CP SC 8810
Data Visualization

Joshua Levine
levinej@clemson.edu
Lecture 07
Processing Intro
Sept. 16, 2014
Agenda

- Lab 01 Questions — due tonight
- Lab 02 is posted
- Design Critique: Junyan and Chaoren’s are due tomorrow
- No questions on the paper readings are due this week (instead, spend time reading and trying out examples of Processing)
Continuing from Lec06
Color $\neq$ Wavelength

But rather, a combination of wavelengths and energy
All colors visible to the average human eye are contained inside the diagram.

The colors along any line between two points can be made by mixing the colors at the end points. In this case Green + Red = Yellow.

The edge of the diagram, called the spectral locus, represents pure, monochromatic light measured by wavelength in nanometers. These are the most saturated colors.

The least saturated colors are at the center, emanating from white.

Color gamut: subset of colors that can be represented by mixing the colors at its corners.

“Line of purples”: these colors are fully saturated but can only be made by mixing two colors (red and blue).

Anatomy of a CIE Chromaticity Diagram
What is a Colormap?

• Specifies a mapping between color and values
  • Sometimes called a transfer function

• Colormaps can be:
  • categorical vs. ordered
  • sequential vs. diverging
  • segmented vs. continuous
  • univariate vs. bivariate

• Recall: expressiveness in visual encoding — Match colormap to attribute type characteristics!
What is a Colormap?

• Specifies a mapping between color and values
  • Sometimes called a transfer function

• Colormaps can be:
  • categorical vs. ordered
  • sequential vs. diverging
  • segmented vs. continuous
  • univariate vs. bivariate

• Recall: expressiveness in visual encoding — Match colormap to attribute type characteristics!
What is a Colormap?

• Specifies a mapping between color and values
  • Sometimes called a **transfer function**

• Colormaps can be:
  • categorical vs. ordered
  • sequential vs. diverging
  • segmented vs. continuous
  • univariate vs. bivariate

• Recall: expressiveness in visual encoding — Match colormap to attribute type characteristics!
What is a Colormap?

- Specifies a mapping between color and values
  - Sometimes called a **transfer function**

- Colormaps can be:
  - categorical vs. ordered
  - sequential vs. diverging
  - segmented vs. continuous
  - univariate vs. bivariate

- Recall: expressiveness in visual encoding — Match colormap to attribute type characteristics!
What is a Colormap?

- Specifies a mapping between color and values
  - Sometimes called a **transfer function**

- Colormaps can be:
  - categorical vs. ordered
  - sequential vs. diverging
  - segmented vs. continuous
  - univariate vs. bivariate

- Recall: expressiveness in visual encoding — Match colormap to attribute type characteristics!
What is a Colormap?

• Specifies a mapping between color and values
  • Sometimes called a **transfer function**

• Colormaps can be:
  • categorical vs. ordered
  • sequential vs. diverging
  • segmented vs. continuous
  • univariate vs. bivariate

• Recall: expressiveness in visual encoding —
  Match colormap to attribute type characteristics!
Categorical Colormaps

- Categorical colors are easier to remember if they are nameable

- Typically designed by using color as an integral identity channel to encode a single attribute

Colin Ware, Information Visualization: Perception for Design
Form

• When applying shading to define the shape of a curved surface, use adequate luminance (as opposed to chromatic) variation.

• If large areas are defined using nearly equiluminous colors, consider using thin border lines with large luminance differences (from the colors of the areas) to help define the shapes.

Even large shapes are seen more clearly if a luminance contrast boundary is provided.

Colin Ware, Information Visualization: Perception for Design
Summary/Guidelines

• Saturation interacts strongly with size

• In small regions:
  • Saturation is more difficult to perceive, in particular saturation and hue are not separable
  • Use bright, highly saturated colors
  • For points and lines use just two saturation levels

• In large regions
  • Higher saturation makes large areas look bigger
  • Use low saturation pastel colors for large regions and backgrounds
Encoding Ordered Data
Order These Colors?

Based on slide from Stasko
Order These Colors?
Order These Colors?
Colormaps & Ordered Attributes

- Ordered colormaps are the most effective if they vary along saturation or luminance.

