Somewhere Over the Rainbow: How to Make Effective Use of Colors in Scientific Visualizations

Reto Stauffer

http://hclwizard.org
Two ‘random’ forecast maps from wetterzentrale.de.
Introduction

**Color:**
- Integral element in graphical displays.
- Easily available in (statistical) software.

**Problem:** Little guidance about how to choose appropriate colors for a particular visualization task.

**Question:** What are useful color palettes for coding qualitative and quantitative variables?
Introduction

Main goal of our work:

• Raise awareness of the issue.
• Introduce Hue-Chroma-Luminance (HCL) model.
  • Based on human perception.
  • Better control for choosing color palettes.
• Provide convenient software for exploring and assessing HCL-based palettes.
RGB Rainbow

**RGB color space:** And the (in)famous rainbow color palette.
• The default color in many software packages.
• Conveniently used by many practitioners.
• Defaults only change slowly (if at all).

Question: Everybody does it – why should it be wrong?
What’s Wrong?

Gradients: Very strong

Original figure as published by the NOAA.
What’s Wrong?

Original figure as published by the NOAA.

Gradients:
Very strong

Saturation
Highly-saturated colors
What’s Wrong?

Gradients: Very strong

Saturation: Highly-saturated colors

Discontinuous: Bright, dark, bright, dark, ...

Original figure as published by the NOAA.
What’s Wrong?

Assignment
No longer unique

Hurricane Sandy
120-hour Day 1-5 Rainfall Forecast

Desaturated version of the original figure.
What’s Wrong?

Assignment
No longer unique

Interpretation
Where is the maximum?

Desaturated version of the original figure.
What’s Wrong?

Assignment
No longer unique

Interpretation
Where is the maximum?

Focus
On dark artefacts

Desaturated version of the original figure.
What’s Wrong?

Hurricane Sandy
120-hour Day 1-5 Rainfall Forecast

Desaturated version of the original figure.
What’s Wrong?

End-user
Who is it?
Consider
Visual constraints?

What color-blind people see (red-green weakness).

About 5% of all Europeans are affected.
Challenges

**Summary:** The colors in a palette should

- be simple and natural,
- not be unappealing,
- highlight the important information,
- not mislead the reader,
- work everywhere and for everyone.

**In practice:**

- People often do not think about it at all.
- . . . and simply use default colors.

**Potential problems:**

- For end users – reviewers, supervisor, colleague, customer.
- For your own day-to-day work.
The Hue-Chroma-Luminance Color Space

A Perception-Based Color Space
Perception-Based Way: HCL

**Advantages:**

- **Hue:** Type of color.
- **Chroma:** Colorfullness.
- **Luminance:** Brightness.
Perception-Based Way: HCL

- **Hue** (*defines the color*)
- **Chroma** (*defines the colorness*) and
- **Luminance** (*defines the brightness*)
HCL Version

Colors:
Smooth gradients.

Information:
Guiding, no hidden information.

Works:
Screen, projector, gray-scaled device.

Same information, changed color scheme.
HCL Version

Assignment:
Higher values $\Rightarrow$ lower luminance.

Desaturated representation of the HCL-version.
Assignment:
Higher values ⇒ lower luminance.

Focus:
leads readers to most important areas.

Desaturated representation of the HCL-version.
HCL Version

Assignment:
Higher values ⇒ lower luminance.

Focus:
leads readers to most important areas.

Summary:
Solved a lot of problems by changing the color palette.

Desaturated representation of the HCL-version.
Warning Map Example

Colorized
Original (left)
HCL idea (right)

Gray-scale

Deuteranopia
Red-Green weakness
Color Palettes: Qualitative

**Goal:** Code quantitative data.

![Color Palette Diagram]
Color Palettes: Qualitative

**Goal:** Code quantitative data.

- **dynamic [30, 300]**
- **harmonic [60, 240]**
- **cold [270, 150]**
- **warm [90, -30]**
**Color Palettes: Qualitative**

**Goal:** Code quantitative data.

**Solution:** Take colors with different hues, but keep chroma and luminance constant. E.g.: \((H, 50, 70)\)
Color Palettes: Sequential

