximally separated under two named illuminants may be unattractive, unbrandable, or poor under a third light nobody declared. This is the standard hazard of inverse design and is not fixable by better optimisation.
    - **Basis order bounds expressible designs.** A low-order polynomial basis cannot represent narrowband features, so designs depending on sharp structure are outside the search space.
    - **Prior art was surveyed informally.** Spectral optimisation and palette design are both established; a search found no tool designing reflectance palettes against a multi-condition minimum separation objective with post-hoc physical verification.

6. Acknowledgements and disclosure

The instruments described here were implemented in Mary's Lab, an open-source computational color science project. Software design, implementation, and the preparation of this report were carried out with substantial assistance from a large language model (Anthropic Claude), under the author's direction and review. All numerical results were produced by executing the released code; none were generated by a language model. The author is responsible for the claims made here.

References

1. Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Frommann-Holzboog.
2. Hansen, N. (2016). The CMA evolution strategy: a tutorial. *arXiv:1604.00772*.
3. Ottosson, B. (2020). *A perceptual color space for image processing (Oklab)*. bottosson.github.io.
4. CIE (2004). *Colorimetry, 3rd edition*. CIE 15:2004. Commission Internationale de l'Éclairage, Vienna.
5. Wyszecki, G. and Stiles, W. S. (1982). *Color Science: Concepts and Methods, Quantitative Data and Formulae*, 2nd edition. Wiley.
6. Morovič, J. (2008). *Color Gamut Mapping*. Wiley.
7. Machado, G. M., Oliveira, M. M., and Fernandes, L. A. F. (2009). A physiologically-based model for simulation of color vision deficiency. *IEEE Transactions on Visualization and Computer Graphics*, 15(6), 1291–1298.
8. Dupont, D. (2002). Study of the reconstruction of reflectance curves based on tristimulus values. *Color Research &*

Application, 27(2), 88–99.

Source, engines, and the script that produced every number in this report: `github.com/JeetuSK0808/maryslab`. Results regenerate with `node papers/build/experiments.mjs`. Inputs: synthetic where §3 says so, and measured USGS splib07a spectra (snapshot `usgs-splib07a-1`) where a real-data section says so. Prepared with AI assistance (see Acknowledgements).