**Figure 10.5.** The luminance and saturation channels are automatically interpreted as ordered by our perceptual system, but the hue channel is not.
Summary/Guidelines

• Number of hues, and distribution on the colormap, should be related to which, and how many structures in the data to emphasize

  • min or max, ends or middle, etc…

• Show ordinal data with a discrete set of colors (and hence have to limit the number)

• If encoding ordinal data with color, place marks on solid, neutral background
Encoding Quantitative Data
Quantitative Data

- Sequential vs. Diverging Colormaps

**Figure 10.11.** Rainbow versus two-hue continuous colormap. (a) Using many hues, as in this rainbow colormap, emphasizes mid-scale structure. (b) Using only two hues, the blue–yellow colormap emphasizes large-scale structure. From [Bergman et al. 95, Figures 1 and 2].
The Rainbow Colormap

Figure 10.12. Rainbow versus multiple-hue continuous colormap with monotonically increasing luminance. (a) Three major problems with the common continuous rainbow colormap are perceptual nonlinearity, the expressivity mismatch of using hue for ordering, and the accuracy mismatch of using hue for fine-grained detail. (b) A colormap that combines monotonically increasing luminance with multiple hues for semantic categories, with a clear segmentation at the zero point, succeeds in showing high-level, mid-level, and low-level structure. From [Rogowitz and Treinish 98, Figure 1].

Where is zero in (a)?
Other Colormaps for Quantitative Data

Lightness scale

Lightness scale with hue and chroma variation

Hue scale with lightness variation
Rainbow Color Map (Still) Considered Harmful

Research has shown that the rainbow color map is rarely the optimal choice when displaying data with a pseudocolor map. The rainbow color map confuses viewers through its lack of perceptual ordering, obscures data through its uncontrolled luminance variation, and actively misleads interpretation through the introduction of non-data-dependent gradients.

Despite much published research on its deficiencies, the rainbow color map is prevalent in the visualization community. We present survey results showing that the rainbow color map continues to appear in more than half of the relevant papers in IEEE Visualization Conference proceedings; for example, it appeared on 61 pages in 2005. Its use is encouraged by its selection as the default color map used in most visualization toolkits that we inspected. The visualization community must do better.

In this article, we reiterate the characteristics that make the rainbow color map a poor choice, provide examples that clearly illustrate these deficiencies even on simple data sets, and recommend better color maps.

Confusing

For all tasks that involve comparing relative values, the color map used should exhibit perceptual ordering. A simple example of a perceptually ordered color map is the gray-scale color map. Increasing luminance from black to white is a strong perceptual cue that indicates values mapped to darker shades of gray are lower in value than values mapped to lighter shades of gray. This mapping is natural and intuitive.

The rainbow color map is certainly ordered—from a shorter to longer wavelength of light (or vice versa)—but it’s not perceptually ordered. If people are given a series of gray paint chips and asked to put them in order, the results vary (see Figure 1). However, if people are given paint chips colored red, green, yellow, and blue and asked to put them in alphabetical order, the results are consistent. This shows that people will consistently place red, orange, yellow, and green in light-to-dark order, but will put blue, indigo, violet, and red in dark-to-light order.

The visual system perceives high spatial frequencies (i.e., edges) with the highest sensitivity and therefore makes it easier to see small changes in data displayed with a perceptually ordered color map. Conversely, the visual system perceives low spatial frequencies (i.e., large areas) with the highest sensitivity and is therefore more susceptible to changes in data displayed with a non-perceptually ordered color map.

When we use a color map that is not perceptually ordered, we make the viewer work harder and more slowly to understand the data. The results of the analysis are not immediately evident, and we must infer them—either through remembering (an error-prone task) or construction (a time-consuming task). The results in Figure 1 illustrate this point clearly.

Here we will discuss the rainbow color map’s characteristics of confusing the viewer, obscuring data, and actively misleading interpretation.
Effects of Color on Large Areas
Effects of Color on Large Areas
Consider Color Segmentation to Improve Accuracy
Summary/Guidelines

- Because of contrast effects, it is difficult to perceive absolute luminance of noncontiguous regions
  - Use only 2-4 bins when background is nonuniform
  - For text, ideally use 10:1 ratio, 3:1 minimum
- Show quantitative data with either a discrete set of colors or continuous (discrete for accuracy)
- Redundantly vary lightness and saturation
Color Deficiencies
RG Color Blindness
Can be explained by opponent color theory
Missing Cones

Note that wavelength is plotted right-to-left
Rainbow Colormaps and Color Deficiency

http://blog.visual.ly/rainbow-color-scales/
http://geog.uoregon.edu/datagraphics/EOS/
Color Illusions
Color Illusions

- Primary cause: the Retinal Ganglion Response, per color.

- Recall: opponent process model.

- Triggered by light in the center, suppressed by light in the surround

- Causes selective sensitivities to discontinuities in color as well.
Simultaneous Contrast
Simultaneous Contrast
Simultaneous Contrast
Simultaneous Contrast
Chromatic Induction
Chromatic Induction

P. Monnier
Chromatic Adaptation
Chromatic Adaptation

What is the color of the flower?
Chromatic Adaptation

What is the color of the flower?
Chromatic Adaptation

What is the color of the flower?
Summary

- Color is a perceptual phenomena
- “Get it right in black and white.” — Maureen Stone
Tools for Color
Intro to Processing

Slide Credits:
Moritz Bächer
http://baecher.info/teaching.html
What is Processing?