**Goal:** Code quantitative data (e.g., probabilities) where one side is of main interest.
Color Palettes: Sequential

**Goal:** Code quantitative data (e.g., probabilities) where one side is of main interest.

**Solution:** Constant hue and changing chroma/luminance. E.g., (90 – 0, 30 – 100, 90 – 50).
Color Palettes: Diverging

**Goal:** Code quantitative data and highlight both ends of the spectrum (e.g., anomalies, wet/dry, probabilities, ...).
Color Palettes: Diverging

**Goal:** Code quantitative data and highlight both ends of the spectrum (e.g., anomalies, wet/dry, probabilities, ...).

**Solution:** Diverging color schemes; combine sequential schemes with smooth transition.
Color Palettes: Diverging

**Goal:** Code quantitative data and highlight both ends of the spectrum (e.g., anomalies, wet/dry, probabilities, . . .).

**Solution:** Diverging color schemes; combine sequential schemes with smooth transition.
Experiences With Practitioners

In the beginning

- Hesitation of colleagues.
- “Not necessary!”
- “Why should we change existing products?”
- “Everybody does it like this . . . ”

A few days later

- Mainly positive feedback.
- Decrease of misinterpretations in classroom (“Weather & Forecast”).
- “Much easier to interpret . . . ”
- “How can I make use of those palettes (in my software)?”
hclwizard.org
A Perception-Based Color Space
The R colorspace package

- Available on c-ran.
- Provides methods to explore, choose, and assess HCL based color maps.

For non-R users: We set up a website to provide the same interface to everyone. Just visit: http://hclwizard.org.
colorsace Package

Using the colorspace package in R

The open source software R provides a package called (Ihaka, Murrell, Hornik, Fisher, & Zeileis, 2016) which uses Ihaka's colorspace library. The package offers some presetted color palettes (rainbow_hcl, terrain_hcl, heat_hcl, ...) to compare to the default color palettes. Furthermore there is a graphical user interface (GUI) where you can design your own color palette. The function therefore is called choose_palette() (needs no dependent packages) and use them for your own work.

The first thing you need is an R installation on your computer. The installation packages for available operating systems (Windows, OSX, Linux) can be found on the http://cran.r-project.org. The installation is no adventure and the R base version just needs a few MB of space. For beginners the R-Studio GUI is recommended (or a similar R editor).

If R is successfully installed on your system you can install optional packages like the colorspace package. You can simply do that over your GUI interface or install them by using the following code line:

install.packages('colorspace')
hclwizard.org
** Define colors palette
```
' set rgb 20 0  24 127'
' set rgb 21 44 12 137'
' set rgb 22 79 0 138'
' set rgb 23 103 0 140'
' set rgb 24 123 0 141'
' set rgb 25 141 0 141'
' set rgb 26 156 0 139'
' set rgb 27 171 20 136'
' set rgb 28 183 38 132'
' set rgb 29 195 54 126'
' set rgb 30 205 70 118'
' set rgb 31 214 85 107'
' set rgb 32 222 101 95'
' set rgb 33 228 117 79'
' set rgb 34 232 133 58'
' set rgb 35 235 150 25'
' set rgb 36 237 166 0'
' set rgb 37 236 183 0'
' set rgb 38 224 200 0'
' set rgb 39 230 218 0'
' set rgb 40 224 236 0'
' set rgb 41 218 255 71'
' set coolw 0  0 10 15 20 25 30 35 40 45 50 55 60 65 70 75 80 85 90 95 100'
```
```
** Open data set via BODS
** Open data set via DODS
'sdfopen http://nomads.ncep.noaa.gov:9090/dods/gfs_1p0f/gfs20170710/gfs_1p0f_00r_ani'
** - Setting region to load
' set ion -20 40'
' set lat 39 85'
** - Map projection
' set map proj hirc'
```
The R colorspace Package
A Perception-Based Color Space
R colorspace

> library('colorspace')
> # Interactively choosing color palettes
> #
> # Variant A:
> # pal <- choose_palette()
> #
> # Variant B (requires shiny and shinyjs):
> # pal <- hclwizard()
R colorspace

Figure: Screenshot of the tikz choose_palette interface.
R colorspace

**Figure:** Screenshot of the hclwizard interface.
R colorspace

Use colorspace package on command-line level

```r
> # choose_palette and hclwizard return a colormap function
> class(pal)
[1] "function"

> # function (n, h = c(12, 265), c = 80, l = c(25, 95), power = 0.7,
> # fixup = TRUE, gamma = NULL, alpha = 1, ...)
```
Use colorspace package on command-line level

> # choose_palette and hclwizard return a colormap function
> class(pal)

[1] "function"

> # function (n, h = c(12, 265), c = 80, l = c(25, 95), power = 0.7,
> #          fixup = TRUE, gamma = NULL, alpha = 1, ...)