Simple **programming environment** to develop **visually-oriented** applications with an emphasis on **animation** and **interaction**.

Powerful **design** and **prototyping** tool
Where to get it?

Go to:  http://www.processing.org/

Click on “Download”:

Pick your system:

- Linux
- Mac OS X
- Windows
- Windows (Without Java)*
Processing Development Environment
Processing Development Environment

Run   Stop
Processing Development Environment

Run    Stop
Open
New
Save
Export
Commands

```plaintext
name(arguments);

ellipse(50,50,100,100);
```
A First Sketch

```
ellipse(50,50,100,100);
```
A First Sketch

```javascript
ellipse(50, 50, 100, 100);
```
Errors

```
ellipse(50, 50, 100, 100)
```
Errors

highlighted line

error message

line number
Save And Export

Save

eEllipse(50, 50, 100, 100);
Save And Export

```java
ellipse(50, 50, 100, 100);
```

Save

Export
Language and Libraries

Shape, Color, Image, Text, Interaction, ...

Video, Network, PDF, Audio, ...
Simple Shapes

point(x,y);

line(x1,y1,x2,y2);

rect(x,y,width,height);

ellipse(x,y,width,height);
Colors

stroke(red, green, blue);

stroke(grey);

noStroke();

fill(red, green, blue);

fill(grey);

noFill();

**fill** color:

**stroke** color:

red, green, blue, grey in [0, 255]
Example

```plaintext
line(1,0,6,0);
rect(3,3,9,5);
rect(9,11,5,5);
```
Example

```
stroke(255,0,0);
line(1,0,6,0);
rect(3,3,9,5);
rect(9,11,5,5);
```
Example

```
stroke(255,0,0);
line(1,0,6,0);
stroke(0,255,0);
rect(3,3,9,5);
rect(9,11,5,5);
```
Example

```
stroke(255,0,0);
line(1,0,6,0);
stroke(0,255,0);
fill(0,0,255);
rect(3,3,9,5);
rect(9,11,5,5);
```
stroke(255,0,0);
line(1,0,6,0);
stroke(0,255,0);
fill(0,0,255);
rect(3,3,9,5);
noFill();
stroke(20);
rect(9,11,5,5);
Animation

```cpp
void setup() {
    // init code
}

void draw() {
    // drawing code
}
```
Animation

```cpp
void setup() {
    // init code
}

void draw() {
    // drawing code
}
```

```cpp
setup();
draw();
draw();
draw();
draw();
...```
Example

```java
void setup() {
    size(200,200);
}

void draw() {
    background(255);

    stroke(0);
    fill(255,0,0);
    rect(mouseX,mouseY,50,50);
}
```
Example

```java
void setup() {
    size(200, 200);
}

void draw() {
    background(255);
    stroke(0);
    fill(255, 0, 0);
    rect(mouseX, mouseY, 50, 50);
}
```

- set window size
- set background color
- (mouseX, mouseY)
- mouse position
Interaction

```cpp
mouseX, mouseY  // current mouse position

to mousePressed() {  
    // handle event
}

to keyPressed() {  
    // handle event
}
void setup() {
    size(200, 200);
    background(255);
}

void draw() {
}

void mousePressed() {
    stroke(0);
    fill(255, 0, 0);
    rect(mouseX, mouseY, 16, 16);
}

void keyPressed() {
    background(255);
}
**Example**

```java
void setup() {
    size(200, 200);
    background(255);
}

void draw() {
}

void mousePressed() {
    stroke(0);
    fill(255, 0, 0);
    rect(mouseX, mouseY, 16, 16);
}

void keyPressed() {
    background(255);
}
```

draw red rectangle when *mouse* is pressed

clear window when a *key* is pressed
Displaying Text

- `text()` - draw text to the screen
- `createFont()` - load a system font (slow!)
- `loadFont()` - load a font from a file (slow!)
- `textSize()` - set font size, in points
Find in Reference

1. Mark
2. Right Click
3. Find in Reference
Textbooks

*Exploring and Explaining Data with the Processing Environment*

*Visualizing Data*

*LEARNING PROCESSING*

*A Beginner’s Guide to Programming Images, Animation, and Interaction*

*Getting Started with Processing*

*Casey Reas & Ben Fry*
Lec08

• In class lab day, in McAdams 110B

• You’re welcome to bring your own laptop, but if not you should be able to download and run processing on the machines.

• Instructions for the lab will be posted beforehand.