Draw a color map with \( N \) colors:

> pal(3)

[1] "#0092E7" "#E2E2E2" "#E45279"

> pal(9)

[1] "#0092E7" "#3BA5E6" "#86B8E5" "#B7CDE5" "#E2E2E2" "#EAC0C7" "#EC9DAD"

[8] "#EA7A93" "#E45279"
R colorspace

Basic colorspace wrapper methods:

```r
> qual <- rainbow_hcl(n=11)
> seq <- sequential_hcl(n=11, h=0, l=c(90,40), c.=c(0,60))
> heat <- heat_hcl(n=11, h=c(0,-120), l=c(70,40), c.=c(30,60))
> div <- diverge_hcl(n=11, h=c(270,120), c=60, l=c(50,80))
```
R colorspace

Assess the spectrum of a color map:

```r
> div <- diverge_hcl(n=91, h=c(270,120), c=60, l=c(50,80))
> specplot(div)
```
R colorspace

Assess the spectrum of a color map:

```r
> rainbow <- rainbow(91)
> specplot( rainbow )
```
R colorspace

Use colorspace to convert colors:

```r
> div <- diverge_hcl(n=5, h=c(270,120), c=60, l=c(50,80))
> RGB <- hex2RGB(div); RGB

R   G   B
[1,] 0.4705882 0.4274510 0.7215686
[2,] 0.6666667 0.6549020 0.7568627
[3,] 0.7764706 0.7764706 0.7764706
[4,] 0.6039216 0.6862745 0.5803922
[5,] 0.2823529 0.5215686 0.1529412

> # Convert to HCL
> HCL <- as(RGB,"polarLUV"); HCL

L     C    H
[1,] 49.96609 60.378199003 270.31045
[2,] 69.49331 20.598091195 270.26623
[3,] 79.88122 0.006140369 94.09931
[4,] 69.23223 21.080046408 119.96933
[5,] 49.85643 59.608000199 119.81271
```
R colorspace

One of the “core functions” is `polarLUV`:

```r
> L <- seq(100, 30, length=12)
> C <- seq(40, 80, length=12)
> H <- rep( c(0,120,240), c(4,4,4) )
> HCL <- polarLUV(H=H, C=C, L=L)
```
R colorspace

One of the “core functions” is `polarLUV`:

```r
> L <- seq(100, 30, length=12)
> C <- seq(40, 80, length=12)
> H <- rep( c(0,120,240), c(4,4,4) )
> HCL <- polarLUV(H=H, C=C, L=L)

Convert colors to hexadecimal representation:

```r
> hexT <- hex( as(HCL,"RGB"), fixup=TRUE)
> hexF <- hex( as(HCL,"RGB"), fixup=FALSE)
```
One of the “core functions” is `polarLUV`:

```r
L <- seq(100, 30, length=12)
C <- seq(40, 80, length=12)
H <- rep( c(0,120,240), c(4,4,4) )
HCL <- polarLUV(H=H, C=C, L=L)
```

What does the `fixup=TRUE`:

```r
as(HCL,"RGB")
```

```
   R            G            B
[1,] 1.50358973 0.85466492 0.95620852
[2,] 1.33963389 0.70133126 0.80120053
[3,] 1.18684461 0.56676651 0.66377437
[4,] 1.04495144 0.44977227 0.54287660
[5,] 0.27519179 0.56279146 0.20191278
[6,] 0.19728039 0.46266385 0.12965811
[7,] 0.13385835 0.37503736 0.07240051
[8,] 0.08354332 0.29909188 0.02861425
[9,] -0.01705634 0.20360461 0.48078327
[10,] -0.05184201 0.15500028 0.41481690
[11,] -0.07965869 0.11583956 0.36140338
[12,] -0.10459109 0.08556191 0.32440858
```
Summary

Choice of colors:

• Use color with care!
• Think about who the readers/users are.
• Avoid large areas of flashy, highly-saturated colors.
• Employ monotonic luminance scale for numerical data.

Try it yourself:

• http://hclwizard.org
• colorspace in R.
References


Thank you for your attention!